Έγγραφο απαιτήσεων λογισμικού (SRS)

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

[ELON-CHARGE]

# Εισαγωγή

## 1.1 Εισαγωγή: σκοπός του λογισμικού

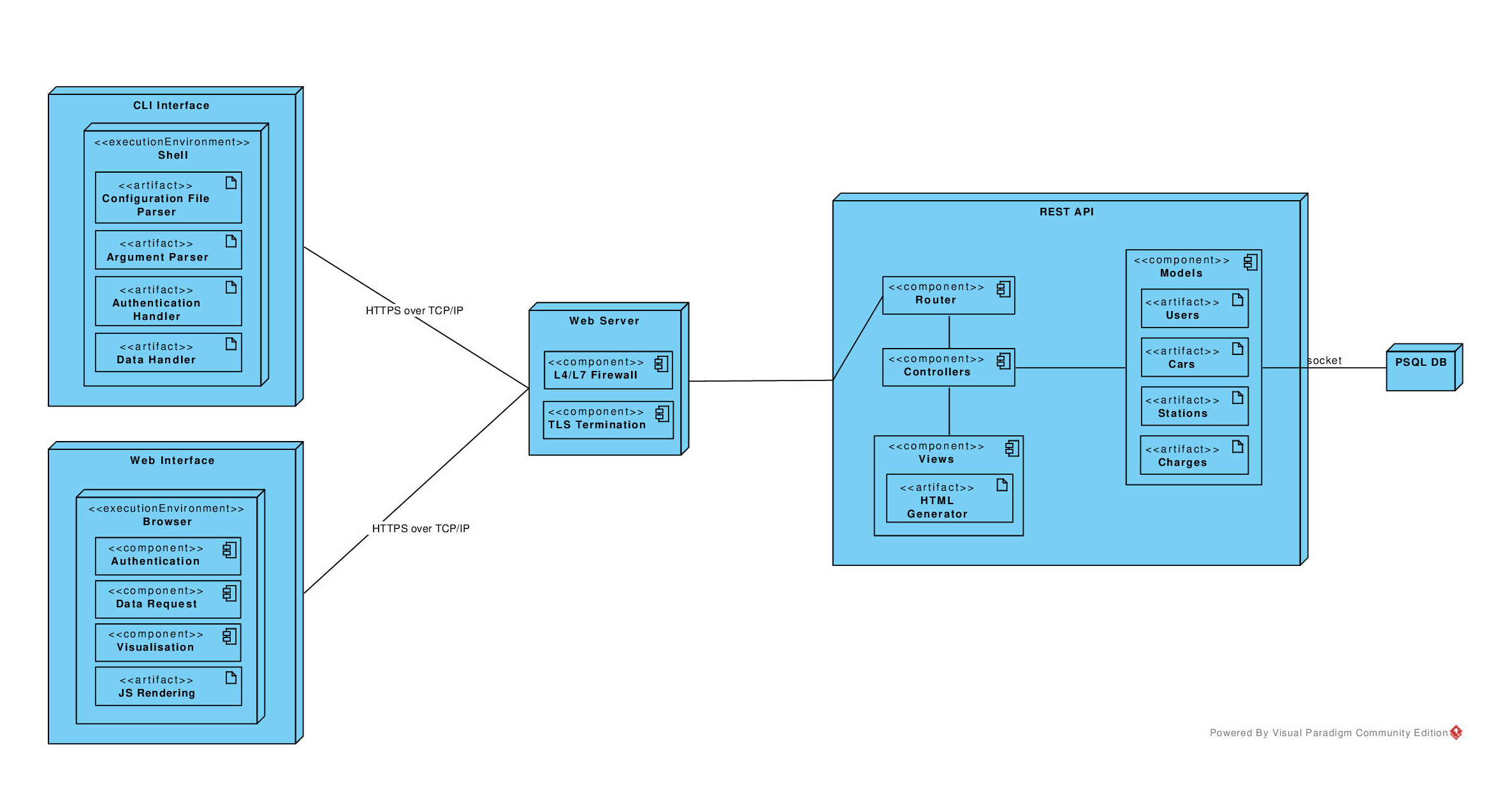
Το λογισμικό αφορά τη δημιουργία μιας διαδικτυακής πλατφόρμας η οποία θα εξυπηρετεί ιδιοκτήτες ηλεκτρικών οχημάτων, διαχειριστές χώρων στάθμευσης και φόρτισης καθώς και ερευνητές. Συγκεκριμένα, οι ιδιοκτήτες ηλεκτρικών οχημάτων θα έχουν πρόσβαση σε μια σύγχρονη πλατφόρμα που θα τους επιτρέπει να συγκρίνουν και να επιλέγουν σταθμούς φόρτισης με βάση πληθώρα κριτηρίων όπως τοποθεσία, μέθοδοι πληρωμής, τιμές, προσφορές κ.α. Ταυτόχρονα, η πλατφόρμα οδηγεί τους ιδιοκτήτες/διαχειριστές σταθμών φόρτισης να έχουν μια άμεση επαφή με το κοινό που εξυπηρετούν επιτρέποντας τους να εισάγουν και να διαχειριστούν τις υπηρεσίες που προσφέρουν. Οι παραπάνω λειτουργίες δημιουργούν μια πληθώρα από δεδομένα με τα οποία μπορούν να επωφεληθούν οι οδηγοί και διαχειριστές σταθμών. Οι χρήστες θα μπορούν να παρατηρούν την εξέλιξη της φόρτισης του οχήματος τους απομακρυσμένα, ενώ οι συναλλαγές τους θα αποθηκεύονται για μελλοντική επισκόπηση. Οι διαχειριστές σταθμών, θα μπορούν να επιβλέπουν δεδομένα που θα τους επιτρέπουν να πάρουν επιχειρησιακές αποφάσεις όπως τον αριθμό των μηνιαίων πελατών και τα έσοδα τους. Τέλος, τα δεδομένα θα μπορούν να παροχηθούν στην ερευνητική κοινότητα έπειτα από απαραίτητη ανωνυμοποίηση.

## 1.2 Επισκόπηση λογισμικού

Η υλοποίηση του λογισμικού μπορεί να διαχωριστεί σε τρείς υψηλού επιπέδου συνιστώσες.

1. **CLI Interface***:* Πρόκειται για την διεπαφή χρήστη μέσω του command line. Συγκεκριμένα, υλοποιεί μεθόδους για την αποδοτική ανάγνωση configuration files και παραμέτρων που θα διατηρούν στοιχεία και επιλογές του χρήστη όπως authentication tokens. Ταυτόχρονα, παρέχει μεθόδους για την επικοινωνία με το *REST API* για την αποστολή και λήψη δεδομένων. Προορίζεται για χρήση από διαχειριστές συστήματος.
2. **Web Interface**: Πρόκειται για την διεπαφή χρήστη από περιβάλλον web browser. Προορίζεται για ευρύ κοινό και χρησιμοποιείται από τους οδηγούς οχημάτων και τους διαχειριστές των σταθμών φόρτισης. Για να εκπληρώσει τις λειτουργίες του επικοινωνεί με το *REST API*.
3. **REST API**: Πρόκειται για τον σκελετό της εφαρμογής. Υλοποιεί μεθόδους ανάκτησης, εισαγωγής και επεξεργασίας δεδομένων, ενώ διαχειρίζεται την διεπαφή με την βάση δεδομένων. Είναι προσαρμοσμένος ώστε να επιστρέφει δεδομένα σε μορφή JSON καθώς και πιο γραφικές αναπαραστάσεις των δεδομένων.

