Έγγραφο απαιτήσεων λογισμικού (SRS)

Δούφας Ευγκένι 03115728

Ζαράνης Ιάσονας 03115729

Μήλιας-Αργείτης Παναγιώτης 03112137

Χρήστος Κίνας 03116103

[ ChargeHub ]

# Εισαγωγή

## 1.1 Εισαγωγή: σκοπός του λογισμικού

Σκοπός του συγκεκριμένου λογισμικού είναι η γρήγορη και άμεση φόρτιση των ηλεκτρικών οχημάτων με βάση τις διαθέσιμες θέσεις παροχής ηλεκτρικής ενέργειας καθώς και η συλλογή δεδομένων σχετικά με τη φόρτισή τους για την καλύτερη γνώση και προσαρμογή των εμπλεκομένων.

## 1.2 Διεπαφές (interfaces)

### 1.2.1 Διεπαφές με εξωτερικά συστήματα

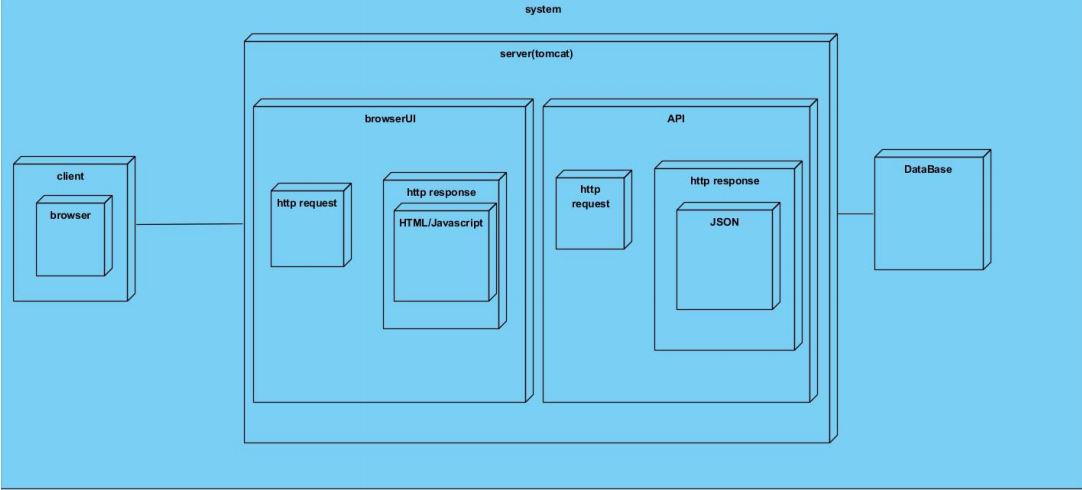
Οι εξωτερικές διεπαφές του συστήματός μας αφορούν:

Τα άκρα της σύνδεσης του φυλλομετρητή (πλατφόρμα του χρήστη της εφαρμογής) με τον διακομιστή που φιλοξενεί το API της εφαρμογής μας αλλά και το σύνολο των HTML σελίδων και της JavaScript που αποτελούν το User Interface.

Την επικοινωνία ανάμεσα σε User Interface και API για τη σωστή ενημέρωση του περιεχομένου που επιστρέφεται στον χρήστη.

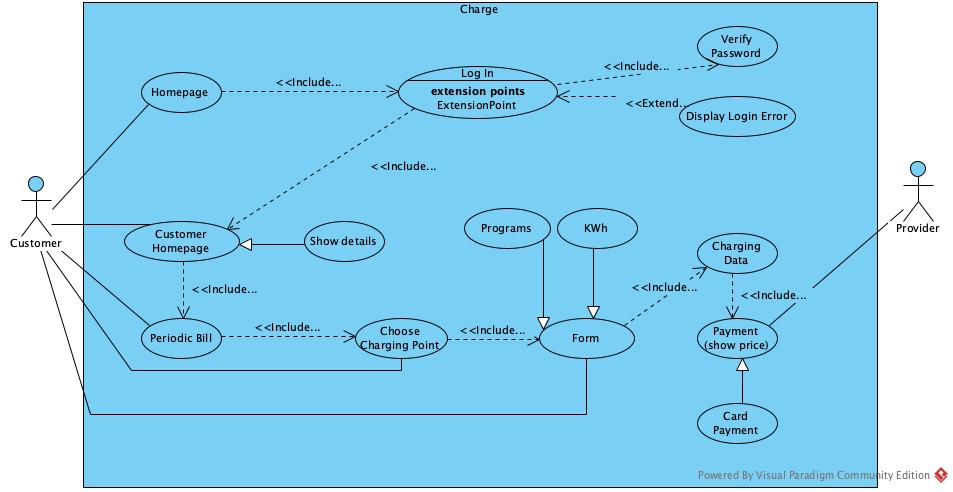
Τις εξόδους του API από και προς τη Βάση Δεδομένων. Ο Client επικοινωνεί μέσω http requests με το UI και το API.

Από το UI δέχεται τα http responses που ενσωματώνουν τα html pages που πρέπει να προβάλλει (μαζί με τη JavaScript που τα συνοδεύει). Από το API δέχεται μόνο http responses που περιέχουν JSON αρχεία με τις πληροφορίες που χρειάζεται.

