A blue circle with a person running

Description automatically generated

Extrasize

Get more from every workout.

Christopher Shannon

Brian Del Carpio

Melvin Wong

Ian Wu

Simon Fraser University

CMPT 276

Contents

[Project Overview 3](#_gjdgxs)

[Kanban Methodology 3](#_30j0zll)

[User Stories 4](#_1fob9te)

[Technology Stack](#_3znysh7) 4-5

[JavaScript/React 4](#_2et92p0)

[HTML 4](#_tyjcwt)

[CSS 4](#_3dy6vkm)

Overview 5

[APIs](#_1t3h5sf) 5

[Google Calendar](#_35nkun2) 5

[Strava](#_4d34og8) 5

[Project Features](#_2s8eyo1) 5-6

[Work Breakdown Structure](#_17dp8vu) 6

[Project Timeline](#_3rdcrjn) 7

[Wireframes/Prototype](#_26in1rg) 7-8

[Data Flow Diagrams](#_lnxbz9) 9

# Project Overview

Our project is centered on developing an innovative running workout application that integrates seamlessly with APIs like Google Calendar and Strava to enhance the user experience in planning and tracking fitness activities. Utilizing a dynamic Kanban methodology to manage our workflow, we are able to address the challenges posed by our team's relative inexperience with large-scale projects by allowing for flexible task management and prioritization. The application is being built with a robust technology stack, leveraging the interactive capabilities of React for the frontend, coupled with the foundational web technologies HTML and CSS for structure and style. This combination provides users with a responsive and intuitive interface to schedule workouts, track their progress over time, and analyze their performance with visually engaging maps and statistics. As we progress, our focus on effective communication within the team and a rigorous review process ensures that we maintain high standards of quality and functionality in each phase of the project, from initial planning through to deployment and closure. This way, we can make sure our app is easy to use and works well from the time we start building it until we're ready to share it with everyone.

# Kanban Methodology

Why Kanban? One of the problems we face in this project is the lack of experience we have in a project of this scale. It would be difficult to use a waterfall method for example as we do not know all the steps that would be required before we get started.

With Kanban, we can dynamically add tasks to the list of to-do things. Allowing us to be flexible and solve problems as they come. With such a small team the biggest advantage we have is communication. The biggest risk with doing kanban is overcomplication if we add more tasks then we complete, leading us to be overwhelmed.

For our Kanban billboard, we will have 4 lists as follows:

1. **To-Do** - tasks that need to be actioned.

2. **In Progress -** tasks that are currently being worked on.

3. **Review** - tasks that are completed but should be reviewed by another member.

4. **Completed** - tasks that have passed the review process

# User Stories

**Cindy:** She loves to plan. She uses her google calendar to schedule her life. She wants to plan running workouts. Using Extrasize Cindy can schedule her workouts and record data in one application

**Clifford:** When he runs it is a spur of the moment decision. He wants to keep track of his runs on his google calendar, while also showing off to his friends on strava. With Extrasize Clifford can do both of these using our application.

**Cornelius:** Runs bi weekly on saturdays. He wants to keep track of his stats on strava but wants to keep his activity there to himself. He does this using Extrasize.

**Constantine:** They are constant in their racewalking. Constantine’s coach David wants him to upload his times to strava. Constantine struggles using technology so he uses our simple interface to upload his times to strava.

# Technology Stack

## JavaScript/React

**Why Choose It:** React is a powerful and popular frontend library for building interactive user interfaces. Its component-based architecture promotes reusability and maintainability, making it easier to develop and expand upon the application over time.

**Benefits:** It offers a virtual DOM, which ensures efficient updates and rendering. This is especially beneficial for applications like ours where users might interact frequently, like checking available slots or booking workouts.

## HTML

**Why Choose It:** HTML is the standard markup language for creating web pages. It structures the content and forms the backbone of any web application.

**Benefits:** Ensures content is organized, accessible, and can be rendered on any web browser.

## CSS

**Why Choose It:** CSS is essential for styling the web pages, making them visually appealing and user-friendly.

**Benefits:** Allows for responsive design, ensuring the app looks good on all devices, and provides a customizable user experience.

 In wrapping up our technology stack choices for the frontend, React, HTML, and CSS emerge as foundational tools ideal for modern web applications. React empowers us with its capability to produce dynamic and efficient user interfaces, while HTML provides the structural framework essential for web content. Complementing this, CSS grants the application a polished and adaptable aesthetic. Together, these technologies form a cohesive combination, ensuring the web application is both functionally robust and visually engaging. With such a stack, users can anticipate a seamless and enjoyable experience when booking their daily running workouts.

