University of Glasgow | School of Computing Science

Developing a Bike Share System Web Application

Ananth Padakannaya
Guangyang Li
Hanchi Zhang
Hendivan Achmad
Klementina Mezesova
Li Wang
Peiyi Zhang
Shirui Zhou
Zihe Xu

Level M Team Project — 3 March 2021

Abstract

Bike share applications are gaining popularity across the world due to the rise of the sustainability movement. Creating a real-world bike share system requires a combination of mobile payment, remote tracking and locking technologies. The goal of this project is to develop a functioning end-to-end prototype of a bike share system with basic functionality and interface. The team followed the seven phase system development life cycle and used a combination of agile and waterfall methodology. Firstly, the requirements of the bike share system were informed by market research and analysed using the MoSCoW prioritisation method. Secondly, the team designed the system architecture, MySQL database and webpage interface. The final product is a web application that gives customers the ability to rent, return and pay for a bike near their location. Operators can repair and move bikes, visually aided by a map. Managers are able to generate various types of charts that inform the success of the application. Lastly, the team reflects on the final product and makes suggestions for its further development.

Education Use Consent

We hereby give our permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic format.

Name:	Ananth Padakannaya	Signature:	Ananth Padakannaya
Name:	Guangyang Li	Signature:	Guangyang Li
Name:	Hanchi Zhang	Signature:	Hanchi Zhang
Name:	Hendivan Achmad	Signature:	Hendivan Achmad
Name:	Klementina Mezesova	Signature:	Klementina Mezesova
Name:	Li Wang	Signature:	Li Wang
Name:	Peiyi Zhang	Signature:	Peiyi Zhang
Name:	Shirui Zhou	Signature:	Shirui Zhou
Name:	Zihe Xu	Signature:	Zihe Xu

Contents

Chapter 1

Introduction

1.1 Motivation

Bike sharing applications are gaining in popularity all across the world. The period of lockdown in the United Kingdom created an unprecedented level of demand for bike rentals [1]. This trend is likely to continue as the younger, more environmentally conscious generation views cycling to be the most sustainable option of public transport [2]. Bike share systems as they are known today were pioneered in China in 2015. They combined existing technologies such as mobile payments, GPS tracking and remote locking to create the product familiar to users today [3]. Noticing the emergence of bike share systems, car share companies such Uber and Lyft developed their own versions. Uber found itself in a good market position to advertise bike sharing as "Uber, but for bicycles" [4]. To combat poor quality of air, major cities such as London, New York and Shanghai set up publicly funded bike share schemes [2]. The presence of bike share systems in cities has been found to increase urban mobility, improve health and reduce travel costs [5]. After initial investment in a stock of bicycles, upkeep of the system has generally low maintenance costs [2]. Therefore, it can be concluded that bike share applications are important and the process of developing such software system presents a relevant challenge.

1.2 Aims & Objectives

The aim of this project is to develop a software system that incorporates basic principles of a bike share scheme. The final product will be a functioning end-to-end prototype of a bike share web application, with basic functionality expected of such system. The product will provide a basic interface that allows customers to rent, return and pay for a bike. Operators will be alerted about any defective bikes and oversee their maintenance. Managers will be able to generate reports that provide insight on usage data and total profit in a given time period.

The main objectives are that the finished application is intuitive, user-friendly and visually appealing. The user interface should be easy to navigate even for an inexperienced user. The application should give cues on how to use its features, either with prompts or descriptive buttons. Finally, the hope is that the customer will be inspired to become a premium member of the scheme.

Chapter 2

Market Research

2.1 Existing Bike Share Systems

2.1.1 Nextbike

Nextbike, found at `www.nextbike.co.uk`, is one of the most popular and well known companies that operate a public bike share system. Founded in Germany in 2004, it now operates in over 200 cities around the world. Their mission is to increase the use of bicycles as means of public transport in cities [6]. They believe this can be achieved by providing people with an easy way to access a bike. Nextbike works closely with communities to ensure that the bike share system makes sense for the local area and allows cities to brand the scheme [6]. The company has an approximate annual revenue of £7.5M and operates around 30,000 bikes worldwide. Many universities in the United Kingdom offer free membership of the scheme, including University of Glasgow [7]. The Glaswegian scheme was originally set up with 400 bikes across 30 stations and hopes to hit the target of 1000 bikes across 100 stations [8].

Nextbike is a mobile application available via Android and iOS app stores. To become a member of the community, the user must sign up in the app or on the website. Users can also phone the helpline, which improves accessibility [6]. By becoming a member, the customer has the availability to rent a bicycle in any other city in the world where Nextbike operates. New costumers are required to pay a £5 deposit at the registration stage. This later becomes credit on their account that the customer can spend.

Customers can choose between renting an e-bike or a standard bicycle. Each bicycle has a unique bike number or QR code. To rent the bike, the customer scans the code to release the lock. This starts the renting period until the bike is returned. Nextbike also allows customers to park the bicycle, during which the renting period continues to run. The very first rented bike is free, after which the customer can choose from multiple subscription options. Pay per ride option costs £1 per 30 minutes, full day rental costs £10. Monthly membership costs £10 and discounts the bike price to 50p per 30 minutes. Bikes must be returned to the official station, otherwise the customer will be fined [6].

2.1.2 Lime

Another interesting bike share system is Lime, found at `www.li.me`. Lime was founded with a fleet of electric scooters, later expanding to include electric bikes and mopeds. Founded in United States in 2017, the company's mission is to decrease the level of air pollution in major cities [9]. They aim to reduce carbon emissions by providing an alternative to car rides. Up to 40% of Lime customers report that they have replaced their most recent car trip and reduced dependability on their vehicle overall [10].

Similarly to other bike share applications, new customers sign up and view a map that shows nearby vehicles. Lime allows customers to pre-pay for rides by adding money to their 'Wallet'. Customers scan the bike code to begin their rental, which is charged per minute at the local rate. Lime also offers full day or unlimited ride passes [9].

Lime has a notable model for their fleet maintenance. Lime users can sign up to become a 'Lime Juicer', a gig worker who completes tasks for the company [11]. These users can view the application in 'Juicer Mode' and receive payments of $8-12 for completing tasks such as relocating bikes, putting them in the charging station or driving bikes to the repair depot.

Despite Lime deploying the free-docking model, most reviews mention that it is difficult to find a bike [12]. In the free-docking model, customers can end the ride anywhere. This leaves the fleet scattered around the city until gig workers pick up the job. Customers also mention that e-bikes are heavy and rides too expensive [13].