Οι εξυπηρετητές συνδέονται με το REST API μέσω ενός web server ο οποίος υλοποιεί αποδοτικές μεθόδους ασφάλειας (λ.χ. firewall) ενώ η βάση δεδομένων είναι η PostgreSQL. Παρακάτω, παρουσιάζεται το Deployment Diagram.

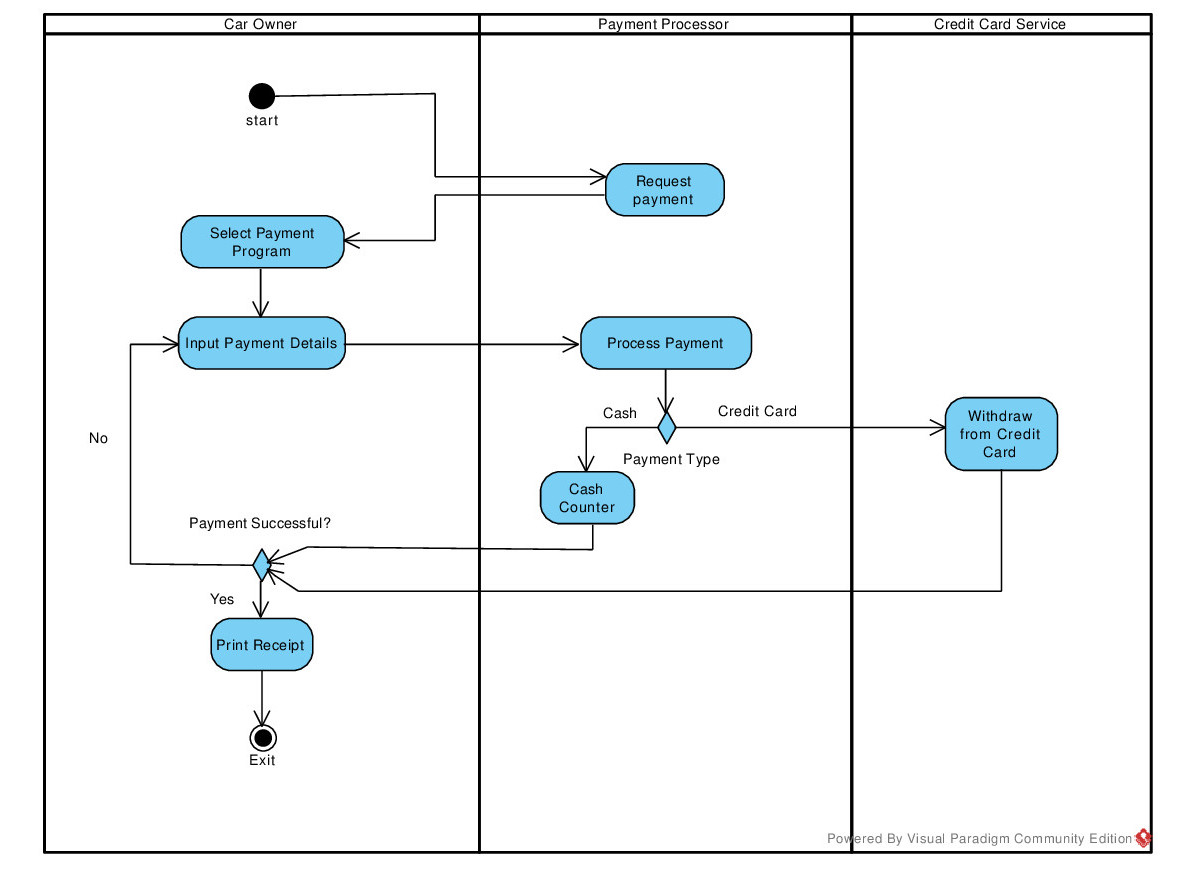


## 1.3 Διεπαφές (interfaces)

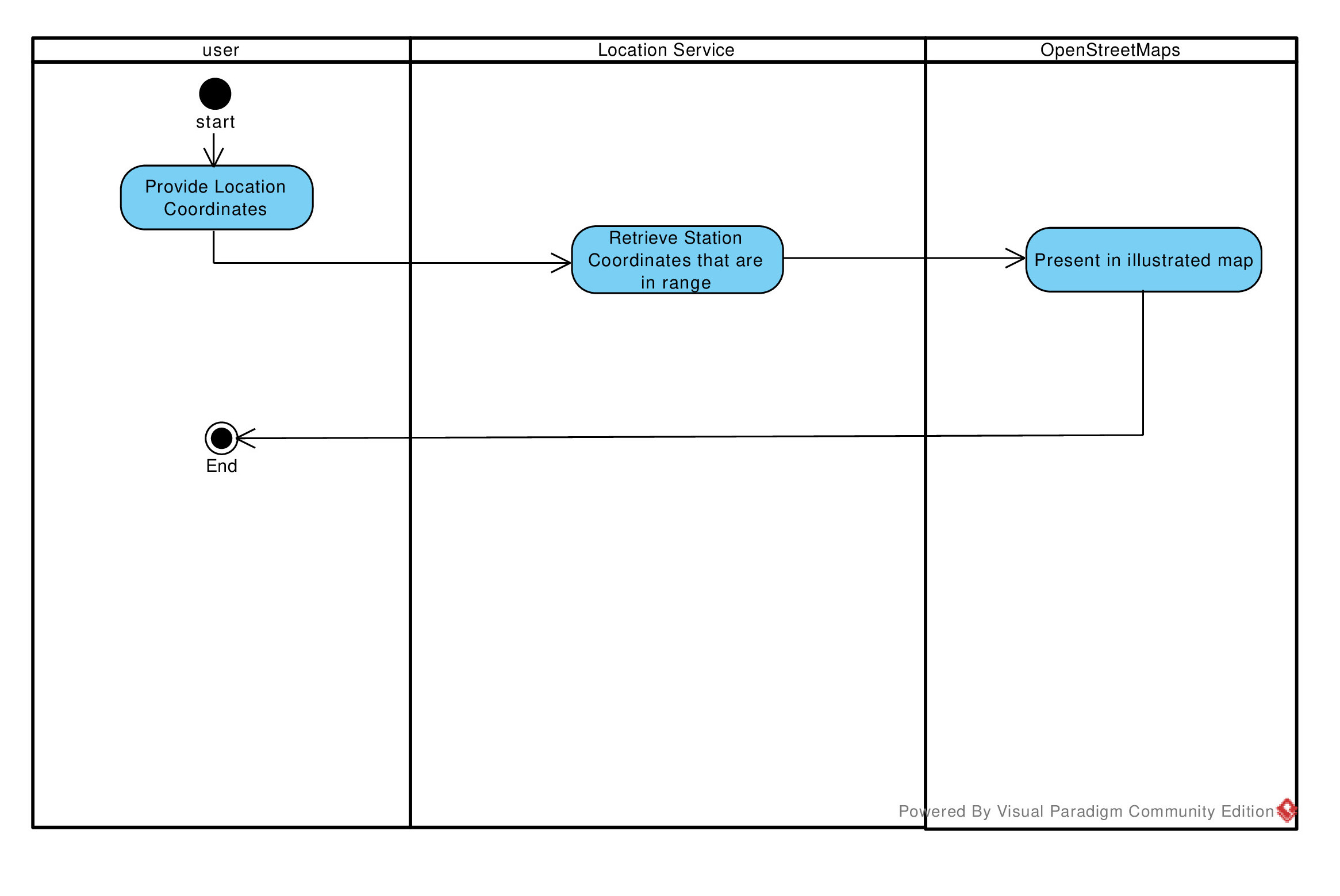
### 1.3.1 Διεπαφές με εξωτερικά συστήματα

