Divvy Van Asst. Final Summary

Group 1: Vivek Ganesan, Juan Zambrano, Alfonso Arias

Description

The *Divvy Van Assistant* is an application based on the *Divvy* bike rental system, operated by Motivate International Inc. for delivering bikes from their compound to their appropriate stations all around Chicago, as well as managing any necessary maintenance work to those already at a station. Efficiency is key, meaning delivery van drivers would be able to make all bicycle transfers in a more well-ordered and timely manner, compared to that of times previously. With the recent addition to scooter-sharing to the system, the application would accommodate for those, as well.

How does system account for scooters not at stations? Do

Requirements

In terms of functional requirements here are some of the requirements that need to be achieved and used. In terms of data processing, the application must be able to process data from the Divvy share ride database. It must process the data via getting each location of the Divvy bike station and determining how many bikes or scooters are in each station. It must also process the data to find if any customer(s) reported any broken bikes to be fixed or replaced. In order for this app to work, this functional feature must be included without it the app will not function correctly and will not be useful. This requirement can be checked off as successful once the application can analyze the data correctly and display the number of bikes and scooters and how many that are needed for repairs, at each station correctly.

Describing the r

Awkward

Another requirement would be the precision and accuracy of real-time data provided by the individual stations to ensure vans are equipped to efficiently transfer its inventory, minimizing over- or underestimation. The system will collect the available inventory data from the aforementioned stations and must be constantly updating. This functional feature is needed to allow drivers to monitor their inventory, also allowing their supervisor to monitor all other vans as well. Without this feature, many drivers could potentially miss manage their inventory and causing a delay when delivering bikes. This requirement can be labeled a successful one when the application can correctly update the drivers inventory.

The application must be simple and easy to use, providing that is will have a short learning curve so that current drivers and new drivers can adapt to the application. Thus as a requirement, the application must have a simple GUI display, that is both easy to navigate and takes less effort for the driver to use. We want the drive to be safe on the road, having this feature will allow the drive to focus on the road and if the driver must correct or adjust his route while the go it will be quick and simple. Without this feature drivers will waste time navigating through the application potentially endangering themselves and others while on the road. This requirement can be checked off as being successful by user-testing where they can navigate the application within a minute.

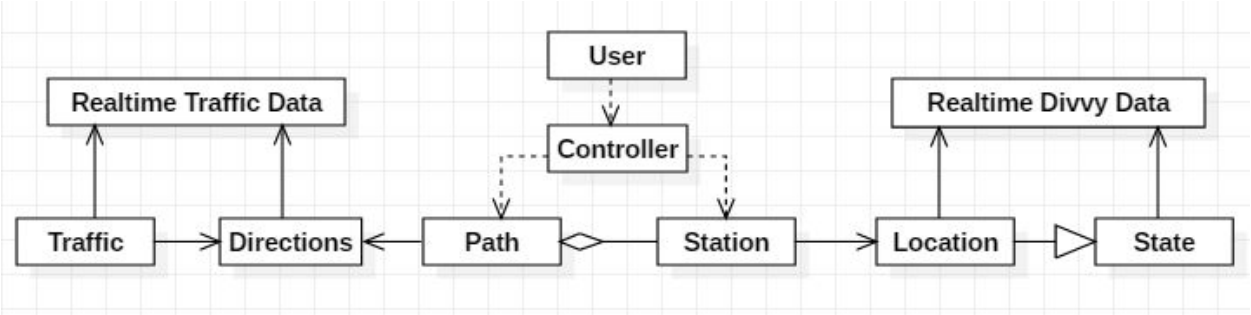
Awkward

Testing

To complete the requirements from above, unit tests and inspection are to be conducted. The application should be tested on its basic functionalities, such as whether it can report back how many bikes should be transferred from one station to another. Also, does the application

show the shortest route to the next station for the driver? All of these requirements must be tested.

Design



Project Issues

There are various issues that can come up during the development process, during the testing phase(s) or even after the product is released in its Alpha, Beta or Final release. Some of those issues can come in the form of whether or not the application can account for road closures, the station is closed due to construction, the application display the fastest route correctly, or whether the customer return the bike correctly to the station: did they put the bike back in place so that the system can recognize it? Some of these issues may be avoided through the process of fault detection by careful programming and detailed testing and inspection of the code before releasing the final product.

Okay summaries of different sections of the proj