

# Portfolio

Projects for Software Engineering & Data Engineering

**Jui Shah**

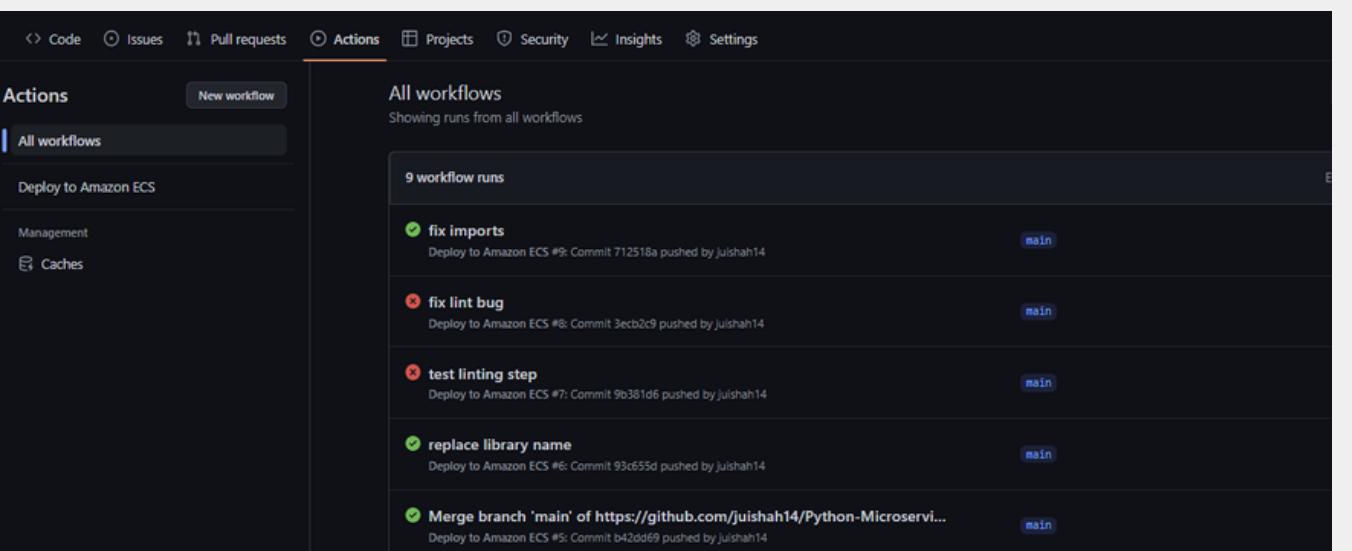
