# WeCasa

Project Plan Document

Team HAGS JP

**Team Lead:**

Allison Austin

**Team Members:**

Githel Lynn Suico

Haley Nguyen

Joshua Quibin

Judy Li

Matthew Chung

**Date Submitted: October 5, 2022**

# Project Plan Version Table

| Version | Description | Date |
|---|---|---|
| **1.0** | Initial Project Plan<br>- Budget/Resources<br>- Project Risks<br>- Project Schedule<br>- Staff Organization | 10/5/2022 |

# Table of Contents

# Overview

## Scope

WeCasa will target the California, United States market at launch with future plans to expand nationwide and other English speaking countries. California privacy laws as well as the Web Content Accessibility Guidelines (WCAG, 2022) will be met at initial launch. The browser of choice will be Chrome 104 due to it being the most popular browser.

## Goals

The WeCasa team is focused on building a professional and scalable product while following proper coding and teamwork frameworks. The primary goal is to brainstorm and execute the implementation of useful features for our app. Though it is just as important to make sure the processes of development are as productive as possible. The team is also driven to learn more about the entire software engineering process and how to properly build a full stack application.

# Technological Process

## Project Management Framework

- SCRUM - Agile Methodology

## Tools

- Version Control: Git
- Project Information Website: Github Pages
- Scheduling Software: Google Sheets
- Team Communication: Discord
- Client Communication: Microsoft Outlook
- Task Management : Trello
- SCRUM Meetings: FigJam
- Documentation Storage/Management: Google Drive, GitHub

## Versioning

WeCasa will follow semantic versioning #.#.# (Major/Minor/Patch) with initial releases starting with '0.1.0' and Minor/Patch versioning incrementation based on client feedback. Client approval will involve Major versioning incrementation which signifies official documentation release.

# Deliverables

Deliverables are derived from the senior project requirements and project-specific requirements. At a high level, major deliverables for the project include documentation and a centralized code package.

# Project Risks

The project risks are documented in the risk management table. Each risk will be assigned the following attributes:

- **Threat Level**: describes the impacts of the risk.
    - Low: the impact of the risk is minimal and can be easily managed.
    - Medium: the risk might negatively affect performance, but won't stop progress.
    - High: the risk would result in an un-shippable product.
- **Likelihood of Event**: Corresponds with how likely the risk is.
- **Tolerance Level**: Quantifiable metric (hours) specifying the amount of buffer allocated to the risk.
- **Mitigation Plan**: Details plans to prevent the risk from happening.
- **Management Plan:** Explains how risk will be reduced if it occurs.

High Priority

| Risk | Threat Level | Likelihood of Event | Tolerance Level | Mitigation Plan | Management Plan |
|---|---|---|---|---|---|
| Hardware equipment failure | **High** | **Unlikely** | **10-12** | Pay attention to hardware maintenance. | In case of hardware failure, we will borrow CSULB laptops through the CSUCCESS program. |
| Technology stack failure | **High** | **Unlikely** | **10-12** | Use well-tested components. | In case of technology infrastructure failures, we will research and switch to a more optimal technology. |

| | | | | | |
|---|---|---|---|---|---|
| Dependency Failure | **High** | **Unlikely** | **10-12** | Avoid dependency failures by making sure libraries are compatible with one another. Also, avoid this failure by feasibly building an abstraction around the library for easier replacement. | In case a dependency fails, we will remove the incompatible library and utilize the abstraction we built in order to replace it with a more efficient package. |
| Changes in user and functional requirements | **High** | **Likely** | **10-12** | Build components with control abstraction that allows easy replacement of new user and functional requirements. | In case laws, regulations, user and functional requirements change, the abstraction we built allows easy replacement for the new requirements. |
| Late deliverables | **High** | **Unlikely** | **10-12** | Follow the project timeline with recurring meetings, and milestones | In case of any late deliverables, the team will reassess deadlines, and make a detailed plan to reach the end goal. |
| Losing team members | **High** | **Likely** | **10-12** | Leaving room (28 hours of capacity) at the end of the project schedule. | In the case that we have less team members than at the start of the project, we will have less tasks in the last few sprints to allow for planned tasks to be punted. |