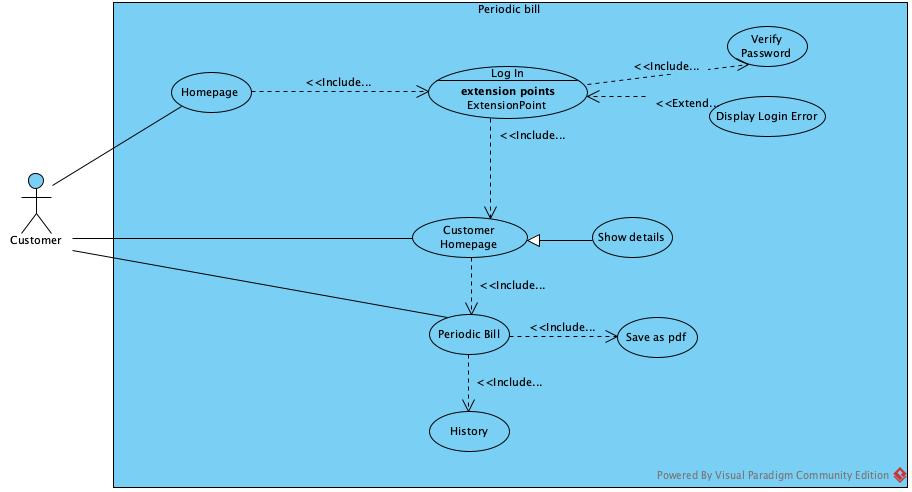


### 1.2.2 Διεπαφές με το χρήστη

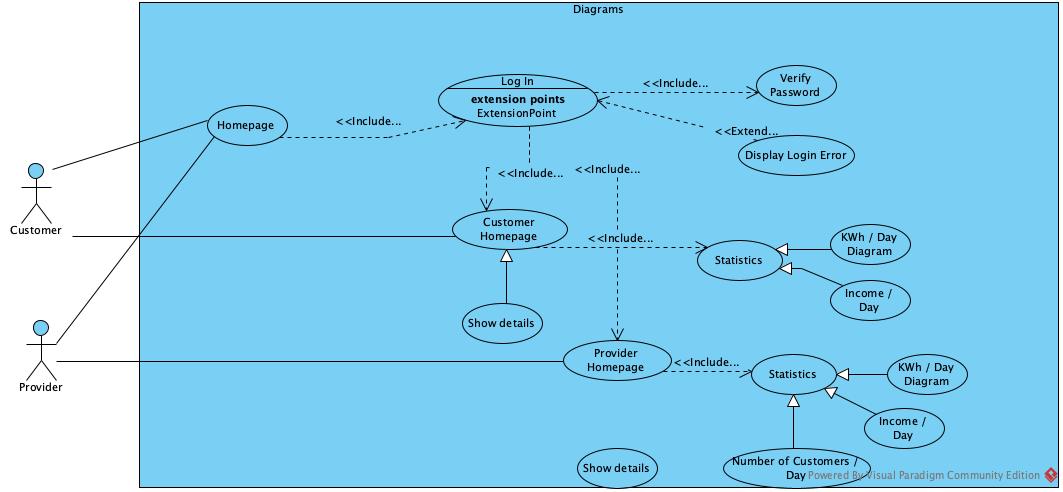
* *Use Case 1: Φόρτιση ηλεκτρικού οχήματος*



*Use Case 2: Έκδοση Περιοδικού λογαριασμού*



*Use Case 3: Απεικόνιση δεδομένων φόρτισης σε διάγραμμα*



# Αναφορές - πηγές πληροφοριών

Δεν υπάρχουν

# Προδιαγραφές απαιτήσεων λογισμικού

## 3.1 Περιπτώσεις χρήσης

### 3.1.1 ΠΕΡΙΠΤΩΣΗ ΧΡΗΣΗΣ 1: Φόρτιση ηλεκτρικού οχήματος

#### 3.1.1.1 Χρήστες (ρόλοι) που εμπλέκονται

Αυτή η χρήση του API περιλαμβάνει το χρήστη ο οποίος θέλει να φορτίσει το σε κάποιον σταθμό το ηλεκτρικό του όχημα, καθώς και τον ιδιοκτήτη του σταθμού ώστε να επιτευχθεί η πληρωμή.

#### 3.1.1.2 Προϋποθέσεις εκτέλεσης

Για να κάνει log in ο χρήστης χρειάζεται να έχει έναν λογαριασμό ήδη ώστε να έχει το δικαίωμα να μπορεί να επιλέξει το πρόγραμμα φόρτισης που επιθυμεί.

