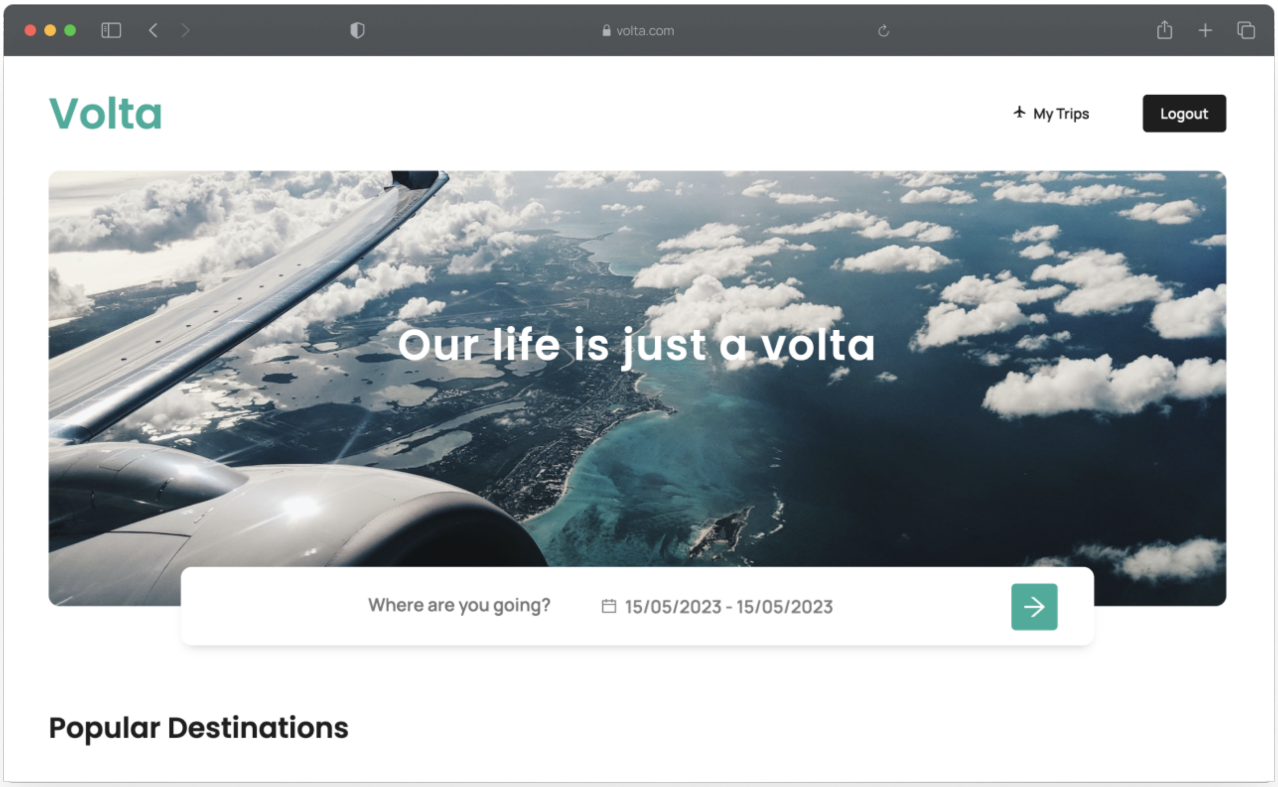
# CFG Degree Group Project

## Travel App: VOLTA



## Spring 2023 Cohort

Adriana Balbuena, Fahima Alim, Gina Kondyleniou, Urslar Madden and Anna Clyde

# Page of Contents

**3** **1. Introduction**

3 1.1 Aim

3 1.2 Objectives

3 1.3 Project Overview and Background

**3** **2. Specifications and Design**

3 2.1 Functional Requirements

4 2.2 Non-functional Requirements

4 2.3 Design and Architecture

**4** **3. Implementation and Execution**

4 3.1 Development Approach and Team Roles

6 3.2 Tools and Libraries

**7** **4. Testing and Evaluation**

7 4.1 Testing Strategy

8 4.2 System Limitations

8 4.3 Evaluation

**9** **5. Conclusion**

**9** **6. Acknowledgments**

## 1) Introduction

### 1.1 Aim

Volta is a Travel app that aims to make travel planning effortless for users by providing them with a tool to create a well-organised itinerary for their holiday. The app enables users to explore new destinations, attractions, and activities and add them to their itinerary. In addition, users can save their favourite activities itinerary for future use.