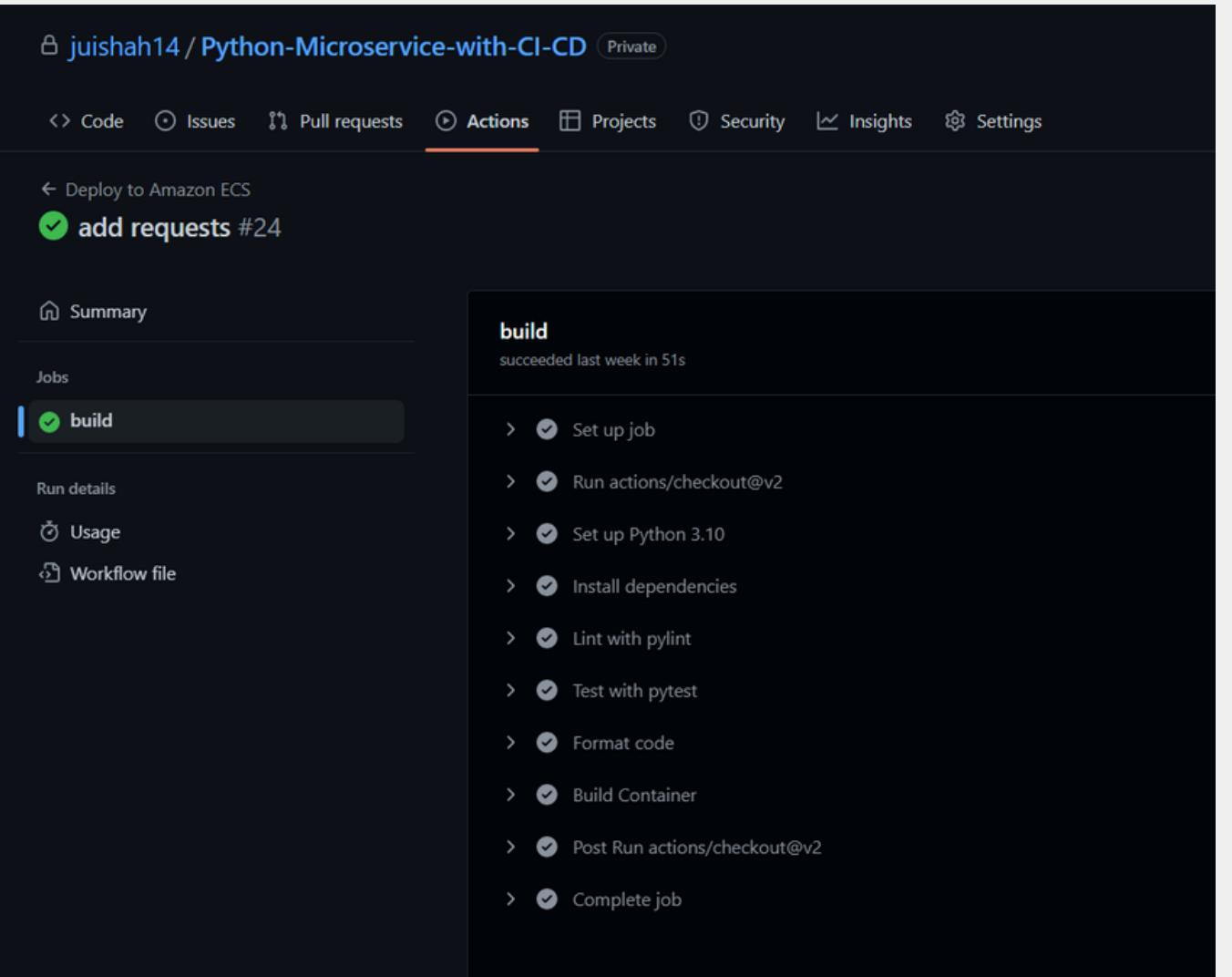
Software Engineering, University of Waterloo

