6-12-2023

Whether-Weather

Lucas Voron

Graham Purvis

Malcolm Weaver

Josh Kelleran

Whether-Weather

by

Lucas Voron
Graham Purvis
Malcolm Weaver
Josh Kelleran

Submitted in partial fulfillment of the requirements

for the degree of

Bachelor of Science in Computer Science & Engineering

School of Engineering
Santa Clara University

Whether-Weather

Lucas Voron
Graham Purvis
Malcolm Weaver
Josh Kelleran

Department of Science and Computer Engineering
Santa Clara University
June 12, 2023

ABSTRACT

The WhetherWeather project is a comprehensive weather and traffic forecasting system that provides real-time insights and predictions. It utilizes Python programming language and various data sources, including Inrix data for traffic information and weather data from Meteostat. The system employs machine learning techniques, specifically XGBoost, to train models on historical data. Once trained, the models can accurately forecast traffic conditions based on weather data. To visualize the results, a Flask API is available, allowing users to access the system's functionalities through a web interface. For more information and to access the WhetherWeather website, visit WhetherWeather.org.

Table of Contents **4**
1 Introduction **6**
2 Requirements **7**
3 Use Cases **9**
Table 3.1 Find Current Weather Caused Traffic Use Case 9
Table 3.2 Determine Weather-Impact of Particular Roads Use Case 9
Table 3.3 Predict Generally Weather-Impacted Roads Use Case 9
Table 3.4 Predict Weather-Impact At a Specific Future Time Use Case 10
Figure 3.1 Use Case Diagram 11
4 Activity Diagrams **12**
Figure 4.1 User Wanting to View Visual Models Activity Diagram 12
Figure 4.2 User Wanting to Read About the Project/ Preventative Measure 12
5 Architectural Diagram **13**
Figure 5.1: Architecture Diagram 13
6 Design Rational **14**
7 Testing Plan **15**
8 Conceptual Model **16**
Figure 8.1 Landing Page 17
Figure 8.4 Current Weather Conditions Map Page 18
Figure 8.5 ML Driven Map Page 19
Figure 8.6 Top Tier List of Dangerous Roads Page 19
9 Final Design **20**
Figure 9.1 Landing Page 20
Figure 9.2 About Us Page 21
Figure 9.3 Data Findings Page 22
Figure 9.4 Top Tier List of Dangerous Roads Page 23
Figure 9.5 Current Weather Conditions Map Page 23
Figure 9.6 ML Driven Map Page 24
10 Risk Analysis **25**
Table 10.1: Risk Analysis Table 25
11 Development Timeline **26**
Table 11.1: Development Timeline 26
12 Societal Issues **27**
13 User Manual **29**
13.1 Landing Page 29
13.2 About Us Page 29
13.3 Data Findings 29
13.4 Top 10 Dangerous Roads Page 29

13.5 Live Map 29
13.6 Machine Learning Map 30
14 Development Manual 31
15 Lessons Learned 34
16 Conclusion 36
17 References 38

1 Introduction

Harsh weather conditions pose a threat to transportation safety and human lives. While there are many transportation based institutions, products, and policies for safe driving in general, we believe that there is still work to be done on the front of driving safety in bad weather conditions. A few examples of a lack of weather based safety measures include speed limits that are only made for optimal driving conditions, but give little guidance when conditions are non-optimal, and the fact that individuals rarely factor rainfall into their trip planning time.

On average, **nearly 5,000 people are killed** and over 418,000 people are injured in weather-related crashes each year[1]. These numbers are likely to increase over time when one considers increases in erratic weather behavior due to climate change.

We want to provide a website/platform to analyze the relationship between harsh weather and transportation safety. Specifically, our platform will aim to analyze how a variety of weather conditions impact the statistical safety of every segment of road within certain geographical regions.

The data our platform will display can be leveraged to make actionable recommendations during extreme (or even moderate) weather events. Such actions could include road maintenance, emergency services planning, trip planning of individuals in impacted regions, and car feature selection based on liability to use weather impacted roads.

Our project has been implemented in a multipage website. It includes suggestions of preventative measures and infrastructure recommendations for safety, a display of historical weather patterns, and safety warnings for drivers under various weather conditions such as heavy rain, snow, winds, or wildfires. We aimed to create a smooth user experience characterized by a clear frontend, effective and efficient backend algorithms, and high-quality traffic data that is used in today's current traffic data-based applications industry at large.

Worsening weather conditions will likely lead to more traffic related injuries and deaths. By compiling and analyzing data on weather and traffic events, we will identify problematic areas of traffic infrastructure under extreme weather. Our website will help to mitigate the negative impacts of climate change on transportation safety.

2 Requirements

Functional requirements

- The system must display website homepage
- The system must display static subpages
- The system must show weather based slowdowns for at least 10 U.S. cities
- The system must display a predictive weather based crash map for various future times

Non-functional requirements:

- The system, specifically the predictive visual model, should run quickly and smoothly
- The system should be easy to navigate and understand for non-technical users
- The system should communicate important information on driving under specific weather conditions, and the dangers associated with them

Constraints

- Lack of weather & traffic data in remote locations across the United States
- Limited size of API calls for front-end efficiency
- Different percentage reporting on
- Recency of traffic data

Technology requirements:

- Front-end:
    - React Javascript framework
    - Charts.js packages for graphic visualizations
    - Deck.gl mapping
- Middleware:
    - NGINX listener to listen to fetch calls by front end
- Back-end:
    - PostgreSQL database
        - Regression models trained from INRIX traffic data and weather data
            - Historical speed/segment data
        - Line Shape for each road segment
        - Historical average speed for the segment
- Machine Learning Model:
    - XGBoost is a Gradient Boosting machine learning technique that iteratively combines weak prediction models to create a strong, accurate prediction system.
    - We combine historical traffic data for each segment of road mapped to the closest weather station's historical weather data and use the weather features to train and eventually predict the target speed

3 Use Cases

As shown in the tables and figures below, we have two primary use cases. Users will view the Live Weather-Based Traffic Slowdowns page to find the weather traffic impact on specific areas that they care about at the current time. A user could alternatively view the Predictive Weather-Based Safety Model page in order to find the roads that are generally most weather-impacted or weather-safe at various future times.

