

## CONTENTS



**AUTHOR:** Jade Harris | 12SDD

**MENTOR:** Adam Leserve

OVERVIEW.....	2
PROBLEM DEFINITION.....	2
SOCIAL, ETHICAL AND LEGAL CONSIDERATIONS .....	2
COMPATIBILITY AND PERFORMANCE REQUIREMENTS.....	2
DESIGN SPECIFICATIONS/ SYSTEM MODELLING:	
STORYBOARD.....	4
PROTOTYPING TOOLS (FOR INTERFACE MOCKUPS).....	5
MODELLING DATA FLOW DIAGRAM.....	9
DATA DICTIONARY.....	15
STRUCTURE CHART.....	18
RESOURCES ALLOCATION & GANTT CHART.....	20
QUALITY ASSURANCE.....	31



## REQUIREMENTS REPORT AND PROJECT PLAN

# AIMS AND OBJECTIVES

### OVERVIEW

CONNECT provides users (aged 18-60) with an effortless Windows software to organise, centralise and manage real-life events. CONNECT simplifies organisation for users, whenever and wherever, by generating event invitations, facilitating communication between attendees, displaying reminders, and maintaining a 'spending' budget. CONNECT is driven by a MySQL database stored on an Amazon-Elastic-Compute-Cloud virtual server (EC2), allowing the software to connect users through the Internet.

### PROBLEM DEFINITION

#### End User Requirements / Interface Objectives

Organising events is often a tedious task, complicated by personal feelings and misaligning schedules. CONNECT overcomes this problem for its general-public end-user with an interface that is:

- **USER-FRIENDLY** thus interface must be **minimalistic, self-documenting, intuitive and consistent** so it is not overwhelming, especially for inexperienced users
- **ROBUST** as errors could be fatal for the interface and communication with the external database. This involves prevention for inputting harmful data
- **CUSTOMISABLE** to **aid accessibility** (e.g colour-blindness) and enhance **user-experience** with **settings**
- **SMOOTH/HIGH-SPEED** with minimal response-times for **user-efficiency and convenience**
- **REDUCING SUBJECTIVITY** by **calculating priority** (mostWantedOption=3, secondPreference=2...) and using the randomClass

### PROBLEM DEFINITION

#### Boundaries

CONNECT uses **internet access from a Windows-10-OS machine** to **interface between the database server(EC2) and program.**

## SOCIAL, ETHICAL AND LEGAL CONSIDERATIONS

### Boundaries

CONNECT greatly benefits society by **connecting users in real-life** to **enhance health, wellbeing, and unity** of the community. It also **reduces the cost and waste** of over-booking events. However, CONNECT raises concerns for **potential real-life interaction with a stranger, stalking, data-privacy/security, and identity theft**.

These are also significant ethical implications. Furthermore, the ethicality of communicating information via the Internet could result in **exploitation of neighbour/public networks**. **Unintentional discrimination** is another consideration. However, the accessibility of CONNECT also promotes **diversity and inclusiveness in the community**.

Legal concerns involve **piracy of CONNECT, manipulation/leak of user data (particularly storing in external database) and copyright of graphics used** in the interface. Legal advantages are that CONNECT's **concept is not copyrighted**.

## COMPATIBILITY AND PERFORMANCE REQUIREMENTS

### Hardware and Software Compatibility and Performance Requirements

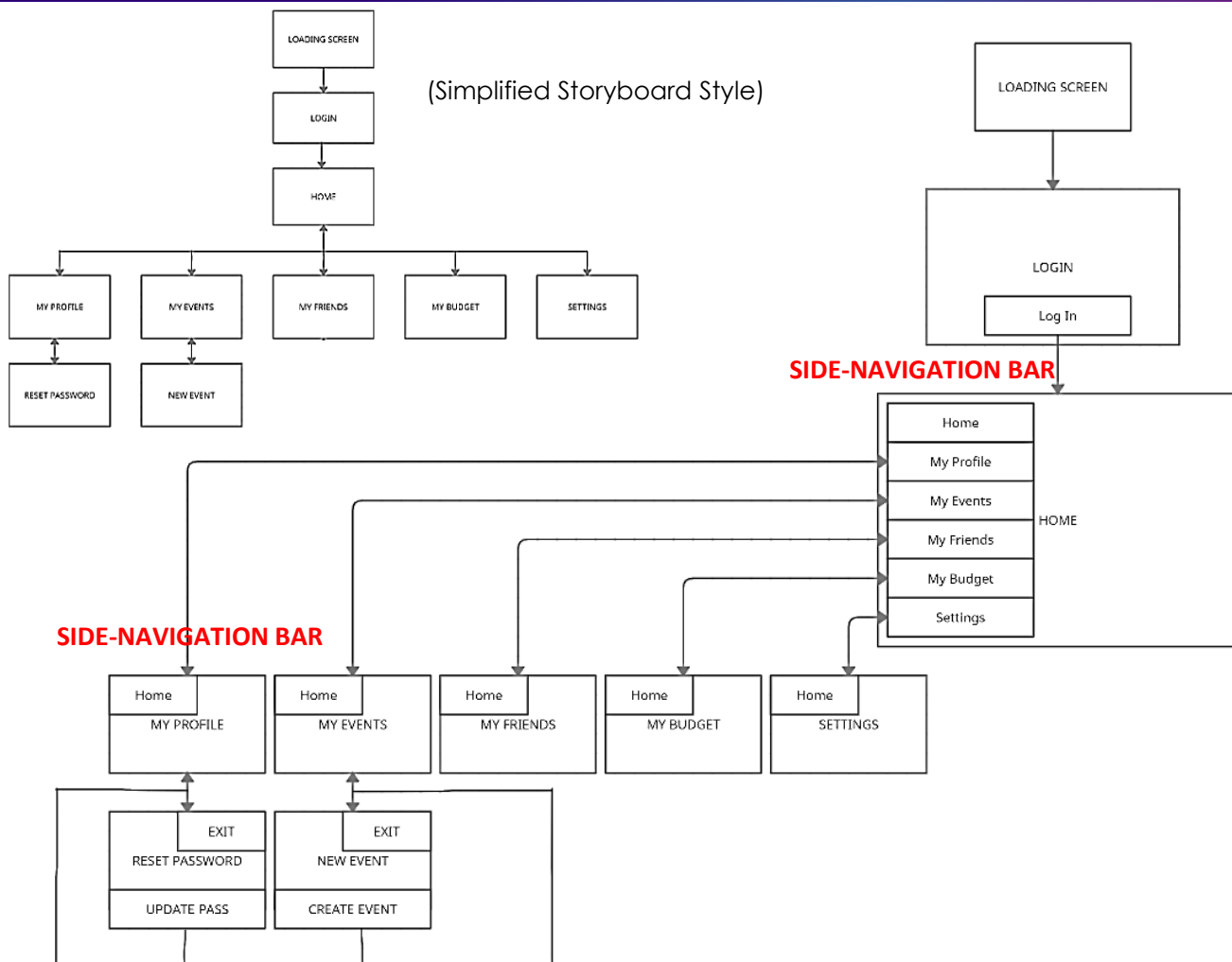
Due to the **small processing power required** to process and send strings to an external server, receive MySQL database table information, and display a minimalistic graphical interface, most systems running **Windows 10 OS** with an **internet connection** are suitable for CONNECT to run **smoothly**.

Peripheral hardware such as **monitor, mouse, and keyboard** (or appropriate substitutes) is required.

# DESIGN SPECIFICATIONS / SYSTEM MODELLING

## PROTOTYPING TOOLS

### Storyboard



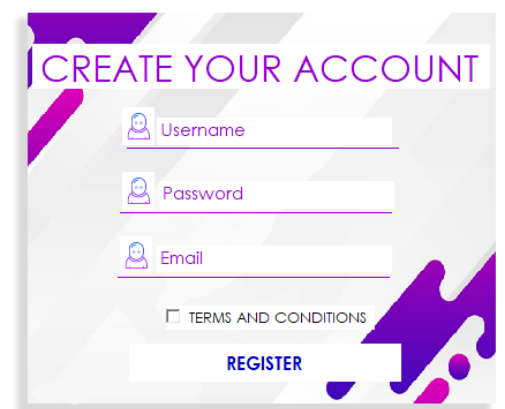
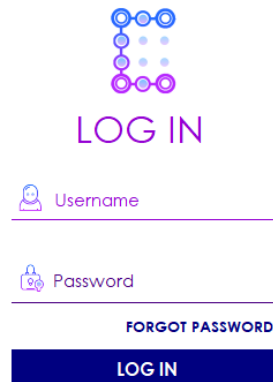
CONNECT first loads in with a brief loading screen. Once the progress bar has complete, a welcome screen automatically loads for the user (where they can either log-in or sign-up then log-in with these new credentials). The user is then taken to the home screen. Here, there is a **side-navigation bar** which can take them to different interfaces/forms of the program. MyProfile provides the option for the user to reset their password, which will open a separate ResetPassword form. MyEvents allows users to create a new event which will load a NewEvent form. The MyFriends, MyBudget and Settings screens can also be loaded from this Home screen. At any time, the user can return to the Home screen by using either the navigation bar or if they have the 'Return to Home' setting on, then they can close the form and return to home.

