SEG2105 – Introduction to Software Engineering

Android Project: Student Course Booking App

Submitted to
Dr. Omar Badreddin
& TA Ammar Rashed

Group 7
Fahmi Ahmed
Salina Bee
Karl-W Gueyie
Jaewoon Kim
Henry Tan

Faculty of Engineering
School of Electrical Engineering and Computer Science
July 31st, 2022

Table of Contents

List of Figures

List of Tables

Introduction

The objective of the project was to design an application that implemented basic course selection services intended for instructors and students on Android. The project allowed us to explore the functionalities of Android Studio while also applying the concepts that were learned in the Introduction to Software Engineering course. Users are able to register either as an instructor or as a student. Student accounts are able to search and enroll/unenroll in courses. Instructor accounts are able to assign/unassign themselves to courses and edit the information on those courses as needed. Moreover, the administrator account is able to manage courses and other user accounts.

1. UML Class Diagram

The following figure shows the final UML Class Diagram for the student course booking application that was designed.

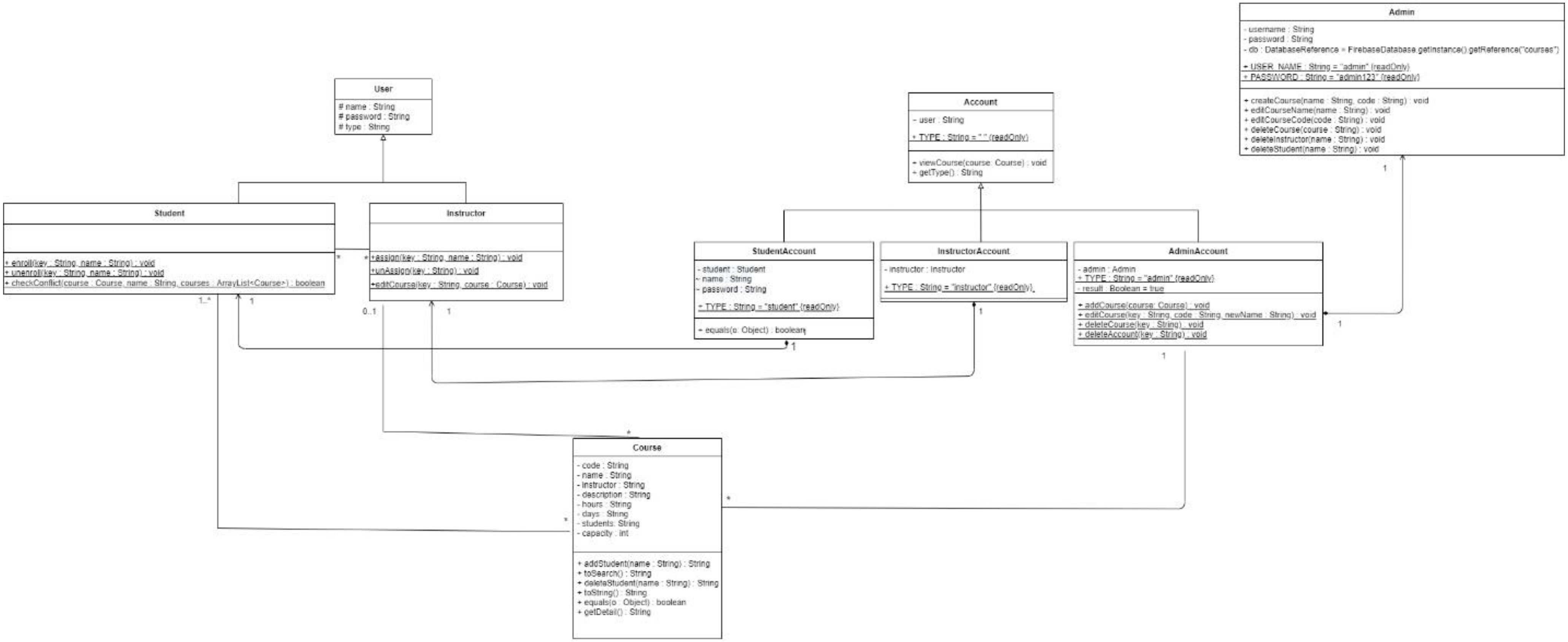

User
# name : String
# password : String
# type : String
Student
+ enroll(key : String, name : String) : void
+ unenroll(key : String, name : String) : void
+ checkConflict(course : Course, name : String, courses : ArrayList<Course>) : boolean
Instructor
+assign(key : String, name : String) : void
+unAssign(key : String) : void
+editCourse(key : String, course : Course) : void
Account
- user : String
+ TYPE : String = " " {readOnly}
+ viewCourse(course: Course) : void
+ getType() : String
Admin
- username : String
- password : String
- db : DatabaseReference = FirebaseDatabase.getInstance().getReference("courses")
+ USER_NAME : String = "admin" {readOnly}
+ PASSWORD : String = "admin123" {readOnly}
+ createCourse(name : String, code : String) : void
+ editCourseName(name : String) : void
+ editCourseCode(code : String) : void
+ deleteCourse(course : String) : void
+ deleteInstructor(name : String) : void
+ deleteStudent(name : String) : void
StudentAccount
- student : Student
~ name : String
- password : String
+ TYPE : String = "student" {readOnly}
+ equals(o: Object) : boolean
InstructorAccount
- instructor : Instructor
+ TYPE : String = "instructor" {readOnly}
AdminAccount
- admin : Admin
+ TYPE : String = "admin" {readOnly}
- result : Boolean = true
+ addCourse(course: Course) : void
+ editCourse(key : String, code : String, newName : String) : void
+ deleteCourse(key : String) : void
+ deleteAccount(key : String) : void
Course
- code : String
- name : String
- instructor : String
- description : String
- hours : String
- days : String
- students: String
- capacity : int
+ addStudent(name : String) : String
+ toSearch() : String
+ deleteStudent(name : String) : String
+ toString() : String
+ equals(o : Object) : boolean
+ getDetail() : String
*
*
1..*
1
0..1
1
1
1
1
1
1
*
*
*

***Figure 1**. UML Class Diagram from Deliverable 3.*

2. Task Distribution

The following table shows the roles and contributions of the members for each deliverable.

