Southwest Innovation Research Lab (SwIRL) Skehdule - Sprint 1 Plan

Team Roles:

We have elected Glen Browne as scrum master and Eric McGonagle as product owner.

Weekly Customer Meeting:

Our weekly meeting is set for Thursdays at 9:00 via Zoom with Jonathan Tan of SwIRL.

First Meeting Summary:

We met with Jon for the first time and introduced the group and discussed the general scope of the project, such as the timeline, weekly meetings, and team roles. Jon had some recommendations with regards to the project as a whole such as assigning roles (i.e. frontend, backend), which the previous groups that he worked with lacked. He suggested a comprehensive review of the previous code so everyone is familiar with where to pick up (assuming we continue with the legacy code). Something to note is his idea of starting the project from scratch, which will be discussed with the group as well as with Professor Ritchey. We got four user stories, which we describe in the "User Stories" section of this document. Finally, we agreed to have weekly meetings on Thursdays at 9:00 AM CST via Zoom.

Customer Need Summary:

The main customer need is to create an event invitation system that the employees of SwIRL can use to invite people of all kinds to events. The invitees include SwIRL employees, other researchers, guest speakers, friends and community members. SwIRL needs to be able to create two types of event invitations. The first type of invitation is an "event" invitation, which is an invitation to a one-time event that fills up to capacity on a first-come, first-serve basis. The second type of invitation, which sets Skehdule apart from competitors, is a "series" invitation. The series invitation is sent to a group of people who can choose which event in the series they will attend. Once a series attendee is confirmed, the spot is filled and the remaining invitees can only choose to attend one of the remaining events in the series. This system is geared toward a guest-speaker type of scenario, where each spot needs to be filled by exactly one invitee.

The application will meet these needs by providing the customers (initially the employees of SwIRL) with the ability to create events and series in Skehdule, to send invitations to individuals or lists of individuals, and to monitor event and invitation statuses, including confirmations or declinations. Users (both hosts and guests) will be able to create accounts and add events to their calendars in order to effectively monitor these statuses. The primary stakeholders of Skehdule will be SwIRL and our development team. The secondary stakeholders of Skehdule will be the invitees.

User Stories:

<u>User Story 1</u>:

Feature: Send an event invitation to everyone or mass of people
As an event host
So that I can fill x amount of slots for an event with those who might be interested
I want to send an invitation to a mass amount of people

Scenario: Send invite
Given I am on home page for an event
When I follow “Send General Invitation”
Then I should be on the event details page
When I fill out the details of the event with 1000 available tickets/slots
And I specify the date and time
And I press “Confirm Invitation and Send”
Then I should be on a page with a confirmation message regarding the invitation
And I can view the number of those who have accepted the invitation
And I can modify the invitation time constraint per event invitation

<u>User Story 2</u>:

Feature: Send an event invitation to a priority list
As an event host
So that I can have a guest speaker come to an event from a list of those I am interested in

I want to go down a list of people I want to invite only moving on to the next after previous declines invitation

Scenario: Send invite to priority list of guests
Given I am on home page for an event as an event host
When I follow “Send Priority Invitation”
Then I should be on the event details page with option to customize invitation with details
When I fill out the details with the time and date
And the special message to the guest
And I add a list of guests that I want to send in order
And I press “Confirm Priority Invitation and Send”
Then I should be directed to a confirmation page that displays who is at the top of the list
And I can view if the person at the top of the queue rejects thus sending invitation to next priority
And I can see how much time is left for the current invitation starting at 24 hours

User Story 3:

Feature: sign in to view current events
As an event host
So that I can view all of the current events I am hosting
I want to view the details of a given event as well like the number of those accepted and total number of invitations sent out

Scenario: Log in and view event details
Given that I am on the home page
When I follow “Event Details” regarding a specific event
Then I should be on the event home page
And I can see the details of the event like the time and the date
And I can see how many invitations are sent and how many have been accepted
And when I go back to the home page and click on another event
Then I should be able to view the same information about this additional event

