PostgreSQL Triggers
A Full Stack Approach

Στολτίδης Αλέξανδρος, Κουτσούκης Νικόλαος

January 6, 2022

Contents

1 Εισαγωγή

Ο σωστός σχεδιασμός και η κατάλληλη υλοποίηση μιας βάσης δεδομένων προυποθέτουν την επίλυση ποικίλων ζητημάτων. Δύο από τα πιο βασικά ζητήματα αποτελούν η διατήρηση ορθών δεδομένων σε όλο το εύρος της βάσης καθώς και η προβολή της βάσης σε διάφορες εφάρμογες επιπέδου χρήστη. Κύριος μηχανισμός για την διατήρηση ορθότητας και ομοιομορφίας στην βάση, που προσφέρεται από το DBMS, αποτελούν τα *Triggers*. Σε αυτή την εργασία θα εξερευνήσουμε σε μεγάλο βάθος τόσο την λειτουρία όσο και στις διάφορες πρακτικές εφαρμόγες των Triggers. Στην συνέχεια, υλοποιούμε μια μικρή σε μέγεθος βάση η οποία ωστόσο να αξιοποιεί τις περισσότερες λειτουργίες που προσφέρονται από τα Triggers. Τέλος, προβάλουμε την βάση μας στον Internet Browser του χρήστη μέσω ενός Server.

