

# Stegord: A better way to send messages project plan

Avi Lance, Isaac Perkins, Jaxon Simmons, Aleks Stevens

<https://github.com/Avi-Lance/Painted-Penguins-Stegord>

<b>1. Project Plan Revision History</b>	<b>1</b>
<b>2. Stegord Application</b>	<b>1</b>
<b>3. Management Plan</b>	<b>2</b>
<b>4. Project Timeline</b>	<b>5</b>
<b>5. Building Plan</b>	<b>6</b>
<b>6. Monitoring and Reporting</b>	<b>6</b>
<b>7. Rationale</b>	<b>7</b>

## 1. Project Plan Revision History

This lists every modification to the document. Entries are ordered chronologically.

Date	Author	Description
5/19/2023	Isaac P.	Added application overview, management plan, building plan, and rationale.
5/19/2023	Jaxon S.	Added project timeline and monitoring and reporting

## 2. Stegord Application

The need for secure messaging is ever so important in a digital world. The downside of messaging is security. We need the ability to securely talk to our friends and family without the fear of having our conversations be intercepted. The solution is Stegord. Stegord allows users to simply send messages to one another with the peace of mind knowing their messages are being guarded with steganography.

### 3. Management Plan

Our team organized the project into two components: the client side and the server side. We divided our team into different areas of development. One member develops the steganography module while another works on the database module and then there are the members connecting and developing the front end. Our architecture is as follows:

- Client side (This is what the user sees)
- Log-In/Sign-Up
  - This allows the user to login or create an account. This is the first page any user will interact with.
- Profile
  - This is where the user can personalize their account with a bio and profile picture. This page is what differentiates users from each other.
- Dashboard
  - This is the navigation structure of the main application. A user will see this as soon as they log in and it will allow them to select between the different uses of the application such as messaging and friending.
- Finding/Pending Friends
  - Users will be able to use this section to search and find other Stegord users as well as accept other users to be their friend.
- Chat
  - Here users can start or continue an existing chat. This is the main feature of the application as it allows users to communicate through the Stegord application.

On the client side we designated Isaac Perkins and Jaxon Simmons and we had additional help from Avi Lance when needed. Aleks Stevens and Avi Lance managed and developed the server side.

Our team's primary focus is to maximize organization and management efficiency by adopting Scrum Agile methodologies. Our approach involves weekly sprints using Jira, allowing us to swiftly progress through tasks and implement new features. Additionally, we will schedule regular Discord or in-person meetings during the week to address any uncertainties, seek clarification from team members, assess our progress, and determine if any course corrections or adjustments are necessary.

For document and code organization we created a google drive and github repo as shown below:

## Shared with me > CS422 Project 2 ▾ 👤

File type ▾


People ▾


Last modified ▾

Name ↑

 3-Page Proposal 👤

 Project\_Plan 👤

 SDS 👤

 SRS 👤

main ▾ 1 branch 0 tags

















Go to file

Add file ▾

<> Code ▾

 iperkins987 Added UI documents

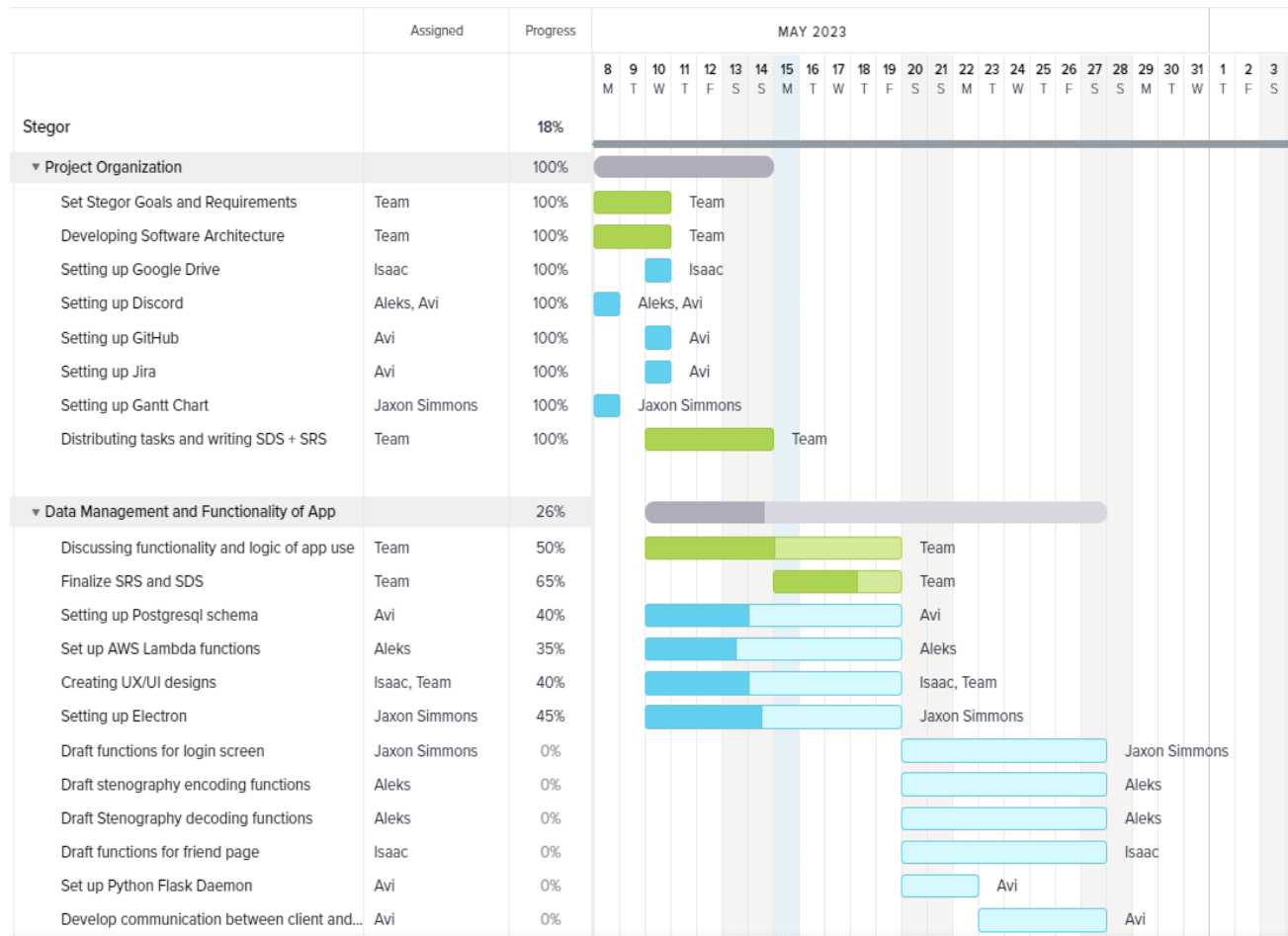
1a1ea06 yesterday 8 commits

	.erb	Added template webpack configs among other scripts	2 days ago
	.vscode	Added AWS Amplify	2 days ago
	assets	Initial commit configuring Electron, React, and Webpack. Also creat...	2 days ago
	documents	Added UI documents	yesterday
	release/app	Initial commit configuring Electron, React, and Webpack. Also creat...	2 days ago
	src	Initial commit configuring Electron, React, and Webpack. Also creat...	2 days ago
	.editorconfig	Initial commit configuring Electron, React, and Webpack. Also creat...	2 days ago
	.eslintignore	Initial commit configuring Electron, React, and Webpack. Also creat...	2 days ago
	.eslintrc.js	Initial commit configuring Electron, React, and Webpack. Also creat...	2 days ago
	.gitattributes	Initial commit configuring Electron, React, and Webpack. Also creat...	2 days ago
	.gitignore	Added template webpack configs among other scripts	2 days ago
	LICENSE	Initial commit configuring Electron, React, and Webpack. Also creat...	2 days ago
	README.md	Wrote initial readme	2 days ago
	package-lock.json	Added AWS Amplify	2 days ago
	package.json	Added AWS Amplify	2 days ago
	tsconfig.json	Initial commit configuring Electron, React, and Webpack. Also creat...	2 days ago

