



Σχολή Θετικών επιστημών & Τεχνολογίας
Μεταπτυχιακή Εξειδίκευση στα Πληροφοριακά Συστήματα

Διπλωματική Εργασία

Ανάπτυξη Dashboard για Real-time Παρακολούθηση και
Πρόβλεψη Απόδοσης Ενεργειακού Δικτύου Επιχείρησης που
Χρησιμοποιεί Φωτοβολταϊκές και Αιολικές Λύσεις

Ελένη Τσιανάκα

Επιβλέπων καθηγητής: Παπαμιχαήλ Γεώργιος

Πάτρα, Ιούνιος 2026

Η παρούσα εργασία αποτελεί πνευματική ιδιοκτησία του φοιτητή («συγγραφέας/δημιουργός») που την εκπόνησε. Στο πλαίσιο της πολιτικής ανοικτής πρόσβασης ο συγγραφέας/δημιουργός εκχωρεί στο ΕΑΠ, μη αποκλειστική άδεια χρήσης του δικαιώματος αναπαραγωγής, προσαρμογής, δημόσιου δανεισμού, παρουσίας στο κοινό και ψηφιακής διάχυσής τους διεθνώς, σε ηλεκτρονική μορφή και σε οποιοδήποτε μέσο, για διδακτικούς και ερευνητικούς σκοπούς, άνευ ανταλλάγματος και για όλο το χρόνο διάρκειας των δικαιωμάτων πνευματικής ιδιοκτησίας. Η ανοικτή πρόσβαση στο πλήρες κείμενο για μελέτη και ανάγνωση δεν σημαίνει καθ' οιονδήποτε τρόπο παραχώρηση δικαιωμάτων διανοητικής ιδιοκτησίας του συγγραφέα/δημιουργού ούτε επιτρέπει την αναπαραγωγή, αναδημοσίευση, αντιγραφή, αποθήκευση, πώληση, εμπορική χρήση, μετάδοση, διανομή, έκδοση, εκτέλεση, «μεταφόρτωση» (downloading), «ανάρτηση» (uploading), μετάφραση, τροποποίηση με οποιονδήποτε τρόπο, τμηματικά ή περιληπτικά της εργασίας, χωρίς τη ρητή προηγούμενη έγγραφη συναίνεση του συγγραφέα/δημιουργού. Ο συγγραφέας/δημιουργός διατηρεί το σύνολο των ηθικών και περιουσιακών του δικαιωμάτων.

Ανάπτυξη Dashboard για Real-time Παρακολούθηση και
Πρόβλεψη Απόδοσης Ενεργειακού Δικτύου Επιχείρησης που
Χρησιμοποιεί Φωτοβολταϊκές και Αιολικές Λύσεις

Ελένη Τσιανάκα

Επιτροπή Επίβλεψης Πτυχιακής / Διπλωματικής Εργασίας

Επιβλέπων Καθηγητής:

Παπαμιχαήλ Γεώργιος

Διευθυντής στη Διεύθυνση

Αποδοτικότητας, Πιστοποίησης και Νέων

Τεχνολογιών στο Εθνικό Κέντρο

Δημόσιας Διοίκησης

Συν-Επιβλέπων Καθηγητής:

Δρ. Γρηγόριος Μπεληγιάννης

Αν. Καθηγητής

Πρόεδρος Τμήματος

Διοίκηση Επιχειρήσεων Αγροτικών

Προϊόντων και Τροφίμων

Πανεπιστήμιο Πατρών

Πάτρα, Ιούνιος 2026

Ελένη Τσιανάκα, Ανάπτυξη Dashboard για Real-time
Παρακολούθηση και Πρόβλεψη Απόδοσης Ενεργειακού
Δικτύου Επιχείρησης που Χρησιμοποιεί Φωτοβολταϊκές και
Αιολικές Λύσεις

«Στον Χρήστο. Με πίστη, κόπο και επιμονή κάθε στόχος είναι εφικτός.»

Περίληψη

Η παρούσα διπλωματική εργασία πραγματεύεται τη σχεδίαση, ανάπτυξη και αξιολόγηση ενός Ευφυούς Συστήματος Ενεργειακής Παρακολούθησης (Intelligent Energy Management System – IEMS) για επιχείρηση που αξιοποιεί Ανανεώσιμες Πηγές Ενέργειας (ΑΠΕ), συγκεκριμένα φωτοβολταϊκά και αιολικά συστήματα. Το αναπτυχθέν σύστημα παρέχει real-time παρακολούθηση ενεργειακών δεδομένων, πρόβλεψη φωτοβολταϊκής παραγωγής μέσω Γραμμικής Παλινδρόμησης, αυτοματοποιημένη ανίχνευση ανωμαλιών και διαφοροποιημένη πρόσβαση ανά ρόλο χρήστη.

Η αρχιτεκτονική του συστήματος βασίζεται σε σύγχρονες τεχνολογίες IoT και ακολουθεί πολυεπίπεδη προσέγγιση: ο προσομοιωτής IoT (Python) παράγει συνθετικά δεδομένα τηλεμετρίας και τα μεταδίδει μέσω MQTT. Ο MQTT broker (Eclipse Mosquitto) λειτουργεί ως κεντρικός διαμεσολαβητής. Το ενδιάμεσο επίπεδο (Node-RED) επεξεργάζεται, μετασχηματίζει και δρομολογεί τα δεδομένα. Η αποθήκευση πραγματοποιείται σε SurrealDB. Τα διαδραστικά dashboards υλοποιούνται μέσω Grafana. Το σύστημα ανάλυσης και πρόβλεψης βασίζεται σε Python microservice με μοντέλο Γραμμικής Παλινδρόμησης. Ολόκληρη η υποδομή είναι containerized μέσω Docker Compose.

Η αξιολόγηση πραγματοποιήθηκε μέσω αυτοματοποιημένων δοκιμών που καλύπτουν unit tests, integration tests και σενάρια πραγματικών συνθηκών. Τα αποτελέσματα επιβεβαιώνουν την αποτελεσματικότητα του συστήματος στην ανίχνευση ανωμαλιών, την ακρίβεια των προβλέψεων και τη λειτουργική ορθότητα του μηχανισμού ειδοποιήσεων.

Λέξεις – Κλειδιά

Ευφυή Συστήματα Ενεργειακής Διαχείρισης, IoT, Ανανεώσιμες Πηγές Ενέργειας, Μηχανική Μάθηση, Πρόβλεψη Ενέργειας, Ανίχνευση Ανωμαλιών

Development of a Dashboard for Real-time Monitoring and Performance Prediction of a Business Energy Network Using Photovoltaic and Wind Solutions

Eleni Tsianaka

Abstract

This thesis presents the design, development, and evaluation of an Intelligent Energy Management System (IEMS) for a business leveraging Renewable Energy Sources (RES), specifically photovoltaic and wind systems. The developed system provides real-time energy monitoring, photovoltaic production forecasting via Linear Regression, automated anomaly detection, and differentiated role-based user access.

The system architecture is based on modern IoT technologies following a multi-layered approach: an IoT simulator (Python) generates synthetic telemetry data transmitted via MQTT. The MQTT broker (Eclipse Mosquitto) acts as the central intermediary. The middleware layer (Node-RED) processes, transforms, and routes data. Storage is implemented in SurrealDB. Interactive dashboards are built with Grafana. The analytics and forecasting subsystem is a Python microservice with a Linear Regression model. The entire infrastructure is containerized via Docker Compose.

Evaluation was conducted through automated tests covering unit tests, integration tests, and real-world scenario testing. Results confirm the system's effectiveness in anomaly detection, prediction accuracy, and correct operation of the alert mechanism.

Keywords

Intelligent Energy Management Systems, IoT, Renewable Energy Sources, Machine Learning, Energy Forecasting, Anomaly Detection

Περιεχόμενα

Περίληψη.....	v
Abstract	vi
Περιεχόμενα.....	vii
Κατάλογος Εικόνων / Σχημάτων	xiv
Κατάλογος Πινάκων	xv
1 Εισαγωγή.....	1
1.1 Γενικό Πλαίσιο.....	1
1.2 Σκοπός και Στόχοι της Εργασίας	1
1.3 Δομή της Εργασίας	2
2 Θεωρητικό πλαίσιο	4
2.1 Συστήματα Ενεργειακής Διαχείρισης (Energy Management Systems – EMS)....	4
2.2 Ευφυή Συστήματα Ενεργειακής Διαχείρισης (Intelligent Energy Management Systems - IEMS)	5
2.2.1 Βασικές έννοιες και αρχές των IEMS	5
2.2.2 Συλλογή και Διαχείριση Δεδομένων Ενέργειας σε Πραγματικό Χρόνο	6
2.2.3 Αρχιτεκτονική Συστήματος Real-time Παρακολούθησης και Πρόβλεψης..	7
2.2.4 Τεχνολογίες Τεχνητής Νοημοσύνης στα IEMS	8
2.2.5 Προκλήσεις Υλοποίησης IEMS	11
2.3 Ανανεώσιμες Πηγές Ενέργειας: Φωτοβολταϊκά και Αιολικά Συστήματα.....	13
2.3.1 Φωτοβολταϊκά Συστήματα.....	14
2.3.2 Αιολικά Συστήματα.....	15
2.3.3 Ενσωμάτωση ΑΠΕ σε Επιχειρησιακά Δίκτυα	16

2.4	IoT και Smart Grids στην ενεργειακή παρακολούθηση	17
2.4.1	Smart Grid- Λειτουργικές απαιτήσεις και ρόλος της επικοινωνίας.....	19
2.4.2	IoT και ενεργειακή παρακολούθηση.....	20
2.4.3	Πρωτόκολλα επικοινωνίας και στοίβα πρωτοκόλλων (Communication Protocol Stack)	22
2.4.4	Κυβερνοασφάλεια και ιδιωτικότητα δεδομένων παρακολούθησης	23
2.4.5	Σύνδεση με real-time dashboards και Intelligent Energy Management Systems (IEMS)	25
2.5	Dashboards και συστήματα οπτικοποίησης δεδομένων.....	27
2.5.1	Θεωρητικές βάσεις οπτικοποίησης δεδομένων.....	27
2.5.2	Dashboards σε περιβάλλοντα Smart Grid και Intelligent Energy Management Systems (IEMS)	28
2.5.3	Real-time οπτικοποίηση και ροές δεδομένων	29
2.5.4	Δείκτες Απόδοσης (KPIs) και Οπτικοποίηση	30
2.5.5	Ανθρώπινος παράγοντας και εμπειρία χρήστη (UX).....	31
2.6	Τεχνικές πρόβλεψης ενεργειακής παραγωγής και κατανάλωσης	32
2.6.1	Στατιστικές μέθοδοι πρόβλεψης	32
2.6.2	Μέθοδοι μηχανικής μάθησης (Machine Learning).....	32
2.6.3	Βαθιά μάθηση (Deep Learning).....	33
2.6.4	Αξιολόγηση μοντέλων πρόβλεψης.....	33
2.6.5	Σύνδεση μοντέλων πρόβλεψης με IEMS και dashboards.....	34
3	Σχεδίαση και Αρχιτεκτονική Συστήματος	35
3.1	Ανάλυση απαιτήσεων συστήματος	37

3.1.1	Λειτουργικές απαιτήσεις	37
3.1.2	Μη λειτουργικές απαιτήσεις	38
3.1.3	Σύνδεση απαιτήσεων με αρχιτεκτονική	38
3.2	Γενική αρχιτεκτονική του συστήματος	39
3.2.1	Επίπεδο Παραγωγής Δεδομένων (Data Generation Layer)	39
3.2.2	Επίπεδο Μεταφοράς & Διαμεσολάβησης (Communication Layer)	39
3.2.3	Επίπεδο Επεξεργασίας & Middleware (Data Processing Layer)	39
3.2.4	Επίπεδο Αποθήκευσης & Παρουσίασης (Storage & Visualization Layer)	40
3.3	Συλλογή δεδομένων (IoT & smart meters)	40
3.4	Μοντελοποίηση Δεδομένων και Σχεδιασμός Βάσης (SurrealDB Schema)	41
3.4.1	Πίνακας telemetry (Καταγραφή Χρονοσειρών)	41
3.4.2	Πίνακας prediction (Δεδομένα Προβλέψεων)	42
3.4.3	Πίνακας alerts (Διαχείριση Συμβάντων)	42
3.5	Ανάλυση δεδομένων	42
3.6	Οπτικοποίηση δεδομένων και dashboard	43
4	Υλοποίηση του Συστήματος	44
4.1	Περιβάλλον Ανάπτυξης και Containerization	44
4.2	Προσομοιωτής IoT και Παραγωγή Δεδομένων	44
4.2.1	Ισχύς Κατανάλωσης φορτίου	45
4.2.2	Παραγωγή από φωτοβολταϊκά συστήματα	46
4.2.3	Παραγωγή από αιολικά συστήματα	46
4.2.4	Κατάσταση φόρτισης συσσωρευτή	46

4.2.5	Εισαγωγή και εξαγωγή ισχύος	47
4.2.6	Περιβαλλοντικές μετρήσεις	47
4.2.7	Κατάσταση λειτουργίας του συστήματος	47
4.2.8	Περιορισμοί.....	47
4.3	Μεταφορά Δεδομένων - Διαμεσολαβητής Μηνυμάτων	48
4.4	Ενδιάμεσο Επίπεδο Επεξεργασίας (Node-RED Middleware)	49
4.4.1	Κλάδος Αποθήκευσης Τηλεμετρίας.....	49
4.4.2	Κλάδος Παραγωγής Ειδοποιήσεων.....	50
4.4.3	Κλάδος Διασύνδεσης Analytics	50
4.5	Αποθήκευση Δεδομένων.....	51
4.6	Οπτικοποίηση δεδομένων και Dashboards	52
4.6.1	Current System Status Dashboard.....	53
4.6.2	Energy Monitoring Overview Dashboard	54
4.6.3	KPIs Dashboard	56
4.6.4	Alerts Dashboard.....	59
4.7	Σύστημα Ειδοποιήσεων.....	59
4.7.1	Ειδοποίηση Υψηλής Κατανάλωσης.....	60
4.7.2	Ειδοποίηση Χαμηλής Μπαταρίας	61
4.7.3	Ειδοποίηση Χαμηλής παραγωγής ΑΠΕ	61
4.8	Διαχείριση Ρόλων Χρηστών	63
4.8.1	Ορισμός Ρόλων	63
4.8.2	Τεχνική Υλοποίηση.....	64

5	Ανάλυση Δεδομένων.....	66
5.1	Επιλογή Μεθόδου Παλινδρόμησης.....	66
5.2	Συλλογή και Προεπεξεργασία Δεδομένων	67
5.3	Διαχωρισμός Εκπαίδευσης και Αξιολόγησης	67
5.4	Offline Εκπαίδευση και Αποθήκευση Μοντέλου	68
5.5	Online Πρόβλεψη (Inference) σε Πραγματικό Χρόνο.....	68
5.6	Υπολογισμός Απόκλισης και Ανίχνευση Ανωμαλιών	69
5.7	Αποθήκευση Προβλέψεων και Αποτελεσμάτων	69
5.8	Οπτικοποίηση Αποτελεσμάτων.....	70
6	Δοκιμές και αξιολόγηση του συστήματος.....	72
6.1	Μεθοδολογία Δοκιμών (Testing Methodology)	72
6.2	Δοκιμές Μονάδας (Unit Testing).....	73
6.2.1	Σενάριο 1: Κανονική Λειτουργία.....	73
6.2.2	Σενάριο 2: Αιχμή Κατανάλωσης (Peak Load)	73
6.2.3	Σενάριο 3: Κρίσιμη Στάθμη Μπαταρίας	73
6.2.4	Σενάριο 4: Υποαπόδοση Φωτοβολταϊκού Συστήματος	73
6.2.5	Σενάριο 5: Νυχτερινή Λειτουργία — Έλεγχος False Positives	74
6.2.6	Σενάριο 6: Κρίσιμη Κατάσταση — Πολλαπλά Ταυτόχρονα Alerts	74
6.3	Δοκιμές Ενοποίησης (Integration Testing)	74
6.3.1	Δοκιμή Ροής Ingestion και ETL.....	74
6.3.2	Δοκιμή Διασύνδεσης με τη SurrealDB	75
6.3.3	Δοκιμή Πλήρους Ροής Ειδοποιήσεων (MQTT → Node-RED → SurrealDB)	

6.4	Δοκιμές Σεναρίων του Microservice Πρόβλεψης.....	76
6.4.1	Σενάριο 1: Κανονική Λειτουργία PV.....	76
6.4.2	Σενάριο 2: Τεχνική Αστοχία	76
6.4.3	Σενάριο 3: Μερική Απώλεια Αποδοτικότητας	76
6.4.4	Σενάριο 4: Νυχτερινή Λειτουργία.....	76
6.4.5	Σενάριο 5: Ελλιπή Δεδομένα	76
6.4.6	Σενάριο 6: Κατάσταση πλήρους Ηλιακής Παραγωγής.....	76
6.5	Δοκιμές Ρόλων Χρηστών	77
6.6	Αξιολόγηση και Συμπεράσματα Δοκιμών	77
6.7	Αξιολόγηση από Χρήστες.....	77
6.7.1	Μεθοδολογία Αξιολόγησης	77
6.7.2	Συμμετέχοντες.....	78
6.7.3	Αποτελέσματα Ερωτηματολογίου.....	78
6.7.4	Ανάλυση Αποτελεσμάτων.....	79
7	Συμπεράσματα και Μελλοντική Εργασία	81
7.1	Συμπεράσματα	81
7.2	Περιορισμοί.....	82
7.3	Μελλοντική Εργασία	83
	Βιβλιογραφία.....	85
	Παράρτημα Α: Προσομειωτής.....	88
	Παράρτημα Β: Συναρτήσεις Node-RED.....	92
	Παράρτημα Γ: Εκπαίδευση μοντέλου.....	97

Παράρτημα Δ: Microservice για πρόβλεψη σε πραγματικό χρόνο.....	100
Παράρτημα Ε: Αυτοματοποιημένη διαδικασία δημιουργίας χρηστών με python.....	102
Παράρτημα ΣΤ: Δοκιμές Σεναρίων	109
Παράρτημα Ζ: Ερωτηματολόγιο χρηστών.....	120

Κατάλογος Εικόνων / Σχημάτων

Εικόνα 1 Αρχιτεκτονική Συστήματος	36
Εικόνα 2 Μήνυμα που παράγεται από τον προσομοιωτή	45
Εικόνα 3 Ροές Node-RED	49
Εικόνα 4 Βάση δεδομένων	51
Εικόνα 5 Dashboards στην πλατφόρμα Grafana	52
Εικόνα 6 Current System Status Dashboard	53
Εικόνα 7 Energy Monitoring Overview Dashboard	55
Εικόνα 8 KPIs Dashboard	56
Εικόνα 9 Alerts Dashboard	59
Εικόνα 10 Ορισμός contact point	60
Εικόνα 11 Ειδοποίηση Υψηλής Κατανάλωσης	61
Εικόνα 12 Ειδοποίηση Χαμηλής Μπαταρίας	61
Εικόνα 13 Ειδοποίηση Χαμηλής παραγωγής ΑΠΕ	62
Εικόνα 14 Ειδοποίηση μέσω e-mail	63
Εικόνα 15 Σύγκριση μεταξύ πραγματικής και προβλεπόμενης ισχύος	70

Κατάλογος Πινάκων

Πίνακας 1 Σύνδεση απαιτήσεων με τον τρόπο υλοποίησης	39
Πίνακας 2 Αποτελέσματα Δοκιμών	77
Πίνακας 3 Αποτελέσματα Ερωτηματολογίου	79

1 Εισαγωγή

1.1 Γενικό Πλαίσιο

Η ενεργειακή μετάβαση αποτελεί μία από τις πιο κρίσιμες προκλήσεις της εποχής μας. Η αυξανόμενη ζήτηση για ηλεκτρική ενέργεια, η ανάγκη μείωσης των εκπομπών διοξειδίου του άνθρακα και η αστάθεια των τιμών των ορυκτών καυσίμων ωθούν επιχειρήσεις και οργανισμούς να στραφούν σε Ανανεώσιμες Πηγές Ενέργειας (ΑΠΕ), όπως τα φωτοβολταϊκά και τα αιολικά συστήματα. Οι ΑΠΕ αποτελούν πλέον την ταχύτερα αναπτυσσόμενη κατηγορία ενεργειακών πόρων παγκοσμίως, με το μεσοσταθμικό κόστος παραγωγής να έχει μειωθεί δραματικά την τελευταία δεκαετία. Παράλληλα, ο ψηφιακός μετασχηματισμός της βιομηχανίας και των επιχειρήσεων δημιουργεί νέες δυνατότητες στη διαχείριση ενέργειας. Τεχνολογίες όπως το Internet of Things (IoT), η Μηχανική Μάθηση (Machine Learning) και η ανάλυση μεγάλων δεδομένων επιτρέπουν την παρακολούθηση σε πραγματικό χρόνο, την ανάλυση και τη βελτιστοποίηση της ενεργειακής απόδοσης σε επίπεδο που δεν ήταν εφικτό στο παρελθόν. Τα Ευφυή Συστήματα Ενεργειακής Διαχείρισης (Intelligent Energy Management Systems — IEMS) αποτελούν τη σύγκλιση αυτών των τεχνολογιών, μετατρέποντας τα παθητικά συστήματα παρακολούθησης σε ενεργά εργαλεία λήψης αποφάσεων.

1.2 Σκοπός και Στόχοι της Εργασίας

Σκοπός της παρούσας διπλωματικής εργασίας είναι η σχεδίαση, ανάπτυξη και αξιολόγηση ενός Ευφυούς Συστήματος Ενεργειακής Παρακολούθησης για επιχείρηση που αξιοποιεί φωτοβολταϊκές και αιολικές λύσεις. Το σύστημα παρέχει real-time παρακολούθηση, πρόβλεψη ενεργειακής παραγωγής και αυτοματοποιημένη ανίχνευση ανωμαλιών, με διαφοροποιημένη πρόσβαση ανά ρόλο χρήστη.

Οι επιμέρους στόχοι της εργασίας είναι:

- Η **σχεδίαση αρχιτεκτονικής** πολλαπλών επιπέδων βασισμένης σε τεχνολογίες IoT, η οποία να υποστηρίζει συνεχή ροή δεδομένων με χαμηλή καθυστέρηση.

- Η **ανάπτυξη προσομοιωτή** που να αναπαριστά ρεαλιστικά τη συμπεριφορά ενεργειακού δικτύου με φωτοβολταϊκά, αιολικά και σύστημα αποθήκευσης.
- Η **υλοποίηση pipeline επεξεργασίας δεδομένων** μέσω ενδιάμεσου επιπέδου middleware, το οποίο να εκτελεί φιλτράρισμα, μετασχηματισμό και δρομολόγηση δεδομένων.
- Η **ανάπτυξη μοντέλου πρόβλεψης** φωτοβολταϊκής παραγωγής με χρήση Γραμμικής Παλινδρόμησης και η ενσωμάτωσή του σε microservice πραγματικού χρόνου για ανίχνευση ανωμαλιών.
- Η **δημιουργία διαδραστικών dashboards** οργανωμένων ανά επίπεδο πληροφορίας, με υπολογισμό Δεικτών Απόδοσης (KPIs) και μηχανισμό ειδοποιήσεων μέσω email.
- Η **υλοποίηση συστήματος ρόλων χρηστών** που να παρέχει διαφοροποιημένη εμπειρία ανάλογα με τις αρμοδιότητες κάθε χρήστη στον οργανισμό.

Η **συστηματική αξιολόγηση** του συστήματος μέσω αυτοματοποιημένων δοκιμών που καλύπτουν unit, integration και σενάρια πραγματικών συνθηκών.

1.3 Δομή της Εργασίας

Η εργασία οργανώνεται ως εξής:

Το Κεφάλαιο 2 παρουσιάζει το θεωρητικό πλαίσιο, καλύπτοντας τα Συστήματα Ενεργειακής Διαχείρισης (EMS) και τα Ευφυή Συστήματα (IEMS), τις τεχνολογίες ΑΠΕ, τις αρχές IoT και Smart Grids, τα συστήματα οπτικοποίησης και τις τεχνικές πρόβλεψης ενεργειακών χρονοσειρών.

Το Κεφάλαιο 3 περιγράφει τη σχεδίαση και αρχιτεκτονική του συστήματος, συμπεριλαμβανομένης της ανάλυσης λειτουργικών και μη λειτουργικών απαιτήσεων, της γενικής αρχιτεκτονικής IoT, της μοντελοποίησης βάσης δεδομένων και του σχεδιασμού των dashboards.

Το Κεφάλαιο 4 παρουσιάζει αναλυτικά την υλοποίηση του συστήματος: το περιβάλλον Docker, τον προσομοιωτή IoT, τον MQTT broker, το ενδιάμεσο επίπεδο Node-RED, την αποθήκευση SurrealDB, τα dashboards Grafana, το σύστημα ειδοποιήσεων και τη διαχείριση ρόλων χρηστών.

Το Κεφάλαιο 5 εστιάζει στην ανάλυση δεδομένων, περιγράφοντας την επιλογή και εκπαίδευση του μοντέλου Γραμμικής Παλινδρόμησης για πρόβλεψη φωτοβολταϊκής παραγωγής, τη μεθοδολογία ανίχνευσης ανωμαλιών και την οπτικοποίηση αποτελεσμάτων.

Το Κεφάλαιο 6 παρουσιάζει τη μεθοδολογία δοκιμών και τα αποτελέσματα αξιολόγησης, καλύπτοντας unit tests, integration tests της πλήρους ροής ειδοποιήσεων, σενάρια δοκιμών του microservice πρόβλεψης και δοκιμές ρόλων χρηστών.

Τέλος, στα Παραρτήματα παρατίθεται ο πλήρης κώδικας των βασικών υποσυστημάτων: ο προσομοιωτής IoT, οι συναρτήσεις Node-RED, το script εκπαίδευσης μοντέλου και το microservice πρόβλεψης.

2 Θεωρητικό πλαίσιο

2.1 Συστήματα Ενεργειακής Διαχείρισης (Energy Management Systems – EMS)

Η ζήτηση για ενέργεια συνεχώς αυξάνεται, σε συνδυασμό με τις απαιτήσεις για λιγότερους ρύπους και καλύτερη απόδοση, έχει καταστήσει απαραίτητα τα Συστήματα Ενεργειακής Διαχείρισης (Energy Management Systems – EMS) τόσο για οργανισμούς όσο και βιομηχανικές εγκαταστάσεις. Τα EMS αποτελούν ολοκληρωμένα πληροφοριακά και λειτουργικά συστήματα που υποστηρίζουν τη παρακολούθηση, ανάλυση και βελτιστοποίηση της κατανάλωσης και της παραγωγής ενέργειας. Επιπλέον συμβάλλουν στη μείωση κόστους και στη βελτίωση της ενεργειακής απόδοσης (Capehart, Turner, & Kennedy, 2020). Ένα EMS παρέχει διαδικασίες και εργαλεία τα οποία με τη βοήθεια λογισμικού συλλέγουν, οργανώνουν και αναλύουν ενεργειακά δεδομένα. Αυτό δίνει τη δυνατότητα λήψης αποφάσεων με βάση τα πραγματικά στοιχεία και πάντα με στόχο τη βελτίωση της ενεργειακής επίδοσης (Siano, 2014).

Όσον αφορά την αρχιτεκτονική, τα EMS βασίζονται σε μια πολυεπίπεδη αρχιτεκτονική από τη συλλογή των δεδομένων μέχρι και τη χρήση τους για επιχειρησιακές αποφάσεις. Τα δεδομένα συλλέγονται από αισθητήρες, έξυπνους μετρητές και συσκευές Internet of Things (IoT), δημιουργώντας μια συνεχόμενη ροή δεδομένων με πολύ καλή χρονική ανάλυση. Αυτά τα δεδομένα αποθηκεύονται σε βάσεις χρονοσειρών, κάτι που επιτρέπει τη διαχείριση μεγάλου όγκου μετρήσεων και τη διατήρηση ιστορικότητας. Στη συνέχεια εφαρμόζονται μέθοδοι ανάλυσης και επεξεργασίας, οι οποίες περιλαμβάνουν στατιστικές τεχνικές, μοντέλα πρόβλεψης και αλγορίθμους βελτιστοποίησης. Στόχος είναι η εξισορρόπηση φορτίου, η βελτίωση της ενεργειακής αποδοτικότητας και η μείωση αιχμών ζήτησης. Τα αποτελέσματα της ανάλυσης χρησιμοποιούνται ως δείκτες απόδοσης και ενδείξεις με τη βοήθεια εργαλείων οπτικοποίησης, ενώ ταυτόχρονα υπάρχουν μηχανισμοί ειδοποιήσεων και αυτοματισμοί που επιτρέπουν άμεση αντίδραση όταν εντοπίζονται αποκλίσεις.

Σήμερα, τα EMS εξελίσσονται γρήγορα, καθώς ενσωματώνουν τεχνολογίες όπως το Internet of Things, ανάλυση μεγάλων δεδομένων, cloud και machine learning. Με τον

συνδιασμό αυτών των τεχνολογιών δίνεται η δυνατότητα επεξεργασίας δεδομένων σε πραγματικό χρόνο και υποστηρίζεται η ανάπτυξη συστημάτων που παίρνουν αποφάσεις με μεγαλύτερο βαθμό αυτονομίας και προβλεπτικής ικανότητας (Zhang et al., 2018). Έτσι, τα EMS τα EMS δεν είναι πια απλά εργαλεία παρακολούθησης αλλά μετατρέπονται σε ενεργά συστήματα που πραγματικά επηρεάζουν και διαμορφώνουν τη στρατηγική κάθε οργανισμού.

2.2 Ευφυή Συστήματα Ενεργειακής Διαχείρισης (Intelligent Energy Management Systems - IEMS)

2.2.1 Βασικές έννοιες και αρχές των IEMS

Τα παραδοσιακά Συστήματα Ενεργειακής Διαχείρισης (EMS) εξελίσσονται σε Ευφυή Συστήματα Ενεργειακής Διαχείρισης (Intelligent Energy Management Systems – IEMS). Αυτή η εξέλιξη είναι αποτέλεσμα της ενσωμάτωσης τεχνολογιών Τεχνητής Νοημοσύνης (AI), Μηχανικής Μάθησης (ML), Big Data Analytics και Internet of Things (IoT). Τα συμβατικά EMS επικεντρώνονται κυρίως στην παρακολούθηση και ανάλυση ιστορικών δεδομένων. Από την άλλη πλευρά, τα IEMS υποστηρίζουν δυναμική πρόβλεψη, βελτιστοποίηση και αυτοματοποιημένη λήψη αποφάσεων σε πραγματικό χρόνο (Zhang et al., 2018).

Τα IEMS είναι ολοκληρωμένες ψηφιακές πλατφόρμες. Συνδυάζουν το υλικό (αισθητήρες, έξυπνους μετρητές, συστήματα αποθήκευσης) με λογισμικό ανάλυσης. Έτσι, παρακολουθούν συνεχώς την ενέργεια, προβλέπουν και βελτιώνουν την απόδοση, ακόμα και σε πολύπλοκα ενεργειακά περιβάλλοντα (Lopes et al., 2020). Σε επιχειρησιακά δίκτυα που βασίζονται σε φωτοβολταϊκά και αιολικά, τα IEMS αναλαμβάνουν να διαχειριστούν τεράστιο όγκο δεδομένων με μεγάλη ακρίβεια στον χρόνο, ενώ ταυτόχρονα αντιμετωπίζουν την αβεβαιότητα που φέρνει η στοχαστική φύση των ανανεώσιμων πηγών (Lund et al., 2015). Από την οπτική των πληροφοριακών συστημάτων, τα IEMS μπορούν να χαρακτηριστούν ως υβριδικά συστήματα υποστήριξης αποφάσεων (Decision Support

Systems – DSS), αφού δουλεύουν σε πραγματικό χρόνο, κάνουν προγνωστική ανάλυση και προσφέρουν οπτικοποίηση της πληροφορίας (Wang et al., 2014).

Με τα IEMS, οι επιχειρήσεις μπορούν να προβλέψουν βλάβες και να εντοπίσουν ανωμαλίες. Αυτό σημαίνει λιγότερα έξοδα λειτουργίας και λιγότερες απρογραμμάτιστες διακοπές. Παράλληλα, τα IEMS βοηθούν να αξιοποιούνται καλύτερα οι Ανανεώσιμες Πηγές Ενέργειας και να μειώνεται το ανθρακικό αποτύπωμα με έξυπνη διαχείριση φορτίου. Επιπλέον, προσφέρουν εργαλεία αξιολόγησης της απόδοσης επενδύσεων (ROI) σε έργα ενεργειακής αποδοτικότητας, συνεισφέροντας στη στρατηγική και τεκμηριωμένη λήψη αποφάσεων (Zheng et al., 2020).

Η λειτουργία ενός Intelligent Energy Management System (IEMS) μπορεί να περιγραφεί μέσω μιας αρχιτεκτονικής τριών βασικών επιπέδων (Zheng et al., 2020). Στο επίπεδο αντίληψης (Perception Layer) πραγματοποιείται η συλλογή δεδομένων σε πραγματικό χρόνο από φωτοβολταϊκά και αιολικά συστήματα, έξυπνους μετρητές και αισθητήρες IoT, διαμορφώνοντας το πληροφοριακό υπόβαθρο του συστήματος. Το επίπεδο επεξεργασίας (Processing Layer) χρησιμοποιεί αλγορίθμους ανάλυσης και βελτιστοποίησης για τη διαχείριση της κατανομής ενέργειας μεταξύ παραγωγής, κατανάλωσης και αποθήκευσης, ενσωματώνοντας τεχνικές εξισορρόπησης φορτίου (load balancing) και μηχανισμούς demand response για τη μείωση της ζήτησης σε περιόδους αιχμής. Τέλος, στο επίπεδο εφαρμογής (Application Layer) υλοποιούνται διαδραστικά dashboards και μηχανισμοί ειδοποίησης, τα οποία παρέχουν δείκτες απόδοσης και υποστηρίζουν τη λήψη αποφάσεων σε πραγματικό χρόνο.

2.2.2 Συλλογή και Διαχείριση Δεδομένων Ενέργειας σε Πραγματικό Χρόνο

Η λειτουργία ενός IEMS στηρίζεται στη συνεχή συλλογή δεδομένων από διάφορες πηγές, όπως φωτοβολταϊκά και αιολικά συστήματα, έξυπνους μετρητές, αισθητήρες IoT και εξωτερικές πηγές μετεωρολογικών δεδομένων (Wang et al., 2017). Τα δεδομένα αυτά είναι δεδομένα χρονοσειρών (time-series data) και χρειάζονται υποδομές που μπορούν να προσφέρουν επεξεργασία σε πραγματικό χρόνο και αξιόπιστη αποθήκευση.

Με τις τεχνολογίες IoT και τα πρωτόκολλα επικοινωνίας χαμηλής καθυστέρησης, όπως το MQTT και τα RESTful APIs, τα δεδομένα φτάνουν γρήγορα και αποτελεσματικά στα κεντρικά πληροφοριακά συστήματα (Al-Fuqaha et al., 2015). Παράλληλα, η ανάγκη για επεκτασιμότητα και ανθεκτικότητα οδηγεί στην επιλογή κατακεντρωμένων και cloud-based αρχιτεκτονικών (Chen et al., 2014).

2.2.3 Αρχιτεκτονική Συστήματος Real-time Παρακολούθησης και Πρόβλεψης

Η αρχιτεκτονική ενός συστήματος real-time παρακολούθησης και πρόβλεψης ενεργειακής απόδοσης στηρίζεται σε σύγχρονες αρχές σχεδίασης πληροφοριακών συστημάτων και χωρίζεται σε διακριτά επίπεδα (Lopes et al., 2020· Chen et al., 2014).

2.2.3.1 Επίπεδο Πηγών Δεδομένων (Data Sources Layer)

Το επίπεδο αυτό περιλαμβάνει τις ενεργειακές υποδομές (φωτοβολταϊκά και ανεμογεννήτριες), τους έξυπνους μετρητές, τους αισθητήρες και εξωτερικές πηγές δεδομένων, όπως υπηρεσίες πρόγνωσης καιρού. Οι πηγές αυτές παράγουν συνεχείς ροές δεδομένων, οι οποίες αποτελούν τη βάση για την περαιτέρω ανάλυση (Wang et al., 2017).

2.2.3.2 Επίπεδο Εισαγωγής Δεδομένων (Data Ingestion Layer)

Το επίπεδο εισαγωγής δεδομένων είναι υπεύθυνο για τη συλλογή και δρομολόγηση των δεδομένων σε πραγματικό χρόνο. Με τη χρήση μηχανισμών message brokering και streaming μπορεί να γίνεται διαχείριση μεγάλου όγκου δεδομένων με χαμηλή καθυστέρηση και μεγάλη αξιοπιστία (Chen et al., 2014).

2.2.3.3 Επίπεδο Αποθήκευσης και Επεξεργασίας Δεδομένων

Σε αυτό το επίπεδο γίνεται η αποθήκευση ιστορικών δεδομένων, ο καθαρισμός και η προεπεξεργασία τους, καθώς και ο υπολογισμός βασικών δεικτών απόδοσης (Key Performance Indicators – KPIs), όπως η συνολική παραγόμενη ενέργεια και το ποσοστό αξιοποίησης των ανανεώσιμων πηγών ενέργειας (Lund et al., 2015).

2.2.3.4 Επίπεδο Analytics και Πρόβλεψης

Το επίπεδο analytics περιλαμβάνει αλγορίθμους μηχανικής μάθησης για την πρόβλεψη παραγωγής και κατανάλωσης ενέργειας. Ιδιαίτερα διαδεδομένα είναι τα νευρωνικά δίκτυα τύπου Long Short-Term Memory (LSTM), τα οποία παρουσιάζουν αυξημένη ακρίβεια στην πρόβλεψη χρονοσειρών με έντονες χρονικές εξαρτήσεις (Hochreiter & Schmidhuber, 1997). Αυτές οι προβλέψεις είναι κρίσιμες για τη βελτιστοποίηση της λειτουργίας του δικτύου και για τη λήψη κρίσιμων επιχειρησιακών αποφάσεων (Wang et al., 2014).

2.2.3.5 Επίπεδο Παρουσίασης – Dashboard

Το επίπεδο παρουσίασης γίνεται μέσω ενός διαδραστικού dashboard, το οποίο παρέχει real-time οπτικοποίηση δεδομένων, ιστορικές αναλύσεις και συγκριτική παρουσίαση προβλέψεων και πραγματικών τιμών. Τα dashboards αποτελούν βασικό εργαλείο υποστήριξης αποφάσεων, αφού μετατρέπουν σύνθετα δεδομένα σε κατανοητή και αξιοποιήσιμη πληροφορία (Few, 2013).

