EduSys

Combined-SRS-SDD-Test (CRDT) Document

squ4re

Ata Duru 1942002

Onat Büyükakkuş 2035772

Onur Adıgüzel 1941665

Enver Evci 1942085

Table of Contents

1. Introduction

This document consists of system requirements, system design and testing documents for a complete education system with complete statistics and analysis. In this document, firstly we are going to give the system requirements part that includes problem definition, brief information about project plan and system requirements and use case diagrams. Second part of the project is system design. This part contains model structure, source code structure, component and deployment diagrams. Finally, we will give information about testing. This section contains test plan and scenario, testing code structure and test results.

2. System Requirements

2.1. Problem Definition

Traditional attendance techniques are time consuming and insufficient. They are also not reliable. There is no processed statistics for students and instructors considering students' interest to certain courses and lecturers' teaching techniques . Students' advisors do not take enough care about students' possible career paths according to courses at which they are successful. There is also no available solution for proper and trustworthy feedback for administrators about instructors. During our senior project, we will be dealing with this problem.

EduSys is a web application that helps lecturers with taking attendance by face recognition through cameras in the classroom. It also gather statistics of students' interest to lecture by detecting where individuals sit in the class.

EduSys shows statistics it gathered in three different dashboards: Student, Lecturer and Administration. Student are able to see their interest and attendance to certain lectures, and they are able to enter examination information so that EduSys can generate future interest areas according to their success, attendance and interest to certain course combinations. Also, EduSys will notify students with their attendance percentage.

Lecturers are able to see daily, weekly, monthly and semesterly statistics of attendance and interest of student to their certain courses so that it will be easy for them to understand which teaching methods are effective throughout the semester.

Administration is given a proper feedback about lecturers and attendance to their courses rather than evaluation surveys completed by students.

2.2 System Requirements

2.2.1 Functional Requirements

2.2.1.1 IP Camera

- IP Camera continuously gathers information and forwards to server. It is provided that the device is enabled.
- IP Camera shall be connected to a network via Ethernet.
- IP Camera shall be connected to a power source with a 12 Volt supply.
- IP Camera shall be able to produce at least 30 frame per second in order to generate enough data for server to analyze.

2.2.1.2 Server Component (EduSys)

- Incoming data from IP Camera is forwarded to the server and then stored in the database.
- Server checks if IP Camera component is connected and working.
- Server checks whether the Internet connection is available or not.
- Server performs further analysis on data and generates statistics to be sent to web application.
- Users shall be able to log into the system with his/her username and password.
- Server recognizes the user by checking his/her username in the database and creates cookies knowing their roles.
- Students who are logged in to the system shall be able to view his/her profile information, update profile information, view attendance record, interest information, grades and possible future career path.
- Instructors who are logged in to the system shall be able to view profile information, update profile information, view attendance information, interest analysis of his/her course, create exam, update exam, take attendance, update attendance.
- Admins who are logged in to the system shall be able to view profile information, update profile information, view user list, create, delete and update users.

2.2.2 Non-Functional Requirements

Major requirements are separated into performance, safety, security requirements subsections. Other related requirements are explained in software quality subsection.

2.2.2.1 Performance Requirements

- Intensive data collection, transmission and analyze leads to concerns about performance of the system. Strict timing of the IP Camera shall be satisfied.

- In order to meet performance targets; response time, workload, scalability and platform issues are needed to be addressed precisely.

- System shall be able to operate without falling behind the speed of IP Camera's data flow in 99% of time.

- Since number of users are not estimated yet, storage and connection capacity of web server is not specified and needed to be scalable. More servers can be used and efficient service can be provided using reverse proxy server in the future.

- TCP and UDP protocols are considered for transmission between middleware device and web server. UDP is a faster protocol comparing to TCP but suffering from lack of reliability. Since reliability is a topic which cannot be compromised from, TCP is a valid choice. Performance requirements shall be satisfied using this protocol.

2.2.2.2 Safety Requirements

There are no safety requirements in this project since there will be no life or health threatening situation when some of the components fails.

2.2.2.3 Security Requirements

Data integrity is required to avoid misinformation. It shall be infeasible to break the system with the purpose of malicious activities. Security implementations shall be provided within the scope of privacy, integrity and authentication. Encryption schemes are ought to be employed due to these concerns. Educational data shall not be observable by anybody but authorized accounts in web application. Authorized users shall be identified by username and password. Legal issues and ethics about monitoring the classroom shall be complied and privacy of the students shall be respected. Exposure of images from the classroom needed to be prevented.

2.3. Project Plan

We periodically meet with Dr. Cevat Şener for consultancy. We roughly shared the work since our project has multiple components.

Despite this division of labor, all members of our group is following every update of other member's works and we are trying to maintain a balance so that everyone would contribute to every part of the project equally.