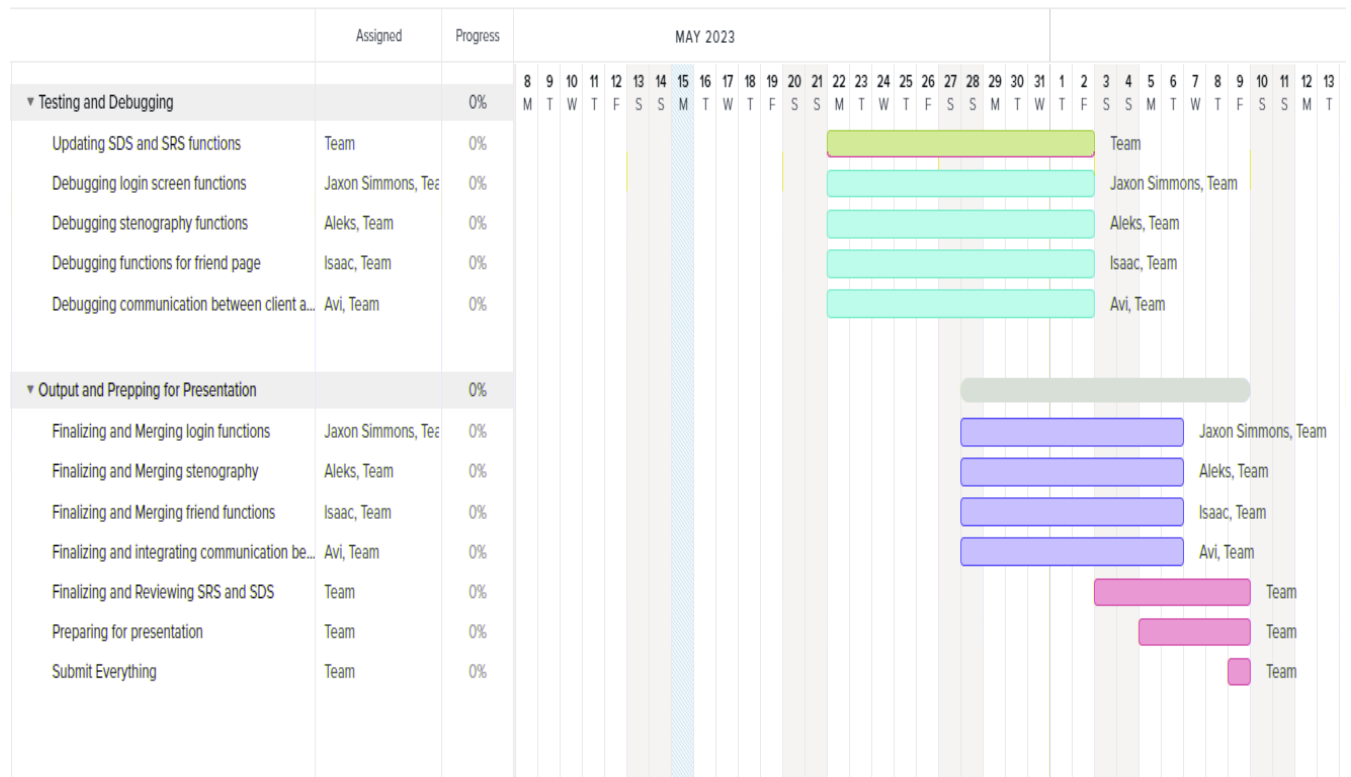
## 4. Project Timeline

In order to stay accountable and make consistent progress, tasks were divided based on the modules we were given with due dates on each task. Our project timeline is shown below:

### Phase 1:



### Phase 2



## 5. Building Plan

Team members were assigned sections to work on. Sections were broad such as frontend and backend, but it was enough for project organization. Our deadlines strictly follow the deadlines of the client (Project 2 Schedule). As for planning, our first stage was deciding on a project and picking which libraries and languages to work with. We then discussed the scope of our application as well as additional features. Each group member has their own specialty such as frontend design or steganography so we divided roles based on amount of expertise. As for actual design of the modules we let the group members design those with the knowledge of what we expect the application to do.

## 6. Monitoring and Reporting

We will monitor our work through active communication through Discord, and viewing the project timeline often to keep everyone accountable. In the above timeline, the green bars represent tasks that the whole team is responsible for. So for every one of those tasks, every team member will contribute either by creating a rough draft, finalizing what needs to be done for the task, or reviewing the task to make sure that they are fully complete.

## 7. Rationale

We decided to break our system into the sections to encourage swift and concise development. Our Discord channel, shared Google Drive, and Github repo allow us to efficiently manage and develop this project.