Για την ασφαλή και αποδοτική υλοποίηση των λειτουργιών της εφαρμογής χρησιμοποιούμε εξωτερικά συστήματα για την πραγματοποίηση πληρωμών και για την παρουσίασή χάρτη με τοποθεσίες.

Συγκεκριμένα, για την πραγματοποίηση πληρωμών χρησιμοποιούμε σύστημα εξωτερικής τράπεζας η οποία μας επιτρέπει να επεξεργαζόμαστε πληρωμές μέσω πιστωτικής κάρτας. Στο παρακάτω διάγραμμα φαίνεται ο τρόπος που επιτυγχάνεται αυτό. Στην πρώτη στήλη βρίσκεται ο χρήστης, στην δεύτερη το δικό μας εσωτερικό σύστημα και στην τρίτη το εξωτερικό σύστημα.

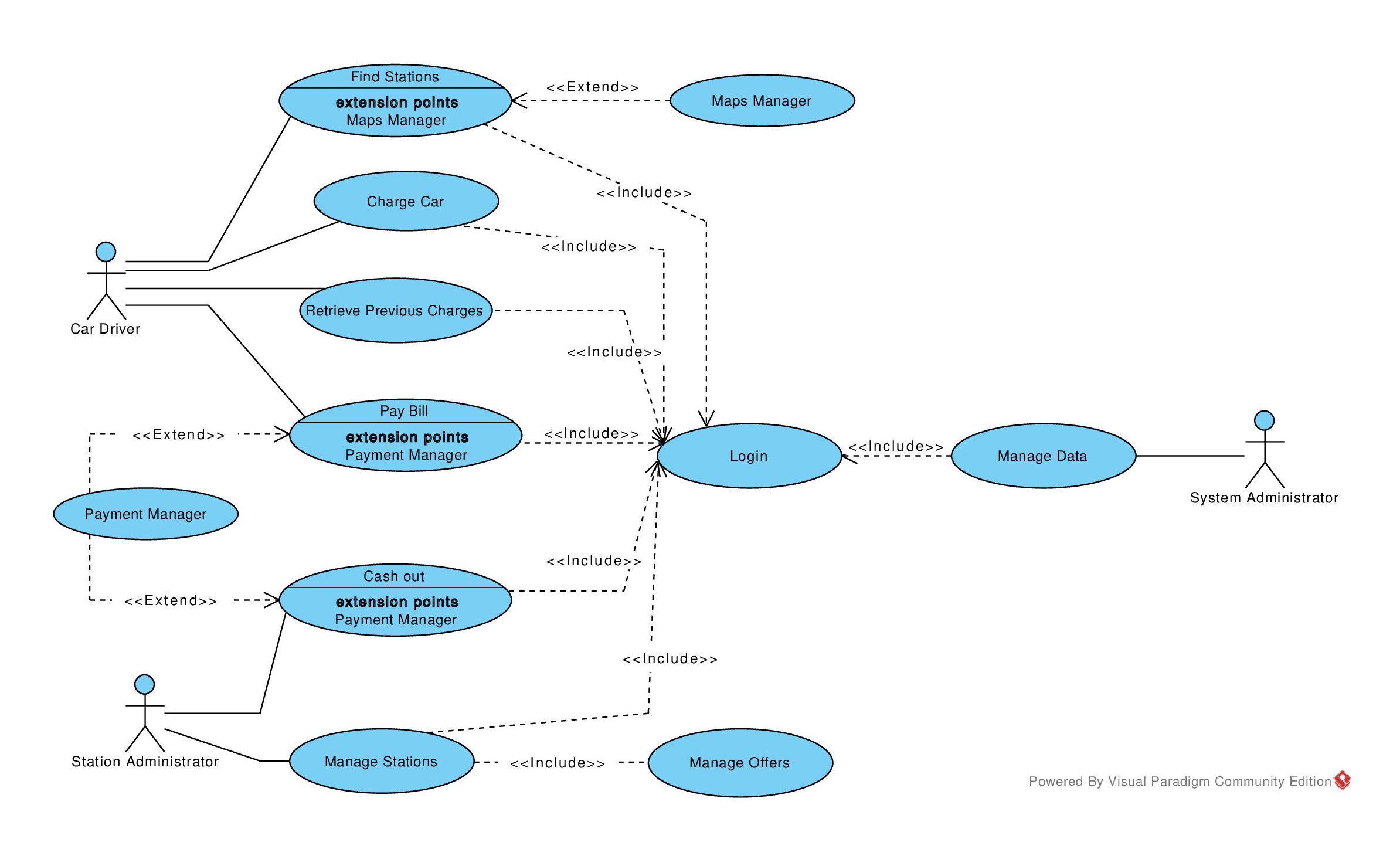


Για την ανασκόπηση γραφικής απεικόνισης χαρτών, χρησιμοποιούμε το OpenStreetMaps. Η διαδικασία φαίνεται στο ακόλουθο διάγραμμα όπου στην πρώτη στήλη βρίσκεται ο χρήστης, στην δεύτερη το εσωτερικό σύστημα και στην τρίτη το εξωτερικό σύστημα.