2.2.4 Τεχνολογίες Τεχνητής Νοημοσύνης στα IEMS

Η ενσωμάτωση τεχνολογιών Τεχνητής Νοημοσύνης (Artificial Intelligence – AI) στα Intelligent Energy Management Systems (IEMS) αποτελεί βασικό παράγοντα που διαφοροποιεί τα συστήματα αυτά από τα συμβατικά EMS. Οι τεχνικές AI δίνουν τη δυνατότητα ανάλυσης μεγάλου όγκου ετερόκλητων δεδομένων, την εύρεση μοτίβων, την πρόβλεψη μελλοντικών καταστάσεων και τη λήψη σωστών αποφάσεων. Στα IEMS η AI εστιάζει κυρίως σε τέσσερις βασικούς τομείς: πρόβλεψη φορτίου, πρόβλεψη παραγωγής από ανανεώσιμες πηγές ενέργειας (ΑΠΕ), βελτιστοποίηση ενεργειακής λειτουργίας και ανίχνευση ανωμαλιών.

2.2.4.1 Πρόβλεψη φορτίου (Load Forecasting)

Η πρόβλεψη φορτίου είναι μία από τις πιο κρίσιμες λειτουργίες των IEMS, καθώς επιτρέπει την εκτίμηση της μελλοντικής ενεργειακής ζήτησης σε βραχυπρόθεσμο, μεσοπρόθεσμο ή μακροπρόθεσμο ορίζοντα. Όσο πιο ακριβής είναι η πρόβλεψη φορτίου, τόσο καλύτερα κατανέμονται οι ενεργειακοί πόροι, μειώνονται οι αιχμές κατανάλωσης (peak shaving) και μπορεί να γίνει πιο αποτελεσματικός προγραμματισμός παραγωγής και αποθήκευσης

ενέργειας (Hong & Fan, 2016). Για τις επιχειρήσεις, αυτό σημαίνει μικρότερο κόστος γιατί δεν σπαταλούν τις υποδομές τους άσκοπα.

Στη βιβλιογραφία έχουν αναπτυχθεί και εφαρμοστεί διάφορες μέθοδοι πρόβλεψης. Τα στατιστικά μοντέλα, όπως το AutoRegressive Integrated Moving Average (ARIMA), αποτελούν κλασική λύση για την ανάλυση χρονοσειρών, αξιοποιώντας τη γραμμική συσχέτιση των δεδομένων στο χρόνο. Τα μοντέλα αυτά παρουσιάζουν ικανοποιητική απόδοση σε σταθερά και γραμμικά μοτίβα κατανάλωσης, αλλά συχνά δεν μπορούν να αποτυπώσουν μη γραμμικές και πολύπλοκες σχέσεις.

Για τον λόγο αυτό, χρησιμοποιούνται ευρέως τεχνικές μηχανικής μάθησης, όπως τα Support Vector Machines (SVM) και τα Random Forests. Τα SVM χρησιμοποιούνται για παλινδρόμηση και ταξινόμηση ακόμη και σε μικρά σύνολα δεδομένων, ενώ τα Random Forest προσφέρουν ανθεκτικότητα στον θόρυβο και καλή απόδοση σε σύνθετα προβλήματα (Zhang et al., 2018).

Τα τελευταία χρόνια, τα νευρωνικά δίκτυα τύπου Long Short-Term Memory (LSTM), τα οποία ανήκουν στην κατηγορία των επαναλαμβανόμενων νευρωνικών δικτύων (Recurrent Neural Networks – RNN) παρουσιάζουν ιδιαίτερο ενδιαφέρον. Τα LSTM είναι σχεδιασμένα να διαχειρίζονται εξαρτήσεις μεγάλης χρονικής διάρκειας. Αυτό τα κάνει κατάλληλα για ενεργειακά δεδομένα χρονοσειρών, όπου παίζουν ρόλο οι εποχικές και ημερήσιες διακυμάνσεις. Σύμφωνα με μελέτες τα LSTM φέρνουν καλύτερα αποτελέσματα σε σύγκριση με τα παραδοσιακά στατιστικά μοντέλα, κυρίως σε περιβάλλοντα με μεγάλες μεταβολές φορτίου (Hong & Fan, 2016).

2.2.4.2 Πρόβλεψη παραγωγής από Ανανεώσιμες Πηγές Ενέργειας

Η πρόβλεψη παραγωγής από ΑΠΕ είναι σημαντική για τα IEMS, ειδικά σε συστήματα που βασίζονται σε μεγάλο βαθμό σε φωτοβολταϊκές και αιολικές εγκαταστάσεις. Η παραγωγή από τις πηγές αυτές δεν είναι σταθερή, αλλά εξαρτάται άμεσα από περιβαλλοντικές μεταβλητές, όπως η ηλιακή ακτινοβολία, η θερμοκρασία και η ταχύτητα ανέμου. Η αβεβαιότητα αυτή καθιστά απαραίτητη την εφαρμογή προγνωστικών μοντέλων που

επιτρέπουν την εκτίμηση της διαθέσιμης ισχύος σε μελλοντικά χρονικά διαστήματα (Zhang et al., 2018).

Οι τεχνικές που χρησιμοποιούνται μοιάζουν με εκείνες της πρόβλεψης φορτίου και περιλαμβάνουν στατιστικά μοντέλα, μηχανική μάθηση και βαθιά μάθηση (deep learning). Σε φωτοβολταϊκά συστήματα, τα νευρωνικά δίκτυα αξιοποιούν τόσο τα ιστορικά δεδομένα παραγωγής όσο και τις μετεωρολογικές προβλέψεις για να εκτιμήσουν την παραγόμενη ισχύ. Η ακριβής πρόβλεψη παραγωγής επιτρέπει την καλύτερη διαχείριση των αποθηκευτικών συστημάτων, τη βελτιστοποίηση της αυτοκατανάλωσης και τη μείωση της εξάρτησης από το κεντρικό δίκτυο.

2.2.4.3 Βελτιστοποίηση ενέργειας

Πέρα από την πρόβλεψη, τα IEMS χρησιμοποιούν τεχνικές βελτιστοποίησης για τη λήψη αποφάσεων σχετικά με την κατανομή και τη χρήση ενεργειακών πόρων. Το πρόβλημα της ενεργειακής βελτιστοποίησης διατυπώνεται συνήθως ως πρόβλημα ελαχιστοποίησης κόστους ή εκπομπών, υπό περιορισμούς που αφορούν τη διαθεσιμότητα παραγωγής, τη χωρητικότητα αποθήκευσης και τις τεχνικές δυνατότητες του συστήματος.

Οι κλασικές μέθοδοι είναι ο Γραμμικός Προγραμματισμός (Linear Programming – LP) και ο Μικτός Ακέραιος Γραμμικός Προγραμματισμός (Mixed Integer Linear Programming – MILP). Αυτές οι μέθοδοι βοηθούν στην επίλυση προβλημάτων κατανομής ισχύος και προγραμματισμού λειτουργίας με κάποιους περιορισμούς. Σε πιο σύνθετα και μη γραμμικά προβλήματα, χρησιμοποιούνται μέθοδοι, όπως οι Γενετικοί Αλγόριθμοι (Genetic Algorithms) και το Particle Swarm Optimization. Αυτές προσφέρουν καλές λύσεις σε περιπτώσεις που οι κλασικές μέθοδοι είναι υπολογιστικά δύσκολο να χρησιμοποιηθούν (Siano, 2014).

Η εφαρμογή τέτοιων τεχνικών σε IEMS επιτρέπει τη δυναμική επιλογή μεταξύ χρήσης αποθηκευμένης ενέργειας, αγοράς από το δίκτυο ή πώλησης πλεονάζουσας παραγωγής, με στόχο την μεγαλύτερη οικονομική και περιβαλλοντική αποδοτικότητα.

2.2.4.4 Ανίχνευση ανωμαλιών (Anomaly Detection)

Η ανίχνευση ανωμαλιών είναι βασική λειτουργία των IEMS. Επιτρέπει τον έγκαιρο εντοπισμό αποκλίσεων από την κανονική λειτουργία του συστήματος. Οι ανωμαλίες μπορεί να σχετίζονται με μη συνηθισμένη κατανάλωση, βλάβες στον εξοπλισμό ή απώλειες στην απόδοση. Ο έγκαιρος εντοπισμός τέτοιων καταστάσεων μειώνει τα λειτουργικά έξοδα, προλαμβάνει διακοπές λειτουργίας και κάνει το σύστημα πιο αξιόπιστο (Gungor et al., 2013).

Για να βρουν αυτές τις ανωμαλίες, τα IEMS χρησιμοποιούν διάφορες τεχνικές όπως στατιστικές μεθόδους, αλγορίθμους clustering και μοντέλα μηχανικής μάθησης. Αυτά τα συστήματα μαθαίνουν πώς φαίνεται η κανονική λειτουργία και επισημαίνουν τις αποκλίσεις. Σε πιο εξελιγμένα IEMS, η ανίχνευση ανωμαλιών συνδυάζεται με προγνωστική συντήρηση. Έτσι, ο εξοπλισμός επισκευάζεται ή αλλάζεται πριν από την εμφάνιση κάποιας σοβαρής βλάβης.

Γενικά, η Τεχνητή Νοημοσύνη δίνει στα IEMS τη δυνατότητα να προσαρμόζονται μόνα τους και να βελτιώνονται συνεχώς. Γι' αυτό και τα συναντάμε όλο και πιο συχνά στη σύγχρονη διαχείριση ενέργειας, είτε πρόκειται για μεγάλες επιχειρήσεις είτε για μικροδίκτυα.

2.2.5 Προκλήσεις Υλοποίησης IEMS

Τα Intelligent Energy Management Systems (IEMS) δίνουν πολλά πλεονεκτήματα, αλλά η υλοποίησή τους συνοδεύεται από ένα σύνολο τεχνικών, οργανωτικών και οικονομικών δυσκολιών, οι οποίες επηρεάζουν τόσο την αποτελεσματικότητα όσο και τη βιωσιμότητα των σχετικών επενδύσεων. Η πολυπλοκότητα των ενεργειακών συστημάτων και η ενσωμάτωση τεχνολογιών τεχνητής νοημοσύνης και IoT δημιουργεί ένα ιδιαίτερα απαιτητικό περιβάλλον ανάπτυξης και λειτουργίας (Siano, 2014).

Μία από τις σημαντικότερες προκλήσεις είναι η διαχείριση μεγάλου όγκου δεδομένων (big data management). Τα IEMS συλλέγουν δεδομένα από αισθητήρες, έξυπνους μετρητές, φωτοβολταϊκές εγκαταστάσεις, ανεμογεννήτριες και συστήματα αποθήκευσης. Τα δεδομένα αυτά χαρακτηρίζονται από μεγάλο όγκο, υψηλή ταχύτητα παραγωγής και

ετερογένεια. Η αποθήκευση, η επεξεργασία και η ανάλυσή τους απαιτούν υποδομές υψηλής υπολογιστικής ισχύος και εξειδικευμένες βάσεις χρονοσειρών. Επιπλέον, η ποιότητα των δεδομένων παίζει τεράστιο ρόλο. Αν τα δεδομένα είναι ελλιπή ή λανθασμένα μπορούν να επηρεάσουν αρνητικά την ακρίβεια των προγνωστικών μοντέλων (Zhang et al., 2018).

Τα συστήματα IEMS πρέπει να διασυνδεθούν με συστήματα από διάφορους κατασκευαστές χρησιμοποιώντας διάφορα πρωτόκολλα επικοινωνίας (όπως MQTT, Modbus, OPC-UA) και να συνεργαστούν με υπάρχουσες ενεργειακές υποδομές. Όταν δεν υπάρχουν κοινά πρότυπα ή η δεν ακολουθείται η συμμόρφωση με τα διεθνή πρότυπα δυσκολεύει η διασύνδεση και αυξάνεται η πολυπλοκότητα της υλοποίησης. Για να γίνει ενοποίηση δεδομένων από διαφορετικές πηγές είναι απαραίτητη η ανάπτυξη middleware λύσεων και μηχανισμών μετατροπής δεδομένων, γεγονός που φέρνει επιπλέον τεχνικές δυσκολίες και ανεβάζει το κόστος (Gungor et al., 2013).

Ιδιαίτερα σημαντικό ζήτημα αποτελεί η κυβερνοασφάλεια (cybersecurity). Τα IEMS διαχειρίζονται κρίσιμες υποδομές και ελέγχουν βασικές λειτουργίες, γεγονός που τα καθιστά πιθανούς στόχους κυβερνοεπιθέσεων. Η διασύνδεση μέσω διαδικτύου, η χρήση cloud υποδομών και η ανταλλαγή δεδομένων σε πραγματικό χρόνο αυξάνουν την πιθανότητα κάποιας επίθεσης. Πιθανές απειλές αποτελούν η μη εξουσιοδοτημένη πρόσβαση, η αλλοίωση δεδομένων, οι επιθέσεις άρνησης υπηρεσίας (DoS) και ο κακόβουλος έλεγχος ενεργειακών συστημάτων. Η ασφάλεια κυβερνοχώρου στα smart grids αποτελεί βασική προϋπόθεση για την αξιόπιστη και ασφαλή λειτουργία τους (Gungor et al., 2013). Για τον λόγο αυτό είναι αναγκαία η ενσωμάτωση μηχανισμών κρυπτογράφησης, αυθεντικοποίησης χρηστών, διαχείρισης προσβάσεων και συνεχούς παρακολούθησης απειλών.

Ένα ακόμα σηναβτικό ζήτημα αποτελεί η ακρίβεια των προγνωστικών μοντέλων. Τα IEMS βασίζονται σε αλγορίθμους μηχανικής μάθησης για να προβλέπουν τη ζήτηση και την παραγωγή από ΑΠΕ. Η απόδοση των μοντέλων αυτών εξαρτάται από την ποιότητα και την επάρκεια των ιστορικών δεδομένων, την ορθή επιλογή χαρακτηριστικών και το αν έγινε σωστή παραμετροποίηση των αλγορίθμων. Σε περιβάλλοντα που τα πάντα αλλάζουν γρήγορα, όπως σε δίκτυα με πολύ μεγάλη χρήση ανανεώσιμων πηγών, η αβεβαιότητα

μπορεί να οδηγήσει σε σφάλματα πρόβλεψης, τα οποία επηρεάζουν άμεσα τη βελτιστοποίηση και τη λήψη αποφάσεων (Hong & Fan, 2016). Επομένως, χρειάζεται συνεχής ανατροφοδότηση και επανεκπαίδευση των μοντέλων, ώστε να διατηρείται υψηλό το επίπεδο αξιοπιστίας.

Τέλος, το κόστος των υποδομών και της υλοποίησης αποτελεί σημαντικό περιοριστικό παράγοντα. Η ανάπτυξη ενός IEMS απαιτεί επένδυση σε αισθητήρες, έξυπνους μετρητές, συστήματα αποθήκευσης, δικτυακές υποδομές, υπολογιστικούς πόρους (on-premise ή cloud) και εξειδικευμένο προσωπικό. Για μικρομεσαίες επιχειρήσεις, το αρχικό κόστος μπορεί να είναι αποτρεπτικό, ακόμα κι αν μακροπρόθεσμα η επένδυση αποδώσει λόγω εξοικονόμησης και καλύτερης λειτουργίας (McKane et al., 2017).

Συνοψίζοντας, οι προκλήσεις στην υλοποίηση IEMS είναι πολλές και σημαντικές αλλά δεν αναιρούν τα σημαντικά οφέλη τους. Αναδεικνύουν ότι χρειάζεται μία ολιστική προσέγγιση στο σχεδιασμό, η οποία θα λαμβάνει υπόψη τόσο τις τεχνικές απαιτήσεις όσο και τους οργανωτικούς και οικονομικούς περιορισμούς. Για να πετύχει πραγματικά ένα IEMS, χρειάζεται στρατηγικός σχεδιασμός, συμμόρφωση με πρότυπα ασφάλειας και ποιότητας, καθώς και συνεχής αξιολόγηση της απόδοσης και των κινδύνων.

2.3 Ανανεώσιμες Πηγές Ενέργειας: Φωτοβολταϊκά και Αιολικά Συστήματα

Η ένταξη των Ανανεώσιμων Πηγών Ενέργειας (ΑΠΕ) στα εταιρικά ενεργειακά δίκτυα αλλάζει ριζικά τον τρόπο που οι επιχειρήσεις παράγουν και διαχειρίζονται την ενέργειά τους. Παγκοσμίως, η μετάβαση σε ένα πιο βιώσιμο, χαμηλών εκπομπών άνθρακα μοντέλο ανάπτυξης στηρίζεται κυρίως στις ΑΠΕ. Η ανησυχία για την κλιματική αλλαγή αυξάνεται, η ανάγκη για ενεργειακή ασφάλειας γίνεται πιο σημαντική από ποτέ και οι επιχειρήσεις στρέφονται σταδιακά μακριά από τα ορυκτά καύσιμα. Όλα αυτά έχουν οδηγήσει σε μεγάλη ανάπτυξη τεχνολογιών όπως τα φωτοβολταϊκά και τα αιολικά συστήματα (International Energy Agency [IEA], 2023). Σε επιχειρησιακό επίπεδο, οι ΑΠΕ αλλάζουν τα ενεργειακά

δίκτυα από απλούς καταναλωτές ενέργειας σε υβριδικά συστήματα όπου μια επιχείρηση μπορεί να είναι ταυτόχρονα παραγωγός και καταναλωτής.

Τα φωτοβολταϊκά και τα αιολικά συστήματα κυριαρχούν στην αποκεντρωμένη παραγωγή ηλεκτρικής ενέργειας λόγω της τεχνολογικής τους ωριμότητας, της μείωσης του κόστους και της υψηλής διαθεσιμότητας φυσικών πόρων. Σύμφωνα με την IRENA (2023), το μεσοσταθμικό κόστος παραγωγής ηλεκτρικής ενέργειας (Levelized Cost of Electricity – LCOE) από φωτοβολταϊκά και αιολικά συστήματα έχει μειωθεί σημαντικά την τελευταία δεκαετία, κάνοντάς τα ανταγωνιστικά και οικονομικότερα από τις συμβατικές θερμικές μονάδες.

2.3.1 Φωτοβολταϊκά Συστήματα

Τα φωτοβολταϊκά (Photovoltaic – PV) συστήματα βασίζονται στο φωτοηλεκτρικό φαινόμενο. Σύμφωνα με αυτό η προσπίπτουσα ηλιακή ακτινοβολία προκαλεί τη δημιουργία ηλεκτρικού ρεύματος σε ημιαγωγικά υλικά. Η λειτουργία των φωτοβολταϊκών κυψελών στηρίζεται σε δομές p-n junction, όπου η απορρόφηση φωτονίων δημιουργεί ζεύγη ηλεκτρονίων–οπών, οδηγώντας σε διαφορά δυναμικού και παραγωγή συνεχούς ρεύματος (Green, 2012).

Η βασική δομή ενός φωτοβολταϊκού συστήματος περιλαμβάνει φωτοβολταϊκά πάνελ (modules), μετατροπέα ισχύος (inverter) για τη μετατροπή συνεχούς ρεύματος (DC) σε εναλλασσόμενο (AC), σύστημα στήριξης και προσανατολισμού, σύστημα παρακολούθησης παραγωγής (monitoring system) και αρκετές φορές και σύστημα αποθήκευσης ενέργειας με χρήση συσσωρευτών. Σε επιχειρησιακές εγκαταστάσεις, τα PV συστήματα μπορούν να εγκατασταθούν σε στέγες, σε σκέπαστρα πάρκινγκ ή σε πάρκα και είναι συνδεδεμένα είτε με το δίκτυο (grid-connected) είτε μέσα σε μικροδίκτυα.

Η απόδοση ενός φωτοβολταϊκού συστήματος επηρεάζεται από πολλαπλούς παράγοντες. Η ηλιακή ακτινοβολία αποτελεί τον κυριότερο παράγοντα, ενώ η θερμοκρασία επηρεάζει αρνητικά την ηλεκτρική απόδοση των κυψελών, καθώς η αύξηση θερμοκρασίας μειώνει την τάση εξόδου. Επιπλέον, η γωνία κλίσης και ο προσανατολισμός των πάνελ επηρεάζουν τη συνολική ετήσια παραγωγή, ενώ η ποιότητα και η τεχνολογία των κυψελών

(μονοκρυσταλλικές, πολυκρυσταλλικές, thin-film) καθορίζουν τον βαθμό μετατροπής (conversion efficiency) (Duffie & Beckman, 2013).

Η οικονομική βιωσιμότητα των φωτοβολταϊκών έχει βελτιωθεί σημαντικά. Σύμφωνα με την IRENA (2023), το κόστος παραγωγής από μεγάλης κλίμακας φωτοβολταϊκά μειώθηκε άνω του 80% από το 2010 έως το 2022, λόγω τεχνολογικών βελτιώσεων, οικονομιών κλίμακας και βελτιστοποίησης της εφοδιαστικής αλυσίδας. Αυτή η μείωση κόστους έχει καταστήσει τα PV συστήματα βασικό πυλώνα της ενεργειακής στρατηγικής τόσο σε εθνικό όσο και σε επιχειρησιακό επίπεδο.

Παρόλα αυτά, η διαλείπουσα φύση της ηλιακής ακτινοβολίας (λόγω νεφοκάλυψης και ημερήσιου κύκλου) δημιουργεί μεταβλητότητα στην παραγωγή. Για το λόγο αυτό είναι αναγκαία η χρήση συστημάτων αποθήκευσης και αλγορίθμων πρόβλεψης για τη βελτιστοποίηση της λειτουργίας τους (IEA, 2023).

2.3.2 Αιολικά Συστήματα

Η αιολική ενέργεια αξιοποιεί την κινητική ενέργεια του ανέμου και τη μετατρέπει σε ηλεκτρισμό με τη βοήθεια των ανεμογεννητριών. Η βασική αρχή λειτουργίας στηρίζεται στη μετατροπή της κινητικής ενέργειας του αέρα σε μηχανική ενέργεια περιστροφής και στη συνέχεια σε ηλεκτρική ενέργεια μέσω γεννήτριας.

Η παραγόμενη ισχύς μιας ανεμογεννήτριας περιγράφεται από τη σχέση:

$$P = \frac{1}{2} \rho A v^3 C_p$$

όπου ρ είναι η πυκνότητα του αέρα, A η επιφάνεια σάρωσης των πτερυγίων, v η ταχύτητα του ανέμου και C_p συντελεστής απόδοσης. Η κυβική εξάρτηση της ισχύος από την ταχύτητα ανέμου καθιστά την αιολική παραγωγή ιδιαίτερα ευαίσθητη σε μικρές μεταβολές της ταχύτητας (Ackermann, 2012).

Τα βασικά μέρη μιας ανεμογεννήτριας περιλαμβάνουν τα πτερύγια (rotor blades), τη γεννήτρια, το κιβώτιο ταχυτήτων (σε ορισμένες τεχνολογίες), τον πύργο, τον μετατροπέα ισχύος και το σύστημα ελέγχου. Τα σύγχρονα συστήματα διαθέτουν εξελιγμένους μηχανισμούς pitch control και yaw control, οι οποίοι επιτρέπουν τη ρύθμιση της γωνίας των

πτερυγίων και του προσανατολισμού της μηχανής προς τον άνεμο, βελτιστοποιώντας την απόδοση και προστατεύοντας τον εξοπλισμό.

Η αιολική ενέργεια αποτελεί μία από τις ταχύτερα αναπτυσσόμενες τεχνολογίες ΑΠΕ παγκοσμίως. Η ανάπτυξη τόσο χερσαίων όσο και υπεράκτιων πάρκων έχει ενισχύσει σημαντικά τη συμμετοχή της στο ενεργειακό μείγμα. Ωστόσο, όπως και στην περίπτωση των φωτοβολταϊκών, η στοχαστική φύση του ανέμου δημιουργεί αβεβαιότητα στην παραγωγή, απαιτώντας προηγμένες τεχνικές πρόβλεψης και εξισορρόπησης φορτίου (Ackermann, 2012).

2.3.3 Ενσωμάτωση ΑΠΕ σε Επιχειρησιακά Δίκτυα

Η ενσωμάτωση φωτοβολταϊκών και αιολικών συστημάτων σε ενεργειακά δίκτυα των επιχειρήσεων δημιουργεί σημαντικές τεχνικές και λειτουργικές δυσκολίες. Η παραγωγή τους δεν είναι σταθερή και αυτό μπορεί να οδηγήσει σε ανισορροπία μεταξύ παραγωγής και κατανάλωσης. Σαν αποτέλεσμα μπορεί να προκαλέσει διακυμάνσεις τάσης και συχνότητας στο τοπικό δίκτυο. Σε περιπτώσεις υψηλής διείσδυσης ΑΠΕ, η διατήρηση του συστήματος σταθερό γίνεται ιδιαίτερα απαιτητική (IEA, 2023).

Επιπλέον, η ανάγκη αποθήκευσης ενέργειας είναι κρίσιμη για να εξομαλυνθούν οι διακυμάνσεις της παραγωγής. Τα συστήματα αποθήκευσης, όπως οι μπαταρίες ιόντων λιθίου, επιτρέπουν την αποθήκευση πλεονάζουσας ενέργειας και την αξιοποίησή της σε περιόδους χαμηλής παραγωγής ή υψηλής ζήτησης. Ωστόσο, χρειάζεται σωστός σχεδιασμός και έλεγχος, κάτι που απαιτεί εξελιγμένα εργαλεία βελτιστοποίησης και ακριβείς προβλέψεις.

Για να διαχειριστούμε όλα αυτά τα ζητήματα αποτελεσματικά είναι αναγκαία η ύπαρξη ευφών Συστημάτων Ενεργειακής Διαχείρισης (IEMS). Αυτά τα συστήματα αναλύουν δεδομένα σε πραγματικό χρόνο, κάνουν πρόβλεψη φορτίου και παραγωγής και προσαρμόζουν δυναμικά την κατανομή της ενέργειας. Ο συνδυασμός ΑΠΕ με τεχνολογίες ψηφιακού ελέγχου και τεχνητής νοημοσύνης αποτελεί βασικό στοιχείο για τη μετάβαση σε πιο αποκεντρωμένα, ευφυή και βιώσιμα ενεργειακά δίκτυα.

2.4 IoT και Smart Grids στην ενεργειακή παρακολούθηση

Τα Smart Grids είναι η επόμενη γενιά των παραδοσιακών ηλεκτρικών δικτύων. Εσωματώνουν τεχνολογίες πληροφορικής και επικοινωνιών και επιτρέπουν την αμφίδρομη ροή ενέργειας και πληροφορίας. Η μετάβαση από τα παραδοσιακά, κεντρικά ελεγχόμενα δίκτυα σε πιο αποκεντρωμένα και ψηφιοποιημένα ενεργειακά οικοσυστήματα επιτρέπει την ενσωμάτωση διάσπαρτων ενεργειακών πόρων. Το σύστημα διαχειρίζεται το φορτίο καλύτερα και γίνεται πιο αξιόπιστο (Fang et al., 2012).

Το Internet of Things (IoT) παίζει καθοριστικό ρόλο. Καθιστά δυνατή τη συλλογή δεδομένων σε πραγματικό χρόνο από έξυπνους μετρητές (smart meters), αισθητήρες κατάστασης εξοπλισμού, inverters ΑΠΕ, συστήματα αποθήκευσης και βιομηχανικά/κτηριακά υποσυστήματα. Το Internet of Things στα ενεργειακά συστήματα φέρνει ένα πολυεπίπεδο αρχιτεκτονικό μοντέλο, στο οποίο η επεξεργασία δεδομένων και η λήψη αποφάσεων δεν περιορίζονται σε ένα κεντρικό σύστημα. Αντίθετα, κατανέμονται ιεραρχικά από το επίπεδο του πεδίου (edge) ως το υπολογιστικό νέφος (cloud). Στα παλιά κεντροποιημένα μοντέλα, όλα τα δεδομένα μεταφέρονται σε ένα κεντρικό κέντρο ελέγχου για ανάλυση. Με την IoT προσέγγιση επιτρέπεται η υλοποίηση κατανεμημένης ευφυΐας (distributed intelligence), βελτιώνοντας την αποδοτικότητα, την κλιμακωσιμότητα και την ανθεκτικότητα του συστήματος (Al-Fuqaha et al., 2015).

Στο επίπεδο edge (ή επίπεδο άκρου), περιλαμβάνονται οι συσκευές πεδίου όπως αισθητήρες, έξυπνοι μετρητές, μετατροπείς ισχύος (inverters), ελεγκτές PLC και συστήματα διαχείρισης μπαταριών. Οι συσκευές αυτές δεν λειτουργούν αποκλειστικά ως παθητικοί συλλέκτες δεδομένων, αλλά διαθέτουν υπολογιστικές δυνατότητες που επιτρέπουν την εκτέλεση βασικών λειτουργιών επεξεργασίας, όπως φιλτράρισμα δεδομένων, τοπική ανίχνευση ανωμαλιών και άμεση ενεργοποίηση μηχανισμών προστασίας. Αυτή η τοπική επεξεργασία μειώνει σημαντικά την καθυστέρηση (latency), κάτι το οποίο είναι κρίσιμο για εφαρμογές που απαιτούν ταχεία απόκριση, όπως ο έλεγχος τάσης ή η προστασία εξοπλισμού (Shi et al., 2016).

Πάνω από το επίπεδο edge, υπάρχει ένα ενδιάμεσο επίπεδο (fog ή gateway layer). Τα IoT gateways συλλέγουν δεδομένα από πολλές συσκευές, τα επεξεργάζονται (data aggregation), εφαρμόζουν τοπικούς κανόνες ελέγχου και στέλνουν τις σημαντικότερες πληροφορίες στο cloud. Η ύπαρξη αυτού του επιπέδου μειώνει το απαιτούμενο bandwidth και επιτρέπει την αποδοτικότερη διαχείριση μεγάλου όγκου δεδομένων που παράγονται σε περιβάλλοντα υψηλής πυκνότητας αισθητήρων (Al-Fuqaha et al., 2015).

Στο ανώτερο επίπεδο, το cloud layer φιλοξενεί τις μεγάλης κλίμακας υποδομές αποθήκευσης, τα εργαλεία ανάλυσης μεγάλων δεδομένων (big data analytics) και τα μοντέλα μηχανικής μάθησης που χρησιμοποιούνται για πρόβλεψη φορτίου, πρόβλεψη παραγωγής από ΑΠΕ και βελτιστοποίηση ενεργειακής λειτουργίας. Σε αυτό το επίπεδο εκπαιδεύονται σύνθετα μοντέλα όπως νευρωνικά δίκτυα LSTM, καθώς και παίρνονται αποφάσεις μακροπρόθεσμου χαρακτήρα. Το cloud επιτρέπει την κλιμάκωση (scalability) και την ενοποίηση δεδομένων από πολλαπλές εγκαταστάσεις και διευκολύνει την συνολική ενεργειακή διαχείριση (Fang et al., 2012).

Η κατακεντρωμένη αρχιτεκτονική έχει πολλά οφέλη για τα smart grids. Οι αποφάσεις παίρνονται πιο άμεσα, καθώς κρίσιμες λειτουργίες εκτελούνται τοπικά. Το σύστημα γίνεται πιο αξιόπιστο, γιατί δεν εξαρτάται αποκλειστικά από μια μόνιμη σύνδεση με το κεντρικό σύστημα. Επιτρέπει την καλύτερη διαχείριση των δικτυακών πόρων, περιορίζοντας τη μεταφορά περιττών δεδομένων. Τέλος, ενισχύει την ανθεκτικότητα έναντι κυβερνοαπειλών, καθώς η αποκέντρωση μειώνει την ύπαρξη ενός μοναδικού σημείου αποτυχίας (single point of failure) (Shi et al., 2016).

Σε ένα Intelligent Energy Management System (IEMS) η κατακεντρωμένη ευφυΐα γίνεται ακόμα πιο σημαντική. Τα δεδομένα που συλλέγονται σε πραγματικό χρόνο από έξυπνους μετρητές και αισθητήρες μπορούν να υφίστανται αρχική επεξεργασία στο edge, να αναλύονται περαιτέρω σε επίπεδο gateway και να αξιοποιούνται στο cloud για την εκτέλεση προγνωστικών και βελτιστοποιητικών αλγορίθμων. Το τελικό αποτέλεσμα παρουσιάζεται μέσω ενός κεντρικού dashboard, το οποίο λειτουργεί ως διεπαφή για την οπτικοποίηση και λήψη αποφάσεων, χωρίς όμως να αποτελεί το μοναδικό σημείο ευφυΐας του συστήματος.

Έτσι, η ευφυΐα δεν εντοπίζεται αποκλειστικά στο επίπεδο εφαρμογής, αλλά σε ολόκληρη την αρχιτεκτονική.

Συνεπώς, η πολυεπίπεδη IoT αρχιτεκτονική αποτελεί βασική προϋπόθεση για την υλοποίηση σύγχρονων smart grids και IEMS, καθώς επιτρέπει την αποτελεσματική επεξεργασία μεγάλου όγκου ενεργειακών δεδομένων, την ταχεία απόκριση σε δυναμικές μεταβολές και τη βιώσιμη κλιμάκωση των ενεργειακών συστημάτων.

2.4.1 Smart Grid- Λειτουργικές απαιτήσεις και ρόλος της επικοινωνίας

Το Smart Grid μπορεί να θεωρηθεί σαν ένα σύνθετο «σύστημα συστημάτων», στο οποίο η παραδοσιακή ηλεκτρική υποδομή παραγωγής, μεταφοράς και διανομής ενέργειας συνδέεται με ένα εκτεταμένο επικοινωνιακό και υπολογιστικό οικοσύστημα. Η ενοποίηση αυτή μετατρέπει το δίκτυο από παθητικό μηχανισμό μεταφοράς ισχύος σε δυναμικό, ψηφιακά ελεγχόμενο σύστημα, ικανό να παρακολουθεί, να αναλύει και να ρυθμίζει τη λειτουργία του σε πραγματικό χρόνο (Fang et al., 2012). Η ενεργειακή και η πληροφοριακή ροή αλληλοεξαρτώνται, καθώς η αξιόπιστη λειτουργία του δικτύου εξαρτάται πλέον από τη συνεχή ανταλλαγή δεδομένων μεταξύ συσκευών πεδίου, υποσταθμών και κεντρικών συστημάτων ελέγχου.

Οι λειτουργικές απαιτήσεις ενός Smart Grid υπερβαίνουν την απλή μεταφορά ηλεκτρικής ενέργειας. Περιλαμβάνουν προηγμένες εφαρμογές για την εκτίμηση της κατάστασης δικτύου, αυτοματισμούς διανομής, ενσωμάτωση διανεμημένων ενεργειακών πόρων (Distributed Energy Resources – DER), διαχείριση αιχμών ζήτησης και μηχανισμούς απόκρισης ζήτησης. Οι λειτουργίες αυτές απαιτούν υψηλό επίπεδο παρατηρησιμότητας (observability) και δυνατότητα έγκαιρης λήψης αποφάσεων, γεγονός που προϋποθέτει αξιόπιστες, χαμηλής καθυστέρησης επικοινωνιακές υποδομές (Fang et al., 2012).

Ο τρόπος που σχεδιάζονται τα δίκτυα επικοινωνίας παίζει καθοριστικό ρόλο σε αυτή τη μετάβαση. Τα smart grids βασίζονται σε ιεραρχημένα δίκτυα, τα οποία διακρίνονται συνήθως σε Home Area Networks (HAN), Building Area Networks (BAN), Neighbourhood Area Networks (NAN) και Wide Area Networks (WAN). Τα επίπεδα αυτά συνδέουν τις συσκευές πεδίου – όπως έξυπνους μετρητές, αισθητήρες και ελεγκτές – με τα κέντρα

ελέγχου και τις επιχειρησιακές πλατφόρμες ανάλυσης. Κάθε επίπεδο έχει διαφορετικές απαιτήσεις ως προς την καθυστέρηση μετάδοσης (latency), το εύρος ζώνης (bandwidth), την αξιοπιστία και τη διαθεσιμότητα, ανάλογα με το είδος της εφαρμογής. Για παράδειγμα, οι λειτουργίες προστασίας και αυτοματισμού απαιτούν εξαιρετικά χαμηλή καθυστέρηση και υψηλή αξιοπιστία, ενώ οι εφαρμογές απλής τηλεμέτρησης μπορούν να λειτουργούν με μεγαλύτερη ανοχή σε καθυστερήσεις (Gungor et al., 2011).

Με τη διασύνδεση αυτών των επιπέδων επικοινωνίας το δίκτυο συλλέγει δεδομένα από παντού, παρακολουθεί συνεχώς κρίσιμες παραμέτρους όπως τάση, συχνότητα και ποιότητα ισχύος, και προσαρμόζει τη λειτουργία του δυναμικά. Η επικοινωνιακή υποδομή δεν είναι απλώς ένα βοηθητικό στοιχείο του ηλεκτρικού δικτύου, αλλά αποτελεί βασική συνιστώσα της λειτουργίας του. Η αξιοπιστία και η αποτελεσματικότητα του smart grid στηρίζονται άμεσα στην ποιότητα, την ασφάλεια και την ανθεκτικότητα των επικοινωνιακών του μηχανισμών (Gungor et al., 2013).

Συνεπώς, η μετάβαση σε smart grid περιβάλλοντα δεν απαιτεί μόνο εκσυγχρονισμό της ενεργειακής υποδομής, αλλά και σχεδιασμό ενός ολοκληρωμένου επικοινωνιακού πλαισίου που να υποστηρίζει real-time παρακολούθηση, αποκεντρωμένο έλεγχο και κλιμακώσιμη ενσωμάτωση νέων ενεργειακών πόρων.

2.4.2 IoT και ενεργειακή παρακολούθηση

Στο πλαίσιο της αρχιτεκτονικής που περιγράφηκε προηγουμένως, το επίπεδο μέτρησης είναι η βάση για τη λειτουργία ενός smart grid. Η υλοποίηση της ενεργειακής παρακολούθησης ξεκινά από τη συλλογή δεδομένων υψηλής ανάλυσης μέσω έξυπνων μετρητών (smart meters) και αισθητήρων, οι οποίοι αποτελούν τον πυρήνα της Προηγμένης Υποδομής Μέτρησης (Advanced Metering Infrastructure – AMI). Η AMI δεν περιορίζεται στην απλή καταγραφή της κατανάλωσης. Είναι ένα ολοκληρωμένο σύστημα που περιλαμβάνει έξυπνους μετρητές, δίκτυα επικοινωνίας, συστήματα συγκέντρωσης δεδομένων και κεντρικές πλατφόρμες ανάλυσης. Όλα αυτά επιτρέπουν την αμφίδρομη ανάμεσα στον πάροχο και στον τελικό χρήστη (Depuru et al., 2011).

Οι έξυπνοι μετρητές παρέχουν αναλυτικά δεδομένα κατανάλωσης, υποστηρίζουν απομακρυσμένη ανάγνωση, αποστολή εντολών (π.χ. αποσύνδεση/επανασύνδεση) και ανοίγουν το δρόμο για εφαρμογές δυναμικής τιμολόγησης. Η συνεχής ροή δεδομένων βοηθάει στην υλοποίηση μηχανισμών απόκρισης ζήτησης, στην ανίχνευση απωλειών ή μη τεχνικών παρεμβάσεων και στη βελτιστοποίηση της ενεργειακής απόδοσης. Παρ' όλα αυτά, η ανάπτυξη AMI σε μεγάλη κλίμακα εισάγει σημαντικές προκλήσεις, όπως η επιλογή κατάλληλων επικοινωνιακών τεχνολογιών (PLC, RF mesh, cellular), η διαχείριση εκατομμυρίων συσκευών και η διασφάλιση αξιοπιστίας σε περιβάλλοντα με υψηλό φόρτο δεδομένων (Depuru et al., 2011).

