Project Proposal: My Traffic Wizard

Professor: Dr. Mike Mireku Kwakye

CSCI441: Software Engineering

Group: Team A

Jan 28, 2024

Repository URL: https://github.com/fhsu-csci441-team-a/myTrafficWizard

Team A Group Members:

Nicole-Rene Newcomb

Tyler Anderson

Philip Baldwin

Jacob Spalding

Team Personnel Profiles

Nicole-Rene Newcomb: Strong collaborator, brainstorming facilitator, and troubleshooter with moderate project management and organization skills. Primary programming languages include: Python, Java, C++, JavaScript, SQL, HTML, and CSS. Familiar with Kotlin, as well as WordPress and Android App development.

Tyler Anderson (Team Leader): Skill set includes Python, T-SQL, Java, and Azure Data services, complemented by a solid foundation in data modeling, database design, and data integration (APIs). Proficient in technical writing - documenting and communicating complex technical concepts via diagrams and text.

Philip Baldwin: In my current role, I field daily questions from customers about errors in the reports and services we provide. To answer, I must interpret gps and location data for vehicles, and cross-reference it against map data to determine the cause. Some of the technical knowledge I must employ on a daily basis includes the use of SQL, direct data retrieval via API calls, and virtual private server (VPS) administration. As a senior at FHSU, I have made use of programming languages (C++, Java, Python) as well as web development technologies (HTML, CSS, Javascript) to complete projects.

Jacob Spalding: Before going to FHSU I got a trade school degree in computers and networking. I work at the University of Kansas Health System as a PACS Admin. In my current role I support over 135 workstations for the Radiologists at the Hospital, satellite locations, and in their homes. At Fort Hays State University, I have used C++, JavaScript, Python, SQL, HTML and Java.

Introduction, Key Importance, and Context

This project seeks to increase the accessibility of critical travel information by providing users with the ability to receive schedulable traffic and weather alerts through multiple channels (SMS, email, Discord, and Slack). This is particularly important for users who live in extreme weather areas for whom standard traffic estimates may be insufficient. In potentially hazardous environmental conditions, it is especially important for drivers to be aware of whether they will need to budget extra time to ensure safe driving for the conditions. While many drivers use apps for directions to new locations, most don't use such apps for familiar routes. This may result in drivers not being aware when dangerous road conditions arise, such as when a sudden heavy rain or freeze occurs that can increase the risk of accidents.

As driving is safest when drivers don't feel rushed and drive at an appropriate speed for the current conditions, our proposed program can play an important role in helping users budget enough time to travel safely when weather conditions require a reduction in usual speed. Making it easy for drivers to estimate their drive time given current traffic and weather conditions should help prevent them from being late and being more tempted to drive at unsafe speeds. Making essential traffic and weather information available via a user-friendly GUI interface while also providing API functions that allow information to be delivered directly to a user's Discord or Slack channel can help improve the safety and time-management of travel. It also simplifies drivers' routine by allowing them to access this vital information through a familiar chat app without needing to install a dedicated app.

Problem Diagnosis

Severe weather conditions can quickly result in unanticipated dangerous driving environments. Due to the seasonal and hard-to-predict nature of these types of dangers, drivers don't always remember to plan their routes to account for them. Drivers who fail to anticipate a longer commute time during such conditions can then be at an increased risk of accidents due to exceeding safe speeds for the conditions. Many drivers don't pull up an app every day before their usual commute, so this project aims to make it easier for users to get daily updates and estimates of the time needed to ensure they will be able to get to where they need to go safely. Users able to receive daily route alerts to their email, SMS, Discord, or Slack will be able to quickly see if their travel will take longer than usual without needing to install a dedicated app.

Proposed Solution

Our solution is tailored to address the problem of inefficient travel planning due to a lack of easily accessible real-time traffic updates and notifications about incidents affecting common travel routes. This gap is often overshadowed by conventional navigation services like Google Maps that mostly seek to provide guidance for new routes. We aim to bridge this gap by providing seamless traffic updates and incident notifications across multiple channels, targeting the everyday routes that our users frequent.

