ΤΕΧΝΟΛΟΓΙΕΣ ΛΟΓΙΣΜΙΚΟΥ

Ανάπτυξη Λογισμικού με Χρήση του Rational Unified Process: Ανάλυση, Σχεδίαση και Ανάπτυξη με Βάση την UML

ΣΤΥΛΙΑΝΟΣ - ΚΩΝΣΤΑΝΤΙΝΟΣ ΒΑΡΥΜΠΟΜΠΙΩΤΗΣ, **Π20028**

ΣΤΑΥΡΟΣ ΚΟΛΟΥΑΣ, **Π18077**

ΓΙΩΡΓΟΣ ΚΩΝΣΤΑΝΤΙΝΟΣ ΜΠΟΥΤΣΙΚΟΣ, **Π20141**

Κεφάλαιο 1ο: Εισαγωγή

1. Στόχοι της Εργασίας

Κύριος σκοπός της εργασίας είναι η δημιουργία λογισμικού τύπου ημερολογίου ραντεβού ανάμεσα σε φοιτητές και καθηγητές.
Οι χρήστες θα είναι οι διαχειριστές της εφαρμογής/ιστοσελίδας. Οι φοιτητές που θα εγγράφονται στην πλατφόρμα για την διεξαγωγή ραντεβού με καθηγητές κι οι τελευταίοι που θα δηλώνουν διαθεσιμότητα για ραντεβού.

Θα έχει τις παρακάτω ιδιότητες:

1. Ο χρήστης θα μπορεί να δηλώσει τα στοιχεία του (Username, Password, Email)

α. Φοιτητές:

i) Sign Up στο σύστημα για να μπορεί να συνδεθεί.
ii) Για να κλείσει ραντεβού πρέπει να είναι συνδεδεμένος στο σύστημα.
iii) Θα πρέπει να αποδεχτεί ο καθηγητής το ραντεβού.
iv) Θα επιβεβαιώνεται η εγγραφή του από κάποιον διαχειριστή.
v) Έξτρα στοιχεία (όνομα, επώνυμο, ΑΜ, σχολή κλπ.)
vi) ακύρωση ενός επερχόμενου ραντεβού.
vii) προβολή ραντεβού ανά Α) βδομάδα, Β) μήνα και Γ) θέμα ραντεβού

β. Διαχειριστής:

i) Θα επιβεβαιώνει τυχόν διπλοκλεισμένα και διπλωμένα ραντεβού.
ii) Επιβεβαίωση εγγραφών των καθηγητών και φοιτητών.
iii) Συγκεκριμένοι κωδικοί διαχειριστή για
σύνδεση χωρίς εγγραφή

γ. Καθηγητές:

i) Sign up στο σύστημα για να μπορεί να συνδεθεί.
ii) Για να αποδεχτεί ραντεβού πρέπει να είναι συνδεδεμένος στο σύστημα.
iii) Αποδοχή ραντεβού των φοιτητών
iv) Θα επιβεβαιώνεται η εγγραφή του από κάποιον διαχειριστή.
v) Έξτρα στοιχεία (όνομα, επώνυμο, ΑΜ, σχολή κλπ.)
vi) αναβολή ενός επερχόμενου ραντεβού
vii) ή ακύρωση του.
viii) προβολή ραντεβού ανά Α) βδομάδα, Β) μήνα και Γ) θέμα ραντεβού.

2. Ταξινόμηση σημαντικότητας των ραντεβού για τυχόν αλλαγές και αναβολές ραντεβού άλλων φοιτητών με κάποιο καθηγητή. (θέμα προτεραιότητας)

2. Ορισμός του προβλήματος προς επίλυση

Το πρόβλημα που πρέπει να λύσουμε είναι η ανάπτυξη μιας εφαρμογής καταχώρησης ραντεβού μεταξύ καθηγητών και φοιτητών. Ανάλογα με τη φύση του ραντεβού, ο φοιτητής θα μπορεί να επιλέξει τον καθηγητή που επιθυμεί να συναντήσει. Η εφαρμογή πρέπει επίσης, να παρέχει πληροφορίες σχετικά με τη διαθεσιμότητα του καθηγητή και τις διαθέσιμες ημερομηνίες και ώρες για ραντεβού. Ο φοιτητής θα μπορεί να καταχωρεί την ημερομηνία και την ώρα του ραντεβού με τον εκάστοτε καθηγητή.

Επιπλέον, η εφαρμογή πρέπει να διαχειρίζεται την προτεραιότητα των ραντεβού ανάλογα με την επείγουσα φύση τους, όπως για παράδειγμα την ανάθεση πτυχιακής εργασίας ή την προετοιμασία για ένα πρόγραμμα ανταλλαγής. Ακόμη, πρέπει να παρέχει τη δυνατότητα ακύρωσης ραντεβού τόσο από τον φοιτητή όσο και από τον καθηγητή, καθώς και την λειτουργία αναβολής ραντεβού, αποκλειστικά από τον καθηγητή.

Τέλος, η πρέπει να υπάρχει δυνατότητα προβολής των ραντεβού του φοιτητή ανά εβδομάδα, ανά μήνα και ανά θέμα για την ευκολότερη παρακολούθηση και οργάνωση των δραστηριοτήτων του.

Κεφάλαιο 2ο: Σύντομη παρουσίαση της RUP (Rational Unified Process)

Η **RUP**(Rational Unified Process) είναι μια διαδικασία-μεθοδολογία ανάπτυξης λογισμικού που δημιουργήθηκε σε ένα τμήμα της IBM το 2003. Η διαδικασία της ανάπτυξης πρέπει, σύμφωνα με την RUP, να έχει μια σειρά επαναλήψεων μέχρι να εξελιχθεί το τελικό λογισμικό. Η RUP αποτελείται από οδηγίες σχετικά με τις τεχνικές ανάπτυξης λογισμικού, κυρίως για την ανάλυση απαιτήσεων και τον σχεδιασμό. Η δομή ενός έργου σε σχέση με τον χρόνο έχει 4 φάσεις:

Α) Έναρξη (Inception): Αφορά την προοπτική του έργου.

