SOFTWARE DESIGN DOCUMENT

SMART WARNING SYSTEM

(Project no.-9)

Created by :

Group no.-18

Abhishek Kumar (160101007)
Abhishek Ranjan (160101008)
Hemant Yadav (160101034)

Supervised by :
Prof. Samit Bhattacharya

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to give a detailed description of the design for the Smart Warning System. This includes use case models, sequence diagrams, class diagrams and other supporting information. It will illustrate the complete declaration for the development of system along with the system constraints. This document is primarily intended as a reference for developing the first version of the system for the development team.

1.2 Document Conventions

Term	Definition
Instructor/Professor	Person who shall be using the software for monitoring
Student	Person who shall be monitored by the instructor
Users	Collectively refers to the students and instructors
Device	An electronic device using which the instructor is delivering their lecture
Accelerometer	Sensor that helps determine the shaking frequency
Gyroscope	Sensor that helps determine the orientation of student's device
Proximity Sensor	Sensor that helps determine the distance between the student and their device
Sensors	Collectively refers to the Accelerometer, Gyroscope and Proximity sensors on the students' devices
UI	User Interface
Node Server	Backend Server written in Node.JS
Socket	Two-way communication gateway layer between student and instructor devices
Android	Android is a mobile operating system developed by Google

1.3 Project Scope

This software is meant to be deployed in an IT-enabled large classroom environment where this software shall allow the instructor to conveniently monitor the attention of their students in real time. The intended audience will be professors and the students of those schools and colleges where a high speed working wifi is available and all the devices in the classroom are connected to this main wifi.

This software has 3 major components one is the server running on the device of the instructor, next is the client side application running on the students' devices which monitors the sensors and the third is an online portal that helps the students and instructors review past attention data.

2. Design Overview

2.1 Introduction

The Design Overview section is to introduce and give a brief overview of the design. The System Architecture is a way to give the overall view of a system and to place it into context with external systems. This allows the reader and user of the document to orient themselves to the design and see a summary before proceeding into the details of the design.

2.2 Background

The responsibility of the proposed system shall be to monitor the attention of the students and ensure that they are not chatting/slacking or are indulged in entertainment. As soon as the attention of a student changes, the system gives a warning in form of an alarm or a pop-up or a notification as chosen by the user.

2.3 System Architecture

The use case environment consists of an Instructor/Professor and students attending the ongoing lecture. A server is initiated by the professor which overlays the connection to the students who connect to the server from their mobile devices. The server uses socket connections for relaying data across the professor and student devices.

The application also asks permission to monitor sensors of the students devices. If the permission is not given, then it gives a notification to prof and if the permission is granted then the application instantiates a SensorEventListener which is triggered whenever the readings of the following sensors: Accelerometer, Gyroscope, Proximity Sensor change.

The sensor manager then determines the attention flag based on the following conditions:

- Distance between student and device <= 25 cm(approx.)
- Screen is in interactive mode for more than 5 seconds continuously.
- Device is neither parallel to vertical nor horizontal.
- Shaking frequency is >= 2/min

The attention flag values are set according to the following-

(1) If none of the above conditions is false continuously for more than last 2 minutes, then the student is said to have attention of 10.
(2) If above conditions have been false continuously for more than 1 minute but less than 2 minutes, set attention flag to value 9.
(3) If the above conditions have become false in last 1 minute, set the attention flag to 8 for that particular student.
(4) If the conditions have become true in last 1 minute, set the attention flag to 7.
(5) If these conditions are true for more than 1 minute but less than 2 minutes continuously, set attention flag to 6.
(6) If above conditions are true for last 2 minutes but less than 5 minutes continuously, set attention flag to 5.
(7) If above conditions are true for more than 5 minutes, set the attention flag to 4.

(8) If the student has a total attendance of more than 90% till now, but is not connected to the server today, set his attention flag to 3.
(9) If the student is absent today but has attendance greater 75% but less than 90%, set his attention flag to 2.
(10) If the student is absent today and has attendance less than 75%, set the attention flag to 1.

Attention flag between 8 to 10 both inclusive means that the student is not using his phone in class. Attention flag between 4 to 7 both inclusive means that the student is present in class but is not paying attention, while attention flag between 1 and 3 both inclusive means that the student is not present in class.
In scenario of a change in attention, professor is notified by a socket emit call and this event is recorded in the database with the session details which can later be accessed by both the professor and student after logging into the system portal whenever the user wants to see his/her previous attention reports.

2.4 System Interfaces and Implementing Technologies

2.4.1 User Interfaces

The user interface will allow the instructor and students to establish a socket connection between their devices. The application will then fetch the real time attention report of all the students to professor's device and give a warning whenever the attention changes.

2.4.2 Hardware Interfaces

The devices carried by the students need to measure the shaking frequency, distance and orientation, this will be done by using the sensors: gyroscope, accelerometer, proximity sensor which will be interfaced using the Android Sensor Manager Library.

2.4.3 Software Interfaces

The sensor data collected will be stored in a relational MySQL database. Node-mysql pooling will be used to update this database when students' device emit socket message to the professors device.

Further, the portal for application will be written in Node.JS framework.

2.4.4 Communications Interfaces

For establishing the connection between the student and instructor devices in the form of a wireless local area network, wifi will be used and Node.JS socket communication library will be used to implement the two way socket communication between the devices.

2.5 Design and Implementation Constraints

For the above purpose of implementing the app, we are not using the camera for measuring the attention. The students might exploit the sensors used to portray themselves as paying attention even when they are not.

This gives a false positive in cases when the student uses a stand to maintain the orientation and turning the screen off every 5 seconds rendering the accelerometer and gyroscopic functions redundant.

2.6 Assumptions and Dependencies

We assume the presence of an IT-enabled large classroom with the availability of a system where each and every device is connected to a main wifi network. We assume that the devices owned by the students will be based on the Android Operating System.