# Python Microservice with CI-CD

## Tools: Docker, AWS, Fast API

With this microservice, users can retrieve information on NASA and its various missions and scientific discoveries, as well as perform a search of the NASA Image and Video library.

This containerized PaaS microservice has been designed such that it can be scaled to create a production-level Dev-Ops workflow and infrastructure, equipped with unit and end-to-end tests, formatting and lint checks, and continuous deployment of its container to AWS.

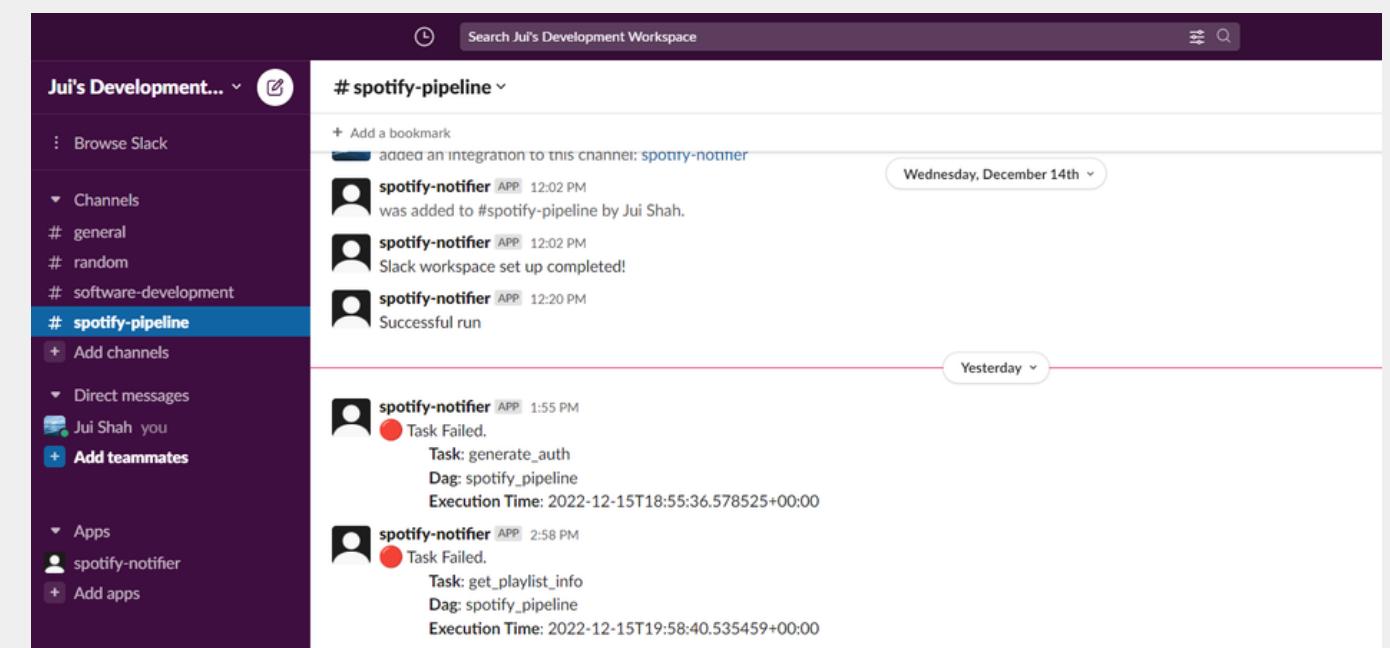
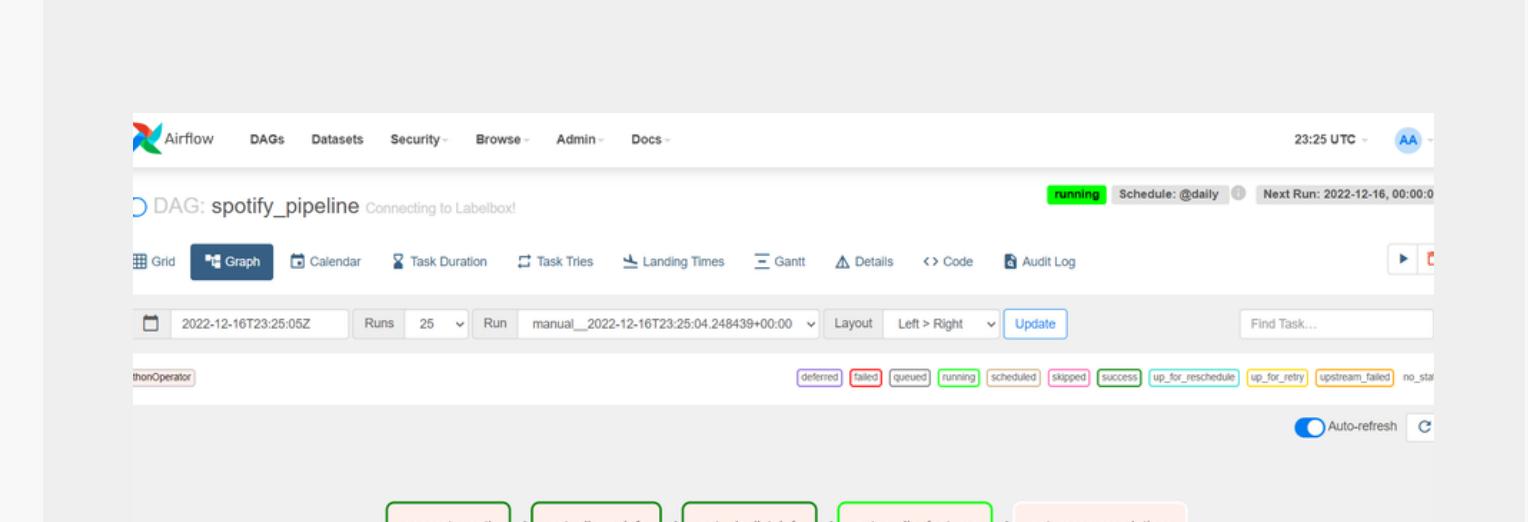


