Έγγραφο απαιτήσεων λογισμικού (SRS)

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

[E-Charge]

# Εισαγωγή

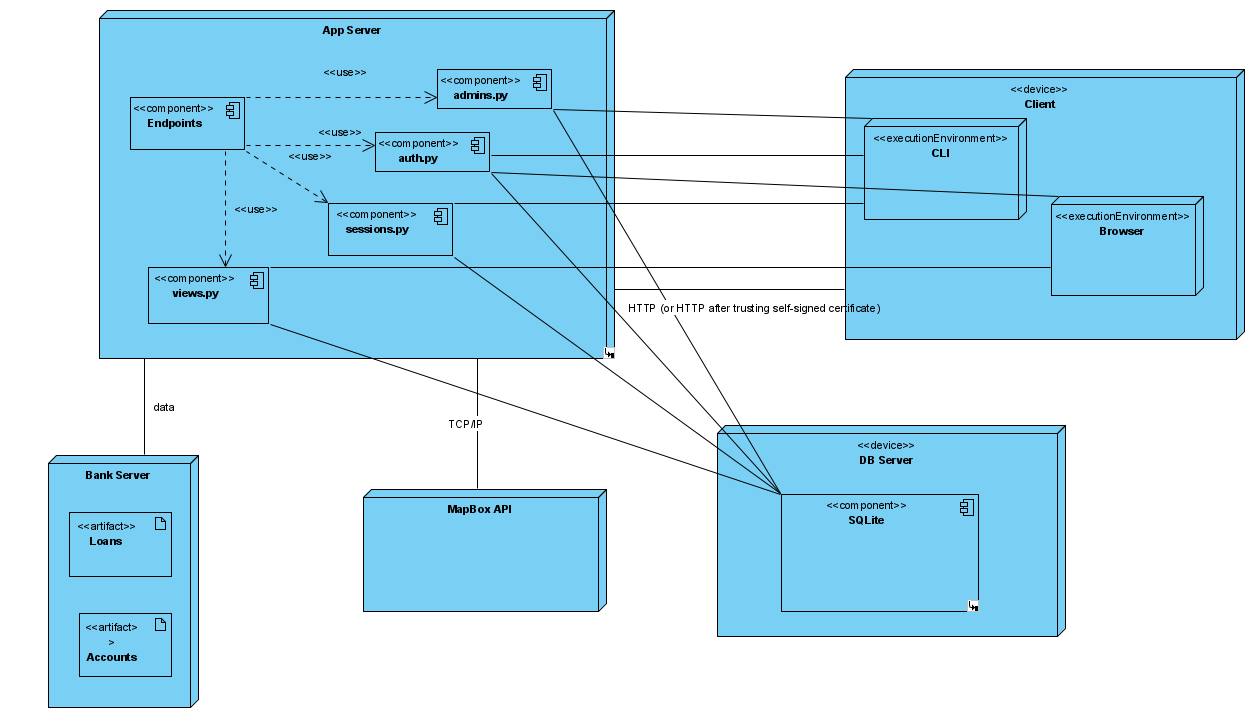
## 1.1 Εισαγωγή: σκοπός του λογισμικού

Σκοπός του λογισμικού είναι να πληροφορεί και να υποστηρίζει τον χρήστη της εφαρμογής σχετικά με τη διαδικασία φόρτισης του ηλεκτρικού του οχήματος, η οποία χαρακτηρίζεται από μεγάλη διάρκεια και ιδιαίτερες απαιτήσεις σε ηλεκτρική ισχύ από τους φορτιστές. Συγχρόνως, τα δεδομένα φόρτισης που προκύπτουν από τους χρήστες είναι διαθέσιμα προς επεξεργασία από διάφορους stakeholders.