One assumption about the product is that it will always be used on mobile phones that have enough performance. If the phone does not have enough hardware resources available for the application, for example the users might have allocated them with other applications, there may be scenarios where the application does not work as intended or even at all. Also, we assume the devices have sophisticated sensors to meet our requirements of measuring even minute shaking to make the software precise.

3. Use Cases

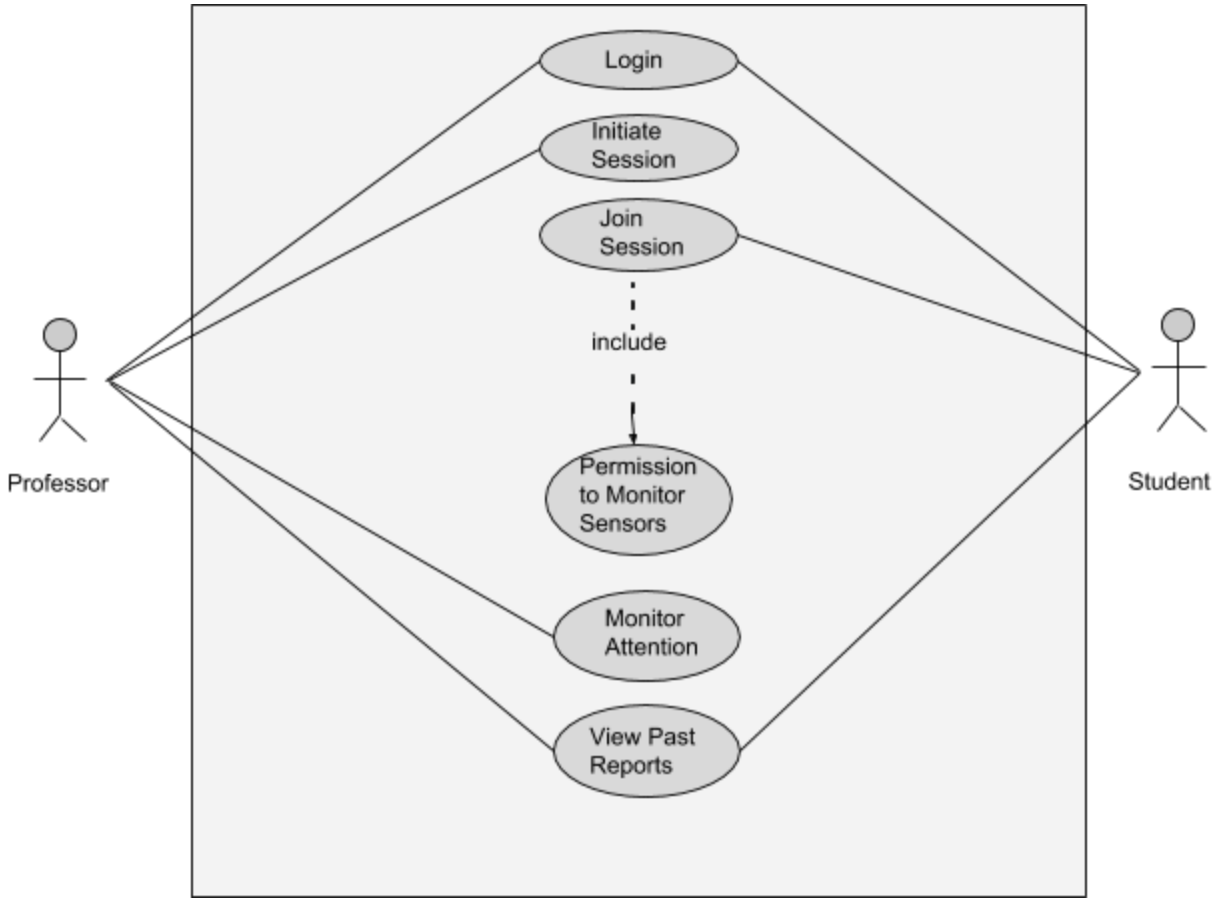


Figure 3.1: Use Case Diagram

3.1 U1: Login

Actors : Student, Professor (both are collectively referred to as the user)

Scenario 1: Mainline Sequence:

1.) User : Requests the login page.
2.) System : Prompts to enter details.
3.) User : Enter the details i.e username and password.
4.) System : Displays option to initiate session and view past session attention reports for that particular professor, change the alarm attributes.

Scenario 2: At step 4 in mainline:

4.) System : Error in case of incorrect username and/or password.

3.2 U2: Initiate Session

Actors：Professor

Precondition : The professor is logged in.

Scenario 1: Mainline Sequence:

1.) Professor : Select Session initiate option.
2.) System : Display prompt to enter course code and date-time for the session.
3.) Professor : Enter the required values.
4.) System : Create a local server and returns server address.

Scenario 2: At step 4 of mainline sequence:

4.) System : Displays error if a problem is encountered while creating the session.

Scenario 3: At step 4 of mainline sequence:

4.) System : Displays the message if some of the required information has not been entered correctly or is missing and it displays a prompt to enter the wrong or missing information. Wrong information will be the case When there exists another session with same credentials.

3.3 U3: Join Session

Actors : Student

Precondition : Student is logged into the system and a server has been created for the session.

Scenario 1: Mainline Sequence:

1.) Student : Requests to Join the session.
2.) System : Displays a prompt requesting the session address.
3.) Student : Enters the session address as directed by the professor.
4.) System : Shows a message acknowledging joining the session and requests for permission to monitor the sensors.

5.) Student : The student accepts the request for permission to monitor sensors.

Scenario 2: At step 4 of mainline sequence:

4.) System : Shows an error message when he enters an incorrect session address and prompts to re enter the session address.

Scenario 3: At step 5 of mainline sequence:

5.) Student : Rejects the permission to monitor sensors.
6.) System : Sends a message to the professor showing a student has not accepted

3.4 U4: Monitor Attention Detail

Actors : Professor

Precondition : Both student and professor are logged in, the session is initiated by the professor and at least one student has joined the session and the student has given permission to monitor the sensors.

