

CS 320 Course Project Final Report

Schedule Helper

Version <2.0>

Prepared by

Group Name: Test Team

|  |  |  |
| --- | --- | --- |
| Rawad Bader | 011678766 | rawad.bader@wsu.edu |
| Steven Arbuckle | 011689992 | Steven.arbuckle@wsu.edu |
| Caleb Hooper | 011684910 | caleb.hooper@wsu.edu |
|  |  |  |
|  |  |  |

|  |  |
| --- | --- |
|  |  |
| Date: | 12/13/2019 |
|  |  |
|  |  |
|  |  |

Contents

Contents ii

1 Introduction 1

1.1 Project Overview 1

1.2 Definitions, Acronyms and Abbreviations 1

1.3 References and Acknowledgments 1

2 Design 2

2.1 System Modeling 2

2.2 Interface Design 2

3 Implementation 3

3.1 Development Environment 3

3.2 Task Distribution 3

4 Testing 4

4.1 Testing Plan 4

4.2 Tests for Functional Requirements 4

4.3 Tests for Non-functional Requirements 4

4.4 Hardware and Software Requirements 4

5 Analysis 5

6 Conclusion 6

Appendix A - Group Log 7

# 

# Introduction

This document is the Software Requirements Specification (SRS) for the Web Application to help people with their schedule.

This document was developed by our team for the Fundamentals of Software Engineering

course offered by the Department of Computer Science at the Washington State University Vancouver.

This project is a tool to help in managing people schedule and published to other team members. It is more useful in a school project team and it organize people free time for meeting to work in a specific martial. In this case sharing document & data related to project the team working on it.

## Project Overview

The purpose of this document is to present a detailed description of the Web scheduler system that helps the member of team to manage their time and availability for work or any other subject they are working on. It will explain the purpose and features of the system, the interfaces of the system, what the system will do to manage the register entries from a different people, and make sure there is empty space for each request. This development tools will make tasks easy for people trying to fit manage their schedule for study groups or meetings for school related project.

This software system will be a Web scheduler system will be running on a local editor system that requires login administers and proved the code for the people only will be included in the project or work in the same job. This system will have error check system to make sure the people do not pick the similar days in the schedule, which would otherwise have to be performed manually. By adding the error checking the amount of work and effort will be minimized which make setting the schedule efficient then setting it manually.

More specifically, this system is designed to allow a student to manage and communicate with a group to manage their time accordingly and make their schedule available to view by every member of the team. The application as a mentioned before it let the person what are the spots free to add their schedule. The platform of this web application will be designed mainly in JavaScript.

## Definitions, Acronyms and Abbreviations

Admin Administrator

EDS ﻿Event-Driven System

HTML Hyper Text Markup Language

JS JavaScript

PM Project Manager

RSC Research, Development, Test and Evaluation Directorate.

SLT Software Lifecycle Tools

SD Sequence diagrams

SRS Software Requirements Specification

TDD ﻿Test-Driven Development

UML Unified Modeling Language

﻿

## References and Acknowledgments

This document is the Software Requirements Specification (SRS) for the Web Application to help people with their schedule.

This document is based on IEEE Std 830-1998 [1].