Table 3.1 Find Current Weather Caused Traffic Use Case

Goal	Be able to see, within a geographical region, which areas of road are currently experiencing weather impact
Actor(s)	Web users
Precondition(s)	Actor has selected a geographical location
Postcondition(s)	Live Weather-Based Traffic Slowdowns are displayed for that region using color coding to clearly identify the weather-impacted and weather-safe road sections
Exception(s)	Actor has not selected a valid geographical location, insufficient current data about roads/weather has been recorded

Table 3.2 Determine Weather-Impact of Particular Roads Use Case

Goal	Be able to see a geographically close set of roads to determine whether these roads are weather-safe or weather-impacted
Actor(s)	Web users
Precondition(s)	Actor has selected a geographical location based around the roads they wish to view
Postcondition(s)	Live Weather-Based Traffic Slowdowns are displayed for that region using color coding to clearly identify the whether the particular road sections are weather impacted
Exception(s)	Actor has not selected a valid geographical location, insufficient current data about roads/weather has been recorded

Table 3.3 Predict Generally Weather-Impacted Roads Use Case

Goal	Be able to see, within a geographical region, which areas of road are most and least likely to be weather-impacted (in terms of crashes)
Actor(s)	Web users
Precondition(s)	Actor has selected a geographical location, and actor has selected a time pattern of “General”
Postcondition(s)	Predictive Weather-Based Traffic Safety Model displays general future crash likelihood using color coding to clearly identify the weather-impacted and weather-safe road sections for the given region
Exception(s)	Actor has not selected a valid geographical location/time pattern, insufficient historical data about roads has been recorded and weather data has been recorded

Table 3.4 Predict Weather-Impact At a Specific Future Time Use Case

Goal	Be able to see, within a geographical region, which areas of road are most and least likely to be weather-impacted (in terms of crashes)
Actor(s)	Web users
Precondition(s)	Actor has selected a geographical location, and has selected a time pattern
Postcondition(s)	Predictive Weather-Based Traffic Safety Model displays future crash likelihood for the given time parameters using color coding to clearly identify the weather-impacted and weather-safe road sections for the given region
Exception(s)	Actor has not selected a valid geographical location/time pattern, insufficient historical data about roads has been recorded and weather data has been recorded

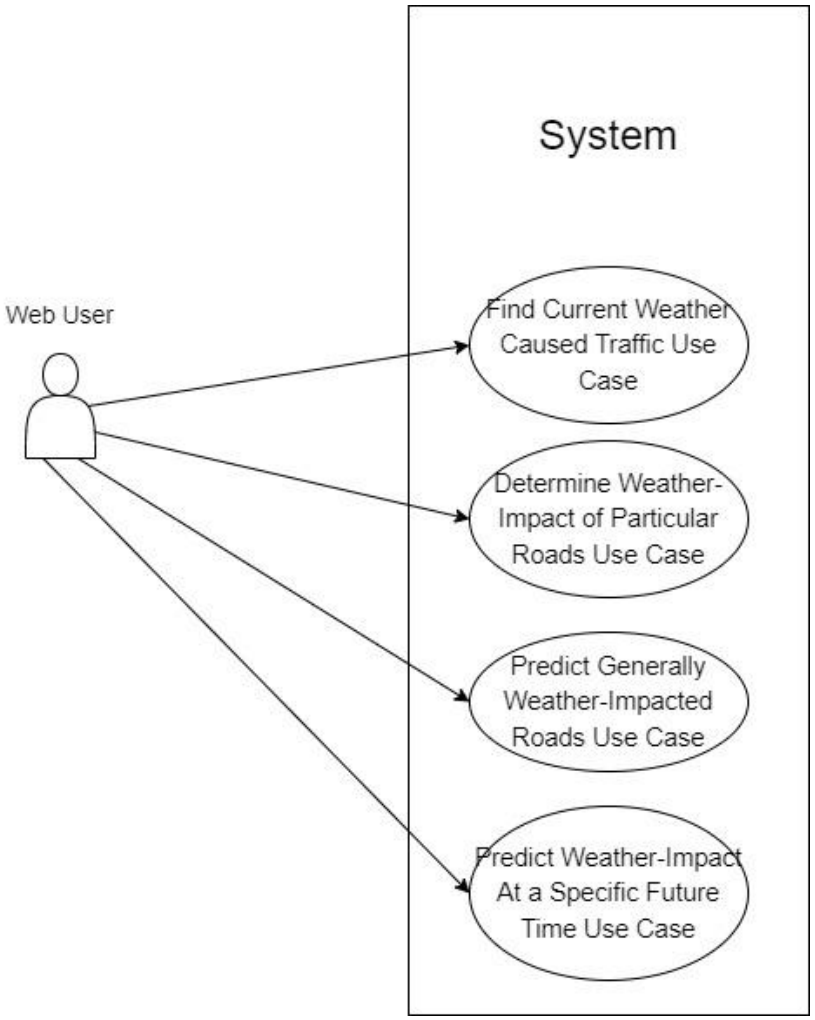

Figure 3.1 Use Case Diagram

4 Activity Diagrams

Below we have activity diagrams outlining different use cases for our web page.

Our first diagram outlines a user wanting to interact with our two map models:

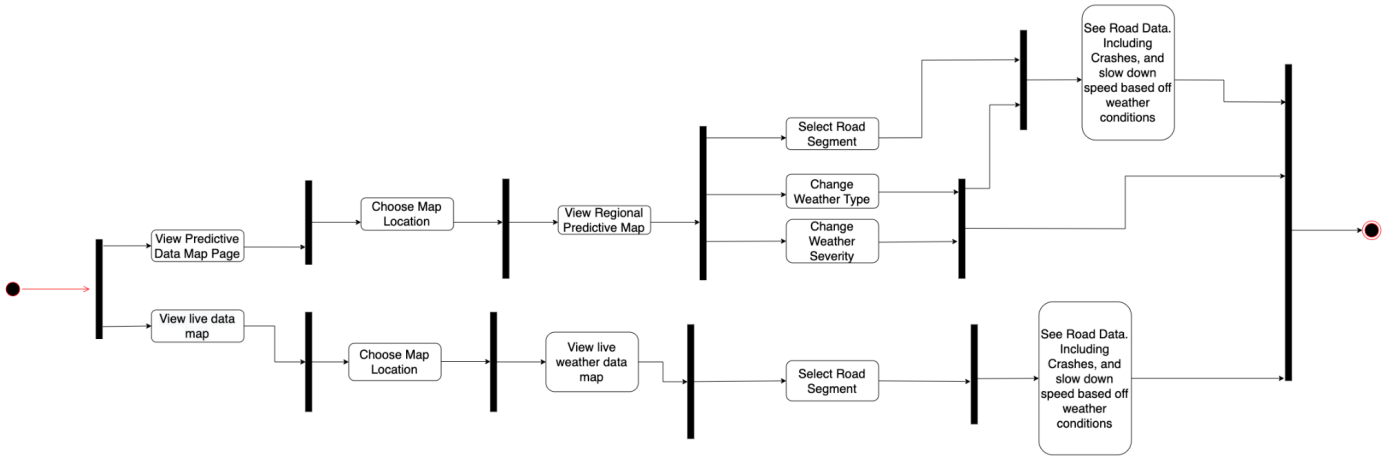

Figure 4.1 User Wanting to View Visual Models Activity Diagram

Our second activity diagram outlines a user interacting with the informational aspects of the webpage:

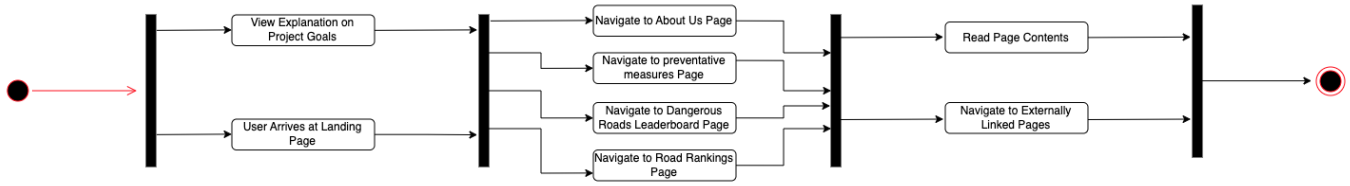

Figure 4.2 User Wanting to Read About the Project/ Preventative Measure

5 Architectural Diagram

As shown in figure 5.1, our architecture is based on a client-server model. Our machine training data is housed in a database in one server, and trained AI will be cached in an AWS Lambda server. Users will interact with our website which will be issued an API to communicate with the backend.

