

ATHLEAT

Eat. Move. Feel.

Charles Richardson, Saketh Renangi, Preston Goren, Humberto Garcia

Advisor: Dr. Sumi Helal - helal@cise.ufl.edu

CIS4914 - Senior Project

Abstract

Understanding one's lifestyle is increasingly recognized as an important part of ensuring peak personal health. As a result of this, this project was built around the idea of developing a lifestyle tracking and analysis platform aiming to provide a comprehensive view of an individual's daily habits and activities. The main component of this platform is integrating user-generated data from a variety of sources. This integration provides the user with a complete picture of their physical activities, emotional well-being, and dietary patterns. A key feature of the platform is its data aggregation capability. Instead of viewing fragmented pieces of information, users get a holistic overview of their lifestyle, facilitating the interpretation and actionability of the data. The platform emphasizes the importance of visualization and reporting. With graphical dashboards, users can quickly grasp their lifestyle metrics, track their progress over time, and even compare their current performance against set personal goals. This visual feedback is invaluable for users to identify areas that need attention and improvement. In conclusion, the proposed platform offers a promising solution for individuals seeking a holistic approach to personal well-being, bridging the gap between disparate data sources and actionable insights.

Keywords

Web Architecture, Database Design, Object-Oriented Programming, UI/UX Design, Health Informatics, Data Aggregation, User Experience Design, Data Visualization, Lifestyle Tracking, Personal Health Management, Natural Language Processing, Macronutrient Analysis, User Authentication, Serverless Architecture, Frontend Development, Strava API, ChatGPT API, ReactJS, Amplify, Typescript, NextJS, Amazon Web Services, Lambda, DynamoDB

Introduction

Problem Statement

The contemporary lifestyle, characterized by hectic schedules and information overload, has made it increasingly difficult for individuals to manage and comprehend their health indicators effectively. The absence of a unified platform that seamlessly integrates exercise data, nutritional information, and health trends poses a significant challenge for individuals seeking a holistic approach to their well-being. ATHLEAT aims to bridge this gap by providing users with an intuitive and insightful tool to monitor and analyze their health metrics effortlessly.

Background or Related Work

In the realm of health monitoring, various tools exist, each focusing on specific aspects of an individual's health journey. Our project, ATHLEAT, draws inspiration from existing research and applications, seamlessly integrating exercise data from Strava and employing ChatGPT for personalized macro calculations to enhance user experience and efficacy, setting it apart from conventional health apps.

Solution Statement

ATHLEAT's solution revolves around creating a unified platform that simplifies the complex process of health monitoring. By seamlessly syncing with Strava to capture exercise data and employing ChatGPT for nutritional analysis, users can effortlessly input their daily meals and receive accurate macro calculations. The incorporation of trend graphs further enhances the user experience, providing a visual representation of their health journey over time.

Contribution

Our solution represents a step forward in the realm of health and wellness technology.

ATHLEAT builds upon existing technologies by integrating diverse data sources and cutting-edge natural language processing capabilities. This amalgamation not only streamlines the user experience but also sets the stage for a more personalized and adaptive approach to health monitoring. ATHLEAT's contribution extends beyond conventional fitness applications, marking a paradigm shift in how individuals engage with and interpret their health data.

Problem Domain

The problem domain for ATHLEAT lies at the intersection of health informatics and user experience design in computer science. It addresses the challenge of fragmented and uncoordinated health data, a common issue in personal health management. ATHLEAT integrates multiple data streams, including physical activity, dietary patterns, and emotional well-being, into a single, user-friendly platform. By employing advanced data aggregation and visualization techniques, along with natural language processing for enhanced interaction, the solution aims to provide users with a comprehensive and easily understandable overview of their health and lifestyle patterns. This approach seeks to bridge the gap between disparate health data sources and actionable personal health insights.

Literature Review

[Strava Documentation](#) – Strava’s documentation details and is an example of how to hook up to their API to begin working with Strava data.

[Benefits of Mood Tracking](#) – This blog post by the Calm app details the benefits of tracking one’s mood.

[Wrangling data from a variety of fitness apps and devices](#) – This New York Times article discusses aggregating data from the plethora of health/wellness monitoring apps and devices we have and provides some cool examples.