#### 3.1.1.3 Περιβάλλον εκτέλεσης

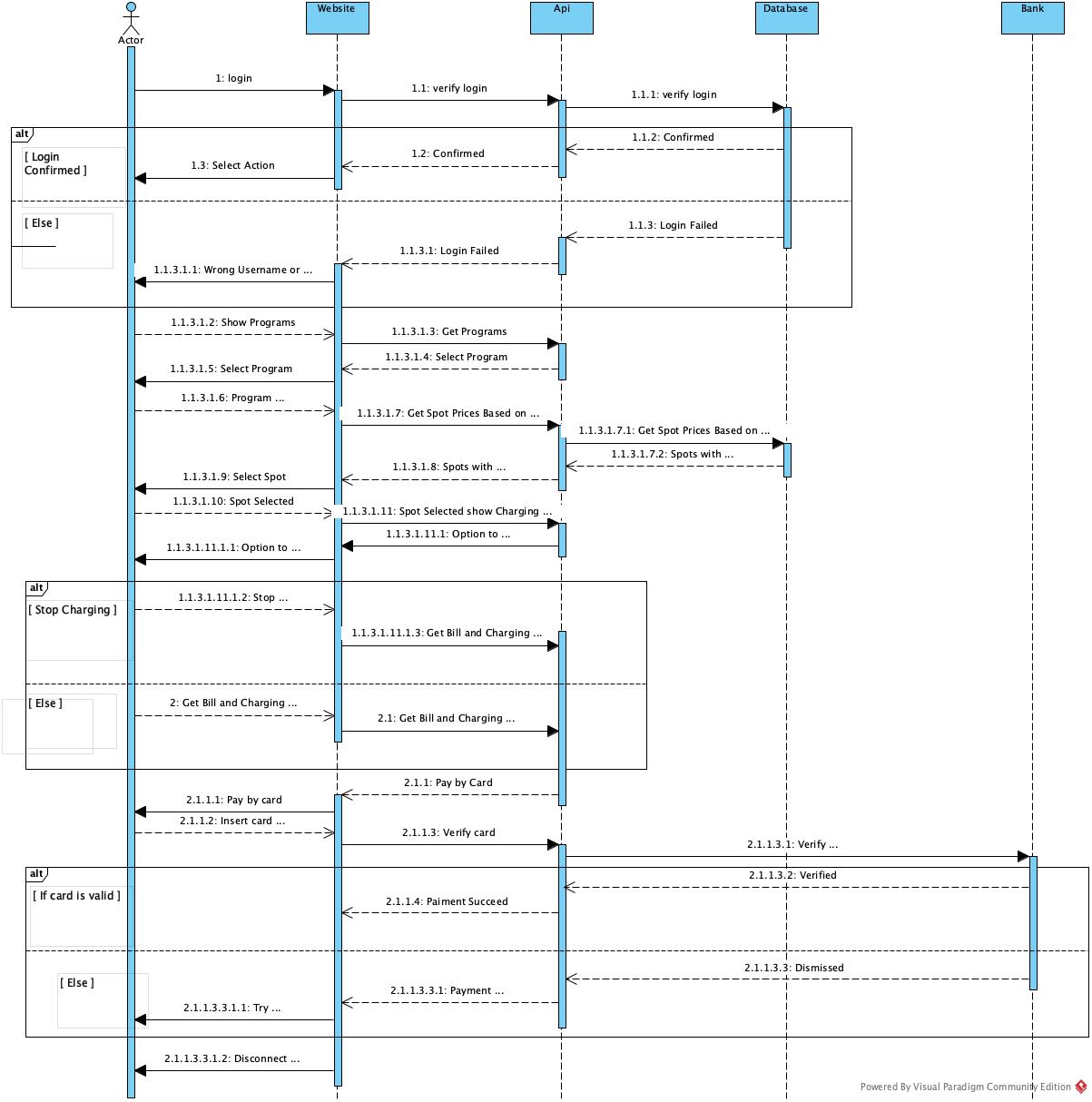
Για την εκτέλεση αυτής της χρήσης χρειάζεται η διαδικτυακή διεπαφή χρήστη καθώς και η επικοινωνία του API με την βάση για την εύρεση των υπάρχοντων προγραμμάτων φόρτισης καθώς και τα σημεία στα οποία αυτά διατίθενται, ή επικύρωση στοιχείων (log in).

#### 3.1.1.4 Δεδομένα εισόδου

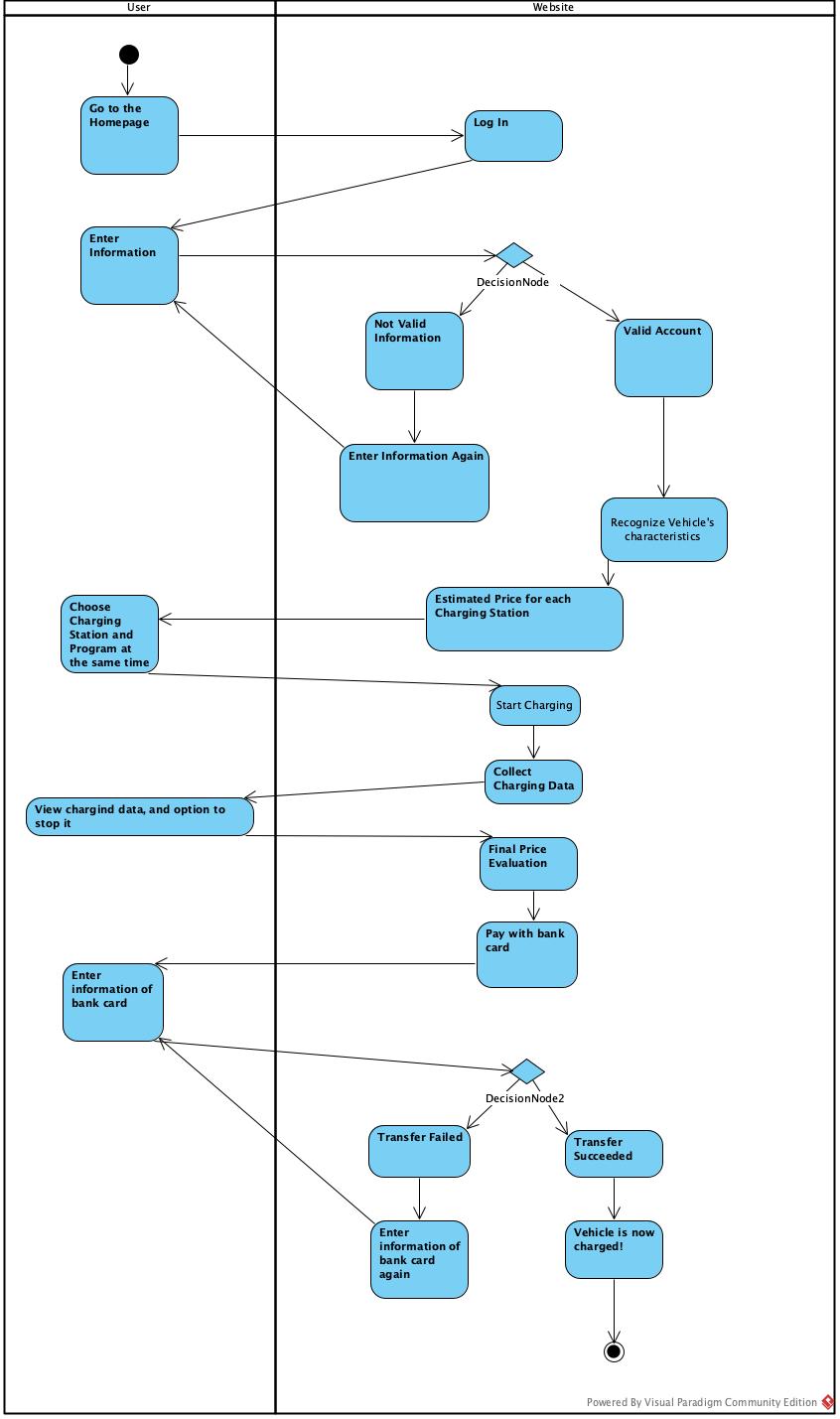
Τα δεδομένα εισόδου από τον administrator περιλαμβάνουν : 1) Τα διαθέσιμα προγράμματα φόρτισης , 2) Τα σημεία στα μπορεί να πραγματοποιηθεί η φόρτιση 3) Log in: τα ατομικά στοιχεία του εγγεγραμμένου χρήστη (username, password).

