*Santa Clara University*

*COEN 174L: Software Engineering Lab*

**Group 5**

**Design Document**

By

Giovanni Briggs

Justin Wong

Alex Hoff

October 12th, 2015

Contents

[Introduction 1](#_Toc432403105)

[Requirements 2](#_Toc432403106)

[Functional Requirements 2](#_Toc432403107)

[Non-functional requirements 2](#_Toc432403108)

[Design Constraints 2](#_Toc432403109)

[Use Cases 3](#_Toc432403110)

[Submit request to join via a form 3](#_Toc432403111)

[Verify Request to Join 4](#_Toc432403112)

[Access Waitlist for all courses offered by a department 4](#_Toc432403113)

[Configure and Maintain System 5](#_Toc432403114)

[Activity Diagram 6](#_Toc432403115)

[Conceptual Model 6](#_Toc432403116)

[Technologies Used 9](#_Toc432403117)

[Architectural Design 9](#_Toc432403118)

[Design Rationale 10](#_Toc432403119)

[Test Plan 10](#_Toc432403120)

[General usability 10](#_Toc432403121)

[Form Functionality 10](#_Toc432403122)

[Waitlist Access 10](#_Toc432403123)

[Waitlist generation 10](#_Toc432403124)

[Verification E-mails 10](#_Toc432403125)

[Risk Analysis 11](#_Toc432403126)

# Introduction

Recently, students are having a harder time graduating in four years at colleges and universities across the country. Higher class enrollment creates more competition to get into classes and thus a greater need for the use of waitlists to track which student is next to enroll in a class should a seat open up for him or her.

Unfortunately, waitlists are often not as streamlined and efficient as they could be. At Santa Clara University, the original waitlist system required a student to e-mail a professor or department administrative assistant with his or her name, student ID, class section number and title, and the reason for addition to the waitlist. However, this system made a professor and an administrative assistant maintain two separate lists for the same course. Due to a lack of communication between the two individuals, no real structure or order existed, and students had no idea where they fell on a given waitlist. Professors were unable to accurately prioritize a waitlist because they did not have a complete picture of who was still waiting on a given class.

Santa Clara University’s School of Engineering saw a need for change and instructed all engineering students to e-mail a department’s administrative assistant when the student wanted to join a waitlist. This new system created one easily maintained list for each course, but also resulted in more traffic being directed towards the administrative assistants. Most engineering students do not seem to be aware of the current policy and still e-mail a professor directly requesting to join the waitlist for a course. Oftentimes, a student does not include the correct information in his or her request, resulting in more overhead for the administrative assistant as he or she has to e-mail the student back with further instructions. An administrative assistant is also responsible for managing the waitlists for multiple classes and sometimes misses a student’s request to join. Cross-listed courses further complicate matters since each department’s administrative assistant maintains his or her own waitlist for the same course.

We propose a website that will allow a student to join a waitlist for a particular class. A student can fill out a form with all necessary information including name, e-mail, SCU ID number, the class he or she wants to get into, and why he or she needs to be in that class. An administrative assistant will have the opportunity to access the waitlist and a professor will be able to prioritize the waitlists for his or her classes. This system will also provide students with feedback about their spots on the waitlist so they will know in advance if they should make other plans. We intend to make the system easily configurable so that it requires low levels of maintenance while also providing an easy to use interface so that students can quickly join a waitlist and administrative assistants can focus on placing students into classes.

# Requirements

## Functional Requirements

* Critical
  + The system will keep a waitlist in a database that can only be accessed by an admin
  + The system will keep one list for cross listed classes
  + The system will allow students to input their required information
  + The system will allow students to choose their class they want to wait list from a series of two drop down lists (by dept and class section)
  + The system will allow admins to change students priorities
* Recommended
  + The system will allow students to see what number they are on the waitlist
* Suggested
  + The system will allow students to remove themselves from the waitlist
  + The system will allow admins to remove students from the waitlist