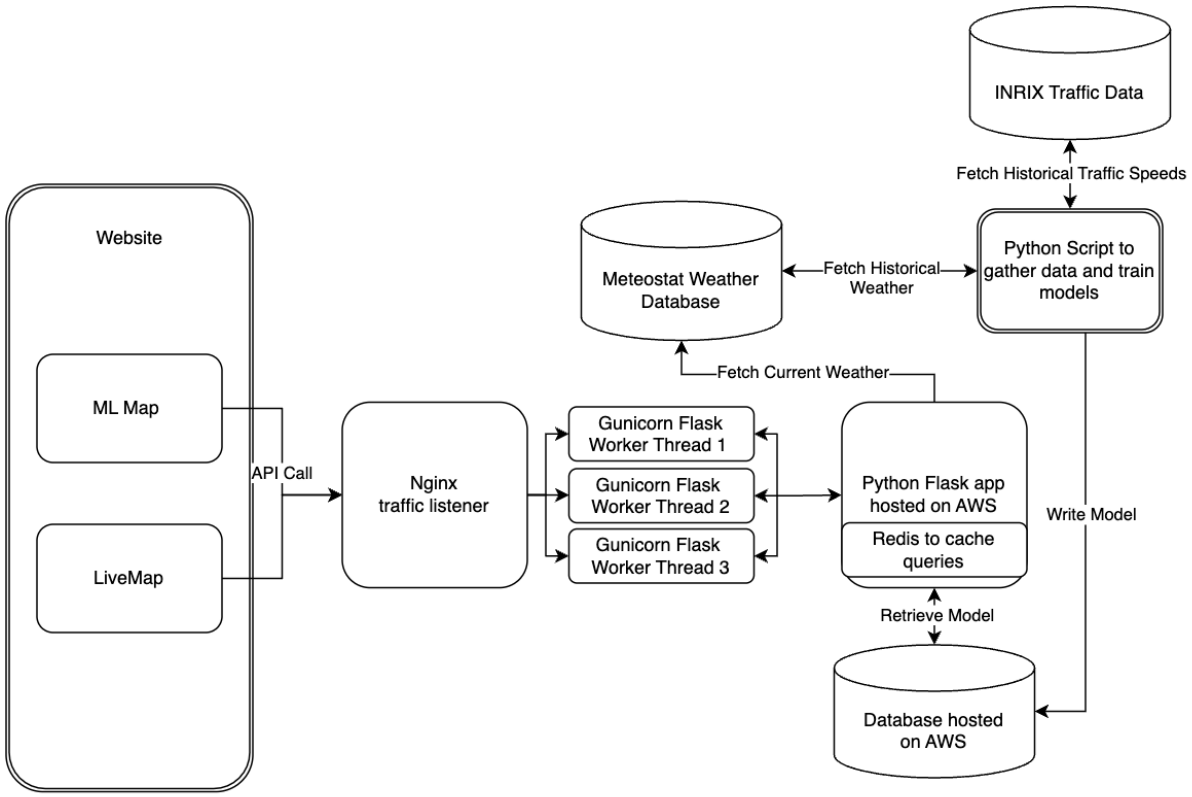

Figure 5.1: Architecture Diagram

6 Design Rational

For our frontend, we used React. Many members of our group had previous experience with React and felt that it would most smoothly allow us to encapsulate our project in an intuitive manner.

For our backend, we used python, as it allows for fast development and we are all comfortable with its libraries. Additionally, Deck.gl will be used to display our maps. This tool conveniently and quickly displays map data.

Our backend infrastructure operates on AWS Lightsail, which offers a cost-effective solution suitable for supporting the anticipated number of users and the system's low-speed requirements. In terms of machine learning algorithms, we heavily relied on XGBoost in conjunction with scikit-learn. Scikit-learn, an open-source Python library, provides a comprehensive implementation of fundamental machine learning functionality. On the other hand, XGBoost is specifically designed for gradient boosting, a powerful technique for creating accurate predictive models.

To train our machine learning model, we employed personal computers and the ECC computers, leveraging their processing capabilities. The trained models were then saved on our computers, awaiting encoding in base64 format for subsequent uploading to our database. These models had to be base64 encoded so they could be readable to our AWS instances. In order to optimize input and output times within our system, we implemented the use of pickle files. Pickle is a Python module that enables the serialization and deserialization of Python objects, allowing us to store and retrieve data structures efficiently. By serializing our machine learning models and other relevant data using pickle, we were able to significantly reduce the time required for loading and saving these objects. Each model was associated with a specific segment and encompassed details such as the segment's shape and historical average speed. This granular approach allowed us to generate precise predictions for traffic speed on a per-segment basis.

Furthermore, our backend system interacts with meteostat to obtain live weather data, which is crucial for generating the live weather map. This integration ensures that the machine learning model receives up-to-date weather information for the relevant region. By incorporating the weather data into the individual models associated with each segment in that region, our system can accurately account for the impact of weather conditions on traffic patterns.

Additionally, the backend retrieves line shapes and historical average speed information from the database. This data is essential for constructing a GeoJSON object, which can be seamlessly displayed on the frontend. By combining the retrieved information with the dynamically acquired weather data, our system provides users with a comprehensive and visually appealing

representation of traffic conditions, allowing them to make informed decisions based on real-time weather-related insights.

Overall, our backend infrastructure, powered by AWS Lightsail, effectively supports our system's requirements. The utilization of XGBoost and scikit-learn enables robust and accurate machine learning predictions, while the integration of live weather data enhances the precision of our models. By combining all these components, we deliver a reliable and user-friendly platform that raises awareness about the influence of weather on traffic speed.

Our Database used postgreSQL because of its compatibility with python as well as its cost effectiveness. We run this on a separate AWS instance and our other instance runs the flask logic in order to call and cache important data as well as executing other backend logic. We use redis as an in-memory database to manage our RAM, specifically our database queries, between Gunicorn threads since python has difficulty sharing memory with threads.

7 Testing Plan

Testing is very important to ensure adequate functionality. We have implemented testing at multiple levels during and after development. Specifically, we implemented AI Model Testing and Unit Testing on our backend server using pytest.

AI Model Testing

Machine learning is notoriously opaque and difficult to test. Therefore, we needed to rely on having a diverse and large dataset and training it according to machine learning algorithms. After this, tested our model by checking various logical invariants on randomly selected road segments. We ensured for example that increases in rainfall are not negatively correlated with increases in danger.

Unit Testing

In order to minimize late stage debugging and verify the functionality of the project, we incrementally tested each piece of our project. We spent especially long amounts of time on verifying the correctness of our model, as well as the AWS back end, as we see these as being the core of the visualization aspect of our project. We worked on testing the functionality of our APIs by using Postman to make calls to the service and verify their output. In addition to this we also wrote unit tests for our flask app to verify that our queries, input parameters, output objects, as well as additional logic needed in the flask app were performed correctly.

