Έγγραφο απαιτήσεων λογισμικού (SRS)

ΠΡΟΣΑΡΜΟΓΗ ΤΟΥ ΑΝΤΙΣΤΟΙΧΟΥ ΕΓΓΡΑΦΟΥ ΤΟΥ ΠΡΟΤΥΠΟΥ ISO/IEC/IEEE 29148:2011

*elephntclt*

# Εισαγωγή

## 1.1 Εισαγωγή: σκοπός του λογισμικού

Η εργασία αφορά τη διαχείριση δεδομένων της αγοράς ηλεκτρικής ενέργειας στην Ευρώπη, τα οποία διατίθενται ως ανοικτά δεδομένα (Open Data) από το δικτυακό τόπο transparency.entsoe.eu. Τα δεδομένα αυτά είναι κρίσιμα για τη διαφάνεια στη λειτουργία της αγοράς ηλεκτρικής ενέργειας και η διάθεση, οπτικοποίηση και ανάλυσή τους είναι απαραίτητες υπηρεσίες προς τους συμμετέχοντες στην αγορά, ανεξάρτητα από το ρόλο τους (παραγωγοί, πωλητές, συνεταιρισμοί κ.ά.).

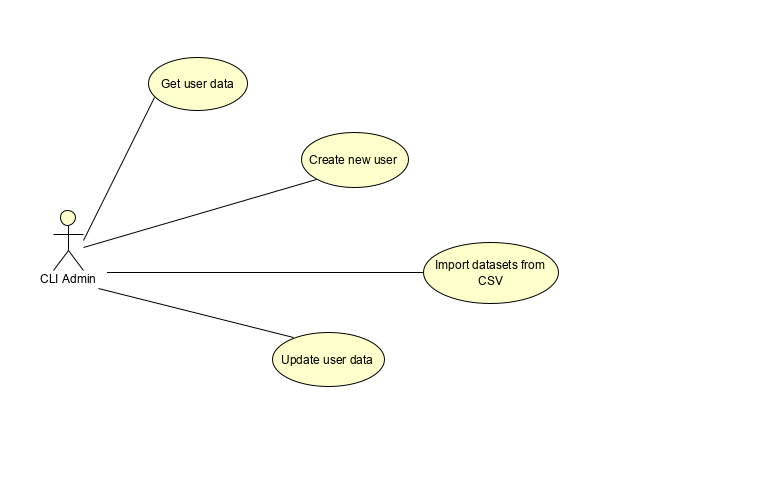
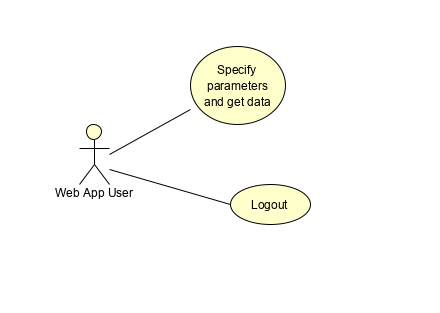
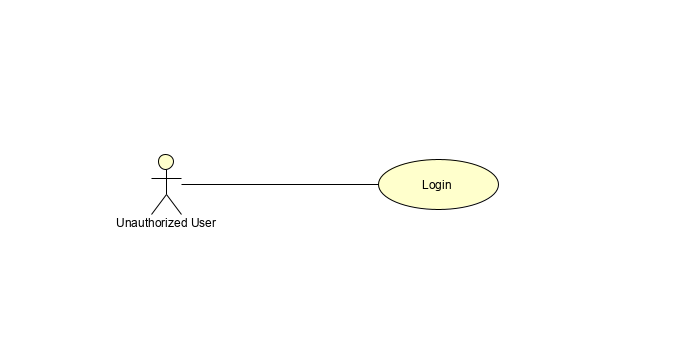
Η εμβέλεια της εργασίας περιορίζεται σε 3 επιλεγμένα datasets, συγκεκριμένα τα Actual Total Load (η πραγματική ενέργεια που καταναλώθηκε (φορτίο) σε μια χωρική εμβέλεια στην οποία αναφέρεται), Day Ahead Total Load Forecast (μια πρόβλεψη για τη συνολική ζήτηση σε ενέργεια κατά την επόμενη μέρα, σε μια χωρική εμβέλεια στην οποία αναφέρεται) και Aggregated Generation Per Type (ανάλυση της πραγματικής παραγωγής ενέργειας στη χώρα ή ζώνη στην οποία αναφέρεται, ανάλογα με τον τρόπο παραγωγής).