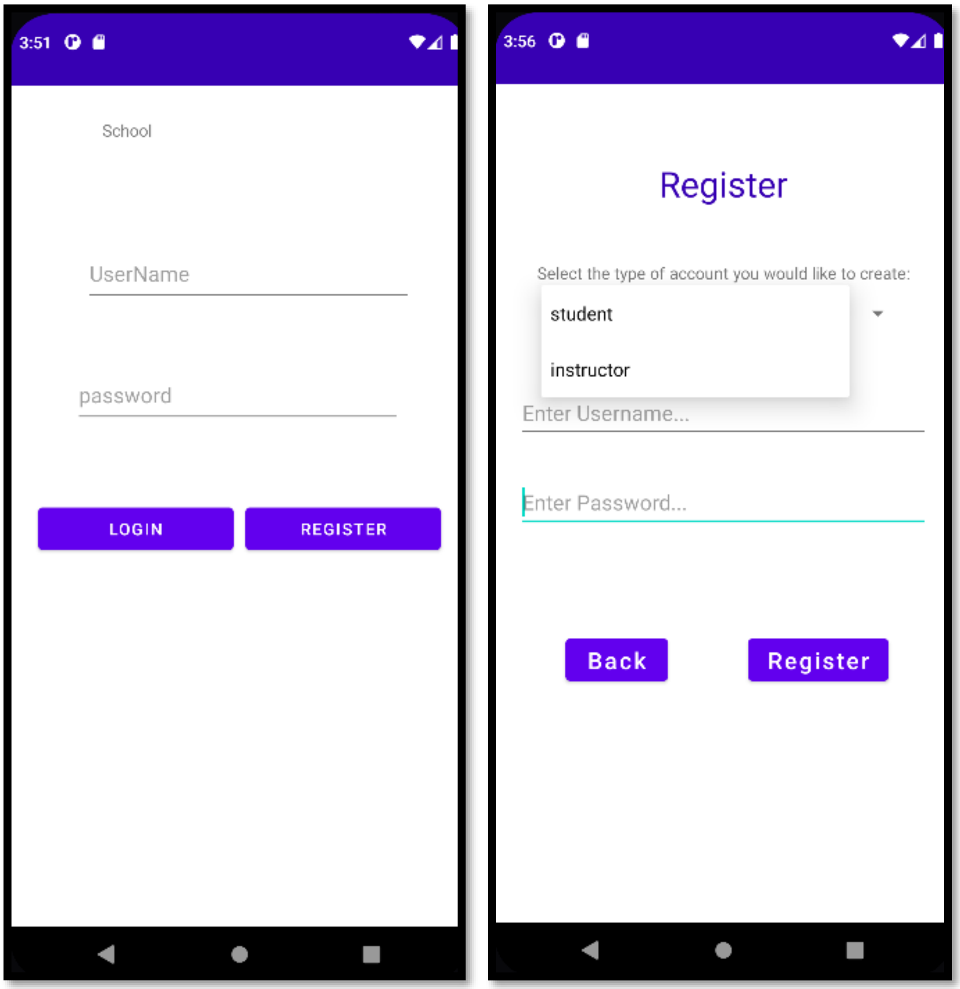
Table 1. *Roles and Contributions.*

Group Member	Deliverable 1	Deliverable 2	Deliverable 3
Fahmi	UML Class Diagram	Searching Courses for Instructors	Searching Courses for Students
Salina	Create Course and View/Edit Course for Admin	Edit Course for Instructors	View Courses for Students and Instructors
Karl	Login Page	UML Class Diagram	UML Class Diagram Project Report
Jaewoon	Register Page	Unit Testing	Unit Testing
Henry	Register Page Welcome Page Login Page	Assign/Unassign for Instructors	Enroll/Unenroll Time Conflict Checking

3. Application Screenshots

This section contains the screenshots of the user interfaces of the student course booking application. The screenshots demonstrate how the different users are able to navigate through the application.

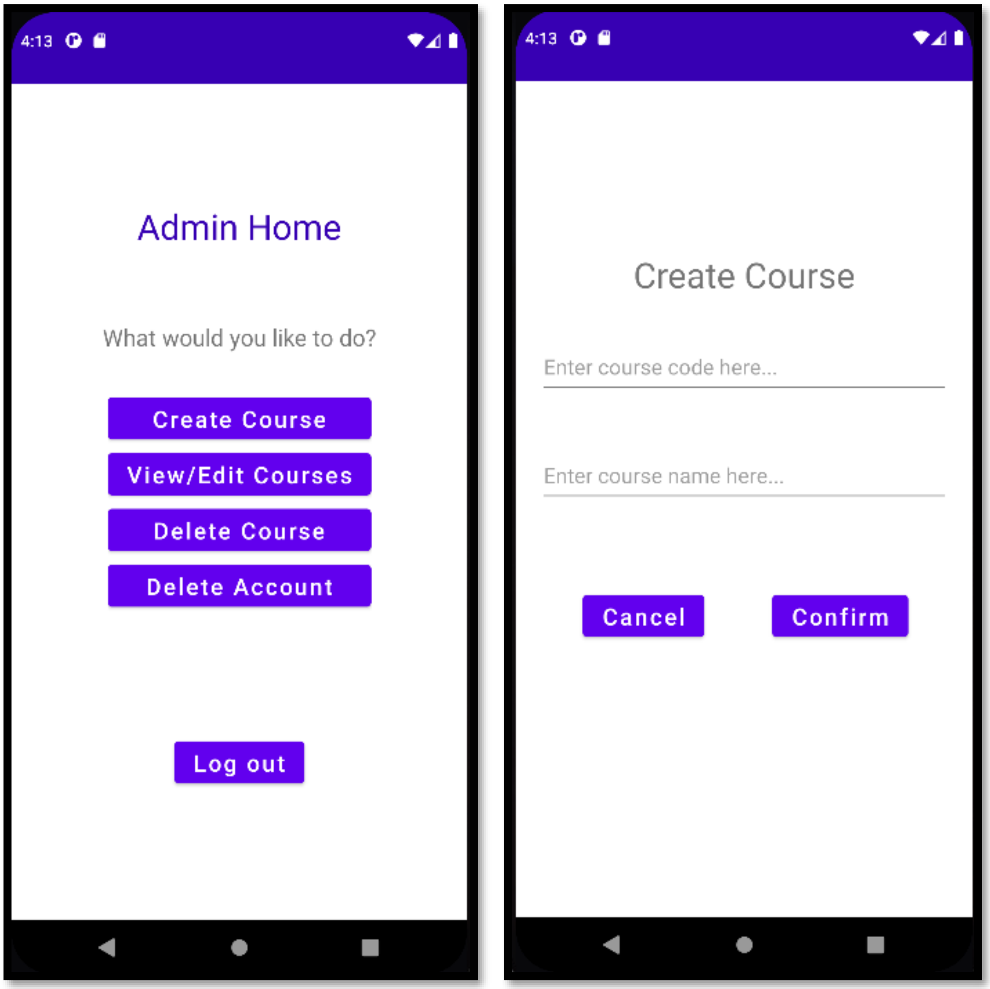
3.1. Main Page (Login/Register)



Select the type of account you would like to create:

__Figure 2.__ Screenshots of the login (left) and register (right) pages.

3.2. Admin Account



Admin Home

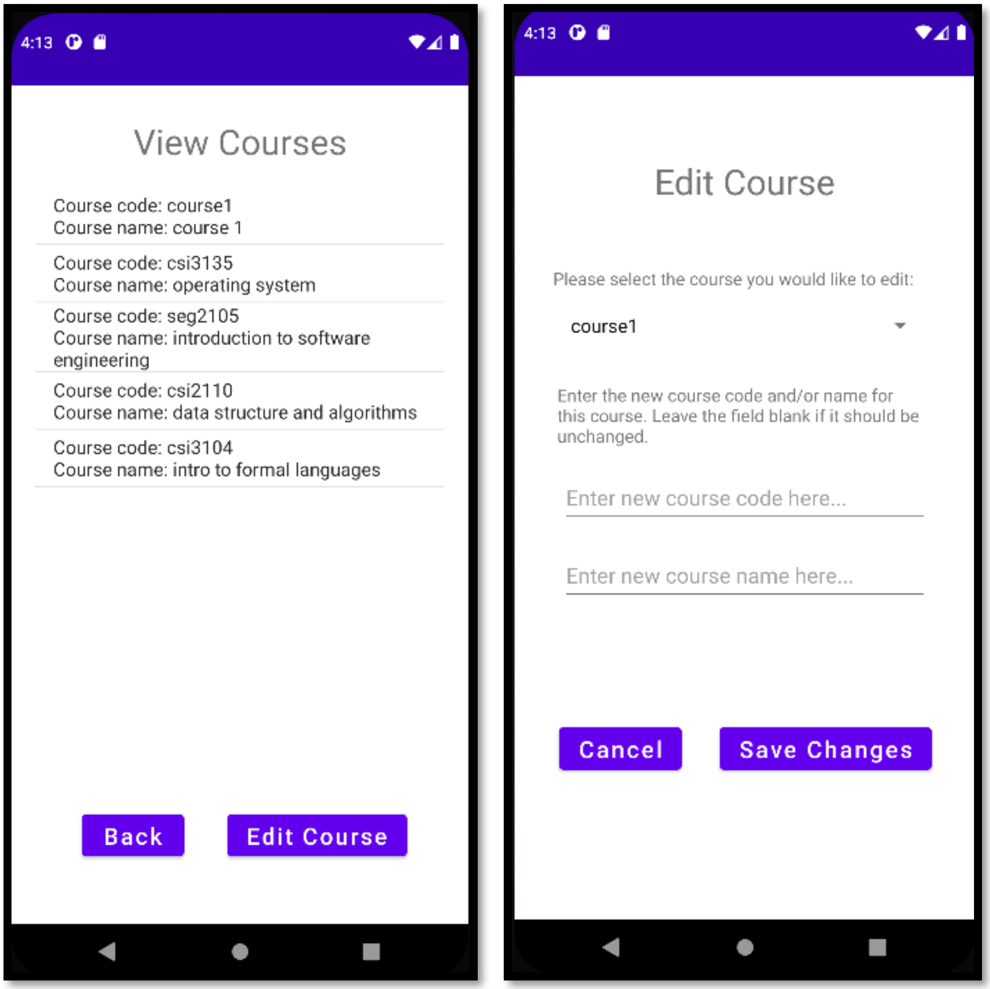
What would you like to do?

Create Course

Enter course code here...

Enter course name here...

Figure 3. *Screenshots of the Admin homepage (left) and Create Course page (right).*



View Courses

Course code: course1
Course name: course 1

Course code: csi3135
Course name: operating system

Course code: seg2105
Course name: introduction to software engineering

Course code: csi2110
Course name: data structure and algorithms

Course code: csi3104
Course name: intro to formal languages

Edit Course

Please select the course you would like to edit:

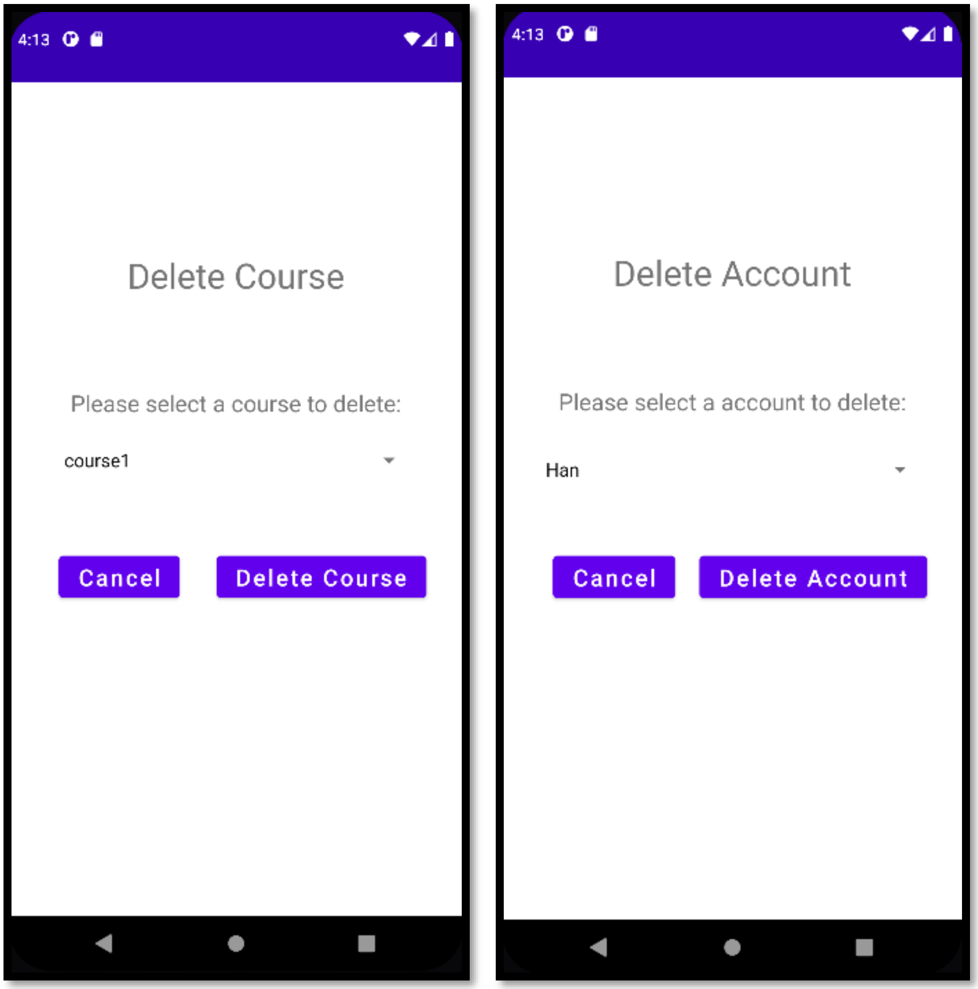
course1

Enter the new course code and/or name for this course. Leave the field blank if it should be unchanged.

Enter new course code here...

Enter new course name here...