# Design

## System Modeling

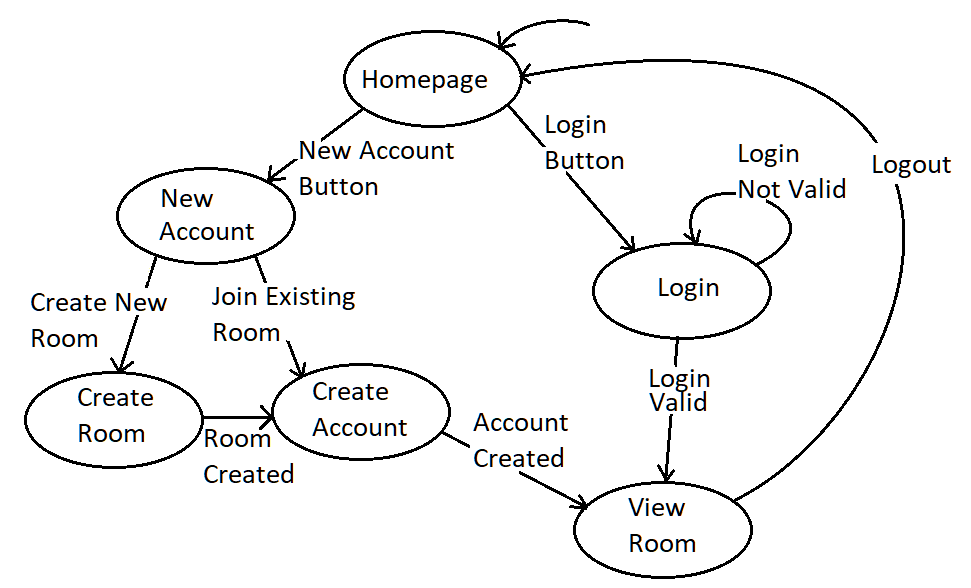
< Update your UML diagrams in milestone 2, to reflect the real implementation of this software.

Figure 1 A state diagram that represents the event-transitions of the web-application