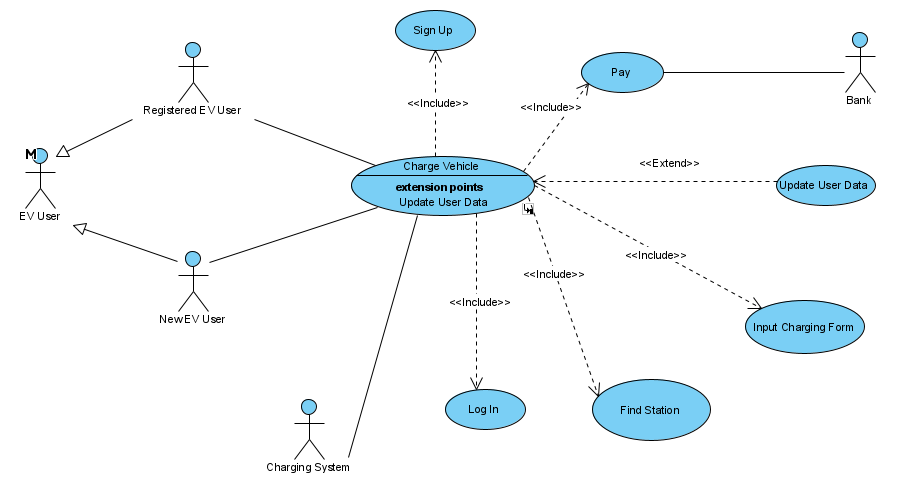
## 1.2 Διεπαφές (interfaces)

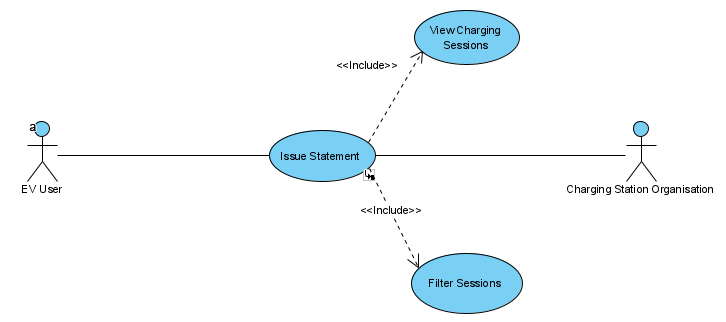
### 1.2.1 Διεπαφές με εξωτερικά συστήματα

Προδιαγραφή διεπαφών με εξωτερικά συστήματα και λογισμικό, με αναφορά σε πρότυπα ανταλλαγής δεδομένων και κλήσης υπηρεσιών.



### 1.2.2 Διεπαφές με το χρήστη

Διεπαφή για τον χρήστη αποτελεί ο Web Browser της επιλογής του, μέσω του οποίου συνδέεται στην εφαρμογή μας. Εκεί, μπορεί να αξιοποιήσει κάθε υπηρεσία που του προσφέρεται από το λογισμικό που σχεδιάσαμε. Παρακάτω, φαίνονται τα UML Use Case διαγράμματα που σχεδιάσαμε: 



# Αναφορές - πηγές πληροφοριών

# Προδιαγραφές απαιτήσεων λογισμικού

## 3.1 Περιπτώσεις χρήσης

### 3.1.1 ΠΕΡΙΠΤΩΣΗ ΧΡΗΣΗΣ 1: (Φόρτιση ηλεκτρικού οχήματος)

#### 3.1.1.1 Χρήστες (ρόλοι) που εμπλέκονται

Κάθε εγγεγραμμένος χρήστης μπορεί να αξιοποιήσει τις υπηρεσίες που προσφέρει η εφαρμογή μας. Συγχρόνως, όμως, κάθε μη εγγεγραμμένος χρήστης έχει τη δυνατότητα να εγγραφεί, και, στη συνέχεια, έχει στη διάθεση του τις υπηρεσίες που προορίζονται για τους εγγεγραμμένους χρήστες.

#### 3.1.1.2 Προϋποθέσεις εκτέλεσης

Αρχικά, ο χρήστης οφείλει να εγγραφεί συμπληρώνοντας τα απαραίτητα στοιχεία που απαιτούνται, και να είναι μοναδικά, δηλαδή να μην υπάρχουν πανομοιότυπα στη βάση δεδομένων της εφαρμογής μας, καθώς αποκλείουμε τα διπλότυπα. Ύστερα, εφόσον διαθέτει ηλεκτροκίνητο όχημα, για το οποίο υποστηρίζεται η φόρτιση από τον σταθμό, μπορεί να φορτίσει επιτυχώς το όχημα του. Τέλος, οφείλει να είναι συνδεδεμένος στο Διαδίκτυο, ώστε να έχει πρόσβαση στην εφαρμογή μας.