Β) Εκπόνηση μελέτης (Elaboration): Σχεδιασμός των απαιτούμενων δραστηριοτήτων και πόρων. Καθορισμός των χαρακτηριστικών και σχεδιασμός της δομής.

Γ) Κατασκευή (Construction): Ανάπτυξη του προϊόντος σε μία σειρά επαναλήψεων.

Δ)Μετάβαση (Transition): Προώθηση του προϊόντος στους χρήστες για εκπαίδευση ,αποσφαλμάτωση και εκτίμηση της ποιότητας του τελικού προϊόντος σύμφωνα με τους στόχους που τέθηκαν στην έναρξη.

Κεφάλαιο 3ο : Έναρξη

1. Σύλληψη απαιτήσεων

- Επιλογή Καθηγητή: Ο φοιτητής πρέπει να μπορεί να επιλέγει τον καθηγητή ή την καθηγήτρια με την οποία επιθυμεί να έχει ραντεβού.

- Διαθεσιμότητα Καθηγητή: Η εφαρμογή πρέπει να επιτρέπει στον φοιτητή να ενημερώνεται για τη διαθεσιμότητα του καθηγητή και να βλέπει ποιες μέρες και ώρες είναι διαθέσιμες για ραντεβού.

- Καταχώρηση Ραντεβού: Ο φοιτητής πρέπει να μπορεί να καταχωρεί και να δεσμεύει μια συγκεκριμένη ημερομηνία και ώρα για ραντεβού με τον καθηγητή, προσδιορίζοντας τον λόγο του ραντεβού.

- Προτεραιότητα Ραντεβού: Το σύστημα πρέπει να αποδίδει προτεραιότητα στα ραντεβού ανάλογα με την επείγουσα φύση τους, όπως η ανάθεση πτυχιακής εργασίας ή προθεσμία για επιστολές συστάσεων.

- Διαχείριση Ραντεβού: Τόσο ο φοιτητής όσο και ο καθηγητής πρέπει να έχουν τη δυνατότητα ακύρωσης, αναβολής ενός ραντεβού.

- Προβολή Ραντεβού: Η εφαρμογή πρέπει να παρέχει στον φοιτητή και στον καθηγητή τη δυνατότητα προβολής των ραντεβού του ανά εβδομάδα, ανά μήνα και ανά θέμα.

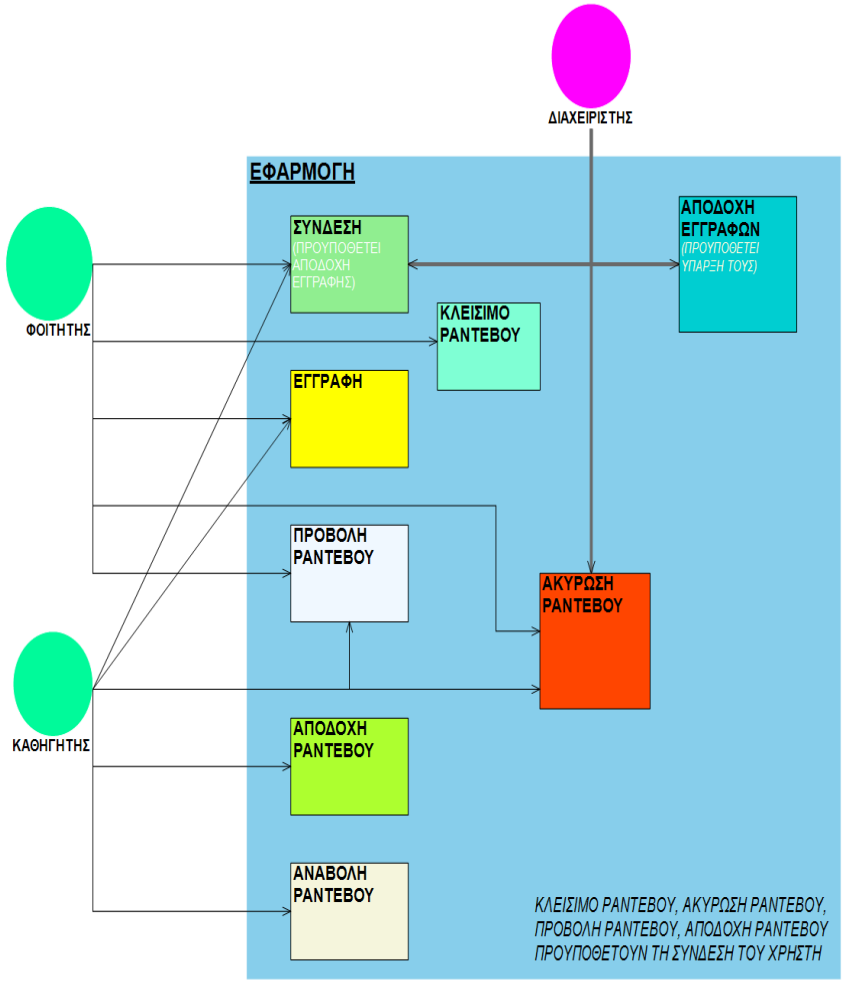
- Δημιουργία Χρηστών: Οι διαχειριστές πρέπει να έχουν τη δυνατότητα να επιβεβαιώνουν τους φοιτητές και καθηγητές στην εγγραφή τους στην εφαρμογή.

- Επιβεβαίωση Ραντεβού: Οι διαχειριστές και η εφαρμογή θα κρίνουν αν είναι έγκυρα τα ραντεβού.

