**Project Planning Report**

Daniel Doan

Mitra Rouhipour

Maryam Faraji

**Repository:**

**https://github.com/doda2391/group5\_final\_project**

**Table Of Content**

[Overview 3](#_Toc99791879)

[SDLC Model 3](#_Toc99791880)

[High Level List of Implemented features 4](#_Toc99791881)

[Application Features Per each API 4](#_Toc99791882)

[Tests Associated with the code (Unit and Integration Test) 4](#_Toc99791883)

[CI/CD infrastructure (Test, Build and Deploy) 5](#_Toc99791884)

[High Level Data Flow Diagram 6](#_Toc99791885)

[Takeaways 6](#_Toc99791886)

# Overview

For this project, team members decided to design a webpage that gets the weather condition based on a city and country from user input from an [Weather API](https://openweathermap.org/api) and a Shoe store in which each shoe has a name, a price and description from a [Shoe API](https://rapidapi.com/kaushiksheel9/api/shoes-collections/). To implement this project, 3 phases have been considered including project preparation, planning, and execution. As implementing this kind of project is new for the team members, the team decided to follow the Agile methodology to have the flexibility to change each design based on the requirements.

The requirements changed over time in the projects from the planning phase into the developing phase. In the planning phase, we needed a web application that suggested the clothes based on the weather, however, the team members were not able to find an appropriate clothes API, therefore the requirement of the project changed based on the new API, the shoe API, that the team selected. Based on the new requirement, the scope of the project changed, and the team decided to select new features for the shoe API. Finally, because there were 3 members in the team instead of 4, the number of features has been decreased from 6 features to 4 features.

# SDLC Model

Other reasons that team members decided to choose Agile as the System Development Life Cycle for the project are as follows:

* Agile has more flexibility rather than plan-driven models. In plan-driven models, requirements are defined, and after planning and setting requirements, the implementation step will start. But, In the Agile model, requirements can alter in any phase of the project.
* There is regular feedback and pair working in Agile model which leads smoothly delivering the project.
* The Agile model is so popular in the industry, and it is a good practice for a future career path.

# High Level List of Implemented features

* Getting the weather based on the city and country entered by user
* Displaying a weather icon based on the current weather
* A toggle switch to sort shoes’ prices ascending or descending.
* Identify items eligible for sale and assign a 20% discount in front of their price.

# Application Features Per each API

**Weather API Features**

* A form to ask input from the user for the city and the country with a button named get Weather. Whenever it is clicked it shows the weather and the second API will be triggered.
* Showing the Icon of the current weather based on specific range in the API documentation.

**Shoe API**

* Order the list of shoes based on the price through toggle bottom
* A sale feature to identify shoes items more expensive than the average of the price of a list of shoes and show 20% discount for them.

# Tests Associated with the code (Unit and Integration Test)

**8-unit tests:**

Test the output of function calCelsius: we pass the function a sample temperature to see if it works properly or not

* test if the output of shoes API function exists
* test if the function that changes the product order is called
* test if the data is received from the shoes API
* test the validity of the city input: we check if put a valid city name it works properly
* test the validity of the country input: we check if put a valid country name it works properly
* test if the weather API is called successfully
* test the validity of set state function

**4-integration tests:**

* test that when the value is inputted and button is clicked, the function getWeather is called to trigger the API
* test if the icon is selected correctly when displaying icon feature
* test if the order state is switched when the function SwitchesGroup is called
* test if the items that less than the average price is sale off

# CI/CD infrastructure (Test, Build and Deploy)

The web application has been built using Travis and Heroku as the infrastructure to implement CI/CD. At First, a test file is created with Jest and Enzyme tools to test the application. When there is a pull request to make a change to the main branch, Travis is triggered to run the test file - continuous testing, build the app based on the continuously changing source code - continuous integration, and deploy the app to Heroku which is connected to Traverse through API key - continuous deployment. On our last deployment, the app was tested and built successfully, however during the deploy phase, there are remaining errors that prevent the app from being displayed. Using heroku.log and investigating the Travis details, we have navigated the problem; yet the team doesn’t have enough time to afford fixing it and expect to work more on it in the futures.

# High Level Data Flow Diagram

**Diagram

Description automatically generated**

# Takeaways

It was the first time that the team was using Agile and React.js for working with an API. Therefore, all steps in the project were new to them.

The primary requirement of the project was using an clothes API to be able to sort clothes based on the gender and category and relate it to the weather API. In action, the team tested several clothes API, and the APIs did not work, therefore we selected a shoe API, and the team decided the weather API and the shoe API do not have any connection to each other.

Team members collaborated with each other twice per week, through zoom meetings, to talk about the progress of the projects, what they have done and what is needed to do.

The team used a [tutorial](https://www.youtube.com/watch?v=IxuqmfO6p28) for the weather API, and also a [git](https://github.com/erikflowers/weather-icons) repository to use icons.

The team members had several challenges during different stages of the project as following:

* Find a clothing API which finally changed to a shoe API and the scope of the project.
* All the requirements of the project were new to us. So, we were supposed to do some research about topics, watch videos, test them on the system to be able implement it on our project.
* The time constraint

Fixing bugs in different steps of developing and testing the project

* stress management
* time management
* The bright side of project:
* team members were responsible and collaborated in a good way.

The team learned following lessons:

* Having regular Scrum meeting are so important
* Time is a key
* It’s better to write the test codes during the development and don’t postpone it until the end.
* A high quality of team working can decrease the work pressure and stress level.

In the bellow there is the list of tasks and the member in charge of it.

|  |  |
| --- | --- |
| **Task** | **Assigned to** |
| Choosing the APIs | Maryam, Mitra, Daniel |
| Developing the react app | Maryam, Mitra |
| Tests | Daniel |
| CI/CD | Daniel |
| Deployment | Daniel |
| Report | Mitra, Maryam, Daniel |
| Creating the Presentation | Daniel |
| Demo of the APP | Mitra, Maryam |
| High level flow Diagram | Mitra, Maryam |