Figure 4. *Screenshots of the View Courses page (left) and Edit Course page (right) for admin account.*



Delete Course

Please select a course to delete:

course1

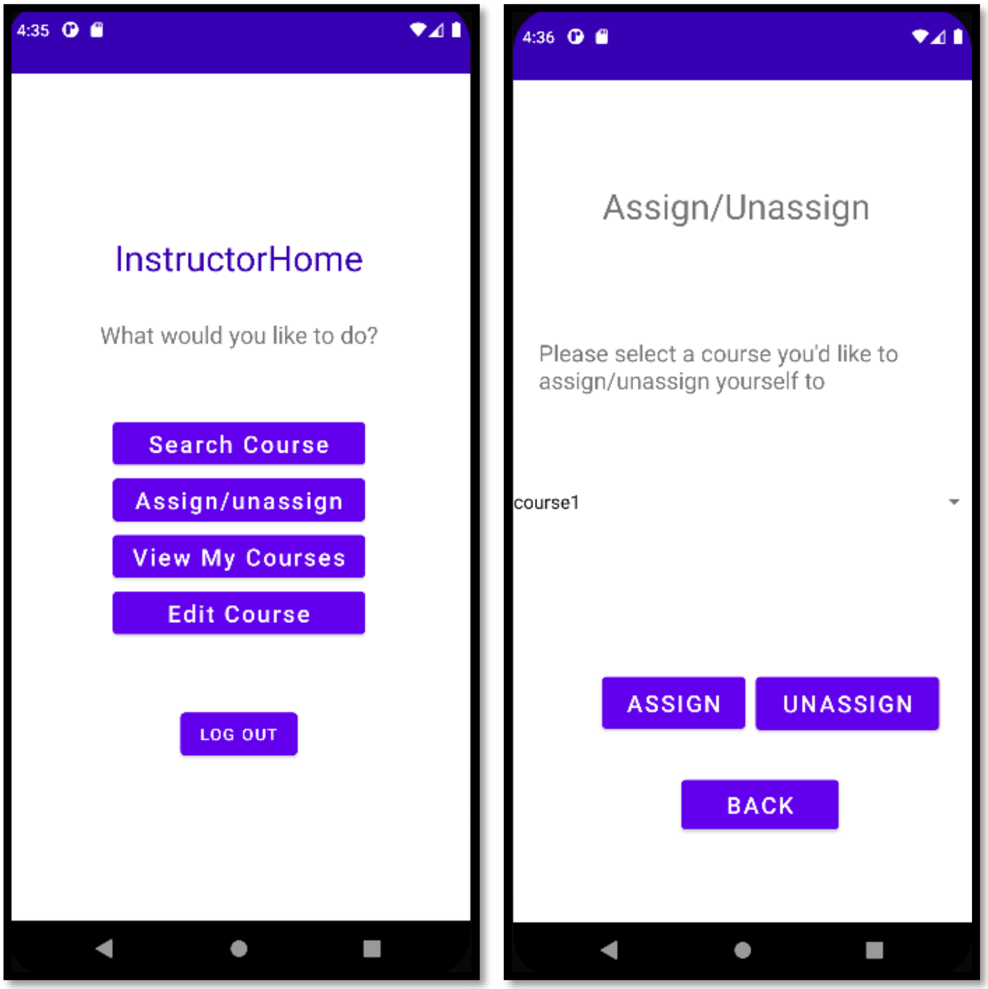
Delete Account

Please select a account to delete:

Han

Figure 5. *Screenshots of the Delete Course page (left) and Delete Account page (right) for admin account.*

3.3. Instructor Account



InstructorHome

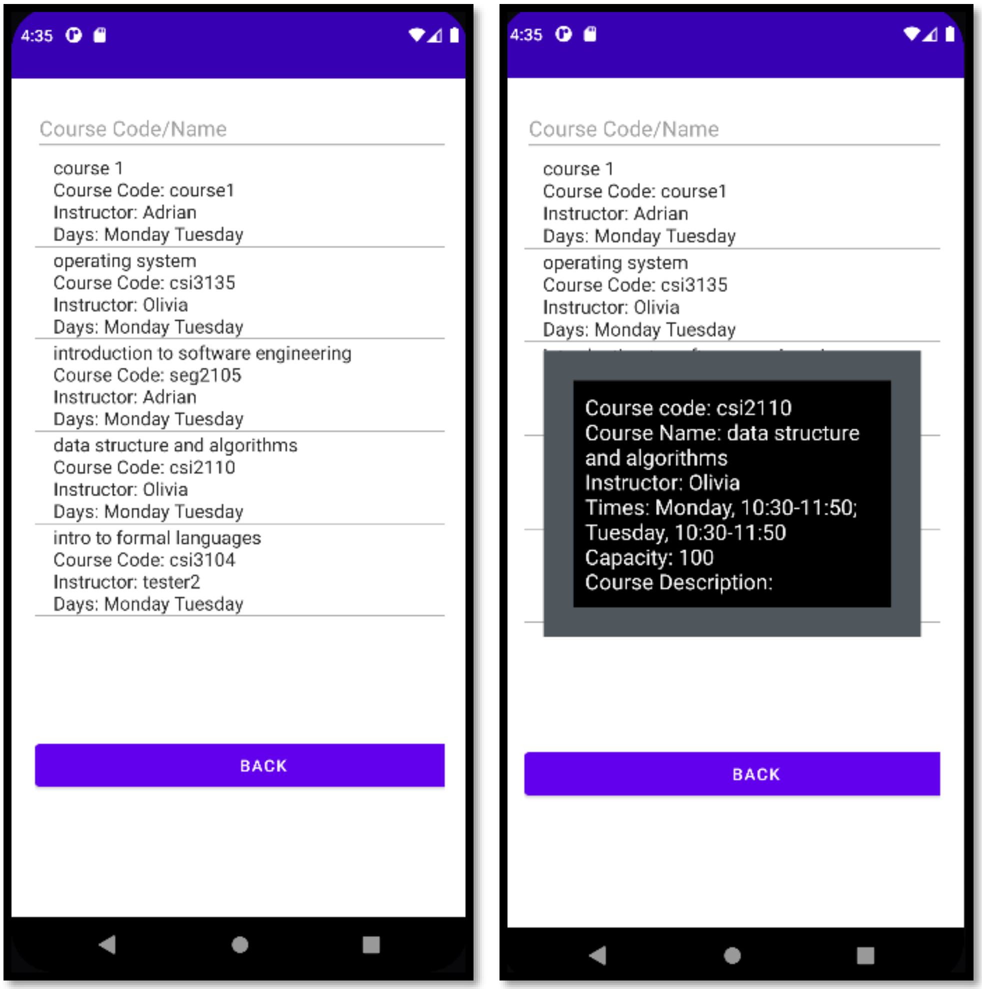
What would you like to do?