# APIs

The two chosen api's to implement are, the google calendar and strava. These apis form a symbiotic relationship for our applications. The google calendar api allows for future events to be planned in the user’s google calendar. Meanwhile the strava api records the information of the workout and also posts to strava. Using the google calendar api to store the date, time and strava events ID. Using the latter to recall the information of the workout.

## 

# Project Features

The first feature is the ability to plan future workouts. The user can input the time and date which will get stored in google calendar. Can choose for workout to be recurring.

The second feature is to see upcoming events. Since we inputed the events into google calendar we can see in the actual calendar app or by going to our application. The calendar stores the date and time of the specified workout after getting input from the user.

The third feature is inputting workouts into Strava. Users can either input from the list of scheduled workouts or create a new workout that was not scheduled. Users input distance and time of workout along with date and time if not scheduled. A Strava event is created and the id for this event is stored in the google calendar event for recall. Users input’s distance, time and if they want to post it to Strava. Along with a date if not scheduled workout.

The fourth feature is the recall of past workouts. Using the google calendar api we can see all past workouts. Storing the Strava id number in the google calendar event, we can recall the information of the workouts.

The fifth feature is Local Notification Sync. When users plan their workouts, they can opt-in to receive local notifications. This feature utilizes the Google Calendar API to create an event and then leverages the local storage to set up reminders. Users will receive alerts on their devices for upcoming workouts, enhancing their ability to stay on schedule.

The sixth feature is Activity Summaries. Our application integrates with the Strava API to provide users with a snapshot of their recent workouts directly on their dashboard. This summary includes key metrics such as distance, pace, and overall time, drawn from the latest synced data with Strava. By storing this data locally, users can access their activity overview quickly, offering insights into their performance without the need for constant data retrieval from a backend server.

# Work Breakdown Structure

# 

A screenshot of a diagram

Description automatically generated

For this running workout app, the project is split into eight big steps, like Project Planning, Design, Presentation, and others. Each step has smaller, detailed tasks to make sure every part of the project is covered. This includes choosing how to develop the software, putting the app out for people to use, and writing down how it was made. This careful organization helps keep track of everything and makes the work go smoothly.

# Project Timeline

A blue line with black text

Description automatically generated

Link to larger image <https://tinyurl.com/2z774545>

# Wireframes/Prototype

A screenshot of a computer

Description automatically generated

The design indicates functionality that combines activity tracking with calendar management to help users plan and track their running sessions and related activities. The interface employs a clean, inspired by ChatGPT, and structured layout with clear visual markers for ease of use.

# Prototype Elements

1. Calendar Integration Panel:

* A mini-calendar view is available for quick date selection.
* Users can click on a date to view or plan workouts.
* The calendar shows marked dates indicating scheduled or completed workouts.

1. Workout Scheduling Module:

* Interface for inputting new workout details, including date, time, and whether the workout is recurring.
* Option to store workout events in Google Calendar with a few clicks.
* A "Sync with Google Calendar" button to confirm and save the workout event.

1. Upcoming Events Viewer:

* A section where users can view a list of upcoming workouts and events, imported from Google Calendar.
* Each event entry may include the workout type, planned date, and time.

1. Strava Workout Input Form:

* A form to input workout details such as distance and time, with a checkbox option to post to Strava.
* An interface to select a scheduled workout or input a new unscheduled workout.

1. Past Workout Log:

* A log or history view that retrieves past workouts using the Strava event ID stored in Google Calendar events.
* Displays detailed workout information such as distance, pace, duration, etc.

1. Activity Summary Dashboard:

* A summary area showing recent workout statistics, utilizing data from the Strava API.
* Includes metrics like total distance run, average pace, and overall time.
* Graphical representations (like charts or graphs) for visual analysis of performance over time. Data Flow Diagrams

# Data Flow Diagrams

Data Flow Diagram Level 0

A diagram of a company

Description automatically generated

Data Flow Diagram level 1

A diagram of a company

Description automatically generated

The Data Flow Diagram (DFD) provides a visual representation of the data processing within the application. It illustrates how data flows through the system, from the initial page loading to the final storage of workout details in the database. The DFD for this application indicates the logical sequence of steps, such as fetching current location data, accessing workout history, and interacting with Google Calendar and Strava APIs.