



ΧΑΡΟΚΟΠΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ  
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΜΑΤΙΚΗΣ

# Εργασία 2022:

## Κατανεμημένα Συστήματα

Ομάδα 7:

it22033, [Christos Kazakos](#)

it22045, [Konstantinos Katsaras](#)

it22064, [Manousos Linardakis](#)

## Περιεχόμενο:

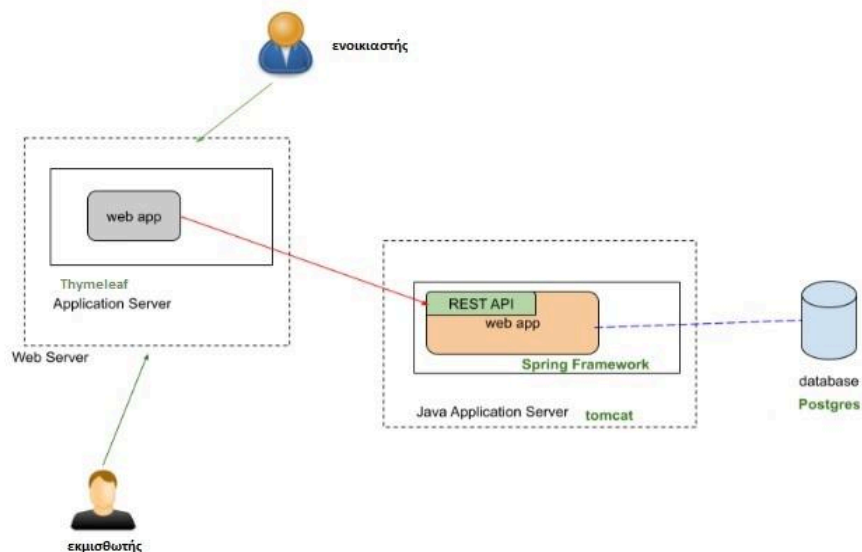
- 1ο Παραδοτέο: Σχεδιασμός
  - Παραδοχές
  - Διάγραμμα Αρχιτεκτονικής
  - Class Diagram
  - ER Diagram
  - Use Case Diagram
  - Activity Diagrams
  - State Diagrams
  - Sequence Diagrams
- 2ο Παραδοτέο: Υλοποίηση
  - Component/Deployment Diagram
  - Repository Link (old)
  - Εγχειρίδιο χρήσης REST API (backend token auth)
- 3ο Παραδοτέο: Σχεδιασμός:
  - Ασφάλεια
  - Επικοινωνία Front - Back
  - Πρόσβαση
  - Screenshots of Frontend
- 4ο Παραδοτέο: Υλοποίηση:
  - Repository links (new)
  - Εγχειρίδιο Χρήσης (backend basic & frontend)

## 1ο Παραδοτέο: Σχεδιασμός

### Παραδοχές:

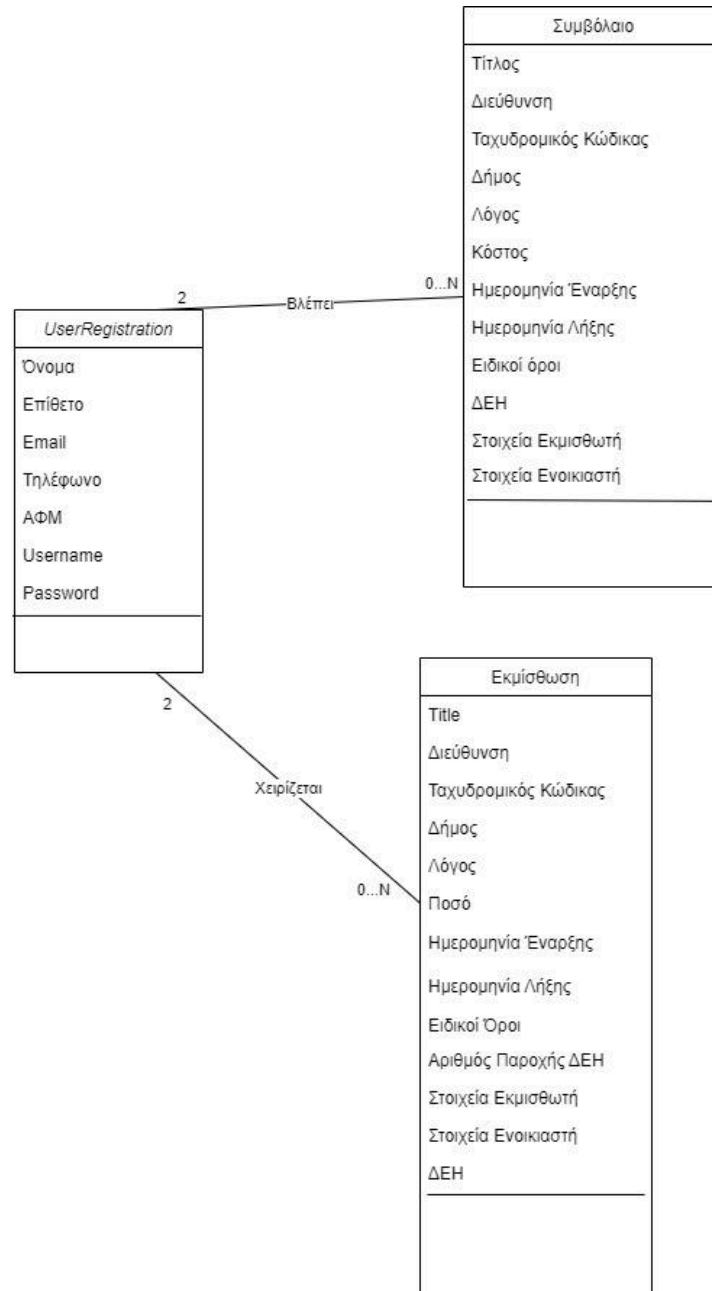
- Η πληροφόρηση για το ακίνητο έχει γίνει είτε από άλλο site, είτε από σχετικές αγγελίες. Η εφαρμογή που θα υλοποιήσουμε στοχεύει μόνο στην έκδοση συμβολαίων για την ενοικίαση του επιλεγμένου ακινήτου.
- Ο ενοικιαστής επικοινωνεί τηλεφωνικά με τον εκμισθωτή για την δήλωση ενδιαφέροντος για το ακίνητο (αφού πρώτα έχει κάνει εγγραφή στο σύστημα). Αφού του πει το username, ο εκμισθωτής παίρνει τα απαραίτητα στοιχεία (του ενοικιαστή) από τη βάση για να συμπληρωσει την εκμίσθωση.
- Ο ενοικιαστής διαβάζει τα στοιχεία της εκμίσθωσης και τότε συμφωνεί ή στέλνει σχόλια. Το συμβόλαιο οριστικοποιείται μόνο αν αποδεχθεί ο ενοικιαστής τα στοιχεία της εκμίσθωσης. Αυτό οδηγεί στη διαγραφή της εκμίσθωσης (από τη βάση) και στην δημιουργία του αντίστοιχου συμβολαίου.
- Τα στοιχεία της εκμίσθωσης μπορούν να ελεγχθούν/συμπληρωθούν από έναν ενοικιαστή/εκμισθωτή (αντίστοιχα).
- Για την συμπλήρωση – δημιουργία της εκμίσθωσης χρειάζεται να καταγραφούν και τα στοιχεία του αντίστοιχου ενοικιαστή, όπως και του εκμισθωτή. Αυτό προϋποθέτει την “ύπαρξή” τους στη βάση πριν την δημιουργία της εκμίσθωσης.
- Οι λειτουργίες για τον εκάστοτε χρήστη είναι διαθέσιμες αφού γίνει login/signup στην αρχή της εφαρμογής (η αυθεντικοποίηση γίνεται με tokens).
- Κάθε εκμίσθωση έχει μόνο έναν εκμισθωτή και ενοικιαστή.
- Ο κάθε χρήστης μπορεί να έχει μόνο έναν ρόλο.

Σχήμα με τη συνολική αρχιτεκτονική όλων των τμημάτων του συστήματος:



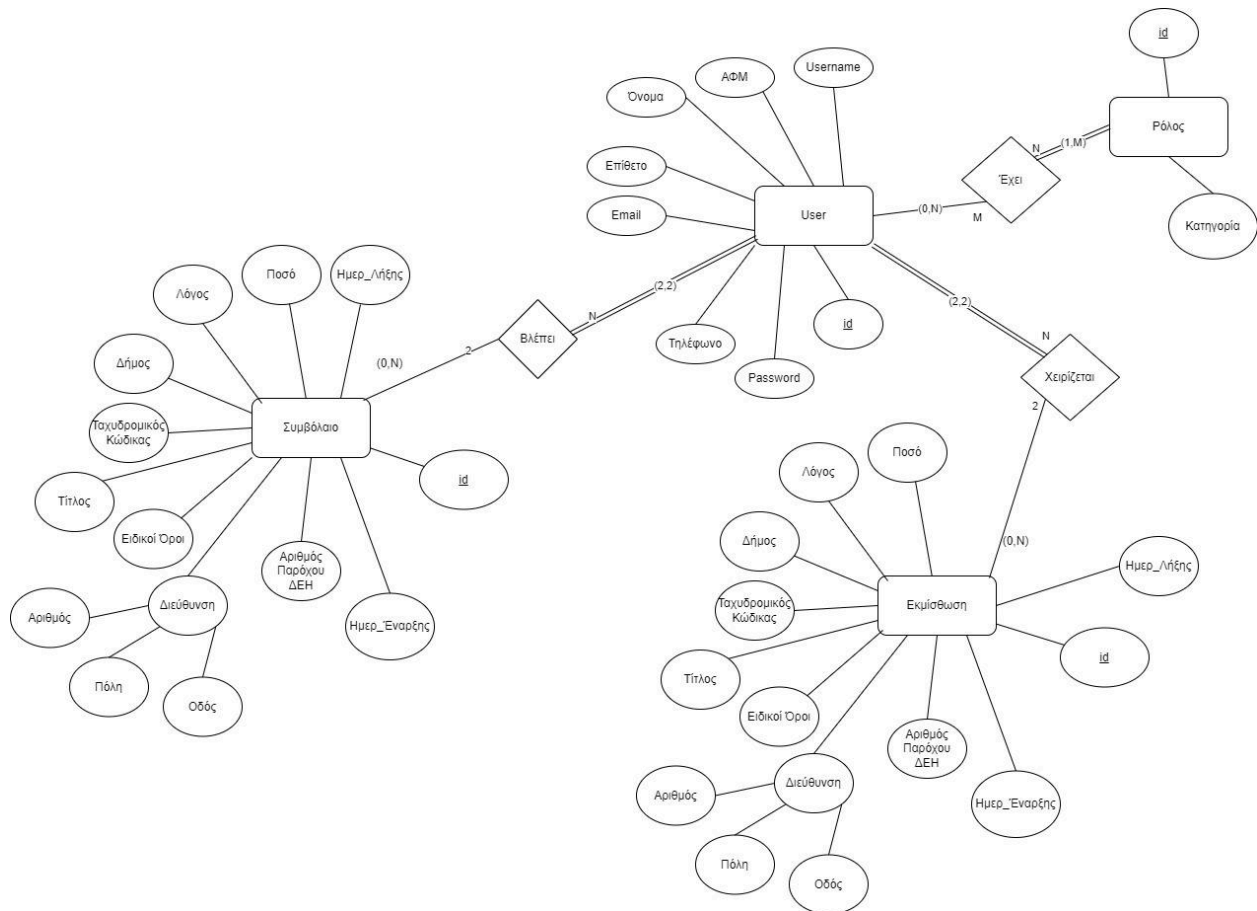
Η βασική δομή για την υλοποίηση και τη διαχείριση ενός καταλόγου χρηστών, χρησιμοποιώντας τυποποιημένες μεθοδολογίες, είναι η εξής:

### Class Diagram:



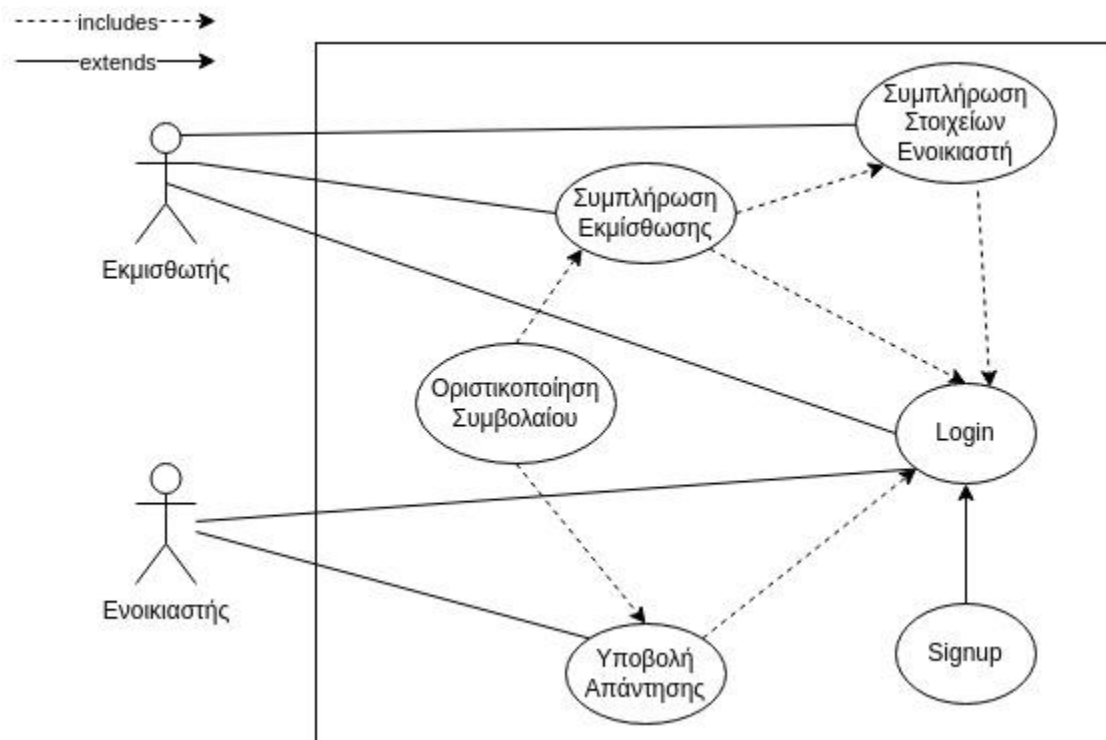
Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

## ER Diagram:



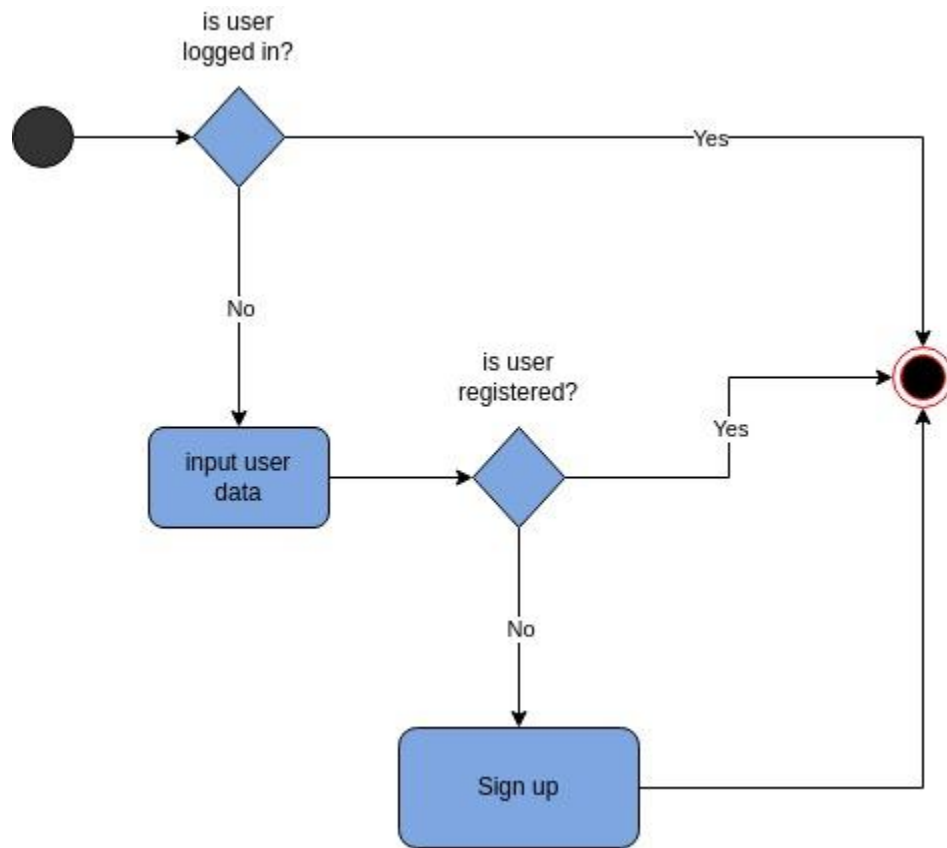
Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

Η αναφορά και σχολιασμός των υπηρεσιών που παρέχονται σε κάθε διαφορετικό ρόλο περιγράφεται με το εξής Use Case Diagram:



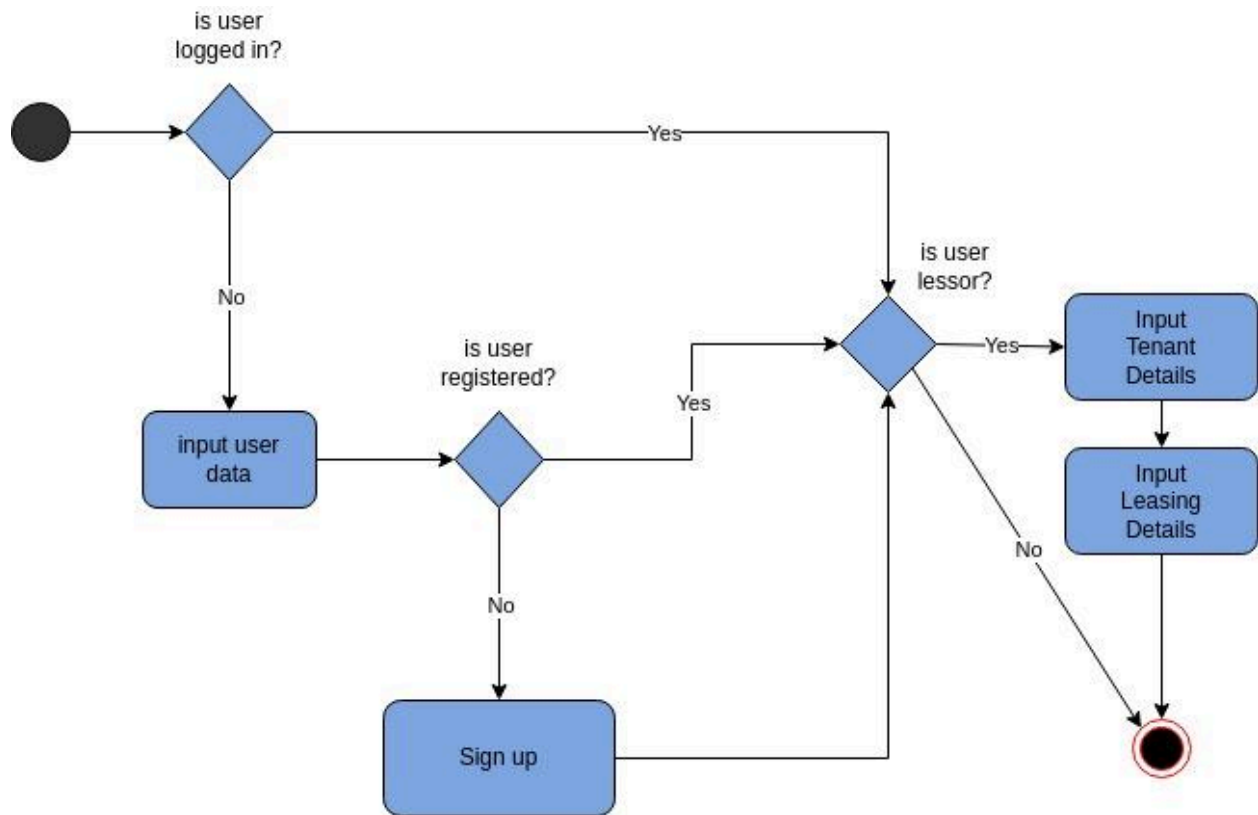
Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