Assign/Unassign

Please select a course you'd like to assign/unassign yourself to

course1

Figure 6. *Screenshots of the Instructor homepage (left) and Assign/Unassign page (right).*



course 1
Course Code: course1
Instructor: Adrian
Days: Monday Tuesday

operating system
Course Code: csi3135
Instructor: Olivia
Days: Monday Tuesday

introduction to software engineering
Course Code: seg2105
Instructor: Adrian
Days: Monday Tuesday

data structure and algorithms
Course Code: csi2110
Instructor: Olivia
Days: Monday Tuesday

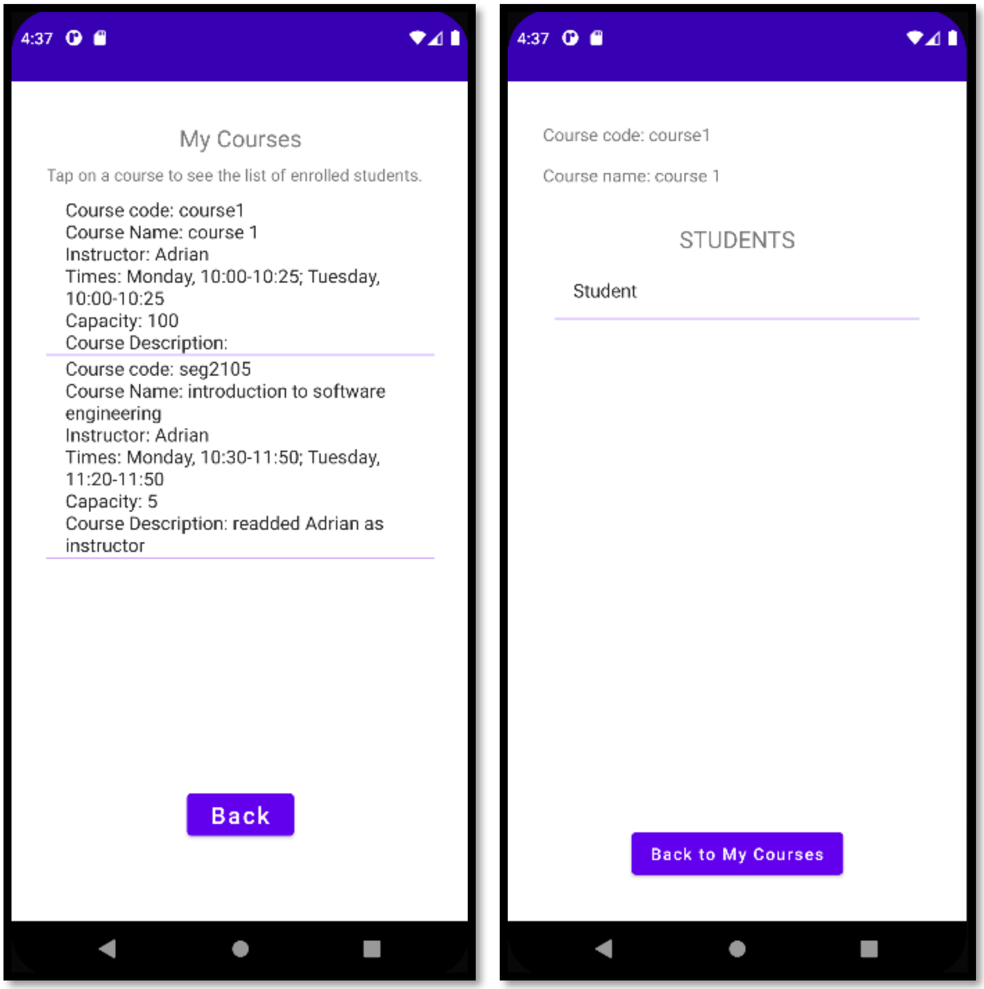
intro to formal languages
Course Code: csi3104
Instructor: tester2
Days: Monday Tuesday

course 1
Course Code: course1
Instructor: Adrian
Days: Monday Tuesday

operating system
Course Code: csi3135
Instructor: Olivia
Days: Monday Tuesday

Course code: csi2110
Course Name: data structure and algorithms
Instructor: Olivia
Times: Monday, 10:30-11:50; Tuesday, 10:30-11:50
Capacity: 100
Course Description:

Figure 7. *Screenshots of the Search Course page, without (left) and with popup window (right) for instructor account.*



My Courses

Tap on a course to see the list of enrolled students.

Course code: course1
Course Name: course 1
Instructor: Adrian
Times: Monday, 10:00-10:25; Tuesday, 10:00-10:25
Capacity: 100
Course Description:

Course code: seg2105
Course Name: introduction to software engineering
Instructor: Adrian
Times: Monday, 10:30-11:50; Tuesday, 11:20-11:50
Capacity: 5
Course Description: readded Adrian as instructor

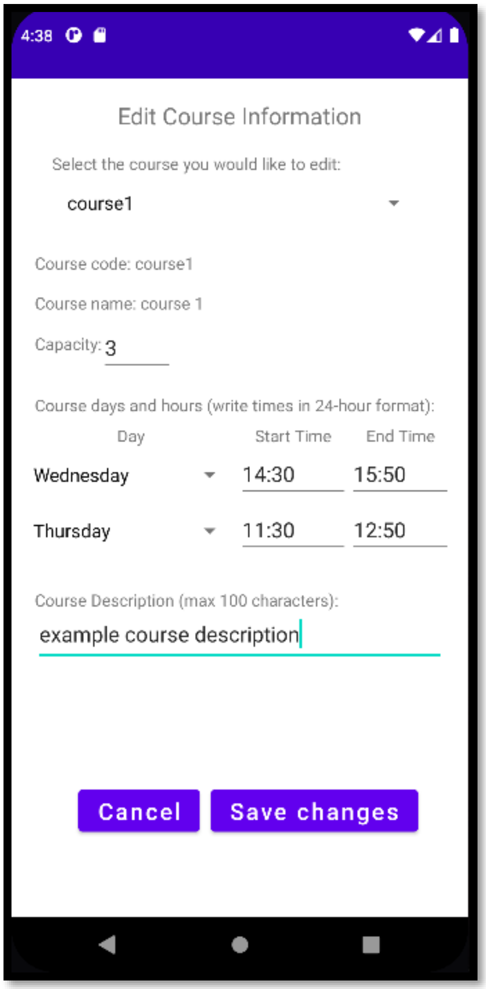
Course code: course1

Course name: course 1

STUDENTS

Student

Figure 8. *Screenshots of the View My Courses page (left) and the Student List page (right) for instructor account.*



Edit Course Information

Select the course you would like to edit:

course1

Course code: course1

Course name: course 1

Capacity: 3

Course days and hours (write times in 24-hour format):