8 Conceptual Model

Users will interact with the Whether-Weather system by viewing our website. The figures below show the initial design for the website. We have included six pages: Landing, About Us, Preventative Measures, Current Weather Conditions Map, ML Driven Map, Top Tier List of Dangerous Roads.

Figure 8.1 Landing Page

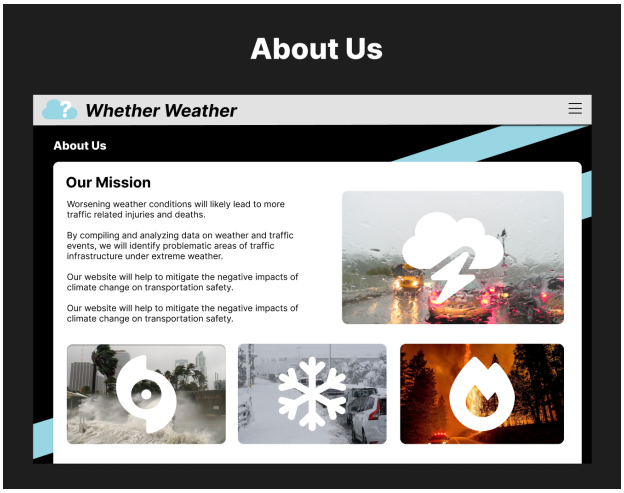

Figure 8.2 About Us Page

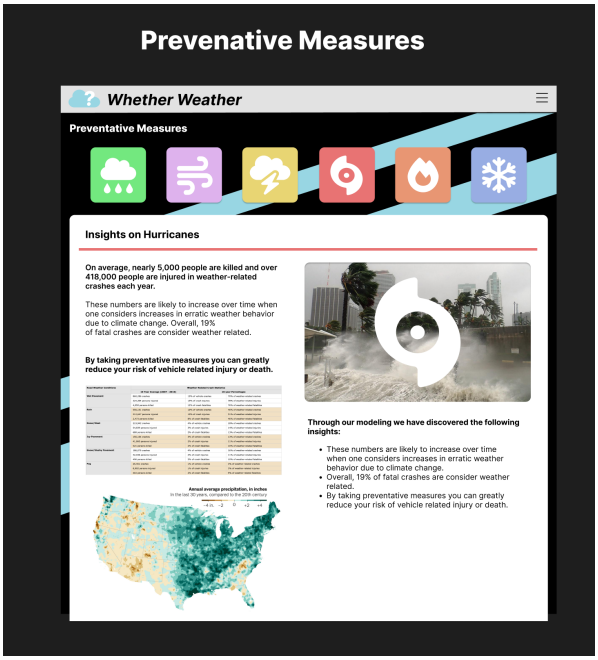

Figure 8.3 Preventative Measures Page

Figure 8.4 Current Weather Conditions Map Page

Figure 8.5 ML Driven Map Page

Figure 8.6 Top Tier List of Dangerous Roads Page

9 Final Design

The landing page serves as the entry point to the website, where users first encounter the Whether-Weather project. It provides a visually appealing and informative introduction, capturing the attention of visitors and directing them to explore further. The landing page may include captivating images, a brief overview of the project's mission, and key features or functionalities that the website offers.

Figure 9.1 Landing Page

The About Us page provides valuable insights into the Whether-Weather project and the team behind it. This page introduces the project's goals, objectives, and the motivation behind its creation. It may showcase the team members, their expertise, and their contributions to the project. Additionally, the About Us page might highlight the project's impact and achievements, fostering trust and credibility among users.

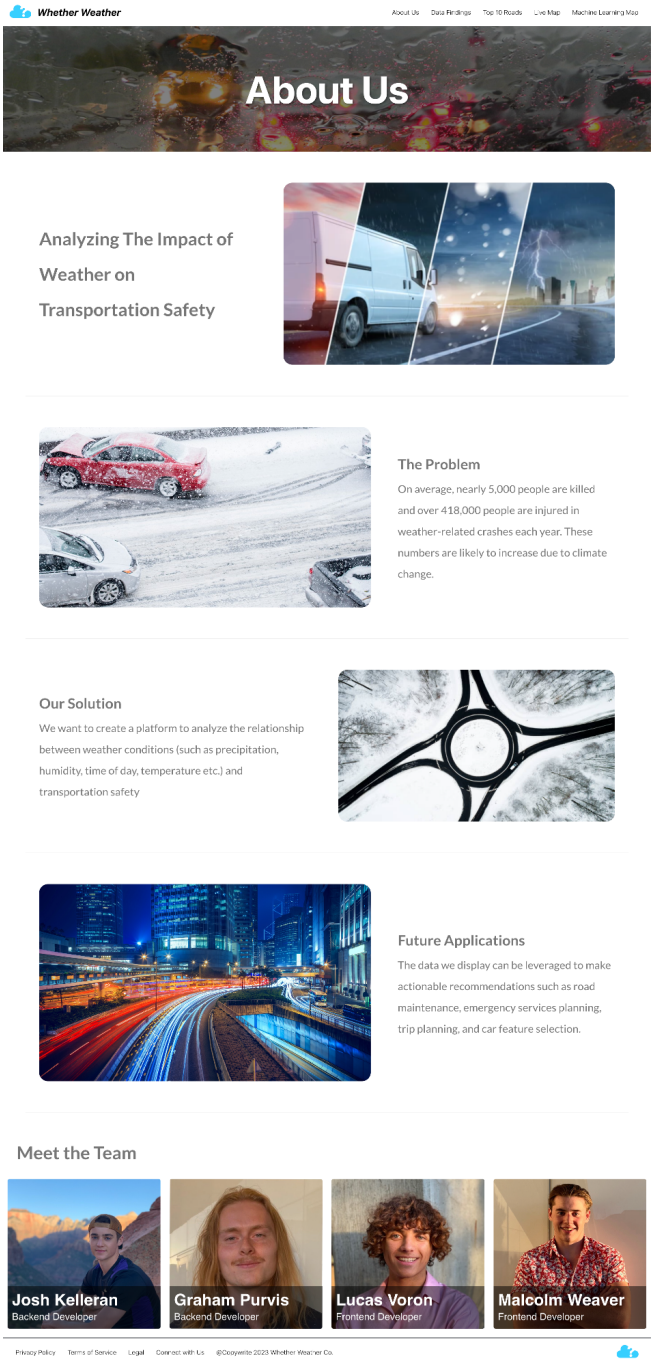

Figure 9.2 About Us Page

The Data Findings page presents the results and insights obtained from the analysis of weather and traffic data. This page communicates the correlations, patterns, and trends discovered during the project's research phase. It includes some key facts that we found when training our model.