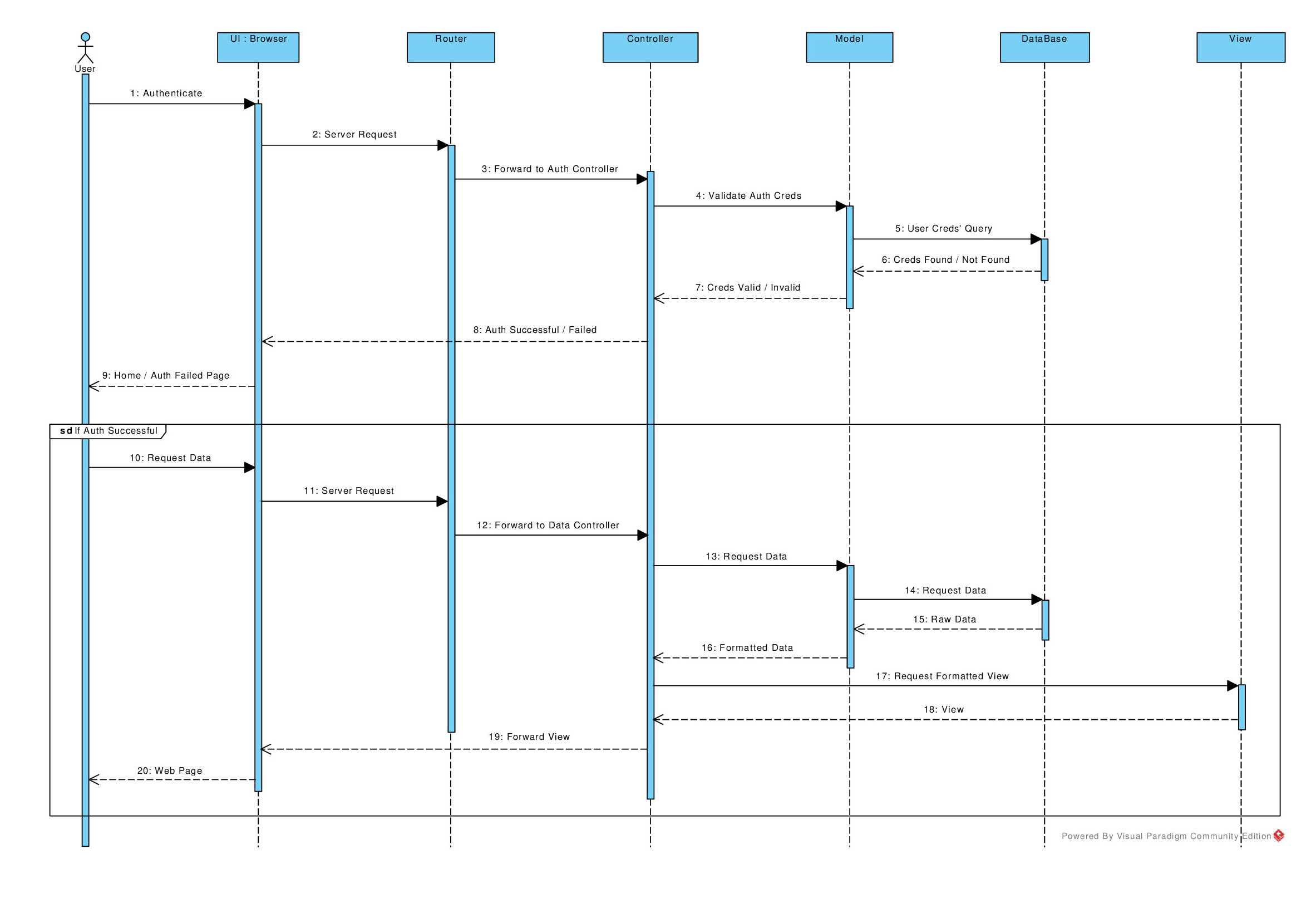


### 1.3.2 Διεπαφές με το χρήστη

Στο παρακάτω UML Use Case Diagram παραθέτουμε μια υψηλού επιπέδου επισκόπηση των λειτουργιών που προσφέρει η εφαρμογή.



Επίσης, μια ακολουθία της αλληλουχίας ανταλλαγής μηνυμάτων μεταξύ του χρήστη και των διάφορων επιμέρους στοιχείων της υπηρεσίας φαίνεται στο ακόλουθο UML Sequence Diagram.



# Αναφορές - πηγές πληροφοριών

*Ν/Α*

# Προδιαγραφές απαιτήσεων λογισμικού

## 3.1 Περιπτώσεις χρήσης

### 3.1.1 Φόρτιση Οχήματος

#### 3.1.1.1 Χρήστες (ρόλοι) που εμπλέκονται

#### Οι χρήστες χωρίζονται σε οδηγούς οχημάτων και ιδιοκτήτες/διαχειριστές σταθμών φόρτισης.

#### 3.1.1.2 Προϋποθέσεις εκτέλεσης

#### Για να έχουν πρόσβαση στην εφαρμογή οι χρήστες πρέπει να εγγραφούν με έγκυρα στοιχεία. Μετά την εγγραφή, οι χρήστες χρειάζεται να εισάγουν τα στοιχεία των οχημάτων τους ώστε να μπορούν να εκτελέσουν γεγονότα φόρτισης μέσω της εφαρμογής.

#### 3.1.1.3 Περιβάλλον εκτέλεσης

#### Το περιβάλλον εκτέλεσης συνίσταται ως διαδικτυακή διεπαφή του χρήστη που αλληλοεπιδρά με την Βάση δεδομένων.

#### 3.1.1.4 Δεδομένα εισόδου

Οι ιδιοκτήτες ηλεκτρικών οχημάτων, αναζητούν σταθμούς φόρτισης με επιλεγμένα φίλτρα που παρέχει η διαδικτυακή διεπαφή, όπως εύρος τιμής, προσφορές και τοποθεσία. Αφού επιλέξουν έναν σταθμό, καλούνται να επιλέξουν το όχημα το οποίο θέλουν να φορτίσουν με επιπλέον προαιρετικές επιλογές όπως διάρκεια ή επιθυμητό ποσοστό φόρτισης. Τα δεδομένα που εισάγουν οι χρήστες ελέγχονται ως προς την ορθότητα τους και την αντιστοίχιση σε πραγματικά δεδομένα στην βάση. Ταυτόχρονα, επεξεργάζονται κατάλληλα ώστε να εγγυηθούν την ασφάλεια του συστήματος από πιθανόν επιθέσεις (CSRS, XSS, SQL Injection κ.α.).

#### 3.1.1.5 Παράμετροι

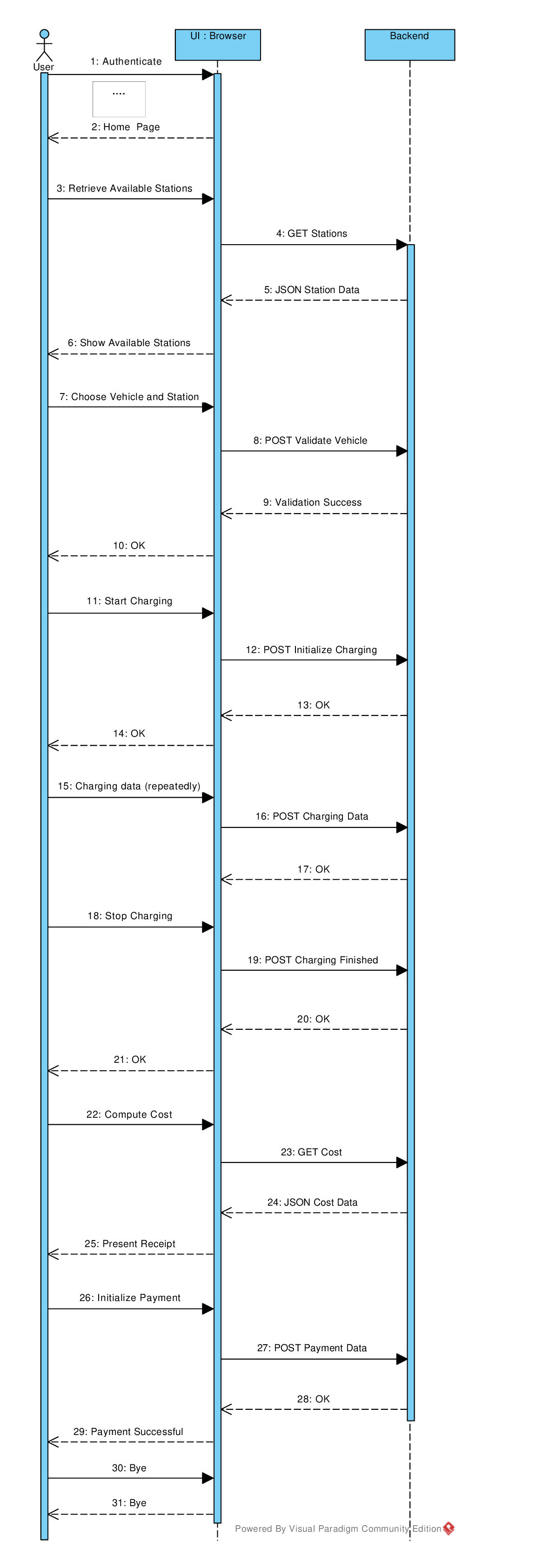
Οι παράμετροι που ορίσαμε παραπάνω πρέπει να αντιστοιχούν σε πραγματικά δεδομένα στην βάση καθώς και να έχουν έγκυρη σύνταξη. Συγκεκριμένα, οι σταθμοί (μαζί με τις προσφορές τους) και τα οχήματα που θα επιλεγούν πρέπει να αντιστοιχίζονται σε οντότητες στην βάση δεδομένων.