[API Management Best Practice](#) - An Article by Google Cloud discussing the design principles and common pitfalls when working with APIs

[ReactJS Documentation](#) - Guide to managing the application’s user interfaces

[NextJS Documentation](#) - Guide to managing the application’s front-end infrastructure and development

[Amplify Documentation](#) - Guide to managing the backend and its integration with the front-end

Solution

ATHLEAT aims to solve the problem of fragmented and uncoordinated health data by providing a central location to aggregate and visualize this data. AWS Amplify is used to facilitate the hosting, user authentication(aws Cognito), and data management(aws DynamoDB) of ATHLEAT. Serverless lambda functions are used to communicate between the front and backend as well as facilitate integrations such as Strava. OpenAI’s API is utilized to leverage LLMs to provide calorie and macronutrient estimates – this functionality is also contained within

serverless Lambda functions that retrieve a meal description from the form provided on the front end and store the resulting estimates in DynamoDB.

API Integrations

AuthenticateStrava Lambda Function

`handler(event, context)`

Handles an AWS Lambda function invocation to update a user's Strava access token and refresh the token in a DynamoDB table.

Inputs:

- ``event``: A dictionary-like object that contains data for the Lambda function to process. Expected to contain a key ``body`` which is a JSON string or dictionary containing keys ``id`` (user's identifier) and ``stravaCode`` (authorization code from Strava).
- ``context``: An object providing information about the invocation, function, and runtime environment. (Not used in the function, but standard in AWS Lambda handlers).

Outputs:

- A dictionary with two keys:
 - ``statusCode``: HTTP status code representing the outcome of the operation (200 for success, 500 for internal server error).
 - ``body``: A JSON string containing a message about the outcome ('User updated successfully' or an error message).

Process:

1. Parses ``event['body']`` to extract ``userId`` and ``stravaCode``.
2. Calls ``exchange_strava_token(stravaCode)`` to exchange the Strava code for access and refresh tokens.
3. Updates the user's record in the DynamoDB table with the new tokens, current date, and time for ``dateLastUpdated``, ``createdAt``, and ``updatedAt``.
4. Handles exceptions and returns an appropriate response.

`exchange_strava_token(stravaCode)`

Exchanges a Strava authorization code for an access token and a refresh token.

Inputs:

- ``stravaCode``: A Strava authorization code provided by the user.

Outputs:

- A dictionary parsed from the JSON response of the Strava OAuth token API. Typically contains keys such as ``access_token`` and ``refresh_token``.

Process:

1. Prepares a payload with the Strava authorization code, client ID, and client secret.
2. Make a POST request to the Strava OAuth token endpoint.
3. Checks the HTTP status of the response; raises an exception for non-200 status codes.
4. Returns the parsed JSON response containing the tokens.

AddActivities Lambda Function

`handler(event, context)`

Handles an AWS Lambda function invocation to fetch a user's activities from Strava and store them in a DynamoDB table.

Inputs:

- ``event``: A dictionary-like object containing data for the Lambda function to process. It should have a key ``body``, which is a JSON string or dictionary containing the key ``user_id``.
- ``context``: An object providing information about the invocation, function, and runtime environment. (Not used in this function but standard in AWS Lambda handlers).

Outputs:

- A dictionary with two keys:
 - ``statusCode``: An HTTP status code representing the outcome (200 for success, 500 for internal server error, 404 for user not found).
 - ``body``: A JSON string containing a success message or an error message.

Process:

1. Parses `event['body']` to extract the `user_id`.
2. Queries the DynamoDB `User` table to retrieve the user's information.
3. Check if the user's Strava token needs refreshing and refresh it if necessary.
4. Fetches activities from Strava using the user's access token.
5. Iterates over the fetched activities, formatting each activity's data and adding it to the DynamoDB `Event` table.
6. Each event item includes fields like `id`, `type`, `eventJSON`, `date`, `time`, `userID`, `createdAt`, `updatedAt`, `_lastChangedAt`, `_version`, and `_deleted`.
7. Handles exceptions and returns an appropriate response.

`refresh_strava_token(user)`

Refreshes a Strava access token using a user's refresh token.

Inputs:

- user: A dictionary representing a user, expected to contain at least the key `stravaRefreshToken` which holds the user's Strava refresh token.

Outputs:

- Returns a dictionary parsed from the JSON response of the Strava OAuth token API, usually containing keys like `access_token` and `refresh_token`.