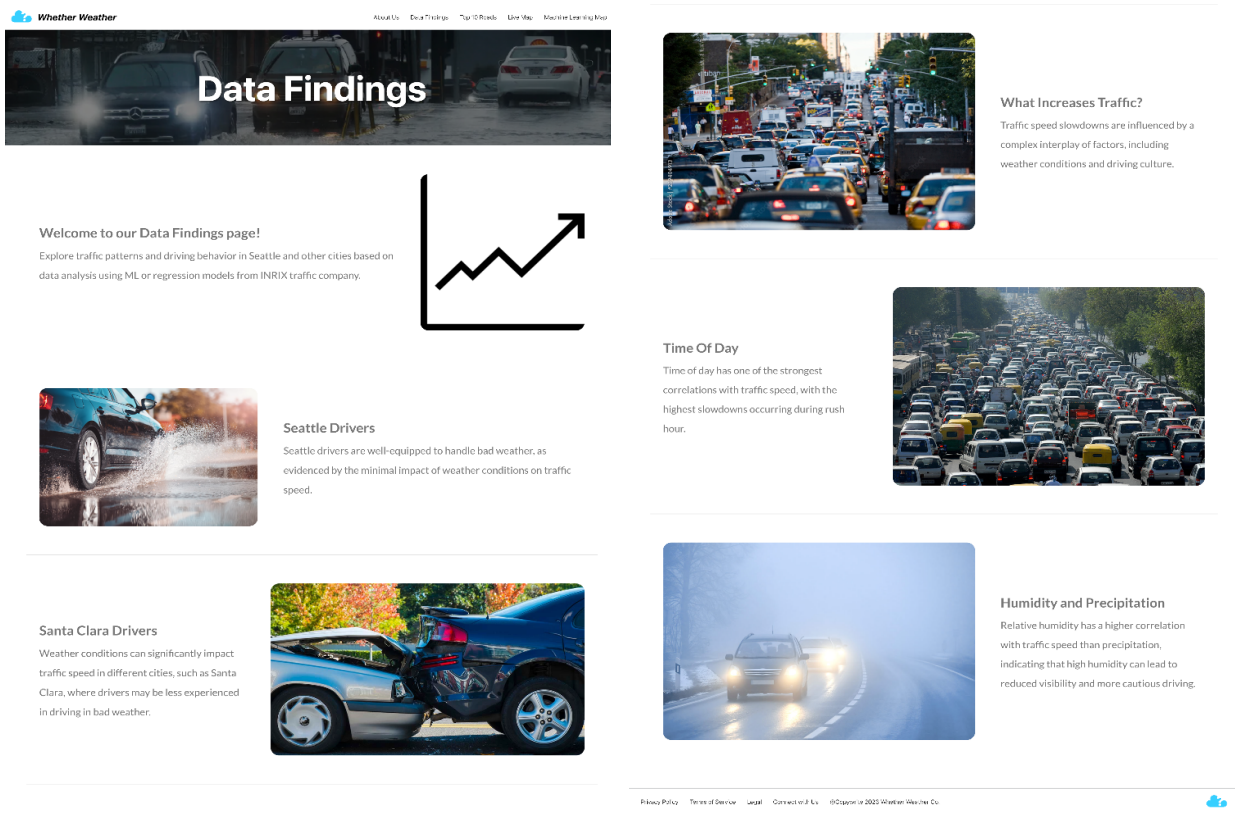

Figure 9.3 Data Findings Page

The Top Tier List of Dangerous Roads page highlights specific road segments or locations that are particularly prone to weather-related traffic risks. This page ranks these roads based on the biggest difference in speed based on good weather versus bad weather. Since we use weather as the feature input we define a set of good weather features and a set of bad weather features and find the biggest difference in our output or target (average speed) values. We sort all of our segment models by this metric to find the biggest slowdowns. The current content was hard coded by looking at the sorted slow-down speeds for Santa Clara county, and assigning road names to segment IDs. In the future we aim to improve this page by allowing users to search the top 10 dangerous roads from various counties which would be stored in a table within our database. This table would be periodically updated to provide our users with up-to-date danger information. Providing this information empowers users to make informed decisions about their routes, take necessary precautions, and raises awareness about hazardous areas prone to weather-related accidents.

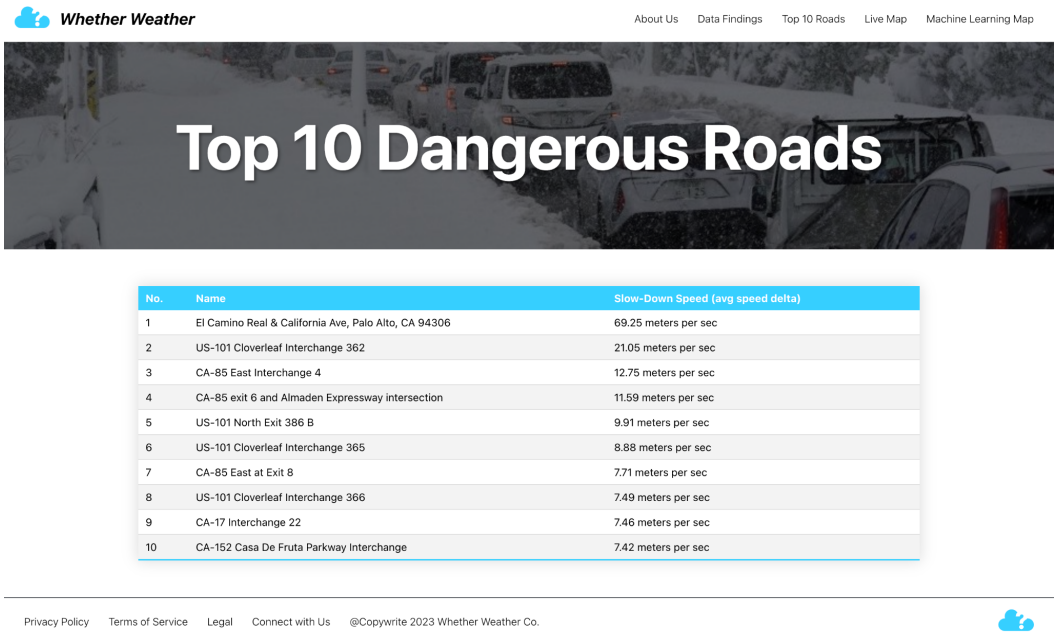

Figure 9.4 Top Tier List of Dangerous Roads Page

The Current Weather Conditions Map page displays a real-time or near-real-time map that visualizes the current weather's impact on road speeds in different areas. Users can explore the map to check how the current weather conditions are affecting traffic in different counties. The map works by making an api request to our backend with parameters such as the county and a specifier indicating a request to display the live map.

Our backend is running on an AWS instance that has Nginx listening to any requests from our specific URL. The listener then tasks one of our three Gunicorn threads to check if we have the county data already stored in our memory managed by Redis. If it isn't in Redis it will make a call to our database to get the encoded models, line shapes, and historical reference speed and serialize it and put it into Redis cache. For the Live Map we make an api call to meteostat for the live weather feature inputs (rain, humidity, etc.) and deserialize the models, line shapes, and historical reference speed one by one and build up a geojson object. The created object contains the line shapes in the specified county, the target speed based on the current weather inputs for that segment, as well as the desired color in comparison between the target speed and historical reference speed. Our backend builds this geojson object which is around 10-25 megabytes and returns it to the frontend which will take this built geojson object and display it using deck.gl.