2.2 Key Takeaways

The development of the bike share system follows the structured process known as the System Development Life Cycle (SDLC) [14]. The background survey was performed to inform the later stages of development. Firstly, companies either have docking stations or allow bikes to be parked anywhere. The dockless model faces the additional strain of having to collect or move bicycles to other locations in the city. While researchers are in favour of the docking station model, there is no consensus on the ideal number of stations or their placement [5]. The dockless model enjoys better reviews when the bike share system employs operators rather than gig workers.

Secondly, many companies are offering both standard and electric bicycles. The choice of extra motor aid certainly makes the application more disability-friendly. The main disadvantage of offering e-bikes is the cost of maintenance, which is passed on to the customer. For example, Nextbike charges £2 per 30 minutes on the e-bike as opposed to £1 on the standard bike. Some countries also require customers to be at least 18 years old to ride electric vehicles, which reduces the potential user base.

Lastly, almost all companies offer a premium subscription, loyalty offers or student discounts. This acts as a motivating factor for users to use the application regularly and invite their friends. If the application does not offer some type of discount, other bike share schemes would gain a competitive advantage. Most applications do not have the ability to pre-pay or rent multiple bicycles, which could also differentiate the final product.

Chapter 3

Analysis of Requirements

3.1 Gathering Requirements

Systems are typically developed by businesses for clients [14]. In the context of this project, the lecturer takes up the client role. After receiving the project specifications from the lecturer at a presentation, the team asked questions about the lecturer's expectations for the final product. This helped clarify the scope of the project and determined the minimum capabilities of the system. Requirements typical of bike share systems were gathered from the market survey discussed in Chapter 2. The following analysis of requirements reflects on the needs of the end users and considers both functional and non-functional requirements.

3.2 The MoSCoW Method

The MoCSoW method is a prioritisation technique that categorises the ideas for system requirements by importance [15]. The acronym stands for what requirements the system must have, should have, could have and will not have. In order, they are the critical, essential and nice to have functionalities. Ideas for any extra functionality that the team could develop under the time constraint are included in the could haves. The aim is to implement one of these features into the final product.

3.2.1 Functional Requirements

Must Have

- **Database.** This stores user information, operational information about bicycles and records payment information. Collects relevant data only.
- **User interface.** Users need to interact with the system to select which features they want to use. Can be a simple GUI or web page.

- **Registration and login page.** Users can set their own username and password. Operators and managers have separate login page or internal staff login details.
- **Customer, operator and manager module.** The system will correctly assign privileges to users based on type.
  - **Customers:** Want to rent a bike close to their current location. The bike should be without defects and unlock using a pin. Customers can report defects. They return the bike at their final destination and pay per minute.
  - **Operators:** View the operational status of every bike. They can repair bikes and move them around the city.
  - **Managers:** Generate reports on bike usage and earnings in a chosen time frame. The reports should use appropriate data visualisation.
- **Available bicycles.** The system should have bikes initialised, so they can be rented straight away.

Should Have

- **Map.** Either a live map or another drawn up representation of the bike's present location. This helps customers and operators visualise where the bicycle is to rent or repair it.

Could Have

- **Discount.** A discount for new, premium or frequent customers. This would reduce the price to rent the bicycle.
- **Loyalty points or coupons.** Customers collect loyalty points for using the app, redeem the points or input codes for free rides.
- **Bike types.** Customers choose between standard bike or e-bike. Prices would vary.
- **Multiple bike rentals.** Ability to rent more than 1 bike at the time. Customer would be asked how many bikes they wish to rent.
- **Park.** Ability to park the bike for a short period of time, during which the rental period is still running.

Will Not Have

- **Secure payment system.** The payment system will not be official or secure. It is not advised for the customer to use real bank details.
- **Customer support.** Users will not be able to call or chat with someone if they experience difficulty.

3.2.2 Non-functional Requirements

- Bike share system is intuitive and easy to use.
- User interface is visually pleasing and interactive.

Chapter 4

Design & Implementation

4.1 Methodology

To design and implement the bike share system, the team used a combination of waterfall and agile methods. The waterfall method is based on the concept of phase-wise building and integration of the application over a long period of time [16]. The waterfall method was used to sequentially complete the phases of requirement analysis, planning, implementation and testing. The main advantage of using the waterfall method was its structured approach to documenting and reviewing each phase. On the other hand, the agile method breaks down tasks into deliverable and usable modules [16]. The agile method was mainly used during the design and implementation stage. Each week, the team listed tasks that individuals then worked on. Closer to the delivery date, there was a collective call for feedback and the individual modules fed into the final product.

4.2 Choosing Technologies

The main tools the team used for developing the bike share web application were the programming language Python, database service MySQL, web framework Django, prototyping tool Axure and HTML toolkit Bootstrap. The project was managed using a Gitlab code repository, where work was coordinated among all team members.

4.2.1 Python

The main programming language used in this project is Python. The team was introduced to Python in the Programming and Systems Development course. The team members come from different academic backgrounds and have various levels of programming expertise. Python has syntax that is easy to learn, which made the process of developing the project easier for beginner programmers. The team members used Pycharm and Spyder to compile the code.

4.2.2 MySQL

The team decided to use the MySQL to create and support the database. MySQL is an open-source database managing tool that uses Structured Query Language (SQL). Most of the team members were new to software development and MySQL is known to be easy and flexible to use. Additionally, this database service supports different programming languages including Python.

4.2.3 Django

Given the choice of main programming language Python, the team decided to use the web framework Django to create the web application. Django is free and open-source. Django is designed to aid web development and allows the team to focus on back-end development.

4.2.4 Axure

The team decided to use Axure to create a prototype. Axure helps create visual representation of the web page without any coding involved.

4.2.5 Bootstrap

To finalise the admin's interface and web page the team used Bootstrap, one of the most popular and free sources toolkits used for front-end development. Bootstrap includes CSS, HTML and JavaScript based design templates and other interface objects.

4.3 System Architecture

Firstly, the team created a simple system diagram depicted in Figure 4.1. This diagram shows a relationship between the modules and features accessible within a particular module, in line with analysis of requirements.

One of the goals of the project is to integrate one additional feature. After reflecting on the background survey and MoCSoW analysis, the team decided to implement a discount or voucher in the customer module. This completed the review of functional requirements. Next, the team to created a full visual representation of the features, see Figure 4.2.

The final system architecture diagram is depicted in Figure 4.3. It summarises the main components, modules and technologies chosen to develop them.