# Spotify ETL Pipeline

## Tools: Airflow, Docker, Slack, Spotify API

This pipeline was designed using Apache Airflow and Docker to programmatically generate song recommendations using the Spotify API, as well as inform the user of any failures via Slack.

It extracts data on your favourite playlists, albums, and tracks on Spotify, and processes its key features (artist, tempo, energy, etc.) to generate song recommendations, with a bias towards songs with higher valence and 'danceability' scores. I was pleasantly surprised with the outcome and have loved many of songs recommended!

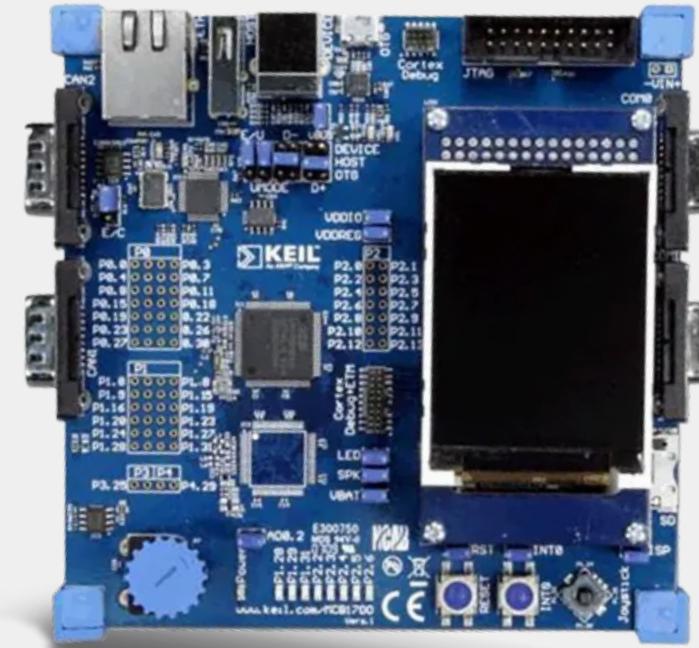


# Real-Time Executive (RTX)

# Tools: C, Keil Board, NXP Microcontroller

As part of the lab for SE350 (Operating Systems), our group designed a small real-time executive (RTX) and implemented it on a Keil MCB1700 board populated with an NXP LPC1768 microcontroller.

The executive provided a basic multiprogramming environment, with five priority levels, preemption, simple memory management, message-based inter-process communication, a basic timing service, and system console I/O and debugging support. The RTX is also suitable for embedded computers which operate in real time.



```
C k_process.c 2, M X
manual_code > Context_Switching > src > C k_process.c > scheduler(void)
170
171     int process_switch(PCB *p_pcb_old)
172     {
173         PROC_STATE_E state;
174
175         state = gp_current_process->m_state;
176
177         if (state == NEW)
178         {
179             if (gp_current_process != p_pcb_old && p_pcb_old->m_state != NEW)
180             {
181                 p_pcb_old->m_state = RDY;
182                 p_pcb_old->mp_sp = (U32 *)__get_MSP();
183             }
184             gp_current_process->m_state = RUN;
185
186             U32 ctrl = __get_CONTROL();
187             ctrl &= ~BIT(0); // clear bit 0, want to be at priv. level when exit from the kernel
188             __set_CONTROL(ctrl);
189             __set_MSP((U32)gp_current_process->mp_sp);
190             __ rte(); // pop exception stack frame from the stack for a new processes
191         }
192
193         /* The following will only execute if the if block above is FALSE */
194
195         if (gp_current_process != p_pcb_old)
196         {
197             if (state == RDY)
198             {
```

# Costco Management Platform



## Tools: Golang, MongoDB

This backend for a Costco Management Platform was designed using Golang and Mongo DB. It features user authentication using JWT, database access using Mongo DB, and handles HTTP requests, route management, and JSON validation using Gin Gonic.

