Marathon: Scrum App

Lucas Kopp, Shreya Thothathri, Hannah Solis, Mollie Mero

Problem Statement

Many companies struggle with finding project management systems that are suited for their team's needs. Some teams prefer longer sprints and daily standups, while others prefer the opposite.

Explanation and Rationale for Proposed Solution

Many companies struggle with scheduling sprints. A standard sprint doesn't always suit every team. With many teams having different requirements from their sprint, they need the ability to customize how often they do standups and how long their sprints are. It could even account for periods between sprints where teams can prepare for upcoming projects by gaining skills necessary or discuss with other teams how the sprint may be improved. When necessary, it could even mark in calendars when people within or across teams need to meet at the end of their sprint to discuss their next step. Our more customizable platform will allow for teams to change sprint requirements for their needs.

Use Cases

Case 1: Design a custom sprint

Preconditions:

- A project team is formed
- A project manager is assigned

Main flow:

1. Manager opens the current sprint configuration [S1]
2. Manager adjust settings using the text inputs and sliders [S2]
3. Manager chooses to deploy this sprint configuration to the current project [S3]

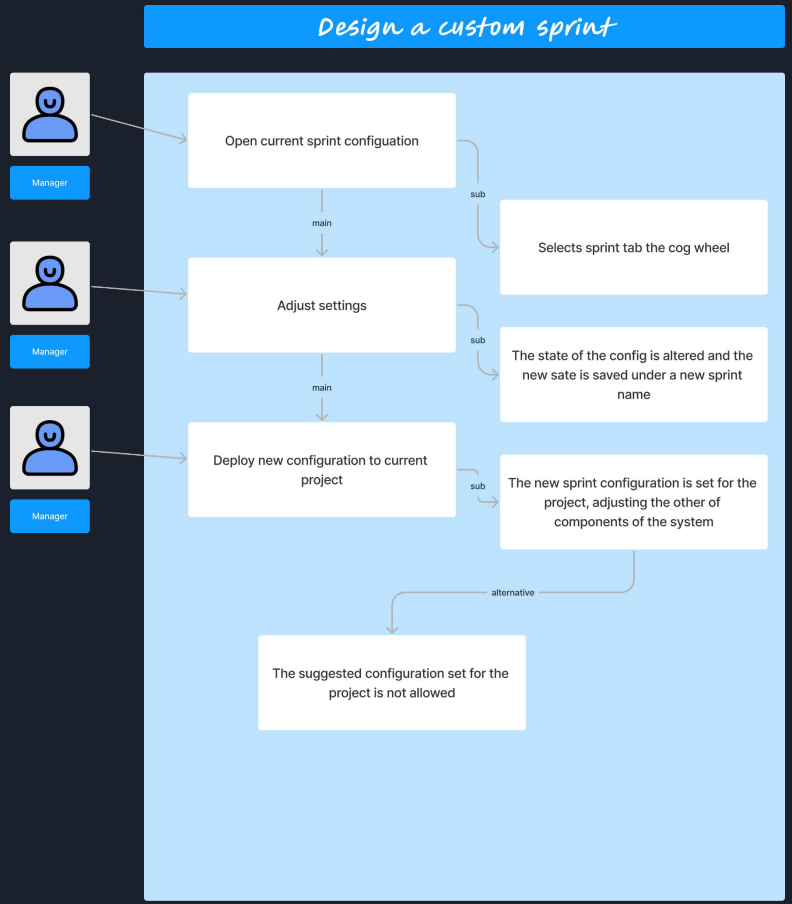
Sub flows:

1. [S1] Manager clicks on the sprint tab, then the settings cog wheel
2. [S2] The state of the default config is altered and the new state is saved under a new sprint name
3. [S3] The new sprint configuration is set for the project, adjusting the other components of the system and what the team sees

Alternative Flows:

1. The suggested configuration is not allowed and needs to be changed

Case 1 Diagram



Case 2: Create meeting time based off of developer's time

Precondition:

- A project exist
- Developers have filled out a survey specifying their availability

Main flow:

1. The Manager opens up the schedule tab in the main interface
2. The Manager selects the create meeting time option and enter the length of the meeting [S1]
3. A suggested time is given to the manager
4. The manger is given the option to schedule the meeting [S2]