User Story 4:

Feature: Add event to Google Calendar
As an event host or user
So that I can visually see the event on my Google Calendar

I want the events to sync up with my Google Calendar

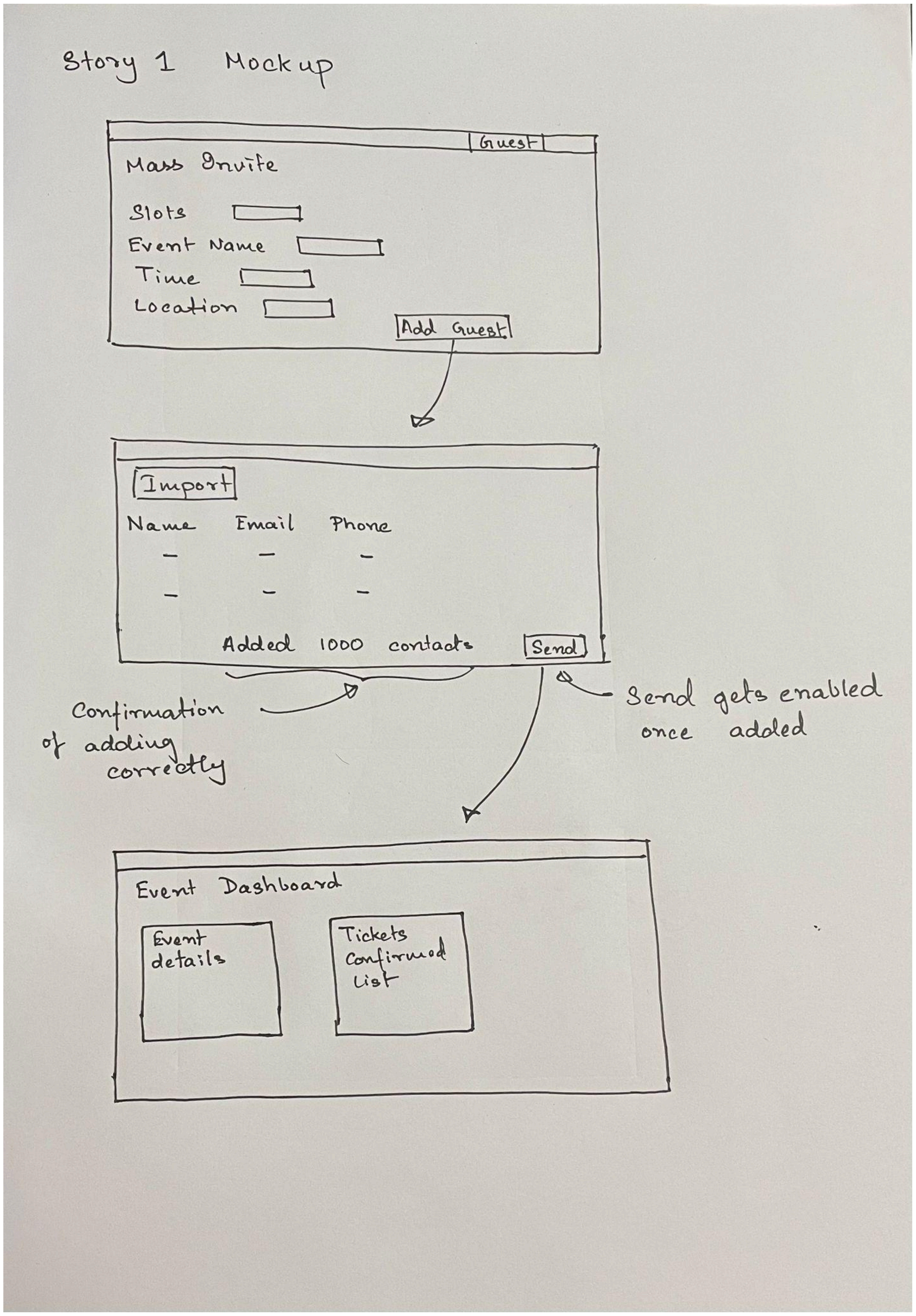
Scenario: Sync event to Google Calendar
Given that I am logged in and have signed up for an event
When I confirm my RSVP to an event
Then I should have the option to add/sync event with Google Calendar
And I can view other events that I am signed up for on my Google Calendar
And when I click the details on that event
Then it will direct me to the event details page assuming that I am still logged in