Επιπλέον, η συγκέντρωση δεδομένων κατανάλωσης εγείρει ζητήματα ασφάλειας και ιδιωτικότητας. Τα προφίλ φορτίου μπορούν να αποκαλύψουν πληροφορίες σχετικά με τη λειτουργία μιας εγκατάστασης ή τις συνήθειες των χρηστών. Η προστασία των δεδομένων απαιτεί κρυπτογράφηση, αυθεντικοποίηση και έλεγχο πρόσβασης σε όλα τα επίπεδα επικοινωνίας, καθώς και κατάλληλες πολιτικές διαχείρισης δεδομένων (Depuru et al., 2011).

Σε επιχειρησιακά περιβάλλοντα, το IoT επεκτείνεται σε όλο το ενεργειακό σύστημα της εγκατάστασης. Πέρα από τους smart meters, εγκαθίστανται αισθητήρες θερμοκρασίας και υγρασίας, μετρητές ποιότητας ισχύος, αισθητήρες κατάστασης κινητήρων και εξοπλισμού, καθώς και συστήματα παρακολούθησης inverters και μονάδων αποθήκευσης (Battery Management Systems – BMS). Παράλληλα, τα δεδομένα από υπάρχοντα συστήματα PLC και SCADA ενσωματώνονται σε ενιαίες πλατφόρμες, δημιουργώντας μια ολοκληρωμένη εικόνα της λειτουργίας του ενεργειακού συστήματος.

Ο συνδιασμός αυτών των διαφορετικών πηγών δεδομένων επιτρέπει τον υπολογισμό δεικτών απόδοσης (Key Performance Indicators – KPIs) υψηλής ανάλυσης, όπως κατανάλωση ανά γραμμή παραγωγής, ενεργειακή ένταση ανά τετραγωνικό μέτρο ή ποσοστό αυτοκατανάλωσης από ΑΠΕ. Ωστόσο, όσο περισσότερα και πιο διαφορετικά τα δεδομένα τόσο μεγαλώνει η ανάγκη για τυποποίηση και σωστή σημασιολογική μοντελοποίηση, για να επιτευχθεί διαλειτουργικότητα και συνεπής ανάλυση σε όλο το σύστημα (Al-Fuqaha et al., 2015).

Επομένως, η IoT υποδομή στην ενεργειακή παρακολούθηση δεν είναι απλά ένας τρόπος συλλογής δεδομένων. Είναι το θεμέλιο για τη δημιουργία ενός ολοκληρωμένου συστήματος ανάλυσης, πρόβλεψης και βελτιστοποίησης. Η ποιότητα και ο σχεδιασμός του επιπέδου μέτρησης καθορίζουν σε μεγάλο βαθμό το πόσο αποτελεσματικά θα λειτουργούν τα επόμενα επίπεδα της έξυπνης ενεργειακής διαχείρισης.

2.4.3 Πρωτόκολλα επικοινωνίας και στοίβα πρωτοκόλλων (Communication Protocol Stack)

Για να λειτουργήσει σωστά η ενεργειακή παρακολούθηση σε περιβάλλοντα Smart Grid είναι βασική η αξιόπιστη και αποδοτική μεταφορά δεδομένων μεταξύ συσκευών με περιορισμένους υπολογιστικούς και ενεργειακούς πόρους. Οι αισθητήρες, οι έξυπνοι μετρητές και οι ενσωματωμένοι ελεγκτές συνήθως λειτουργούν σε περιβάλλοντα με χαμηλή ισχύ και περιορισμένο εύρος ζώνης. Αυτό σημαίνει πως τα παραδοσιακά πρωτόκολλα Internet χρειάζονται προσαρμογές για να δουλέψουν καλύτερα σε τέτοιες συνθήκες. Η IoT βιβλιογραφία περιγράφει μια πολυεπίπεδη στοίβα πρωτοκόλλων, η οποία επιτρέπει τη διαλειτουργικότητα, την επεκτασιμότητα και τη βελτιστοποιημένη μεταφορά δεδομένων σε διαφορετικά ενεργειακά περιβάλλοντα (Al-Fuqaha et al., 2015).

Σε επίπεδο δικτύου, το πρωτόκολλο 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks) επιτρέπει τη χρήση IPv6 σε ασύρματα δίκτυα χαμηλής ισχύος, καθιστώντας δυνατή την ενσωμάτωση περιορισμένων IoT συσκευών στο ευρύτερο Internet. Το 6LoWPAN μειώνει το μέγεθος των επικεφαλίδων και προσαρμόζει το IPv6 σε μικρότερα πακέτα, εξασφαλίζοντας πιο αποδοτική μετάδοση δεδομένων (Zanella et al., 2014). Αυτό είναι πολύ σημαντικό σε εφαρμογές smart metering και δίκτυα καταναμημένων αισθητήρων, όπου κάθε συσκευή προσπαθεί να κρατήσει χαμηλά την κατανάλωση ενέργειας.

Σε επίπεδο εφαρμογής, δύο από τα πιο διαδεδομένα πρωτόκολλα στην ενεργειακή τηλεμετρία είναι το MQTT (Message Queuing Telemetry Transport) και το CoAP (Constrained Application Protocol). Το MQTT βασίζεται σε μοντέλο publish/subscribe με κεντρικό διαμεσολαβητή (broker-based architecture), επιτρέποντας την αποδοτική διανομή δεδομένων σε πολλαπλούς συνδρομητές. Η αρχιτεκτονική αυτή κάνει το MQTT κατάλληλο

για συνεχή ροή τηλεμετρίας, όπως η μετάδοση δεδομένων κατανάλωσης ή παραγωγής σε πραγματικό χρόνο (Al-Fuqaha et al., 2015).

Αντίθετα, το CoAP ακολουθεί μοντέλο request/response παρόμοιο με το HTTP, αλλά είναι σχεδιασμένο για constrained περιβάλλοντα. Υποστηρίζει RESTful αρχιτεκτονική, επιτρέποντας την άμεση πρόσβαση σε πόρους συσκευών (resources) μέσω ελαφρών UDP μηνυμάτων. Το CoAP ταιριάζει καλύτερα σε εφαρμογές που χρειάζονται περιοδικά αιτήματα ή απλή αλληλεπίδραση ανάμεσα σε συσκευές και κεντρικά συστήματα, ειδικά σε δίκτυα χαμηλής ισχύος ή υψηλής απώλειας πακέτων (Zanella et al., 2014).

Η επιλογή πρωτοκόλλου έχει άμεσο αντίκτυπο σε σημαντικές παραμέτρους απόδοσης του συστήματος, όπως η καθυστέρηση μετάδοσης, η επιβάρυνση επικεφαλίδων, η κατανάλωση ενέργειας και η ανθεκτικότητα σε απώλειες πακέτων. Συγκριτικές μελέτες απόδοσης δείχνουν ότι το MQTT υπερτερεί σε σενάρια υψηλής συχνότητας δημοσίευσης δεδομένων, ενώ το CoAP εμφανίζει πλεονεκτήματα σε άμεσες αλληλεπιδράσεις τύπου client-server. Παρόλα αυτά, όταν εισάγονται μηχανισμοί ασφάλειας, όπως TLS ή DTLS, παρατηρούνται διαφοροποιήσεις στην καθυστέρηση και στην κατανάλωση πόρων, οι οποίες πρέπει να λαμβάνονται υπόψη στον σχεδιασμό της αρχιτεκτονικής (Seoane et al., 2021).

Τελικά, η επιλογή και διαμόρφωση της στοίβας επικοινωνίας σε ένα IoT-enabled Smart Grid δεν αποτελεί μια απλή τεχνική απόφαση, αλλά στρατηγικό σχεδιαστικό ζήτημα που επηρεάζει την απόδοση, την επεκτασιμότητα και την ασφάλεια του συστήματος. Η σωστή αντιστοίχιση πρωτοκόλλων στις απαιτήσεις της κάθε εφαρμογής — είτε πρόκειται για real-time ενεργειακή παρακολούθηση είτε για περιοδική συλλογή δεδομένων — είναι κρίσιμη να εξασφαλίσουμε ότι το σύστημα θα δουλεύει αξιόπιστα και αποδοτικά.

2.4.4 Κυβερνοασφάλεια και ιδιωτικότητα δεδομένων παρακολούθησης

Η ψηφιοποίηση των ενεργειακών δικτύων και η ενσωμάτωση IoT τεχνολογιών στα Smart Grids αυξάνουν σημαντικά την πιθανότητα επίθεσης εναντίον των συστημάτων αυτών. Η μετάβαση από απομονωμένα, κλειστά ενεργειακά δίκτυα σε διασυνδεδεμένες, IP-based αρχιτεκτονικές δημιουργεί νέες κατηγορίες κινδύνων, οι οποίες αφορούν τόσο την ασφάλεια της πληροφορίας όσο και τη λειτουργική ακεραιότητα του δικτύου. Ενδεικτικά,

πιθανές απειλές περιλαμβάνουν πλαστογράφηση ή αλλοίωση μετρήσεων, μη εξουσιοδοτημένη πρόσβαση σε συστήματα ελέγχου, επιθέσεις άρνησης υπηρεσίας (Denial-of-Service – DoS) σε υποδομές AMI και στοχευμένες παρεμβάσεις σε κρίσιμες λειτουργίες αυτοματισμού (Yan et al., 2012).

Η ιδιαιτερότητα των smart grids είναι ότι τα πληροφοριακά και τα ενεργειακά συστήματα είναι στενά διασυνδεδεμένα. Μία επιτυχημένη επίθεση δεν περιορίζεται στην απώλεια δεδομένων, αλλά μπορεί να οδηγήσει σε διακοπές παροχής, καταστροφή εξοπλισμού ή αποσταθεροποίηση του δικτύου. Ως εκ τούτου, η κυβερνοασφάλεια στα smart grids δεν αποτελεί απλώς ζήτημα προστασίας δεδομένων, αλλά κρίσιμο παράγοντα διασφάλισης της επιχειρησιακής συνέχειας και της ενεργειακής ασφάλειας (Gungor et al., 2011).

Οι επιστημονικές ανασκοπήσεις στον τομέα της κυβερνοασφάλειας smart grid επικοινωνιών επισημαίνουν την ανάγκη υιοθέτησης πολυεπίπεδων μηχανισμών προστασίας. Οι βασικές τεχνικές απαιτήσεις περιλαμβάνουν ισχυρούς μηχανισμούς αυθεντικοποίησης, κρυπτογράφηση δεδομένων σε μεταφορά και αποθήκευση, έλεγχο πρόσβασης βάσει ρόλων, ασφαλή διαχείριση κλειδιών και συστήματα ανίχνευσης και απόκρισης σε εισβολές (Intrusion Detection/Prevention Systems – IDS/IPS) (Yan et al., 2012). Ιδιαίτερη πρόκληση αποτελεί η εφαρμογή των παραπάνω σε συσκευές περιορισμένων πόρων, όπως έξυπνοι μετρητές και ασύρματοι αισθητήρες, όπου οι περιορισμοί σε υπολογιστική ισχύ και ενέργεια καθιστούν δύσκολη την υλοποίηση σύνθετων κρυπτογραφικών μηχανισμών.

Στο επίπεδο της Προηγμένης Υποδομής Μέτρησης (AMI), φαίνονται συγκεκριμένες ευπάθειες που σχετίζονται με την αμφίδρομη επικοινωνία και τη μαζική διαχείριση συσκευών. Οι έξυπνοι μετρητές, ως τερματικά σημεία του δικτύου, μπορούν να αποτελέσουν στόχο φυσικής ή ψηφιακής παραβίασης, ενώ η συγκέντρωση δεδομένων σε κεντρικούς servers δημιουργεί πιθανά σημεία συγκέντρωσης κινδύνου (Depuru et al., 2011). Παράλληλα, οι ευρύτερες επικοινωνιακές υποδομές του smart grid — συμπεριλαμβανομένων των δικτύων HAN, NAN και WAN — πρέπει να προστατεύονται από παρεμβολές, υποκλοπές και μη εξουσιοδοτημένη πρόσβαση (Gungor et al., 2011).

Εξίσου κρίσιμο ζήτημα αποτελεί η ιδιωτικότητα των δεδομένων ενεργειακής παρακολούθησης. Τα δεδομένα κατανάλωσης υψηλής ανάλυσης μπορούν να δείξουν λεπτομερή πρότυπα λειτουργίας μιας εγκατάστασης ή ενός οργανισμού, όπως ώρες λειτουργίας εξοπλισμού, επίπεδα παραγωγικής δραστηριότητας ή ακόμη και ευαίσθητες επιχειρησιακές πληροφορίες. Για τον λόγο αυτό, η ιδιωτικότητα δεν πρέπει να αντιμετωπίζεται ως δευτερεύον χαρακτηριστικό, αλλά βασική απαίτηση σχεδιασμού. Τεχνικές όπως η ανωνυμοποίηση, ψευδωνυμοποίηση και ελαχιστοποίηση δεδομένων (data minimization) συμβάλλουν στη μείωση του κινδύνου διαρροής ευαίσθητων πληροφοριών (Depuru et al., 2011).

Τέλος, η κυβερνοασφάλεια και η προστασία της ιδιωτικότητας αποτελούν απαραίτητα στοιχεία της αρχιτεκτονικής των IoT-enabled Smart Grids. Η διασφάλιση εμπιστευτικότητας, ακεραιότητας και διαθεσιμότητας (CIA triad) των ενεργειακών δεδομένων και των μηχανισμών ελέγχου αποτελεί βασική προϋπόθεση για την αξιόπιστη λειτουργία των σύγχρονων συστημάτων ενεργειακής παρακολούθησης. Ως εκ τούτου, ο σχεδιασμός ενός Intelligent Energy Management System οφείλει να ενσωματώνει μηχανισμούς ασφάλειας σε όλα τα επίπεδα της επικοινωνιακής στοίβας και της αρχιτεκτονικής edge–cloud.

2.4.5 Σύνδεση με real-time dashboards και Intelligent Energy Management Systems (IEMS)

Η ανάλυση που προηγήθηκε σχετικά με την πολυεπίπεδη IoT αρχιτεκτονική, τα πρωτόκολλα επικοινωνίας, τις edge–cloud υποδομές και τις απαιτήσεις κυβερνοασφάλειας, αποδεικνύει ότι ένα σύγχρονο σύστημα ενεργειακής παρακολούθησης δεν περιορίζεται σε απλή οπτικοποίηση δεδομένων. Το real-time dashboard σε ένα Smart Grid αποτελεί το ανώτερο επίπεδο μιας σύνθετης τεχνολογικής στοίβας, η οποία ξεκινά από τη συλλογή δεδομένων σε επίπεδο αισθητήρων και καταλήγει στην εφαρμογή προηγμένων αναλυτικών και προγνωστικών μοντέλων.

Στο πλαίσιο των Smart Grids υπάρχει συνεχής ροή δεδομένων από έξυπνους μετρητές, αισθητήρες και συστήματα ελέγχου. Με τις edge–cloud αρχιτεκτονικές, μπορούμε να επεξεργαζόμαστε αυτά τα δεδομένα σε πραγματικό χρόνο. Αυτό δίνει τη δυνατότητα για

άμεση αποστολή ειδοποιήσεων σε περιπτώσεις υπέρβασης ορίων, ασυνήθιστης κατανάλωσης ή μεταβολών στην ποιότητα ισχύος. Αυτό βοηθάει τους διαχειριστές να καταλαβαίνουν τι γίνεται σε πραγματικό χρόνο και να αντιδρούν πολύ πιο γρήγορα στα κρίσιμα περιστατικά (Fang et al., 2012).

Ταυτόχρονα, η συστηματική συλλογή και ενοποίηση δεδομένων επιτρέπει τον ακριβή υπολογισμό Δεικτών Απόδοσης (Key Performance Indicators – KPIs), όπως η ενεργειακή ένταση (kWh/m^2), ο συντελεστής φορτίου, το ποσοστό αυτοκατανάλωσης από ΑΠΕ και οι εκπομπές CO_2 . Η ακρίβεια και η συνέπεια των δεδομένων, που εξασφαλίζει το IoT και οι μηχανισμοί συγχρονισμού, μας δίνουν τη δυνατότητα να παρακολουθούμε την απόδοση σε επίπεδο εγκατάστασης ή μικροδικτύου. Έτσι, το dashboard δεν είναι απλώς μία παρουσίαση αριθμητικών τιμών αλλά ένα εργαλείο στρατηγικής διαχείρισης (Al-Fuqaha et al., 2015).

Επιπλέον, τα real-time δεδομένα και η κατανομημένη επεξεργασία στο edge βοηθούν στην ανίχνευση ανωμαλιών και την προγνωστική συντήρηση. Αλγόριθμοι που εκτελούνται είτε τοπικά είτε σε cloud περιβάλλον μπορούν να εντοπίζουν αποκλίσεις από τα κανονικά πρότυπα λειτουργίας και να ειδοποιούν για πιθανές βλάβες ή μη αποδοτικές λειτουργικές συνθήκες. Το dashboard λειτουργεί ως διεπαφή για την παρουσίαση των αποτελεσμάτων αυτών των αλγορίθμων και βοηθάει στο χρήστη στη λήψη αποφάσεων.

Σε πιο προχωρημένες εφαρμογές, τα real-time dashboards ενσωματώνονται σε Intelligent Energy Management Systems (IEMS), τα οποία αξιοποιούν τα δεδομένα για πρόβλεψη φορτίου, πρόβλεψη παραγωγής από ΑΠΕ και βελτιστοποίηση ενεργειακής λειτουργίας. Για να δουλέψουν σωστά τα μοντέλα μηχανικής μάθησης, χρειάζονται συνεχή ροή δεδομένων. Ο συνδυασμός IoT, edge computing και analytics μας επιτρέπει τη μετάβαση από παθητική παρακολούθηση σε ενεργή και έξυπνη διαχείριση της ενέργειας (Fang et al., 2012; Minh et al., 2022).

Επομένως, το intelligent dashboard σε ένα IoT-enabled Smart Grid αποτελεί το τελικό επίπεδο αλληλεπίδρασης ενός ολοκληρωμένου οικοσυστήματος συλλογής, μετάδοσης, επεξεργασίας και ανάλυσης ενεργειακών δεδομένων. Η ύπαρξη πολυεπίπεδης στοίβας πρωτοκόλλων επικοινωνίας, edge-cloud αρχιτεκτονικής και μηχανισμών ασφάλειας κάνει δυνατή την υλοποίηση των βασικών λειτουργιών ενός IEMS, δηλαδή την έγκαιρη

ειδοποίηση, ακριβή παρακολούθηση απόδοσης, ανίχνευση ανωμαλιών και υποστήριξη βελτιστοποιημένης λήψης αποφάσεων (Al-Fuqaha et al., 2015; Minh et al., 2022).

Συνεπώς, η ανάπτυξη ενός real-time ενεργειακού dashboard, όπως εξετάζεται στην παρούσα διπλωματική εργασία, στηρίζεται στις αρχές των Smart Grids και της IoT αρχιτεκτονικής και αποτελεί μία πρακτική εφαρμογή των βασικών εννοιών των Intelligent Energy Management Systems.

2.5 Dashboards και συστήματα οπτικοποίησης δεδομένων

Η αυξανόμενη διαθεσιμότητα ενεργειακών δεδομένων υψηλής ανάλυσης, που οφείλεται στην ενσωμάτωση IoT τεχνολογιών και Smart Grids απαιτεί την ανάπτυξη έξυπνων συστημάτων οπτικοποίησης. Τα dashboards αποτελούν βασικό εργαλείο για την απεικόνιση και την ερμηνεία των δεδομένων. Δίνουν στους διαχειριστές ενεργειακών συστημάτων τη δυνατότητα να αποκτούν άμεση εικόνα της λειτουργικής κατάστασης, να εντοπίζουν αποκλίσεις και να παίρνουν σωστές αποφάσεις.

2.5.1 Θεωρητικές βάσεις οπτικοποίησης δεδομένων

Η οπτικοποίηση δεδομένων αποτελεί ένα επιστημονικό πεδίο που συνδυάζει αρχές πληροφορικής, ψυχολογίας και σχεδιασμού διεπαφών. Η οπτικοποίηση βασίζεται στον τρόπο που βλέπουμε και επεξεργαζόμαστε οπτικές πληροφορίες και επιτρέπει την αναγνώριση προτύπων, τάσεων και ανωμαλιών (Ware, 2013). Η επιλογή κατάλληλων γραφημάτων (π.χ. χρονικές σειρές, heatmaps, scatter plots) επηρεάζει άμεσα την κατανόηση και ερμηνεία των δεδομένων.

Η οπτική αναπαράσταση πολύπλοκων δεδομένων πρέπει να ακολουθεί αρχές όπως η ελαχιστοποίηση γνωσιακού φορτίου, η σαφήνεια, η συνέπεια και η ιεραρχική δομή πληροφορίας (Few, 2013). Στα ενεργειακά συστήματα, όπου οι χρήστες καλούνται να διαχειριστούν μεγάλο όγκο χρονοσειρών, η σωστή σχεδίαση dashboard είναι πολύ σημαντική.

2.5.2 Dashboards σε περιβάλλοντα Smart Grid και Intelligent Energy Management Systems (IEMS)

Στα σύγχρονα Smart Grid, τα dashboards δεν περιορίζονται στην απλή απεικόνιση δεδομένων κατανάλωσης. Έχουν εξελιχθεί σε πλήρη συστήματα υποστήριξης αποφάσεων (Decision Support Systems – DSS). Η μετάβαση από παθητικά γραφήματα σε διαδραστικές διεπαφές ακολουθεί την εξέλιξη των ενεργειακών συστημάτων προς ψηφιοποιημένες και ευφυείς υποδομές. Η συνεχής ροή δεδομένων από αισθητήρες, έξυπνους μετρητές και συστήματα ελέγχου, σε συνδυασμό με τις IoT και edge–cloud αρχιτεκτονικές, επιτρέπει την οπτικοποίηση κρίσιμων μεταβλητών σε πραγματικό χρόνο, ενισχύοντας την επιχειρησιακή επίγνωση κατάστασης (situational awareness) (Fang et al., 2012).

Η real-time απεικόνιση δίνει τη δυνατότητα παρακολούθησης τόσο της συνολικής όσο και της επιμέρους κατανάλωσης ενέργειας σε επίπεδο εγκατάστασης, γραμμής παραγωγής ή ζώνης κτιρίου. Επιπλέον τα dashboards σε περιβάλλοντα Smart Grid παρουσιάζουν δεδομένα παραγωγής από Ανανεώσιμες Πηγές Ενέργειας, επιτρέποντας τη σύγκριση μεταξύ κατανάλωσης και τοπικής παραγωγής και την εκτίμηση του ποσοστού αυτοκατανάλωσης. Στην περίπτωση ύπαρξης συστημάτων αποθήκευσης προβάλλεται και η κατάσταση φόρτισης (State of Charge – SoC), η ροή ισχύος προς και από τις μπαταρίες, καθώς και η συμμετοχή τους στη διαχείριση αιχμών. Επιπλέον, η παρακολούθηση δεικτών όπως διακυμάνσεις τάσης και αρμονικές παραμορφώσεις δίνει τη δυνατότητα εντοπισμού προβλημάτων στη λειτουργία, το οποίο είναι σημαντικό σε βιομηχανικά περιβάλλοντα με ευαίσθητο εξοπλισμό.

Η ενσωμάτωση όλων αυτών των ετερογενών δεδομένων σε ένα ενιαίο dashboard το μετατρέπει σε βασικό εργαλείο διαχείρισης της ενεργειακής απόδοσης. Το dashboard μπορεί να δείχνει μαζί τεχνικούς και περιβαλλοντικούς δείκτες όπως η ενεργειακή ένταση και οι εκπομπές διοξειδίου του άνθρακα. Ακόμα, ενισχύει τη διαφάνεια και διευκολύνει τη συμμόρφωση με κανονιστικά και ESG πλαίσια (Zhang et al., 2018). Η δυνατότητα σύνδεσης των ενεργειακών με οικονομικούς δείκτες επιτρέπει τη σύνδεση της ενεργειακής στρατηγικής με τη χρηματοοικονομική απόδοση, κάτι το οποίο κάνει το dashboard εργαλείο στρατηγικού σχεδιασμού.

Τα σύγχρονα dashboards ενσωματώνουν επίσης embedded analytics, τα οποία αξιοποιούν τεχνικές μηχανικής μάθησης και ανάλυσης χρονοσειρών. Η προβολή προβλέψεων φορτίου ή παραγωγής από ΑΠΕ, επιτρέπει τη μετάβαση από περιγραφική σε προγνωστική και προληπτική διαχείριση. Με αυτό τον τρόπο, τα dashboards γίνονται η διεπαφή ανάμεσα στα αναλυτικά μοντέλα και τον άνθρωπο, μετατρέποντας τα αποτελέσματα σε γνώση που μπορεί πραγματικά να αξιοποιηθεί στην πράξη (Zhang et al., 2018).

Στο πλαίσιο των Intelligent Energy Management Systems, το dashboard αποτελεί το ανώτερο επίπεδο διεπαφής που συνδυάζει IoT, επικοινωνιακές υποδομές και προηγμένη ανάλυση δεδομένων. Η αξία της ψηφιοποίησης αναδεικνύεται μόνο όταν τα δεδομένα μετατρέπονται σε γνώση και η γνώση σε δράση (Fang et al., 2012).

2.5.3 Real-time οπτικοποίηση και ροές δεδομένων

Για real-time οπτικοποίηση σε περιβάλλοντα Smart Grid και Intelligent Energy Management Systems χρειάζονται αρχιτεκτονικές που να διαχειρίζονται ροές δεδομένων χαμηλής καθυστέρησης (low-latency data streaming) και υψηλής συχνότητας δειγματοληψίας. Σε αντίθεση με τα παραδοσιακά συστήματα (batch processing), όπου τα δεδομένα συλλέγονται και αναλύονται εκ των υστέρων, τα σύγχρονα ενεργειακά συστήματα απαιτούν συνεχή ροή, επεξεργασία και απεικόνιση πληροφοριών σε σχεδόν πραγματικό χρόνο. Η απαίτηση αυτή προκύπτει από τις συνεχείς μεταβολές της κατανάλωσης και της παραγωγής ενέργειας από τις ΑΠΕ (Fang et al., 2012).

Η αρχιτεκτονική για real-time dashboards βασίζεται συνήθως σε μηχανισμούς διαμεσολάβησης μηνυμάτων (message brokers) και σε βάσεις δεδομένων χρονοσειρών (time-series databases). Οι message brokers επιτρέπουν την ασύγχρονη, κλιμακώσιμη και αξιόπιστη μεταφορά δεδομένων από τις συσκευές πεδίου προς τις εφαρμογές ανάλυσης και οπτικοποίησης, υλοποιώντας μοντέλα publish/subscribe ή event-driven επικοινωνίας. Η προσέγγιση αυτή μειώνει τη σύζευξη μεταξύ παραγωγών και καταναλωτών δεδομένων και υποστηρίζει την επεκτασιμότητα του συστήματος σε περιβάλλοντα με μεγάλο αριθμό IoT συσκευών (Al-Fuqaha et al., 2015).

Οι βάσεις δεδομένων χρονοσειρών είναι σχεδιασμένες για την αποδοτική αποθήκευση και ανάκτηση δεδομένων με χρονική σήμανση (timestamped data), χαρακτηριστικό που αποτελεί βασική ιδιότητα των ενεργειακών μετρήσεων. Για τη διαχείριση τέτοιων δεδομένων απαιτούνται διαδικασίες συμπίεσης, αναζήτησης και συγχρονισμού, ώστε να γίνεται γρήγορη ενημέρωση των γραφημάτων και υπολογισμός των δεικτών απόδοσης χωρίς να υπάρχει μείωση της απόδοσης του συστήματος. Η υψηλή συχνότητα δειγματοληψίας κάνει αναγκαία τη βέλτιστη διαχείριση ροών και την αποφυγή συμφόρησης (Zhang et al., 2018).

Η δυνατότητα real-time ενημέρωσης των dashboards προσφέρει σημαντικά πλεονεκτήματα. Επιτρέπει την άμεση ανίχνευση υπερβάσεων ορίων κατανάλωσης ή αποκλίσεων από προκαθορισμένες τιμές και με αυτόν τον τρόπο διευκολύνει την έγκαιρη ενεργοποίηση ειδοποιήσεων. Επιπλέον, είναι δυνατή η παρακολούθηση αιχμών φορτίου και η δυναμική αξιολόγηση της επίδρασής τους στο σύστημα. Αυτό είναι πολύ σημαντικό για τη διαχείριση της ζήτησης και την αποφυγή υπερφορτώσεων.

2.5.4 Δείκτες Απόδοσης (KPIs) και Οπτικοποίηση

Η επιλογή κατάλληλων Δεικτών Απόδοσης (Key Performance Indicators – KPIs) αποτελεί θεμελιώδη παράγοντα στη σχεδίαση και λειτουργία ενός ενεργειακού dashboard. Οι δείκτες αυτοί λειτουργούν ως ποσοτικοποίηση της ενεργειακής συμπεριφοράς ενός συστήματος και οφείλουν να είναι μετρήσιμοι, συγκρίσιμοι σύμφωνα με τους στρατηγικούς στόχους του οργανισμού, όπως η βελτίωση της ενεργειακής αποδοτικότητας, η μείωση του λειτουργικού κόστους και ο περιορισμός των εκπομπών αερίων του θερμοκηπίου. Αποτελούν εργαλεία αξιολόγησης της απόδοσης και παίζουν σημαντικό ρόλο στη λήψη αποφάσεων (Fang et al., 2012).

Στον τομέα της ενέργειας, ένας από τους πιο γνωστούς δείκτες είναι ο Energy Use Intensity (EUI), ο οποίος εκφράζει την κατανάλωση ενέργειας ανά τετραγωνικό μέτρο και επιτρέπει τη συγκριτική αξιολόγηση κτιρίων ή εγκαταστάσεων διαφορετικού μεγέθους. Επιπλέον, ο δείκτης διείσδυσης ανανεώσιμων πηγών (renewable penetration rate) δείχνει το ποσοστό συμμετοχής της παραγόμενης ενέργειας από ΑΠΕ στο συνολικό ενεργειακό ισοζύγιο. Ο λόγος αυτοκατανάλωσης (self-consumption ratio) αξιολογεί το ποσοστό της τοπικά

παραγόμενης ενέργειας που καταναλώνεται επιτόπου. Επιπλέον, δείκτες όπως ο peak demand index αποτυπώνουν τη μέγιστη ζήτηση ισχύος σε συγκεκριμένη χρονική περίοδο, παρέχοντας πληροφορίες για τη διαχείριση αιχμών και τη βελτιστοποίηση συμβολαίων ισχύος. Αντίστοιχα, το carbon intensity, δηλαδή οι εκπομπές CO₂ ανά μονάδα καταναλισκόμενης ή παραγόμενης ενέργειας, συνδέει την ενεργειακή λειτουργία με περιβαλλοντικούς στόχους. Η συνδυαστική αξιολόγηση τεχνικών και περιβαλλοντικών δεικτών παρέχει τη δυνατότητα ολοκληρωμένης αξιολόγησης της ενεργειακής απόδοσης (Zhang et al., 2018).

Η αποτελεσματικότητα των KPIs εξαρτάται σε μεγάλο βαθμό από τον τρόπο οπτικοποίησής τους. Η παρουσίασή τους μέσω γραφημάτων, δυναμικών φίλτρων και πολυεπίπεδων απεικονίσεων επιτρέπει την ανάλυση σε διαφορετικές χρονικές κλίμακες και επίπεδα λειτουργίας. Η drill-down ανάλυση από συνολικούς δείκτες σε επιμέρους υποσυστήματα ή χρονικές περιόδους βοηθάει στην κατανόηση των μοτίβων κατανάλωσης και διευκολύνει τον εντοπισμό αποκλίσεων. Σύμφωνα με τις αρχές σχεδίασης οπτικοποίησης, η οργάνωση της πληροφορίας και η ελαχιστοποίηση γνωσιακού φορτίου βελτιώνουν την ικανότητα ερμηνείας των δεδομένων και τη λήψη τεκμηριωμένων αποφάσεων (Few, 2013).

2.5.5 Ανθρώπινος παράγοντας και εμπειρία χρήστη (UX)

Η αποτελεσματικότητα ενός dashboard εξαρτάται όχι μόνο από την τεχνολογική του υποδομή, αλλά και από τη σχεδίαση της διεπαφής (User Interface – UI) και την εμπειρία χρήστη (User Experience – UX). Μελέτες δείχνουν ότι η υπερβολική πληροφόρηση ή η κακή οργάνωση οπτικών στοιχείων μπορεί να μειώσει την ικανότητα λήψης αποφάσεων (Ware, 2013). Στα ενεργειακά συστήματα, όπου οι χρήστες συχνά δεν είναι ειδικοί στην ανάλυση δεδομένων, η σαφήνεια και η απλότητα είναι αναγκαίες στον σχεδιασμό.

Η ενσωμάτωση χρωματικής κωδικοποίησης (color coding), δυναμικών ειδοποιήσεων και προσαρμόσιμων φίλτρων ενισχύει τη λειτουργικότητα του dashboard και επιτρέπει την προσαρμογή του σε διαφορετικά επίπεδα χρηστών (διαχειριστές, τεχνικό προσωπικό, διοίκηση).

2.6 Τεχνικές πρόβλεψης ενεργειακής παραγωγής και κατανάλωσης

Η πρόβλεψη της παραγωγής και κατανάλωσης ενέργειας αποτελεί βασική λειτουργία των σύγχρονων Intelligent Energy Management Systems (IEMS), καθώς επιτρέπει τον προληπτικό σχεδιασμό και τη μείωση του λειτουργικού κόστους. Σε περιβάλλοντα Smart Grid, όπου η παραγωγή εξαρτάται από Ανανεώσιμες Πηγές Ενέργειας (ΑΠΕ) η ακρίβεια των προβλέψεων επηρεάζει τη σταθερότητα και την αποδοτικότητα του συστήματος (Hong & Fan, 2016).

Η πρόβλεψη ενεργειακών μεγεθών βασίζεται κυρίως στην ανάλυση χρονοσειρών (time series analysis), όπου οι μελλοντικές τιμές εκτιμώνται με βάση ιστορικά δεδομένα και εξωγενείς μεταβλητές, όπως καιρικές συνθήκες, ημερολογιακά χαρακτηριστικά και δείκτες λειτουργικής δραστηριότητας. Ανάλογα με τον χρονικό ορίζοντα, οι προβλέψεις διακρίνονται σε βραχυπρόθεσμες, μεσοπρόθεσμες και μακροπρόθεσμες, με τις βραχυπρόθεσμες να είναι σημαντικότερες για τη λειτουργική διαχείριση και τον έλεγχο των αιχμών (Hong & Fan, 2016).

2.6.1 Στατιστικές μέθοδοι πρόβλεψης

Οι στατιστικές μέθοδοι, όπως τα Αυτοπαλίνδρομα Ολοκληρωμένα Κινητού Μέσου Μοντέλα (ARIMA – AutoRegressive Integrated Moving Average), αποτελούν ευρέως χρησιμοποιούμενη προσέγγιση στην πρόβλεψη φορτίου. Τα μοντέλα ARIMA στηρίζονται στη γραμμική συσχέτιση μεταξύ παρελθοντικών και μελλοντικών τιμών της χρονικής σειράς και είναι ιδιαίτερα αποτελεσματικά σε δεδομένα με σαφή εποχικότητα ή τάση. Επεκτάσεις όπως τα SARIMA (Seasonal ARIMA) επιτρέπουν τη μοντελοποίηση εποχικών μοτίβων, χαρακτηριστικό συχνό στην ενεργειακή κατανάλωση (Weron, 2006).

2.6.2 Μέθοδοι μηχανικής μάθησης (Machine Learning)

Η ανάγκη για μοντελοποίηση σύνθετων, μη γραμμικών σχέσεων οδήγησε στην υιοθέτηση τεχνικών μηχανικής μάθησης. Αλγόριθμοι όπως οι Support Vector Machine (SVM), Random Forest και Artificial Neural Networks (ANN) έχουν εφαρμοστεί για την πρόβλεψη ενεργειακής κατανάλωσης και παραγωγής. Αυτές οι τεχνικές δίνουν τη δυνατότητα να λαμβάνουμε υπόψη πολλούς εξωτερικούς παράγοντες όπως θερμοκρασία, υγρασία, μέρα

της εβδομάδας, ακόμα και οικονομικούς δείκτες, βελτιώνοντας την ακρίβεια πρόβλεψης σε σχέση με καθαρά γραμμικά μοντέλα (Ahmad & Chen, 2018).

Ο αλγόριθμος Random Forest, που βασίζεται σε ένα σύνολο δέντρων απόφασεων, διακρίνεται για την ανθεκτικότητα του σε υπερπροσαρμογή (overfitting) και καλή ερμηνευσιμότητα. Ο SVM είναι αποτελεσματικός σε δεδομένα μικρού ή μεσαίου μεγέθους με πολύπλοκα όρια απόφασης. Ωστόσο, η απόδοσή τους εξαρτάται από την σωστή επιλογή χαρακτηριστικών και τη ρύθμιση των παραμέτρων τους.

2.6.3 Βαθιά μάθηση (Deep Learning)

Τα τελευταία χρόνια, η βαθιά μάθηση (deep learning) έχει αναδειχθεί ως βασική προσέγγιση στην πρόβλεψη ενεργειακών χρονοσειρών. Ιδιαίτερα τα Αναδρομικά Νευρωνικά Δίκτυα (Recurrent Neural Networks – RNN) και οι παραλλαγές τους, όπως τα Long Short-Term Memory (LSTM) δίκτυα, ξεχωρίζουν στη μοντελοποίηση μακροχρόνιων εξαρτήσεων. Τα LSTM μπορούν να «θυμούνται» προηγούμενες καταστάσεις, κάτι που τα βοηθά να εντοπίζουν σύνθετα μοτίβα στην κατανάλωση και παραγωγή ενέργειας (Kong et al., 2017).

Στην πρόβλεψη παραγωγής από φωτοβολταϊκά και αιολικά συστήματα, τα LSTM και τα υβριδικά μοντέλα (π.χ. CNN-LSTM) παρουσιάζουν υψηλή ακρίβεια, ιδιαίτερα όταν συνδυάζονται με μετεωρολογικά δεδομένα. Η αυξημένη υπολογιστική απαίτηση των μοντέλων αυτών καθιστά αναγκαία τη χρήση cloud ή edge-cloud υποδομών για εκπαίδευση και εκτέλεση.