Scenario 1: Mainline Sequence:

1.) System : Checks the sensor (gyroscope, accelerometer, proximity sensor) data and isInteractive function data and creates a report for every student and sends it to the professor's device.
2.) Professor : Receives the notification and acknowledges it.
3.) System : Notifies the student to pay attention.

Scenario 2: At step 1 of mainline sequence:

1.) System : Checks sensor data and determines that the student's attention state has not changed.
2.) Professor : Receives the current status of all the students.

3.5 U5: View Past reports

Actors： Student, Professor (both are collectively referred to as the user)

Precondition : Previous sessions data has been recorded and the user is logged in to view the report.

Scenario 1: Mainline Sequence:

1.) User : Selects the option of viewing the past report.
2). System : Displays a prompt to enter the course code and date-time.
3.) User : Enters the required details of the needed class.
4.) System : Displays the attention statistics of the session for the specified class. to a student only his report is shown while the professor has access to every student's report.

Scenario 2: At step 4 of mainline sequence:

4.) System : Displays an error message that the particular class details are incomplete or wrong and prompts to enter the correct details.

4. Object Oriented Model

4.1 Domain Modelling

4.1.1 Entity Objects

1. AttentionData
2. Student
3. Professor

4.1.2 Controller Objects

1. Socket : Server
2. Socket : Client
3. Monitor

4.1.3 Boundary Objects

1. Main Activity
2. SensorEventListener

4.2 Class Diagram

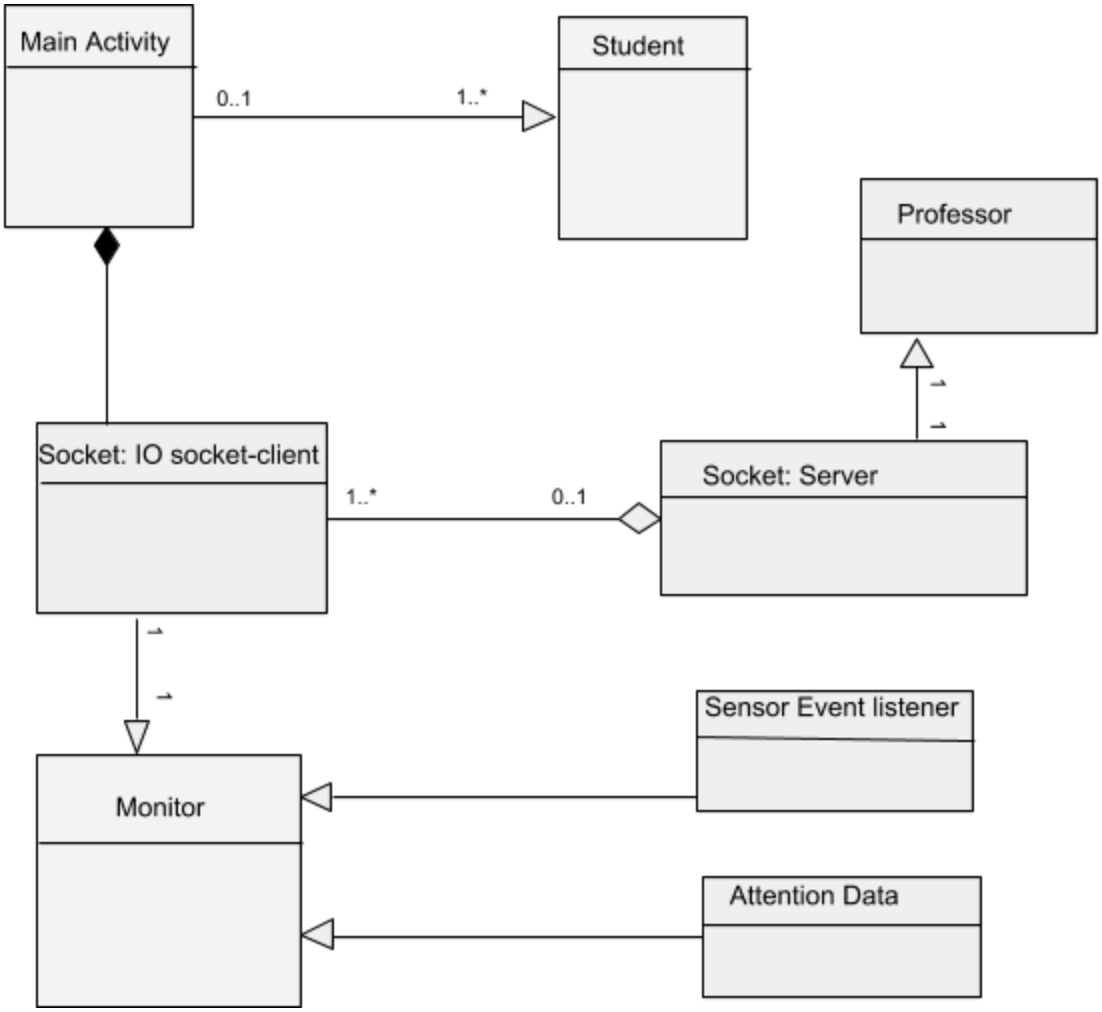


Figure 4.1: Class Diagram

4.3 Class Descriptions

4.3.1 Main_Activity

Extends: Activity

Description: An activity is a single, focused thing that the user can do .Almost all activities interact with the user, so the Activity class takes care of creating a window for user in which user can place his/her UI.

Main_Activity-Data Members

1. monitor;
Description: This object controls the activities related to the sensor data.

2. socket;
Description: This is a reference to the class that manages the connections between the student and instructor devices

3. student;
Description: It stores the details of the student(username, password, past report).

4. Professor;
Description: It stores the details of the professor(username and password,past session report).

4. view;
Description: It is a reference to the UI elements present in the mobile application running on user's devices.

