Functional Specification for Meeting-Organiser

1.Introduction

1.1 Overview

The system will be a webapp which will utilise its user's calendar to organise meetings between multiple users. Users will create logins for the webapp and will have a calendar associated to each login, which has all their scheduled events for the weeks ahead. The users can then create groups of users to organise a

meeting with. Each group will have an admin - the user who created the group by default, though this privilege can be passed among users in the group, and multiple admins will be allowed but not advised. Groups will have a built in chat client, and the admin of the group can set parameters for a meeting to organise, as well as assign some users as high priority for meetings, such as a meeting between students and a lecturer would assign high priority to the lecturer. Some of the possible parameters will include: how long a meeting will last; meeting locations as specified (see below for description); high priority users for meetings. Meeting locations will adhere to necessities for users, such as accessibility for disabled users, computers, conference call functionality and more. Behind the scenes, the application will compare each user in a group's calendar to find suitable times which adhere to user's available time. In the instance of the application returning multiple possible meeting times, users in the group can vote for which time suits them best. High priority users will have a higher weighting for voting. In the event of a tie, the option voted for by the highest priority user will be chosen, or the group admin will gain privilege to choose which time should be used. Once a meeting is organised, the webapp will add this meeting to the user's calendars.

Other functionality of the webapp will include automated emailing to users with group invitations and meeting details. Users in a group will have the option to upload notes of scribed meetings, which can be sent via email from the webapp to each user in the group, or to users in the group who could not attend a meeting. Businesses will be able to create groups, which can host all their meeting locations and all their members of staff. These business groups can then have subgroups to organise meetings from, and will be able to use the meeting rooms the business owns as suitable meeting locations. This will allow our webapp to note when a location is available or unavailable also. It is aspired that the webapp will be able to handle meetings between different time zones, without setting a "redeye" meeting for some user(s).

1.2 Glossary

Webapp - a web based application
API - Application Programming Interface
Admin - Abbreviation of administrator, although an admin on our site will not do much administrating, we decided this is the most appropriate word to define the creator of a group.
HTML - HyperText Markup Language
CSS - Cascading Style Sheet
SQL - Standard Query Language
PHP - PHP originally stood for Personal Home Page, but it now stands for the recursive acronym PHP: Hypertext Preprocessor.
Java - Slang lingo for coffee. Also refers to the programming language by oracle which will be used in the development of the application.

2.General Descriptions

2.1 System Functions

Users
Users of the webapp will create a login for our site using a gmail account. Through use of Google API this will allow the webapp to access the user's google calendar and send emails to the users. Users will be able to alter some settings, such as what they will receive email notifications about, which users can contact them, if their details display to other users in groups and other expected user functions.

Groups
Groups will hold any number of users from 2+. High user count groups will be unadvised but will still be permitted. Each group will have a chat client. Groups can have subgroups, which are purposed for businesses but can be used by anyone. Groups will have the option to be for a once off meeting. This means that after a meeting is scheduled the group will be disbanded. This function is for once-off meetings.

Group Admins

When a user creates a group they automatically become the admin of the group. Admins decide when all users have joined the group, when to organise a meeting, which users are high priority, settle tied votes, set meeting requirements.

Interface
The interface of our website will be programmed through HTML and CSS. It will be easy to use for a first time user, but also rewardingly fast for an experienced user through use of shortcuts and memory of users most used tasks.

Meetings
The core function of the webapp. When the admin of a group clicks the “Organise a Meeting” button, the meeting function begins operating. It will dissect each user in the group’s calendar, taking out each available free time which adheres to parameters set by the group admin. It will compare each user’s list of free times and return all the suitable times it found.

Voting
In the event of the meeting function returning multiple possible meeting times, user’s will be given the option to vote for their preferential meeting times through highest importance voting - ie they rank their most suited time 1, and their least suited time n where n is the number of meetings returned.

2.2 User Characteristics and Objectives

Types of Users

There are 2 primary types of Users we are catering towards.
1. Business people looking to organise internal or external meetings 2. College students looking to organise meetings with lecturers or with groups for projects.

Each of these users will probably be daily users of the internet and as a result should be able to access our website without any issues. Some may also be familiar with Calendars and how to import, export, and sync them. However it would still be a good idea to provide instruction on how to complete these tasks. Both these groups will probably be familiar with the website “Doodle” which has a similar function to our webapp. This will help us as users will be familiar with this idea of figuring out common free times using a website. It may also however lead to users finding it tough to adapt to the changes between “Doodle” and our website.