Medium Priority

| **Risk** | **Threat Level** | **Likelihood of Event** | **Tolerance Level** | **Mitigation Plan** | **Management Plan** |
|---|---|---|---|---|---|
| Going over budget | **Medium** | **Unlikely** | **5-8** | Ensure that all software being used is under a free license. | Keep a tab of expenses and divide up costs between each team member at the end of the project. |
| Changes to client schedule/deadlines | **Medium** | **Unlikely** | **5-8** | Leave room in the project schedule for deadlines to be moved up. | Re-evaluate work item priorities to adhere to feature deadline changes<br><br>Perform backlog grooming |

Low Priority

| Risk | Threat Level | Likelihood of Event | Tolerance Level | Mitigation Plan | Management Plan |
|---|---|---|---|---|---|
| Requirements Creep | **Low** | **Unlikely** | **>4** | Conduct stand-ups 2x per week to monitor progress.<br>Scope is also initially defined in project plan, reviewed monthly by team to prevent undetected scope creep. | In case there is an undetected scope creep, we can re-evaluate our initial scope, and reduce or expand if necessary. |

# Resources

## Team

While a complete team would contain all the following staff members, HAGS JP has six members. Each team member will be performing multiple jobs.

- **Team Leader/ Lead Engineer**
- **Scrum Master**
- **Web Developer**
- **Backend Developer**
- **Data Engineer**
- **Beta Tester**
- **UI/UX Designer**

## Resources

The development team will use software packages and solutions that are free and open source. This includes the front-end/back-end framework, data store, development environment, web server, and testing framework. Computers and other devices that will be used to develop and test code, document the project, and connect with team members will be the personal computers already owned by each team member.

### Hardware

Minimum specification recommendation for personal computers:

CPU: Intel Core i5 Processor

Display Resolution: 1920 x 1080
RAM: 8GB
Storage: 128GB+
Operating System: Windows/macOS

### Required Software

Front-end Framework: React
Back-end Framework: .NET 6.x
Web Server: Nginx 1.22+
Data Store: Maria DB 10.x
IDE: Visual Studio Code 2022 Community Edition (1.7x Windows/Mac)
Interface Design Tool: Figma 107.0