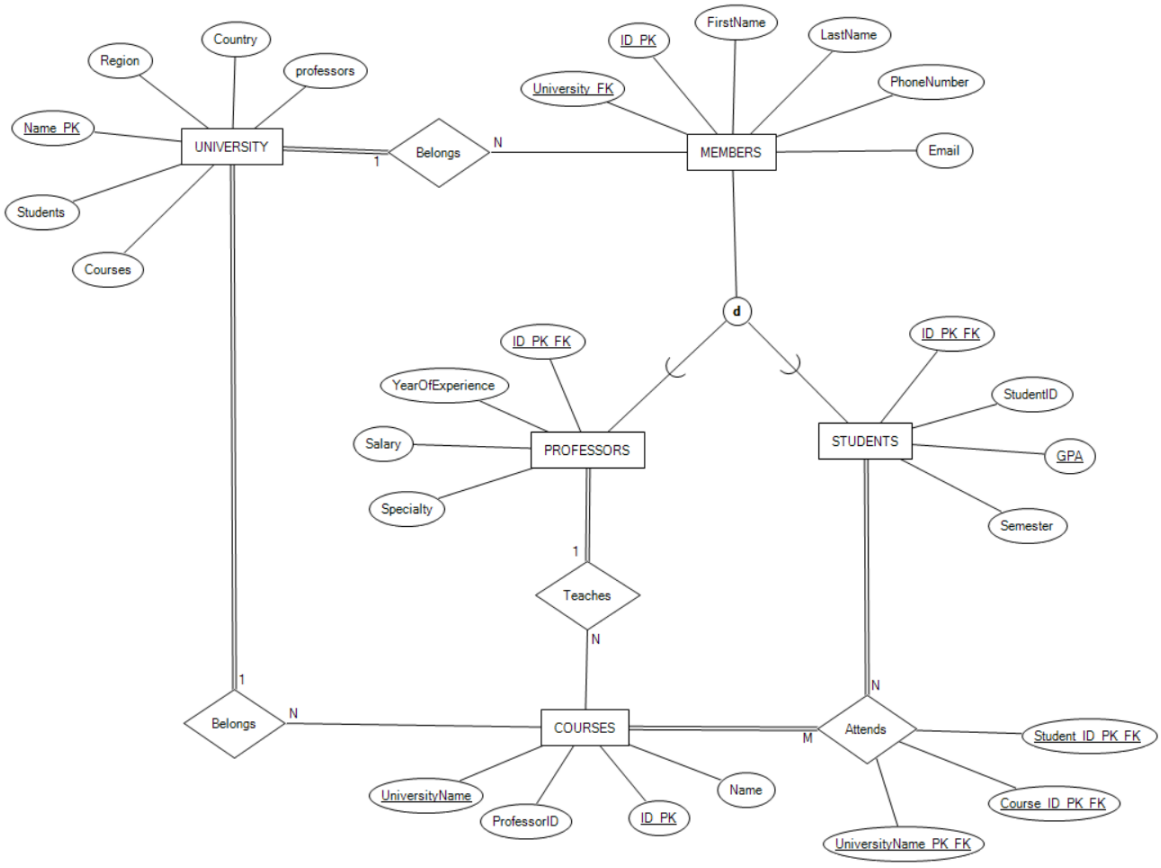
2 Εισαγωγή στα Triggers

Για την υλοποίηση μιας βάσης η οποία να εκμεταλλεύεται την δυνατότητα των Triggers πρέπει αρχικά να κατανοήσουμε τι ακριβώς είναι αυτοί οι μηχανισμοί καθώς και σε ποιους τόμεις της βάσης μπορούν να φανούν απαραίτητοι ή τουλάχιστον χρήσιμοι. Ένα Trigger, αρχικά, αποτελεί ένα σύνολο από εντολές οι οποίες εκτελούνται όταν προκύψει κάποιο Event σε ένα Table ή View της βάσης δεδομένων. Ένα Trigger εκτελείτε ώστε να τροποιηθουν δεδόμενα σε ένα ή περισσότερα σημεία της βάσης. Τα δεδομένα αυτά ειναι αλληλοεξαρτώμενα και συνεπώς μια αλλαγή σε ένα σημείο της βάσης επιβάλει την ενημέρωση των εξαρτημένων δεδομένων. Με την χρήση των Tiggers επιτυγχάνεται η ορθότητα των δεδομένων στο εύρος της βάσης.

3 Σχεδιασμός Βάσης Δεδομένων

Με σκοπό να εντρυφήσουμε σε μεγαλύτερο βάθος στην λειτουργία των Triggers δημιουργούμε μια βάση δεδομένων για ένα σύνολο πανεπιστημίων. Σκοπός της βάσης μας αποτελεί η εφαρμογή των περισσότερων σχέσεων μεταξύ των διαφόρων Entities (1-1, 1-N, N-N) καθώς και η υλοποίηση κληρονομικότητας (ISA) όπου αυτή είναι αναγκαία. Η βάση μας περιέχει τα διαθέσιμα πανεπιστήμια, τα μαθήματα που περιέχονται στο πρόγραμμα σπουδών του κάθε πανεπιστημίου, τους καθηγητές που εργάζονται και διδάσκουν μαθήματα στα αντίστοιχα πανεπιστήμια και τέλος τους φοιτητές οι οποίοι επιλέγουν τα μαθήματα τους ανα εξάμηνο. Μια πιο εκτενής περιγραφή της βάσης μπορεί να φανεί στο παρακάτω ERD (Entity - Relationship Diagram)