## PROTOTYPING TOOLS (For Interface Mock-ups)

### INTERFACE MOCK-UPS (Made with WinForms in Visual Studio)

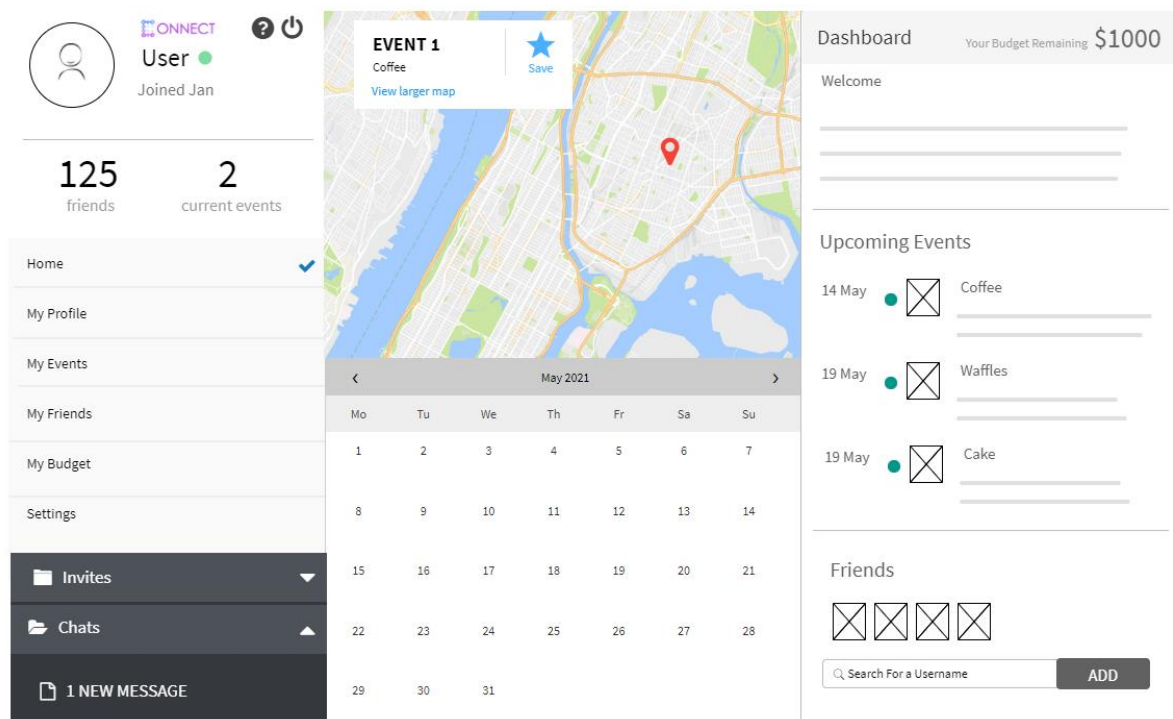
Loading Screen

Login-In Screen



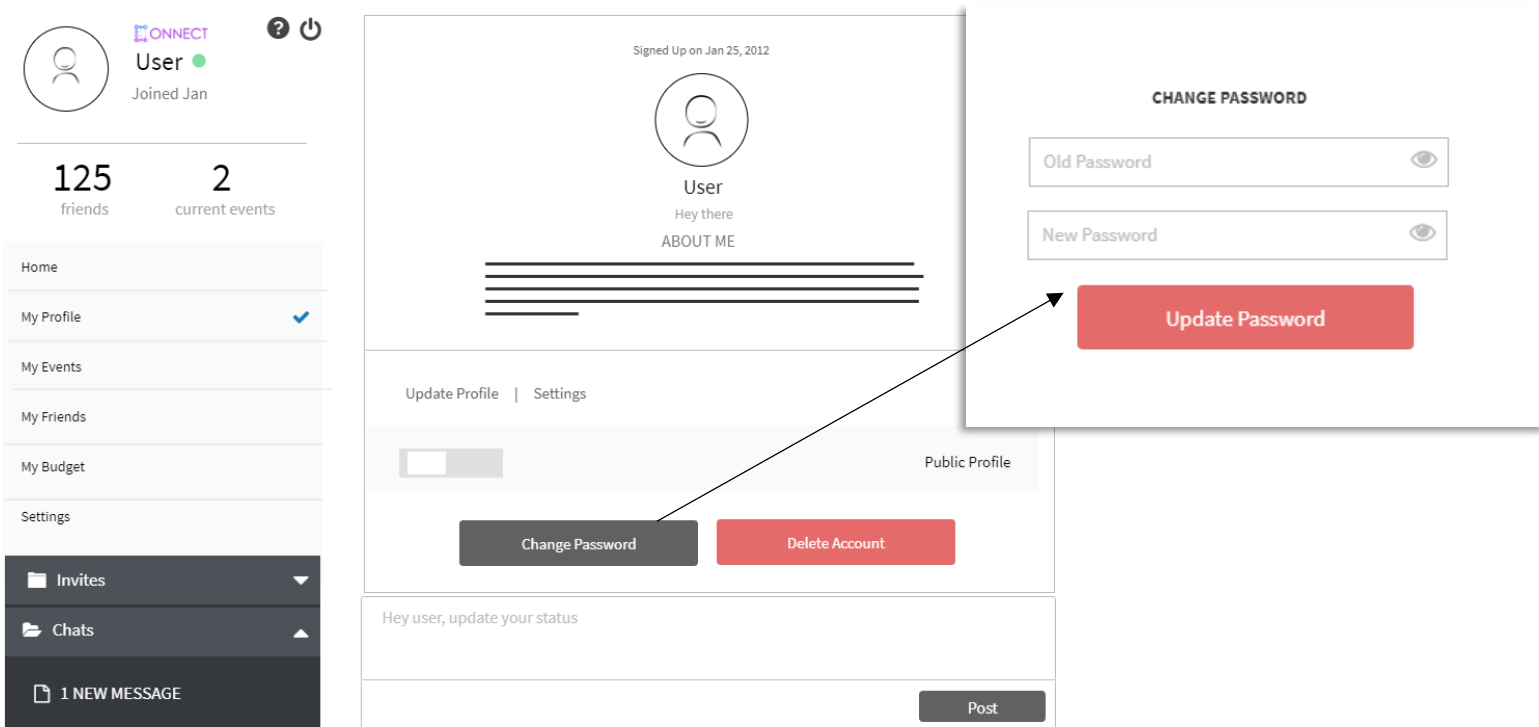
### INTERFACE MOCK-UPS (Made with Interface Designer tool)