Process:

1. Prepares a payload with the Strava client ID, client secret, grant type, and the user's refresh token.
2. Sends a POST request to the Strava OAuth token endpoint.
3. Checks the HTTP status of the response and raises an exception for non-200 status codes.
4. Returns the parsed JSON response containing the new access token and refresh token.

intakeChatgpt Lambda Function

`handler(event, context)`

Receives the results of the nutrition form from the front end, leverages chatGPT to provide an estimate of calories and macronutrients, and stores this data in our “events” DynamoDB table.

Inputs:

- ``event``: A dictionary-like object that contains data for the Lambda function to process. Expected to contain a key ``body`` which is a JSON string or dictionary containing keys ``id`` (user's identifier), ``food`` (text description of the user's meal), and metadata surrounding the date and time of the meal.
- ``context``: An object providing information about the invocation, function, and runtime environment. (Not used in the function, but standard in AWS Lambda handlers).

Outputs:

- A dictionary with two keys:
 - ``statusCode``: An HTTP status code representing the outcome (200 for success, 500 for internal server error, 404 for user not found).
 - ``body``: A JSON string containing a success message or an error message.

Process:

1. Parses ``event['body']`` to extract the ``user_id``.
2. Queries the DynamoDB ``User`` table to ensure the user exists.
3. Extract the meal description and call an OpenAI chat completion to retrieve the estimated calories and macronutrients
4. Parses the output of the OpenAI chat completion and organize it into the proper structure
5. Place all relevant data into an `"event_item"` variable and insert the event into the DynamoDB ``Event`` table
7. Handles exceptions and returns an appropriate response.

User-specific access

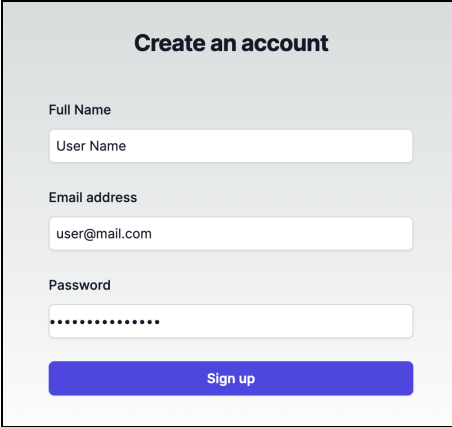
As an application that manages users' health information, we need separate datasets for each user. Using the Amplify Auth API, we simplify the implementation of AWS Cognito and delegate data isolation to the proven AWS infrastructure, thus, respecting user privacy and building a secure application.

User Registration (Sign-Up)

The user navigates to the sign-up page.

The user fills in the required information (e.g., email, password).

Submission and verification process.



Create an account

Full Name
User Name

Email address
user@mail.com

Password
.....

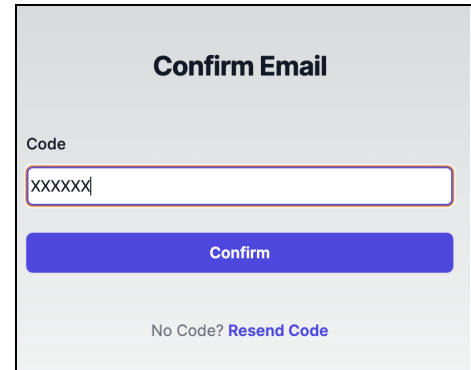
Sign up

User Confirmation Process

The user receives a code in their email account upon sign-up/sign-in.

The user inputs the received code on the confirmation page.

Validation and completion of the authentication process.



Confirm Email

Code

Confirm

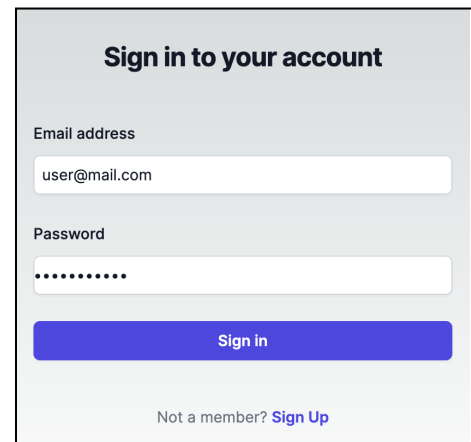
No Code? [Resend Code](#)

User Login (Sign-In)

The user accesses the sign-in page.

The user enters login credentials.

Authentication and redirection to the user dashboard.



