SwIRL Final Report

Team Members:

Eric McGonagle
Glen Browne
Debal Goswami
Carlos Meisel
Erhan Wang
Prakhar Singh
Pankaj Kumar Tiwari

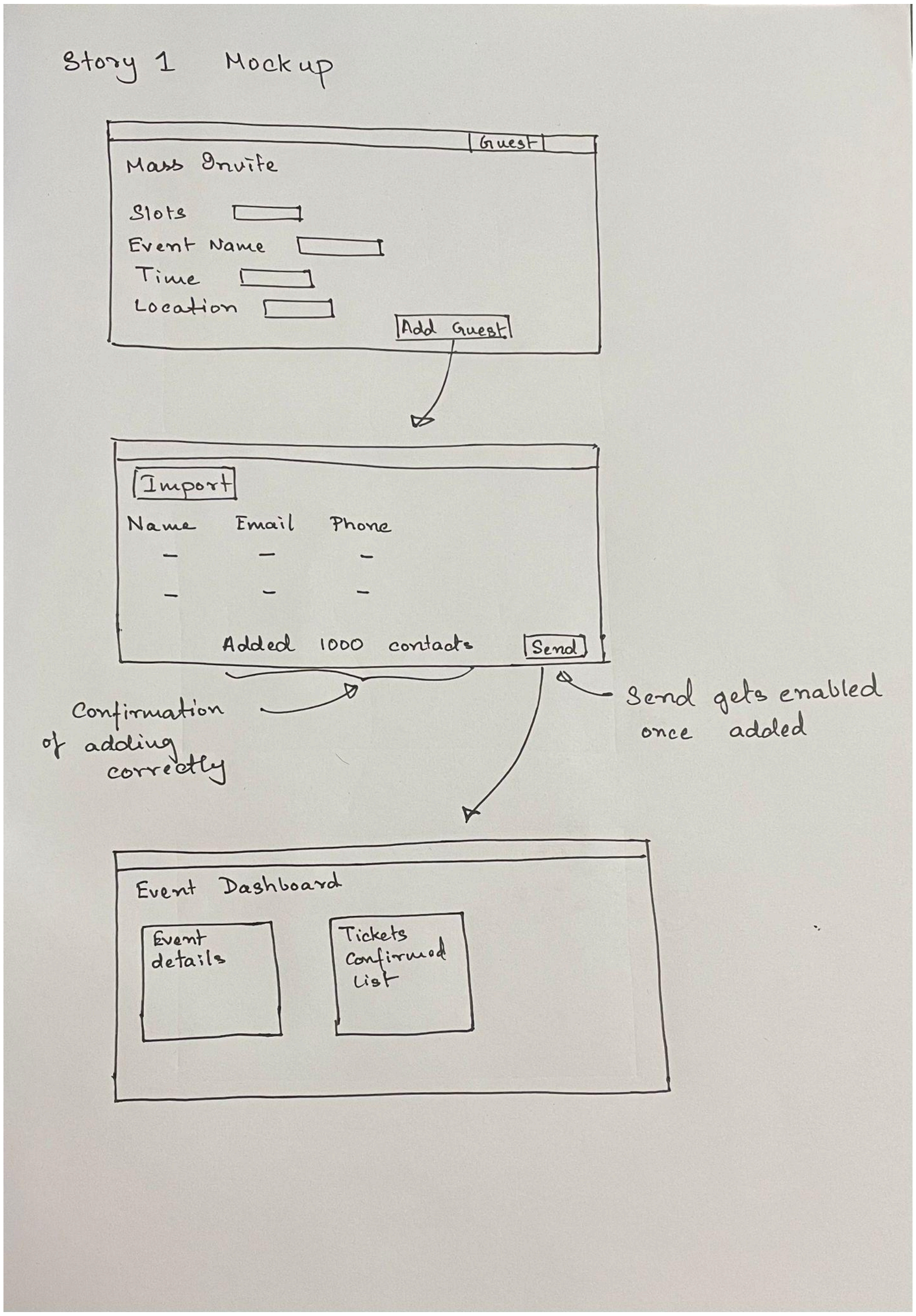
Summary

Our project is a scheduling platform primarily used for events that have specific attendance needs or capacities. An event scheduler that handles automated email invitations addresses the concern of wanting to fill out all available slots for a given event without promising more than the allotted number of specified seats. The stakeholder for our project is the Southwest Innovation Research Lab (SwIRL), who will be using it for event scheduling. We allow for the scheduling of two types of events: singular and series. Singular events are for one-time occasions where a specific number of attendees must be accommodated. An example of this would be a dinner party, where the host aims to invite guests until all available seats are occupied. Series events are for recurring events that have multiple dates and times within them. An example of this could be a lecture series hosted by universities, with multiple speakers giving talks on different occasions. Every event allows for acceptance and rejection, as well as the ability to RSVP.

We provide several other features that meet scheduling needs. For one, we allow for both singular and series events to be added to Google Calendar, allowing for increased convenience. In addition, we allow for reminder emails to be sent to guests who have not accepted the event, which is run on a timer. This allows for improved event attendance and reduced manual effort in managing events. Besides this, we include quality of life features such as account creation, invitation status indicators, and acceptance/rejection emails to the inviter.

User Stories

User Story 1:
Points: 5
Assigned to: Glen Browne
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



User Story 2:

Points: 5

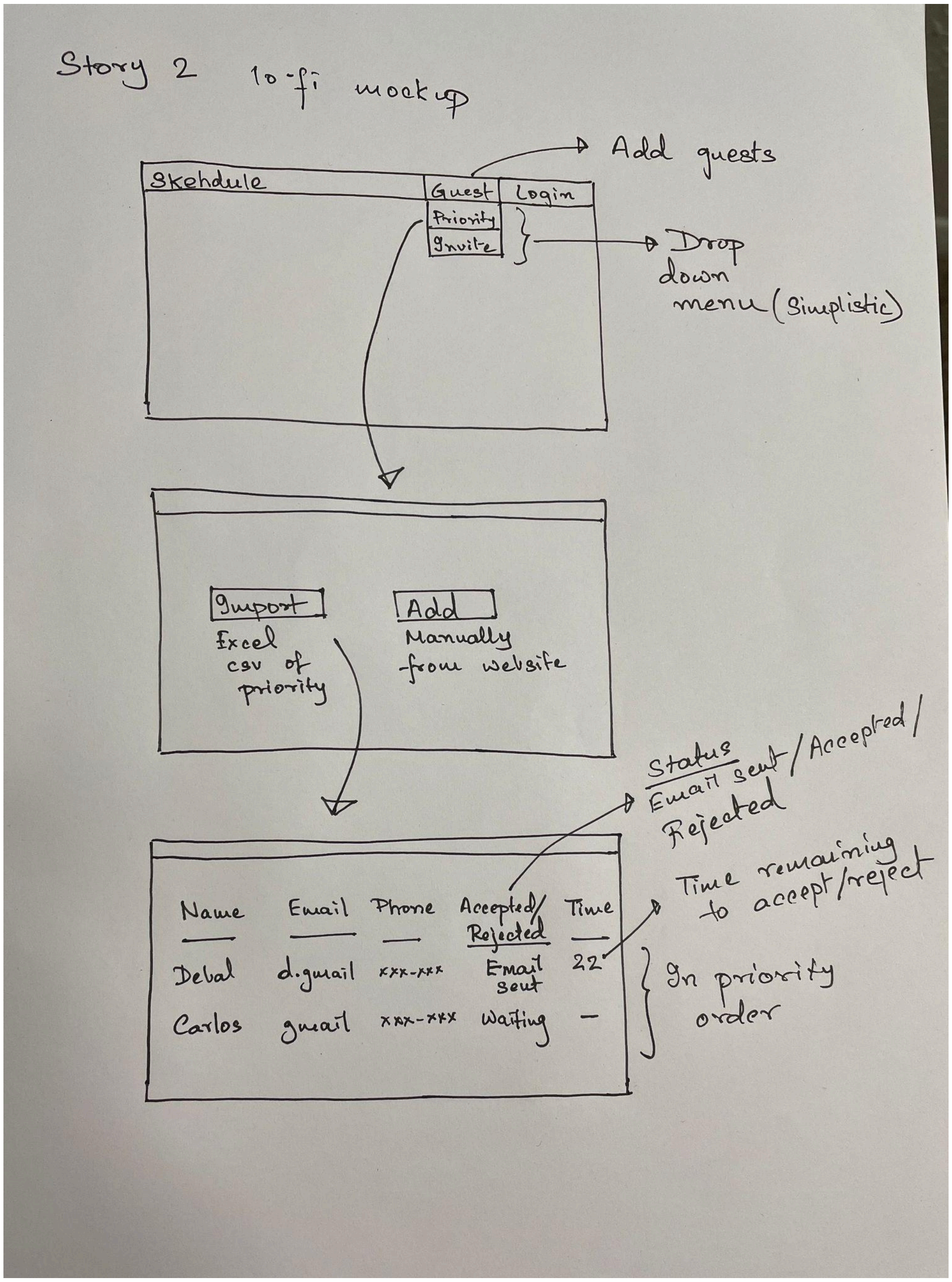
Assigned to: Pankaj Tiwari

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

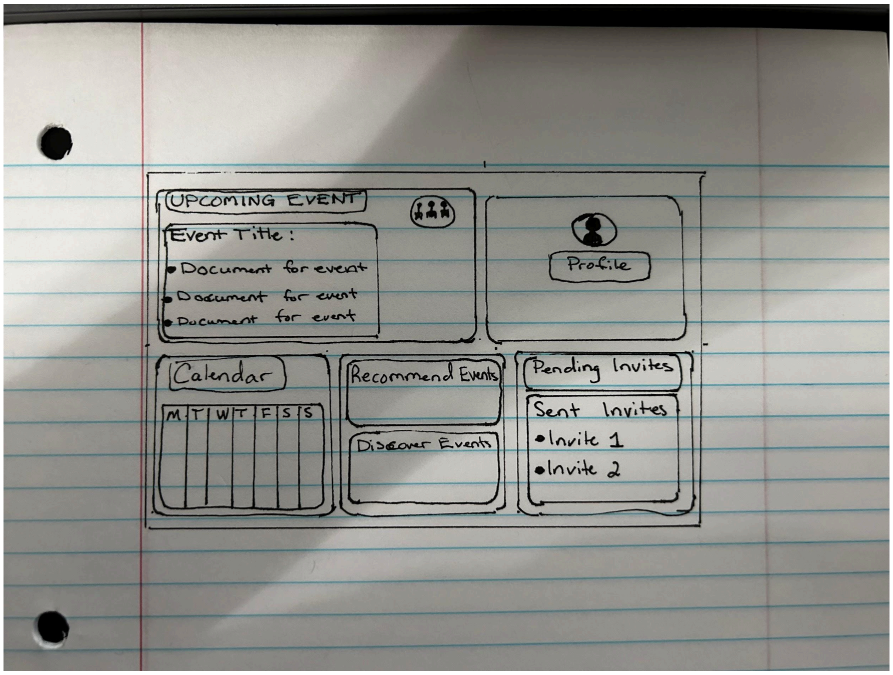
I want to go down a list of people I want to invite only moving on to the next after previous declines invitation



Name	Email	Phone	Accepted/Rejected	Time
Debal	d.gmail	xxx-xxx	Email sent	22
Carlos	gmail	xxx-xxx	Waiting	–

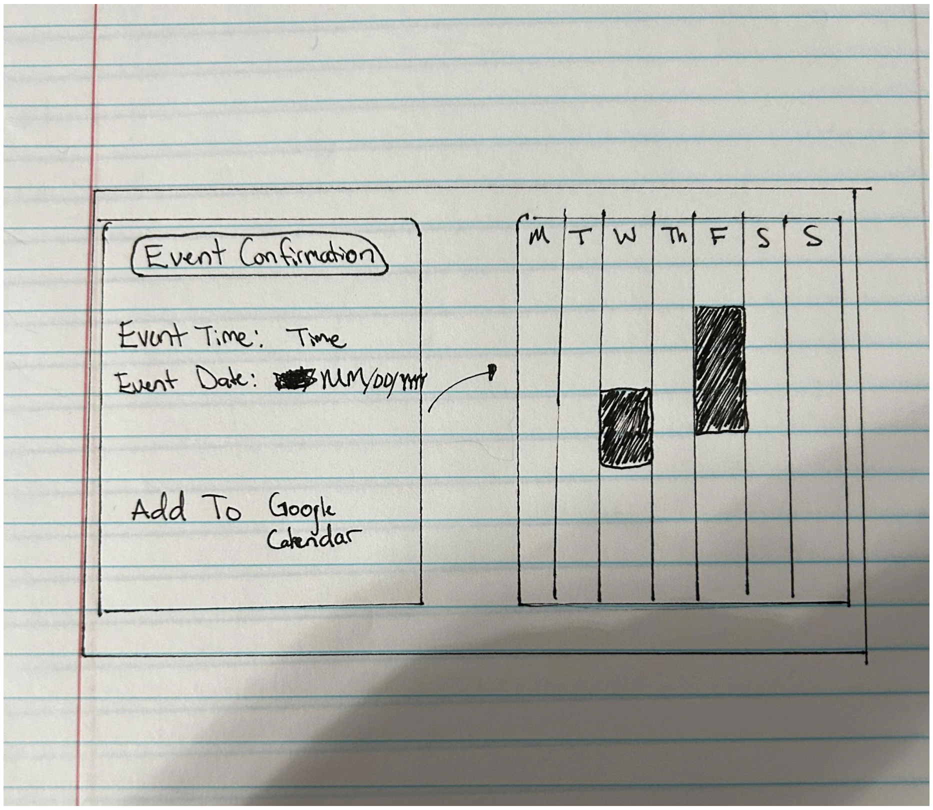
<u>User Story 3</u>:
Points: 5
Assigned to: Debal Goswami
Sign in to view current events
As an event host
So that I can view all of the current events I am hosting

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



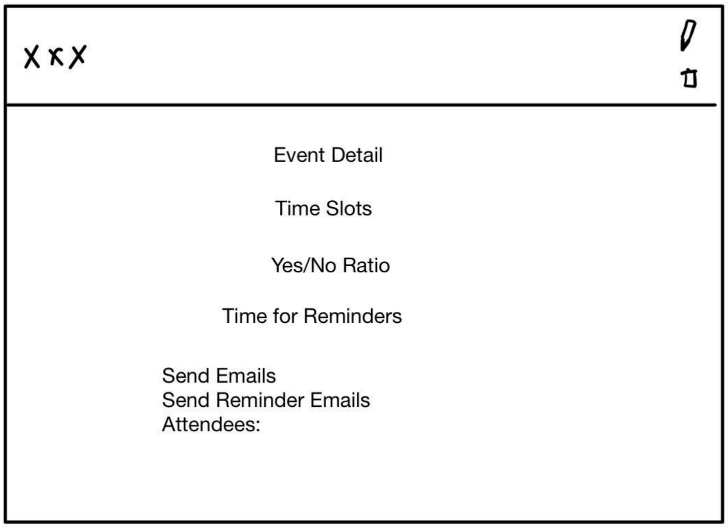
<u>User Story 4</u>: Add event to Google Calendar
Points: 3
Assigned to: Prakhar Singh

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



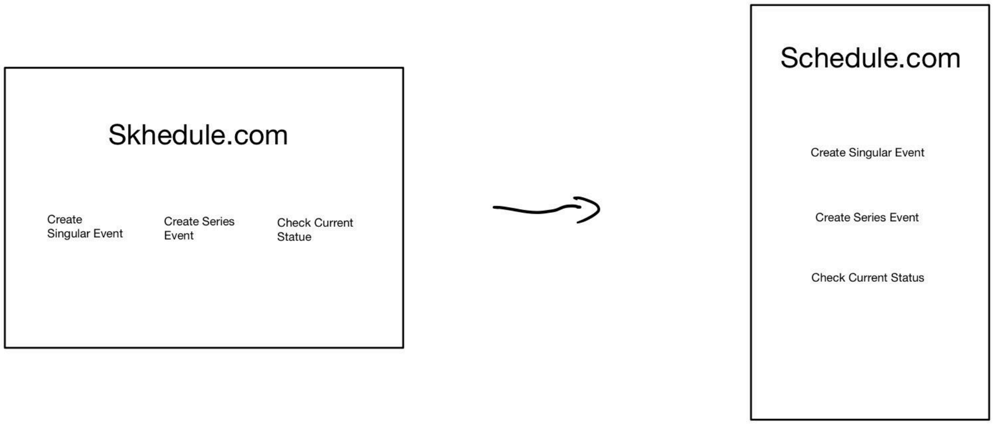
User Story 5: Viewing Countdown Timer for the Series Event
Points: 3
Assigned to: Debal Goswami

As a user, I want to add a countdown timer to the Series Event to schedule reminder emails to be sent after the timer ends.
For legacy projects, include a discussion of the process for understanding the existing code, and what refactoring/modification was performed on the code, in addition to the user stories listed above.



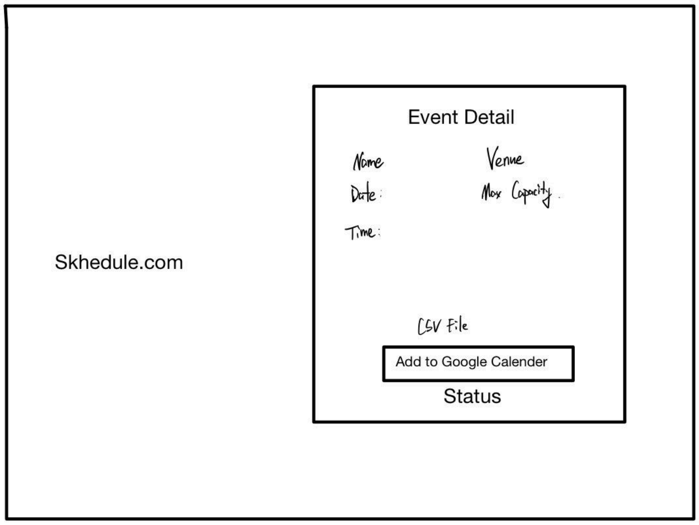
<u>User Story 6</u>: Making the website dynamic to fit across all the devices
Points: 3
Assigned to: Debal Goswami

As a user accessing the website from various devices, I want the website to adjust its layout and functionality dynamically to ensure a seamless experience on any screen size.



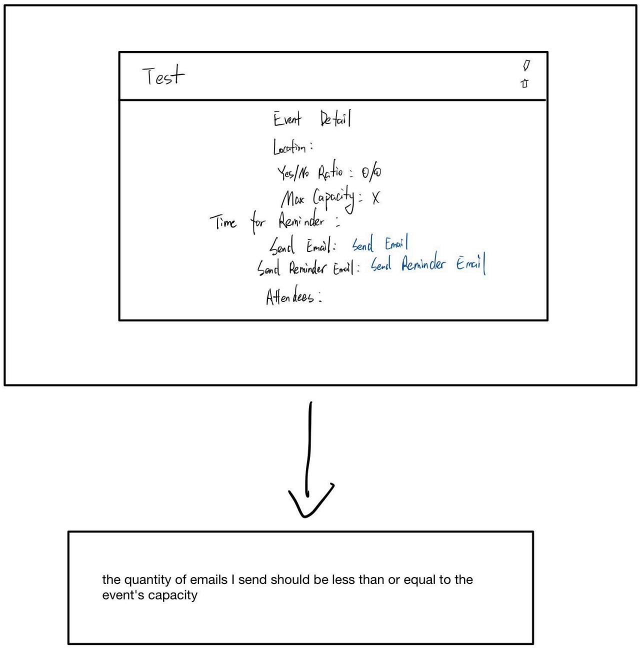
<u>User Story 7</u>: Adding Google Calendar functionality to the Series Event
Points: 2
Assigned to: Prakhar Singh

As a user, I want to add my Series Event to Google Calendar to view and manage it more easily



<u>User Story 8</u>: Send no more invitations than the maximum amount allowed.
Points: 3
Assigned to: Pankaj Kumar Tiwari

As the event planner, so that no more invitations than the event capacity are sent, I want to send out as many invitations as possible, up to the maximum amount that I determine.

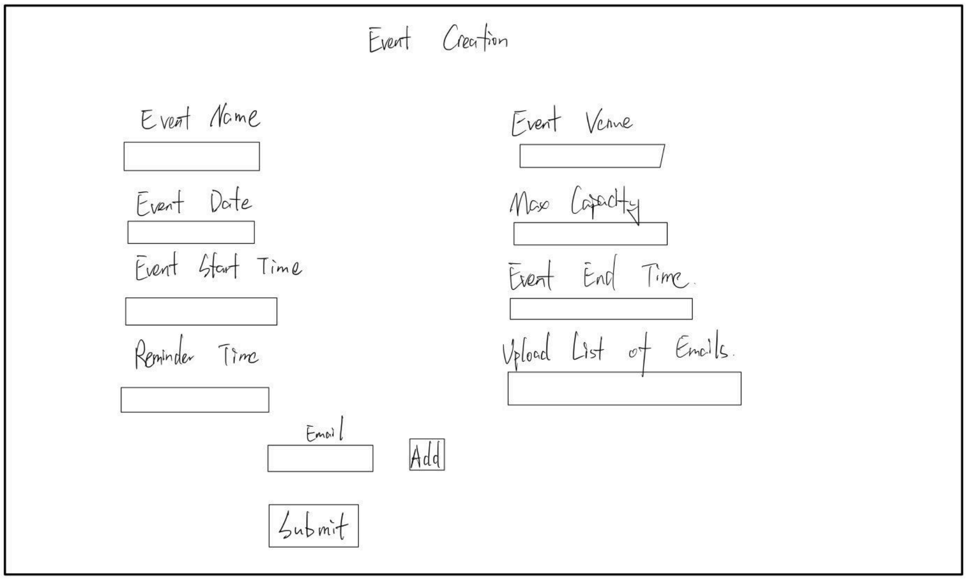


the quantity of emails I send should be less than or equal to the event's capacity

User Story 9: Add Text Box for Email

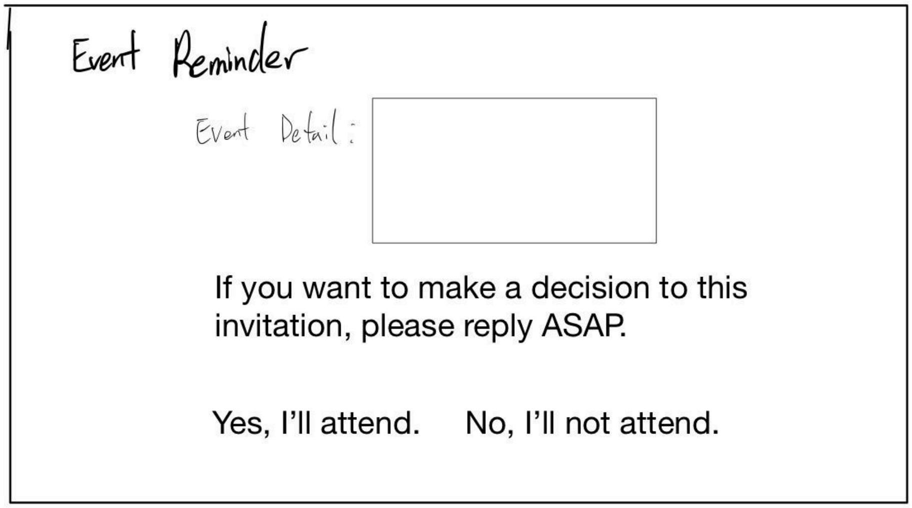
Points: 3
Assigned to: Glen Browne

As a user, so that I can simplify the process of sending invites, I want to add a text box for entering participant emails during Series Event and Single Event creation.



User Story 10: Make Reminder Email Different from Initial Email
Points: 2
Assigned to: Eric McGonagle

As a user, in order to avoid confusion among participants, I want the reminder email for the Series Event and the Single Event to be different from the initial invitation email.



Event Reminder

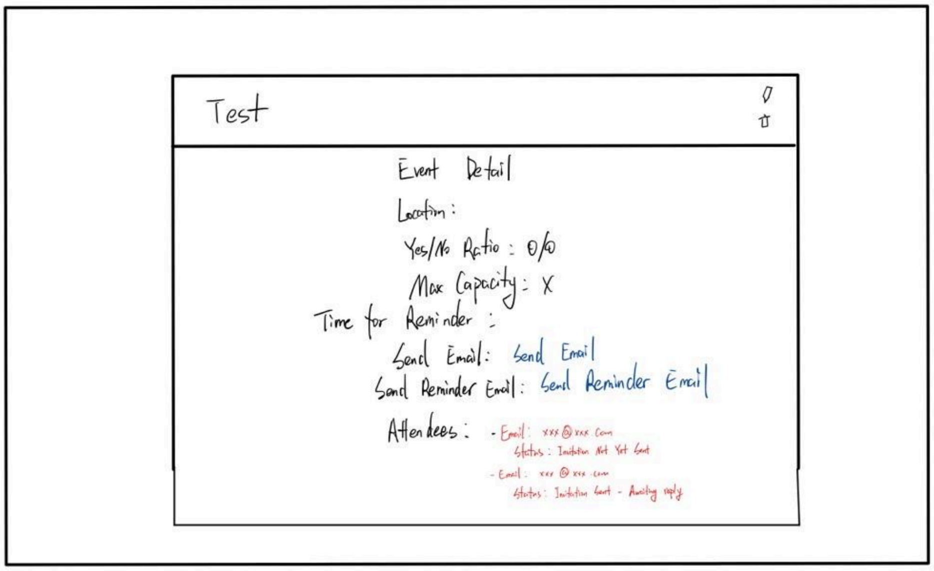
Event Detail :

If you want to make a decision to this invitation, please reply ASAP.

Yes, I'll attend. No, I'll not attend.

User Story 11: Update Invitation Status Indicators
Points: 3
Assigned to: Erhan Wang

Update the status indicators for invitations to accurately reflect the current state of participant responses.



Test

Event Detail

Location :

Yes/No Ratio : 0/0

Max Capacity : X

Time for Reminder :

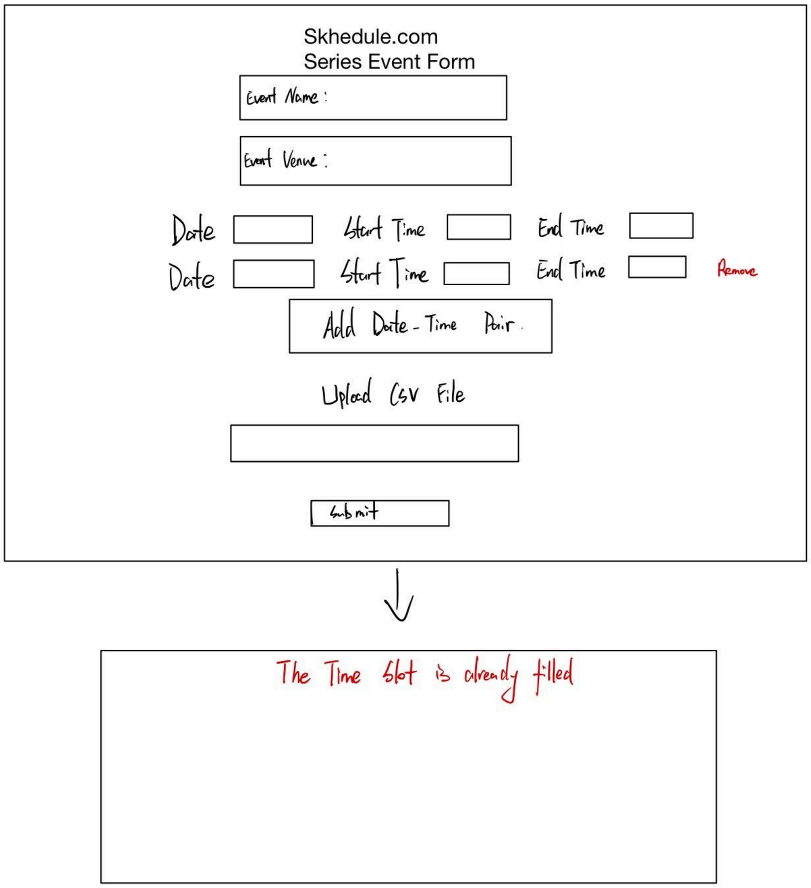
Send Email: Send Email

Send Reminder Email : Send Reminder Email

Attendees :
- Email : xxx@xxx.com
  Status : Invitation Not Yet Sent
- Email : xxx @ xxx .com
  Status : Invitation Sent - Awaiting reply

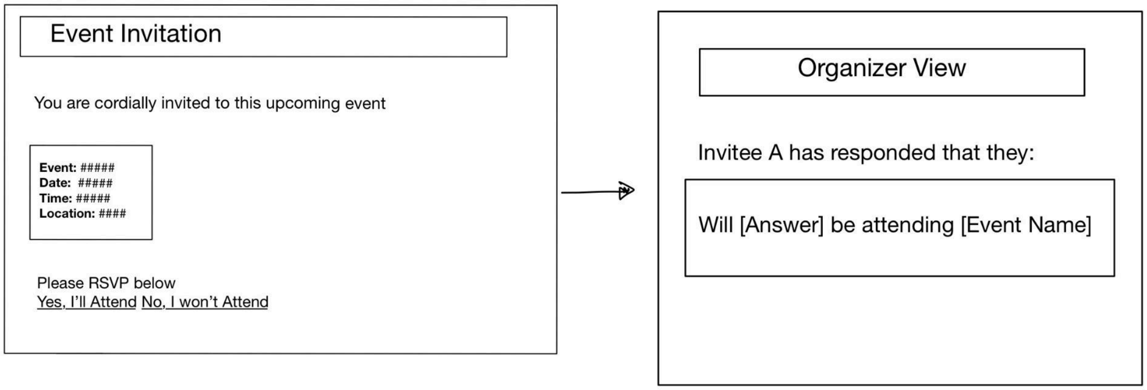
User Story 12: Handling Time Slot Conflicts for Series Event
Points: 1 (Least Priority)
Assigned to: Glen Browne

As an event organizer, so that time slot conflicts are handled gracefully, I want to notify users when a selected time slot for a Series Event is already filled and provide options to select an empty slot.



User Story 13: Send Event Rejection/Regret
Points: 3
Assigned to: Erhan Wang

As an invitee
So that I can report to the organizer that I won't be attending the event to which I was invited.



Event Invitation

You are cordially invited to this upcoming event

Event: #####
Date: #####
Time: #####
Location: ####

Please RSVP below
Yes, I'll Attend No, I won't Attend

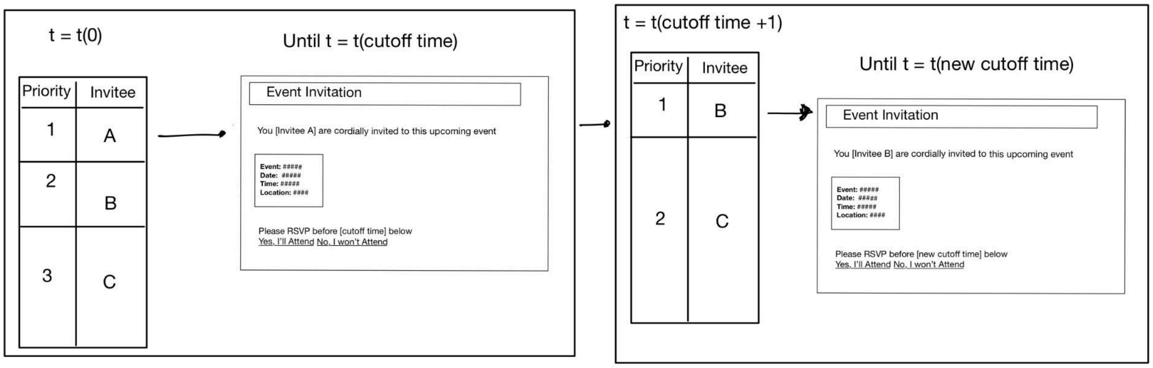
Organizer View

Invitee A has responded that they:

Will [Answer] be attending [Event Name]

User Story 14: Send Email to Next Invitee Once Slot Becomes Available
Points: 5
Assigned to: Debal Goswami

As an event organizer
So that an invitation is automatically sent to the next invitee on the list in order of priority when a slot becomes available.



t = t(0)

Until t = t(cutoff time)

Priority	Invitee
1	A
2	B
3	C

Event Invitation

You [Invitee A] are cordially invited to this upcoming event

Event: #####
Date: #####
Time: #####
Location: ####

Please RSVP before [cutoff time] below
Yes, I'll Attend No, I won't Attend

t = t(cutoff time +1)

Until t = t(new cutoff time)

Priority	Invitee
1	B
2	C

Event Invitation

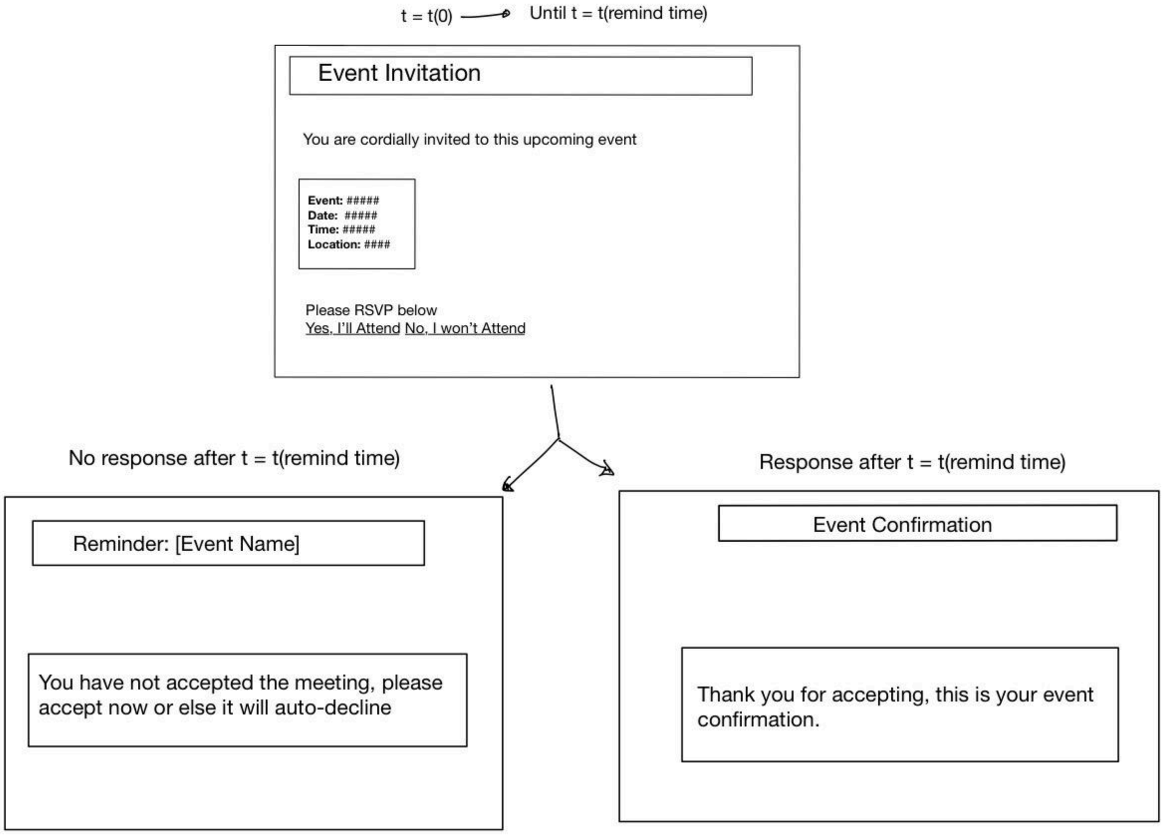
You [Invitee B] are cordially invited to this upcoming event

Event: #####
Date: #####
Time: #####
Location: ####

Please RSVP before [new cutoff time] below
Yes, I'll Attend No, I won't Attend

User Story 15: Auto Send Email Reminder
Points: 3
Assigned to: Eric McGonagle

As an event organizer
So that I can automatically send an email reminder to accept or reject an invitation when the timer runs out.



t = t(0)
Until t = t(remind time)
No response after t = t(remind time)
Response after t = t(remind time)

Event Invitation

You are cordially invited to this upcoming event

Event: #####
Date: #####
Time: #####
Location: ####

Please RSVP below
Yes, I'll Attend No, I won't Attend

Reminder: [Event Name]

You have not accepted the meeting, please accept now or else it will auto-decline

Event Confirmation

Thank you for accepting, this is your event confirmation.

User Story 16: Auto Send Email Confirmation of Attendance RSVP Decision
Points: 5
Assigned to: Prakhar Singh

As an invitee
So that I can confirm my choice when I choose to attend or not via the emailed invitation.

User Story 17: Limit Number of Invitations Sent to Max Capacity
Points: 2
Assigned to: Pankaj Tiwari

As an event organizer
So that I can limit the number of invitations sent to the max capacity I choose.

Roles

For the duration of the project, while all team members worked as devs, three members settled into primary roles. Glen Browne was our Scrum Master, as he took up responsibility for the Scrum workflow around the project. Glen also managed the Github repository, so he also functioned as our System Administrator. Glen worked diligently along with the help of Prakhar to ensure the project had successful Continuous Deployment. Eric McGonagle functioned as our Product Owner, he met with the customers and set up meetings in order to relay requirements and manage communications. Carlos Meisel settled into the role of Test Master as he was head of testing, he oversaw the writing of Rspec tests and made sure code coverage was sufficient, maintainable and ensured that target behaviors were achieved by features. These three team members, along with Debal, Pankaj, Prakhar, and Erhan all worked as full-stack developers throughout the course of the semester. Below is a detailed view of each team member and their roles.

Iteration 0:

Pankaj Tiwari	Debal Goswami	Eric McGonagle	Glen Browne	Carlos Meisel	Erhan Wang	Prakhar Singh
Full Stack Dev	Full Stack Dev	Product Owner / Full Stack Dev	Scrum Master / Full Stack Dev /Sys Admin	Test Master / Back End Dev	Full Stack Dev	Full Stack Dev

Iteration 1:

Pankaj Tiwari	Debal Goswami	Eric McGonagle	Glen Browne	Carlos Meisel	Erhan Wang	Prakhar Singh

Full Stack Dev	Full Stack Dev	Product Owner / Full Stack Dev	Scrum Master / Full Stack Dev /Sys Admin	Test Master / Back End Dev	Full Stack Dev	Full Stack Dev

Iteration 2:

Pankaj Tiwari	Debal Goswami	Eric McGonagle	Glen Browne	Carlos Meisel	Erhan Wang	Prakhar Singh
Full Stack Dev	Full Stack Dev	Product Owner / Full Stack Dev	Scrum Master / Full Stack Dev /Sys Admin	Test Master / Back End Dev	Full Stack Dev	Full Stack Dev

Iteration 3:

Pankaj Tiwari	Debal Goswami	Eric McGonagle	Glen Browne	Carlos Meisel	Erhan Wang	Prakhar Singh
Full Stack Dev	Full Stack Dev	Product Owner / Full Stack Dev	Scrum Master / Full Stack Dev /Sys Admin	Test Master / Back End Dev	Full Stack Dev	Full Stack Dev

Iteration 4:

Pankaj Tiwari	Debal Goswami	Eric McGonagle	Glen Browne	Carlos Meisel	Erhan Wang	Prakhar Singh
Full Stack Dev	Full Stack Dev	Product Owner / Full Stack Dev	Scrum Master / Full Stack Dev /Sys Admin	Test Master / Back End Dev	Full Stack Dev	Full Stack Dev

Summation of points:

Pankaj Tiwari	Debal Goswami	Eric McGonagle	Glen Browne	Carlos Meisel	Erhan Wang	Prakhar Singh
31	33	30	32	29	29	30

Sprint Iteration Summaries

Scrum 1:

The main goal of this sprint was to add some values to the existing codebase with the main concern of understanding the legacy code as well as the file structure; this would help the team gain a more complex understanding of how the project works as a whole to maximize the efficiency in actual code development and testing when adding features to the project.

Achievements:

Reviewed legacy code

Deployed app on Heroku

Created login form

Story: read through excel file

Story: add login feature

Scrum 2:

After having spent some time understanding the codebase, this sprint was more focused on improving some of the features previously started in sprint 1. This includes addressing many of the above-mentioned user stories. With the login page in its very basic stages, before moving onto any major design changes to this page, we wanted to ensure more complete functionality of the login page to include sign-up features in order to create accounts as well as continue as a guest for users who don’t want to create an account for let’s say one single dinner party event.

Our team finished the implementation of the following features:

More complete login functionality
User account creation feature
Invitations being sent by clicking the “Send Emails” link
Timer for invitation reminder emails and invitation revocation

Completed stories:

See format details for CSV and Excel email files on Send Invitation page
Properly parse csv/excel email addresses
See list of upcoming events
Accept or decline invitation
Upload CSV file of invitees
Upload Excel File of Invitees
Employ timer that expires if invitee does respond to invitation
Send an invitation

Create a user account

Scrum 3:

This sprint was intended to get another large chunk of work done on ensuring some user functionalities as well as touching up on some design changes before final features and UI design were put in place in the fourth and final sprint. This third sprint featured finalizing the sign-in features for the login page as well as starting to make this page look more uniform and in sync with the rest of the website. After some initial testing with the google calendar and timers, these two features were to be fully implemented by the end of this sprint as well as starting to get the reminder emails sent out, namely at least the first round of reminder emails in addition to completion of the RSVP functions featured in the email invitations.

Sprint Achievements:

Our team finished the implementation of the following features:

The countdown timer fully implemented on the website

Add to Google Calendar Functionality

Reminder emails are sent after the countdown timer runs out

RSVP functions being addressed

Scrum 4:

This sprint's main goal was to resolve any bugs from the previous sprint as well as finalize some features that needed to be added to make the product more complete. On top of this, any design changes to features that were previously added/are going to be added need to take effect so the look and feel of the product will be more appealing to users. This fourth sprint included various feature changes that added more functionality to the product to increase the effectiveness as well as increase the ability for the product to carry out more automated tasks.

Sprint Achievements:

Our team finished and implemented the following features:
Google Calendar functionality fully implemented with all bugs resolved
RSVP functions working as well as the according response/status shown on the event status
Series event creation works properly
Reminder time/countdown timer update
Reminder email implementation
Invitation status indicators correctly indicate the status

Email address text box works - user can enter email addresses in the text box, upload a csv or xlxs file, both
The app adjusts dynamically to fit the device it’s displayed on
Invitations are limited to the max capacity

Customer Meetings

February 1, 2024 Meeting with Jonathan Tan via Zoom

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 like 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 pick up the legacy code; something to note is his idea of starting a project from scratch which will be discussed with the group as well as with Professor Ritchey (not sure if this is allowed or not).

We got four user stories as follows

First user story is the general invitation with the goal of filling up all of the available slots for a given event (i.e. if 1000 tickets available for a concert, ideally all 1000 slots will be sold/filled). This is a general invitation that can be sent to anyone. Time constraint can be set per event

Second user story is a priority invitation for special guests like speakers at events. This invitation goes in order of priority (if Eric is first priority he gets an invitation, assuming Glen is second in priority he will only get the invitation should Eric decline his) 24 hour time constraint for now (subject to change)

Third user story is user accounts so users can log in and view their current events and possibly other events that they are invited to/can view and sign up for

Fourth user story is linking events on one's account to his or her calendar let's say through Google calendar for now

We agreed to have weekly meetings Thursday at 9:00 AM CST.

February 20, 2024 Meeting with Jonathan via Zoom

We met with Jon again to show him any value we added to the project throughout the first sprint. The team discussed the login page allowing users to login and be directed to the dashboard for which they can create and view events. The team mentioned that the login page was in its first stages of development so there weren't any features for creating users yet, just mainly logging in for users that are already created i.e. developers as of now for testing.

Jonathan didn't have many questions or concerns except for mentioning the importance of dynamic viewing for various sized screens like tablets and phones in addition to computer viewing since there is a high chance users will be viewing and creating events through mobile means.

The team will continue to work on the features that were discussed from the first meeting with the client, ensuring to break them down to more specific user stories such that the next sprint has more accurate story points. The next meeting will be Thursday Feb 29 at 9:00 AM CST to go over any progress that will be made shortly after the second sprint starts.

March 1, 2024 Meeting with Jon and his colleague Marcus via Zoom

We met with Jon again to discuss some minor changes throughout the course of the second sprint (halfway through). The main point of discussion for this meeting was to meet with Jon's colleague Marcus to provide some insight on the project mainly about role assignments, specification on event details, and timed responses and other functionalities dealing with the invitations as a whole. A document will be sent in the links chat detailing the suggestions that Marcus made. The next meeting is planned for Thursday the 7th at 9:00 AM to review any changes in the latter half of this sprint in preparation of Spring Break preventing an immediate time frame to meet and display MVP from sprint 2

March 8, 2024 Meeting with Jon

The team met with Jon to show the MVP from sprint 2 where some features were added such as adding a single dinner party event to Google calendar, sign-up and continue as "guest", and adding a reminder timer to events, and more. The team discussed that while these features were merged with the main deployment, there are still some stories left that will be addressed in the next sprint such that these features can be fully complete such as adding Google calendar events for series events and sending "confirmation" and "reminder" emails after a series of timer countdowns. In addition to this, the team notified Jon that the login page's main priority right now is functionality thus the design and appeal of the page will come in later sprints; as a result of this the team asked what his initial idea for the design should be so that we can have a basic understanding and he suggested keeping the theme in sync with the rest of the website (like when being directed to the home page after logging in). The next meeting will be on Thursday March 21 at 9:00am CST

April 4, 2024 Meeting with Jon

The team met with Jon to show the MVP from sprint 3 where we displayed some features like the countdown timer on single dinner party events, reminder emails being sent after the timer runs out, logging in and signing up being implemented with proper design updates, and more.

Jonathon went more in-depth on certain testing details which provided some very valuable insight into what the team can focus on in this fourth and final sprint. There are some features that will be added to single dinner events to make this aspect of the project more complete as well as getting the series events up to speed with single dinner events regarding functionality.

April 24, 2024 Meeting with Jon via Zoom

The team met with Jon to demo the features added to the project in sprint 4. Jon was pleased to see that we had addressed a chunk of the suggestions that Marcus had provided to the team regarding improving the app. The team notified Jon that we'll keep in touch with him in the coming week as we finish the final report and will send him the customer satisfaction survey as well.

BDD / TDD Process

Throughout the course of the semester, we had to not only adhere to an iterative approach for testing but also develop tests retroactively. All team members iteratively wrote tests for their implemented features, spearheaded by Carlos Meisel as our appointed Test Master. Throughout the course of the semester, we changed some of the backend functionalities of the legacy project, providing the unique challenge of writing new tests for legacy code while also ensuring minimal overlap.

At each iteration (sprint), the team met up for an evening meeting where we would review the functionalities of each feature and further discussed possible testing avenues. In these meetings we discussed the requirements for each feature, success criteria, failure criteria, edge case discovery, edge case solutions and maintainability of each feature.

BDD Process with Cucumber:

- **Scenario Design**: We started with the desired behavior of a given feature, from there we scoped out the key components of successful behavior. After our test team agreed on success criteria, we drafted up a feature file, articulating the high-level requirements.
- **Implementation:** It was important to design step definitions with enough detail so that we could ensure that at each step the correct behavior was being achieved. Not only was this a good practice but it served a double purpose as it helped with debugging and error diagnostics. As the system grew more complex, it can become increasingly more difficult to trace the sources of error, this is why the need for a debugging framework was crucial. We were able to leverage the granularity of our step definitions to double as a debugging tool.

TDD Process with Rspec:

- **Test Case Design:** Here the test team started by scoping out the key behaviors that we'd want each method to achieve. At the convergence of agreement on the core behaviors, we'd then write some tests to attempt to cover the method.
- **Implementation:** Once the method has begun to be covered by some preliminary tests, we'd then begin to write new tests to attempt to discover edge failure cases to expose any potential weaknesses in the codebase.
- **Refactoring**: Once lines of code have been covered and tests have been drafted, we begin the code refactoring process. This is done alongside assessing Rubocop progress as this also assists us with determining code styling and code quality.
- **Iterations:** Now after this whole process has been completed, we iterate for each sprint to assess what dependencies and functionalities have changed.

Challenges Faced with Legacy BDD and TDD:

- **Stale Rspec Tests and Low Coverage:** At the beginning of the semester there was a very large amount of tech debt due to the need for legacy Rspec refactoring and the need for increasing code coverage before adding new code to the codebase.
- **Stale Cucumbers:** In the beginning, none of the cucumbers were passing for the codebase, this created a tremendous amount of tech debt since it was assumed that we'd use the cucumbers to start to build an understanding for the codebase. After a lot of sweat and tears, we decided to part ways with all of the legacy cucumber scenarios and to re-write our own. This was much more reasonable as the codebase had changed so much with the new requirements provided from the customer and with their specific requests for us to revisit legacy features and modify them.
- **Large Requirement for Code Refactoring:** The legacy code was in dire need of code refactoring as there were multiple methods with the same name, and several instances where lines of code achieved the same thing. Code Climate helped tremendously with diagnosing these issues, and reduced debugging time for our team.

Challenges of BDD and TDD:

- **Local Development vs. Production**: Here "Production" just refers to our deployed version of the codebase. Our testing team faced a lot of challenges when trying to implement BDD while in the local development environment. We had Carlos focus on trying to research methods to attempt to make local testing as close to deployment as possible. While we were able to achieve some mirror scenarios, we still faced challenges with certain scenarios. One of these was the verification of successful google calendar behaviors. We were able to successfully determine that the functionalities were achieved via Rspec testing but we were unable to directly access them behaviorally via cucumbers (BDD). In order to directly test these functions, we had to manually deploy and verify by going through the motions. This is significantly slower and technically expensive, drastically slowing down our testing process.

- **Code Complexity Increase:** As the semester progressed, the codebase only got bigger, tremendously bigger. As the codebase grew, coverage had to continuously increase in order to account for fresh code. Not only coverage but new tests had to be written, old tests had to be refactored in order to be kept up-to-date. Perhaps most difficult, the cucumber scenarios had to be continuously modernized with each sprint. We found ourselves having to change core features more frequently than anticipated, resulting in the need for continuous BDD upkeep.

Benefits of BDD and TDD:

- **Clear Success and Failure Criteria:** By implementing BDD and TDD, we were able to layout clear failure points in our codebase. We were able to quickly diagnose system failures and had an efficient method to test out our solution processes.
- **Iterative Design Improvement:** We were able to use the BDD and TDD methods as a mechanism for assessing design improvements. Not only this but they were used as a mechanism to partition ideas for future feature brainstorming. For example, while designing a cucumber test requiring one to login to the website and interact with it, it would be reasonable to assume the user would need to login right? Well what happens if that person is a new user, at this point they would need to sign up and make an account. Hence, by designing a login cucumber, we discovered the need for a sign up option. This sign up feature must have an ability to add the new user-password combination to the database, the cycle continues.

In conclusion, there were many hurdles and triumphs for our testing team this semester. We primarily focused on improving the overall code quality and most importantly improving the codebase's testing. We did so in a way where future teams would be able to maintain these tests, and in hopes that our BDD and TDD will aid in developing an understanding of the codebase, hoping that the future team members would not have such a difficult time being brought up to speed as we did. Our testing team really wanted to focus on getting things right, we wanted to make sure that we not only achieved coverage and test passing but we wanted to make sure that we established robust mechanisms that would help future teams moving forward.

Configuration Management Approach

The team used Git to handle version control which helped a great deal with managing the multitude of deployments the project went through in addition to providing a method of being able to rollback certain deployments given any environment issues that otherwise would not have been experienced in the local development environment. The team utilized pull requests such that no commits and modifications ever touched the main branch but rather another team member would have to look at any changes before being able to merge the changes into the main branch.

Deployments were made automatically after changes were looked over and confirmed before merging into the main branch. Being that this was a legacy code project, many of the errors that the team ran into were brought to attention after various deployments which oftentimes indicated that there was a dependency and environment concern rather than something being faulty regarding the foundation of the codebase; this resulted in little to no need for spikes. The final count came in at 68 branches and 2 releases. The high branch count was due to our approach for developments, where we created a new branch for each story. This was done so that code backtracking and debugging was easier to document. We also set apart some testing branches which were kept up to date in order to actively and retroactively create tests for the ever changing codebase.

Production release was quite simple and convenient with Heroku, but one issue to be aware of is that of managing credentials contained in encrypted files. We used an encrypted file to provide our username and password for our automated email service, and the master.key file that is necessary to decrypt the credentials.yaml file will not get pushed to an online repository for the sake of security. Accordingly, these credentials have to be configured manually as environment variables using the Heroku command-line interface in order for the email service to authenticate.

The Heroku deployment environment is configured for automatic deployments, so that every time code is pushed to the main branch on the github repo, a new version of the app is deployed on heroku.

Gems and Other Tools

Describe the tools/Gems you used, such as GitHub, CodeClimate, SimpleCov, and their benefits and problems.

1. **Rspec:** without repeating too much from the BDD/TDD Process portion, Rspec is a testing framework for Ruby. Specifically, it helped tremendously for facilitating test-driven development (TDD). It helped elevate code quality and allowed for us to use it as a method for code refactoring.
2. **Rspec-rails**: This is a specifically tailored version for Rails application. It basically extended the capabilities of Rspec, allowing it to integrate with the Rails framework.
3. **Rubocop**: Rubocop is a static code analyzer and formatter for Ruby. It helps enforce many of the guidelines outlined in the community Ruby Style Guide. Rubocop can automatically detect and sometimes correct stylistic errors, inconsistencies, and code smells in your Ruby code, making it more readable and maintainable.
4. **SimpleCov**: SimpleCov is a code coverage analysis tool for Ruby. It provides a detailed report of which lines of your code are being executed and which are not under test coverage. This is crucial for maintaining a healthy test suite and ensuring that all critical paths are tested in your application.
5. **Timecop:** Timecop is a gem that allows "time travel" and "time freezing" capabilities in tests, making it simple to mock and manipulate time within Ruby. This can be invaluable for testing time-dependent code, such as expiration dates, timeouts, or age calculations without relying on the system clock.
6. **Capybara:** Capybara is an integration testing tool designed to simulate how a user would interact with a web application. Capybara supports multiple backends, such as Selenium, to automate web browsers, allowing you to write tests that mimic real user actions like clicking buttons, submitting forms, and verifying page content.
7. **WebMock**: WebMock is a library for stubbing and setting expectations on HTTP requests in Ruby. It allows you to stub out HTTP interactions in your tests, making it possible to test your application's logic without actually making external requests, which can speed up your tests and make them more reliable.
8. **Signet**: Signet is a gem used for OAuth 1.0 and OAuth 2.0 client-side implementations. It helps handle authentication and authorization in an application by managing tokens and building authorization headers, which is especially useful when interacting with APIs that require OAuth.
9. **Particles-js-rails**: Particles.js is a lightweight JavaScript library for creating particles-based visual effects. The particles-js-rails gem integrates this library into the Rails asset pipeline, allowing you to easily add and manage these effects in a Rails application.
10. **Google-apis-calender_v3:** Particles.js is a lightweight JavaScript library for creating particles-based visual effects. The particles-js-rails gem integrates this library into the Rails asset pipeline, allowing you to easily add and manage these effects in a Rails application.

Project Links

Links to your Project Management tool page, public GitHub repo, and Heroku deployment, as appropriate. Make sure these are up-to-date.

Project management tool: **https://www.pivotaltracker.com/n/projects/2690139**
GitHub link: **https://github.com/gdbrowne85/SwIRL-CSCE-606**
Deployed app link: **https://swirlskehdule-f316b598c688.herokuapp.com**

Links to your presentation video and demo video.

Project Presentation: **https://youtu.be/zF25324Oouk**

Project Demonstration: **https://youtu.be/ofwE-XH64AE**

Combined Videos: **https://youtu.be/MQI-c8KRxjU**