However due to the broad appeal of a meeting planner it could be expected that a variety of users would use the webapp.. While we can assume they at least have a basic knowledge of the web, our webapp must still be easy to use so as to cater to the lower technical skills of some of the users.

Users Expectations / Requirements

Business / College Expectations

- The ability to organise a meeting between groups of people. The people in the group may or may not be users of the website.
- An easy way to fill in when they are unavailable. This could be manually filling in a calendar or importing their calendar.
- Ability to vote on which time suits them best for a meeting.
- Be able to create a group containing users of the website who they have frequent meetings with.
- The ability to add in requirements for a meeting so as to show the available meeting rooms that meet the requirements.
- Have certain users (for example Lecturers or Managers) carry a greater weighting so as to prioritise

their free times over other members of the meeting / group.

- Add meetings to the users calendar and send out notification emails.
- Inter group / meeting chat. So as to communicate between members.

User's Wishlist

- Upload minutes or notes of meeting which is then sent to all members of the group.
- Organise meetings between different timezones.
- Create a business group with multiple sub-groups.

2.3 Operational Scenarios

User Creates an Account

1. User clicks on create account button.
2. User is asked to fill in a form. They must enter a unique username and a password that conforms to the security standards we set. For example at least 6 characters long with letters and numbers but no miscellaneous symbols. They also must enter a First and Last name. However other information such as Address, Date of Birth, College/Business, and Occupation is optional.
3. User is sent an email to confirm email address.
4. User enters code sent in email address and this activates the account.

User Logs into account

1. User clicks on login button.
2. User is given choice between logging in with an account they created or a google account they currently possess.
3. User enters username and password.
4. If correct they can enter the site.
5. If incorrect given a fixed number of tries to correctly input details.
6. Account is locked after enough incorrect tries. User is sent an email allowing them to unlock it.
7. If a user has forgotten their username or password they can either have their username sent out to them or they can reset the password.

Import Calendar to Google Calendar

1. User is given option to import a .csv or .ics file.
2. User click import button.
3. Window pops up that allows user to select the file to be imported or drag and drop file into window.
4. User presses ok when correct file is selected.
5. If file is imported correctly then user is given a message "Events have been imported".
6. Otherwise "Import Failed" message is given.

Organise a Meeting With Website Users

1. User selects create a meeting button
2. User adds users to meeting using their email address or sending them a link.
3. User selects a start and end time and or date for the meeting to occur during as well a duration.
4. User select any additional requirements for the meeting for example size of room required, projector, or conference phone.
5. User sets any high priority users for example Lecturer/Manager. This gives this user more weighting in the voting process.
6. User presses create button.

7. User is informed if group is successfully created.

2.4 Constraints

The webapp will run into multiple constraints throughout the development process. These constrains are listed below:

Time Constraints
The development team will encounter issues arising from trying to have the project completed and operational in the six weeks they have to complete it.

Hardware Constraints
In order for the webapp to be operational it is necessary that it can be hosted online on a server. This will have to be acquired by the development team.

User Requirements
It is necessary for the program that it meets the requirements of its users.

3.Functional Requirements

3.1 User Login/Registration

Description
This system will allow Users to log in to an existing account or create a new account. These will be stored in the database so the webapp can authenticate a user's username and and password.
This will be done using the Google Firebase Api. This allows users to login using their pre-existing google accounts.

Criticality
This system is critical to the webapp. The webapp needs to be able to access the user's Google Calendars to select the best times for meetings, if they aren't logged in application has no way to access their calendar. Users are also required to make groups.

Technical Issues
The webapp must deal with all the usual issues that exist in implementing a login system. For example storing the usernames and password securely, checking that details are entered correctly, and providing a system for users to recover locked accounts as well as a good UI.
However the application must also allow users to log in with a pre-existing google account this adds another layer of intricacy.

Dependencies
User depends on being able to query the database for the user's login information.

3.2 Creating a group

Description
The webapp allows users to set up a group. These will allow users to create meetings within the group as well as other expanded setting not available to users making a solitary meeting. Some examples of this are adding meeting rooms to groups as well as creating subgroups. Users can have multiple groups and groups can be organised for just two users up to a seemingly unreachable cap. It is from groups that meetings will be organised.

Criticality
It goes without question that without groups there will be no meetings. In a webapp based around making the organisation of a meeting easier, meetings are important. Thus, groups are extremely important to the webapp.

Technical Issues
Groups will need to be stored somewhere, thus adding the necessity for databasing. UI wise it should be clear how to create a group and change group settings. Functionally it should be easy for users to create and join groups.

Dependencies
Groups depend on the existence of users, such that, there cannot be a collection of users without users to collect.

3.3 Adding Users to a Group

Description
Users can add new users into their group. A user must be an admin of a group to add users into it. The user who is added to the group is sent an email. They can accept the invitation and join the group or choose not to join the group. Groups will have a limit on how many users can join them.

Criticality
This is very important for making groups work properly however it is not critical to the webapp.

Technical Issues
The webapp must be able to set users as admins of their respective groups. It also needs some way of sending invites to groups to users via their email addresses. Lastly, the application must be able to keep track of how many users are in a group and enforce an upper limit on it.

Dependencies
Requires groups to be working as well as users registration and login. Also needs database queries to be working.

3.4 Creating a Meeting

Description
Meetings are created by a user. The creator of the meeting will input an acceptable timeframe for the meeting to occur within. They will also add the users they wish to attend the meeting. Lastly they can chose to make the meeting a repeating one eg Daily, Weekly, or Monthly. The system then searches for the time that most of the attendees are free within that timeframe. It returns a couple of times and lets the users decide on the best time for the meeting to occur. It will also allow users to chat with other members of the meeting using a secure group chat.

Criticality
This is the main driving idea behind the webapp and as such is the most critical part.

Technical Issues
It will be necessary to be able to access each user's calendar. The application will need to be able to send users invited to a meeting an email.

Dependencies
This depends on querying a database and the ability to sync with user's calendars and the ability of the users to login to the webapp.

3.5 Voting on the Best Time for a Meeting

Description
Users are given a couple of options as to which time suits them best for a meeting. A user will rank these options from best time to worst time. The system will then choose the most popular option will keeping in account certain rules. These rules might include things like a certain person must be available at that time or the meeting room must have the ability to video conference.

Criticality
While this is not a key feature of the meeting organiser, it will add an element of user control, and is important to settle instances where users have multiple common available times.

Technical Issues
The webapp needs to find a way of allowing users to vote on certain options. This will probably be done by using a mixture of html and css for the webapp and php and sql to calculate and store the result.

Dependencies
Depends on Meetings, Users, Database Queries.

3.6 Calendars

Description
Calendars for the webapp refer to the Google Calendars associated with each user's profile. They will be read by the webapp in order to organise meetings. The webapp will also write new meetings to the user's calendar.

Criticality
High importance: Without calendars the webapp will not be able to find free time for all users to organise meetings.

Technical Issues
As the calendars will not be hosted by the webapp or the database associated, the program may have slow periods depending on internet traffic, as it will be sourcing its calendar information from the internet.

Dependencies
Each calendar is associated to a user. Calendar interacts with meetings.

3.7 Calendar Sync

Description
The process of syncing a users currenting calendars into their google calendar attached to their login. This can be done by means of importing a .csv or .ics file. They are also given the option to either share their other calendar to this google calendar or manually enter times they are busy on the calendar itself.

Criticality
This is a critical feature as without this users will not be able to enter when they are busy.

Technical Issues
Finding out how to allow users to share other calendars to this calendar.

Dependencies
Depends on calendars to work.

3.8 Database Queries

Description
The database will store information related to users and groups. The webapp will need to store and pull data from the database for almost every action it performs. The database will consist of two major databases - users and groups.

Criticality
The database for the system is trivially important, as without it groups will not be an operable function.

Technical Issues
Database queries adds the need for SQL and PHP to the project. Requires a physical server to host database on, however this can be the server the webapp will be hosted on.

Dependencies
The User and Group information is stored on the database. These are the only functional requirements which interact with the database.

4.System Architecture

Below is a system architecture diagram, which shows where the interactions between users, the system, and the databases occur.

System Architecture Diagram

System Architecture Diagram

5.High Level Design

Displayed in this section are several diagrams which help to describe and display the functionality of our webapp. The diagrams that will be shown below are:

- *Context Diagram*
- *Data FlowDiagram*
- *Logical Data System*

Context Diagram

Context Diagram

Data Flow Diagram

Data Flow Diagram

Logical Data System Diagram

Logical Data System Diagram

6.Preliminary Schedule

The Gantt chart sets out the order we want to build our webapp and the timeframe for each item.

Gantt Chart

7.Appendices

- Toodle - http://doodle.com/
- Google Calendar API - https://developers.google.com/google-apps/calendar/
- Google Sign In API - https://developers.google.com/identity/
- DCU Redbrick - https://www.redbrick.dcu.ie/
- Stack Overflow - http://stackoverflow.com/
- W3Schools - http://www.w3schools.com/