Software Design Document

Version 1.0

26 November 2021

QResent

Attendance list generator based on QR codes.

Software Management Team 3

Faculty of Automatic Control and Computers, University POLITEHNICA of Bucharest.

1. Project purpose

This document is intended to accurately and completely describe the solution designed for the attendance list software system named QResent. The document serves as a unique solution-building guide for the project development team.

2. Document Contents

The document consists of four essential sections:

- Data model - presents the main data structures used, as well as the schema of the database data
- Architectural model - presents the architectural templates used, the system architecture and describes the components of the architecture
- User interface model - shows the user interface and the flow between windows
- Testing infrastructure - present all the tests inside the suite

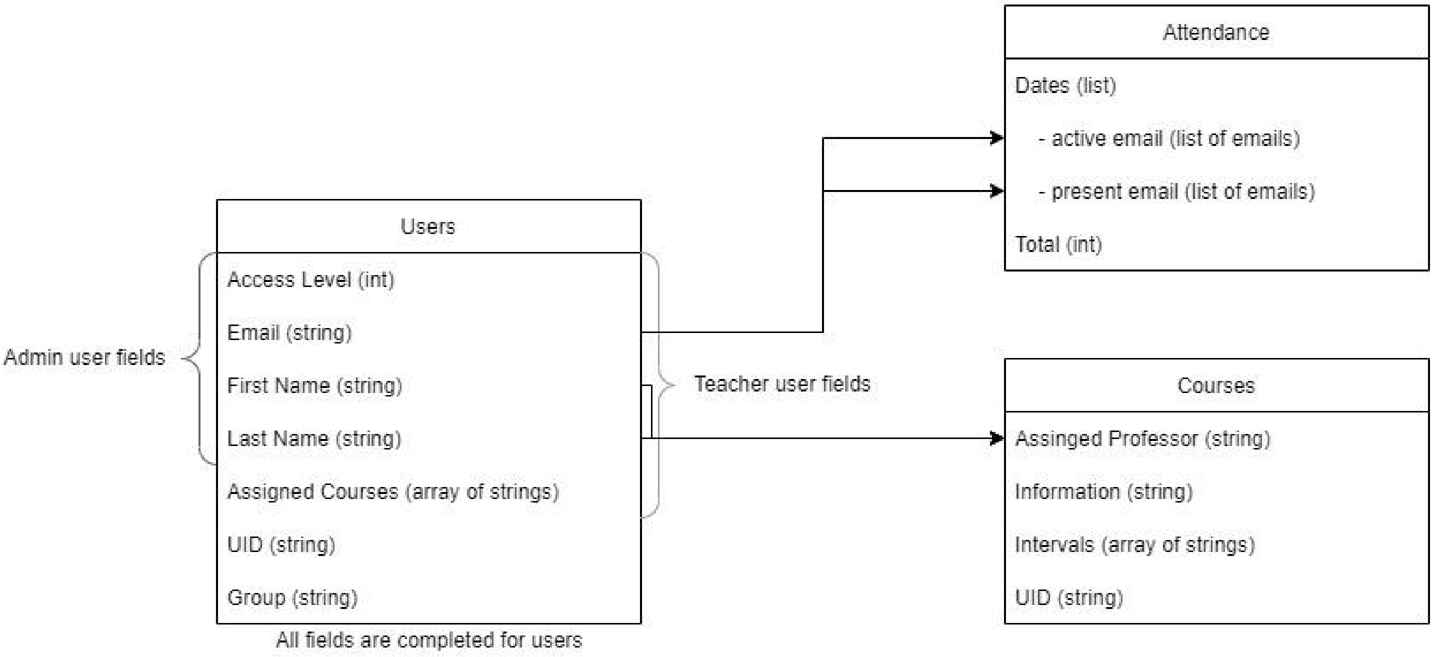
3. Data model

a. Temporary data structures

b. Data file format

c. Database structure

i. Database diagram



ii. Tables

1. Users

The main data table, storing all the users. The access level field is an integer type and differentiates the users between them. The three different types are students (level: 0), teachers (level: 1), and administrators (level: 2). Data stored for all users is the first and last name and email. The assigned courses field is a list of course names and is stored for teachers and students. The student also has a group assigned as a string.
The main key for the data table is the UID (unique ID).

2. Courses

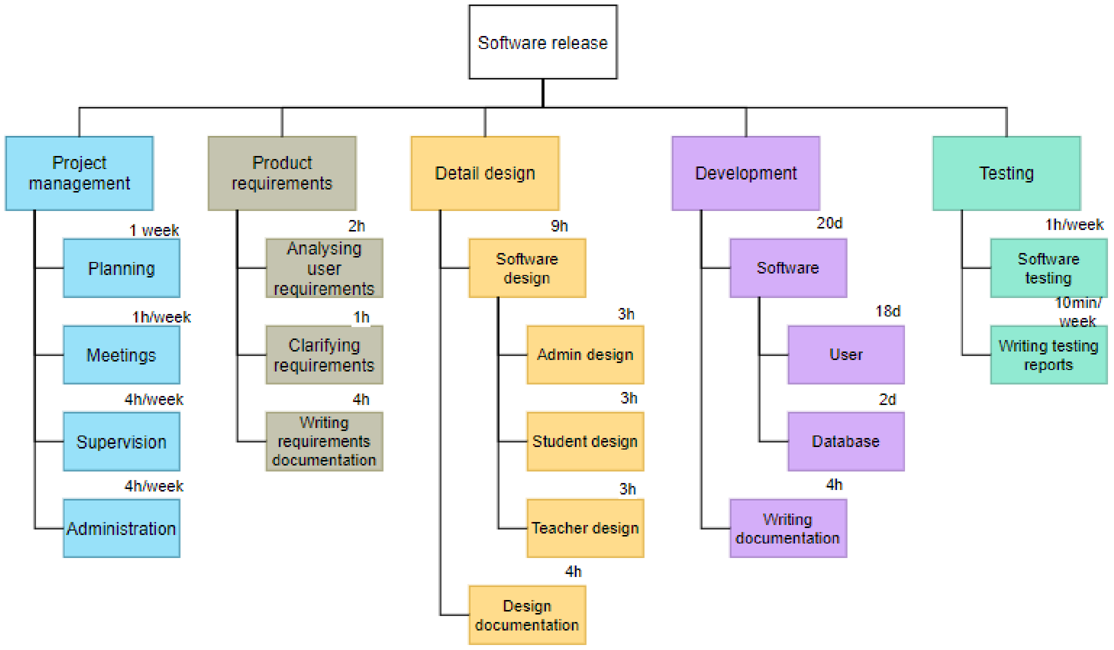
The courses table has the association between the course details and their intervals and the assigned teacher. The data stored in this table is only available to the related teacher and the assigned students, and it can be modified by the teacher.

3. Attendance

The attendance table holds information for each interval of each course. The main data structure is a map that associates each date to two lists, one of the active students and one of the present students. These lists are created when a new QR code is generated and are filled when a student scans the code. Also present in the table is the total number of students for each interval.