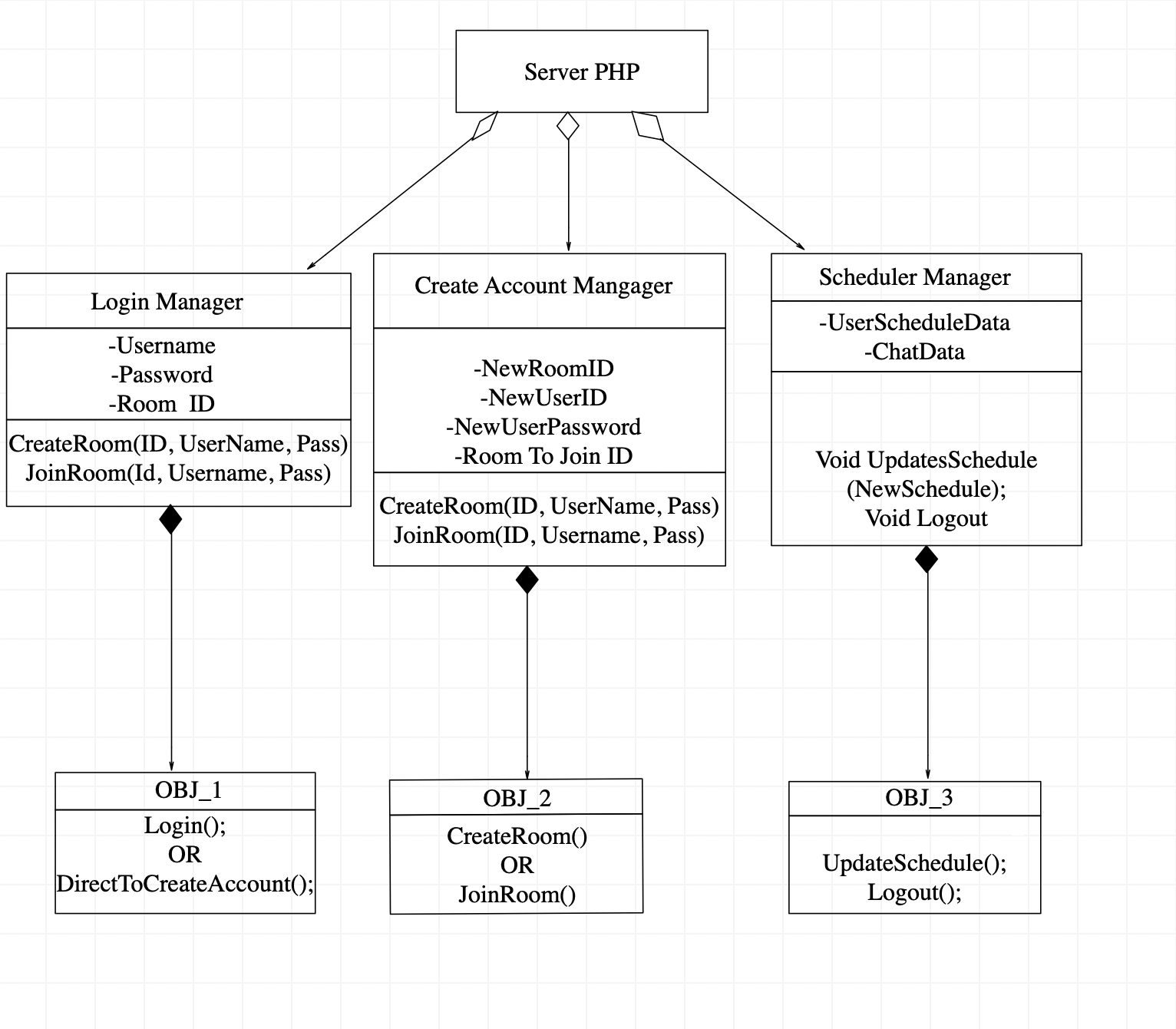


Figure 2 the class diagram that represents different functions used to operate the web-application