Στόχος υπήρξε η ανάπτυξη ενός συστήματος λογισμικού για την εισαγωγή επιλεγμένων συνόλων δεδομένων αγοράς ηλεκτρικής ενέργειας που διατίθενται από το transparency.entsoe.eu σε μία ΒΔ (data import) και τη διάθεσή τους, με ελεγχόμενη πρόσβαση, μέσω ενός RESTful Web API.

### 1.2.1 Διεπαφές με εξωτερικά συστήματα

Δεν υπάρχουν.

### 1.2.2 Διεπαφές με το χρήστη

Το παρόν έργο λογισμικού περιλαμβάνει 2 διεπαφές με το χρήστη, ένα Web App, για πρόσβαση μέσω browser και ένα custom Command Line Interface (CLI), για χρήση μέσω τερματικού.

# Αναφορές - πηγές πληροφοριών

Αναφορά σε πηγές πληροφοριών στο μέτρο της αναγκαιότητας για την κατανόηση του συστήματος

ΝΑ ΜΗΝ ΣΥΜΠΛΗΡΩΘΕΙ ΑΝ ΔΕΝ ΑΠΑΙΤΕΙΤΑΙ

# Προδιαγραφές απαιτήσεων λογισμικού

## 3.1 Περιπτώσεις χρήσης

### 3.1.1 ΠΕΡΙΠΤΩΣΗ ΧΡΗΣΗΣ 1: Λήψη δεδομένων ενέργειας μέσω Web App

#### 3.1.1.1 Χρήστες (ρόλοι) που εμπλέκονται

Ο διαπιστευμένος (Logged in) χρήστης του Web app.

#### 3.1.1.2 Προϋποθέσεις εκτέλεσης

Αρχικά, ο χρήστης θα πρέπει να είναι authorized ή/και τα credentials του να είναι σωστά προκειμένου να μπορεί να γίνει η κλήση στο API. Επίσης, πρέπει ο χρήστης να έχει επαρκή αριθμό quotas στην κατοχή του. Τέλος, απαιτείται η υποβολή παραμέτρων οι οποίες συμφωνούν με την μορφή που αναμένει το API.

#### 3.1.1.3 Περιβάλλον εκτέλεσης

Η συγκεκριμένη περίπτωση χρήσης απευθύνεται στο χρήστη του Web app, μέσω κάποιου browser.

#### 3.1.1.4 Δεδομένα εισόδου

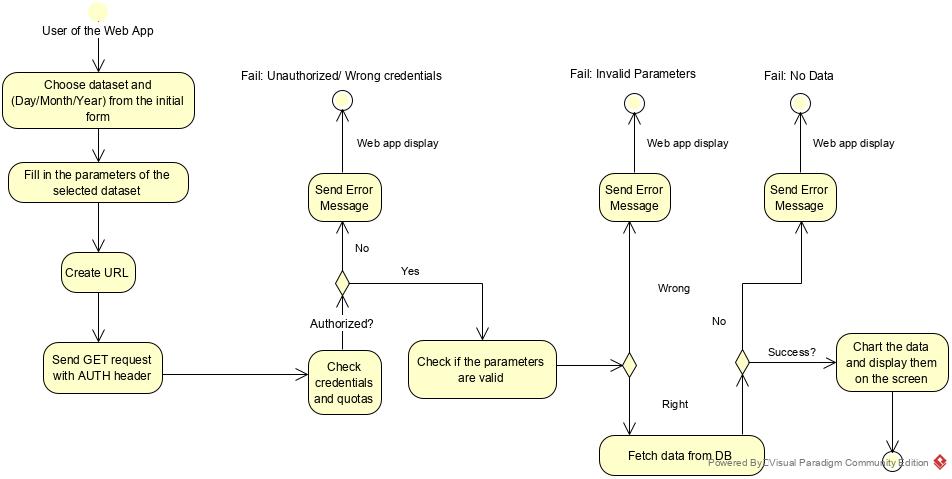
Τα δεδομένα εισόδου που παρέχει ο χρήστης είναι:

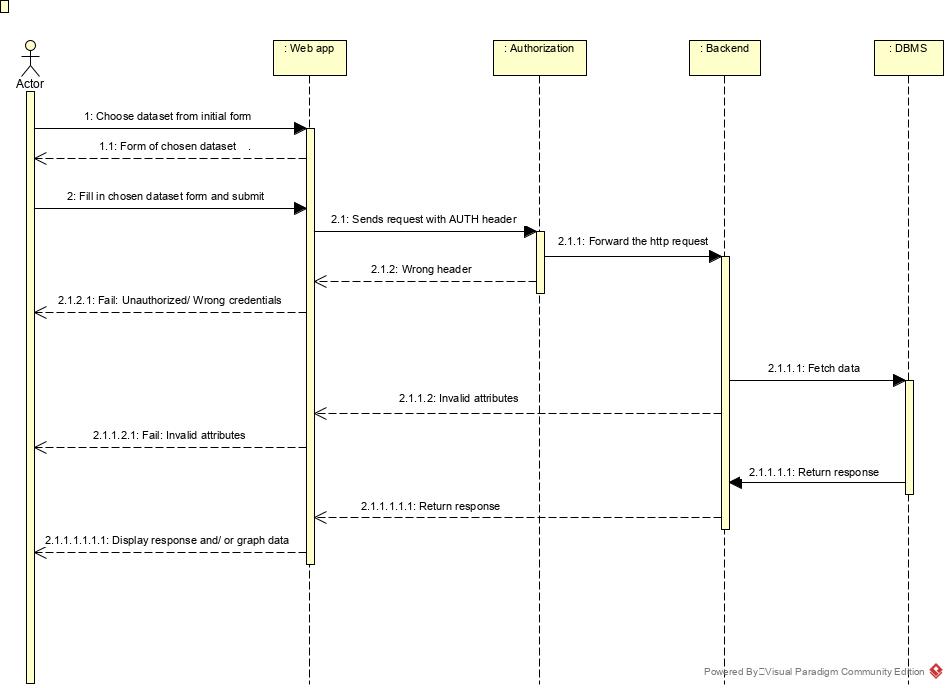
* Το dataset το οποίο επιθυμούμε να εμφανιστεί.
* Το custom header που περιλαμβάνει τα αντίστοιχα credentials.
* Area name: Το όνομα της χώρας στην οποία αφορούν τα δεδομένα.
* Resolution Code: Ο κωδικός χρονικής ανάλυσης.
* Production Type: Ο τίτλος του τύπου παραγωγής.
* Year: Το έτος.
* Month: Ο μήνας.
* Day: Η μέρα.

Όλα τα δεδομένα εισόδου από το Resolution Code μέχρι το Day παρουσιάζονται μέσω λίστας στον χρήστη για την αποφυγή λανθασμένης εισόδου.

#### 3.1.1.5 Αλληλουχία ενεργειών - επιθυμητή συμπεριφορά

* Βήμα 1: Επιλογή dataset από την σχετική λίστα που δίνεται από το interface.
* Βήμα 2: Συμπλήρωση των παραμέτρων του επιλεγμένου dataset σύμφωνα με την νέα φόρμα που εμφανίζεται στον χρήστη.
* Βήμα 3: Δημιουργία του URL στο οποίο θα γίνει η κλήση, από το Web app.
* Βήμα 4: Πραγματοποίηση του GET request από το Web app στο API παρέχοντας και τα credentials του εν ενεργεία χρήστη.
* Βήμα 5: Έλεγχος credentials και αριθμού quotas από το Backend.
* Βήμα 6: Αν τα παρεχόμενα credentials είναι σωστά και απομένουν quotas στον χρήστη, έλεγχος για εγκυρότητα παραμέτρων, αλλιώς εμφάνιση κατάλληλου μηνύματος λάθους στον χρήστη.
* Βήμα 7: Αν ο χρήστης έχει υποβάλει έγκυρες παραμέτρους, ανάκτηση δεδομένων από τη βάση, αλλιώς εμφάνιση κατάλληλου μηνύματος λάθους στον χρήστη.
* Βήμα 8: Αν βρεθούν τα απαιτούμενα δεδομένα στη βάση, επιστροφή τους στο Web app, μορφοποίηση του σε μορφή διαγράμματος και εμφάνιση τους στον χρήστη, αλλιώς εμφάνιση κατάλληλου μηνύματος λάθους.

