TIMMY'S TUTORS
USER MANAGEMENT SYSTEM

Jack Gadsby, Brayden Munro, Jordan Lovaszy and Reuben Lyndon

SAD Assignment 2

TASK 1

Requirements Report

Requirements Report

Author: J. Lovaszy, R. Lyndon, J. Gadsby, B. Munro   Date:02/11/19   Version:1.1

Introduction: This requirements report outlines the functional and non-functional requirements that need to be considered when designing and implementing our student management system.

Purpose: The purpose of this document is to outline what needs to be considered when designing and implementing the student management system.

Background: We are automating and digitalising a new cyber education start-up's student management system. The system specialises in micro-credentials where students participate in various events and complete a range of activities in order to achieve badges. We are tasked with designing and implementing an attendance record and badge tracking system (student management system) by storing details in SQLite.

Assumptions: We are assuming that we have the ability to make any architectural decisions around the design and implementation of the student management system, as long as they are not outside the scope of the initial task sheet provided by the principal, Timmy.

Constraints: The main constraint during the requirements gathering phase was that we were limited to only the documentation provided by principal, Timmy. We had no access to question, interview or observe the current use of the system and the operatives of.

References: There is only one reference for this report, that being the assignment 2 task sheet provided by principal, Timmy.

Methodology: For the design and implementation, we used a mixture of SCRUM and KANBAN methodology. This worked well for how we divided up our workload, providing the designers and implementers autonomy and freedom, whilst also having reviewers to ensure quality control. GitHub provided great KANBAN services, with the added element of peer reviewability incorporating SCRUM techniques.

Functional Requirements:

1. **Manage Students**
   - 1.1. Add, edit and view students
   - 1.2. Manage meetings and intensives involving students
   - 1.3. Track what badges and what topics the students have done
   - 1.4. Implement a leader board
2. **Manage Badges**
   - 1.1. Add, edit and view badges
   - 1.2.

Non-Functional Requirements:

1. **Operational Requirements**
   - 1.1. The system will operate in a windows environment through a console
   - 1.2. The system should automatically update and back up
   - 1.3. The system should be compatible and able to connect with an SQLite3 database
   - 1.4. The system will record attendance on a weekly basis to the SQLite3 database
   - 1.5. The system will track and record the students' progress
2. **Performance Requirements**
   - 1.1. The system will store new data in 2 seconds or less
3. **Security Requirements**

1.1. Only the teachers have the authority to create, edit and delete attendance records, students and meetings/intensives
1.2. Every teacher's password must be in accordance with the latest ASD security recommendations of minimal 13 characters long including upper- and lower-case letters, numbers, and unique characters
1.3. There is a time out of 10 minutes for each user