Activity Diagram για τη Λειτουργία “Login” (και “Sign Up”):



Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

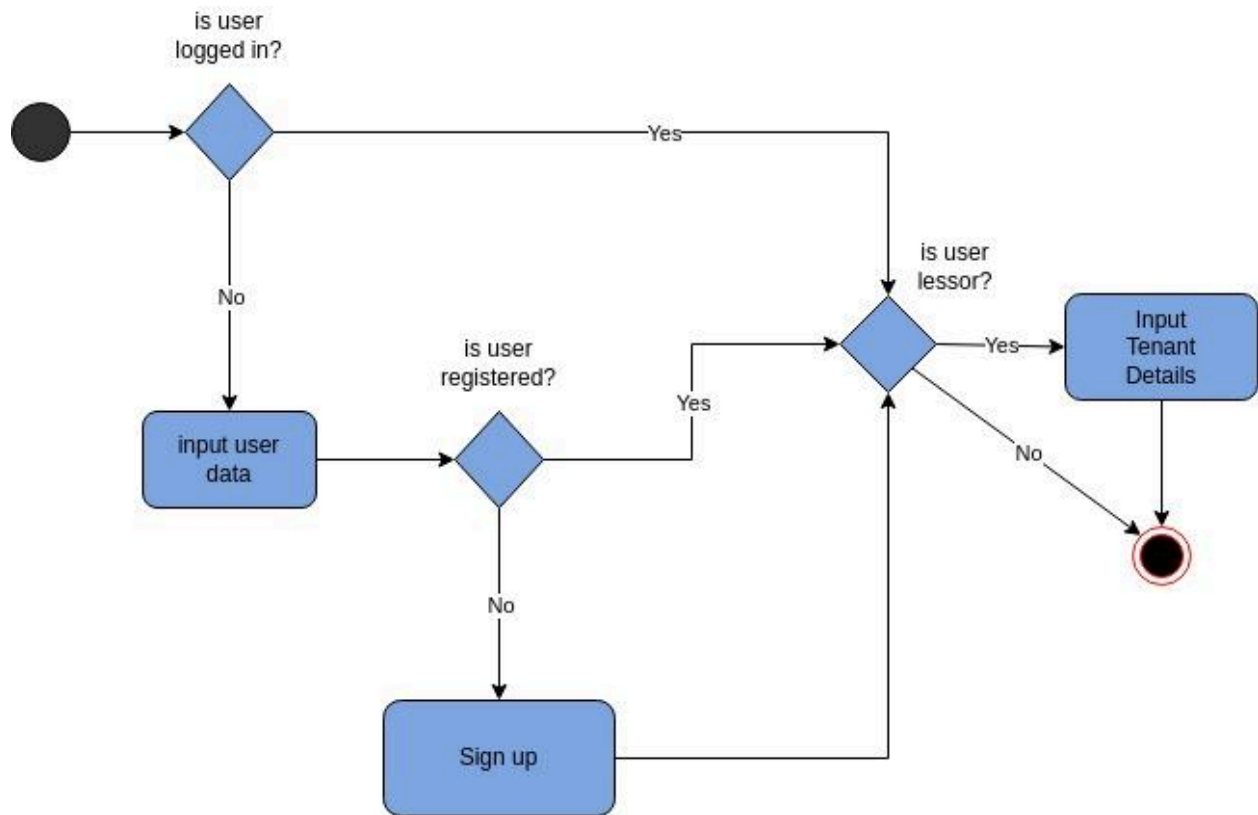
Activity Diagram για τη Λειτουργία “Συμπλήρωση εκμίσθωσης”:



Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

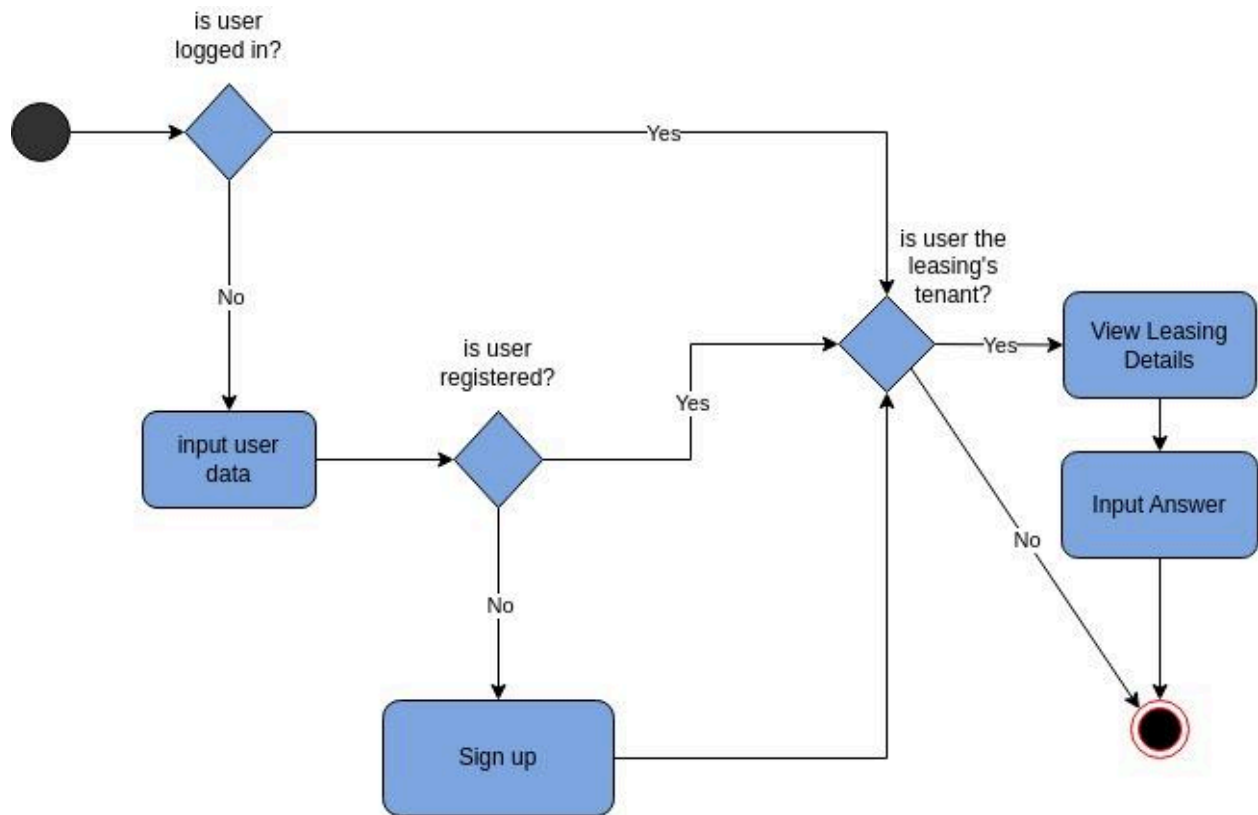


Activity Diagram για τη Λειτουργία “Συμπλήρωση στοιχείων Ενοικιαστή”:



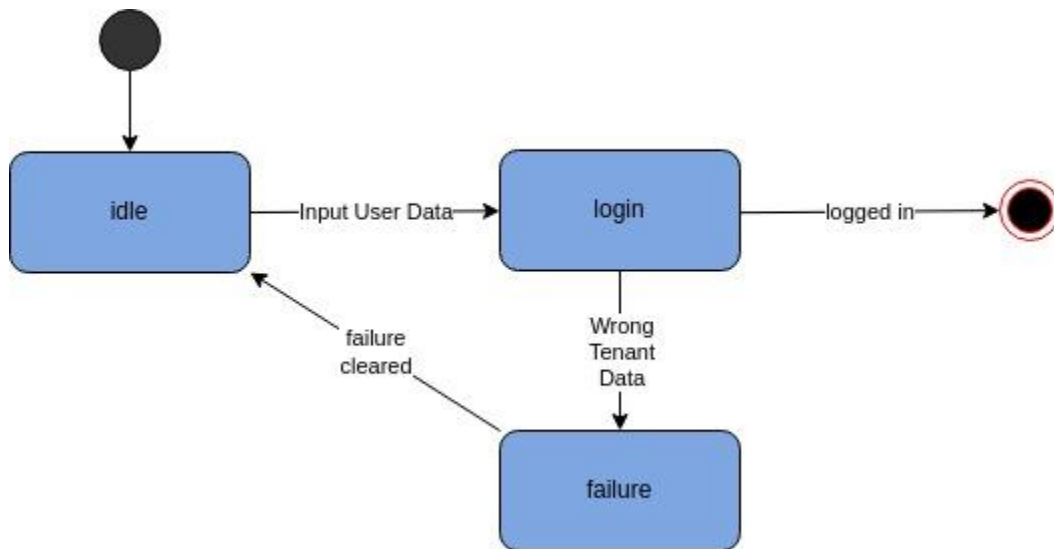
Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

Activity Diagram για τη Λειτουργία “Υποβολή Απάντησης”:



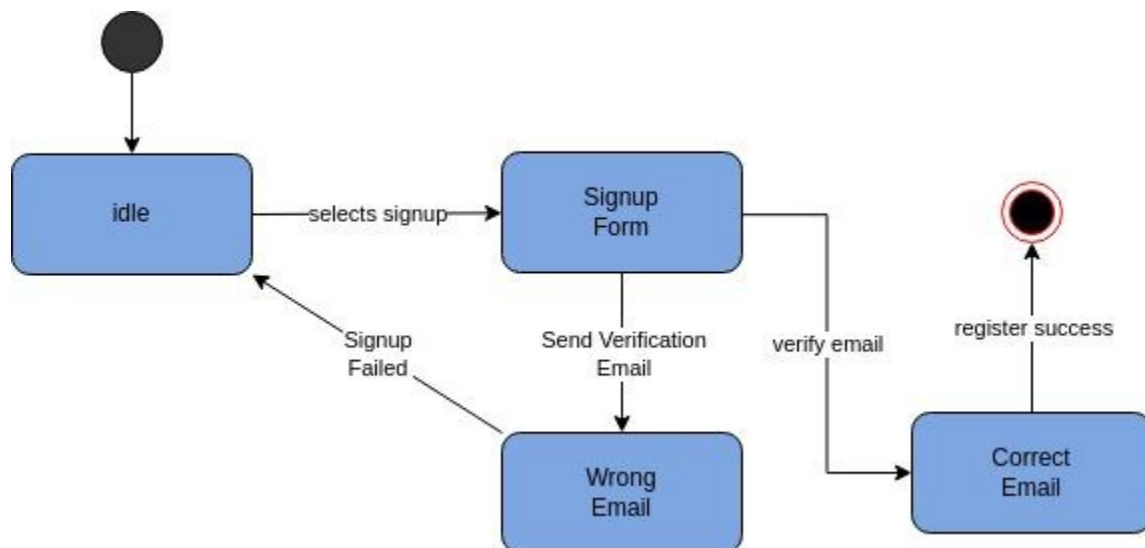
Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

State Diagram για τη Λειτουργία “Login”:



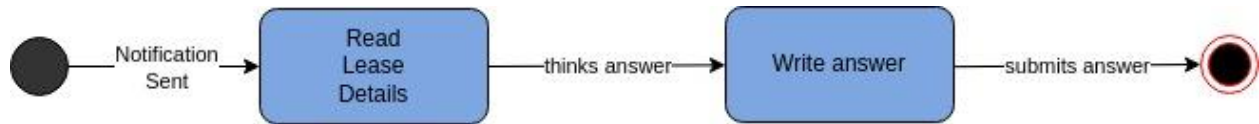
Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

State Diagram για τη Λειτουργία “Signup”:



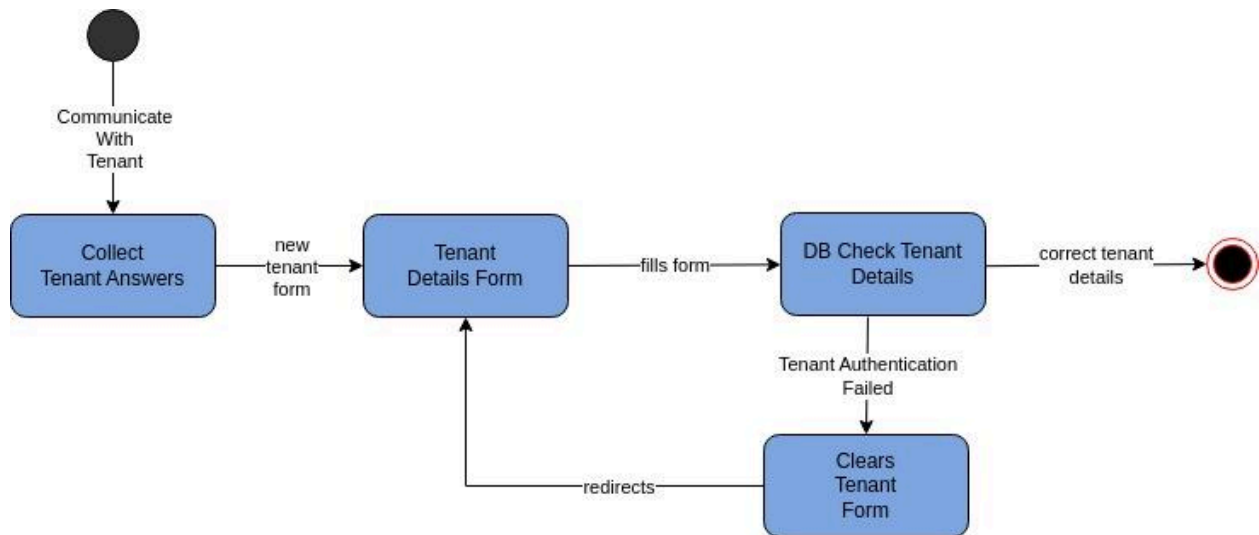
Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

State Diagram για τη Λειτουργία “Υποβολή Απάντησης”:



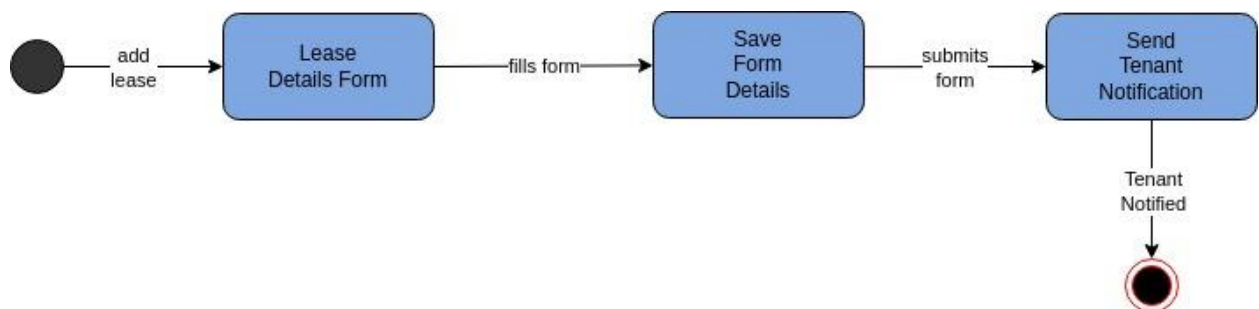
Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

State Diagram για τη Λειτουργία “Συμπλήρωση στοιχείων ενοικιαστή”:



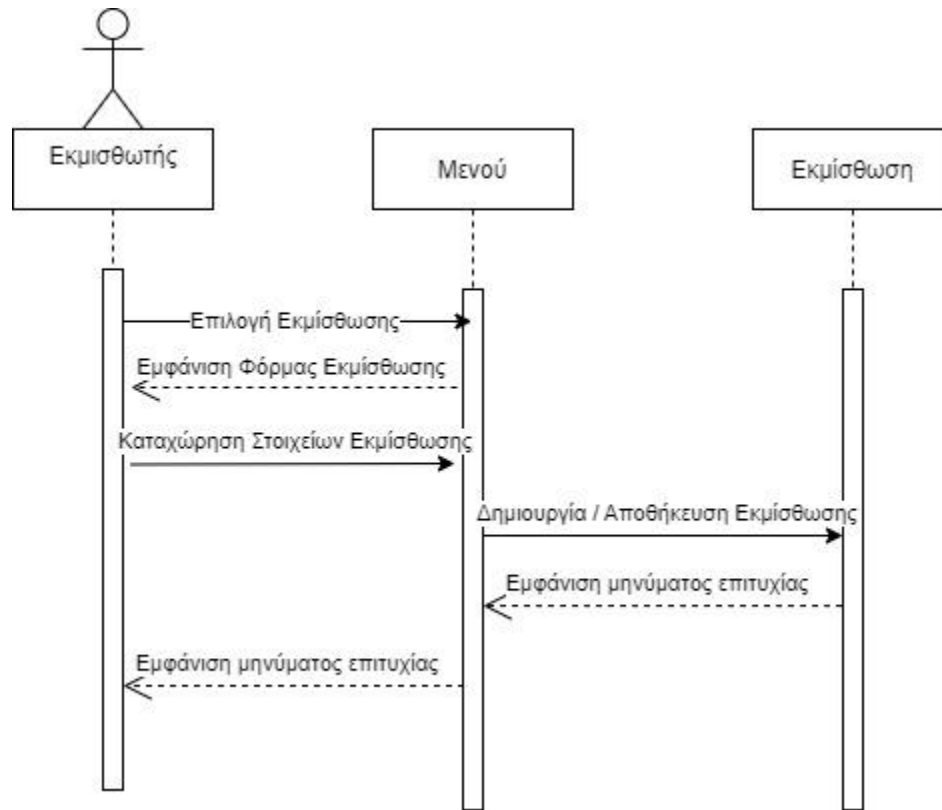
Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

State Diagram για τη Λειτουργία “Συμπλήρωση εκμίσθωσης”:



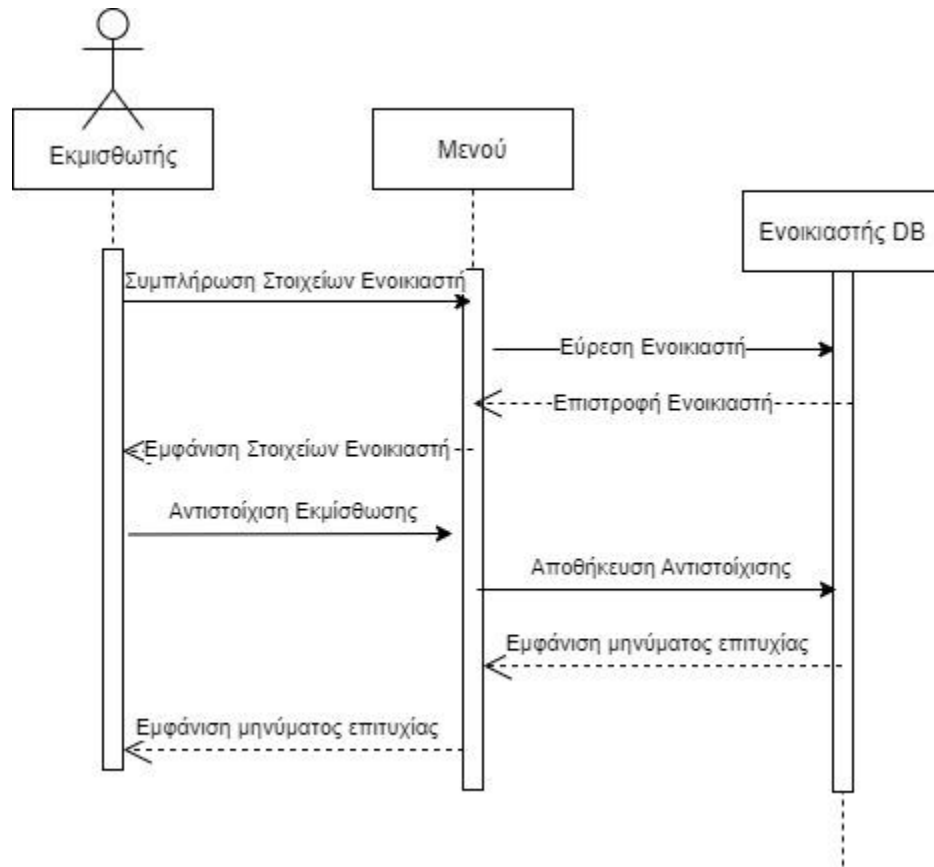
Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

