

Jot-Down

Software Plan

Software Project Management (EGR427), Fall 2021
December 8, 2021
Dr. Daniel Grissom

Team Leader: Tomas Perez

Draft Leader: David Barsamian

Presentation Leader: Kaleb Coggins

Team Member 4: Daniel Furdui

Team Member 5: Isaiah Swanson

Table of Contents

PRODUCT PROPOSAL	5
1 Problem Description	6
2 Proposed Solution	6
3 Expected Business Value	6
3.1 Monetization Strategies & Rationale	6
3.2 Growth Opportunity: Sustainability, Scalability & Investibility	7
3.3 Total Addressable Market (TAM)	7
4 Alternate Solutions	8
5 Personal Investment	8
6 Qualifications of Team	9
PRODUCT REQUIREMENTS	10
1 Purpose and Scope	11
1.1 Business Purpose & Scope	11
1.2 Tech Purpose & Scope	11
2 Stakeholder Identification & Regulatory Requirements	12
3 Persona (User Role) Identification	13
4 User Stories & Requirements	13
4.1 Functional User Stories	13
4.2 Technical & Usability Requirements	14
4.3 Performance Requirements	14
5 High Level Workflow	15
5.1 Milestones	15
5.1.1 Minimum Viable Product	15
5.1.2 Full Release 1.0	15
5.1.3 Full Release 2.0	15
5.2 Timeline	16
6 Acceptance Criteria	17
7 Rapid Prototype	18
FUNCTIONAL SPECIFICATIONS	21

1 Platform Overview	22
1.1 Platform Requirements	22
1.2 Platform Rationale	22
2 Architecture & Organization	23
2.1 High-Level Architecture	23
3 Application Screen Flowchart	24
4 Wireframe Layouts	25
Screen 1: Landing Page	25
Screen 2: Login Page	26
Screen 3: Purchase Page	27
Screen 4: User Dashboard	28
Screen 5: Microphone Setup	29
Screen 6: Assign Editors Page	30
Screen 7: Meeting Started Page	31
Screen 8: Meeting Paused Page	32
Screen 9: Mark Section Page	33
Screen 10: Icon Editor Page	34
Screen 11: Stop Recording Page	35
Screen 12: Edit Transcript	36
Screen 13: Send Notes	37
Screen 14: Edit Email for Attendees	38
Screen 15: Contact Sales	39
Screen 16: Admin Page	40
Screen 17: Manage Users	41
Screen 18: Transcripts	42
Screen 19: Permissions Page	43
5 User Feedback	44
 SOFTWARE PROJECT MANAGEMENT PLAN	 45
1 Risk Analysis	46
1.1 Risk Analysis Matrix	47
1.2 Risk Management Strategies	47
1.2.1 Risk Response Strategy for R1 (Lack of Interest)	47
1.2.2 Risk Response Strategy for R2 (React Update)	48
2 Deliverables, Components & Actions	48
3 Teams and Organization	53
3.1 Team Roles & Organization	54
3.2 Supporting Tools	56
4 Schedule	56
4.1 High-Level Gantt Chart	57

4.2 Sprint Cycle Gantt Charts	57
4.3 Gantt Chart Source	60
5 Kanban Board	60
6 Cost	61
7 Testing	62
8 Maintenance	63
9 Source Code Management & Git Repository	63
 INDUSTRY TRENDS	 65
1 Software Development Process	66
1.1 Industry Summary	66
1.2 Application to Project	66
2 Feature Selection	66
2.1 Industry Summary	66
2.2 Application to Project	66
3 Team Roles and Organizational Structure	66
3.1 Industry Summary	66
3.2 Application to Project	67
4 Testing & Maintenance	67
4.1 Industry Summary	67
4.2 Application to Project	67
5 Communication	67
5.1 Industry Summary	67
5.2 Application to Project	68

Jot-Down

Product Proposal

1 Problem Description

During company meetings, there can be numerous significant items being discussed, ranging from planning the next quarter, business forecasting, or perhaps even a presentation to or from a significant client. There are many important conversations or points that are being made during these meetings that can be difficult for employees to track or take notes on to reference later. Tools such as WebEx have a feature that allows users to create a transcript of the meeting that had taken place; however, the transcript lacks a layout or organization that optimizes the productivity of the user. The user will spend more time trying to track conversations through the transcript in a cohesive manner, rather than being able to quickly reference a point in the meeting that they need, thus reducing productivity and wasting money.

The same issue arises in universities during their lectures with students attempting to both understand what is being presented, as well as trying to keep up in note taking for any reference at another time. This problem is only amplified with students that have disabilities that would hinder their abilities to take notes in lectures. Again, using WebEx, it can be difficult for the student to filter out the information that they need because of other students talking in class being picked up in the transcript, narrowing down what discussion references which slide, and so on. This would take time away from the student studying and understanding the lecture, thus hindering the student's ability to be successful in the class overall.

2 Proposed Solution

Meeting Notes Helper will give presenters the ability to craft detailed outlines of their content in real-time as they are presenting. One common pain point for audiences is that they often must split their attention between the presenter's presentation and their visual content. Our tool will let the audience fully focus on the presenters' content by letting the presenter create rich outlines of their content based on an advanced artificial intelligence-generated transcript through verbal commands or a simple user interface. For example, when presenting on a specific topic, the presenter can create a new section specifically for that content by saying a voice command or pressing a button. Meeting Notes Helper will then categorize all the content that follows as belonging to that section. This removes the tedious work of editing a detailed notes document for the presenters' audience post-presentation. For larger presentations and lectures, presenters can also assign any number of editors to do the work for them, letting the content of the presentation be the focus.

3 Expected Business Value

In the next three sections, we will be addressing how we intend to monetize and grow our product, as well as our addressable market.

3.1 Monetization Strategies & Rationale

This note taking application will (1) offer a paid-monthly package, (2) offer an annual renewal membership, (3) and have a heavy social media presence.

1. The note taking application will allow customers the option to purchase a month-to-month subscription for a limited number of users within the customers business, as a sort of trial in order to encourage customers to purchase the annual membership.
2. The note taking application will give clients the ability to purchase an annual membership that will allow the clients to manage an admin version of the site. From this admin site, the clients will be able

to give access to any number of users, remove access, and have access to a database containing the saved lectures along with their transcripts.

3. Social media is a huge part of everybody's everyday lives. Therefore, the note-taking application will create accounts across various social media platforms to better advertise itself.

3.2 Growth Opportunity: Sustainability, Scalability & Investibility

1. **Sustainability:** Jot-Down offers a variety of licenses that must be renewed annually in order to continue using the product. Businesses and schools will learn about the versatility and flexibility provided by our application, which will prompt them to purchase a license to try out the Jot-Down app. Then, through both word of mouth and marketing campaigns, new clients can be secured which will- in turn- drive up the revenue produced by our software.
2. **Scalability:** This product is sold as licenses which are valid for each user within an organization. This means that any organization can buy as many licenses as they feel necessary and means that this solution is a viable option for organizations of any size. In addition, the application will be flexible by allowing the presenter (whoever is controlling Meeting Notes Helper) to control where the notes are piped, and this can include as many recipients as needed, within reason.
3. **Investibility:** If presenters and speakers want the content for their meetings/classes to be well understood and fully comprehended, the solution to their problem lies with the Jot-Down. Jot-Down is a piece of software which is specifically tailored to providing a condensed, organized transcript of a lecture/meeting in which pertinent information can be recalled immediately. The benefits provided by Jot-Down will no doubt entice investors and clients from all walks of life.

3.3 Total Addressable Market (TAM)

Jot-Down is meant for companies with greater than 100 employees, and universities with over 1,000 students, which would be most likely to allocate funds to purchase the product. This equates to 170,600 businesses¹ and around 2,000 universities², totaling 172,600 potential organizations. Pricing for Jot-Down will be \$150 per year per license. If the average company purchases 80 licenses, the total addressable market comes to \$2.1 billion annually.

4 Alternate Solutions

Several solutions currently exist, however none of them offer the flexibility or power that our product offers. Popular virtual meeting platform Cisco Webex³ offers an AI-supported assistant service that provides some advanced transcription services; however, the service lacks the editorial control that our product offers. In-person services, such as the services provided by GMR⁴ with hand-written transcripts can be accurate, however these services are costly compared to our offering.

FEATURES	CISCO WEBEX ASSISTANT VOICE INTELLIGENCE	GMR IN-PERSON TRANSCRIPTION	OURS
AUTOMATED CAPTIONING	✓	⚠ <i>Hand-written transcript</i>	✓
POST-MEETING EDITING	✗	⚠ <i>Edited by hand</i>	✓
RECORDING CONTROL PANEL	✗	✗	✓
OUTLINED MEETING NOTES	✗	⚠ <i>Edited by hand</i>	✓
HIGHLIGHTED NOTES	✓	⚠ <i>Edited by hand</i>	✓
TRANSLATION	✓	✗	✓
ACCESSIBILITY	✓	✗	✓

Table 1: Comparison between our product and our competitors

5 Personal Investment

Our team is determined in getting Jot-Down built because each of us have personally experienced a time in our lives where we weren't able to pay attention during lectures or take efficient notes.

As students, we empathize with the struggles our peers face. Taking notes during a lecture can result in some information slipping through the cracks or being lost in translation. This problem can result in disappointing performance on important assignments such as exams and quizzes, especially if the lecture material is not available outside of class. That is where Jot-Down would save you from having to go

through the headache of frantically scrambling to find information last-minute. A colleague of ours mentioned:

“I was in an upper division physics class where the professor was moving quickly from topic to topic, with little breathing room in between. It is impossible to take efficient notes in such conditions. As a result of these adverse conditions, I generally struggled when it came time for an exam. Rarely did I ever score higher than a 75% on these exams, and the cycle of frustration never seemed to end for me.”

After experience and hearing stories like the one above, we’re even more driven to create Jot-Down. We believe it would ease the process of note taking by having a well-documented transcript sent to the audience, which would give the audience the capability to fully focus on the presentation.

6 Qualifications of Team



QUALIFICATIONS	OUR TEAM
FORMAL EDUCATION IN SOFTWARE DEVELOPMENT	
MOBILE DEVELOPMENT EXPERIENCE	
DIVERSE EXPERIENCE WITH PROGRAMMING TECHNOLOGIES	
STRONG LEADERSHIP AND COLLABORATION	

Table 2: Team qualifications.

Jot-Down

Product Requirements

1 Purpose and Scope

1.1 Business Purpose & Scope

All businesses that regularly host meetings or presentations will benefit from our product because our tool increases the productivity of its employees by giving them a thoroughly organized transcript of the presentation. As a result, companies will spend less time and money getting their teams on the same page, bringing value to the company. Universities will see a rise in overall student effective learning and grade averages because our note taking tool will give students more time to focus on actual content presented in class, thus creating better educated and prepared students. Our own business will see greater financial savings due to the use of our software in meetings.

1.2 Tech Purpose & Scope

Jot-Down will allow presenters to craft detailed outlines of their content in real-time as they present. The presenter, and any number of editors that they assign, can use an AI-assisted automatically generated transcript and several editing tools to create highly organized outlines of their content. The presenter can optionally use voice commands to organize their outlines hands-free. This will give the presenter total control over the notation of their content and will improve their audiences' retention and comprehension of it.

With these powerful real-time editing tools, Jot-Down will stand out compared to our competitors. Other apps that offer automatically generated transcripts don't give their users control over the content of the transcript, resulting in massive walls-of-text that are difficult to read and edit.

Jot-Down also gives the audience the tools to give feedback and share important information from the presentation. Viewers can add their own markup to documents they have access to, letting them provide input as to what they found valuable. They can also share these markups publicly on the document, and others can give feedback for these markups through a simple rating system. This lets the best content from the presentation rise to the top, making it more visible and accessible.

2 Stakeholder Identification & Regulatory Requirements

The following table describes classes of people who might hold stakes in the success of Jot-Down as a product, and how Jot-Down will benefit them in return.

STAKEHOLDER (SH)	WHY SH MATTERS TO PRODUCT	WHAT SH HAS TO GAIN FROM PRODUCT
INVESTORS	Investors will act as a kickstart for our product and provide funding for the product.	Investors would receive a share of our profits, proportional to their initial contribution.
UNIVERSITIES	Universities pay for licenses to give to either students or teachers or whoever uses the product.	This product will enable better learning and higher success rates for their students.
BUSINESSES	Businesses pay for licenses to give to supervisors and any other employee that might need to use the product.	This product would allow better documentation from company meetings, which could be archived and viewed later.
EMPLOYEES, TEACHERS	Employees and Teachers are the primary users of our product.	Employees and Teachers will no longer have to worry about whether their audience is retaining the information they present.
STUDENTS	Students with disabilities, and students in general, might find it difficult to take notes effectively during a lecture.	This product will allow students to be able to more easily keep track of pertinent information without needing to divide their attention.

Table 3: Table identifying the key stakeholders to this product, why the stakeholder matters to the product's success and why the product's success matters to the stakeholder. Stakeholders are listed in decreasing order of importance, from top to bottom.

3 Persona (User Role) Identification

The table below describes classes of people who will use Jot-Down and defines why each class is unique.

PERSONA	ROLE & WHY NEEDED	UNIQUE FUNCTIONALITY
ADMIN	An administrative user for the product. Needed as license holders will need a way to manage their license, end users, and organization.	View or Manage currently active licenses. Manage archive of transcripts which correspond to each presentation.
PRESENTER	The primary user for the product. Needed as they are in control of how the program organizes the notes.	Categorize, outline, organize, and markup transcript; Strike words/statements from the transcript; Controls start/stop of transcription; Delegate editor/viewer access to transcript.
EDITOR	Secondary user for the product, editing the presenters notes live. Needed to allow the program to scale for larger audiences/presentations.	Categorize, outline, organize, and markup transcript; Strike words/statements from the transcript;
VIEWER	Tertiary user for the product, has read-only access to transcript and notes; Needed to allow audience of presentation to view notes.	Read-only view of published transcript/notes; Add comments to document; Publicly markup document with highlights, bookmarks, or labels; Rate other viewers' markups based on usefulness

Table 4: Table identifying the personas/users for this product.

4 User Stories & Requirements

The upcoming section discusses critical user stories that are needed to create a functional and satisfying application for all potential users.

4.1 Functional User Stories

1. As an Admin, I would like to manage all users, so that I can give the appropriate access to those that need it.
2. As an Admin, I would like to have access to all recorded transcripts of presentations that my business has been a part of, so that I can redistribute said transcripts to those that need it or moderate transcripts.
3. As a Presenter, I would like fine-grained control over the starting-and-stopping of the transcription, so that I can control what is recorded to the transcript.
4. As a Presenter, I would like to give or revoke a user's Editor role on a transcript, so that I can focus more on the presentation and less on the editing.
5. As a Presenter, I would like to give or revoke a user's Viewer role on a transcript, so that I can control who can view the information.
6. As an Editor, I would like to add notes of key elements to the recorded transcript, so that I can let the viewer know what to focus on.
7. As an Editor, I would like to edit the transcript after the recording has finished, so that I can fix any mistakes or append on new information afterwards.

8. As an Editor, I would like to markup what I have said on the recorded transcript, so that the viewer can better interpret the document.
9. As a Viewer, I would like to read a well-organized, and properly formatted document from the recorded transcript, so that it would be easier to locate specific content.
10. As a Viewer, I would like to be able to give feedback to the Presenter, so that I can communicate with the Presenter on the contents of the transcript.
11. As a Viewer, I would like to be able to publicly markup the document, including highlighting, bookmarking, and labelling, in a nondestructive way, so that I can share with the presenter and other viewers what I valued most from the document.
12. As a Viewer, I would like to be able to rate other viewers' markups, so that I can make more valuable markups visible to more people.
13. As a Viewer, I would like to view an aggregated set of the highest rated markups on a document, so I can quickly see the most valuable information rated by others.