Παρακάτω φαίνονται τα αντίστοιχα UML διαγράμματα (Activity, Sequence) που περιγράφουν τις προηγούμενες λειτουργίες:



#### 3.1.1.7 Δεδομένα εξόδου

Τα δεδομένα εξόδου που παρέχονται στον χρήστη είναι:

* Οι φόρμες υποβολής του dataset και των παραμέτρων του.
* Τα μηνύματα λάθους που μπορεί να προκύψουν (“Fail: Unauthorized/ Wrong credentials” στο Βήμα 6, “Fail: Invalid Parameters” στο Βήμα 7 και “Fail: No Data” στο Βήμα 8).
* Τα δεδομένα που ζήτησε ο χρήστης σε μορφή γραφήματος.

3.1.2 ΠΕΡΙΠΤΩΣΗ ΧΡΗΣΗΣ 2: Δημιουργία νέου χρήστη (μέσω CLI)

*3.1.2.1 Χρήστες (ρόλοι) που εμπλέκονται*

Η συγκεκριμένη περίπτωση χρήσης αφορά τον διαχειριστή του συστήματος λογισμικού (admin) και όχι απλούς ή μια διαπιστευμένους χρήστες.

*3.1.2.2 Προϋποθέσεις εκτέλεσης*

Προκειμένου να μπορέσει ο διαχειριστής του συστήματος να δημιουργήσει έναν νέο χρήστη είναι απαραίτητο να έχει εισέλθει στο σύστημα με τα στοιχεία του (Login), ώστε να έχει διαπιστευθεί η ταυτότητα του.

*3.1.2.3 Περιβάλλον εκτέλεσης*

Η συγκεκριμένη περίπτωση χρήσης εκτελείται στο command line, μέσω του CLI εργαλείου.

*3.1.2.4 Δεδομένα εισόδου*

Κατά την σύνταξη της εντολής απαιτείται από τον διαχειριστή ο προσδιορισμός του SCOPE, που στη συγκεκριμένη περίπτωση είναι Admin.

*3.1.2.5 Παράμετροι*

Ο διαχειριστής είναι υποχρεωμένος, κατά την σύνταξη της εντολής, να προσδιορίζει τα στοιχεία του νέου χρήστη που δημιουργεί. Απαιτείται να εισάγει τις παραμέτρους:

* --newuser με τιμή το όνομα νέου χρήστη
* --password με τιμή τον κωδικό πρόσβασης χρήστη
* --email με τιμή την διεύθυνση ηλεκτρονικού ταχυδρομείου χρήστη
* --quota με τιμή το πλήθος των επιτρεπτών κλήσεων ανά 24 ώρες

Προκειμένου να είναι έγκυρη η εντολή και να δημιουργηθεί νέος χρήστης θα πρέπει οι τιμές των παραμέτρων, που εισάγονται, να ικανοποιούν συγκεκριμένους περιορισμούς. Συγκεκριμένα, θα πρέπει το όνομα χρήστη να είναι αλφαριθμητικό με λατινικούς χαρακτήρες και μοναδικό, ο κωδικός πρόσβασης να μήν αποτελείται από κενά και η διεύθυνση ηλεκτρονικού ταχυδρομείου να είναι επίσης μοναδική για κάθε χρήστη.

*3.1.2.6 Αλληλουχία ενεργειών - επιθυμητή συμπεριφορά*

Βήμα 1: Ο διαχειριστής συντάσσει την εντολή δημιουργίας νέου χρήστη και προσδιορίζει τις κατάλληλες παραμέτρους.

Βήμα 2: Πραγματοποιείται έλεγχος για να διαπιστωθεί ότι δίνονται όλες οι απαιτούμενες παράμετροι για την δημιουργία ενός νέου χρήστη. Σε περίπτωση που εντοπισθεί πρόβλημα στις παραμέτρους εμφανίζεται κατάλληλο μήνυμα λάθους.