Sign in to your account

Email address

Password

Sign in

Not a member? [Sign Up](#)

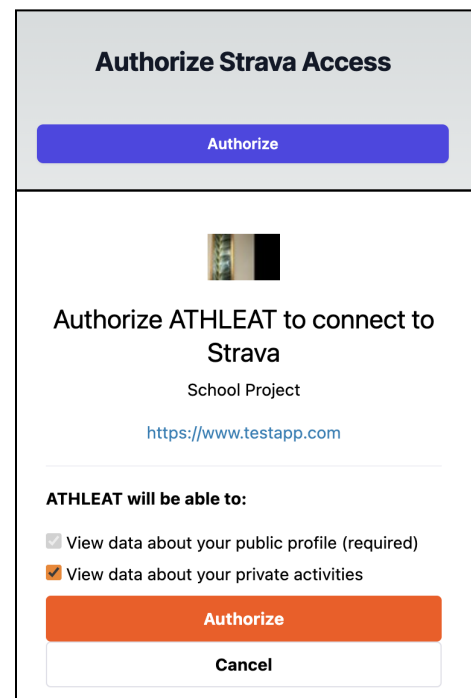
Strava Authorization (Data Integration)

The user opts for Strava integration.

The user is redirected to the Strava authorization page.


The user consents to data sharing and is redirected back to the application.

On redirect, the Strava authentication process is run, detailed in the [Authenticate Strava](#) section above, which is succeeded by an initial data synchronization as outlined in the [Add Activities](#) section



Authorize Strava Access

Authorize



Authorize ATHLEAT to connect to Strava

School Project

<https://www.testapp.com>

ATHLEAT will be able to:

☐ View data about your public profile (required)

☒ View data about your private activities

Authorize

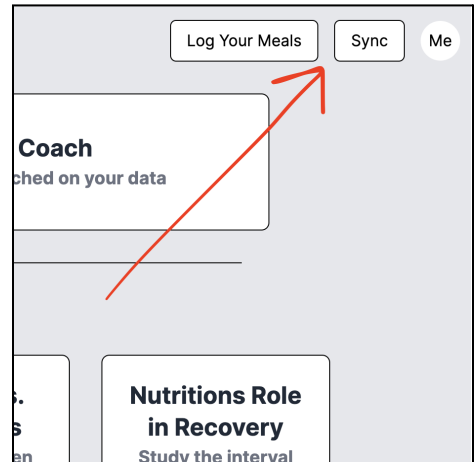
Cancel

User-generated Data

Activity

Remotely resynced with Strava on demand via the **Sync** button on the home page. This button is connected to [AddActivities](#).

```
{
  date: string;
  time: string;
  activityType: string;
  duration: number;
  calories: number;
  distance: number;
  pace: number;
};
```



Nutrition

Nutrition data is manually input via the form, which is connected to [intakeChatgpt](#).

```
{
  date: string;
  time: string;
  food: string;
  calories: number;
  carbs: number;
  fat: number;
  protein: number;
}
```

A screenshot of a 'Nutrition Form'. It has a title 'Nutrition Form' and a 'Now' button. Below the title, there are three input fields: 'Date:' with a calendar icon, 'Time:' with a clock icon, and 'Food:'. The 'Date' field contains '12/02/2023', the 'Time' field contains '09:23:34 PM', and the 'Food' field contains 'A regular chicken tender Publix sub & 12oz Coke'. At the bottom, there is a 'Submit' button.

Mood

Mood data is manually input via a form, which is connected directly to the database as there are no additional processing needs.

```
{
  date: string;
  time: string;
  moodIndex: number;
}
```

A screenshot of a 'Mood Form'. It has a title 'Mood Form' and a question 'How are you feeling right now?'. Below the question, there is a horizontal slider with a red dot in the middle. To the right of the slider, the word 'Neutral' is displayed. At the bottom, there is a 'Submit' button.