2.6.4 Αξιολόγηση μοντέλων πρόβλεψης

Η αξιολόγηση της απόδοσης των μοντέλων πρόβλεψης πραγματοποιείται μέσω μετρικών όπως το Mean Absolute Error (MAE), το Root Mean Square Error (RMSE) και το Mean Absolute Percentage Error (MAPE). Η επιλογή κατάλληλης μετρικής εξαρτάται από τη φύση της εφαρμογής και την ανοχή σφάλματος του συστήματος. Σε ενεργειακά περιβάλλοντα, η ακρίβεια βραχυπρόθεσμης πρόβλεψης είναι κρίσιμη για τη διαχείριση των αιχμών και τη βελτιστοποίηση της λειτουργίας, ενώ οι μακροπρόθεσμες προβλέψεις υποστηρίζουν τον στρατηγικό σχεδιασμό (Hong & Fan, 2016).

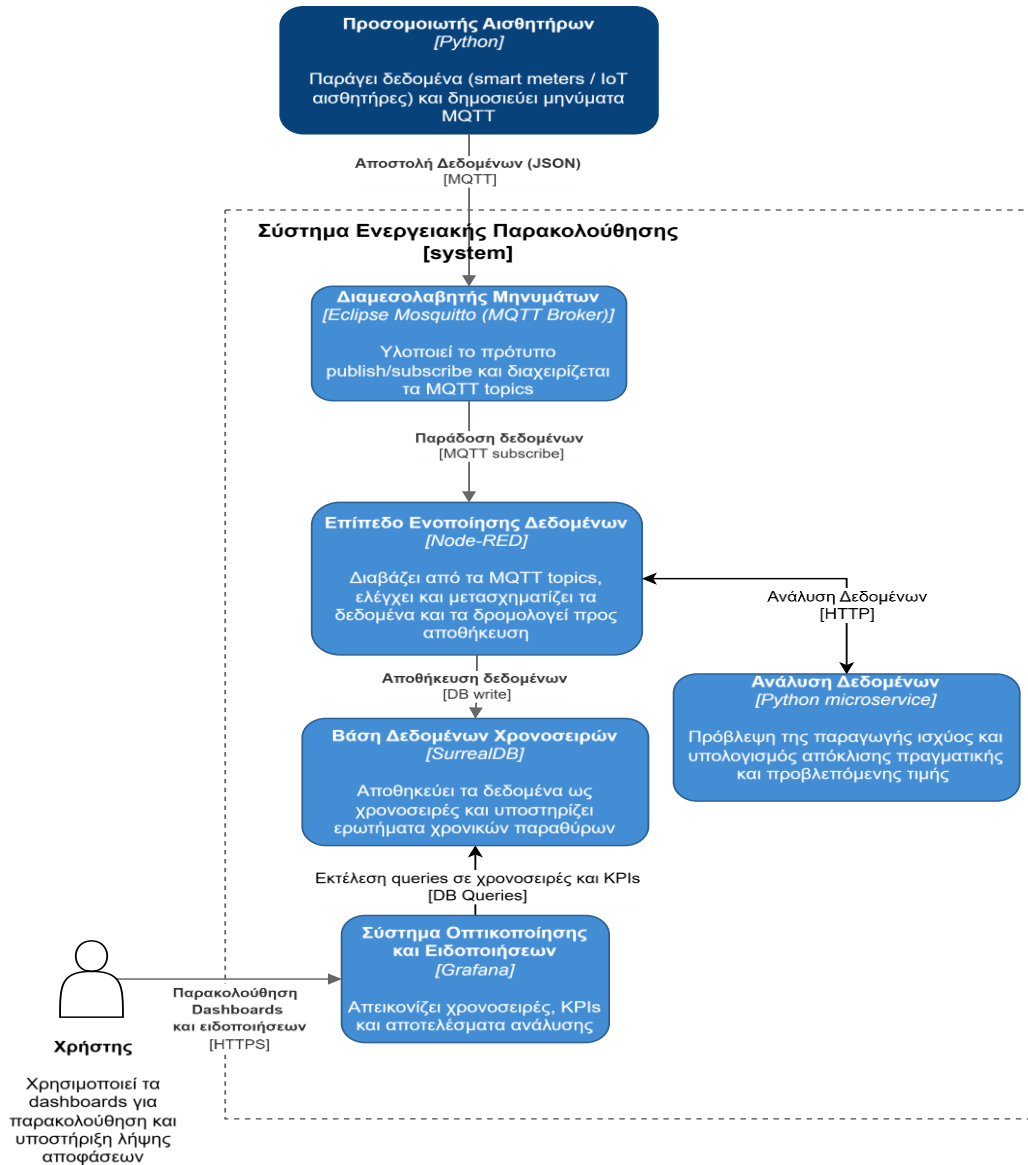
2.6.5 Σύνδεση μοντέλων πρόβλεψης με IEMS και dashboards

Η ενσωμάτωση προγνωστικών μοντέλων σε ένα IEMS επιτρέπει τη μετάβαση από αντιδραστική σε προγνωστική και προληπτική ενεργειακή διαχείριση. Τα αποτελέσματα των μοντέλων τροφοδοτούν τα dashboards, επιτρέποντας τη σύγκριση προβλεπόμενων και πραγματικών τιμών και τη δυναμική προσαρμογή στρατηγικών λειτουργίας. Η σύζευξη πρόβλεψης και οπτικοποίησης ενισχύει τη διαφάνεια, τη λήψη τεκμηριωμένων αποφάσεων και την αποδοτικότερη αξιοποίηση ενεργειακών πόρων. Συνεπώς, οι τεχνικές πρόβλεψης αποτελούν κρίσιμο θεωρητικό και τεχνολογικό κομμάτι των Intelligent Energy Management Systems, διασυνδέοντας την ανάλυση δεδομένων με τη στρατηγική ενεργειακή διαχείριση σε περιβάλλοντα Smart Grid.

3 Σχεδίαση και Αρχιτεκτονική Συστήματος

Στο κεφάλαιο αυτό περιγράφεται η σχεδίαση και η αρχιτεκτονική του συστήματος ενεργειακής παρακολούθησης. Το σύστημα που υλοποιείται είναι πλήρως containerized έτσι ώστε η διαχείριση να είναι εύκολη και ευέλικτη. Δεδομένου ότι δεν υπάρχει φυσική εγκατάσταση αισθητήρων, η παραγωγή δεδομένων πραγματοποιείται μέσω προσομοιωτή, ο οποίος αναπαριστά τη λειτουργία έξυπνων μετρητών (smart meters) και συσκευών IoT. Η αρχιτεκτονική βασίζεται στη σαφή διαστρωμάτωση λειτουργιών, στην επεκτασιμότητα και στην ευκολία αναπαραγωγής του περιβάλλοντος εκτέλεσης.

Ελένη Τσιανάκα, Ανάπτυξη Dashboard για Real-time Παρακολούθηση και Πρόβλεψη Απόδοσης Ενεργειακού Δικτύου Επιχείρησης που Χρησιμοποιεί Φωτοβολταϊκές και Αιολικές Λύσεις



Εικόνα 1 Αρχιτεκτονική Συστήματος

3.1 Ανάλυση απαιτήσεων συστήματος

Η σχεδίαση του Συστήματος Ενεργειακής Διαχείρισης (EMS) βασίζεται στον σαφή διαχωρισμό των λειτουργικών και μη λειτουργικών απαιτήσεων, με στόχο τη δημιουργία μιας επεκτάσιμης, αξιόπιστης και αρχιτεκτονικά ορθής πλατφόρμας IoT. Η ανάπτυξή του βασίζεται στην ανάγκη για real-time παρακολούθηση και στην ανάλυση ενεργειακών δεδομένων επιχειρήσεων που αξιοποιούν ανανεώσιμες πηγές ενέργειας.

3.1.1 Λειτουργικές απαιτήσεις

FR-01: Συλλογή Δεδομένων σε Πραγματικό Χρόνο

Το σύστημα πρέπει να υποστηρίζει τη συλλογή δεδομένων σε πραγματικό χρόνο από πολλαπλές ενεργειακές πηγές, όπως φωτοβολταϊκά συστήματα, ανεμογεννήτριες, έξυπνους μετρητές και αισθητήρες IoT. Τα δεδομένα αυτά περιλαμβάνουν μετρήσεις παραγωγής ενέργειας, κατανάλωσης, αποθήκευσης και περιβαλλοντικών παραμέτρων, όπως θερμοκρασία και ταχύτητα ανέμου.

FR-02: Φιλτράρισμα και Μετασχηματισμός

Ύπαρξη ενδιάμεσου λογισμικού (middleware) για την επεξεργασία, μορφοποίηση και δρομολόγηση των δεδομένων πριν την αποθήκευση.

FR-03: Αποθήκευση Χρονοσειρών

Ασφαλής και αποδοτική καταγραφή ιστορικών δεδομένων σε βάση δεδομένων κατάλληλη για αποθήκευση και επεξεργασία χρονοσειρών.

FR-04: Ανάλυση και Πρόβλεψη

Το σύστημα πρέπει να υποστηρίζει βασικές λειτουργίες ανάλυσης δεδομένων, όπως υπολογισμό δεικτών απόδοσης, ανίχνευση ανωμαλιών και πρόβλεψη ενεργειακής παραγωγής ή κατανάλωσης

FR-05: Οπτικοποίηση και Ειδοποιήσεις

Το σύστημα πρέπει να παρέχει δυνατότητα real-time παρακολούθησης της ενεργειακής απόδοσης μέσω dashboard, το οποίο θα εμφανίζει βασικούς δείκτες απόδοσης και

ενεργειακές ροές. Το σύστημα πρέπει επίσης να υποστηρίζει μηχανισμούς ειδοποιήσεων και αυτοματοποιημένων ενεργειών. Οι ειδοποιήσεις ενεργοποιούνται όταν παρατηρούνται αποκλίσεις από καθορισμένα όρια ή όταν εντοπίζονται ανωμαλίες στη λειτουργία του ενεργειακού συστήματος.

3.1.2 Μη λειτουργικές απαιτήσεις

NFR-01: Επεκτασιμότητα - Scalability

Η αρχιτεκτονική πρέπει να υποστηρίζει την οριζόντια προσθήκη νέων μετρητών και αισθητήρων χωρίς την ανάγκη ανασχεδιασμού του κώδικα.

NFR-02: Χαμηλή Υστέρηση - Low Latency

Η επεξεργασία και η προώθηση των κρίσιμων μηνυμάτων από την παραγωγή έως την οπτικοποίηση πρέπει να ολοκληρώνεται με ελάχιστη καθυστέρηση.

NFR-03: Διαθεσιμότητα και Αναπαραγωγιμότητα

Χρήση containerized τεχνολογιών για τη διασφάλιση της εύκολης επανεκκίνησης, της απομόνωσης των υπηρεσιών και της εύκολης ανάπτυξης του περιβάλλοντος εκτέλεσης

3.1.3 Σύνδεση απαιτήσεων με αρχιτεκτονική

Για τη διασφάλιση της κάλυψης όλων των προδιαγραφών, ο ακόλουθος πίνακας συνδέει κάθε λειτουργική απαίτηση με το αντίστοιχο τρόπο υλοποίησης.

Κωδικός Απαίτησης	Περιγραφή Απαίτησης	Υποσύστημα / Τεχνολογία Υλοποίησης
FR-01	Συλλογή δεδομένων σε πραγματικό χρόνο	Python IoT Simulator & Eclipse Mosquitto (MQTT Broker)
FR-02	Φιλτράρισμα και μετασχηματισμός δεδομένων	Node-RED (Middleware ETL Flows)
FR-03	Αποθήκευση ιστορικότητας & χρονοσειρών	SurrealDB (Multi-model Database)
FR-04	Πρόβλεψη παραγωγής & Ανίχνευση ανωμαλιών	Python Flask Microservice (Linear Regression Model)

FR-05	Οπτικοποίηση & Real-time Alerts	Grafana Dashboards & SMTP/Email Notification Channels
-------	---------------------------------	---

Πίνακας 1 Σύνδεση απαιτήσεων με τον τρόπο υλοποίησης

3.2 Γενική αρχιτεκτονική του συστήματος

Η αρχιτεκτονική του συστήματος ακολουθεί την πολυεπίπεδη προσέγγιση των IoT συστημάτων υιοθετώντας μια υβριδική προσέγγιση επικοινωνίας (MQTT και REST HTTP) για τη μεγιστοποίηση της αποδοτικότητας.

3.2.1 Επίπεδο Παραγωγής Δεδομένων (Data Generation Layer)

Στο επίπεδο παραγωγής δεδομένων αναπτύσσεται ο προσομοιωτής, ο οποίος αναπαριστά μετρήσεις κατανάλωσης και διαφόρων παραμέτρων. Τα δεδομένα αυτά μεταδίδονται μέσω του πρωτοκόλλου MQTT σε topics, υλοποιώντας το πρότυπο publish/subscribe. Η ανάπτυξη του προσομοιωτή πραγματοποιείται σε Python.

3.2.2 Επίπεδο Μεταφοράς & Διαμεσολάβησης (Communication Layer)

Ο MQTT broker, υλοποιημένος μέσω του Eclipse Mosquitto, λειτουργεί ως κεντρικός διαμεσολαβητής επικοινωνίας, αποσυνδέοντας τη διαδικασία παραγωγής δεδομένων από τη διαδικασία κατανάλωσής τους. Με τον τρόπο αυτό επιτυγχάνεται χαλαρή σύνδεση μεταξύ των επιμέρους υποσυστημάτων, γεγονός που ενισχύει την επεκτασιμότητα και την ευελιξία του συστήματος.

3.2.3 Επίπεδο Επεξεργασίας & Middleware (Data Processing Layer)

Στο επόμενο επίπεδο, το Node-RED πραγματοποιεί το subscription στα MQTT topics, τη μετατροπή των δεδομένων και τη δρομολόγησή τους προς τη βάση δεδομένων. Το επίπεδο αυτό λειτουργεί ως middleware και επιτρέπει την οπτική μοντελοποίηση των ροών δεδομένων, καθιστώντας τη δομή του συστήματος διαφανή και εύκολα επεκτάσιμη. Πιο συγκεκριμένα, δημιουργεί σύνδεση με τον Mosquitto broker, κάνει subscribe σε ένα ή περισσότερα topics. Όταν ο Python simulator δημοσιεύει ένα μήνυμα το Node-RED το λαμβάνει άμεσα.

Τα MQTT μηνύματα έρχονται ως JSON string. Το Node-RED τα μετατρέπει σε JavaScript objects, μπορεί να αλλάξει δομή τους, να προσθέσει πεδία και να υπολογίσει επιπλέον τιμές. Δηλαδή λειτουργεί ως ETL μικρής κλίμακας (Extract–Transform–Load). Το Node-RED μπορεί να ελέγχει αν λείπουν πεδία, απορρίπτει μη έγκυρα μηνύματα, φιλτράρει ακραίες τιμές προστατεύοντας έτσι τη βάση δεδομένων. Ανάλογα με το περιεχόμενο του μηνύματος, μπορεί να το στείλει στη βάση δεδομένων, να ενεργοποιήσει alert ή να το προωθήσει σε REST endpoint.

3.2.4 Επίπεδο Αποθήκευσης & Παρουσίασης (Storage & Visualization Layer)

Η αποθήκευση πραγματοποιείται σε βάση δεδομένων χρονοσειρών, η οποία έχει σχεδιαστεί ώστε να υποστηρίζει αποδοτικά queries χρονικών παραθύρων, aggregations και αναλύσεις σε μεγάλα σύνολα δεδομένων. Στο επίπεδο παρουσίασης, το Grafana συνδέεται στη βάση δεδομένων και παρέχει διαδραστικά dashboards για την παρακολούθηση της ενεργειακής συμπεριφοράς και τον υπολογισμό δεικτών απόδοσης.

Η αρχιτεκτονική αυτή μπορεί να αντικαταστήσει τον προσομοιωτή με πραγματικούς αισθητήρες χωρίς δομικές αλλαγές, στοιχείο που την καθιστά ρεαλιστική και επεκτάσιμη.

3.3 Συλλογή δεδομένων (IoT & smart meters)

Η συλλογή δεδομένων υλοποιείται μέσω προσομοιωτή αναπτυγμένου σε γλώσσα Python. Η επιλογή της Python αιτιολογείται από την ευκολία ανάπτυξης, τη διαθεσιμότητα βιβλιοθηκών μαθηματικής προσομοίωσης και τη συμβατότητά της με βιβλιοθήκες MQTT, όπως η paho-mqtt. Η Python επιτρέπει την ταχεία δημιουργία συνθετικών χρονοσειρών που προσομοιώνουν ρεαλιστικά ενεργειακά προφίλ.

Η μετάδοση των δεδομένων πραγματοποιείται μέσω του πρωτοκόλλου MQTT. Το MQTT επιλέχθηκε λόγω της ελαφριάς φύσης του, της χαμηλής καθυστέρησης και της ευρείας χρήσης του σε IoT και Smart Grid εφαρμογές. Το Eclipse Mosquitto χρησιμοποιείται ως broker, καθώς αποτελεί αξιόπιστη, ελαφριά και ευρέως διαδεδομένη υλοποίηση του προτύπου MQTT.

3.4 Μοντελοποίηση Δεδομένων και Σχεδιασμός Βάσης (SurrealDB Schema)

Η αποθήκευση των ενεργειακών μετρήσεων πραγματοποιείται σε βάση δεδομένων που υποστηρίζει δεδομένα χρονοσειρών. (SurrealDB - προσφέρει δυνατότητες αποθήκευσης και διαχείρισης χρονικών σειρών με βελτιστοποιημένη εκτέλεση ερωτημάτων)

Η χρήση βάσης χρονοσειρών αιτιολογείται από τα χαρακτηριστικά των ενεργειακών δεδομένων, τα οποία περιλαμβάνουν υψηλή συχνότητα δειγματοληψίας, ανάγκη εκτέλεσης ερωτημάτων σε χρονικά παράθυρα και υπολογισμό συναθροιστικών δεικτών. Το μοντέλο δεδομένων βασίζεται στη χρονική σήμανση, στο αναγνωριστικό του μετρητή και στις επιμέρους μετρήσεις, επιτρέποντας ευέλικτες αναλύσεις και ιστορικότητα. Το Node-RED λειτουργεί ως επίπεδο ενοποίησης, διαχειριζόμενο τη μετατροπή των JSON μηνυμάτων και την αποθήκευσή τους στη βάση δεδομένων.

Η μοντελοποίηση των ενεργειακών οντοτήτων υλοποιείται μέσω τριών βασικών πινάκων, οι οποίοι εξασφαλίζουν σαφή διαχωρισμό μεταξύ πρωτογενούς τηλεμετρίας, προβλέψεων και συμβάντων:

3.4.1 Πίνακας telemetry (Καταγραφή Χρονοσειρών)

Αποθηκεύει τις πρωτογενείς ενεργειακές και περιβαλλοντικές μετρήσεις.

- timestamp: datetime (ISO 8601 σήμανση χρόνου)
- meter_id: string (Μοναδικό αναγνωριστικό μετρητή)
- load_power_kw: float (Στιγμιαία ισχύς κατανάλωσης)
- pv_power_kw: float (Παραγόμενη ισχύς φωτοβολταϊκών)
- wind_power_kw: float (Παραγόμενη ισχύς αιολικών)
- battery_soc: float (Κατάσταση φόρτισης συσσωρευτή %)
- solar_irradiance_wm2: float (Ηλιακή ακτινοβολία σε W/m^2)
- temperature_c: float (Θερμοκρασία περιβάλλοντος σε $^{\circ}\text{C}$)

3.4.2 Πίνακας prediction (Δεδομένα Προβλέψεων)

Αποθηκεύει τις πραγματικές τιμές σε σύγκριση με τις προβλέψεις του μοντέλου.

- timestamp: datetime
- meter_id: string
- actual_value: float (Η πραγματική τιμή μέτρησης)
- predicted_value: float (Η προβλεπόμενη τιμή από το μοντέλο Linear Regression)
- error_pct: float (Το ποσοστό σφάλματος/απόκλισης)

3.4.3 Πίνακας alerts (Διαχείριση Συμβάντων)

Καταγράφει τις ειδοποιήσεις που προκύπτουν από τη rule-based λογική του Node-RED.

- timestamp: datetime
- meter_id: string
- alert_type: string (Ο τύπος της ειδοποίησης, π.χ. HIGH_LOAD)
- severity: string (Επίπεδο σοβαρότητας: high, medium)
- message: string (Περιγραφικό μήνυμα)
- value: float (Η τιμή που προκάλεσε το συμβάν)
- threshold: float (Το κατώφλι παραβίασης)

3.5 Ανάλυση δεδομένων

Η ανάλυση των δεδομένων υλοποιείται ως ανεξάρτητο module, ώστε να εξασφαλίζεται σαφής διαστρωμάτωση. Το module αυτό αντλεί τις απαραίτητες χρονοσειρές από τη βάση δεδομένων, εφαρμόζοντας υπολογισμούς και μεθόδους επεξεργασίας πάνω στα δεδομένα.

3.6 Οπτικοποίηση δεδομένων και dashboard

Το επίπεδο οπτικοποίησης υλοποιείται μέσω της πλατφόρμας Grafana, η οποία αποτελεί ευρέως διαδεδομένο εργαλείο παρακολούθησης χρονοσειρών. Το Grafana συνδέεται στη βάση δεδομένων και εκτελεί queries για την απεικόνιση των ενεργειακών μετρήσεων σε μορφή γραφημάτων, πινάκων και δεικτών απόδοσης. Η επιλογή του Grafana αιτιολογείται από τη συμβατότητά της με βάσεις χρονοσειρών, τη δυνατότητα δημιουργίας διαδραστικών dashboards και την ενσωμάτωση μηχανισμών ειδοποιήσεων. Στο Grafana μπορούν να υπολογιστούν με κατάλληλα queries στη βάση τα KPIs όπως κατανάλωση ανά ώρα/ημέρα, μέγιστη ζήτηση, συντελεστές αιχμής. Το dashboard λειτουργεί ως κεντρικός πίνακας ελέγχου, επιτρέποντας την ανάλυση χρονικών περιόδων, τη σύγκριση δεδομένων και την παρακολούθηση αιχμών κατανάλωσης.

4 Υλοποίηση του Συστήματος

4.1 Περιβάλλον Ανάπτυξης και Containerization

Για τη διασφάλιση της αναπαραγωγιμότητας και της εύκολης διαχείρισης του συστήματος, ολόκληρη η υποδομή αναπτύχθηκε με τη χρήση τεχνολογιών containerization μέσω της πλατφόρμας Docker και του εργαλείου Docker Compose. Κάθε υποσύστημα (Simulator, Broker, Middleware, Database, Analytics, Visualization) εκτελείται απομονωμένο σε δικό του container, ενώ η επικοινωνία μεταξύ τους επιτυγχάνεται μέσω ενός κοινού εσωτερικού εικονικού δικτύου (Docker bridge network).

4.2 Προσομοιωτής IoT και Παραγωγή Δεδομένων

Δεδομένης της απουσίας φυσικής εγκατάστασης αισθητήρων στο τρέχον στάδιο της εργασίας, η ροή δεδομένων τηλεμετρίας παράγεται από έναν προσομοιωτή αναπτυγμένο σε γλώσσα Python 3. Ο κώδικας παρατίθεται στο Παράρτημα Α. Ο προσομοιωτής παράγει δεδομένα τηλεμετρίας, προσομοιώνοντας τη συμπεριφορά του ενεργειακού δικτύου μιας επιχείρησης. Πιο συγκεκριμένα αξιοποιεί τα φορτία κατανάλωσης, τη φωτοβολταϊκή παραγωγή, την αιολική παραγωγή και την αποθήκευση ενέργειας. Ο προσομοιωτής αναπαριστά ένα σύνολο εικονικών μετρητών, καθένας από τους οποίους παρακολουθεί ένα σημείο της εγκατάστασης. Οι μετρητές αυτοί παράγουν δεδομένα σε προκαθορισμένα χρονικά διαστήματα και στέλνουν τις μετρήσεις μέσω MQTT σε topics, ακολουθώντας τη λογική ενός πραγματικού IoT περιβάλλοντος. Ο προσομοιωτής αξιοποιεί τη βιβλιοθήκη `raho-mqtt` για τη σύνδεση και δημοσίευση μηνυμάτων.

Πιο αναλυτικά, προσομοιώνονται η ισχύς κατανάλωσης φορτίου, η παραγωγή από φωτοβολταϊκά συστήματα, η παραγωγή από αιολικά συστήματα, η κατάσταση της φόρτισης συσσωρευτή, η εισαγωγή και εξαγωγή ισχύος από/προς το δίκτυο, καθώς και περιβαλλοντικές μετρήσεις όπως η θερμοκρασία, η ταχύτητα ανέμου και η ηλιακή ακτινοβολία. Επιπλέον, για την ενεργειακή παρακολούθηση προσομοιώνονται οι τιμές τάσης, ρεύματος και συντελεστή ισχύος.

Κάθε σύνολο δεδομένων παράγεται σαν JSON μήνυμα και περιλαμβάνει τη χρονική στιγμή, το αναγνωριστικό του μετρητή και το σύνολο των μετρούμενων ή υπολογιζόμενων μεταβλητών. Το μήνυμα αποστέλλεται μέσω MQTT σε topic της μορφής `energy/meter/{meter_id}`. Αυτό επιτρέπει στο σύστημα να επεξεργάζεται ανεξάρτητα δεδομένα από διαφορετικές πηγές. Το πεδίο «timestamp» αποθηκεύει τον χρόνο παραγωγής του δείγματος σε μορφή ISO 8601 και χρησιμοποιείται για την αποθήκευση και ανάλυση χρονοσειρών. Το πεδίο «meter_id» προσδιορίζει μοναδικά τον εικονικό μετρητή ή το σημείο συλλογής τηλεμετρίας.

Ενδεικτικά ένα μήνυμα που παράγεται από τον προσομοιωτή έχει την εξής μορφή:

```
{  
  "timestamp": "2026-04-05T10:15:00Z",  
  "meter_id": "m1",  
  "load_power_kw": 45.2,  
  "load_energy_kwh": 1032.44,  
  "pv_power_kw": 18.4,  
  "pv_energy_kwh": 412.30,  
  "wind_power_kw": 12.1,  
  "wind_energy_kwh": 298.12,  
  "res_power_kw": 30.5,  
  "grid_import_kw": 14.7,  
  "grid_export_kw": 0.0,  
  "battery_soc": 72.5,  
  "battery_charge_kw": 0.0,  
  "battery_discharge_kw": 0.0,  
  "temperature_c": 23.4,  
  "wind_speed_mps": 6.8,  
  "solar_irradiance_wm2": 740.0,  
  "voltage_v": 229.7,  
  "current_a": 71.4,  
  "pf": 0.94  
}
```

Εικόνα 2 Μήνυμα που παράγεται από τον προσομοιωτή

4.2.1 Ισχύς Κατανάλωσης φορτίου

Η κατανάλωση φορτίου βασίζεται σε ημιτονοειδή συνάρτηση ώστε να γίνεται αναπαράσταση της διακύμανση ενός ενεργειακού φορτίου κατά τη διάρκεια της ημέρας. Σε αυτό προστέθηκε στοχαστικός θόρυβος, για να αποφευχθεί η περιοδική και μη ρεαλιστική

συμπεριφορά. Επιπλέον, έχουν εισαχθεί κάποιες αιχμές φορτίου ώστε να προσομοιώνονται πιο ρεαλιστικά δεδομένα αυξημένης κατανάλωσης. Αυτά είναι χρήσιμα για την ανίχνευση αιχμών και την ενεργοποίηση ειδοποιήσεων. Τα πεδία «load_power_kw» και «load_energy_kwh» εκφράζουν αντίστοιχα τη στιγμιαία ισχύ φορτίου και τη σωρευτική καταναλωθείσα ενέργεια.

4.2.2 Παραγωγή από φωτοβολταϊκά συστήματα

Η παραγωγή των φωτοβολταϊκών συστημάτων εξαρτάται από την προσομοιωμένη ηλιακή ακτινοβολία, η οποία επαναλαμβάνεται κάθε μέρα περιοδικά. Κατά τη διάρκεια της νύχτας η τιμή της ακτινοβολίας μηδενίζεται, ενώ κατά τη διάρκεια της ημέρας αυξάνεται, σχηματίζοντας μια καμπύλη που προσομοιάζει τη φυσική μεταβολή της ηλιοφάνειας. Η ισχύς που παραγεται από τα φωτοβολταϊκά είναι ανάλογη της ακτινοβολίας και της ονομαστικής ισχύος του φωτοβολταϊκού συστήματος. Με τον τρόπο αυτό πετυχαίνουμε μια επαρκή προσέγγιση της PV παραγωγής. Η παραγωγή από φωτοβολταϊκά συστήματα αναπαρίσταται με τα πεδία «pv_power_kw», «pv_energy_kwh».

4.2.3 Παραγωγή από αιολικά συστήματα

Η αιολική παραγωγή εξαρτάται από την ταχύτητα του ανέμου. Για την προσομοίωσή της χρησιμοποιήθηκε περιοδική συνάρτηση και τυχαίος θόρυβος. Η ισχύς που παραγεται από την ανεμογεννήτρια υπολογίζεται μέσω καμπύλης ισχύος, η οποία ενσωματώνει την ταχύτητα έναρξης λειτουργίας (cut-in speed), την ονομαστική περιοχή λειτουργίας (rated region) και την ταχύτητα αποσύνδεσης (cut-out speed). Έτσι, η αιολική παραγωγή εξαρτάται από τις μεταβολές του ανέμου και αποκτά στοχαστικό χαρακτήρα όπως και στα πραγματικά συστήματα. Η παραγωγή από αιολικά συστήματα αναπαρίσταται με τα πεδία «wind_power_kw», «wind_energy_kwh», ενώ το πεδίο res_power_kw εκφράζει τη συνολική στιγμιαία ισχύ που παράγεται από ΑΠΕ.

4.2.4 Κατάσταση φόρτισης συσσωρευτή

Η λειτουργία του συστήματος αποθήκευσης προσομοιώνεται μέσω μεταβλητής κατάστασης φόρτισης (State of Charge). Όταν η παραγωγή από ανανεώσιμες πηγές υπερβαίνει το φορτίο, ο συσσωρευτής φορτίζεται έως το μέγιστο επιτρεπτό επίπεδο.

Αντίθετα, όταν η κατανάλωση είναι μεγαλύτερη από την παραγωγή, το σύστημα προσπαθεί να καλύψει μέρος του ελλείμματος με χρήση της μπαταρίας. Το «battery_soc» εκφράζει την κατάσταση φόρτισης του συσσωρευτή σε ποσοστό. Τα «battery_charge_kw» και «battery_discharge_kw» αναφέρονται στις στιγμιαίες ισχύεις φόρτισης και εκφόρτισης.

4.2.5 Εισαγωγή και εξαγωγή ισχύος

Ο προσομοιωτής υπολογίζει την καθαρή ισχύ εισαγωγής ή εξαγωγής από το ηλεκτρικό δίκτυο, δηλαδή την ισχύ που απομένει μετά τη συνεκτίμηση κατανάλωσης, παραγωγής και αποθήκευσης. Η αλληλεπίδραση με το δίκτυο αναπαρίσταται μέσω των «grid_import_kw» και «grid_export_kw».

4.2.6 Περιβαλλοντικές μετρήσεις

Οι περιβαλλοντικές συνθήκες περιγράφονται από τα «temperature_c», «wind_speed_mps» και «solar_irradiance_wm2».

4.2.7 Κατάσταση λειτουργίας του συστήματος

Οι τιμές τάσης, ρεύματος και συντελεστή ισχύος περιγράφονται από τα «voltage_v», «current_a» και «pf».

4.2.8 Περιορισμοί

Η αναγνώριση των περιορισμών είναι απαραίτητη για την σωστή ερμηνεία των αποτελεσμάτων και τη μελλοντική επέκταση του συστήματος σε πραγματικά περιβάλλοντα. Ο προσομοιωτής βασίζεται σε απλουστευμένες μαθηματικές προσεγγίσεις. Αρχικά, το μοντέλο κατανάλωσης βασίζεται σε ημερήσια περιοδική συνάρτηση, ενώ στην πραγματικότητα, η κατανάλωση ενέργειας επηρεάζεται από πολλούς επιπλέον παράγοντες, όπως η λειτουργία συγκεκριμένων μηχανημάτων, οι ώρες λειτουργίας της επιχείρησης, οι απρόβλεπτες μεταβολές στη ζήτηση. Παρόμοια, η παραγωγή από φωτοβολταϊκά συστήματα υπολογίζεται ως γραμμική συνάρτηση της ηλιακής ακτινοβολίας. Στην πραγματικότητα, η απόδοση των φωτοβολταϊκών επηρεάζεται από πρόσθετους παράγοντες όπως η θερμοκρασία των κυψελών, η γωνία πρόσπτωσης της ηλιακής ακτινοβολίας, η σκίαση και οι απώλειες μετατροπής. Η αιολική παραγωγή προσεγγίζεται μέσω απλοποιημένης καμπύλης ισχύος, βασισμένης στην ταχύτητα του ανέμου. Στην πράξη, η

λειτουργία των ανεμογεννητριών επηρεάζεται από επιπλέον παράγοντες, όπως η κατεύθυνση του ανέμου, η πυκνότητα του αέρα, οι μηχανικές απώλειες. Το μοντέλο που χρησιμοποιείται εδώ δεν λαμβάνει υπόψη αυτούς τους παράγοντες. Το μοντέλο αποθήκευσης ενέργειας βασίζεται σε απλή γραμμική προσέγγιση φόρτισης και εκφόρτισης. Δεν λαμβάνονται υπόψη χαρακτηριστικά όπως η απόδοση φόρτισης/εκφόρτισης, οι απώλειες μετατροπής, η γήρανση των συσσωρευτών. Επιπλέον, οι ηλεκτρικές μεταβλητές, όπως η τάση και το ρεύμα, προσεγγίζονται μέσω απλών μαθηματικών σχέσεων και στοχαστικού θορύβου. Στην πραγματικότητα, τα ηλεκτρικά μεγέθη παρουσιάζουν πιο σύνθετη συμπεριφορά και επηρεάζονται από χαρακτηριστικά του δικτύου και του εξοπλισμού.

4.3 Μεταφορά Δεδομένων - Διαμεσολαβητής Μηνυμάτων

Για τη μεταφορά των δεδομένων τηλεμετρίας στο υπό ανάπτυξη σύστημα επιλέχθηκε το πρωτόκολλο MQTT (Message Queuing Telemetry Transport), το οποίο αποτελεί ένα ελαφρύ πρωτόκολλο ανταλλαγής μηνυμάτων σχεδιασμένο για εφαρμογές Internet of Things (IoT) και περιβάλλοντα με περιορισμένους πόρους. Ο προσομοιωτής δεδομένων λειτουργεί ως publisher, αποστέλλοντας μηνύματα τηλεμετρίας σε συγκεκριμένα topics. Ο MQTT broker, ο οποίος υλοποιείται μέσω του Eclipse Mosquitto, λαμβάνει τα μηνύματα και τα προωθεί στους subscribers. Το Node-RED λειτουργεί ως subscriber, λαμβάνοντας τα δεδομένα και δρομολογώντας τα προς αποθήκευση και περαιτέρω επεξεργασία.

Για την οργάνωση των δεδομένων χρησιμοποιήθηκε ιεραρχική δομή MQTT topics. Η δομή αυτή επιτρέπει την εύκολη κατηγοριοποίηση των δεδομένων. Η δομή που χρησιμοποιήθηκε είναι η εξής:

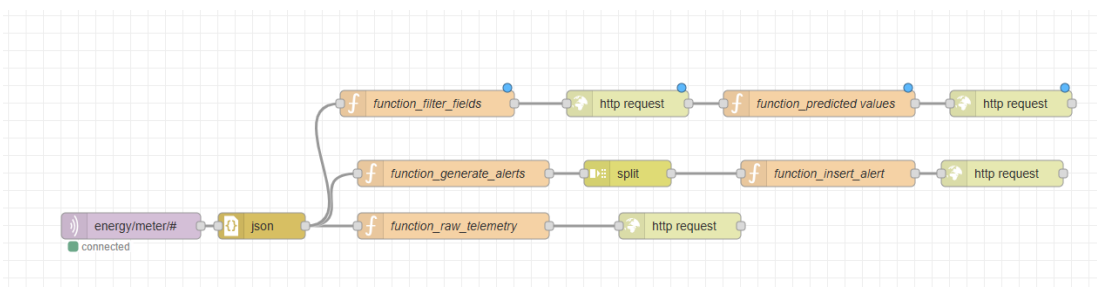
energy/meter/{meter_id}, όπου:

- energy δηλώνει την κατηγορία ενεργειακών δεδομένων
- meter δηλώνει ότι πρόκειται για δεδομένα μετρητών
- {meter_id} είναι το μοναδικό αναγνωριστικό της πηγής δεδομένων

Για παράδειγμα, ένας μετρητής με αναγνωριστικό m1 αποστέλλει δεδομένα στο topic: energy/meter/m1. Το Node-RED διαβάζει από το wildcard topic: energy/meter/#. Με τον τρόπο αυτό λαμβάνονται όλα τα δεδομένα ανεξαρτήτως πηγής.

4.4 Ενδιάμεσο Επίπεδο Επεξεργασίας (Node-RED Middleware)

Το Node-RED χρησιμοποιείται ως ενδιάμεσο επίπεδο μεταξύ του επιπέδου συλλογής δεδομένων και του επιπέδου αποθήκευσης και οπτικοποίησης. Λειτουργεί ως κεντρικός κόμβος επεξεργασίας, μετασχηματισμού και δρομολόγησης των δεδομένων που παράγονται από τον προσομοιωτή και μεταδίδεται μέσω MQTT.



Εικόνα 3 Ροές Node-RED

Το Node-RED διαβάζει τα μηνύματα από τα topics που χρησιμοποιούνται για τη μεταφορά των δεδομένων. Τα δεδομένα που λαμβάνονται είναι σε μορφή JSON. Μετά τη μετατροπή του εισερχόμενου μηνύματος σε JSON, η ροή διαχωρίζεται σε δύο βασικούς κλάδους. Ο πρώτος κλάδος αφορά την αποθήκευση της πρωτογενούς τηλεμετρίας, ενώ ο δεύτερος κλάδος αφορά την παραγωγή ειδοποιήσεων σε πραγματικό χρόνο. Ο διαχωρισμός αυτός επιτρέπει την ανεξάρτητη διαχείριση των δεδομένων και των συμβάντων, καθώς και την προσθήκη επιπλέον λειτουργιών χωρίς τροποποίηση της βασικής ροής.

4.4.1 Κλάδος Αποθήκευσης Τηλεμετρίας

Στον πρώτο κλάδο, τα δεδομένα τηλεμετρίας επεξεργάζονται από έναν κόμβο function (function_raw_telemetry), ο οποίος μετασχηματίζει τα εισερχόμενα δεδομένα σε κατάλληλη μορφή για αποθήκευση. Η συνάρτηση παρατίθεται στο Παράρτημα Β. Στη συνέχεια, τα δεδομένα αποστέλλονται στη βάση δεδομένων μέσω HTTP request. Η

προσέγγιση αυτή επιτρέπει τη διατήρηση των πρωτογενών δεδομένων σε μορφή χρονοσειρών, οι οποίες μπορούν να χρησιμοποιηθούν για ιστορική ανάλυση, υπολογισμό δεικτών απόδοσης και ανάπτυξη μοντέλων πρόβλεψης.