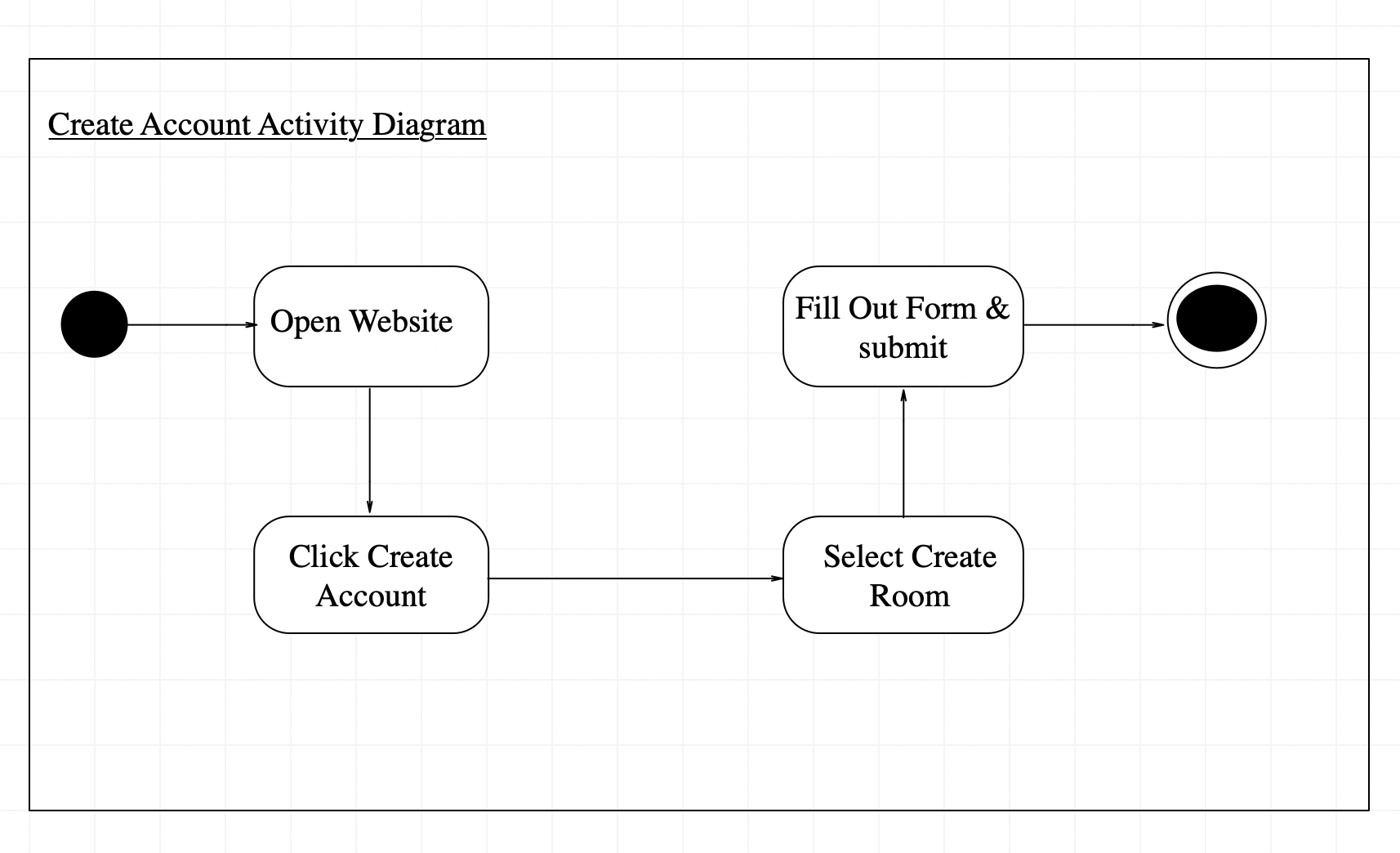


Figure 3 the activity diagram for when a user creates an account

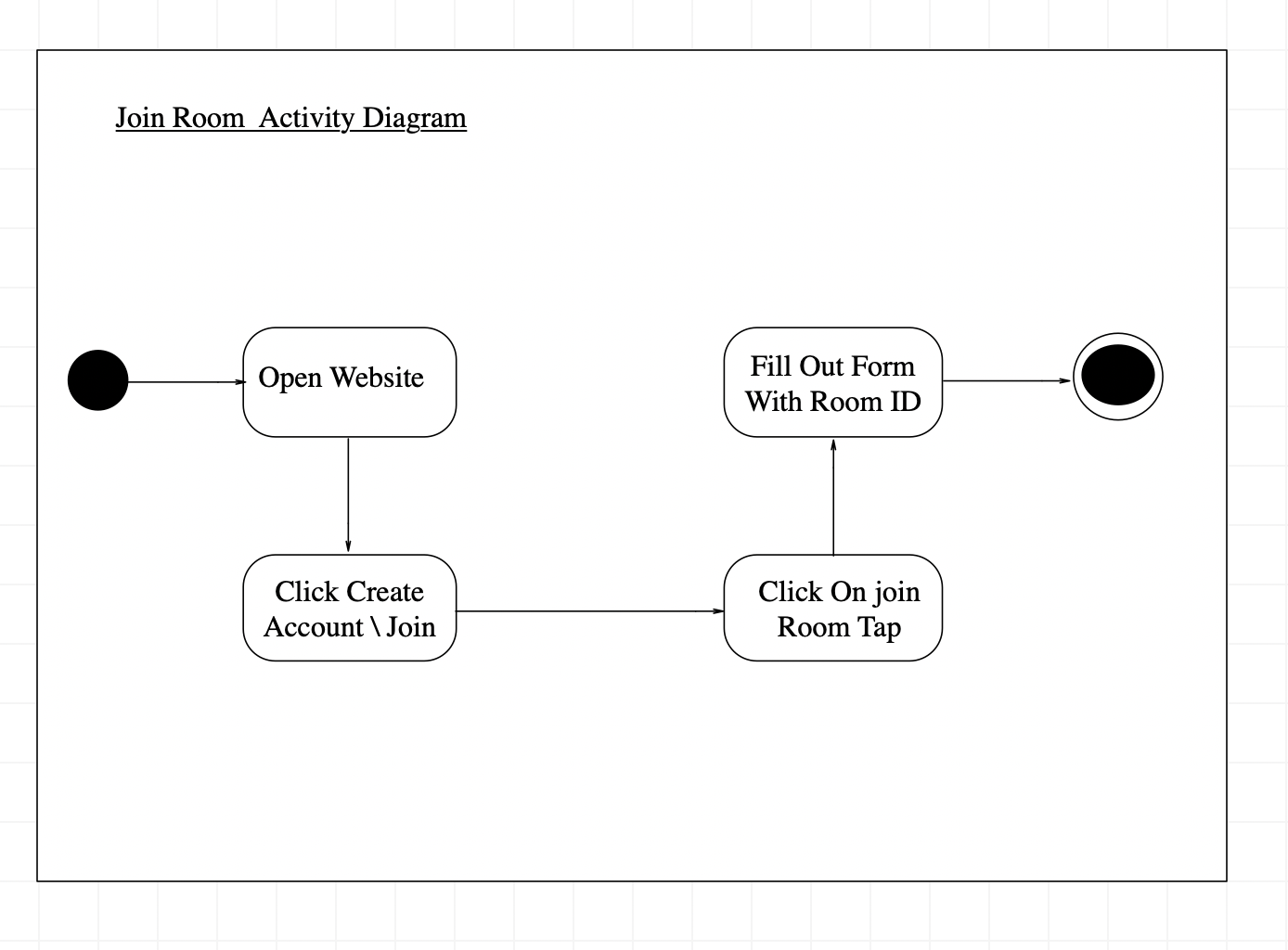


