



Extrasize

Get more from every workout.

Christopher Shannon

Brian Del Carpio

Ian Wu

Simon Fraser University

CMPT 276

Contents

| | |
|------------------------------|---|
| Project Overview | 3 |
| Kanban Methodology | 3 |
| Features | 4 |
| Code Testing | 5 |
| CICD Infrastructure | 5 |
| High-Level Data Flow Diagram | 6 |
| Challenges | 7 |
| Lessons Learned | 7 |
| Work Division | 8 |

Project Overview

Our project is centred 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 can address the challenges posed by our team's relative inexperience with large-scale projects by allowing for flexible task management and prioritization. The application was built with a robust technology stack, leveraging the interactive capabilities of React for the front end, coupled with the foundational web technologies HTML and CSS for structure and style. This combination provides our 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. Late in our build process, we discovered that Strava would not provide us with a map to display. Leveraging our flexibility, we implemented a third API, Leaflet, to be able to bring in what we believe is one of the highlights of the application, a visually appealing map. After struggling and attempting to use Heroku to manage our CICD process we had to resort back to using GitHub Actions alongside Docker (Since GitHub can not host servers), and aws ec2 to iteratively run our web application.

Kanban Methodology

Our intention originally was to make use of the Kanban Agile methodology. This was because it was an easy way to visualize which tasks need to be done, and which tasks are in progress or already done. We saw potential and researched it in the beginning stages of the project. The member in charge of carrying out these tasks was Melvin who dropped out of the course without notice. Because we did not know he dropped the course until the TA check-in, we carried out as best as we could through messages for the whole of Milestone 1. After the first check-in, once we realized Melvin was not going to be part of the group, our priority shifted to making up for the tasks that should have been done. Throughout the whole project, however, we did maintain good communication and used Discord as our makeshift kanban board. Although there was no visual representation of the board, tasks were laid out clearly along with their priorities.

High-Level Features


The goal of our application is to provide a way for athletes to plan and track their workouts. To achieve this, users can perform one of two actions. The first of these actions is to plan a workout. The second of these is to input a completed workout. Along with these features they need to be able to see the workouts that they planned and completed.

Google Calendar Features

The Google Calendar API allows access for our application to view and change the users' calendars. Using the API, we can create events that store the name, date, time, location and description. Once an event is created it can be seen in any location that calendar is viewed in.

Schedule

Schedule your workouts and keep track of your progress.



Please fill out the form below to schedule your workout.

Title *

Title should be at least 2 characters

Date *

mm/dd/yyyy

From *

--:-- --

To *

--:-- --

Location *

Location should be at least 2 characters

Description *

Description should be at least 2 characters

CANCEL

SCHEDULE

Another feature that the Google Calendar API provides is the ability to display calendars. In our application, we can use the Google Calendar API to display this calendar allowing the users to see the upcoming workouts. This allows the users to see the schedule for the coming weeks


| Today ◀ ▶ December 2023 ▾ | | | | | | | Week | Month | Agenda |
|---------------------------|--|--|-----|------------------------|------------------------|-----------------------|------|-------|--------|
| Sun | Mon | Tue | Wed | Thu | Fri | Sat | | | |
| 26 3:12pm test3 | 27 9am test 1 8:12pm running workout | 28 12:10am workout test 3:47pm test 4:06pm test | 29 | 30 4:09pm daily run | 31 4:15pm daily run | 1 8:54pm daily run | | | |
| 3 4:18pm daily run | 4 | 5 | 6 | 7 | 8 | 9 | | | |
| 10 | 11 | 12 | 13 | 14 | 15 | 16 | | | |
| 17 | 18 | 19 | 20 | 21 | 22 | 23 | | | |
| 24 | 25 | 26 | 27 | 28 | 29 | 30 | | | |

Strava Features

Connecting our application to the Strava API allows for users to upload their workouts to Strava. Users input the title, date, duration, distance, and location of the workout. The workout gets uploaded to their Strava account. Anyone who follows them on Strava can see their workout.

Record

Record your workouts and keep track of your progress.


Strava

Current Date: Thu Dec 07 2023(America/Los_Angeles)

Please fill out the form below to Record your workout.

Title *

T...

Title should be at least 2 characters

Date *

mm/dd/yyyy

Duration (seconds)

Distance (KM)

City

Country

State


CANCEL

RECORD

The Strava API allows for the recall of past events. Using this feature, we provide our users with a list of their past completed workouts. This allows users to compare their current results to their past to check for improvement.


WorkOuts

dailyworkout on 2023/12/8




Distance: 0.00Mil Speed: 0.00Mil/h
Duration: 1h 0m

test1 on 2023/11/28




Distance: 0.00Mil Speed: 0.00Mil/h
Duration: 1h 0m

daily running on 2023/11/24




Distance: 0.01Mil Speed: 0.00Mil/h
Duration: 2h 0m

weekend run on 2023/11/18



Distance: 0.01Mil Speed: 0.00Mil/h
Duration: 1h 23m

daily workout on 2023/11/17



Distance: 0.00Mil Speed: 0.00Mil/h
Duration: 1h 0m

Code Testing / CICD

We used GitHub as our version control software. This allowed for all to have the most updated version of the project across all our computers.