4.2 Technical & Usability Requirements

Jot-Down will require the following technical specifications to be fulfilled:

1. The product must support all platforms that popular virtual meeting platforms support, including Windows, macOS, iOS, and Android.
2. The product must have the option to be available offline as well as with an internet connection.
3. The product must be available either through a publicly available cloud solution or a self-hosted solution.
4. The product must be able to run on low-end workstation laptops with included microphones.
5. The product must be power efficient to a degree that allows users to use the product for extended periods of time, at least two hours of continuous usage.
6. The product must have a consistent user interface across operating systems, regardless of the platform it is running on.
7. The product must have a user interface that is easy to use and is simple enough that anyone can pick up the product and use it quickly and efficiently.
8. The product must encrypt user data that is collected.
9. The product must be highly available, with high uptime and acceptable latency.

4.3 Performance Requirements

Jot-Down will need to satisfy the following performance requirements in order to be widely available:

1. The product must be accessible and ready to use within 10 seconds of opening it on a standard business or university network, assumed to have at least a 10 Mbps connection.
2. The product must only consume 10% or less of a laptop's battery life during a 2-hour period of prolonged use. Consumption of battery life while the device's screen is powered off must be reduced by roughly 50%.
3. The product must be operable on contemporary laptop and desktop devices developed by Dell, Apple, Asus, Lenovo, Microsoft, Acer, Compaq, and Toshiba.

5 High Level Workflow

5.1 Milestones

This section will outline the milestones that the Jot-Down app will be going through. There will be three stages on which the development process will undergo. Jot-Down will begin at the Minimum Viable Product, or MVP phase, then once we're comfortable with the first milestone, we'll continue to the Full Release 1.0 phase, and lastly it will move onto the Full Release 2.0 phase.

5.1.1 Minimum Viable Product

The minimum viable product for Jot-Down will include the following core functionality:

- Presenters will be able to record a full lecture / meeting.
- Presenters will be able to utilize an integrated user interface with applications such as Zoom or WebEx, that allows them to start and stop certain points of the recording.
- Presenters and editors will be able to manually add sections, headers, and timestamps in the transcript to help organize the transcript.
- All users will be able to download a PDF version of the transcript.

5.1.2 Full Release 1.0

The Jot-Down application's full release version 1.0 should be able to automate many of the transcript-generating features, such as:

- Presenters will be able to fully integrate the Jot-Down software with both Microsoft PowerPoint and Google Slides.
- Presenters and editors will have automatic time stamps created when they change slides, allowing all conversations to stay attached to their relevant slides.
- Presenters and editors will be allowed to edit their transcripts before finalizing them in PDF form.
- Admins will have access to an admin site where they can control access of Jot-Down to specific users.
- Admins will have a cloud-based database containing their recorded lectures as well as the corresponding transcripts.
- Presenters can create templates for their transcripts and customize automatic timestamps to fit their own needs, thus automating the transcripts further during recordings.

5.1.3 Full Release 2.0

The Jot-Down application's full release version 2.0 will allow more customization for users in terms of reading and write permissions to transcripts with the following features being added:

- Presenters will be able to give edit and view access to the generated transcripts.
- Admins and presenters will be able to store transcripts to a class or business page, where users can make comments, highlights, or make requests on the transcripts.

5.2 Timeline

The following schedules, broken down into three epics, represents the tentative timeline for the development of Jot-Down. It begins with approximately three months dedicated to the formation of a minimum viable product in 2021 and ends with version 2.0 Full Release shipping in 2022.

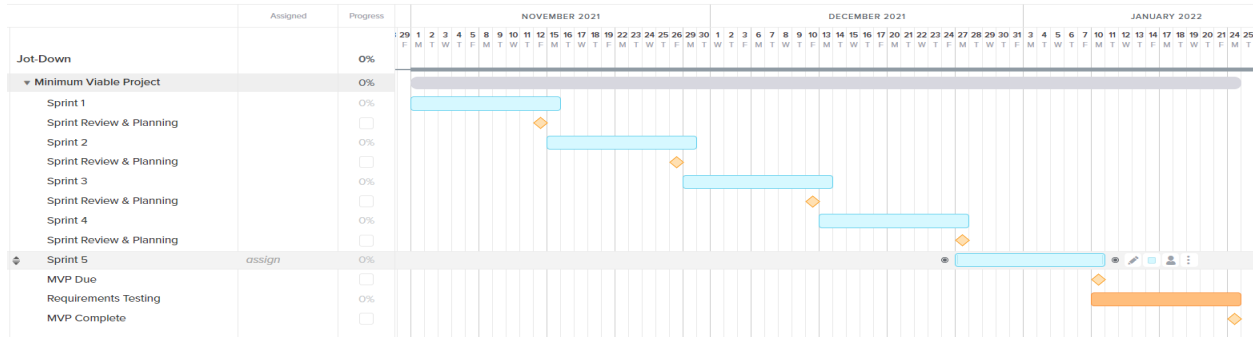


Fig. 1: Screenshots showing the first epic of tasks for the Minimum Viable Product.

Starting in November 2021, we will begin the process of building a minimum viable product, which we expect to have finished and tested by the end of January 2022.

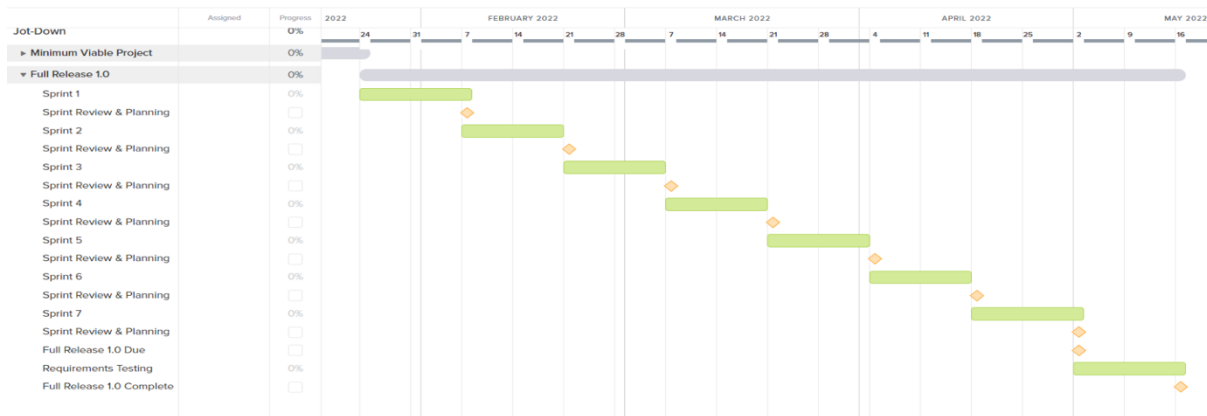


Fig. 2: Screenshots showing the second epic of tasks for version 1.0 Full Release.

Following the completion of our minimum viable product, the development of the first Full Release version will commence, with a two-week cadence of sprints ending in mid-May.

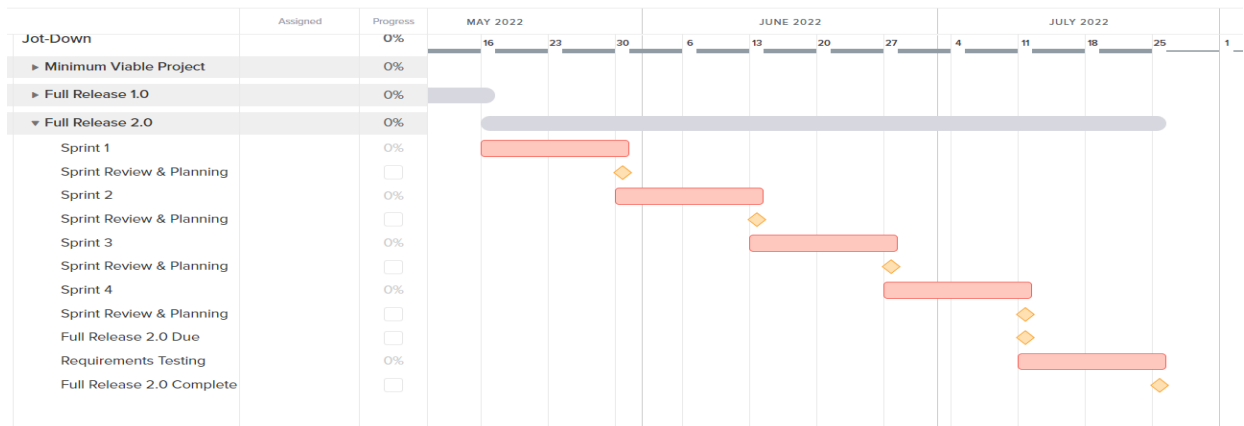


Fig. 3: Screenshots showing the third epic of tasks for version 2.0 Full Release.

Finally, with the completion and release of 1.0 Full Release, we will begin working on the first post-release update. This epic will again follow a two-week cadence of sprints, ending in late July.

The full schedule for the development of Jot-Down can be found at the following link.

Gantt Chart Link:

https://prod.teamgantt.com/gantt/schedule/?ids=2833849&public_keys=HdjzGccF2UPg

6 Acceptance Criteria

Jot Down's final version should fulfill all the following criteria:

- A viewer can view an edited transcript.
- A viewer can comment on an edited transcript.
- A viewer must be notified if the transcript has been edited.
- An editor or presenter can edit the raw transcript.
- An editor or presenter can organize the transcript with headers, sections, and/or notes.
- An editor or presenter can markup the edited transcript after it has been edited.
- A presenter can start and stop the transcription process.
- A presenter can delegate access to the editing process of the transcription.
- A presenter can delegate access to the viewing of the edited transcript.
- A presenter can outline, categorize, and highlight the transcript in real-time.
- A presenter can communicate to the viewers of the transcript.
- A presenter must be notified of any changes to the transcript, made by the editor.
- An admin can manage the access of all users to the program.
- An admin can manage the data recorded by all users of the program.
- An admin can add users to the program.

7 Rapid Prototype

The following figures give a rough idea of the screens that a user may see while using Jot-Down.

In the first figure, the user is in a meeting, in an instance of Zoom or WebEx or any similar video conferencing tool, and Jot down shows up as a panel with multiple options that are relevant to Jot-Down. This panel includes the ability to stop or pause Jot-Down's automatic notetaking feature. After the user ends the meeting via the stop button, the configured editor will receive the notes and be able to edit them in the editor screen.

The editor screen allows the editor to edit the produced notes, including highlighting and reorganizing the notes. As shown in figure 2, these options will be available in a single ribbon at the top of the screen, and the user could make use of the options as they desire.

Before a meeting, a presenter may need to configure who the editor is and who will receive the meeting notes. Using the screen in figure 3, presenters can add editors and recipients to their respective fields, before starting the meeting.

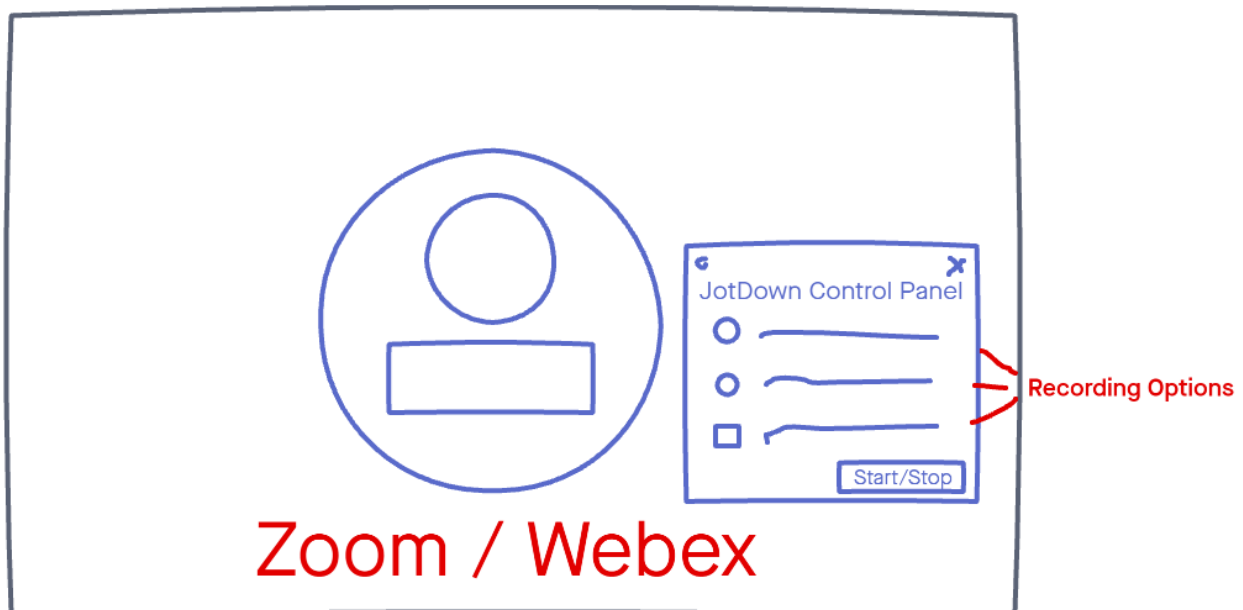


Fig. 4: Screenshot showing Jot-Down as a panel over a WebEx or Zoom client, where the presenter would configure recording settings and start or stop the recording.

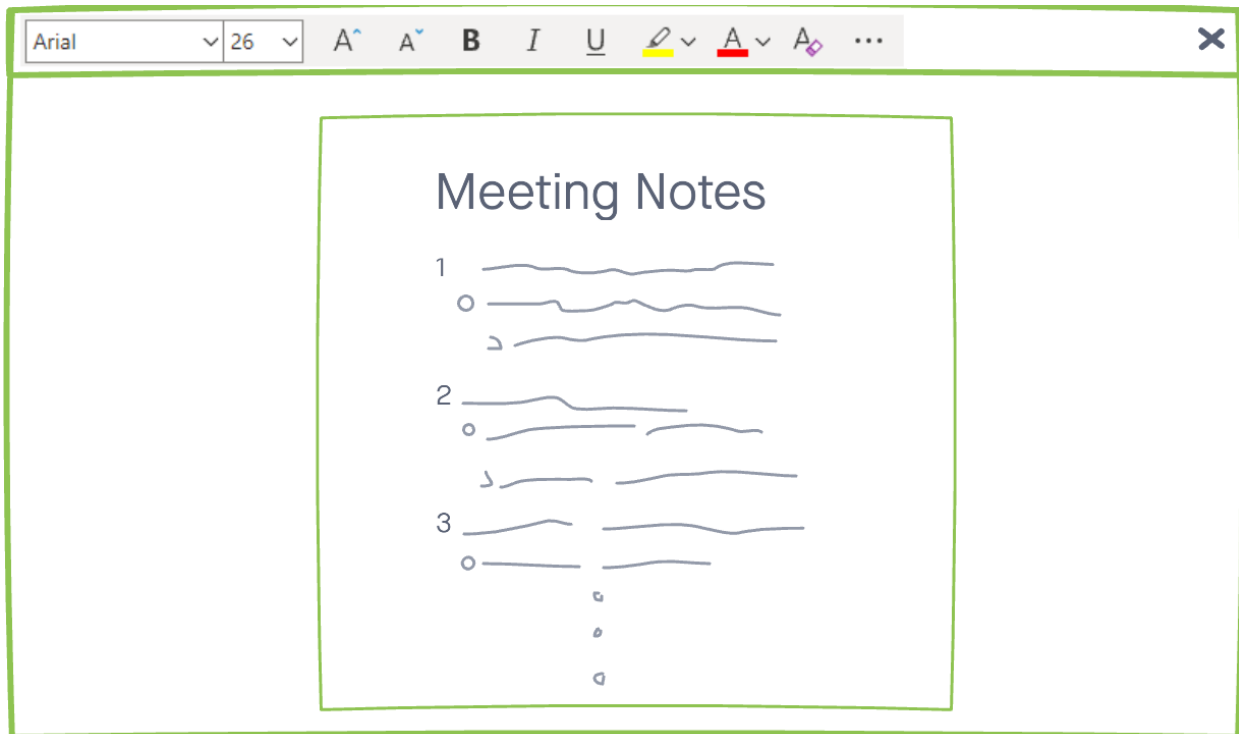


Fig. 5: Screenshot showing the notes generated after a meeting, with tools the editor can use to reformat, highlight, or edit the notes as they desire. After editing, the editor can send the notes to recipients (configured by the presenter).