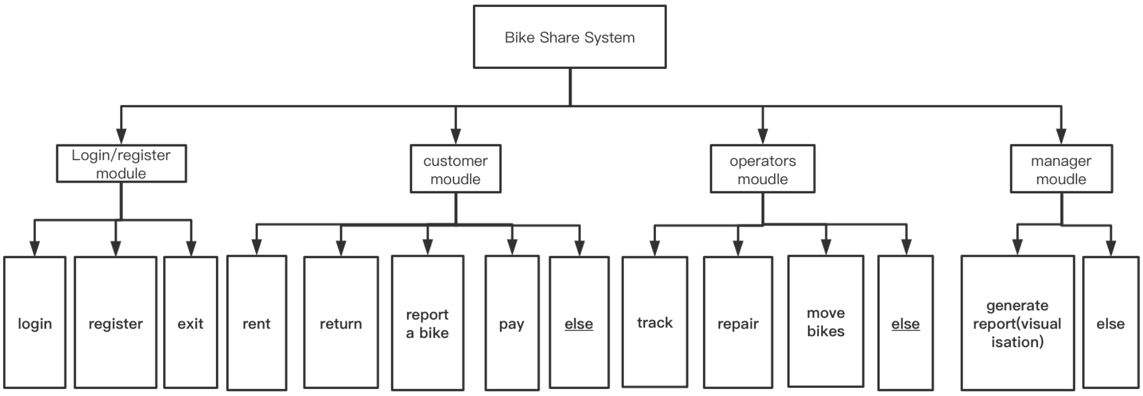


Figure 4.1: Bike share system modules

4.4 Database Design

Database is recognised as a must have feature of the bike share system. It stores the incoming user information and supports all the functions that the system needs to achieve. In fact, each of the modules records or receives information about users or bicycles. The basic structure of the database design can be seen in the entity-relationship (E-R) diagram, see Figure 4.4.

There are 3 tables in the database: customer_info, bike_info and pay_info. It was clear from the registration & login module that a table to store user information is required. For the functions in the customer and operator module, it was necessary to set up a table with bicycle information. Both customers and managers want to keep track of past rides and payments made on the app, hence a table for payment information was needed. Note that fields in the database cannot be NULL.

First, let us look at the details of the customer_info table. This table stores all information regarding the customers. Each customer gets a unique ID upon registration, thus user ID became the primary key of this table. The customer sets a username and password, then fills in personal details, such as phone number, e-mail address and bankcard number. The database also takes in the customer's GPS coordinates, where x represents the latitude and y represents the longitude. These are necessary to calculate user's distance to the nearest bicycle. The table stores the rent time in minutes and remaining balance on customer's account. Lastly, the table stores the user type: 0 represents a normal customer, 1 the operator, 2 the manager and 3 the prime (premium) customer.

Secondly, the bike_info table stores all information regarding the bicycles owned by the scheme. Each bike has a unique ID, thus bike ID became the primary key of the table. The operational status of each bike is stored in the bike status column, where 0 means the bike is functional and 1 means it is broken. The bike problem column stores the message customer wrote about the issue. Available bikes have bike usage set to 0, when the bike is rented out its status is 1. Each bike has a unique 4-digit bike password. Each bicycle has GPS coordinates, again with latitude and longitude. Additionally, each bike falls into one of the 5 bike areas (A, B, C, D, E). These roughly represent the city centre and the compass rose, see Figure 4.5 for details.

Finally, the pay_info table's primary key is the payment ID. Each processed payment will set a unique payment ID and takes note of the user and bike IDs. The table also records a start time and end time of the bike rental. This is used to calculate the duration column, which is then used