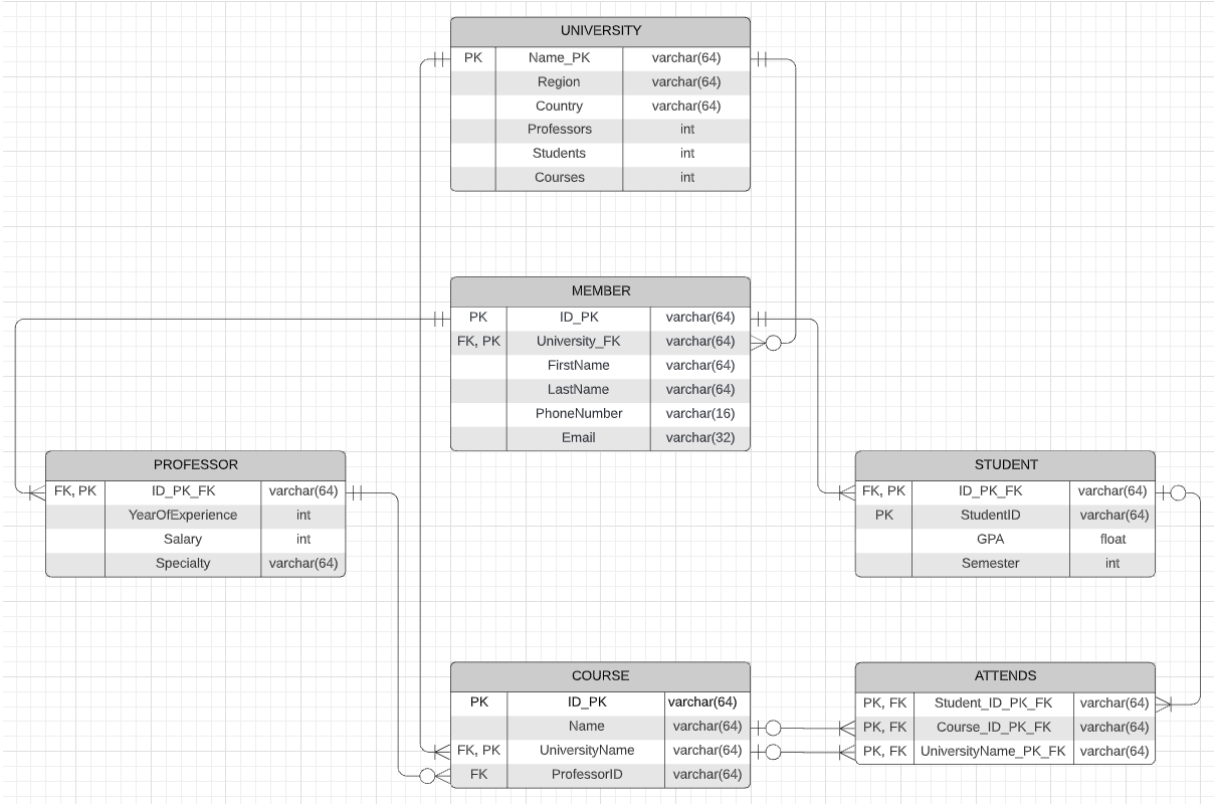
3.1 Entity-Relation Diagram



Σχήμα 1: ER Diagram

Μια καλύτερη αποτύπωση της βάσης για τον προγραμματιστή η οποία ταυτόχρονα ταιριάζει και με την απεικόνιση της βάσης από την οπτική του DBMS αποτελεί το Relational Data Model το οποίο φαίνεται παρακάτω.

3.2 Relational Data Model



Σχήμα 2: RD Diagram

4 Triggers

Για την υλοποίηση της βάσης δεδομένων μας δημιουργήθηκαν εννέα συνολικά triggers, τρία εκ των οποίων χρησιμοποιήθηκαν για την εντολή insert, ενώ τα υπόλοιπα για τη χρήση της εντολής delete για τα tables professors, students και courses αντίστοιχα. Παρακάτω παρουσιάζουμε αναλυτικότερα την λειτούργια του κάθε trigger.

4.1 Inserting Data

4.1.1 Professors

Για την εισαγωγή ενός professor στην βάση δεδομένων πρέπει να τροποποιήσουμε τόσο το table που περιέχει τους professors όσο και το table που περιέχει τα members καθώς ένας professor κληρονομεί τις ιδιότητες των members. Για την υλοποίηση αυτής της κληρονομικής σχέσης (ISA) δημιουργούμε ένα View το οποίο περιέχει τις πληροφορίες των professors και των members. Στην συνέχεια, χρησιμοποιούμε ένα instead of trigger το οποίο αντικαθιστά το insert ώστε η πληροφορία του νέου professor να κατανεμηθεί στα σωστά tables των members και των professors. Τέλος, γίνεται και ένα update στο table universities ώστε να αυξηθεί η τιμή του professors attribute κατά ένα στο αντίστοιχο university.

4.1.2 Students

Αντίστοιχα για τους students με την ίδια λογική επιτυγχάνεται η κατανομή της πληροφορίας στα tables, members και students καθώς και η ενημέρωση του αντίστοιχου students attribute για το κατάλληλο university.

4.1.3 Courses

Κατά την διαδικασία εισαγωγής των courses τα πράγματα είναι πιο απλά. Αρχικά υλοποιούμε ένα after insert trigger το οποίο εκτελείτε αμέσως μετά την εισαγωγή ενός course και έχει ως σκοπό να αυξήσει την τιμή του courses attribute για το αντίστοιχο university κατα ένα.

4.2 Deleting Data

4.2.1 Courses

Για την διαγραφή ενός μαθήματος δημιουργούμε ένα before delete trigger το οποίο εκτελείτε πριν διαγραφή του μαθήματος για να διαγράψει από το table attends τις εγγραφές όπου υπάρχει το αντίστοιχο μάθημα προς διαγραφή. Στην συνέχεια εκτελείται μια ενημέρωση στο table universities μειώνοντας την τιμή των courses κατά 1 για το αντίστοιχο υνιερσιτψ.

4.2.2 Students

Για να διαγράψουμε ένα φοιτητή χρησιμοποιούμε δύο triggers. Αρχικά εκτελείτε ένα before delete trigger το οποίο διαγράφει από το table attends κάθε εισαγωγή όπου υπάρχει ο αντίστοιχος φοιτητής και στην συνέχεια κάνει ένα update στο table universities μειώνοντας την τιμή των students κατά ένα για το αντίστοιχο university. Τέλος, αφού ο φοιτητής έχει διαγραφεί ένα επιπλέον after delete trigger εκτελείται το οποίο διαγράφει απο το table members τον αντίστοιχο φοιτητή.