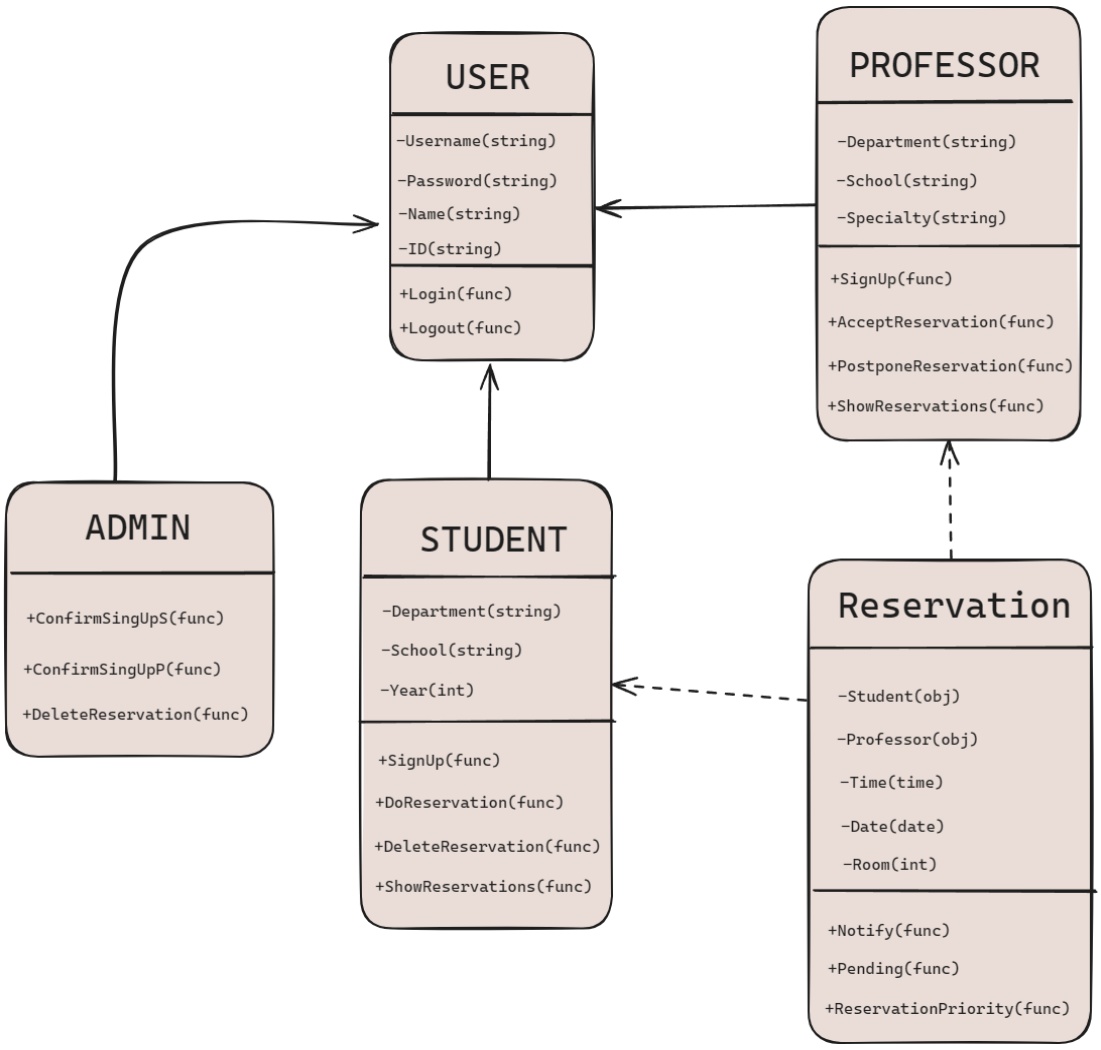
- Αυτές είναι οι βασικές απαιτήσεις που θα πρέπει να ληφθούν υπόψη κατά την ανάπτυξη της εφαρμογής.

2. Ανάλυση-Σχεδιασμός

1. Διαγράμματα Περιπτώσεων Χρήσης (1η έκδοση)



2. Διαγράμματα Κλάσεων/Τάξεων (1η έκδοση)



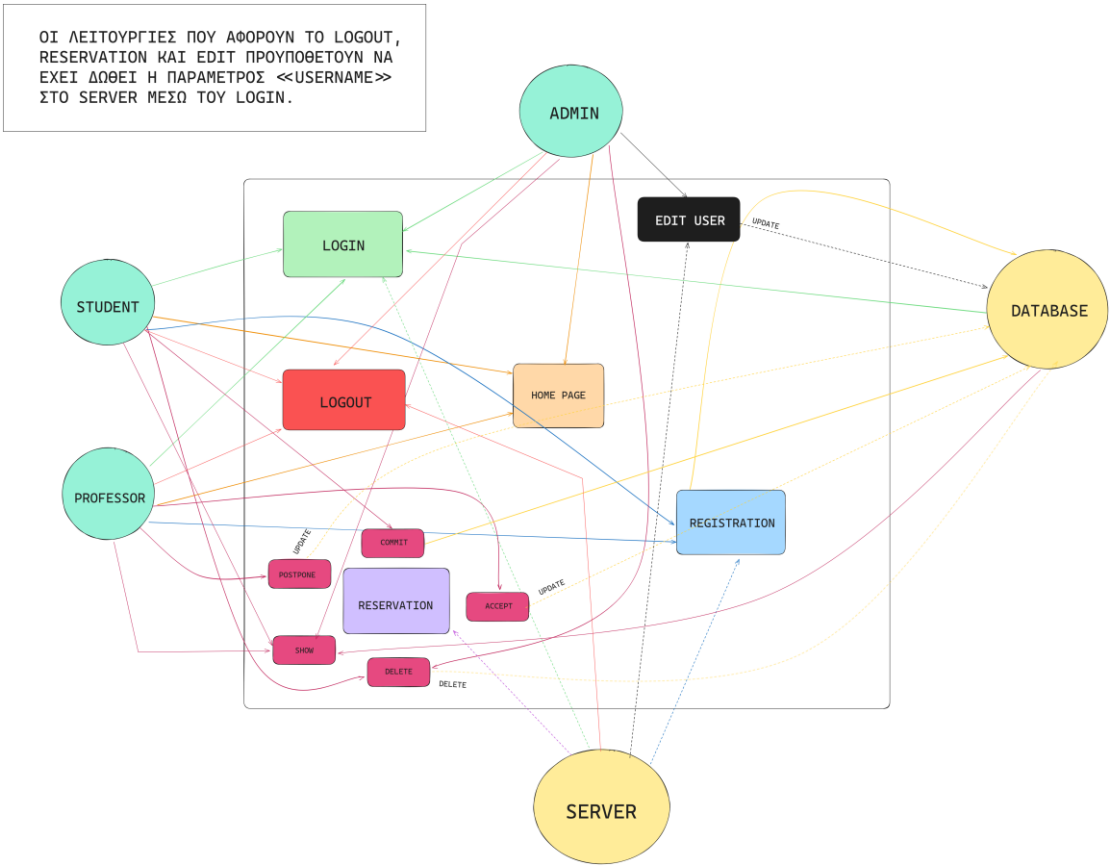
Κεφάλαιο 4ο: Εκπόνηση μελέτης (Elaboration)