#### 3.1.1.6 Αλληλουχία ενεργειών - επιθυμητή συμπεριφορά

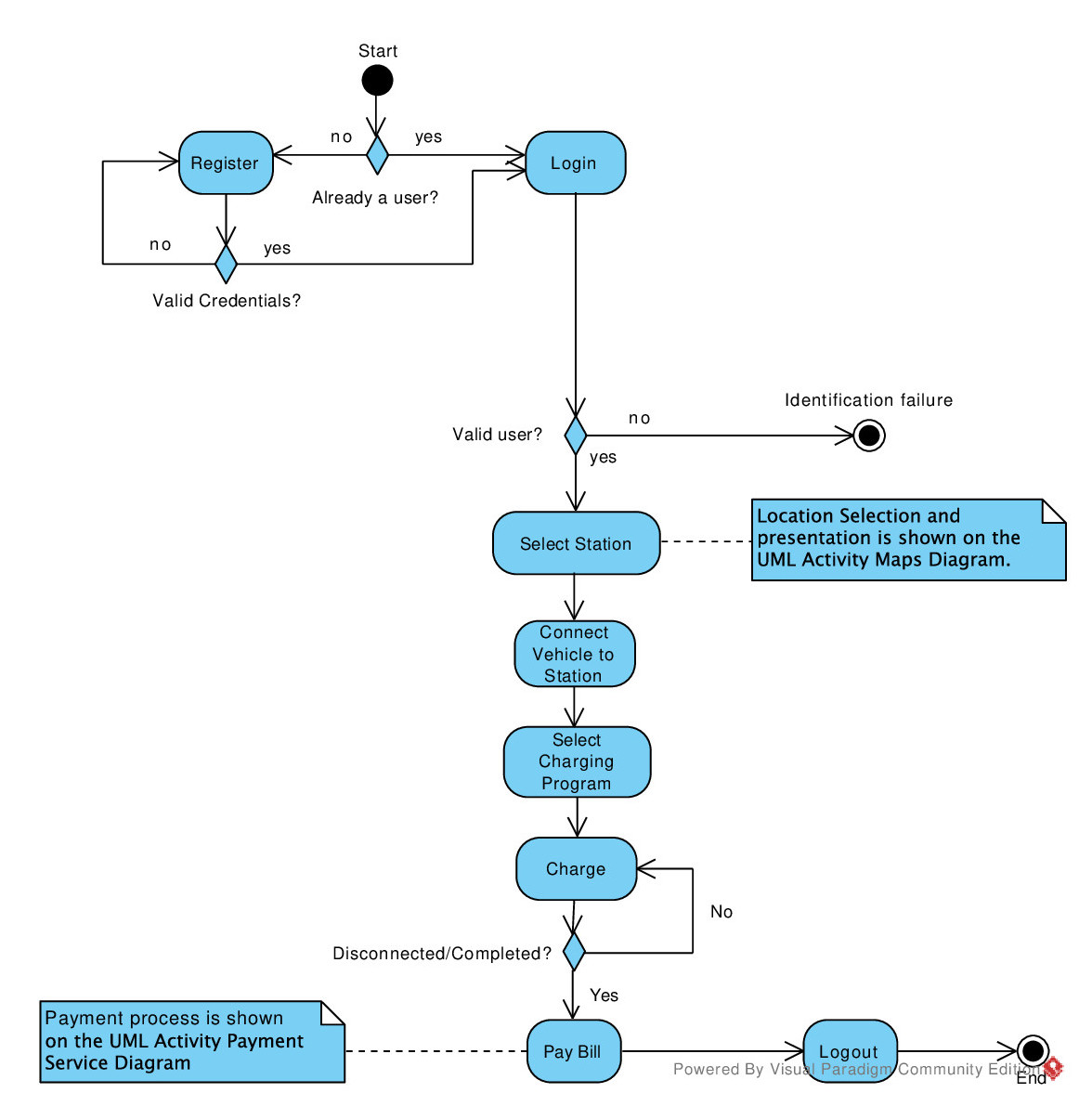
Η διαδικασία φόρτισης περιλαμβάνει τα βήματα:

1. Login/Register.
2. Υπολογισμός αναμενόμενου κόστους φόρτισης για κάθε σταθμό.
3. Αναγνώριση των στοιχείων του οχήματος.
4. Έναρξη φόρτισης με επιλογή προγράμματος χρέωσης από τα διατιθέμενα.
5. Καταγραφή γεγονότων και δεδομένων φόρτισης.
6. Υπολογισμός κόστους φόρτισης που ολοκληρώνεται.
7. Πληρωμή με μετρητά ή κάρτα.
8. Συλλογή δεδομένων οχήματος σε κάθε φόρτιση.

Παρακάτω, παραθέτουμε ένα UML Sequence Diagram το οποίο περιγράφει την διαδικασία φόρτισης. Είναι μια επέκταση του γενικού διαγράμματος που παρουσιάστηκε στην Ενότητα *1.3.2* όπου οι λειτουργίες των Router, Controller, Model, Database, View συμπυκνώνονται στην λειτουργία Model.



Επίσης, παραθέτουμε το παρακάτω UML Activity Diagram το οποίο δίνει μια εικόνα της λειτουργίας εξ ’ολοκλήρου από την πλευρά του χρήστη.



#### 3.1.1.7 Δεδομένα εξόδου

Η αλληλουχία ενεργειών που αφορά την έξοδο εμπεριέχεται στα διαγράμματα της παραγράφου 3.1.1.6. Συγκεκριμένα, για την φόρτιση δημιουργούνται δεδομένα για το γεγονός της φόρτωσης, των διαδοχικών στατιστικών φόρτισης που παράγονται ανά ένα συγκεκριμένο χρονικό διάστημα ώστε να μπορεί ο χρήστης να παρακολουθεί την διαδικασία της φόρτισης καθώς και τα δεδομένα πληρωμών.