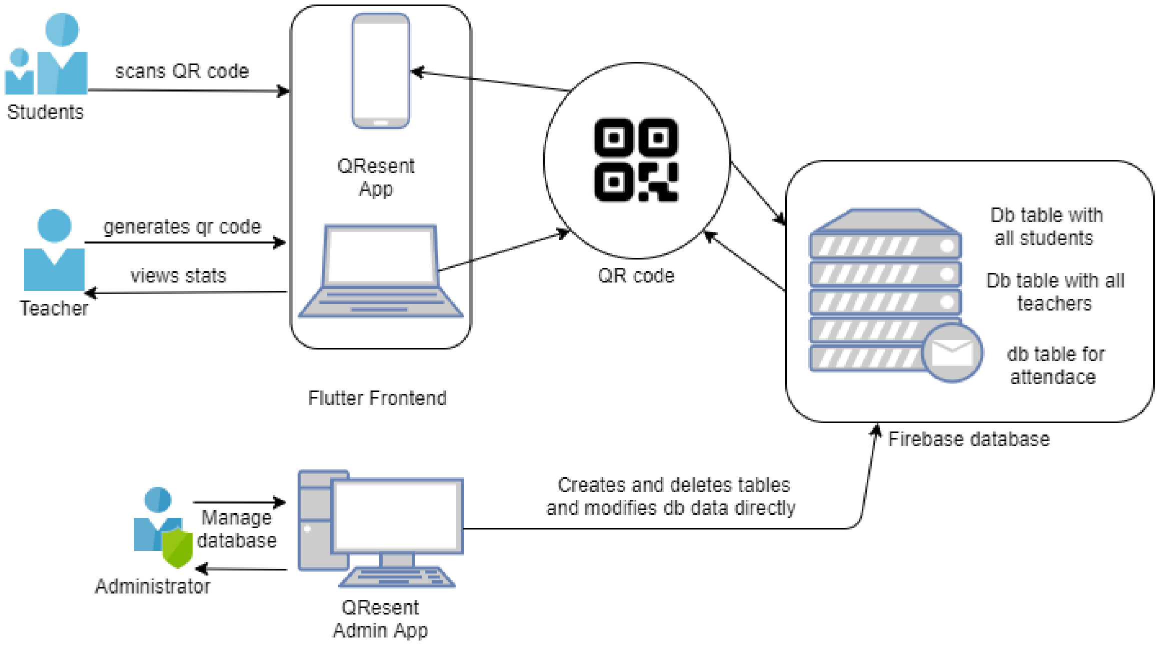
4. Architectural model

a. Architectural template



The project organization is split between the areas showcased in the diagram above. QResent is organized as a mobile app, it is realized using the Google tech stack. The mobile/web/desktop app communicates with the database through its user types.

b. Architectural diagram



The diagram below shows the design that has been chosen for the application, as well as the interaction between the components

c. Component Description

Software Components:

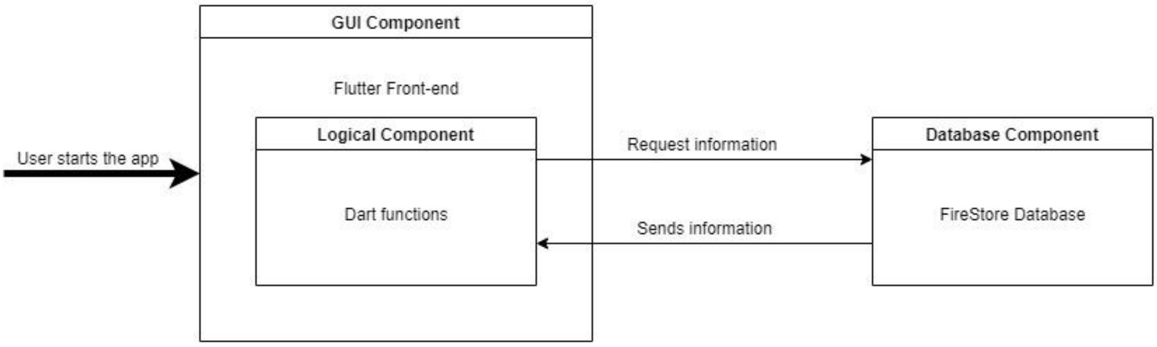
- **GUI Component -** The interface is realized with the flutter, and has a modern and intuitive look. It represents all the elements drawn on the screen.
- **Logical Component -** It is intertwined with the GUI component as they are both realized in dart. It represents all the functionality and actions taken by the app when the users interact with it.
- **Database Component -** It is responsible for the interactions with the database server and CRUD operations (Create, Update, Delete)**.**

d. Technological limitations

- The database that has been chosen is Firebase. It is a real-time (No)SQL database that provides both advantages and disadvantages. It is also limited in the case of large write operations.
- The front-end is developed in the flutter environment using Dart. This limits the ability to use vector graphics and animations and is also limited in application complexity. We are also limited in speed compared to native platform code.

These limitations do not impede the development of QResent

e. Component interaction

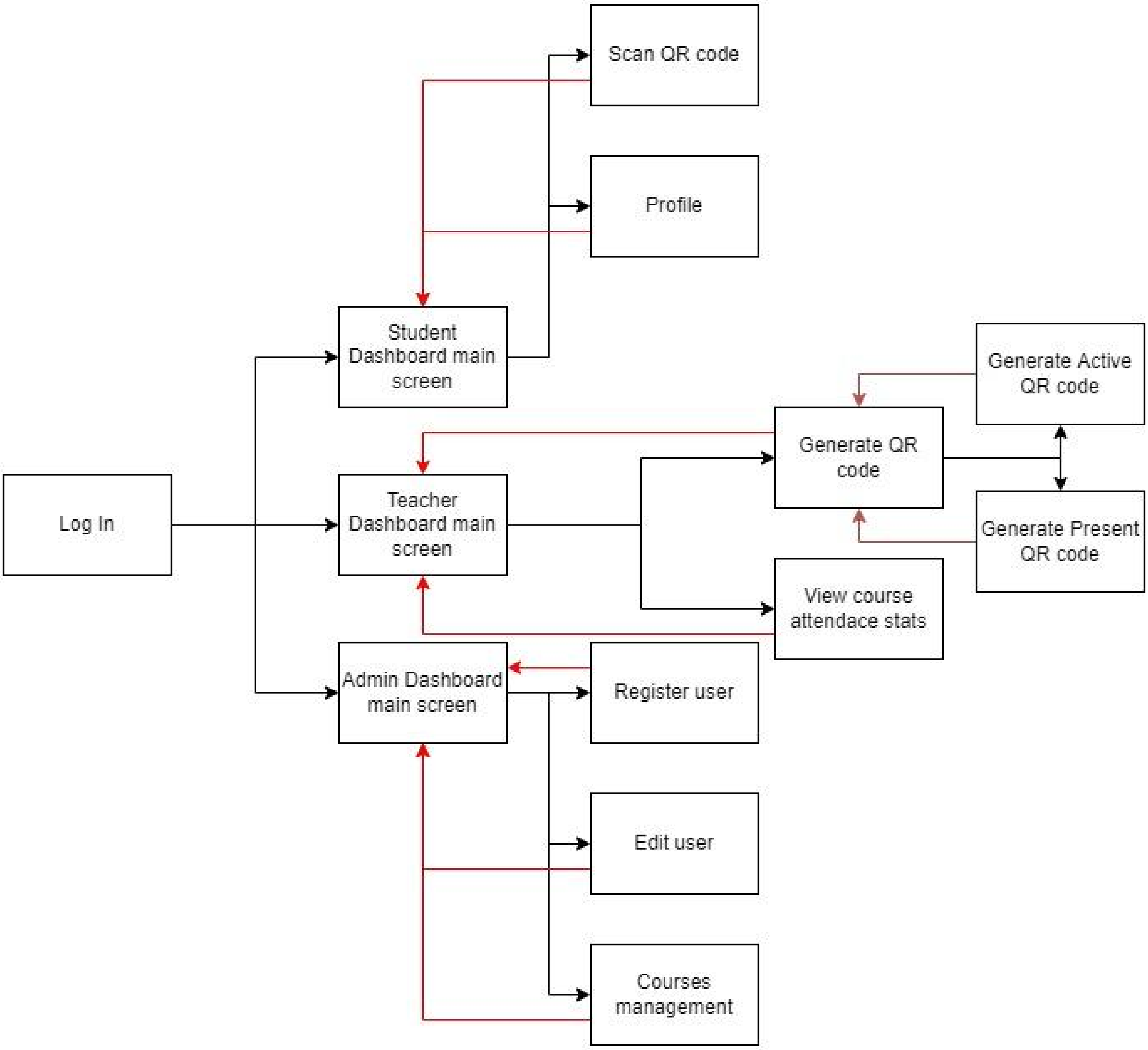


When the user starts the app he is greeted by the flutter front-end, behind it the dart code ensures the functionality and communication with the FireStore database. Firestore holds all the information about the users (email, name, assigned courses), information about the available courses, available intervals, and attendance.