1: Εξήγηση Αλλαγών

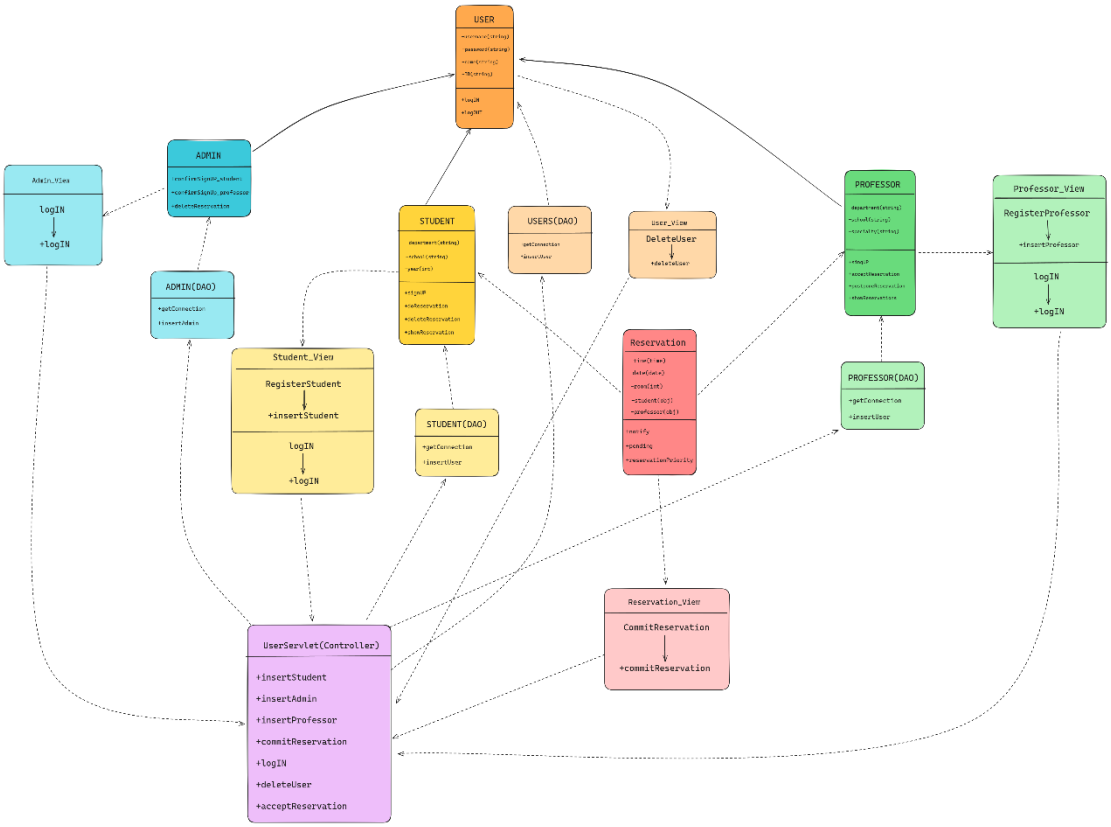
Αλλαγή στον τρόπο με τον οποίο διαχειριζόμαστε την εφαρμογή. Θα υλοποιηθεί ως εφαρμογή ιστού, οπότε προστέθηκε βάση δεδομένων (MySQL) και server (Apache Tomcat 9.0). Χρησιμοποιείται πλέον το design pattern CRUD in MVC.

2: Ανάλυση – Σχεδιασμός

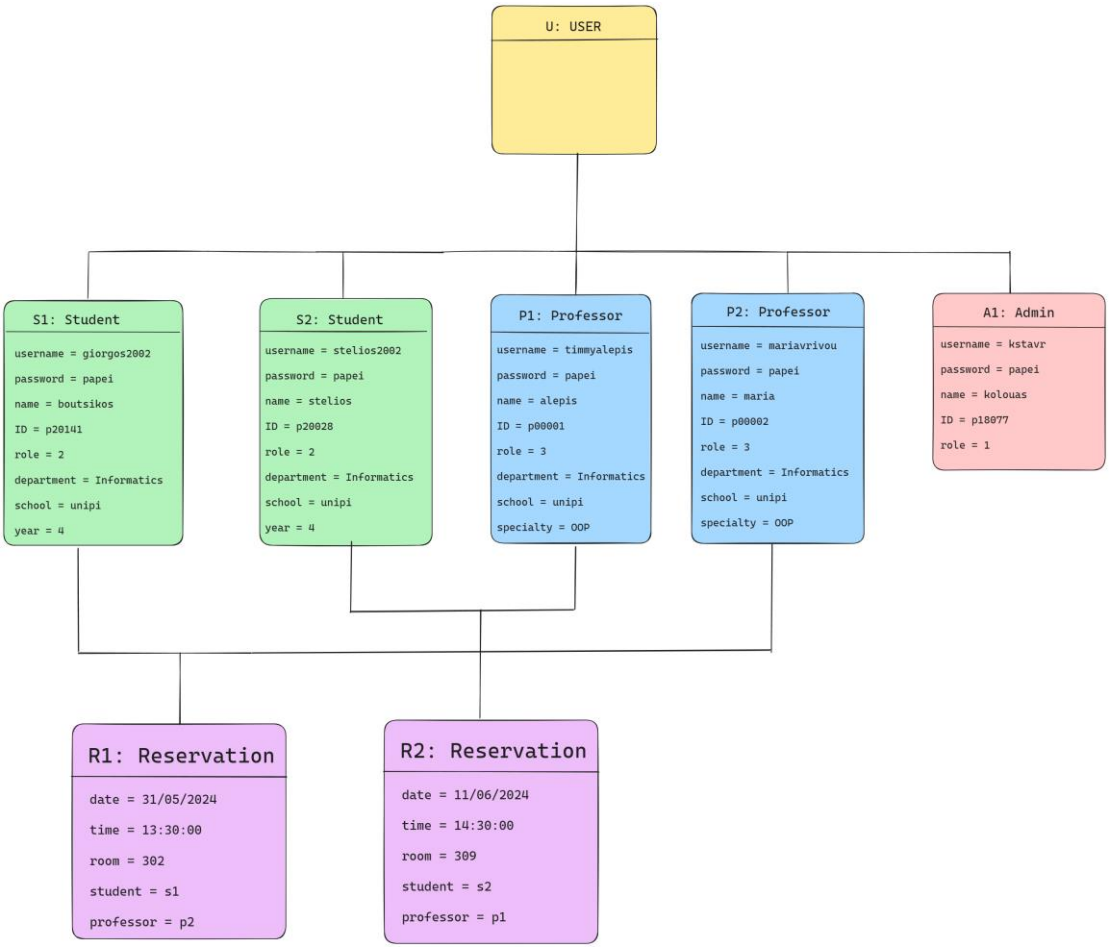
1. Διάγραμμα Περιπτώσεων Χρήσης (2η έκδοση) (Users Case Diagram)



