SPRING 2023-2024  
SE112 – INTRODUCTION TO SOFTWARE ENGINEERING

Class Attendance System

GROUP 33  
  
22243810049 – YİĞİT ŞEVKİ KAPLAN  
22244710058 – YUSUF ÖZDOĞAN

Table Of Contents

[1. Project Proposal 1](#_Toc165504086)

[2. Project Management 2](#_Toc165504087)

[2.1. Project Plan 2](#_Toc165504088)

[2.2. Risk Plan 3](#_Toc165504089)

[3. Requirements 4](#_Toc165504090)

[4. PlantUML 5](#_Toc165504091)

[4.1. UseCase 5](#_Toc165504092)

[4.2. Class Diagram 6](#_Toc165504093)

[4.3. Sequence Diagram 8](#_Toc165504100)

[5. Figma 10](#_Toc165504101)

# Project Proposal

**CLASS ATTENDANCE SYSTEM**

**Project/Problem Description:** The aim of this project is to creating a system that allows the instructer to see who participate in the class safe and accurate. For this instance, system through image processing will be developed. Class Attendance system is a image processing based application designed for the use of all kinds of instructers. The data base on this system will contain student details (id number, name, surname, photograph). System will automatically detect the student and will update the data system. Users can see or search for all the details about the person in an efficent and consistant way. The system will show the users most absent student at that time. Image processing and ChatGPT will be used to develop this project. We’ll collect the data using Image processing through camera located in the classroom; ChatGPT for the student who changed their looks, ChatGPT will put them in to a list for instructor to give authorization for.

# Project Management

## Project Plan

A screenshot of a computer

Description automatically generated

## Risk Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Likelihood | Consequences | | | |
|  | Insignificant | Acceptable | Significant | Destructive |
| High |  |  |  |  |
| Medium |  |  | The components are used in the system can’t process as many as transactions per second as expected. | The underlying technology on which the system is built is superseded by new technology. |
| Low |  |  |  |  |

Very Effective, Effective, Ineffective

* Plan B for the “The underlying technology on which the system is built is superseded by new technology.”

Add a “Update” period lasts 21 days to keep up with the current technology.

* Plan B for the “The components are used in the system can’t process as many as transactions per second as expected.”

Use the reserved resources to upgrade the current hardware and current components.

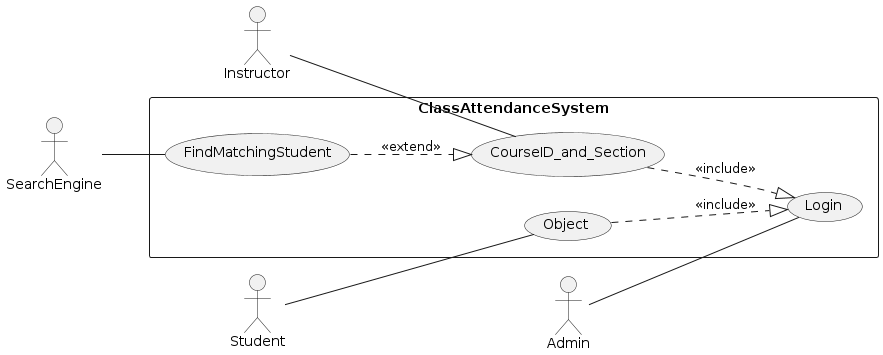
# Requirements

|  |  |  |
| --- | --- | --- |
| Functional Reqs ID | Explanation | MOSCOW priority (M, S, C, W) |
| FR1 | The software system should be integrated with ATACS system | M |
| FR2 | Only instructers and IT departmant have the right to edit the attandance data. | M |
| FR3 | System must allow instructer to login. | M |
| FR4 | Instructer must choose the course ID and section. | M |
| FR5 | Instructer can edit the list until 10 minutes after class ends | C |
| FR6 | Instructer can see the previous attandance of the students with searching up his/her name. | S |
| FR7 | Instructer can change the course ID and section. | M |
| FR8 | Image processing should begin after the course ID and section enterd. | S |
| FR9 | The software system could be integrated with CHATGPT for better recognision. | C |
| FR10 | Student can object. | C |
| Non-Functional Reqs ID | Explanation | MOSCOW priortiy  (M, S, C, W) |
| NFR1 | System must backup daily | M |
| NFR2 | Software must be multilingual | M |
| NFR3 | The camera must send the processed data to system within 1 second | M |

Class Attandance System is used to arrange and manage the attandance list of the specified class. Instructers should be able to enter the course ID and section, Admin may authorize the action, students can object at the end of the course if needed. All users (Instructer, Admin, Student) need to login to ATACS before using it. Instructer should be able to search the students name and get the all attandance information about him/her.