Βήμα 3: Στη συνέχεια, μέσω του commander σχηματίζεται το επιθυμητό URL για την αντίστοιχη της εντολής του CLI κλήση στο API.

Βήμα 4: Αποστέλλεται POST request στο API, με τα διαπιστευτήρια του χρήστη στο custom header X-OBSERVATORY-AUTH. Σε περίπτωση που το header δεν παρέχεται ή περιέχει εσφαλμένα διαπιστευτήρια εμφανίζεται κατάλληλο μήνυμα λάθους.

Βήμα 5: Ελέγχονται τα διαπιστευτήρια χρήστη και σε περίπτωση που το header δεν παρέχεται ή περιέχει εσφαλμένα διαπιστευτήρια εμφανίζεται κατάλληλο μήνυμα λάθους.

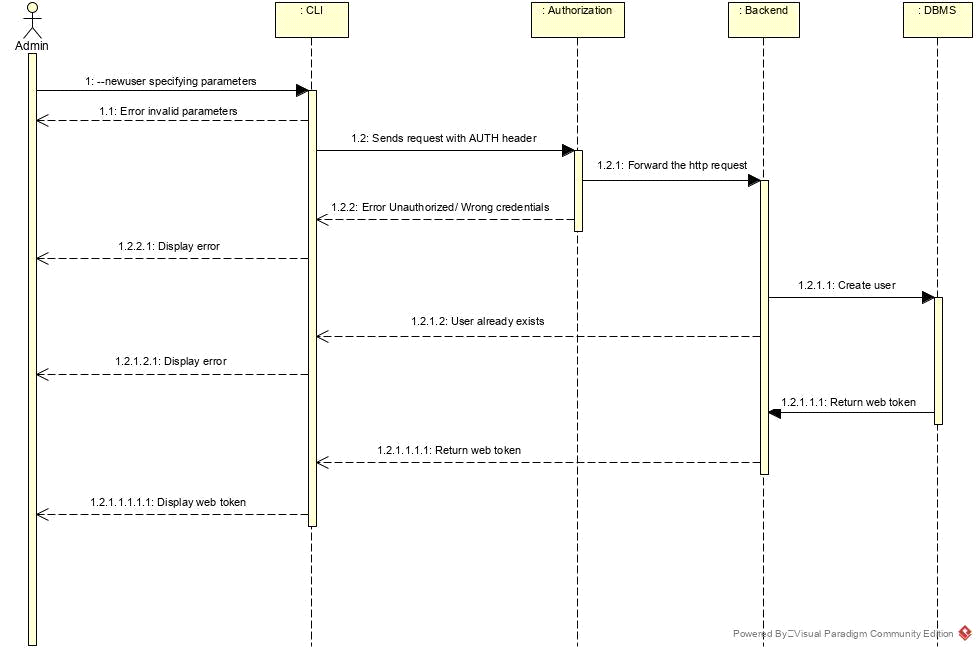
Βήμα 6: Ελέγχονται τα στοιχεία χρήστη που παρέχονται ώστε να διαπιστωθεί ότι δεν υπάρχει άλλος ήδη εγγεγραμμένος χρήστης με ίδιο username ή email. Σε περίπτωση που υπάρχει, εμφανίζεται κατάλληλο μήνυμα λάθους.

Βήμα 7: Τα στοιχεία του νέου χρήστη αποθηκεύονται στη βάση δεδομένων και με βάση αυτά δημιουργείται νέο μοναδικό token για την μελλοντική διαπίστευση του. Σε περίπτωση που προκύψει σφάλμα κατά την παραπάνω διαδικασία εμφανίζεται κατάλληλο μήνυμα λάθους.

Βήμα 8: Επιστρέφεται το αποτέλεσμα εκτέλεσης της εντολής και το token που δημιουργήθηκε σε προηγούμενα βήματα.

Παρατίθενται, ακολούθως, τα UML διαγράμματα Activity και Sequence:

### 



*3.1.2.7 Δεδομένα εξόδου*

Τα δεδομένα εξόδου που επιστρέφονται στο διαχειριστή είναι τα ακόλουθα:

* Το αποτέλεσμα εκτέλεσης της εντολής.
* Το token του νέου χρήστη που δημιουργήθηκε.