We need to finish our project until June 2017 and divide the work into two because the Computer Engineering Design course is divided into two parts for two different semester. We planned the semester works like:

First Semester:

Architectural design;

- Deciding which architectural design we will be following.
- Deciding which open source libraries, ready-to-use materials we will be using.
-

Installation of IP camera;

- Installation of camera software.
- Learning its software.
- Getting data from the camera and process this data frame by frame.
-

Detection and Recognition of faces of students;

- Finding proper algorithms to detect faces in frames coming from IP camera.
- Implementing algorithms.
- Generating a dataset of faces of students photographed from different angles with different light.
- Finding proper algorithms to train data of faces in frames coming from IP camera.
- Finding proper algorithms to detect resemblance queried face and trained data.
- Implementing those algorithms.

Second Semester:

Generating interest information about individuals;

- Generating data of where individuals sit in the classroom according to face recognition.
- Keeping data in server databases.

Generating attendance information about individuals;

- Generating data of if individuals are in the classroom according to face recognition in WP3.
- Keeping data in server databases.

Generating future interest areas;

- Defining a graph where combination of certain courses in our department leads to certain areas of interest with collaboration of academics in department.
- Defining an algorithm that includes data stored in item 3 and 4 and also students' examination information to generate future areas of interest defined by graph in this item.

Implementing student, lecturer and admin dashboards;

- Implementing authentication Sign Up and Log In features to dashboards.
- Dashboards will include following.

a) Daily, weekly, monthly and semesterly attendance statistics of students in certain courses with graphics.
b) Notification system that notifies student if they are going to fail from attendance with their current attendance ratio.
c) Visualization of the graph in WP6 and coloring it with statistics of students leading to certain areas such as Bioinformatics, Computer Vision etc.

In the second semester, we added some new and revised features to our project.

First lesson training;

It can be seen from “Detection and Recognition of faces of students” item above that we thought that we photographed every student in order to train them in the first place. We wanted to improve training and implemented first lesson training. Lecturer takes short video of students by clicking a button on the dashboard in the lecture and system detects faces, takes a couple of frame of them and puts them to different directories in the dataset.

Automated dataset organizer;

In the beginning, we have given all directory names in the dataset manually after training. In the revised version, we implemented that directory names started to be given after students logged in the system for the first time. System recognizes the face of the student, finds the directory of the student and changes the name of it according to student id.

Offline attendance;

This was our advisor Cevat Şener’s idea. When a classroom does not have a camera, the attendance can be taken by taking photographs from the smart phone of the lecturer. Lecturer upload one or more photograph of the students to the system and system takes the attendance.

2.4. Use Case Diagrams

Student

Student is a person who can view profile information, update profile information, view attendance record, view interest analysis, view exam results, view possible career path. More detailed information can be seen in the below diagrams.

Instructor

Instructor is a person who can view profile information, update profile information, view attendance information, interest analysis of his/her course, create exam, update

exam, take attendance, update attendance. More detailed information can be seen in the below diagrams.

Admin

Admin is a person who can view profile information, update profile information, view user list, create, delete and update users. More detailed information can be seen in the below diagrams.