Figure 4 the activity diagram for when a user joins a room

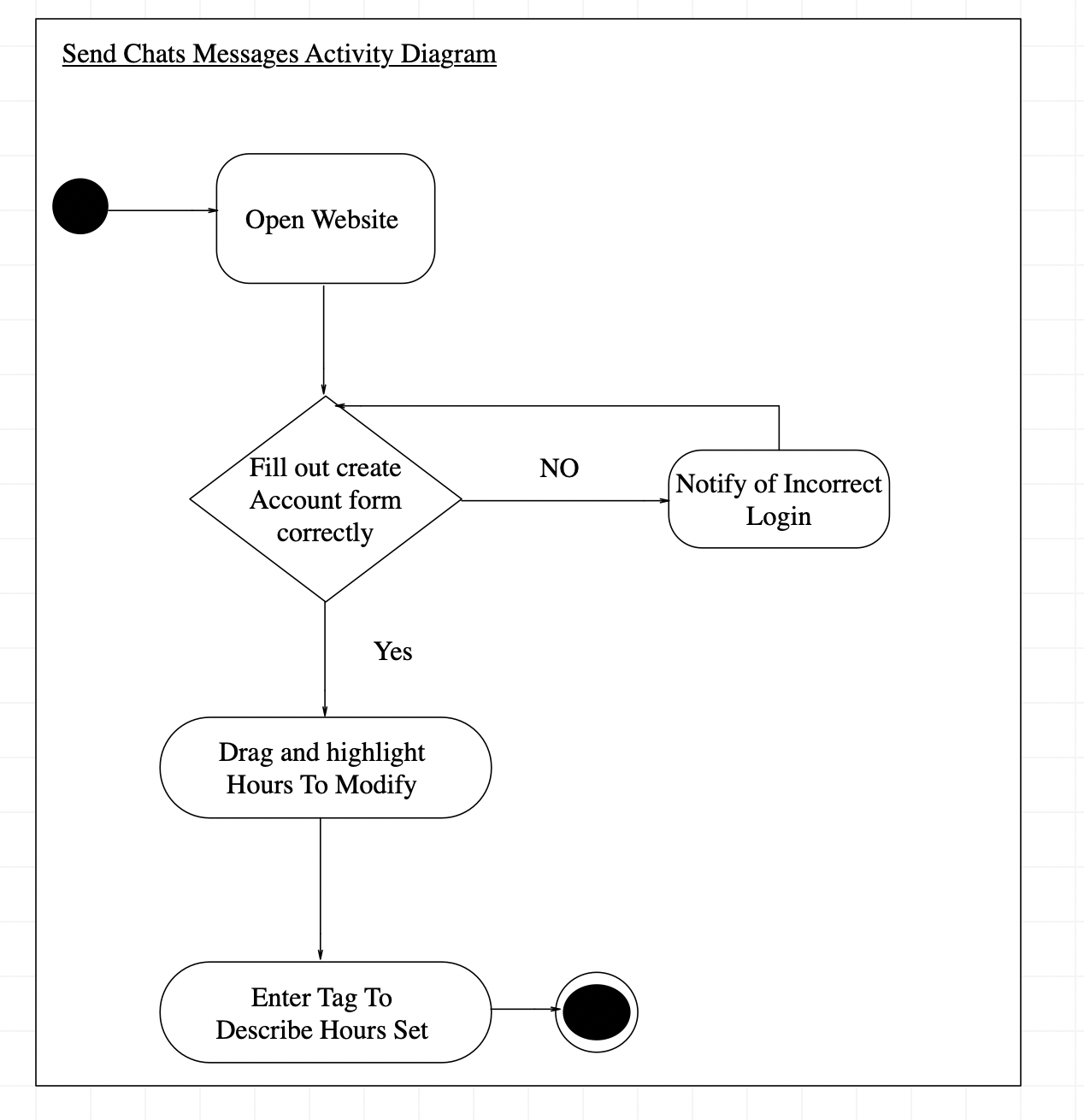


Figure 5 the activity diagram for when a user edits their schedule