#### 3.1.1.5 Αλληλουχία ενεργειών - επιθυμητή συμπεριφορά

*Uml Sequence diagram*



*Uml Activity Diagram*



#### 3.1.1.7 Δεδομένα εξόδου

Ο χρήστης λαμβάνει δεδομένα εξόδου αφότου πραγματοποιήσει την φόρτιση και οριστεί η τελική τιμή την οποία καλείται να πληρώσει.

### 3.1.2 ΠΕΡΙΠΤΩΣΗ ΧΡΗΣΗΣ 2: Έκδοση Περιοδικού Λογαριασμού

#### 3.1.2.1 Χρήστες (ρόλοι) που εμπλέκονται

Αυτή η χρήση του API περιλαμβάνει το χρήστη ο οποίος ζητά από το σύστημα την έκδοση ενός περιοδικού λογαριασμού με βάση τα δεδομένα που έχουν καταχωρηθεί στο λογαριασμό του.

#### 3.1.2.2 Προϋποθέσεις εκτέλεσης

Ο χρήστης θα πρέπει να διαθέτει έγκυρο λογαριασμό ώστε να συλλεχθούν τα απαραίτητα δεδομένα.

#### 3.1.2.3 Περιβάλλον εκτέλεσης

Για την εκτέλεση αυτής της χρήσης χρειάζεται η διαδικτυακή διεπαφή χρήστη καθώς και η επικοινωνία του API με την βάση για την εύρεση των δεδομένων που αφορούν τα γεγονότα φόρτισης τα οποί αντιστοιχούν σε μια συγκεκριμένη χρονική περίοδο.

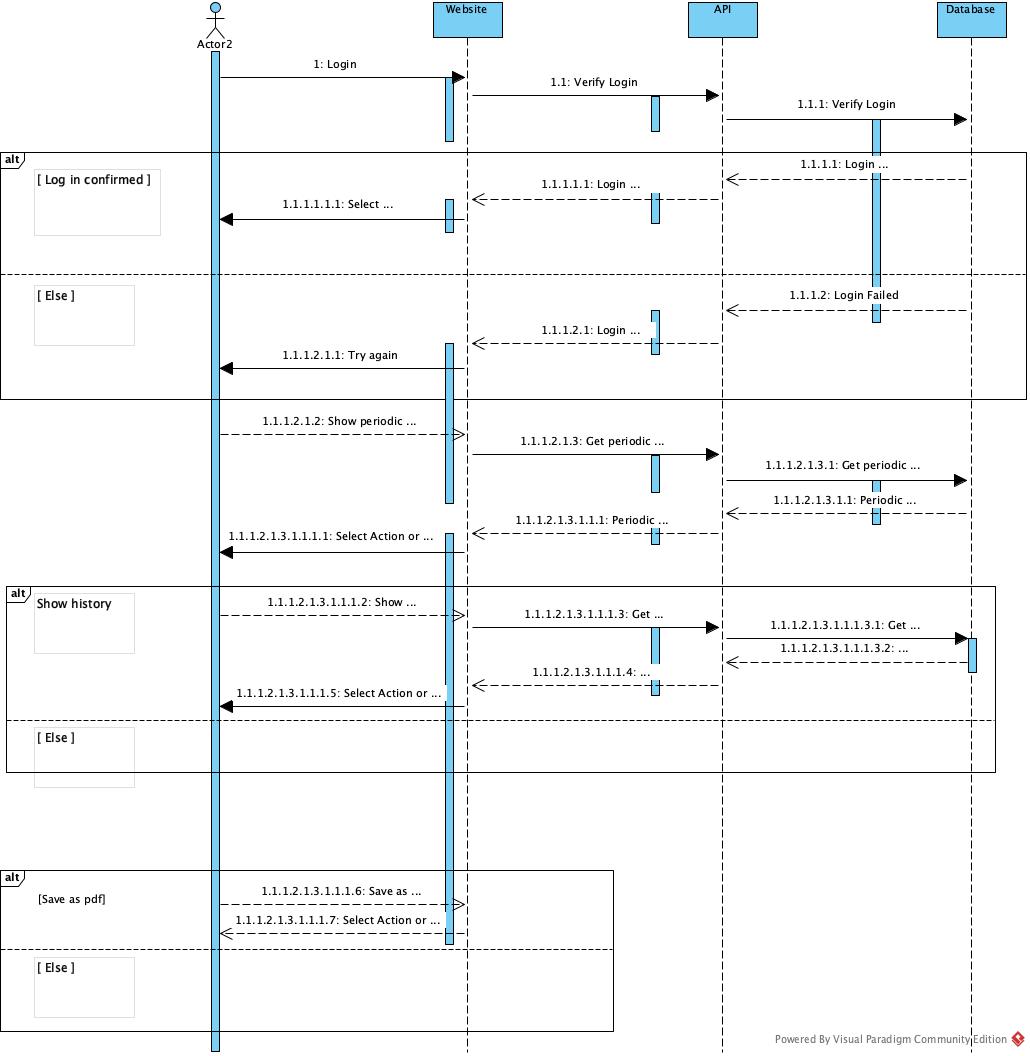
#### 3.1.2.4 Δεδομένα εισόδου

Τα δεδομένα εισόδου περιλαμβάνουν τα στοιχεία των γεγονότων φόρτισης του χρήστη σε μια συγκεκριμένη χρονική περίοδο

