1 November 2021

Z5308045 Caroline – Z5320443 Adam Lindsell – Z5308201 Minwoo Lee – Z5308142 Noah Hayes – Z532044 Joel Koshy

Management System Report

ZEIT2105 SAD Assignment 2

# Introduction

The aim of this report is to outline the processes taken to design and implement an app that streamlines student attendance and achievement recording processes for a new cyber-education start up. It is composed of team management processes, a requirement report, database design processes, deliverables inclusive of a use case diagram, relational database schema, activity diagrams, class diagram and windows navigation diagram, app implementation processes and a discussion.

# Git:

Throughout the execution of this project, GitHub was used as a primary means to share and store the artefacts created by group members. The following link opens the Git that was used: <https://github.com/AdamLindsell/sadAss>

As displayed in *Table 1,* Each member took primary responsibility for different tasks within the project and using Git we were able to further collaborate and enhance each other’s work.

|  |  |
| --- | --- |
| Group Member | Contributions |
| Adam Lindsell | **Majority of the application design and implementation**, aided with database design, created the WND, report write up. |
| Minwoo Lee | **Majority of diagrams,** advised on database and application design, project manager - schedule keeping, report write up. |
| Caroline Taylor | **Database Design**, Diagrams (Relational Database Schema), Programming (created complex queries and aided app implementation), report write up. |
| Joel Koshy | **Requirements gathering** |
| Noah Hayes | **Requirements gathering** |

# Requirements

## Purpose

The purpose of this requirements document is to discuss the development opportunities of the proposed badge tracking app. It will significantly minimise the workload of the teachers and provide better tracking of the achievements accomplished by students. This app will use a simple database that will hold all relevant data for the teachers, allowing them to edit and monitor their students details and badges.

## Background

The current curriculum at the school requires extensive human input from the teacher to account for and record the badges/activities completed by students. This is not only unsustainable on a bigger scale as it lacks any level of automation, but it is also fully dependent on the teacher allowing for more human error. The proposals for the app will mitigate this and can be operational on a bigger scale.

## Assumptions

The following assumptions have been made based on the collection of requirements:

1. There is no existing technology and processes in relation to software that the school runs. Therefore, these will all be developed and created in the project.
2. The app will have no cultural sensitivity and is suitable for all ages 16+; since 16 and older will be assumed to be responsible for using the app.
3. The school is in an English-speaking country and as such the default language will be English.
4. The finance and scheduling of this project is separate and completed.
5. Training for the app is done internally without any additional costs.
6. A member of the school (tech department) will need to be taught how to use the app. They will then oversee promulgating this knowledge to the rest of the staff. The user also should not be expected to understand the controls as the programmer does.
7. The quality of the app will be measured by the fulfillment of the requirements.
8. Feedback from the school to improve the quality/experience of the app
9. Once the app has been built and is functional, the support and maintenance will be taken care of. This includes potential updates for any bugs, as well as any updates requested by the school.
10. The hardware used by the school can operate the app without much strain on the processor and memory.

## Functional requirements

### Implemented

* Add student full name, DOB and group activity attended
* Record attendance and date of weekly meetings or weekend intensives
* Record parts, topics or tests completed
* Search by name for a student and display their personal details and achievements
* Edit student details
* A schedule to add upcoming meetings/activities

### not implemented

* Secure personalised login before entering the app.
* Notify teaching staff when a student has completed a badge (7 compulsory tests + 3 optional)
* Weekly updated leader board of name of students with highest number of badges completed

## Non-functional requirements

### IMPLEMENTED

* Simple search interface for students displaying student name and details. Allows for display for all students in the curriculum (just names). Once searched allows editing of any details of the student.
* Easy to use home interface for students/search, record and schedule.
* Schedule allowing for display and editing of upcoming activities/meetings with name, date/time and teacher running it.

### NOT IMPLEMENTED

* Login password determined by teaching staff required to use the app.
* Simple notification popping up on the appropriate teacher’s device displaying the student's name completing a certain badge.
* Leader board of top 10 students with highest number of badges completed displaying number of badges completed for each badge level and overall total.

## Constraints

### Storage and cache sizes

Potential hardware constraints can be a causal factor to consider. Although the amount of memory available is not the key constraint, exploitation of memory may lead the operating system to ask the application to shut down or sacrifice cached data or slowing program execution. To mitigate for this, the application will need to minimise its power and memory footprint while considering performance during this process.

### Budget

Should the app be made by professionals; a budget would need to be set for the app development. This includes, but is not limited to - Analysis and Planning, UI/UX Design, App Development, Testing, as well as Development and Support.

### Network issues

A possible constraint with using the app is having multiple teaching staff having access to the app. Conflict can occur with the data if multiple teachers are editing the details on the app at the similar time.

### Network connections

The app will need to be optimised to use its network efficiently. Longer latency periods for wireless connections can influence how quickly the user will receive information. This also ties into the update rate where the information may be required at multiple devices simultaneously.

### Screen size and OS

If the app is to be built for a phone, the screen sizes are much smaller and offer a limited canvas to play and interact in. iOS also has fixed screen sizes and resolutions to plan for, as opposed to Android, where there are more variations. Due to the limitation in screen size, the information presented will also need to be optimised to only show relevant information. The UI design will also need to accommodate for this to allow users to perform tasks easily and access information quickly.

## Risk considerations

### Insufficient authentication and authorisation control

Missing or poor authentication schemes can allow third parties to anonymously execute functionalities within the application. Unlike web applications, mobile apps do not need the use to be always online during their session. Mobile apps these days may have uptime requirements when the app is not being used. This requires offline authentications, which may pose security risks that the developer will need to consider.

### Poor encryption

Apropos the above point, devices and data will need to be encrypted properly. Although the app is only used for a school level, the data will need to be encrypted. Poor encryption can lead to data loss and all of the repercussions that follows from the loss of that information. Should this happen, it can reflect poorly on those who designed the app and legal cases could even be filled. This can be mitigated for with the implementation of modern encryption algorithms and by having layered encryption.

### Adding too many features

Overloading the information available/provided to the user can result in a negative connotation to the app. This is an easy pitfall to take for the programmer and should be mitigated for by sticking to the cardinal commandment in app development - “Keep it Simple, Stupid”.

### Miscellaneous issues

Among some of the potential risks in this system, the main one remains to be with technology related issues. Any fault in the data storage servers will likely cause the entire system to be rendered not working until it is rectified.

# Database Design

## Relational Database Schema

Diagram

Description automatically generated

As can be seen, many relationships exist between the tables within the database inclusive of one-to-many relationships and a many-to-many relationship between *Schedule* and *StudentAttendance*. Additionally, each table is in 3NF as each of the following conditions hold:

1. There is a primary key
2. There are no repeating groups
3. Each attribute only holds one value per entry
4. All non-key attributes are dependent on the table’s primary key
5. The values in each non-primary-key column depend on values in only the primary key columns.

## Table Description

The following table will describe the purpose of each table and define the primary keys

|  |  |  |
| --- | --- | --- |
| Table Name | Use Description | Primary Key |
| Staff | Holds the information for each staff member | StaffID |
| Schedule | Each ScheduleID uniquely describes which staff member is teaching which parts to which group of students in which week. | ScheduleID |
| StudentAttendance | Records the ID of students that attend certain weeks so that the parts taught in the week can be related to the student. | Composite Key  (StudentID, weekNum) |
| Student | Holds information relating to each student inclusive of personal details and which group they belong to. | StudentID |
| Parts | Holds the information for each part that then uniquely belongs to a topic. | PartID |
| Topics | Holds the information for each topic that then uniquely belongs to a test. It also holds a value for how many parts belong to it. | TopicID |
| Tests | Holds the information for each Test that then uniquely belongs to a badge. It also holds a value for how many tests belong to it. | TestID |
| Badges | Holds badge information inclusive of which level and achievement type it is. Does not need to have a set value indicating how many tests belong to it as this is constant for all badges. | BadgeID |

## Database/query design Assumptions

The following outlines the assumptions made when creating the database and queries, based on the projects requirements, and the projects key functionality and scope:

1. Only tracking of the weekly student attendance was required, not the attendance for weekend activities.
2. As staff would be unable to monitor parts completed in their own time, only the parts completed by students at the weekly meetings would be confirmed as completed.
3. If a student went to the weekly meeting, they finished the parts scheduled as being taught at the meeting.
4. The students complete the tests within the badges in order up until test 7 which is the last mandatory test. After that they can pick any 3 tests.
5. A student only completes the 10 required tests (7 mandatory, 3 additional) for each badge and no more than this.

## QUERY DESIGN

A picture containing text

Description automatically generatedIn order to access information in the database from within the app, instructions (SQL queries) were programmed into the code as shown further in the app implementation section. Most of these queries were simple SELECT/INSERT/DELETE queries such as in the addStudent method; however, there also queries consisting of subqueries and inner joins, one of which will be broken down in this section.

The query as displayed in aboveis used to list the badges that a particular student has completed. It works by obtaining a list of weeks that the student has gone to, then a list of the parts completed in those weeks. After this, it counts the number of parts completed for each topic (Using COUNT and GROUP BY), and if the number of parts completed is equal to the value numParts in the topic table, it is added to a list of complete topics. This pattern is repeated to obtain the lists of completed tests and badges. Additionally, for a badge to be complete, the number of tests has to equal 10 as based on the stated assumptions this would mean that the 7 required tests, and 3 additional tests have been completed.

# Deliverables

## Use case diagram

Diagram

Description automatically generated

## Activity diagrams

### Add Student

Diagram

Description automatically generated

### Update Student

Diagram

Description automatically generated

### Search Student

Diagram

Description automatically generated

### Delete Student

Text, chat or text message

Description automatically generated

### Mark Attendance

Diagram

Description automatically generated

### Display Schedule

Diagram

Description automatically generated

### Add Staff Member

Diagram

Description automatically generated

### Assing staff to part

Diagram

Description automatically generated

### Show student

### Diagram Description automatically generated

### View Badges

### Diagram Description automatically generated

## App Implementation Diagram

Diagram, schematic

Description automatically generated

# App implementation

## Add Student:

As is seen in the screenshot below, when adding a student, the user is required to input their first and last name, their DOB and the group (or day which the students attends) that they are a part of. The DOB is entered in a specific format which allowed us to format our queries in the search student function to search by age instead of by DOB.

Text

Description automatically generated

## Record Attendance

This method also records the topics, parts and tests completed as they are due to complete in accordance with the schedule.

Text

Description automatically generated

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

## Search by name

As seen in the below screenshot, when the query returns a result, the details of the credentials of the entered student, along with the completed tests of the student, represented as badges through the calculateParts() method.

Text

Description automatically generatedText

Description automatically generated

# Conclusion

To conclude, we were able to design and implement an app that satisfied the outlined requirements of student attendance and achievement record keeping. Whilst more than the minimum requirements were completed, the were still more that could have been implemented such as the leader board had there been more time to complete the project. However, meeting the set time was more important for this project than adding further additional features. By assigning tasks to each team member, the team was able to work more effectively and get more accomplished in the time frame.