2.4.1. Student Use Case

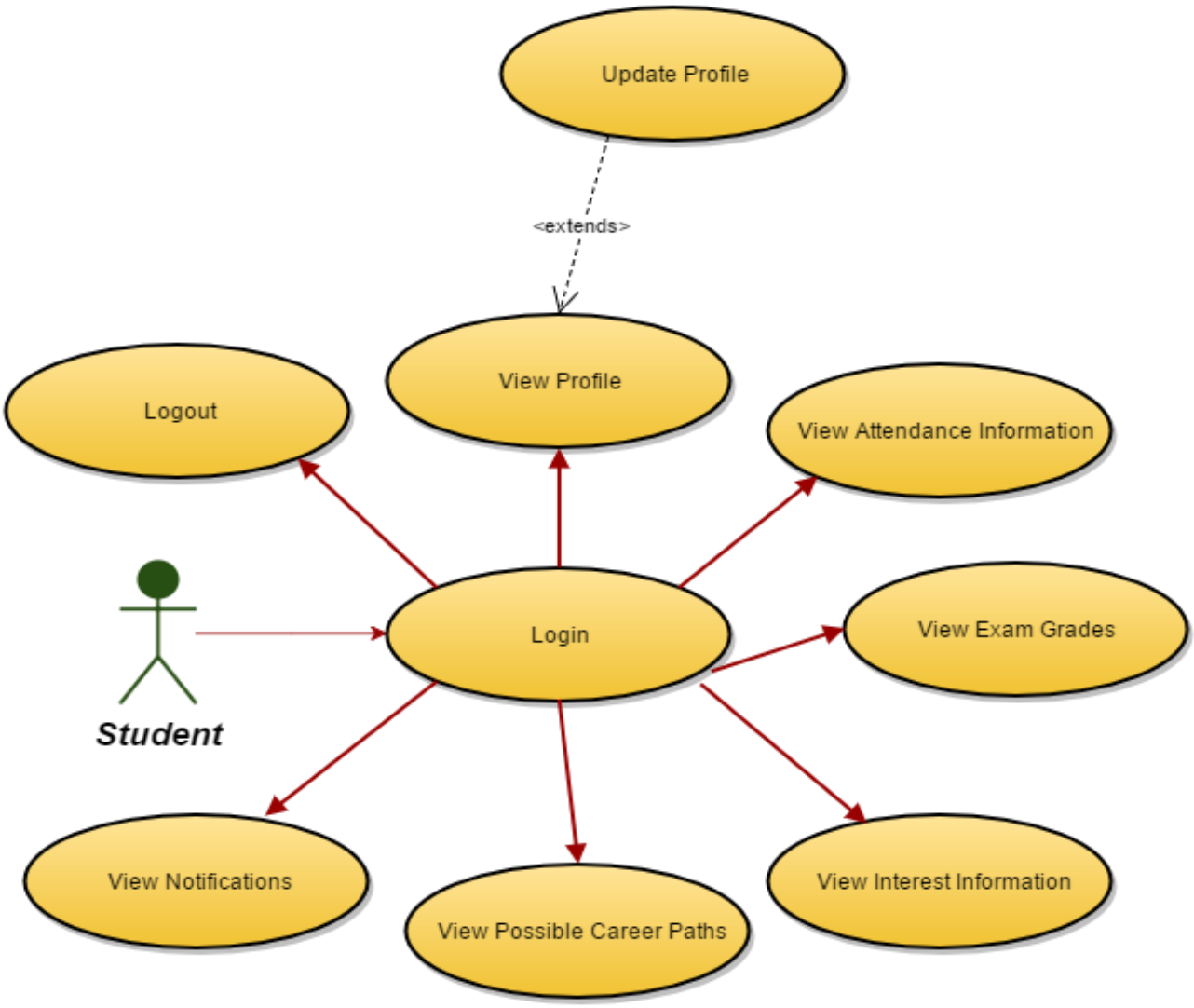


Figure 2 Student Use Case Diagram

Use Case	Description
Login	Student has to login in order to do functionalities below.
View Profile	Every registered student has a profile containing personal details.
Update Profile	Every registered student can update his/her profile containing personal details.
View Notifications	The student can view his/her warning notifications related to critical attendance or exam grades.
View Attendance Information	The student can view his/her attendance information for a specific course or all.
View Exam Grades	The student can view his/her exam grades.
View Interest Information	The student can view his/her interest information for each course or overall.
View Possible Career Paths	The student can view his/her possible career paths according to his/her most interested courses.
Logout	The student can logout to end his/her session.