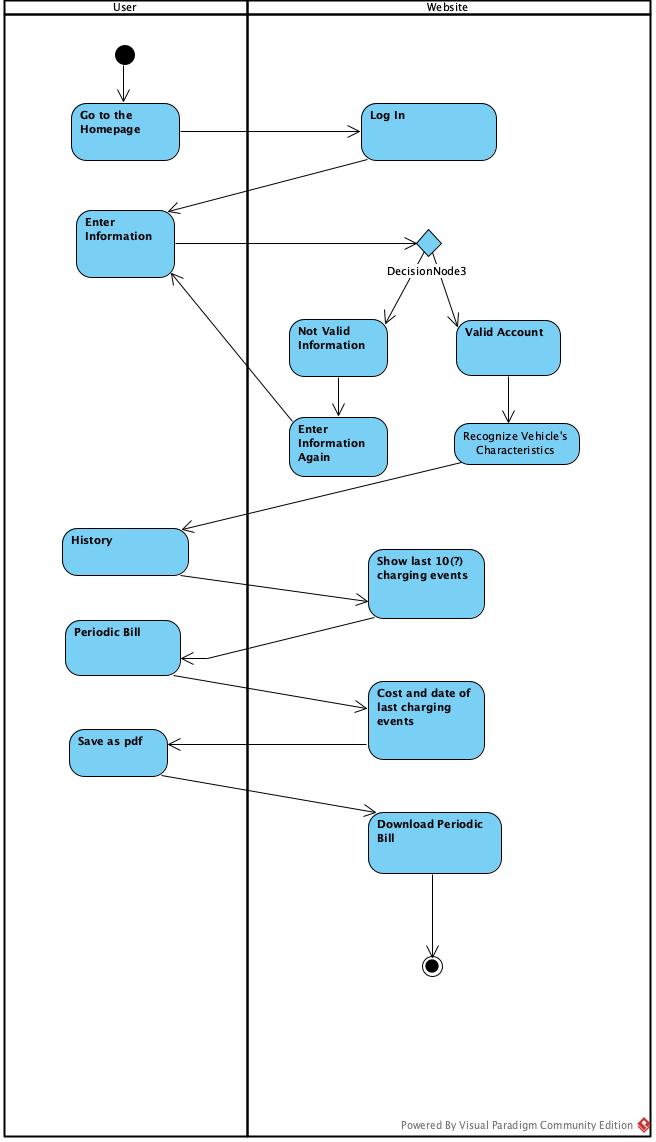
#### 3.1.2.6 Αλληλουχία ενεργειών - επιθυμητή συμπεριφορά

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).

Uml Sequence Diagram



Uml Activity Diagram



#### 3.1.2.7 Δεδομένα εξόδου

Ο χρήστης λαμβάνει δεδομένα εξόδου όταν καταφέρουν τελικά να συλλεχθούν τα δεδομένα και εκδίδεται ο περιοδικός λογαριασμός

3.1.3 ΠΕΡΙΠΤΩΣΗ ΧΡΗΣΗΣ 3: Απεικόνιση δεδομένων σε διάγραμμα

#### 3.1.3.1 Χρήστες (ρόλοι) που εμπλέκονται

Αυτή η χρήση του API περιλαμβάνει το χρήστη ο οποίος ζητά από το σύστημα την απεικόνιση γεγονότων φόρτισης σε διάγραμμα για στατιστική μελέτη αυτών.

#### 3.1.3.2 Προϋποθέσεις εκτέλεσης

Ο χρήστης θα πρέπει να διαθέτει έγκυρο λογαριασμό ώστε να συλλεχθούν τα απαραίτητα δεδομένα.

#### 3.1.3.3 Περιβάλλον εκτέλεσης

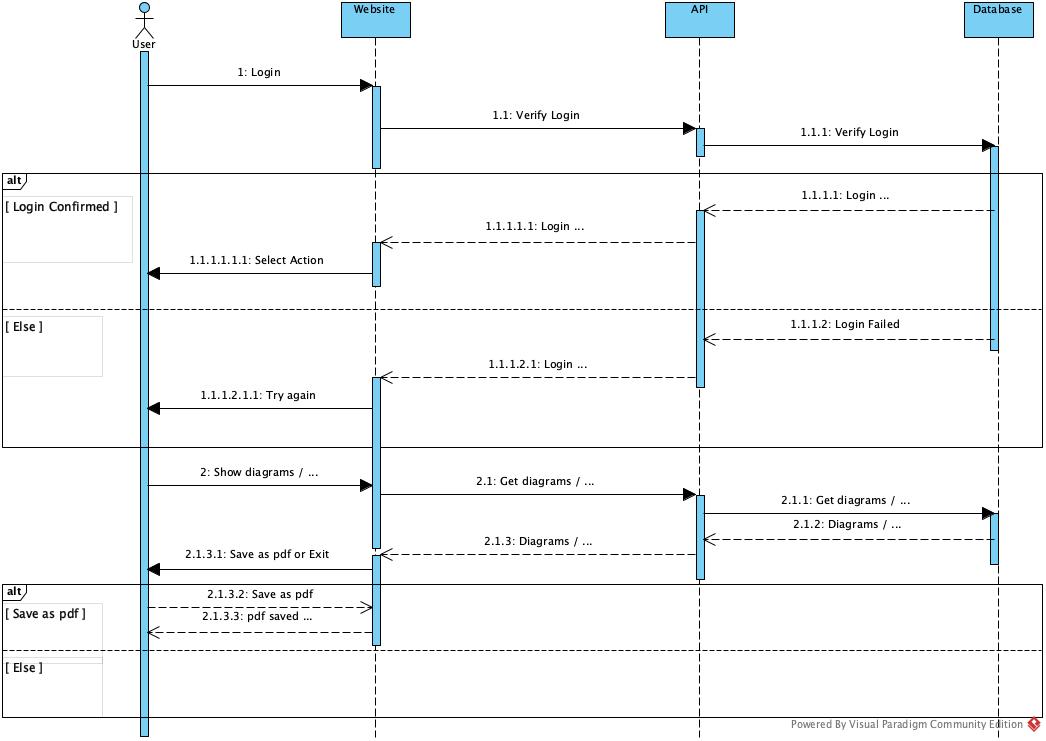
Για την εκτέλεση αυτής της χρήσης χρειάζεται η διαδικτυακή διεπαφή χρήστη καθώς και η επικοινωνία του API με την βάση για την εύρεση των δεδομένων.