## Budget

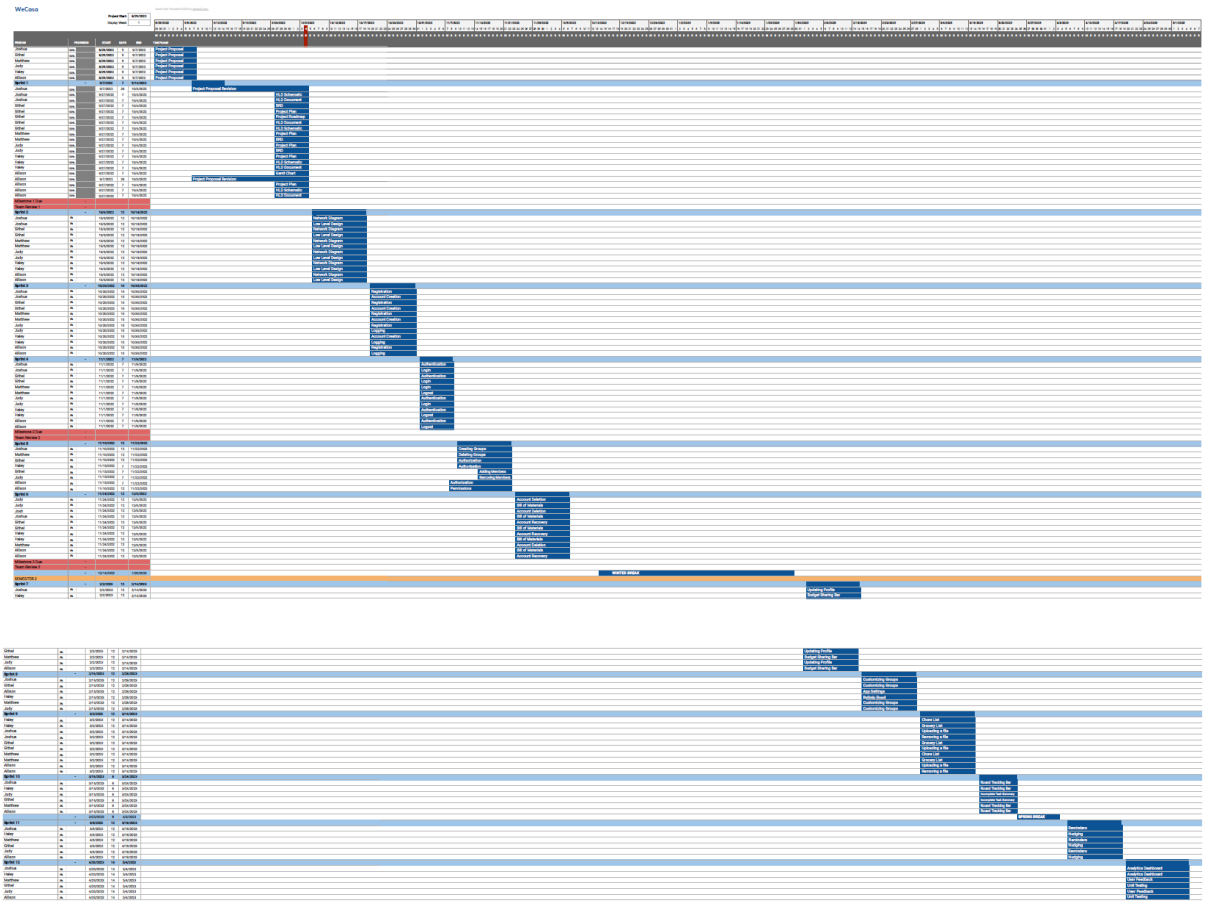
| | |
|---|---|
| Resources<br>- Software (open-source)<br>- Computers (personal items) | $0 |
| Labor (Average Salary in Long Beach, California from Ziprecruiter.com)<br>While each team member will perform the tasks of multiple roles, their overall salary will be only the higher paying role they perform.<br><br>- Team Lead/Lead Engineer: $128,351<br>- Scrum Master: $123,743<br>- Web Developer: $81,120<br>- Backend Developer: $82,120<br>- UI/UX Engineer: $98,196 | ($128,351 x 1) +<br>($123,743 x 1) +<br>($81,120 x 1) +<br>($82,120 x 2) +<br>($98,196 x 1) = |
| **Total** | **$595,650** |

# Timeline

The WeCasa application will be completed by May 5, 2023.

# Schedule

The project schedule coordinates with the CSULB fall and spring semester schedules, and includes fall, winter, and spring breaks. We used a Gantt Chart to map out the dependencies and work item assignments for each sprint, as well as document crucial due dates and academic breaks. Due to the high level of detail in the schedule, it might not be readable in this document. For more detail, this schedule can be viewed in the WeCasa code repository found on GitHub.



# Work Breakdown

Work items can be broken down into smaller tasks based on the business requirements for a particular feature. The Business Requirements Document can be viewed in the WeCasa GitHub repository.