## Interface Design

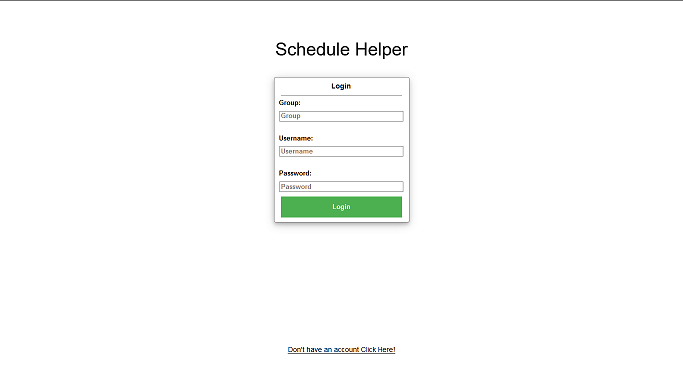
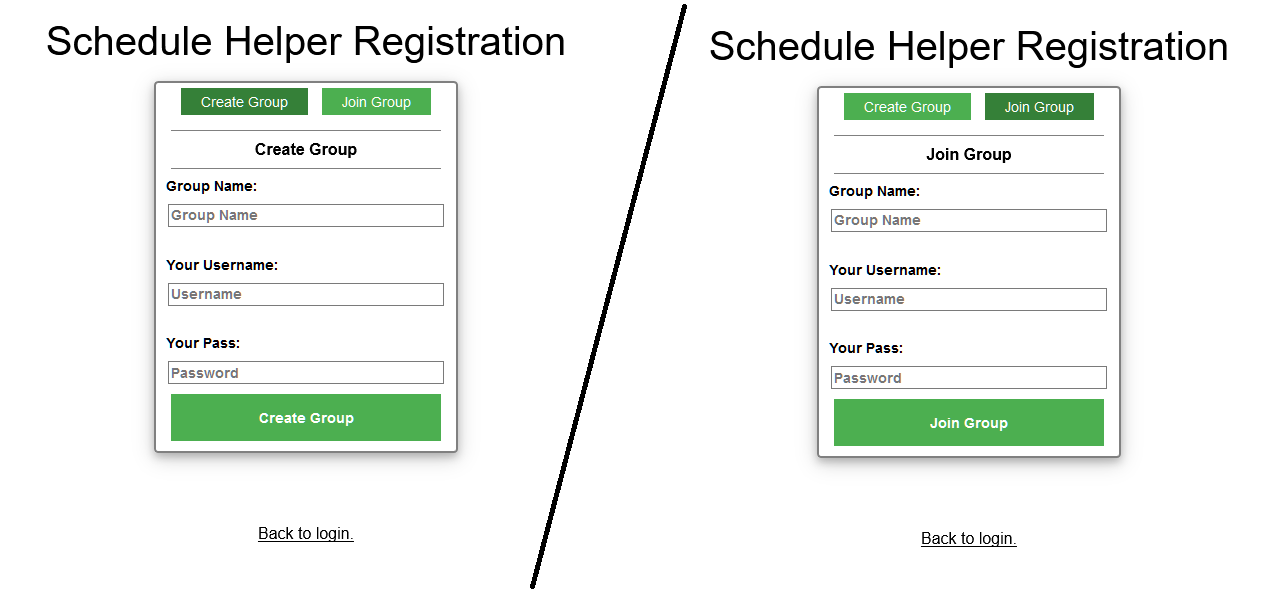
**2.2.1 The Login Interface**