#### 3.1.3.4 Δεδομένα εισόδου

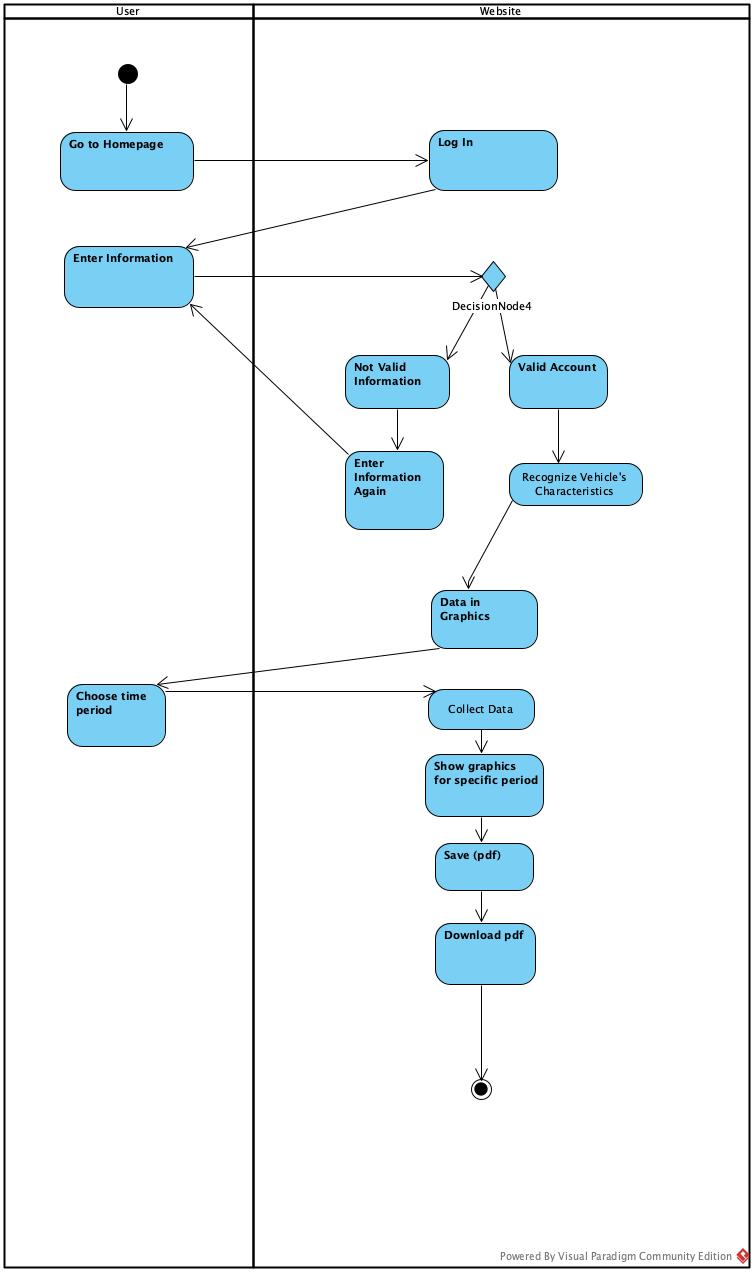
Τα δεδομένα εισόδου περιλαμβάνουν τα στοιχεία των γεγονότων φόρτισης ενός η περισσότερων χρηστών

#### 3.1.3.6 Αλληλουχία ενεργειών - επιθυμητή συμπεριφορά

*Uml Sequence diagram*



*Uml Activity Diagram*



#### 3.1.3.7 Δεδομένα εξόδου

Ο χρήστης λαμβάνει δεδομένα εξόδου διαγράμματα τα οποία καταγράφουν μια γενική εικόνα για τη φόρτιση ηλεκτρικων οχημάτων

## 3.2 Απαιτήσεις επιδόσεων

Ποσοτική τεκμηρίωση μέτρων και κριτηρίων επιθυμητών επιδόσεων με αναφορά στα ποσοτικά χαρακτηριστικά εισόδων και φορτίου του λογισμικού.