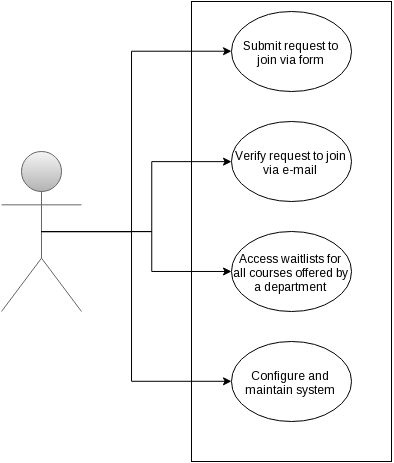
## Non-functional requirements

* Fast and easy to use for both students and admins
* Secure
* Aesthetically appealing

## Design Constraints

* Web based
* Run on design center machines

# Use Cases



Figure

## Submit request to join via a form

**Actor**: Student

**Goal:** Add student to specified waitlist

**Preconditions:**

1. The course exists
2. All fields on form are filed out with valid data

**Postconditions:**

1. System sends verification via email to student

**Steps:**

1. Student accesses form via web browser
2. Student fills out each field
   1. Department offering course
   2. Course’s Name/Section
   3. Student’s Name
   4. Student’s ID
   5. Student’s e-mail
   6. Reason why student needs to be in class
3. Student press submit button

**Exceptions:** N/A

## Verify Request to Join

**Actor**: Student

**Goal:** Verify that the student actually submitted the request to join and then add them to the waitlist

**Preconditions:**

1. Someone requested to add the user with the specified e-mail to a waitlist
2. Student received confirmation e-mail

**Postconditions:**

1. User confirms or denies that he or she is the one who submitted the request and that he or she wants to be added to the waitlist
2. User is added to the waitlist
3. Splash screen displays student’s position on waitlist

**Steps:**

1. User opens e-mail
2. User clicks on verification link
3. The waitlists are divided by quarter and year. There will be one link for each
   1. Example: 2015 Fall

**Exceptions:** N/A

## Access Waitlist for all courses offered by a department

**Actor**: Administrative assistant or Professor

**Goal:** Download waitlist for department

**Preconditions:**

1. There is at least one person on any waitlist for any class offered by that department

**Postconditions:**

1. User downloads local copy of all waitlists for courses offered by that department

**Steps:**

1. User logins in using their department’s username and password
2. User clicks on link to download the appropriate waitlist
   1. The waitlists are divided by quarter and year. There will be one link for each
      1. Example: 2015 Fall