Figure 6 a screenshot of the login page

****

**2.2.2 The Registration Interface**

Figure 7 the two possible views of the registration page, the use can swap between them at any time

**2.2.3 The Schedule Interface**

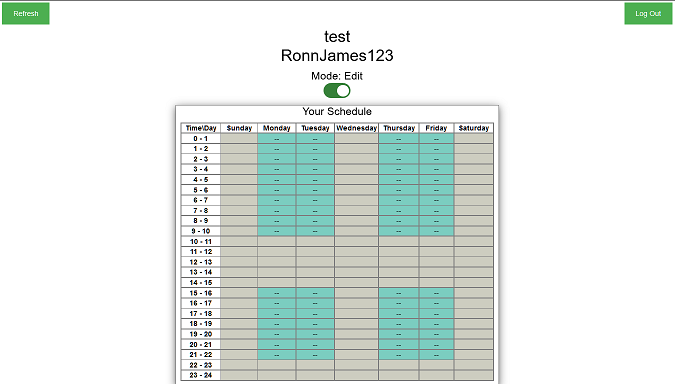


Figure 8 a screenshot of what a user would see when editing their schedule

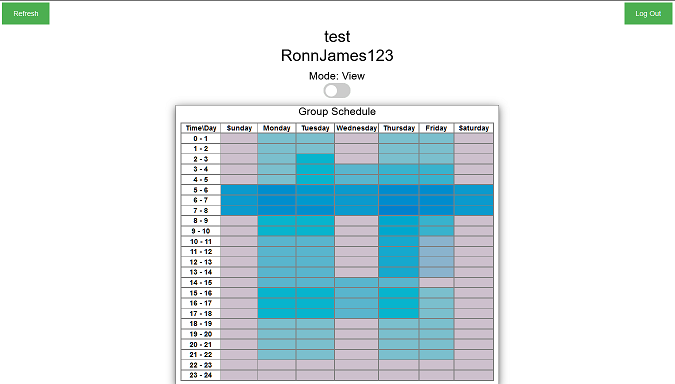


Figure 9 a screenshot of what a user would see when viewing their group’s schedule

# Implementation

## Development Environment

We are using visual studio code for our development environment. The project involves HTML and CSS for the front end, and PHP inserting HTML into the webpage from the back end with AJAX user request calls. We are using MY-SQL for retrieving and storing user data that is used with the webpage.

## Task Distribution

The majority of the project was worked on an implemented as a group but we each took charge of a smaller piece.

**Server/Backend Implementation:** Caleb Hooper

Implemented the SQL database to get and retrieve schedule data then insert into web page dynamically for users to view and modify.

**Website Visuals/Frontend:** Steve Arbuckle & Rawad Bader

Created CSS stylings and form setup for creating new groups, joining groups, and logging into a group.

**Testing:** Whole Group

Experimented with various test cases to stress test the system and discover bugs within the project.

# Testing

## Testing Plan

We have multiple plans to test each of the requirements and make sure they meet the requirements.