Interventions:

- Automated Alert System - a robust alert system that automatically generates notifications based on user-defined parameters
- Multi-Channel Notifications - implementation of a notification service that covers email, SMS, Discord, and Slack
- Multi-Source Data Integration - utilization of traffic and incident data to provide scheduled notifications to users

Metrics for Evaluation:

- Functional interface that allows the user to enter start and end points
- The ability to select a variety of notification methods (email, text, chat)
- Successful notification arrival at the scheduled time
- Alerts include accurate weather and incident details for route

In addition to the metrics above, we expect our customers to enhance their productivity and time management, reduce stress and uncertainty, reduce automobile costs due to unexpected delays, and increase personal safety.

User Scenario 1: Commuter optimizes morning routine

User Story: Mary, an account executive, often encounters unpredictable traffic and weather incidents that cause her to be late.

Solution in Action: Mary receives a traffic delay notification from our system, so she uses an alternate route which saves her 20 minutes and minimizes schedule disruptions.

User Scenario 2: Logistics company enhances operational efficiency

User Story: Global Logistics LLC manages local freight and parcel deliveries, and their operations are often impeded by dynamic traffic conditions.

Solution in Action: Each morning, the logistics control center and dispatching agents receive an alert from our system that highlights potential delays on their common delivery routes. Agents make adjustments based on the alerts and reduce the company delivery times, saving fuel costs and increasing customer satisfaction.

Identifier	Requirement
REQ1	The system shall provide an interface for the creation of alerts accepting the following parameters: origin and destination addresses, preferred channels of notification, e-mail address, telephone number, route nickname, and recurrence.
REQ2	The system shall process requests for alerts and ensure their execution at scheduled times through the chosen communication channels.
REQ3	The system shall facilitate the transmission of notifications across multiple channels, including e-mail, Short Message Service (SMS), Discord, and Slack.
REQ4	The system shall send a notification to the user that includes estimated time, ideal time, travel delay time and any applicable incident information for their respective route.
REQ5	The system shall incorporate and leverage weather and traffic data captured at the time the message is sent for the chosen route.

Plan of Work

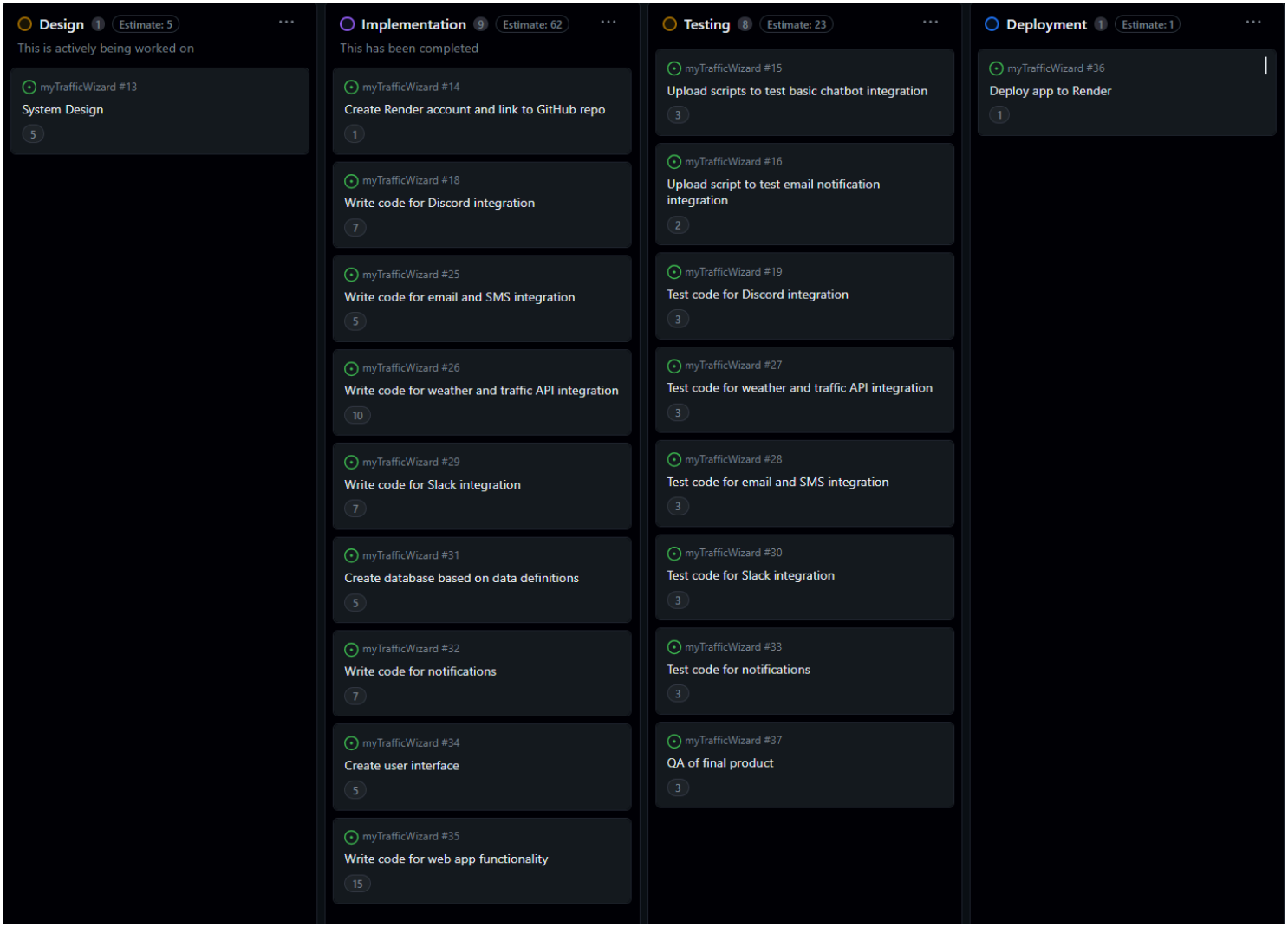
We intend to use commonly available web technologies and freely available services to provide users with weather information along a given route. To that end, we will make calls to mapping services and weather reporting services before integrating that into an alert system. Users will be able to receive alerts by email, text, or in one of their favorite chat apps.