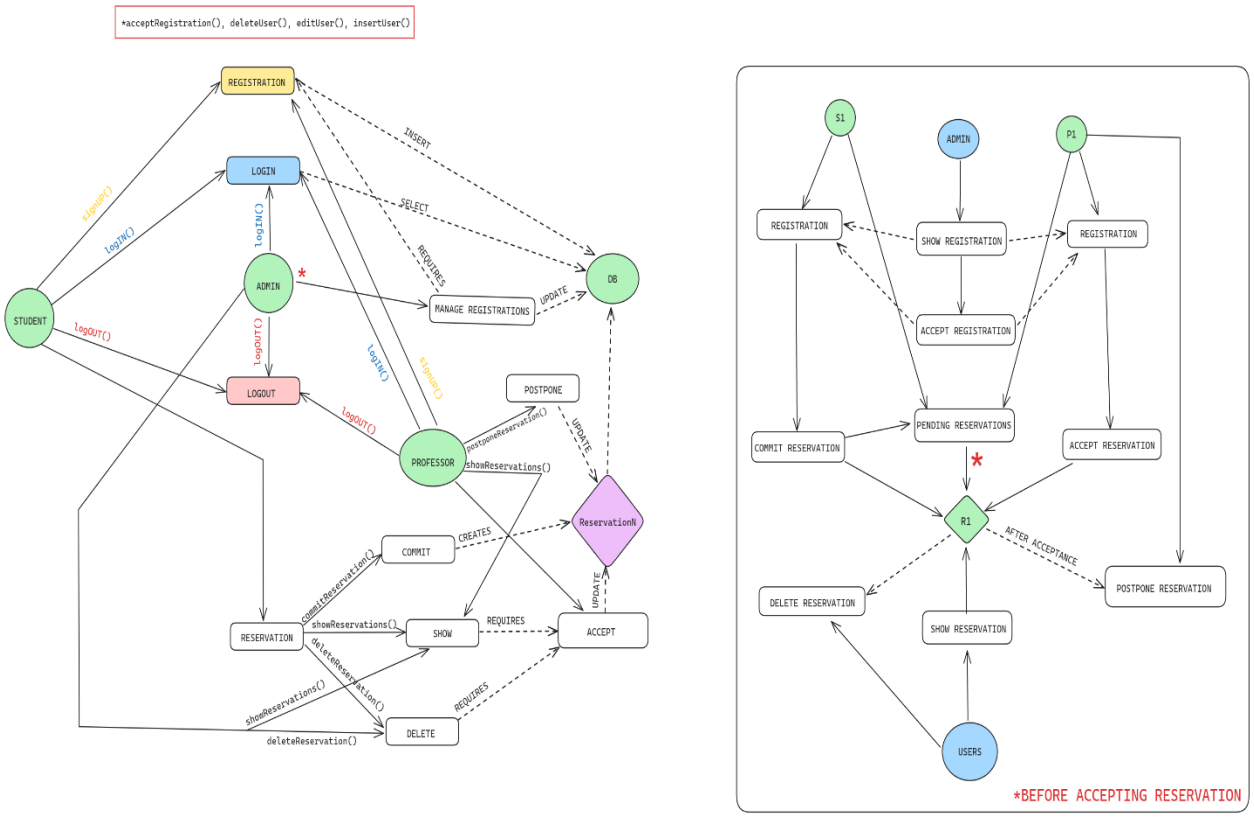
2. Διάγραμμα Τάξεων (2η έκδοση) (Class Diagram)