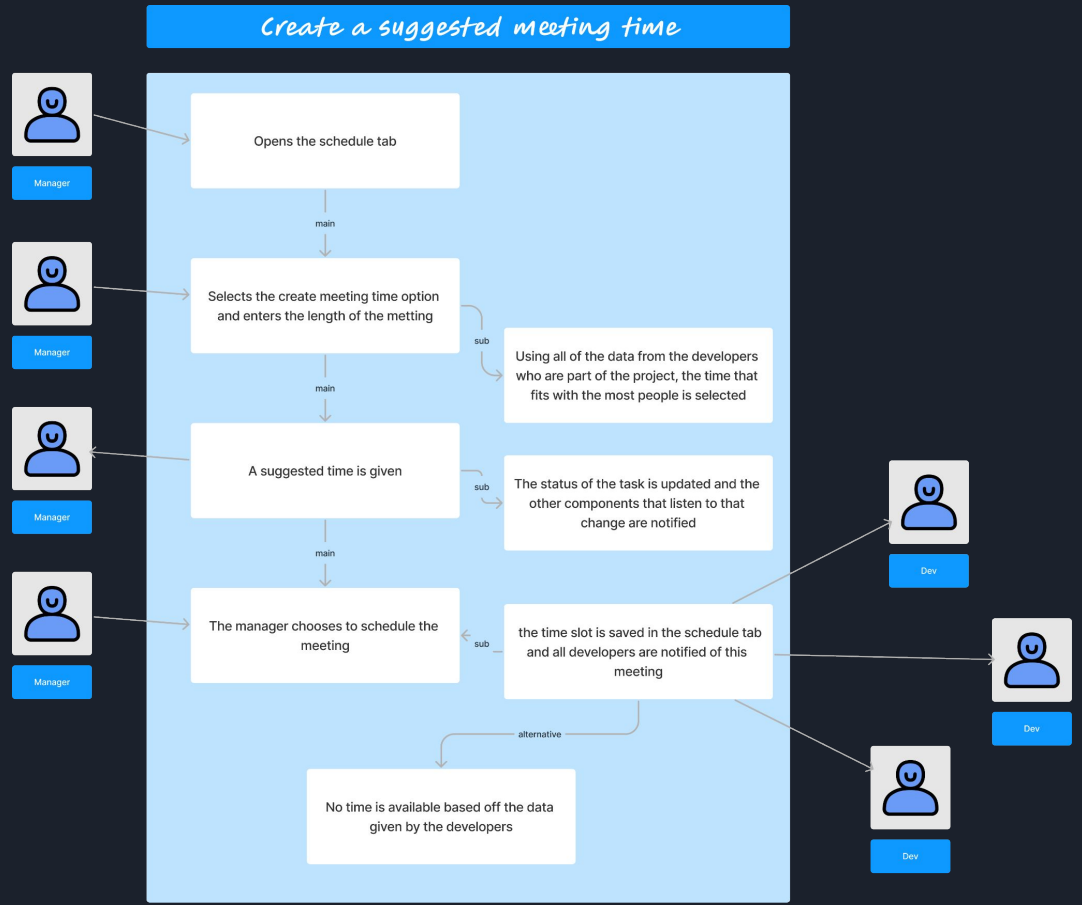
Sub flows:

1. [S1] Using all of the data from the developers who are part of the project, the time that fits with the most people is selected
2. [S2] If yes is selected the time slot is saved in the schedule tab and all developers are notified of this meeting

Alternative Flows:

1. No time is available based off the data given by the developers [S1]

Case 2 Diagram



Case 3: Team enters a swarm

Precondition:

- Team members all know what a swarm is
- Tasks have already been assigned priority levels

Main flow:

1. Manager decides to start a swarm. Team chooses priority items
2. Members are all notified of swarm status and begin to work on high priority tasks
3. As tasks finish, members are given next task to swarm on
4. Swarm is finished and team goes back to their regular work