5. User Interface model

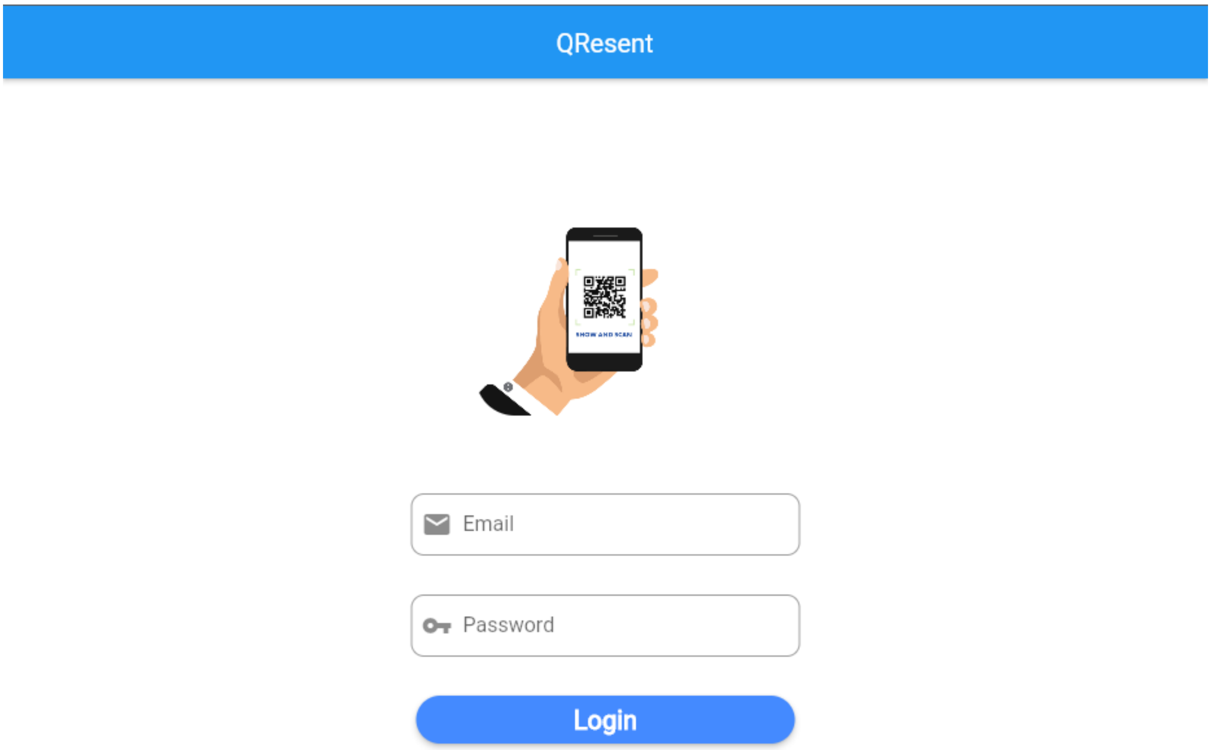
a. Interface succession

The above diagram showcases the interface flow inside the app for every kind of user. From every page, the user can log out and it's redirected to the first screen. The flow is realized through intuitively labeled buttons and icons. The red arrow signifies the back button redirection.



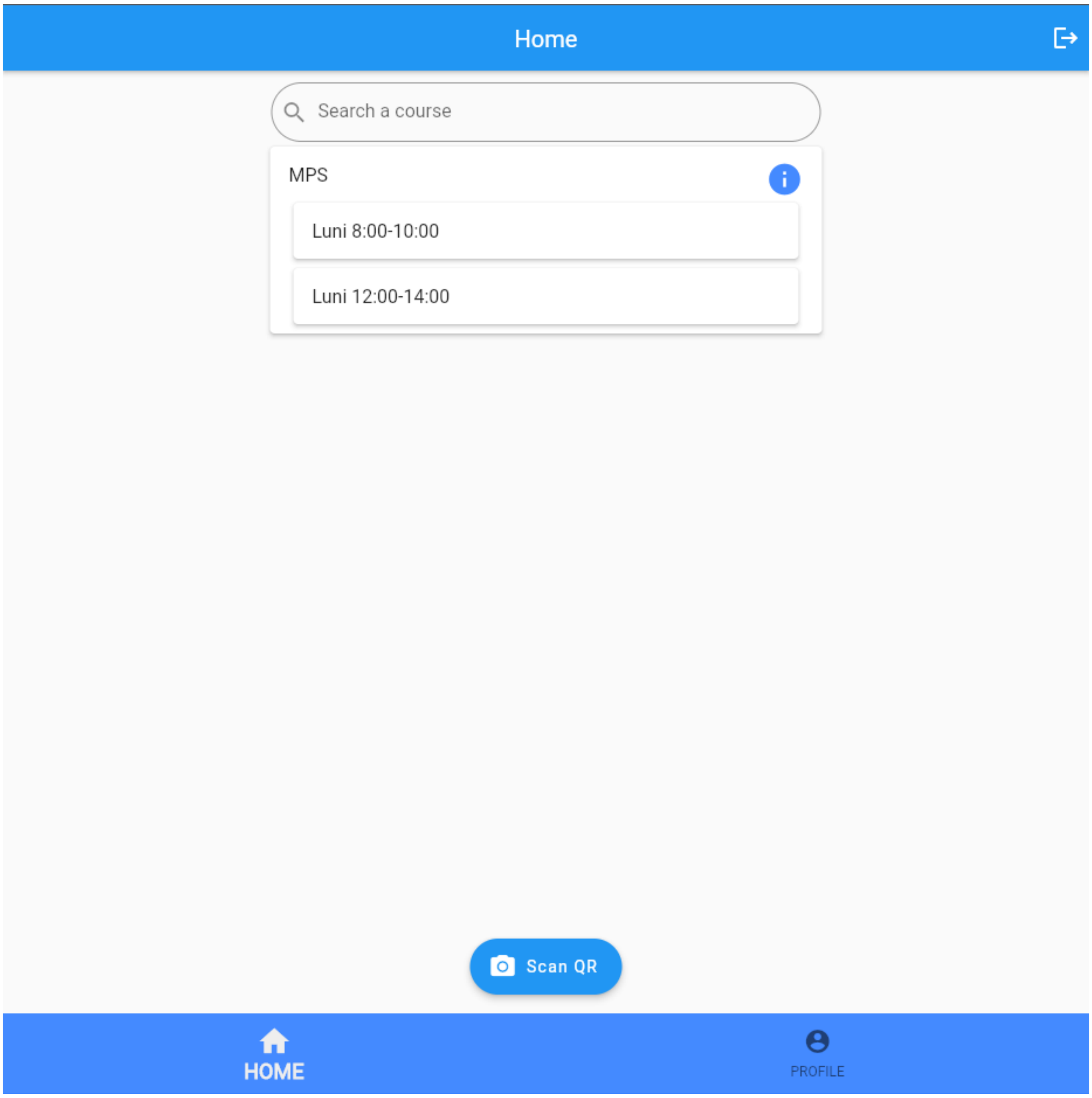
b. Application screens

i. Log In Screen



The login screen is the start of every interaction with QResent. The logo is custom designed for the app and the text fields check for valid emails and for empty values.