2.4.2. Instructor Use Case

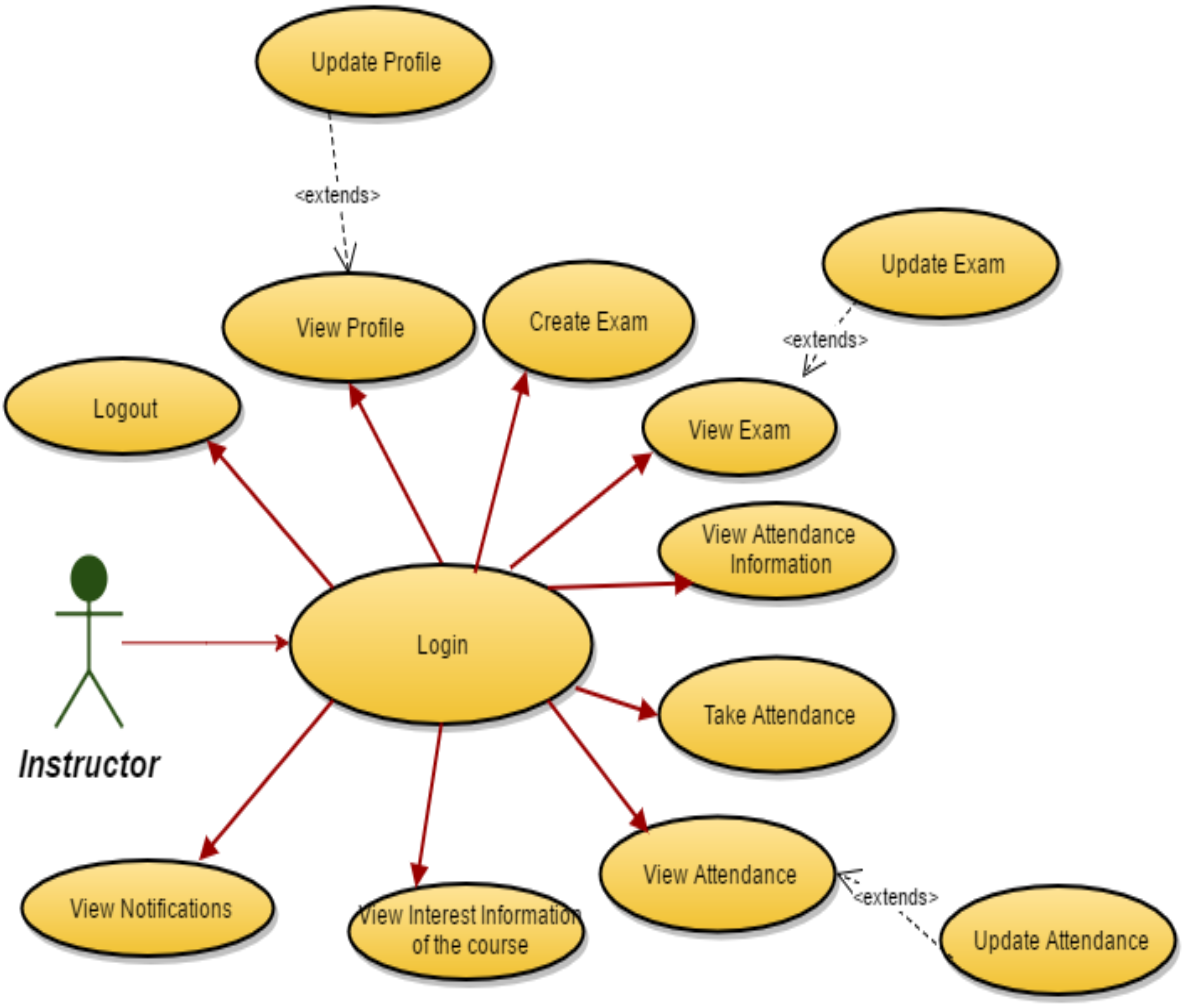


Figure 3 Instructor Use Case Diagram

Use Case	Description
LogIn	Instructor has to login in order to do functionalities below.
View Profile	Every registered instructor has a profile containing personal details.
Update Profile	Every registered instructor can update his/her profile containing personal details.
View Notification	The instructor can view his/her notifications related to critical attendance or exam grades.
Create Exam	The instructor can create an exam specifying the course and the date of the exam.
View Exam	The instructor can view the exam's information.
Update Exam	The instructor can update the information of selected exam.
View Attendance Information	The instructor can view which students attend for a specific lecture.
Take Attendance	The instructor can take attendance of the current lesson.
View Attendance	The instructor can view attendance information for a spesific course or all.
Update Attendance	The instructor can update the attendance information of selected lecture.
View Interest Information of the Course	The instructor can view interest information of students for a specific lecture.
Logout	The instructor can logout to end his/her session.

2.4.3. Admin Use Case

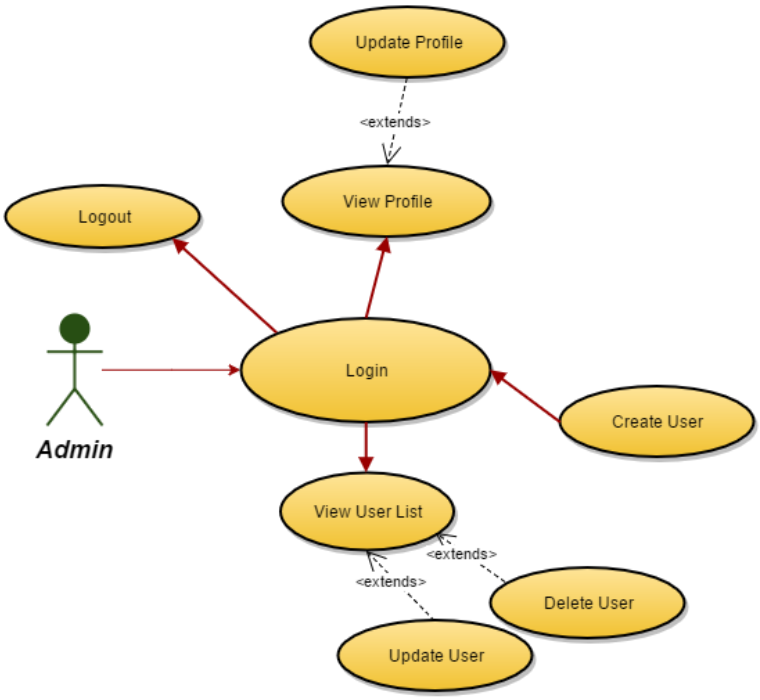


Figure 3 Instructor Use Case Diagram

Use Case	Description
LogIn	Admin has to login in order to do functionalities below.
View Profile	Admin has a profile containing personal details.
Update Profile	Admin can update his/her profile containing personal details.
Create User	Admin can create users specifying their username, roles, password etc.
View User List	Admin can view the user list.
Delete User	Admin can delete users.
Update User	Admin can update information of users.
Logout	Admin can logout to end his/her session.

3. System Design

3.1. Module Structure

Our System has 4 main modules. These main modules are database server, backend server, camera module and GUI module.