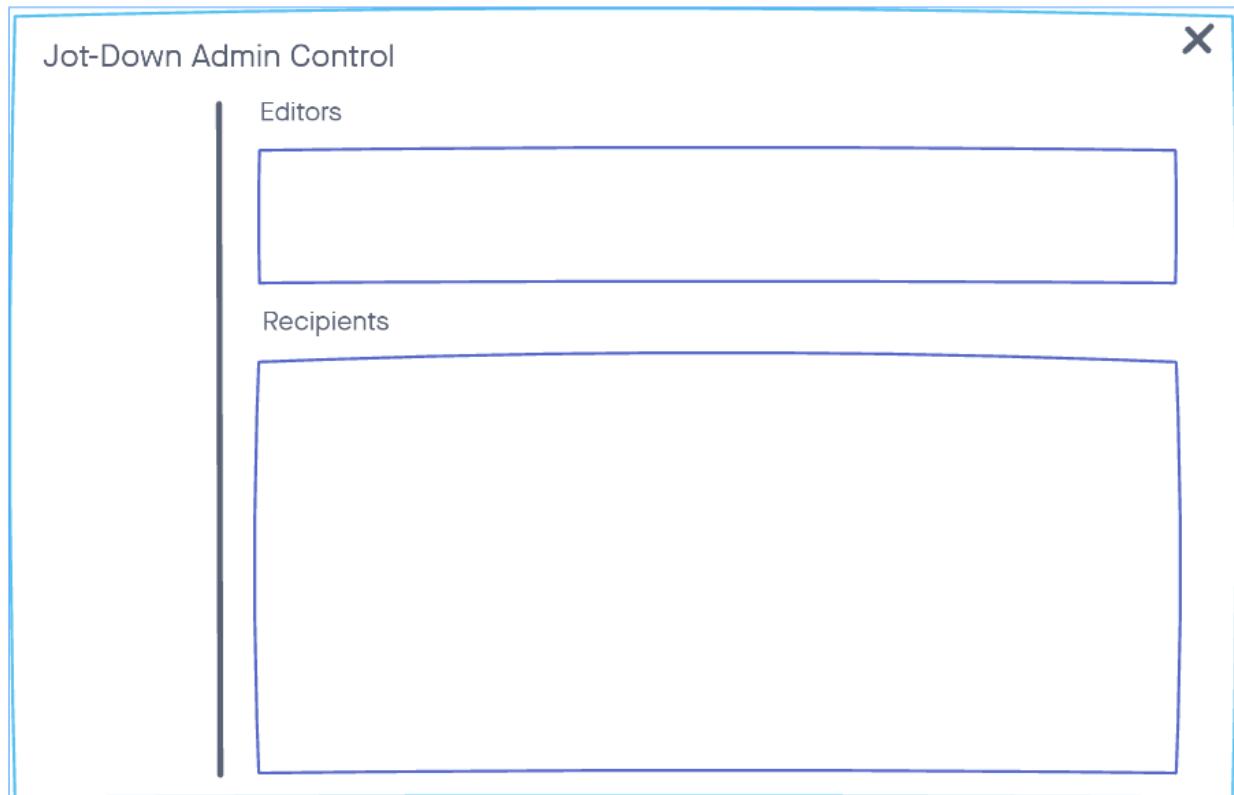


Fig. 6: Screenshot showing a panel where the presenter can configure who can edit the notes and where the notes are sent after the meeting.

Jot-Down

Functional Specifications

1 Platform Overview

Jot-Down's minimum viable product requirements are listed in the following sections:

1.1 Platform Requirements

Software Stack

- Front-end (web): React
- Front-end (mobile): React Native
- Back-end (server): NodeJS (+Express)
- Database: Cloud Firestore (NoSQL)

User Device Target

- Web-app for end-users and administrators.
 - Supports Google Chrome, Mozilla Firefox, Microsoft Edge

Hosting Requirements

- Google Cloud App Engine
 - Hosts the web application in a fully scalable way with zero server management, zero deployment management, and a wide variety of runtime support.

1.2 Platform Rationale

Jot-Down is an audio-recording application which can be used to capture audio from either an application or from the desktop for the sake of producing a transcript for users to be able to look back on in the future. Therefore, we have chosen the following platform requirements:

React was picked for our front-end development because it's free uses a special JSX syntax that lets you combine HTML, CSS and JavaScript together. This will let us rapidly iterate and test our front-end designs, which will let our small team be agile and more efficient.

Google Firebase's Cloud Firestore database will allow us to store data with high availability, able to be accessed across platforms. When we begin work on mobile versions of Jot-Down, this will reduce the time-to-market due to its cross-platform support.

Since we're using React to build our web front-end, we will use React Native to build native versions of Jot-Down for mobile platforms. React Native is built on top of React, so our codebase will be easily portable across web and mobile, reducing the amount of work that must be done and reducing our time-to-market.

2 Architecture & Organization

2.1 High-Level Architecture

The graph below depicts the high-level architecture of Jot-Down and the necessary stack that we will be using. This application will be hosted by the Google Cloud App Engine, with its server-side in Express with a Cloud Firestore NoSQL database. The client side will use React and Google's Speech-To-Text API. These tools would allow for the potential to reach mobile devices on both iOS and Android in future releases.

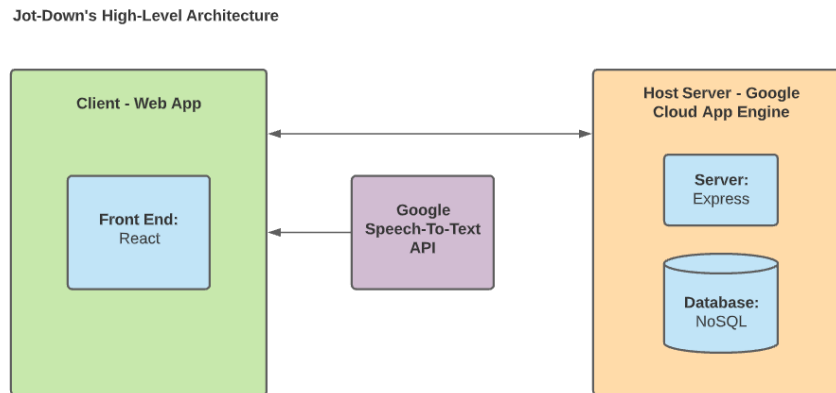
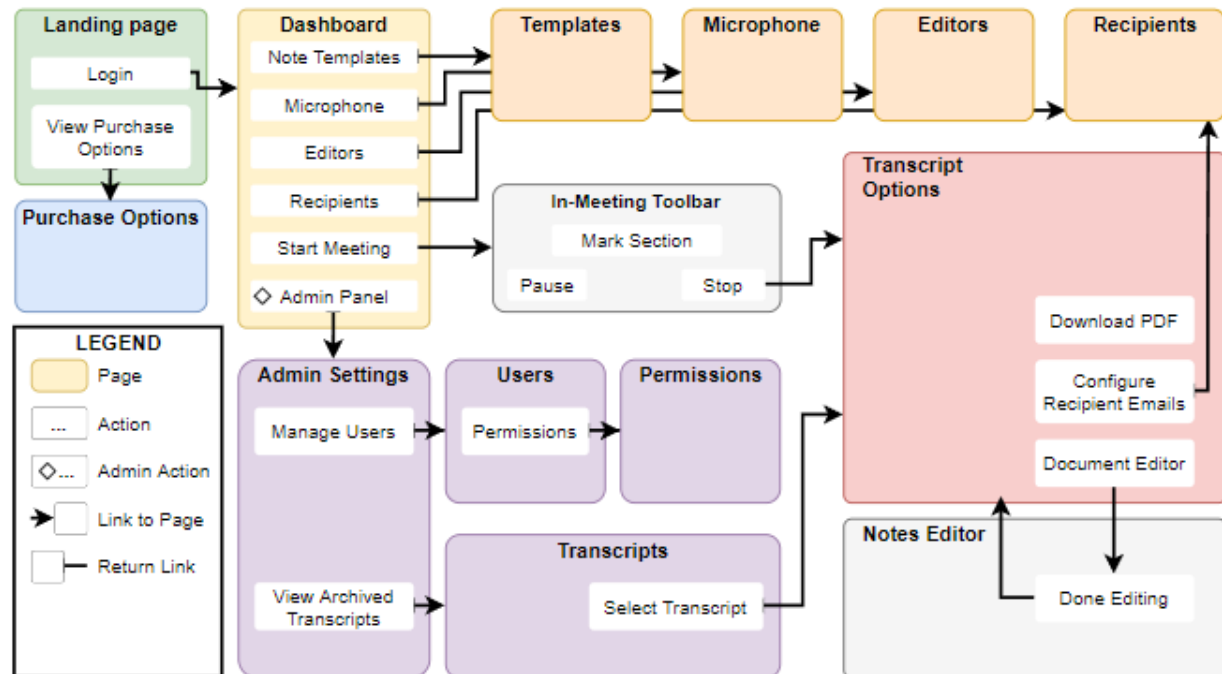


Figure 7: Graph showing the architecture of Jot-Down.

The following flowchart demonstrates how users will use the app and which actions will lead to other screens of the Jot-Down web application.



Source: <https://drive.google.com/file/d/1vUbMdujw-MQGhHplgYo5JeTmtlzC48qN/view?usp=sharing>

4 Wireframe Layouts

Screen 1: Landing Page

Jot-Down

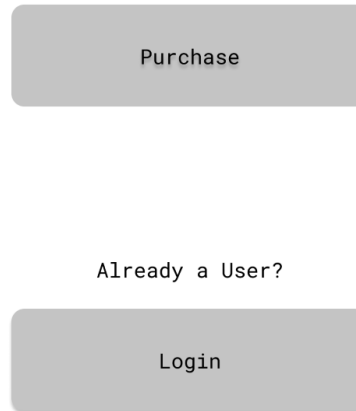


Figure 9: Wireframe demonstrating layout of Screen 1: the Landing Page. The option to purchase licenses is here in place of a 'sign up' option.

Screen 2: Login Page

Welcome Back

Forgot Password? [Reset](#)

Figure 10: Wireframe demonstrating layout of Screen 2: the Login Page.

Screen 3: Purchase Page

Jot-Down

Contact Sales

909-xxx-xxxx

contactsales@sales.com

Our sales representatives are ready to help you
choose the purchase plan that suits your needs.

Back

Figure 11: Wireframe demonstrating layout of Screen 3: the Purchase Page. The user will contact a sales representative and discuss their purchasing options (this process is subject to change).

Screen 4: User Dashboard

Dashboard

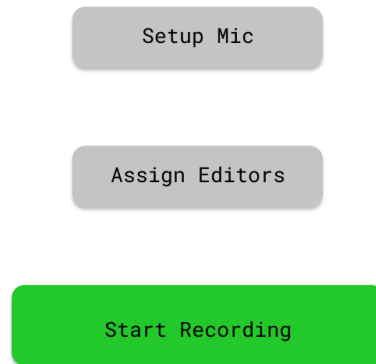


Figure 12: Wireframe demonstrating layout of Screen 4: the User Dashboard.

Screen 5: Microphone Setup

Dashboard

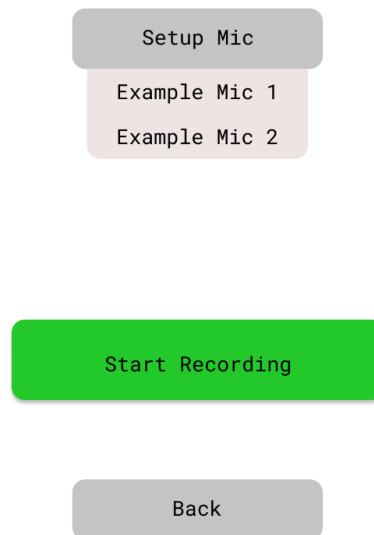


Figure 13: Wireframe demonstrating layout of Screen 5: the Microphone Setup Page. On this page, the user could choose which microphone they want to use and test their mic.

Screen 6: Assign Editors Page

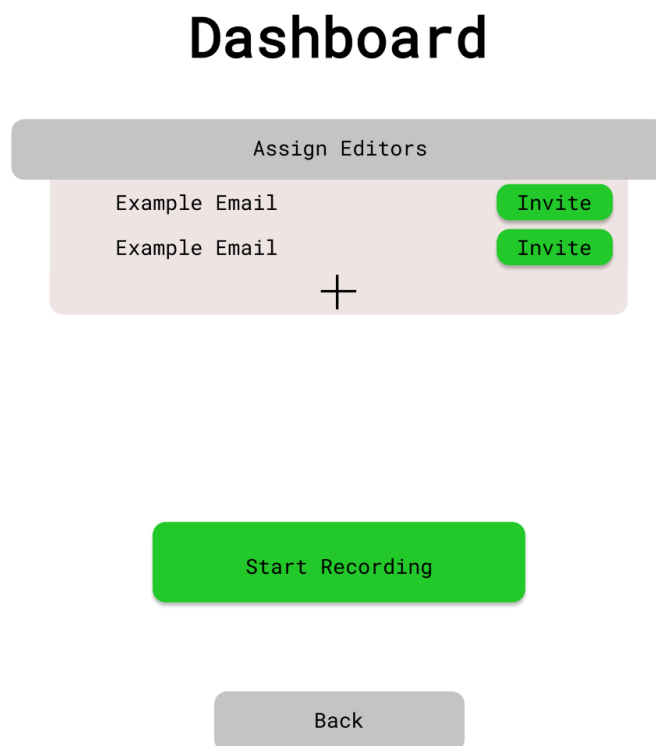


Figure 14: Wireframe demonstrating layout of Screen 6: Assign Editors Page. The user can send emails with links to the meeting out to audience member's email addresses.