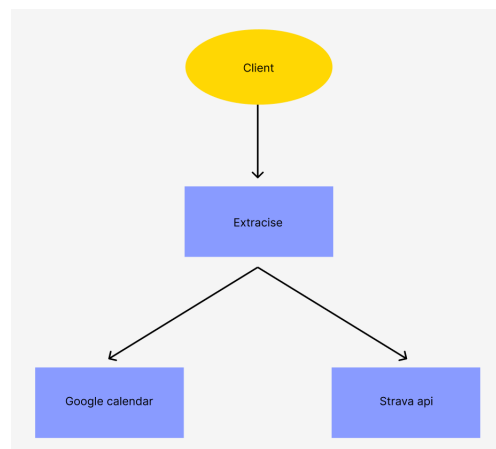
We created different branches for the different milestones. When these branches were complete a merge request would be made. Another member of the team would then review the code and merge to main if no mistakes/problems arose.

Once the merge is made the next step is testing. For this automation is key as if there was manual testing it could be skipped out of laziness. To solve this problem GitHub actions can help us. Using the workflows, we can create .yml files that automatically run sending the code in our repository to Docker.

In docker we have two containers one for the client and one for the server. Unit testing and integration testing occur in both. When the tests pass the client and server information is separately passed to AWS hosting services.

This leads to an up-to-date tested website anyone can access across the web. The one issue is the Strava API requires manual steps by each individual user to get connected to it. Thus leading to some friction in setting up the application for the first time.

High-Level Data Flow Diagram



[Link to larger image](#)

Our application (Center) is the link between the users' Google Calendar and Strava. A client will be able to interact with both apps through a single interface saving them time, and providing an easier way to schedule runs into their Google Calendar app.

Challenges

Through the completion of the project, our group went through many challenges. For Milestone 1, we had to deal with a group member who dropped out of the class without any communication with the team. Although this had many ripple effects, the biggest one was costing us the SDLC methodology.

Soon after Milestone 1, we found out that Strava would not provide us with a map to display on our website, this left us scrambling to find a third API to implement and connect to our application.

While trying to implement a CI/CD process, we had first switched from GitHub to Heroku, but soon found out we had to go back to GitHub, costing us a significant amount of time.

Finally, while trying to test our website from a user's perspective, we found that we had to implement a database to hold our users' accounts.

Even though our group was faced with multiple challenges, we persevered to deliver a useful application to both save peoples' time and facilitate their scheduling processes.

Lessons Learned

Scope of project:

From this project, we learned that we must plan and set in stone exactly what our project will do before we start working on any other steps of the planning stage. This will ensure that we don't fall victims of scope creep.

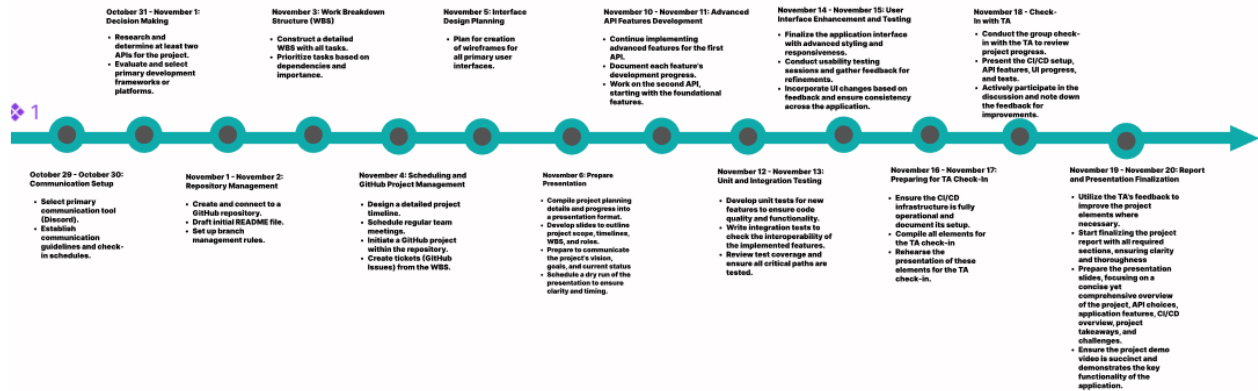
Kanban SDLC Methodology:

Perhaps the most important takeaway for our group was just how important following a proven SDLC methodology is. Although we did not purposefully ignore the creation of our Kanban board, we did try to follow one the best we could through our messaging platform. Safe to say, that we will always prioritize setting up an SDLC methodology in the planning phases of future projects.

Adaptability:

Through the many challenges (mentioned above) we went through to complete this project, our group learned the power of being adaptable when unforeseen circumstances arise. And although we are bound to encounter many more issues throughout our careers, we learned why agile practices are the most popular methodologies in the workplace.

Work Division



[Link to larger image](#)

Brian – Planning, Scheduling

M1 Report: Formatting, Proofreading, Overview, SDLC

M1 Presentation: Formatting, Editing video, Overview, SDLC

M2 Report: Formatting, Proofreading, Overview, SDLC, Lessons Learned, Challenges

M2 Presentation: Formatting, Proofreading, Overview

Ian – Programing, CICD, Testing

M1 Report: WBS, DFD, Project Timeline, Wireframes/Prototype

M1 Presentation: Wireframes/Prototype, DFD

M2 Presentation: Takeaways, Challenges, Video

Coding: Programming, CICD, Testing

Chris –

M1 Report: Project Features, User Stories, Technology Stack, APIs,

M1 Presentation: API, Application features

M2 Report: High-Level Features, Google Calendar Features, Strava Features, Testing, CICD

M2 Presentation: API, Features, CICD,