Main_Activity-Methods

1. startConnection(string, string);
INPUT: Server Address and a unique token
OUTPUT: Reference to Socket connection object
CALLED BY: The Event handler of the UI Element (Button - Start Connection) along with Text Views containing the server address and tokens.
Description: It verifies the validity of token and server address and once validated, instantiates a socket object.

2. onCreate(Bundle savedInstanceState);
INPUT: A Bundle containing the activity's previous state
OUTPUT: Void
CALLED BY: Activity when it is first created.
Description: This method is where app initializes the activity. Most importantly, here app will usually call setContentView(int) with a layout resource defining UI, and using findViewById(int) to retrieve the widgets in that UI that are needed to interact with programmatically.

3. authenticate(string, string);
INPUT: email address, password of the student
CALLED BY: The Event handler of the UI Element (Button - Login) along with Text Views containing the email address and password.
Description: Verifies the email address and password combination and fetches student data from database and fills student object.

4. onPause()
INPUT: Touch
CALLED BY :when the system is about to start resuming a previous activity.
Description: Any changes made by the user at this point are committed (usually to the ContentProvider holding the data)

4.3.2 IO.socket-client

Inherited From: IO

Description: socket-client is a library class that allows establishment and management of the connection between the student and Professor devices. It will be instantiated on the student device and manage connection from students point of view.

IO.socket-client-Data Members

1. url;
Description: The Address of the server

2. socket.id**;**
Description: Socket identifier

IO.socket-client-Methods

1. Socket(string);
INPUT: Server Address
CALLED BY: Main_Activity::StartConnection
Description: Establishes the socket connection and registers all the event listeners.

2. onConnectEventListener();
CALLED BY: Triggered when the connection is established
OUTPUT: Void
Description: Asks for Permission to access Sensors from the student.

3. onDisconnectEventListener();
CALLED BY: Triggered when the connection is destroyed
OUTPUT: Void
Description: Informs student about connection problem.

4. emit(string);
INPUT: Message string
OUTPUT: Void
Description: Sends the message string to Instructor

4.3.3 AttentionData

Description: It stores the data readings given by the sensors and isInteractive() Function creates the attention report in real time according to the conditions described above.

AttentionData-Data Members

1.orientationData;
Description: The Accelerometer reading gives the orientation of the student's device.

2. proximityData;
Description: The Proximity Sensor reading gives the distance between student and his device.

3.gyroscopeData;
Description: The Gyroscope reading gives the shaking frequency of the device.

4. screenData;
Description: isInteractive() function's output gives tells if the screen is turned on. or off.

4.3.4 Monitor

Description: Manages the sensor data and takes actions according to the recordings.

Monitor-Data Members

1. sensorManager;
Description: It lets the application access student's device's sensors.

2. attentionData**;**
Description: Attention data values as produced above.

3. attentionFlag
Description: Attention value associated to a particular student.

4. attentionFlagChanged;
Description: True if the Attention Flag just changed from a higher value to a lower value.

5. sensorEventListener ;
Description: Used for receiving notifications from the SensorManager when there is new sensor data. It triggers whenever the accuracy of the registered sensor changes or there is a new sensor event.

Monitor-Data-Methods

1. Monitor();
CALLED BY: OnConnect EventListener
Description: Instantiates the sensor manager and registers all the event listeners for the sensors. The **onconnect** property of the SharedWorkerGlobalScope interface is an EventHandler representing the code to be called when the connect event is raised

2. checkAttention();
CALLED By: onSensorChanged(SensorEvent event)
Description: Sets the attention flag and checks if it changes and appropriately informs the Professor.
ALGORITHM: For proper attention, following must be satisfied:

a) Distance between student and device = 25 cm(approx.)
b) Shaking frequency >= 2/min
c) orientation- Neither parallel to horizontal nor to vertical
d) screen is turned on continuously for more than 5 seconds.

4.3.5 SensorEventListener

Description: Listens to the sensor Data

SensorEventListener-Methods

1. onSensorChanged(SensorEvent event)
Description: Used for receiving notifications from the SensorManager when sensor values have changed. Called when the accuracy of a sensor has changed or when sensor values have changed.

4.3.6 Server

Description: A Socket is the fundamental class for interacting with browser clients. A socket uses an underlying Client to communicate which is the student in this case.

Server-Data Members

1. socketId;
Description: Socket identifier

2. socketClient;
Description: A reference to the underlying Client object.

Server-Methods

1. Socket();
CALLED BY: UI Event Listener - Trigger by professor to start server.
Description: Establishes the socket Server to which students can connect.

2.onConnectionEventListener();
CALLED BY: Triggered whenever a new student connects.
Description: Updates the student count data.

3. onDisconnectEventListener();
CALLED BY: Triggered when a student disconnects
Description: Notifies the Professor of the event and the student who disconnected and also updates the records database.

4. onAttentionEventListener(SensorData, bool, Student)
INPUT: Sensor Data and Attention Flag and student details
CALLED BY: Socket Message Listener when a ”Attention” message is emitted by the client.
Description: Notifies the professor about the change in attention of the particular student

4.3.8 Student

Description: Encapsulation of the Student Data

Student-Data Members

1. name;
2. rollno;
3. email;
4. course;
5. session;