Το σύστημά μας διαθέτει βάση δεδομένων η οποία περιέχει πληροφορίες για τα αυτοκίνητα , τα σημεία φόρτισης και τους χρήστες , καθώς και API και UI, ρόλος των οποίων είναι η παροχή πρόσβασης χρηστών, τρίτων εφαρμογών και γενικά εξωτερικών συστημάτων στις λειτουργίες της εφαρμογής. Με βάση, λοιπόν, την αρχιτεκτονική αυτή, τα κύρια μεγέθη που αφορούν το λογισμικό μας είναι τα εξής δύο:

* Το μέγεθος των δεδομένων που είναι αποθηκευμένα στη βάση μας και αφορούν χρήστες, σημεία φόρτισης, αυτοκίνητα , προγράμματα, τιμές και αρχείο συναλλαγών (το οποίο μέγεθος μετράται σε GB προς το παρόν και λογικά σε TB στο μέλλον)
* Το πλήθος των ενεργών συνδέσεων με το API (είτε από το δικό μας UI, είτε από τρίτες εφαρμογές)
* Η ταχύτητα ανταλλαγής δεδομένων μέσω αυτών των συνδέσεων (από και προς το API, από και προς τη βάση δεδομένων)

Έχοντας, οπότε, υπόψη μας τα παραπάνω βασικά μεγέθη, κρίσιμες μετρικές της επιθυμητής απόδοσης του λογισμικού μας είναι οι ακόλουθες:

* Η αύξηση του μεγέθους των δεδομένων μας με το χρόνο (ή, καλύτερα, το ποσοστό αύξησης π.χ. ανά εβδομάδα, μήνα και έτος), η οποία μας βοηθά στη κατανόηση των τεχνικών αναγκών της εφαρμογής.
* Χρόνος απόκρισης σε ένα αίτημα χρήστη σε ms. Είναι πολύ σημαντικό ο συνολικός χρόνος επεξεργασίας, καθώς και το latency του δικτύου να μην επηρεάζουν αρνητικά την εμπειρία του χρήστη.
* Throughput εξυπηρέτησης αιτημάτων σε βάση δεδομένων και API. Ενδιαφέρει τόσο το πλήθος των αιτημάτων όσο και το μέγεθός τους (επιτρέπονται κλήσεις στο API που απαιτούν πολλή επεξεργασία και επιστρέφουν μεγάλο πλήθος δεδομένων).
* Η κλιμακωσιμότητα, θέλουμε δηλαδή η οποιαδήποτε αύξηση των δεδομένων λόγω της δραστηριότητας της εφαρμογής να αντισταθμίζεται γραμμικά από μία αντίστοιχη οριζόντια αύξηση των διαθέσιμων πόρων.
* Το πλήθος των σφαλμάτων που επιστρέφουν η βάση και το API, καθώς και το ποσοστό σφαλμάτων σε σχέση με το συνολικό πλήθος των αιτημάτων, με τη πάροδο του χρόνου.
* Η διεκπεραιωτική ικανότητα της εφαρμογής, και, πιο συγκεκριμένα, τόσο το bandwidth του ίδιου του API, όσο και της βάσης, καθώς τα δύο αυτά δεν είναι άρρηκτα συνδεδεμένα. Πιο συγκεκριμένα, για κάθε αίτημα στη βάση, σίγουρα προηγήθηκε ένα τουλάχιστον αίτημα στο API (αφού η βάση δεν είναι άμεσα προσβάσιμη), αλλά δεν είναι απαραίτητο ότι για κάθε αίτημα στο API θα γίνεται και ένα αντίστοιχο αίτημα στη βάση (λόγω περιπτώσεων όπως σφάλματα, caching κλπ.)

## 3.3 Απαιτήσεις οργάνωσης δεδομένων

### 3.3.1 Απαιτήσεις και περιορισμοί πρόσβασης σε δεδομένα

Απαιτήσεις πρόσβασης και περιορισμοί.

Αναφορικά με απαιτήσεις και περιορισμούς πρόσβασης στα δεδομένα τα οποία θα διαχειρίζεται το λογισμικό μας, έχουμε να σημειώσουμε τα εξής:

* Έχοντας αποφασίσει τη σχεδίαση και υλοποίηση του RESTful API, έχουμε ήδη προδιαγράψει ότι κανένας εξωτερικός χρήστης ή λογισμικό δεν θα έχει άμεση / ελεύθερη πρόσβαση στα δεδομένα που διατηρεί το λογισμικό, αλλά όλες οι ανταλλαγές δεδομένων θα περνούν από το API.

Όπως είναι λογικό, δεν θέλουμε να υπάρχει πρόσβαση στα δεδομένα που αφορούν έναν χρήστη από οποιονδήποτε άλλο, και για αυτό το σκοπό οι πληροφορίες για κάθε χρήστη περιλαμβάνουν email και μυστικό κωδικό.

## 3.4 Περιορισμοί σχεδίασης

Λεπτομερής τεχνική τεκμηρίωση των περιορισμών σχεδίασης οι οποίοι επιβάλλονται από απαιτήσεις συμμόρφωσης σε πρότυπα, κανονισμούς, ή άλλους περιορισμούς του έργου. Περιλαμβάνεται η πολιτική ονοματολογίας οντοτήτων δεδομένων και πεδίων. Τέτοιοι περιορισμοί μπορεί να επιβάλλονται από τη χρήση βιβλιοθηκών, frameworks, περιβαλλόντων ανάπτυξης κλπ

## 3.5 Λοιπές απαιτήσεις

### 3.5.1 Απαιτήσεις διαθεσιμότητας λογισμικού

Τεκμηρίωση απαιτήσεων διαθεσιμότητας

### 3.5.2 Απαιτήσεις ασφάλειας

Τα μόνα ευαίσθητα δεδομένα τα οποία διαχειρίζεται το λογισμικό μας είναι τα στοιχεία των χρηστών: όνομα, email, αγαπημένα, και γενικότερα ο λογαριασμός του κάθε χρήστη σαν συνολική υπόσταση. Στα δεδομένα αυτά, και στον κάθε λογαριασμό, θέλουμε να έχει πρόσβαση μόνο ο αντίστοιχος χρήστης, μέσω του μυστικού κωδικού (password) που μας έχει δώσει κατά τη διαδικασία εγγραφής του (sign up).