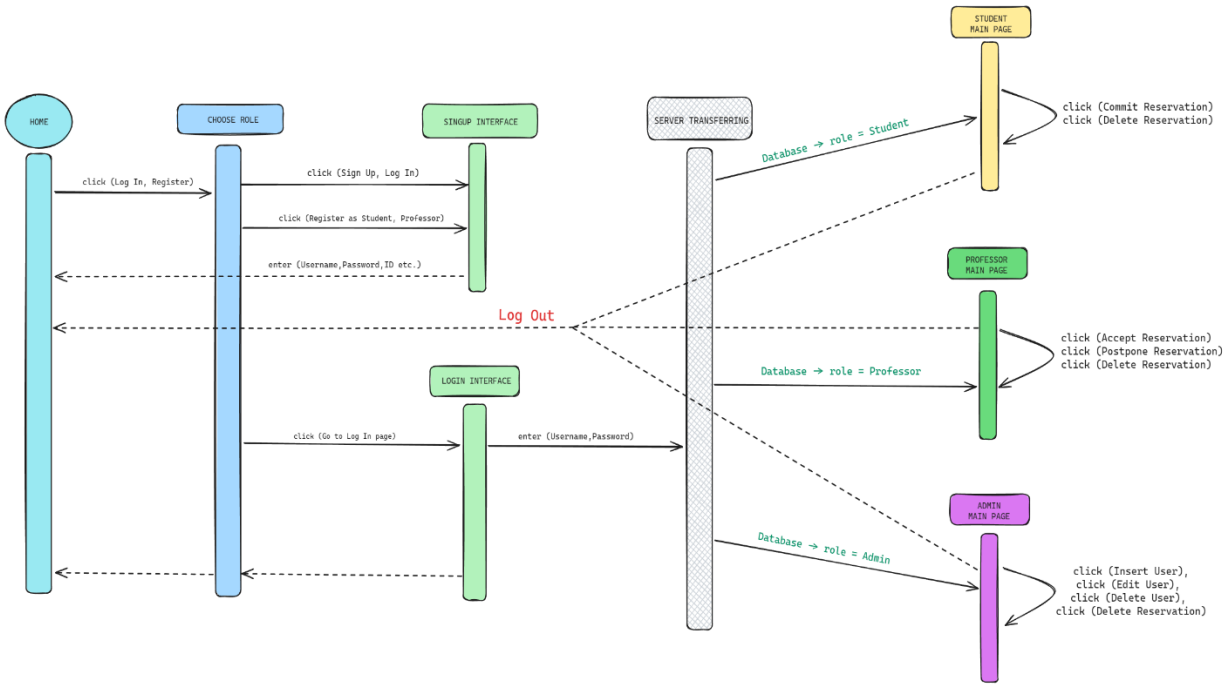
3. Διάγραμμα Αντικειμένων (Objects Diagram) (1η έκδοση)



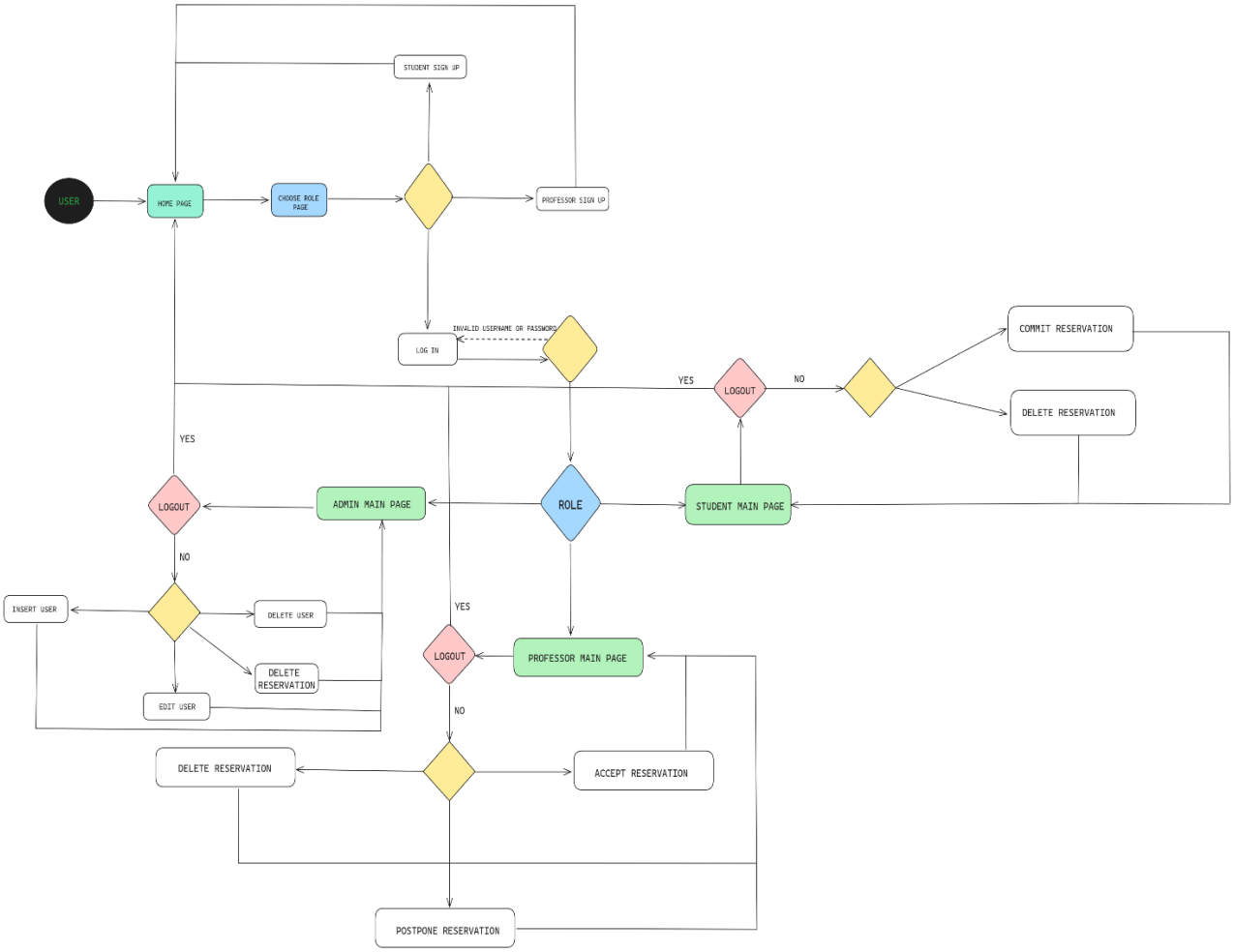
4. Διάγραμμα Συνεργασίας (Synergy Diagram) (1η έκδοση)