Sprint Backlog:

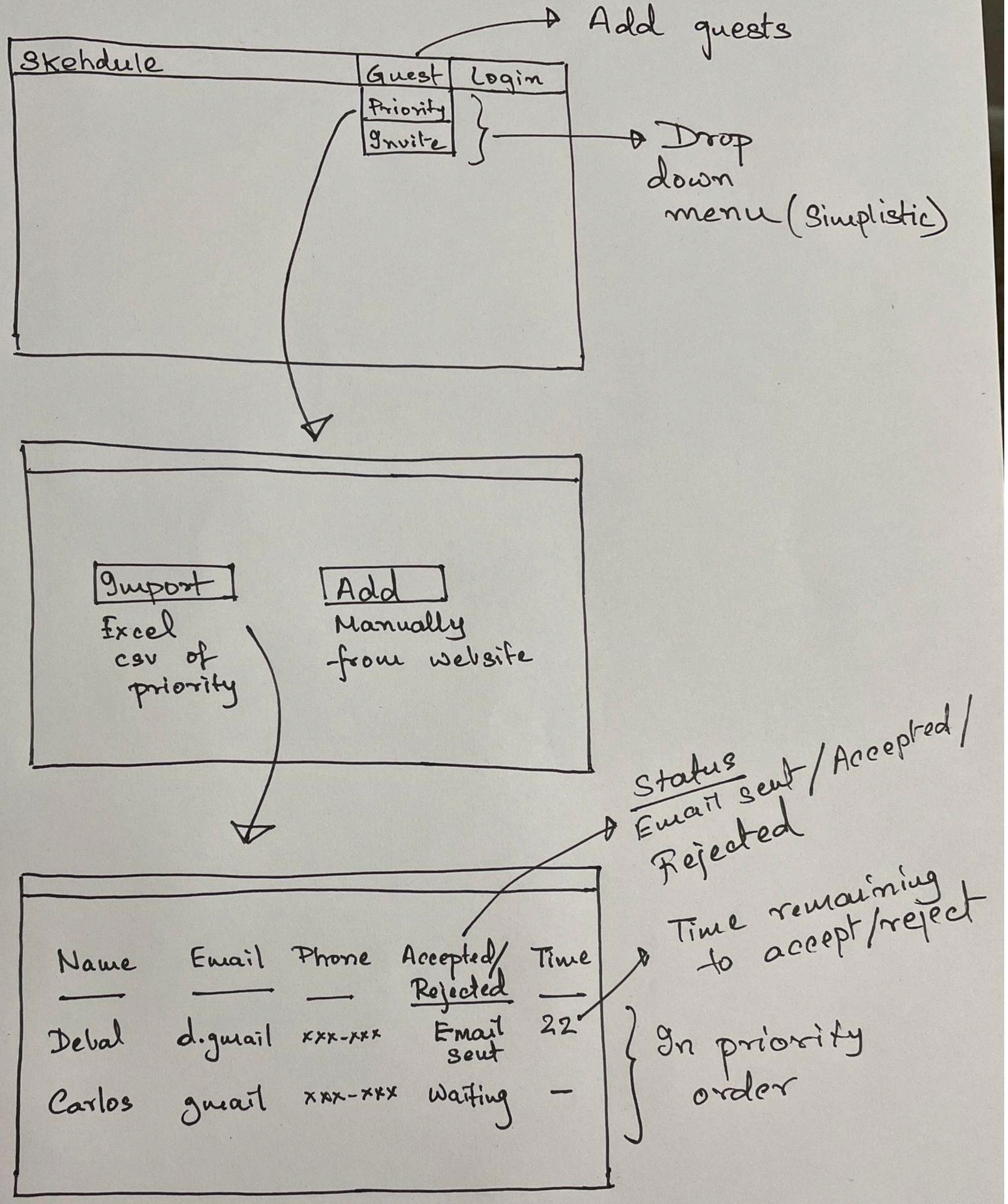
The goal of the Sprint 1 is not only to make progress satisfying the customer's functional requirements for the project, but to make a decision on whether the group will start from scratch or continue with the legacy code. In our meeting with Jonathan Tan of SwIRL, he recommended that we start from scratch and Professor Ritchey OKed the notion. In one of our group scrum meetings, we discussed the idea and decided that the legacy project might have some valuable aspects, such as the front-end design, that would be unwise to throw out. We gave ourselves a hard deadline of the end of the first sprint to make a decision on whether to continue with the legacy code or to start from scratch.