Screen 7: Meeting Started Page

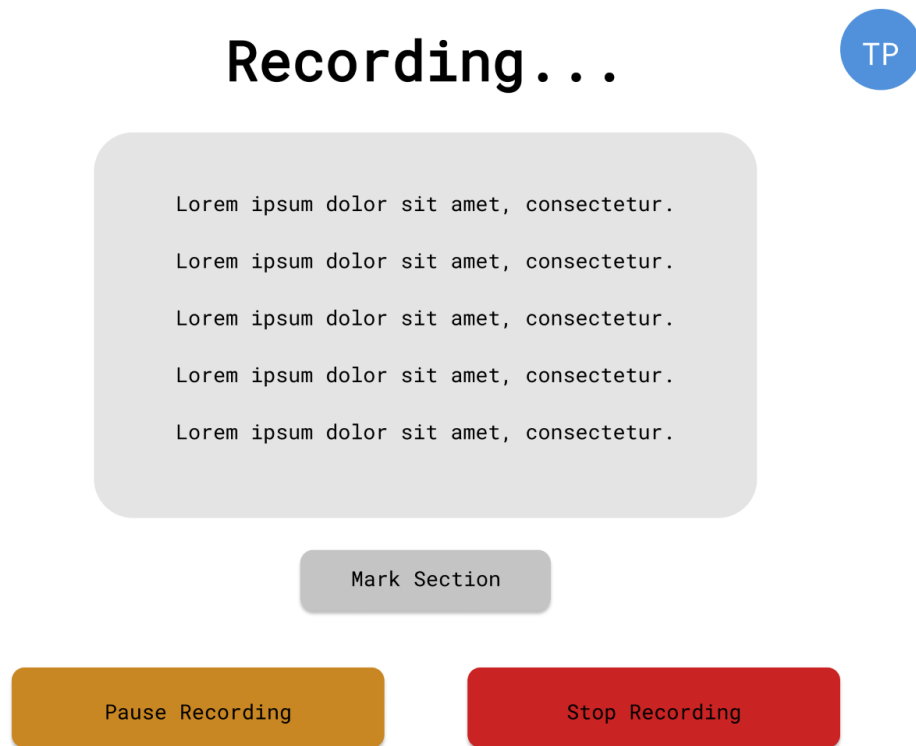


Figure 15: Wireframe demonstrating layout of Screen 7: the Main Meeting Page. The text box in the middle is the transcript being created in real-time.

Screen 8: Meeting Paused Page

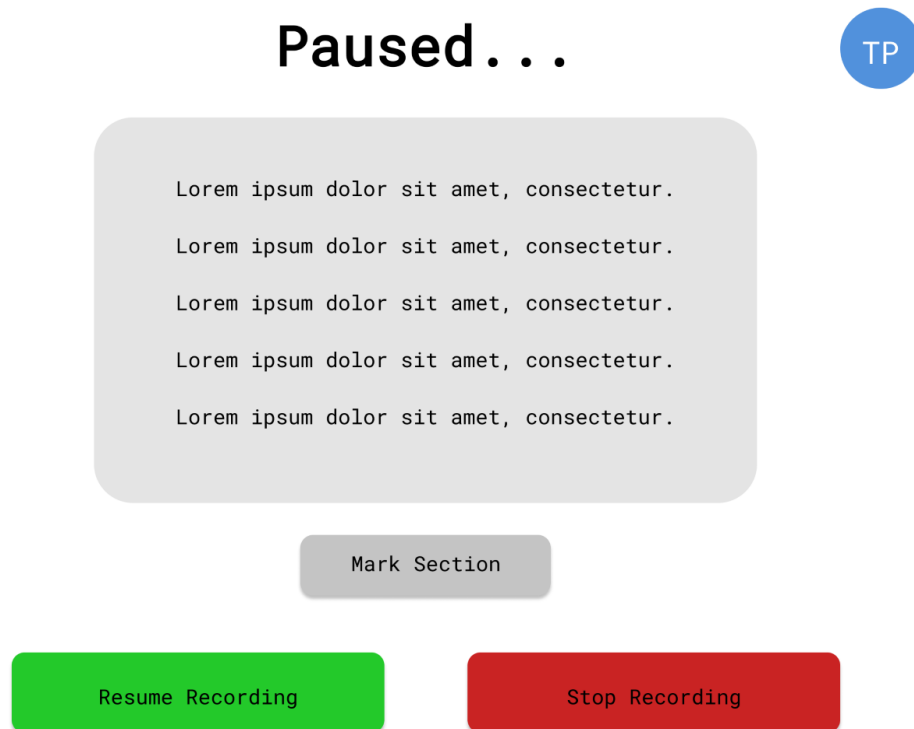


Figure 16: Wireframe demonstrating layout of Screen 8: the Paused Meeting Page. This is the page that would show up when the user clicks the “Pause Recording” button on the meeting page.

Screen 9: Mark Section Page

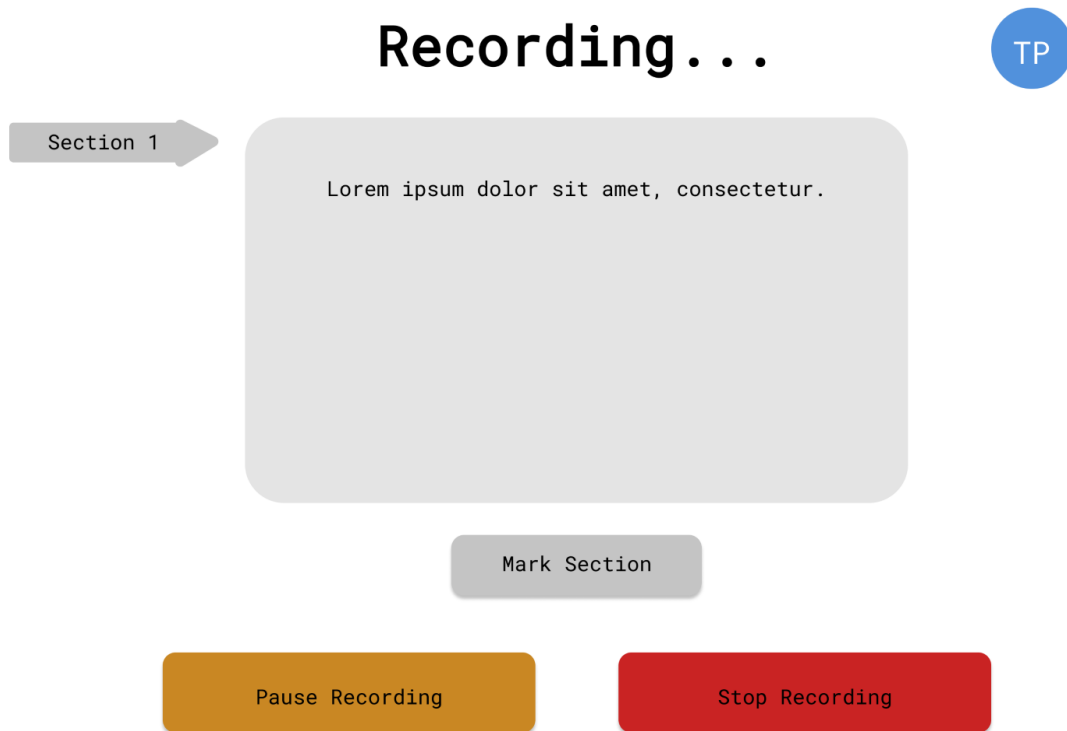


Figure 17: Wireframe demonstrating layout of Screen 9: the Mark Section Page. This is where the speaker could mark off different sections of the transcript that's being generated for readers.

Screen 10: Icon Editor Page

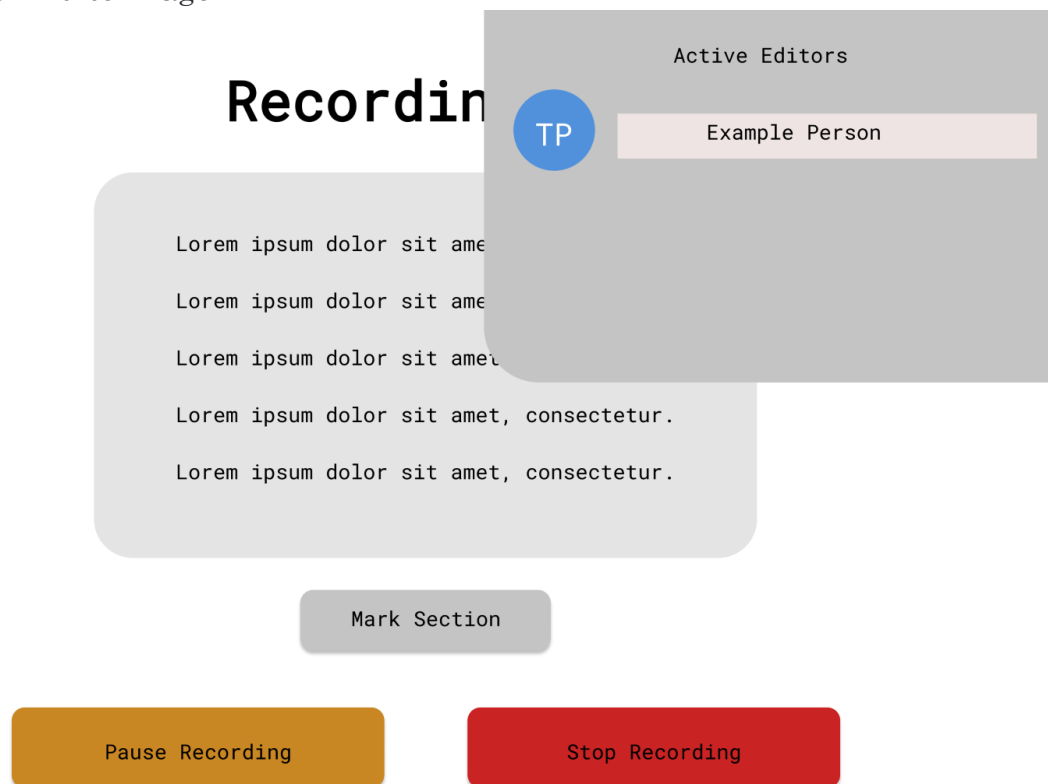


Figure 18: Wireframe demonstrating layout of Screen 10: the Icon Editor Page. On this page, the presenter can see which editors are currently overseeing the transcript being generated.

Screen 11: Stop Recording Page

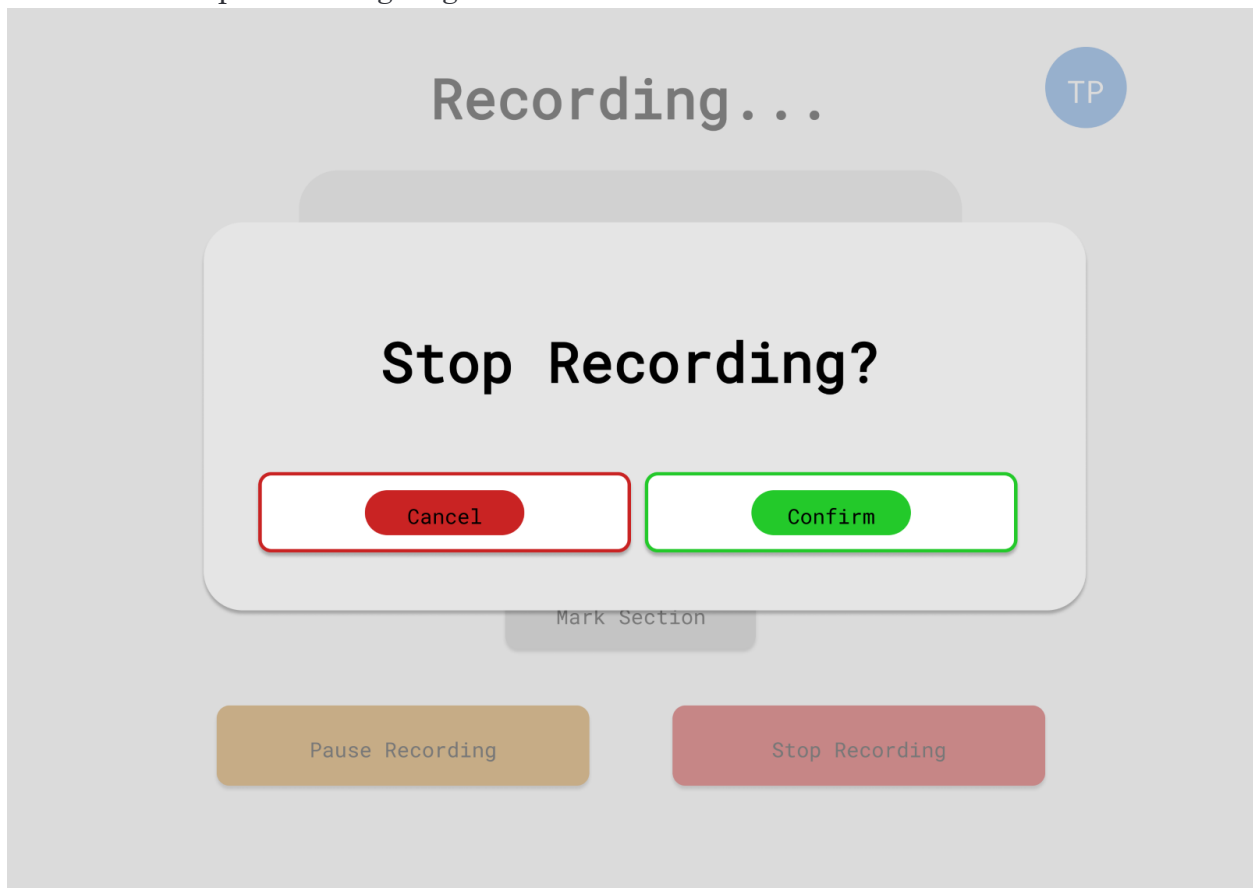


Figure 19: Wireframe demonstrating layout of Screen 11: the Stop Recording Page. This is the page that is pulled up when the presenter presses the “Stop Recording” button.

Screen 12: Edit Transcript

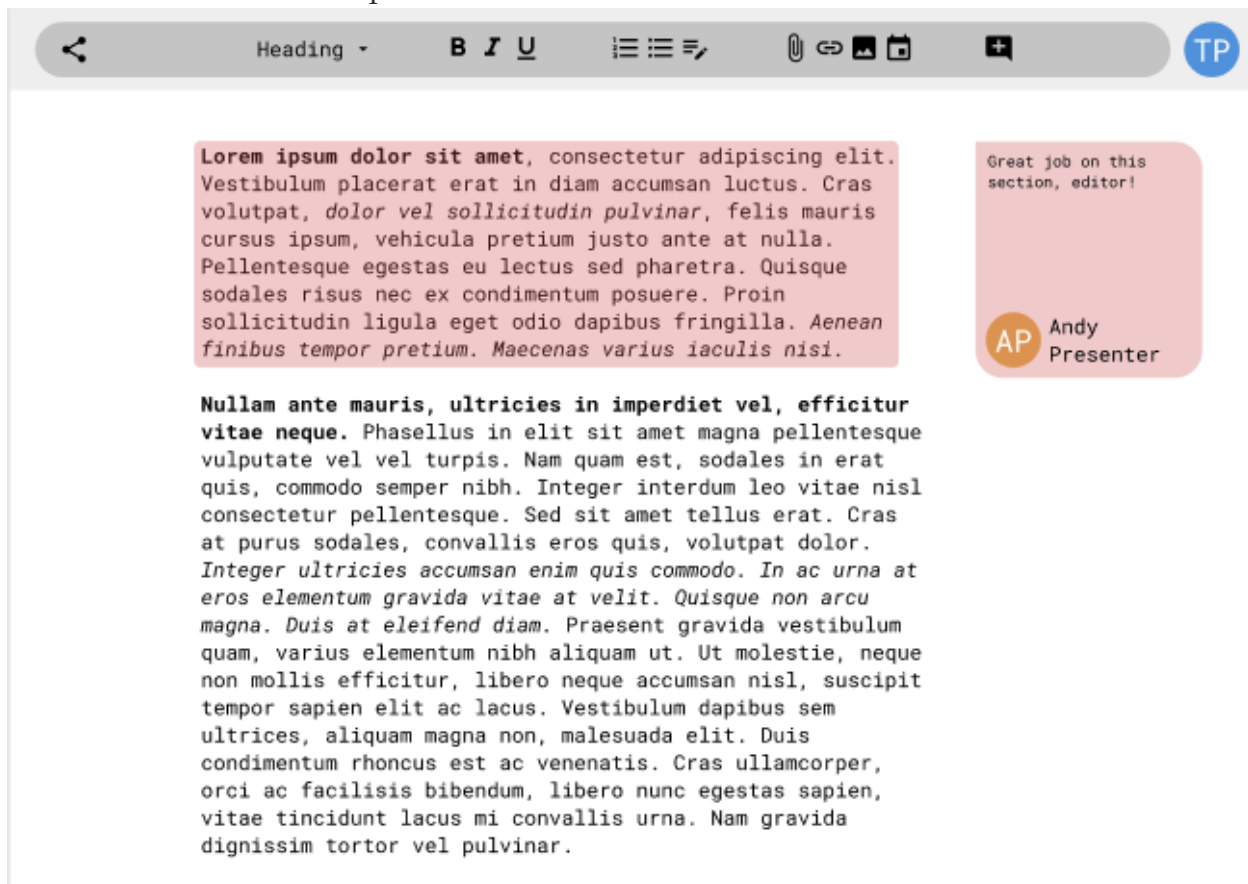


Figure 20: Wireframe demonstrating layout of Screen 12: Edit Transcript Page. This is the page that opens once the user confirms the end of a recording as well as when an Admin goes to view a specific transcript.