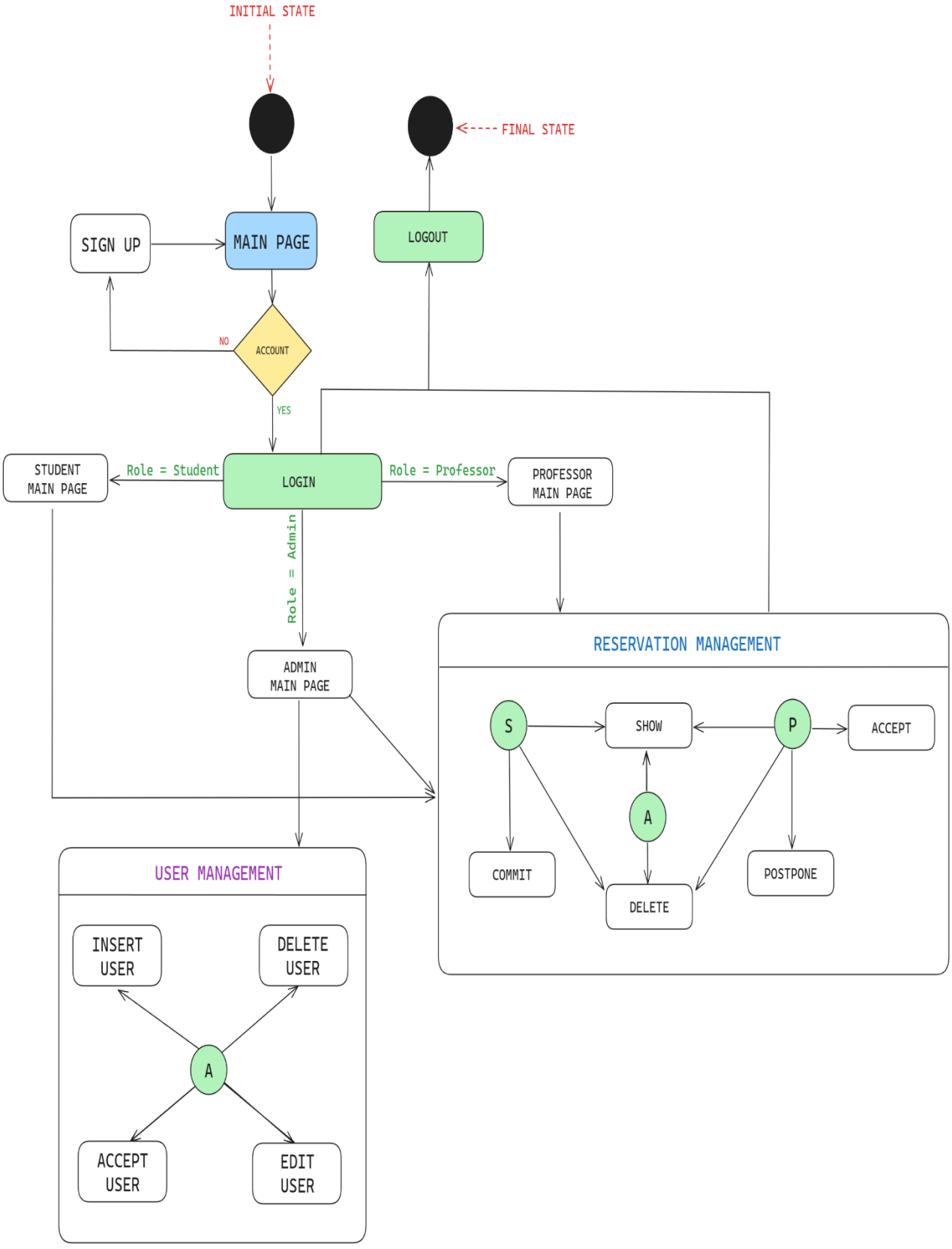
5. Διάγραμμα Σειράς (Sequence Diagram) (1η έκδοση)



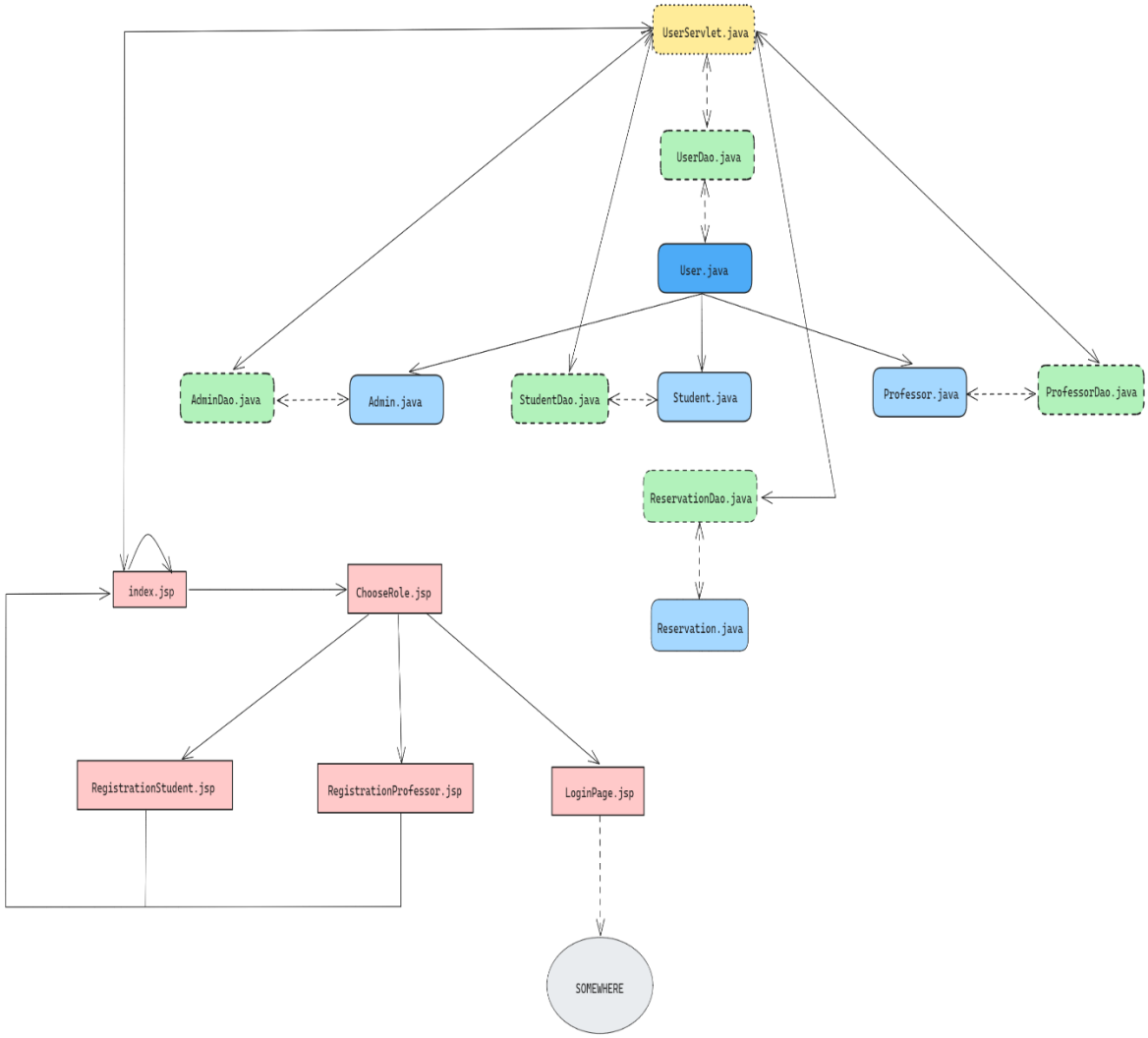
6. Διάγραμμα Δραστηριοτήτων (Activities Diagram) (1η έκδοση)