**Exceptions:** N/A

## Configure and Maintain System

**Actor**: System admin

**Goal:** Configure and maintain system

**Preconditions:**

1. N/A

**Postconditions:**

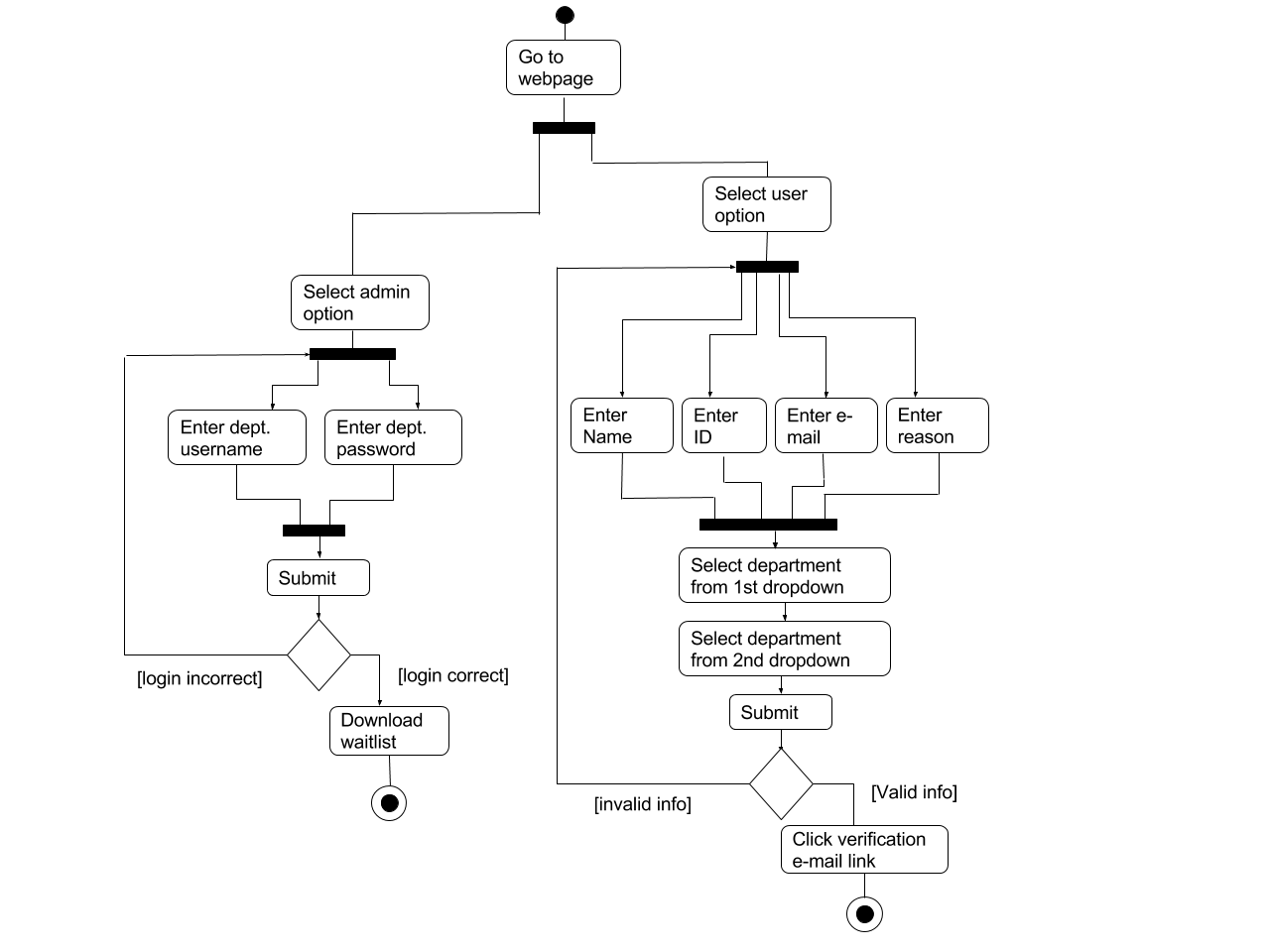
1. System is ready to run

**Steps:**

1. User goes to *admin* page
2. User logs in using correct credentials
3. User has access to:
   1. *Courseavail address*: the address pointing the system to Courseavail in order to pull the right course information
   2. *Database*: all waitlist and departmental information. System admins will be able to maintain the database for optimal performance and perform tasks such as changing department passwords or adding new departments as needed.