Screen 13: Send Notes

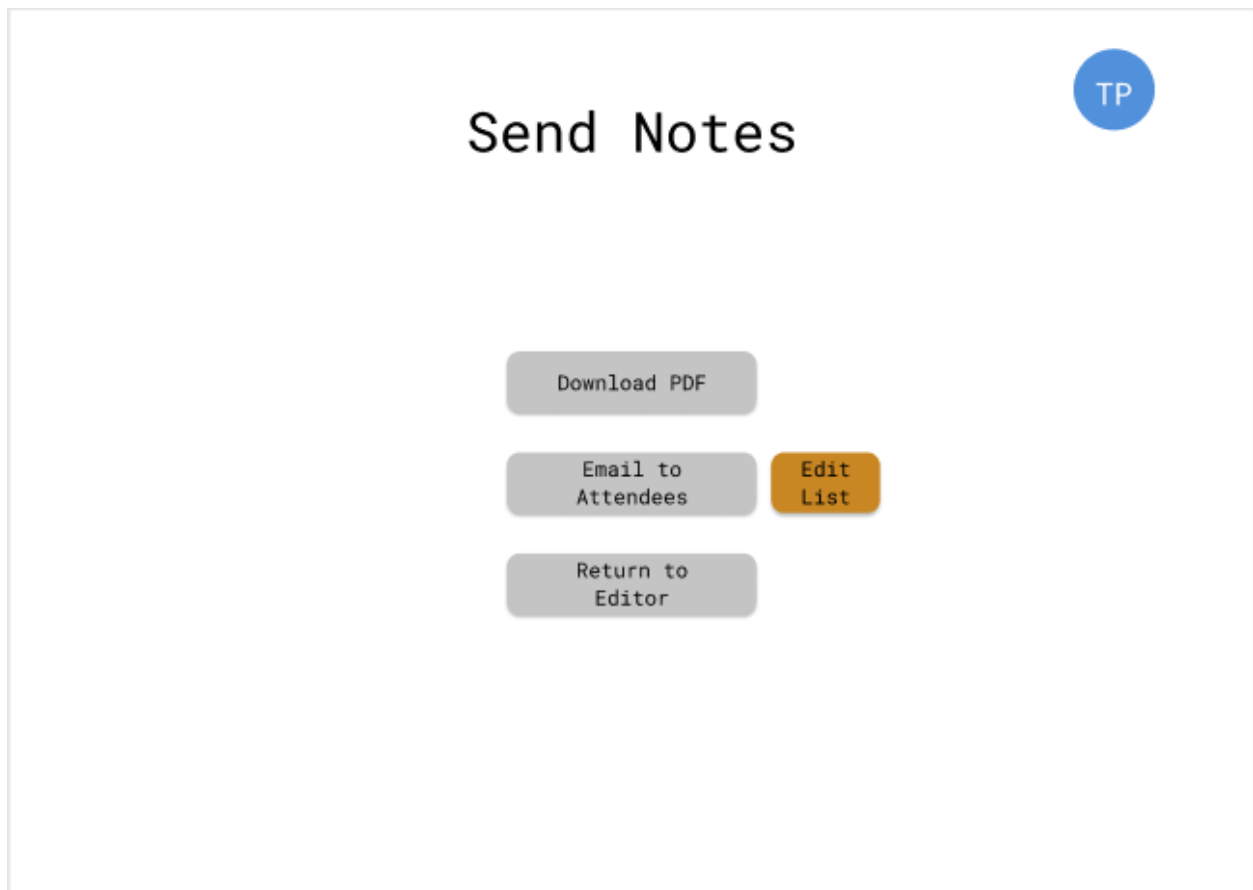


Figure 21: Wireframe demonstrating layout of Screen 13: Send Notes Page. This is the page that opens once the user confirms that they are done editing a transcript.

Screen 14: Edit Email for Attendees

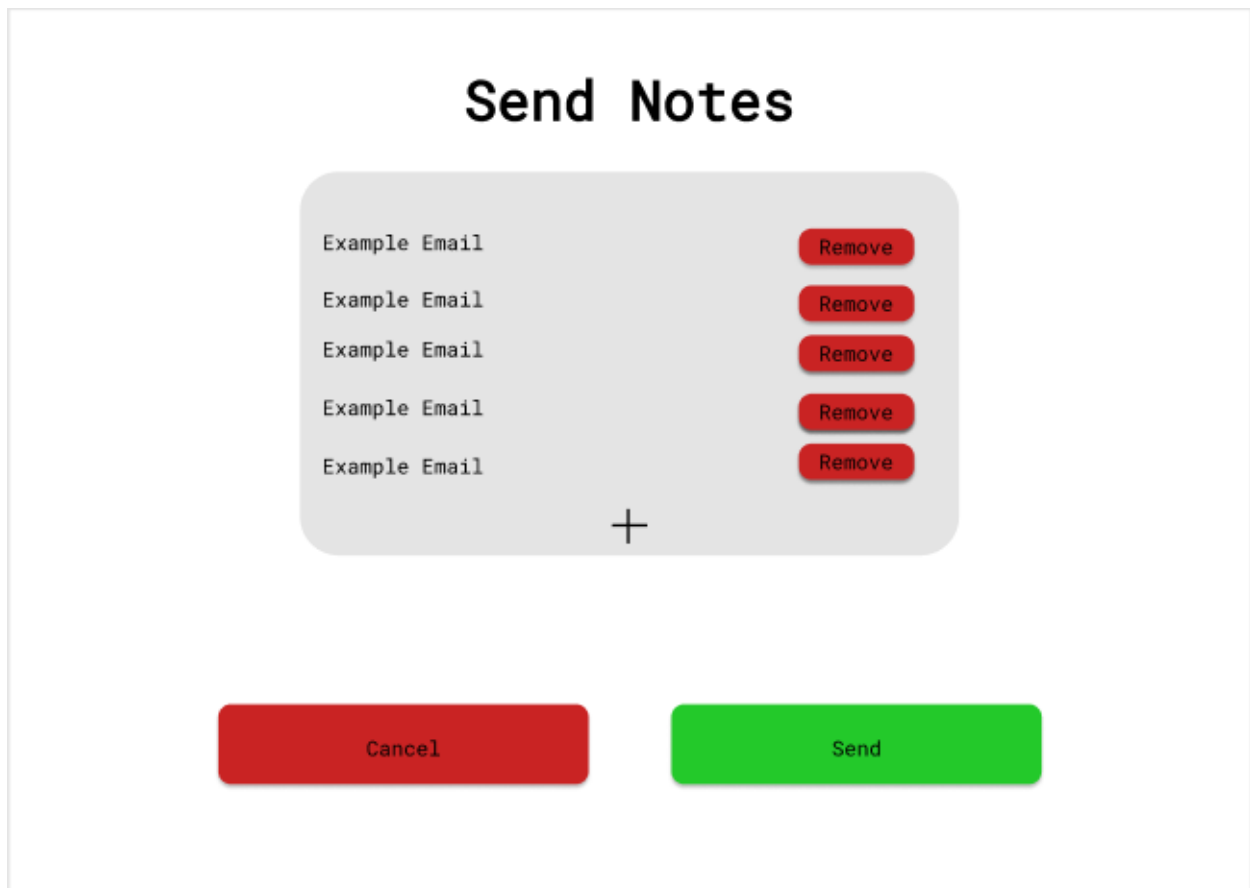


Figure 22: Wireframe demonstrating layout of Screen 14: Edit Email for Attendees Page. This is the page that appears when the user clicks on the Edit button for the attendee list where they can manually remove and add specific users.

Screen 15: Contact Sales



Figure 23: Wireframe demonstrating layout of Screen 15: Contact Sales Page. This is the page that appears if the user does not have an account or login provided by their admin. Here they are given information for sales representatives to contact.

Screen 16: Admin Page

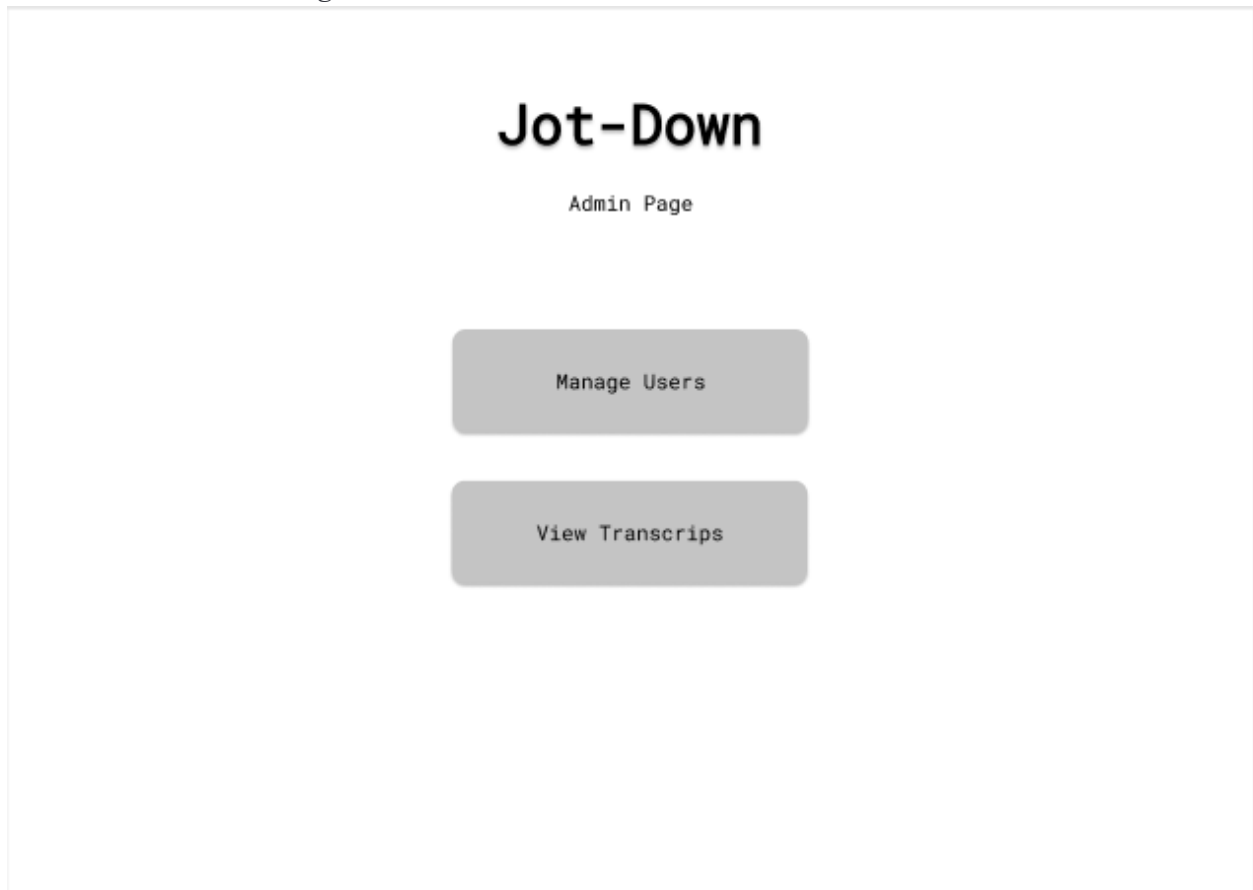


Figure 24: Wireframe demonstrating layout of Screen 16: Admin Page. This is the page that appears if a user signs in with admin credentials.

Screen 17: Manage Users

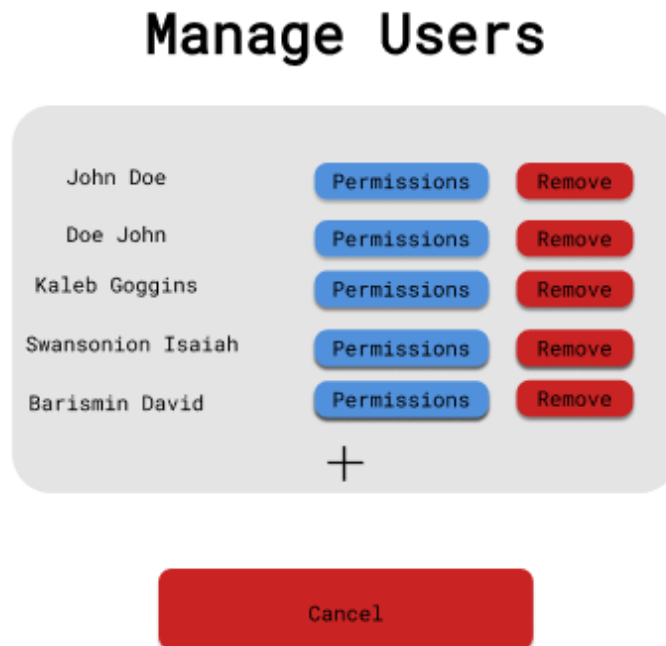


Figure 25: Wireframe demonstrating layout of Screen 17: Manage Users page. This is the page that appears when the user clicks on the “Manage Users” button from the admin screen. Here the user gets a list of all active users under their license.