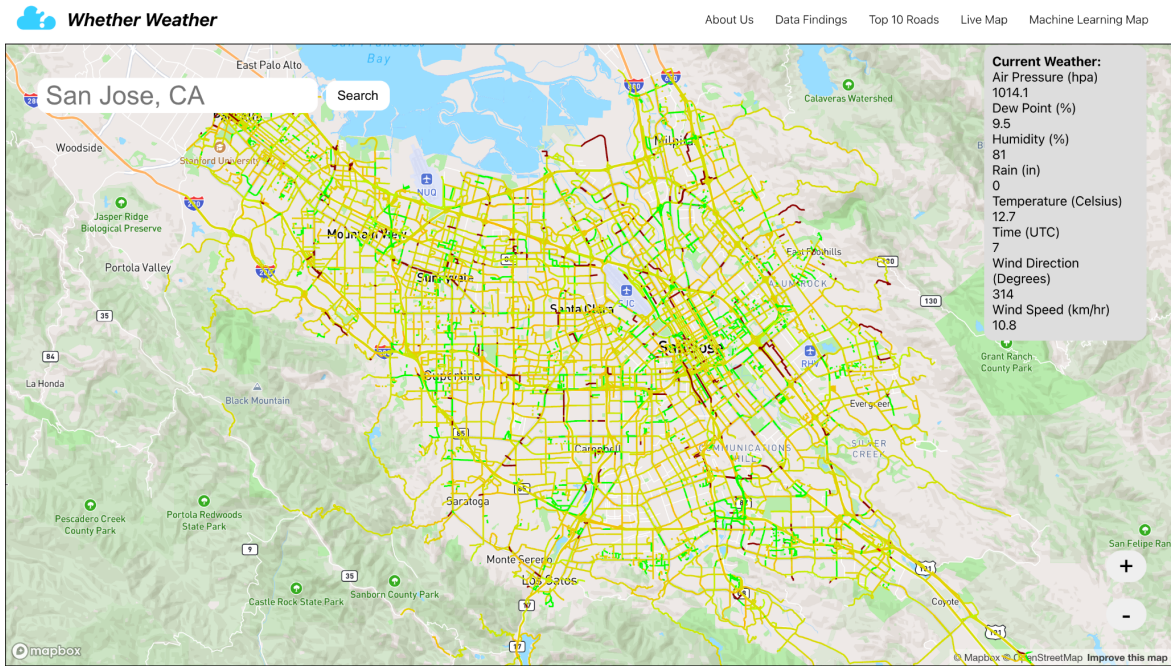

Figure 9.5 Current Weather Conditions Map Page

The ML Driven Map page utilizes machine learning algorithms to generate a map that predicts and highlights areas with increased risk of weather-related traffic incidents. Users can explore the ML-driven map to identify areas that require extra caution during adverse weather conditions, helping them plan their journeys more safely. The map works by making an api request to our backend with parameters such as the county, current inputted weather, and a specifier indicating a request to display our machine learning map.

Our backend is running on an AWS instance that has Nginx listening to any requests from our specific URL. The listener then tasks one of our three Gunicorn threads to check if we have the county data already stored in our memory managed by Redis. If it isn't in Redis it will make a call to our database to get the encoded models, line shapes, and historical reference speed and serialize it and put it into Redis cache. For the Machine Learning Map we take the inputted weather (rain, humidity, etc.) and deserialize the models, line shapes, and historical reference speed one by one and build up a geojson from the line shapes in the specified county, the target speed based on the weather inputs for that segment as well as the desired color in comparison between the target speed and historical reference speed. Our backend builds this geojson object which is around 10-25 megabytes and returns it to the frontend which will take this built geojson object and display it using deck.gl.

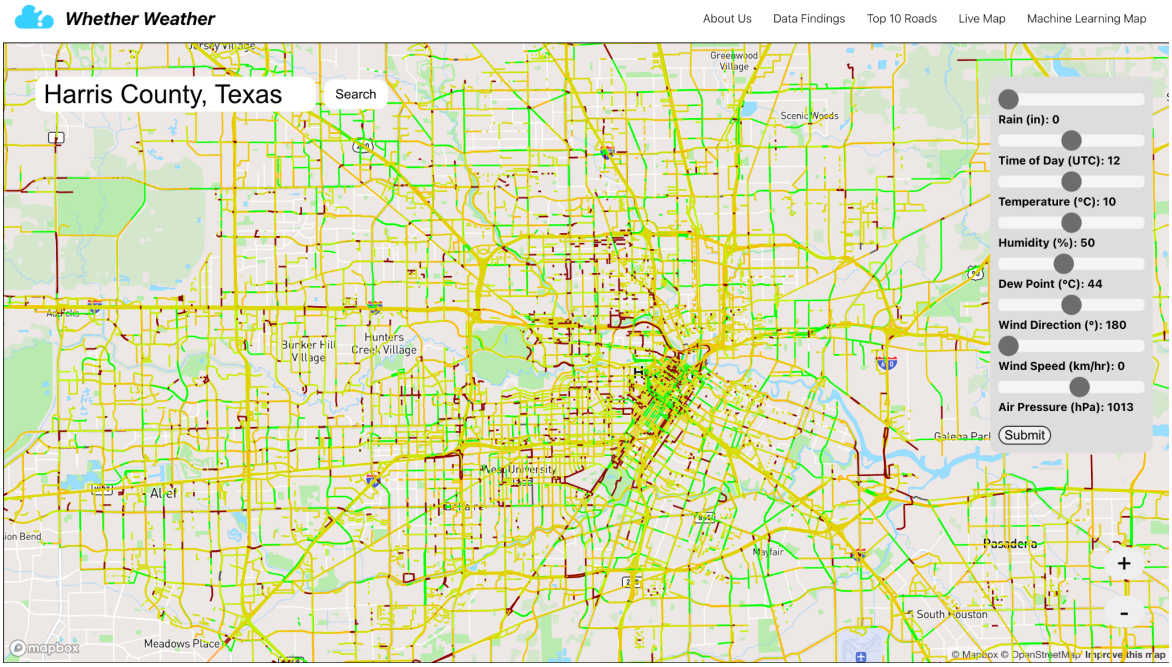

Figure 9.6 ML Driven Map Page

10 Risk Analysis

Table 10.1 outlines the possible risks that we predicted we would encounter during our development process. The numbers provided are based on our current understanding of each risk, and were used to predict which areas of our project we expected to be pain points. The probability below defines the chance of the given risk impacting our work, and the severity of each risk is how impactful the risk would be on our project. By taking the product of these two parameters we arrived at the impact score for each risk which might affect our approach to each phase of the project.

Risk Rating Scale: 1-10:

Risk	Consequences	Probability	Severity	Impact	Mitigation
Scoping	Scope being too large will prevent the completion of all project requirements.	0.5	5	2.5	Reduce the number of locations, reduce the number of factors in the predictive model, reduce the number of road segments done.
Bugs	Will slow down the development process and possibly get in the way of timeline goals.	1	2	2	Debugging
API Integration	If APIs are not giving the desired data, we may need to combine resources in order to get the desired outcome.	0.8	10	8	Create a simplified version of the product early on in order to verify that the APIs will contain all the data we need
Performance Issues	The product cannot be widely adapted if it is unusable do to performance	0.8	6	4.8	Developing algorithms to efficiently display data

Table 10.1: Risk Analysis Table

11 Development Timeline

Below is the initial development timeline for our project. Unfortunately due to heavy course schedules in the winter, we ended up compressing the majority of December, January, February and March work into late March, April, and May while also working on our presentation. Our report ended up being done last in early June.