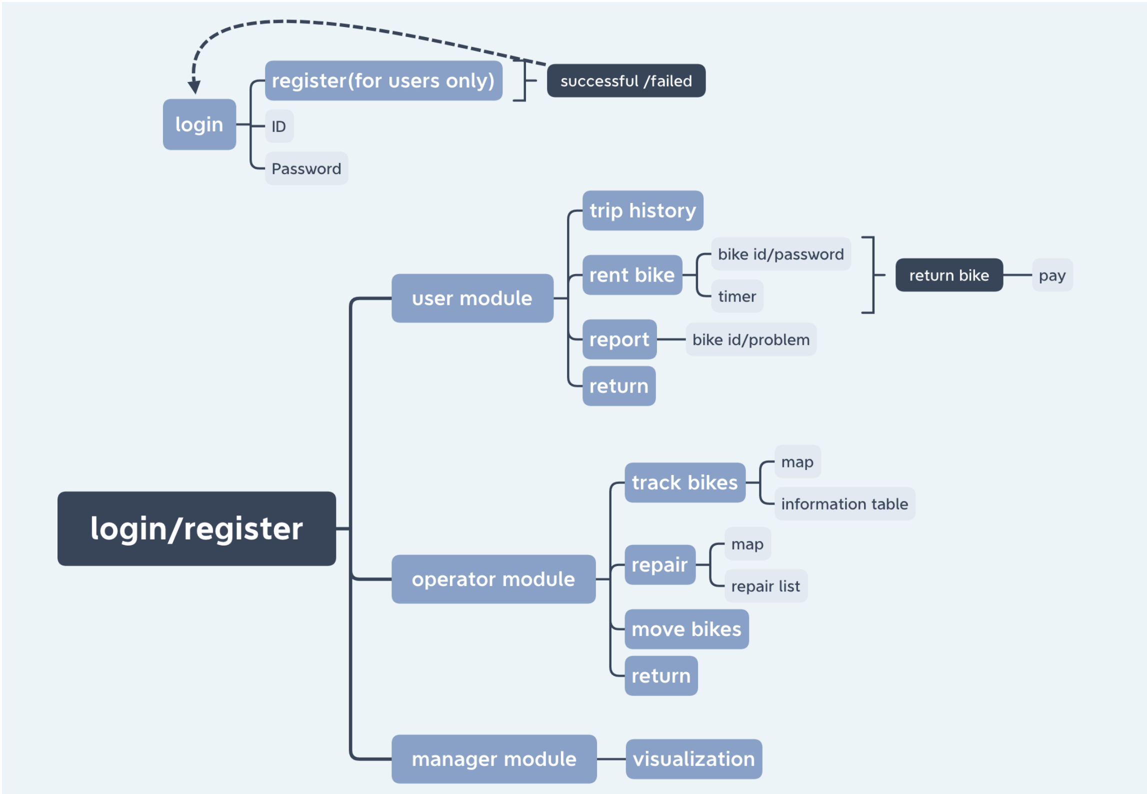


Figure 4.2: Representation of functional requirements

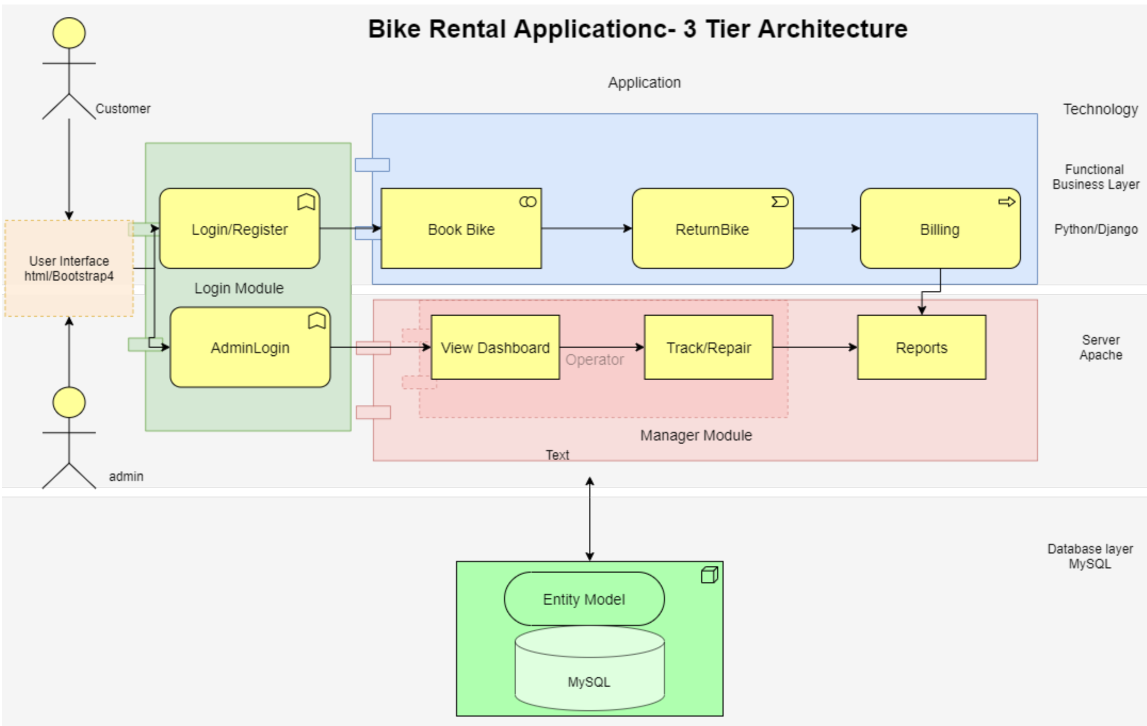


Figure 4.3: System architecture diagram

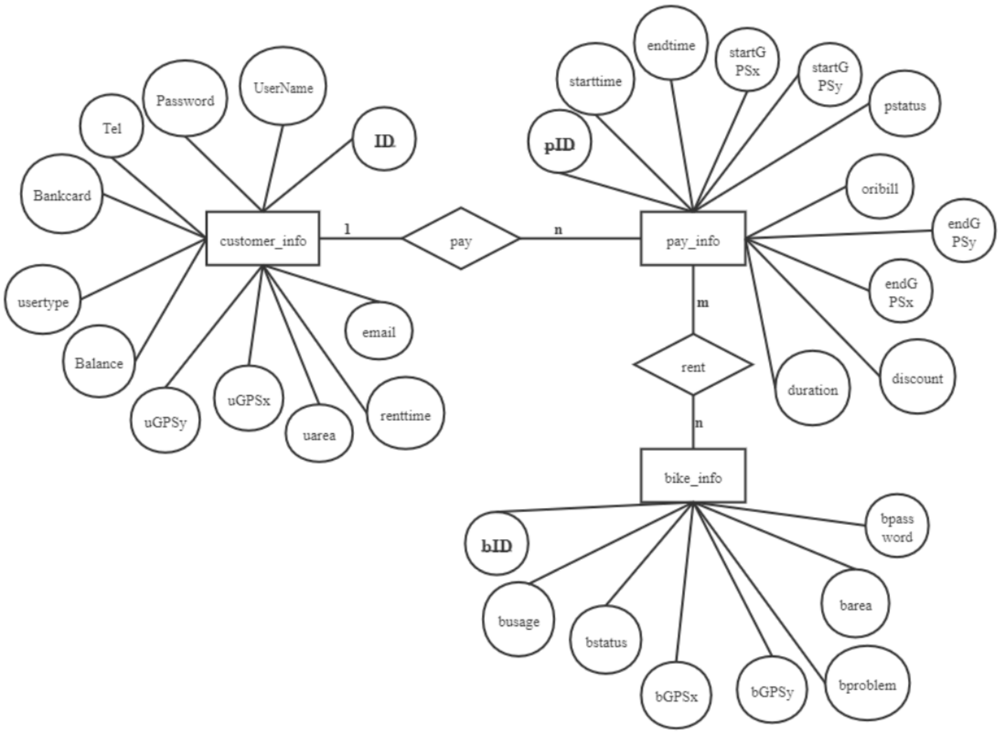


Figure 4.4: E-R Diagram

to calculate the total bill. Start and end coordinates of the ride are noted, in particular the end coordinates become the new bike coordinates. Lastly, the payment status shows if the payment was completed. Status 0 means that the customer has an outstanding balance and 1 means the bill has been paid.

4.5 Prototype

The team created a prototype for the bike share system webpage using Axure. This is a basic code-free model of the system, that helped finalise ideas for the user interface. The prototype became the main reference point during the implementation process. For pictures of the prototype, see Appendix A.

4.6 Implemented Features

This section discusses the bike share system features and how they work. For figures relating to the section, see Appendix B.

4.6.1 Register & Login Module

Register:

User will first need to register. In the registration page, user fills in a form. They set their unique user ID, username and password. They will type the password again to confirm it. The form also needs personal details such as telephone, bank card details. Then, the user clicks on the register button and the data is stored in the back-end database.

A random function is used to generate random coordinates for the customer's location in Glasgow. After the user has successfully registered, the system will store their geographic location coordinates and area in the database. At the same time, the default user type and usage time will also be stored in the database along with registration.

If the registration is successful, the successful registration prompt statement will be displayed and guide the user to log in. If the registration fails, the failure prompt will be displayed and guide the user to re-register.