Sequence Diagram για τη Λειτουργία “Συμπλήρωση Εκμίσθωσης”:



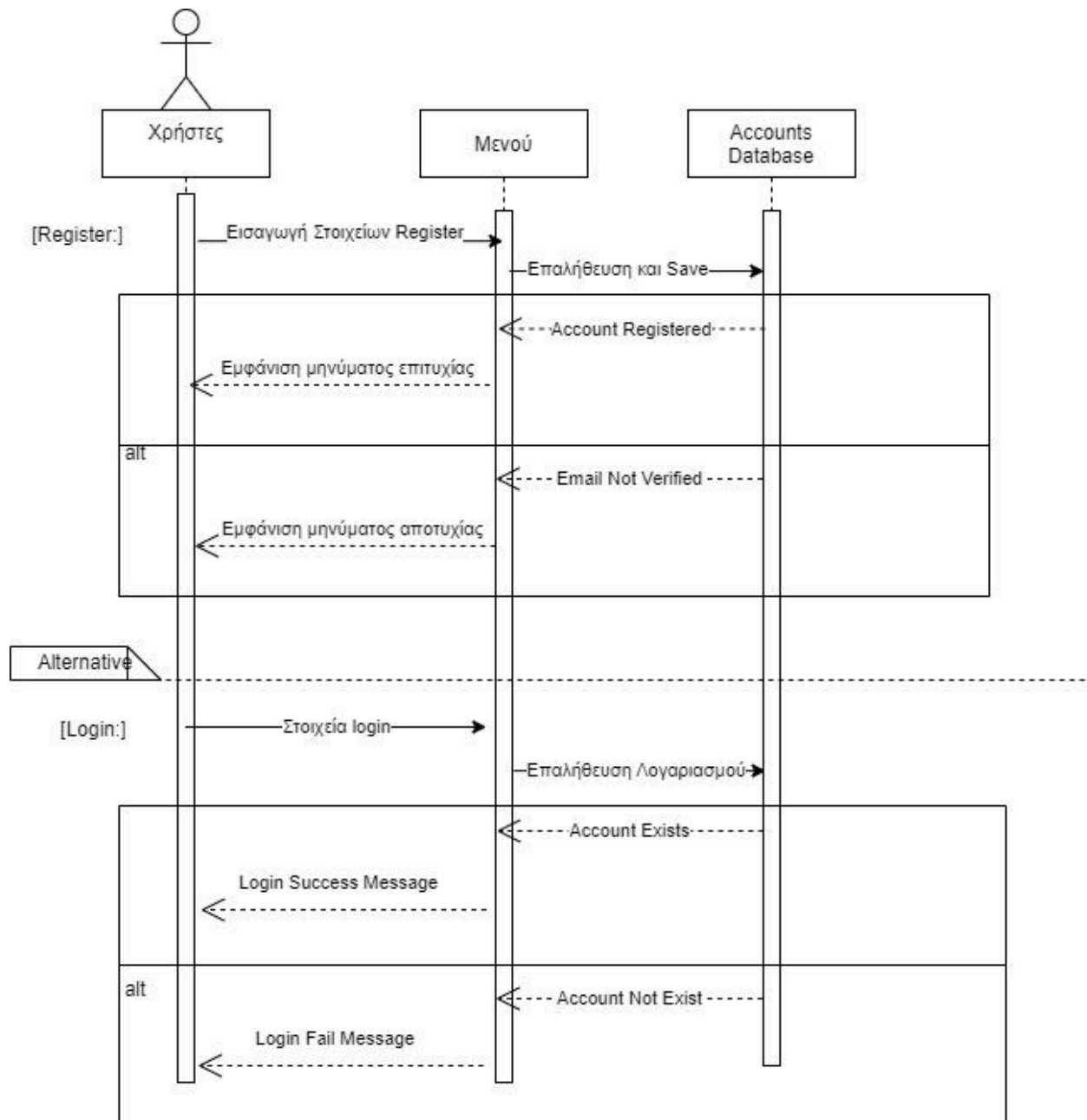
Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

Sequence Diagram για τη Λειτουργία “Συμπλήρωση Στοιχείων Ενοικιαστή”:



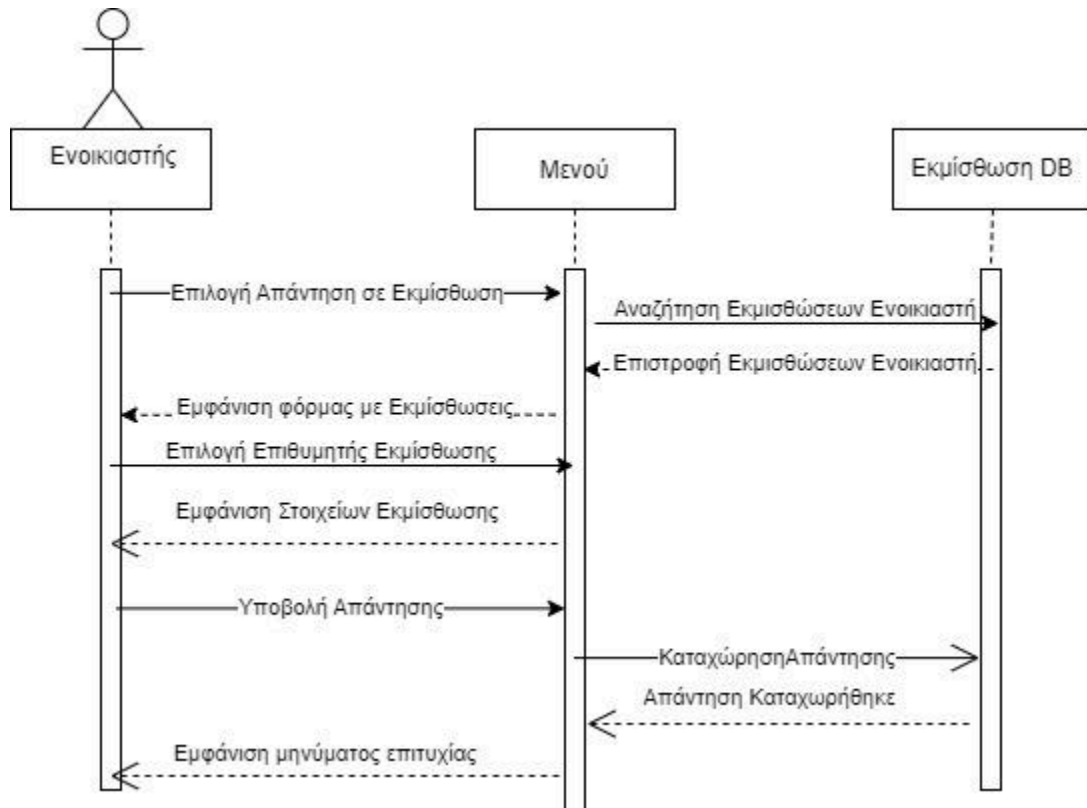
Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

Sequence Diagram για τη Λειτουργία “Login” (και “Sign Up”):



Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

Sequence Diagram για τη Λειτουργία “Υποβολή Απάντησης”:

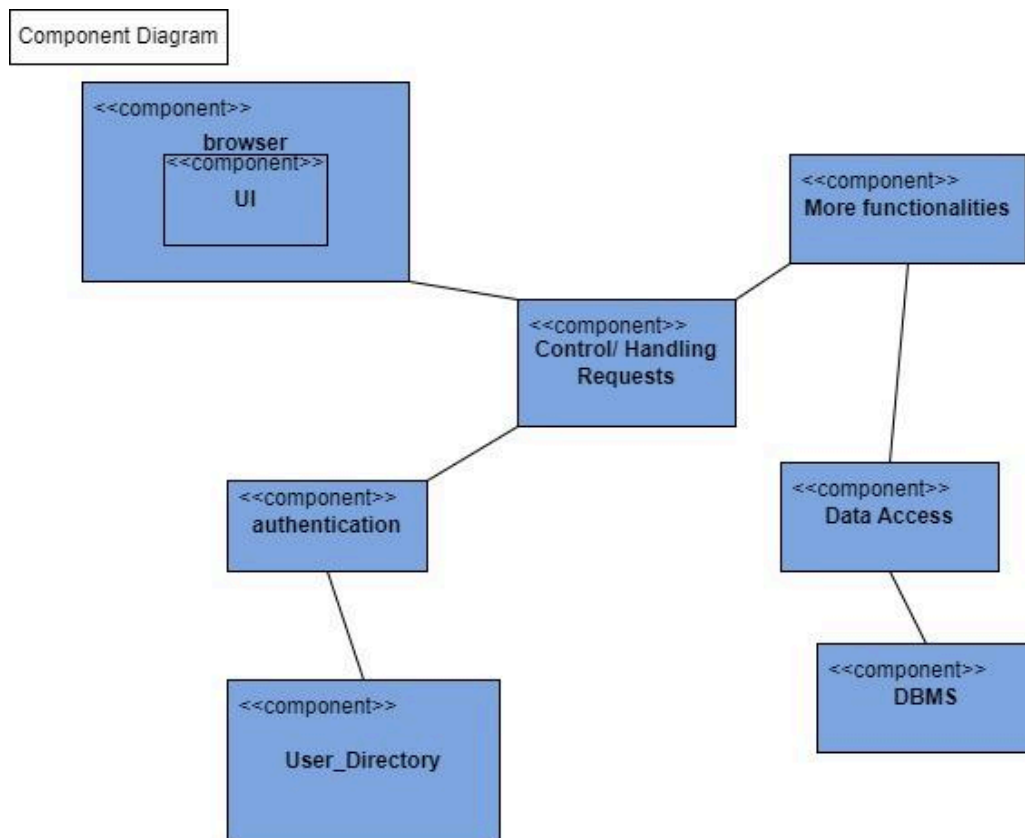


Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).