### 1.2 Objective

The ultimate goal of Volta is to simplify the travel planning process and offer users a hassle-free way to create their dream holiday itinerary, enabling them to make the most of their time abroad, whilst also budgeting.

### 1.3 Project Overview and Background

In order to use the web app it is mandatory to have a registered user. Once registered and logged in, the user can start looking for trip recommendations by entering the destination city and start and end dates on the home page. Once on the results page, the registered user can save their favourite recommended  activities and view them on the 'My Trips' page by clicking 'See Activities' on the newly created trip. It is important to note that the trip is created automatically by clicking the green arrow on the home page, as long as there are activities available for the required destination. The user can also delete all the data associated with that trip, or individual activities from the 'See Activities' page.

## 2) Specifications and Design

### 2.1 Functional Requirements

1. User Registration: The app must offer user registration functionality, so that users can create a personal account and access the app's features.
2. Destination Search: The app should enable users to search for destinations that match their interests, date preferences, and budget.

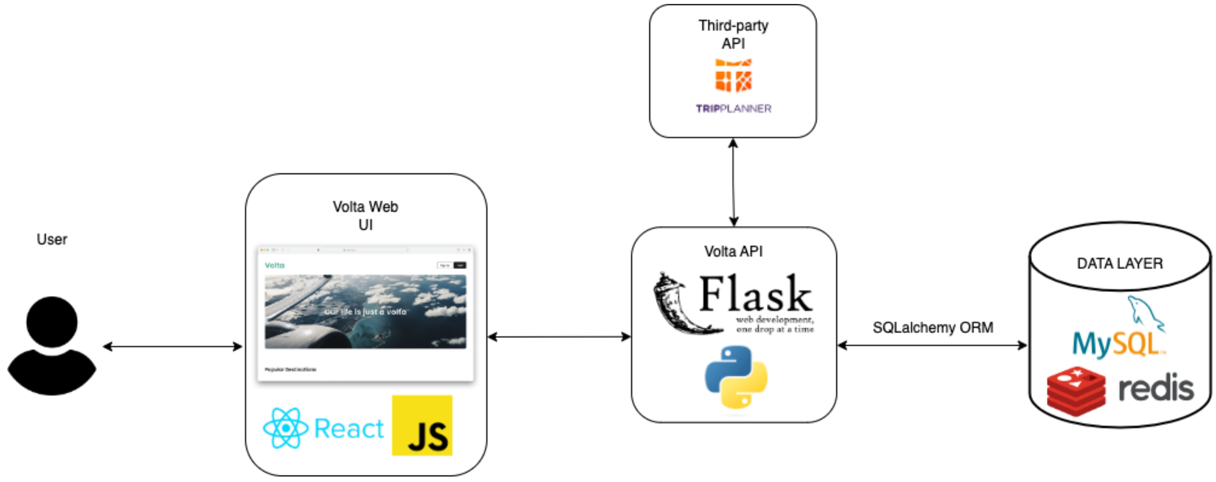
3. Itinerary/Activities Creation: Users must be able to create their travel itinerary by adding destinations, attractions, and activities using the app.
4. Saving Favourites: The app should provide users with the ability to save their preferred destinations and activities for future reference.
5. Deleting Trips: Users must be able to delete their trips data.
6. Deleting Itinerary/Activities: Users must be able to delete individual itineraries.

## 2.2 Non-functional Requirements

1. The app should have a simple and be easy to navigate, to provide a positive user experience.
2. The app should be reliable, fast, and available for use at all times.
3. The app should be designed to handle a growing number of users and data without affecting its performance or causing the app to crash.
4. The app should be compatible with different devices and operating systems, to ensure accessibility for a wide range of users.
5. The app should be easy to maintain and update to address bugs, errors, and other issues promptly.
6. The app should be capable of integrating (in the future) with other travel-related services such as booking platforms, maps, and social media.

## 2.4 Design and Architecture

The Volta architecture is composed by a flask API acting as a backend, which interacts with our MySql database using SQLalchemy, and the frontend developed using React JS, which consumes the 11 endpoints provided by our API through http requests.