Screen 18: Transcripts

Transcripts

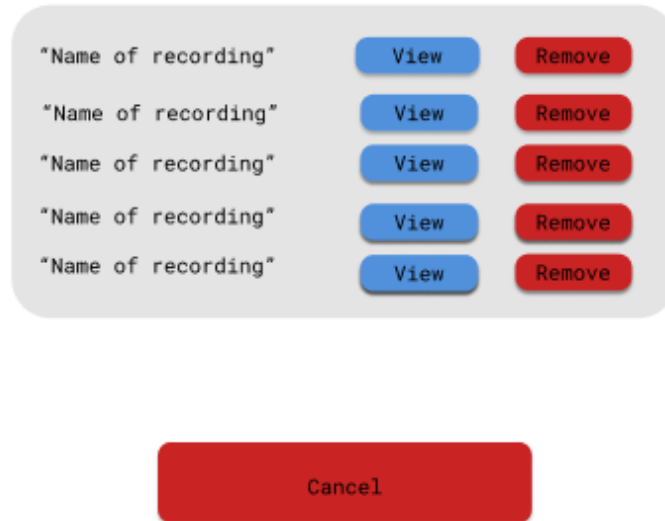


Figure 26: Wireframe demonstrating layout of Screen 18: Transcripts page. This is the page that appears once the admin selects the “View Transcripts” button. Here the user can get access to all recording meeting transcripts and choose to edit or remove them.

Screen 19: Permissions Page

Permissions

	View	Record	Edit
John Doe	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Doe John	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Kaleb Goggins	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Swansonion Isaiah	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Barismin David	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Cancel

Figure 27: Wireframe demonstrating layout of Screen 19: the Permissions Page. This is what appears if the admin clicks on the “Permissions” button on the manage users page. Here, the admin can change all view, record, and edit permissions for each active user.

Figma Link: <https://www.figma.com/file/1oHeGvPZdnGc0fxNF7kCaz/Jot-Down?node-id=0%3A1>

5 User Feedback

The following is a general overview of the significant points that were made during the testing of the Jot-Down prototype:

- User didn't know why there wasn't a way to sign up without having to pay.
- User wanted to know what kind of plans the sales representative would offer them.
- User didn't understand why they wouldn't just record the meeting instead of doing this.
- User didn't know what the process of adding new people to the meeting would look like.
- User suggested making the app more aesthetically pleasing.
- User was confused on where to invite users to have access to the transcript.
- User wants to have an overview of who has access to the transcript.
- User was confused on why they need to select view on a user in the Permission Page since they invited them to see the transcript.
- User suggested to have the landing page to be able to already log in instead of going to another page.
- User would like to have the users that have permission to edit/record/view to be in separate categories.

Jot-Down

Software Project Management Plan

1 Risk Analysis

Risk is computed as (impact × probability). **Impact** is defined on a scale from 1 - 5 where a 1 (5) means the risk coming true would have a minimum (maximum) impact on the project's success. **Probability** is also defined on a scale from 1 - 5 with the following meaning:

- Slight possibility under unusual conditions
- It could happen
- 50/50 chance
- Fair chance it will happen
- High likelihood

1.1 Risk Analysis Matrix

The following risk analysis matrix details the most predominant risks in building Jot-Down.

RISK #	DESCRIPTION	I	P	R
R1	Customers are not interested in our product.	5	4	20
R2	A React update causes the website to require refactoring in order to function properly.	4	4	16
R3	Not finding a suitable speech-to-text software for our product.	5	2	10
R4	Our NodeJs server breaks, causing the flow of data to and from our app to be halted.	5	2	10
R5	Risk of our database bottlenecking	5	2	10
R6	A team members losses access to their computer which prevents future contributions to Jot-Down	4	2	8
R7	Our team decides to move away from an initially chosen platform (such as Cloud Firestore).	3	2	6
R8	Our Cloud Firestore database becomes unavailable.	3	2	6
R9	A team member has a personal emergency which could impact the productivity of creating Jot-Down	5	1	5
R10	Transcripts are saved to the database incorrectly.	4	1	4

Table 5: Risk analysis matrix listing the predominant risks and their risk score, computed as Risk = Impact * Probability ($R = I * P$). Risks with a risk score greater or equal to 12 are highlighted in Red to demonstrate they are high-risk and need to be TAMEd.

1.2 Risk Management Strategies

The responses below are for the possible risks that generate a risk score equal to 12 and higher as described in 1.1. The responses will go over our strategy for the Transfer Risk, Accept Risk, Mitigate Risk, and Eliminate Risk. Each response will also be assigned to capable team members who will be overseeing the risk.

1.2.1 Risk Response Strategy for R1 (Lack of Interest)

Risk R1 (Lack of interest) will be TAMEd in the following ways:

- **Transfer Risk** – N/A
- **Accept Risk** – We will move forward with our plans for Jot-Down without customers who are disinterested in our product.
- **Mitigate Risk** - We will listen to our customers' feedback in order to make our product more appealing to potential clients.
- **Eliminate Risk** – N/A

Risk strategies assigned to: Tomas Perez and Isaiah Swanson.

1.2.2 Risk Response Strategy for R2 (React Update)

Risk R2 (React Update) will be TAMEd in the following ways:

- **Transfer Risk** - Outsource application development
- **Accept Risk** – Refactor the website as needed if react updates
- **Mitigate Risk** – N/A
- **Eliminate Risk** – N/A

Risk strategies assigned to: Kaleb Coggins and Isaiah Swanson

2 Deliverables, Components & Actions

The following list describes the deliverables, components, actions and sub-actions of Jot-Down:

- Landing Screen (Screen 1)
 - Component Layout
 - Placement of Purchase button and Log In button
 - Component Styling
 - Add Jot-Down Logo, style Purchase/Log-In buttons
 - Front-End
 - Implement following methods and associated action handlers
 - btnPurchasePlans() (navigates to Purchase screen)
 - BtnLogIn() (navigates to Log In screen)
 - Back-End
 - No back-end is needed for this screen
- Login Page (Screen 2)
 - Component Layout
 - Title, Username field
 - Password field
 - Forgot password button
 - Register account button
 - Log in button
 - Back button
 - Component Styling
 - Add logo, match fonts & color palette
 - Front-End
 - Implement following methods and associated action handlers
 - BtnRegisterNewUser()
 - BtnLogInUser()
 - BtnDismissScreen() (return to previous screen)
 - Back-End
 - Implement following methods on the server

- UserExists() (for forgot password flow)
 - AddNewUser()
 - LoginUser()
- Purchase Page (Screen 3)
 - Component Layout
 - Title
 - Contact sales button
 - Phone number label
 - Email address label
 - Description label
 - Back button
 - Component Styling
 - Add logo, match fonts & color palette
 - Front-End
 - Implement following methods and associated action handlers
 - BtnContactSalesLink()
 - BtnDismissScreen() (return to previous screen)
 - Back-End
 - No back end functionality required
- User Dashboard (Screen 4)
 - Component Layout
 - Placement of the following buttons:
 - Mic Configuration
 - Editor Configuration
 - Start Meeting
 - Component Styling
 - Add Jot-Down Logo, style buttons
 - Front-End
 - Implement following methods and associated action handlers
 - btnMicSetup() (navigate to Mic Config Screen)
 - btnEditorSetup() (navigate to Editor Config Screen)
 - BtnStartMeeting() (navigate to Meeting Started Page)
 - Back-End
 - No back-end functionality necessary
- Admin Page (Screen 5)
 - Component Layout
 - Title
 - Manage users button
 - View transcripts button
 - Logout button
 - Component Styling
 - Add logo, match fonts & color palette

- Front-End
 - Implement following methods and associated action handlers
 - BtnManageUsers()
 - BtnViewTranscripts()
 - BtnLogoutUser()
- Back-End
 - No back-end functionality required
- Manage Users (Screen 6)
 - Component Layout
 - Title
 - List of users
 - Name label
 - Permissions button
 - Remove button
 - Cancel button
 - Component Styling
 - Add logo, match fonts & color palette
 - Front-End
 - Implement following methods and associated action handlers
 - PopulateUserList()
 - BtnEditUserPermissions(user)
 - BtnRemoveUser(user)
 - BtnDismissScreen()
 - Back-End
 - Implement following methods on server
 - FetchUsers()
 - UpdateUserPermissions(user, permissions)
 - RemoveUser(user)
 - LogoutSession()
- View Transcripts (Screen 7)
 - Component Layout
 - Title
 - List of saved transcripts
 - Recording name label
 - View button
 - Delete button
 - Component Styling
 - Add logo, match fonts & color palette
 - Front-End
 - Implement the following methods and associated action handlers
 - PopulateTranscriptList()
 - BtnViewTranscript()

- BtnDeleteTranscript()
 - BtnDismissScreen()
 - Back-End
 - Implement the following methods on the server
 - FetchTranscripts()
 - FetchTranscript(transcript)
 - DeleteTranscript(transcript)
- In-Meeting Dashboard (Screen 8)
 - Component Layout
 - Title
 - Transcript viewport
 - Buttons
 - Mark Section
 - Pause Recording
 - End Recording
 - Show Editors (Hamburger Button)
 - (hidden) Editor Panel, containing the following components
 - Per Editor:
 - User Icon
 - User Name
 - (hidden) End meeting confirmation pop-up, containing the following components:
 - Title
 -
 - Component Styling
 - Add logo and match styling, colors, etc
 - Front-End
 - Implement following methods and associated action handlers
 - BtnMarkSection()
 - BtnPauseRecording()
 - BtnEndRecording() (show confirmation pop-up)
 - BtnActiveEditors() (toggle Active Editor Panel visibility)
 - Back-End
 - Implement the following back-end methods
 - StartRecording()
 - MarkTranscriptSection()
 - StopRecording()
 - SaveTranscript()
 - ResetTranscript()
- In-Meeting Permissions (Screen 9)
 - Component Layout
 - Title

- List of users
 - View permission checkmark
 - Record permission checkmark
 - Edit permission checkmark
 - Cancel button
 - Component Styling
 - Add logo, match fonts & color palette
 - Front-End
 - Implement following methods and associated action handlers
 - PopulateUserListWithPermissions()
 - BtnUpdatePermission(user, permission)
 - BtnDismissScreen()
 - Back-End
 - Implement following methods on server
 - FetchUsers()
 - FetchUserPermissions()
 - UpdateUserPermissions(user, permission)
- Edit Transcript Screen (Screen 10)
 - Component Layout
 - Editing tools
 - Text style
 - Bold, italic, underline
 - List style, Alignment
 - Add attachment, add link, add picture, add calendar date
 - Share + Export
 - Transcript content
 - User comments
 - Component Styling
 - Add logo, match fonts & color palette
 - Use standard icons from Google's library
 - Front-End
 - Implement the following methods and associated action handlers
 - PopulateTranscriptContent()
 - OnUpdateTranscriptContent()
 - OnSelectText()
 - Editing Tools:
 - BoldSelected()
 - ItalisizeSelection()
 - UnderlineSelected()
 - StyleList()
 - SetBackgroundColor()
 - SetTextColor()

- Back-End
 - Implement the following back-end methods and processes
 - SaveTranscript()
- Share Transcript Screen (Screen 11)
 - Component Layout
 - Download PDF Button
 - Email to attendees button
 - Configure recipients button
 - Edit Transcript
 - Component Styling
 - Add logo, match fonts & color palette
 - Front-End
 - Implement the following methods
 - BtnDownloadPdf()
 - BtnEmailTranscript()
 - BtnEditRecipients() (navigate to recipient config screen)
 - BtnEditTranscript() (navigate to editing screen)
 - Back-End
 - Implement the following back-end methods and processes
 - GetTranscriptPdf()
 - EmailTranscript()

3 Teams and Organization

In the following sections, team roles, organization, and supporting tools that the team will utilize.

3.1 Team Roles & Organization

The structure for the team roles and organization is described below in both a chart and diagram. Within the chart is a brief description of each role, along with its significance to the development process of the application.

ROLE	ASSIGNED PERSON	GENERAL RESPONSIBILITIES	IMPORTANCE OF ROLE TO PROJECT SUCCESS
FULL-STACK	David Barsamian	Assisting in the development and maintenance of the front and back end.	By assisting engineers in both front and back-end sides of the application, there is less of a chance of development being held up by one specific side of the application.
PROJECT MANAGER	Tomas Perez	Plans, organizes, and directs the completion of current and future projects.	By organizing sprints, sprint reviews, and following the SCRUM methodology, the team will be able to stay on track and stay within budget.
FRONT-END	Isaiah Swanson	Maintaining and developing the front end of the application.	By creating a UI and UX that is engaging and appealing, more user traffic is likely to increase.
SERVER-DEV	Kaleb Coggins	Maintaining and developing server development.	By creating a MVC framework that the front-end can communicate with, the application can be routed efficiently to the database for utilizing data.
DATABASE	Daniel Furdui	Creating and managing database architecture.	By creating a solid database architecture, the application's operations with data will be optimized; leading to an overall fast and efficient application.

Table 6: Table identifying the key roles and people/resources for this product's success.

Below is a graph detailing the Jot-Down Team Structure. Most of the engineering team will be split into a front-end and back-end team, with the back-end team also having specialized roles for the database and server. The front-end engineers will be working on the UI/UX design of the application. A full-stack developer will oversee the work of both the front-end and back-end teams, and will be able to assist either team if need be. All roles will report to the project manager who oversees and tracks all development. This project manager is also capable of performing tasks on both the front and back end side of the application.

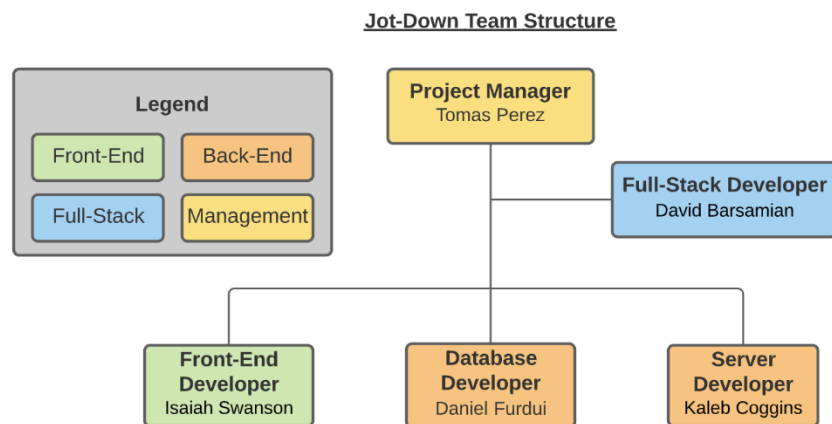


Figure 28: Organizational structure of the team showing who communicates and reports to who.

3.2 Supporting Tools

The below table shows each of the software tools we plan on using as we work on Jot-Down. It gives a brief overview of the purpose of the software, the general functionalities of it, and the yearly cost for using these software tools.

SW TOOL	PURPOSE	GENERAL FUNCTIONALITIES	ANTICIPATED YEARLY COST TO COMPANY
TRELLO	Provides an organized task board for our teams to reference.	Tello allows us to keep track of tasks which either have yet to be done or need to be done so that our team members can plan accordingly.	\$210 annually per user (Enterprise).
SLACK	Provides an easy to access avenue for communication between staff members.	Slack is a communication platform which offers direct messaging, group messaging, and virtual calls as methods of communication between team members.	\$210 annually per user (Business+).
GITHUB	Provides an easy way to save and share source code with one another	Git-Hub allows us to store code in repositories, which could be accessed wherever and whenever. Git-Hub also allows us to share our code with as many people as we want	\$252 annually (Enterprise)

Table 7: Table identifying the key roles and people/resources for this product's success.

4 Schedule

The following sections include both our high-level Gantt chart as well as individual breakdowns of each sprint. For the high-level Gantt chart, there are eight sprints that fall within the scope of the project in order to achieve the minimum viable product. Next to the list of sprints, there is a column for estimated hours for both the sprints and the overall project. There is also a visual timeline of the entire process on the right-hand side.

The next sub-sections describe each sprint in detail, with their corresponding tasks that are necessary to complete each sprint. The same estimation of hours and visual breakdown of each task is present, as it is in the high-level view. There is also a breakdown for when each task is going to start and end within the timeline.