4. Cultural and Political Requirements

1.1. This application needs to meet maintain cultural sensitivity and apolitical, to ensure that it is all-inclusive.

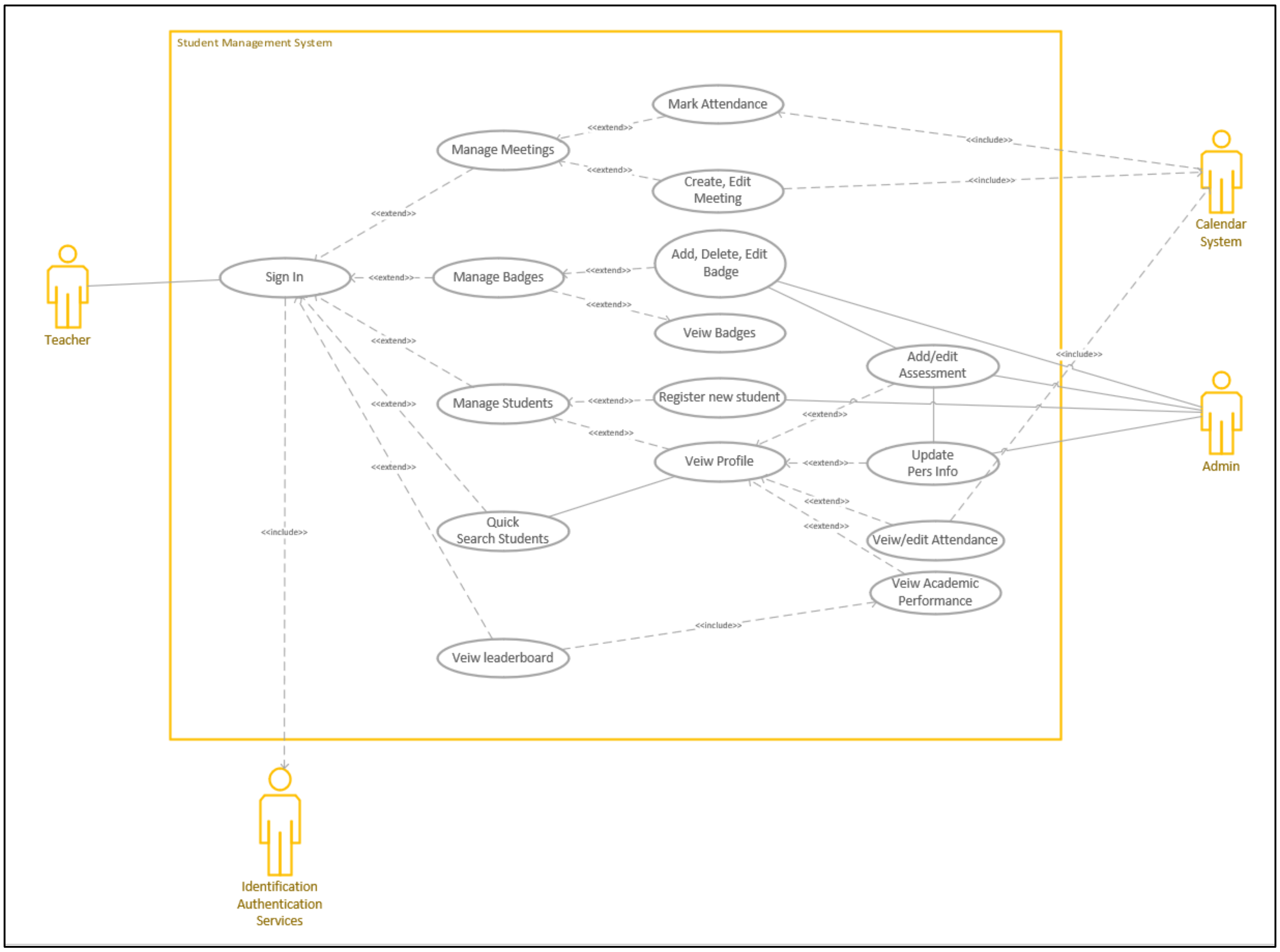
Sign off

I, __________________________, here-by agree with the above requirements list and understand that by signing this document give permission for Timmy's Developers to design and implement the student management system.

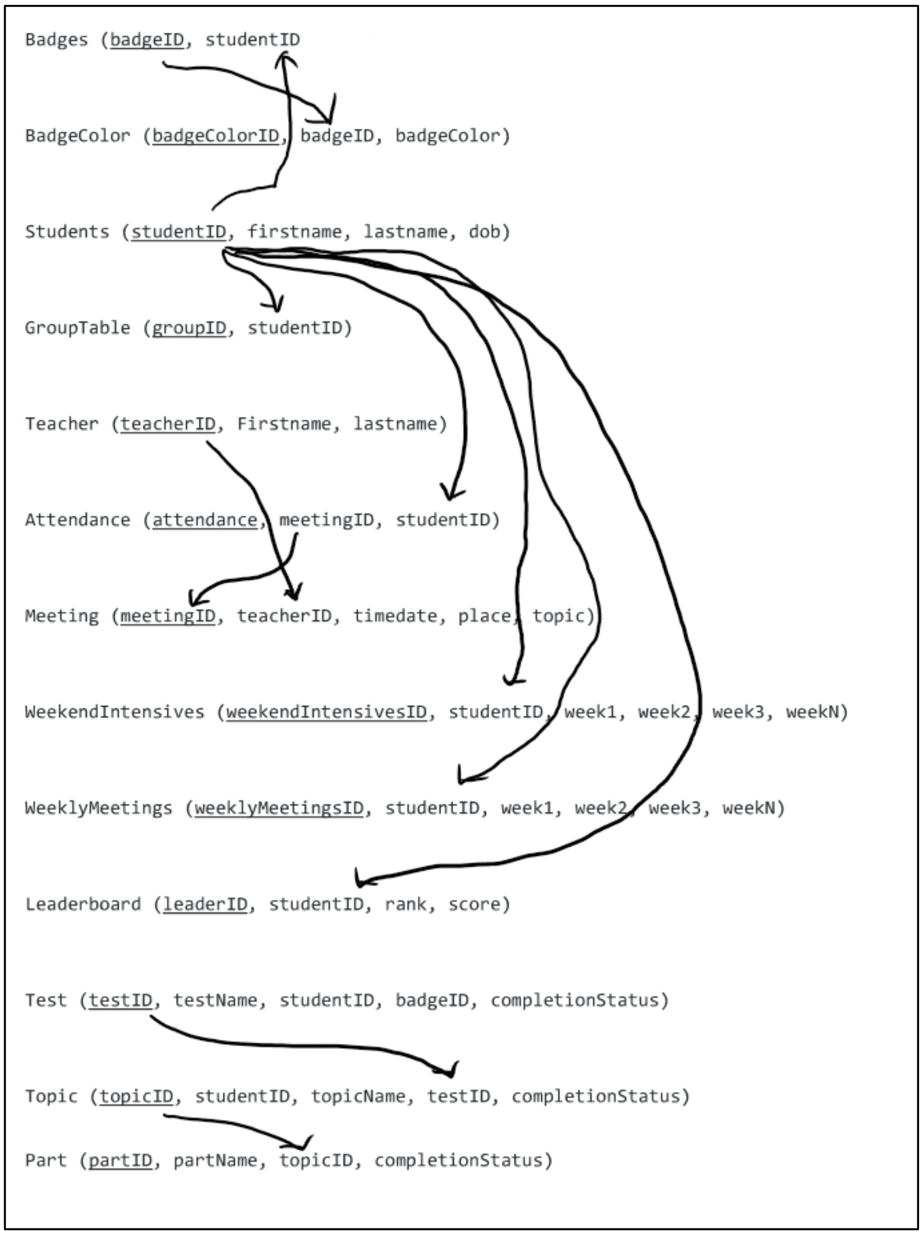
X ______________________________	X ______________________________
Name	Name
Project Manager	Client