**Exceptions:** N/A

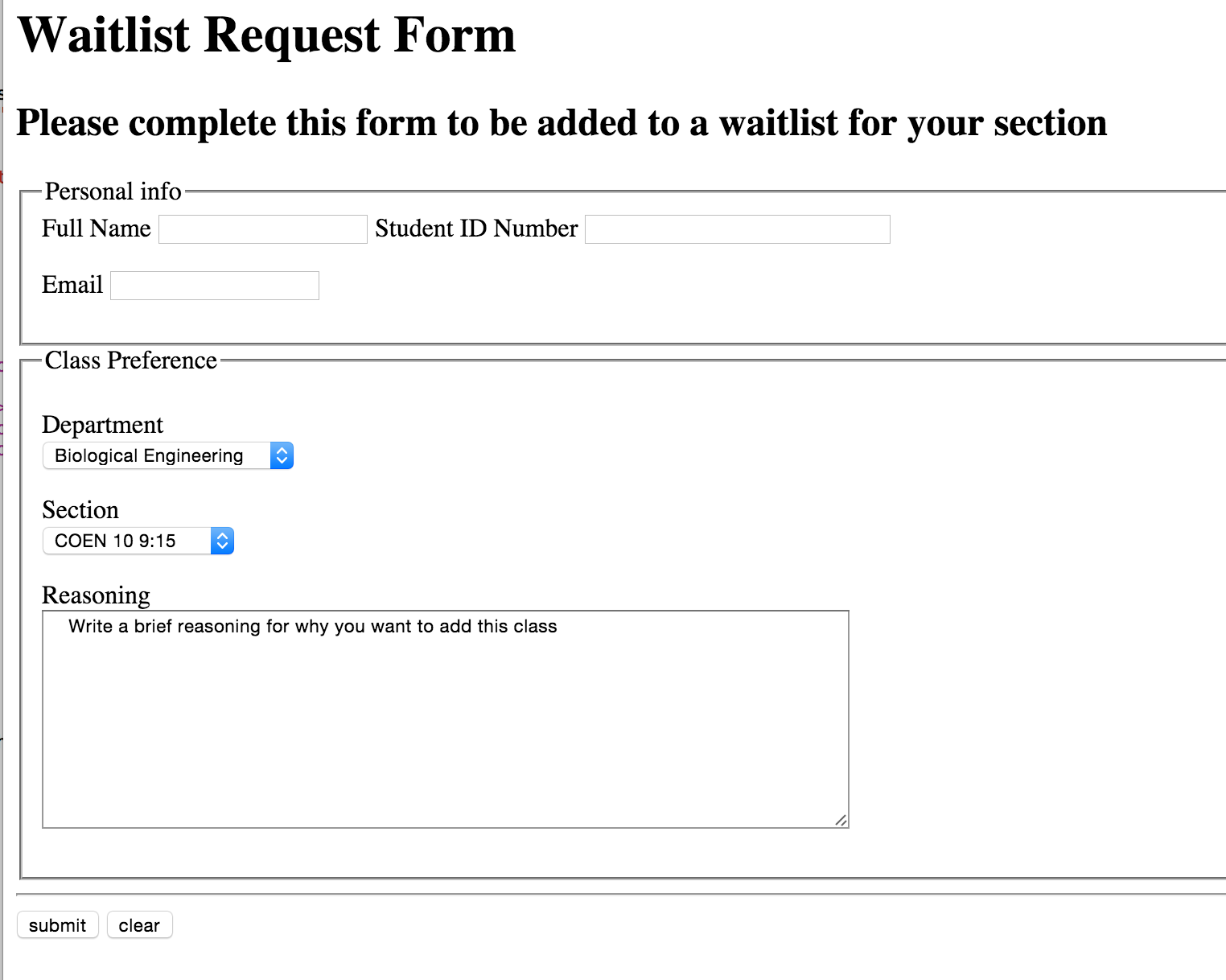
# Activity Diagram



Figure

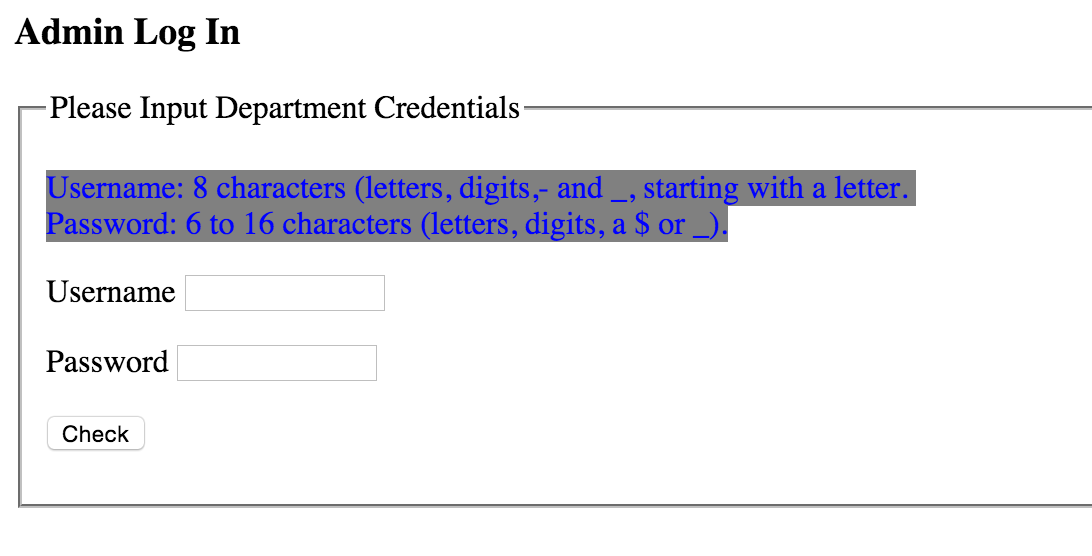
# Conceptual Model

Student users will first be presented with a form much like the one below. They will be able to input their credentials and then choose which class section they would like to be added to using the two drop down menus. The second drop down menu will populate based on the department chosen in the first drop down. Students will then give a short reason for why they would like to add the class and then submit the form which it will then be checked to make sure the personal information is correct. The completed form will then be sent via email to the student where they will click on a link in the email to confirm the validity of the form. The student will then be added to the appropriate waitlist which will display all of their information for the admin to decide on priority.



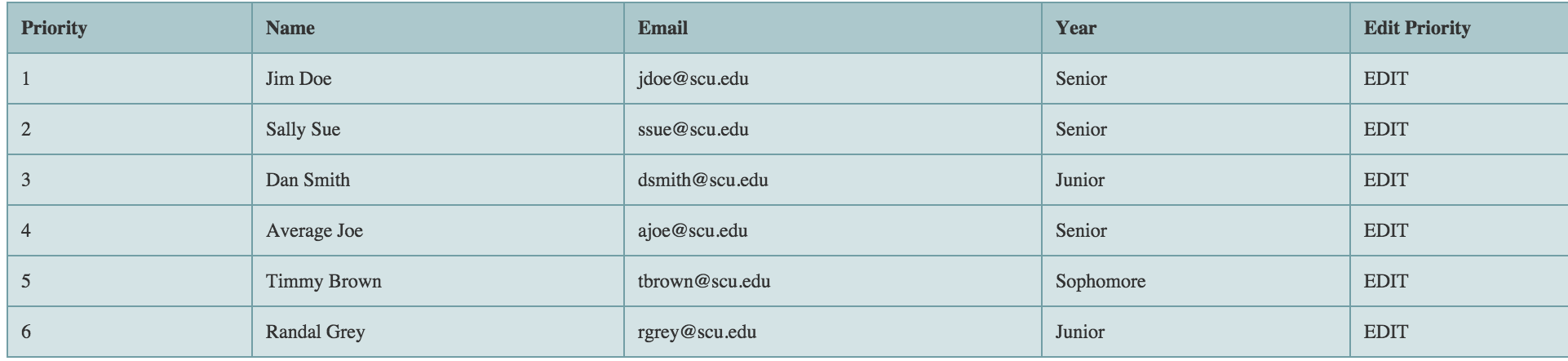
Figure