This CRUD API keeps track of product inventory, orders, invoices, account holders, and their memberships (Business, Executive, or Gold Star).

Costco Login

Quality Goods & Services

Email\*  
Enter your email

Password\*  
Enter your password

Remember me [Forgot password?](#)

[Login](#)

Not registered yet? [Create a new account](#)

Costco Register

Manage Inventory Efficiently  
Enter your details to create an account and begin your journey with Costco Wholesale!

First name  
Enter your name

Last name  
Enter your last name

Email  
Enter your email

Phone no.  
Enter your phone number

Password  
Enter your password

I agree to all terms, privacy policies, and fees

[Sign up](#)

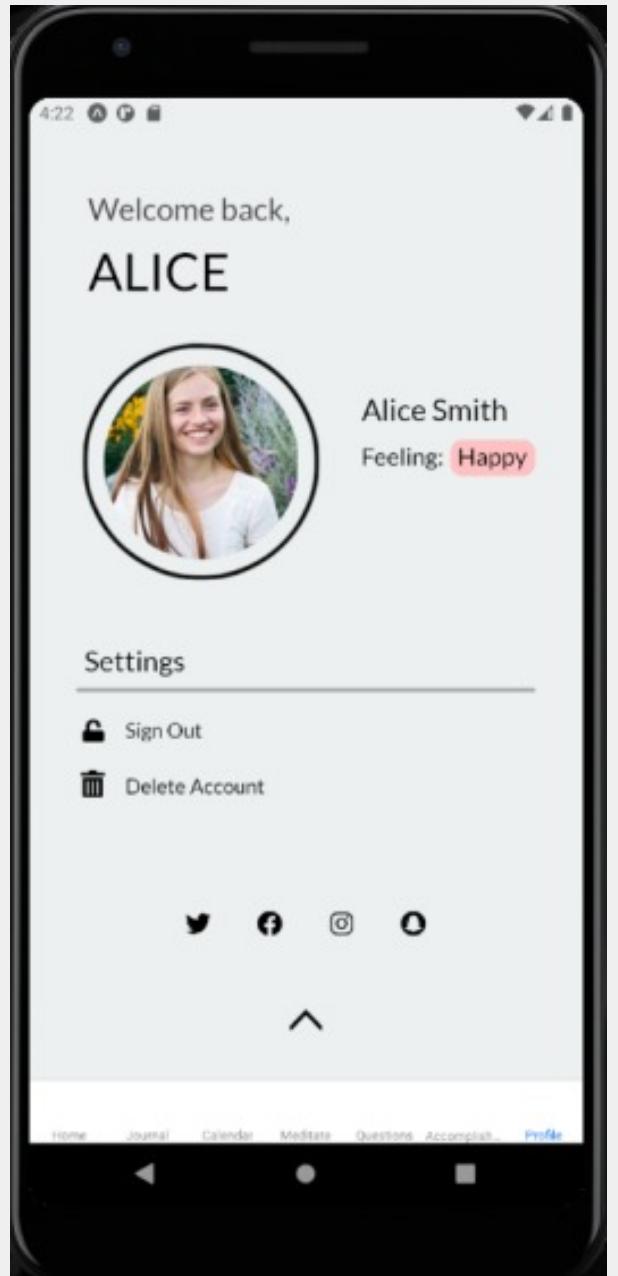
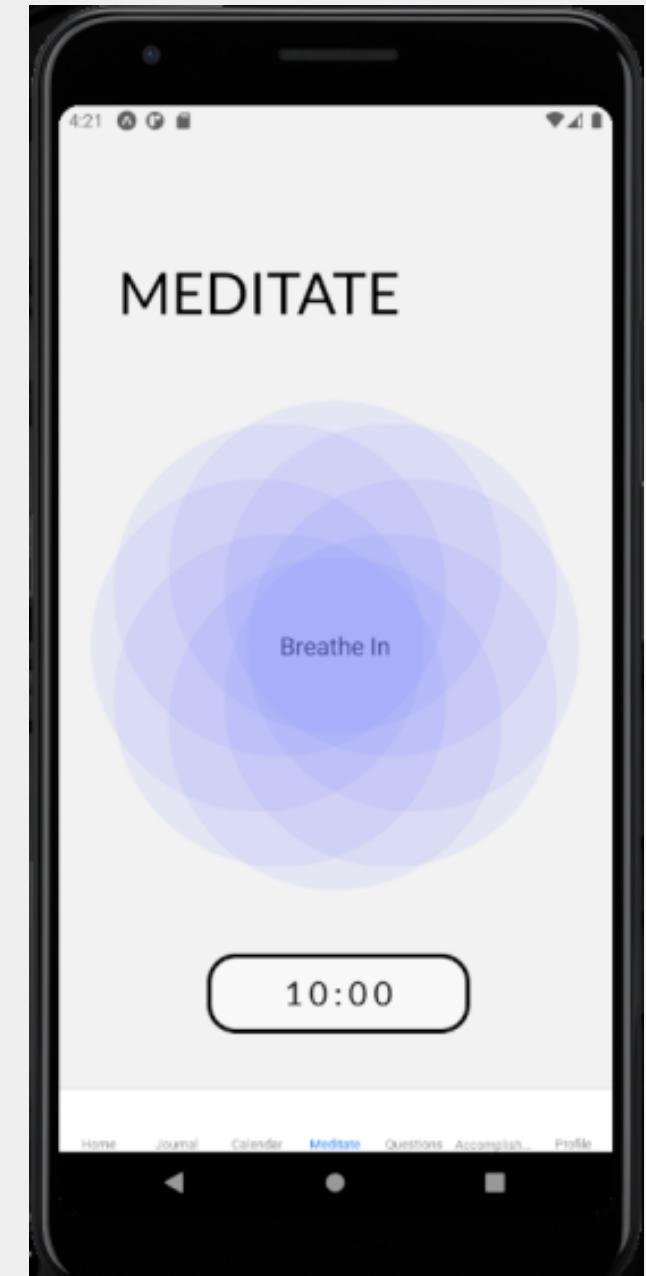