## 2ο Παραδοτέο - Υλοποίηση:

Ακολουθεί το Component/Deployment διάγραμμα που χρησιμοποιήσαμε για την υλοποίηση της εφαρμογής:



Για καλύτερη προβολή του σχήματος, πατήστε [εδώ](#).

### Repository Link:

To link στο github repository για το backend (token authentication) είναι το ακόλουθο:

[https://github.com/manouslinard/dist\\_sys\\_2022/tree/backend-token-auth](https://github.com/manouslinard/dist_sys_2022/tree/backend-token-auth)

To link για το repository (γενικά):

[https://github.com/manouslinard/dist\\_sys\\_2022](https://github.com/manouslinard/dist_sys_2022)

Επίσης έχουμε υλοποιήσει το backend και με basic authentication (μαζί με τις επιπλέον λειτουργίες που χρειαζόμαστε στο [frontend](#)). Το link του backend repository (με basic authentication) είναι το εξής:

[https://github.com/manouslinard/dist\\_sys\\_2022/tree/backend-basic](https://github.com/manouslinard/dist_sys_2022/tree/backend-basic)

Αντίστοιχα, το εγχειρίδιο χρήσης για τις καινούργιες λειτουργίες μπορείτε να το βρείτε [εδώ](#).

## Εγχειρίδιο Χρήσης REST API:

Για να χρησιμοποιηθούν τα παρακάτω “links”, χρειάζεται το request να ξεκινάει με “http://localhost:8080” (πχ η μέθοδος “/lessor/getAllLessors” θα γραφτεί ως εξής για να δουλέψει: “http://localhost:8080/lessor/getAllLessors” – αντίστοιχα και για τις άλλες μεθόδους).

Είναι αξιοσημείωτο ότι στις μεθόδους χρησιμοποιούμε curly brackets “{ ... }” για να επισημάνουμε ότι αυτό το “κομμάτι” είναι μεταβλητή και θα χρειαστεί να αλλάξει αναλόγως, για να υπάρξει το επιθυμητό αποτέλεσμα (δείτε και τις περιγραφές των μεθόδων). Επίσης, στο τελικό request **δεν** τοποθετούνται curly brackets αλλά μόνο το value της μεταβλητής.

Η **πρόσβαση** των “μεθόδων” που ξεκινάνε με /lessor/... μπορούν να εκτελεστούν από τον lessor. Αντίστοιχα η πρόσβαση των “μεθόδων” που ξεκινάνε με /tenant/... μπορούν να εκτελεστούν από τον tenant. Επιπρόσθετα οτιδήποτε ξεκινάει με /api/auth/... μπορεί να εκτελεστεί από όλους στο token και basic authentication (backend). Είναι επίσης σημαντικό να σημειωθεί ότι ο κάθε χρήστης μπορεί να δει μόνο τα δικά του δεδομένα και **όχι** άλλων χρηστών (εκτός αν έχει γίνει σύνδεση ως admin, ο οποίος μπορεί επιπλέον να εκτελέσει functions που δημιουργούν lease, σβήνουν χρήστες ή να βλέπει τα στοιχεία τους – κυρίως για να βοηθάει στην επίλυση προβλημάτων του χρήστη αν υπάρχουν).

<u>Authentication</u> ( <u>λειτουργεί μόνο</u> <u>για το token</u> <u>authentication</u> )	<u>Περιγραφή</u>	<u>Μέθοδος</u>	<u>Test Input (JSON)</u>
/api/auth/signup	Οι χρήστες κάνουν signup στο σύστημα.	POST	Lessor Signup:  <pre>{"username": "lessor1", "email": "lessor1@gmail.com"}</pre>

			<pre>, "password": "pass123", "role": ["lessor"]}</pre> <p>Tenant Signup:</p> <pre>{"username": "tenant1", "email": "tenant1@gmail.com", "password": "pass123"}</pre> <p>Admin Signup:</p> <pre>{"username": "admin1", "email": "admin1@gmail.com", "password": "pass123", "role": ["admin"]}</pre> <p>(optional: firstName, lastName, afm and phone attributes – see structure of entire json <a href="#">here</a>)</p>
/api/auth/signin	Οι χρήστες κάνουν signin στο σύστημα.	POST	<pre>{"username": "tenant", "password": "pass123"}</pre>

<u>Lessor (/lessor)</u>	<u>Περιγραφή</u>	<u>Μέθοδος</u>	<u>Test Input(JSON)</u>
/lessor/getAllLessors	Επιστρέφει όλους τους lessors.	GET	
/lessor/{lessorUsername}/leases	Φέρνει όλα τα leases του lessor με το δοσμένο lessorUsername.	GET	
/lessor/{lessorUsername}/leases/{lid}	Φέρνει ένα lease με id=lid του lessor με το δοσμένο lessorUsername	GET	

/lessor/{lessorUsername}/leases/{lid}	Σβήγει το lease με id=lid του lessor με το δοσμένο lessorUsername	DELETE	
/lessor/{lessorUsername}/leases/{lid}	Κάνει update το lease με id=lid του lessor με το δοσμένο lessorUsername.	PUT	<pre>{   "title": "lease2",   "startDate": "2023-01-01",   "endDate": "2023-01-13" }</pre> <p>(only mandatory attr are lease's title, start and end Date - the other are optional - see other attr in test JSON <a href="#">createLease</a>)</p>
/lessor/{lessorUsername}/{tenantUsername}/createLease	Δημιουργεί ένα lease και εισάγει τα στοιχεία του tenant (με το δοσμένο tenantUsername) όπως και στοιχεία του lessor (με το δοσμένο lessorUsername) μέσα στο νέο lease.	POST	<pre>{   "title": "test",   "address": "address",   "tk": "12137",   "dimos": "Peristeri",   "reason": "Reason",   "cost": 15000,   "startDate": "2023-01-01",   "endDate": "2023-01-13",   "sp_con": "Special Conditions",   "dei": "1234567" }</pre>
/lessor/createTenant	Δημιουργεί έναν tenant.	POST	<pre>{   "username": "christos3",   "email": "christos3@gmail.com",   "password": "pass123",   "firstName": "Chris",   "lastName": "Papas",   "afm": "12345678901",   "phone": "698731311" }</pre> <p>(attributes firstName,</p>

			lastName, afm and phone are optional – afm should be numbers only and exact size of 11! Also afm is unique and cannot be “used” again)
/lessor/getAllTenants	Εμφανίζει όλους τους tenants.	GET	
/lessor/{lessorUsername}/contracts	Φέρνει όλα τα contracts του lessor με το δοσμένο lessorUsername.	GET	
/lessor/{lessorUsername}/contracts/{cid}	Φέρνει ένα contract με id=cid του lessor με το δοσμένο lessorUsername.	GET	
/lessor/{lessorUsername}	Επιστρέφει τα στοιχεία του lessor με το input lessorUsername.	GET	
/lessor/{lessorUsername}	Διαγράφει τον lessor με το input lessorUsername.	DELETE	

<u>Tenant</u>	<u>Περιγραφή</u>	<u>Μέθοδος</u>	<u>Test Input (JSON)</u>
/tenant/{tenantUsername}/leases	Φέρνει όλα τα leases του tenant με το δοσμένο tenantUsername.	GET	

/tenant/{tenantUsername}/leases/{lid}	Φέρνει ένα lease με id=lid του tenant με το δοσμένο tenantUsername.	GET	
/tenant/{tenantUsername}/leases/{lid}/answer	Αποθηκεύει την απάντηση του tenant με το δοσμένο tenantUsername σε ένα lease του, το οποίο έχει id=lid.  Αν tenantAnswer: hasAgreed == true, οριστικοποιείται το contract.	POST	<pre>{ "hasAgreed": false,   "tenantComment": "Needs Fixing" }</pre> <p>OR:</p> <pre>{ "hasAgreed": true }</pre> <p>(comment is optional)</p>
/tenant/getAllLessors	Επιστρέφει όλους τους lessors που είναι καταγεγραμμένοι στο σύστημα.	GET	
/tenant/{tenantUsername}/contracts	Φέρνει όλα τα contracts του tenant με το δοσμένο tenantUsername	GET	
/tenant/{tenantUsername}/contracts/{cid}	Φέρνει ένα contract με id=cid του tenant με το δοσμένο tenantUsername.	GET	
/tenant/{tenantUsername}	Διαγράφει τον tenant με το input tenantUsername.	DELETE	

/tenant/{tenantUsername}	Επιστρέφει τα στοιχεία του tenant με το input tenantUsername.	GET	
--------------------------	---	-----	--

### 3ο Παραδοτέο: Σχεδιασμός

#### Ασφάλεια:

Στο backend, ορίζουμε την ασφάλεια με το Spring Boot Security στο αρχείο [SecurityConfig](#). Επιπλέον, χρησιμοποιούμε το βασικό http request για το authentication από το frontend στο backend, για αυτό και ο χρήστης χρειάζεται να κάνει login (ή register) για να μπει στην εφαρμογή. Σχετικά με το basic auth στο frontend, αν ο χρήστης εισάγει λανθασμένα στοιχεία, τότε εμφανίζεται μήνυμα "Bad Credentials". Επιπλέον, ο κωδικός πρόσβασης κάθε χρήστη κωδικοποιείται μετά την εγγραφή.

Επιπρόσθετα ο ρόλος του χρήστη ελέγχεται παραμετρικά για να διασφαλιστεί η σωστή λειτουργία και η "ακεραιότητα" της εφαρμογής. Για να το εφαρμόσουμε αυτό στο frontend, χρησιμοποιούμε το Thymeleaf, το οποίο προσφέρει άμεσους τρόπους για να ελέγξουμε τον ρόλο του χρήστη, αν είναι αυθεντικοποιημένος και άλλες χρήσιμες λειτουργίες.

