

EE/CprE/SE 491 WEEKLY REPORT 9

04/2/24 - 04/9/24

Group number: 22

Project title: CyRide Visualization

Client: Mohammed Soliman

Advisor: Mohamed Selim

Team Members & Role:

Bradon Buckalew: Programmer

Endi Odobasic: Programmer

Evan Schlarmann: Programmer


Andrew McMahon: Programmer

Week Summary

The team worked towards displaying mock data on the Google Maps interface. This requires changes in every component so that they communicate together. It also requires the implementation of the Google Maps API. Team also worked on documentation for the app as a whole and researched testing frameworks for our backend and frontend

Accomplishments

App Documentation:

 App Documentation

Django WebSocket:

Pulls the location data from MySQL server line by line and sends it to any connections on the WebSocket. Data queried from the database is converted into Python datatypes using a model serializer. The data is then converted into JSON for users.

Testing Research:

Created a Trello Testing Ticket; in the ticket, there are hyperlinks leading to libraries, examples, and technical explanations about potential testing frameworks our project could use. The Frontend will use Jest, a test runner, and the React Testing library, designed to test React components. Below is a table showing the difference between the two.

Jest	React Testing Library
Is a test runner	Is a Library for writing tests specific to the React ecosystem
Is Framework agnostic	Is not Framework agnostic
Helps with writing snapshots, assertions, test suites, etc	Helps in writing actual tests by rendering component
Provides assertion utilities and code coverage report	Provides built-in support for manually checking user interactions like typing, clicking, etc. with <code>fireEvent</code>
Nearly zero-config setup. Automatically finds and runs the test cases	Automatically re-renders the component based on user interaction (i.e. typing or clicking).
Helps in checking the output of rendered components with assertions	Helps in rendering components with virtual DOM so the output can be checked via assertion utilities. Ensures maintainable and trustworthy tests
Built-in support for mocking, snapshot, and assertion utilities	Provides a simple API for testing components. Encourages testing of accessibility and user experience
Provides support for testing asynchronous code	Provides support for rendering & testing React Hooks.

We plan on writing unit tests for our backend with Python's standard library module: unittest. Unittest was inspired by JUnit, it supports test automation, sharing of setup, and shutdown code for tests.

Polyline Bus Route:

We got the Orange 23 route and the route structure for where the bus might be traveling through the GoogleMaps package, but it was done within the Django/HTML/JS initially, so now the migration over to the google-maps-react package is raising some difficulty. We have the structure of how it should look but it doesn't transfer over directly and still working on moving it over to Typescript.

Header and Footer:

I created a basic header and a footer. The header shows our project and the footer has links to our social media accounts and our project website. The white space in the middle is where our Google API will be displayed after its pushed to main. Used our figma templates for designs

Cyride UE Visualization Senior Design December 2024 Group 22

Braden Buckalew

Evan Schlarmann

Endi Odobasic

Andrew McMahon

Advisor: Mohamed Y. Selim

Client : Mohammed Soliman

Project Website

Pending Issues

- Migration from Javascript/HTML to React/Typescript of Orange 23 route

Individual Contributions

<u>NAME</u>	<u>Individual Contributions</u>	<u>Description</u>	<u>Week Hours</u>	<u>Cum. Hours</u>
Evan Schlarmann	1. Created location WebSocket 2. Created documentation for Django and Server 3. Fixed database models	1. The location WebSocket currently retrieves locations from the database line by line and sends them in JSON format to any connections that have been made. 2. The documentation goes into detail about the different Django classes and what purpose each one serves. This provides insight when building upon the project and separates the functionality. The server documentation goes over the CI/CD pipeline and useful commands.	8	51

	and migrations	3. Fixed models so they matched the database. Django migrations now work and update the tables correctly. The data can be retrieved using the models and serializers.		
Braden Buckalew	<ul style="list-style-type: none"> - Research Unit Testing frameworks for React and Django - Create Footer and Header -React Folder Structure -Helped with JS to TS conversion for google api 	<ul style="list-style-type: none"> - Created Issues in Gitlab to implement basic unit tests and examples for the team. I created a ticket in Trello with valuable links to documentation and examples. -Our header and Footer display information about our project; some links include undergrad teammates, advisor, and client links. -Implement a quick and easy way for our team to stay organized while navigating through the frontend files -Helped Endi move our project to follow the React ts framework so we can dynamically change components on the fly and avoid template-driven web applications. 	10	47
Endi Odobasic	<ul style="list-style-type: none"> 1) Routes to Polyline 2) Change from Javascript to Typescript. Research + debugging 	<ul style="list-style-type: none"> 1) initially, I had the bus path displayed as a route, so I changed it over to a polyline path (something that is more static and clearer) 2) I had everything originally in javascript (html + javascript) and had to convert it into typescript, but the process isn't so easy, so its taken a long time to figure out. 	12	50
Andrew McMahon	<ul style="list-style-type: none"> 1) Update backend to store all locations for one bus (Brown), rather than current 2) Added CRUD endpoints to Locations, Vehicles, and Stops 3) Research ML models to implement 	<ul style="list-style-type: none"> 1) Change the location table to contain historical location data (all), add a timestamp to each location update, and use WebSocket for current data. Also, change the location table's data to use one Bus (Brown, the one with UE on it). 2) Finalized all basic backend endpoints (CRUD operations for both tables, and Stops for the Google Maps UI), working with Postman. 3) Researched Machine Learning models and will propose to the group (will be included in design and eventually implemented) at this Thursday's meeting. 	10	45

Plans For the Upcoming Week

- Update the table's relationships and data
 - Ensure the tables are created correctly and use the correct keys to retrieve data about a vehicle and its location.
 - Set up the testing frameworks
 - Initialize testing frameworks on both layers of the application with basic tests
 - Include the testing frameworks in the CI/CD pipeline
 - Have websocket connecting to Django so location on Google Maps
 - Create Icons to use for buses to display, one showing green for connected and green for unconnected
-

Weekly Client/Advisor Meeting Summary

In our weekly meeting, Team Members list out our past week's accomplishments. Our work was focused on getting a basic understanding of WebSocket communication, setting up our SQL database on our ETG server, and importing Google Maps. We discussed the importance of using Django for our project by using its built-in security features, making it a popular and easy framework to develop.

We discussed our plan for the next few weeks to work on technical tasks. Nearing the end of the semester, we want to transition to preparing for our design presentation. Our team will prepare and put our slides in our shared box folder with our Client and Advisor.