Login:

To log in, the user enters the username and password. If the login is successful, the user will enter the main interface of the system. The interface will show the user what features of the bike share are available to them according to their user type. If the login fails, the system will display a login error message and remind the user to register.

4.6.2 Customer Module

Rent a bike:

When the customer wishes to rent a bike, they click on the rent button. Upon request, the function finds all the bikes in the database that are available to use and without defects. In the next step, the function needs to find the nearest bike in this pool. The user is located at uGPSx, uGPSy and each bike is located at bGPSx, bGPSy. The function calculates the distances between user and bikes using the distance formula,

$$d = \sqrt{(uGPSx - bGPSx)^2 + (uGPSy - bGPSy)^2}$$

and selects the bike which corresponds to that value. The usage of that bike will be updated. It returns the bike ID and password so that the user can unlock the bike. The rental period starts running and the function records the user, bike, start time and start location for the payment information table. If there is no available bike, the function returns a message. If there are more than bikes nearest to the user, the function will assign one of these bikes.

Return a bike:

When the customer returns a bike, the function sets the bike usage field to 0. This makes the bike available to the next user. The rental period stops running. Since the bike is returned by the customer at their final destination, so the new bikes coordinates are the user's present GPS coordinates. The function updates the record in the payment table with the end time and end location. This allows the pay function to calculate the bill by taking the difference in start and end time.

Pay & Discount:

This function will be called when user clicks on the pay button. The function will calculate the duration of the rental period in minutes. If the user rides for more than 500 minutes, they receive a 20% discount on their bill. This is around 8 hours, representing a day trip outside Glasgow or similar longer rental. If the payment is successful, the back-end database completes the payment table information with duration, amount paid and clears the users outstanding balance status for that ride.

Report defective bike:

Every user can report a defective bike. The operator needs to know which bike is broken, so the reporting form takes in the bike ID and description of the problem. Users can report any bike in the city for repair. If the report is submitted successfully, a success prompt will be returned. Otherwise, a failure prompt appears. The database will change the operational bike status from 0 to 1, meaning that the bike will not be available to rent out. This also reports the problem to the operator, who is able to see the defective bike and user's message about the issue in the operator mode.

4.6.3 Operator Module

Track bikes:

To track all bikes, the operator is provided with a map and a table. There are 3 types of markers on the map, the usage status and operational status will appear when the operator hovers the mouse over the markers. The green dot represents unoccupied, functional bikes. The yellow dot represents bikes that are occupied. The caution marker signals a broken bike. Underneath the map, there is a table that contains all the information about bikes relevant to the operator.

Repair bikes:

The map contains a prompt that tells the operator how many bikes need to be repaired. When the operator clicks on the button, they can view information about each broken bicycle. Notably, they will see what the customer wrote about the defect. After the bike is repaired in real life, the operator can change the bike's status by choosing the bike ID and clicking repair. This will update the database and makes the bike functional and available to use. The repaired bike will no longer appear in the table.

Move bikes:

In real life, the dockless bike share model leaves bikes scattered around the city. Operators can move bikes wherever they are needed. Firstly, using the set up areas of Glasgow, the page will have a hint about which areas need more bikes as well as the total number of bikes in that area. For example, if the more frequented city centre does not have enough bicycles, the operator might wish to move more bikes to area A. There is a set threshold amount for each area, thus the area needs more bikes if the available bikes are below the set amount. The table will display all the bike information needed to find the bike, including bike ID, status, usage and location. The operator can choose which bike to move by checking the box of the corresponding bike and selecting which area to move the bike into. If they input the wrong area code, they move will not be successful and an error prompt will be displayed.

4.6.4 Manager Module

Data visualization:

The data visualization is the main function of the manager module. It has been created for managers of the bike share scheme, who can view all the data about bikes, rentals and payments. It includes pie charts of user structure and bike status, heat maps of the rate of bike use and density of payments and bar charts of rental duration and daily payment quantity.

The basic information is shown on a map, this displays bike ID number, location, area, availability and defects. Managers can hover their mouse over the map to view this information.

The heat map was created using the folium package and the open-source map openstreetmap. The heat maps informs managers about which locations on the map are more popular with users. Managers can tell where people rent bikes and where the highest demand is.

The remaining charts, which include bar charts, pie charts and liquid charts, were created with the pyecharts package. All the charts are dynamic and interactive. Managers can choose which chart to display and set the label they want. There will be a transition animation when something is changed. Managers can hover their cursor over the charts to see specific numbers.

4.6.5 Bikes & Map Rendering

The bike share system will initialise a number of bikes. They will be randomly scattered on the map, assigned an operational status and availability. This allows the first user to rent a bike and staff will have access to some data for maintenance and data analysis.

To show the bike location and other information on a map of Glasgow, the team used Google Maps API. This is free for developers and only requires to be integrated into the system with the project's database. First it is necessary to establish a database connection and retrieve bike data, particularly GPS coordinates. Secondly, the data is passed to the frond-end templates using the render function. The render function transmits the content in functions to a template, meaning the HTML files, through an HttpResponse object [17].

So far, the data in the database can be passed to the front-end for generating bike information. After a series of data processing, the locations of bikes can be drawn using latitude and longitude. They are shown as markers through the HTML Rendering method. Finally, strings are concatenated, the data is passed from the database, and mouse functions are added to show the information. If users move their cursor and hover at a mark, information about the bike will be shown.

Glasgow city was roughly divided into 5 areas. The city centre is the central area A and the remaining 4 areas point in each cardinal direction. Although this type of distribution may increase computational workload, this structure was preferred to reflect that the city centre is more frequented by users. To see how the areas are distributed, see Figure 4.5.