#### Επικοινωνία Front - Back:

Η επικοινωνία front - back γίνεται με REST που έχουμε υλοποιήσει στο backend. Για να καλέσουμε "εύκολα" κάποιες από τις μεθόδους που έχουμε υλοποιήσει, χρησιμοποιούμε τη jquery (πχ για το delete ενός χρηστή πατώντας ένα κουμπί). Είναι αξιοσημείωτο ότι αξιοποιούμε τις λειτουργίες του thymeleaf για να ελέγξουμε αν τα δεδομένα υπάρχουν ήδη στη βάση, ενημερώνοντας τον χρήστη καταλλήλως (πχ αν ο χρήστης κατά την εγγραφή βάζει ένα username που ήδη υπάρχει στη ΒΔ → επιστροφή κατάλληλου μηνύματος στο αντίστοιχο πεδίο της φόρμας συμπλήρωσης στο frontend). Ένας ακόμη λόγος που χρησιμοποιήσαμε την thymeleaf στο frontend είναι για να ελέγχουμε άμεσα αν ο viewer της σελίδας έχει ή δεν έχει αυθεντικοποιηθεί και, ανάλογα με το ρόλο του, να του εμφανίζεται το κατάλληλο περιεχόμενο. Για να εφαρμόσουμε τις διάφορες συναρτήσεις – ελέγχους (για το input) και άλλες λειτουργίες (όπως slider φωτεινότητας) στο frontend χρησιμοποιήσαμε javascript.

#### Πρόσβαση:

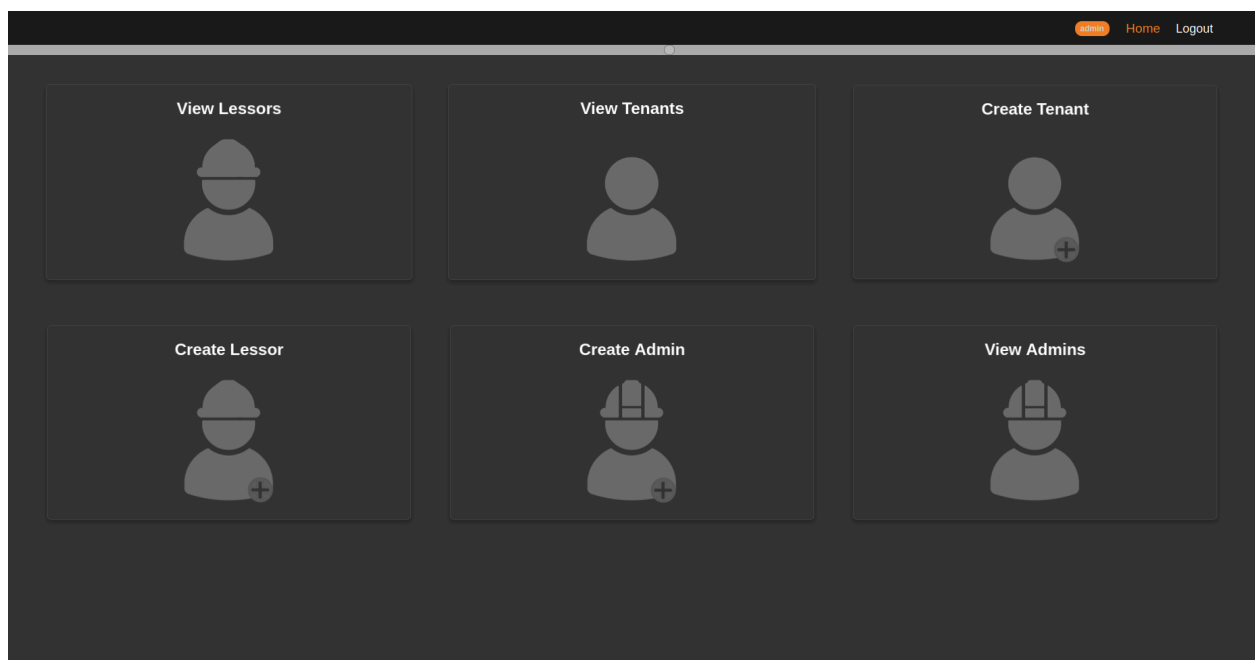
Η πρόσβαση κάθε χρήστη περιγράφεται στις αντίστοιχες μεθόδους στη στήλη "Πρόσβαση" (δείτε στο [εγχειρίδιο χρήσης](#)). Για την πρόσβαση του χρήστη στο backend, έχουμε ορίσει με το Spring Security (και συγκεκριμένα στο αρχείο [SecurityConfig](#)) ποια REST calls επιτρέπεται να χρησιμοποιήσει ο κάθε ρόλος. Επίσης, στο frontend χρησιμοποιούμε το Thymeleaf sec:authorize για να εμφανίζουμε στους χρήστες μόνο τις μεθόδους – λειτουργίες που έχουν

πρόσβαση. Είναι αξιοσημείωτο ότι το security που έχουμε ορίσει στο backend “συμφωνεί” με τις λειτουργίες που δείχνουμε στον κάθε ρόλο (στο frontend).

### Screenshots of Frontend:

Στο frontend έχουμε ένα brightness slider, στο οποίο ο χρήστης μπορεί να ρυθμίσει τη φωτεινότητα (dark & light mode). Ενδεικτικά για το light mode δείτε τις οθόνες του lessor.

### Admin Frontend (Dark Mode):





adminHomeLogout

Admin Username \*

Username

Email \*

Email

Password \*

Password

First Name

First Name

Last Name

LastName

Phone

Phone

AFM

AFM

Submit

admin			admin@gmail.com	Current User
-------	--	--	-----------------	--------------

adminHomeLogout

Username	First Name	Last Name	Email	Actions
lessor			lessor@gmail.com	<div>Delete</div>


adminHomeLogout

Username	First Name	Last Name	Email	Actions
tenant			tenant@gmail.com	<div>Delete</div>


Lessor Frontend (Dark Mode):

[lessor](#) [Home](#) [Logout](#)


View Lessors




View Tenants




Create Tenant




View Leases




Create Lease



Update Lease



View Contracts



[lessor](#) [Home](#) [Logout](#)

Username	First Name	Last Name	Email
lessor			lessor@gmail.com

lessor

Home

Logout

Username	First Name	Last Name	Email
tenant			tenant@gmail.com

lessor

Home

Logout

Title	Comment	Address	Start Date	End Date	Dei	TK	Cost	Actions
test		address	2023-01-01	2023-01-13	1234567	12137	15000.0	<div>Delete</div>

lessorHomeLogout

Title \*

Title

Tenant Username \*

Tenant Username

Address

Address

Municipality

Municipality

Postal Code

Code

Cost

Cost

Reason

Reason

Start Date

mm / dd / yyyy

End Date

mm / dd / yyyy

Special Condition

Special Condition

Dei Number

Number

Submit

lessorHomeLogout

Requested Lease Title \*

Title

New Title

New Title (Optional)

New Tenant Username

New Tenant Username (Optional)

Address

Address

Municipality

Municipality

Postal Code

Code

Cost

Cost

Reason

Reason

Start Date

mm / dd / yyyy

End Date

mm / dd / yyyy

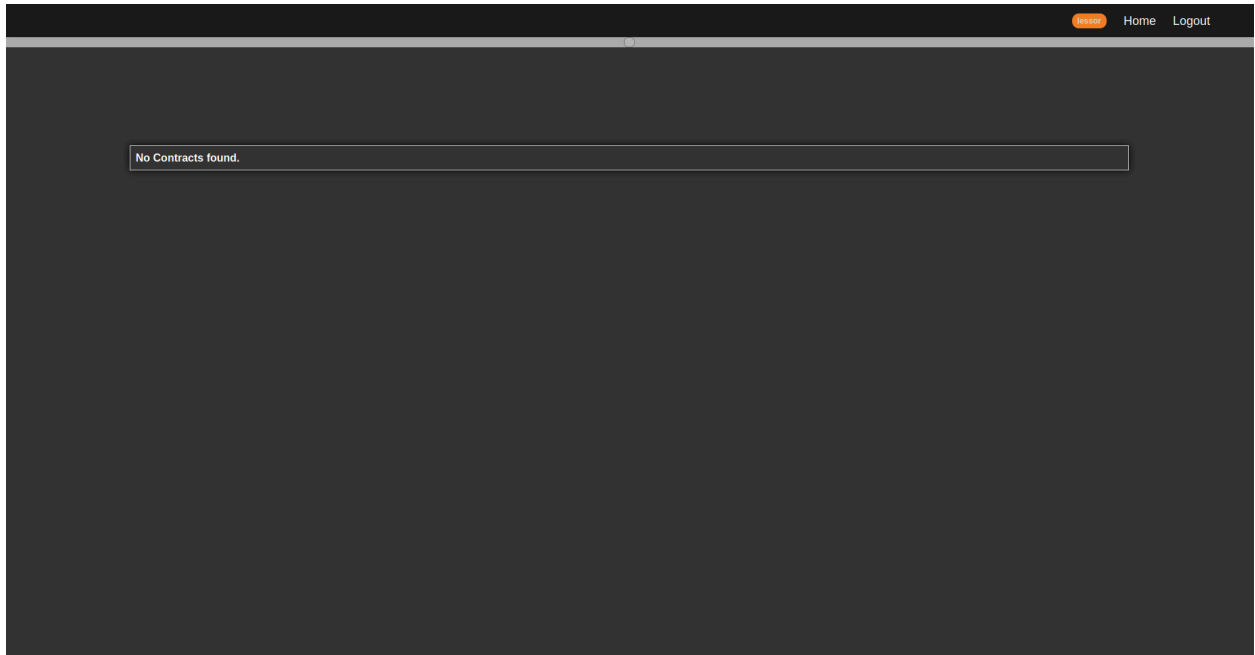
Special Condition

Special Condition

Dei Number

Number

Submit



Lessor Frontend (Light Mode):

View Lessors



View Tenants



Create Tenant



View Leases



Create Lease



Update Lease



View Contracts



Username	First Name	Last Name	Email
lessor			lessor@gmail.com

Username	First Name	Last Name	Email
tenant			tenant@gmail.com

Title	Comment	Address	Start Date	End Date	Dei	TK	Cost	Actions
test		address	2023-01-01	2023-01-13	1234567	12137	15000.0	Delete



Title \*

Title

Tenant Username \*

Tenant Username

Address

Municipality

Postal Code

Address

Municipality

Code

Cost

Reason

Cost

Reason

Start Date

End Date

mm / dd / yyyy

mm / dd / yyyy

Special Condition

Dei Number

Special Condition

Number

Submit

Requested Lease Title \*

Title

New Title

New Title (Optional)