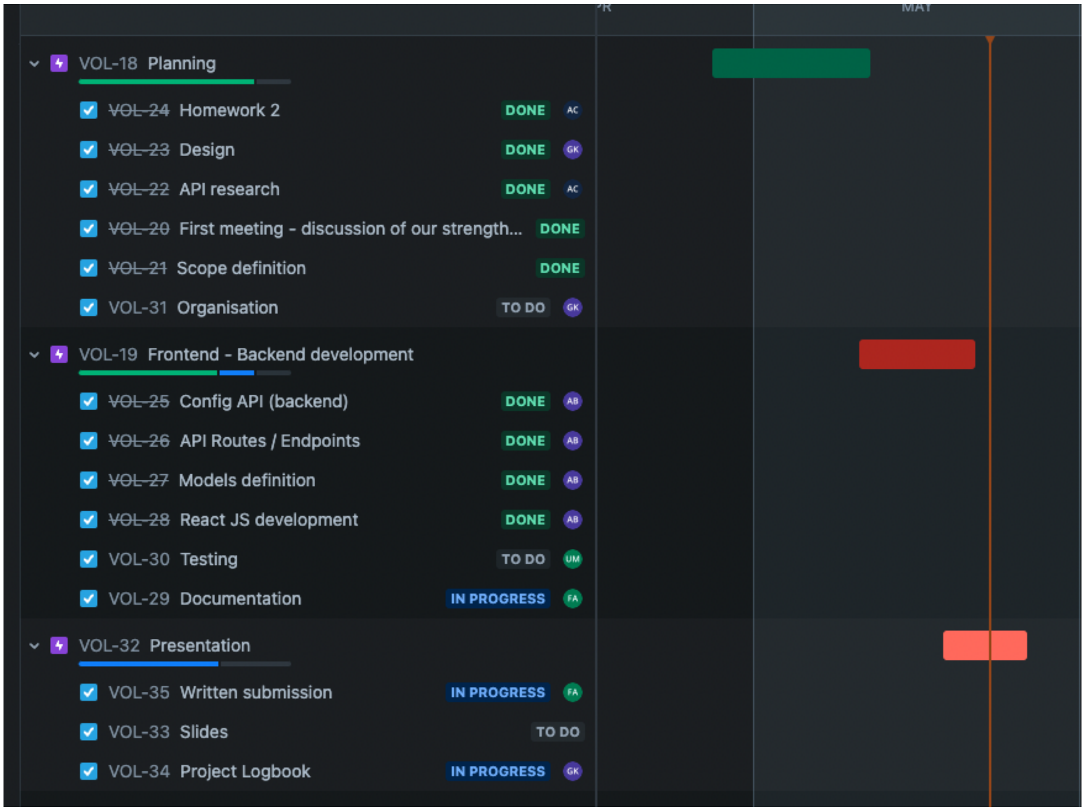


## 3) Implementation and Execution

### 3.1 Development Approach

In creating Volta, we chose to work with agile methodologies, specifically scrum, because it prioritised flexibility, collaboration, and responsiveness to change.  We broke down the project into smaller tasks called user stories, and organised them using Jira's project management tool. We had regular meetings to discuss progress and challenges, making sure everyone was working together. This iterative process enabled us to incorporate feedback and make adjustments as needed, resulting in a travel app that meets the needs of our users.