4.3.9 Professor

Description: Encapsulation of the Instructor Data

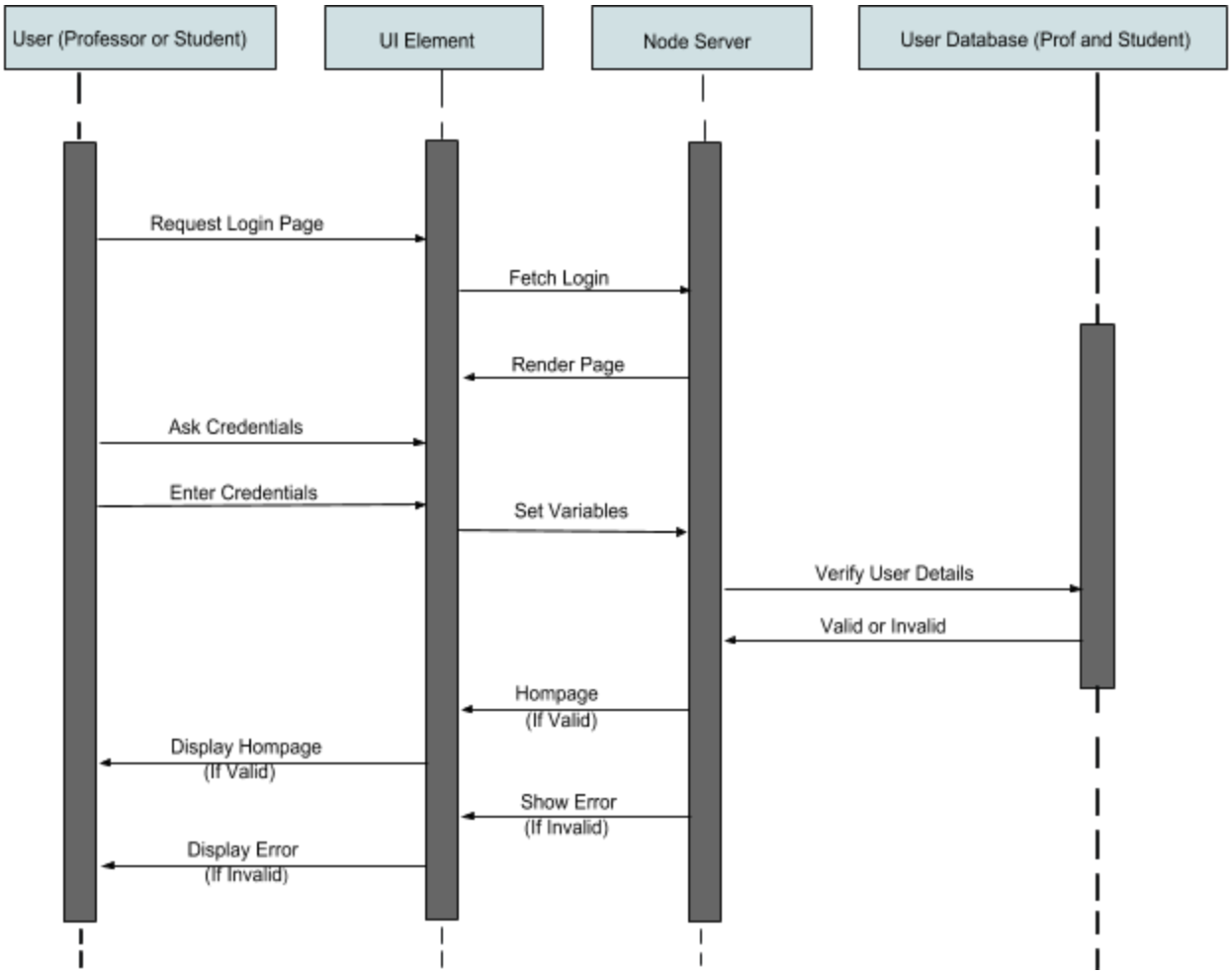
Professor-Data Members

1. name;
2. department;
3. email;
4. course;
5. session;

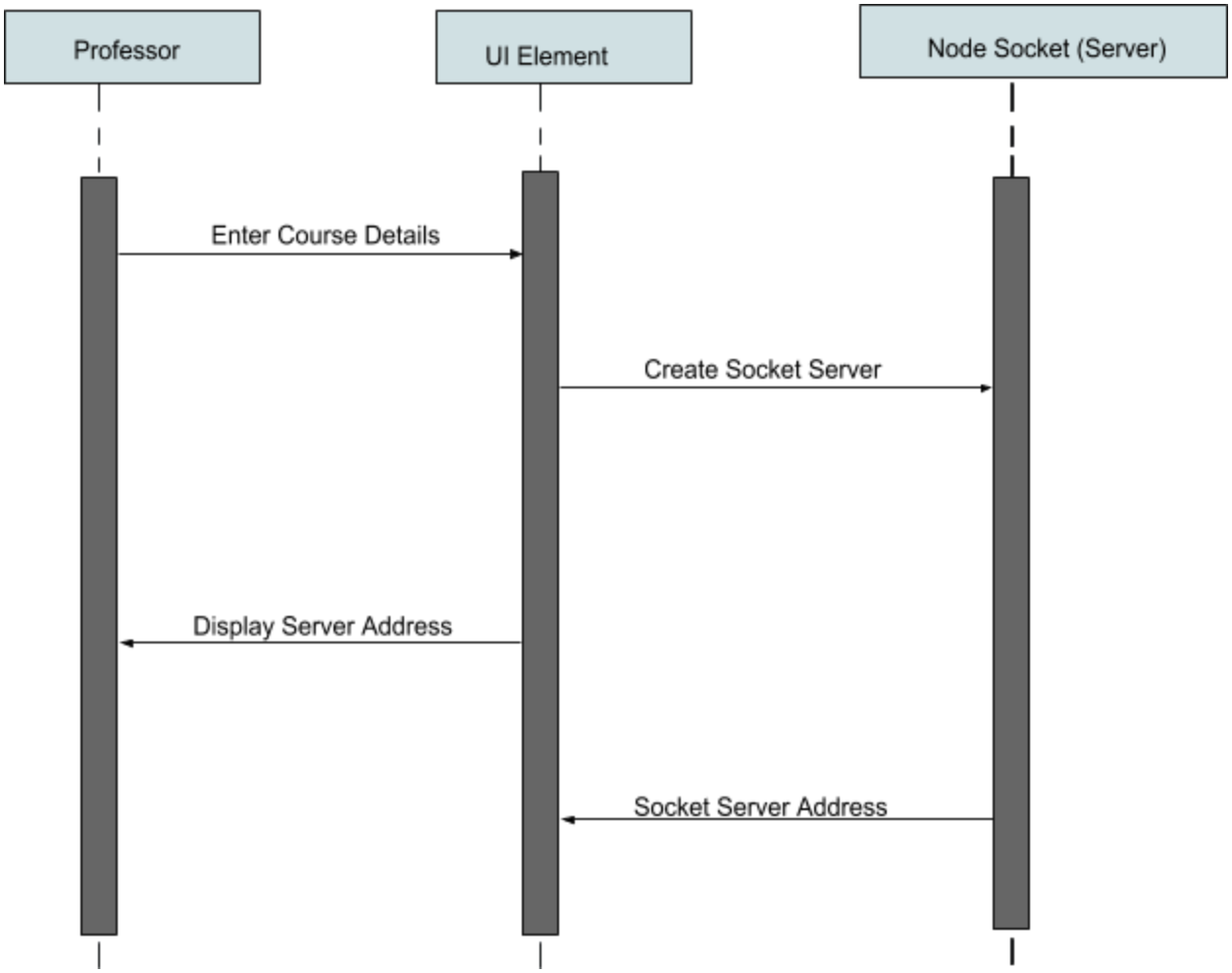
5. Sequence Diagram

The following Diagrams depict the sequences followed during each of the use cases mentioned.

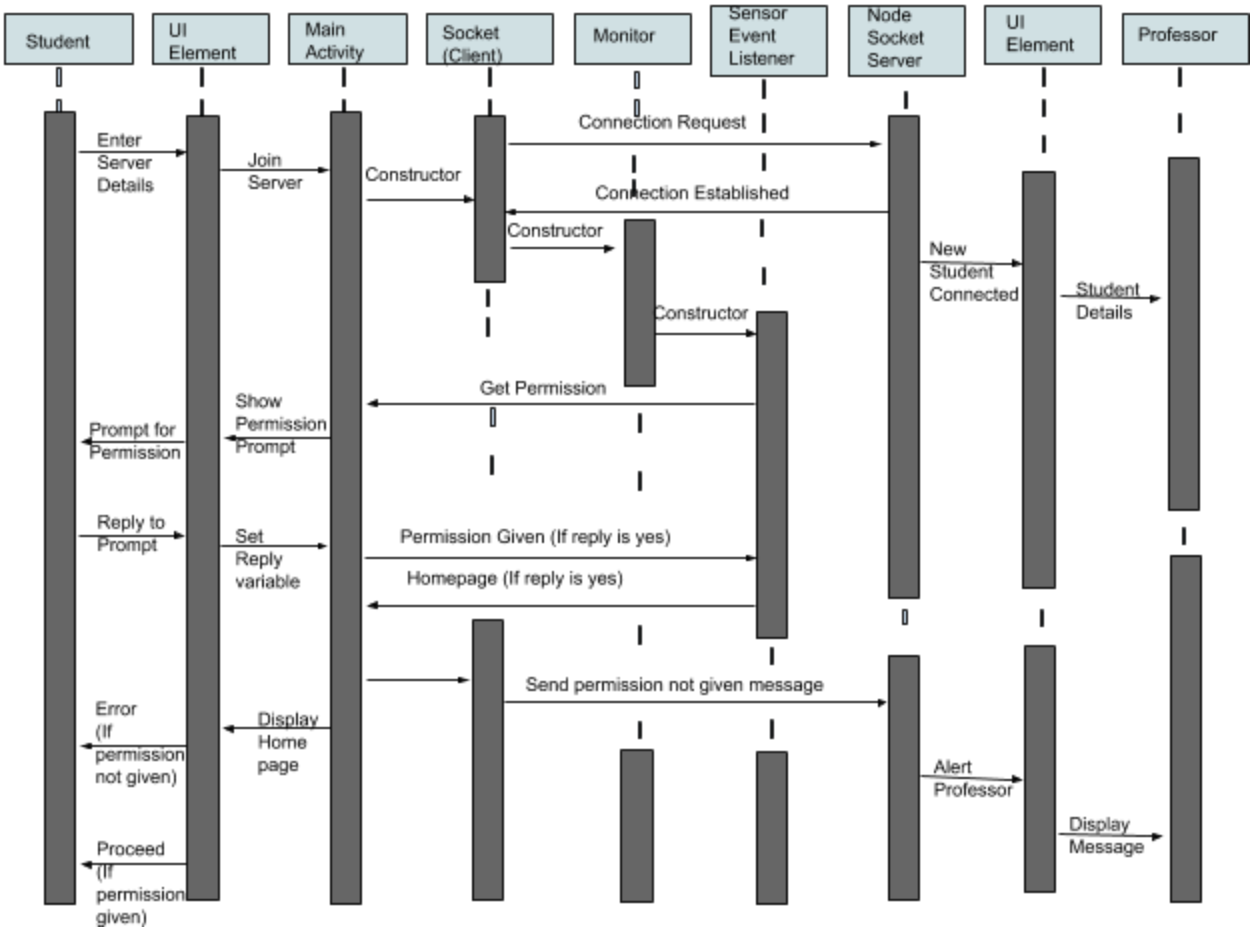
Use Case: <u>LOGIN SEQUENCE</u>



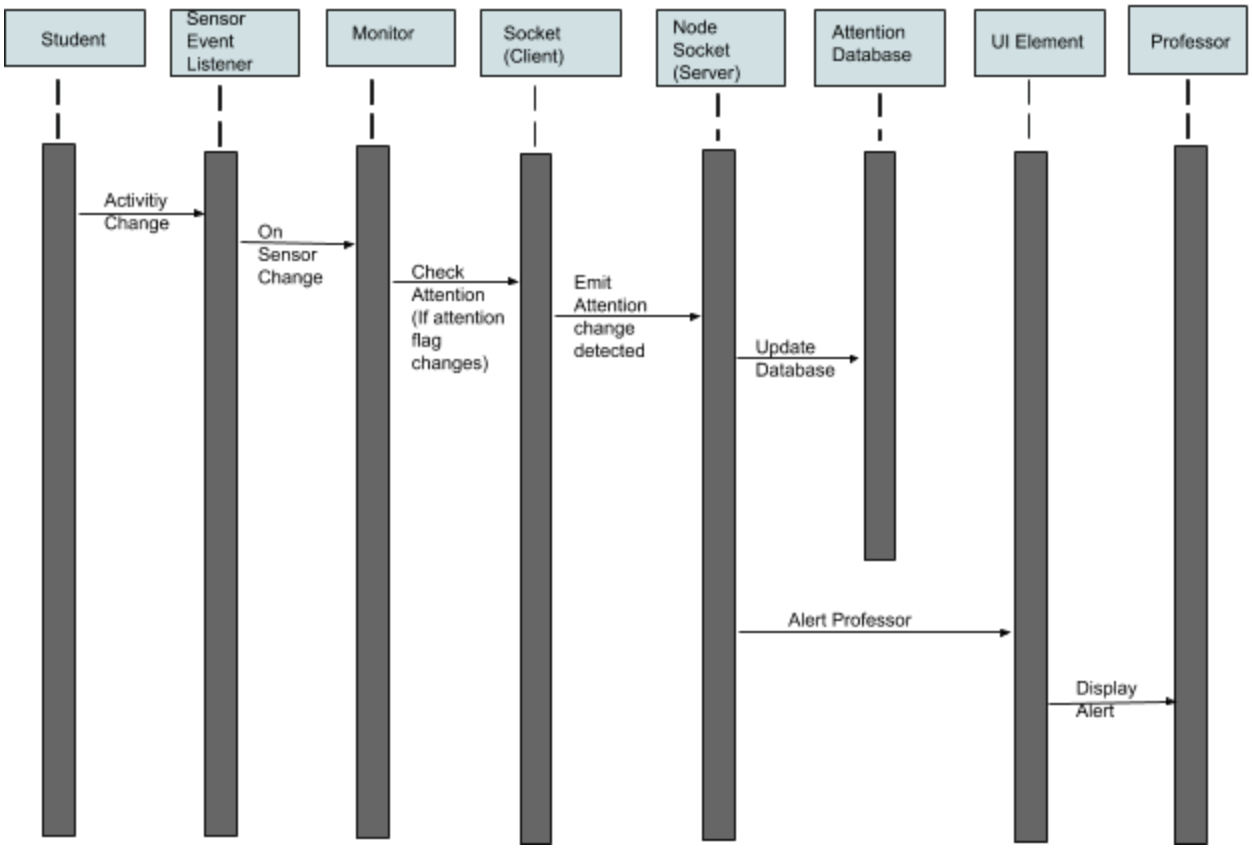
Use Case: <u>Initiate Session Sequence</u>



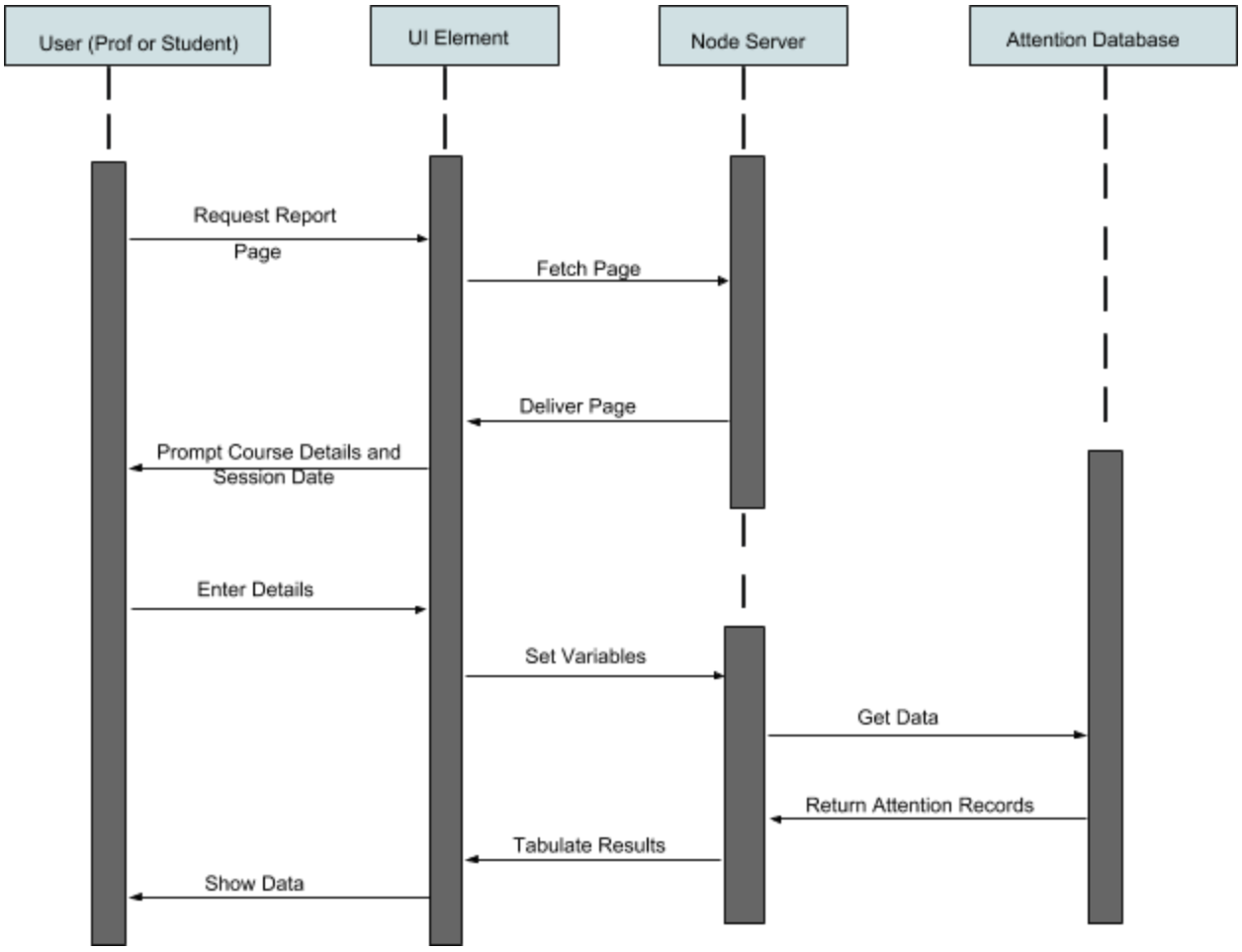
Use Case: <u>Join Session Sequence</u>



Use Case: <u>Attention Monitor Sequence</u>

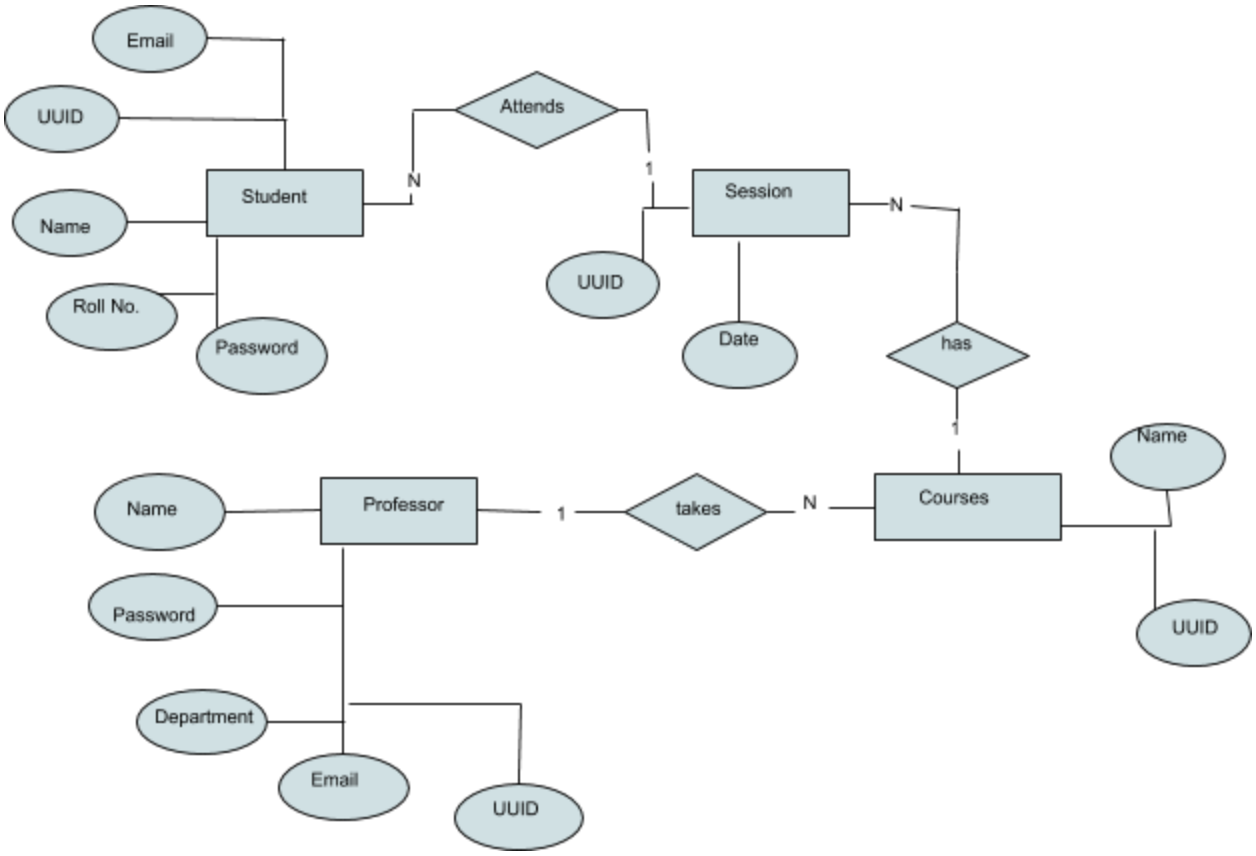


Use Case: <u>View Past Report</u>



6. Data Design

6.1 Entity Relationship Diagram



7 Nonfunctional Requirements

7.1 Performance Requirements

- The search enabling the instructor to view the summary of a student and give his analysis should be clear and well understandable.
- The program must update the real time statistics at a fast enough pace so that the instructor can have a clear view on the current attention of the class.
- The program must be able to handle the information of the entire class i.e in full attendance almost 90 students and also individually not be glitch in a given mobile device.
- Time to give the error message on losing a particular connection should be minimal (almost immediate).

7.2 Security Requirements

- The students enter their authentication (webmail).

References

[1] Software Requirement Specification for Student Activity Monitor and Alert Generator Version 1.2, Dated: 16/2/2017

[2] Android Developer Reference https://developer.android.com/reference/ packages.html Google 2017

[3] Node.js Documentation https://nodejs.org/api/ v7.8.0

[4] Share Latex Learn Documentation https://www.sharelatex.com/learn

5] Stack Overflow Forum https://stackoverflow.com/

[6] Activity - Android https://developer.android.com/reference/android/app/ Activity.html/

[7] Socket Client API Reference https://github.com/socketio/socket. Io-client-java

[8] Socket Server API Reference https://socket.io/docs/server-api/

[9] Sensors Reference - Android https://developer.android.com/guide/topics/ sensors/sensors_overview.html