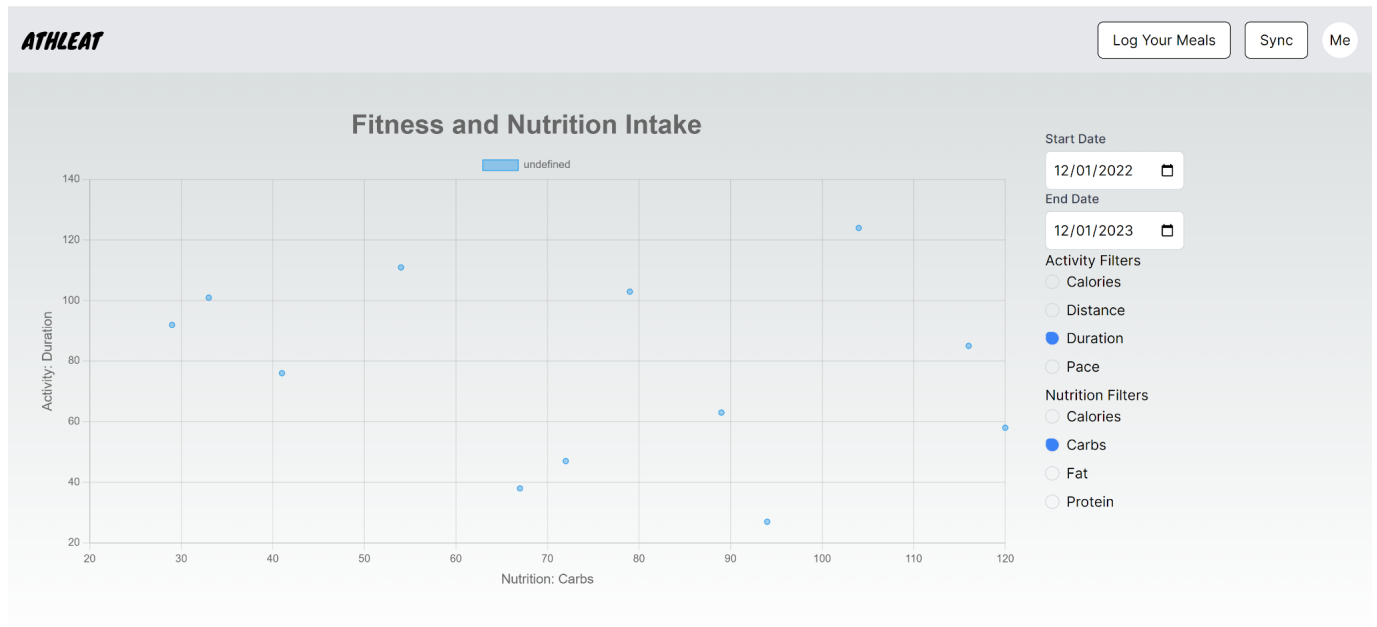
Results

By combining, aggregating, and visualizing data from multiple lifestyle-related sources, we can display insightful charts that can reveal helpful trends in the relationships of data sources.