#### 3.1.1.8 Παρατηρήσεις

Ν/Α

### 3.1.2 Παρουσίαση δεδομένων στο Dashboard

#### 3.1.2.1 Χρήστες (ρόλοι) που εμπλέκονται

#### Οι χρήστες χωρίζονται σε οδηγούς οχημάτων και ιδιοκτήτες/διαχειριστές σταθμών φόρτισης.

#### 3.1.2.2 Προϋποθέσεις εκτέλεσης

Για να έχουν πρόσβαση στην εφαρμογή οι χρήστες πρέπει να εγγραφούν με έγκυρα στοιχεία. Μετά την εγγραφή, οι οδηγοί ηλεκτρικών οχημάτων χρειάζεται να εισάγουν τα στοιχεία των οχημάτων τους ενώ οι ιδιοκτήτες σταθμών φόρτισης πρέπει να εισάγουν τα στοιχεία των σταθμών τους. Με την προϋπόθεση ύπαρξης επάρκειας δεδομένων, στη συνέχεια παρουσιάζονται στατιστικά χρήσης όπως γεγονότα φόρτισης και πληρωμές.

#### 3.1.2.3 Περιβάλλον εκτέλεσης

#### Το περιβάλλον εκτέλεσης συνίσταται ως διαδικτυακή διεπαφή του χρήστη που αλληλοεπιδρά με την Βάση δεδομένων.

#### 3.1.2.4 Δεδομένα εισόδου

#### Τα δεδομένα εισόδου αφορούν υπάρχον δεδομένα που προέρχονται από την βάση δεδομένων. Συγκεκριμένα, αφορούν γεγονότα φόρτισης (τρέχων και προηγουμένων) καθώς και πληρωμών. Συνεπώς, δεν επιδέχονται κάποιον έλεγχο εγκυρότητας. Ταυτόχρονα, δίνεται η δυνατότητα στον χρήστη να πληρώσει λογαριασμούς με την πιστωτική του κάρτα όπου ο έλεγχος της εγκυρότητας των δεδομένων εξουσιοδοτείται στην συνεργαζόμενη υπηρεσία πληρωμών.

#### 3.1.2.5 Παράμετροι

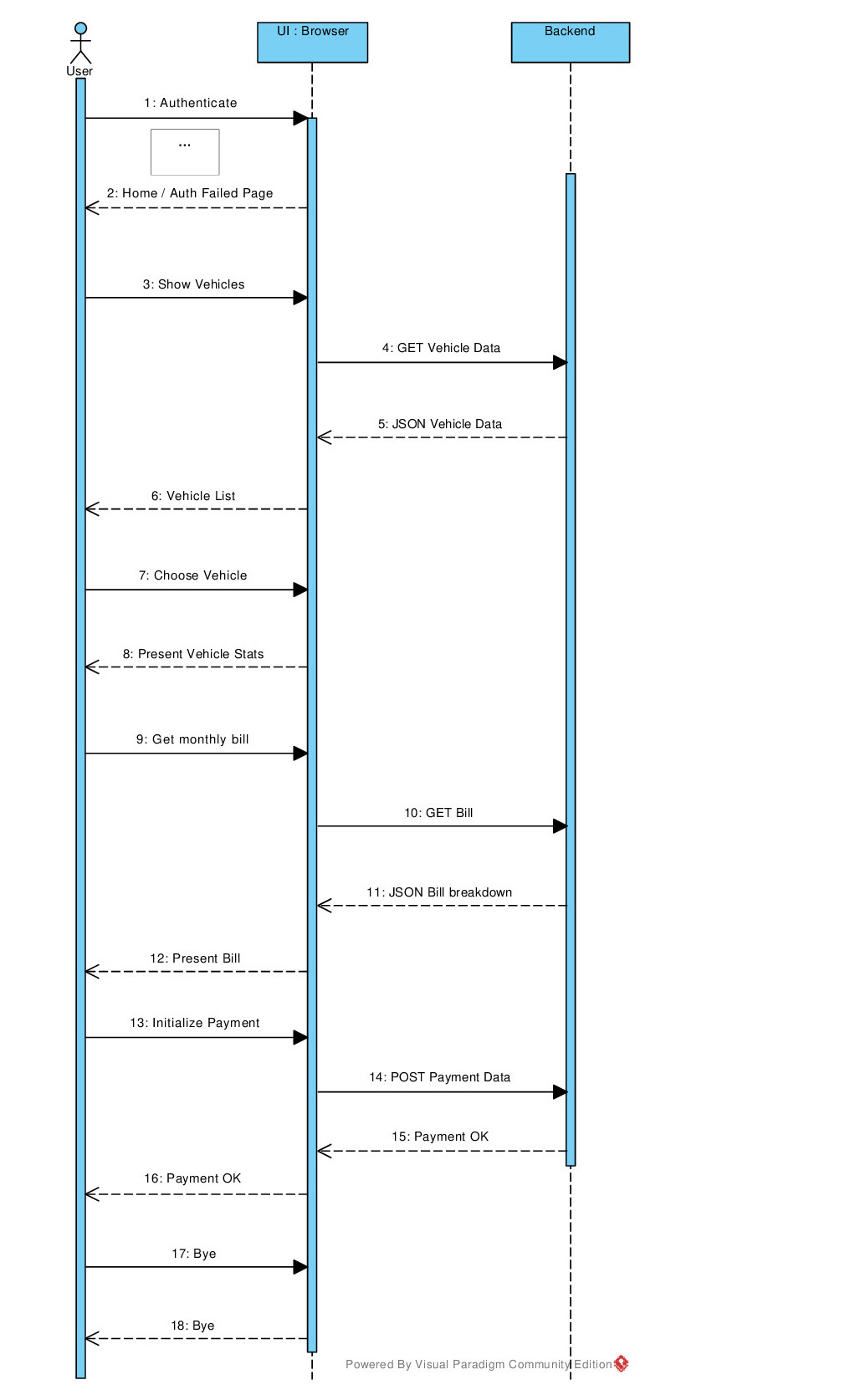
Η μόνη παράμετρος που δίνεται από τον χρήστη είναι η πιστωτική κάρτα της οποίας ο έλεγχος εξουσιοδοτείται σε εξωτερική υπηρεσία.