4.2.3 Professors

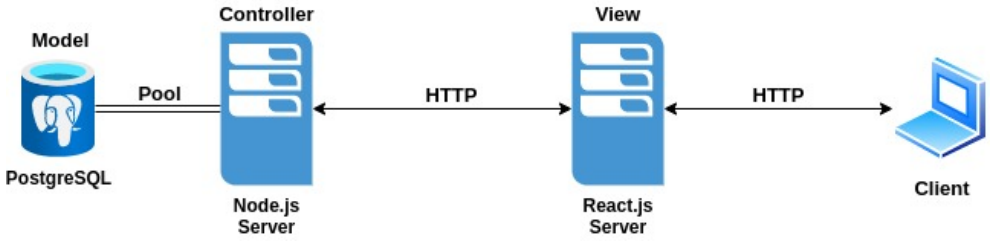
Η διαγραφή ενός καθηγητή χρειάζεται, όπως και του φοιτητή, δύο triggers. Αρχικά εκτελείτε ένα before delete trigger το οποίο καλεί την εντολή delete courses για τα μαθήματα που διδάσκει ο συγκεκριμένος καθηγητής. Η εντολή αυτή με την σειρά της εκτελεί τα αντίστοιχα triggers που προαναφερθήκαν για την διαγραφή ενός μαθήματος. Επιπλέον, εκτελείται μια ενημέρωση στο table universities μειώνοντας την τιμή των professors κατά ένα για το αντίστοιχο university. Τέλος, αφού ο καθηγητής έχει διαγραφεί ένα επιπλέον after delete trigger εκτελείται και διαγράφει απο το table members τον αντίστοιχο καθηγητή.

4.2.4 University

Για την διαγραφή ενός πανεπιστήμιου χρησιμοποιείται ένα before delete trigger το οποίο καλεί τις εντολές delete professors και delete students οι οποίες με την σειρά τους πυροδοτούν τα δικά τους triggers. Οι συνεχόμενες αυτές πυροδοτήσεις έχουν ως αποτέλεσμα την διαγραφή οποιασδήποτε πληροφορίας σχετικής με το πανεπιστήμιο που είναι προς διαγραφή. Η διαγραφή των μαθημάτων του πανεπιστημίου γίνεται μέσω των triggers των φοιτητών ή των καθηγητών.

5 Server Architecture

Για την παρουσίαση της βάσης στις διάφορες εφαρμογές επιπέδου χρήστη θα χρειαστούμε έναν δυναμικό server ο οποίος να συνδέεται άμεσα με την βάση και ταυτόχρονα να απαντάει στα διάφορα requests των χρηστών. Ο server αυτός μπορεί εύκολα να χρησιμοποιηθεί από εφαρμογές όπως το cURL ή το Postman. Για την βελτίωση της εμπειρίας αλληλεπίδρασης του χρήστη με την βάση μέσω του server εισάγουμε έναν ακόμη server ο οποίος αναλαμβάνει το rendering μιας απλής σε περιεχόμενο σελίδας η οποία ωστόσο να εκμεταλλεύεται τις περισσότερες δυνατότητες της βάσης. Η δομή που περιγράφουμε παραπάνω ονομάζεται MVC (Model-View-Controller) και παρουσιάζεται αναλυτικότερα στην παρακάτω εικόνα.



Σχήμα 3: Implemented MVC Architecture

6 Node.js Server (Controller)

Για την υποστήριξη επικοινωνίας μεταξύ της βάσης και των διάφορων εφαρμογών επιπέδου χρήστη ο προγραμματισμός ενός RESTful API είναι απαραίτητος. Το REST API παρέχει πρόσβαση μέσω προγραμματισμού στις διάφορες οντότητες της βάσης δεδομένων. Το API που σχεδιάσαμε εμείς είναι γραμμένο σε Node.js με χρήση του Framework Express.js για πιο εύκολο σχεδιασμό του routing. Παρακάτω παρουσιάζονται αναλυτικά τα διαθέσιμα routes, οι μέθοδοι από requests που υποστηρίζονται από τα αντίστοιχα routes καθώς και οι διαδικασίες που εκτελούνται ανάλογα με τα requests των χρηστών.