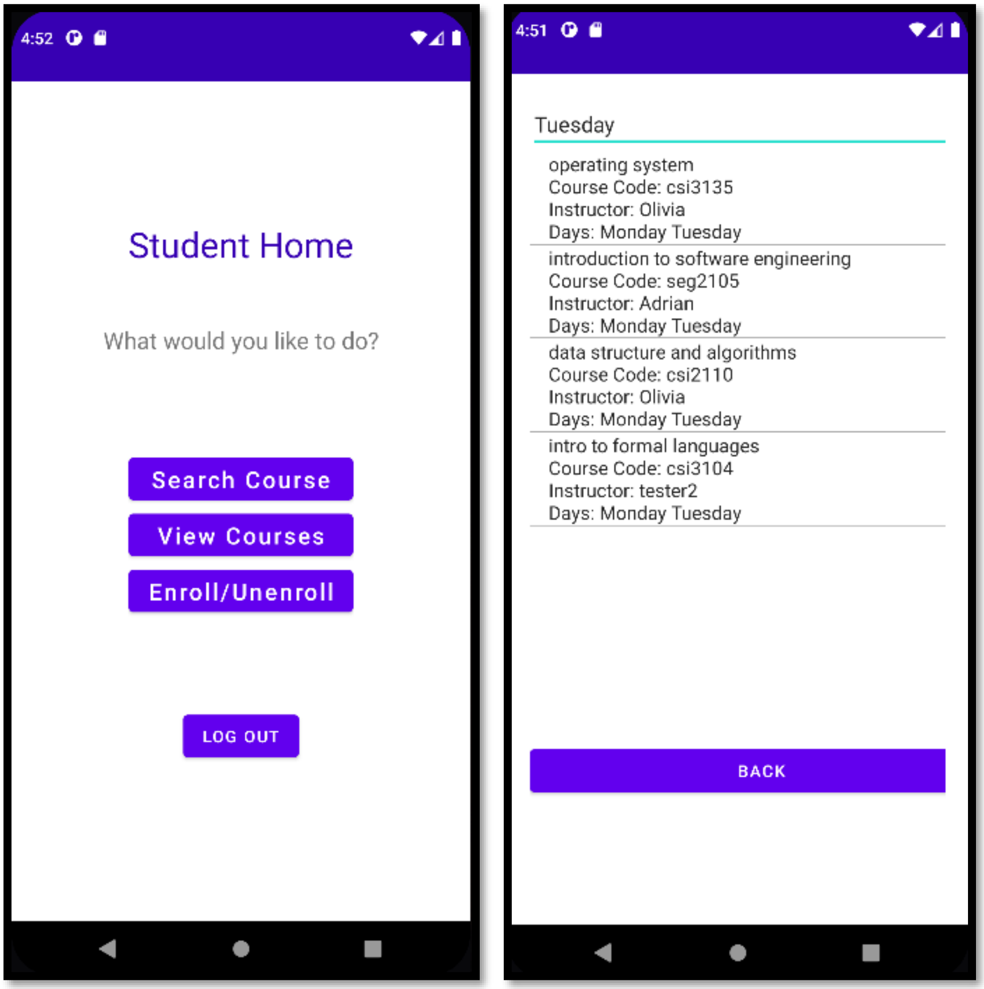
Day	Start Time	End Time
Wednesday	14:30	15:50
Thursday	11:30	12:50

Course Description (max 100 characters):

example course description

Figure 9. *Screenshot of the Edit Course page for instructor account.*

3.4. Student Account



Student Home

What would you like to do?

Tuesday

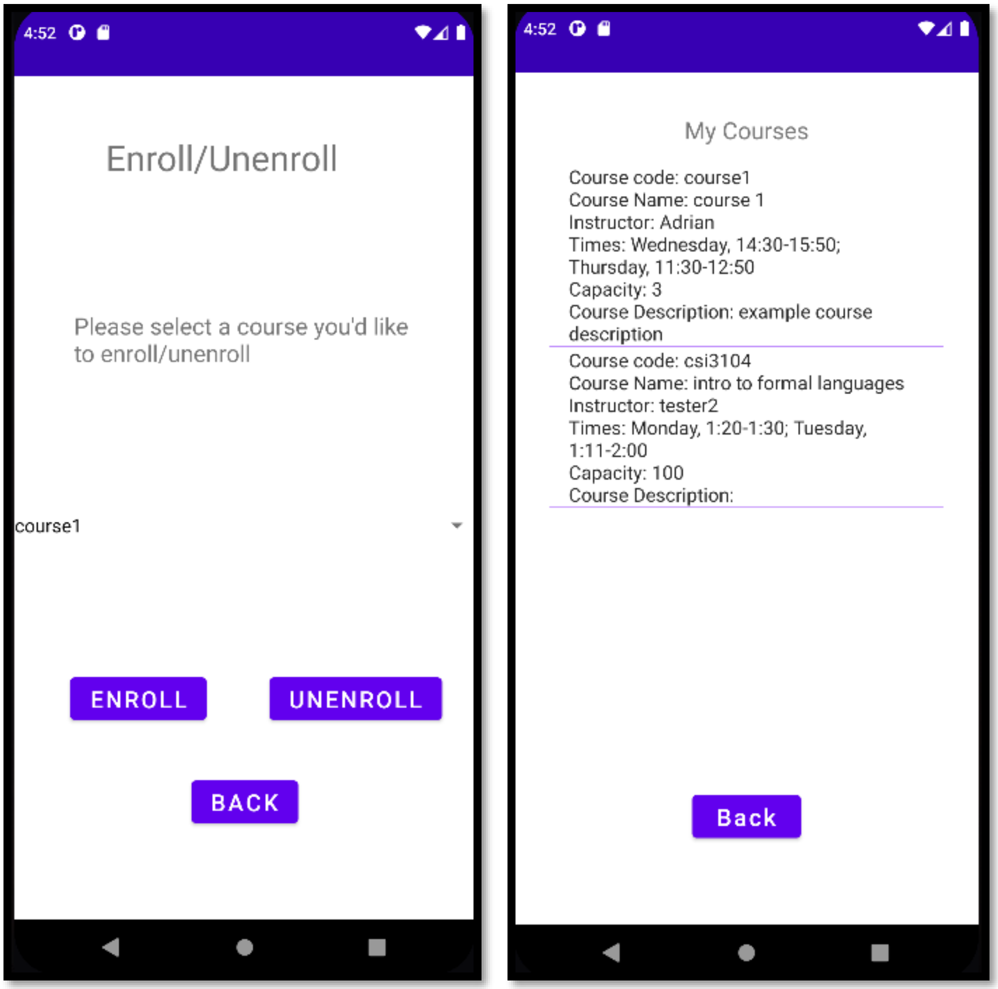
operating system
Course Code: csi3135
Instructor: Olivia
Days: Monday Tuesday

introduction to software engineering
Course Code: seg2105
Instructor: Adrian
Days: Monday Tuesday

data structure and algorithms
Course Code: csi2110
Instructor: Olivia
Days: Monday Tuesday

intro to formal languages
Course Code: csi3104
Instructor: tester2
Days: Monday Tuesday

***Figure 10**. Screenshots of the Student homepage (left) and Search Course page (right).*



Enroll/Unenroll

Please select a course you'd like to enroll/unenroll

course1

My Courses

Course code: course1
Course Name: course 1
Instructor: Adrian
Times: Wednesday, 14:30-15:50; Thursday, 11:30-12:50
Capacity: 3
Course Description: example course description

Course code: csi3104
Course Name: intro to formal languages
Instructor: tester2
Times: Monday, 1:20-1:30; Tuesday, 1:11-2:00
Capacity: 100
Course Description:

***Figure 11**. Screenshots of the Enroll/Unenroll page (left) and the View Courses page (right) for student account.*

Conclusion & Challenges

The objective of the project was successfully achieved. The Android application for course booking was built according to the project description.

During the project many challenges were faced and many lessons were learned. The most important challenges were the logistic ones. In deliverable 1, time management and communication were a big problem. However, that was fixed on deliverable 2 by having the roles outlined, setting smaller goals and deadlines, and scheduling weekly meetings to discuss the project.

Some technical challenges involved figuring out how Android Studio and Firebase worked and solving the unexpected issues. To deal with these challenges, it was best to start early and take into consideration unexpected problems in the deadline.