#### 3.1.2.6 Αλληλουχία ενεργειών - επιθυμητή συμπεριφορά

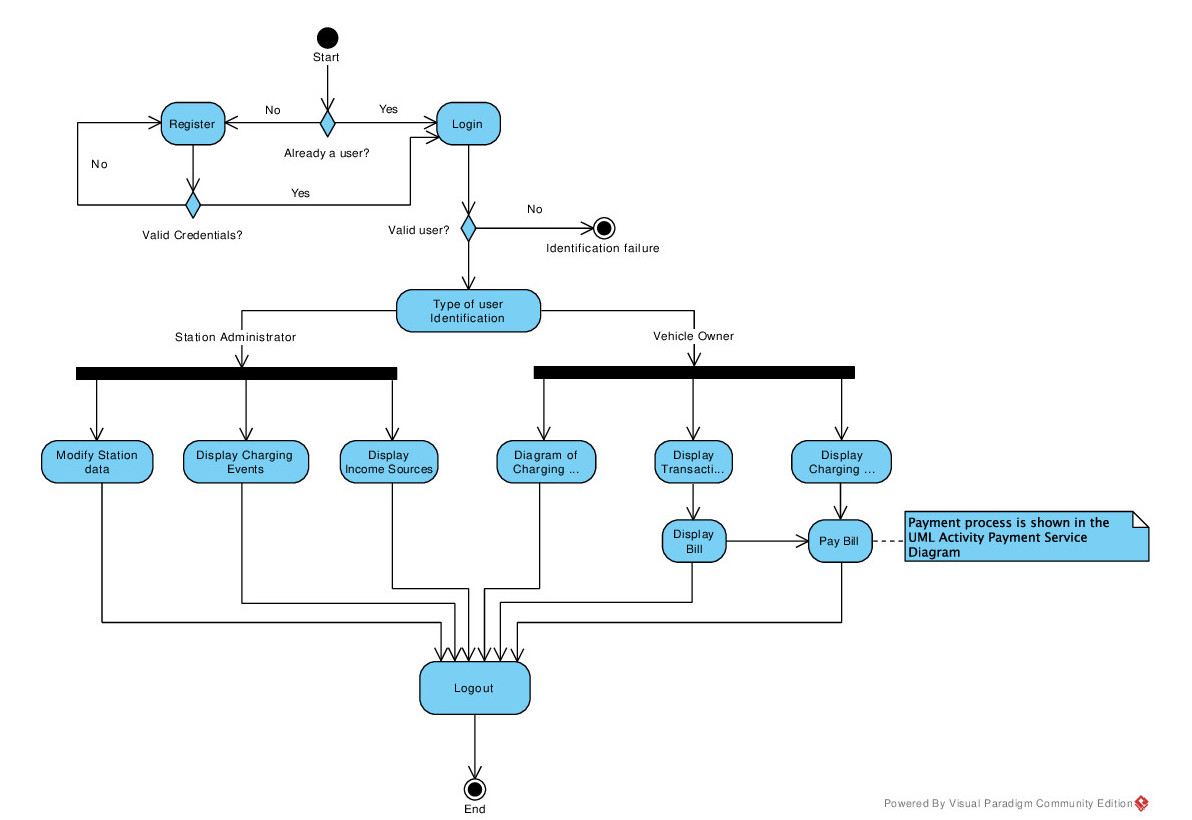
Η διαδικασία χρήσης του dashboard εμπεριέχει τα ακόλουθα βήματα:

1. Login.
2. Επιλογή οχήματος.
3. Απεικόνιση δεδομένων/γεγονότων φόρτισης σε διάγραμμα.
4. Έκδοση περιοδικού λογαριασμού ανά όχημα.
5. Πληρωμή.

Ακολουθεί ένα UML Sequence Diagram το οποίο περιγράφει την αλληλεπίδραση του χρήστη με το dashboard. Είναι μια επέκταση του γενικού διαγράμματος που παρουσιάστηκε στην Ενότητα *1.3.2* όπου οι λειτουργίες των Router, Controller, Model, Database, View συμπυκνώνονται στην λειτουργία Model.



Επίσης, παραθέτουμε ένα UML Activity Diagram το οποίο περιγράφει την λειτουργία εξ 'ολοκλήρου από την μεριά του χρήστη.



#### 3.1.2.7 Δεδομένα εξόδου

Η αλληλουχία ενεργειών που αφορά την έξοδο εμπεριέχεται στα διαγράμματα της παραγράφου 3.1.2.6. Συγκεκριμένα, δημιουργούνται νέα δεδομένα στην βάση δεδομένων όταν πραγματοποιείται πληρωμή ενός λογαριασμού. Ταυτόχρονα, δεδομένα εξόδου θεωρούμε και τα στατιστικά που απεικονίζονται για τα οχήματα του χρήστη.

#### 3.1.2.8 Παρατηρήσεις

Ν/Α

## 3.2 Απαιτήσεις επιδόσεων

### Το παρόν σύστημα καλείται να επεξεργάζεται δεδομένα μεγάλης κλίμακας και ταχείου ρυθμού. Συγκεκριμένα, το μεγαλύτερο bottleneck της εφαρμογής είναι τα συνεχή δεδομένα φόρτισης που διατίθενται κατά την διάρκεια φόρτισης ενός οχήματος ώστε να παρέχουν μια συνεχή εικόνα της διαδικασίας φόρτισης στον χρήστη. Οι υποδομές που υποστηρίζουν την λειτουργία του συστήματος πρέπει να επιλεγούν με τέτοιο τρόπο ώστε να μπορούν να διεκπεραιώνουν την έντονη ροή δεδομένων σε πραγματικούς χρόνους. Συγκεκριμένα, επιλέγουμε να έχουμε μέσο χρόνο καθυστέρησης το 1 δευτερόλεπτο με βάση τα Miller Response-Time Limits.