Αν κατά τη διάρκεια της εκτέλεσης προκύψει σφάλμα, εμφανίζεται στο χρήστη το αντίστοιχο μήνυμα λάθους:

* Invalid parameters- Βήμα 2
* No token provided (401 Unauthorized)/ Wrong credentials (400 Bad Request)- Βήμα 4
* User/Email already exists (400 Bad Request)- Βήμα 6
* User not created- Βήμα 7

### 3.1.3 ΠΕΡΙΠΤΩΣΗ ΧΡΗΣΗΣ 3: Εισαγωγή στη βάση δεδομένων από CSV (μέσω CLI)

#### 3.1.3.1 Χρήστες (ρόλοι) που εμπλέκονται

Η συγκεκριμένη περίπτωση χρήσης αφορά μόνο κάποιον διαχειριστή του συστήματος (admin) και όχι τους απλούς χρήστες.

#### 3.1.3.2 Προϋποθέσεις εκτέλεσης

Ο χρήστης θα πρέπει να είναι διαπιστευμένος ως διαχειριστής του συστήματος για να εκτελέσει τη συγκεκριμένη περίπτωση.

#### 3.1.3.3 Περιβάλλον εκτέλεσης

Η συγκεκριμένη περίπτωση χρήσης εκτελείται στο command line, μέσω του CLI εργαλείου.

#### 3.1.3.4 Δεδομένα εισόδου

* Το scope Admin και το header X-OBSERVATORY-AUTH που φέρει το token με τα credentials του.

Για να γίνει επιτρεπτή η εκτέλεση της εντολής θα πρέπει τα credentials να ανταποκρίνονται σε αυτά κάποιου χρήστη εγγεγραμμένου στη βάση, με δικαιώματα admin.

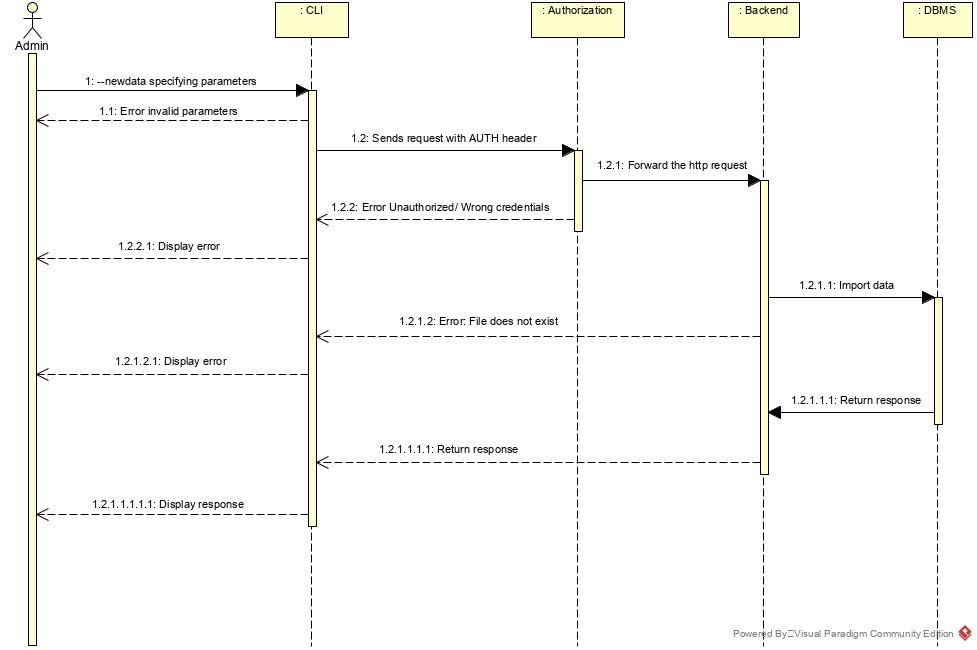
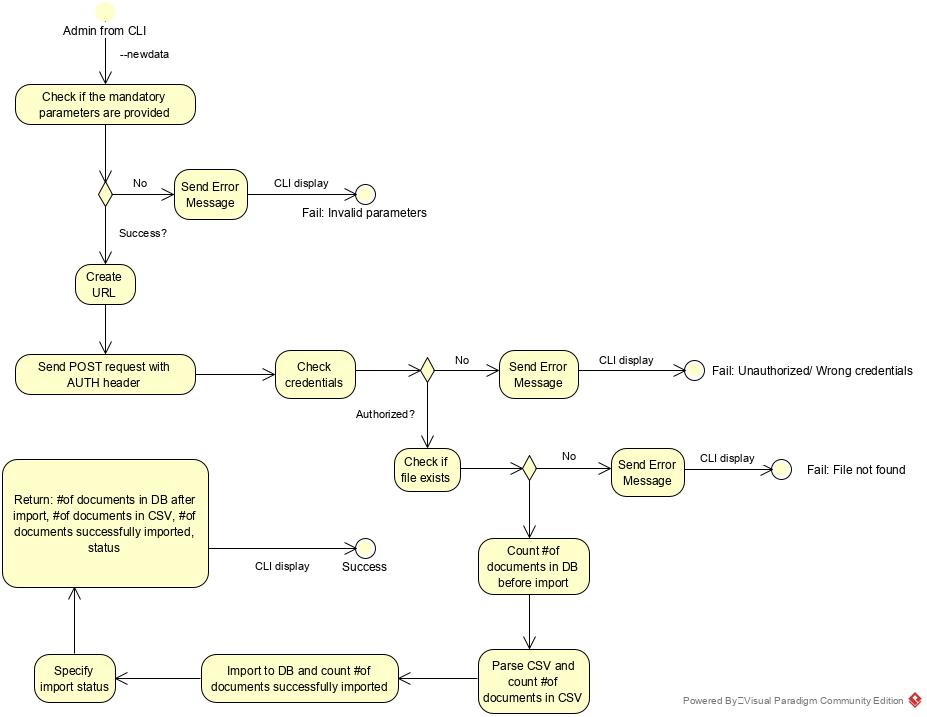
#### 3.1.3.5 Παράμετροι