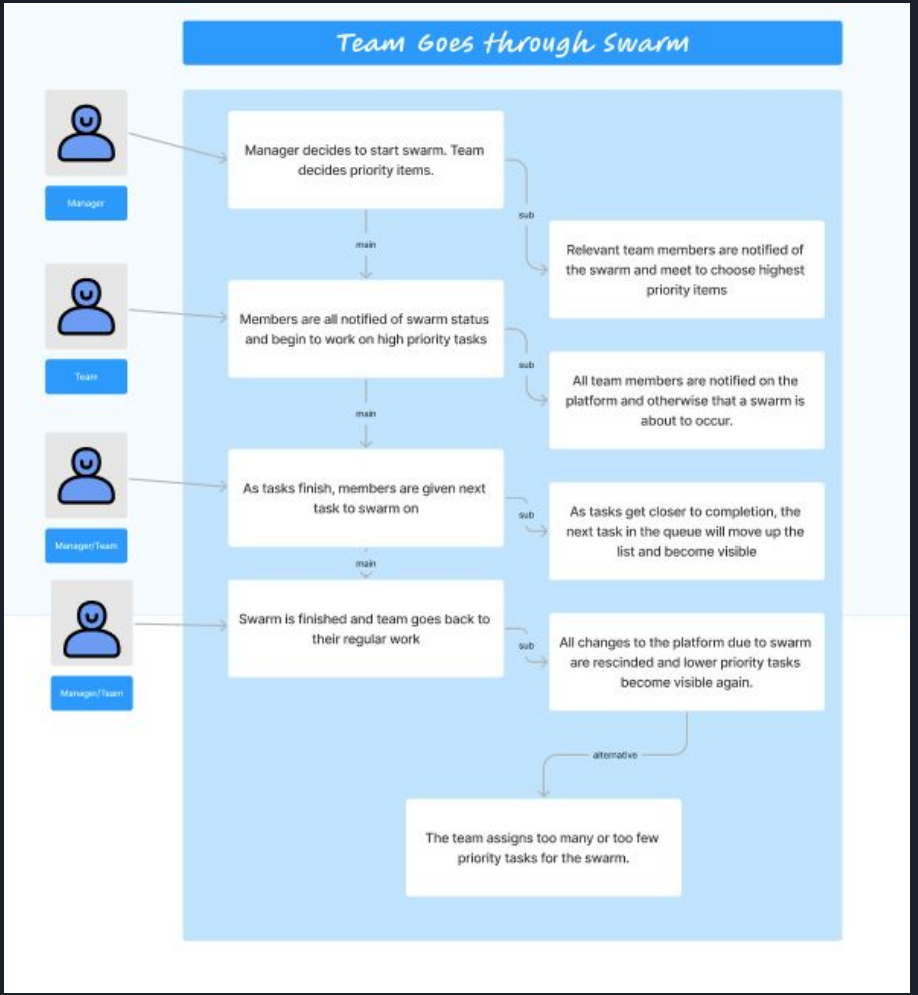
Sub flows:

1. Relevant team members are notified of the swarm and meet to choose highest priority items. This list will be the only thing team members will be able to access during the swarm
2. All team members are notified on the platform and otherwise that a swarm is about to occur. They are only given access to high priority tasks on the board
3. As tasks get closer to completion, the next task in the queue will move up the list and become visible
4. All changes to the platform due to swarm are rescinded and lower priority tasks become visible again.

Alternative Flows:

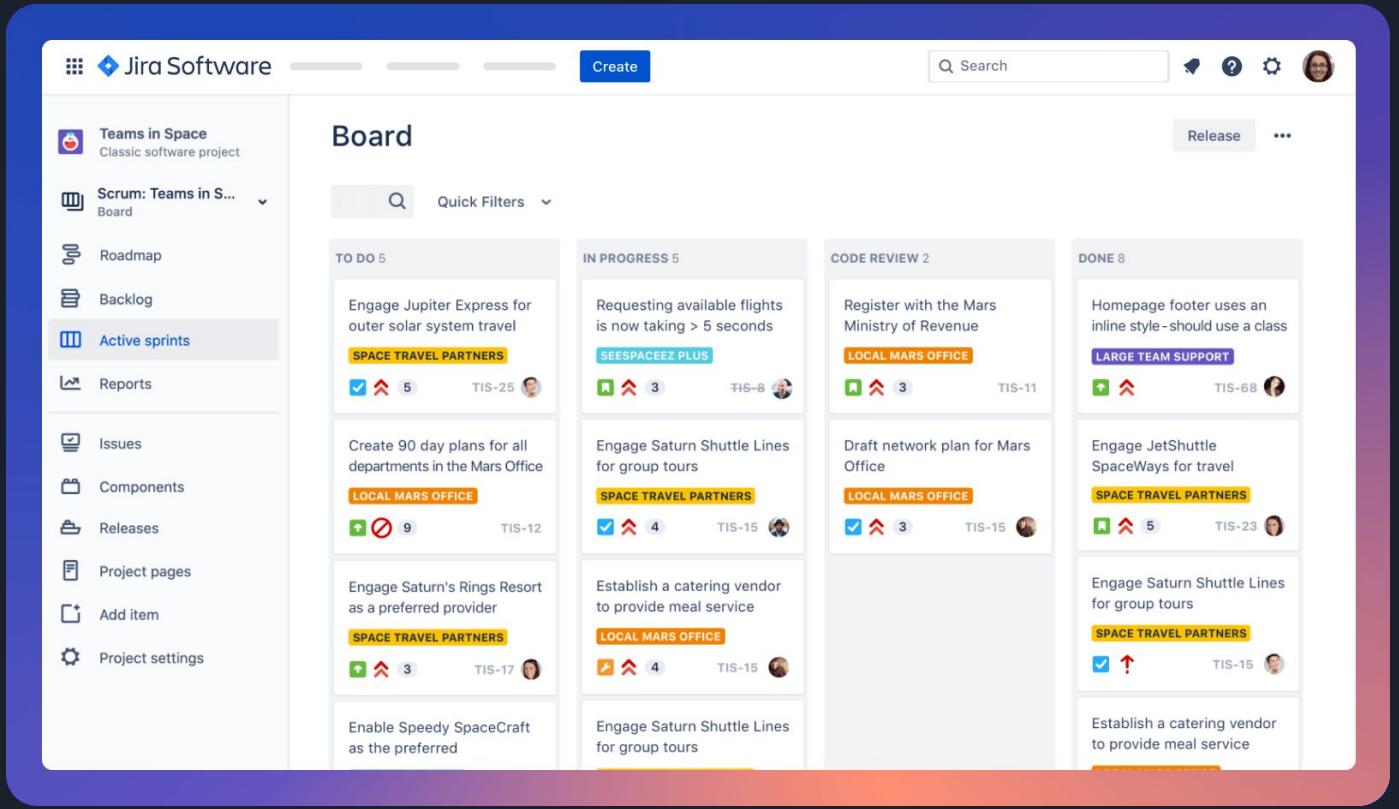
1. Not enough or too many priority items
2. Members accidentally do the same work at the same time
3. No more tasks to swarm on before swarm ends
4. Team's work was finished by swarm

Case 3 Diagram



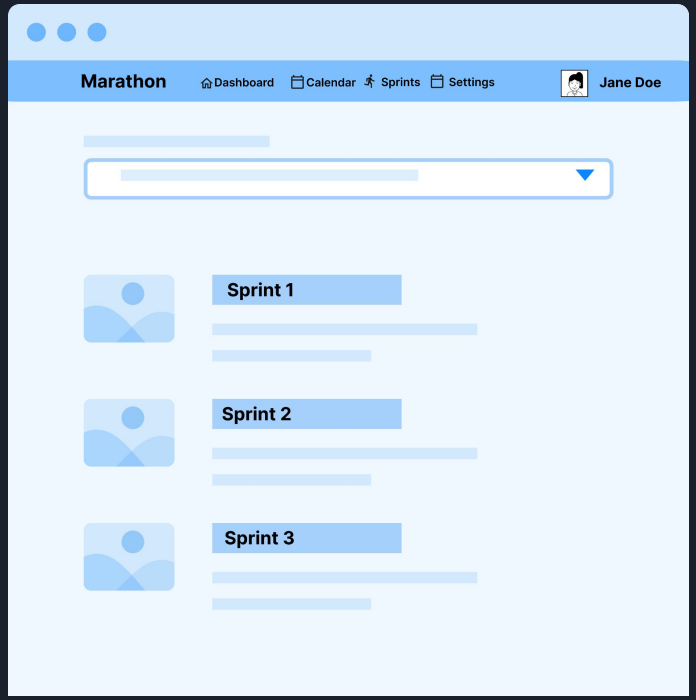
Visual Representation of Project

Inspiration for design



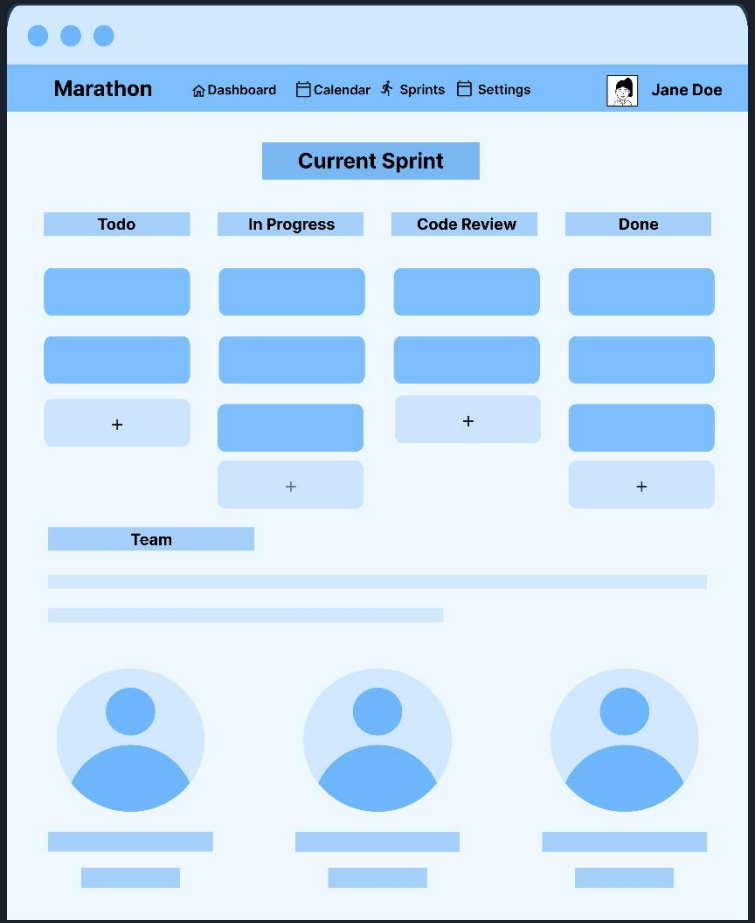
Home Page

- Allows users to navigate between their sprints
- Allows users to search for current and past sprints
- Automatically shows current sprints first, then upcoming sprints
- Shows the name of a sprint, the current stage, a short description, and a project picture



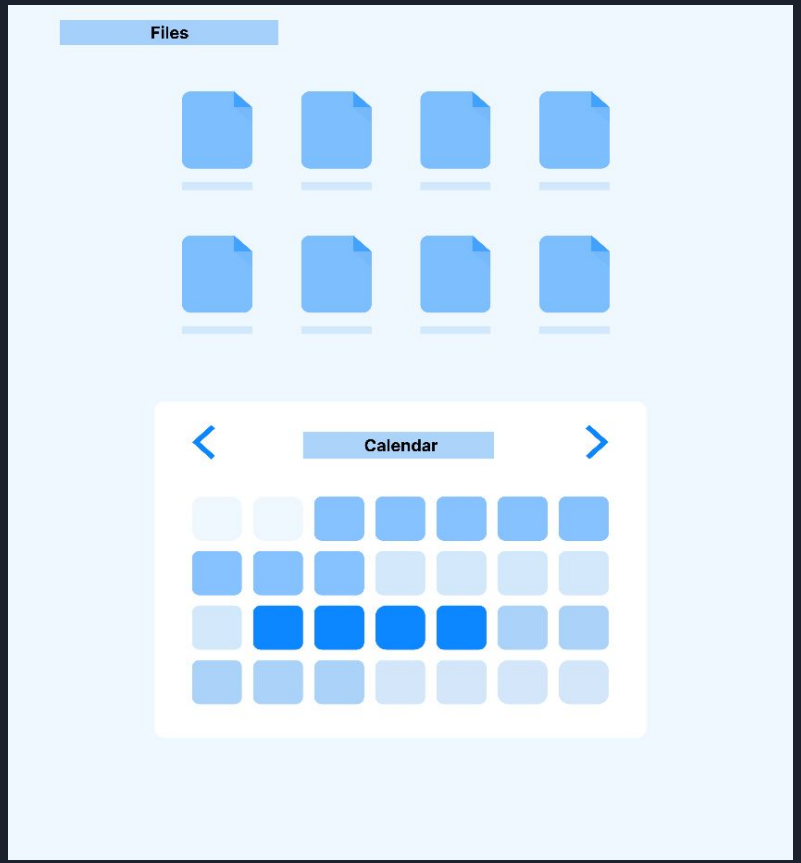
Specific Sprint Page

- Shows the five basic sprint sections - Todo, In Progress, Code Review, and Done
- Each section will have a task with
  - Name of task
  - People working on it
  - Short Description including necessary skills
  - Story Points
- Allows any user to add to a section
- Shows the team members that are a part of the sprint



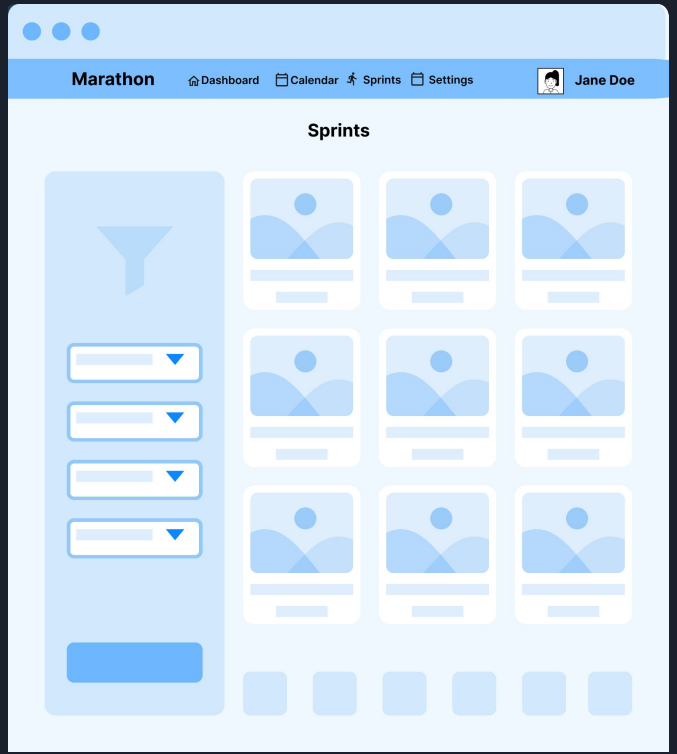
Specific Sprint Page

- Allows team members to upload and access all relevant files to the sprint
- Shows the stages of the sprint in the calendar in different colors with the current stage highlighted
- Sprint is highly customizable with phase type, total length, and phase length
- Will also have a phase for learning included in the sprint to make sure team members are prepared for upcoming phases



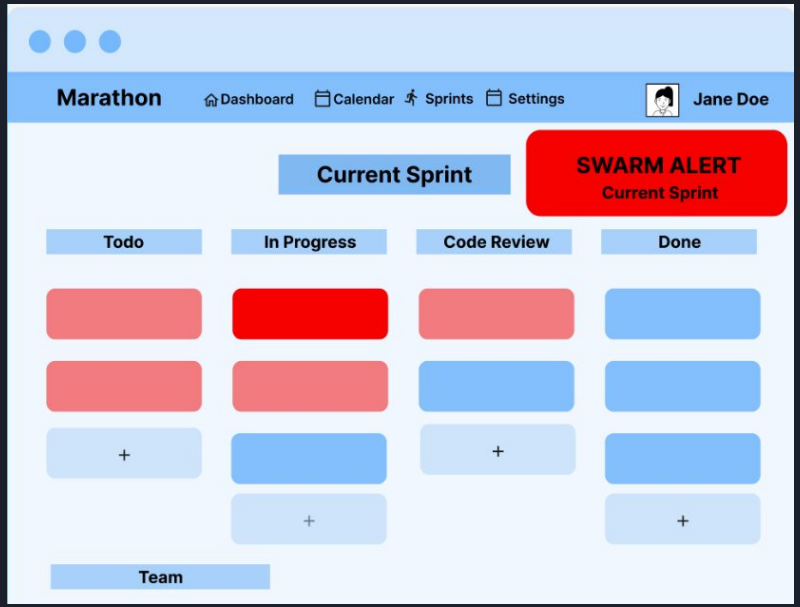
Sprint Page

- Allows users to filter through past and current sprints
- Can search by
  - Team members involved
  - Projects
  - Length of Sprint
- Will list recently used sprints before search
- Can be used for sprint reviews or to easily find certain files