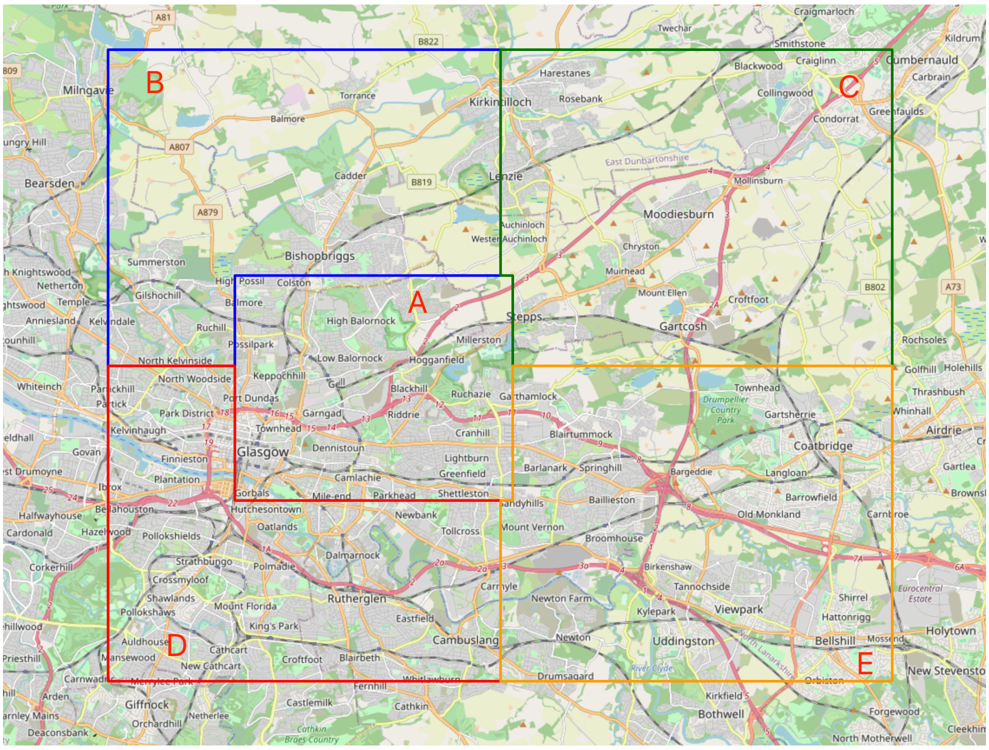


Figure 4.5: Map areas based on GPS coordinates

4.7 User Interface

After developing the prototype of the webpage and completing the modules, the next step was to create the interface. The team's choice of interface-development tool was Bootstrap. The team needed the webpage not only to be modern, but also adjustable to different devices and browsers. This is something that Bootstrap provided. Additionally, Bootstrap is easy to use and learn, which certainly sped up the build-up process. Bootstrap provides a wide range of ready-made templates that can easily be integrated. The team's choice of template can be found at `www.startbootstrap.com/theme/sb-admin-2`. Using a template allowed the team to have more time for back-end development.

Chapter 5

Testing & Demonstration

The bike system was tested by going through different test scenarios as per the developer and verifying the results. The following table shows the test cases after integration and their pass or fail status. The test cases marked with * are expected negative cases.

Table 5.1: Test Table

Test case	**Pass/Fail**
Register user	Pass
Log in	Pass
Log out	Fail
Log in with incorrect credentials*	Pass
Log in redirects to the corresponding user module	Pass
Rent a bike	Pass
Handling null exception in bike coordinates	Fail
Return a bike	Pass
Pay for a bike	Pass
Report a defective bike	Pass
Operator views all bike information	Pass
Repair a broken bike	Pass
Move bike to another location	Pass
Operator moves to invalid area*	Pass
Manager generates available charts	Pass
Map rendering	Pass
Interface	Pass
Page navigation	Fail

For demonstration of the final product please view the attached User Guide and watch the video at

`https://web.microsoftstream.com/video/7a7c51b1-006a-44dd-a074-0b954d833174`

Chapter 6

Conclusion

6.1 Reflection

The requirements of the project were to develop a bike share system that would have basic functionality and interface. The team was inspired by applications that are currently on the market. Nextbike, a bike share that operates in Glasgow functions as a web application as well and the team decided to challenge themselves and design a similar system. The requirements and technologies were chosen by the team appropriately. Using agile and waterfall methodology, the system was implemented and runs locally. The final product is a web application that implements the required modules. The interface is modern and adjustable to different devices and browsers. The system was then evaluated, noting down known errors and suggestions for future work.

The team worked on this project online and from home during trying times in the world. The team members were located in different time zones and consisted of students coming from different backgrounds and pursuing Master's degrees in Data Science or Data Analytics. This meant that the pool of knowledge varied and the team members learned from each other and together. Despite the circumstances, the team believes that this project was a success and aimed to highlight the positive aspects of the finished system.

6.2 Future Work

The team believes that the bike-sharing system met all the minimum requirements, however there are several aspects that need to be improved due to constraints in time or capability. First, the database needs improvements as standard ID was not used, only auto-incremented primary keys. Next the user ID is set by the user and used to log in. The prime user type has been set up but a function for user to become prime needs to be implemented. Customers and operators should also see the exact location of the bike on the map. Some of the features of the sidebar are not well connected and require details work. User can exit the payment page. They will an outstanding balance, but they are prevented from renting future rides.

References

[1] Vice News. Cycling Has Never Been More Popular – Now Could Be the Time to Revolutionise Travel. https://www.vice.com/en/article/n7w3gb/cycling-popularity-coronavirus-lockdown.

[2] Ashish Kabra, Elena Belavina, and Karan Girotra. Bike-share systems: Accessibility and availability. *Management Science*, 66(9):3803–3824, 2020.

[3] CB Insights. The Global Bike-Share Boom: Dockless Models Look To Solve Urban Commutes & Transit Access. https://www.cbinsights.com/research/bike-sharing-boom/.

[4] The New York Times. Uber's new training wheels: Testing bike sharing in san francisco. https://www.nytimes.com/2018/01/31/technology/uber-bike-sharing-san-francisco.html.

[5] Miriam Ricci. Bike sharing: A review of evidence on impacts and processes of implementation and operation. *Research in Transportation Business & Management*, 15:28–38, 2015.

[6] Nextbike. nextbike - Bike sharing company. https://www.nextbike.co.uk/en/about-us/.

[7] University of Glasgow - Cycling with nextbike. https://www.gla.ac.uk/myglasgow/sustainability/travel/cycling/nextbike/.

[8] Nextbike. nextbike Glasgow - 11 new nextbike stations across Glasgow. https://www.nextbike.co.uk/en/glasgow/news/11-new-nextbike-stations-across-glasgow/.

[9] Lime - Electric Scooter and Bike Rentals. https://www.li.me/en-us/home.

[10] Lime - 2018 Year End Report Examines Stunning Growth Of Micromobility. https://www.li.me/second-street/lime-year-end-report-2018.

[11] Lime. Lime - Juicer Mode. https://help.li.me/hc/en-us/articles/360015628874-Juicer-Mode.

[12] Lime. Lime Bikes in London: One Hot, Green, Expensive Mess - ShareCast. https://www.sharecast.com/news/technology/lime-bikes-in-london-one-hot-green-expensive-mess--3684875.html.

[13] Lime. We tried out London's new Lime-E electric bikes and it did not go well - MyLondon. https://www.mylondon.news/news/mylondon-tried-out-electric-bike-16274138.