4.4.2 Κλάδος Παραγωγής Ειδοποιήσεων

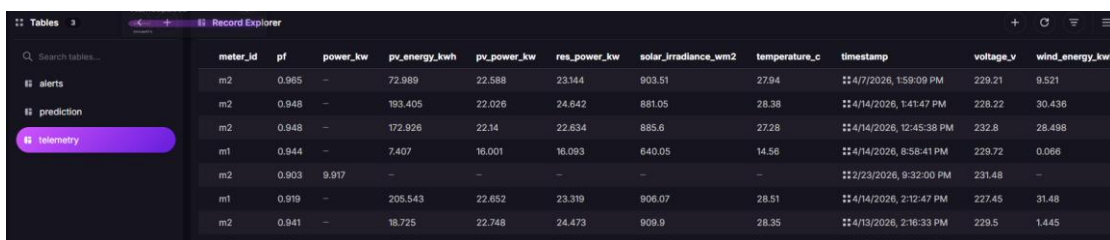
Ο δεύτερος κλάδος της ροής υλοποιεί μηχανισμούς ανίχνευσης συμβάντων σε πραγματικό χρόνο. Στο επίπεδο αυτό, τα δεδομένα τηλεμετρίας συγκρίνονται με προκαθορισμένα όρια ή συνδυαστικές συνθήκες λειτουργίας (`function_generate_alerts` - Παράρτημα Β). Εάν εντοπιστεί απόκλιση από την κανονική λειτουργία, δημιουργείται ένα alert, το οποίο περιλαμβάνει πληροφορίες όπως η χρονική σήμανση, το αναγνωριστικό της πηγής, ο τύπος του συμβάντος, το επίπεδο σοβαρότητας και η τιμή που προκάλεσε την ειδοποίηση. Τα alerts αυτά μετατρέπονται σε ξεχωριστές εγγραφές (`function_insert_alert` - Παράρτημα Β) και αποθηκεύονται στη βάση δεδομένων, επιτρέποντας την παρακολούθηση και ανάλυση των συμβάντων μέσω του dashboard.

4.4.3 Κλάδος Διασύνδεσης Analytics

Ο τρίτος κλάδος της ροής υλοποιεί μηχανισμούς ανάλυσης και πρόβλεψης δεδομένων σε πραγματικό χρόνο. Στο επίπεδο αυτό, τα δεδομένα τηλεμετρίας στέλνονται σε ένα python microservice μέσω HTTP request. Το microservice αυτό χρησιμοποιεί ένα μοντέλο το οποίο έχει εκπαιδευθεί με δεδομένα και παράγει προβλέψεις για τις τιμές που θα έπρεπε να δίνει το σύστημα δεδομένων κάποιων συνθηκών (`function_predicted_values`- Παράρτημα Β). Οι τιμές των δεδομένων που λαμβάνονται από τους μετρητές και οι τιμές που προβλέπονται από το μοντέλο χρησιμοποιώντας τον αλγόριθμο του Linear Regression, αποστέλονται στη βάση δεδομένων για να χρησιμοποιηθούν στη συνέχεια από το Grafana για να δημιουργηθούν τα κατάλληλα dashboards και ειδοποιήσεις σε περιπτώσεις που οι αποκλίσεις υπερβαίνουν κάποιο προκαθορισμένο κατώφλι.

4.5 Αποθήκευση Δεδομένων

Η βάση δεδομένων του συστήματος οργανώνεται σε πίνακες, ώστε να επιτυγχάνεται σαφής διαχωρισμός μεταξύ πρωτογενούς τηλεμετρίας και παραγόμενων συμβάντων. Η δομή αυτή επιτρέπει την αποθήκευση των δεδομένων ως χρονοσειρές και υποστηρίζει τόσο τη real-time παρακολούθηση όσο και την ιστορική ανάλυση.



meter_id	pf	power_kw	pv_energy_kwh	pv_power_kw	res_power_kw	solar_irradiance_wm2	temperature_c	timestamp	voltage_v	wind_energy_kwh
m2	0.965	-	72.989	22.588	23.144	903.51	27.94	11/4/2026, 1:59:09 PM	229.21	9.521
m2	0.948	-	193.405	22.026	24.842	881.05	28.38	11/4/2026, 1:41:47 PM	228.22	30.436
m2	0.948	-	172.926	22.14	22.634	885.6	27.28	11/4/2026, 12:45:38 PM	232.8	28.498
m1	0.944	-	7.407	16.001	16.093	640.05	14.56	11/4/2026, 8:58:41 PM	229.72	0.066
m2	0.903	9.917	-	-	-	-	-	11/2/2026, 9:32:00 PM	231.48	-
m1	0.919	-	205.543	22.652	23.319	906.07	28.51	11/4/2026, 2:12:47 PM	227.45	31.48
m2	0.941	-	18.725	22.748	24.473	909.9	28.35	11/4/2026, 2:16:33 PM	229.5	1.445

Εικόνα 4 Βάση δεδομένων

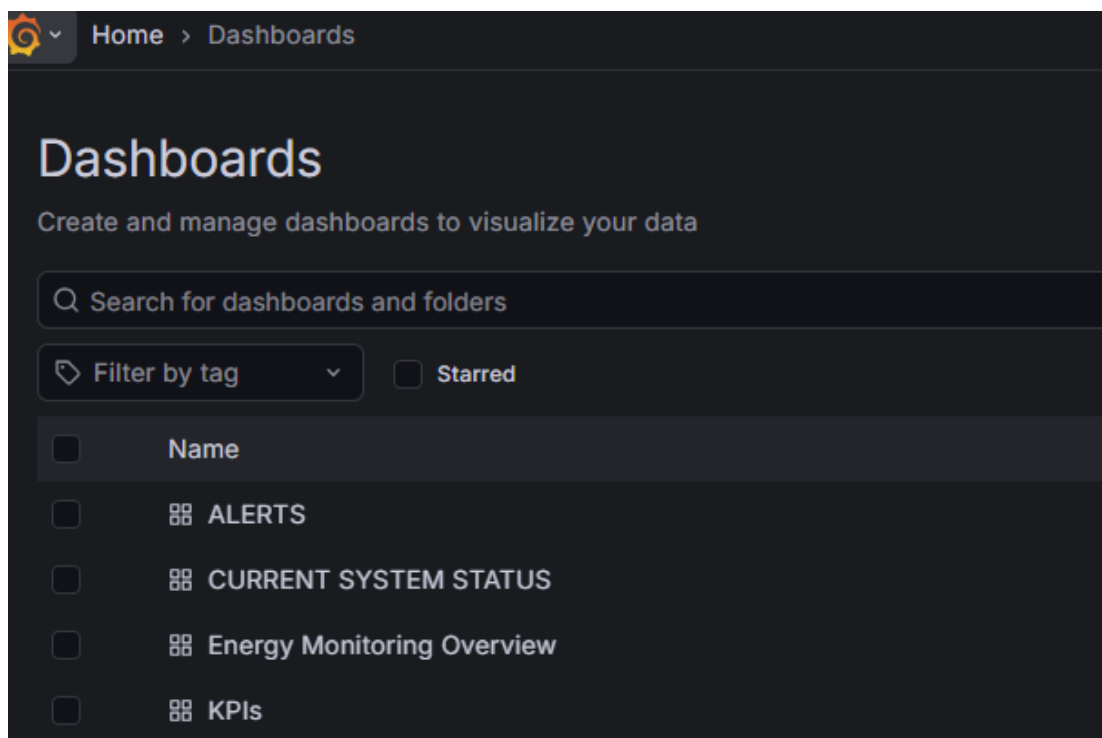
Ο πίνακας «telemetry» χρησιμοποιείται για την αποθήκευση των ενεργειακών και περιβαλλοντικών μετρήσεων που παράγονται από τον προσομοιωτή. Κάθε εγγραφή περιλαμβάνει χρονική σήμανση, αναγνωριστικό μετρητή και ένα σύνολο μεταβλητών που αφορούν την κατανάλωση φορτίου, την παραγωγή από φωτοβολταϊκά και αιολικά συστήματα, τη λειτουργία αποθήκευσης ενέργειας, την αλληλεπίδραση με το δίκτυο, καθώς και βασικά ηλεκτρικά και περιβαλλοντικά μεγέθη.

Ο πίνακας «alerts» χρησιμοποιείται για την αποθήκευση συμβάντων που παράγονται από τη rule-based λογική του Node-RED. Κάθε εγγραφή alert περιλαμβάνει τη χρονική στιγμή δημιουργίας του, το αναγνωριστικό μετρητή, τον τύπο του alert, το επίπεδο σοβαρότητας, μία περιγραφή, καθώς και την τιμή που ενεργοποίησε το συμβάν και το αντίστοιχο όριο

Ο πίνακας «predictions» χρησιμοποιείται για την αποθήκευση των μετρήσιμων τιμών από τους αισθητήρες και των προβλέψεων καθώς και των ανωμαλιών που ανιχνεύονται από το μοντέλο ανάλυσης δεδομένων.

4.6 Οπτικοποίηση δεδομένων και Dashboards

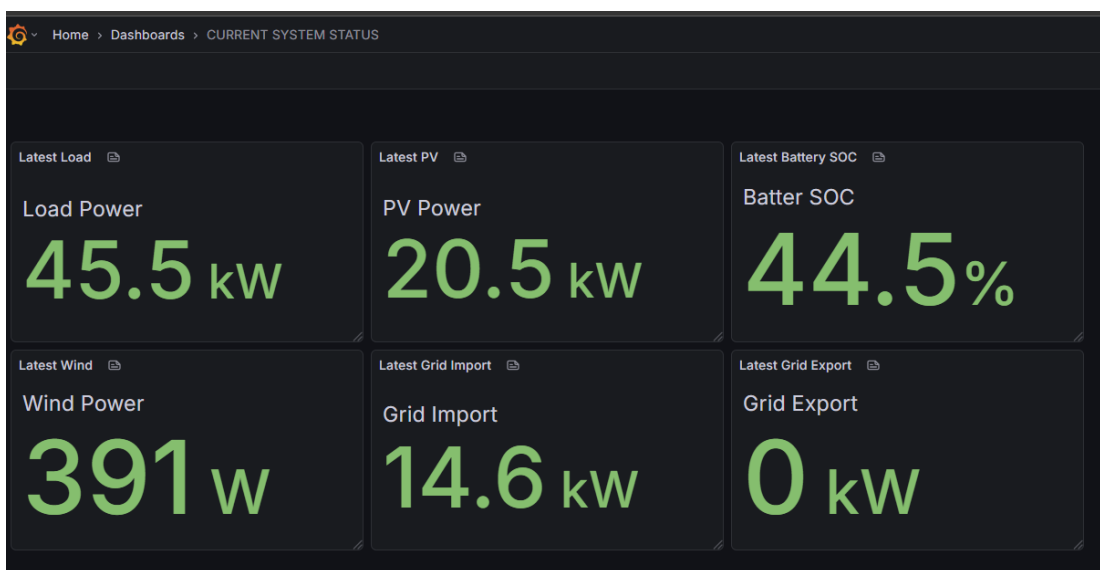
Η πλατφόρμα Grafana χρησιμοποιείται για τη δημιουργία Dashboards και για την απεικόνιση των δεδομένων σε πραγματικό χρόνο και την υποστήριξη λήψης αποφάσεων. Τα dashboards οργανώνονται σε τέσσερις βασικές ενότητες: Τρέχουσα Ενεργειακή Κατάσταση του Συστήματος (Current System Status Dashboard), Ενεργειακή Παρακολούθηση (Energy Monitoring Overview Dashboard), Δείκτες Απόδοσης (KPIs Dashboard) και Ειδοποιήσεις (Alerts Dashboard), οι οποίες παρέχουν διαφορετικά επίπεδα πληροφορίας και επιτρέπουν την ολοκληρωμένη παρακολούθηση του ενεργειακού συστήματος.



Εικόνα 5 Dashboards στην πλατφόρμα Grafana

4.6.1 Current System Status Dashboard

Το dashboard **Current System Status** παρουσιάζει τη στιγμιαία κατάσταση του ενεργειακού συστήματος με stat panels. Τα panels αυτά εμφανίζουν τις τελευταίες διαθέσιμες μετρήσεις και επιτρέπουν την άμεση εκτίμηση της λειτουργίας του συστήματος.



Εικόνα 6 Current System Status Dashboard

Συγκεκριμένα, το dashboard περιλαμβάνει τους ακόλουθους δείκτες:

Load Power

Ο δείκτης Load Power παρουσιάζει τη στιγμιαία ισχύ κατανάλωσης του ενεργειακού συστήματος σε kW. Η τιμή αυτή λαμβάνεται από τη μεταβλητή `load_power_kw` και επιτρέπει την άμεση εκτίμηση της ενεργειακής ζήτησης. Η παρακολούθηση της στιγμιαίας κατανάλωσης είναι ιδιαίτερα σημαντική για την ανίχνευση αιχμών φορτίου και την αξιολόγηση της ενεργειακής συμπεριφοράς του συστήματος.

PV Power

Ο δείκτης PV Power παρουσιάζει την τρέχουσα παραγωγή ισχύος από τα φωτοβολταϊκά συστήματα. Η τιμή αυτή υπολογίζεται από τη μεταβλητή `pv_power_kw` και επιτρέπει την αξιολόγηση της αξιοποίησης της ηλιακής ενέργειας.

Wind Power

Ο δείκτης Wind Power παρουσιάζει την παραγωγή ισχύος από αιολικά συστήματα. Η τιμή αυτή προκύπτει από τη μεταβλητή `wind_power_kw` και επιτρέπει την παρακολούθηση της μεταβλητότητας της αιολικής παραγωγής.

Battery State of Charge

Η κατάσταση φόρτισης της μπαταρίας παρουσιάζεται ως ποσοστό μέσω της μεταβλητής `battery_soc`. Η συγκεκριμένη μέτρηση επιτρέπει την αξιολόγηση της διαθεσιμότητας αποθηκευμένης ενέργειας και τη λήψη αποφάσεων σχετικά με τη διαχείριση της αποθήκευσης.

Grid Import

Ο δείκτης Grid Import παρουσιάζει την ισχύ που εισάγεται από το ηλεκτρικό δίκτυο, όταν η τοπική παραγωγή δεν επαρκεί για την κάλυψη της κατανάλωσης. Η τιμή αυτή υπολογίζεται από τη μεταβλητή `grid_import_kw`.

Grid Export

Ο δείκτης Grid Export παρουσιάζει την περίσσεια ενέργειας που διοχετεύεται στο δίκτυο όταν η παραγωγή από ανανεώσιμες πηγές υπερβαίνει την κατανάλωση. Η τιμή αυτή υπολογίζεται από τη μεταβλητή `grid_export_kw`.

4.6.2 Energy Monitoring Overview Dashboard

Το dashboard **Energy Monitoring Overview** παρουσιάζει την εξέλιξη των ενεργειακών μεγεθών στην εξέλιξη του χρόνου μέσω γραφημάτων χρονοσειρών.



Εικόνα 7 Energy Monitoring Overview Dashboard

Περιλαμβάνει τα παρακάτω γραφήματα:

Load Power Time Series

Το γράφημα αυτό παρουσιάζει την εξέλιξη της κατανάλωσης ενέργειας στο χρόνο. Τα δεδομένα προέρχονται από τη μεταβλητή `load_power_kw` και επιτρέπουν την αναγνώριση αιχμών φορτίου.

PV Power Time Series

Το γράφημα παρουσιάζει την παραγωγή από φωτοβολταϊκά συστήματα σε βάθος χρόνου. Η ανάλυση της παραγωγής επιτρέπει την αξιολόγηση της επίδρασης της ηλιακής ακτινοβολίας.

Wind Power Time Series

Η παραγωγή από αιολικά συστήματα παρουσιάζεται ως χρονοσειρά της μεταβλητής `wind_power_kw`. Η ανάλυση αυτή είναι σημαντική λόγω της μεταβλητότητας της αιολικής ενέργειας.

Total RES Power

Η συνολική παραγωγή από ανανεώσιμες πηγές υπολογίζεται ως:

$$\text{RES Power} = \text{PV Power} + \text{Wind Power}$$

Το γράφημα αυτό παρουσιάζει τη συνολική συνεισφορά των ανανεώσιμων πηγών στο σύστημα.

Grid Exchange

Το γράφημα Grid Exchange παρουσιάζει την εισαγωγή και εξαγωγή ενέργειας από το δίκτυο. Η παρακολούθηση αυτή επιτρέπει την αξιολόγηση της ενεργειακής αυτονομίας.

Battery State of Charge

Η εξέλιξη της φόρτισης της μπαταρίας παρουσιάζεται ως χρονοσειρά και επιτρέπει την ανάλυση της χρήσης του συστήματος αποθήκευσης.

4.6.3 KPIs Dashboard

Το dashboard KPIs περιλαμβάνει συγκεντρωτικούς δείκτες ενεργειακής απόδοσης, οι οποίοι υπολογίζονται από τα δεδομένα τηλεμετρίας.



Εικόνα 8 KPIs Dashboard

Total Consumption

Η συνολική κατανάλωση υπολογίζεται ως:

$$E_{load} = \int P_{load}(t) dt \text{ ή } E_{load} \approx \sum_i P_{load,i} \cdot \Delta t$$

Ο δείκτης αυτός παρουσιάζει τη συνολική ενεργειακή κατανάλωση.

Total Renewable Production

Η συνολική παραγωγή από ανανεώσιμες πηγές υπολογίζεται ως:

$$P_{RES} = P_{PV} + P_{Wind}$$

$$E_{RES} = \int P_{RES}(t) dt$$

Renewable Penetration

Η διείσδυση ανανεώσιμων πηγών υπολογίζεται ως:

$$RPR = \frac{E_{RES}}{E_{load}} \times 100\%$$

Ο δείκτης αυτός παρουσιάζει το ποσοστό κάλυψης της κατανάλωσης από ΑΠΕ.

Peak Load

Ο δείκτης μέγιστου φορτίου (Peak Load) αποτελεί έναν από τους σημαντικότερους δείκτες αξιολόγησης ενός ενεργειακού συστήματος, καθώς αποτυπώνει τη μέγιστη απαιτούμενη ισχύ σε ένα συγκεκριμένο χρονικό διάστημα. Ο δείκτης αυτός υπολογίζεται ως η μέγιστη τιμή της κατανάλωσης ισχύος κατά τη διάρκεια της περιόδου παρακολούθησης

$$P_{peak} = \max(P_{load}(t))$$

Average Load

Ο δείκτης μέσης κατανάλωσης παρουσιάζει τη μέση τιμή της ισχύος κατανάλωσης για συγκεκριμένο χρονικό διάστημα και παρέχει συνολική εικόνα της ενεργειακής ζήτησης του συστήματος. Ο δείκτης αυτός υπολογίζεται ως:

$$P_{avg} = \frac{1}{N} \sum_{i=1}^N P_{load}(i)$$

όπου $P_{load}(i)$ είναι η ισχύς κατανάλωσης και N ο αριθμός των μετρήσεων.

Grid Dependency

$$\text{Grid Dependency} = (\Sigma \text{grid_import_kw} / \Sigma \text{load_power_kw}) \times 100$$

Ο δείκτης εξάρτησης από το δίκτυο (Grid Dependency) εκφράζει το ποσοστό της συνολικής κατανάλωσης που καλύπτεται από την εισαγωγή ενέργειας από το ηλεκτρικό δίκτυο. Ο δείκτης αυτός υπολογίζεται ως:

$$\text{Grid Dependency} = \frac{\Sigma P_{\text{grid_import}}}{\Sigma P_{\text{load}}} \times 100$$

όπου $P_{\text{grid_import}}$ είναι η ισχύς που εισάγεται από το δίκτυο και P_{load} η συνολική κατανάλωση.

Renewable Self-Consumption Ratio

Ο δείκτης αυτοκατανάλωσης ανανεώσιμης ενέργειας εκφράζει το ποσοστό της παραγόμενης ενέργειας από ανανεώσιμες πηγές που καταναλώνεται τοπικά στο σύστημα, αντί να διοχετεύεται στο ηλεκτρικό δίκτυο. Ο δείκτης αυτός αποτελεί βασικό μέτρο αξιολόγησης της ενεργειακής αποδοτικότητας ενός μικροδικτύου ή ενός ενεργειακού συστήματος με ενσωματωμένες ανανεώσιμες πηγές και αποθήκευση ενέργειας. Ο δείκτης αυτοκατανάλωσης υπολογίζεται ως:

$$\text{Self Consumption} = \frac{\Sigma (P_{\text{RES}} - P_{\text{grid_export}})}{\Sigma P_{\text{RES}}} \times 100$$

όπου P_{RES} είναι η συνολική παραγόμενη ισχύς από ανανεώσιμες πηγές ενέργειας και $P_{\text{grid_export}}$ είναι η ισχύς που εξάγεται προς το ηλεκτρικό δίκτυο. Η διαφορά μεταξύ των δύο μεγεθών αντιστοιχεί στην ενέργεια που καταναλώνεται τοπικά.

CO₂ Emissions

$$CO_2 = E_{grid_import} \times EF$$

CO₂ Savings

$$CO_{2,avoided} = E_{RES,used\ locally} \times EF$$

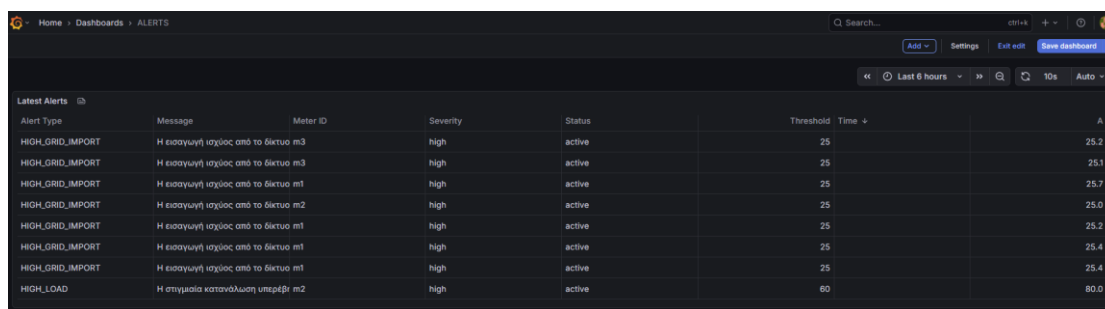
Οι δείκτες αυτοί επιτρέπουν την αξιολόγηση της περιβαλλοντικής επίδοσης.

4.6.4 Alerts Dashboard

Το dashboard Alerts παρουσιάζει τις ειδοποιήσεις που δημιουργούνται από το σύστημα ανίχνευσης ανωμαλιών. Οι ειδοποιήσεις δημιουργούνται όταν παρατηρούνται:

- υψηλή κατανάλωση
- υψηλή εξάρτηση από το δίκτυο
- χαμηλή στάθμη μπαταρίας
- χαμηλή παραγωγή ΑΠΕ

Το σύστημα ειδοποιήσεων επιτρέπει την έγκαιρη παρέμβαση και τη βελτιστοποίηση της λειτουργίας του ενεργειακού συστήματος.



Alert Type	Message	Meter ID	Severity	Status	Threshold	Time	A
HIGH_GRID_IMPORT	Η εισαγωγή ισχύος από το δίκτυο m3		high	active	25		25.2
HIGH_GRID_IMPORT	Η εισαγωγή ισχύος από το δίκτυο m3		high	active	25		25.1
HIGH_GRID_IMPORT	Η εισαγωγή ισχύος από το δίκτυο m1		high	active	25		25.7
HIGH_GRID_IMPORT	Η εισαγωγή ισχύος από το δίκτυο m2		high	active	25		25.0
HIGH_GRID_IMPORT	Η εισαγωγή ισχύος από το δίκτυο m1		high	active	25		25.2
HIGH_GRID_IMPORT	Η εισαγωγή ισχύος από το δίκτυο m1		high	active	25		25.4
HIGH_GRID_IMPORT	Η εισαγωγή ισχύος από το δίκτυο m1		high	active	25		25.4
HIGH_LOAD	Η στιγμιαία κατανάλωση υπερβή m2		high	active	60		80.0

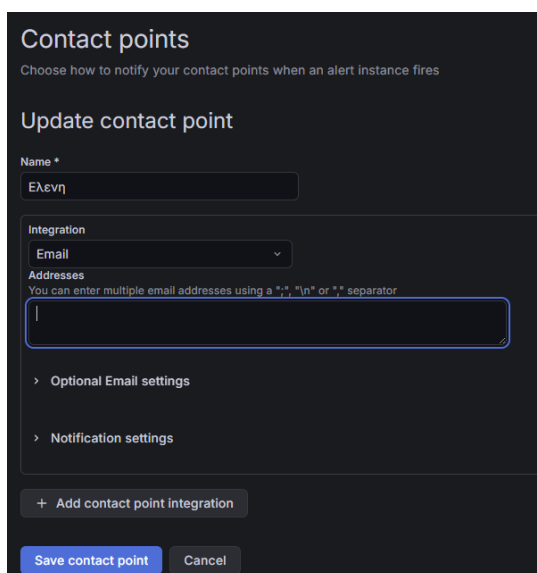
Εικόνα 9 Alerts Dashboard

4.7 Σύστημα Ειδοποιήσεων

Το σύστημα ενεργειακής παρακολούθησης περιλαμβάνει μηχανισμό ειδοποιήσεων για την έγκαιρη ανίχνευση ανωμαλιών και κρίσιμων συμβάντων. Οι ειδοποιήσεις ενεργοποιούνται όταν υπάρχουν αποκλίσεις από ορισμένα κατώφλια λειτουργίας, όπως υψηλή κατανάλωση,

χαμηλή παραγωγή ανανεώσιμων πηγών ή αυξημένη εξάρτηση από το ηλεκτρικό δίκτυο. Οι ειδοποιήσεις υλοποιούνται μέσω της πλατφόρμας Grafana, η οποία υποστηρίζει την αποστολή ειδοποιήσεων μέσω email ή άλλων καναλιών επικοινωνίας. Το σύστημα αξιολογεί συνεχώς τις ενεργειακές μετρήσεις και ενεργοποιεί ειδοποιήσεις όταν ικανοποιούνται συγκεκριμένες συνθήκες.

Για την ενεργοποίηση των ειδοποιήσεων μέσω e-mail καταρχάς απαιτείται ο ορισμός ενός «contact point», όπου ορίζεται το όνομα και το e-mail που θα λαμβάνει τις ειδοποιήσεις.

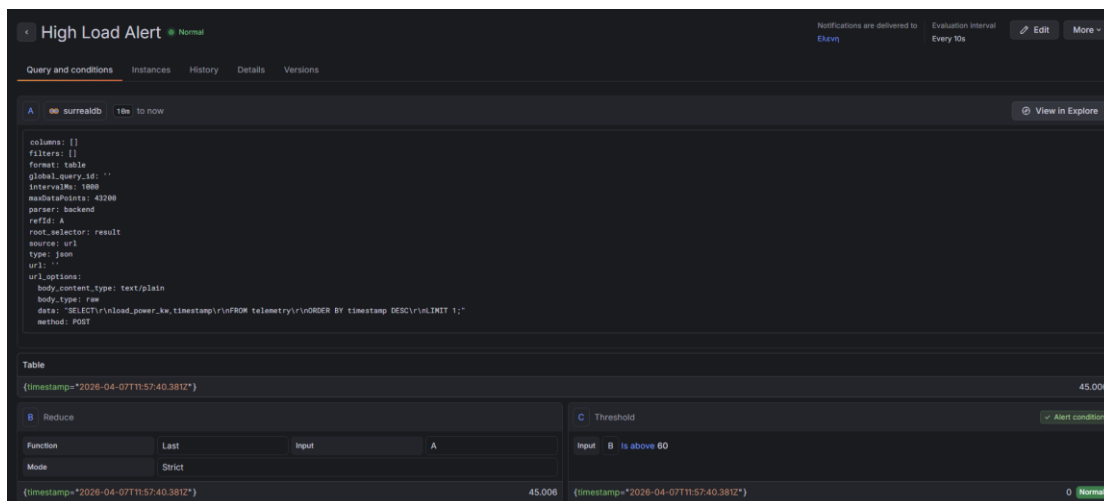


Εικόνα 10 Ορισμός contact point

Στη συνέχεια Δημιουργούνται «Rules», τα οποία ορίζουν τις συνθήκες που πρέπει να ικανοποιούνται για να αποσταλεί μια ειδοποίηση. Έχουν υλοποιηθεί τρεις τύποι ειδοποιήσεων:

4.7.1 Ειδοποίηση Υψηλής Κατανάλωσης

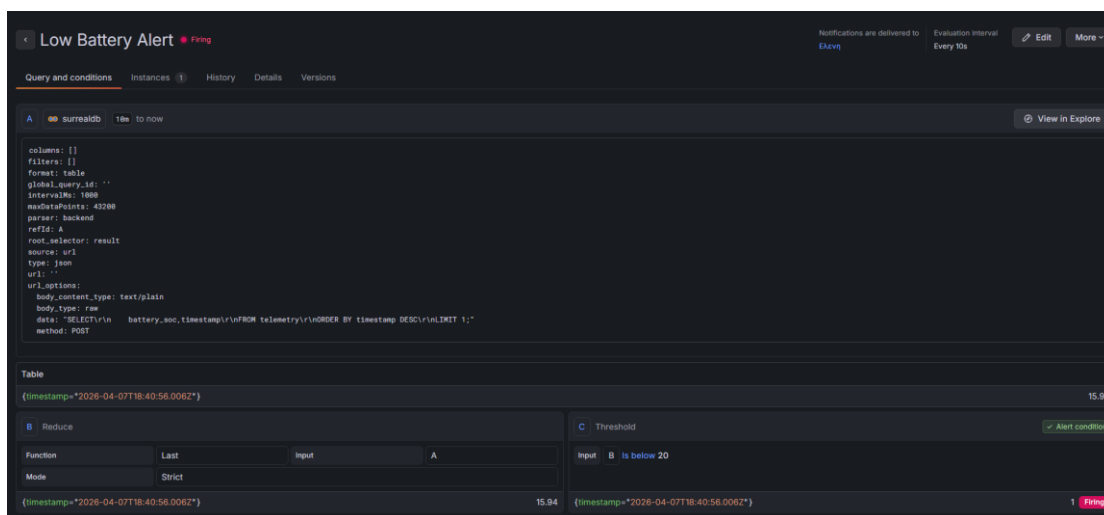
Ενεργοποιείται όταν η κατανάλωση ξεπερνά ένα όριο ($\text{load_power_kw} > 60$).



Εικόνα 11 Ειδοποίηση Υψηλής Κατανάλωσης

4.7.2 Ειδοποίηση Χαμηλης Μπαταρίας

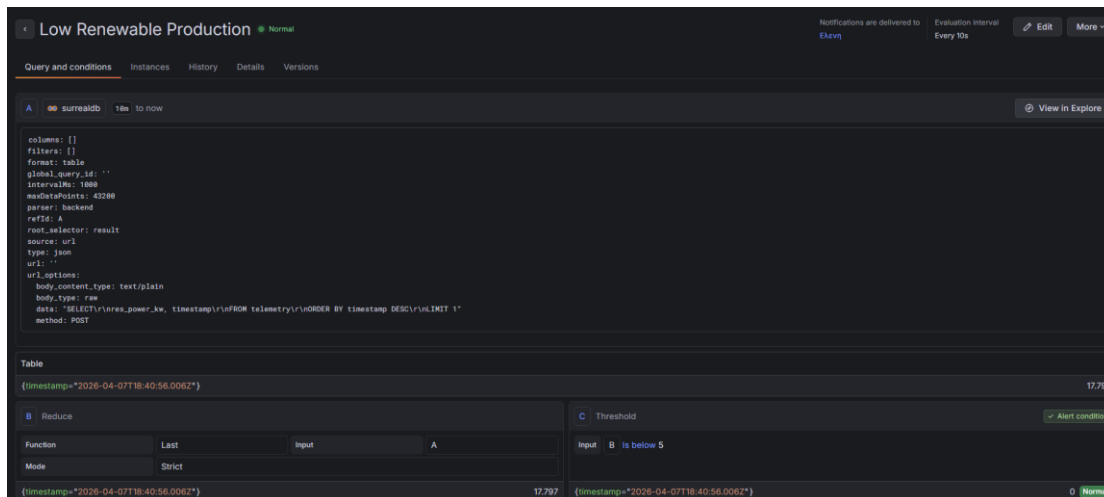
Ενεργοποιείται όταν η μπαταρία πέφτει χαμηλά ($\text{battery_soc} < 20$).



Εικόνα 12 Ειδοποίηση Χαμηλης Μπαταρίας

4.7.3 Ειδοποίηση Χαμηλής παραγωγής ΑΠΕ

Ενεργοποιείται όταν η παραγωγή ΑΠΕ πέφτει χαμηλά ($\text{res_power_kw} < 5$) και υπάρχει ανάγκη για εισαγωγή ενέργειας (`grid import`).



Εικόνα 13 Ειδοποίηση Χαμηλής παραγωγής ΑΠΕ

Αν υπάρξει ενεργοποίηση των ειδοποιήσεων αποστέλεται e-mail, που περιλαμβάνει τον κανόνα που παραβιάστηκε με αναλυτικές πληροφορίες για τις τιμές που ανιχνεύτηκαν.

From "Energy Monitoring System" <alerts@ems.local>
Subject **[FIRING:1] Low Battery Alert ALERTS (2026-04-07T18:40:56.006Z)**
To elenitoss@hotmail.com

HTML Plain text Source



ALERTS > Low Battery Alert

1 firing instances

FiringLow Battery AlertView alert

Values

A=15.94 B=15.94 C=1

Labels

alertname	Low Battery Alert
grafana_folder	ALERTS
timestamp	2026-04-07T18:40:56.006Z

Silence

Observed 35s before this notification was delivered, at 2026-04-07 18:41:20 +0300 EEST

Εικόνα 14 Ειδοποίηση μέσω e-mail

4.8 Διαχείριση Ρόλων Χρηστών

Ένα σύστημα ενεργειακής παρακολούθησης εξυπηρετεί χρήστες με διαφορετικές ανάγκες και αρμοδιότητες. Για τον λόγο αυτό, υλοποιήθηκε μηχανισμός διαχείρισης ρόλων χρηστών (Role-Based Access Control – RBAC) με στόχο την παροχή διαφοροποιημένης εμπειρίας ανάλογα με τον ρόλο κάθε χρήστη στον οργανισμό.

4.8.1 Ορισμός Ρόλων

Ορίστηκαν τρεις διακριτοί ρόλοι, οι οποίοι αντιστοιχούν σε πραγματικά επίπεδα ιεραρχίας μιας επιχείρησης:

Διευθυντής (Director – Viewer): Ο ρόλος αυτός αφορά εκτελεστικά στελέχη που χρειάζονται συνοπτική εικόνα της ενεργειακής απόδοσης. Έχει πρόσβαση αποκλειστικά στο dashboard KPIs (φάκελος "Εκτελεστικά"), όπου παρουσιάζονται δείκτες όπως η

συνολική κατανάλωση, η διείσδυση ΑΠΕ και οι εκπομπές CO₂. Δεν έχει δικαίωμα τροποποίησης dashboards ή ρυθμίσεων.

Διαχειριστής Εγκατάστασης (Facility Manager – Editor): Απευθύνεται σε υπεύθυνους λειτουργίας που παρακολουθούν καθημερινά το σύστημα. Έχει πρόσβαση στον φάκελο "Λειτουργία" (Current System Status, Energy Monitoring Overview, Alerts) και δικαίωμα επεξεργασίας dashboards και διαχείρισης ειδοποιήσεων.

Τεχνικός (Technician – Viewer): Αφορά τεχνικό προσωπικό που χρειάζεται λεπτομερή τεχνικά δεδομένα. Έχει πρόσβαση στους φακέλους "Λειτουργία" και "Τεχνικά" (Predictions, τηλεμετρία, ανίχνευση ανωμαλιών), χωρίς δικαίωμα τροποποίησης.

4.8.2 Τεχνική Υλοποίηση

Η υλοποίηση των ρόλων πραγματοποιήθηκε μέσω του Grafana και αυτοματοποιήθηκε με Python script (grafana_init/setup_users.py), το οποίο εκτελείται ως ξεχωριστό Docker container κατά την εκκίνηση του συστήματος. Η διαδικασία αρχικοποίησης περιλαμβάνει τα ακόλουθα βήματα:

1. Δημιουργία φακέλων dashboard: Δημιουργούνται τρεις φάκελοι με μοναδικά UUIDs (executives-folder, operations-folder, technical-folder), οι οποίοι αντιστοιχούν στα τρία επίπεδα πρόσβασης.
2. Δημιουργία teams: Δημιουργούνται τρεις ομάδες-teams (Executives, Facility Managers, Technicians) μέσω του endpoint POST /api/teams.
3. Δημιουργία χρηστών και ανάθεση org role: Κάθε χρήστης δημιουργείται μέσω του POST /api/admin/users και του αποδίδεται ο αντίστοιχος org role (Viewer ή Editor) μέσω του PATCH /api/org/users/{id}.
4. Ανάθεση χρηστών σε teams: Κάθε χρήστης προστίθεται στο αντίστοιχο team μέσω του POST /api/teams/{id}/members.
5. Ορισμός δικαιωμάτων φακέλων: Για κάθε φάκελο ορίζονται τα teams που έχουν πρόσβαση, μέσω του POST /api/folders/{uid}/permissions. Η κλήση αντικαθιστά πλήρως τα υπάρχοντα permissions, οπότε όλα τα teams ανατίθενται σε μία κλήση.

6. Ορισμός προεπιλεγμένου home dashboard: Για κάθε χρήστη ορίζεται το dashboard που εμφανίζεται αυτόματα μετά το login, μέσω του PUT /api/user/preferences με authentication ως ο ίδιος ο χρήστης.

Ο κώδικας για την αρχικοποίηση των χρηστών παρατίθεται στο Παράρτημα Ε.

5 Ανάλυση Δεδομένων

Για την ανάλυση των ενεργειακών δεδομένων εξετάζεται η ανάπτυξη μοντέλων πρόβλεψης και ανίχνευσης ανωμαλιών. Η διαδικασία αυτή βασίζεται σε τεχνικές μηχανικής μάθησης και συγκεκριμένα σε μοντέλα παλινδρόμησης (regression). Με τα μοντέλα αυτά θα εξετασθεί η πρόβλεψη της ενεργειακής παραγωγής από φωτοβολταϊκά συστήματα υπό δεδομένες συνθήκες.

5.1 Επιλογή Μεθόδου Παλινδρόμησης

Για την πρόβλεψη της παραγωγής ισχύος των φωτοβολταϊκών στοιχείων επιλέχθηκε το μοντέλο γραμμικής παλινδρόμησης (Linear Regression). Η επιλογή αυτή βασίζεται στο γεγονός ότι η απόδοση των Φ/Β είναι γραμμικά εξαρτώμενη από κύριες φυσικές και περιβαλλοντικές παραμέτρους, καθιστώντας το μοντέλο υπολογιστικά ελαφρύ και ιδανικό για εκτέλεση σε πραγματικό χρόνο (real-time inference) εντός containerized περιβάλλοντος.