6.1 Routes

Όλα τα routes απαντάνε με τα αντίστοιχα δεδομένα σε μορφή JSON. Σε περιπτώσεις σφάλματος ο server απαντάει με HTTP Status 400 στα requests που προσπαθούν να κάνουν κάποια αλλαγή στην βάση και με HTTP Status 500 στα requests που απλά ζητάνε δεδομένα για προβολή από την βάση. Παρακάτω περιγράφονται τα routes για τα αντίστοιχα πεδία της βάσης.

6.1.1 Universities

/universities/all/	GET	Return all available from universities.
/universities/:university/professors/	GET	Return professors registered at a certain university.
/universities/:university/students/	GET	Return students registered at a certain university.
/universities/:university/courses/	GET	Return every course that a certain university. entails
/universities/register/	PUT	Register a university and return it's information.
/universities/:university/	DELETE	Delete a university and return it's information.

6.1.2 Professors

/professors/all/	GET	Return all available professors.
/professors/:professor/	GET	Return information about a certain professor.
/professors/register/	PUT	Register a professor and return it's information.
/professors/:professor/	DELETE	Delete a professor and return it's information.

6.1.3 Students

/students/all/	GET	Return all available students.
/students/:student/	GET	Return information about a certain student.
/students/register/	PUT	Register a student and return it's information.
/students/:student/	DELETE	Delete a student and return it's information.

6.1.4 Courses

/courses/all/	GET	Return all available courses.
/courses/university/:university/	GET	Return courses registered to certain university.
/courses/professor/:professor/	GET	Return courses that a certain professor is teaching.
courses/student/:student/	GET	Return courses that a certain student is attending.
/courses/:course/:university/students/	GET	Return all students attending a certain course.
/courses/register/	PUT	Register a course to a university.
/courses/:student/register/	PUT	Register a certain student to a course.
/courses/:course/:university/	DELETE	Delete a certain course from a university.
/courses/:student/:course/:university/	DELETE	Delete a student a course course.

6.2 Τεχνικά Χαρακτηριστικά

6.2.1 Σύνδεση με Clients

Ο server ακούει στο port 8080 και δέχεται AJAX requests από τον client. Ο client γνωρίζει εκ των προτέρων το port του server. Σε στάδιο εκτός του development σε περίπτωση που αλλάξει το port πρέπει να ενημερώσουμε τον client.

6.2.2 Σύνδεση με Βάση

Η βάση ακούει στο port 5432. Ο server δημιουργεί ένα pool συνδέσεων με την βάση και επικοινωνεί άμεσα μαζί της. Τα διαπιστευτήρια για την σύνδεση στην βάση φαίνονται παρακάτω και προφανώς είναι προσωρινά για την διαδικασία του development.

```
{
  user: "postgres",
  host: "127.0.0.1",
  database: "postgres",
  password: "postgres",
  port: 5432
}
```

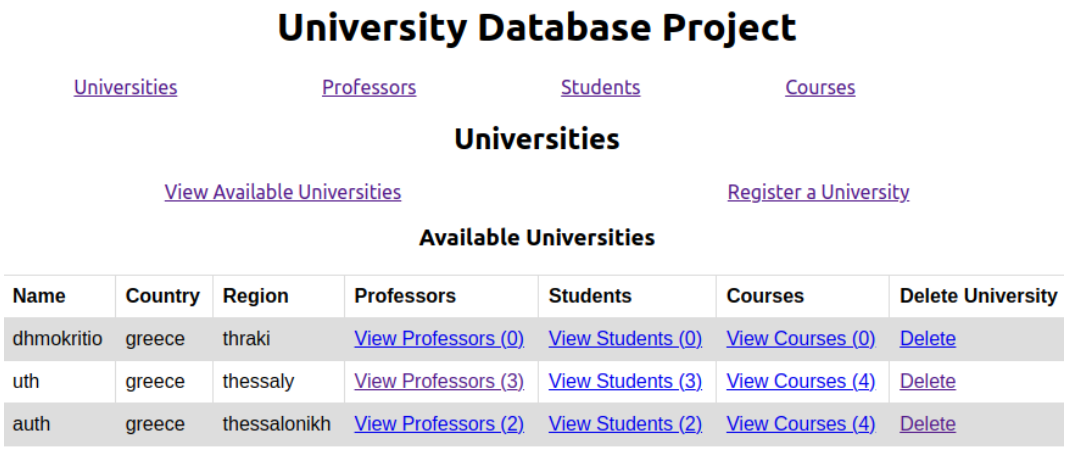
7 React.js Server (View)