# Mental Health App

## Tools: React Native, MongoDB, Node.js & Express.js

"Happy Sapling" is an app that encourages mindfulness and daily self-reflection. When the user first enters the app, they're greeted with a sapling, which slowly grows over time as the user continues to use the app. The user has the opportunity to reflect on their day via daily questions, journal entries, and a timed meditation bubble, as well as celebrate their successes via an 'Accomplishments' tab.

Through the Mood Calendar, the user can also have a birds' eye view of how they were feeling that month, through the mood colour they logged when answering the mindfulness questions.



# Personal Website

## Tools: Javascript, HTML, CSS

To showcase my projects and past experience, I designed a personal website using Javascript, HTML, and CSS. With this project I was able to learn how to implement a frontend and iterate on the designs I implemented.

I added various features, such as a Light & Dark mode, a carousel demonstrating my recent projects, and a timeline to demonstrate my past experiences & extracurriculars. I aimed to implement a clean design, and hope to add to this website in the future.

Jui Shah

Home About Skills Portfolio Projects Contact Me

### Hi, I'm Jui Shah

Here's who I am and what I do

Hi, I'm Jui, a second-year Software Engineering student at the University of Waterloo.

I'm passionate about Data Science and Data Analytics, and I recently had the pleasure of interning at [IROC](#) as a Data Analyst/Software Developer.

I really look forward to learning more about this field, so if you'd love to chat, please feel free to connect or message me!