#### 3.1.1.3 Περιβάλλον εκτέλεσης

Ο χρήστης, εάν επιθυμεί να εγγραφεί, συμπληρώνει τα απαιτούμενα στοιχεία μέσω της διαδικτυακής διεπαφής χρήστη, με την οποία συνδέεται με την εφαρμογή μας. Ύστερα, τα στοιχεία στέλνονται στον DBMS, όπου είτε υπάρχουν ήδη και αποστέλλεται στον χρήστη μήνυμα για αποτυχία της εγγραφής, είτε δεν υπάρχουν, οπότε αποστέλλεται μήνυμα επιτυχημένης εγγραφής και τα στοιχεία αποθηκεύονται στη βάση δεδομένων μας Ύστερα, δεδομένου του μοντέλου του αμαξιού, το REST API αναζητά στη βάση δεδομένων μας και επιστρέφει τα διαθέσιμα προγράμματα φόρτισης στη διαδικτυακή διεπαφή χρήστη. Ύστερα, κατά τη διάρκεια της φόρτισης, αφού ο χρήστης επιλέξει πρόγραμμα φόρτισης (ενέργεια και ρυθμό φόρτισης), στη διαδικτυακή διεπαφή χρήστη εμφανίζονται δεδομένα σχετικά με τη φόρτιση που επιτελείται.

#### 3.1.1.4 Δεδομένα εισόδου

Ο χρήστης, είτε συμπληρώνει το username και το password για να κάνει log in, όντας εγγεγραμμένος, είτε συμπληρώνει όλα τα απαιτούμενα στοιχεία για να επιτευχθεί η εγγραφή του. Στη συνέχεια, καλείται να επιλέξει την ενέργεια και το ρυθμό φόρτισης που επιθυμεί και εμφανίζονται στη διαδικτυακή διεπαφή χρήστη.

#### 3.1.1.5 Αλληλουχία ενεργειών - επιθυμητή συμπεριφορά

Βήμα 1: Ο χρήστης επισκέπτεται τη σελίδα της εφαρμογής μας.

Βήμα 2: α) Αν ο χρήστης είναι ήδη εγγεγραμμένος στη βάση δεδομένων μας, τότε αρκεί να κάνει log in με το username και το password που του αντιστοιχούν.

β) Αν ο χρήστης δεν έχει εγγραφεί, συμπληρώνει τα απαραίτητα στοιχεία, ώστε να προστεθεί στη βάση δεδομένων μας ως νέος χρήστης, δηλαδή να επιτευχθεί η εγγραφή του. Αν η εγγραφή απέτυχε, λόγω ύπαρξης των στοιχείων που εισήχθησαν στη βάση δεδομένων μας, ο χρήστης ενημερώνεται με μήνυμα, ώστε να επαναλάβει το βήμα 2.

Βήμα 3: Μετά το προηγούμενο βήμα, ο χρήστης οδηγείται στην αρχική σελίδα ως συνδεδεμένος χρήστης. Πιθανώς, επιλέγει να βρει τον προτιμότερο σταθμό σε αυτόν από τον χάρτη συγκρίνοντας κοντινούς του σταθμούς.

Βήμα 4: Έπειτα αφού φτάνει στον σταθμό φόρτισης, συμπληρώνει τη φόρμα με τα στοιχεία οχήματος, επιθυμητή ενέργεια, ρυθμό φόρτισης και τρόπο πληρωμής.