TASK 2

Use Case Diagram



Relational Database Schema



Badges (<u>badgeID</u>, studentID

BadgeColor (<u>badgeColorID</u>, badgeID, badgeColor)

Students (<u>studentID</u>, firstname, lastname, dob)

GroupTable (<u>groupID</u>, studentID)

Teacher (<u>teacherID</u>, Firstname, lastname)

Attendance (<u>attendance</u>, meetingID, studentID)

Meeting (<u>meetingID</u>, teacherID, timedate, place, topic)

WeekendIntensives (<u>weekendIntensivesID</u>, studentID, week1, week2, week3, weekN)

WeeklyMeetings (<u>weeklyMeetingsID</u>, studentID, week1, week2, week3, weekN)

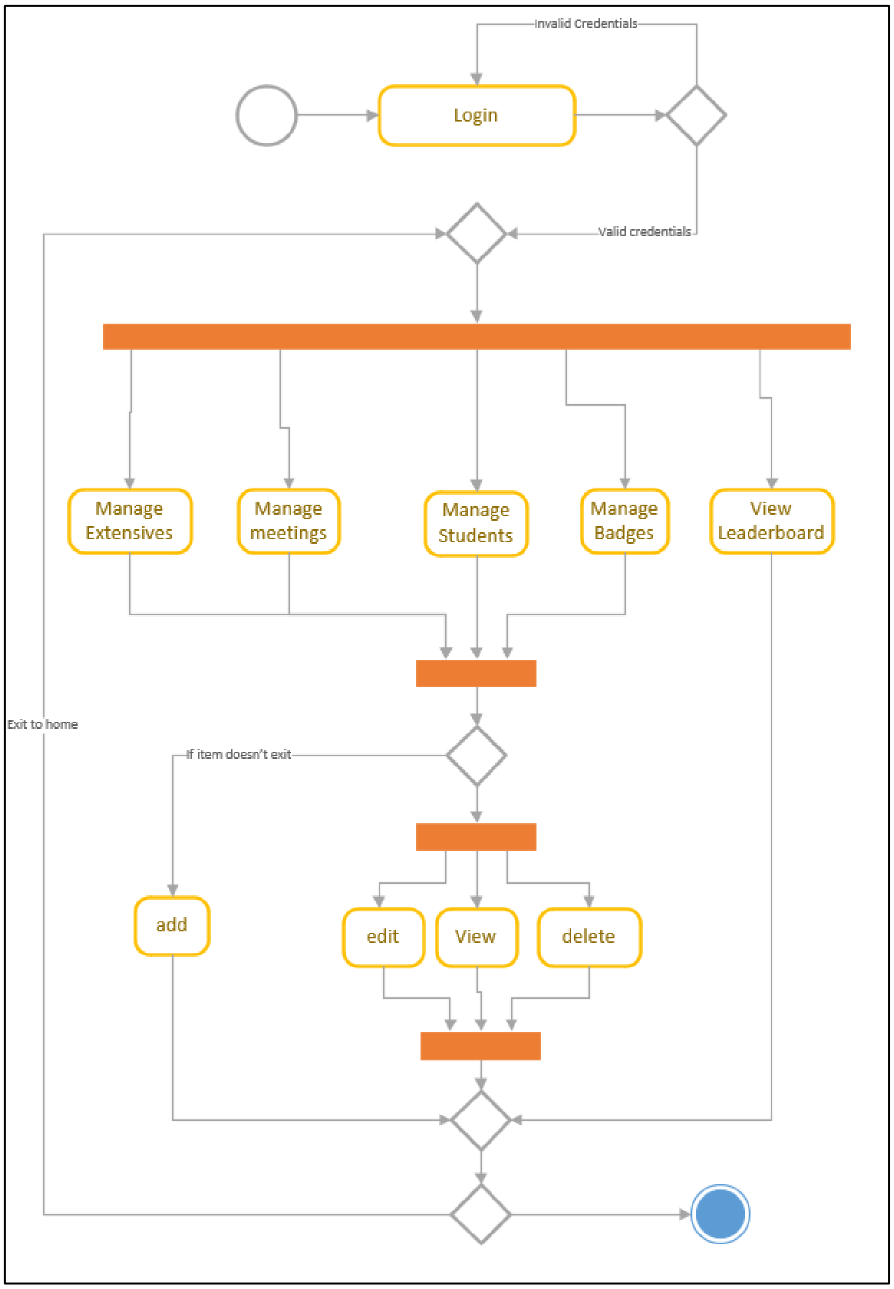
Leaderboard (<u>leaderID</u>, studentID, rank, score)

Test (<u>testID</u>, testName, studentID, badgeID, completionStatus)

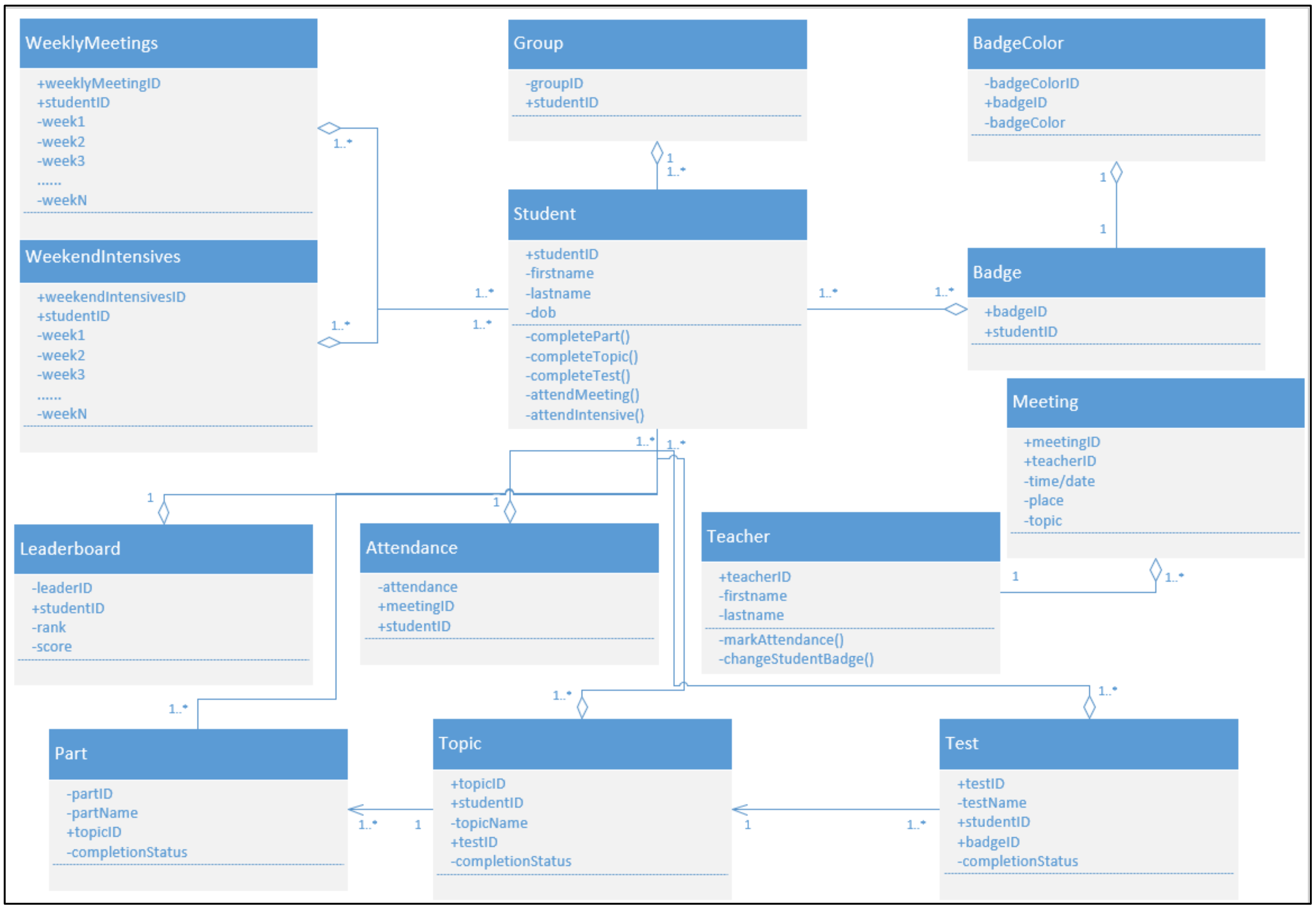
Topic (<u>topicID</u>, studentID, topicName, testID, completionStatus)

Part (<u>partID</u>, partName, topicID, completionStatus)

Activity Diagram



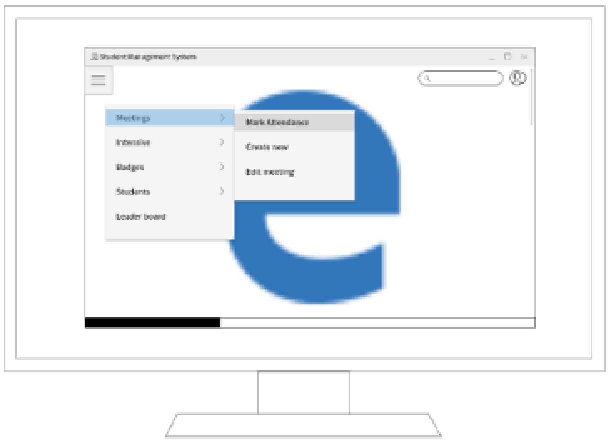
Class Diagram

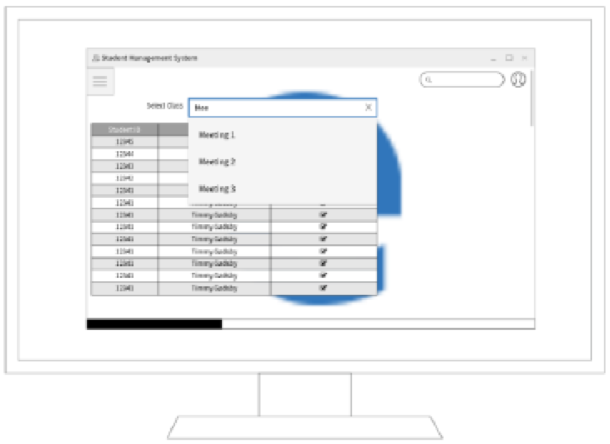


Windows Navigation Diagram

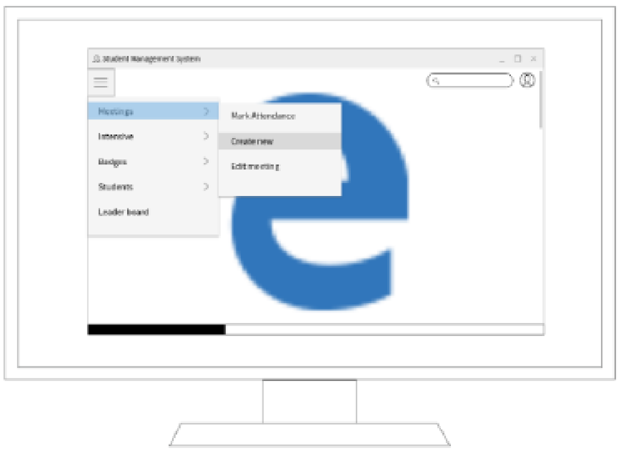
The windows diagrams flow from left to write, showing the progression of selecting an activity or function and using it. The design flows well, allowing the user to have control over the system with little training involved.

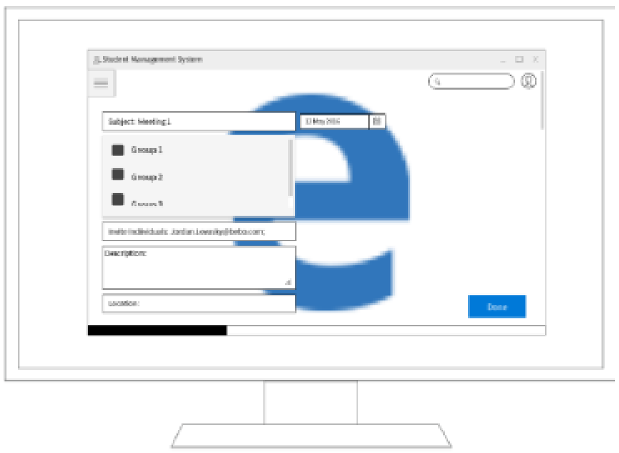
1. Marking attendance for a meeting: you have a dropdown search box that allows you to select the appropriate class and then mark attendance.





2. Creating a new meeting:

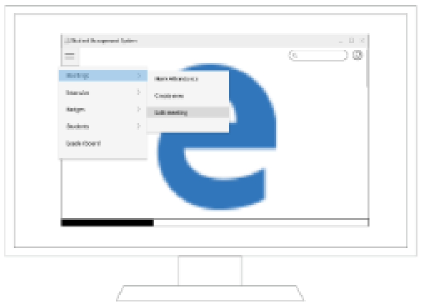


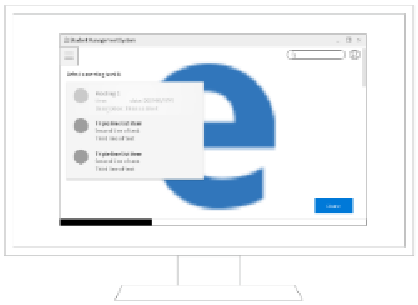

Subject: Meeting1

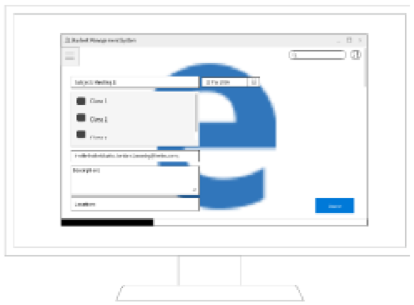
12 May 2016

Invite individuals: Jordan.Lewsky@bebo.com;

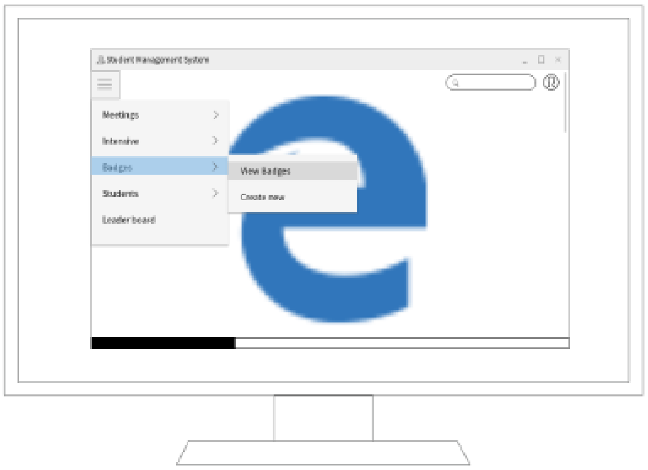
3. Editing a meeting: here you select the meeting you wish to edit before being directed to a page like that of when you're making the meeting.

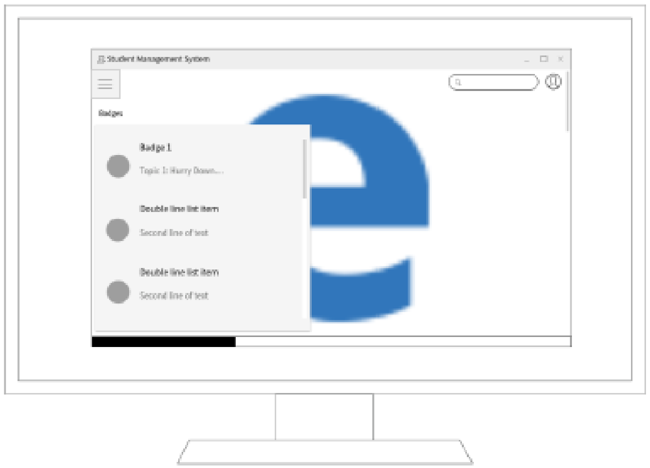




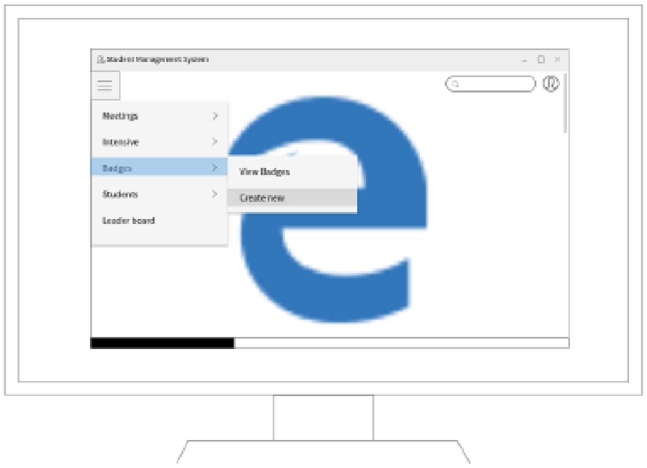


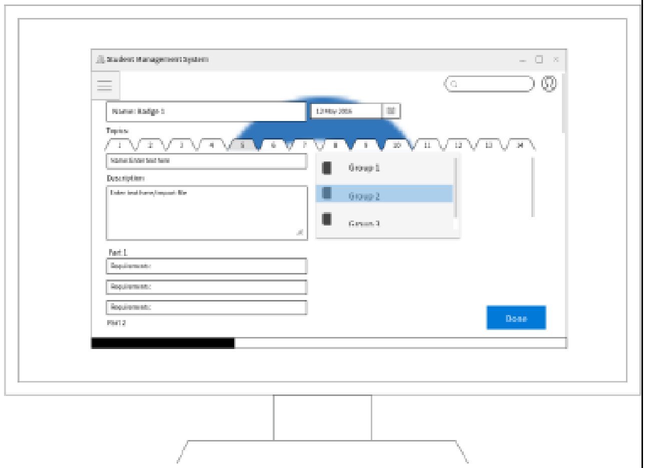
4. Intensives are extremely similar to that of meetings, where instead of meeting, you have intensive. Thus it was redundant to create window diagrams for them.
5. Viewing badges: here you select a badge to view, and then the information regarding the badge – name, type, groups, parts, topics and requirements – are shown.

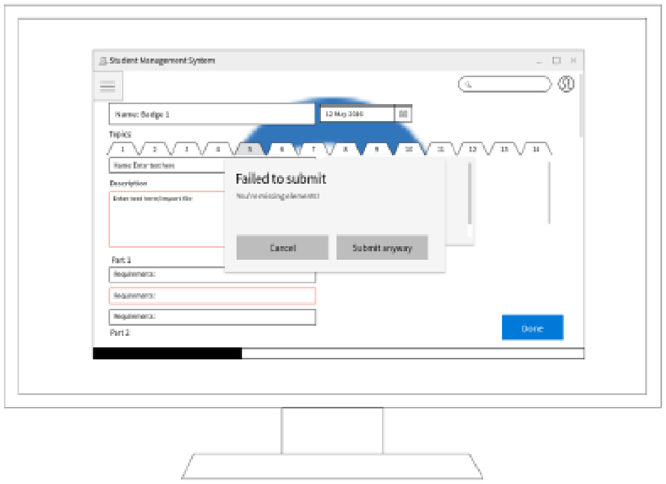




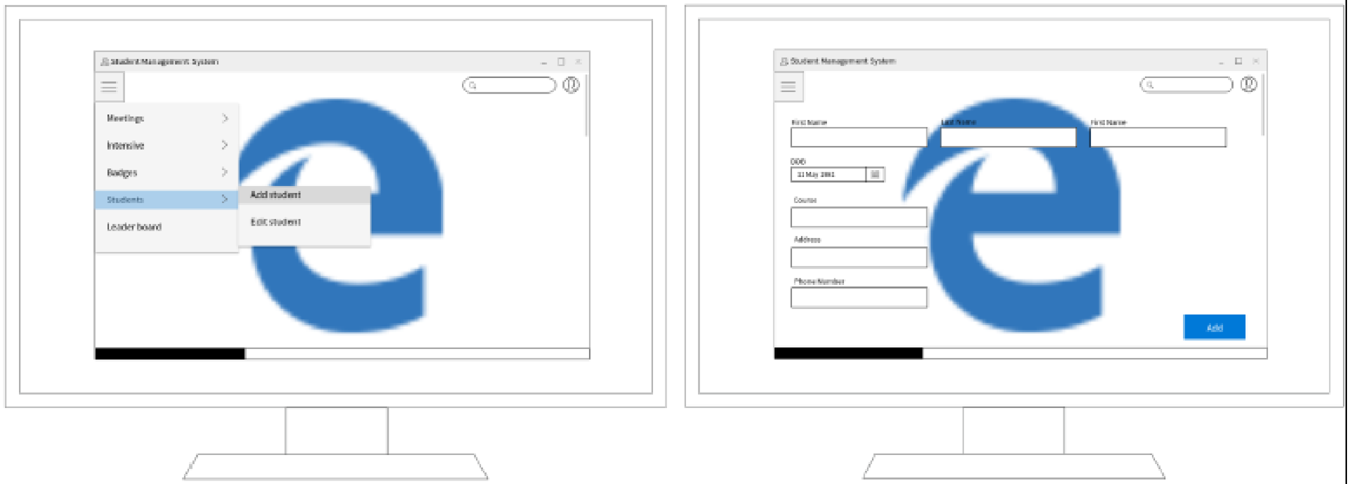
6. Creating a new badge: here you have tabs in which the teacher must work through, filling in all the necessary information required to sufficiently supply students with information and requirements to achieve the badge. There is a warning functionality to ensure this happens.



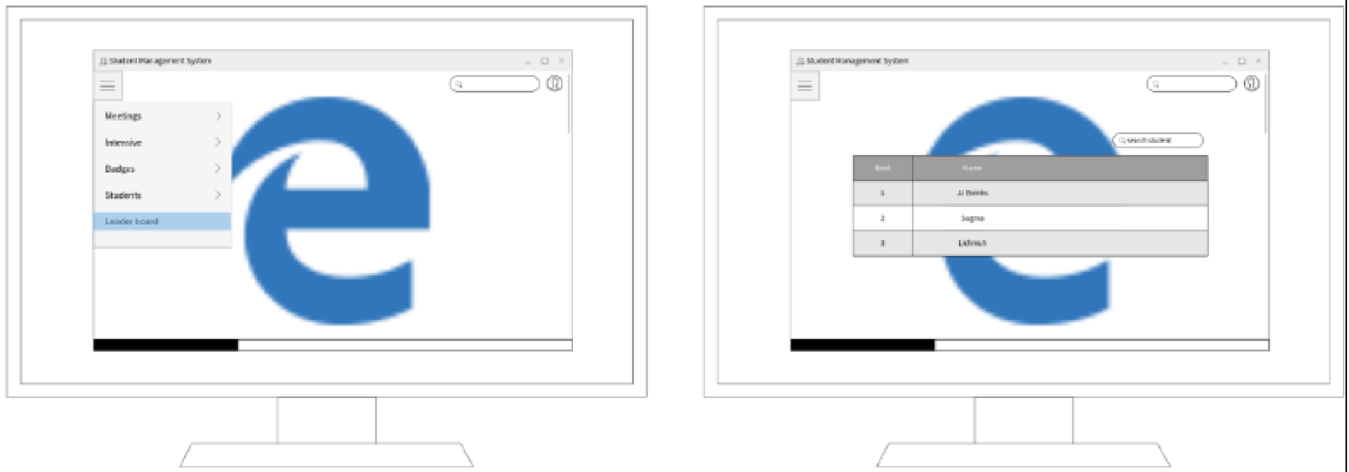




7. Adding a student: user selects to add a student and are taken to a new page where they input basic student information. Editing a student takes the user to the same page.



8. Viewing the leader board:



Rank	Name
1	JJ Banks
2	Sagna
3	Lathrush

TASK 3

See GitHub for the completed code.

TASK 4

As per individual submission.