ii. Student Dashboard



Home

Search a course

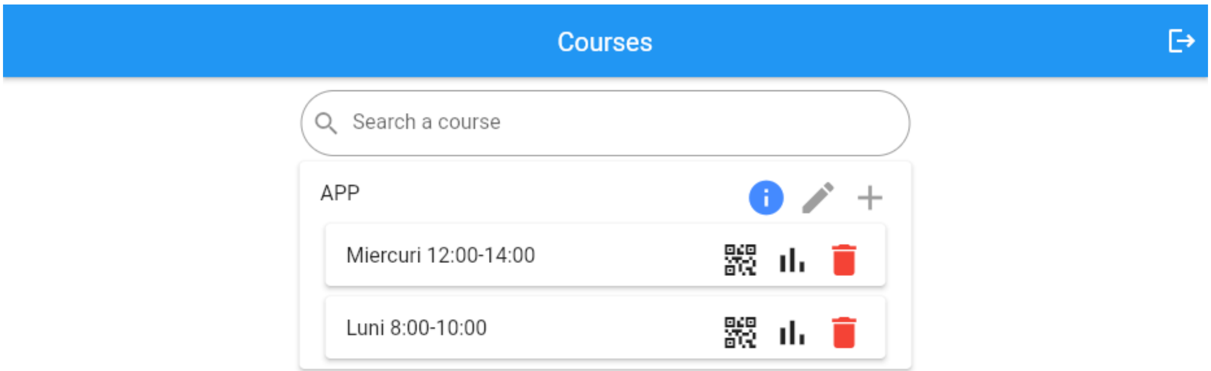
MPS

Luni 8:00-10:00

Luni 12:00-14:00

The student home screen displays all the courses the student has in his program, he can also view course information and scan a QR code for attendance.