* –newdata: Ορίζει την πραγματοποίηση του συγκεκριμένου use case και παίρνει ως παράμετρο το dataset που πρόκειται να γίνει import στη Βάση, μεταξύ των ActualTotalLoad, DayAheadTotalLoadForecast και AggregatedGenerationPerType.
* --source: Το πλήρες path για το αρχείο csv από το οποίο θα γίνει η συλλογή των νέων δεδομένων. Θα πρέπει να αποτελεί έγκυρο path προς κάποιο υπάρχον csv αρχείο.

#### 3.1.3.6 Αλληλουχία ενεργειών - επιθυμητή συμπεριφορά

* Βήμα 1: Ελέγχεται ότι όλες οι υποχρεωτικές παράμετροι και μόνο αυτές έχουν δοθεί. Αν όχι, εμφανίζεται κατάλληλο μήνυμα λάθους.
* Βήμα 2: Στη συνέχεια, μέσω του commander σχηματίζεται το επιθυμητό URL για την αντίστοιχη της εντολής του CLI κλήση στο API.
* Βήμα 3: Αποστέλλεται POST request στο API, με τα διαπιστευτήρια του χρήστη στο custom header X-OBSERVATORY-AUTH. Σε περίπτωση που το header δεν παρέχεται ή περιέχει εσφαλμένα διαπιστευτήρια εμφανίζεται κατάλληλο μήνυμα λάθους.
* Βήμα 4: Ελέγχεται ότι το δοθέν path είναι έγκυρο και ότι το αρχείο υπάρχει. Σε περίπτωση σφάλματος, επιστρέφεται κατάλληλο μήνυμα λάθους.
* Βήμα 5: Υπολογίζεται ο αριθμός των εγγραφών στη βάση.
* Βήμα 6: Γίνεται parse του csv και υπολογίζεται, ταυτόχρονα, ο αριθμός των εγγραφών που περιέχει.
* Βήμα 7: Τα δεδομένα που ανασύρθηκαν από το csv γίνονται μαζικά import στη Βάση Δεδομένων, ενώ υπολογίζεται ο αριθμός των εγγραφών που έγιναν επιτυχώς.
* Βήμα 8: Επιστρέφονται το status του import, ο αριθμός των εγγραφών στο csv, o αριθμός των επιτυχώς εισαχθέντων στη Βάση και ο συνολικός αριθμός εγγραφών στη Βάση μετά το import.