A database server is a computer program that provides database services to other computer programs or to computers, as defined by the client–server model. Since database server needs a management system, as database management system component, we use PostgreSQL. PostgreSQL is an object-relational database. Its primary functions are to store data securely and return that data in response to requests from other software applications. [1]

A backend server system is responsible for communication between database and web. It has three main components. First component is Apache Tomcat. Tomcat Server, is an open-source Java Servlet Container. Tomcat implements several Java EE specifications including Java Servlet, JavaServer Pages (JSP), Java EL, and WebSocket, and provides a "pure Java" HTTP web server environment in which Java code can run. [2] Our second component is rest server. Rest is the underlying architectural principle of the web. The system about the web is the fact that clients (browsers) and servers can interact in complex ways without the client knowing anything beforehand about the server and the resources it hosts. [3] The third main component of the backend server is that hibernate. Hibernate ORM (Hibernate in short) is an object-relational mapping tool for the Java programming language. It provides a framework for mapping an object-oriented domain model to a relational database. Hibernate solves object-relational impedance mismatch problems by replacing direct, persistent database accesses with high-level object handling functions. Hibernates primary feature is mapping from Java classes to database tables, and mapping from Java data types to SQL data types. Hibernate also provides data query and retrieval facilities. It generates SQL calls and relieves the developer from the manual handling and object conversion of the result set. Hibernate provides an SQL inspired language called Hibernate Query Language (HQL) that allows SQL-like queries to be written against Hibernate's data objects. Criteria Queries are provided as an object-oriented alternative to HQL. Criteria Query is used to modify the objects and provide the restriction for the objects. HQL (Hibernate Query Language) is the object-oriented version of SQL. It generates database independent queries so that there is no need to write database-specific queries. Without this capability, changing the database would require individual SQL queries to be changed as well, leading to maintenance issues. Under the lights of explanations above the rest server and Hibernate communication is provided by HQL. [4]

Camera module includes both hardware and software. The hardware component is either an IP Camera that is installed in class or any device that uploads photos to the system. When the system gets the images (these images can be either some photos or frames that is fetched from the video of IP Camera), software component of camera module processes the image by using OpenFace. OpenFace is a free and open source Python and Torch implementation of face

recognition library with deep neural networks. It uses dlib and OpenCV to detect and crop faces.[5]

Last main module of the project is GUI module. This mission coordination center makes communication between user and backend server over rest calls. These calls basically means sending request to rest server and taking response from server and database association. GUI component has three main files. These are lib, templates and views. Lib basically includes CSS files. Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging web pages and user interfaces for web applications. Templates file includes HTML files of the project. Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. Lastly, views file includes JavaScript files of our project. Javascript is a high-level, dynamic, untyped, and interpreted run-time language. Also Jquery is used in order to simplify the client-side scripting of HTML. [6][7][8]

3.2. Source Code Structure

Our project implementation is under 1 main repository called EduSys.

https://gitlab.ceng.metu.edu.tr/squ4re/EduSYS

In this main repository, there are 2 main parts of the project: Cam Module and Server Module.

Cam Module can be reached from:
https://gitlab.ceng.metu.edu.tr/squ4re/EduSYS/tree/master/CamModule

Face detection, recognition and training implementation can be found under this module. Also, there are dataset, feature of this dataset and openface and dlib library directories under this module. Main processes about training and taking attendance is implemented in this module.

Server Module can be reached from:
https://gitlab.ceng.metu.edu.tr/squ4re/EduSYS/tree/master/Server

Server module contains GUI of EduSys. All of the back-end and front-end implementation is done under this module. This implementations can be found in:

https://gitlab.ceng.metu.edu.tr/squ4re/EduSYS/tree/master/Server/src/main

Source part of the server has three parts: Java, Resources and Webapp.
https://gitlab.ceng.metu.edu.tr/squ4re/EduSYS/tree/master/Server/src/main/java

First part is Java. Content of it can be reached from the link above. Java also has 6 parts: Composite_PKEY, edutest, models, rest, service and utility.

- Composite_PKEY is used to implement composite key relations in rest classes.
- Edutest contains the test module of the project. It has rest, database and Java test applications.
- Models contain the database tables as the Java classes. It is implemented for ORM.
- Rest is the Rest classes of the database tables. These classes have the methods that connect the Hibernate and the Java back end parts.
- Service has the service class that is used to call hibernate functions in the Rest part.
- Final part is utility. This has some back end part of the project. There are some helper classes to provide completeness of the project. For example, it has Camera utility that provides communication between Python implementation in the Cam module and GUI.

https://gitlab.ceng.metu.edu.tr/squ4re/EduSYS/tree/master/Server/src/main/resources

Second part is the resources part. This contains hibernate.cfg.xml and project.properties.