### Experience

Past Work and Extracurricular Experience

Work Education

- Software Engineering University of Waterloo Sep 2020 - Present
- IBM Data Science Professional Certificate Coursera June 2020 - Aug 2020
- UoM Python for Everybody Specialization Coursera June 2020 - Aug 2020

### Skills

- Data Scientist
- Software Engineering
- Data Analyst
- Frontend Developer
- Power BI 80%
- SQL 75%
- Azure Databricks Notebooks 70%
- Jupyter Notebooks 75%

### Projects

Take a look at my recent projects



Happy Sapling  
An all-in-one mental health, meditation, and goal-fulfilling React Native mobile app.  
Tools: React Native

< View More >



# Comic Book API

---

## Tools: Golang, GraphQL, MongoDB

Inspired by comic books and Marvel, I used Golang, GraphQL, and MongoDB to create a CRUD API to keep track of comic books, its characters and authors, as well as the location and details of comic book stores all around the world.

This project gave me a chance to learn and familiarize myself with GraphQL and use it with a backend to store information in a MongoDB database.

A screenshot of a GraphQL playground interface. The query is:

```
mutation CreateComicBook($input: CreateComicBookInput!) {
  createComicBook(input:$input) {
    _id
    name
    description
    publication_date
    author
  }
}
```

The variables are:

```
input: {
  name: "The Amazing Spider-Man",
  description: "Abandoned by his parents and raised by an aunt and uncle, teenager Peter Parker is trying to sort out who he is and his new superpowers.",
  publication_date: "May 3, 2002",
  author: "Stan Lee"
}
```

The response shows the created comic book:

```
{ "data": { "createComicBook": { "_id": "638ebbd73ba69ae6949c9b0", "name": "The Amazing Spider-Man", "description": "Abandoned by his parents and raised by an aunt and uncle, teenager Peter Parker is trying to sort out who he is and his new superpowers.", "publication_date": "May 3, 2002", "author": "Stan Lee" } }}
```

A screenshot of a GraphQL playground interface. The query is:

```
mutation CreateCharacter($input: CreateCharacterInput!) {
  createCharacter(input:$input) {
    _id
    name
    description
    comic_book_series
    superpower
    danger_level
  }
}
```

The variables are:

```
input: {
  name: "Peter Parker",
  description: "Sometimes student, sometimes scientist and sometimes photographer, Peter Parker is a full-time super hero better known as Spider-Man.",
  comic_book_series: "The Amazing Spider-Man",
  superpower: "Shoots spider webs",
  danger_level: 7
}
```

The response shows the created character:

```
{ "data": { "createCharacter": { "_id": "638eb5223ba69ae6949c9ad", "name": "Peter Parker", "description": "Sometimes student, sometimes scientist and sometimes photographer, Peter Parker is a full-time super hero better known as Spider-Man.", "comic_book_series": "The Amazing Spider-Man", "superpower": "Shoots spider webs", "danger_level": 7 } }}
```