	People	SEPT	OCT	NOV	DEC	JAN	FEB	MAR	APR	MAY
Research & Planning	ALL	X	X	X						
Design	ALL	X	X	X						
Get API Access	ALL			X						
Front-end Development	ALL				X	X	X	X		
Middleware Development	ALL				X	X	X	X		
Back-end Development	Graham, Malcolm, & Josh				X	X	X	X		
AI Modeling	ALL					X	X	X		
Unit Testing	ALL					X	X	X		
Integration Testing	ALL					X	X	X		
Report	ALL								X	
Presentation	ALL									X

Table 11.1: Development Timeline

12 Societal Issues

12.1 Ethical

The development of our website/platform to analyze the relationship between harsh weather and transportation safety raises ethical considerations regarding data access and privacy. We recognize the importance of providing users with the ability to access the data while also preserving their privacy. We implemented measures that allow users to control the visibility of certain fields to protect their privacy and prevent potential retaliation. By prioritizing data accessibility and privacy, we aim to create an ethical and user-centric platform.

12.2 Social

Our project addresses a pressing social concern by focusing on driving safety in bad weather conditions. Harsh weather poses a significant threat to transportation safety and human lives, and existing safety measures often overlook weather-related factors. By developing a platform that analyzes the impact of various weather conditions on road safety, we aim to fill this gap and provide valuable insights for road maintenance, emergency services planning, and individual trip planning. Our project aims to contribute to the overall well-being and safety of communities impacted by harsh weather conditions.

12.3 Political

In the realm of transportation safety, there may be political considerations regarding the responsibility for road maintenance and infrastructure improvements in relation to weather conditions. Our platform will present data on problematic areas of traffic infrastructure under extreme weather, helping to identify areas that require attention and potential policy changes. However, as software engineers, we understand that policy decisions are outside our purview. We will provide the necessary data and analysis to stakeholders and policymakers, enabling them to make informed decisions and take appropriate actions.

12.4 Economic

The long-term economic sustainability of our project will rely on the maintenance of the website/platform and the continuous availability of high-quality traffic data. We will ensure that the codebase is well-documented and accessible for future maintenance and feature additions. Additionally, the costs associated with hosting and managing the website/platform will be considered to ensure affordability for our clients and stakeholders. By addressing the economic aspects of the project, we aim to create a sustainable solution that can continue to provide value in the long run.

12.5 Health and Safety

Our project directly contributes to health and safety by focusing on transportation safety in harsh weather conditions. By analyzing the relationship between weather and traffic events, we can identify areas of traffic infrastructure that pose risks during extreme weather events. This information can be used to implement preventative measures, plan emergency services, and provide safety warnings to drivers. Through our project, we aim to enhance health and safety outcomes by mitigating the negative impacts of worsening weather conditions on transportation safety.

12.6 Manufacturability

As a website/platform, our project does not involve traditional manufacturing processes. However, we will ensure that the development and deployment of the website/platform follow industry-standard practices. The implementation of effective frontend design, efficient backend algorithms, and reliable traffic data integration will contribute to a well-manufactured and user-friendly solution.

12.7 Sustainability

To ensure the sustainability of our project, we will focus on the maintainability and scalability of the website/platform. Thorough documentation of the codebase and software stack will facilitate future maintenance and updates. We will also consider feedback from users and stakeholders to incorporate additional features and improvements over time. By prioritizing sustainability, we aim to create a solution that can adapt to evolving needs and provide long-term value.

12.8 Environmental Impact

While our project primarily focuses on transportation safety, it indirectly contributes to environmental conservation efforts. By promoting safer driving practices and optimizing traffic infrastructure under extreme weather conditions, we can reduce accidents and their associated environmental impact. By addressing the risks posed by climate change-induced weather patterns, our project aligns with broader environmental conservation goals and supports efforts to create a sustainable future.

12.9 Usability

Usability is a core aspect of our project, as we aim to create a smooth user experience for individuals accessing the website/platform. Through user feedback and iterative design with our peers, we tried to make our website as accessible as possible.

13 User Manual

13.1 Landing Page

The landing page is the first page you encounter when visiting WhetherWeather.org. It provides an overview of the website's purpose, highlighting the problem of harsh weather conditions and their impact on transportation safety as well as our solution, which involves analyzing data to identify problematic areas and provide actionable recommendations. Scrolling down, you can see icons to jump to any of the other pages.

13.2 About Us Page

To access the "About Us" page, you can either click on the "About Us" link in the navigation bar at the top of the page or scroll down to find the section. The "About Us" page offers detailed information about WhetherWeather.org. Here, you can learn about our team, mission, and the future applications.

13.3 Data Findings

To access the "Data Findings" page, you can either click on the "Data Findings" link in the navigation bar at the top of the page or scroll down to find the section. The page is designed to provide insights into various aspects related to traffic and driving, leveraging the power of data analysis and advanced algorithms.

13.4 Top 10 Dangerous Roads Page

To access the "Top 10 Dangerous Roads" page, you can either click on the "Top 10 Roads" link in the navigation bar at the top of the page or scroll down to find the section. Once you have reached the section, you will see a table displaying the top ten roads. The information presented in the table highlights the roads with the highest average speed reduction, indicating potential areas of concern in terms of safety and traffic conditions. The higher the value in the "Slow-Down Speed (avg speed delta)" column, the more significant the impact on average speed due to various factors.

13.5 Live Map

To access the "Live Map" page, you can either click on the "Live Map" link in the navigation bar at the top of the page or scroll down to find the section. The Live Map page of Whether Weather provides an interactive interface for users to explore real-time weather conditions in different locations. The page features a search bar where users can enter a specific location, such as "San Jose, CA," to retrieve weather data for that area. You can click and drag the map to navigate around and explore different areas. By moving the map, you can focus on specific regions and view relevant weather and road information. Roads will be colored based on how weather dependent they are with red meaning severely affected and green meaning less significantly affected.

13.6 Machine Learning Map

To access the "Machine Learning Map" page, you can either click on the "Machine Learning Map" link in the navigation bar at the top of the page or scroll down to find the section. The Machine Learning Map page of Whether Weather displays a user interface allowing users to search for specific locations and view a map with color-coded roads based on their level of weather impact. The page includes a search bar, zoom controls, and sliders to adjust various weather parameters such as rain, time of day, temperature, humidity, dew point, wind direction, wind speed, and air pressure. Users can submit their preferred settings using the submission button.

14 Development Manual

Cloning from Git:
Open your terminal or command prompt.

Navigate to the directory where you want to clone the repository. You can use the cd command to change directories. For example, if you want to clone the repository to your desktop, use the following command:

cd ~/Desktop
Clone the repository by running the following command:

git clone https://github.com/username/WhetherWeather.git
Replace username with the actual username of the repository owner. This command will create a new directory named "WhetherWeather" and clone the repository into it.

Once the cloning process completes, you can navigate into the cloned repository by using the cd command:

cd WhetherWeather
You now have the repository cloned to your local machine. You can start working with the code or explore the files within the repository.