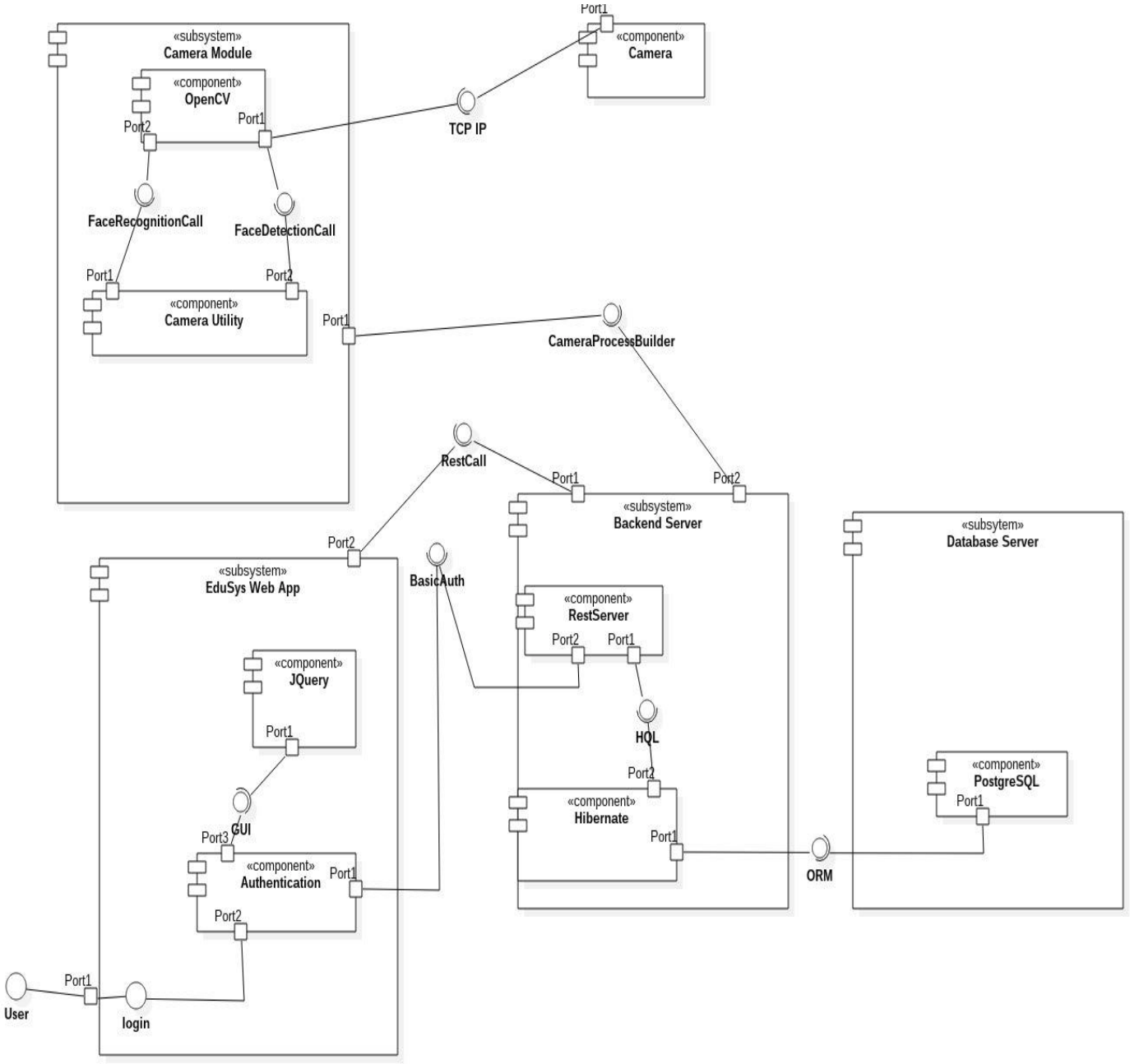
- Hibernate.cfg.xml shows the database server, location and Java classes of the tables.
- Project.properties has the general features and paths to use them in the other classes.