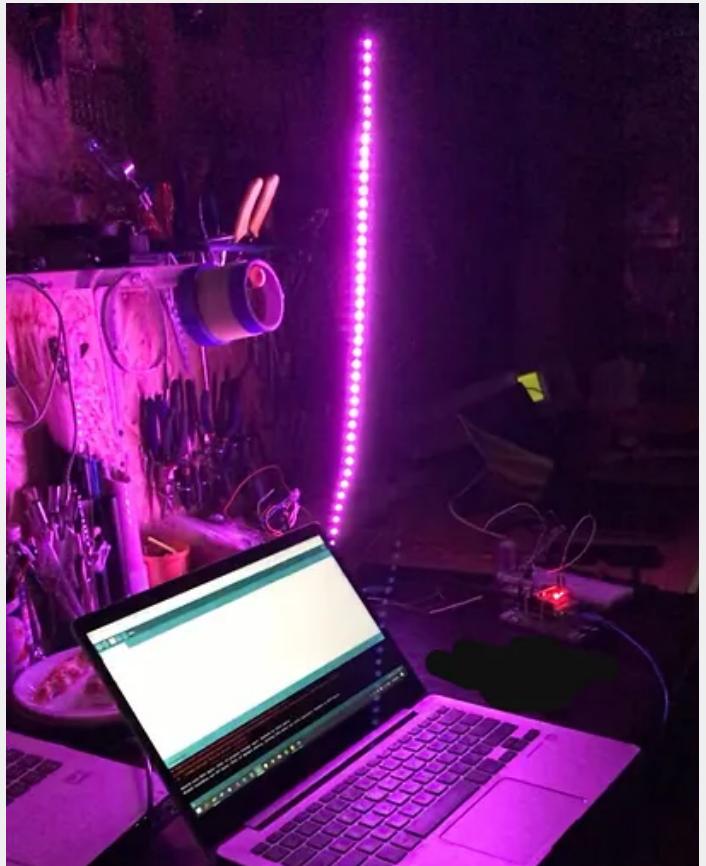
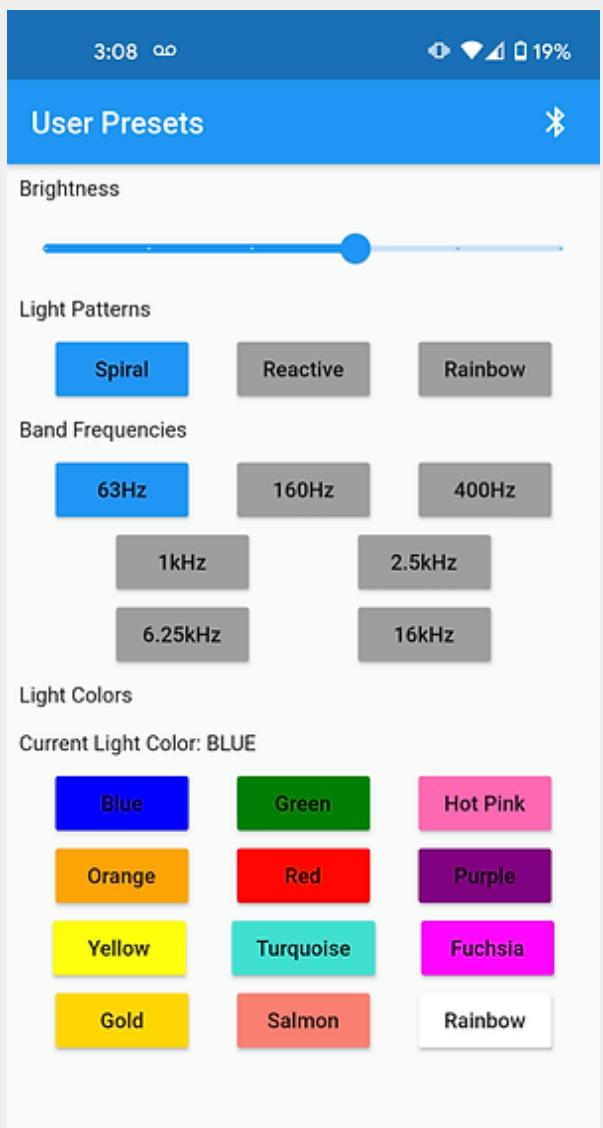


# LED Music Visualizer

## Tools: Arduino, Flutter

This LED Music Visualizer allows an LED light strip to instantaneously react to music played by the user. Its hardware consists of an Arduino microprocessor, a Bluetooth module, and a Spectrum Shield.

The Bluetooth-connected app, created using Flutter, provides an interface for the user to select from 3 presets and 12 colours for the LED strip to display, as well as set the brightness level of the LEDs. The 'Reactive' preset allows for the user to select from 7 audio frequency bands (eg. Bass) for the LED strip to react to.



# Data Science & ML

---

## Tools: Python, Jupyter Notebooks

In order to learn the basics of data science, data analysis, and machine learning, I completed various courses offered by IBM on Coursera. These courses gave me the opportunity to use Python (Pandas, Numpy, Seaborn, Matplotlib) to clean and visualize data, as well as build ML regression models.

I later used my learnings from these courses to build various classifiers, such as a Titanic Passenger Survival classifier and a Car Price classifier. All the labs from the IBM courses, as well as the above classifiers, can be found on my Github.