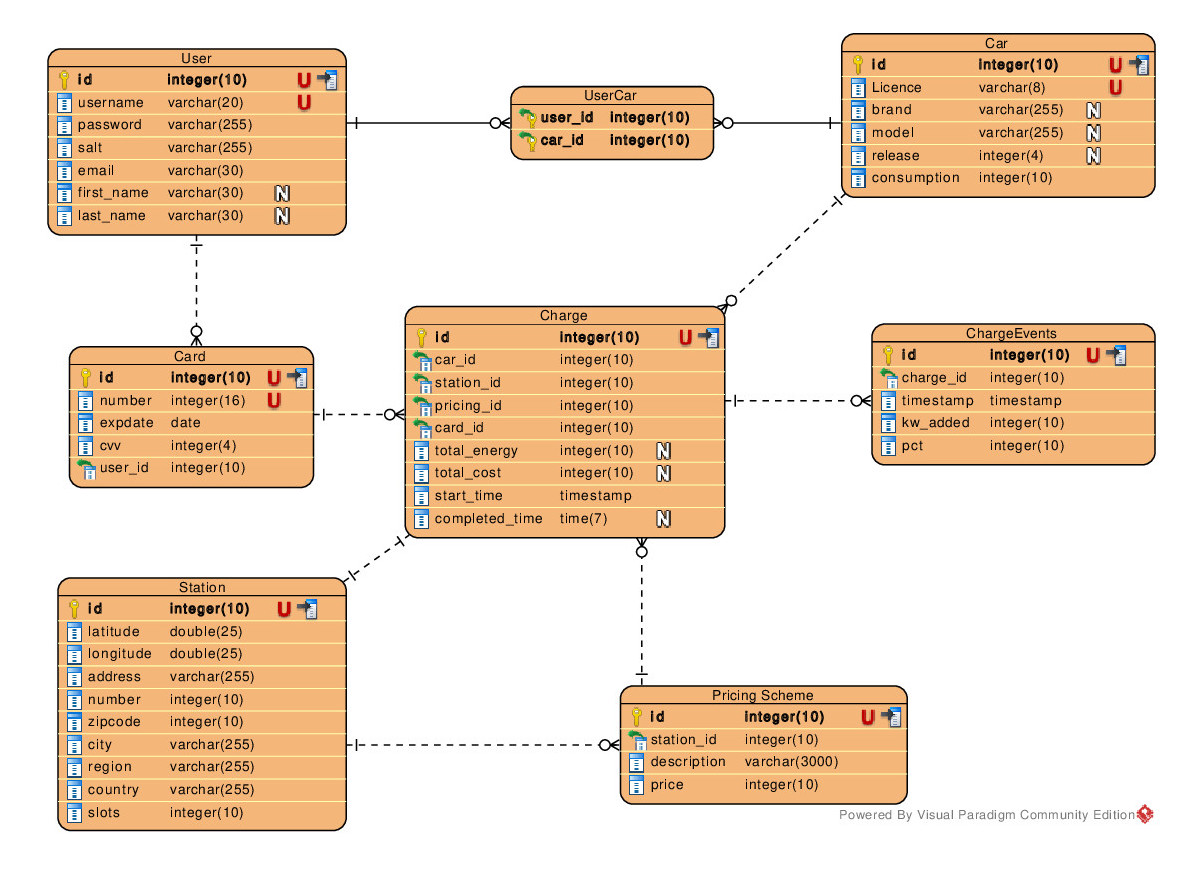
## 3.3 Απαιτήσεις οργάνωσης δεδομένων

### 3.3.1 Απαιτήσεις και περιορισμοί πρόσβασης σε δεδομένα

Η εφαρμογή απαιτείται να συμβιβάζεται με τους τρέχοντες κανονισμούς σχετικά με την προστασία προσωπικών δεδομένων (GDPR). Συγκεκριμένα, πρόσβαση στα δεδομένα των χρηστών, όπως δεδομένα πληρωμών, τοποθεσίας και φορτίσεων πρέπει να είναι διαθέσιμα μονάχα στους χρήστες στους οποίους ανήκουν τα οχήματα ή οι σταθμοί που οδήγησαν στην δημιουργία τους. Τυχόν προσβάσεις σε δεδομένα χρηστών από τους διαχειριστές της εφαρμογής επιτρέπεται να γίνονται μόνο με την ρητή συγκατάθεση των χρηστών. Ταυτόχρονα, οι χρήστες έχουν την δυνατότητα να αφαιρέσουν οποιαδήποτε στιγμή την συγκατάθεση τους.

### 3.3.2 Μοντέλο Δεδομένων

Παραθέτουμε το ER διάγραμμα που περιγράφει το μοντέλο των δεδομένων που αποθηκεύει η εφαρμογή ώστε να εκπληρώσει με ακρίβεια τις υπηρεσίες της.



## 3.4 Περιορισμοί σχεδίασης

Υπάρχει απαίτηση συμμόρφωσης με τα πρότυπα σχετικά με την προστασία των προσωπικών δεδομένων, σύμφωνα με το GDPR. Επίσης, δεδομένου ότι το CLI θα υλοποιηθεί μέσω Python, υπάρχει ως προαπαιτούμενο η εγκατάσταση της. Τέλος, το framework “Django” της Python που θα χρησιμοποιηθεί για την υλοποίηση του REST API, απαιτεί πως τα μοντέλα και οι οντότητες που ορίζουν να ξεκινούν από κεφαλαίο γράμμα, ενώ τα test cases πρέπει να είναι ορίζονται σε αρχεία που ξεκινούν με την λέξη “test” και να είναι υπό-κλάσεις της “*django.test.TestCase”* .

## 3.5 Λοιπές απαιτήσεις

### 3.5.1 Απαιτήσεις διαθεσιμότητας λογισμικού

### Το σύστημα καλείται να επεξεργάζεται τα δεδομένα, να διεκπεραιώνει τους ζητούμενους υπολογισμούς και να αποκρίνεται σε κατάλληλους χρόνους με βάση τις προβλεπόμενες συνθήκες λειτουργίας. Παράλληλα, θα πρέπει να έχει την δυνατότητα προσαρμοστικότητας σε διαφορετικά περιβάλλοντα δίχως την απαίτηση αλλαγών από το χρήστη. Ταυτόχρονα, θα πρέπει να μπορεί να μεταφερθεί σε διαφορετικές υποδομές χωρίς αλλαγές στο λογισμικό, γεγονός που θα επιτρέψει την διατήρηση ποιότητας λειτουργίας σε περίπτωση αύξησης του αριθμού χρηστών μέσω scaling των υποδομών. Επιπλέον, το σύστημα θα πρέπει να έχει ένα απλό και κατανοητό interface ώστε να μπορεί να χρησιμοποιηθεί και από ομάδες που έχουν δυσκολίες πρόσβασης στην τεχνολογία όπως ΑμεΑ και ηλικιωμένοι. Συμπληρωματικά, κρίνεται αναγκαίο να πραγματοποιούνται τακτικά backups των δεδομένων, και να υπάρχει προκαθορισμένη διαδικασία επαναφοράς αυτών σε περίπτωση data corruption. Τέλος, το σύστημα θα πρέπει να επιδιώκει να χρησιμοποιεί μικρές απαιτήσεις δεδομένων έτσι ώστε να είναι προσβάσιμο και σε περιπτώσεις χρηστών με μικρό bandwidth.

### 3.5.2 Απαιτήσεις ασφάλειας

### Η εφαρμογή πρέπει να είναι πλήρως συμμορφωμένη με τους υπάρχοντες νόμους και διατάξεις αυτών και κατά συνέπεια μέσω των δικλείδων ασφαλείας που παρέχουμε κανείς εκ των εγγεγραμμένων χρηστών δεν έχει πρόσβαση στα ευαίσθητα δεδομένα άλλων χρηστών, όπως για παράδειγμα οι συναλλαγές αυτών. Η υπηρεσία θα πρέπει να έχουν μεριμνήσει για την τυχόν άμυνα σε επιθέσεις από κακόβουλους χρήστες, είτε αυτές αφορούν επιθέσεις τύπου Web (XSS, CSRF, SQL Injection κ.α.), είτε επιθέσεις τύπου Denial of Service. Επίσης, υπάρχει η ανάγκη να οριστούν διαδικασίες σε περίπτωση Data Breach μέσω ενός Incident Response Team. Ακόμη, είναι σημαντικό να υπάρχει αυτοματοποιημένη διαδικασία ελέγχου της υπολειπόμενης διάρκεια ζωής του issued TLS certificate.

### 3.5.3 Απαιτήσεις συντήρησης

### Αρχικά, η ανάπτυξη του λογισμικού θα πρέπει να γίνεται ακολουθώντας τα τρέχοντα best practices (code reviews, CI/CD, automated testing, multiple deployment environments). Συγκεκριμένα, οι αλλαγές στο λογισμικό θα πρέπει να συνοδευόνται πάντα από επαρκή test cases, καθώς επίσης, θα πρέπει να μην προκαλούν αποτυχίες σε υπάρχον test cases. Επιπλεόν, απαιτείται να οριστούν διαδικασίες για το ορθό deployment της εφαρμογής οι οποίες θα τρέχουν όλους τους απαραίτητους ελέγχους και θα κάνουν deploy την νέα έκδοση της εφαρμογής με το ελάχιστο δυνατό downtime.