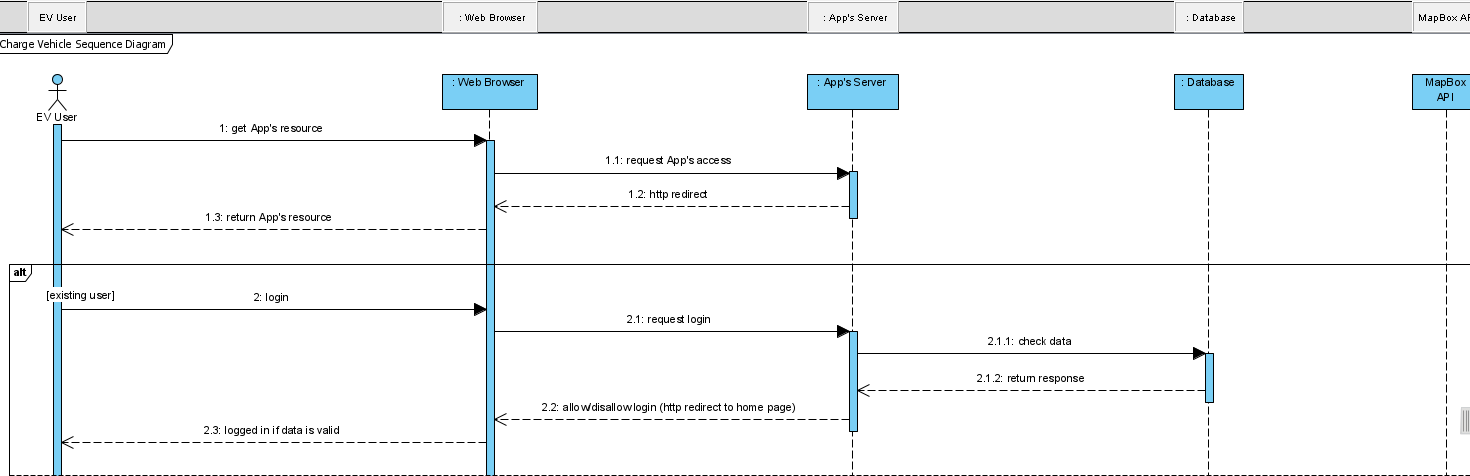
Βήμα 5: Ο χρήστης ξεκινάει τη διαδικασία φόρτισης με το “Charge”.

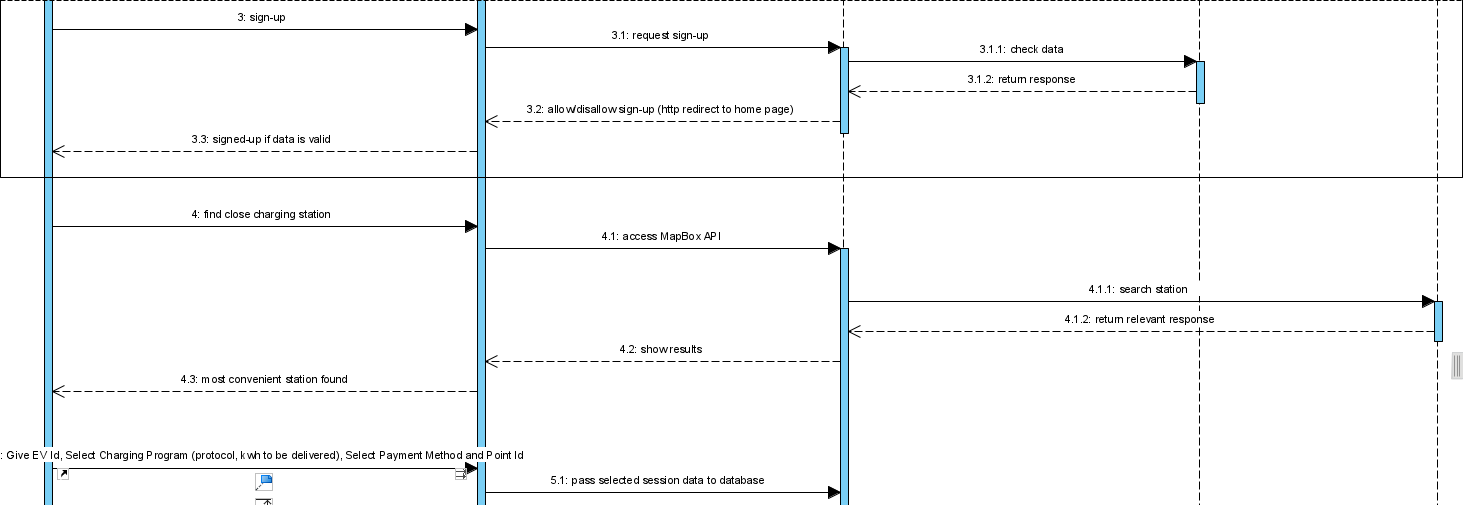
Βήμα 6: Ξεκινάει η φόρτιση. Κατά τη διάρκεια της, εμφανίζονται κάποια από τα δεδομένα της φόρτισης στη διαδικτυακή διεπαφή του χρήστη (Διάρκεια φόρτισης, μπάρα φόρτισης, εκτιμώμενο κόστος).