Home Screen



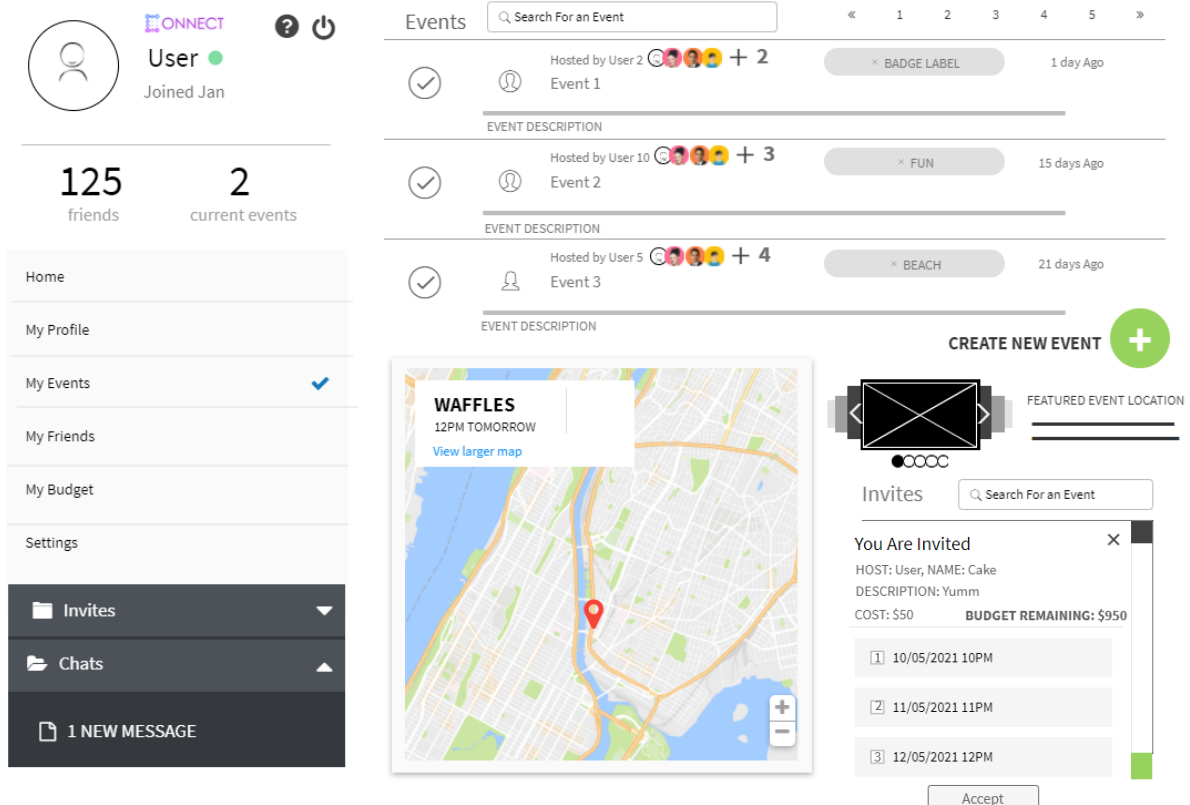
## INTERFACE MOCK-UPS

### My Profile > Change Password



## INTERFACE MOCK-UPS

### My Events



## New Event

## Create New Event

Name

Description

Invitees

+

▼

Location

+

Cost for Location

\$

+

10/05/2021

Enter Times... +

11/05/2021

Enter Times... +

12/05/2021

Enter Times... +

<

May 2021

>

Mo	Tu	We	Th	Fr	Sa	Su
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

☒ Create Event Group-Chat

☒ Private

☒ Allow Users to Invite Other Friends


☒ New Users Can Alter Invite Preferences

Send Invites



## My Friends

## INTERFACE MOCK-UPS

### My Budget



**CONNECT**  
**User**  
Joined Jan

125

friends

2

current events

Home

My Profile

My Events

My Friends

My Budget ✓

Settings

Invites

Chats

1 NEW MESSAGE

Set on Jan 25, 2012

Your Allocated Budget
**\$1000**


Change Budget


Event History

<input type="checkbox"/>	Event	Date	Cost	Host	Description
<input checked="" type="checkbox"/>	Project Scandal	07/05/2021	\$1	Mike	App
<input checked="" type="checkbox"/>	Project Riga	07/05/2021	\$1	Mike	App
<input checked="" type="checkbox"/>	Project Counter ...	07/05/2021	\$1	Mike	App
<input checked="" type="checkbox"/>	Project PMS	07/05/2021	\$1	Mike	App
<input checked="" type="checkbox"/>	Project HMS	07/05/2021	\$1	Mike	App
<input checked="" type="checkbox"/>	Project HOUSIN...	07/05/2021	\$1	Mike	App
<input checked="" type="checkbox"/>	Project VIRA	07/05/2021	\$1	Mike	App



Amount Remaining **\$993**

## INTERFACE MOCK-UPS

### Settings



**CONNECT**  
**User**  
Joined Jan

125

friends

2

current events

Home

My Profile

My Events

My Friends

My Budget

Settings ✓

Invites

Chats

1 NEW MESSAGE

☐

Save Settings Preferences

☐

Return to Home Before Exiting Program

☐

Block Invites

☐

Block Friend Requests

☐

Public Profile

☐

Save Events Past 30 Days

Save Settings

Delete Account



### DATAFLOWDIAGRAM 1 | Log-in/Sign up to Home Primary Module

**PURPOSE:** Allow user to log-in (or optionally sign up then log-in) and then redirecting them to the home screen.

#### COMPREHENSIVE DESCRIPTION OF DATA IN/OUT:

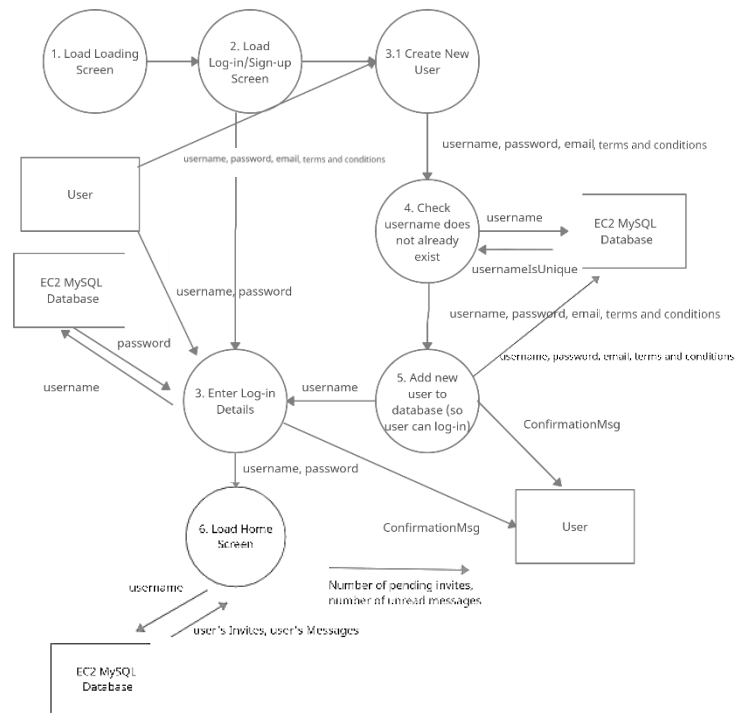
To begin, the user will be faced with the loading screen. Following this, the log-in or sign-up interface will automatically appear after the timer has completed. Here, the data flow diagram branches into two options: providing the user the ability to either create a new account or log-in. If the user decides to create a new user, then they will have to supply a unique username along with an associated password, email and terms and conditions response (whether they agree or not). This data is sent into the next process where the username is sent to the MySQL database. Here, it is checked if the username is unique (using a MySQL command). Alternatively, this could be achieved by another process where a list of usernames is provided by the MySQL database and the current username is checked against it).

The MySQL database will then output a boolean variable named `usernamesUnique` which will then be retrieved by the process. If this is true, then a user does not already exist with this username. Thus, the new user data is added to the external MySQL server and the username. Furthermore, this username flows into the next function so that the username field can be auto-filled (aesthetics). The program will also output a confirmation message to the user.

Whether the user decided to immediately log-in to their account or create a user first, the log-in process will then occur by receiving a username and password from the user. This username will then be sent to the external server to retrieve the password associated with the username. If this password matches, then the user has entered the correct credentials and then the home screen will display.

The last process loads the home screen. This sends the username data to the external database to receives the user's invites and user's messages information used in the interface.

The following dataFlowDiagrams use the username data variable from this diagram.



### DATAFLOWDIAGRAM 2 | MyProfile Primary Module

**PURPOSE:** Provide users with an interface to customise their personal profile that is stored on the MySQL database and for other users to see.

#### COMPREHENSIVE DESCRIPTION OF DATA IN/OUT:

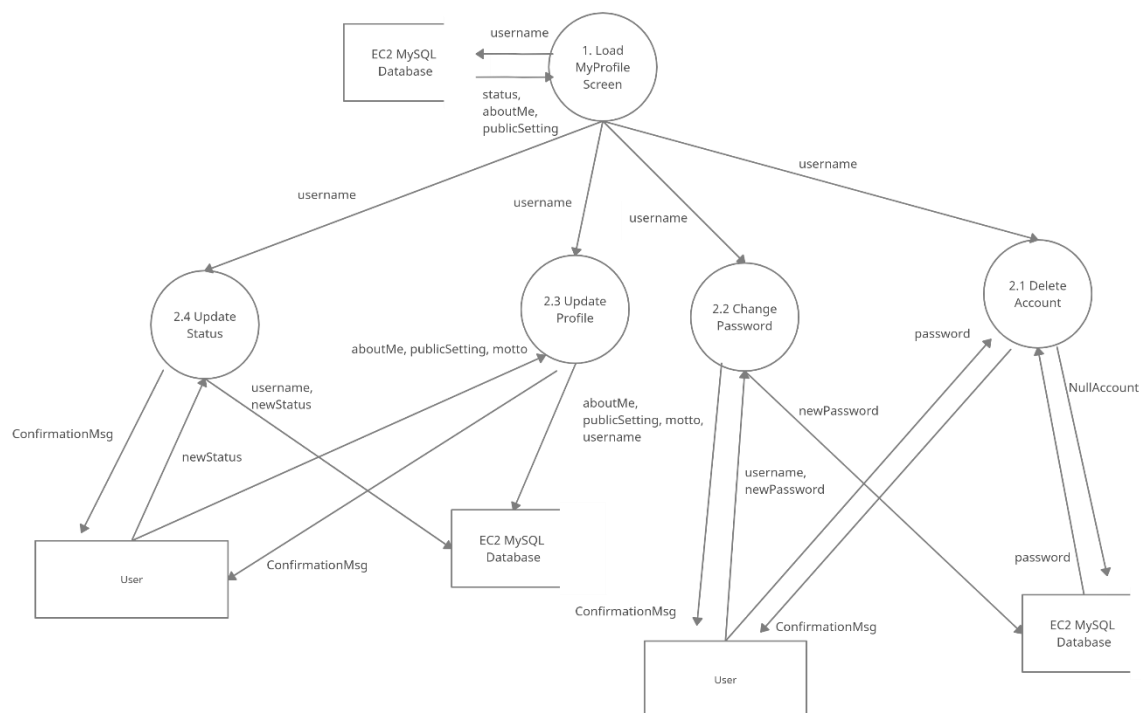
If the user decides to load the MyProfile screen from the sideNavigation bar, then the loadMyProfile screen process will occur. This loads the appropriate interface and to gather the data necessary for the elements, it will send the username data string to the MySQL database. Here, it will receive the status, aboutMe and publicSetting variables associated with the user currently logged in. After this (again due to the event-driven nature of winForms) the use has 4 possible processes.

The first is the user's option to update their profile status. This process requires the input of the user's username from the loading interface process which it will send to the MySQL database along with the newStatus which the external entity of the user is required to input into the system. Once this process has been successfully complete a confirmation message will be outputted to the user.

The next process that the user can choose is to update their profile. This process requires the input of the username which will then be sent to the MySQL database along with the user's new aboutMe, publicSetting and motto inputs.

Thirdly, the user can engage in the process to Change Password. Here, the process requires newPassword data input from the user then along with the username, this is sent to update the user's login credentials in the MySQL database. The user receives a confirmation message if this has been successfully reset.

Lastly, the delete account process may occur which requires the user to input their password. This is compared against their old password which is received from the EC2 MySQL database server to reduce the likelihood of accidentally deleting their profile. However, if this is successful and the user's account is deleted then the process will output a confirmation message and nullAccount data (a string replacing the user's name with "-") will be returned to update the mySQL server with the information that the user's profile has been removed. Alternatively, this would involve removing the user's database record.

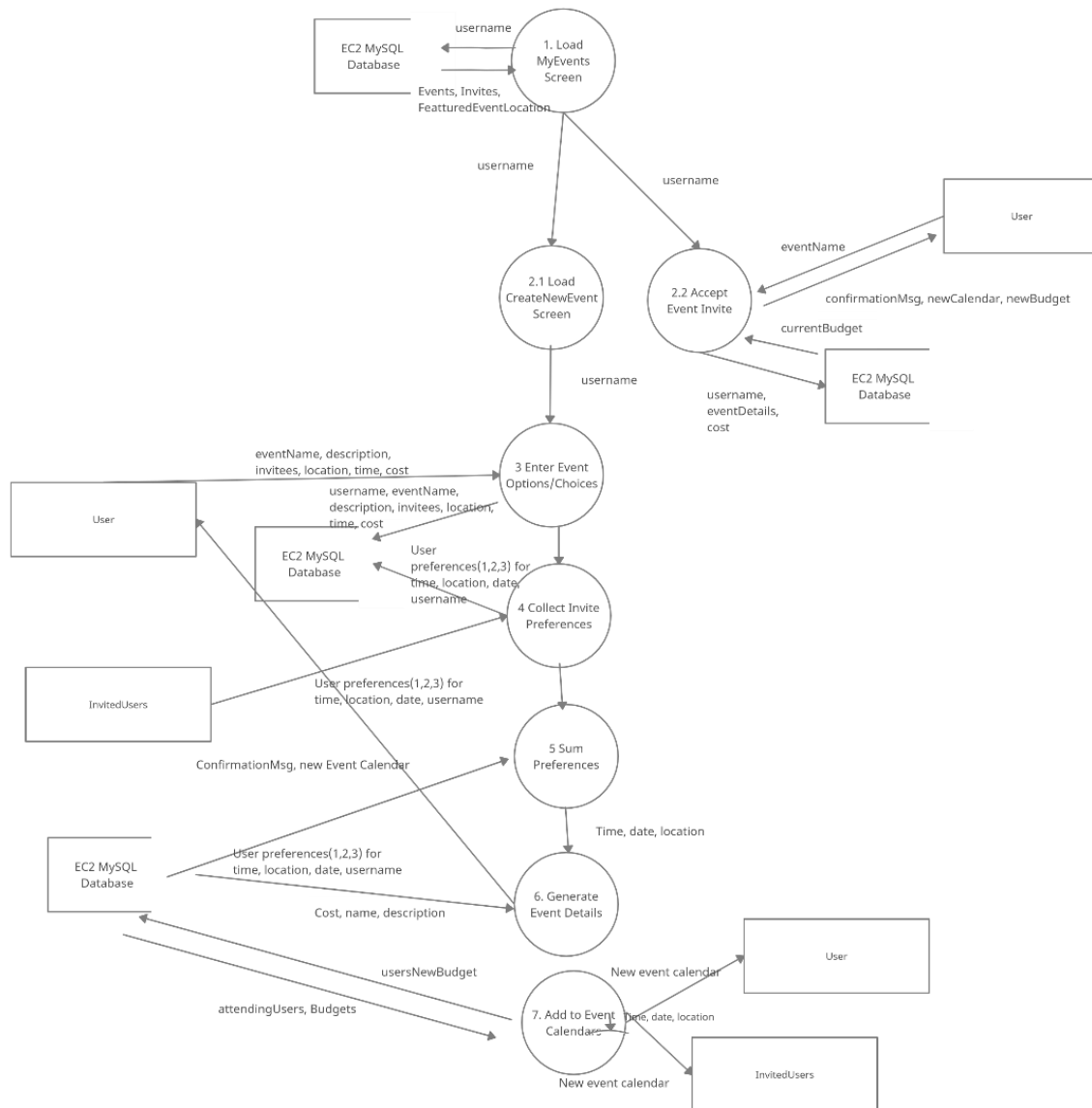


**PURPOSE:** An interface which allows users to centralise, invite other's, accept invitations of and create new events that are stored on the MySQL database.

**COMPREHENSIVE DESCRIPTION OF DATA IN/OUT:**

This module loads and functions the My events form. Firstly, it collects data using the user's username for the MySQL database to fill the information of the interface - events, invites, and the featured event location.

As the interface is event driven, there are two possible processes branching from here. The user can either accept an event invite which requires the event name from the user, or create a new event which involves loading the createNewEvent screen. If the user chooses to accept an event, the database will receive the username, details of the event (to update the users event list and the cost of the event. This will be used to calculate the user's new budget, which will then be returned to the function (to showcase if the user can afford the event). Once this process is complete, the user will receive a confirmation message that the event has been accepted, their calendar will be updated with the new event and they will be able to see their new budget. Otherwise, if the user has decided to create a new event they will be required to input the details of the events. These details are stored in the MySQL database. Next, then invited user's external entity receive an invitation retrieved from the MySQL database where they are required to input their preferences. This is summed (where the highest score for each option is selected). Lastly, the event details are generated completing with information from the MySQL server and after updating the user's budgets (and returning this to store in the database, the users event calendars will be updated with the new event.



**DATAFLOWDIAGRAM 1 | MyFriends Primary Module**

**PURPOSE:** Create an interface which allows users to view, message and control their friends.

#### COMPREHENSIVE DESCRIPTION OF DATA IN/OUT:

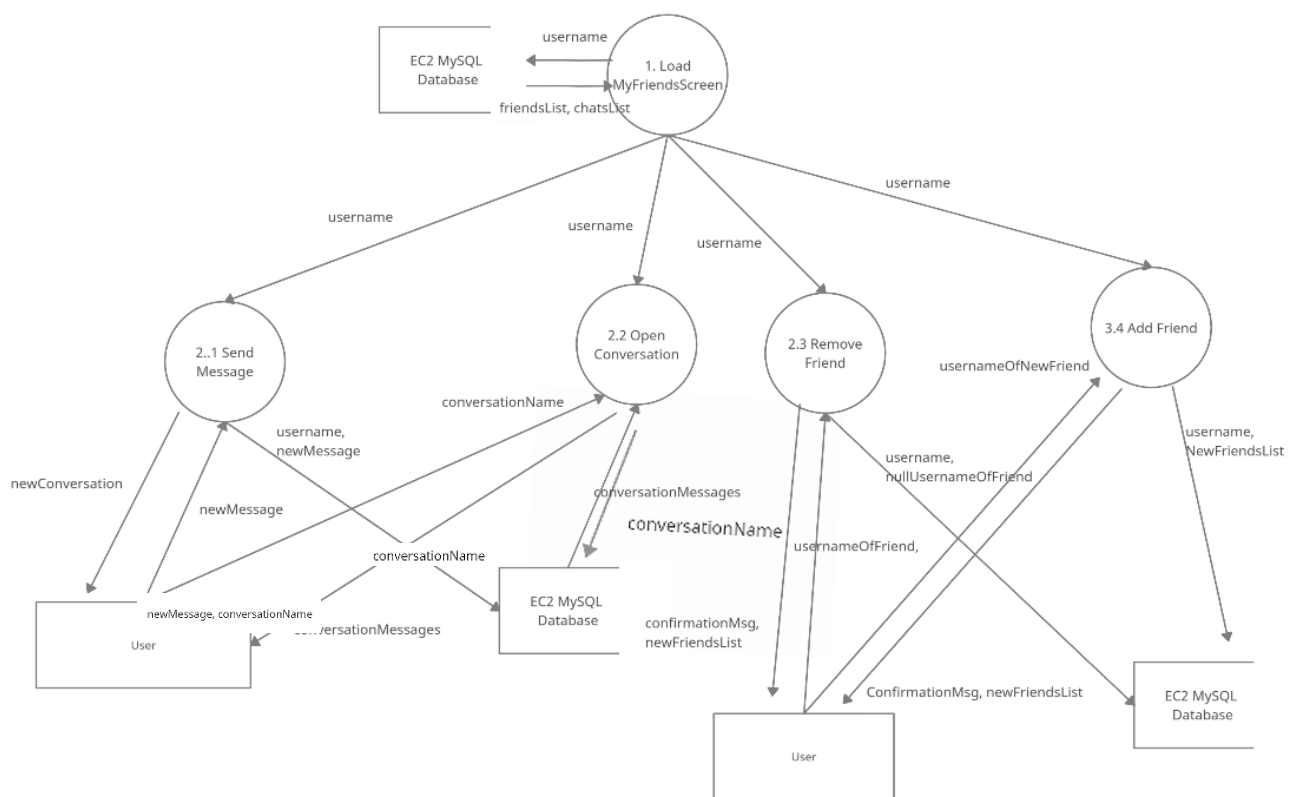
The MyFriends primary module begins with a process to load the MyFriends interface. This screen requires the friends list and chats list data from the EC2 MySQL Database which it retrieves by sending the username of the current user. Once more, because winForms is event-driven, there are 4 different possible branches for the next processes.

The user can send a message by inputting a newMessage and conversationName into the system. Here, the program sends the data of a new message, the username and the conversationName to the MySQL to update the stored conversation with the new message. The external entity user then receives the output of the process as a new conversation.

The next process that the user can use is Open Conversation. This requires the user to input the name of the conversation they wish to open to the process which is sent to the MySQL Database which then outputs the conversationMessages to the process. This then displays the conversation to the user.

Another process that the module's purpose encompasses is the functionality to remove a friend. Here, the user inputs the username of a friend to the process which is then sent to the MySQL data store and marked to remove (either by actually removing the record or by setting the friend's username to "-"). This returns a confirmation message to the user to let them know that their friend has successfully been removed as well as displaying the new friends list.

Finally, this primary module provides the user the ability to add new friends, by using their input of a username to add it to the MySQL data store. Once this is complete, the user receives a confirmation message and the interface is updated with the new friends list.



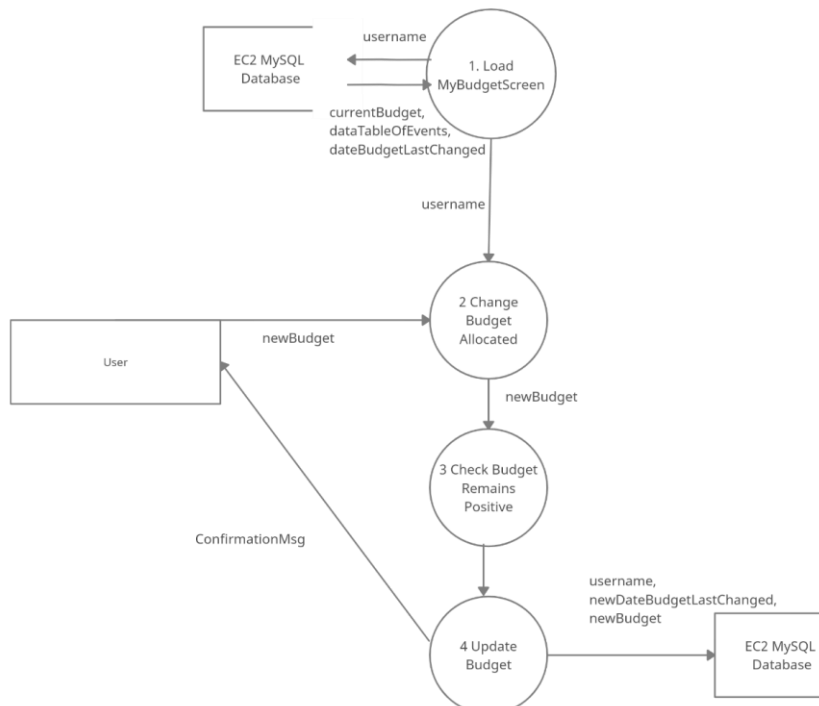
**DATAFLOWDIAGRAM 5 | My Budget Primary Module**

**PURPOSE:** Create an interface that allows users to view, manage and reallocate their 'spending' budget.

#### COMPREHENSIVE DESCRIPTION OF DATA IN/OUT:

The MyBudget module loads the myBudget interface by sending the username to the mySQL database which outputs the according user's data for the interface. This allows the process to receive the users currentBudget. The user can then input the new amount they wish for their budget to be. This is followed by a process that ensure that the budget will remain positive by reading the list of events and costs to ensure that the budget remains positive. If this is successful, then the

module outputs and confirmation message which the user receives as a display as well as adjusting the budget to display this new budget.

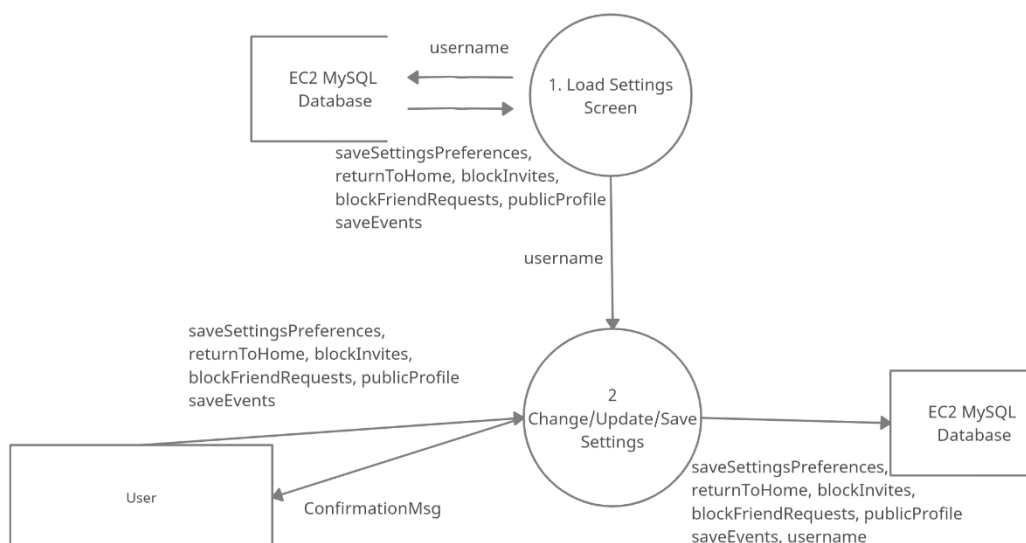


**DATAFLOWDIAGRAM 6 | Settings Primary Module**

**PURPOSE:** Create an interface that allows users manage their account and customise the program.

#### COMPREHENSIVE DESCRIPTION OF DATA IN/OUT:

The settings module loads the settings screen by retrieving the Boolean data values from the MySQL database that are associated with the current user (by providing it the username data). The only process available for users here is to change their current settings. Once the user has inputted these new settings (the new Boolean data values) are sent to the MySQL database for cloud storage (thus it is customisable on an CONNECT program). The user receives a confirmation once this has been successful.



## DATA DICTIONARIES

### Log-in Form

FIELD NAME	TYPE	DESCRIPTION	EXAMPLE
Password	String	The password the user entered	"Password"
Username	String	The username the user entered	"Username"
TermsAndConditions	Bool	Whether user agrees or disagrees (checkbox)	True
Password2	String	Stored password for the user (password compared against)	"Password"
Email	String	Email for a new user	"Email@email.com"
UsernameIsUnique	Bool	Test if the username is unique for a new user	True
UsersMessages	Int	Counts all of users messages	1
UsersInvites	int	Counts all of users messages	1

## DATA DICTIONARIES

### MyProfile

FIELD NAME	TYPE	DESCRIPTION	EXAMPLE
Password	String	The password the user logged in	"Password"
Username	String	The username of the user logged in	"Username"
AboutMe	String	User's about me section	"Nice to meet you"
publicSetting	Bool	Stores if user has their profile on public or private	True
Motto	String	User's short motto	"Hi there"
ConfirmationMsg	String	Notify user of successful action	"You have successfully updated your status"

<b>NullAccount</b>	String	Remove user from the MySQL field (or - set as “-” for null)
--------------------	--------	---

## DATA DICTIONARIES

### MyEvents

FIELD NAME	TYPE	DESCRIPTION	EXAMPLE
<b>Username</b>	String	The username of the user logged in	“Username”
<b>Events</b>	List (STRUCT)	List of events	
<b>FeaturedEventsLocation</b>	String	Stores the featured event	“Dinner at Bazaar”
<b>EventName</b>	String	Name associated with an event	“Breakfast”
<b>Description</b>	String	Description for an event	“Let’s go catch-up”
<b>Invitees</b>	List of Strings (STRUCT)	Contains list of usernames of all users - invited	
<b>Location</b>	List of Strings (STRUCT)	Contains list of all possible locations - with their cost	
<b>Time, Date</b>	Date	Time and date for event	10/10/2021 10:05
<b>User Preferences</b>	Int	Can be either 1,2 or 3 (up to n amount of options) for events. This is added and the option with the highest value is selected.	3
<b>Cost</b>	Float	Cost for the event	123.12
<b>ConfirmationMsg</b>	String	Notify user of successful action	“You have successfully updated your status”

## DATA DICTIONARIES

### MyFriends

FIELD NAME	TYPE	DESCRIPTION	EXAMPLE
------------	------	-------------	---------



<b>Username</b>	String	The username of the user logged in	"Username"
<b>UsernameOfNewFriend / UsernameOfFriend</b>	String	Name of the friend the user wishes to add as a friend or remove; the username of the user which is currently selected for interactions	"User2"
<b>NewFriendsList</b>	String or List of Strings (STRUCT)	Stores the currently logged in user's friend list	"User 2, User 3"
<b>nullUsernameOfFriend</b>	String	Remove the username as a friend from the MySQL field (or set as "-" for null)	"-"
<b>newMessage</b>	String	Message the user is sending to the conversation	"Hey all"
<b>ConfirmationMsg</b>	String	Notify user of successful action	"You have successfully updated your status"

## DATA DICTIONARIES

### My Budget

FIELD NAME	TYPE	DESCRIPTION	EXAMPLE
<b>Username</b>	String	The username of the user logged in	"Username"
<b>NewDateBudgetLastChanged</b>	Date	Last time the user changed their budget	10/21/2021
<b>newBudget</b>	Float	New budget the user wants	123.12
<b>ConfirmationMsg</b>	String	Notify user of successful action	"You have successfully updated your status"