Για την αλληλεπίδραση των χρηστών με την βάση μέσω του server δημιουργούμε ένα γραφικό περιβάλλον το οποίο να τρέχει στο browser του κάθε χρηστη. Για την σχεδίαση του περιβάλλοντος αυτού χρησιμοποιούμε την React.js, μια βιβλιοθήκη δηλαδή της JavaScript η οποία αναλαμβάνει το rendering. Ο σχεδιασμός του front end δεν είναι απαραίτητος να υλοποιηθεί αποκλειστικά με JavaScript καθώς αυτό που πραγματικά μας απασχόλει είναι η αλληλεπίδραση της βάσης με τον χρήστη. Αντίστοιχο γραφικό περιβάλλον μπορεί σίγουρα να δημιουργηθεί με διαφορετικές γλώσσες προγραμματισμού. Εμείς επιλέξαμε την χρήση JavaScript τόσο για το front end (view) όσο και για το back end (controller) για να διατηρησουμε μια καθολική ομοιομορφία στο προτζεκτ μας.

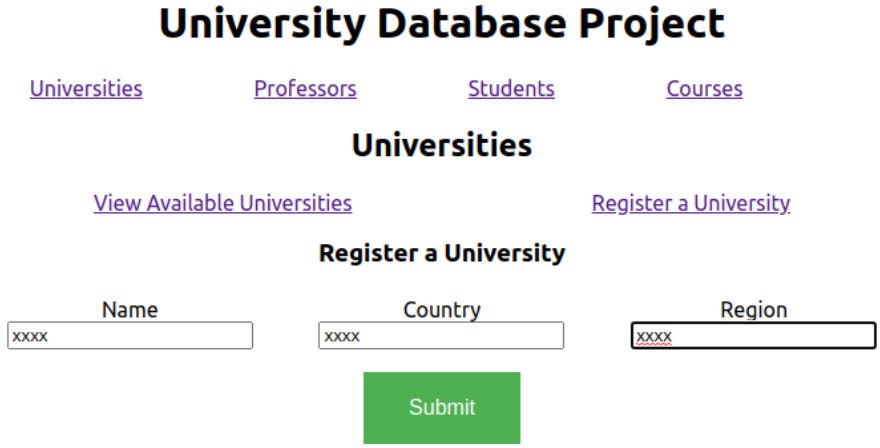
7.1 Εικόνες από την Εφαρμογή

Τέλος βλέπουμε λίγες εικόνες από την εφαρμογή. Προφανώς υπάρχουν πολλές περισσότερες δυνατότητες οι οποίες περιέχονται στο πρότζεκτ αλλά δεν φαίνονται σε αυτές τις εικόνες. Για την εκτέλεση της εφαρμογής ακολουθήστε τις οδηγίες που σας δίνουμε στο αρχείο install_and_run.pdf

7.1.1 Show Universities



7.1.2 Register a University



7.1.3 Show Professors

University Database Project

Universities Professors Students Courses

Professors

View Available Professors Register a Professor

Available Professors

ID	First Name	Last Name	E-Mail	Phone	Specialty	Universtity	Experience (Years)	Salary	Courses	Delete Professor
1	ABC	ABC	name1@gmail	6944325	ABC	uth	1	1000	Show Courses	Delete Professor
2	BCD	BCD	name2@gmail	6944565	BCD	uth	10	2000	Show Courses	Delete Professor
3	CDE	CDE	name3@gmail	6948994	CDE	auth	7	3000	Show Courses	Delete Professor
4	DEF	DEF	name@4gmail	6944122	DEF	auth	8	1500	Show Courses	Delete Professor
10	DEF	DEF	name@10gmail	6914122	DEF	uth	8	1500	Show Courses	Delete Professor

7.1.4 Show Courses

University Database Project

Universities Professors Students Courses

Courses

View Available Courses Register a Course

Available Courses

ID	Name	University	Professor	Students	Delete Course
ECE1	vaseis	uth	Professor Information	View Students	Delete
ECE2	vaseis2	uth	Professor Information	View Students	Delete
ECE1	vaseis	auth	Professor Information	View Students	Delete

8 Contact Information

Αλέξανδρος Στολτίδης (2824): stalexandros@uth.gr
Νικόλαος Κουτσούκης (2907): nkoutsoukis@uth.gr