# Work Item POCs

## Documentation

| Work item | Team member |
|---|---|
| Project Proposal | Allison |
| Business Requirement Documentation | Joshua |
| High Level Design | Judy |
| Project Plan | Haley |
| Project Roadmap | Githel |
| Network Diagram | Haley |
| Low Level Design | Githel |
| Bill of Materials | Matthew |

## Core Components

| Work item | Team member |
|---|---|
| Registration | Githel |
| Logging | Allison |
| Authentication | Judy |
| Authorization | Allison |
| User Management | Joshua |

## Features

| Work item | Team member |
|---|---|
| App Settings | Joshua |

| | |
|---|---|
| Budget Sharing Bar | Judy |
| Bulletin Board | Judy |
| Calendar | Githel |
| Dashboard | Joshua |
| Group Lists | Githel |
| Incomplete Task Summary | Haley |
| Reminders | Haley |
| Round Tracking Bar | Matthew |
| Nudging | Matthew |
| Photo/Documentation Upload | Allison |
| User Feedback | Joshua |

## Schedule Changes

All schedule changes and updates will be made within the Gantt Chart.

# Team Organization

HAGS JP uses a “Hybrid” Agile Software Development Team Structure which consists of a combination of generalists and specialists.

## Team Structure

### Role Definitions:

*Allison Austin*

**Team Lead/Lead Engineer:** Allison is responsible for guiding, monitoring and leading HAGS JP.

**Data Engineer:** Allison is responsible for setting up the data pipeline, preparing data for downstream dependencies, and developing the Data Access Layer for feature implementation.

*Githel Lynn Suico*

**Web Developer:** Githel is a part of the interface team and will be coding a portion of the UI.
**QA Tester:** Githel is partially responsible for trying out unreleased features, reporting bugs/errors, and providing feedback.

*Haley Nguyen*

**Web Developer:** Haley is a part of the interface team and will be coding a portion of the UI.
**UI/UX Designer:** Haley is responsible for WeCasa brand design and UI/UX design.
**QA Tester:** Haley is partially responsible for trying out unreleased features, reporting bugs/errors, and providing feedback.

*Joshua Quibin*

**Scrum Master:** Joshua is responsible for ensuring HAGS JP uses SCRUM methodologies.
**Web Developer:** Joshua is a part of the interface team and will be coding a portion of the UI
**QA Tester:** Joshua is partially responsible for trying out unreleased features, reporting bugs/errors, and providing feedback.

*Judy Li*

**Backend Engineer:** Judy is partially responsible creating for server-side web application logic and integration of the web developers' work
**QA Tester:** Judy is partially responsible for trying out unreleased features, reporting bugs/errors, and providing feedback.

*Matthew Chung*

**Backend Engineer:** Matthew is partially responsible creating for server-side web application logic and integration of the web developers' work
**QA Tester:** Matthew is partially responsible for trying out unreleased features, reporting bugs/errors, and providing feedback.

# Team Reporting and Communication

## Mechanisms for Progress Reporting

Progress reporting to the client is communicated through email or in-person team reviews. All files sent to team members are done via email or Discord. These communications are done informally, unless special documentation of the progress is required.

### Mechanisms for Intra Team Communication

HAGS JP conducts bi-weekly in-person (twice a week) stand-up meetings to update other team members on their progress. These meetings also act as a medium for the team to ask questions and remove blockers that may not be communicated electronically. In the case a team member cannot make the meeting in-person, the meeting can be moved to Discord. All other communication is done electronically through Discord or email.

# Glossary

| **Term** | **Definition** |
|---|---|
| Beta tester | User that tests a product in a production environment to uncover bugs/issues before it gets released |
| IDE | Integrated Development Environment |
| POC | Point-Of-Contact |
| QA | Quality Assurance |
| UI/UX | User Interface/User Experience |

# References

*Entry Level*. (2022). ZipRecruiter; ZipRecruiter.

https://www.ziprecruiter.com/Salaries/Entry-Level-Salary-in-Long-Beach,CA

*Simple Gantt Chart*. (2022). Vertex42.com.

https://www.vertex42.com/ExcelTemplates/simple-gantt-chart.html

What Agile Software Development Team Structure Looks Like. (2022).

Relevant.software

https://relevant.software/blog/what-agile-software-development-team-structure-looks-like/