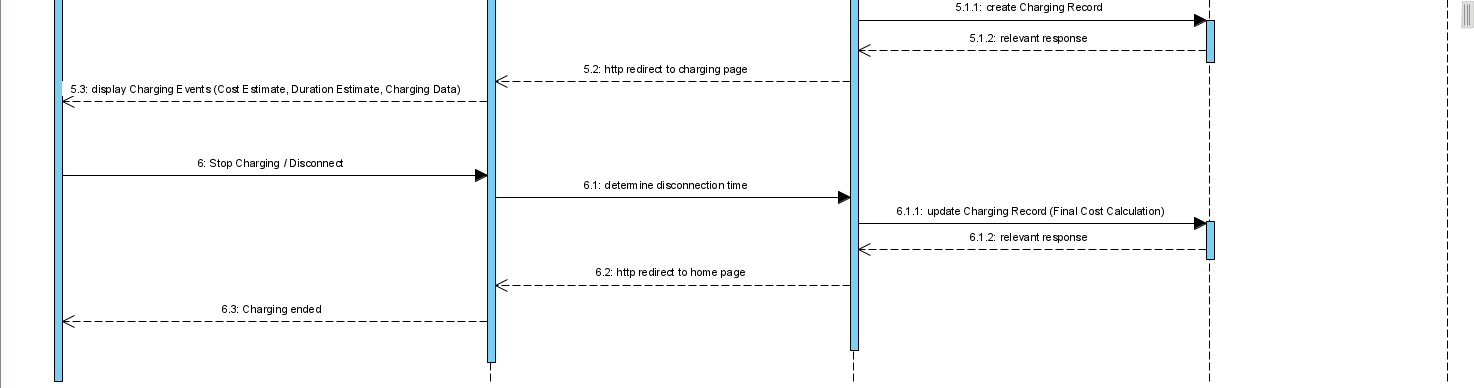
Βήμα 7: Μόλις τελειώσει η φόρτιση ή αποσυνδεθεί ο χρήστης, υπολογίζεται η ημερομηνία/ώρα αποσύνδεσης, η ενέργεια που παραδόθηκε και το συνολικό κόστος της φόρτισης που ολοκληρώθηκε.

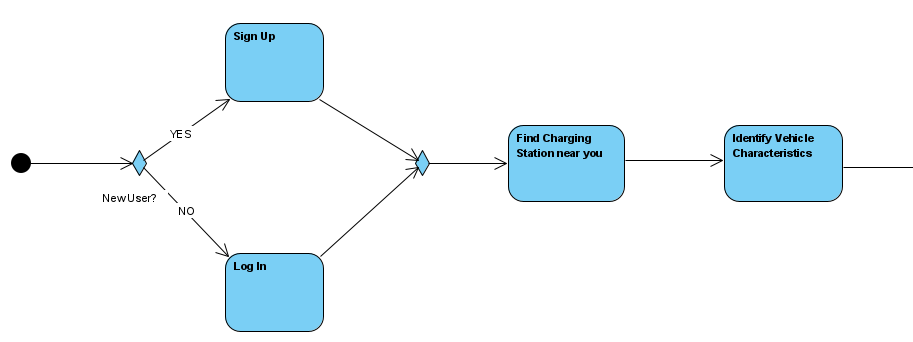
Βήμα 8: Καταγράφεται η συγκεκριμένη φόρτιση, με συγκεκριμένα αναγνωριστικά και τα δεδομένα που προέκυψαν από αυτή, στο σύστημά μας.

