20/09/2022

Isaac Naylor

Jet2

Synoptic Project

Holiday Chat Bot

Contents

[Design 2](#_Toc114668357)

[Requirements. 2](#_Toc114668358)

[Data. 2](#_Toc114668359)

[Wireframes. 2](#_Toc114668360)

[Use Cases. 3](#_Toc114668361)

[Development 4](#_Toc114668362)

[Implementation 4](#_Toc114668363)

[Data 4](#_Toc114668364)

[Project Setup. 5](#_Toc114668365)

[Testing 6](#_Toc114668366)

[Unit Testing 6](#_Toc114668367)

[Manual Testing 6](#_Toc114668368)

# Design

Before deciding how I would like the user interface to look I would like to get an abstracted list of requirements.

## Requirements.

* The chat bot must ask minimum 2 questions
* The chat bot must feedback to invalid input from the user
* The chat bot must provide a set of recommended holidays

## Data.

I have decided that I would like to load my dataset from a JSON format. This is because I find the JSON format easier to read and write, both myself and from code.

## Wireframes.

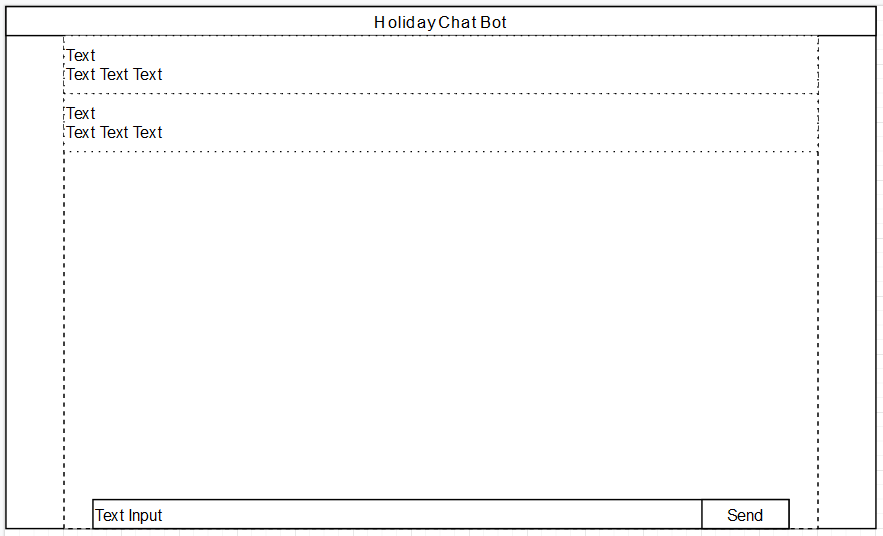
Figure 1 Shows the basic Wireframe that I would like to create my application to look like.

Figure 1: Chat Bot Wireframe

I have decided that the most logical way to implement the Minimal Viable Product is by having it be a full-page application, however I think if this were to be incorporated into a larger website, it could also be shrunk down to be a small component in the corner of the screen.

The wireframe in figure 1 shows the trail of conversation the user has had with the chat bot, with the top row of text being used to show who has sent a message (Chat bot or User), with the bottom row of text representing the message sent by either party.

## Use Cases.

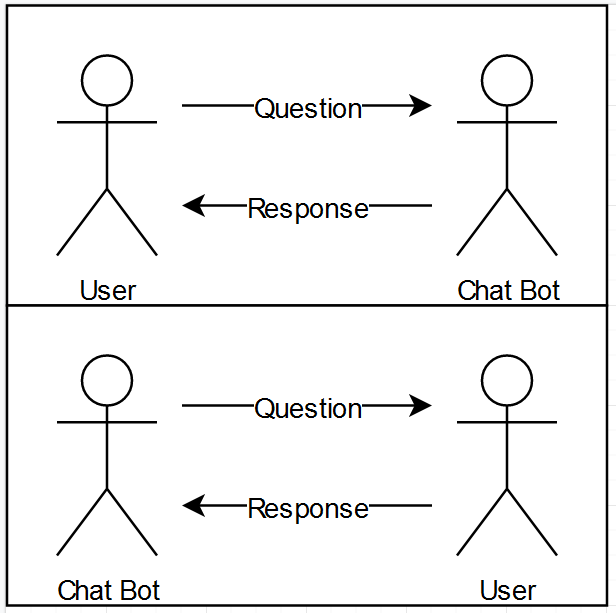


Figure 2: Use Case Diagram

The main use cases for this application are for the user to ask the bot questions to get a holiday recommendation, or the user to be asked questions by the bot and replying to be given a holiday.

# Development

## Implementation

I have decided to implement this project using the Python programming language. Within Python I am using a web development library called Flask. This is because I am familiar with Python and Flask and I think that they suit this sort of quick turnaround style of development.

I am also going to be using the Bootstrap CSS framework in order to get a good looking final product without having to spend a lot of time concerned with small details.

## Data

As stated earlier, I decided to use JSON to store the holiday data. I chose to do this because it is a standard way of sending and receiving data on the web, and it is easy to read both from a human and code perspective.

Below is a screenshot of one of the data sets converted to JSON.



I wrote and ran a small python script which read in the data csv file and converted it’s contents to JSON. This script is included in my project (“csv\_to\_json.py”)

## Project Setup.

The setup for my project is relatively simple.

The folder configuration is as follows:

.env – This is a local python environment for running the project

.pytest\_cache – This is generated by the testing library that I am using

.vscode – This is a local configuration for my development environment

app – This is where most of the code for my application will be

app/static/data – This is where I am storing the holidays.json file, and where any other similar data oriented files may go in the future

app/templates – This is flask convention for storing HTML files

instance – This is a folder created to store any data required by the application while running

SDT Project C – This contains the project brief, the original CSV data file, and the confirmation file for the Synoptic Project.

tests – This folder contains all the tests I will write for my application.

# Testing

## Unit Testing

For this project I used 2 python libraries, Pytest and Coverage. Pytest is a library used to write and run tests, whereas coverage is used to generate code coverage reports. Using these 2 libraries together allowed me to write unit tests for my code and receive a code coverage report, which allowed me to reach 100% code coverage.

## Manual Testing

Manual Testing is a big part of testing as it allows me to experience what the end user will experience and find any issues I find with the functionality, as well as the usability.

My Manual tests have been documented in the Excel file named “Manual Testing”

# Conclusion