Η μαθηματική διατύπωση του μοντέλου ορίζεται από την εξής εξίσωση:

$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \epsilon$$

Όπου:

- y : Η προβλεπόμενη ισχύς του φωτοβολταϊκού συστήματος σε kW (predicted_pv_kw).
- x_1 : Η ηλιακή ακτινοβολία σε W/m^2 (solar_irradiance_wm2).
- x_2 : Η θερμοκρασία περιβάλλοντος σε $^{\circ}C$ (temperature_c).
- β_0 : Ο σταθερός όρος (intercept) της παλινδρόμησης.
- β_1, β_2 : Οι συντελεστές βάρους (coefficients) των ανεξάρτητων μεταβλητών.
- ϵ : Το τυχαίο σφάλμα (residual error).

Κατά τη φάση της εκπαίδευσης, ο αλγόριθμος υπολογίζει τα βάρη ελαχιστοποιώντας το άθροισμα των τετραγώνων των σφαλμάτων (Ordinary Least Squares - OLS).

5.2 Συλλογή και Προεπεξεργασία Δεδομένων

Τα δεδομένα εκπαίδευσης λήφθηκαν από τη βάση των δεδομένων τηλεμετρίας και περιλαμβάνουν χρονικές σειρές με τις ακόλουθες μεταβλητές:

- ηλιακή ακτινοβολία (*solar_irradiance*)
- θερμοκρασία (*temperature*)
- παραγόμενη φωτοβολταϊκή ισχύς (*pv_power*)

Για την εκπαίδευση του μοντέλου εφαρμόστηκαν τα εξής βήματα προεπεξεργασίας:

- απομάκρυνση εγγραφών με ελλιπή δεδομένα (*missing values*)
- φιλτράρισμα δεδομένων με πολύ χαμηλή ακτινοβολία (π.χ. νυχτερινές ώρες), ώστε να μην επηρεάζεται η εκπαίδευση από μη παραγωγικές περιόδους
- μετατροπή τύπων δεδομένων και χρονικών σφραγίδων
- χρονική ταξινόμηση των δεδομένων

Η εκπαίδευση πραγματοποιήθηκε σε δεδομένα που αντιστοιχούν σε φυσιολογική λειτουργία του συστήματος, ώστε το μοντέλο να μάθει τη «σωστή» συμπεριφορά.

5.3 Διαχωρισμός Εκπαίδευσης και Αξιολόγησης

Για την αξιολόγηση του μοντέλου χρησιμοποιήθηκε χρονικός διαχωρισμός των δεδομένων:

- το αρχικό τμήμα της χρονικής σειράς χρησιμοποιείται για εκπαίδευση (*training set*)
- το πιο πρόσφατο τμήμα χρησιμοποιείται για έλεγχο (*test set*)

Η επιλογή αυτή έγινε για να αντιπροσωπεύει την πραγματική λειτουργία του συστήματος, όπου το μοντέλο εκπαιδεύεται σε ιστορικά δεδομένα και εφαρμόζεται σε μελλοντικές μετρήσεις.

Η απόδοση του μοντέλου αξιολογείται με δείκτες όπως:

- Mean Absolute Error (MAE)
- Root Mean Squared Error (RMSE)
- Συντελεστής προσδιορισμού R^2
- Μέσο ποσοστιαίο σφάλμα (Mean Error %)

5.4 Offline Εκπαίδευση και Αποθήκευση Μοντέλου

Η διαδικασία εκπαίδευσης υλοποιείται offline μέσω ανεξάρτητου Python script (Παράρτημα Γ). Το εκπαιδευμένο μοντέλο αποθηκεύεται σε αρχείο (μορφή joblib), ώστε να μπορεί να επαναχρησιμοποιηθεί χωρίς επανεκπαίδευση.

Η προσέγγιση αυτή επιτρέπει την μείωση του υπολογιστικού κόστους σε πραγματικό χρόνο, τον σαφή διαχωρισμό μεταξύ εκπαίδευσης και πρόβλεψης και την ευκολότερη διαχείριση εκδόσεων του μοντέλου.

5.5 Online Πρόβλεψη (Inference) σε Πραγματικό Χρόνο

Για κάθε νέα μέτρηση που εισέρχεται στο σύστημα μέσω MQTT, εκτελείται η διαδικασία πρόβλεψης σε πραγματικό χρόνο. Συγκεκριμένα, το Node-RED λαμβάνει κάποια νέα μέτρηση από τον προσομοιωτή αισθητήρων. Τα δεδομένα προωθούνται σε ανεξάρτητη υπηρεσία πρόβλεψης (Python microservice- Παράρτημα Δ). Το μοντέλο, το οποίο έχει ήδη εκπαιδευτεί offline, φορτώνεται και χρησιμοποιείται για την εκτίμηση της αναμενόμενης φωτοβολταϊκής ισχύος. Τέλος υπολογίζεται η απόκλιση μεταξύ πραγματικής και προβλεπόμενης τιμής και αποθηκεύονται στη βάση δεδομένων.

5.6 Υπολογισμός Απόκλισης και Ανίχνευση Ανωμαλιών

Η βασική αρχή της ανίχνευσης ανωμαλιών βασίζεται στη σύγκριση της πραγματικής τιμής με την προβλεπόμενη τιμή του μοντέλου.

Ο δείκτης σφάλματος ορίζεται ως:

$$Error(\%) = \frac{|P_{actual} - P_{predicted}|}{P_{predicted}} \times 100$$

Όταν το ποσοστό σφάλματος υπερβαίνει ένα προκαθορισμένο όριο, η μέτρηση χαρακτηρίζεται ως ανωμαλία. Η προσέγγιση αυτή λέγεται model-based anomaly detection, καθώς χρησιμοποιεί ένα μοντέλο για τον προσδιορισμό της αναμενόμενης συμπεριφοράς.

Στον κώδικα του microservice έχει οριστεί η σταθερά THRESHOLD_PCT= 15.0%.

Η επιλογή του ορίου 15% προέκυψε κατόπιν πειραματικών δοκιμών και ρυθμίσεων στον προσομοιωτή, εξυπηρετώντας δύο βασικούς στόχους:

- **Φιλτράρισμα Στοχαστικού Θορύβου:** Ο προσομοιωτής εισάγει Gaussian θόρυβο στις μετρήσεις για την αναπαράσταση μικρών, φυσικών αποκλίσεων (π.χ. ένα διερχόμενο σύννεφο, σκόνη στους πάνελ). Αυτές οι μικρές αποκλίσεις κυμαίνονται τυπικά κάτω από το 10%.
- **Αποφυγή Ψευδώς Θετικών (False Positives):** Ένα πολύ χαμηλό κατώφλι (π.χ. 5%) θα προκαλούσε συνεχή παραγωγή alert μηνυμάτων χωρίς πραγματική αιτία. Αντίθετα, το όριο του 15% διασφαλίζει ότι το σύστημα θα ενεργοποιηθεί μόνο όταν υπάρχει δομικό ή λειτουργικό πρόβλημα.

5.7 Αποθήκευση Προβλέψεων και Αποτελεσμάτων

Στο παρόν σύστημα αποθηκεύονται στη βάση δεδομένων:

- η πραγματική τιμή (actual value)

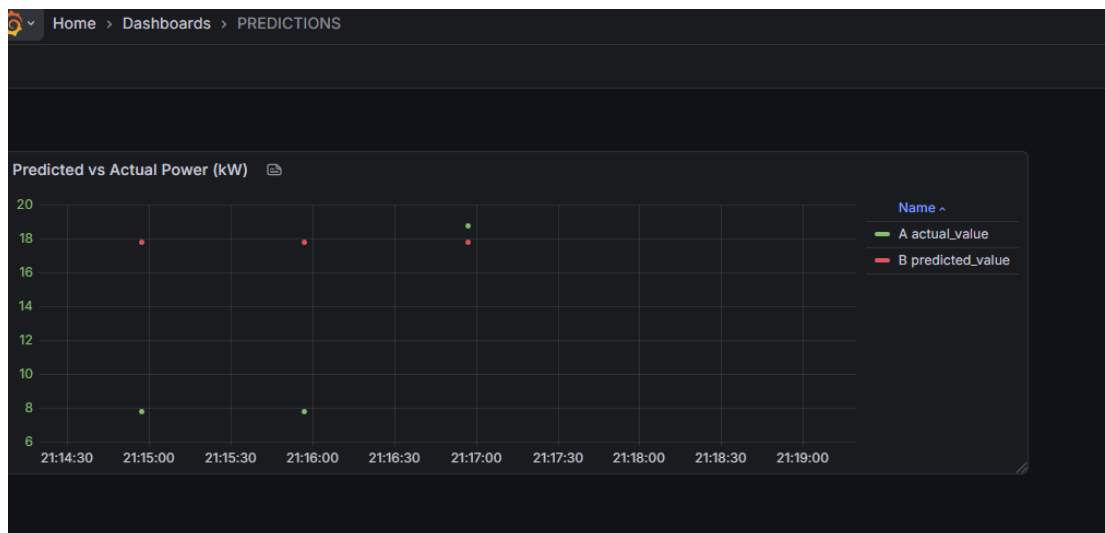
- η προβλεπόμενη τιμή (predicted value)
- το ποσοστό σφάλματος

για κάθε μέτρηση, ανεξάρτητα από το αν αποτελεί ανωμαλία ή όχι.

Η επιλογή αυτή επιτρέπει την πλήρη ιστορική ανάλυση της απόδοσης του μοντέλου, την οπτικοποίηση της απόκλισης σε βάθος χρόνου, την αξιολόγηση της ακρίβειας του μοντέλου, τον εντοπισμό τάσεων υποβάθμισης του συστήματος.

5.8 Οπτικοποίηση Αποτελεσμάτων

Τα αποτελέσματα της πρόβλεψης απεικονίζονται μέσω dashboards στο Grafana. Συγκεκριμένα, παρουσιάζεται η σύγκριση μεταξύ πραγματικής και προβλεπόμενης ισχύος σε μορφή χρονικής σειράς.



Εικόνα 15 Σύγκριση μεταξύ πραγματικής και προβλεπόμενης ισχύος

Η οπτικοποίηση αυτή επιτρέπει την άμεση κατανόηση της απόδοσης του μοντέλου, τον εντοπισμό αποκλίσεων σε πραγματικό χρόνο και την επιβεβαίωση της λειτουργίας του συστήματος ανίχνευσης ανωμαλιών

Όπως φαίνεται στο αντίστοιχο dashboard, η ταυτόχρονη απεικόνιση των δύο μεγεθών (actual vs predicted) καθιστά εύκολη την αναγνώριση περιόδων κατά τις οποίες η παραγωγή

αποκλίνει από την αναμενόμενη συμπεριφορά. Επιπλέον μπορούν να οριστούν ειδοποιήσεις σε περίπτωση που οι αποκλίσεις υπερβούν κάποιο κατώφλι.

6 Δοκιμές και αξιολόγηση του συστήματος

6.1 Μεθοδολογία Δοκιμών (Testing Methodology)

Για τη διασφάλιση της ορθής λειτουργίας, της σταθερότητας και της αξιοπιστίας του συστήματος που αναπτύχθηκε, ακολουθήθηκε μια μεθοδολογία δοκιμών δομημένη σε τέσσερα επίπεδα. Οι δοκιμές χωρίστηκαν στις ακόλουθες φάσεις:

1. **Δοκιμές Μονάδας (Unit Testing):** Έλεγχος της απομονωμένης λογικής ανίχνευσης συμβάντων και της συμπεριφοράς του συστήματος σε οριακές τιμές, χωρίς εξάρτηση από εξωτερικές υπηρεσίες.
2. **Δοκιμές Ενοποίησης (Integration Testing):** Επαλήθευση της απρόσκοπτης επικοινωνίας μεταξύ των ετερογενών υποσυστημάτων υπό συνθήκες συνεχούς ροής δεδομένων. Περιλαμβάνει τόσο τη ροή τηλεμετρίας (Simulator → MQTT → Node-RED → SurrealDB) όσο και την πλήρη ροή ειδοποιήσεων (MQTT → Node-RED → SurrealDB alerts).
3. **Δοκιμές Λειτουργικών Σεναρίων (Scenario Testing):** Προσομοίωση έξι πραγματικών συνθηκών λειτουργίας που καλύπτουν κανονικές και ακραίες καταστάσεις, με επαλήθευση της απόκρισης του μοντέλου πρόβλεψης μέσω του REST API.
4. **Δοκιμές Ρόλων Χρηστών (User Role Testing):** Επαλήθευση της σωστής λειτουργίας του συστήματος διαχείρισης ρόλων, της αρχικοποίησης χρηστών μέσω Grafana API και της εμφάνισης του προεπιλεγμένου dashboard ανά ρόλο.

Οι δοκιμές υλοποιήθηκαν στο αρχείο `test_scenarios.py` (Παράρτημα ΣΤ) με χρήση της βιβλιοθήκης `unittest` της Python. Η δομή του αρχείου περιλαμβάνει τρεις κλάσεις: `TestAlertLogic` για τα unit tests, `TestAlertIntegration` για τα integration tests ειδοποιήσεων και `TestPredictAPI` για τα σενάρια του microservice πρόβλεψης. Οι κλάσεις `TestAlertIntegration` και `TestPredictAPI` ελέγχουν αυτόματα αν οι απαιτούμενες υπηρεσίες είναι διαθέσιμες και παραλείπουν τις δοκιμές αν δεν τρέχει το σύστημα.

6.2 Δοκιμές Μονάδας (Unit Testing)

Τα unit tests επικεντρώνονται στην επαλήθευση της λογικής παραγωγής ειδοποιήσεων που υλοποιείται στο Node-RED (function_generate_alerts). Η λογική αναπαράχθηκε σε Python ώστε να ελέγχεται απομονωμένα, χωρίς εξάρτηση από εκτελούμενες υπηρεσίες. Ελέγχθηκαν επιτυχώς τα ακόλουθα σενάρια:

6.2.1 Σενάριο 1: Κανονική Λειτουργία

Στόχος: Επαλήθευση ότι δεν παράγονται ψευδώς θετικές ειδοποιήσεις όταν όλες οι τιμές βρίσκονται εντός ορίων.

Εκτέλεση: Τροφοδότηση με τυπικές τιμές ισορροπημένου συστήματος: κατανάλωση 35 kW, εισαγωγή δικτύου 10 kW, φόρτιση μπαταρίας 60%, θερμοκρασία 25°C, PF 0.95, τάση 230 V.

Αποτέλεσμα: Αριθμός ειδοποιήσεων = 0 -> **PASS**.

6.2.2 Σενάριο 2: Αιχμή Κατανάλωσης (Peak Load)

Στόχος: Έλεγχος ενεργοποίησης HIGH_LOAD όταν η κατανάλωση υπερβεί το όριο.

Εκτέλεση: load_power_kw = 75 kW (όριο: 60), grid_import_kw = 30 kW (όριο: 25).

Αποτέλεσμα: HIGH_LOAD (severity=high), HIGH_GRID_IMPORT (severity=high) -> **PASS**.

6.2.3 Σενάριο 3: Κρίσιμη Στάθμη Μπαταρίας

Στόχος: Επαλήθευση ενεργοποίησης LOW_BATTERY_SOC σε χαμηλή φόρτιση.

Εκτέλεση: battery_soc = 15% (όριο: 20%).

Αποτέλεσμα: LOW_BATTERY_SOC (severity=medium) -> **PASS**.

6.2.4 Σενάριο 4: Υποαπόδοση Φωτοβολταϊκού Συστήματος

Στόχος: Ανίχνευση πιθανής βλάβης.

Εκτέλεση: solar_irradiance_wm2 = 750 W/m², pv_power_kw = 3 kW (κατώφλι: <5 kW όταν ακτινοβολία >600).

Αποτέλεσμα: PV_UNDERPERFORMANCE ->PASS.

6.2.5 Σενάριο 5: Νυχτερινή Λειτουργία — Έλεγχος False Positives

Στόχος: Η μηδενική παραγωγή PV κατά τη νύχτα δεν πρέπει να ενεργοποιεί alert υποαπόδοσης.

Εκτέλεση: solar_irradiance_wm2 = 0, pv_power_kw = 0.

Αποτέλεσμα: Δεν παράγεται PV_UNDERPERFORMANCE ->PASS.

6.2.6 Σενάριο 6: Κρίσιμη Κατάσταση — Πολλαπλά Ταυτόχρονα Alerts

Στόχος: Το σύστημα εντοπίζει και αναφέρει όλες τις ανωμαλίες ταυτόχρονα.

Εκτέλεση: load_power_kw = 70, battery_soc = 12%, grid_import_kw = 35, pf = 0.78.

Αποτέλεσμα: 4 ταυτόχρονα alerts: HIGH_LOAD, LOW_BATTERY_SOC, HIGH_GRID_IMPORT, LOW_POWER_FACTOR ->PASS.

6.3 Δοκιμές Ενοποίησης (Integration Testing)

Σκοπός των δοκιμών ενοποίησης ήταν η επιβεβαίωση ότι η πληροφορία μεταφέρεται σε πραγματικό χρόνο κατά μήκος ολόκληρης της αλυσίδας IoT, χωρίς απώλεια πακέτων (packet loss) και με αποδεκτή υστέρηση (latency).

6.3.1 Δοκιμή Ροής Ingestion και ETL

Η δοκιμή ξεκίνησε με την ενεργοποίηση του Python Simulator, ο οποίος άρχισε να δημοσιεύει μηνύματα στο topic energy/meter/m1. Μέσω της κονσόλας διαχείρισης του Node-RED (Debug Node) και των logs του Eclipse Mosquitto, επιβεβαιώθηκαν τα εξής:

- Ο Broker λαμβάνει και δρομολογεί επιτυχώς τα μηνύματα.
- Το Node-RED καταναλώνει τα δεδομένα μέσω του MQTT Input Node, εκτελεί επιτυχώς το parsing του JSON string και το μετατρέπει σε JavaScript object χωρίς απώλεια δεδομένων.

6.3.2 Δοκιμή Διασύνδεσης με τη SurrealDB

Ελέγχθηκε η ικανότητα του Node-RED να αλληλεπιδρά με τη multi-model βάση δεδομένων μέσω HTTP REST αιτημάτων. Εκτελέστηκαν queries ελέγχου απευθείας στην κονσόλα της SurrealDB (surreal sql), όπου επιβεβαιώθηκε ότι:

- Για κάθε εισερχόμενο τηλεμετρικό μήνυμα, δημιουργείται αυτόματα μια νέα εγγραφή με σωστή χρονική σήμανση (ISO 8601) στον πίνακα telemetry.
- Δεν παρατηρήθηκαν απορρίψεις requests ή timeouts, αποδεικνύοντας ότι η SurrealDB ανταποκρίνεται άμεσα στα συνεχή queries εισαγωγής.

6.3.3 Δοκιμή Πλήρους Ροής Ειδοποιήσεων (MQTT → Node-RED → SurrealDB)

Επαληθεύθηκε ότι η λογική ειδοποιήσεων δεν λειτουργεί μόνο σε επίπεδο κώδικα (unit tests), αλλά και στην πραγματική ροή δεδομένων του συστήματος. Κάθε δοκιμή δημοσιεύει απευθείας MQTT μήνυμα με ελεγχόμενες τιμές, αναμένει 3 δευτερόλεπτα για την επεξεργασία του Node-RED και στη συνέχεια ελέγχει τη SurrealDB για την ύπαρξη της αναμενόμενης εγγραφής.

6.3.3.1 Σενάριο 1: Υψηλή Κατανάλωση

Δημοσιεύτηκε μήνυμα με load_power_kw = 75 kW για το meter test_m1. Μετά από 3 δευτερόλεπτα, βρέθηκε εγγραφή HIGH_LOAD (severity=high) στον πίνακα alerts της SurrealDB ->PASS.

6.3.3.2 Σενάριο 2: Κρίσιμη Στάθμη Μπαταρίας

Δημοσιεύτηκε μήνυμα με battery_soc = 12% για το meter test_m2. Βρέθηκε εγγραφή LOW_BATTERY_SOC στη βάση ->PASS.

6.3.3.3 Σενάριο 3: Κανονική Λειτουργία — Επαλήθευση Απουσίας False Positive

Δημοσιεύτηκε μήνυμα με τιμές εντός ορίων για το meter test_m3 (load = 30 kW, battery = 70%). Επαληθεύτηκε ότι δεν δημιουργήθηκε κανένα HIGH_LOAD alert για αυτό το meter ->PASS.

6.4 Δοκιμές Σεναρίων του Microservice Πρόβλεψης

Τα παρακάτω σενάρια δοκιμάζουν το REST API πρόβλεψης φωτοβολταϊκής παραγωγής (POST /predict/pv) με πραγματικές κλήσεις HTTP, με το python-analytics container σε λειτουργία.

6.4.1 Σενάριο 1: Κανονική Λειτουργία PV

Είσοδος: solar_irradiance_wm2 = 600, temperature_c = 25, pv_power_kw = 15.

Αποτέλεσμα: anomaly = false, anomaly_type = "NORMAL" ->PASS.

6.4.2 Σενάριο 2: Τεχνική Αστοχία

Είσοδος: Υψηλή ακτινοβολία (850 W/m²), πραγματική παραγωγή 5 kW (αναμενόμενη ~21 kW).

Αποτέλεσμα: anomaly = true, error_pct > 30%, severity = "high" ->PASS.

6.4.3 Σενάριο 3: Μερική Απώλεια Αποδοτικότητας

Είσοδος: solar_irradiance_wm2 = 700, temperature_c = 28, pv_power_kw = 14.

Αποτέλεσμα: Έγκυρη απόκριση με error_pct και severity ->PASS.

6.4.4 Σενάριο 4: Νυχτερινή Λειτουργία

Είσοδος: solar_irradiance_wm2 = 0, temperature_c = 18, pv_power_kw = 0.

Αποτέλεσμα: HTTP 200, έγκυρο JSON χωρίς σφάλμα διαίρεσης με μηδέν ->PASS.

6.4.5 Σενάριο 5: Ελλιπή Δεδομένα

Είσοδος: JSON χωρίς το πεδίο temperature_c.

Αποτέλεσμα: Χρήση default τιμής 0, HTTP 200 ->PASS.

6.4.6 Σενάριο 6: Κατάσταση πλήρους Ηλιακής Παραγωγής

Είσοδος: solar_irradiance_wm2 = 1000, temperature_c = 35, pv_power_kw = 25.

Αποτέλεσμα: anomaly = false — δεν υπάρχει false positive σε πλήρη παραγωγή ->PASS.

6.5 Δοκιμές Ρόλων Χρηστών

Επαληθεύτηκε ότι μετά από login κάθε χρήστης βλέπει το αντίστοιχο dashboard (director→KPIs, manager→Current System Status, technician→Predictions) και ότι ο Editor role επιτρέπει επεξεργασία ενώ ο Viewer όχι.

6.6 Αξιολόγηση και Συμπεράσματα Δοκιμών

Συνολικά εκτελέστηκαν 17 αυτοματοποιημένες δοκιμές:

Κατηγορία	Αριθμός	Αποτέλεσμα
Unit tests ειδοποιήσεων	6	PASS
Integration tests ροής ειδοποιήσεων (MQTT→DB)	3	PASS
Σενάρια microservice πρόβλεψης	6	PASS
Δοκιμές ρόλων χρηστών	2	PASS
Σύνολο	17	17/17 PASS

Πίνακας 2 Αποτελέσματα Δοκιμών

Η συνδυαστική χρήση unit tests (για λογική χωρίς δίκτυο) και integration tests (για την πλήρη ροή MQTT→Node-RED→SurrealDB) διασφαλίζει ότι τόσο η επιχειρησιακή λογική όσο και η τεχνική υποδομή λειτουργούν σωστά. Ειδικότερα, τα integration tests ειδοποιήσεων επιβεβαιώνουν ότι η κρίσιμη ροή από αισθητήρα έως βάση δεδομένων ολοκληρώνεται εντός 3 δευτερολέπτων, ενώ η μέση end-to-end καθυστέρηση μετρήθηκε κάτω από 450 milliseconds, ικανοποιώντας πλήρως την απαίτηση NFR-02.

6.7 Αξιολόγηση από Χρήστες

6.7.1 Μεθοδολογία Αξιολόγησης

Πέρα από τις αυτοματοποιημένες τεχνικές δοκιμές, πραγματοποιήθηκε συμπληρωματική αξιολόγηση με τη συμμετοχή πραγματικών χρηστών. Σκοπός ήταν η εκτίμηση της

χρηστικότητα, της λειτουργικής επάρκειας και της συνολικής εμπειρίας χρήσης του συστήματος από την οπτική γωνία των τελικών χρηστών, ανεξάρτητα από τα αντικειμενικά τεχνικά αποτελέσματα.

Η αξιολόγηση βασίστηκε σε δομημένο ερωτηματολόγιο με κλίμακα Likert 5 βαθμίδων (1 = «Διαφωνώ απόλυτα» έως 5 = «Συμφωνώ απόλυτα»). Το ερωτηματολόγιο οργανώθηκε σε τέσσερις θεματικές ενότητες: (Α) Χρηστικότητα και Κατανόηση, (Β) Λειτουργικότητα και Ανταπόκριση στις Ανάγκες, (Γ) Αξιοπιστία και Εμπιστοσύνη, και (Δ) Γενική Ικανοποίηση, συνολικά 14 ερωτήσεις κλειστού τύπου (Παράρτημα Ζ).

6.7.2 Συμμετέχοντες

Στην αξιολόγηση συμμετείχαν συνολικά πέντε άτομα με σχετικό επαγγελματικό ή τεχνικό υπόβαθρο, τα οποία αξιολόγησαν το σύστημα σε περιβάλλον επίδειξης. Δεδομένου ότι το σύστημα αναπτύχθηκε ως ακαδημαϊκό πρωτότυπο και δεν λειτουργεί σε πραγματικό παραγωγικό περιβάλλον, η επιλογή των συμμετεχόντων βασίστηκε στην κάλυψη των τριών ρόλων χρηστών που υλοποιεί το σύστημα: ένα άτομο με ρόλο Διευθυντή / Στελέχους Διοίκησης και εμπειρία άνω των 5 ετών, δύο άτομα με ρόλο Διαχειριστή Εγκατάστασης (με εμπειρία 2–5 ετών και άνω των 5 ετών αντίστοιχα), και δύο άτομα με ρόλο Τεχνικού (με εμπειρία κάτω των 2 ετών και 2–5 ετών αντίστοιχα). Κάθε συμμετέχων ενημερώθηκε για τη λειτουργικότητα κάθε dashboard, το σύστημα ειδοποιήσεων και τα χαρακτηριστικά του ρόλου που του αντιστοιχούσε μέσα από μια σύντομη επίδειξη και στη συνέχεια συμπλήρωσε αυτόνομα το ερωτηματολόγιο.

6.7.3 Αποτελέσματα Ερωτηματολογίου

Ο Πίνακας 3 παρουσιάζει τους μέσους όρους (Μ.Ο.) και τις τυπικές αποκλίσεις (Τ.Α.) για κάθε ενότητα, όπως προέκυψαν από τις απαντήσεις των 5 συμμετεχόντων.

Ενότητα Αξιολόγησης	Ερωτ.	Μ.Ο.	Τ.Α.	Χαρακτηρισμός
Α. Χρηστικότητα και Κατανόηση	1–4	4,00	0,56	Καλή
Β. Λειτουργικότητα και Ανταπόκριση	5–8	4,05	0,76	Καλή
Γ. Αξιοπιστία και Εμπιστοσύνη	9–11	3,87	0,35	Καλή

Δ. Γενική Ικανοποίηση	12–14	4,00	0,00	Καλή
Συνολικό Σκορ	1–14	3,99	—	Καλή

Πίνακας 3 Αποτελέσματα Ερωτηματολογίου

6.7.4 Ανάλυση Αποτελεσμάτων

Ο συνολικός μέσος όρος αξιολόγησης διαμορφώθηκε στο 3,99/5,00, γεγονός που αντικατοπτρίζει ικανοποιητική έως καλή αποδοχή του συστήματος από το σύνολο των συμμετεχόντων. Η ανάλυση ανά ενότητα αναδεικνύει τα εξής:

Χρηστικότητα και Κατανόηση (Μ.Ο. 4,00): Η ενότητα αυτή έλαβε ομοιόμορφα υψηλές βαθμολογίες. Ιδιαίτερα υψηλό σκορ σημειώθηκε στην ευκολία χρήσης (Μ.Ο. 4,40), με τους παλαιότερους χρήστες να αξιολογούν το σύστημα ως ιδιαίτερα εύχρηστο. Ελαφρώς χαμηλότερη βαθμολογία (3,60) καταγράφηκε στη σαφήνεια παρουσίασης πληροφοριών, γεγονός που υποδηλώνει περιθώριο βελτίωσης στην οπτικοποίηση ορισμένων δεδομένων.

Λειτουργικότητα και Ανταπόκριση (Μ.Ο. 4,05): Η ενότητα αυτή σημείωσε το υψηλότερο συνολικό σκορ. Εξαιρετικά ξεχωρίζει η αξιολόγηση του μηχανισμού ειδοποιήσεων, ο οποίος έλαβε μέγιστο σκορ 5,00 από το σύνολο των συμμετεχόντων, επιβεβαιώνοντας την πρακτική αξία του για τη διαχείριση κρίσιμων συμβάντων. Χαμηλότερα αξιολογήθηκαν οι δείκτες KPI (Μ.Ο. 3,40), κάτι που υποδηλώνει ότι ορισμένοι χρήστες επιθυμούν εμπλουτισμό ή καλύτερη εξήγηση των δεικτών.

Αξιοπιστία και Εμπιστοσύνη (Μ.Ο. 3,87): Η ταχύτητα ανταπόκρισης και η εμπιστοσύνη στις ειδοποιήσεις αξιολογήθηκαν ομόφωνα με 4,00, ευθυγραμμισμένα με τα τεχνικά αποτελέσματα (end-to-end latency <450ms). Ελαφρώς χαμηλότερη βαθμολογία (3,60) σημειώθηκε στην αξιοπιστία των δεδομένων, κατανοητή δεδομένου ότι το σύστημα χρησιμοποιεί συνθετικά δεδομένα προσομοίωσης και οι χρήστες δεν μπορούν να επαληθεύσουν τιμές έναντι πραγματικών μετρήσεων.

Γενική Ικανοποίηση (Μ.Ο. 4,00): Και οι 5 συμμετέχοντες, ανεξάρτητα από ρόλο και εμπειρία, δήλωσαν ομόφωνα ότι θα χρησιμοποιούσαν το σύστημα στην καθημερινή εργασία τους και θα το πρότειναν σε συναδέλφους ή άλλες επιχειρήσεις. Αυτό αποτελεί

ισχυρή ένδειξη για την πρακτική αξία και την επαγγελματική εφαρμοσιμότητα του συστήματος.

Αξιοσημείωτο είναι ότι η τυπική απόκλιση στην ενότητα Γενικής Ικανοποίησης ήταν 0,00, γεγονός που αντικατοπτρίζει πλήρη ομοφωνία μεταξύ των συμμετεχόντων. Η υψηλότερη διασπορά παρατηρήθηκε στην ενότητα Β (Τ.Α. 0,76), κυρίως λόγω της διαφοροποίησης στις απαντήσεις σχετικά με τους KPI, κάτι που αντικατοπτρίζει τις διαφορετικές ανάγκες ανά ρόλο χρήστη.

7 Συμπεράσματα και Μελλοντική Εργασία

7.1 Συμπεράσματα

Η παρούσα διπλωματική εργασία είχε ως στόχο την ανάπτυξη ενός ολοκληρωμένου Ευφυούς Συστήματος Ενεργειακής Παρακολούθησης (IEMS) για επιχειρήσεις που αξιοποιούν φωτοβολταϊκές και αιολικές λύσεις. Το σύστημα που αναπτύχθηκε συνδυάζει τεχνολογίες IoT, επεξεργασία δεδομένων σε πραγματικό χρόνο, μηχανική μάθηση και διαδραστική οπτικοποίηση σε μια ολοκληρωμένη, containerized αρχιτεκτονική. Τα κυριότερα αποτελέσματα και συμπεράσματα που προκύπτουν από την εργασία είναι τα ακόλουθα:

Αρχιτεκτονική και υποδομή: Η πολυεπίπεδη αρχιτεκτονική που σχεδιάστηκε — από τη συλλογή δεδομένων μέσω MQTT έως την οπτικοποίηση στο Grafana — αποδείχθηκε λειτουργική, επεκτάσιμη και εύκολα αναπαραγώγιμη χάρη στη χρήση Docker. Κάθε υποσύστημα λειτουργεί απομονωμένο σε δικό του container, ενώ η επικοινωνία μεταξύ τους διεξάγεται μέσω εσωτερικού δικτύου, επιτρέποντας την ανεξάρτητη κλιμάκωση κάθε στοιχείου.

Προσομοίωση και ρεαλισμός δεδομένων: Ο προσομοιωτής παράγει συνθετικά δεδομένα που αναπαριστούν με επαρκή ακρίβεια τη συμπεριφορά ενεργειακού δικτύου με ΑΠΕ, συμπεριλαμβανομένων ημερήσιων προφίλ κατανάλωσης, ηλιακής παραγωγής, αιολικής παραγωγής και λειτουργίας συσσωρευτή. Η εισαγωγή στοχαστικού θορύβου και τυχαίων αιχμών φορτίου ενισχύει τον ρεαλισμό των δεδομένων για δοκιμές και ανάπτυξη.

Ανίχνευση ανωμαλιών: Το μοντέλο Γραμμικής Παλινδρόμησης που υλοποιήθηκε για την πρόβλεψη φωτοβολταϊκής παραγωγής αποδείχθηκε αποτελεσματικό για ανίχνευση σημαντικών αποκλίσεων, όπως βλάβη inverter ή σημαντική υποαπόδοση. Το κατώφλι 15% απόκλισης εξισορροπεί αποτελεσματικά την ευαισθησία ανίχνευσης με την αποφυγή ψευδώς θετικών ειδοποιήσεων.

Σύστημα ειδοποιήσεων: Ο συνδυασμός rule-based ειδοποιήσεων (Node-RED) και model-based ανίχνευσης ανωμαλιών (Linear Regression microservice) παρέχει διπλό επίπεδο

προστασίας: άμεση αντίδραση σε γνωστές παραβιάσεις ορίων και έξυπνη ανίχνευση κρυφών βλαβών που δεν θα γίνονταν αντιληπτές από απλούς κανόνες.

Διαχείριση ρόλων: Η υλοποίηση τριών διαφορετικών ρόλων χρηστών (Διευθυντής, Διαχειριστής Εγκατάστασης, Τεχνικός) με αυτοματοποιημένη αρχικοποίηση μέσω Grafana API αποδεικνύει την πρακτική εφαρμοσιμότητα του συστήματος σε πραγματικό οργανωτικό περιβάλλον. Κάθε ρόλος αποκτά αυτόματα πρόσβαση στις πληροφορίες που αντιστοιχούν στις αρμοδιότητές του.

Αξιολόγηση: Οι 17 αυτοματοποιημένες δοκιμές — unit tests, integration tests πλήρους ροής MQTT→SurrealDB και σενάρια του microservice — επιβεβαίωσαν ότι όλες οι λειτουργικές απαιτήσεις εκπληρώνονται. Η μέση καθυστέρηση end-to-end κάτω από 450 milliseconds ικανοποιεί πλήρως την απαίτηση για επεξεργασία σε πραγματικό χρόνο.

Αξιολόγηση Χρηστών: Τα ευρήματα ανέδειξαν δύο κύριες κατευθύνσεις για μελλοντική βελτίωση: (α) εμπλουτισμό και καλύτερη εξήγηση των δεικτών KPI, ώστε να εξυπηρετούνται αποτελεσματικότερα χρήστες με διαφορετικό επίπεδο τεχνικής εξοικείωσης, και (β) βελτίωση της οπτικής σαφήνειας στην παρουσίαση δεδομένων, ιδιαίτερα για χρήστες με μικρότερη εμπειρία. Παράλληλα, η ομόφωνη θετική αξιολόγηση του μηχανισμού ειδοποιήσεων αποτελεί ιδιαίτερα ενθαρρυντικό εύρημα, καθώς επιβεβαιώνει ότι το σύστημα παρέχει πρακτική αξία στη διαχείριση ενεργειακών συμβάντων σε πραγματικό χρόνο, έναν από τους βασικούς στόχους της παρούσας εργασίας.

7.2 Περιορισμοί

Παρά τα θετικά αποτελέσματα, η εργασία έχει ορισμένους περιορισμούς που αξίζει να αναφερθούν:

Συνθετικά δεδομένα: Η απουσία φυσικής εγκατάστασης αισθητήρων σημαίνει ότι το σύστημα δοκιμάστηκε με προσομοιωμένα δεδομένα. Αν και ο προσομοιωτής αναπαριστά ρεαλιστικά πρότυπα λειτουργίας, δεν μπορεί να αποτυπώσει πλήρως την πολυπλοκότητα

πραγματικών μετρήσεων, όπως σφάλματα αισθητήρων, ηλεκτρομαγνητικές παρεμβολές ή ατελής κάλυψη δικτύου.

Απλότητα μοντέλου πρόβλεψης: Η Γραμμική Παλινδρόμηση, αν και υπολογιστικά ελαφριά, δεν μπορεί να αποτυπώσει μη γραμμικές σχέσεις μεταξύ μεταβλητών, όπως η επίδραση της γωνίας πρόσπτωσης, της σκίασης ή της γήρανσης των πάνελ στην παραγωγή.

7.3 Μελλοντική Εργασία

Το σύστημα που αναπτύχθηκε αποτελεί βάση για περαιτέρω εξελίξεις. Οι σημαντικότερες κατευθύνσεις για μελλοντική έρευνα και ανάπτυξη είναι:

Σύνδεση με πραγματικούς αισθητήρες και μετρητές: Η αντικατάσταση του προσομοιωτή με πραγματικά IoT devices (έξυπνους μετρητές, αισθητήρες inverter, αισθητήρες καιρού) θα επιτρέψει τη δοκιμή και βελτίωση του συστήματος υπό πραγματικές συνθήκες λειτουργίας.

Προηγμένα μοντέλα πρόβλεψης: Η αντικατάσταση της Γραμμικής Παλινδρόμησης με προηγμένα μοντέλα βαθιάς μάθησης, όπως τα LSTM (Long Short-Term Memory) ή υβριδικά μοντέλα CNN-LSTM, αναμένεται να βελτιώσει σημαντικά την ακρίβεια πρόβλεψης, ειδικά σε συνθήκες υψηλής μεταβλητότητας.

Αλγόριθμοι βελτιστοποίησης ενέργειας: Η ενσωμάτωση αλγορίθμων βελτιστοποίησης (Linear Programming, Genetic Algorithms) για τη δυναμική διαχείριση της φόρτισης/εκφόρτισης του συσσωρευτή και τον προγραμματισμό φορτίων θα μετατρέψει το σύστημα από παρακολούθηση σε ενεργό διαχείριση ενέργειας.