Για το σκοπό αυτό, έχουμε εφαρμόσει στο λογισμικό μας τις ακόλουθες, ευρύτερα γνωστές, τακτικές:

* Διατήρηση του hash κάθε κωδικού στη βάση μας, και όχι του ίδιου του κωδικού, ώστε ο κωδικός του χρήστη να είναι ασφαλής από επιθέσεις στη βάση. ConquERRORS ΕΓΓΡΑΦΟ SRS Σελ 22 / 22
* Επικοινωνία που περιλαμβάνει ανταλλαγή οποιωνδήποτε ευαίσθητων πληροφοριών / δεδομένων γίνεται μέσω ασφαλούς σύνδεσης (SSL). Στη φάση του development, αυτό γίνεται με χρήση selfsigned ceritficate, αλλά στο μέλλον σε περιβάλλον production, στόχος είναι να γίνει μέσω CA signed certificate.

Με την υλοποίηση των παραπάνω στρατηγικών, μπορούμε να εγγυηθούμε σε αρκετά μεγάλο βαθμό την ασφάλεια των δεδομένων των χρηστών μας.

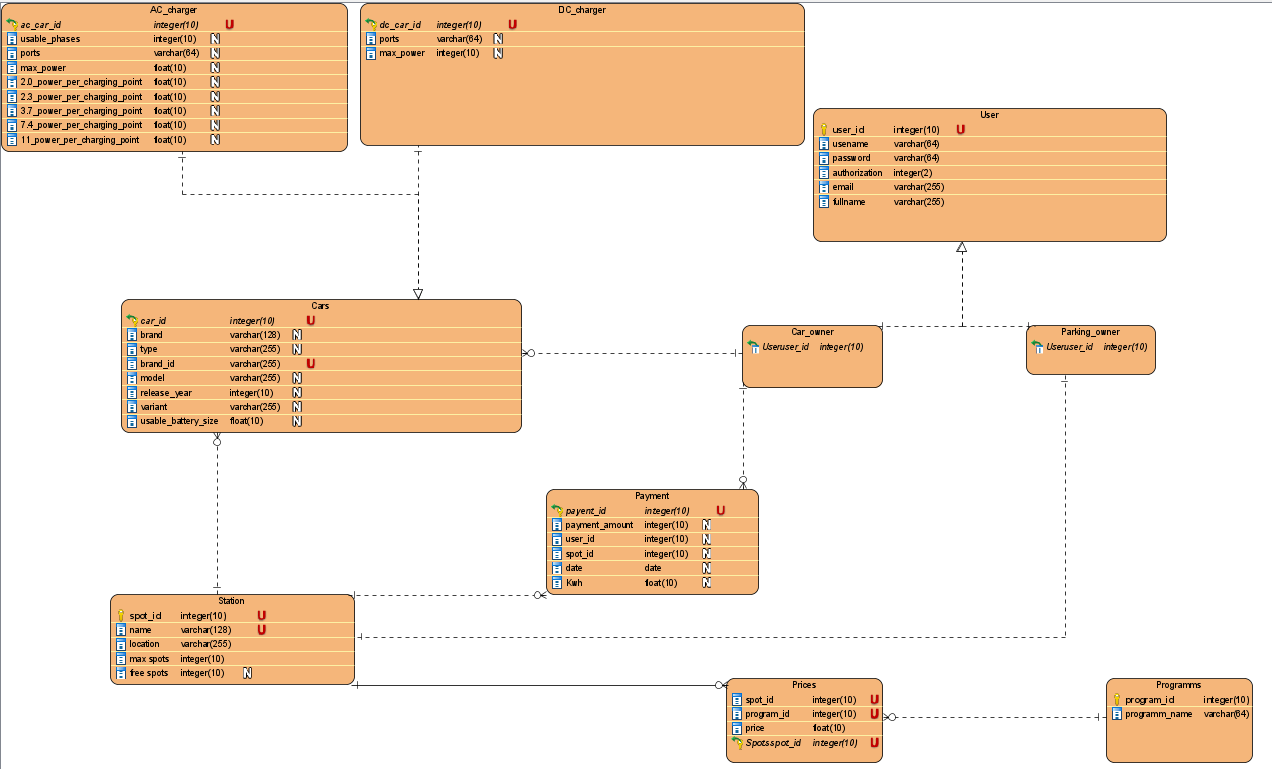
### 3.5.3 Απαιτήσεις συντήρησης

Τεκμηρίωση απαιτήσεων συντήρησης

Όσον αφορά την συντήρηση του λογισμικού, υπάρχουν δύο βασικοί άξονες:

* Η συνεχής επιτήρηση του λογισμικού, για τον εντοπισμό σφαλμάτων. Ίσως, μάλιστα, να είναι απαραίτητη και η προσωρινή διακοπή της λειτουργίας της εφαρμογής σε περίπτωση σοβαρού σφάλματος, το οποίο δεν μπορεί να διορθωθεί όσο η εφαρμογή είναι Live. Αυτό μπορεί να συμβεί τόσο στο back end όσο και στο front end της εφαρμογής.
* Γενικές εργασίες συντήρησης που είναι πιθανό να χρειάζονται, όπως για παράδειγμα ένας ο έλεγχος από developers ή/και administrators για την εγκυρότητα των δεδομένων της εφαρμογής, καθώς από της φύση της, και ειδικά τον πρώτο καιρό λειτουργίας, όπου μάλλον δε θα υπάρχουν ακόμα έμπιστοι crowdsourcers, είναι πολύ πιθανή η συσσώρευση άκυρων πληροφοριών και δεδομένων.

4.1 Παρουσιάζεται παρακάτω το ER- Diagram



4.2 Παρουσιάζεται παρακάτω το Class- Diagram