Παρατίθενται, ακολούθως, τα UML διαγράμματα Sequence και Activity:



#### 3.1.3.7 Δεδομένα εξόδου

Τα δεδομένα εξόδου που επιστρέφονται στο χρήστη είναι τα ακόλουθα:

* Ο αριθμός των εγγραφών που περιέχονταν στο csv.
* Ο αριθμός των εγγραφών που εισήχθησαν επιτυχώς στη Βάση.
* Ο συνολικός αριθμός εγγραφών στη Βάση μετά το import.

Αν κατά τη διάρκεια της εκτέλεσης προκύψει σφάλμα, εμφανίζεται στο χρήστη το αντίστοιχο μήνυμα λάθους:

* Invalid parameters- Βήμα 1
* No token provided (401 Unauthorized)/ Wrong credentials (400 Bad Request)- Βήμα 3
* Invalid path (400 Bad Request)- Βήμα 4

## 3.2 Απαιτήσεις επιδόσεων

Βάσει των μετρήσεων μας για αιτήματα σε βάσεις των 100.000 εγγραφών θεωρείται αποδεκτός χρόνος απόκρισης των 2 sec.

## 3.3 Απαιτήσεις οργάνωσης δεδομένων

### 3.3.1 Απαιτήσεις και περιορισμοί πρόσβασης σε δεδομένα

Για πρόσβαση στα δεδομένα, ο χρήστης θα πρέπει να είναι εγγεγραμμένος στη βάση και το token του να είναι έγκυρο. Για να ισχύουν τα παραπάνω απαιτείται να έχει προηγηθεί log in από τον χρήστη. Επιπλέον, ο χρήστης χρειάζεται να έχει επαρκή αριθμό quotas στην διάθεσή του.

## 3.4 Περιορισμοί σχεδίασης

Η ανάπτυξη του λογισμικού πραγματοποιήθηκε στο περιβάλλον του node.js.

Για τη διαχείριση των δεδομένων επιλέχθηκε η μη σχεσιακή MongoDB.

Χρησιμοποιήθηκε το framework Express για την υλοποίηση του REST API, αλλά και το Commander για το custom CLI.

Η υλοποίηση του Web application βασίστηκε στη χρήση της βιβλιοθήκης του React, ενώ η αποτύπωση των δεδομένων σε διαγράμματα πραγματοποιήθηκε με τη βοήθεια του Chart.

Όλα τα δεδομένα προέρχονται από προηγουμένως κατεβασμένα από το entso-e και τοπικά αποθηκευμένα αρχεία csv. Επομένως, η δομή των δεδομένων στη βάση υπαγορεύθηκε από αυτή του entso-e, ενώ το ίδιο συνέβη και με την ονοματολογία των διάφορων πεδίων.

## 3.5 Λοιπές απαιτήσεις

### 3.5.1 Απαιτήσεις διαθεσιμότητας λογισμικού

Δεν υπάρχει περιορισμός διαθεσιμότητας, δεδομένου, βέβαια, ότι είναι σε λειτουργία ο server.

### 3.5.2 Απαιτήσεις ασφάλειας

Για τη διασφάλιση της ακεραιότητας της εφαρμογής και της ιδιωτικότητας των δεδομένων υποστηρίζεται το πρωτόκολλο HTTPS για όλες τις χρηστικές και προγραμματιστικές διεπαφές μέσω self-signed certificate.

Για τα διαπιστευτήρια των χρηστών επιλέχθηκε κωδικοποίηση μέσω jwt και μεταφορά του σε custom header (X-OBSERVATORY-AUTH) κατά τις κλήσεις στο API.

Στην περίπτωση του login μέσω CLI, τα διαπιστευτήρια του χρήστη αποθηκεύονται τοπικά σε αρχείο στο μηχάνημα, το οποίο θεωρούμε ασφαλές.

### 3.5.3 Απαιτήσεις συντήρησης

Δεν υπάρχουν.