The admin will log into the waitlist using their departments credentials using a form like the one displayed below.



Figure

The admin will then be presented with a table listing the waitlists for all class sections in their department with the students names, emails, years, and reasoning for adding the class. The admins can then edit a student's priority moving them up or down the wait list or remove students they do not deem necessary to be in the waitlist or send permission numbers to students they wish to add.



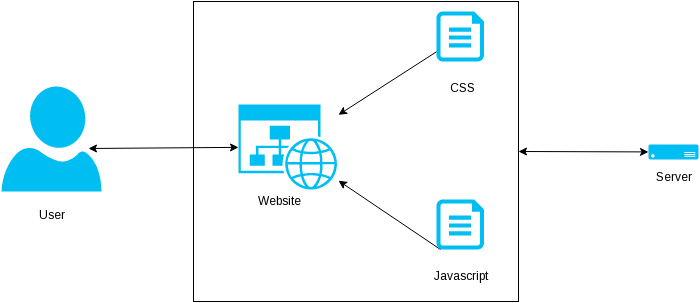
Figure

# Technologies Used

* HTML
* Javascript
* Bootstrap
  + HTML, CSS, and Javascript framework designed for rapid web development  
    Javascript - High-level programming language used for web development that executes on the user’s computer.
* \*MySQL
  + A relational database management system used to provide a high amount of scalability to information storage.
* PHP
  + High-level programming language used for web development that executes on the server-side.