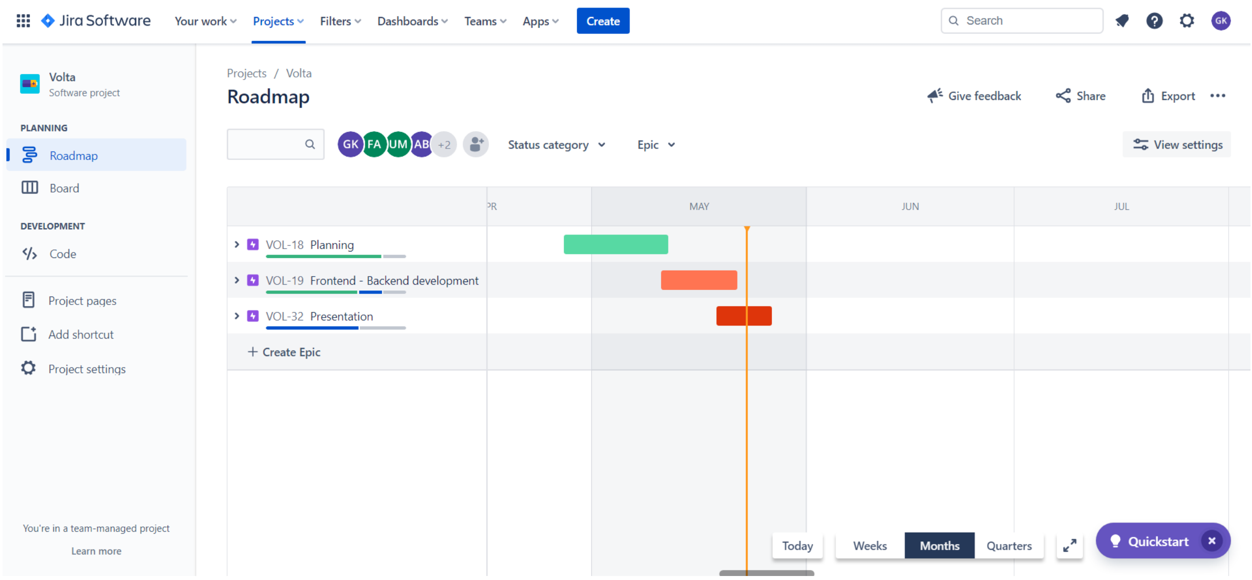
We had 3 sprints, the first about planification, the second about development and the third about documentation:



VOL-18 Planning
- ☑ ~~VOL-24~~ Homework 2 DONE AC
- ☑ ~~VOL-23~~ Design DONE GK
- ☑ ~~VOL-22~~ API research DONE AC
- ☑ ~~VOL-20~~ First meeting - discussion of our strength... DONE
- ☑ ~~VOL-21~~ Scope definition DONE
- ☑ VOL-31 Organisation TO DO GK

VOL-19 Frontend - Backend development
- ☑ ~~VOL-25~~ Config API (backend) DONE AB
- ☑ ~~VOL-26~~ API Routes / Endpoints DONE AB
- ☑ ~~VOL-27~~ Models definition DONE AB
- ☑ ~~VOL-28~~ React JS development DONE AB
- ☑ VOL-30 Testing TO DO UM
- ☑ VOL-29 Documentation IN PROGRESS FA

VOL-32 Presentation
- ☑ VOL-35 Written submission IN PROGRESS FA
- ☑ VOL-33 Slides TO DO
- ☑ VOL-34 Project Logbook IN PROGRESS GK



VOL-18 Planning

VOL-19 Frontend - Backend development

VOL-32 Presentation

Pictures above demonstrate how Jira was used to delegate tasks to team members, as well as showing dates.

## Team Roles:

Project Manager: Gina Kondyleniou

Technical Lead: Adriana Balbuena

Researcher: Anna Clyde

Documentation: Fahima Alim

Testing: Urslar Madden

During the development of Volta, our team encountered several challenges. Time constraints proved to be a significant hurdle, as we had to balance the project's scope with limited resources and tight deadlines. Coordinating group meetings and ensuring everyone's availability was another challenge, considering our different schedules and commitments. Communication and collaboration were crucial, as we had to align our ideas and work together efficiently. Additionally, managing and prioritising tasks effectively within the project management tool required careful attention, to avoid confusion and maintain productivity. Despite these challenges, our team remained resilient and found ways to overcome them. Using our combined knowledge and ability to adapt, we effectively delivered the app.

### 3.2 Tools and Libraries

**Third-party API:** AI Trip Planner API Documentation (nabeeldev1340) I RapidAPI

We decided to use a trip planner API for Volta, which utilises artificial intelligence to generate customised travel plans based on the user inputs - the location and dates of the trip. It produces a detailed itinerary that includes recommendations for tourist attractions, local experiences, and accommodations tailored to the user's preferences. By leveraging this API, users can quickly and easily plan their trips without the need for extensive research, allowing them to focus on enjoying their holiday instead. The travel itineraries produced by the system are flexible and adaptable, enabling users to modify them as needed, to match their requirements. One team member purchased this API from https://rapidapi.com, providing her with an API key; this enabled us to create HTTP requests to the API endpoint with our relevant parameters, being the location and the dates.

**Flask**: we used flask to build the API.

**SQLalchemy:** we used the SQLalchemy (which is an Object Relational Mapping) flask integration to work with tables, sql statements and the mysql db connection programmatically.

**Flask_session:** we used flask session to support our server-sided session and manage sessions. In a web application, a session allows you to store user-specific data on the server and associate it with a session ID. This allows you to maintain user state and store information across multiple requests.