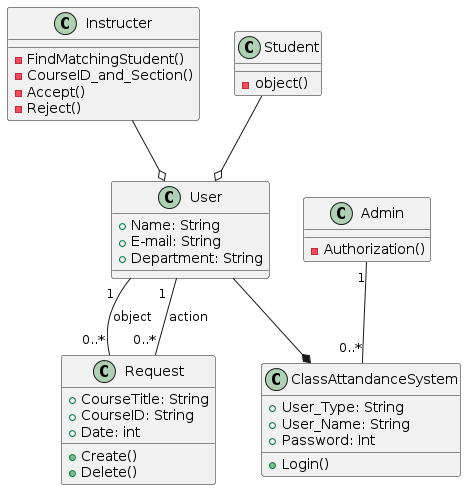
# PlantUML

## UseCase



@startuml  
left to right direction  
  
actor Instructor  
actor Admin  
actor Student  
actor SearchEngine  
  
rectangle ClassAttendanceSystem{  
usecase Login  
usecase CourseID\_and\_Section  
usecase Object  
usecase FindMatchingStudent  
}  
  
Instructor -- (CourseID\_and\_Section)  
Student -- (Object)  
Admin -- (Login)  
SearchEngine -- (FindMatchingStudent)  
(CourseID\_and\_Section) ..|> (Login) : <<include>>  
(Object) ..|> (Login) : <<include>>  
(FindMatchingStudent) ..|> (CourseID\_and\_Section) : <<extend>>  
@enduml

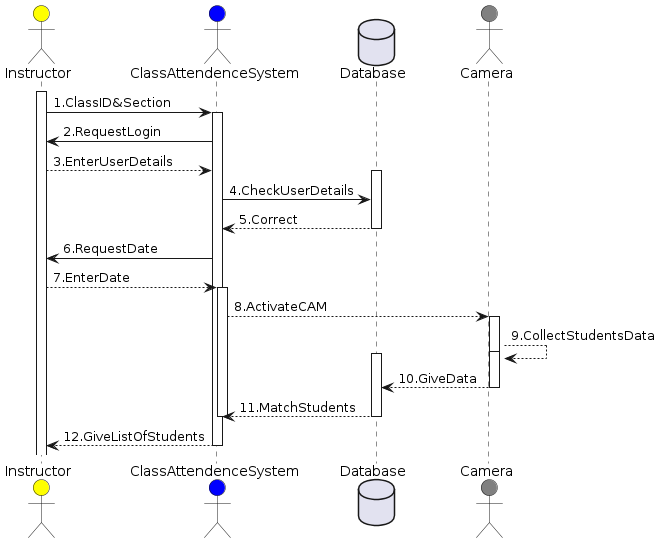
## Class Diagram



@startuml  
top to bottom direction  
  
class Instructer{  
-FindMatchingStudent()  
-CourseID\_and\_Section()  
-Accept()  
-Reject()  
}  
  
Class Student{  
-object()  
}  
  
Class Admin{  
-Authorization()  
}  
  
Class User{  
+Name: String  
+E-mail: String  
+Department: String  
}  
  
Class ClassAttandanceSystem{  
+User\_Type: String  
+User\_Name: String  
+Password: Int  
+Login()  
}  
  
Class Request{  
+CourseTitle: String  
+CourseID: String  
+Date: int  
+Create()  
+Delete()  
}  
  
Student --o User  
Instructer --o User  
User --\* ClassAttandanceSystem  
Admin "1" -- "0..\*" ClassAttandanceSystem  
User "1 " -- "0..\*" Request: object  
User "1 " -- "0..\*" Request: action  
@enduml



## Sequence Diagram



@startuml  
actor Instructor as In #yellow  
actor ClassAttendenceSystem as CAS #blue  
database Database as DB  
actor Camera as CAM #gray  
activate In  
In->CAS: 1.ClassID&Section  
activate CAS  
CAS->In: 2.RequestLogin  
In-->CAS: 3.EnterUserDetails  
activate DB  
CAS->DB: 4.CheckUserDetails  
DB-->CAS: 5.Correct  
deactivate DB  
CAS->In: 6.RequestDate  
In-->CAS: 7.EnterDate  
activate CAS  
CAS-->CAM: 8.ActivateCAM  
activate CAM  
activate CAM  
CAM --> CAM: 9.CollectStudentsData  
deactivate CAM  
activate DB  
CAM --> DB: 10.GiveData  
deactivate CAM  
DB --> CAS: 11.MatchStudents  
deactivate DB  
deactivate CAS  
CAS-->In: 12.GiveListOfStudents  
deactivate CAS