4.1 High-Level Gantt Chart

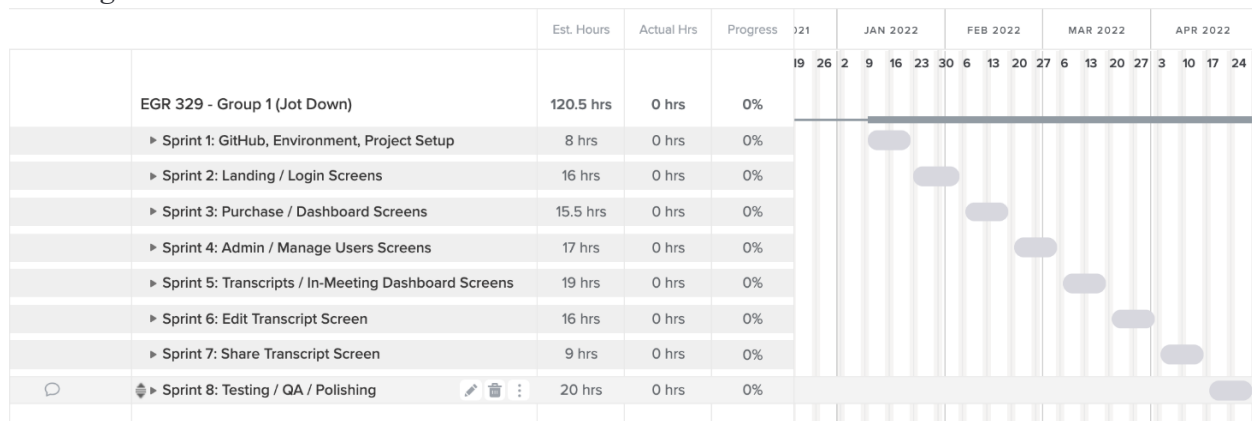


Figure 29: Screenshot of the Gantt chart schedule. See the following sub-sections for more details.

4.2 Sprint Cycle Gantt Charts

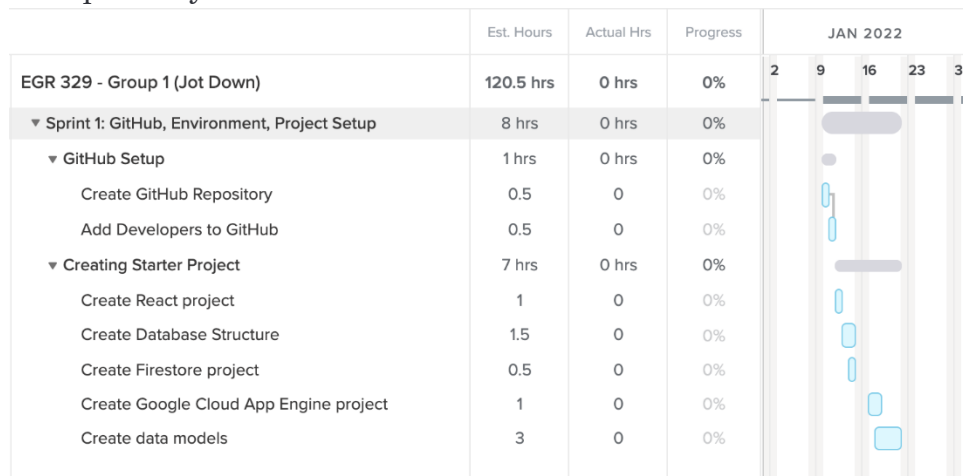


Figure 30: Screenshot of the Gantt chart schedule for Sprint 1.

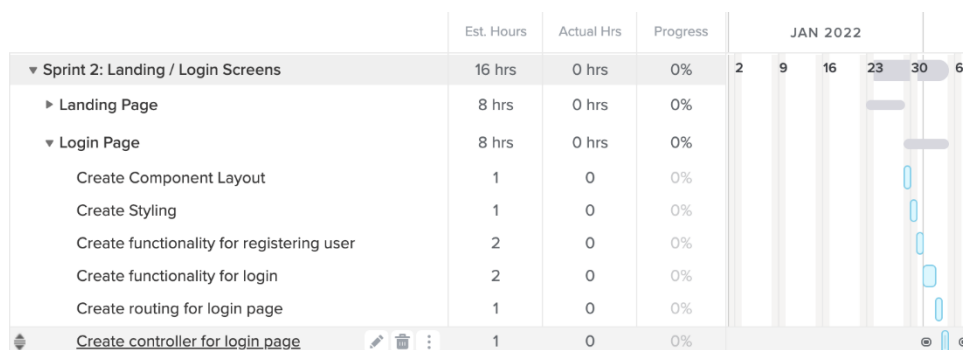


Figure 31: Screenshot of the Gantt chart schedule for Sprint 2.

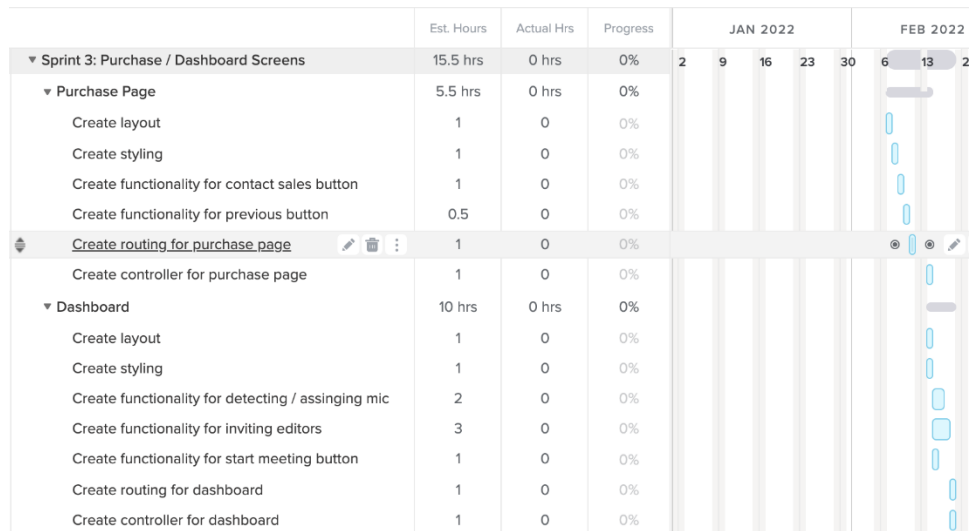


Figure 32: Screenshot of the Gantt chart schedule for Sprint 3.

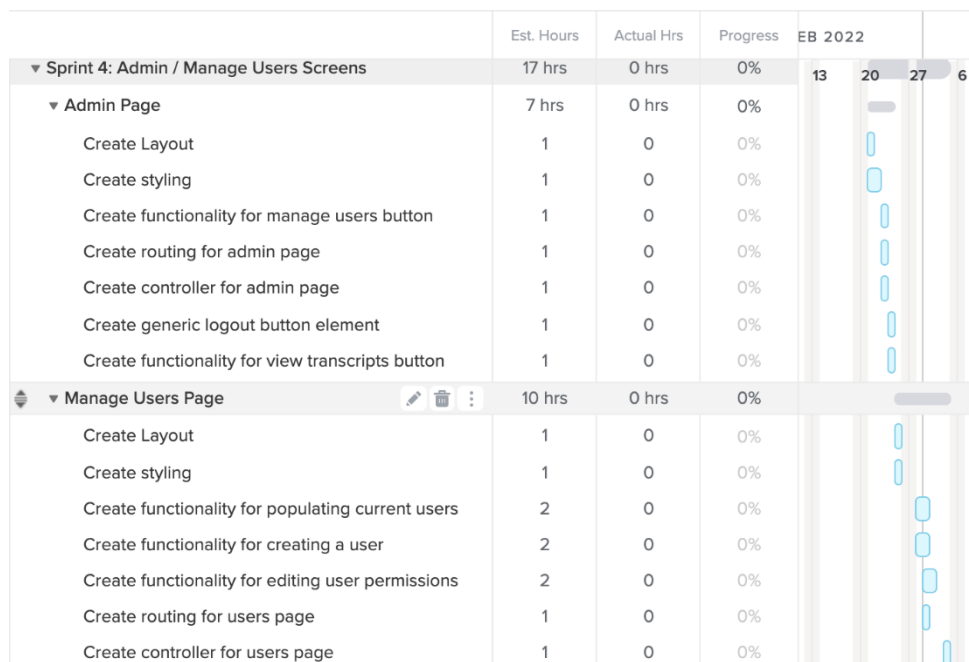


Figure 33: Screenshot of the Gantt chart schedule for Sprint 4.

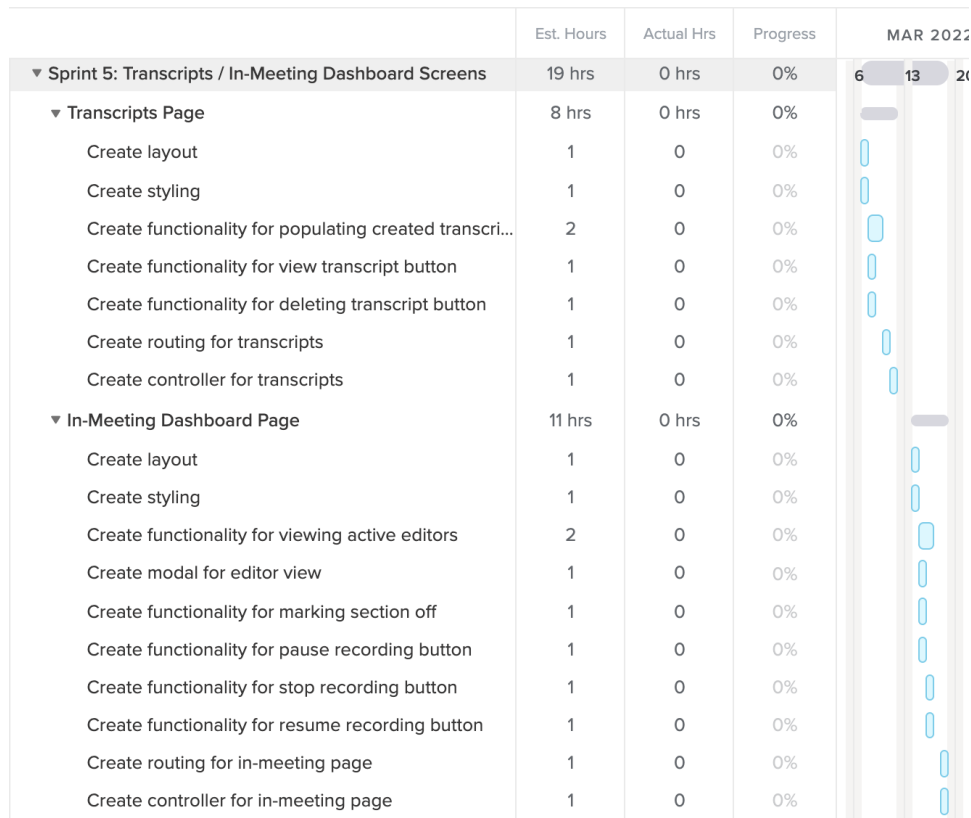


Figure 34: Screenshot of the Gantt chart schedule for Sprint 5.

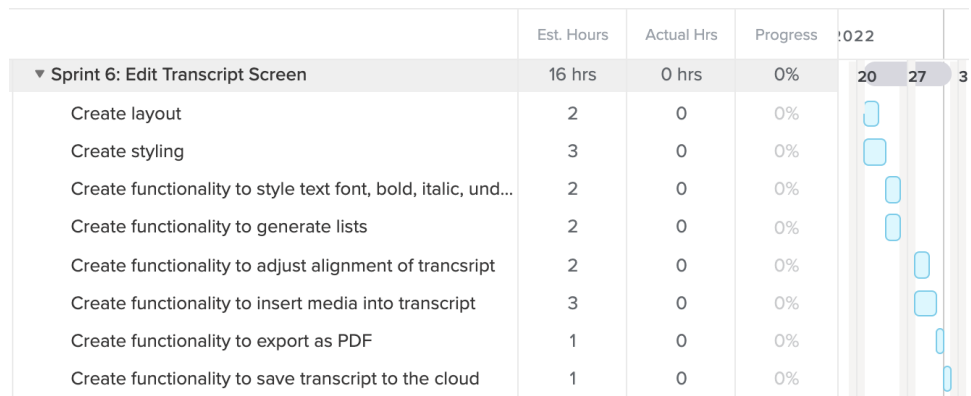


Figure 35: Screenshot of the Gantt chart schedule for Sprint 6.

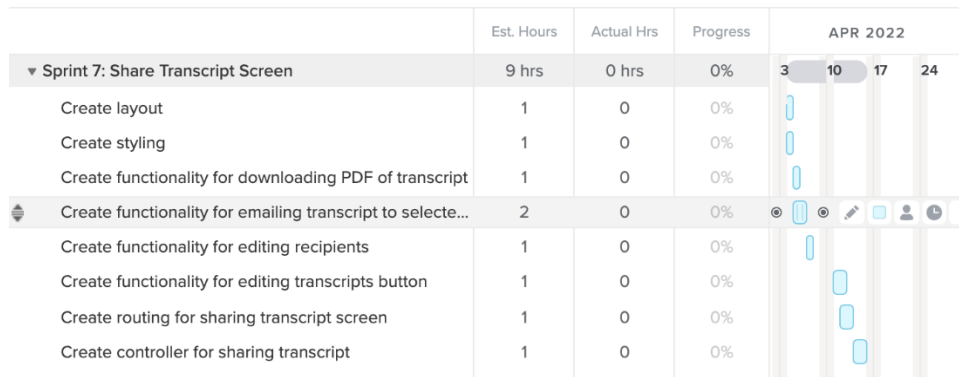


Figure 36: Screenshot of the Gantt chart schedule for Sprint 7.

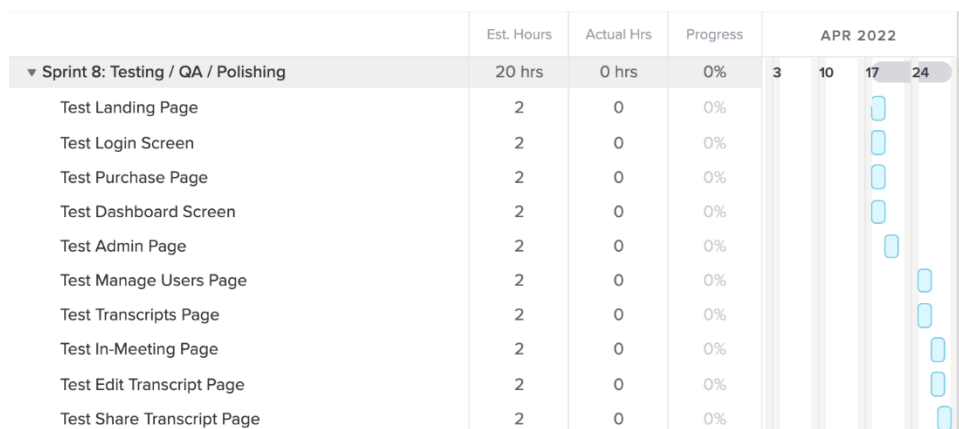


Figure 37: Screenshot of the Gantt chart schedule for Sprint 8.

4.3 Gantt Chart Source

Source: [Link to Team Gantt.](#)

5 Kanban Board

For planning individual sprints, we will use Trello to keep track of upcoming sprints and tasks. We will also assign tasks to different members through Trello and prioritize tasks by seeing how long each task takes.