**Bcrypt:** we used bcrypt to encrypt and check the users passwords.

**Python dotenv:** we used dotenv to manage environment variables such as our secret key and the API key.

**PyMysql:** we used pymysql to connect to our mysql database.

**Redis:** we used redis as the session store because it provides a more scalable and persistent storage solution. Redis's in-memory nature ensures fast access to session data, and its persistence options allow session data to survive server restarts.

**Postman:** we used postman to test our API endpoints in a user-friendly interface.

**React JS:** we used react to develop the frontend.

### 3.3 Implementations process and challenges

One of the faced challenges was finding a free API to work with, since our instructor suggested amazing ideas to integrate in our app, most of the APIs to achieve that were paid, so we had to pay for at least one to make our app more interesting.

Developing a minimally useful app in three weeks was definitely another challenge, even though we already knew that we were trying to do something bigger and riskier than just the main project requirements, but thanks to the organisation and good communication in our group it was possible and we learned a lot about coding, new tools and roughly what it's like to be a developer in real life.

## 4) Testing and Evaluation

### 4.1 Testing Strategy

Our testing objective was the API endpoints, (using unittest, mocks and postman):

- Login: /login
- Register: /register
- Logout: /logout
- Get current user: /@current_user
- Get plan (from external api): /get_plan?days={ number of days}destination={ city destination}
- Add user trips: /trips
- Get user trips: /trips
- Delete user trip: /trips/{trip id}
- Add user plan: /trips/{enter trip id}/plan
- Get user plan from db (saved plans): /trips/{trip id}/plan
- Delete user plan from db (saves plan): /trips/{ trip id}/plan/{plan id}

The test coverage  was primarily planned for the happy path: success responses, with a response status code of 200, and edge cases of each endpoint. Unfortunately due to lack of time and organisation in this part, we were only able to deliver the unit test for the authentication endpoints (login, register and logout) and the get plans from external api endpoints.

### 4.2 System limitations

- Only registered users can save trips and activities, but non-registered users can still make a request to get a plan from the external api without the functionalities of saving.

-The third-party API accepts any word as a 'destination', and provides recommendations even for non-existent places, that is the reason why we are using a general error handling there.

- To test the API endpoints in the provided postman collection it is necessary to disable (commenting) the lines 24 and 25 of the config.py file in the volta-backend repository.

- To test the frontend correctly it may be necessary to uncomment the mentioned lines to set the cookie properly.

### 4.3 Evaluation

In evaluating our travel app, we have identified areas where further improvements could be made. If given the opportunity to expand the app, we would consider incorporating additional features to provide more comprehensive and specific information to users. One such feature would be integrating weather forecasts for chosen travel dates, as long as the dates are not too far in the future. This would enable users to plan their activities and pack accordingly. Additionally, including information on accommodation options, transportation details, and food recommendations would offer users a more holistic travel planning experience. By expanding the app's functionality in these areas, we can further enhance the user's journey and provide a valuable resource for their travel needs.

Furthermore, as part of our evaluation of the travel app project, we have considered the possibility of introducing an alternative user experience. To cater to users who prefer not to create an account, we could explore implementing a guest account option. This would allow individuals to access and utilise the app's features without the need for a formal account registration process. By providing this option, we aim to accommodate a wider range of users and ensure a seamless and inclusive experience for all.

## 5) Conclusion

Testing posed some challenges during the development of our travel app. Specifically, we encountered difficulties related to mocking data and defining proper assertions. Despite our efforts, we faced persistent errors that proved difficult to resolve. As a result, we were only able to successfully implement 4 tests for the app, out of 11. While this limitation was not ideal, we focused on ensuring the functionality and stability of critical features. Moving forward, we recognise the importance of refining our testing processes to improve the overall quality and reliability of the travel app.

However, our team exhibited strong collaboration, effective communication, and mutual support throughout the development of Volta. We actively shared knowledge, motivated one another, and maintained a high level of professionalism. As a result, we successfully created and delivered the Volta app, showcasing our collective dedication and expertise.

## 6) Acknowledgements

We would like to express our sincere gratitude towards each member of our project team, for their support and motivation throughout the duration of this project. Our efforts and collaborative spirit have been pivotal in the successful completion of this project.

Additionally, we extend our thanks to Amy, Youri, Stavros, and Kam for their continued assistance.