The sprint is two weeks long, and we have four user stories to work on. We assigned points to the stories assuming 3-hours of work equaling 1 point. User story 1 (creating a method to send invitations) is a 12 point (36 hour) story and is assigned to Eric. User story 2 (creating a prioritization system for invitations) is a 9 point (27 hour) story and is assigned to Debal. User story 3 (creating a user login system) is a 12 point (36 hour) story and is assigned to Pankaj. User story 4 (allowing a user to add events to their calendar) is a 6 point (18 hour) story and is assigned to Prakar.

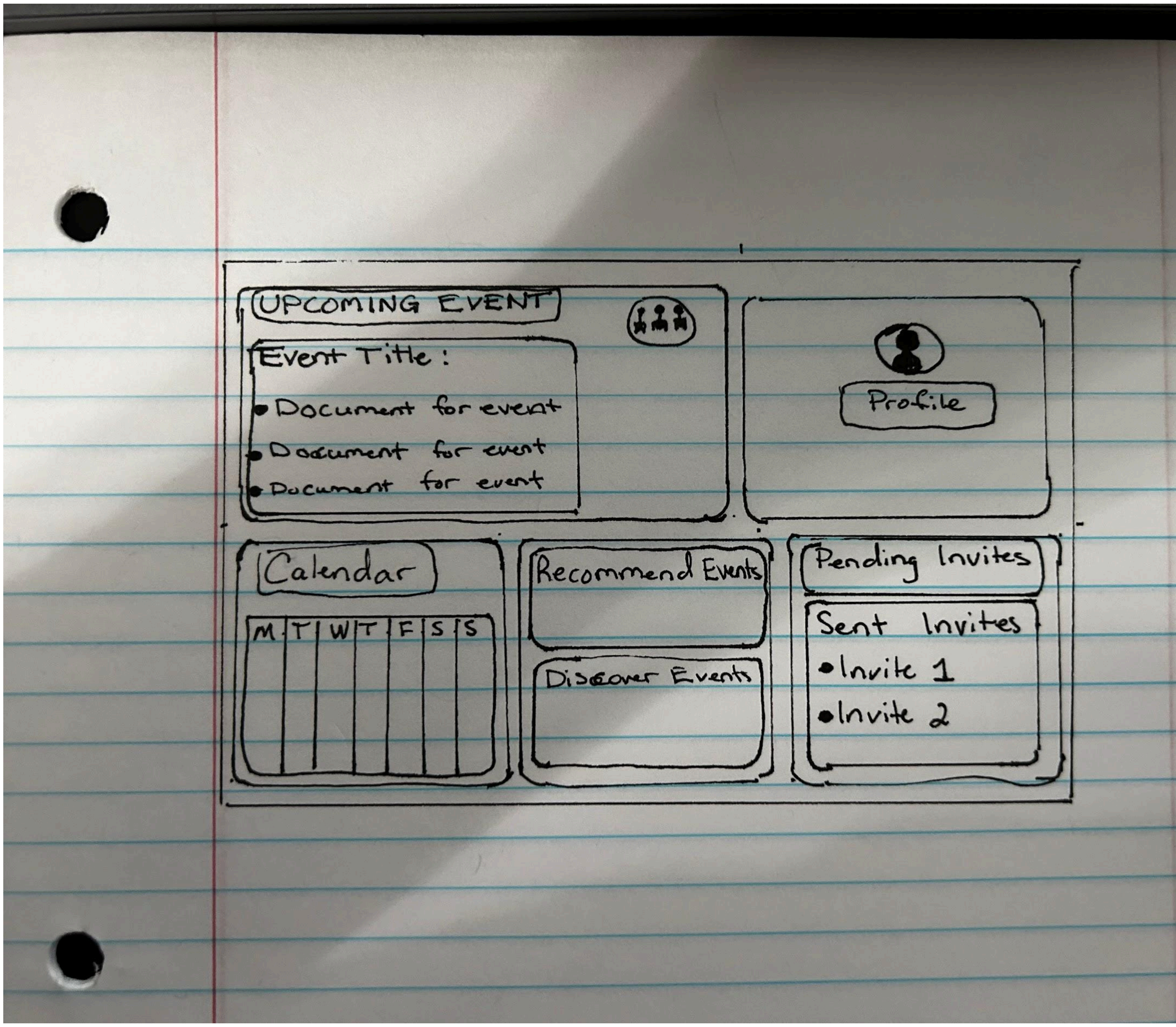
User Interface:



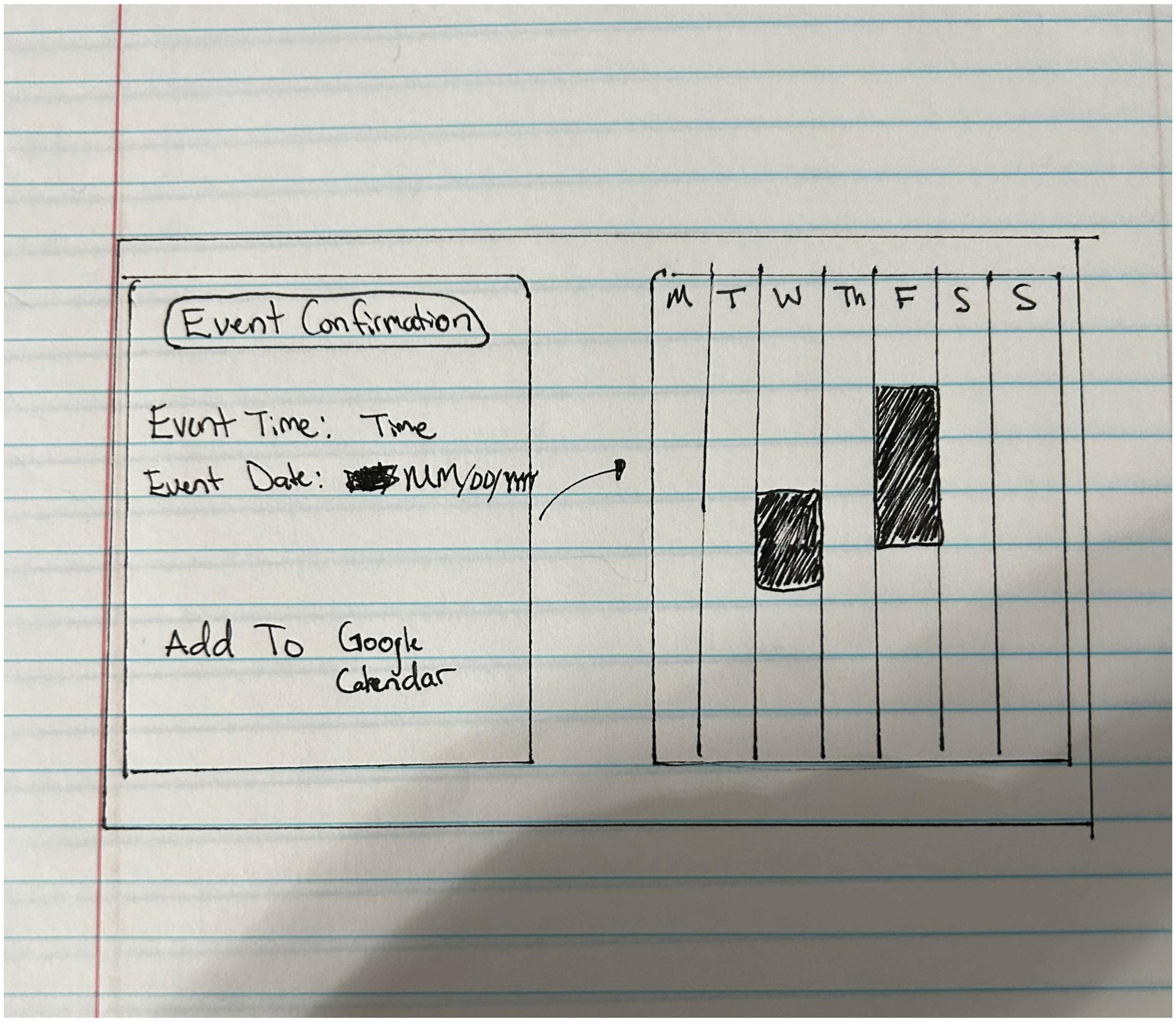
Story 2 lo-fi mockup



User story 3:


UPCOMING EVENT
Event Title :
• Document for event
• Document for event
• Document for event
Profile
Calendar
M T W T F S S
Recommend Events
Discover Events
Pending Invites
Sent Invites
• Invite 1
• Invite 2

User story 4:



Links

Deployed App - https://skhedule-9d55cf93012e.herokuapp.com/home
GitHub Repo - https://github.com/gdbrowne85/SwIRL-CSCE-606
Project Management Page - https://www.pivotaltracker.com/n/projects/2690139
Slack channel - https://swirlskehdule.slack.com/

Legacy Plan:

Since our project is a legacy project, we need to devise a plan to learn and improve the prior code. We think that the best way to do this is to deploy the legacy project and try to use every feature that has been implemented so far. When using the product, we will take notes about what we like or don't like about the functionality and design, and also note any bugs. After testing out every feature of the deployed legacy project, we will go through code and identify which blocks are responsible for which behaviors. We believe this method would be much faster than just looking at the code first and trying to discern its function. We will mark all the code with comments denoting what functionality it controls, and add our notes that we had previously taken to the comments. Then, if there is still code that is unaccounted for, such as incomplete code that has not been deployed yet, we will attempt to learn what the purpose of it is, and decide if we want to try to complete it or deploy it.

Since the paradigm used for this legacy project was likely test-driven development, it is equally likely that there are tests that were written but that never passed. We think that if we look at all the tests that were written, this will give us big clues as to what features were in development but that were never completed. These tests would point us to sections of code that were never deployed. If strict test-first development methodology was used, there shouldn't be any code in the product code that wasn't accounted for in the tests.

With all the code accounted for, we will be ready to make improvements. With the code annotated, we will devise a way to improve it without breaking the things we don't want to change. We can achieve this through unit testing, which we should run frequently. We will do our best to isolate properly working code by objectifying it if it was previously written procedurally. If we have broken something, we can use version control to revert back to the previously working version. We must make sure to create a new branch every time we work on new functionality. We will also try to take advantage of the code that wasn't deployed by trying to make it pass the pre-written tests, if we decide that functionality is something we want in our product.

One of the user stories we noted for Sprint 1 is a replacement of a non-functional login system from the legacy project. We will have to locate that code and decide to correct/complete it or replace it, depending on our assessment of its nearness to the functionality the customer needs.