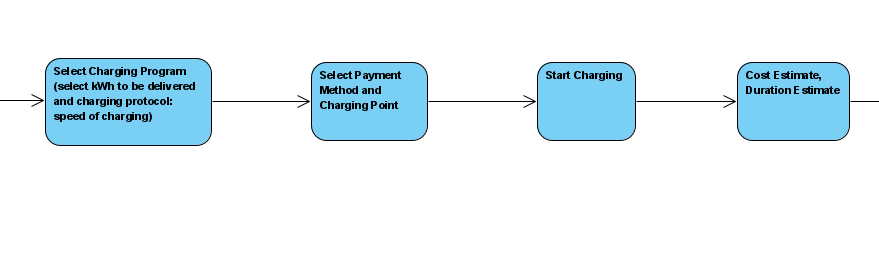
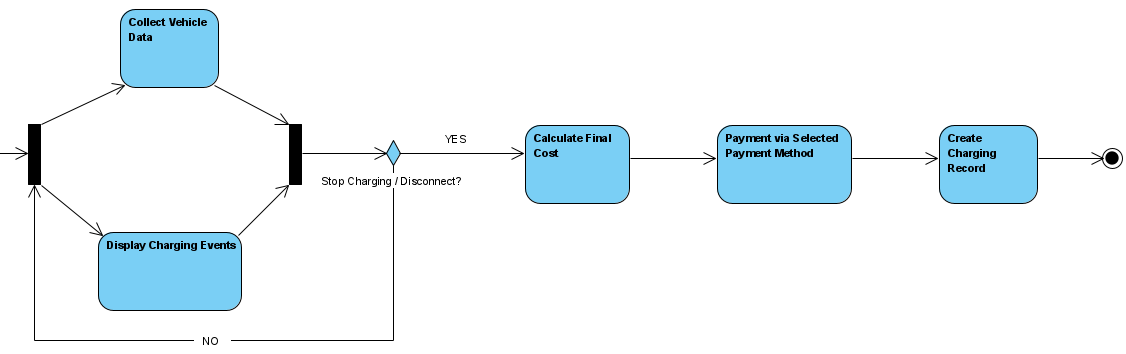
Παρακάτω ακολουθούν τα διαγράμματα UML Sequence και UML Activity με αυτή τη σειρά.









#### 3.1.1.7 Δεδομένα εξόδου

Δεδομένα εξόδου αποτελούν, αρχικά, τα αντίστοιχα μηνύματα (flashes στο frontend) ανάλογα αν η προσπάθειά του χρήστη είναι επιτυχημένη ή απέτυχε.

Κατά τη διάρκεια της φόρτισης, στη συνέχεια, στην διαδικτυακή διεπαφή του χρήστη επιστρέφονται η μπάρα με την πρόοδο της φόρτισης καθώς και δεδομένα που προκύπτουν από την τρέχουσα φόρτιση.

Ύστερα, ο χρήστης μπορεί να αποχωρήσει.

#### 3.1.1.8 Παρατηρήσεις

### 3.1.2 ΠΕΡΙΠΤΩΣΗ ΧΡΗΣΗΣ 2: (Έκδοση καταγραφής γεγονότων φόρτισης)

#### 3.1.2.1 Χρήστες (ρόλοι) που εμπλέκονται

Μόνο εγγεγραμμένοι χρήστες, οι οποίοι έχουν συνδεθεί επιτυχώς στην εφαρμογή, έχουν τη δυνατότητα να ζητήσουν την έκδοση καταγραφής φόρτισης.

#### 3.1.2.2 Προϋποθέσεις εκτέλεσης

Ο χρήστης επιλέγει ημερομηνίες-φίλτρα βάσει των οποίων εμφανίζονται τα γεγονότα φόρτισης του χρήστη στη ζητούμενη περίοδο.

#### 3.1.2.3 Περιβάλλον εκτέλεσης

Ο χρήστης επιλέγει τα φίλτρα που επιθυμεί στη διαδικτυακή διεπαφή χρήστη. Ύστερα, το REST API επικοινωνεί με τη βάση και επιστρέφει τη λίστα των γεγονότων φόρτισης της ζητούμενης περιόδου στη διαδικτυακή διεπαφή του χρήστη.

#### 3.1.2.4 Δεδομένα εισόδου

Ο χρήστης καλείται να πατήσει στην επιλογή “Issue Statement”. Στη συνέχεια, επιλέγει τα επιθυμητά φίλτρα-ημερομηνίες και εφαρμόζοντάς τα παίρνει το ζητούμενο statement.

#### 3.1.2.5 Παράμετροι

Για να εκδοθεί η καταγραφή κάποιας φόρτισης, βασικές προϋποθέσεις είναι ο χρήστης να είναι εγγεγραμμένος στη βάση δεδομένων μας, ώστε να έχει πρόσβαση στις υπηρεσίες της εφαρμογής μας.

Έπειτα, επιλέγει μια περίοδο και βλέπει τη λίστα φορτίσεων του.

#### 3.1.2.6 Αλληλουχία ενεργειών - επιθυμητή συμπεριφορά

Περιγραφή με κείμενο (Βήμα 1, Βήμα 2 κλπ) και διαγράμματα UML αλληλουχίας (Sequence) και δραστηριοτήτων (Activity). Περιλαμβάνεται η συμπεριφορά σε απρόβλεπτες καταστάσεις και σφάλματα (εναλλακτικές ροές).