Trends

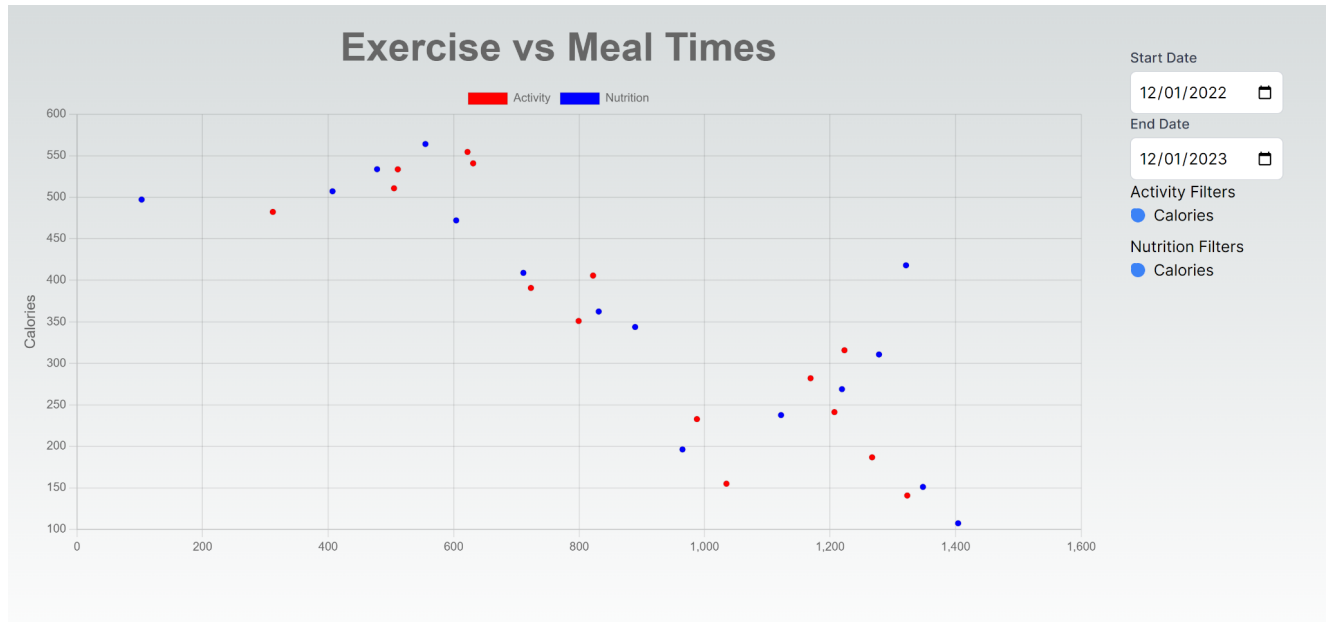
Intake

This trend graph shows the user's intake vs output data. They can swap between various filter options to get a full picture of how their consumption affects their exercise.



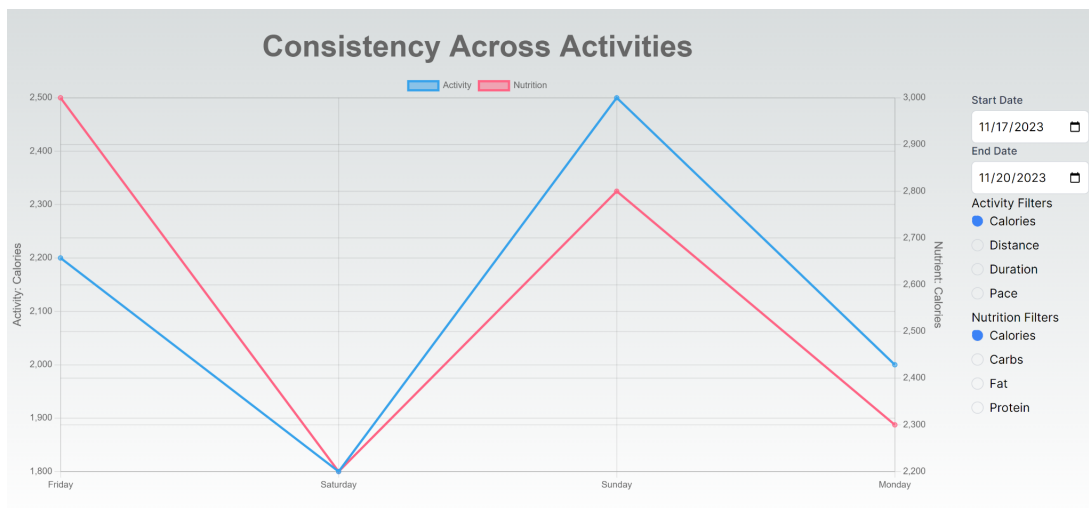
Timing

Shows the user at what times they typically eat and exercise as well as how much of each they do at that time. The x-axis is the minute of the day