New Tenant Username

New Tenant Username (Optional)

Address

Municipality

Postal Code

Address

Municipality

Code

Cost

Reason

Cost

Reason

Start Date

End Date

mm / dd / yyyy

mm / dd / yyyy

Special Condition

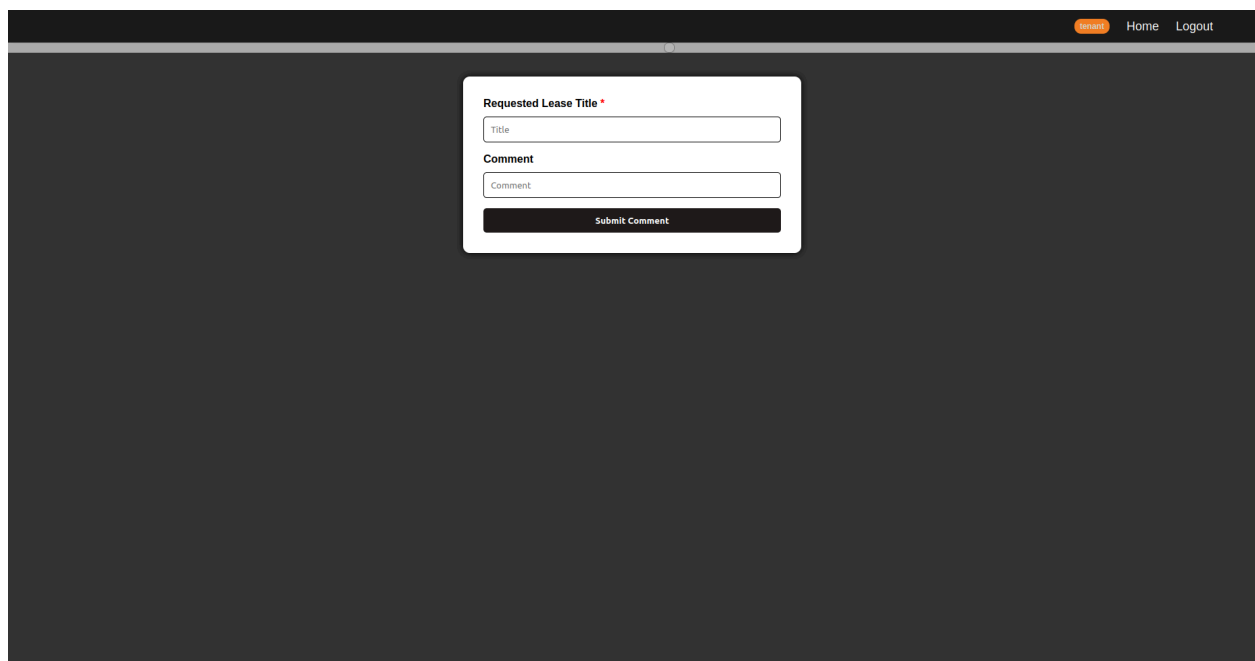
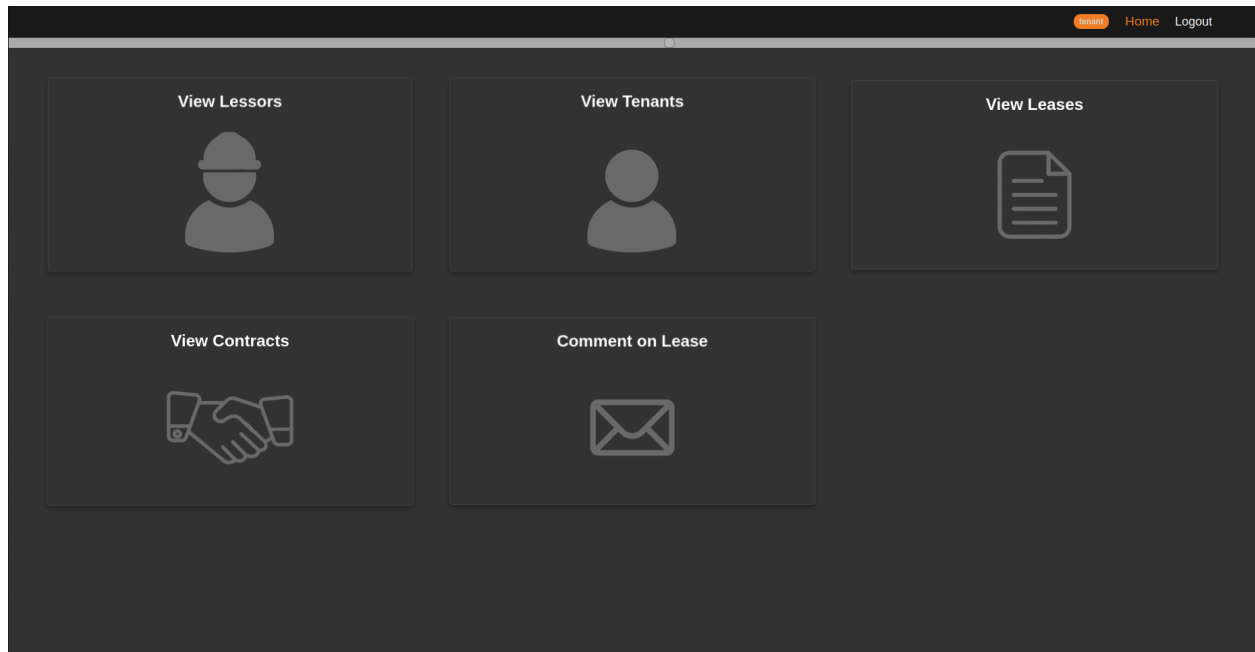
Dei Number

Special Condition

Number

Submit

## Tenant Frontend (Dark Mode):



tenantHomeLogout

Title	Comment	Address	Start Date	End Date	Dei	TK	Cost	Actions
test		address	2023-01-01	2023-01-13	1234567	12137	15000.0	Agree

tenantHomeLogout

Username	First Name	Last Name	Email
lessor			lessor@gmail.com

tenant Home Logout			
Username	First Name	Last Name	Email
tenant			tenant@gmail.com

## 4ο Παραδοτέο: Υλοποίηση

### Repository Links:

- [Main](#)
- [Backend with Token authentication](#)
- [Backend with Basic authentication](#)
- [Frontend](#)

### Εγχειρίδιο Χρήσης:

Το frontend έχει την ίδια βάση με το backend-basic-auth & backend-token-auth, που σημαίνει ότι όσα στοιχεία δημιουργούνται/διαγράφονται στο frontend (ή και τα αντίστοιχα στο backend-basic-auth ή στο backend-token-auth), θα υπάρχει ανάλογη ενημέρωση και στα υπόλοιπα.

Είναι σημαντικό να σημειωθεί ότι στις μεθόδους χρησιμοποιούμε curly brackets “{ ... }” για να επισημάνουμε ότι αυτό το “κομμάτι” είναι μεταβλητή και θα χρειαστεί να αλλάξει αναλόγως, για να υπάρξει το επιθυμητό αποτέλεσμα (δείτε και τις περιγραφές των μεθόδων). Επίσης, στο τελικό request **δεν** τοποθετούνται curly brackets αλλά μόνο το value της μεταβλητής.

Δείτε το προηγούμενο παραδοτέο (και το εγχειρίδιο χρήσης με τις υπόλοιπες μεθόδους του backend) [εδώ](#).

<u>User (Backend)</u>	<u>Περιγραφή</u>	<u>Μέθοδος</u>	<u>Πρόσβαση</u>	<u>Test Input (JSON)</u>
/user/tenant/{id}	Σβήνει τον tenant με το input id.	DELETE	ADMIN	-
/user/lessor/{id}	Σβήνει τον lessor με το input id.	DELETE	ADMIN	-
/user/leases/{id}	Σβήνει το lease με το input id.	DELETE	LESSOR	-
/user/admin/{id}	Σβήνει τον admin με το input id.	DELETE	ADMIN	-
/user/leases/agree/{id}	Συμφωνεί με το lease με το input id (δημιουργεί το αντίστοιχο contract).	POST	TENANT	-

<u>UserForm (Frontend)</u>	<u>Περιγραφή</u>	<u>Μέθοδος</u>	<u>Πρόσβαση</u>	<u>Test Input (JSON)</u>
/	Επιστρέφει το index	GET	ALL	-
/lessorform	Εμφανίζει την φόρμα του lessor	GET	ADMIN	-

/lessorlist	Εμφανίζει την λίστα των lessors	GET	LESSOR, TENANT	-
/leaselist	Εμφανίζει την λίστα των leases	GET	LESSOR, TENANT	-
/contractlist	Εμφανίζει την λίστα των contract	GET	LESSOR, TENANT	-
/lessorform	Αποθηκεύει τον lessor.	POST	ADMIN	-
/tenantform	Εμφανίζει την φόρμα του tenant	GET	ADMIN, LESSOR	-
/tenantlist	Εμφανίζει την λίστα των tenants.	GET	Όλοι οι logged in users.	-
/tenantform	Αποθηκεύει τον tenant.	POST	ADMIN, LESSOR	-
/leaseform	Εμφανίζει τη φόρμα δημιουργίας του lease.	GET	LESSOR	-
/leaseform	Αποθηκεύει το lease.	POST	LESSOR	-
/leaseupdate	Εμφανίζει την φόρμα για το update ενός lease.	GET	LESSOR	-
/leaseupdate	Ανανεώνει τα στοιχεία του lease.	POST	LESSOR	-
/adminlist	Εμφανίζει την λίστα των admin.	GET	ADMIN	-
/adminform	Αποθηκεύει τον Admin.	POST	ADMIN	-

/adminform	Εμφανίζει την φόρμα των admin.	GET	ADMIN	-
/leasecom	Εμφανίζει τη φόρμα για τη συμπλήρωση του lease.	GET	TENANT	-
/leasecom	Βρίσκει το lease που το comment πρόκειται να αναφέρεται με βάση τον τίτλο, και αποθηκεύει το comment.	POST	TENANT	-
/verifyuser	Κάνει Verify ένα verification code.	POST	ANY	-
/verifyuser	Εμφανίζει τη φόρμα για το verification code.	GET	ANY	-