Βήμα 1: Ο χρήστης επισκέπτεται τη σελίδα της εφαρμογής μας.

Βήμα 2: α) Αν ο χρήστης είναι ήδη εγγεγραμμένος στη βάση δεδομένων μας, τότε αρκεί να κάνει log in με το username και το password που του αντιστοιχούν.

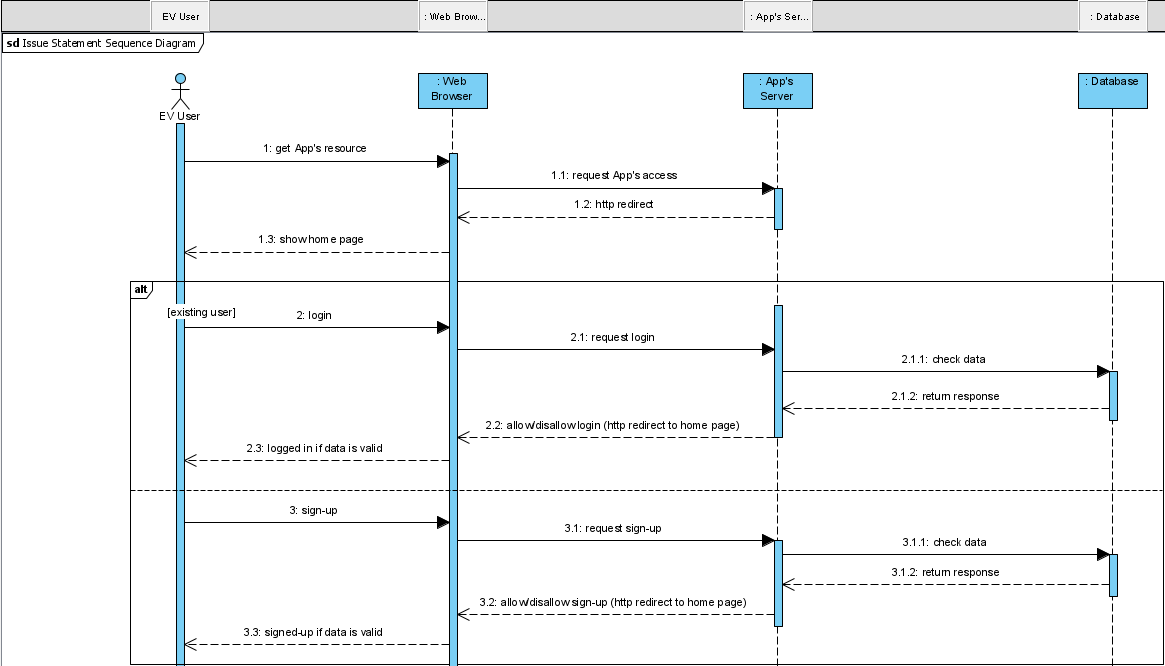
β) Αν ο χρήστης δεν έχει εγγραφεί, συμπληρώνει τα απαραίτητα στοιχεία, ώστε να προστεθεί στη βάση δεδομένων μας ως νέος χρήστης, δηλαδή να επιτευχθεί η εγγραφή του. Αν η εγγραφή απέτυχε, λόγω ύπαρξης των στοιχείων που εισήχθησαν στη βάση δεδομένων μας, ο χρήστης ενημερώνεται με μήνυμα, ώστε να επαναλάβει το βήμα 2.

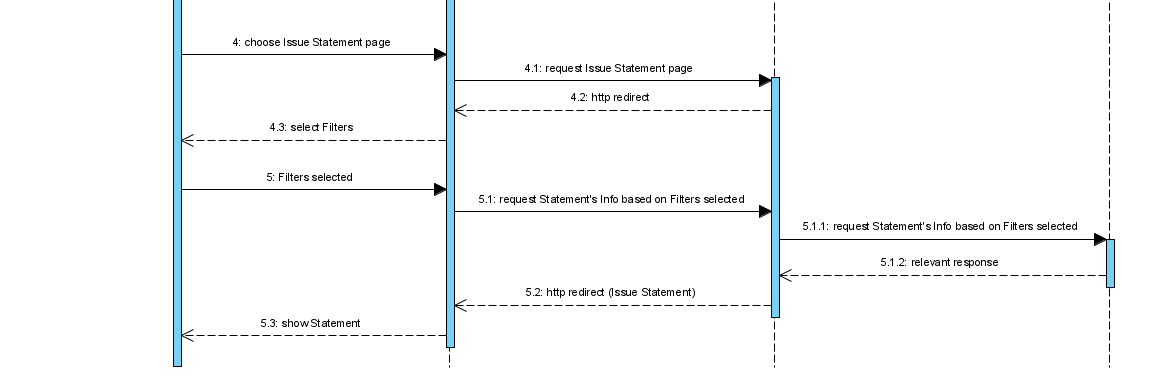
Βήμα 3: Ο χρήστης πατάει στην επιλογή “Issue Statement” της αρχικής σελίδας.