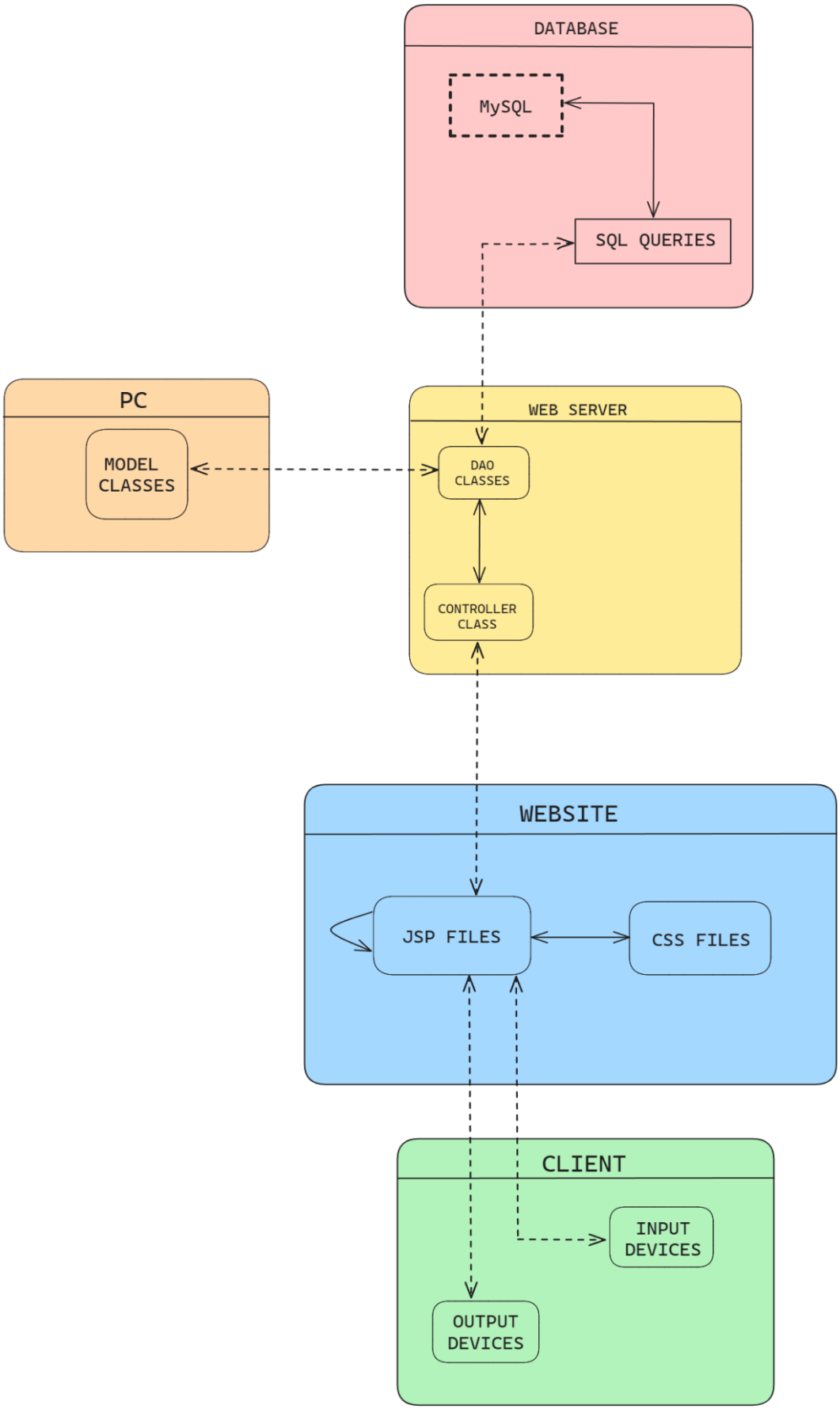
7. Διάγραμμα Καταστάσεων (1η έκδοση)



8. Διάγραμμα Εξαρτημάτων (1η έκδοση)



9. Διάγραμμα Διανομής (1η έκδοση)



3. Υλοποίηση – Έλεγχος

1. Υλοποίηση: 1η εκτελέσιμη έκδοση

Η υλοποίηση βρίσκεται μέσα στους φακέλους project με τα αρχεία .jsp, .xml, και .java.

Επίσης, θα βρείτε και την εργασία μας στο παρακάτω link: https://github.com/stelios2002/ergasia_texnologies_logismikou

2. Αναφορά ελέγχου για την 1η εκτελέσιμη έκδοση

Χρησιμοποιώντας τον server **Apache Tomcat** (Version 9) τρέχει το πρόγραμμα στον επιλεγόμενο browser (ή πληκτρολογώντας την ιστοσελίδα http://localhost:8080/Reservation_App/). Λειτουργούν τα redirections ανάμεσα στις σελίδες που βλέπει ο χρήστης και η σύνδεση μεταξύ των κλάσεων.