# Architectural Design

Our system implements a multi-page website where each page meets a different functional requirement. We have multiple classes of users, but each class only needs access to certain functionality. Students only need access to the request form; whereas, department administrative assistants need access to the actual waitlists for his or her respective departments. Segregating the website based on functionality allows us to better meet the needs of each user group specifically, rather than trying to do a one page fits all method. Each page will have utilize CSS and Javascript and some pages will need to communicate with the server in order to complete tasks.



# Design Rationale

We designed our system in order to meet all functional requirements while keeping the system easy to use and maintain. We have identified three different classes of users, and our web-based solution will have specific web-pages geared to each class. This will allow each class of user to quickly access the information that they need. The use of e-mail verification allows us to implement some form of authentication without having to interface directly with Santa Clara University’s variety of authentication systems. Using Courseavail makes the setup of the system easier, since all a system admin needs to do is point the server to the correct Courseavail address. The system will then automatically pull in class information so departments don’t need to enter their course offerings in multiple places. This method has been proven to work by the SCU Classes (<http://www.scuclasses.com>) website which is also consistently ripping the data from Courseavail. Doing this ensures that the system has minimal overhead while still providing the essential functionality.

# Test Plan

## General usability

1. Test all pages on different operating systems with different browsers to make sure that the site is truly cross platform

## Form Functionality

1. Check form validation functions by submitting faulty requests
2. Submit valid requests and make sure that user is directed to appropriate pages afterwards

## Waitlist Access

1. Test login by supplying correct and incorrect credentials
2. Test download functionality by attempting to download available waitlists
3. Test waitlist viewing functionality by attempting to load the waitlist in browser and make sure that the browser displays all available data for that waitlist

## Waitlist generation

1. Submit requests through form, validate the requests and then check the waitlist to see if all information went through correctly

## Verification E-mails

1. After submitting a form, check to see if server sent corresponding confirmation e-mail
2. Check to make sure confirmation e-mail link is functional by clicking on link and then checking waitlist to see if data is there

# Risk Analysis

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Risks** | **Consequences** | **Probability** | **Severity** | **Impact** | **Mitigation Strategy** |
| Illness/Heavy homework load | Teammate unable to help with completion of project | .3 | 3 | .9 | -Schedule time wisely so appropriate time for sleep can be given  -eat a healthy diet |
| Too much functionality | Unable to complete the project on time | .2 | 8 | 1.6 | -Design the project with reasonable functionalities  -Create a timeline to finish functionalities on time |
| Poor Communication | Bad chemistry between the team, working on the same functionalities, slow development | .2 | 10 | 2 | -Create group message so all group members are in the know at all times  -Create group hierarchy |
| Incorrect Requirements | Product is not completed as the customer wished | .4 | 10 | 4 | -Constantly communicate with customer about questions about the project  -Have the customer test the project throughout development |
| Insufficient Technical Ability | Group members forced to spend large amounts of time learning new skills resulting in slow development | .2 | 8 | 1.6 | -Communicate within the group who has what skills  -identify necessary new skills early in development so they can be learned ahead of time |

# Development Timeline

Giovanni Justin Alex All

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Task | Wk 1 | Wk 2 | Wk 3 | Wk 4 | Wk 5 | Wk 6 | Wk 7 | Wk 8 | Wk 9 | Wk 10 |
| Design Document |  |  | Edit | Comp |  |  |  | Test | Test | Test |
| Write Student Form |  |  |  | Dev | Dev | Dev | Comp | Test | Test | Test |
| E-mail verification system |  |  |  |  | Dev | Dev | Comp | Test | Test | Test |
| Setup Access to Courseavail |  |  |  | Dev | Comp | Test | Test | Test | Test | Test |
| Generate waitlists |  |  |  | Dev | Dev | Comp | Test | Test | Test | Test |
| Waitlist view/download pages |  |  |  | Dev | Dev | Comp | Test | Test | Test | Test |