Appendices

Appendix A: Additional Figures

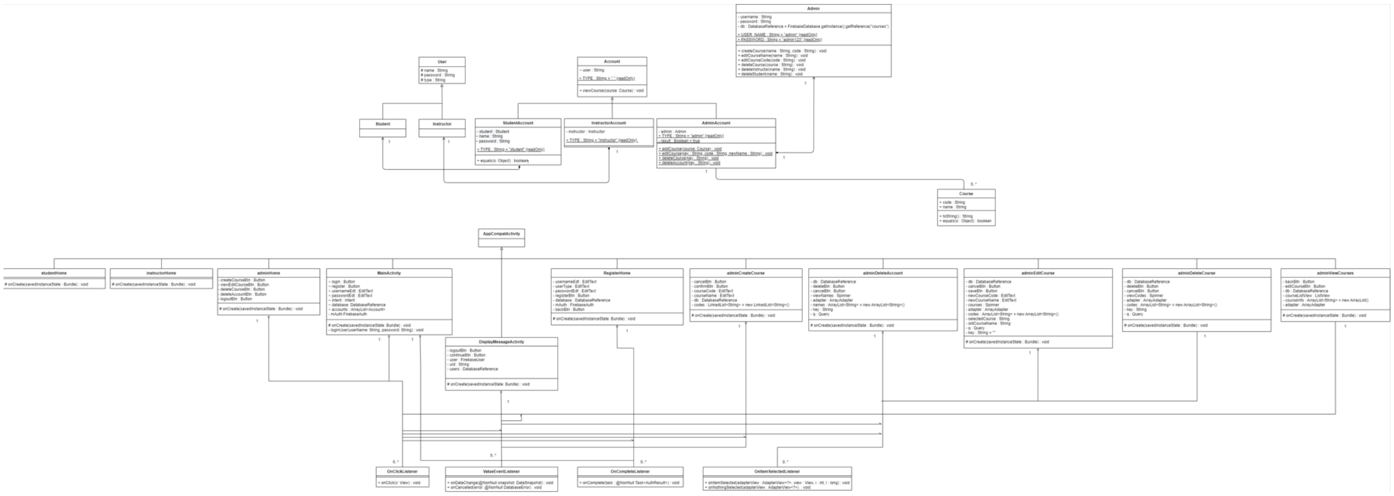


Figure 12. *UML Class Diagram from Deliverable 1.*

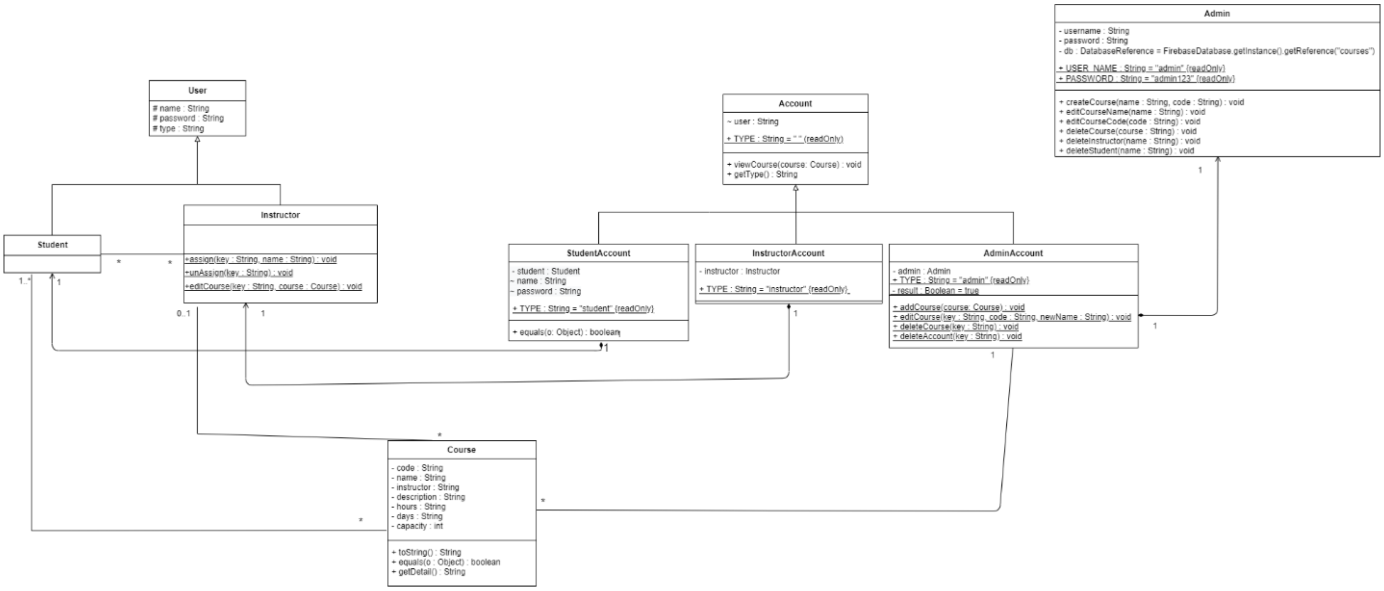

Admin
- username : String
- password : String
- db : DatabaseReference = FirebaseDatabase.getInstance().getReference("courses")
+ USER_NAME : String = "admin" {readOnly}
+ PASSWORD : String = "admin123" {readOnly}
+ createCourse(name : String, code : String) : void
+ editCourseName(name : String) : void
+ editCourseCode(code : String) : void
+ deleteCourse(course : String) : void
+ deleteInstructor(name : String) : void
+ deleteStudent(name : String) : void
User
# name : String
# password : String
# type : String
Account
~ user : String
+ TYPE : String = " " (readOnly)
+ viewCourse(course: Course) : void
+ getType() : String
Student
Instructor
+assign(key : String, name : String) : void
+unAssign(key : String) : void
+editCourse(key : String, course : Course) : void
StudentAccount
- student : Student
~ name : String
~ password : String
+ TYPE : String = "student" {readOnly}
+ equals(o: Object) : boolean
InstructorAccount
- instructor : Instructor
+ TYPE : String = "instructor" {readOnly}
AdminAccount
- admin : Admin
+ TYPE : String = "admin" {readOnly}
- result : Boolean = true
+ addCourse(course: Course) : void
+ editCourse(key : String, code : String, newName : String) : void
+ deleteCourse(key : String) : void
+ deleteAccount(key : String) : void
Course
- code : String
- name : String
- instructor : String
- description : String
- hours : String
- days : String
- capacity : int
+ toString() : String
+ equals(o : Object) : boolean
+ getDetail() : String
1
1
1
1
1
1
*
*
1..*
0..1
*
*
*

Figure 12. *UML Class Diagram from Deliverable 2.*

Appendix B: Project Log

-Keep track of project activities, progress, and task allocation here.