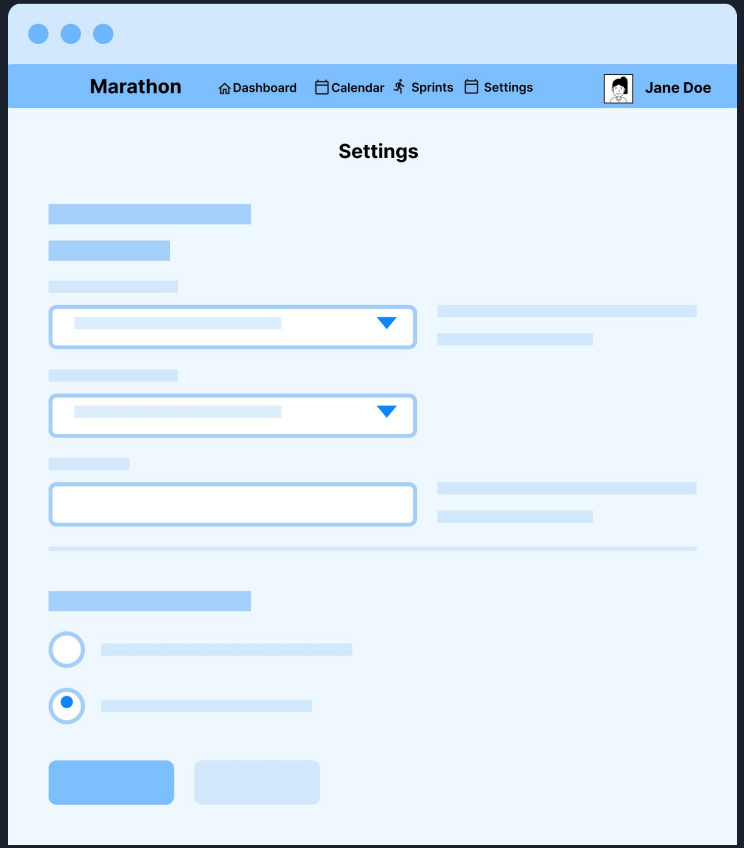
Swarm Changes

- When swarms occur, all team members receive notifications and the sprint page will automatically link to the specific sprint
- High priority tasks will be highlighted in red and lower level tasks will be either blue hidden until the swarm is over
- Only the team manager will be able to call a swarm



Settings Page

- Will allow users to change accessibility settings (ex. Contrast, Color Scheme)
- Will allow users to change their login information, email, language
- Allows users to change notification settings between different projects or phases



Processes

1. Storyboarding : Useful for walking through the application step by step, figuring out all of the necessities

2. UI Designs: Helpful visualization of the app and what frontend functionality is needed
    a. Wireframe

3. Diagram making for use cases: Useful for walking through specific use cases of the application and the user experience

4. Scrum
    a. Meetings: Allowed time to catch up with team and discuss work done or problems that need to be resolved
    b. Review : Helpful for tracking all of the work done thus far and what's left

Tools Used

- Figma - wireframe and use case models
- Google docs - Scrum note taking
- Google slides - present updates to class
- Version control - Git - through Github to manage team files

What we learned

1. Software Engineering process
    a. Hand-on experience with the full scrum process

2. Design detailed use cases
    a. Five cases covering general and advanced use of application

3. Design user interfaces through wireframe
    a. Using Figma to create web wireframes

4. Pick a development process
    a. model-view-controller

Limitations

- Time
  - Due to the project being limited to a portion of a semester and most of that time being spent on planning the furthest we got was creating a wireframe model
- Team size
  - With a team size of four we couldn't split up task as well as a larger team. We all worked on the same task which limited our output
- Skills
  - Team members were bad at designing interfaces and software

Future Work

1. Turn wireframe into actual software
2. Do user testing on interface and functionality
3. Update software based on feedback from users
4. Polish early model to get rid of bugs
5. Add more features:
    a. Ability to share sprint configs (exporting configuration)
    b. Add more roles for team members, other than dev and manager
    c. Direct messaging between members or sharing of contact information on profile
    d. Direct surveys for developers to get project feedback
6. Rinse and repeat