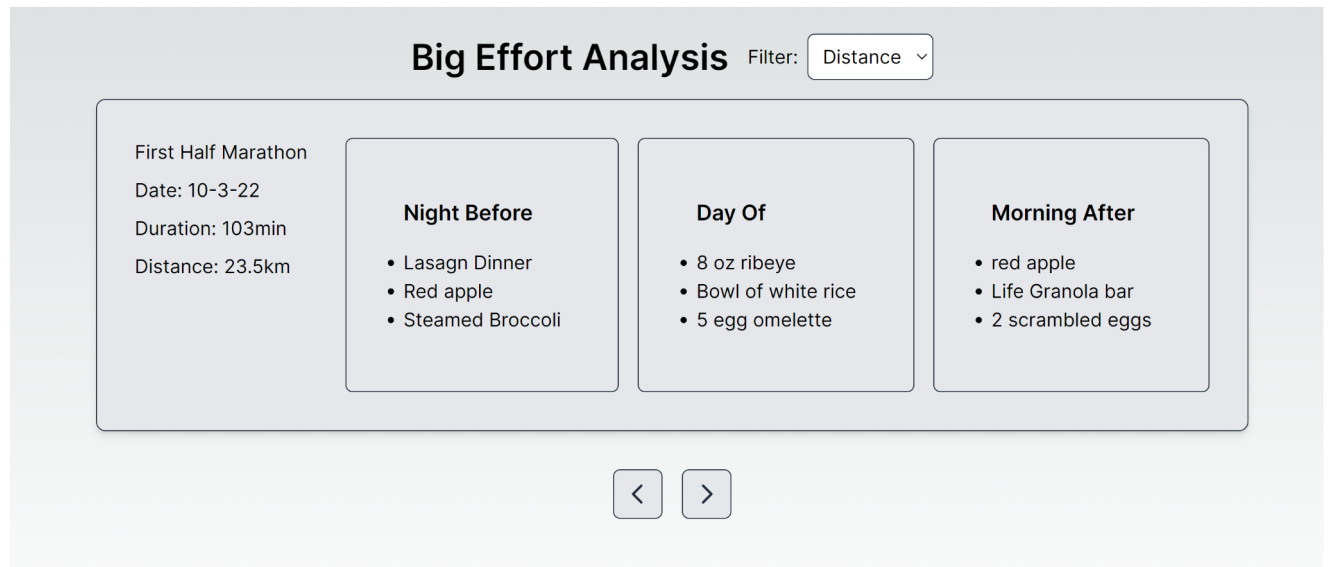
Consistency

Shows the correlation of frequency/duration of your workouts with meal/macronutrient consumption over time.



Big Effort Analysis

This trend page accumulates the five most intense workouts of a user by category (eg if calories are selected it will select the five workouts that burned the most calories). It then shows specific information about those workouts and the food that was eaten before, the day of, and after.



Conclusions

Our senior project team created ATHLEAT, a lifestyle tracking and analysis platform that aims to provide a comprehensive view of an individual's daily habits and activities via aggregated health data and easily digestible data visualization. We integrated our project with Strava (a common application used to track exercise/activity) to make data tracking easier for users as well as developed our entry methods (including Large Language Model powered macronutrient estimation) for other domains of data. ATHLEAT serves as a hub for these various data sources and can provide users with a centralized location to investigate their health data and discover trends that can help them optimize their well-being and performance.

Throughout developing ATHLEAT, there were many lessons learned by our team. We discovered the importance of keeping the user in mind when designing User Interfaces and data visualizations. With multiple sources of data, it is easy to come up with many charts to display to the user, but it is important to be able to understand which of these charts will be most useful and easy to digest for a user. We learned how to plan out, design, and then implement a complex project with several moving parts – including user authentication, database, management of the flow of data, serverless lambda functions, hosting, and user interface design. Throughout the project, our team also learned how to communicate more effectively, brainstorm collaboratively, and in general more productively work as a team.

Although promising, ATHLEAT is not without deficiencies and potential for future work. A true “one-size-fits-all” centralized hub for health data aggregation and analysis would require much more engineering and resources. ATHLEAT has limited sources for data integration, which means for example that someone already needs to be using Strava to integrate their exercise data with ATHLEAT. Future work could involve expanding accepted data sources, allowing the user to create more complex visualizations with their data, providing insights based on the user’s aggregated data, and perhaps even direct integration with wearable health metric trackers such as Apple Watch.

Acknowledgments

The group extends their gratitude to Professor Sumi Helal for his invaluable guidance and unwavering support throughout this project. His expertise and insightful feedback have been instrumental in shaping both the direction and success of the work. The entire group is deeply appreciative of the time and wisdom Professor Helal has imparted.

Biography

Charles Richardson is a project manager and software engineer focused on AI-integrated systems. He works for two startups with AI-driven products offering Web and mobile interfaces. Charles' passion lies in creating value for users with efficient, mobile solutions. Outside of his technical career, Charles likes to go fast using human-powered mechanics.

Humberto Garcia is a passionate software engineer with a primary focus on front-end development and data science. Specializing in Preact, Typescript, and Python, he contributed significantly as a Software Engineering Intern at GE Digital. Humberto is driven by a desire to blend creativity and technology in crafting user-friendly interfaces. Beyond school and work, Humberto enjoys playing the guitar and tennis.

Preston Goren is a Computer Science and Philosophy student specializing in Machine Learning and Artificial Intelligence. Previous internships include AnswerRocket and RallyHere and include experience with data analysis, data visualization, ETL processes, NLP, LLMs, and RAG systems. Preston enjoys mountain biking, slacklining, perusing nature, and gaming.

Saketh Renangi is a skilled software engineer with a strong focus on backend development and system integration. His internships at Surescripts and Pearson have seen him make significant contributions to optimizing data processes and developing scalable software solutions. Outside of his academic and professional pursuits, Saketh is an avid sports enthusiast, enjoys investing, and engages in strategic games of poker.