Remember to replace the repository URL with the actual URL of the "WhetherWeather" repository you want to clone.

Setup Python and Pip
Before proceeding with the development steps, ensure that Python and Pip are installed on your computer. You can download them from the official Python website (https://www.python.org) and follow the installation instructions specific to your operating system.

Setup Python Virtual Environment
A Python virtual environment helps create an isolated environment for your project, ensuring that dependencies are managed separately. Follow the steps below to set up a virtual environment:

Create a new virtual environment
python -m venv traffic_prediction_env

Activate the virtual environment
On Windows
traffic_prediction_env\Scripts\activate

On macOS/Linux

source traffic_prediction_env/bin/activate
Install Requirements
To install the required Python packages, navigate to the project directory and run the following command:

pip install -r requirements.txt

Download Inrix Data
To obtain the necessary traffic data, follow these steps:
Visit the Inrix website at https://inrix.com/products/iq-location-intelligence-solutions/.
Download around three months' worth of hourly data for your desired county. Ensure that the data includes both weather and traffic information. Save the downloaded data to a suitable location on your computer.

Preprocessing Steps
Before training the model, you need to perform some preprocessing steps. This involves fetching weather data, finding midpoints, and creating a consolidated CSV file. Execute the following commands in order:
Fetch Weather Data:

python fetch_weather_data.py --data_path <path_to_weather_data>
Replace <path_to_weather_data> with the path where you saved the downloaded weather data.

Find Midpoints:
python find_midpoints.py --data_path <path_to_traffic_data>
Replace <path_to_traffic_data> with the path where you saved the downloaded traffic data.

Create CSV:
python create_csv.py

Train the Model with XGBoost:
To train the model using the provided CSV file, run the following command:

python train_model.py --csv_path <path_to_csv_file> --county <your_county>
Replace <path_to_csv_file> with the path to the generated CSV file from the previous step.
<your_county> should be replaced with the name of your desired county.

Visualize the Model with Flask API
To visualize the trained model, follow these steps:
Run the Flask API Locally:

flask app.py

(Within the flask directory in the project)

Start the Frontend:

Install Node.js and npm by downloading the appropriate installer from the official Node.js website (https://nodejs.org) and running it.
Install project dependencies using **npm install <package-name>** for each required package.
Test the project locally by running **npm start** in the terminal.
To deploy:
switch to the gh-pages branch using **git checkout gh-pages**
Generate the static files for deployment: **npm run build**
Commit the newly generated static files: **git add .** and **git commit -m "Add static files for deployment"**
Push the gh-pages branch to the GitHub repository: **git push origin gh-pages**

Pushing to Production (Optional)
If you have tested your code and trained model and wish to push them to production, make a request following your organization's production deployment process. Provide the relevant code and trained model files for the production deployment team to incorporate them into the production environment.

Note: Ensure that you have satisfied all the prerequisites, such as having the correct paths for data and the required Python packages installed, before executing the code snippets.

Remember to customize the steps and paths according to your specific project configuration.

15 Lessons Learned

During the development of the Whether-Weather project, we encountered various challenges and gained valuable insights. The following lessons were learned throughout the process:

Clear project scoping: It is crucial to define the project scope early on and ensure it is realistic and manageable within the given timeframe. Managing the scope helps prevent unnecessary delays and ensures the completion of all project requirements.

Effective API integration: Working with external APIs can be complex, and it is essential to thoroughly understand their functionality and limitations. It is advisable to create a simplified version of the product early on to verify that the APIs provide the required data. Additionally, having contingency plans in case the APIs do not meet expectations can help mitigate risks.

Bug tracking and debugging: Bugs are inevitable during the development process, and it is essential to have effective bug tracking and debugging mechanisms in place. Regular testing and debugging sessions help identify and resolve issues promptly, preventing delays and ensuring a smoother development workflow.

Performance optimization: Ensuring the system's performance is optimized is crucial, particularly when dealing with large datasets and complex algorithms. Developing efficient algorithms and data processing techniques can significantly improve system performance and user experience.

Incremental testing: Implementing testing at multiple levels, including unit testing and AI model testing, helps identify and rectify issues early on. Incremental testing allows for a more systematic approach to identifying and resolving problems, reducing the overall development time.

Privacy and ethical considerations: When dealing with sensitive data, such as user information and location data, privacy and ethical considerations must be taken into account. Implementing measures to protect user privacy and providing users with control over the visibility of their data helps maintain trust and adhere to ethical standards.

Collaboration and communication: Effective collaboration and communication among team members are crucial for project success. Regular meetings, clear task assignments, and open lines of communication foster a productive and collaborative work environment.

Documentation and reporting: Maintaining comprehensive documentation throughout the development process helps track progress, clarify requirements, and ensure smooth knowledge

transfer. Creating a well-structured report summarizing the project's objectives, methodologies, and outcomes provides a comprehensive overview of the work done.

By applying these lessons learned, future projects can benefit from a more streamlined development process, improved efficiency, and enhanced outcomes.

16 Conclusion

In conclusion, our project successfully addressed the need for improved driving safety in harsh weather conditions. We recognized the existing gaps in safety measures, such as speed limits designed for optimal driving conditions and the lack of consideration for weather conditions in trip planning. With the completion of our project, we have made progress in bringing awareness to these issues and providing a valuable resource for users.

Our completed website/model analyzes the relationship between weather conditions and transportation safety, specifically focusing on the statistical safety of road segments in specific geographical regions.

During the development process, we encountered several challenges related to manipulating big data, understanding machine learning algorithms, and effectively using mapping frameworks to display data to users. However, through diligent effort and continuous learning, we overcame these obstacles and successfully implemented the necessary functionalities.

The system requirements, both functional and non-functional, were carefully considered and implemented to ensure an effective and user-friendly platform. We incorporated the necessary technologies, such as utilizing frameworks for handling big data, understanding and applying machine learning algorithms, and leveraging mapping frameworks to visualize data to users.

To ensure the quality and functionality of our system, we conducted thorough testing at multiple levels, including testing the AI model, and unit testing. These testing processes allowed us to verify the correctness of our algorithms, ensure smooth integration of different components, and provide a reliable and accurate platform for users.

Our completed conceptual model presents a user-friendly interface that allows users to access current weather-based traffic slowdowns and predictive weather-based safety models. Through visualizations, informative pages, and a top 10 list of dangerous roads, users can make informed decisions and take necessary precautions while driving in harsh weather conditions.

Through the completion of this project, we have gained valuable experience in manipulating big data, understanding machine learning algorithms, and effectively utilizing mapping frameworks. These skills will be instrumental in future projects and allow us to tackle complex challenges with confidence.

In summary, our project successfully addressed the need for improved transportation safety in harsh weather conditions. By leveraging technology, data analysis, and effective user interfaces, we have provided a valuable resource for users to make informed decisions and enhance driving

safety. With the completion of this project, we have taken significant steps towards mitigating the risks associated with harsh weather conditions and ensuring safer driving experiences for all.

17 References

1. “How Do Weather Events Impact Roads?” *How Do Weather Events Impact Roads? - FHWA Road Weather Management*, https://ops.fhwa.dot.gov/weather/q1_roadimpact.htm.