1. A test case that made sure the logging phase working and meet the requirements.
2. A test case to check if each group was created right and follow the requirements.
3. Schedule sharing test to confirm that each member of the group was able to see the other member schedule.
4. Sorting test that made sure the data of each group was loaded to the database and kept separated from other groups.

## Tests for Functional Requirements

### Logging in

We tested the login page by creating an account and trying to connect to the server. This test was successful and we were able to see schedules that we added.

We made a test to make sure only valid login information was accepted by the system. We entered invalid login information to the system and it still let us view the team schedule; this error was later fixed.

**4.2.2 Connecting a group**

We tested our group name option that we proved in our web page to create a new group; we were able to add a new group and check their available time and schedules. We tested if the system allowed to create the same group name and that end up overwriting the existing same group name. we had to go back and fix the problem with adding an existing group.

**4.2.3 Schedule Sharing**

We created a test case that made sure all the group members were able to see and update schedules made by any of the member in the group. The webpage would not update the schedule right after each member edited their own schedule; this problem was later fixed

**4.2.5 Storing Schedule Data**

A test case was created to make sure that all the information and data for each user in the group was saved in the database.

A close up of a mans face

Description automatically generated

## Tests for Non-functional Requirements

Updating schedule test was created to make sure that every member were able to see the changes each of the team member made and calculate the delay time also how long it take the new changes to appear on the schedule after multiple try we got it to update the schedule less than 1 second to update the schedule.

We made a test case to see how many groups we can create, and the results were great which the system can allowed us to add more than 20 people in one group.

## Performance Requirements

All database queries and features from the web page in general should respond quicker than the average human reaction time of .17 seconds. The largest delay will likely be from the user’s internet connection, but we will reduce the response time as much as possible by using efficient database queries and minimal, data efficient web design.

## Safety and Security Requirements

We will implement secure data transfer of user data to our database server but the most important thing we will do to protect the user’s data is not ask anything personal of them. In case of a data breach the user will be protected because we will not store any important data on the user. Just their username password they use for our site and basic schedule information.

## Software Quality Attributes

The most important attribute will be ease-of-use. This app needs to help people connect with their friends, so we need to not waste the users time learning a new piece of software or drive users away with unneeded complexity. The software should be kept reasonably minimal so there is little need for maintain.

It will be important for the user to have access to this software on as many devices as possible so it must run within a web-based environment since almost all devices have a web browser of some kind

## Hardware and Software Requirements

SQL server

Apache web server

Web browser

# Analysis

We thought the project an opportunity to learn more about web developments and the process of creating test as we go through the project. The third milestone took more time then the other two milestone stages.

There were multiple meeting for our group that goes over each phase and cover each member role in the devilment process.

# Conclusion

In conclusion this project was helpful for all of us to experience and learn other ways to approach each project requirements and follow the SRS.

Appendix A - Group Log

**Group Meeting One:**

**People:** All Group Members

**Time:** 1 hour

**Date:** 10-9-19

**Description:** We decided on group project and flushed out details of what we wanted project to be as a group. Then assigned chunks of the milestone 1 project to each group member.

**Group Meeting Two:**

**People:** All Group Members

**Time:** 1 hour

**Date:** 10-9-19

**Description:** We got development environments setup on individual group member PC’s

**Group Meeting Three:**

**People:** Caleb, Rawad

**Time:** 45 mins

**Date:** 10-9-19

**Description:** Finalized SRS document.

**Group Meeting Four:**

**People:** All Group Members

**Time:** 1 hour

**Date:** 11-11-19

**Description:** We decided how the interface will look like and what we should include in our web appellation plus what we are using to code each part of the project.

**Group Meeting Five:**

**People:** All Group Members

**Time:** 1 hour

**Date:** 11-13-19

**Description:** We got some of the coding and interface working and we decided to meet to see each member feedback on the process.

**Group Meeting Three:**

**People:** All Group Members

**Time:** 1 hour

**Date:** 12-02-19

**Description:** Finalized same of the requirements and run some testes to demonstrate the web application and how each thing works also we dive