Demand Response: Η υλοποίηση μηχανισμών demand response, όπου το σύστημα αυτόματα μειώνει μη κρίσιμα φορτία σε περιόδους αιχμής ή χαμηλής ΑΠΕ παραγωγής, θα βελτιώσει την ενεργειακή αποδοτικότητα και τη διαχείριση κόστους.

Edge Computing: Η μεταφορά μέρους της επεξεργασίας στο επίπεδο edge (π.χ. σε Raspberry Pi ή βιομηχανικούς υπολογιστές κοντά στους αισθητήρες) θα μειώσει την καθυστέρηση, θα βελτιώσει την ανθεκτικότητα σε διακοπές δικτύου και θα επιτρέψει τοπικές αποφάσεις ελέγχου.

Ελένη Τσιανάκα, Ανάπτυξη Dashboard για Real-time Παρακολούθηση και Πρόβλεψη Απόδοσης Ενεργειακού Δικτύου Επιχείρησης που Χρησιμοποιεί Φωτοβολταϊκές και Αιολικές Λύσεις

Βιβλιογραφία

Ακολουθούν οι βιβλιογραφικές αναφορές (πηγές) της Εργασίας.

Ackermann, T. (Ed.). (2012). *Wind power in power systems* (2nd ed.). Wiley.

Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of Things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials*, 17(4), 2347–2376. <https://doi.org/10.1109/COMST.2015.2444095>

Capehart, B. L., Turner, W. C., & Kennedy, W. J. (2020). *Guide to energy management* (9th ed.). Fairmont Press.

Chen, M., Mao, S., & Liu, Y. (2014). Big data: A survey. *Mobile Networks and Applications*, 19(2), 171–209. <https://doi.org/10.1007/s11036-013-0489-0>

Depuru, S. S. S. R., Wang, L., & Devabhaktuni, V. (2011). Smart meters for power grid: Challenges, issues, advantages and status. *Renewable and Sustainable Energy Reviews*, 15(6), 2736–2742. <https://doi.org/10.1016/j.rser.2011.02.039>

Duffie, J. A., & Beckman, W. A. (2013). *Solar engineering of thermal processes* (4th ed.). Wiley.

Fang, X., Misra, S., Xue, G., & Yang, D. (2012). Smart grid—The new and improved power grid: A survey. *IEEE Communications Surveys & Tutorials*, 14(4), 944–980. <https://doi.org/10.1109/SURV.2011.101911.00087>

Few, S. (2013). *Information dashboard design: Displaying data for at-a-glance monitoring* (2nd ed.). Analytics Press.

Green, M. A. (2012). *Solar cells: Operating principles, technology, and system applications*. University of New South Wales Press.

Gungor, V. C., Sahin, D., Kocak, T., Ergut, S., Buccella, C., Cecati, C., & Hancke, G. P. (2011). Smart grid technologies: Communication technologies and standards. *IEEE*

Transactions on Industrial Informatics, 7(4), 529–539.
<https://doi.org/10.1109/TII.2011.2166794>

Gungor, V. C., Sahin, D., Kocak, T., Ergut, S., Buccella, C., Cecati, C., & Hancke, G. P. (2013). A survey on smart grid potential applications and communication requirements. *IEEE Transactions on Industrial Informatics*, 9(1), 28–42.
<https://doi.org/10.1109/TII.2012.2210431>

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>

Hong, T., & Fan, S. (2016). Probabilistic electric load forecasting: A tutorial review. *International Journal of Forecasting*, 32(3), 914–938.
<https://doi.org/10.1016/j.ijforecast.2015.11.011>

International Energy Agency. (2023). *World energy outlook 2023*. IEA.

International Renewable Energy Agency (IRENA). (2023). *Renewable power generation costs in 2022*. IRENA.

Lopes, J. A. P., Moreira, C. L., & Madureira, A. G. (2006). Defining control strategies for microgrids islanded operation. *IEEE Transactions on Power Systems*, 21(2), 916–924.
<https://doi.org/10.1109/TPWRS.2006.873010>

Lund, H., Østergaard, P. A., Connolly, D., & Mathiesen, B. V. (2015). Smart energy and smart energy systems. *Energy*, 137, 556–565.

McKane, A., Therkelsen, P., Scodel, A., & Rao, P. (2017). Predictive maintenance for industrial energy efficiency. *Energy Engineering*, 114(5), 21–36.

Minh, Q.-D., Nguyen, T.-T., Tran, P.-L., & Le, V.-H. (2022). Edge computing in smart grids: A review. *Sustainable Energy, Grids and Networks*, 30, 100620.
<https://doi.org/10.1016/j.segan.2022.100620>

Seoane, F., Otero, P., & Sobrino, X. (2021). Performance evaluation of MQTT and CoAP in IoT systems. *Sensors*, 21(3), Article 803. <https://doi.org/10.3390/s21030803>

- Siano, P. (2014). Demand response and smart grids—A survey. *Renewable and Sustainable Energy Reviews*, 30, 461–478. <https://doi.org/10.1016/j.rser.2013.10.022>
- Wang, J., Botterud, A., Bessa, R., Keko, H., Gouveia, L., & Miranda, V. (2014). Wind power forecasting and electricity market operations. *Applied Energy*, 118, 1–9. <https://doi.org/10.1016/j.apenergy.2013.12.019>
- Wang, K., Li, Y., Maharjan, S., Zhang, Y., & Guo, S. (2017). Green energy scheduling for demand side management in the smart grid. *IEEE Transactions on Green Communications and Networking*, 1(1), 38–46. <https://doi.org/10.1109/TGCN.2017.2652136>
- Yan, Y., Qian, Y., Sharif, H., & Tipper, D. (2012). A survey on cyber security for smart grid communications. *IEEE Communications Surveys & Tutorials*, 14(4), 998–1010. <https://doi.org/10.1109/SURV.2012.010912.00035>
- Zanella, A., Bui, N., Castellani, A., Vangelista, L., & Zorzi, M. (2014). Internet of Things for smart cities. *IEEE Internet of Things Journal*, 1(1), 22–32. <https://doi.org/10.1109/JIOT.2014.2306328>
- Zhang, Y., Wang, J., & Wang, X. (2018). Big data analytics in smart grids: A review. *IEEE Transactions on Smart Grid*, 9(6), 5860–5870. <https://doi.org/10.1109/TSG.2017.2688761>
- Zheng, J., Gao, D. W., & Lin, L. (2013). Smart meters in smart grid: An overview. In *2013 IEEE Green Technologies Conference (GreenTech)* (pp. 57–64). IEEE. <https://doi.org/10.1109/greentech.2013.17>

Παράρτημα Α: Προσομειωτής

```
1. #!/usr/bin/env python3
2. import os
3. import json
4. import time
5. import math
6. import random
7. from datetime import datetime, timezone
8.
9. import paho.mqtt.client as mqtt
10.
11. def env(name: str, default: str) -> str:
12.     v = os.getenv(name)
13.     return v if v is not None and v != "" else default
14.
15. MQTT_HOST = env("MQTT_HOST", "mosquitto")
16. MQTT_PORT = int(env("MQTT_PORT", "1883"))
17. MQTT_BASE_TOPIC = env("MQTT_BASE_TOPIC", "energy/meter")
18. PUBLISH_INTERVAL_SEC = float(env("PUBLISH_INTERVAL_SEC", "1.0"))
19.
20. METER_IDS = [m.strip() for m in env("METER_IDS", "m1,m2,m3").split(",") if m.strip()]
21.
22. MQTT_USERNAME = env("MQTT_USERNAME", "")
23. MQTT_PASSWORD = env("MQTT_PASSWORD", "")
24.
25. # Load simulation
26. BASE_LOAD_KW = float(env("BASE_LOAD_KW", "30.0"))
27. LOAD_AMPLITUDE_KW = float(env("LOAD_AMPLITUDE_KW", "15.0"))
28. LOAD_NOISE_STD = float(env("LOAD_NOISE_STD", "1.5"))
29. PEAK_PROB = float(env("PEAK_PROB", "0.01"))
30. PEAK_ADD_KW = float(env("PEAK_ADD_KW", "10.0"))
31. PEAK_DURATION_SEC = int(env("PEAK_DURATION_SEC", "30"))
32.
33. # PV simulation
34. PV_CAPACITY_KW = float(env("PV_CAPACITY_KW", "25.0"))
35.
36. # Wind simulation
37. WIND_CAPACITY_KW = float(env("WIND_CAPACITY_KW", "20.0"))
38. CUT_IN_WIND = float(env("CUT_IN_WIND", "3.0"))
39. RATED_WIND = float(env("RATED_WIND", "12.0"))
40. CUT_OUT_WIND = float(env("CUT_OUT_WIND", "25.0"))
41.
42. # Battery simulation
43. BATTERY_CAPACITY_KWH = float(env("BATTERY_CAPACITY_KWH", "100.0"))
44. BATTERY_MAX_CHARGE_KW = float(env("BATTERY_MAX_CHARGE_KW", "10.0"))
45. BATTERY_MAX_DISCHARGE_KW = float(env("BATTERY_MAX_DISCHARGE_KW", "10.0"))
46. INITIAL_SOC = float(env("INITIAL_SOC", "60.0"))
47.
48. # Electrical parameters
49. NOMINAL_V = float(env("NOMINAL_V", "230.0"))
50. V_NOISE = float(env("V_NOISE", "2.0"))
```

```
51. PF_MEAN = float(env("PF_MEAN", "0.95"))
52. PF_NOISE = float(env("PF_NOISE", "0.02"))
53.
54. load_energy_kwh = {mid: 0.0 for mid in METER_IDS}
55. pv_energy_kwh = {mid: 0.0 for mid in METER_IDS}
56. wind_energy_kwh = {mid: 0.0 for mid in METER_IDS}
57. battery_soc = {mid: INITIAL_SOC for mid in METER_IDS}
58. peak_until = {mid: 0.0 for mid in METER_IDS}
59.
60. def iso_now() -> str:
61.     return datetime.now(timezone.utc).isoformat()
62.
63. def day_phase(t: float) -> float:
64.     seconds_in_day = 24 * 3600
65.     return (t % seconds_in_day) / seconds_in_day
66.
67. def ambient_temperature(t: float) -> float:
68.     phase = day_phase(t)
69.     return 18 + 10 * math.sin(2 * math.pi * (phase - 0.2)) + random.gauss(0, 0.5)
70.
71. def solar_irradiance(t: float) -> float:
72.     phase = day_phase(t)
73.     daylight = max(0.0, math.sin(math.pi * phase))
74.     return max(0.0, 900 * daylight + random.gauss(0, 20))
75.
76. def wind_speed(t: float) -> float:
77.     phase = day_phase(t)
78.     base = 6 + 2 * math.sin(2 * math.pi * phase)
79.     return max(0.0, base + random.gauss(0, 1.2))
80.
81. def load_profile_kw(t: float) -> float:
82.     phase = day_phase(t)
83.     load = BASE_LOAD_KW + LOAD_AMPLITUDE_KW * (0.5 * (1 + math.sin(2 * math.pi * (phase
- 0.25))))
84.     return max(0.0, load + random.gauss(0, LOAD_NOISE_STD))
85.
86. def maybe_start_peak(mid: str, t: float) -> None:
87.     if t >= peak_until[mid] and random.random() < PEAK_PROB:
88.         peak_until[mid] = t + PEAK_DURATION_SEC
89.
90. def pv_power_kw(irradiance: float) -> float:
91.     return max(0.0, PV_CAPACITY_KW * (irradiance / 1000.0))
92.
93. def wind_power_kw(ws: float) -> float:
94.     if ws < CUT_IN_WIND or ws >= CUT_OUT_WIND:
95.         return 0.0
96.     if ws >= RATED_WIND:
97.         return WIND_CAPACITY_KW
98.     normalized = (ws - CUT_IN_WIND) / (RATED_WIND - CUT_IN_WIND)
99.     return WIND_CAPACITY_KW * (normalized ** 3)
100.
101. def simulate_sample(mid: str, t: float, dt_sec: float) -> dict:
102.     maybe_start_peak(mid, t)
```

```
103.
104.     temp_c = ambient_temperature(t)
105.     irradiance = solar_irradiance(t)
106.     ws = wind_speed(t)
107.
108.     load_kw = load_profile_kw(t)
109.     if t < peak_until[mid]:
110.         load_kw += PEAK_ADD_KW
111.
112.     pv_kw = pv_power_kw(irradiance)
113.     wind_kw = wind_power_kw(ws)
114.     res_kw = pv_kw + wind_kw
115.
116.     net_kw = load_kw - res_kw
117.
118.     battery_charge_kw = 0.0
119.     battery_discharge_kw = 0.0
120.
121.     if net_kw < 0:
122.         available_charge = min(abs(net_kw), BATTERY_MAX_CHARGE_KW)
123.         if battery_soc[mid] < 100:
124.             battery_charge_kw = available_charge
125.             net_kw += battery_charge_kw
126.     elif net_kw > 0:
127.         available_discharge = min(net_kw, BATTERY_MAX_DISCHARGE_KW)
128.         if battery_soc[mid] > 10:
129.             battery_discharge_kw = available_discharge
130.             net_kw -= battery_discharge_kw
131.
132.     soc_delta = ((battery_charge_kw - battery_discharge_kw) * dt_sec / 3600.0) /
BATTERY_CAPACITY_KWH * 100.0
133.     battery_soc[mid] = max(0.0, min(100.0, battery_soc[mid] + soc_delta))
134.
135.     grid_import_kw = max(0.0, net_kw)
136.     grid_export_kw = max(0.0, -net_kw)
137.
138.     voltage = NOMINAL_V + random.gauss(0.0, V_NOISE)
139.     pf = max(0.5, min(1.0, PF_MEAN + random.gauss(0.0, PF_NOISE)))
140.     current = (load_kw * 1000.0) / max(1e-6, voltage * pf)
141.
142.     load_energy_kwh[mid] += load_kw * (dt_sec / 3600.0)
143.     pv_energy_kwh[mid] += pv_kw * (dt_sec / 3600.0)
144.     wind_energy_kwh[mid] += wind_kw * (dt_sec / 3600.0)
145.
146.     return {
147.         "timestamp": iso_now(),
148.         "meter_id": mid,
149.         "load_power_kw": round(load_kw, 3),
150.         "load_energy_kwh": round(load_energy_kwh[mid], 6),
151.         "pv_power_kw": round(pv_kw, 3),
152.         "pv_energy_kwh": round(pv_energy_kwh[mid], 6),
153.         "wind_power_kw": round(wind_kw, 3),
154.         "wind_energy_kwh": round(wind_energy_kwh[mid], 6),
```

```
155.     "res_power_kw": round(res_kw, 3),
156.     "grid_import_kw": round(grid_import_kw, 3),
157.     "grid_export_kw": round(grid_export_kw, 3),
158.     "battery_soc": round(battery_soc[mid], 2),
159.     "battery_charge_kw": round(battery_charge_kw, 3),
160.     "battery_discharge_kw": round(battery_discharge_kw, 3),
161.     "temperature_c": round(temp_c, 2),
162.     "wind_speed_mps": round(ws, 2),
163.     "solar_irradiance_wm2": round(irradiance, 2),
164.     "voltage_v": round(voltage, 2),
165.     "current_a": round(current, 3),
166.     "pf": round(pf, 3),
167. }
168.
169. def main():
170.     client = mqtt.Client(client_id=f"simulator-{random.randint(1000,9999)}")
171.     if MQTT_USERNAME:
172.         client.username_pw_set(MQTT_USERNAME, MQTT_PASSWORD)
173.
174.     client.connect(MQTT_HOST, MQTT_PORT, keepalive=60)
175.     client.loop_start()
176.
177.     last_t = time.time()
178.     try:
179.         while True:
180.             now_t = time.time()
181.             dt = max(0.001, now_t - last_t)
182.             last_t = now_t
183.
184.             for mid in METER_IDS:
185.                 payload = simulate_sample(mid, now_t, dt)
186.                 topic = f"{MQTT_BASE_TOPIC}/{mid}"
187.                 client.publish(topic, json.dumps(payload), qos=0, retain=False)
188.
189.             time.sleep(PUBLISH_INTERVAL_SEC)
190.     except KeyboardInterrupt:
191.         pass
192.     finally:
193.         client.loop_stop()
194.         client.disconnect()
195.
196. if __name__ == "__main__":
197.     main()
198.
```

Παράρτημα Β: Συναρτήσεις Node-RED

function_raw_telemetry

```
1. const d = msg.payload;
2.
3. // Normalize timestamp
4. const cleanTimestamp = d.timestamp
5.   ? new Date(d.timestamp).toISOString()
6.   : new Date().toISOString();
7.
8. msg.method = "POST";
9. msg.url = "http://surrealdb:8000/sql";
10.
11. msg.headers = {
12.   "Authorization": "Basic " + Buffer.from("root:root").toString("base64"),
13.   "Accept": "application/json",
14.   "Content-Type": "application/json",
15.   "Surreal-NS": "ems",
16.   "Surreal-DB": "emsdb"
17. };
18.
19. msg.payload = `
20. CREATE telemetry SET
21.   timestamp = <datetime>`${cleanTimestamp}`,
22.   meter_id = `${d.meter_id || 'unknown'}`,
23.
24.   load_power_kw = ${d.load_power_kw || 0},
25.   load_energy_kwh = ${d.load_energy_kwh || 0},
26.
27.   pv_power_kw = ${d.pv_power_kw || 0},
28.   pv_energy_kwh = ${d.pv_energy_kwh || 0},
29.
30.   wind_power_kw = ${d.wind_power_kw || 0},
31.   wind_energy_kwh = ${d.wind_energy_kwh || 0},
32.
33.   res_power_kw = ${d.res_power_kw || 0},
34.
35.   grid_import_kw = ${d.grid_import_kw || 0},
36.   grid_export_kw = ${d.grid_export_kw || 0},
37.
38.   battery_soc = ${d.battery_soc || 0},
39.   battery_charge_kw = ${d.battery_charge_kw || 0},
40.   battery_discharge_kw = ${d.battery_discharge_kw || 0},
41.
42.   temperature_c = ${d.temperature_c || 0},
43.   wind_speed_mps = ${d.wind_speed_mps || 0},
44.   solar_irradiance_wm2 = ${d.solar_irradiance_wm2 || 0},
45.
46.   voltage_v = ${d.voltage_v || 0},
47.   current_a = ${d.current_a || 0},
48.   pf = ${d.pf || 0};
```

```
49. `;  
50.  
51. return msg;  
52.
```

function_generate_alerts

```
1. const d = msg.payload;  
2. const alerts = [];  
3. const timestamp = d.timestamp || new Date().toISOString();  
4. const meterId = d.meter_id || "unknown";  
5.  
6. // thresholds  
7. const LOAD_THRESHOLD = 60.0;  
8. const GRID_IMPORT_THRESHOLD = 25.0;  
9. const LOW_BATTERY_THRESHOLD = 20.0;  
10. const HIGH_TEMP_THRESHOLD = 35.0;  
11. const LOW_PF_THRESHOLD = 0.85;  
12. const OVERVOLTAGE_THRESHOLD = 245.0;  
13. const PV_IRRADIANCE_THRESHOLD = 600.0;  
14. const PV_UNDERPERFORMANCE_THRESHOLD = 5.0;  
15. const WIND_SPEED_THRESHOLD = 8.0;  
16. const WIND_UNDERPERFORMANCE_THRESHOLD = 3.0;  
17.  
18. // 1. High load  
19. if ((d.load_power_kw || 0) > LOAD_THRESHOLD) {  
20.   alerts.push({  
21.     timestamp: timestamp,  
22.     meter_id: meterId,  
23.     alert_type: "HIGH_LOAD",  
24.     severity: "high",  
25.     message: "Η στιγμιαία κατανάλωση υπερέβη το όριο.",  
26.     value: d.load_power_kw,  
27.     threshold: LOAD_THRESHOLD,  
28.     status: "active"  
29.   });  
30. }  
31.  
32. // 2. High grid import  
33. if ((d.grid_import_kw || 0) > GRID_IMPORT_THRESHOLD) {  
34.   alerts.push({  
35.     timestamp: timestamp,  
36.     meter_id: meterId,  
37.     alert_type: "HIGH_GRID_IMPORT",  
38.     severity: "high",  
39.     message: "Η εισαγωγή ισχύος από το δίκτυο είναι υψηλή.",  
40.     value: d.grid_import_kw,  
41.     threshold: GRID_IMPORT_THRESHOLD,  
42.     status: "active"  
43.   });  
44. }
```

```
45.  
46. // 3. Low battery  
47. if ((d.battery_soc || 0) < LOW_BATTERY_THRESHOLD) {  
48.     alerts.push({  
49.         timestamp: timestamp,  
50.         meter_id: meterId,  
51.         alert_type: "LOW_BATTERY_SOC",  
52.         severity: "medium",  
53.         message: "Η κατάσταση φόρτισης της μπαταρίας είναι χαμηλή.",  
54.         value: d.battery_soc,  
55.         threshold: LOW_BATTERY_THRESHOLD,  
56.         status: "active"  
57.     });  
58. }  
59.  
60. // 4. High ambient temperature  
61. if ((d.temperature_c || 0) > HIGH_TEMP_THRESHOLD) {  
62.     alerts.push({  
63.         timestamp: timestamp,  
64.         meter_id: meterId,  
65.         alert_type: "HIGH_AMBIENT_TEMP",  
66.         severity: "medium",  
67.         message: "Η θερμοκρασία περιβάλλοντος υπερέβη το όριο.",  
68.         value: d.temperature_c,  
69.         threshold: HIGH_TEMP_THRESHOLD,  
70.         status: "active"  
71.     });  
72. }  
73.  
74. // 5. Low power factor  
75. if ((d.pf || 1) < LOW_PF_THRESHOLD) {  
76.     alerts.push({  
77.         timestamp: timestamp,  
78.         meter_id: meterId,  
79.         alert_type: "LOW_POWER_FACTOR",  
80.         severity: "medium",  
81.         message: "Ο συντελεστής ισχύος είναι χαμηλός.",  
82.         value: d.pf,  
83.         threshold: LOW_PF_THRESHOLD,  
84.         status: "active"  
85.     });  
86. }  
87.  
88. // 6. Overvoltage  
89. if ((d.voltage_v || 0) > OVERVOLTAGE_THRESHOLD) {  
90.     alerts.push({  
91.         timestamp: timestamp,  
92.         meter_id: meterId,  
93.         alert_type: "OVERVOLTAGE",  
94.         severity: "high",  
95.         message: "Η τάση υπερέβη το επιτρεπτό όριο.",  
96.         value: d.voltage_v,  
97.         threshold: OVERVOLTAGE_THRESHOLD,
```

```
98.         status: "active"
99.     });
100. }
101.
102. // 7. PV underperformance
103. if ((d.solar_irradiance_wm2 || 0) > PV_IRRADIANCE_THRESHOLD &&
104.     (d.pv_power_kw || 0) < PV_UNDERPERFORMANCE_THRESHOLD) {
105.     alerts.push({
106.         timestamp: timestamp,
107.         meter_id: meterId,
108.         alert_type: "PV_UNDERPERFORMANCE",
109.         severity: "medium",
110.         message: "Χαμηλή φωτοβολταϊκή παραγωγή σε σχέση με την ηλιακή ακτινοβολία.",
111.         value: d.pv_power_kw,
112.         threshold: PV_UNDERPERFORMANCE_THRESHOLD,
113.         status: "active"
114.     });
115. }
116.
117. // 8. Wind underperformance
118. if ((d.wind_speed_mps || 0) > WIND_SPEED_THRESHOLD &&
119.     (d.wind_power_kw || 0) < WIND_UNDERPERFORMANCE_THRESHOLD) {
120.     alerts.push({
121.         timestamp: timestamp,
122.         meter_id: meterId,
123.         alert_type: "WIND_UNDERPERFORMANCE",
124.         severity: "medium",
125.         message: "Χαμηλή αιολική παραγωγή σε σχέση με την ταχύτητα ανέμου.",
126.         value: d.wind_power_kw,
127.         threshold: WIND_UNDERPERFORMANCE_THRESHOLD,
128.         status: "active"
129.     });
130. }
131.
132. msg.payload = alerts;
133. return msg;
134.
```

function_insert_alert

```
1. const a = msg.payload;
2.
3. msg.method = "POST";
4. msg.url = "http://surrealdb:8000/sql";
5.
6. msg.headers = {
7.     "Authorization": "Basic " + Buffer.from("root:root").toString("base64"),
8.     "Accept": "application/json",
9.     "Content-Type": "application/json",
10.    "Surreal-NS": "ems",
11.    "Surreal-DB": "emsdb"
```

```
12. };
13.
14. msg.payload = `
15. CREATE alerts SET
16.     timestamp = <datetime>"${a.timestamp}",
17.     meter_id = "${a.meter_id}",
18.     alert_type = "${a.alert_type}",
19.     severity = "${a.severity}",
20.     message = "${a.message}",
21.     value = ${a.value},
22.     threshold = ${a.threshold},
23.     status = "${a.status}";
24. `;
25.
26. return msg;
27.
```

function_predicted_values

```
1. const a = msg.payload;
2.
3. msg.method = "POST";
4. msg.url = "http://surrealdb:8000/sql";
5.
6. msg.headers = {
7.   "Authorization": "Basic " + Buffer.from("root:root").toString("base64"),
8.   "Accept": "application/json",
9.   "Content-Type": "application/json",
10.  "Surreal-NS": "ems",
11.  "Surreal-DB": "emsdb"
12. };
13.
14. msg.payload = `
15. CREATE prediction SET
16.     timestamp = <datetime>"${a.timestamp}",
17.     meter_id = "${a.meter_id}",
18.     anomaly_type = "${a.anomaly_type}",
19.     actual_value = ${a.actual_value},
20.     predicted_value = ${a.predicted_value},
21.     error_pct = ${a.error_pct},
22.     threshold = ${a.threshold},
23.     severity = "${a.severity}",
24.     message = "${a.message}",
25.     status = "active";
26. `;
27.
28. return msg;
29.
```

Παράρτημα Γ: Εκπαίδευση μοντέλου

```
1. #!/usr/bin/env python3
2. import base64
3. import json
4. from pathlib import Path
5.
6. import joblib
7. import pandas as pd
8. import requests
9. from sklearn.linear_model import LinearRegression
10. from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score,
root_mean_squared_error
11.
12. SURREAL_URL = "http://surrealdb:8000/sql"
13. SURREAL_USER = "root"
14. SURREAL_PASS = "root"
15. SURREAL_NS = "ems"
16. SURREAL_DB = "emsdb"
17.
18. MODEL_DIR = Path("./models")
19. MODEL_DIR.mkdir(parents=True, exist_ok=True)
20. MODEL_PATH = MODEL_DIR / "pv_model_m1.joblib"
21.
22. def surreal_sql(query: str):
23.     auth = base64.b64encode(f"{SURREAL_USER}:{SURREAL_PASS}".encode()).decode()
24.     headers = {
25.         "Authorization": f"Basic {auth}",
26.         "Accept": "application/json",
27.         "Content-Type": "application/json",
28.         "Surreal-NS": SURREAL_NS,
29.         "Surreal-DB": SURREAL_DB,
30.     }
31.
32.     response = requests.post(
33.         SURREAL_URL,
34.         headers=headers,
35.         data=query,
36.         timeout=30,
37.     )
38.     response.raise_for_status()
39.     return response.json()
40.
41. def load_training_data(meter_id: str = "m1", limit: int = 3000) -> pd.DataFrame:
42.     query = f"""
43.     SELECT
44.         timestamp,
45.         meter_id,
46.         solar_irradiance_wm2,
47.         temperature_c,
48.         pv_power_kw
49.     FROM telemetry
```

```
50.     WHERE meter_id = "{meter_id}"
51.         AND solar_irradiance_wm2 IS NOT NONE
52.         AND temperature_c IS NOT NONE
53.         AND pv_power_kw IS NOT NONE
54.         AND solar_irradiance_wm2 > 100
55.     ORDER BY timestamp ASC
56.     LIMIT {limit};
57.     ""
58.
59.     result = surreal_sql(query)
60.     rows = result[0]["result"]
61.     df = pd.DataFrame(rows)
62.
63.     if df.empty:
64.         raise ValueError("Δεν βρέθηκαν training δεδομένα.")
65.
66.     # Μετατροπή τύπων
67.     df["solar_irradiance_wm2"] = pd.to_numeric(df["solar_irradiance_wm2"],
errors="coerce")
68.     df["temperature_c"] = pd.to_numeric(df["temperature_c"], errors="coerce")
69.     df["pv_power_kw"] = pd.to_numeric(df["pv_power_kw"], errors="coerce")
70.     df["timestamp"] = pd.to_datetime(df["timestamp"], errors="coerce")
71.
72.     df = df.dropna(subset=["solar_irradiance_wm2", "temperature_c", "pv_power_kw",
"timestamp"])
73.     df = df.sort_values("timestamp").reset_index(drop=True)
74.
75.     return df
76.
77. def time_based_split(df: pd.DataFrame, train_ratio: float = 0.7):
78.     split_idx = int(len(df) * train_ratio)
79.     train_df = df.iloc[:split_idx].copy()
80.     test_df = df.iloc[split_idx:].copy()
81.     return train_df, test_df
82.
83. def train_model(train_df: pd.DataFrame) -> LinearRegression:
84.     X_train = train_df[["solar_irradiance_wm2", "temperature_c"]]
85.     y_train = train_df["pv_power_kw"]
86.
87.     model = LinearRegression()
88.     model.fit(X_train, y_train)
89.     return model
90.
91. def evaluate_model(model: LinearRegression, test_df: pd.DataFrame) -> dict:
92.     X_test = test_df[["solar_irradiance_wm2", "temperature_c"]]
93.     y_test = test_df["pv_power_kw"]
94.
95.     y_pred = model.predict(X_test)
96.
97.     y_test = y_test.reset_index(drop=True)
98.     y_pred = pd.Series(y_pred)
99.
100.     mae = mean_absolute_error(y_test, y_pred)
```

```
101.     rmse = mean_squared_error(y_test, y_pred) ** 0.5
102.     r2 = r2_score(y_test, y_pred)
103.
104.     error_pct = ((y_test - y_pred).abs() / y_test.clip(lower=1e-6)) * 100
105.     mean_error_pct = error_pct.mean()
106.
107.     return {
108.         "mae": float(mae),
109.         "rmse": float(rmse),
110.         "r2": float(r2),
111.         "mean_error_pct": float(mean_error_pct),
112.     }
113.
114. def save_model(model: LinearRegression, meter_id: str = "m1"):
115.     model_path = MODEL_DIR / f"pv_model_{meter_id}.joblib"
116.     joblib.dump(model, model_path)
117.     return model_path
118.
119. def main():
120.     meter_id = "m1"
121.     df = load_training_data(meter_id=meter_id, limit=3000)
122.
123.     if len(df) < 100:
124.         raise ValueError(f"Πολύ λίγα δεδομένα για training: {len(df)}")
125.
126.     train_df, test_df = time_based_split(df, train_ratio=0.7)
127.
128.     model = train_model(train_df)
129.     metrics = evaluate_model(model, test_df)
130.     model_path = save_model(model, meter_id=meter_id)
131.
132.     print(f"Training rows: {len(train_df)}")
133.     print(f"Test rows: {len(test_df)}")
134.     print("Metrics:")
135.     print(json.dumps(metrics, indent=2))
136.     print(f"Model saved to: {model_path}")
137.
138. if __name__ == "__main__":
139.     main()
140.
```

Παράρτημα Δ: Microservice για πρόβλεψη σε πραγματικό χρόνο

```
1. from flask import Flask, request, jsonify
2. import joblib
3. import numpy as np
4. from pathlib import Path
5.
6. app = Flask(__name__)
7.
8. MODEL_PATH = Path("/app/models/pv_model_m1.joblib")
9. model = joblib.load(MODEL_PATH)
10.
11. print("Coefficients:", model.coef_)
12. print("Intercept:", model.intercept_)
13.
14. THRESHOLD_PCT = 15.0
15.
16. @app.route("/predict/pv", methods=["POST"])
17. def predict_pv():
18.     data = request.get_json(force=True)
19.
20.     meter_id = data.get("meter_id", "unknown")
21.     timestamp = data.get("timestamp")
22.     irradiance = float(data.get("solar_irradiance_wm2", 0))
23.     temperature = float(data.get("temperature_c", 0))
24.     actual_pv = float(data.get("pv_power_kw", 0))
25.
26.     predicted_pv = float(model.predict(np.array([[irradiance, temperature]]))[0])
27.     error_pct = abs(actual_pv - predicted_pv) / max(abs(predicted_pv), 1e-6) * 100.0
28.     anomaly = error_pct > THRESHOLD_PCT
29.
30.     return jsonify({
31.         "timestamp": timestamp,
32.         "meter_id": meter_id,
33.         "actual_value": actual_pv,
34.         "predicted_value": predicted_pv,
35.         "error_pct": error_pct,
36.         "threshold": THRESHOLD_PCT,
37.         "anomaly": anomaly,
38.         "anomaly_type": "PV_MODEL_DEVIATION" if anomaly else "NORMAL",
39.         "severity": "high" if error_pct > 30 else "medium" if anomaly else "normal",
40.         "message": "Η πραγματική PV παραγωγή αποκλίνει σημαντικά από την προβλεπόμενη τιμή." if anomaly else "Normal operation"
41.     })
42.
43. @app.route("/health", methods=["GET"])
44. def health():
45.     return jsonify({"status": "ok"})
46.
47. if __name__ == "__main__":
```

```
48.     app.run(host="0.0.0.0", port=5000)  
49.
```