@enduml

1. The instructor chooses the course
2. The user is prompted to log in
3. Instructor logs in
4. Checking whether the password and username are correct
5. Password and username are confirmed
6. The date of the course is asked to the instructor.
7. instructor enters date
8. Activating Camera
9. Data collecting with Image processing and ChatGPT.
10. Students' identities and names, as well as their attendance status, are taken from the database.
11. CAS makes the list
12. List shown to the Instructer

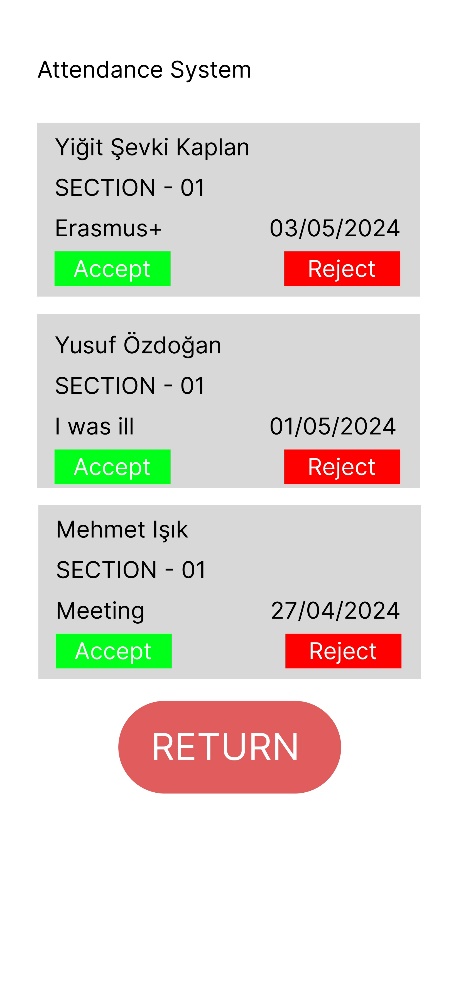
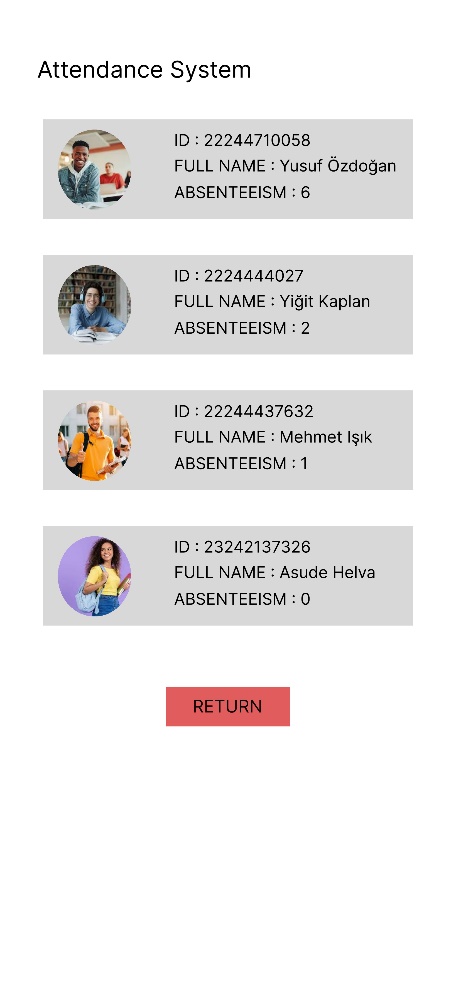
# Figma

[SE112\_Project\_33\_Link](https://www.figma.com/file/zNud8rR9ehBqX8HSWgyogX/SE112_Project_33?type=design&node-id=18%3A101&mode=design&t=yFqMxQyEBs4dxZvJ-1)

metin, ekran görüntüsü, yazı tipi, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, ekran görüntüsü, yazı tipi, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, ekran görüntüsü, yazı tipi, logo içeren bir resim

Açıklama otomatik olarak oluşturuldumetin, ekran görüntüsü, yazı tipi, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu