Team F

CSCI3340.02 Software Engineering Fall 2019 Team Project Stage 1 Peer Evaluation Form			
Team Members	Completion of the assigned tasks (0-20)	Contribution to the team submission (0-5)	Brief comments
Name of Team Member #1 Gwen	20	5	
Name of Team Member #2 Aldo	20	5	
Name of Team Member #3 Edward	20	5	
Name of Team Member #3			

I just want to add that we have all been attending weekly meetings and brain storming ideas. As of yet the team is functional in my mind.

Keith

High Level Requirements For the “when2meet” program

1. Introduction

Dev Ninjas: Gwen, Aldo, Edward, Keith

Team F

1.1 The Purpose

This projects mission is to aid in the planning of group meetings. Often it can be cumbersome to organize a meeting time for people with wildly different (and sometimes conflicting) schedules. We aim to help find a time slot everyone can make and comfortably if possible.

1.2 Scope:

To provide a platform for businesses to schedule meetings without multiple email chains.

1.3 Definitions:

Type 1 Use – defining basic time slot for 25 users or less.

GUI - (Graphical User Interface) a computer program designed to allow a computer user to interact easily with the computer typically by making choices from menus or groups of icons.

API - (Application Programming Interface) a set of rules that allows programmers to develop software for a particular operating system without having to be completely familiar with that operating system.

SQL - (System Query Language) database language.

1.4 References:

The non-GUI portions will be POSIX.1-2017 compliant.

Merriam-Webster dictionary used to define words.

1.5 Overview:

This document outlines the parameters for the when2meet application.

2. Overall Description

2.1 Product Perspective

2.1.1 System interface.

- The application may interface with Google API.
- The application will run on Windows 10.
- The application needs an internet connection.

2.1.2 User Interface.

The user will use a GUI that will allow the user to login with a username and password to view their personal meeting time and other information. See figures in Appendix A GUI Section.

2.1.3 Hardware Interface.

The user will use a keyboard and mouse to interact with the GUI.

2.1.4 Software Interface.

The application will use a MySQL database.

2.1.5 Communication Interface.

The user will receive email and pop up notifications from the application.

2.1.6 Operations.

- The application will subtract occupied time from the given time slots from the user.
- The user can import their calendar data into the program.

2.2 Product Function

- The program will return a time frame which maximizes the number of attendees. Meaning that the program will search for a time frame in which most people are unoccupied for a given period.

2.3 User Characteristics

- Languages supported: English.
- Computing Skills: Have basic knowledge of MS Outlook or similar calendar applications.
- Physical and Social Environment: This program will be appropriate for both corporate and family settings. Will take users' position in company/social group into account.

2.4 Constraints

- Must be human as per the Geneva convention of 1915 and pass a captcha.
- Must have at least Windows 10 1803 or Mac OS 10.14.6 and later.
- Work on Visual C++ 2015 and GNU gcc 9.2.0

3. Specific Requirements

3.1 External interfaces

3.1.1 User Interfaces

When2Meet will have a GUI to allow the user to interact with the program. It will have the tools needed to input and modify schedules, and it will provide means to request meetings from users in an opt-in manner.

3.1.2 Hardware Interfaces

When2Meet will focus on standard keyboard and mouse inputs for data acquisitions that will be translated into monitor output.

3.1.3 Software Interfaces

When2Meet will be a C++ program that uses the QT libraries to generate the GUI. It will also connect with MariaDB for its database needs, possibly on another computer; thus, requiring an internet connection.

3.1.4 Communications Interfaces

When2Meet might require internet at later stages to connect to remote server containing user data.

3.2 Functional requirements

3.2.1 Login requirements

- In order to log in users will need to have a unique username. This username will be their email address.
- Program will allow three (3) attempts at login. After three (3) unsuccessful attempts the system will lock the user out for 30 minutes and send an email that they should change their password.

3.2.2 Database

- At the creation of an account the user will be assigned a unique number (generated in accordance to ___ requirement). Users will be able to assign a one to many relation with that number to their email accounts.
- The unique ID number in conjunction to an email will be a composite key for the schedule entries.

3.3 Performance requirements

3.3.1 Memory Constraints

- The application will use at most 1 GB of RAM for Type 1 Usage.
- The application will use at least 256 kB for the internet connection

3.3.2 Computational constraints

3.4 Logical database requirements

3.4.1 MariaDB

MariaDB will be used as a relational database to store and query user information.

3.5 Design constraints

3.5.1 Standards compliance

Github repo will follow the linux Filesystem Hierarchy Standard 3.0 (FHS).

3.6 Software system attributes

3.6.1 Reliability

3.6.2 Availability

3.6.3 Security

- In order to log in, make changes to submitted schedule, or raise an exception the user must log in and or make an account.

- The user’s password will be a minimum of eight (8) characters long. It must contain at a minimum one number and special character. Special charactes are limited to the one to zero keys (shifted). For example Zucckerburged?0
- Passwords will be stored after going through a hashing algorithm.

3.6.4 Maintainability

3.6.5 Portability

High level functionality must be written to be platform agnostic. That is when possible the program will avoid using non-standard libraries.

3.7 Organizing the specific requirements

3.7.1 Database class

This class will create an object that has the needed methods for placing, retrieving and modifying attributes of the database.

3.7.2 Calendar class

Will be an object that will generate Gregorian Calendars to display user data. Create and handle time objects.

3.7.3 Meetings Class

This class will be tasked with finding the best times to meet. Will also allow for the acceptance and declining of meetings and notify of such actions.

3.8 Use Cases

3.8.1 Account Creation

1. At the log in screen user click “Create an account”
2. Fill in the data specified by data requirements.
3. Click create account.

3.8.2 Actions

3.8.3.1 scheduling meet

1. User logs in and enters dashboard.
2. User clicks on the create meeting button.
3. User selects time frame. That is day or week they want the meeting to be held; along with a list of users to invite.
4. Program returns two meeting options to host to select.
5. Have your meeting and save the world.

3.8.3.2 Accepting a meeting

1. User logs in and enters dashboard.
2. User clicks on notifications.
3. User sees proposed meetings they have been invited to and selects those they can attend.
4. Goes to meetings.

3.8.3.3 Adding data

1. User logs in and enters dashboard.
2. User clicks the Calendar button.
3. User clicks the add personal schedule button.
4. User inputs data into fields specified by the data requirements.
5. Click confirm.

3.8.3 Views

3.8.3.1 Calendar view

- Users can see their own data inputs.
- Users can modify their own data.

3.8.3.2 Upcoming Meetings / Dashboard

- Users will be able to see meetings that they have accepted too. It will be styled after UTRGV's Week at a glance.

3.8.3.3 Meeting invites / Notifications

- Users will be able to see proposed meetings that they have been invited to.
- Users will be able to see reminders to upcoming meetings.
- Users can accept or decline invites from this view.

3.9 Data collection

3.9.1 Creating an account

- A valid email address will be needed. This is both for generating composite keys in our database and sending out notifications.
- A password that is in compliance with 3.6.3 Security.

3.9.2 Inputting Personal Calendar

- A short (15 character length) title for the personal event.
- The time of said event will take place in HH:MM am/pm format.
- The date of said event will take place in DD/MM/YYYY format.

- An optional field for comments (40 character length).

3.9.3 Requesting a meeting

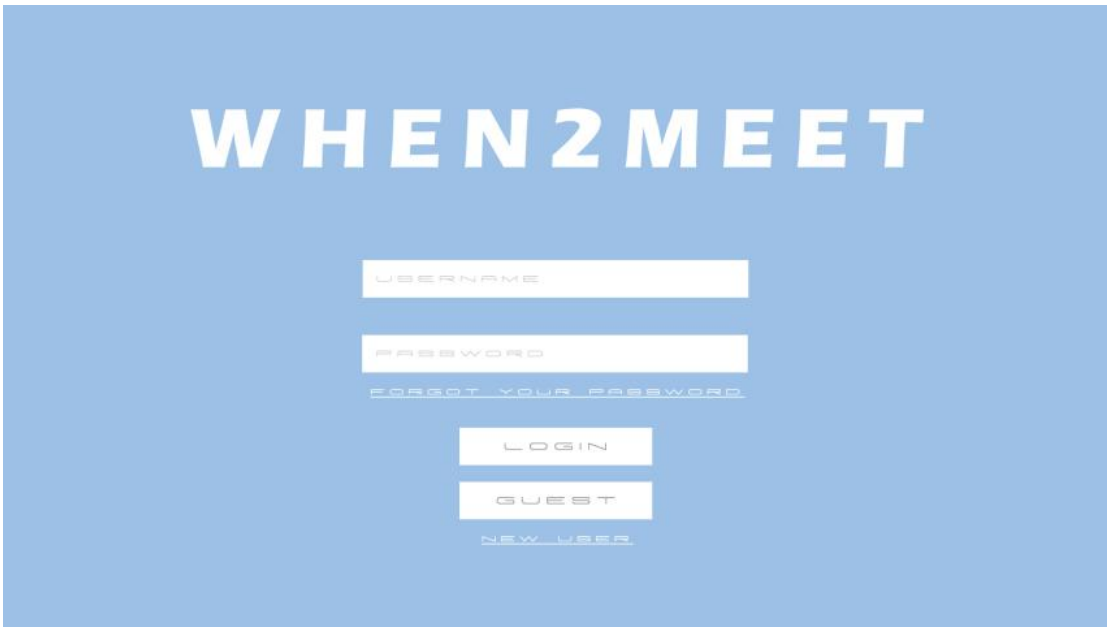
- The emails of people who’s participation is requested.
- A period of time when the meeting should happen. For example tomorrow or between 04/01/2020 and 04/09/2020.

3.10 Additional comments

Appendix A

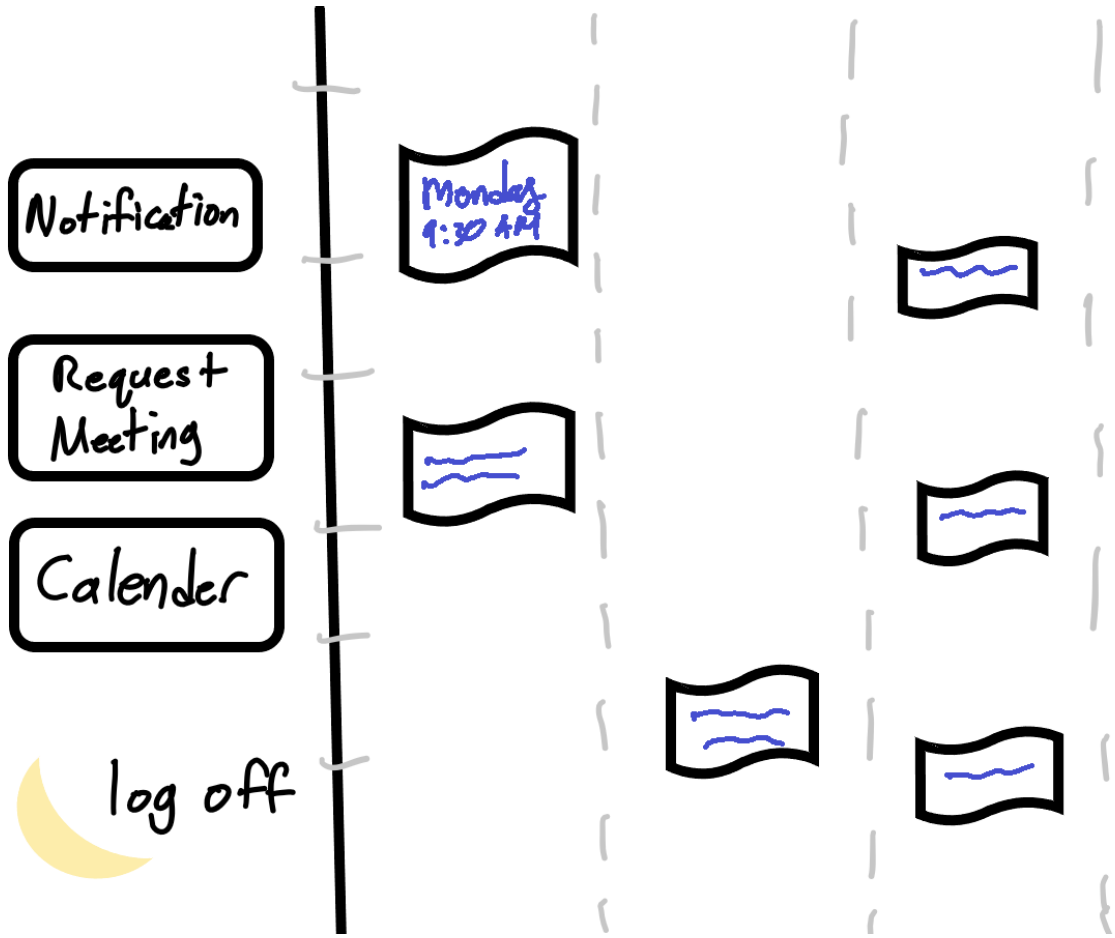
GUIs

Log in Screen



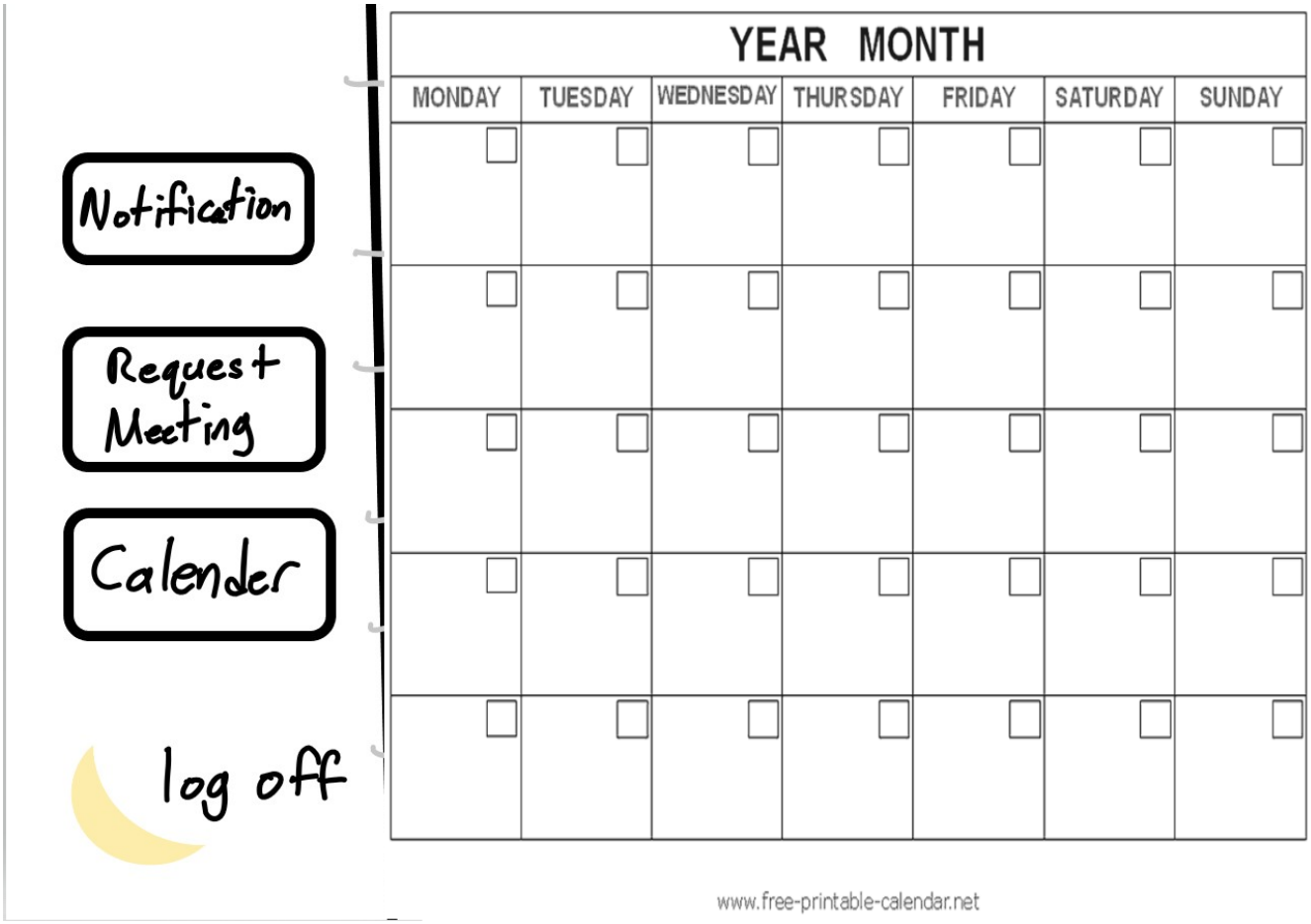
Main Menu / Dashboard

Team F



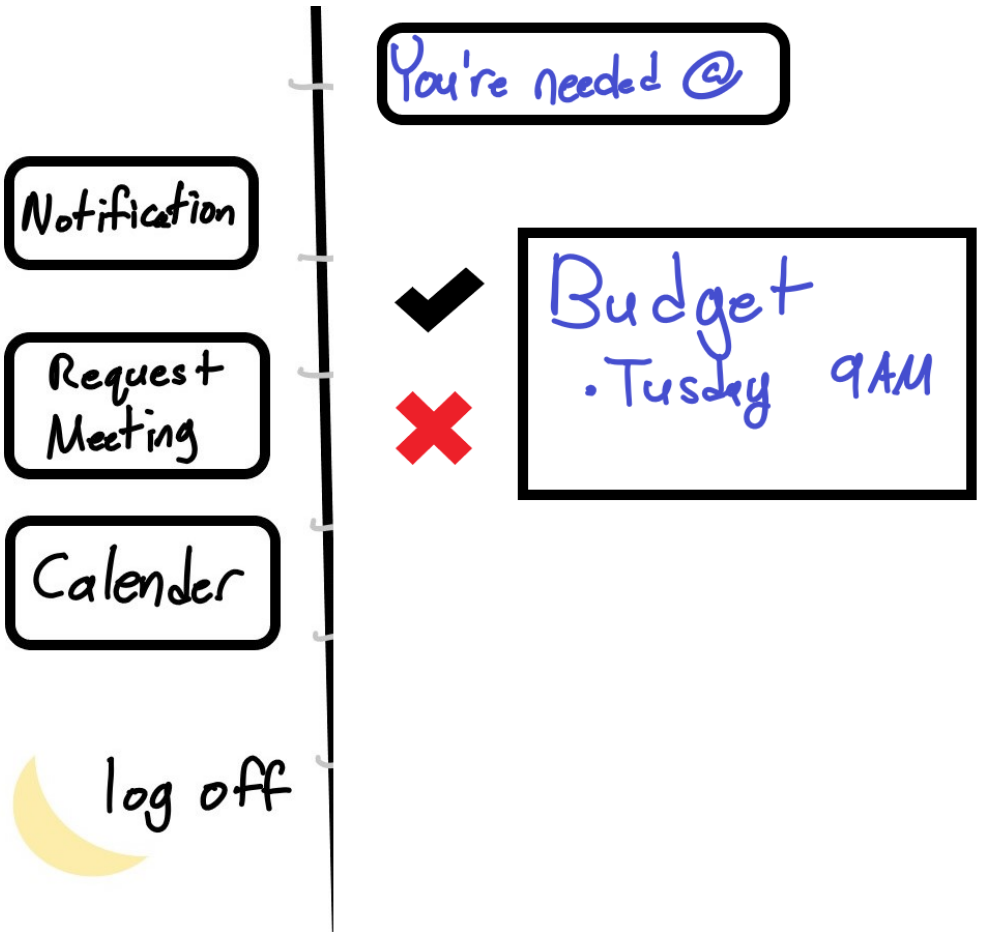
Calendar Menu

Team F



Notifications Menu

Team F



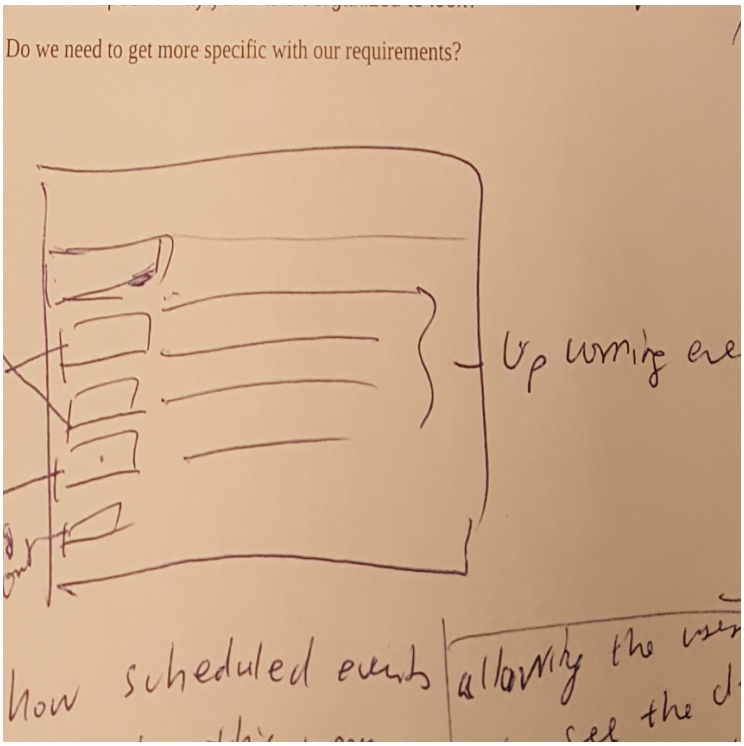
Questions

- What features must the home dashboard have?
- Is there a specific way you want it organized to look?
- Do we need to get more specific with our requirements?
- How do we handle guests?

Team F

Lessons Learned

- We need to ask ourselves if the requirements we have written are enough to design/code off of
- Image of basic dashboard idea



- Assume guests are always available and only need an email