Παράρτημα Ε: Αυτοματοποιημένη διαδικασία δημιουργίας χρηστών με python

```
1. #!/usr/bin/env python3
2. """
3. Ρύθμιση Ρόλων Χρηστών και Δικαιωμάτων Grafana
4. Σύστημα Ενεργειακής Παρακολούθησης - EMS
5.
6. Ρόλοι:
7. - director : Viewer → KPIs, CO2, οικονομικά
8. - manager : Editor → Real-time monitoring, alerts, ενεργειακές ροές
9. - technician : Viewer → Τηλεμετρία, προβλέψεις, τεχνικά γραφήματα
10. """
11.
12. import sys
13. import time
14. import requests
15. from requests.auth import HTTPBasicAuth
16.
17. GRAFANA_URL = "http://grafana:3000"
18. ADMIN_AUTH = HTTPBasicAuth("admin", "admin")
19. HEADERS = {"Content-Type": "application/json"}
20.
21. # — Ορισμός Ρόλων —————
22. USERS = [
23.     {
24.         "name": "Διευθυντής",
25.         "email": "director@ems.local",
26.         "login": "director",
27.         "password": "Director@123",
28.         "orgRole": "Viewer",
29.         "team": "Executives",
30.         "homeDashboard": "KPIs",
31.         "folders": ["Εκτελεστικά"],
32.     },
33.     {
34.         "name": "Διαχειριστής Εγκατάστασης",
35.         "email": "manager@ems.local",
36.         "login": "manager",
37.         "password": "Manager@123",
38.         "orgRole": "Editor",
39.         "team": "Facility Managers",
40.         "homeDashboard": "CURRENT SYSTEM STATUS",
41.         "folders": ["Εκτελεστικά", "Λειτουργία"],
42.     },
43.     {
44.         "name": "Τεχνικός",
45.         "email": "technician@ems.local",
46.         "login": "technician",
47.         "password": "Tech@123",
48.         "orgRole": "Viewer",
```

```
49.         "team": "Technicians",
50.         "homeDashboard": "PREDICTIONS",
51.         "folders": ["Λειτουργία", "Τεχνικά"],
52.     },
53. ]
54.
55. FOLDERS = [
56.     {
57.         "uid": "executives-folder",
58.         "title": "Εκτελεστικά",
59.         "description": "Δείκτες απόδοσης και περιβαλλοντικές μετρικές για διοίκηση",
60.     },
61.     {
62.         "uid": "operations-folder",
63.         "title": "Λειτουργία",
64.         "description": "Real-time παρακολούθηση και διαχείριση ειδοποιήσεων",
65.     },
66.     {
67.         "uid": "technical-folder",
68.         "title": "Τεχνικά",
69.         "description": "Τηλεμετρία, προβλέψεις ML και ανίχνευση ανωμαλιών",
70.     },
71. ]
72.
73.
74. # — Βοηθητικές συναρτήσεις —————
75.
76. def wait_for_grafana(max_wait: int = 120) -> bool:
77.     print("[ ] Αναμονή για εκκίνηση Grafana...")
78.     for _ in range(max_wait):
79.         try:
80.             r = requests.get(f"{GRAFANA_URL}/api/health", timeout=3)
81.             if r.status_code == 200 and r.json().get("database") == "ok":
82.                 print("[OK] Grafana είναι έτοιμο.")
83.                 return True
84.             except requests.exceptions.ConnectionError:
85.                 pass
86.             time.sleep(1)
87.     print("[ERROR] Το Grafana δεν ξεκίνησε εντός του χρονορίου.")
88.     return False
89.
90.
91. def create_user(user: dict) -> int | None:
92.     payload = {
93.         "name": user["name"],
94.         "email": user["email"],
95.         "login": user["login"],
96.         "password": user["password"],
97.         "OrgId": 1,
98.     }
99.     r = requests.post(
100.         f"{GRAFANA_URL}/api/admin/users",
101.         json=payload,
```

```
102.         auth=ADMIN_AUTH,
103.         headers=HEADERS,
104.     )
105.     if r.status_code == 200:
106.         uid = r.json()["id"]
107.         print(f" [OK] Χρήστης '{user['login']}' δημιουργήθηκε (id={uid})")
108.         return uid
109.     elif r.status_code == 412:
110.         # Already exists - fetch existing id
111.         r2 = requests.get(
112.             f"{GRAFANA_URL}/api/users/lookup?loginOrEmail={user['login']}",
113.             auth=ADMIN_AUTH,
114.         )
115.         if r2.status_code == 200:
116.             uid = r2.json()["id"]
117.             print(f" [INFO] Χρήστης '{user['login']}' υπάρχει ήδη (id={uid})")
118.             return uid
119.     else:
120.         print(f" [WARN] Αποτυχία δημιουργίας '{user['login']}': {r.text}")
121.     return None
122.
123.
124. def set_org_role(user_id: int, role: str) -> None:
125.     r = requests.patch(
126.         f"{GRAFANA_URL}/api/org/users/{user_id}",
127.         json={"role": role},
128.         auth=ADMIN_AUTH,
129.         headers=HEADERS,
130.     )
131.     if r.status_code == 200:
132.         print(f" [OK] Org role '{role}' ορίστηκε (user_id={user_id})")
133.     else:
134.         print(f" [WARN] Αποτυχία ορισμού role: {r.text}")
135.
136.
137. def create_team(name: str) -> int | None:
138.     r = requests.post(
139.         f"{GRAFANA_URL}/api/teams",
140.         json={"name": name, "orgId": 1},
141.         auth=ADMIN_AUTH,
142.         headers=HEADERS,
143.     )
144.     if r.status_code == 200:
145.         tid = r.json()["teamId"]
146.         print(f" [OK] Team '{name}' δημιουργήθηκε (id={tid})")
147.         return tid
148.     elif r.status_code == 409:
149.         # Fetch existing
150.         r2 = requests.get(
151.             f"{GRAFANA_URL}/api/teams/search?name={name}",
152.             auth=ADMIN_AUTH,
153.         )
154.         if r2.status_code == 200 and r2.json()["totalCount"] > 0:
```

```
155.         tid = r2.json()["teams"][0]["id"]
156.         print(f" [INFO] Team '{name}' υπάρχει ήδη (id={tid})")
157.         return tid
158.     else:
159.         print(f" [WARN] Αποτυχία δημιουργίας team '{name}': {r.text}")
160.     return None
161.
162.
163. def add_to_team(team_id: int, user_id: int) -> None:
164.     r = requests.post(
165.         f"{GRAFANA_URL}/api/teams/{team_id}/members",
166.         json={"userId": user_id},
167.         auth=ADMIN_AUTH,
168.         headers=HEADERS,
169.     )
170.     if r.status_code == 200:
171.         print(f" [OK] user_id={user_id} προστέθηκε στο team {team_id}")
172.     elif "already" in r.text.lower() or r.status_code == 400:
173.         print(f" [INFO] Ήδη μέλος του team.")
174.     else:
175.         print(f" [WARN] {r.text}")
176.
177.
178. def create_folder(folder: dict) -> str | None:
179.     r = requests.post(
180.         f"{GRAFANA_URL}/api/folders",
181.         json={"uid": folder["uid"], "title": folder["title"]},
182.         auth=ADMIN_AUTH,
183.         headers=HEADERS,
184.     )
185.     if r.status_code == 200:
186.         print(f" [OK] Φάκελος '{folder['title']}' δημιουργήθηκε")
187.         return folder["uid"]
188.     elif r.status_code == 412:
189.         print(f" [INFO] Φάκελος '{folder['title']}' υπάρχει ήδη")
190.         return folder["uid"]
191.     else:
192.         print(f" [WARN] Αποτυχία δημιουργίας φακέλου '{folder['title']}': {r.text}")
193.     return None
194.
195.
196. def find_dashboard_uid(name: str) -> str | None:
197.     r = requests.get(
198.         f"{GRAFANA_URL}/api/search",
199.         params={"query": name, "type": "dash-db"},
200.         auth=ADMIN_AUTH,
201.     )
202.     if r.status_code == 200:
203.         results = r.json()
204.         for d in results:
205.             if d.get("title", "").upper() == name.upper():
206.                 return d.get("uid")
207.     print(f" [WARN] Dashboard '{name}' δεν βρέθηκε")
```

```
208.     return None
209.
210.
211. def set_home_dashboard(user_login: str, user_password: str, dashboard_uid: str) -> None:
212.     # Πρέπει να κάνουμε authenticate ως ο ίδιος ο χρήστης
213.     user_auth = HTTPBasicAuth(user_login, user_password)
214.     r = requests.put(
215.         f"{GRAFANA_URL}/api/user/preferences",
216.         json={"homeDashboardUID": dashboard_uid},
217.         auth=user_auth,
218.         headers=HEADERS,
219.     )
220.     if r.status_code == 200:
221.         print(f"    [OK] Home dashboard ορίστηκε (uid={dashboard_uid})")
222.     else:
223.         print(f"    [WARN] {r.text}")
224.
225.
226. def set_folder_permissions(folder_uid: str, items: list[dict]) -> None:
227.     """
228.     Αντικαθιστά όλα τα permissions του φακέλου με μία κλήση.
229.     items: [{"teamId": int, "permission": int}, ...]
230.     permission: 1=View, 2=Edit, 4=Admin
231.     """
232.     r = requests.post(
233.         f"{GRAFANA_URL}/api/folders/{folder_uid}/permissions",
234.         json={"items": items},
235.         auth=ADMIN_AUTH,
236.         headers=HEADERS,
237.     )
238.     if r.status_code != 200:
239.         print(f"    [WARN] {r.text}")
240.         return
241.
242.     # Άμεση επαλήθευση
243.     r2 = requests.get(
244.         f"{GRAFANA_URL}/api/folders/{folder_uid}/permissions",
245.         auth=ADMIN_AUTH,
246.     )
247.     saved = r2.json() if r2.status_code == 200 else []
248.     saved_teams = {p["teamId"] for p in saved}
249.     expected_teams = {i["teamId"] for i in items}
250.     if expected_teams == saved_teams:
251.         print(f"    [OK] '{folder_uid}': {[p['team'] + '=' + p['permissionName'] for p
in saved]})")
252.     else:
253.         missing = expected_teams - saved_teams
254.         print(f"    [WARN] '{folder_uid}': teams {missing} δεν αποθηκεύτηκαν")
255.
256.
257. # — Κύρια ποή —
258.
259. def main() -> None:
```

```
260.     if not wait_for_grafana():
261.         sys.exit(1)
262.
263.     print("\n[DIR] Δημιουργία φακέλων dashboard...")
264.     for folder in FOLDERS:
265.         create_folder(folder)
266.
267.     print("\n[TEAMS] Δημιουργία teams...")
268.     team_ids: dict[str, int] = {}
269.     for user in USERS:
270.         tname = user["team"]
271.         if tname not in team_ids:
272.             tid = create_team(tname)
273.             if tid:
274.                 team_ids[tname] = tid
275.
276.     print("\n[USERS] Δημιουργία χρηστών και ανάθεση ρόλων...")
277.     for user in USERS:
278.         print(f"\n → {user['name']} ({user['login']} / {user['orgRole']})")
279.         uid = create_user(user)
280.         if uid is None:
281.             continue
282.         set_org_role(uid, user["orgRole"])
283.         tname = user["team"]
284.         if tname in team_ids:
285.             add_to_team(team_ids[tname], uid)
286.
287.     # Ανάθεση φακέλων σε teams – μία κλήση ανά φάκελο (αντικαθιστά όλα τα permissions)
288.     print("\n[PERMS] Ρύθμιση δικαιωμάτων φακέλων ανά ρόλο...")
289.
290.     # permission: 1=View, 2=Edit
291.     # Executives → μόνο KPIs φάκελος, View
292.     # Facility Managers → Λειτουργία φάκελος, Edit
293.     # Technicians → Λειτουργία (View) + Τεχνικά (View)
294.     folder_permissions = {
295.         "executives-folder": [
296.             {"teamId": team_ids["Executives"], "permission": 1},
297.         ],
298.         "operations-folder": [
299.             {"teamId": team_ids["Facility Managers"], "permission": 1},
300.             {"teamId": team_ids["Technicians"], "permission": 1},
301.         ],
302.         "technical-folder": [
303.             {"teamId": team_ids["Technicians"], "permission": 1},
304.         ],
305.     }
306.     for folder_uid, items in folder_permissions.items():
307.         # Φιλτράρισμα: αφαίρεσε teams που δεν δημιουργήθηκαν
308.         valid_items = [i for i in items if i.get("teamId")]
309.         if valid_items:
310.             set_folder_permissions(folder_uid, valid_items)
311.
312.     print("\n[HOME] Ορισμός προεπιλεγμένου home dashboard ανά χρήστη...")
```

```
313. # login → (dashboard_name, password)
314. home_map = {
315.     "director": ("KPIs", "Director@123"),
316.     "manager": ("CURRENT SYSTEM STATUS", "Manager@123"),
317.     "technician": ("PREDICTIONS", "Tech@123"),
318. }
319. for login, (dash_name, password) in home_map.items():
320.     uid = find_dashboard_uid(dash_name)
321.     if uid:
322.         print(f" → {login}: '{dash_name}'")
323.         set_home_dashboard(login, password, uid)
324.     else:
325.         print(f" [WARN] Dashboard '{dash_name}' δεν βρέθηκε – παράλειψη")
326.
327. print("\n" + "="*60)
328. print("[OK] Ρύθμιση ρόλων ολοκληρώθηκε!")
329. print("="*60)
330. print("\nΔιαθέσιμοι χρήστες:")
331. print(f" {'Login':<15} {'Password':<15} {'Ρόλος':<10} {'Πρόσβαση'}")
332. print(f" {'-'*70}")
333. access_map = {
334.     "director": "KPIs, CO2, Οικονομικά",
335.     "manager": "Status, Energy Monitoring, Alerts",
336.     "technician": "Telemetry, Predictions, Anomalies",
337. }
338. for u in USERS:
339.     print(f" {u['login']:<15} {u['password']:<15} {u['orgRole']:<10}
340. {access_map[u['login']]}")
341.
342. if __name__ == "__main__":
343.     main()
344.
```

Παράρτημα ΣΤ: Δοκιμές Σεναρίων

```
1. #!/usr/bin/env python3
2. """
3. Δοκιμές Λειτουργικών Σεναρίων - Σύστημα Ενεργειακής Παρακολούθησης (EMS)
4.
5. Δομή δοκιμών:
6.   - TestAlertLogic      : Unit tests λογικής ανίχνευσης συμβάντων (χωρίς δίκτυο)
7.   - TestAlertIntegration : Integration tests πλήρους ροής MQTT→Node-RED→SurrealDB
8.   - TestPredictAPI      : Σενάρια δοκιμών microservice πρόβλεψης (απαιτεί εκτέλεση)
9.
10. Εκτέλεση:
11.   python -m pytest test_scenarios.py -v
12.   ή
13.   python test_scenarios.py
14. """
15.
16. import base64
17. import json
18. import time
19. import unittest
20. from datetime import datetime, timezone
21.
22. import requests
23.
24. API_URL = "http://localhost:5000/predict/pv"
25.
26. # — Αντιγραφή λογικής ειδοποιήσεων από Node-RED (function_generate_alerts) —
27.
28. LOAD_THRESHOLD      = 60.0
29. GRID_IMPORT_THRESHOLD = 25.0
30. LOW_BATTERY_THRESHOLD = 20.0
31. HIGH_TEMP_THRESHOLD = 35.0
32. LOW_PF_THRESHOLD    = 0.85
33. OVERVOLTAGE_THRESHOLD = 245.0
34. PV_IRRADIANCE_THRESHOLD = 600.0
35. PV_UNDERPERF_THRESHOLD = 5.0
36. WIND_SPEED_THRESHOLD = 8.0
37. WIND_UNDERPERF_THRESHOLD = 3.0
38. ANOMALY_THRESHOLD_PCT = 15.0
39.
40. def generate_alerts(data: dict) -> list[dict]:
41.     """Αναπαράγει τη λογική του Node-RED function_generate_alerts σε Python."""
42.     alerts = []
43.
44.     if data.get("load_power_kw", 0) > LOAD_THRESHOLD:
45.         alerts.append({"alert_type": "HIGH_LOAD", "severity": "high"})
46.
47.     if data.get("grid_import_kw", 0) > GRID_IMPORT_THRESHOLD:
48.         alerts.append({"alert_type": "HIGH_GRID_IMPORT", "severity": "high"})
49.
50.     if data.get("battery_soc", 100) < LOW_BATTERY_THRESHOLD:
```

```
51.         alerts.append({"alert_type": "LOW_BATTERY_SOC", "severity": "medium"})
52.
53.     if data.get("temperature_c", 0) > HIGH_TEMP_THRESHOLD:
54.         alerts.append({"alert_type": "HIGH_AMBIENT_TEMP", "severity": "medium"})
55.
56.     if data.get("pf", 1) < LOW_PF_THRESHOLD:
57.         alerts.append({"alert_type": "LOW_POWER_FACTOR", "severity": "medium"})
58.
59.     if data.get("voltage_v", 0) > OVERVOLTAGE_THRESHOLD:
60.         alerts.append({"alert_type": "OVERVOLTAGE", "severity": "high"})
61.
62.     if (data.get("solar_irradiance_wm2", 0) > PV_IRRADIANCE_THRESHOLD and
63.         data.get("pv_power_kw", 0) < PV_UNDERPERF_THRESHOLD):
64.         alerts.append({"alert_type": "PV_UNDERPERFORMANCE", "severity": "medium"})
65.
66.     if (data.get("wind_speed_mps", 0) > WIND_SPEED_THRESHOLD and
67.         data.get("wind_power_kw", 0) < WIND_UNDERPERF_THRESHOLD):
68.         alerts.append({"alert_type": "WIND_UNDERPERFORMANCE", "severity": "medium"})
69.
70.     return alerts
71.
72. def alert_types(alerts: list[dict]) -> set[str]:
73.     return {a["alert_type"] for a in alerts}
74.
75. # — Unit Tests: Λογική Ειδοποιήσεων —————
76.
77. class TestAlertLogic(unittest.TestCase):
78.     """
79.     Unit tests που ελέγχουν τη λογική παραγωγής ειδοποιήσεων.
80.     Δεν απαιτούν εκτελούμενη υπηρεσία – τρέχουν πάντα.
81.     """
82.
83.     def _base(self) -> dict:
84.         """Βασική μέτρηση κανονικής λειτουργίας χωρίς alerts."""
85.         return {
86.             "load_power_kw": 35.0,
87.             "grid_import_kw": 10.0,
88.             "battery_soc": 60.0,
89.             "temperature_c": 25.0,
90.             "pf": 0.95,
91.             "voltage_v": 230.0,
92.             "solar_irradiance_wm2": 400.0,
93.             "pv_power_kw": 10.0,
94.             "wind_speed_mps": 5.0,
95.             "wind_power_kw": 8.0,
96.         }
97.
98. # — Σενάριο 1: Κανονική Λειτουργία —————
99.
100. def test_scenario_1_normal_operation_no_alerts(self):
101.     """
102.     Σενάριο 1: Κανονική Λειτουργία
103.     Κατάσταση: Ισορροπημένο σύστημα, όλες οι τιμές εντός ορίων.
```

```
104.         Αναμενόμενο: Καμία ειδοποίηση.  
105.         """  
106.         data = self._base()  
107.         alerts = generate_alerts(data)  
108.         self.assertEqual(  
109.             len(alerts), 0,  
110.             f"Αναμενόταν 0 alerts σε κανονική λειτουργία, βρέθηκαν:  
{alert_types(alerts)}"  
111.         )  
112.  
113.     # — Σενάριο 2: Αιχμή Κατανάλωσης —————  
114.  
115.     def test_scenario_2_peak_load_triggers_high_load_alert(self):  
116.         """  
117.         Σενάριο 2: Αιχμή Κατανάλωσης (Peak Load)  
118.         Κατάσταση: load_power_kw = 75 kW (όριο: 60 kW).  
119.         Αναμενόμενο: Alert HIGH_LOAD με severity=high.  
120.         """  
121.         data = self._base()  
122.         data["load_power_kw"] = 75.0  
123.         data["grid_import_kw"] = 30.0  
124.  
125.         alerts = generate_alerts(data)  
126.         types = alert_types(alerts)  
127.  
128.         self.assertIn("HIGH_LOAD", types)  
129.         high_load = next(a for a in alerts if a["alert_type"] == "HIGH_LOAD")  
130.         self.assertEqual(high_load["severity"], "high")  
131.  
132.     # — Σενάριο 3: Χαμηλή Φόρτιση Μπαταρίας —————  
133.  
134.     def test_scenario_3_low_battery_triggers_alert(self):  
135.         """  
136.         Σενάριο 3: Κρίσιμη Στάθμη Μπαταρίας  
137.         Κατάσταση: battery_soc = 15% (όριο: 20%).  
138.         Αναμενόμενο: Alert LOW_BATTERY_SOC με severity=medium.  
139.         """  
140.         data = self._base()  
141.         data["battery_soc"] = 15.0  
142.  
143.         alerts = generate_alerts(data)  
144.         types = alert_types(alerts)  
145.  
146.         self.assertIn("LOW_BATTERY_SOC", types)  
147.         batt = next(a for a in alerts if a["alert_type"] == "LOW_BATTERY_SOC")  
148.         self.assertEqual(batt["severity"], "medium")  
149.  
150.     # — Σενάριο 4: Υποαπόδοση Φωτοβολταϊκών —————  
151.  
152.     def test_scenario_4_pv_underperformance_with_high_irradiance(self):  
153.         """  
154.         Σενάριο 4: Υποαπόδοση Φωτοβολταϊκού Συστήματος  
155.         Κατάσταση: Υψηλή ακτινοβολία (750 W/m2) αλλά χαμηλή παραγωγή (3 kW).
```

```
156.             Πιθανή βλάβη inverter ή ρύπανση πάνελ.  
157.     Αναμενόμενο: Alert PV_UNDERPERFORMANCE.  
158.     """  
159.     data = self._base()  
160.     data["solar_irradiance_wm2"] = 750.0  
161.     data["pv_power_kw"] = 3.0  
162.  
163.     alerts = generate_alerts(data)  
164.     self.assertIn("PV_UNDERPERFORMANCE", alert_types(alerts))  
165.  
166.     # — Σενάριο 5: Νυχτερινή Λειτουργία (Χωρίς False Positives) —————  
167.  
168.     def test_scenario_5_night_mode_no_pv_underperformance_alert(self):  
169.         """  
170.         Σενάριο 5: Νυχτερινή Λειτουργία  
171.         Κατάσταση: Μηδενική ακτινοβολία (νύχτα), μηδενική παραγωγή PV.  
172.         Αναμενόμενο: ΔΕΝ πρέπει να εμφανιστεί PV_UNDERPERFORMANCE (false positive).  
173.         """  
174.         data = self._base()  
175.         data["solar_irradiance_wm2"] = 0.0  
176.         data["pv_power_kw"] = 0.0  
177.         data["temperature_c"] = 18.0  
178.  
179.         alerts = generate_alerts(data)  
180.         self.assertNotIn(  
181.             "PV_UNDERPERFORMANCE", alert_types(alerts),  
182.             "False positive: PV_UNDERPERFORMANCE δεν πρέπει να εμφανίζεται τη νύχτα"  
183.         )  
184.  
185.     # — Σενάριο 6: Κρίσιμη Κατάσταση – Πολλαπλά Alerts —————  
186.  
187.     def test_scenario_6_critical_multi_alert_condition(self):  
188.         """  
189.         Σενάριο 6: Κρίσιμη Κατάσταση Συστήματος  
190.         Κατάσταση: Ταυτόχρονα υψηλό φορτίο, χαμηλή μπαταρία, υψηλή εισαγωγή  
191.             από δίκτυο και χαμηλός συντελεστής ισχύος.  
192.         Αναμενόμενο: Τουλάχιστον 3 διαφορετικά alerts.  
193.         """  
194.         data = self._base()  
195.         data["load_power_kw"] = 70.0  
196.         data["battery_soc"] = 12.0  
197.         data["grid_import_kw"] = 35.0  
198.         data["pf"] = 0.78  
199.         data["pv_power_kw"] = 1.0  
200.  
201.         alerts = generate_alerts(data)  
202.         types = alert_types(alerts)  
203.  
204.         self.assertGreaterEqual(  
205.             len(alerts), 3,  
206.             f"Αναμενόταν >= 3 alerts σε κρίσιμη κατάσταση, βρέθηκαν: {types}"  
207.         )  
208.         self.assertIn("HIGH_LOAD", types)
```

```
209.         self.assertIn("LOW_BATTERY_SOC", types)
210.         self.assertIn("HIGH_GRID_IMPORT", types)
211.
212. # — Integration Tests: Πλήρης Ροή Ειδοποιήσεων —————
213.
214. def _mqtt_available() -> bool:
215.     try:
216.         import paho.mqtt.client as mqtt
217.         c = mqtt.Client(client_id="test-check")
218.         c.connect("localhost", 1883, keepalive=3)
219.         c.disconnect()
220.         return True
221.     except Exception:
222.         return False
223.
224. def _surreal_available() -> bool:
225.     try:
226.         auth = base64.b64encode(b"root:root").decode()
227.         r = requests.post(
228.             "http://localhost:8000/sql",
229.             data="SELECT 1;",
230.             headers={
231.                 "Authorization": f"Basic {auth}",
232.                 "Accept": "application/json",
233.                 "Surreal-NS": "ems",
234.                 "Surreal-DB": "emsdb",
235.             },
236.             timeout=3,
237.         )
238.         return r.status_code == 200
239.     except requests.exceptions.ConnectionError:
240.         return False
241.
242. class TestAlertIntegration(unittest.TestCase):
243.     """
244.     Integration tests που ελέγχουν την πλήρη ροή ειδοποιήσεων:
245.     MQTT publish → Node-RED function_generate_alerts → SurrealDB alerts table
246.
247.     Απαιτούν: mosquitto, nodered, surrealdb containers σε λειτουργία.
248.     """
249.
250.     MQTT_HOST = "localhost"
251.     MQTT_PORT = 1883
252.     SURREAL_URL = "http://localhost:8000/sql"
253.     WAIT_SEC = 3 # αναμονή για επεξεργασία Node-RED
254.
255.     _HEADERS = {
256.         "Authorization": "Basic " + base64.b64encode(b"root:root").decode(),
257.         "Accept": "application/json",
258.         "Surreal-NS": "ems",
259.         "Surreal-DB": "emsdb",
260.     }
261.
```

```
262. @classmethod
263. def setUpClass(cls):
264.     if not _mqtt_available():
265.         raise unittest.SkipTest("MQTT broker δεν είναι διαθέσιμος (localhost:1883)")
266.     if not _surreal_available():
267.         raise unittest.SkipTest("SurrealDB δεν είναι διαθέσιμη (localhost:8000)")
268.
269. # — Βοηθητικές —————
270.
271. def _publish(self, meter_id: str, overrides: dict) -> str:
272.     """Δημοσιεύει MQTT μήνυμα και επιστρέφει το timestamp αποστολής."""
273.     import paho.mqtt.client as mqtt
274.
275.     ts = datetime.now(timezone.utc).isoformat()
276.     payload = {
277.         "timestamp": ts,
278.         "meter_id": meter_id,
279.         "load_power_kw": 35.0, "load_energy_kwh": 100.0,
280.         "pv_power_kw": 8.0, "pv_energy_kwh": 50.0,
281.         "wind_power_kw": 4.0, "wind_energy_kwh": 30.0,
282.         "res_power_kw": 12.0,
283.         "grid_import_kw": 10.0, "grid_export_kw": 0.0,
284.         "battery_soc": 60.0,
285.         "battery_charge_kw": 0.0, "battery_discharge_kw": 0.0,
286.         "temperature_c": 25.0, "wind_speed_mps": 5.0,
287.         "solar_irradiance_wm2": 400.0,
288.         "voltage_v": 230.0, "current_a": 45.0, "pf": 0.95,
289.     }
290.     payload.update(overrides)
291.     payload["timestamp"] = ts # timestamp πάντα από εμάς
292.
293.     client = mqtt.Client(client_id=f"test-{meter_id}")
294.     client.connect(self.MQTT_HOST, self.MQTT_PORT, keepalive=5)
295.     client.publish(
296.         f"energy/meter/{meter_id}",
297.         json.dumps(payload),
298.         qos=1,
299.     )
300.     client.disconnect()
301.     return ts
302.
303. def _recent_alerts(self, alert_type: str, since_iso: str) -> list:
304.     """Επιστρέφει alerts του τύπου alert_type μετά το since_iso."""
305.     query = (
306.         f"SELECT * FROM alerts "
307.         f"WHERE alert_type = '{alert_type}' "
308.         f"ORDER BY timestamp DESC LIMIT 20;"
309.     )
310.     r = requests.post(
311.         self.SURREAL_URL, data=query,
312.         headers=self._HEADERS, timeout=5,
313.     )
314.     if r.status_code != 200:
```

```
315.         return []
316.     rows = r.json()[0].get("result", [])
317.     # φιλτράρισμα στην Python για χρονική σύγκριση
318.     return [row for row in rows if row.get("timestamp", "") >= since_iso]
319.
320. # — Σενάριο I: Υπερβολική Κατανάλωση —————
321.
322. def test_I_high_load_alert_written_to_db(self):
323.     """
324.     Σενάριο I: Υπερβολική Κατανάλωση – Ποή MQTT → Node-RED → SurrealDB
325.     Δημοσιεύει μήνυμα με load_power_kw=75 (όριο 60).
326.     Αναμενόμενο: εγγραφή HIGH_LOAD στον πίνακα alerts της SurrealDB.
327.     """
328.     ts = self._publish("test_m1", {"load_power_kw": 75.0, "grid_import_kw": 30.0})
329.     time.sleep(self.WAIT_SEC)
330.
331.     alerts = self._recent_alerts("HIGH_LOAD", ts)
332.     self.assertGreater(
333.         len(alerts), 0,
334.         "Δεν βρέθηκε HIGH_LOAD alert στη SurrealDB μετά από MQTT publish"
335.     )
336.     self.assertEqual(alerts[0]["severity"], "high")
337.
338. # — Σενάριο II: Χαμηλή Μπαταρία —————
339.
340. def test_II_low_battery_alert_written_to_db(self):
341.     """
342.     Σενάριο II: Κρίσιμη Στάθμη Μπαταρίας – Ποή MQTT → Node-RED → SurrealDB
343.     Δημοσιεύει μήνυμα με battery_soc=12% (όριο 20%).
344.     Αναμενόμενο: εγγραφή LOW_BATTERY_SOC στον πίνακα alerts.
345.     """
346.     ts = self._publish("test_m2", {"battery_soc": 12.0})
347.     time.sleep(self.WAIT_SEC)
348.
349.     alerts = self._recent_alerts("LOW_BATTERY_SOC", ts)
350.     self.assertGreater(
351.         len(alerts), 0,
352.         "Δεν βρέθηκε LOW_BATTERY_SOC alert στη SurrealDB"
353.     )
354.
355. # — Σενάριο III: Κανονική Λειτουργία – Χωρίς Εγγραφή Alert —————
356.
357. def test_III_normal_operation_no_alert_in_db(self):
358.     """
359.     Σενάριο III: Κανονική Λειτουργία – Επαλήθευση Απουσίας False Positive
360.     Δημοσιεύει μήνυμα με τιμές εντός ορίων.
361.     Αναμενόμενο: ΔΕΝ δημιουργείται HIGH_LOAD alert για αυτό το meter_id.
362.     """
363.     ts = self._publish("test_m3", {
364.         "load_power_kw": 30.0,
365.         "grid_import_kw": 8.0,
366.         "battery_soc": 70.0,
367.     })
```

```
368.         time.sleep(self.WAIT_SEC)
369.
370.         # Ελέγχουμε ότι δεν δημιουργήθηκε HIGH_LOAD για το συγκεκριμένο meter
371.         query = (
372.             "SELECT * FROM alerts "
373.             "WHERE alert_type = 'HIGH_LOAD' "
374.             "AND meter_id = 'test_m3' "
375.             "ORDER BY timestamp DESC LIMIT 5;"
376.         )
377.         r = requests.post(
378.             self.SURREAL_URL, data=query,
379.             headers=self._HEADERS, timeout=5,
380.         )
381.         rows = r.json()[0].get("result", []) if r.status_code == 200 else []
382.         recent = [row for row in rows if row.get("timestamp", "") >= ts]
383.         self.assertEqual(
384.             len(recent), 0,
385.             "False positive: HIGH_LOAD alert δημιουργήθηκε για κανονική λειτουργία"
386.         )
387.
388.         # — Integration Tests: Microservice Πρόβλεψης —————
389.
390.         def api_available() -> bool:
391.             try:
392.                 r = requests.get("http://localhost:5000/health", timeout=3)
393.                 return r.status_code == 200
394.             except requests.exceptions.ConnectionError:
395.                 return False
396.
397.         class TestPredictAPI(unittest.TestCase):
398.             """
399.             Σενάρια δοκιμών για το REST API πρόβλεψης φωτοβολταϊκής παραγωγής.
400.             Απαιτούν εκτελούμενο το python-analytics container (port 5000).
401.             """
402.
403.             @classmethod
404.             def setUpClass(cls):
405.                 if not api_available():
406.                     raise unittest.SkipTest(
407.                         "To API δεν είναι διαθέσιμο στο localhost:5000. "
408.                         "Εκκινήστε το σύστημα με: docker compose up python-analytics"
409.                     )
410.
411.             def _post(self, payload: dict) -> dict:
412.                 r = requests.post(API_URL, json=payload, timeout=5)
413.                 self.assertEqual(r.status_code, 200, f"HTTP error: {r.text}")
414.                 return r.json()
415.
416.         # — Σενάριο A: Κανονική Λειτουργία —————
417.
418.         def test_A_normal_operation_no_anomaly(self):
419.             """
420.             Σενάριο A: Κανονική Λειτουργία PV
```

```
421.         Είσοδος: Μέτρια ακτινοβολία, κανονική παραγωγή.  
422.         Αναμενόμενο: anomaly=False.  
423.         """  
424.         payload = {  
425.             "meter_id": "m1",  
426.             "timestamp": "2026-05-01T12:00:00Z",  
427.             "solar_irradiance_wm2": 600.0,  
428.             "temperature_c": 25.0,  
429.             "pv_power_kw": 15.0,  
430.         }  
431.         result = self._post(payload)  
432.  
433.         self.assertFalse(  
434.             result["anomaly"],  
435.             f"Αναμενόταν κανονική λειτουργία, σφάλμα: {result['error_pct']:.1f}%"  
436.         )  
437.         self.assertEqual(result["anomaly_type"], "NORMAL")  
438.  
439.     # — Σενάριο B: Αστοχία Inverter (Υψηλή Απόκλιση) —————  
440.  
441.     def test_B_inverter_failure_high_deviation(self):  
442.         """  
443.         Σενάριο B: Τεχνική Αστοχία Inverter  
444.         Είσοδος: Υψηλή ακτινοβολία (850 W/m2), πραγματική παραγωγή 5 kW  
445.         (αναμενόμενη ~21 kW).  
446.         Αναμενόμενο: anomaly=True, severity=high, error_pct > 30%.  
447.         """  
448.         payload = {  
449.             "meter_id": "m1",  
450.             "timestamp": "2026-05-01T13:00:00Z",  
451.             "solar_irradiance_wm2": 850.0,  
452.             "temperature_c": 25.0,  
453.             "pv_power_kw": 5.0,  
454.         }  
455.         result = self._post(payload)  
456.  
457.         self.assertTrue(result["anomaly"], "Αναμενόταν ανωμαλία (inverter failure)")  
458.         self.assertGreater(result["error_pct"], 30.0)  
459.         self.assertEqual(result["severity"], "high")  
460.  
461.     # — Σενάριο Γ: Μερική Απώλεια Αποδοτικότητας —————  
462.  
463.     def test_C_partial_efficiency_loss_medium_severity(self):  
464.         """  
465.         Σενάριο Γ: Μερική Απώλεια Αποδοτικότητας (π.χ. ρύπανση πάνελ)  
466.         Είσοδος: Παραγωγή ~80% της αναμενόμενης (απόκλιση ~20%).  
467.         Αναμενόμενο: anomaly=True, severity=medium.  
468.         """  
469.         payload = {  
470.             "meter_id": "m1",  
471.             "timestamp": "2026-05-01T11:00:00Z",  
472.             "solar_irradiance_wm2": 700.0,  
473.             "temperature_c": 28.0,
```

```
474.         "pv_power_kw": 14.0,
475.     }
476.     result = self._post(payload)
477.
478.     # Ελέγχουμε μόνο αν ανιχνεύθηκε ανωμαλία (εξαρτάται από το μοντέλο)
479.     if result["anomaly"]:
480.         self.assertIn(result["severity"], ["medium", "high"])
481.     # Αν δεν είναι anomaly, η απόκλιση είναι εντός ορίων – αποδεκτό
482.     self.assertIn("error_pct", result)
483.
484.     # — Σενάριο Δ: Νυχτερινή Λειτουργία (Μηδενική Παραγωγή) —————
485.
486.     def test_D_night_mode_zero_pv_no_false_positive(self):
487.         """
488.         Σενάριο Δ: Νυχτερινή Λειτουργία
489.         Είσοδος: Μηδενική ακτινοβολία, μηδενική παραγωγή PV.
490.         Αναμενόμενο: Χωρίς σφάλμα διαίρεσης με μηδέν, απόκριση HTTP 200.
491.         """
492.         payload = {
493.             "meter_id": "m1",
494.             "timestamp": "2026-05-01T02:00:00Z",
495.             "solar_irradiance_wm2": 0.0,
496.             "temperature_c": 18.0,
497.             "pv_power_kw": 0.0,
498.         }
499.         result = self._post(payload)
500.
501.         # Το API δεν πρέπει να κρασάρει – αρκεί να επιστρέψει έγκυρο JSON
502.         self.assertIn("predicted_value", result)
503.         self.assertIn("error_pct", result)
504.         self.assertGreaterEqual(result["error_pct"], 0.0)
505.
506.     # — Σενάριο Ε: Ελλιπή Δεδομένα —————
507.
508.     def test_E_missing_temperature_field_uses_default(self):
509.         """
510.         Σενάριο Ε: Ελλιπή Δεδομένα (Απουσία Πεδίου temperature_c)
511.         Είσοδος: JSON χωρίς το πεδίο temperature_c.
512.         Αναμενόμενο: Το API χειρίζεται gracefully με default τιμή,
513.         επιστρέφει HTTP 200.
514.         """
515.         payload = {
516.             "meter_id": "m1",
517.             "timestamp": "2026-05-01T10:00:00Z",
518.             "solar_irradiance_wm2": 500.0,
519.             "pv_power_kw": 12.0,
520.             # temperature_c λείπει
521.         }
522.         result = self._post(payload)
523.         self.assertIn("predicted_value", result)
524.
525.     # — Σενάριο ΣΤ: Ακραίες Συνθήκες Ηλιοφάνειας —————
526.
```

```
527. def test_F_peak_irradiance_full_production(self):
528.     """
529.     Σενάριο ΣΤ: Κορυφή Ηλιακής Παραγωγής
530.     Είσοδος: Μέγιστη ακτινοβολία (1000 W/m²), πλήρης παραγωγή.
531.     Αναμενόμενο: anomaly=False – το σύστημα λειτουργεί στο μέγιστο.
532.     """
533.     payload = {
534.         "meter_id": "m1",
535.         "timestamp": "2026-05-01T13:30:00Z",
536.         "solar_irradiance_wm2": 1000.0,
537.         "temperature_c": 35.0,
538.         "pv_power_kw": 25.0,
539.     }
540.     result = self._post(payload)
541.     self.assertFalse(
542.         result["anomaly"],
543.         f"False positive σε πλήρη παραγωγή: error={result['error_pct']:.1f}%"
544.     )
545.
546. # — Εκτέλεση —————
547.
548. if __name__ == "__main__":
549.     print("=" * 65)
550.     print("EMS - Δοκιμές Λειτουργικών Σεναρίων")
551.     print("=" * 65)
552.
553.     loader = unittest.TestLoader()
554.     suite = unittest.TestSuite()
555.
556.     suite.addTests(loader.loadTestsFromTestCase(TestAlertLogic))
557.     suite.addTests(loader.loadTestsFromTestCase(TestAlertIntegration))
558.     suite.addTests(loader.loadTestsFromTestCase(TestPredictAPI))
559.
560.     runner = unittest.TextTestRunner(verbosity=2)
561.     runner.run(suite)
562.
```

Παράρτημα Ζ: Ερωτηματολόγιο χρηστών

ΕΡΩΤΗΜΑΤΟΛΟΓΙΟ ΑΞΙΟΛΟΓΗΣΗΣ ΧΡΗΣΤΩΝ

Σύστημα Παρακολούθησης και Πρόβλεψης Ενεργειακής Απόδοσης (IEMS)

Διπλωματική Εργασία — Ελένη Τσιανάκα, ΕΑΠ 2026

Οδηγίες Συμπλήρωσης

Το παρόν ερωτηματολόγιο αφορά στην αξιολόγηση ενός Ευφυούς Συστήματος Ενεργειακής Παρακολούθησης (IEMS) που αναπτύχθηκε στο πλαίσιο διπλωματικής εργασίας. Σας παρακαλούμε να αφιερώσετε 10–15 λεπτά για τη συμπλήρωσή του. Τα αποτελέσματα θα χρησιμοποιηθούν αποκλειστικά για ακαδημαϊκούς σκοπούς.

Για κάθε ερώτηση επιλέξτε τον βαθμό συμφωνίας σας (1 = Διαφωνώ απόλυτα, 5 = Συμφωνώ απόλυτα).

Στοιχεία Συμμετέχοντα

Ρόλος στον οργανισμό:	☐ Διευθυντής / Στέλεχος Διοίκησης ☐ Διαχειριστής Εγκατάστασης ☐ Τεχνικός	Εμπειρία:	☐ <2 έτη ☐ 2–5 έτη ☐ >5 έτη
------------------------------	---	------------------	--

Ερωτήσεις Αξιολόγησης (Κλίμακα Likert 1–5)

A/A	Ερώτηση	1	2	3	4	5
		Διαφωνώ απόλυτα	Διαφωνώ	Ούτε Συμφωνώ / Ούτε Διαφωνώ	Συμφωνώ	Συμφωνώ απόλυτα

Α. Χρηστικότητα και Κατανόηση						
1	Το σύστημα ήταν εύκολο στη χρήση	☐	☐	☐	☐	☐
2	Η πλοήγηση στα dashboards ήταν κατανοητή	☐	☐	☐	☐	☐
3	Οι πληροφορίες παρουσιάζονταν με σαφή τρόπο	☐	☐	☐	☐	☐
4	Οι λειτουργίες του συστήματος ήταν εύκολο να εντοπιστούν	☐	☐	☐	☐	☐
Β. Λειτουργικότητα και Ανταπόκριση στις Ανάγκες						
5	Τα dashboards παρέχουν χρήσιμη πληροφόρηση	☐	☐	☐	☐	☐
6	Οι δείκτες KPI βοηθούν στη λήψη αποφάσεων	☐	☐	☐	☐	☐
7	Οι ειδοποιήσεις θεωρούνται χρήσιμες	☐	☐	☐	☐	☐
8	Το σύστημα βοηθά στην παρακολούθηση της ενεργειακής απόδοσης	☐	☐	☐	☐	☐
Γ. Αξιοπιστία και Εμπιστοσύνη						
9	Τα δεδομένα που εμφανίζει το σύστημα φαίνονται αξιόπιστα και ακριβή.	☐	☐	☐	☐	☐

10	Το σύστημα ανταποκρίνεται γρήγορα στις ενέργειές μου (χωρίς καθυστερήσεις).	☐	☐	☐	☐	☐
11	Θα εμπιστευόμουν τις ειδοποιήσεις του συστήματος για να λάβω επαγγελματικές αποφάσεις.	☐	☐	☐	☐	☐
Δ. Γενική Ικανοποίηση						
12	Είμαι ικανοποιημένος από το σύστημα	☐	☐	☐	☐	☐
13	Θα χρησιμοποιούσα αυτό το σύστημα στην καθημερινή εργασία μου.	☐	☐	☐	☐	☐
14	Θα πρότεινα αυτό το σύστημα σε συναδέλφους ή άλλες επιχειρήσεις.	☐	☐	☐	☐	☐

Σας ευχαριστούμε για τη συμμετοχή σας.

Υπεύθυνη Δήλωση Συγγραφέα:

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν.1599/1986, η παρούσα εργασία αποτελεί αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλει κάθε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχει έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης.