Integrated Group Project

NA3

Charlie Howes (CH), Lewis Allen (LA),
Adam Howes (AH), Ben Ashing (BA),
Constantinos Ioannou (CI)

March 12, 2017

Contents

1 Planning

1.1 Gantt Chart

1.1.1 Proposed

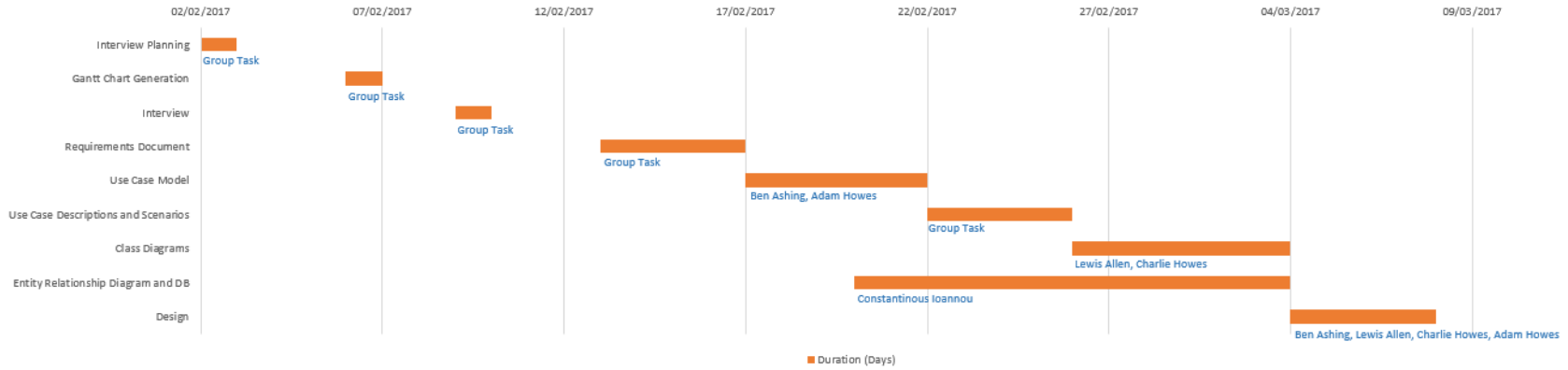

Figure 1: Proposed Gantt Chart

1.1.2 Actual

1.2 Minutes

2 Requirements

2.1 Document

1. Business Requirements
   - B1. The Planning should be completed by March 13th
   - B2. The finished product should be delivered in May.
   - B3. The software will be adopted and used by all staff.
2. User Requirements
   - U1. Users must be able to log in using a user name and password.
   - U2. Users must be able to log out
   - U3. The user must be able to personalize their view.
   - U4. The user must able to easily view calendars for daily, monthly and yearly schedules.
   - U5. The user must be able to set recurring appointments.
   - U6. Staff must be able to create groups.
   - U7. Staff must be able to modify groups
   - U8. Staff must be able to add other staff/groups to events.
   - U9. The user must be able to cancel appointments at any time within a session.
   - U10. The user must be able to use a search feature to find other users.
   - U11. The user should be able to make event requests.
     - U11.1. Users should be able to receive event requests.
     - U11.2. Users should be able to accept event requests.
     - U11.3. Users should be able to decline event requests.
3. Quality Requirements
   - Q1. The application must use symbols to clearly show changes in events.
   - Q2. The application should notify the user in any changes to events such as cancellations.
   - Q3. The application must implement optimized loading times.
4. Functional Requirements
   - F1. Only members of staff should be assigned accounts.
   - F2. The architecture of the project should support different platforms.
   - F3. A method must exist which allows staff to be given administration rights to form an administrative team.
   - F4. Administrators must be able to verify accounts.
5. Non-Functional Requirements
   - NF1. The user should be able to complete any single task with a minimum of eight actions. (LA)
   - NF2. The code needs to be easily maintainable by keeping the code organized, well written, documented and simple. (CH)

NF3. The project must be easily scaled to implement additional features and a larger user base. (CH)

NF4. The program must be executable on the following Operating Systems: (AH)

- Windows 8, 8.1 & 10
- Mac OS
- Linux Ubuntu

NF5. The minimum specifications to run the program should be: (AH)

- Processor: Pentium 4
- RAM: 500MB
- Disk Space: 100MB

NF6. The code must also be able to run on both 32 and 64-bit Operating Systems and be able to view events when not connected to the Internet. (AH)

NF7. All personal data must be fully secure through encryption and hashing. (BA)

NF8. The project must be adaptable in the future for mobile implementations. (CI)

2.2 Stakeholder Diagram

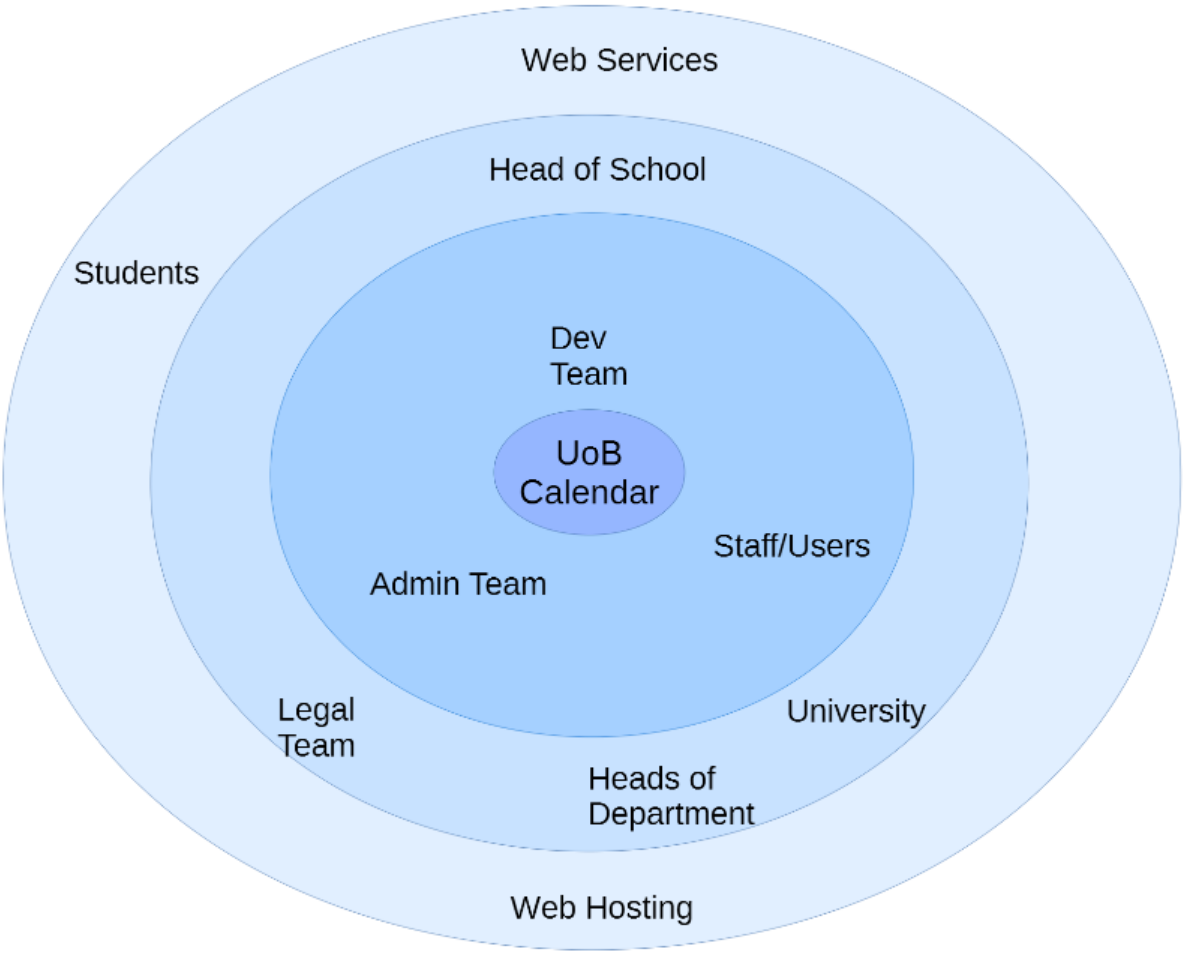

Figure 2: Stakeholder Diagram

3 Use Case Model

3.1 Actors

3.2 Diagram

3.3 Case Descriptions

3.3.1 Login - CI

Use case: Login

Actors: User (Primary)

Goal: TODO?

Precondition: Time, Date & Location from customer (via phone).

Success postcondition: The user fills in their login credentials successfuly and logs into the system.

Failure postcondition: The user does not fill the correct login credentials.

Trigger: The personal calendar of the user is displayed

Main success scenario:

1. User wants to login
2. User inserts credentials into the system
3. User clicks the login button
4. System checks the input credentials.
5. System fetches information and displays their personal calendar

Extensions:

3a User entered invalid data

1. System asks the user to input correct data
2. Restart from 2

3.3.2 Creating an Event - LA

Use case: Creating an Event

Actors: User (Primary)

Goal: TODO?

Precondition: The User has logged into their account.

Success postcondition: The User has the created event displayed on their calendar.

Failure postcondition: The User was unable to create the event.

Trigger: The User clicks the 'add event' button.

Main success scenario:

1. The user clicks the 'add event' button.
2. The User specifies the time, day and duration of the event.
3. The user names the event and gives it a description.
4. The user clicks 'Confirm'.
5. The System adds the event to the user's calendar.

Extensions:

2a The user has input an event time/date and there is a conflicting schedule. (The user already has an event booked for that duration)

1. The System indicates that the user can no longer click confirm.
2. The System informs the user that there is already an event planned for that duration.
3. The user is returned to the event creation screen and is given the option to change to event time/day, along with the other details if required.
4. The user modifies the time/day to a free period.
5. Continue on to step 3.

4 Class Diagram

5 Database

5.1 Documentation

5.1.1 Entities

Event:
This entity will store all the information about the events. The unique ID for the entity is the *Event_ID* and it will be used to determine an event. The entity can hold all the necessary information such as date duration and location. The event is created by a user so the *Host_ID* is used to identify the user that create the event. The option to keep the event private is also available.

Event Request:
The Event Request entity is used store the information about an event request. The unique ID is the foreign key from the Event entity. It stores the response of user to the request as well as date invited and if the user seen the request or not.

User:
All the data of a user are store in the User entity. The unique *User_ID* is used as a primary and is used to log in in to the system with the required password. The basic personal data of the user are also store such as first name, last name, position, email and phone number. An Admin attribute is used to identify if a user is an admin or a normal user.

Attendee:
The attendee entity holds the information about the attendance of the event. The combination of the two attributes *User_ID* and *Event_ID* will give a combine key for the table.

5.1.2 Tables

Table 1: Event

Attributes	PK/FK	Data Type	Constraints
Event_ID	PK	VarChar(10)	Unique, Not Null
User_ID	FK	VarChar(10)	Not Null
Host_ID		VarChar(10)	Not Null
Event_Description		VarChar(30)	Not Null
Event_Start_Date		Date	Not Null
Event_Duration		Integer	
Room_No		VarChar(10)	
Location		VarChar(15)	
Private		Bit	
Comment		VarChar(100)	

Table 2: Attendee

Attributes	PK/FK	Data Type	Constraints
Event_ID	FK	VarChar(10)	Unique, Not Null
User_ID	FK	VarChar(10)	Not Null

Table 3: User

Attributes	PK/FK	Data Type	Constraints
User_ID	PK	VarChar(10)	Unique, Not Null
Admin		Bit	Not Null
Password		VarChar(20)	Not Null
First_Name		VarChar(15)	Not Null
Last_Name		VarChar(15)	Not Null
Position		VarChar(30)	
Email		VarChar(30)	
Phone_Number		VarChar(20)	Not Null

Table 4: Event Request

Attributes	PK/FK	Data Type	Constraints
Event_ID	FK	VarChar(10)	Unique, Not Null
Response		Bit	Not Null
Seen		Bit	Not Null
Date_Invited		Date	Not Null

5.2 Entity Relationship Diagram

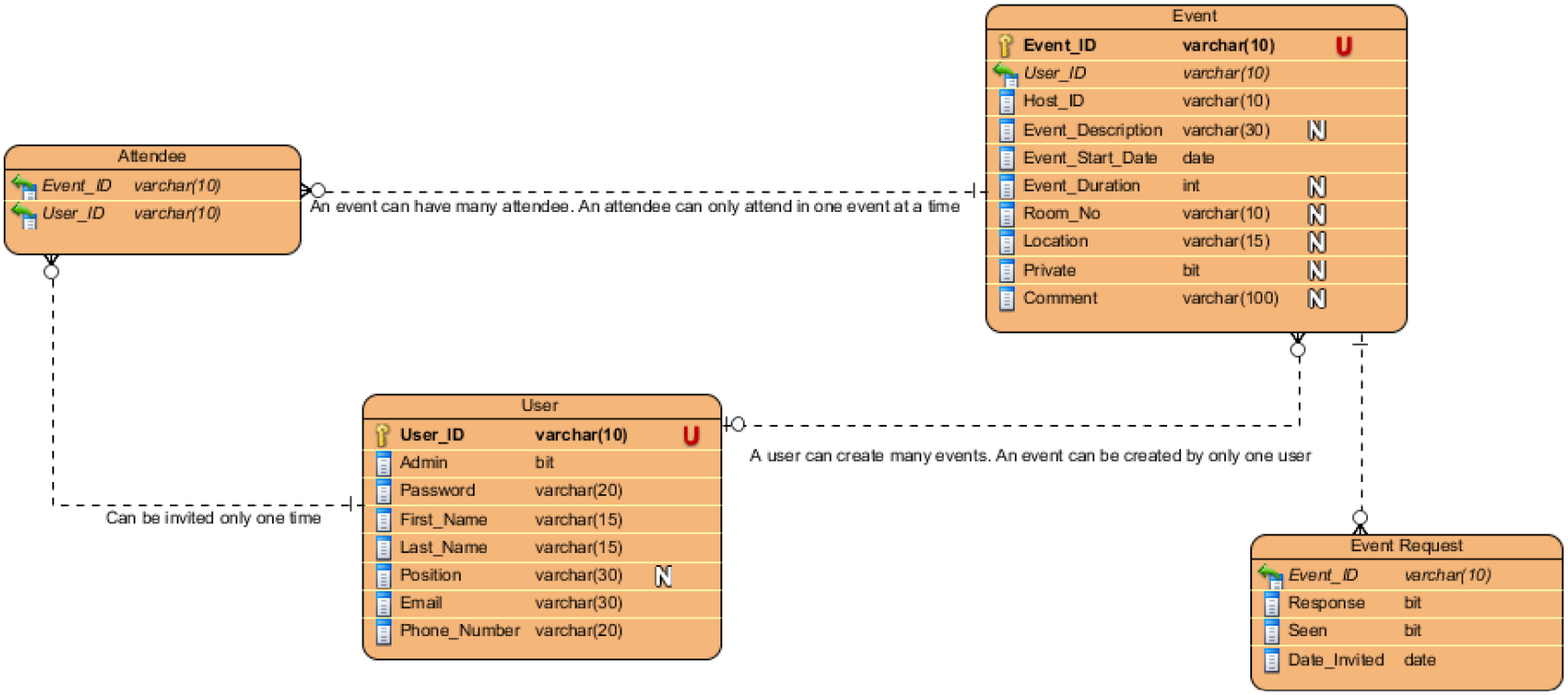

Figure 3: Entity Relationship Diagram

6 Design