https://gitlab.ceng.metu.edu.tr/squ4re/EduSYS/tree/master/Server/src/main/webapp

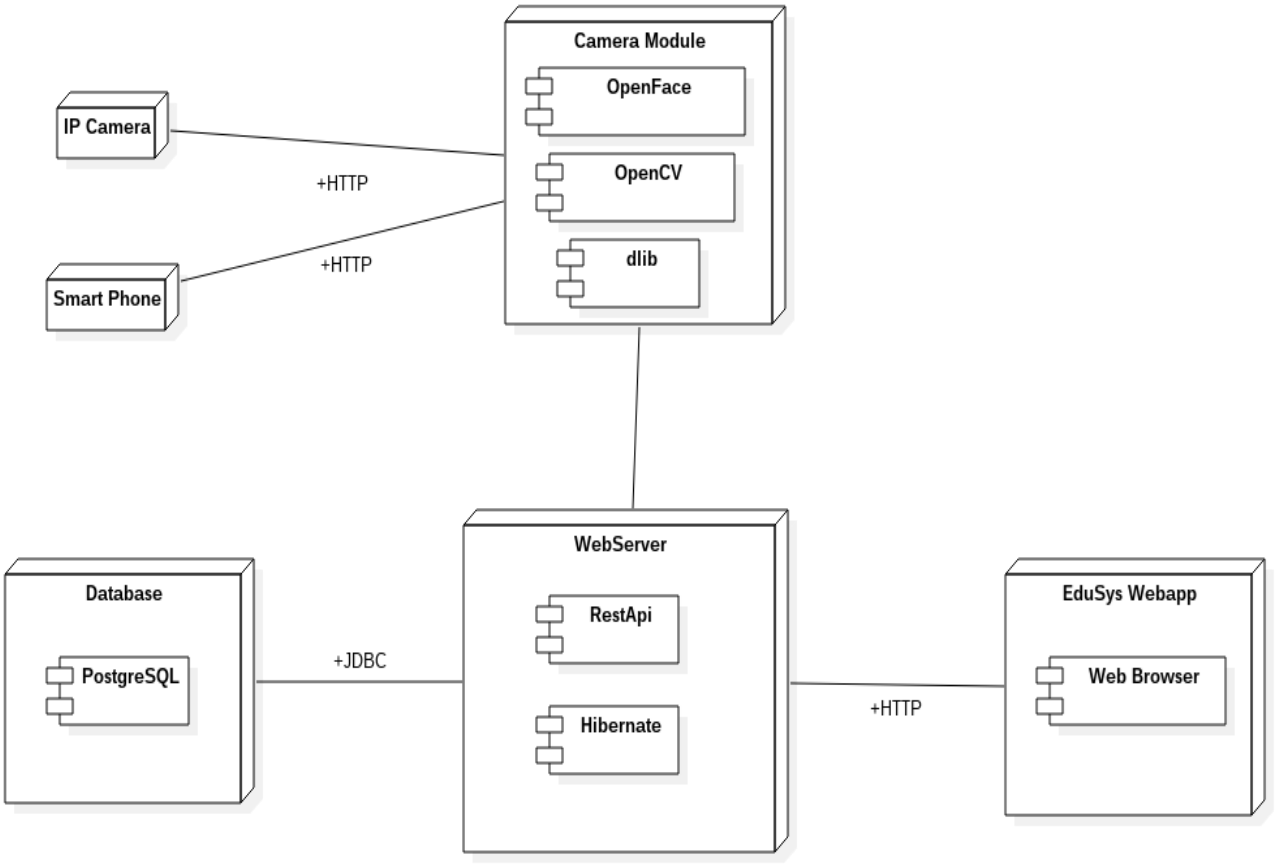
Last part is webapp. This part is the front end of the GUI. There are lib, templates and views in there.

- Lib has the JS implementation and CSS classes that are used in front-end.
- Templates has the HTML files. All page appearances is here.
- Views has the JavaScript implementation of the pages. All functionality of the pages is implemented here.

3.3. Component Diagram



3.4. Deployment Diagram



4. Testing

4.1. Test Plan and Scenarios

The purpose of this section is to provide the test cases of the EduSys project. It defines the objective, scenario and expected outcomes for each test case. This section also describes the specific items to be tested, and general test approach is described with the pass/fail criteria, and test deliverables.

Scenarios:

TC Identifier	TC1
Objective	Login
Input	Username, password
Expected Outcome	Given username and password, user data should be fetched from database and this data should be checked in the server whether this user has rights to login or not. After confirmation, a cookie which stores the TOKEN of the user must be created.
Environmental Needs	Server and database should run
Intercase Dependencies	-

TC Identifier	TC2
Objective	Registering User
Input	Basic User Information
Expected Outcome	Given user information, a new user must be created and it must be stored in database.
Environmental Needs	Server and database should run
Intercase Dependencies	TC1

TC Identifier	TC3
Objective	Deleting User
Input	userID
Expected Outcome	The user with the given userID must be deleted from database.
Environmental Needs	Server and database should run
Intercase Dependencies	TC1

TC Identifier	TC4
Objective	Updating User Profile
Input	Any information which the user wants to change
Expected Outcome	The user with the given userID must be updated in database.
Environmental Needs	Server and database should run
Intercase Dependencies	TC1

TC Identifier	TC5
Objective	Logout
Input	-
Expected Outcome	Clearing all cookies and local storage, redirecting to login page
Environmental Needs	-
Intercase Dependencies	TC1

TC Identifier	TC6
Objective	Viewing Profile Information
Input	-
Expected Outcome	User can see the information of his/her in the profile page
Environmental Needs	Server and database should run
Intercase Dependencies	TC1

TC Identifier	TC7
Objective	Listing Student’s Courses
Input	Selecting Specific Course
Expected Outcome	User must see the courses that he/she registered
Environmental Needs	Server and database should run
Intercase Dependencies	TC1

TC Identifier	TC8
Objective	Listing Student’s Attendance Data
Input	-
Expected Outcome	User must see the attendance data of a specific course
Environmental Needs	Server and database should run
Intercase Dependencies	TC1,TC7

TC Identifier	TC9
Objective	Viewing Exams
Input	Selecting specific course
Expected Outcome	List of exams that belongs to a course should be seen
Environmental Needs	Server and database should run
Intercase Dependencies	TC1,TC7

TC Identifier	TC10
Objective	Viewing Exam Grades
Input	Selecting specific exam
Expected Outcome	List of exam grades that belongs to an exam should be seen
Environmental Needs	Server and database should run
Intercase Dependencies	TC1,TC7,TC9

TC Identifier	TC11
Objective	Viewing Notification
Input	-
Expected Outcome	After clicking notification menu user should see the notifications that are listed in three categories.
Environmental Needs	Server and database should run
Intercase Dependencies	TC1

TC Identifier	TC12
Objective	Viewing Interest
Input	-
Expected Outcome	Interest graph must be filled with the correct values for all courses
Environmental Needs	Server and database should run
Intercase Dependencies	TC1

TC Identifier	TC13
Objective	Creating an Exam
Input	Exam specific inputs(course ID name, percantage of exam...)
Expected Outcome	New exam must be registered to the database correctly.
Environmental Needs	Server and database should run
Intercase Dependencies	TC1,TC7

TC Identifier	TC14
Objective	Adding an exam grade
Input	Grade specific inputs(student ID, section ID)
Expected Outcome	List of exams that belongs to a course
Environmental Needs	Server and database should run
Intercase Dependencies	TC1,TC7

TC Identifier	TC15
Objective	Taking Online Attendance
Input	Course ID, Section No, Classroom ID
Expected Outcome	An attendance should be created and students whose faces are recognized by IP camera are listed in this attendance. And redirect the page to the course home. This action should create a notification and send it to all the students that are recognized.
Environmental Needs	Server and database should run
Intercase Dependencies	TC1,TC7,TC11

TC Identifier	TC16
Objective	Taking Offline Attendance
Input	Course ID, Section No, Classroom ID
Expected Outcome	An attendance should be created and students whose faces are recognized from a list of images that are uploaded are listed in this attendance. And redirect the page to the course home. This action should create a notification and send it to all the students that are recognized.
Environmental Needs	Server and database should run
Intercase Dependencies	TC1,TC7,TC11

TC Identifier	TC17
Objective	Adding Exam Grades via Excel File
Input	Excel File, exam ID
Expected Outcome	User should see all the grades in the specific exam.
Environmental Needs	Server and database should run
Intercase Dependencies	TC1,TC7,TC9,TC13

TC Identifier	TC18
Objective	Seeing Heat Map of Seating Place
Input	-
Expected Outcome	If user is student, he/she should see the heat map of himself/herself of a specific course. Else if user is lecturer, he/she should see overall heat map of where students sit.
Environmental Needs	Server and database should run
Intercase Dependencies	TC1,TC7

TC Identifier	TC19
Objective	Adding Student to an Attendance
Input	Basic Student Information
Expected Outcome	Adding another student to an already existing attendance
Environmental Needs	Server and database should run
Intercase	TC1,TC7

Dependencies	

TC Identifier	TC20
Objective	Deleting a Student from an Attendance
Input	StudentID, attendanceID
Expected Outcome	Deleting a student from an already existing attendance
Environmental Needs	Server and database should run
Intercase Dependencies	TC1,TC7

TC Identifier	TC21
Objective	Login with Wrong Password or Username
Input	Username , password
Expected Outcome	User enters wrong username or password and he/she should not proceed to home page.
Environmental Needs	Server and database should run
Intercase Dependencies	TC1

4.2. Testing Code Structure

One can find test code of our project here;
https://gitlab.ceng.metu.edu.tr/squ4re/EduSYS/tree/master/Server/src/main/java/edutest

There are two kinds of tests in our project, model tests and rest service tests. Model tests are about back-end side functions and operations but rest service tests are more about front-end side functions of the project.

Model Tests:
https://gitlab.ceng.metu.edu.tr/squ4re/EduSYS/tree/master/Server/src/main/java/edutest/modeltester

Rest Tests:
https://gitlab.ceng.metu.edu.tr/squ4re/EduSYS/tree/master/Server/src/main/java/edutest/edu resttester

4.3. Test Results

You can find the results of test cases form the table below;

Test Case Identifier	Result
TC1	Pass
TC2	Pass
TC3	Pass
TC4	Pass
TC5	Pass
TC6	Pass
TC7	Pass
TC8	Pass
TC9	Pass
TC10	Pass
TC11	Pass
TC12	Pass
TC13	Pass
TC14	Pass
TC15	%80 successful with a standard quality IP camera
TC16	More picture, more successful (%74)
TC17	Pass
TC18	Pass
TC19	Pass
TC20	Pass
TC21	Pass

5. References

[1] https://en.wikipedia.org/wiki/PostgreSQL

[2] https://en.wikipedia.org/wiki/Apache_Tomcat

[3] https://en.wikipedia.org/wiki/Representational_state_transfer

[4] https://en.wikipedia.org/wiki/Hibernate_(framework)

[5] https://cmusatyalab.github.io/openface/

[6] https://en.wikipedia.org/wiki/Cascading_Style_Sheets

[7] https://en.wikipedia.org/wiki/HTML

[8] https://en.wikipedia.org/wiki/Javascript