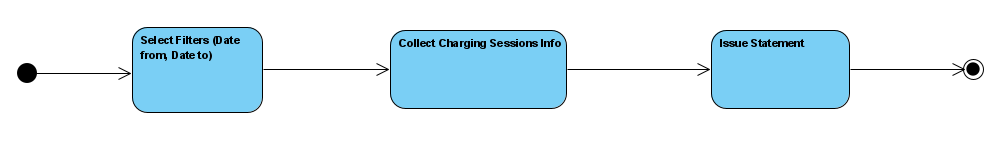
Βήμα 4: Ο χρήστης επιλέγει την ζητούμενη περίοδο (ημερομηνίες - φίλτρα) και κάνει submit.

Βήμα 4: Εμφανίζεται στη διαδικτυακή διεπαφή του χρήστη η λίστα φορτίσεων του κατά τη διάρκεια της ζητούμενης περιόδου. Επίσης, ο χρήστης μπορεί να επιλέξει να διαγράψει κάποιο από τα γεγονότα φόρτισης του.

Παρακάτω ακολουθούν τα διαγράμματα UML Sequence και UML Activity με αυτή τη σειρά.







#### 3.1.2.7 Δεδομένα εξόδου

Στη διαδικτυακή διεπαφή του χρήστη, αφού επιλεγούν τα επιθυμητά φίλτρα εμφανίζεται η λίστα με τα γεγονότα φόρτισης της ζητούμενης περιόδου.

#### 3.1.2.8 Παρατηρήσεις

## 3.2 Απαιτήσεις επιδόσεων

Η εφαρμογή μας σχεδιάστηκε για να τρέχει με ικανοποιητικό αριθμό stakeholders και δεδομένων στη βάση (points, sessions, κλπ).

## 3.3 Απαιτήσεις οργάνωσης δεδομένων

### 3.3.1 Απαιτήσεις και περιορισμοί πρόσβασης σε δεδομένα

Ο ADMIN και οι Privileged Stakeholders έχουν την περισσότερη πρόσβαση όσον αφορά τις πληροφορίες που παρέχονται από το cli (SessionsPer…), ενώ οι χρήστες έχουν πρόσβαση στο frontend και στις δυνατότητες που αυτό παρέχει (base use-cases).

## 3.4 Περιορισμοί σχεδίασης

## 3.5 Λοιπές απαιτήσεις

### 3.5.1 Απαιτήσεις διαθεσιμότητας λογισμικού

Η εφαρμογή οφείλει να είναι πάντα διαθέσιμη στους χρήστες της.

### 3.5.2 Απαιτήσεις ασφάλειας

Οι κωδικοί των χρηστών αποθηκεύονται στη βάση δεδομένων, αφού πρώτα περάσουν από το στάδιο της κρυπτογράφησης, για ενίσχυση της ασφάλειας για την αποτροπή υποκλοπής προσωπικών στοιχείων των χρηστών.

Οι χρήστες γίνονται authenticate τόσο με jwt tokens (cli) όσο και με τα cookies του LoginManager της Flask.

### 3.5.3 Απαιτήσεις συντήρησης

Η εφαρμογή υλοποιήθηκε με σκέψη και επόμενες εκδόσεις αυτής.

Π.χ. Τα αρχεία οργανώνονται κατάλληλα ώστε να επιτρέπεται η εύκολη συντήρησή τους.

Ο σχεδιασμός της βάσης επιτρέπει την ιδιοκτησία πολλών αμαξιών από κάθε χρήστη, κάτι που μπορεί να υλοποιηθεί σε επόμενη έκδοση της εφαρμογής.