05/19/2022

- Document created by Salina; added deadlines, headings, and table of contents for easy navigation

05/20/2022

- Document updated by Fahmi; added a summary of user types and what each user should be capable of doing

05/22/2022

- Initial meeting
- Rough draft of UML diagram for Deliverable 1 presented by Henry
    - there was some uncertainty regarding whether or not separate classes for user accounts and the user themselves were necessary (seems redundant)
    - we'll probably be using draw.io to create the UML diagrams, but Lucidcharts could be a nice alternative if we want to collaborate on one (assuming draw.io has no direct way to collaborate)
    - UML should be finalized by next week (May 29th)
- Goal for the week established: learn Android Studio (at least, general features and basics)

05/27/2022

- created Github repository for project
- discussed roles for Deliverable 1 (**but not assigned yet**)
    - ROLES:
        - UML/Class Creation (1 person)
        - Welcome/Login Page (1 person)
        - Sign-up Page (1 person)
        - Main Page (will differ for each user type; admin features should all be complete) (2 people)
- buttons for navigating between activities created by Fahmi

06/04/2022

- all roles assigned and are as follows:
    - UML/Class Creation (1 person) - Fahmi
    - Welcome/Login Page (1 person) - Karl
    - Sign-up Page (1 person) - Henry
    - Main Page (will differ for each user type; admin features should all be complete) (2 people) - Jaewoon and Salina
- portion of deliverable 1 presented by Henry
    - login page: connected to sign-up page
    - sign-up page: can successfully register user, but may want dropdown menu for user type
    - firebase connectivity complete
    - no roles were assigned prior to this, so comments will need to be added to allow other group members to understand what needs to be done
- checklist for all deliverable 1 requirements added to document by Salina
- portion of deliverable 1 with additional comments has been merged to main branch by Henry

- <u>everyone must finish their part by June 9th at the latest to allow for final adjustments to be implemented and demo to be recorded on June 10th</u>

06/05/2022

- “Welcome screen and minor adjustments” merged to main branch by Salina; Summary of changes is as follows:
    - adjusted RegisterHome.java so that it could pass on username and role to DisplayMessageActivity.java
    - DisplayMessageActivity.xml and its corresponding java file are more or less complete, but login page (MainActivity.java) needs to pass on username and role to it
    - created 3 xml and java files for the homepages of the respective user types: studentHome, instructorHome, and adminHome
    - deleted register3.xml and its corresponding backend class; hypothetically, successfully logging in or registering should lead to the same welcome page
    - any comments made outside of the welcome screen portion start with "EDIT:"

06/10/2022

- "adminfunctions1" branch created by Salina, with the following changes (not merged to main yet):
    - Admin functionalities Create Course and View/Edit courses are complete with error handling (see Errors/Anomalies Checked)
    - The corresponding AdminAccount methods for the functionalities above have been changed to static, as an AdminAccount object isn't really needed to perform the tasks (can be changed back to non-static if need be); parameters were also changed based on what was needed
    - Any AdminAccount methods that were related to the two functionalities but were not used were deleted.
    - Added default constructor to Course.java to allow DatabaseReference method getValue to work
    - Added a drawable so that the ListView in View Courses had a border.
- Errors/Anomalies Checked:
    - Create Course: no fields are empty (text is trimmed, so no strings with empty spaces either) and course code does not already exist
    - View Course: cannot enter Edit Course activity if there are no courses to edit, and if there are no courses to view, a popup notif (Toast) will say as much
    - Edit Course: if both new course code and new course name are empty, popup will prompt user to add something; if only one field is left empty, that field will not be changed; new course code cannot be one that already exists

06/11/2022

- Expected date of completion was discussed.
    - The expected completion date of deliverable 1 is 06/12/2022 (tomorrow)
- Questions regarding UML were asked
    - “How do we integrate the java files that connect the front end to the back end in the class diagram?”
    - “Is there a need to include the files that connect the back end to the front end into the class diagram?”
- Login page sort of implemented by Henry; uses email authentication but pushed to main anyways

06/12/2022

- Discussed Jaewoon’s issues with deletion from Firebase database

    - Delete course deleted a random course from the database; Salina will atttempt to fix it
- 4 main buttons added to adminHome, pushed to repo by Jaewoon
    - needs to implement backend
- Discussed merging all the branches with the master branch.
    - merged “adminfunctions1” and resolved conflicts
- Tested out the app and checked which requirements had been met so far
- TO DO TODAY:
    - Henry and Karl: figure out how to implement login without using email
    - Henry: implement delete account
    - Salina: add dropdown menu to register page (DONE), fix adminHome (DONE), implement delete course (DONE)
- new branch called “register2” merged to main by Salina:
    - merge earlier today messed up MainActivity.java a bit, but it should be fixed
    - RegisterHome is now fully functional (doesn’t use email), with a dropdown menu and the ability to check if the user exists in the database
    - because DisplayMessageActivity no longer works with RegisterHome, it now returns the user to the login page; ensure that MainActivity passes username and role to DisplayMessageActivity
    - adminHome layout fixed and backend implemented
    - adminDeleteCourse and adminDeleteAccount activities have been added (but not implemented)
- delete course implementation pushed to main by Salina

06/18/2022

- Discussed what roles there would be for Deliverable 2.
    - UML Class (Jaewoon and Karl)
    - Unit Testing (Jaewoon and Karl)
    - Viewing/Searching courses (includes second page/screen for displaying info) (Fahmi)
    - Assign/Unassign Instructors (Henry)
    - Add Edit Course Information (Salina)
- Delegated roles to each member of the team.
- Discussed smaller due dates for various stages of deliverable 2.
    - Must be done with the “bare-bones” functionality of deliverable 2 by the 25th.
    - Must be ready to record by the 30th of June
- InstructorHome and Assign/Unassign functionality created and pushed by Henry

06/20/2022

- Errors such as extra constructors and unimplemented spinner fixed by Salina

06/21/2022

- View/Search courses functionality completed and pushed by Fahmi

06/24/2022

- Add/Edit Course Information functionality completed and pushed by Salina

06/25/2022

- changes to be made:
    - remove continue and log out button from DisplayMessageActivity
    - fix layout of Search Course (add back button)
    - remove View Courses from InstructorHome (maybe rename Search Courses?)
    - (tentative) create error message for Assign/Unassign that tells the user when there are no courses to assign/unassign themselves to
    - (tentative) connectivity of the functionalities (awaiting TA feedback)

07/03/2022

- might need to fix Search Course layout again
- Unit test cases will most likely need to write into the database (e.g. addCourse() for AdminAccount.java), create a Course/User object with the same attributes (expected), and then read the database (actual); if expected.equals(actual), unit test case is successful (might need to implement .equals() in Course.java and/or the user classes, depending on what you choose to do unit test cases on).
- Expected completion date: all files, including unit test cases and class diagram, must be complete and pushed to Github by Tuesday (demo will be recorded on Thursday, if not earlier)

07/16/2022

- Roles:
  - Final Report and UML: Karl
  - JUnit: Jaewoon
  - Searching Student: Fahmi
  - Enrol and Unenrol and Time Conflict Checking: Henry
  - View Courses/Students: Salina
- Deadline for barebones functionalities: **July 20th (Wednesday)**

Expected date for recording: **~~July 23rd/24th (Saturday/Sunday)~~ July 20th (Wednesday)**