Languages: HTML, CSS, Javascript, and SQL
Platforms/Tools: VS Code, Node.js, Render (host), PostgreSQL (database), queue, alert service
Integrations: Weather/Traffic API, Discord, Slack, email, SMS

We will know we have succeeded if a user is able to receive a requested notification of weather events along a given route via their chosen method or methods. Our success in addressing the customer's diagnosed problem can be measured according to the metrics in the prior section.

Our plan of work and work scheduling will follow the Kanban chart at:
https://github.com/orgs/fhsu-csci441-team-a/projects/4/views/1

An image of this Kanban chart with details of our work plan can also be seen in the image on the following page below:


Design 1 Estimate: 5
This is actively being worked on
myTrafficWizard #13
System Design
5
Implementation 9 Estimate: 62
This has been completed
myTrafficWizard #14
Create Render account and link to GitHub repo
1
myTrafficWizard #18
Write code for Discord integration
7
myTrafficWizard #25
Write code for email and SMS integration
5
myTrafficWizard #26
Write code for weather and traffic API integration
10
myTrafficWizard #29
Write code for Slack integration
7
myTrafficWizard #31
Create database based on data definitions
5
myTrafficWizard #32
Write code for notifications
7
myTrafficWizard #34
Create user interface
5
myTrafficWizard #35
Write code for web app functionality
15
Testing 8 Estimate: 23
myTrafficWizard #15
Upload scripts to test basic chatbot integration
3
myTrafficWizard #16
Upload script to test email notification integration
2
myTrafficWizard #19
Test code for Discord integration
3
myTrafficWizard #27
Test code for weather and traffic API integration
3
myTrafficWizard #28
Test code for email and SMS integration
3
myTrafficWizard #30
Test code for Slack integration
3
myTrafficWizard #33
Test code for notifications
3
myTrafficWizard #37
QA of final product
3
Deployment 1 Estimate: 1
myTrafficWizard #36
Deploy app to Render
1

References

1. Discord, chat app. https://discord.com/developers/docs/intro
2. GitHub, version control and project planning. https://docs.github.com/en
3. Gmail, email provider. https://developers.google.com/gmail/api/guides/sending
4. Knock app, alerts and push notification system. https://knock.app/
5. Render, web app hosting provider. https://docs.render.com/
6. Slack, chat app. https://api.slack.com/start
7. Tom Tom, traffic API.
https://developer.tomtom.com/traffic-api/documentation/product-information/introduction
8. Tomorrow.io, weather API. https://www.tomorrow.io/weather-api/