## DATA DICTIONARIES

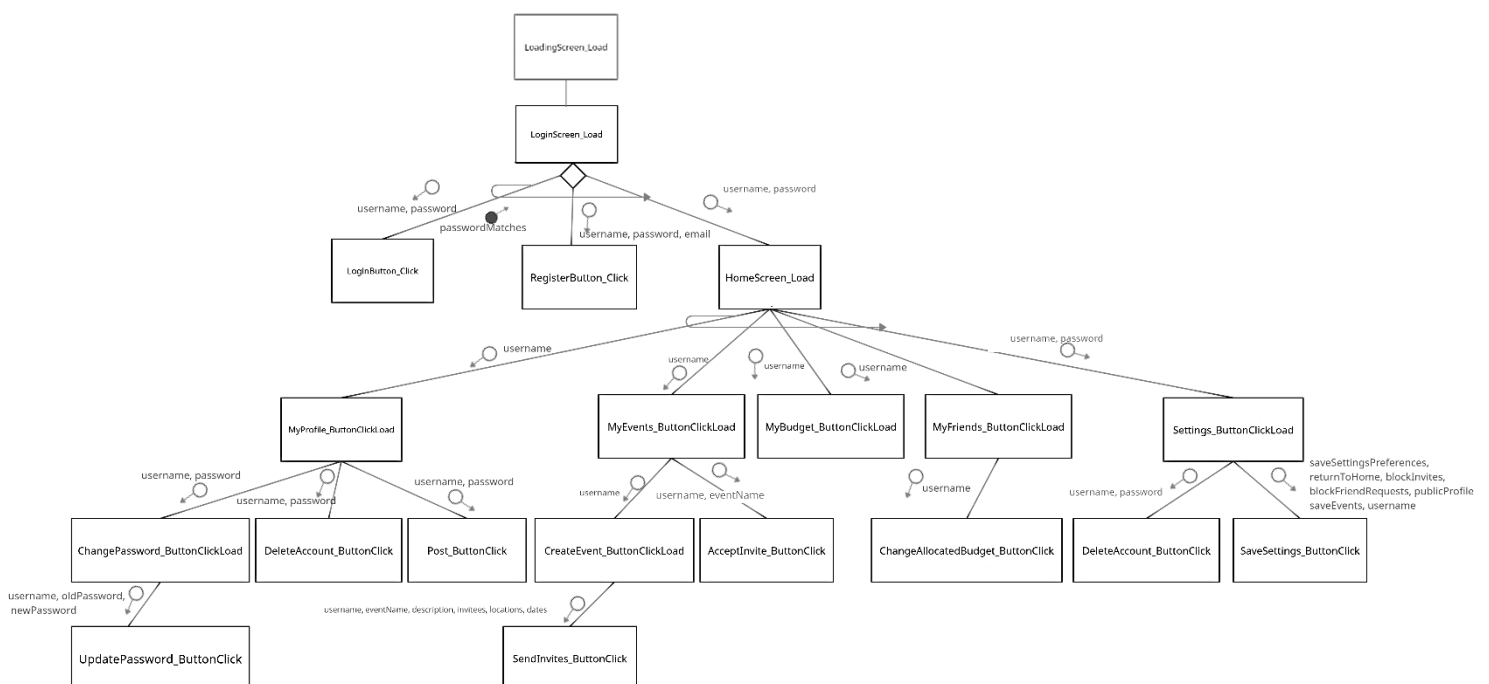
### Settings

FIELD NAME	TYPE	DESCRIPTION	EXAMPLE
------------	------	-------------	---------

<b>Username</b>	String	The username of the user logged in	"Username"
<b>saveSettingsPreferences</b>	Bool	If the settings should be saved	False
<b>returnToHome</b>	Bool	If the settings should be saved	False
<b>blockInvites</b>	Bool	If user wants to block invites	False
<b>blockFriendRequests</b>	Bool	If user wants to block friend requests	False
<b>publicProfile</b>	Bool	If user wants to profile to be public (viewable by anyone not a friend)	False
<b>saveEvents</b>	Bool	If the user wants to save events past 30 days	False

## PROTOTYPING TOOLS

### Structure Chart



To begin, the user experiences a loading screen. This then automatically leads to the LoginScreen\_Load module which loads the welcome/login screen for the user. Here, the user must enter a username and password and click the login button (showcasing the pass of username and

password to the click function). If the password matches, then a flag **passwordMatches** will be sent back to the LoginScreen and the loop will be left – loading the HomeScreen. However, the user may not have an account yet so instead can also enter a new username, password, email and terms and conditions (to register a new account). These parameters are sent to the RegisterButton\_Click module when the user has complete the registration process to create an account. However, this does not return a flag because the user will still need to login with the LoginButon\_Click module using these new credentials to get the passwordMatches flag which will redirect them to the homescreen. Thus, these modules are repeated (also allowing users to create many new accounts without having to log in).

Once the home screen is loaded, due to the event-driven nature of winForms, the user can use the side nav buttons (MyProfile\_ButtonClickLoad, MyEvents\_ButtonClickLoad, MyBudget\_ButtonClickLoad, MyFriends\_ButtonClickLoad and Settings\_ButtonClickLoad) to open the according form. These modules are all passed the username parameter (**because the password and most other necessary information is accessible from the EC2 MySQL Database elaborated in the DataFlowDiagrams**).

In the MyProfile module which loads the MyProfile screen, the user can accordingly to the modules load a separate form allowing them to change their password where they will then click a button to confirm this change, delete their account (which uses their username parameter and password parameter to confirm their action) and post a new update.

Next, the My Events screen allows users to create a new event by using the create new event button. They can also accept invites by passing the selected parameter when the Accept button is activated.

The user can load the MyBudget interface.

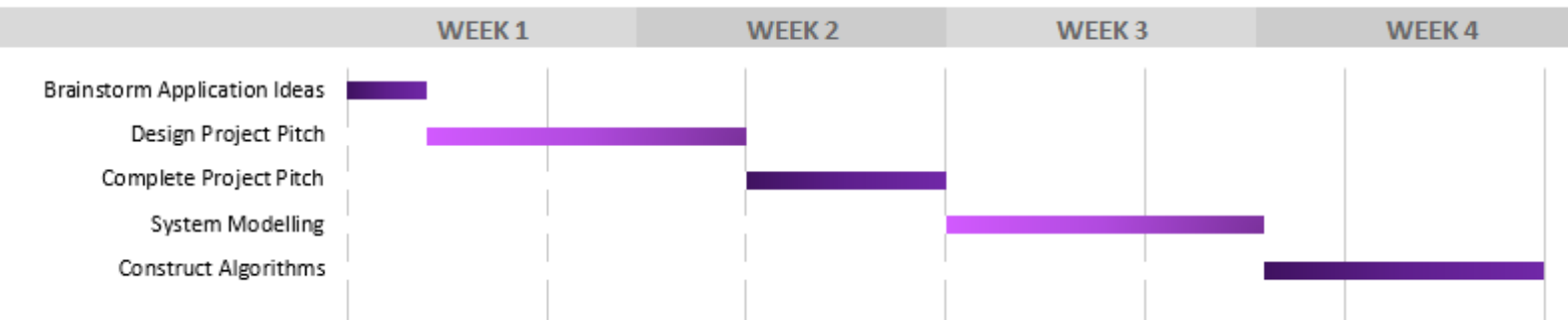
Users can also load the MyFriends module and here there is a changeAllocatedBudget subroutine which will allow them to properly change the budget associated with their username (thus passing the username parameter).

Lastly, users can load the Settings form and here they can either delete their account (using the username and password parameters to confirm deletion and prevent accidental removals) or they can use the saveSettings button. This button receives the parameters of the current settings variables.



## REQUIREMENTS REPORT AND PROJECT PLAN

# RESOURCES ALLOCATION PLAN AND GANTT CHART



### RESOURCES ALLOCATION PLAN AND GANTT CHART

Brainstorm Application Ideas

**DEPENDENCIES:** None

**SOURCING AND ALLOCATION OF -**

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Paper

**TIME RESOURCES:** 3 days

**DESCRIPTION:** Generate ideas for a problem, target audience and interface. Consider scope of assignment and prepare necessary technical resources.

### RESOURCES ALLOCATION PLAN AND GANTT CHART

Design / Complete Project Pitch

**DEPENDENCIES:** Brainstorm Application Ideas

**SOURCING AND ALLOCATION OF -**

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Powerpoint software

**TIME RESOURCES:** 11 days (design and complete)

**DESCRIPTION:** After creating ideas for the program, consolidate information into a brief but concise project proposal to pitch the idea. Involve information such as purpose of software, environment, intended users and concept prototypes.

**DEPENDENCIES:** Complete Project Pitch

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### System Modelling

#### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Modelling software

**TIME RESOURCES:** 11 days

**DESCRIPTION:** Once the pitch has been complete and the vision for the program has been realised, begin modelling the system to create and convey the software implementation of the desired structure and functionality. Produce storyboards, interface mock-ups, a dataflow diagram, data dictionaries and structure charts.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Construct Algorithms (DELAYED)

**DEPENDENCIES:** System Modelling

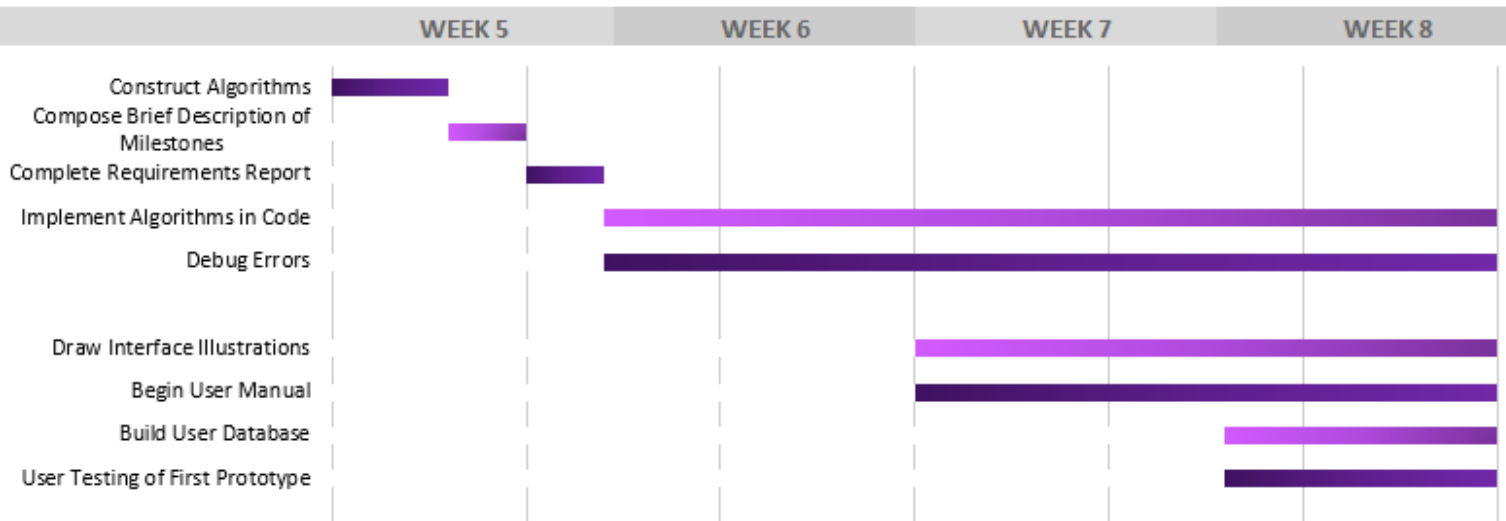
#### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Software or paper to compose algorithms

**TIME RESOURCES:** 14 days

**DESCRIPTION:** Now the idea has been designed into modules, create algorithms in pseudocode to model and describe each function in preparation for implementation



## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Construct Algorithms (CONTINUED)

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Compose Brief Description of Milestones

## **SOURCING AND ALLOCATION OF**

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Gantt chart software and text processor (for descriptions)

**TIME RESOURCES:** 3 days

**DESCRIPTION:** Since the software's modules have been outlined for implementation, resources including Human Resources, TECHNICAL RESOURCES and time can now be planned, allocated, prepared, and sourced in a Gantt chart. Compose a brief description for each milestone while outlining its dependencies.

## **RESOURCES ALLOCATION PLAN AND GANTT CHART**

### **Complete Requirements Report**

**DEPENDENCIES:** Compose Brief Description of Milestones

## **SOURCING AND ALLOCATION OF -**

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Text processor (for report)

**TIME RESOURCES:** 3 days

**DESCRIPTION:** As a developer it is necessary to have a complete understanding of the purpose, environment, and end user of an application. Now that the system's pitch and idea has been realised, system modelling has been complete and a Gantt chart produced, consolidate this information into a report that showcases this comprehension and appreciation. Additionally, compose a series of QA criteria.

## **RESOURCES ALLOCATION PLAN AND GANTT CHART**

### **Implement Algorithms in Code**

**DEPENDENCIES:** Complete Requirements Report, Construct Algorithms

## **SOURCING AND ALLOCATION OF -**

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** VS Community C# Winforms, Virtual Server, sufficient hardware to program

**TIME RESOURCES:** 60 days (total)

**DESCRIPTION:** Once the requirements report has been complete that contains the system modelling tools which allows the creation of algorithms, begin implementing this in code for the software.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Debug Errors

**DEPENDENCIES:** Implement Algorithms in Code

#### **SOURCING AND ALLOCATION OF -**

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** VS Community C# Winforms, Virtual Server, sufficient hardware to program

**TIME RESOURCES:** 60 days (total)

**DESCRIPTION:** Whilst implementing the algorithms, ensure simultaneously debugging errors and recording their nature and occurrence for later documentation (Testing and Evaluating report). Additionally, incorporate amendments and improvements on any algorithms.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Draw Interface Illustrations

**DEPENDENCIES:** Implement Algorithms in Code, Debug Errors

#### **SOURCING AND ALLOCATION OF -**

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Adobe Photoshop, Winforms VS Community

**TIME RESOURCES:** 46 days (total)

**DESCRIPTION:** Now that implementation in code has begun and the software is being created, design interface graphics accordingly.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Begin/Complete User Manual

**DEPENDENCIES:** Implement Algorithms in Code, Debug Errors

#### **SOURCING AND ALLOCATION OF -**

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Text processor (to create user manual)

**TIME RESOURCES:** 34 days (total)

**DESCRIPTION:** While coding and producing interface graphics, the end user should be consistently considered, which will be achieved by simultaneously creating documentation for the user. As a 200 word documentation must be submitted, it is also most time-effective to create this as the program is being produced.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Build User Database

**DEPENDENCIES:** Implement Algorithms in Code, Debug Errors

#### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** EC2 Instance, Winforms VS Community

**TIME RESOURCES:** 7 days

**DESCRIPTION:** Configure the EC2 server, create the MySQL database and connect with the program. Populate with test data and users (and setup for later use with the program)

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### User Testing of First Prototype

**DEPENDENCIES:** Implement Algorithms in Code, Debug Errors, Complete User Manual

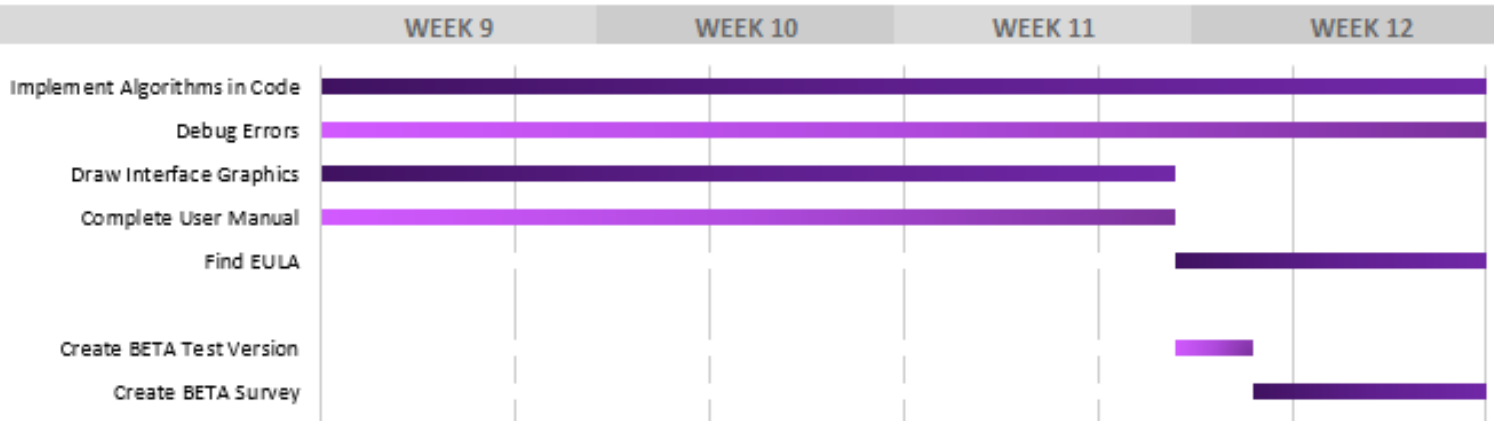
#### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** 5+ Human Testers

**TECHNICAL RESOURCES:** Winforms VS Community, If possible different WindowsOS devices with internet access (preferably that can run simultaneously)

**TIME RESOURCES:** 7 days

**DESCRIPTION:** In preparation for the Beta test task and to ensure that the software is user-friendly, reliable, and consistently error-free to avoid any major functionality issues, begin informal user testing of the software. Furthermore, develop ideas for a Beta test survey.



## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Implement Algorithms in Code (CONTINUED)

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Debug Errors (CONTINUED)



## RESOURCES ALLOCATION PLAN AND GANTT CHART

Begin/Complete User Manual (CONTINUED)

## RESOURCES ALLOCATION PLAN AND GANTT CHART

Find EULA (End User Licence Agreement)

**DEPENDENCIES:** Implement Algorithms in Code, Debug Errors, Complete User Manual

### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Word processor, Internet browser

**TIME RESOURCES:** 7 days

**DESCRIPTION:** While arguably no dependencies are required for this milestone, a holistic understanding of the program will inform the best-suited approach. Appropriate this EULA for the EULA task.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

Create BETA Test Version

**DEPENDENCIES:** Implement Algorithms in Code, Debug Errors, Complete User Manual

### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** EC2 Instance, Winforms VS Community

**TIME RESOURCES:** 2 days

**DESCRIPTION:** Prepare a version of the software to a test-able state for the BETA test task. While development of the program continues, this version will maximise the BETA test.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

Create BETA Test Survey

**DEPENDENCIES:** Create BETA Test Version

### SOURCING AND ALLOCATION OF -

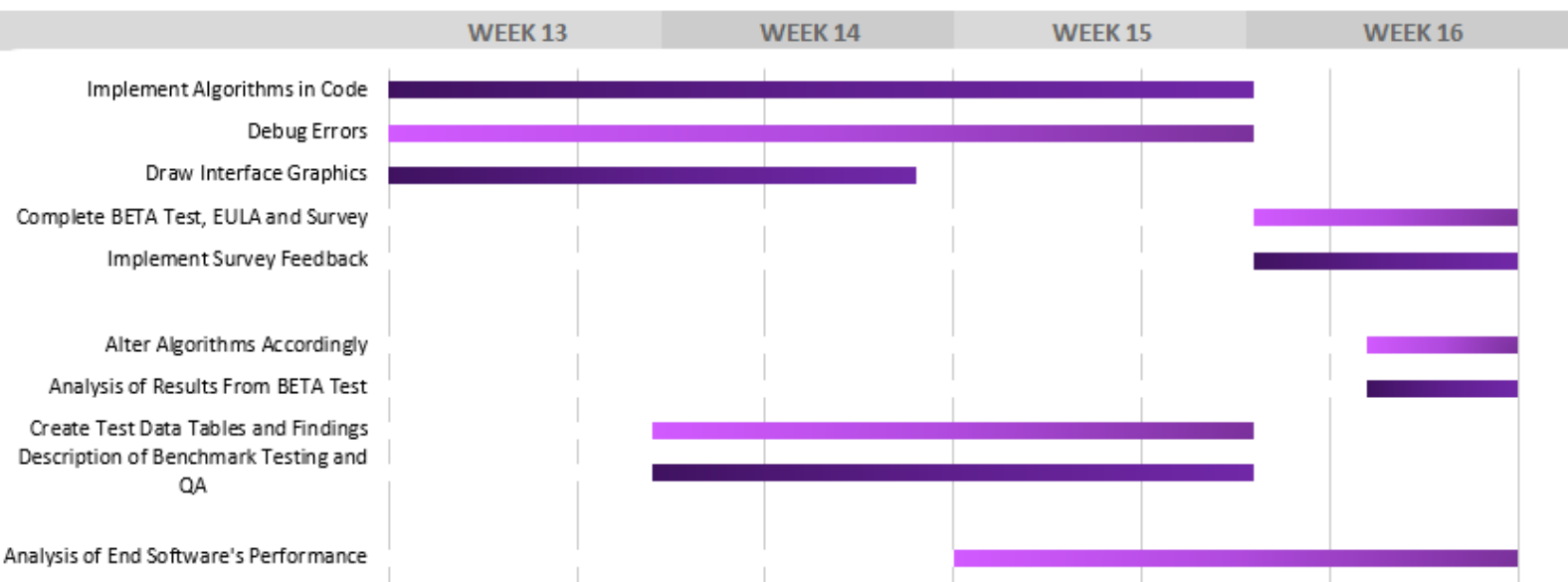
**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Word processor, Survey tool (such as Google Forms)

**TIME RESOURCES:** 6 days

**DESCRIPTION:** With the content and scope of the BETA test version of the software in mind, design a survey for testers to complete that will maximise the feedback from the testing. Additionally,

organise 10+ 'public' testers (especially with diverse backgrounds due to the broad nature of CONNECT's target audience)



## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Implement Algorithms in Code (CONTINUED)

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Debug Errors (CONTINUED)

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Draw Interface Graphics (RESUMED/CONTINUED)

Paused for 7 days during BETA testing preparation as BETA versions of the software can use placeholder graphics. Instead, it is more important that the software has majority of its functionality.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Complete BETA Test, EULA and Survey

**DEPENDENCIES:** Create BETA Test Version, Create BETA Test Survey, Find EULA

#### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** Lead Developer, 10+ testers

**TECHNICAL RESOURCES:** Word processor, Survey tool (such as Google Forms), Winforms VS Community, If possible different WindowsOS devices with internet access (preferably that can run simultaneously)

**TIME RESOURCES:** 7 days

**DESCRIPTION:** Complete the Beta Test, EULA and Survey task using the material created during week 11 and 12. NOTE: Update or create new versions of the test program, EULA and survey if needed.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Implement Survey Feedback

**DEPENDENCIES:** Complete BETA Test, EULA and Survey

**SOURCING AND ALLOCATION OF -**

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Word processor, Survey tool (such as Google Forms)

**TIME RESOURCES:** 7 days

**DESCRIPTION:** Begin simultaneously implementing feedback while receiving feedback from the BETA test, EULA and Survey task to reduce workload.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Alter Algorithms Accordingly

**DEPENDENCIES:** Implement Survey Feedback

**SOURCING AND ALLOCATION OF -**

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Text processor

**TIME RESOURCES:** 4 days

**DESCRIPTION:** Now implementing near-final changes to the code, alter the algorithms that are to be submitted with the Testing and Evaluating report.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Analysis of results from Beta Test

**DEPENDENCIES:** Complete BETA Test, EULA and Survey

**SOURCING AND ALLOCATION OF -**

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Text processor

**TIME RESOURCES:** 4 days

**DESCRIPTION:** Compose an analysis of the results from the Beta testing for the Testing and Evaluating report while the testing is still recent and testers can still recall majority of the program's functionality in case brief clarification is required.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Create Test Data Tables and Findings

**DEPENDENCIES:** Implement Algorithms in Code, Debug Errors, Complete User Manual

#### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Text processor

**TIME RESOURCES:** 14 days

**DESCRIPTION:** Prepare for the Testing and Evaluating report by beginning to outline and produce test data tables for the program. Also, begin composing the report on these findings.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Description of Benchmark Testing and Quality Assurance

**DEPENDENCIES:** Complete Requirements Report

#### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Text processor

**TIME RESOURCES:** 14 days

**DESCRIPTION:** Outline and compose description of benchmark testing and quality assurance for the Testing and Evaluating report.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Analysis of End Software's Performance

**DEPENDENCIES:** Implement Algorithms in Code, Debug Errors, Complete User Manual

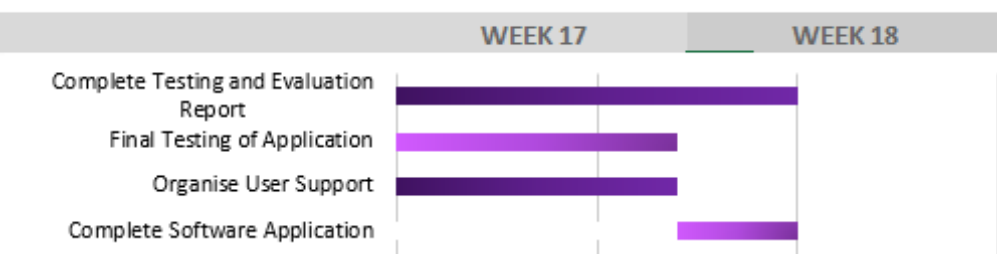
#### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Text processor

**TIME RESOURCES:** 14 days

**DESCRIPTION:** Compose an analysis of the software's end performance (that can be adjusted accordingly at the completion of the Testing and Evaluating report).



## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Complete Testing and Evaluation Report

**DEPENDENCIES:** Alter Algorithms Accordingly, Analysis of results from Beta Test, Create Test Data Tables and Findings, Description of Benchmark Testing and Quality Assurance, Analysis of End Software’s Performance

#### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Text processor

**TIME RESOURCES:** 10 days

**DESCRIPTION:** Compile, adjust and conclude the Testing and Evaluating Report. Include formal documentation of how testing and evaluating was executed and include all algorithms for the program.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Final Testing of Application

**DEPENDENCIES:** Implement Survey Feedback, Debug Errors Complete User Documentation

#### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** Lead Developer, if possible 2+ ‘public’ testers

**TECHNICAL RESOURCES:** EC2 Instance, Winforms VS Community

**TIME RESOURCES:** 7 days

**DESCRIPTION:** To ensure that the final state of the program is still error-free and complete, test the application completely and involve two ‘public’ testers as well.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Organise User Support

**DEPENDENCIES:** Complete BETA Test, EULA and Survey

#### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** Internet browser, Text processor

**TIME RESOURCES:** 7 days

**DESCRIPTION:** Required as part of the Software Application submission is 'appropriate user support'. Organise and create suitable support whether distributing the documentation or creating a community support form or Youtube demonstration videos.

## RESOURCES ALLOCATION PLAN AND GANTT CHART

### Complete Software Application

**DEPENDENCIES:** Implement Survey Feedback

#### SOURCING AND ALLOCATION OF -

**HUMAN RESOURCES:** Lead Developer

**TECHNICAL RESOURCES:** EC2 Instance, Winforms VS Community, Text Processor (Software Application and Testing and Evaluating Report), Display Folder, Printer

**TIME RESOURCES:** 4 days

**DESCRIPTION:** Conclude the project and submit the Software Application and PDR google drive folder through email and a PDR printed version in a display folder.



## REQUIREMENTS REPORT AND PROJECT PLAN

# QUALITY ASSURANCE

STATEMENT	TESTING CRITERIA
Smooth/Quick-Responding Interface	Manually test responds to buttons within 1 second
Communicates smoothly with server	Manually test server-related interactions complete within 1 second.
Organise events in objective manner	Software calculates priority and uses randomNumbers to create events.
User-friendly	Distribute 10+ program prototypes with a survey and receive positive/satisfied feedback. Interface uses consistent buttons and messageboxes.
Robust	Test each input with a variety of illegal data. Use checkboxes/comboboxes where possible Distribute 10+ program prototypes and ensure no errors occur.
Customisable	Distribute 10 program prototypes with a survey and receive positive/satisfied feedback. 4+ different settings.
Functions on Windows OS platforms	Test program works on range of 5+ windowsOS systems (with internet).
Connects users anywhere	Information is stored and accessed in a Cloud-server using the Internet. Test program works on range of 5+ systems (with internet).