Islamic University of Technology (IUT)

AUTOMATION OF ATTENDANCE SYSTEM AT IUT

Final Report

Course

CSE 4408

System Analysis and Design Lab

Mohammad Ishrak Abedin
160041051
Shakleen Ishfar
160041029
Anika Tasnim Preoty
160041044

Contents

Team Members

Member Name	Member ID
Mohammad Ishrak Abedin	160041051
Shakleen Ishfar	160041029
Anika Tasnim Preoty	160041044

Introduction

The current attendance system at Islamic University of Technology (IUT) depends on the use of an attendance sheet in most cases. The sheet is either passed through the students or the teacher calls the name or ID of each of the students one by one. In some cases, some teachers do take attendance through the use of Excel or other spreadsheet applications, but none the less, the process is time consuming, inaccurate and prone to proxy and error in all the cases mentioned above. Moreover, when it comes to the case of performing calculation on the data set of attendance, it requires manual labor and calculation which is difficult to do accurately at a large scale and cannot give any instant snapshot of the attendance of a specific student at a given moment.

On the other hand, the automated attendance systems that are currently in use at other universities or places make use of sophisticated hardware like finger print or any other biometric scanners or NFC based card scanners. These are both costly and difficult to manage and the teacher has no direct control or overview over the attendance of the students.

We wanted to build an automated attendance system that is fast, reliable and easy to implement as well as less sophisticated-hardware oriented. Our system aims at solving the problems of current analogue attendance system through a way that is convenient for both teachers and students. Through the implementation of regular day hardware that everyone uses, like mobile phones and computers, we wish to automate the attendance system in a simple but effective manner.

Problems of the Current System

i. Teacher has either to pass the attendance sheet and students sign it manually or the teacher him/herself takes attendance by calling each of the students.
ii. Both passing down of sheet or calling requires a large amount of time, especially in the cases where class size is around hundred and it wastes valuable class time.
iii. Calling attendance can often lead to mismarking in the attendance sheet.
iv. In a large class, it is quite easy for the students to give proxy for other students since it is not possible for the teacher to physically track each of the responses. Passing down signature sheet is also prone to proxy since a student can sign on behalf of another absent student.
v. As the semester goes on, the attendance sheet grows larger and it becomes very hard to keep track of the previous records. Moreover, since it is a physical object, there is no proper way to keep any back up of it. So if it is lost once, it is lost forever.
vi. Finally, there is no proper way to get instant information from the data present at the attendance sheet. If the teacher wants to know whether a student has attendance percentage over marginal percentage or not, he needs to manually acquire the attendance and then calculate them. This is

both tedious and there is every chance of mistakes. Moreover, for the whole data set, it is really difficult to carry on such calculation.

Objectives

By analyzing the aforementioned problems, we decided to design such a system that will contain the following properties:

i. An automated system that does not require much manual effort
ii. A system that takes attendance quickly and effectively
iii. A system that is accurate and resistant to mistakes
iv. A system that is secured and will prevent proxies
v. A system that will automatically manage and store all of the gathered data
vi. A system that can perform instantaneous and automated calculation over the collected data set at any time and generate required report automatically on pre-specified events and user requirement.

Proposed Solution

By analyzing the current problems and the identified objects, we went on to design a system that works in the following steps. The design consists of two major parts, teacher's and students', with an admin part which mainly consists of database management.

1. At the beginning of each class, the teacher generates a unique QR code that contains the information of the teacher, class and a randomizer. The teacher displays the QR code in the projector screen. This app can reside in the computers that are situated in the class rooms and the teacher can log into their accounts and generate the QR for the specific class. The teacher can keep the QR active for any suitable amount of time he/she desires.
2. The students would log into their own respective application that would preferably run on Android mobile devices, and log in using their SIS (Student Information System) account and scan the QR code which would give them attendance for the current class.
3. Once the students have given their attendance, the teacher can view the attendance on his/her app and modify it in case of any error or if any student fails to provide attendance using the automated system.

4. At the same time, the system would run its calculations and show the list of defaulters to the teacher. A notification will also go to the defaulter's app notifying him/her.
5. The teacher can also perform other calculations over the attendance list such as checking the attendance of any specific students, or overall class count or even the attendance list of any certain day.
6. By tracking how many presents are being submitted from one device and time frame between submissions of attendances, the system can identify and block proxies, thus providing security.

Requirements for the Project

- The system must be secured.
- It should only be operable from inside IUT campus.
- The system should be using the same user accounts as the ones currently being used in the existing system of IUT.
- It must provide a user-friendly interface.
- The application will be running on smart phones from users' end. So, it should have both forward and backward operating system compatibility.
- The application should be light-weight in case of storage, performance and memory consumption.

Constraints

- Limited amount of time.
- The knowledge and skills required to build the application.
- Permission is required to access and store the server based system modules and information in the official server of IUT.

Feasibility Analysis

Economical Feasibility:

Software Requirements:

1) **Software Development Kits (SDKs)**
   a) Java SDK: We chose Java to create the system that works on teacher's end and runs on computer.
   b) Android SDK: Android SDK shall be used to create the apps for the students.
2) **Integrated Development Environments (IDEs)**
   a) **IntelliJ IDEA:** IntelliJ IDEA is known as the most powerful IDE for java for its powerful intellisense, code refactoring and parsing capabilities along with huge libraries and multiple language integration. Moreover being students of IUT also provides free usage of the Ultimate (Complete) version of the software for free.
   b) **Android Studio:** Android Studio will be used for the development of students' apps.
3) **Interface Design**
   a) **JavaFX** is used for the design of teacher's application.
   b) **Android XML** will be used to design the students' application.

Hardware Requirements:

- **Computers:** Required for development of the system. Class room computers are required to implement teacher's application.
- **Mobile phones:** Required for testing and debugging the developed system.
- **Routers and networks:** Required for the attendance system network.
- **Data storage and IUT Server**: Required to store information regarding attendance system and leave forms for female students.

Developers and Debuggers:

- During primary stage, we will be designing the systems alongside testing and debugging it.
- On closed beta stage, selected students and teachers will be trying out and testing the features.
- On open beta stage, any student or teacher of IUT are eligible to use and test the system.

Technical Feasibility:

1) **Development Possibilities:** Far more sophisticated attendance systems like biometric or NFC based attendance systems are being used at many places. Compared to those, a simple QR code based attendance system will not prove difficult to be developed.
2) **Technical Abilities:** We are using Java to develop the app at teacher's end. We have already learnt Java in the current semester. Moreover, Android uses a modified version of Java. So, implementing the students' app will not prove difficult.

Operational Feasibility:

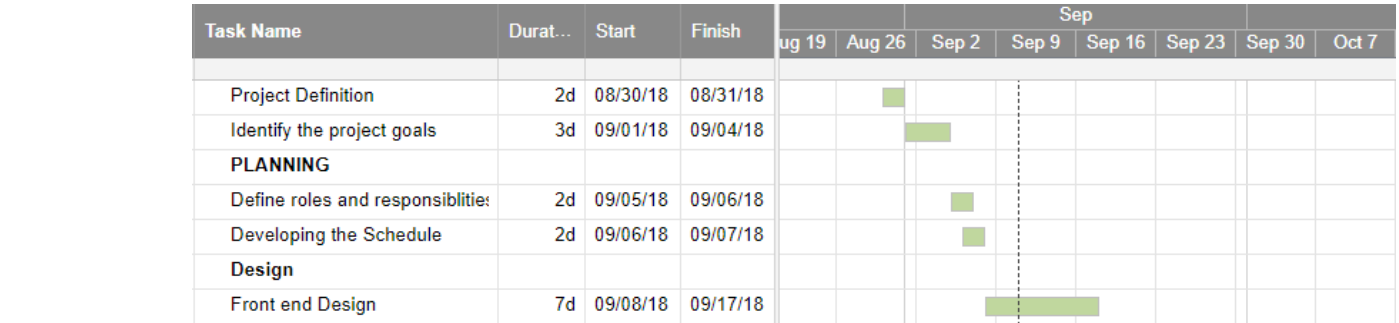
- **Supported systems:** The whole system will be running on Android for students, Java on Windows operation system for teachers and the backend database would run on SQL server of IUT. Almost all the students in IUT uses Android smart phones and all the class room computers run on Windows. IUT already runs a SQL servers for its database. So, implementing the new system on the current devices will not pose any problem.
- **User Interface:** We will be trying to build an intuitive, easy to control and simple user interface so that anyone can operate it with ease. Through prototypes and continuous improvements, we wish to achieve so.
- **Automation at hand:** Our system will be working to solve the issues with the current system. The proposed system, as seen, is faster, more accurate and securer than the old system at all ways. Moreover, it requires very little manual work, just the initiation and checking process. At the same time, it is capable of keeping records of a running semester along with the past ones. It can also generate instant reports and give answer to different queries of the teacher over the attendance list. Overall, it streamlines the attendance process for both the teachers and the students and switching to the new system should not prove to be much difficult. The system will be launched through prototypes and staged builds for further improvements.

Activity Planning

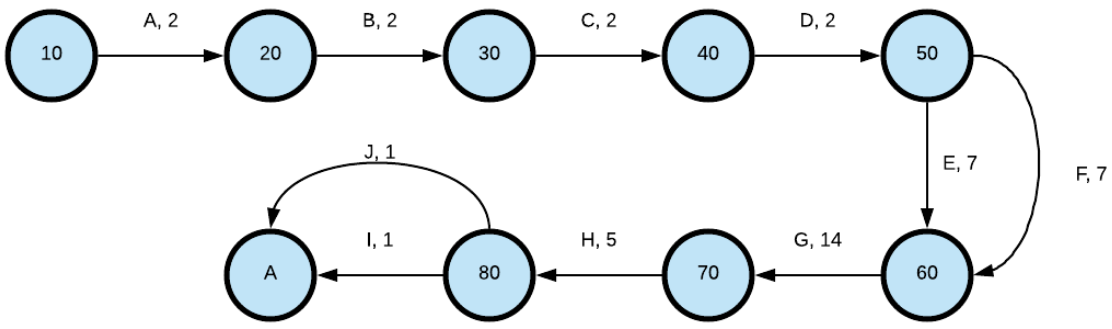
We started out project at 30 August, 2018 and it is currently ongoing. The activity planning for the development of the first prototype of the system is given below. The snapshot was taken at 04 September, 2018.

Gantt chart:

Task Name	Durat...	Start	Finish
Project Definition	2d	08/30/18	08/31/18
Identify the project goals	3d	09/01/18	09/04/18
PLANNING			
Define roles and responsiblities	2d	09/05/18	09/06/18
Developing the Schedule	2d	09/06/18	09/07/18
Design			
Front end Design	7d	09/08/18	09/17/18
Back end Design	7d	09/08/18	09/17/18
Implementation			
System modules	14d	09/18/18	10/05/18
Testing	5d	10/06/18	10/11/18
Monitoring and Controlling			
Measuring the ongoing project	1d	10/12/18	10/12/18
Identify corrective actions	1d	10/13/18	10/13/18



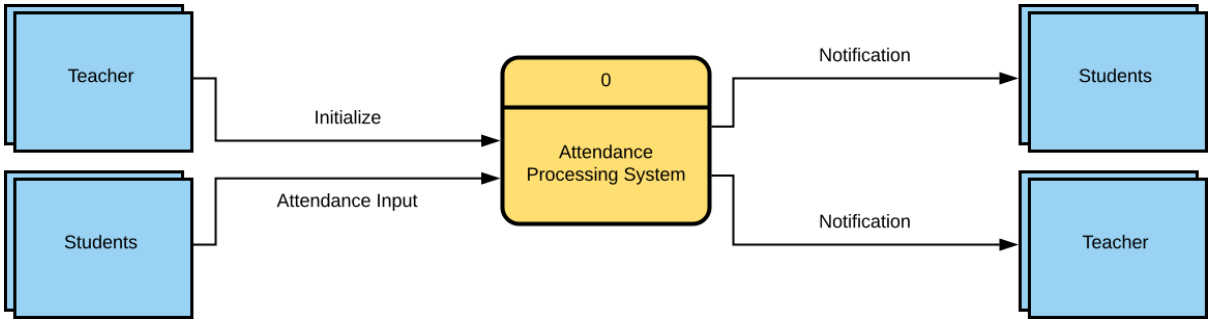
PERT Diagram:



Diagrams

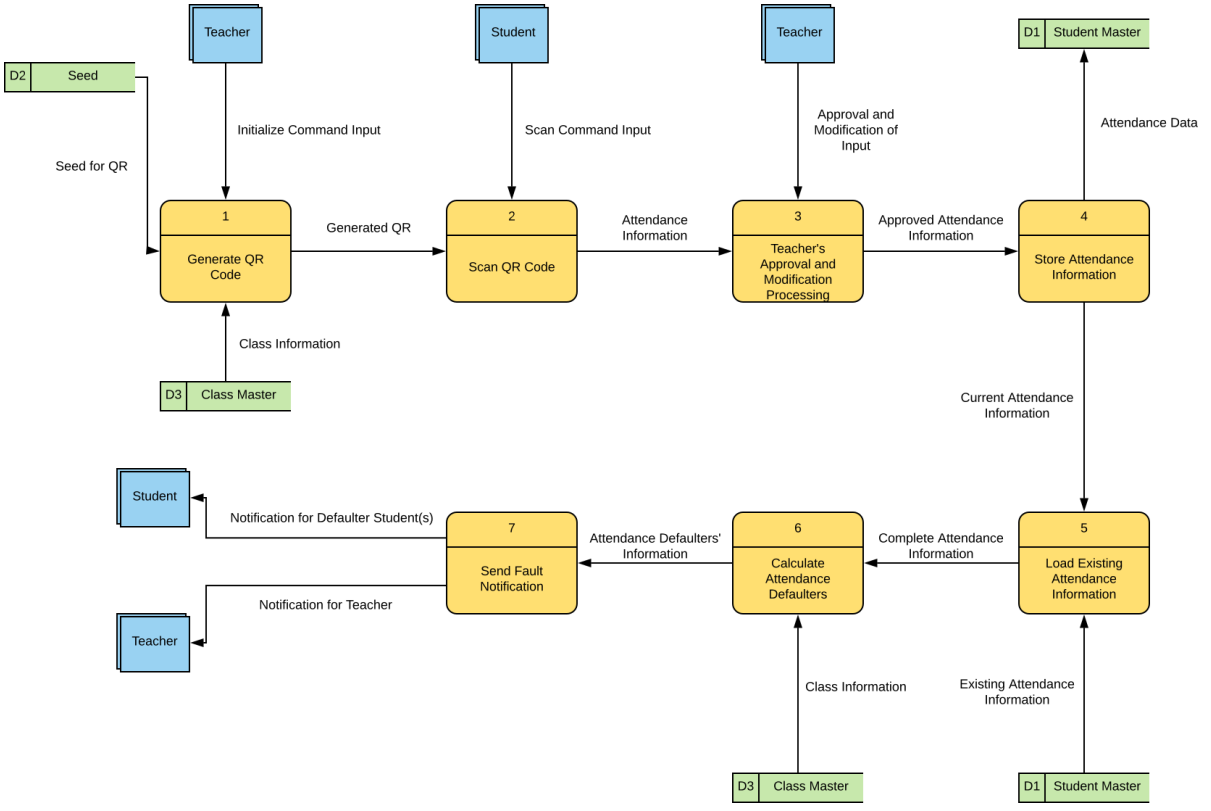
Data Flow Diagrams

Context Diagram:



The context diagram shows the complete overview of the project, in short, the picture. The main two entities that are connected are teacher and students. The main or 0 process is “Attendance Processing System”. The teacher initialized the system and students provide attendance. The end result is report and notification for teacher and defaulter students respectively.

Zero Diagram:



If we explode the context diagram or process 0, we get the 0 diagram. It gives the data flow in the overall project. It starts with the teacher initiating the project. The system generates a QR code from data storage “Seed” through the “Generate QR Code” process. It also uses “Class Master” data storage to get the necessary information about the class. Then the students scans the generate QR code using “Scan QR Code” process. The collected information is sent for teacher’s approval in “Teacher’s Approval and Modification Processing” process. Once the attendance is approved, it is stored at “Student Master” data storage using “Store Attendance Information” process. Then the existing attendance is load from “Student Master” using the “Load Existing Attendance Information” process. The complete attendance information is sent to “Calculate Attendance Defaulters”. Information is drawn from “Class Master” to provide class information. Finally fault notification is sent to teacher and defaulter students using “Send Fault Notification” process.

If we further explode Zero Diagram, we get child diagrams. The child diagrams and respective description are given next.

Diagram 1:

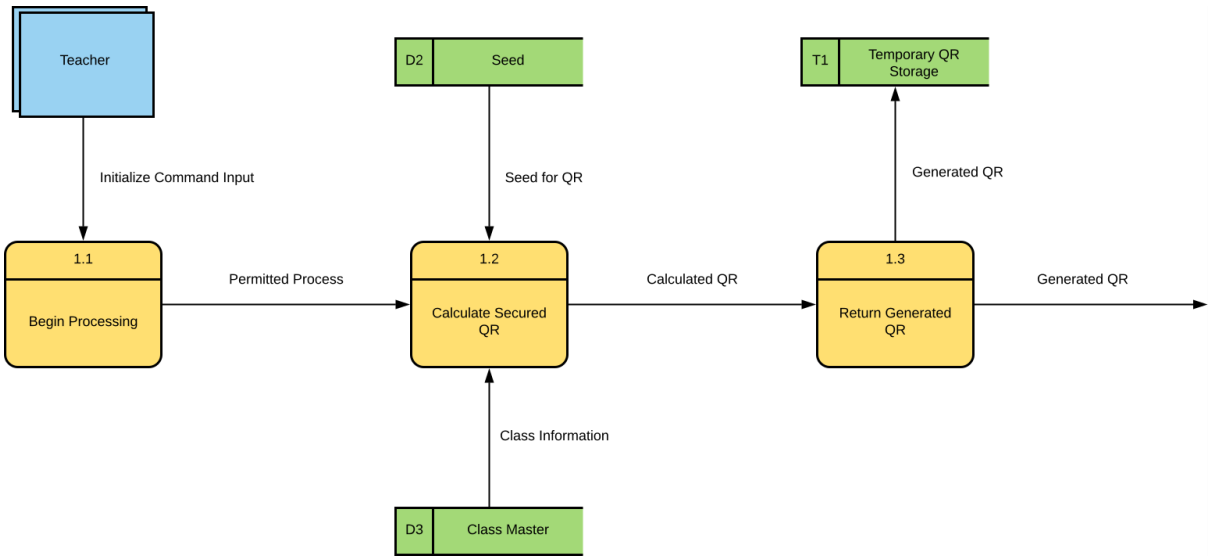


Diagram 1 shows the exploded process of “Generate QR Code”. The teacher initializes the system through “Begin Processing”. Then the permitted process calculates QR code using “Calculate Secured QR” process which draws information from “Seed” which randomizes the QR and “Class Master” for class information. Then it returns generated QR code using “Return Generated QR” process and keeps the QR temporarily in “Temporary QR Storage”.

Diagram 2:

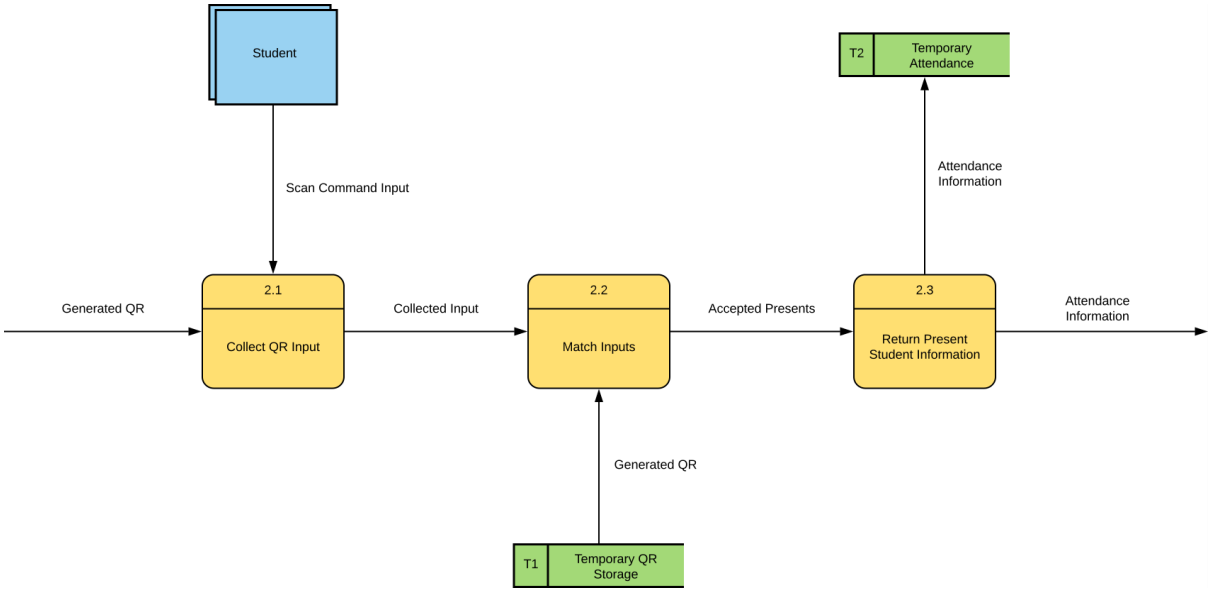


Diagram 2 shows the exploded process of "Scan QR Code". The students scan QR code and it is collected through "Collect QR Input" process. Then it is matched with the generated QR by extracting it from "Temporary QR Storage" in "Match Inputs" process. Finally the accepted attendance information is returned through "Return Present Student Information" process and is temporarily kept in "Temporary Attendance" data storage.

Diagram 3:

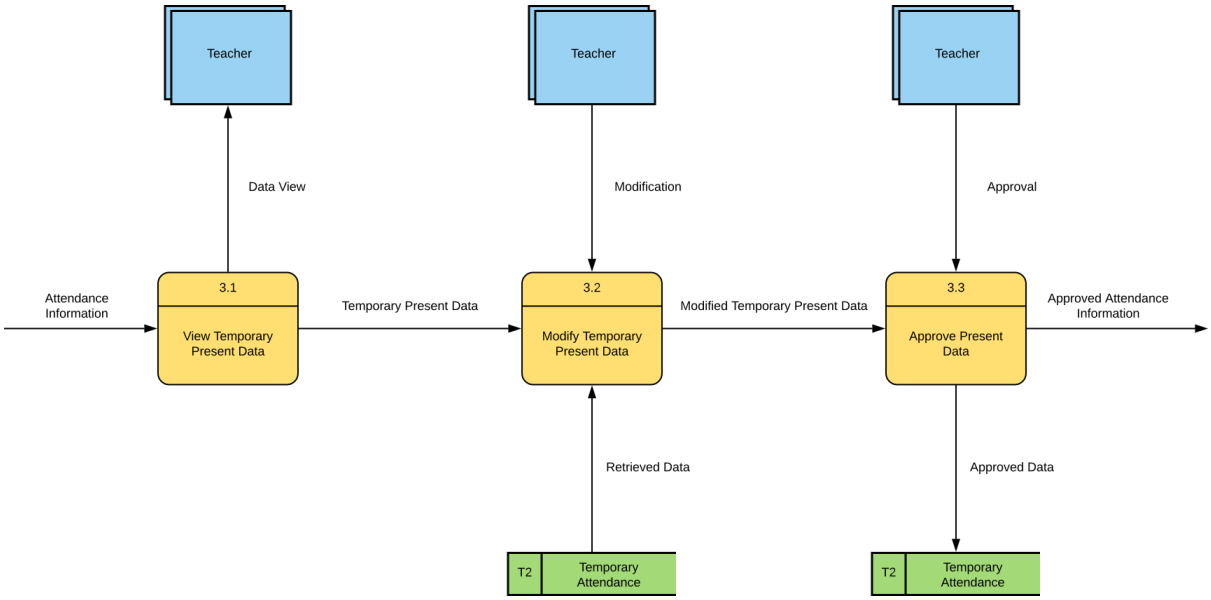


Diagram 3 shows the exploded process of “Teacher’s Approval and Modification Processing”. The teacher can view the temporary present data using “View Temporary Present Data” process. After that, if necessary, he/she can modify it using “Modify Temporary Present Data” process. It draws information from “Temporary Attendance”. Once done, the teacher needs to approve the accepted attendance using “Approve Present Data” process.

Diagram 4:

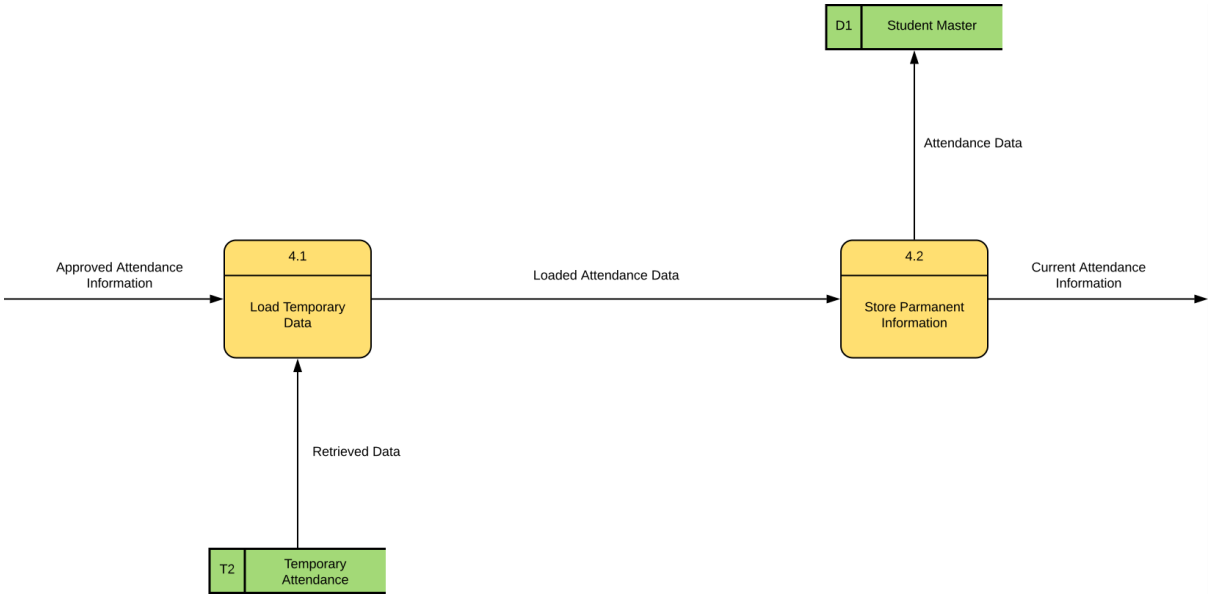


Diagram 4 shows the exploded process of “Store Attendance Information”. It first retrieves attendance information from “Temporary Attendance” data storage using “Load Temporary Data” process and then it stores it permanently in “Student Master” storage using “Store Permanent Information” process.

Process 5, “Load Existing Attendance Information” is trivial and hence no child diagram is shown.

Diagram 6:

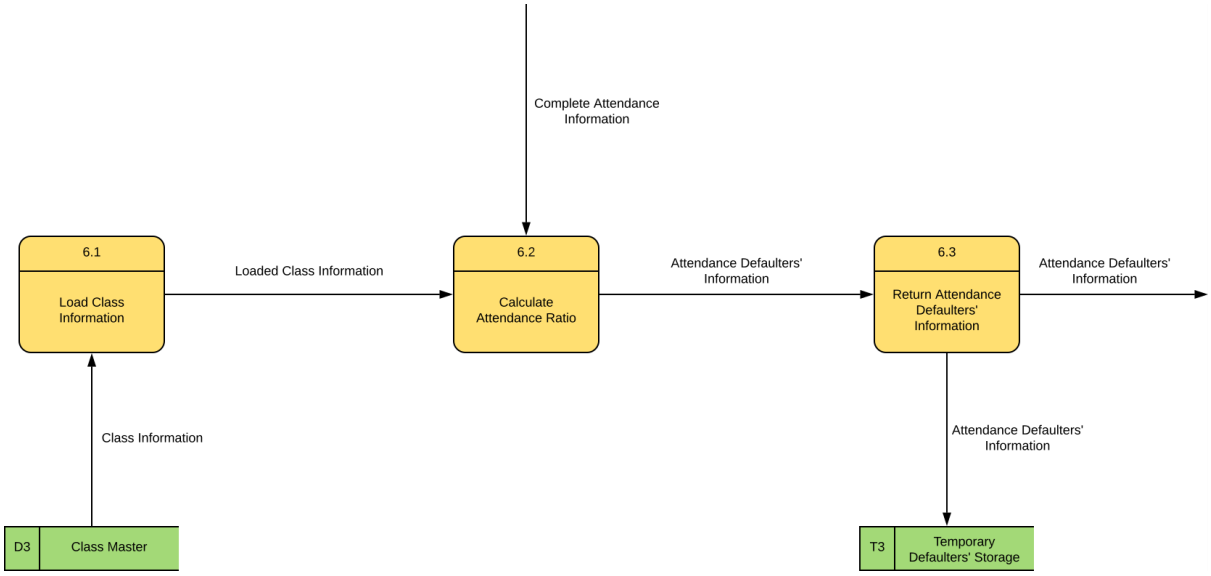


Diagram 6 shows the exploded process of "Calculate Attendance Defaulters". First it loads the class information from "Class Master" using "Load Class Information" process. Then from the complete attendance information from previous process, it calculates the attendance ratio using "Calculate Attendance Ratio" process. Once done, it returns the defaulters' information using "Return Attendance Defaulters' Information" process and stores it temporarily in "Temporary Defaulters' Storage".

Diagram 7:

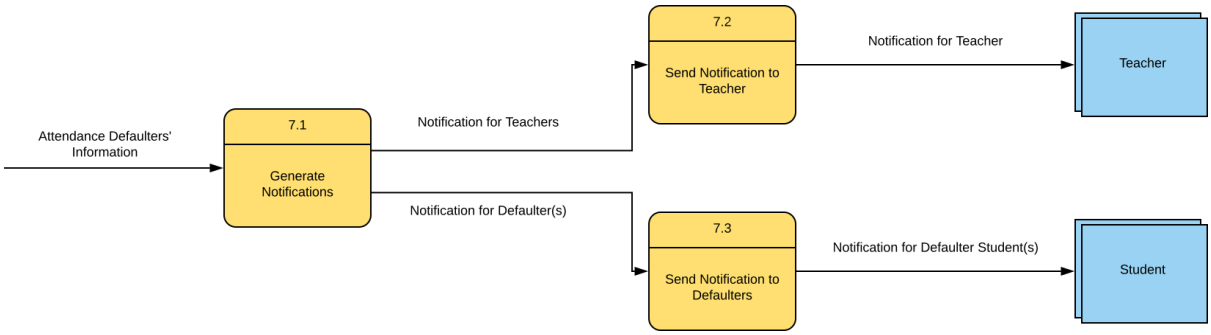
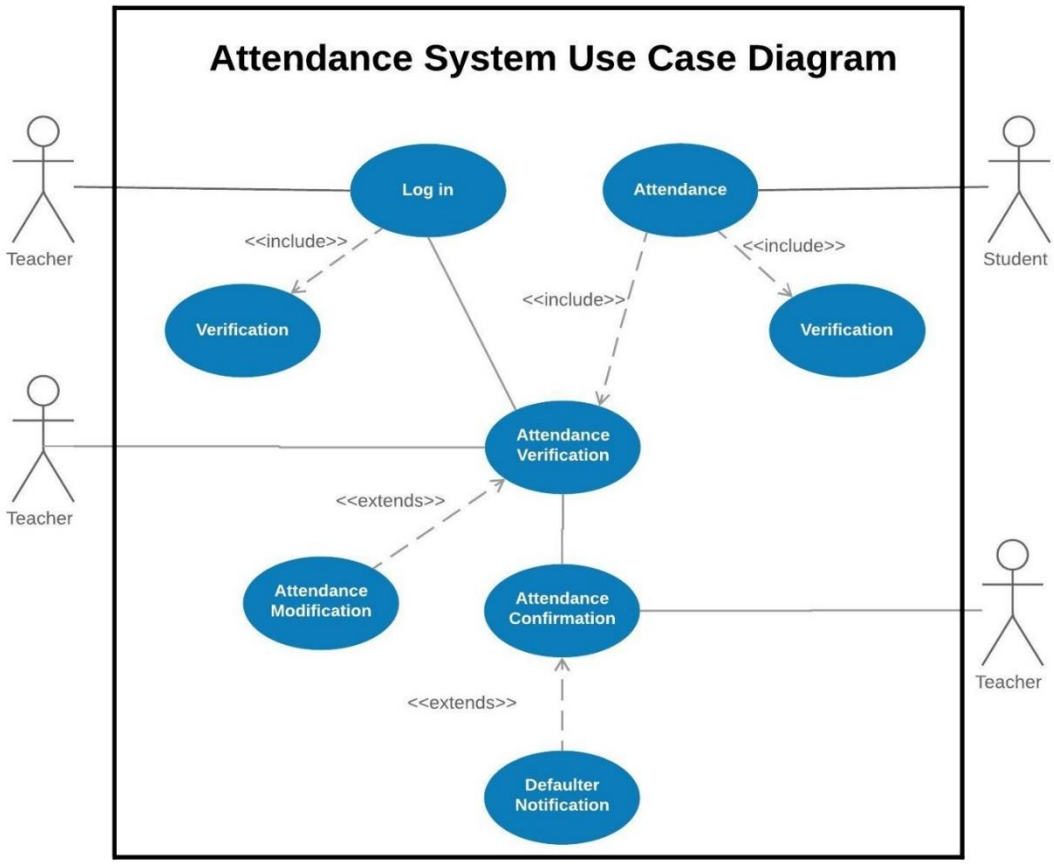


Diagram 7 shows the exploded process of "Send Fault Notification". Once the attendance defaulters' information is collected, it generates notifications accordingly using "Generate Notifications" process. From there notification is sent to both teacher and defaulter students using "Send Notification to Teacher" and "Send Notification to Defaulters" process respectively.

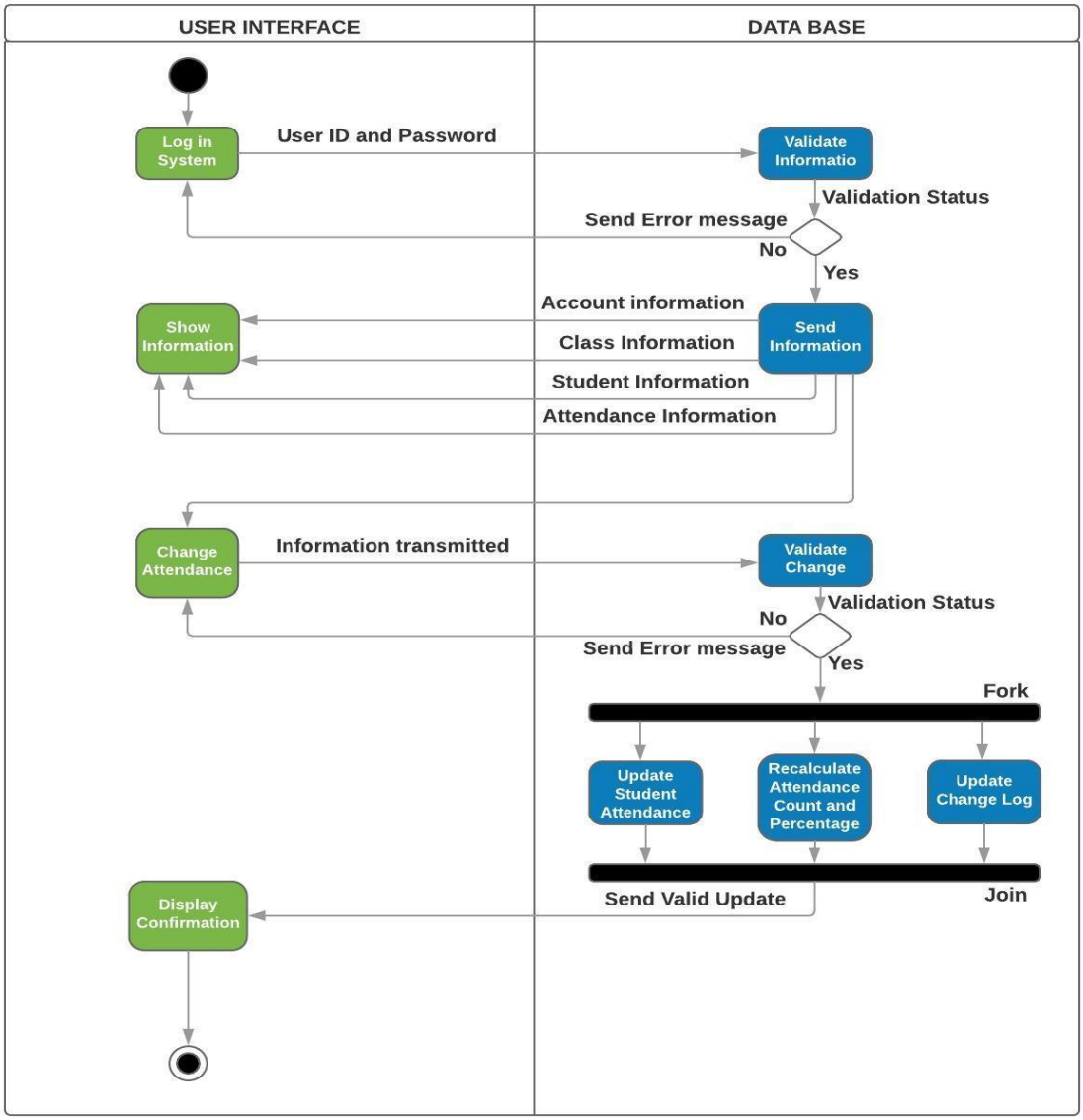
UML Diagrams

Use Case Diagram



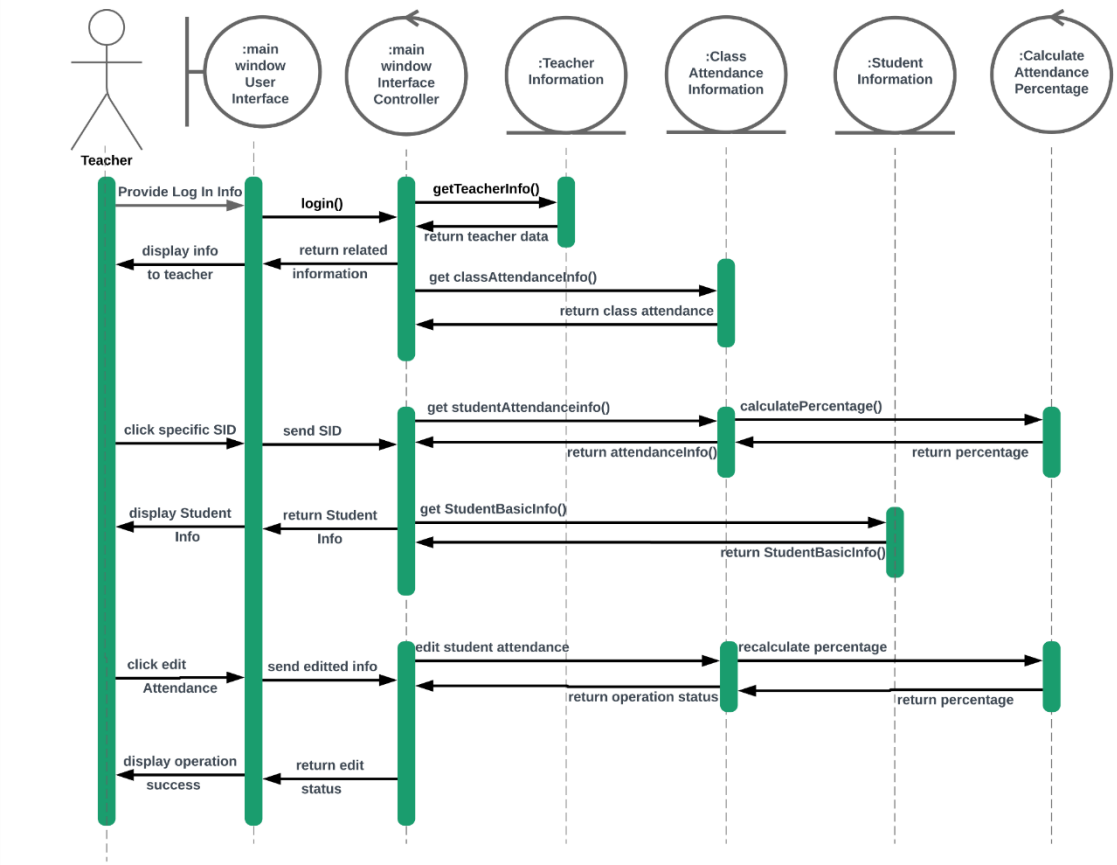
The Attendance System Use Case Diagram shows the teacher logging in via the **log in use case**. The **log in use case** includes **Verification use case** to validate the data inputted. A student can give attendance via the **Attendance use case** which includes the **Attendance verification use case** to check the attendance. The teacher can verify and view attendance via the **Attendance verification use case.** The teacher can also edit attendance information as the **Attendance verification use case** extends **Attendance Modification use case**. Once the attendance information is verified, the teacher can confirm the attendance via **Attendance Confirmation use case.** Once that is done, the teacher will be notified about the defaulter students along with the students themselves. To do this, **Attendance Confirmation use case** extends **Defaulter Notification use case**.

Activity Diagram



The sequence diagram expresses the sequence of activities as they are done in order. The activity diagram starts with the user, in this case the teacher, logging into the system. The teacher inputs data into the user interface which is transferred to the data base for validation. If the user and password is not valid, then an error message is thrown back to the user interface which it displays to the user. If the validation is successful, then the teacher is allowed to view and edit the attendance. The teacher enters the changed information into the UI and the UI passes it to the data base. The data base checks whether the change is valid or not. If it is not, then it returns an error to the user interface, which informs the user of the error in changed data. Otherwise, when the changed data is valid, 3 things happen simultaneously. The edited information is updated, attendance count and percentage is recalculated and a change log is updated. Afterwards, the data base returns a confirmation to the user interface. Thus the user is shown the confirmation which is the end of the activity diagram

Sequence Diagram



Sequence diagram description

Our attendance system sequence diagram consists of the following classes:

1. **Main window user interface:** This is a boundary class tasked with providing a simple user interface to the user who is the teacher.
2. **Main window interface controller:** This is a controller class which handles all the requests generated by the user interface class and responds accordingly.
3. **Teacher information:** This is an entity class. It holds information regarding a teacher's account, courses and sections s/he teaches in.
4. **Class Attendance Information:** This is also an entity class. It holds attendance information for a particular course of a particular section.
5. **Student information:** This entity class holds basic information of a student such as name, address, CGPA, contact information etc.

6. **Calculate attendance percentage:** This controller class is responsible for calculating the percentage of attendance for any student.

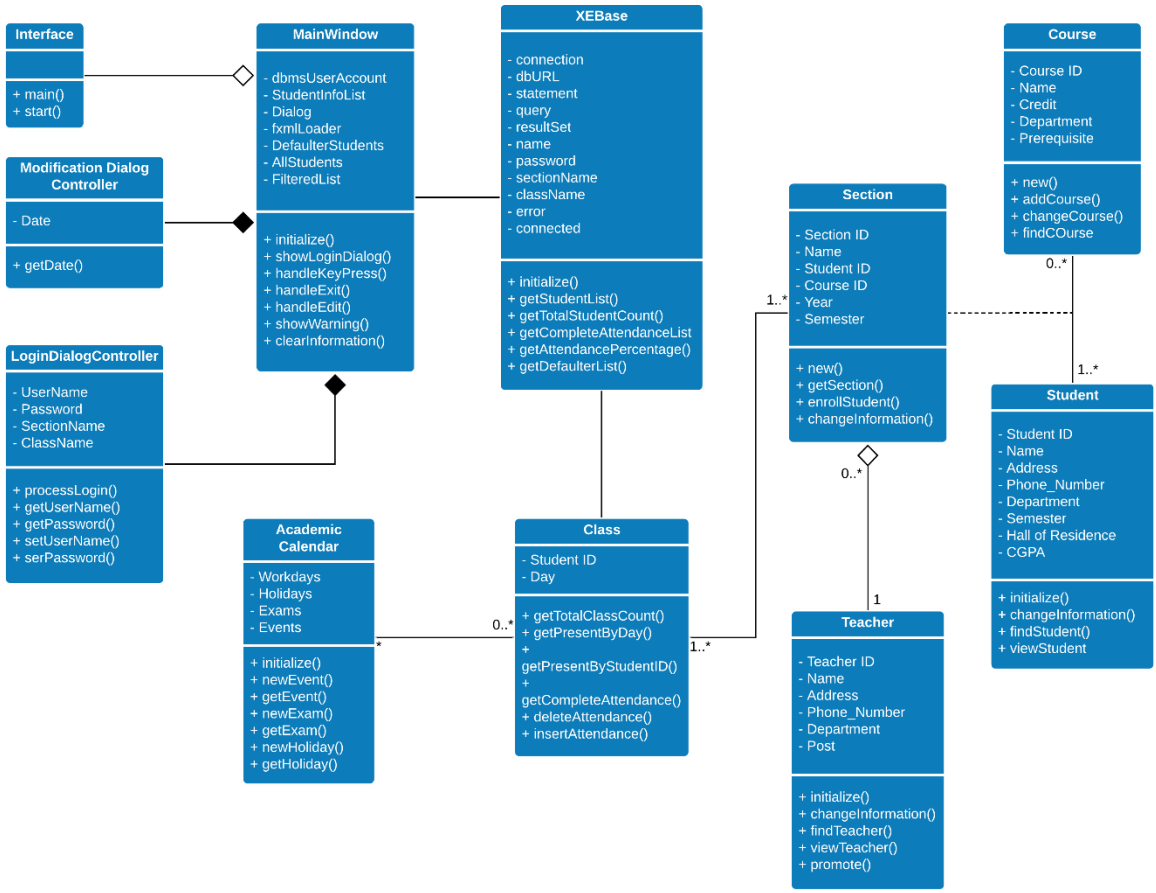
The sequence diagram shows the following activities.

The 1st activity of is that of the teacher logging in. The teacher will to log in to his account via the main window user controller class. The user interface will pass the log in request with the user name and password to the main window controller class. The controller class will then check the information provided with that in Teacher Information class. If a valid match is found, then the proper teacher account information will be returned. The user interface class will also fetch the attendance information for the classes the teacher takes from class attendance information class. When all the data is ready to be displayed, it is passed to the user interface class.

The 2nd activity is that the teacher viewing a student's basic and attendance information by clicking on any student ID. When a specific student ID is clicked, the user interface requests the controller to fetch information of SID. The controller asks the class attendance information class for the attendance information of that SID. The attendance information class then sends necessary info to the controller class Calculate Attendance Percentage and receives the percentage of that SID. The total information is then passed to the main interface controller class. The interface controller also fetches the basic information of that student from student information class and then passes it to the user interface class to show. The teacher may view the information of the defaulter students in this way.

The 3rd activity is the teacher editing the attendance of a student. The user interface passes the edited information to its controller class. The controller class then passes the information to the class attendance information class for saving. The class attendance information class recalculates the attendance percentage via the calculate attendance percentage controller class and saves the value.

Class Diagram



The class diagram consists of 11 classes of three types, Entity, Control and Interface/ Boundary class. Teacher, Student, Course, Section, Class and Academic Calendar are entity classes. XEBase, Main Window, Modification Dialog Controller and Login Dialog Controller are control classes. Interface is the boundary class. All the variables and methods of a class are shown in the diagram.

Student and Course forms a many to many relationship between themselves and Section works as a junction table between them. Every section ‘has a’ teacher, so it is a part or aggregate collection relationship and relation is many to one between section and teacher. Academic calendar works as a tracker for the dates of all academic activities. All the entities mentioned up to now are already implemented in IUT and these are just taken as place holder replicas for development purpose. Class is an entity class that holds the attendance information for students for a specific section and course. It uses Student ID and Day as foreign keys from Section and Academic Calendar and keeps track of attendance of the students. Up to now, all the classes are implemented in backend and they are built using SQL, Oracle SQL to be precise and run in Oracle XE 11g local server for debugging and testing purpose.

XEBase works as a control class between the backend and frontend and connects them together. At the same time, it works as the control for database and performs various calculations on the attendance data sets. Main Window works as the control class between XEBase and the Interface. It provides the necessary information to Main Window class and responds data base method calls. It is in an aggregate composition relationship with Modification Dialog Controller and Login Dialog Controller classes and them, in together, forms the actual Main Window class.

Finally, Interface class is the boundary class that stands in the frontend of the system works as the graphical user interface for the user. It calls the methods of Main Window class, which handles the calls and display using Modification Dialog Controller and Login Dialog Controller classes and in turn, calls the methods of XEBase class. Then XEBase communicates with the entity classes and retrieves and stores necessary information.

Current Stage of the Project and Prototype

Currently, only the first prototype of the project is developed and it is a prototype with selected features. Currently, only the teacher's part is operable, that means it can insert, remove, modify and view attendance of students and be filtered using date or student ID.

Conclusion and Future Works

In this project, we attempted to introduce a cost efficient but effective attendance system that would solve the current issues in the attendance system of IUT. Due to the lacking of time and various other works, we only managed to develop prototype for one part of the system. In the future, we wish to complete the whole system through continuous developments and prototyping and eventually, put it for a test run. At the same time, in the big picture, we wanted to develop a complete app system for IUT that would let students do all of their various activities such as attendance giving, class schedule management, teachers' and course's information, online notice board, academic calendar and many others while managing daily and academic life. So as a continuation of this project, we wish to achieve such a system in the future.