iii. Teacher Dashboard

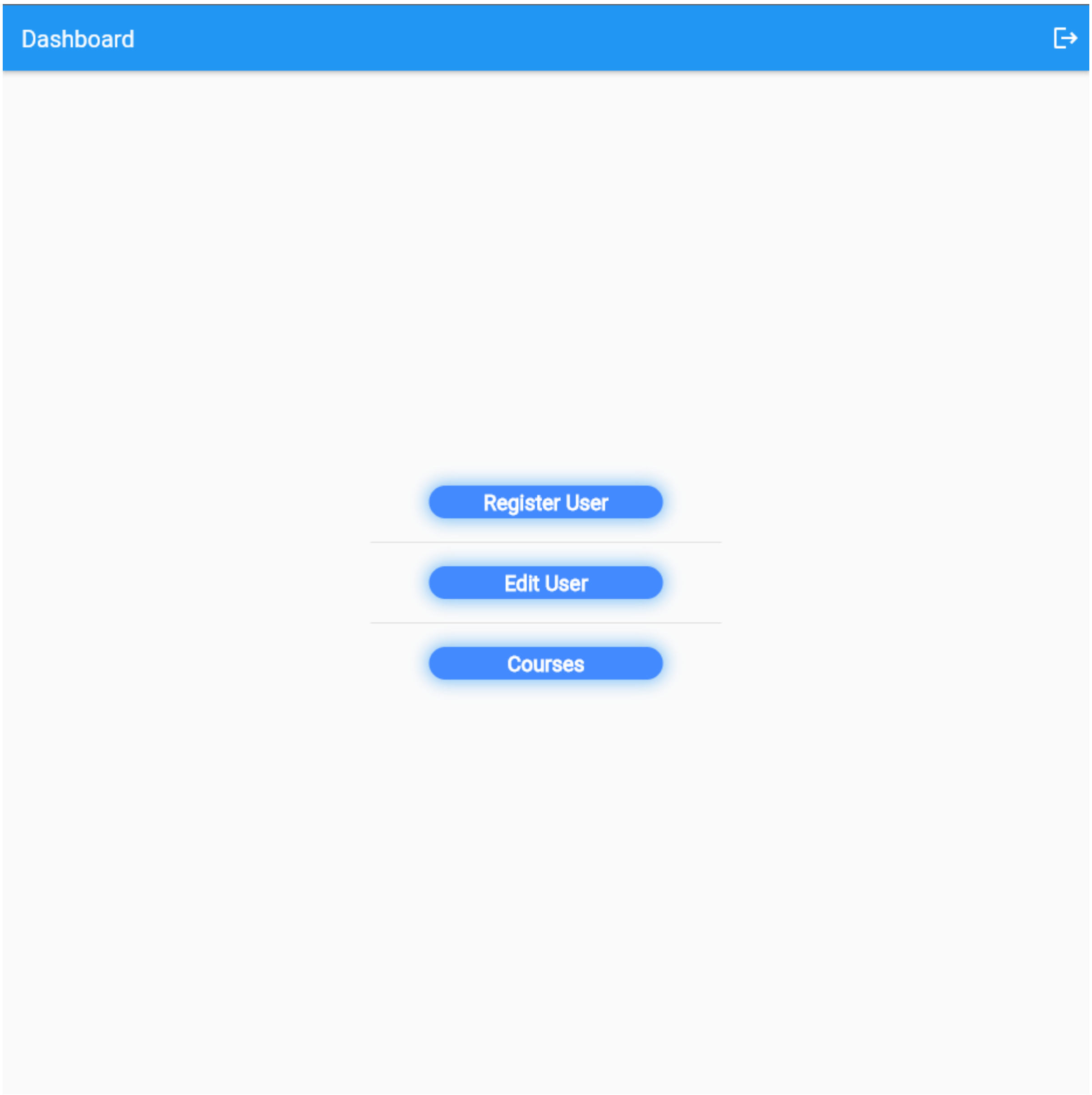


Miercuri 12:00-14:00

Luni 8:00-10:00

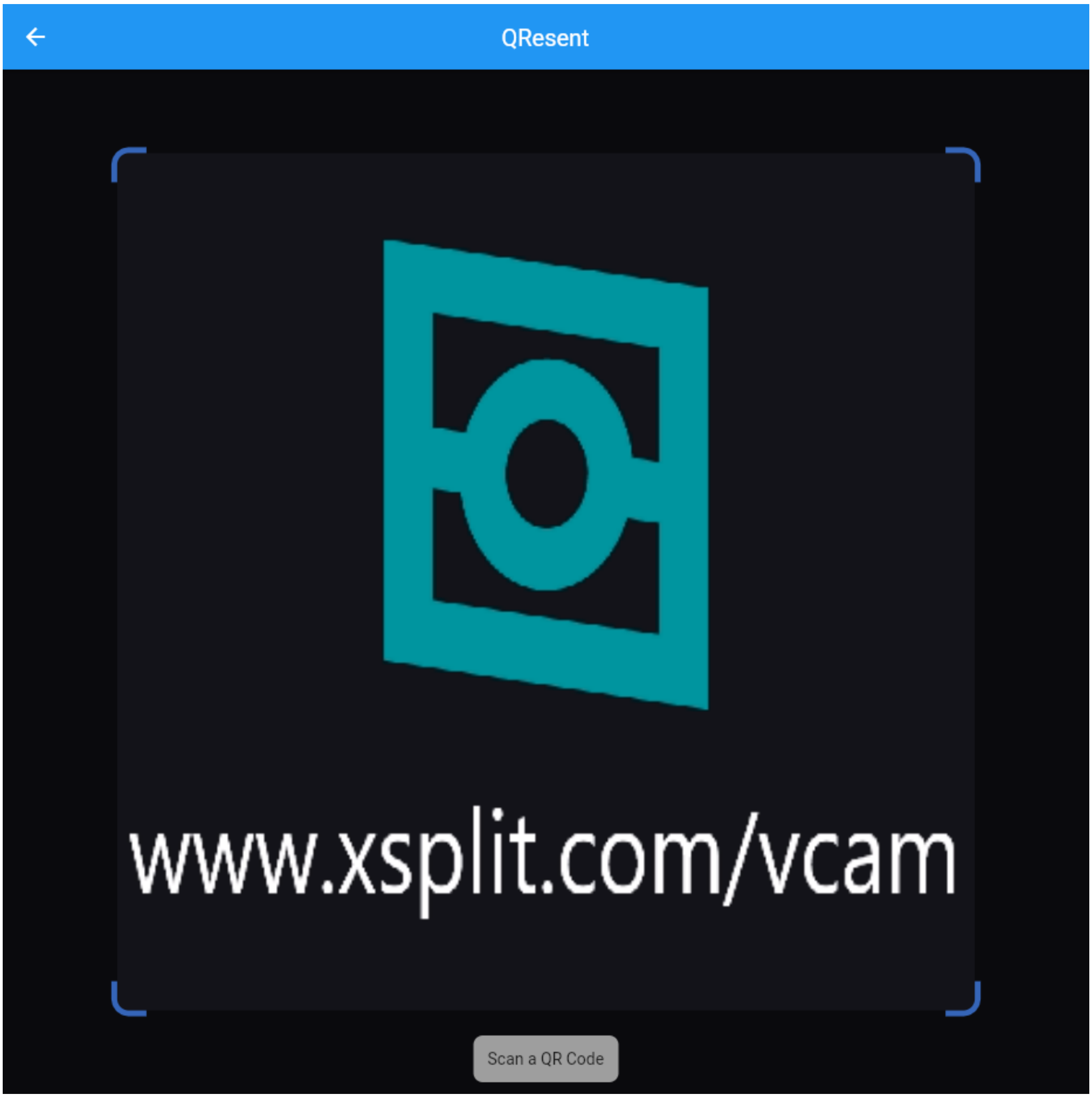
The main dashboard for the teacher. From here the teacher can manage all his courses. He can add new intervals to a course, change its information or just preview it. The teacher can also generate a QR code for an interval or look at the attendance statistics.

iv. Admin Dashboard



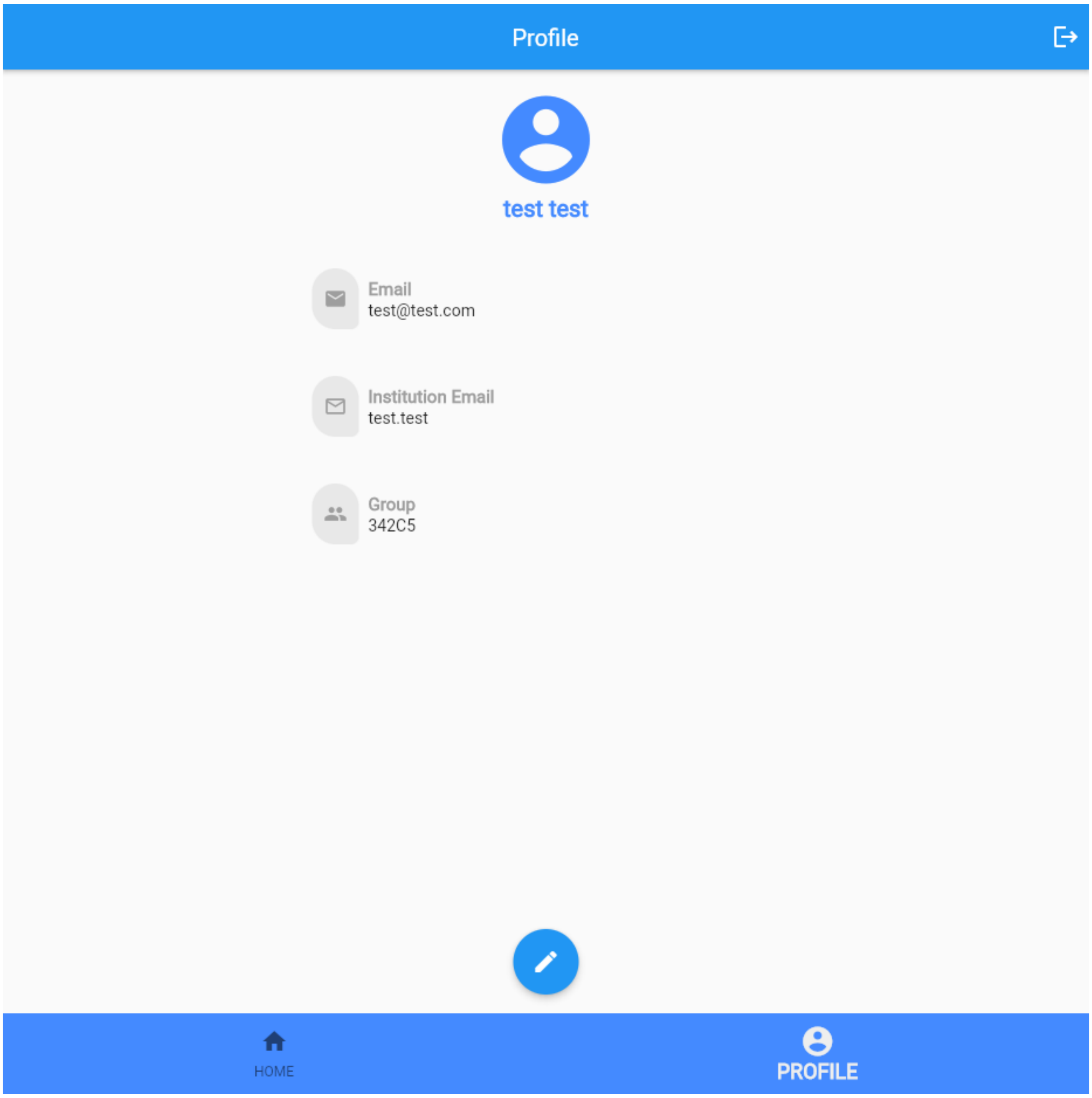
The main screen for the administrator acts as a control board for all the admin functions.

v. Scan QR code



This screen accesses the device camera and allows the student to scan the QR code and mark his presence or activity.

vi. Profile



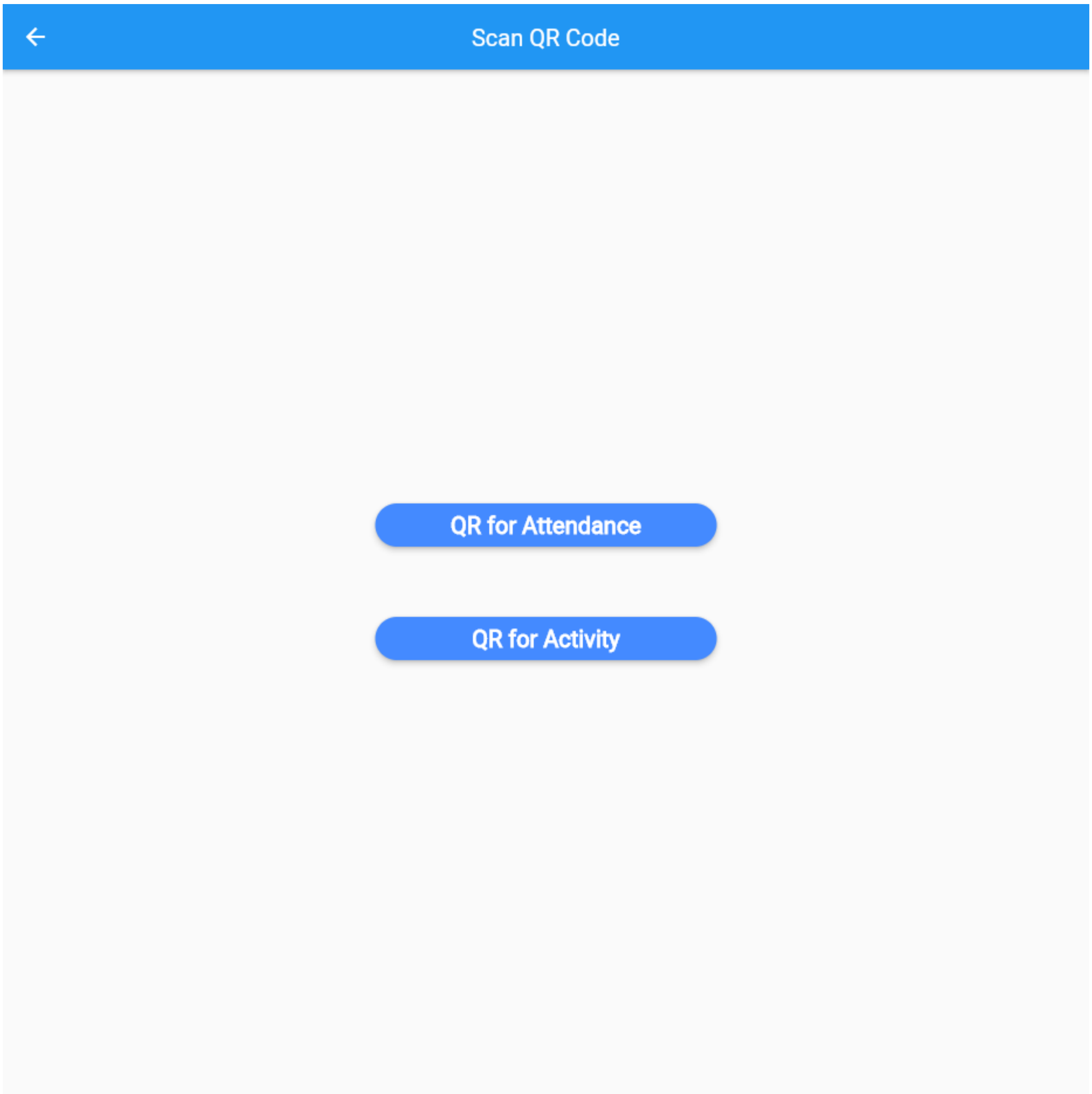
Email
test@test.com

Institution Email
test.test

Group
342C5

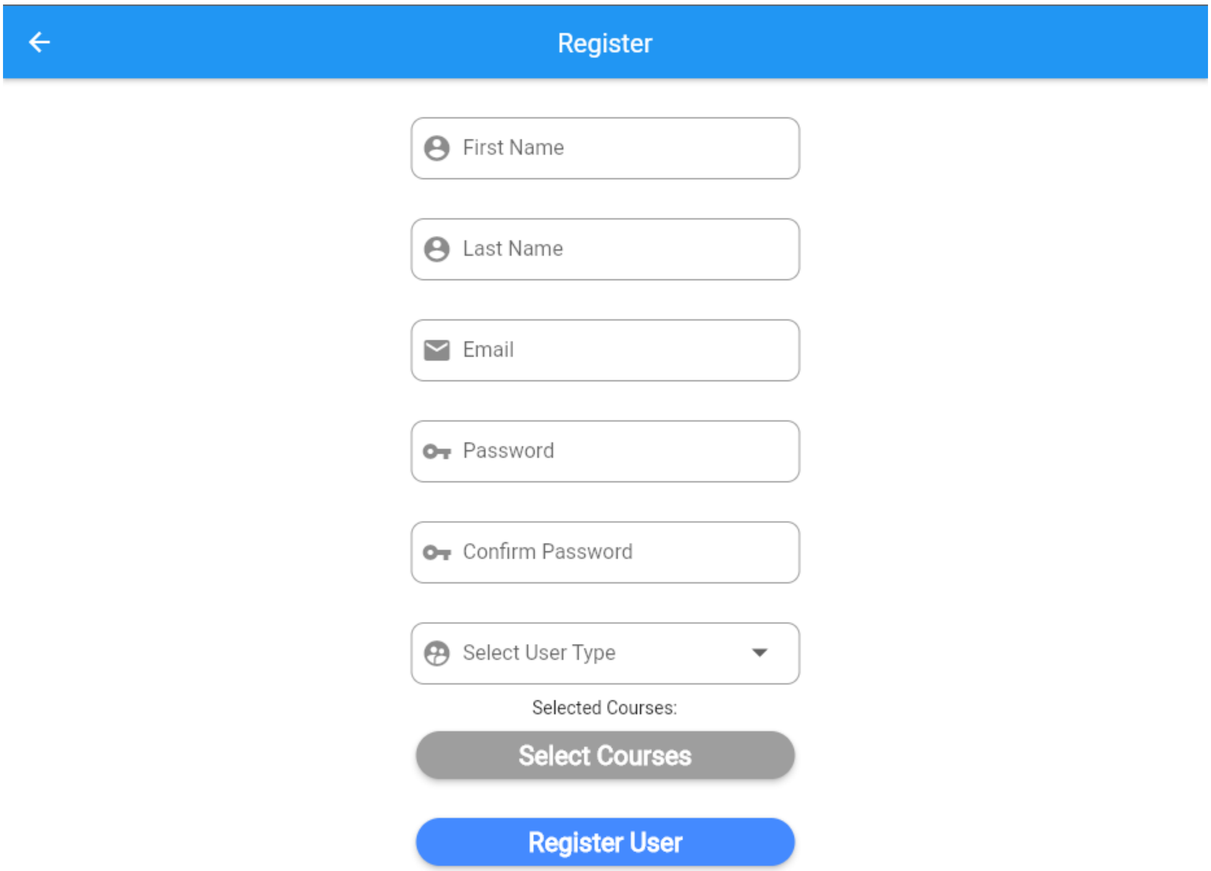
The profile tab shows all the relevant information for every student (email, LDAP user, and group), and it also lets the student modify his group in case he transfers.

vii. Generate QR code option menu



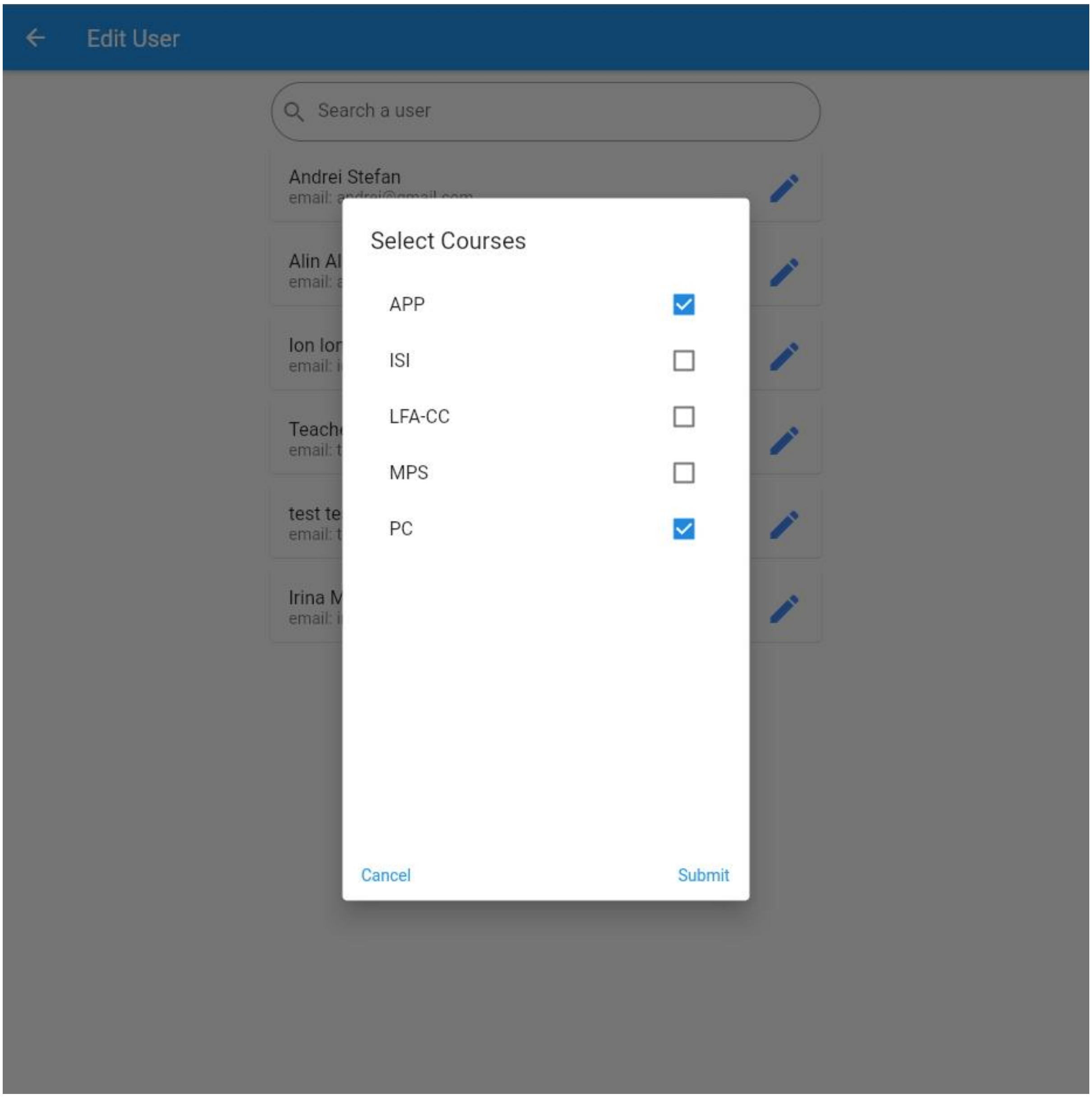
This screen gives the teacher the possibility to choose what type of QR code he wants to generate in order to keep track of his students.

viii. Register user



The administrator can register both students and teachers and he is responsible to enroll them in the courses. To add a user the administrator must set its name, email, and password.

ix. Edit user



Select Courses

APP ☑

ISI □

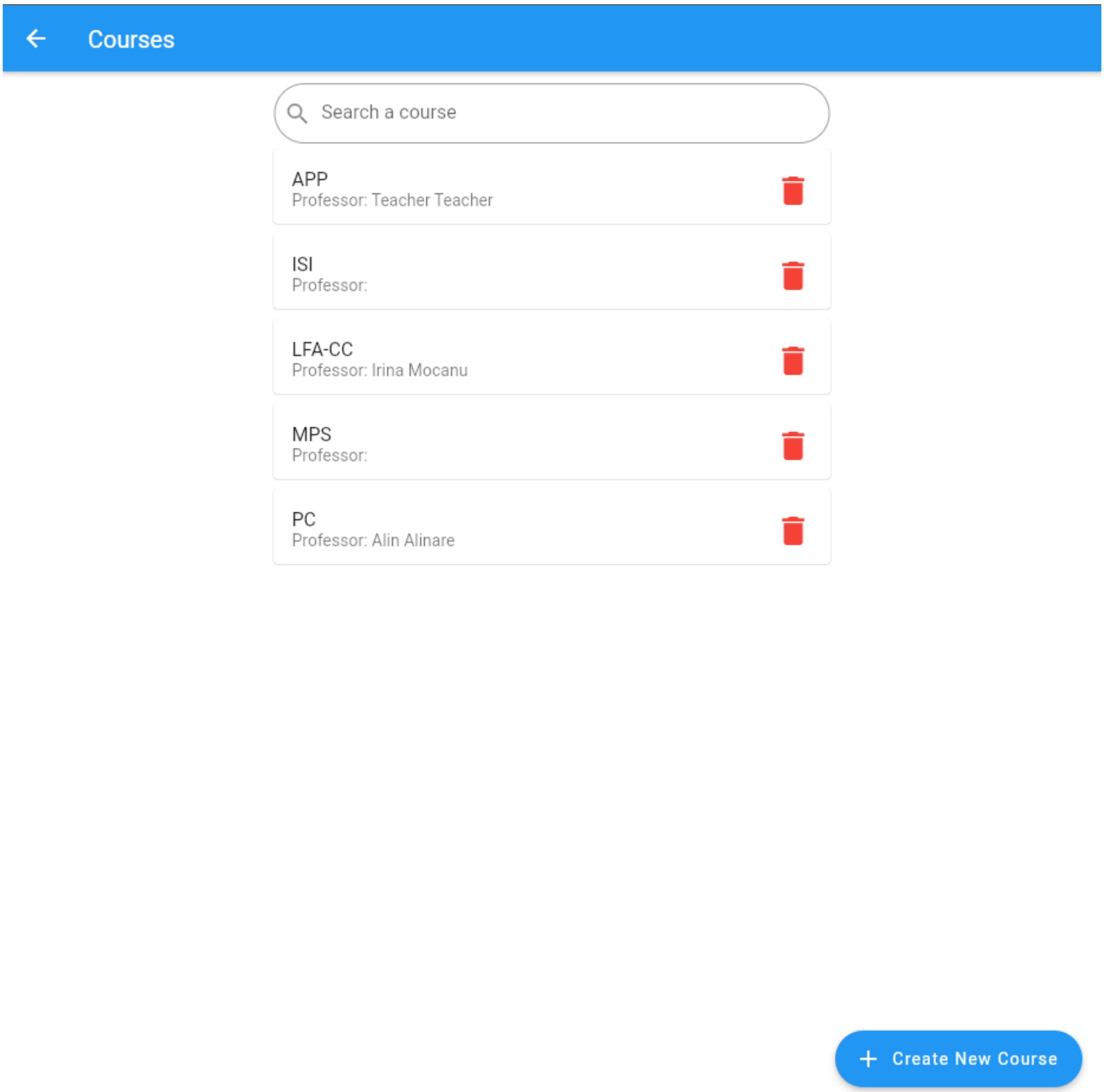
LFA-CC □

MPS □

PC ☑

The edit user screen shows all the students and teachers present in the database and allows the administrator to change the enrollment for each one. When pressing the edit button (the pen next to each name) the pop-up in the image will appear and the enrollment can be modified. The search bar causes live updates in the name list and can do matching anywhere in the “first name last name” string.

x. Courses management



APP
Professor: Teacher Teacher

ISI
Professor:

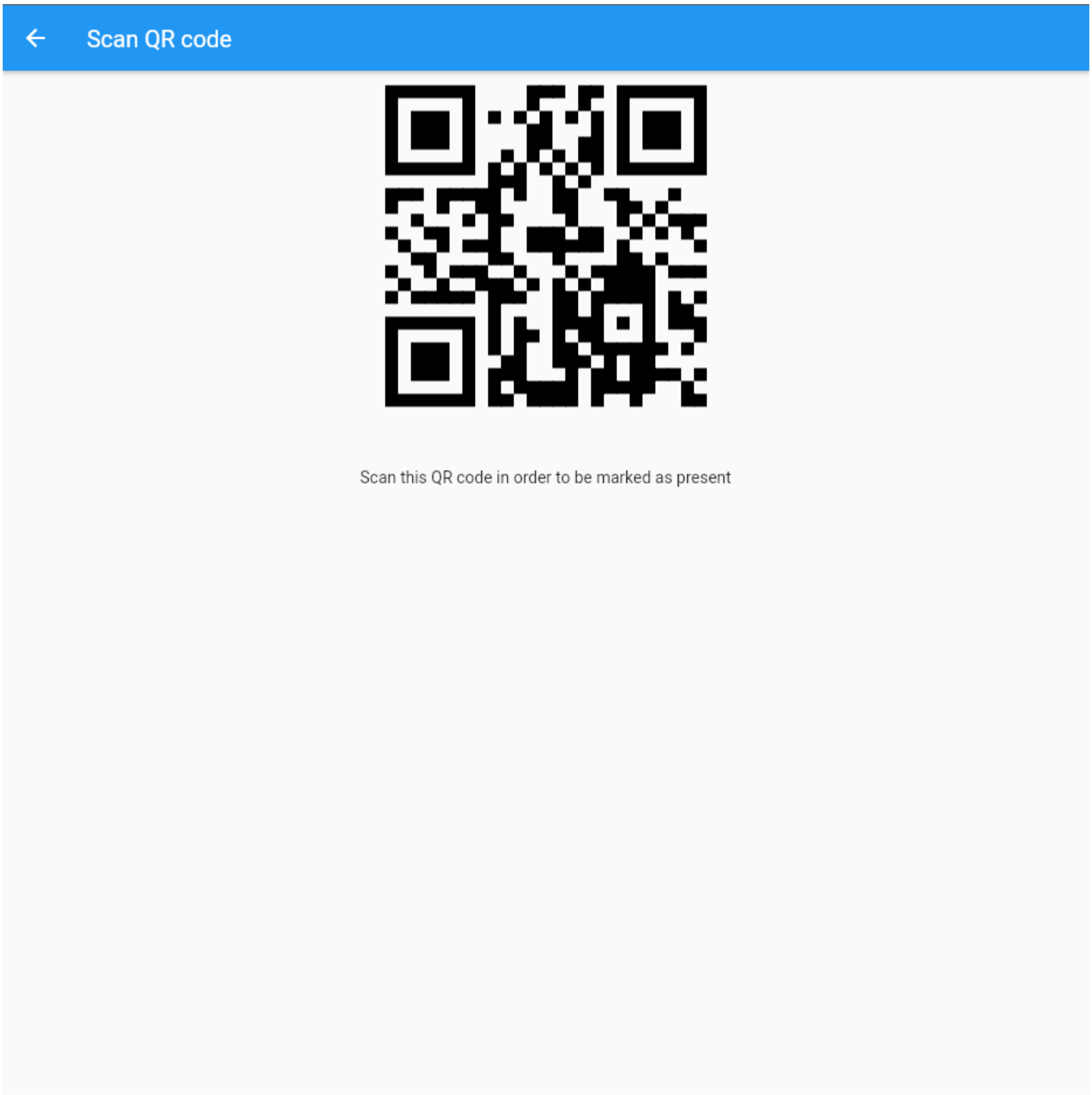
LFA-CC
Professor: Irina Mocanu

MPS
Professor:

PC
Professor: Alin Alinare

The course management page allows the administrator to create a new course. The course can be assigned to a teacher in the edit user screen, where the administrator can also enroll a student.

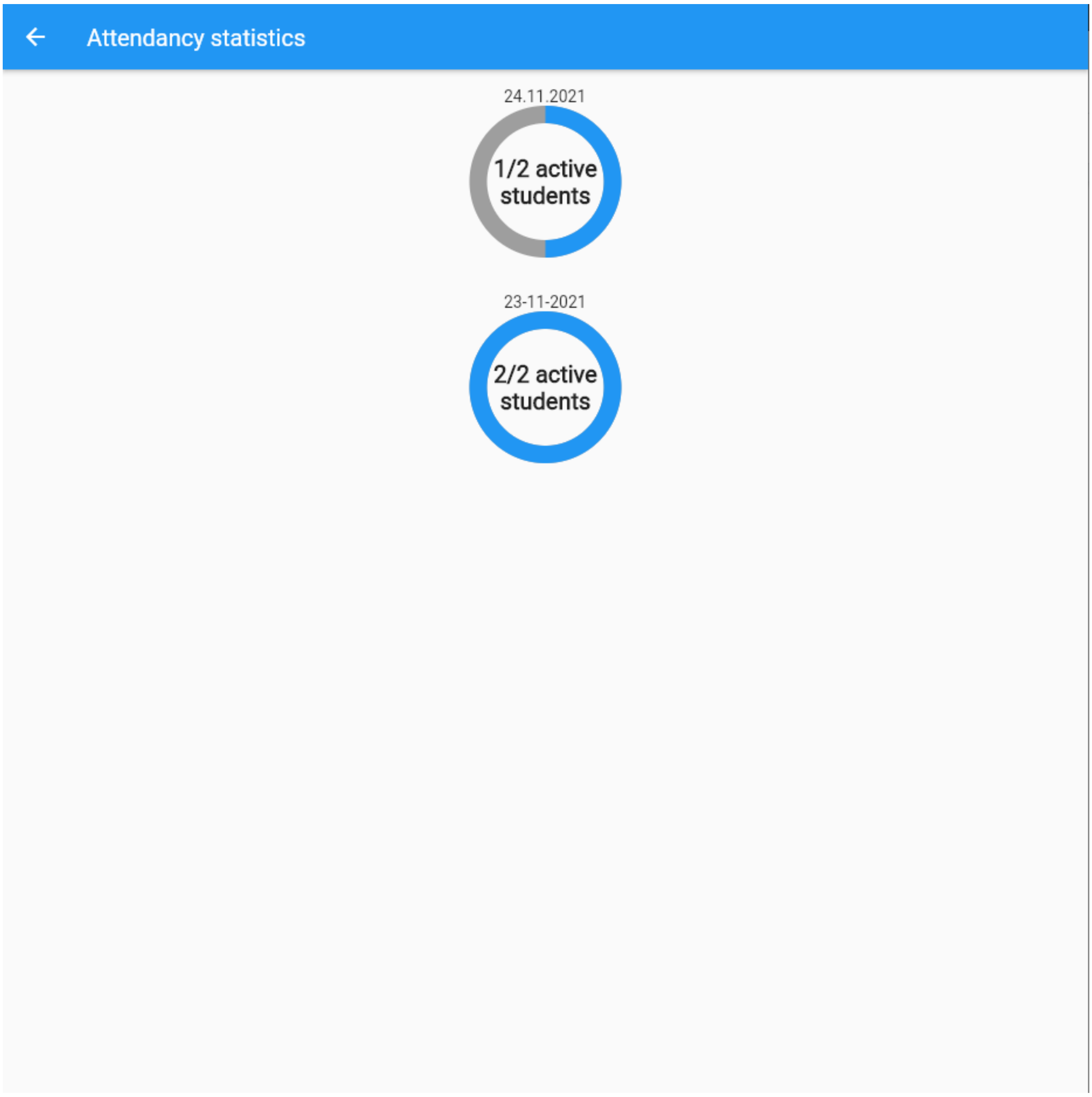
xi. Generate QR codes for active/present



Scan this QR code in order to be marked as present

The QR code display page can be displayed by the teacher on a projector or the share screen to be scanned by the students.

xii. View Course attendance stats



This screen shows the statistics for every interval for all the generated QR codes so far. It splits the students as present and active and displays a percentage graph.

6. Testing Environment

a. Manual tests

Every feature is manually tested by the developer before the pull request is created. This is the first step in the testing process.

b. Testing Infrastructure

The testing infrastructure is created using the dart testing capabilities and is deployed using GitHub actions. The testing suite is run on every pull request and every commit to the main branch. All the tests can also be run locally.

c. Test Suite

The test suite includes the following tests:

- Build
- Code format
- Code analysis (static analysis)
- Run flutter tests
- Application deploy
- Cleanup