For every task in the Sprint Backlog and Next Sprint, they are assigned Story Points which gives the team members an idea of how long the task will take. 1 Story Point covers between 0 – 3 hours, 2 Story Points is between 3 – 6 hours, and 3 Points is between 6 – 10 hours. Using Story Points helps the assigned team member to see how long the task should take, which helps them to prioritize their work.

[Link to Trello board.](#)

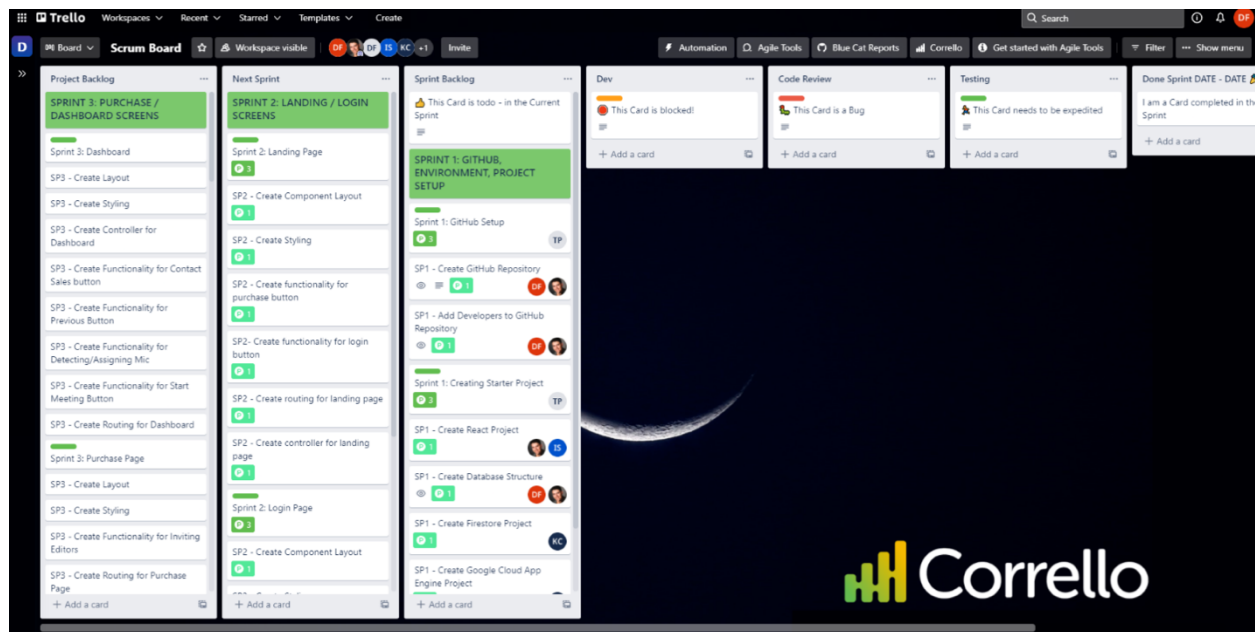


Figure 38: Screenshot of Trello Board showing the Backlog, Next Sprint, and Sprint Backlog.

6 Cost

The table below is a breakdown of the approximate cost for running our business in the first year. These numbers are subject to change since costs such as Rent & Utilities are rough estimates based on values we researched.

ITEM	TOTAL FIRST YEAR COST	INCLUDES
RAW ENGINEERING SALARIES	~\$343K	1 Sr. Level Engineering @ ~\$113K, 1 Mid-Level Engineers @ ~\$90K, 2 Jr. Level Engineers @ ~\$70K
RAW ADMIN SALARIES	\$40K	1 Admin @ ~\$40K
BENEFITS PACKAGES	\$0	We are cutting benefits
EQUIPMENT	~\$11.2K	5 Employees w/ PCs, monitors, and an IDE @ ~\$2.23K
PRODUCTIVITY SOFTWARE	~\$2.1K	5 Employees w/ Trello, Slack, and GitHub (private repos) @ ~\$410
SERVER COSTS	\$7.3K	10 Servers @ \$732
RENT & UTILITIES (IF HAVE SITE)	~\$76K	5 Employees @ ~\$6.3K monthly
MARKETING EXPENSES	\$500K	5 Viral Marketing Campaigns @ ~\$100K
TOTAL:	~\$980K	

Table 8: A rough estimate of first-year costs associated with the creation and release of an MVP.

The average pay for a senior level engineer, mid-level, and entry level engineer is about \$113,000, \$90,000, and \$70,000 respectively. Assuming we have one senior level engineer, one mid-level engineers,

and two entry level engineers, that puts the total salary expenditure up to about \$343,000 per year on engineers alone.

The average pay for an admin on a software project is about \$40,000. Assuming there's only 1 admin, that adds up to a total expenditure of about \$40,000 annually.

For benefits packages, assuming we have a team of approximately thirteen employees - including the five admins - and with the average cost of benefits per year being roughly \$21,000 per year per employee, the total would be roughly \$273,000.

For equipment, Microsoft Visual studio is roughly \$15 monthly per seat, the cost of a workstation for each user is roughly \$700~\$2000, and a decent monitor costs roughly \$300~\$400. In total, this brings the sum to about \$29,000 for the first year (the workstations and monitors would not be repeat purchases).

For productivity software, Trello Enterprise is \$10 per user per month, Slack is \$12.50 per user per month, and GitHub Enterprise is \$9 per user per month, giving us a total of about \$5,330 monthly for productivity software for each of the thirteen employees.

Servers cost \$731.94 per year¹ and assuming we have about ten servers (at the most), that is about \$7,300.

For utilities and rent, since the rent for an average office space in California is roughly \$42.18 per square foot and the average office building is ~150 square feet (on the smaller size, since we only have 5 employees). That puts the total at about \$6,327 monthly and \$75,924 annually to rent out an office space for 5 employees in California.

For our marketing expenses, we planned to launch viral marketing campaigns. Viral marketing campaigns can cost anywhere from \$100,000 to a few million dollars. Assuming we spend roughly \$100,000 on five viral marketing campaigns yearly, that would bring the total to about \$500,000.

7 Testing

This section details how the Jot-Down team will test new features. This includes both a broad overview of the development workflow per new feature and the responsibilities of each developer regarding testing.

GitHub's branching feature will be used extensively in the development of Jot-Down, described in-depth in section 9. In brief, stable releases will be pushed to a Production Branch. While implementing new features, the Jot-Down development team will pull and push to a parallel branch, the Development Branch, as to not introduce service-breaking bugs into the Production Branch. When a developer starts a new feature, he will create his own branch and make changes to that branch and will create a push request once his feature is complete.

In order to increase efficiency and allow all developers to work in parallel with each other, developers will perform their own testing on their designated features. All testing is expected to be done before pushing a feature to the development branch.

As the Jot-Down application is implemented with React, developers will perform testing using Jest and the JavaScript Testing Library. These allow the running of both component and integration tests, and where necessary, will also allow mocking.

8 Maintenance

The following is regarding Jot-Down's strategy for handling maintenance activities for the live application – namely, our strategy for bug reporting and handling.

For end-user reports, Jot-Down will be having a designated email which users will send their reports to regarding any bugs they run in to. This email will be monitored by the Jot-Down team who would begin testing the code, as described in sections 7 and 9, in search of this bug. Once a resolution has been found, the team would run tests to ensure it doesn't happen again.

Jot-Down will be monitoring dependencies for potential bugs and/or security threats by using GitHub security features. GitHub security feature allows us to see any security issues in our pull request from GitHub, preventing new vulnerabilities from making it into our code. GitHub security feature lets us easily see what dependencies have been changed in a pull request from GitHub and it automatically monitors your dependencies for known vulnerabilities and gives us suggested fixes. GitHub security feature also watches our repositories and notifies us of any security risks.

9 Source Code Management & Git Repository

In the following sections, we'll detail our source code management strategy, as well as the process as to how code will be reviewed, tested, and approved or denied. The same process will occur whether the code is pertaining to a bug fix or a feature request for the application.

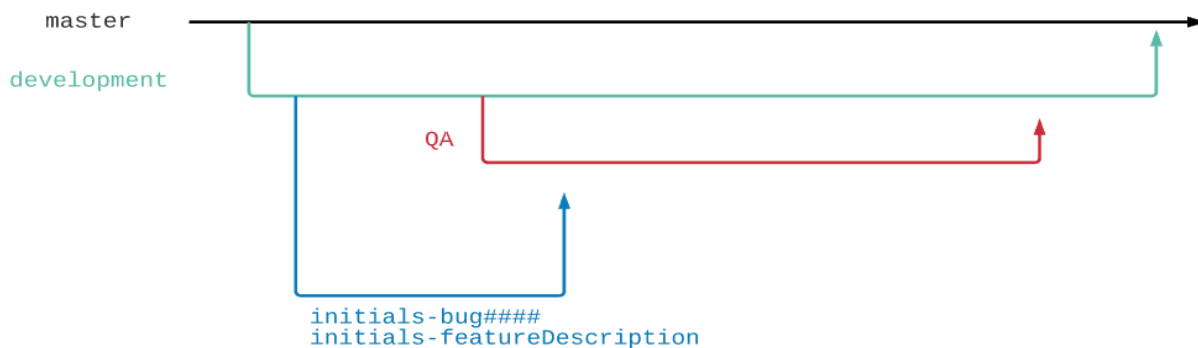


Figure 39: An outline of the current branch structure and naming convention for Jot-Down.

There is a single master branch which contains the code for the current live production build of the application. From this branch, a development branch diverges from the master branch to serve as a staging point for new additions. This branch will act as a preview of how the build in the master branch will operate with the changes, allowing for testing in a non-live environment. From the development branch a developer, using our standard naming convention for branches, will branch out from the development branch to address a single bug fix or feature request. Our standard naming convention for branches is initials, followed by either the bug number provided from the team's ticket board, or a brief description of the feature the developer is implementing.

Once a developer is done on a branch, the developer will merge their branch to a quality assurance, or “QA”, branch that is a copy of the development branch. From here, another developer will perform quality assurance testing on their changes. After quality assurance testing is performed, the developer will return the branch to the person who submitted said branch, along with a list of things to fix for completion. Once the QA branch meets all requirements, the developer that performed the QA testing will then merge the QA branch with the development branch.

Upon successful merge of these branches, a dedicated developer will run suite of tests on the development branch to ensure that no area of the product had been negatively affected by the recent additions. If there is a conflict, the developer will return the branch to the original developer. However, if there is no conflict at the end of the sprint, then the development branch will be merged into the master branch and sent to production.

GitHub Repository Link: <https://github.com/tomas13perez/Jot-Down>

Jot-Down

Industry Trends

1 Software Development Process

1.1 Industry Summary

After listening to the various speakers throughout the semester, we've learned that company's software development process can vary. While most companies tend to follow the same development process, such as the agile method, each company seems to have a slight variation in the approach that tailors to its own needs for the company. This process is different when I listened to speakers from Facebook and ZyBooks because Facebook seems to have a clearer vision of what their process is in terms of projects. Still, with ZyBooks, they have a heavy influence from investors and clients that can affect their development process if something comes up that takes precedence, meaning that ZyBooks developers need to be a bit more flexible in their process.

1.2 Application to Project

Seeing the many different variations between companies and their development process makes it more apparent that a significant amount of time needs to be made on identifying the key factors that will affect our own company's development process. Being aware of who the investors are as well as the clients' or customers' needs will be crucial in determining the structure and flexibility of the development team.

2 Feature Selection

2.1 Industry Summary

Feature selection is another aspect of the development process that can vary drastically between companies. Again, using ZyBooks as an example, they had heavy influence from investors and clients on what features should be added to the software. Similarly, with Jordan and his startup, the features that were chosen in the beginning were heavily focused on the front-end side of the software in order to impress investors to gain traction. Once Jordan had obtained a significant number of investors and support, he transitioned into a more traditional feature selection process.

2.2 Application to Project

For beginning in a startup company, the feature selection process would closely resemble the same process Jordan followed in order to gain traction and attention. However, in an already established company, the feature process would most likely resemble a more structured process, like the way Facebook, PayPal, and Google uses.

3 Team Roles and Organizational Structure

3.1 Industry Summary

Depending on the size of the company, Team Roles and Organization Structure can go a few ways. In startup companies that are not necessarily software companies, as we learned from Josh Park, the company could hire a software engineer, and that engineer would answer directly to the company director and be responsible for any software development needed. On software startups, the whole company is one team, that is working and communicating with each other directly, and so the only assigned team role is the director. In larger companies, according to Michael Yoon at Facebook, engineers are divided into teams usually consisting of multiple junior engineers, one or two senior engineers, and a project manager. The project manager makes sure that things are getting done, and may or may not help in programming.

Optionally, some teams also have a product engineer whose role is to manage what features are being implemented and to review whether those features are in line with the company vision.

3.2 Application to Project

As a startup, we would structure our team so that we have multiple engineers and two of us would take the roles of project manager and product engineer, respectively. This would allow the engineers to work on getting the product up and running and the product engineer can focus on keeping the product on track for our final output and on keeping the stakeholders up to date.

4 Testing & Maintenance

4.1 Industry Summary

Testing and debugging are imperative to the successful development of a product. Every aspect of a product, from hardware to software is stress-tested at Intel, according to Joseph Tarango, so that the best possible product can be shipped. Any respectable company will test their products before they sent them out so that they can ensure that their customers will receive a product without defect. Especially in software, bug or issues that are let out into the field are much more costly to fix than the issues caught during the development phase. An interesting point brought up by Scott Sirowy from PeopleAI is that developers didn't test their own products back in the early 2010s when he began working as an engineer. Nico Chera brought up three types of testing which take place at Google: unit testing (testing individual components), black box testing (seeing what output some input produces), and open-box testing (where a desired response is elicited).

4.2 Application to Project

At our company, we would have a team of developers who work on testing and debugging. Unit testing will be critical to ensuring a competently developed final product and black box testing would probably be performed after unit testing is complete, to ensure that every part of our software functions as intended together. Maintenance would probably be performed by the same team which performs testing and debugging on Jot-Down. We, as developers of Jot-Down, want to ensure that we put out the best possible product that we can. Naturally, if we want to release the best possible product that we can, we must perform maintenance and testing on Jot-Down for the good of our relationship with our customers and stakeholders.

5 Communication

5.1 Industry Summary

The way in which communication is facilitated among companies depends on several factors such as the size of the company and the makeup of the development teams. Most of them claimed that their transition to a remote style of working during the beginning of the COVID pandemic was relatively smooth, with Zoom, WebEx, and Slack serving as an intermediary for communication between teammates or other employees. For example, Scott Sirowy, from PeopleAI, mentioned Zoom, Trello, Slack, and Oculus Quest (an experimental VR conferencing software) as methods of communication among employees at the company. Jefferey McDaniel, a developer from Door Dash, also mentioned that Slack is their primary communication tool. Many speakers who discussed their communication tools mentioned Slack as a communication facilitator at their company.

5.2 Application to Project

Given the relatively small size of our company, we wouldn't have to worry too much about a lack of direct communication between developers and other sections of our company. We'd have no issue abiding by the current industry standard of using Slack for communication between employees, since it could provide us with an avenue through which we could still communicate if we were to be forced to work remotely again for whatever reason. Ultimately, we'd keep our company relatively small so that communication would never be stifled (as it seemed to be at JPL), and we'd use Slack as a tool through which immediate and/or private communication could be facilitated.