[14] Innovative Architects. SDLC: Seven Phases of the System Development Life Cycle. https://www.innovativearchitects.com/KnowledgeCenter/basic-IT-systems/system-development-life-cycle.aspx.

[15] Parker Software. Software development and the MoSCoW method - Parker Software. https://www.parkersoftware.com/blog/software-development-and-the-moscow-method/.

[16] Agile Vs Waterfall: Know the Difference Between Methodologies. https://www.guru99.com/waterfall-vs-agile.html.

[17] Django documentation. Django shortcut function. https://docs.djangoproject.com/en/3.1/topics/http/shortcuts/.

Appendix A

Prototype

This appendix contains screenshots of the bike share system prototype.

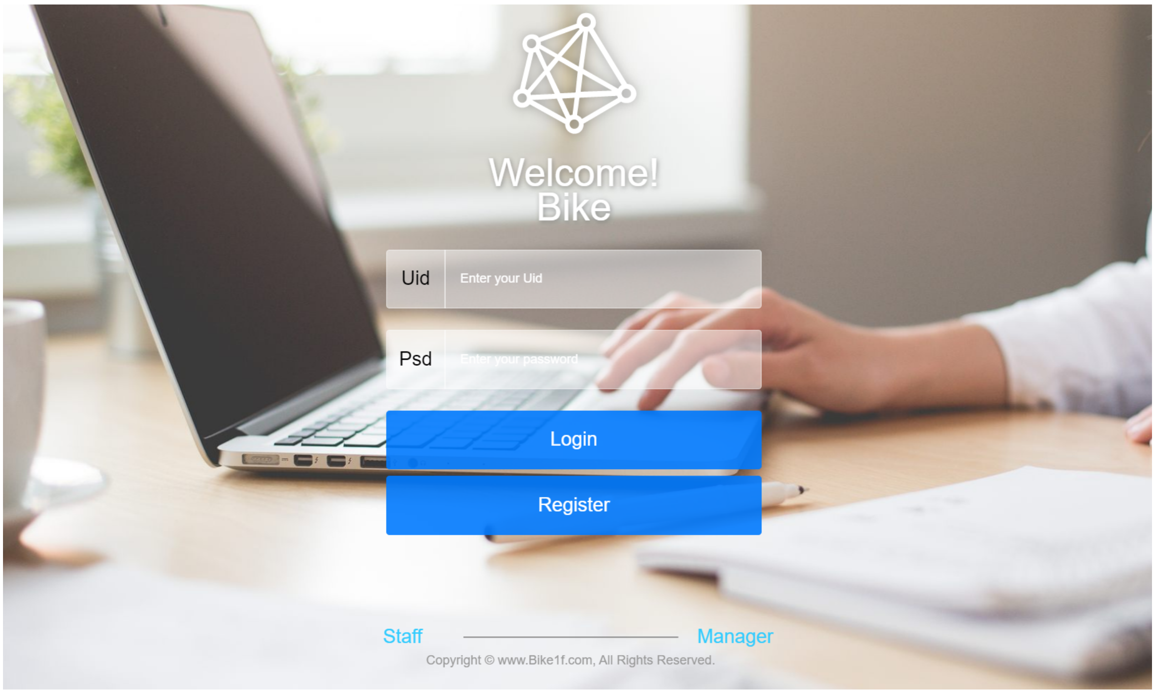


Figure A.1: Prototype - Log-in Page

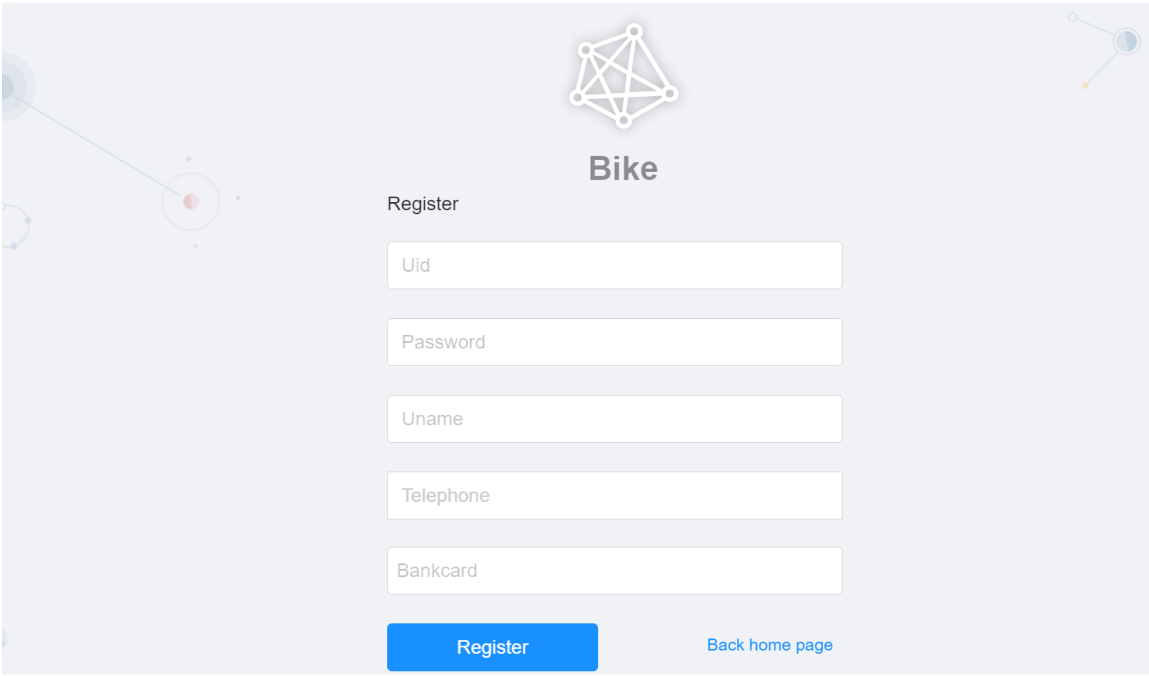


Figure A.2: Prototype - Register Page

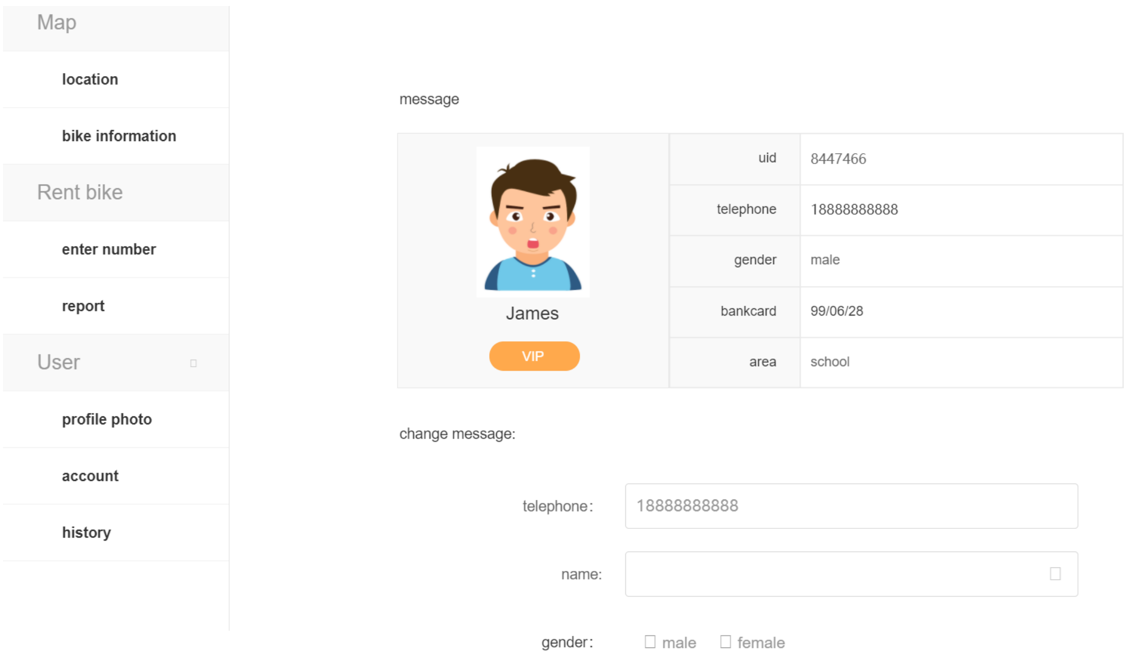


Figure A.3: Prototype - User Page

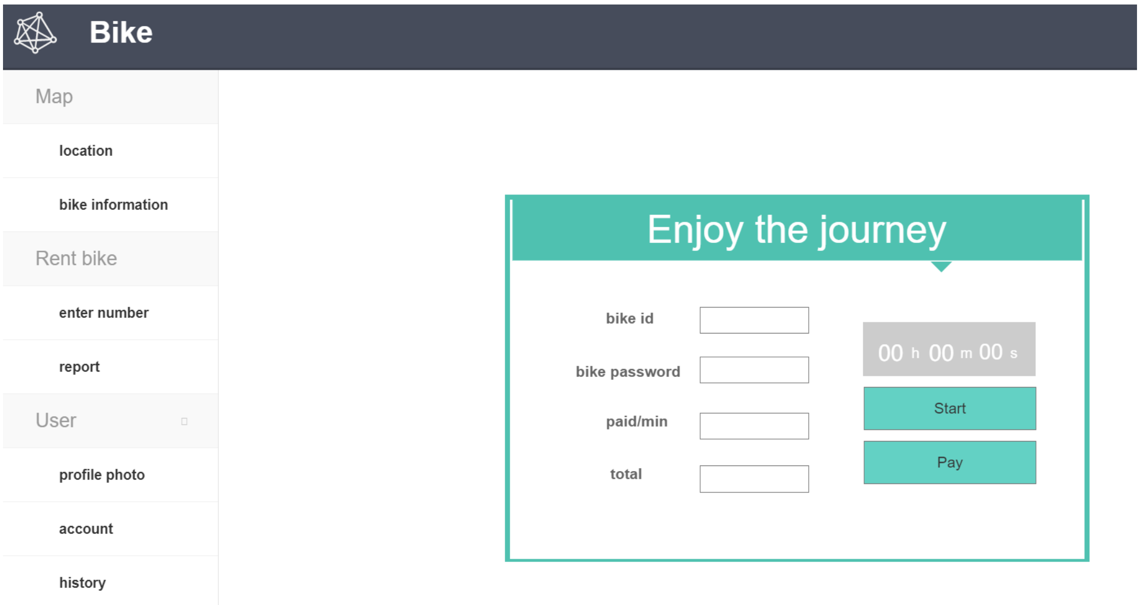


Figure A.4: Prototype - Rent a Bike

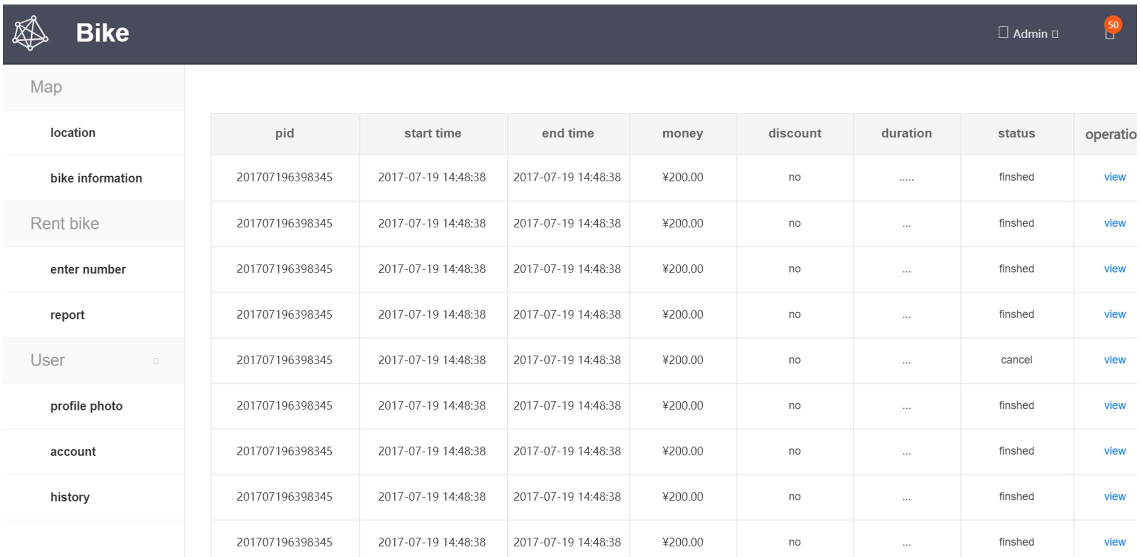


Figure A.5: Prototype - User's Activity History

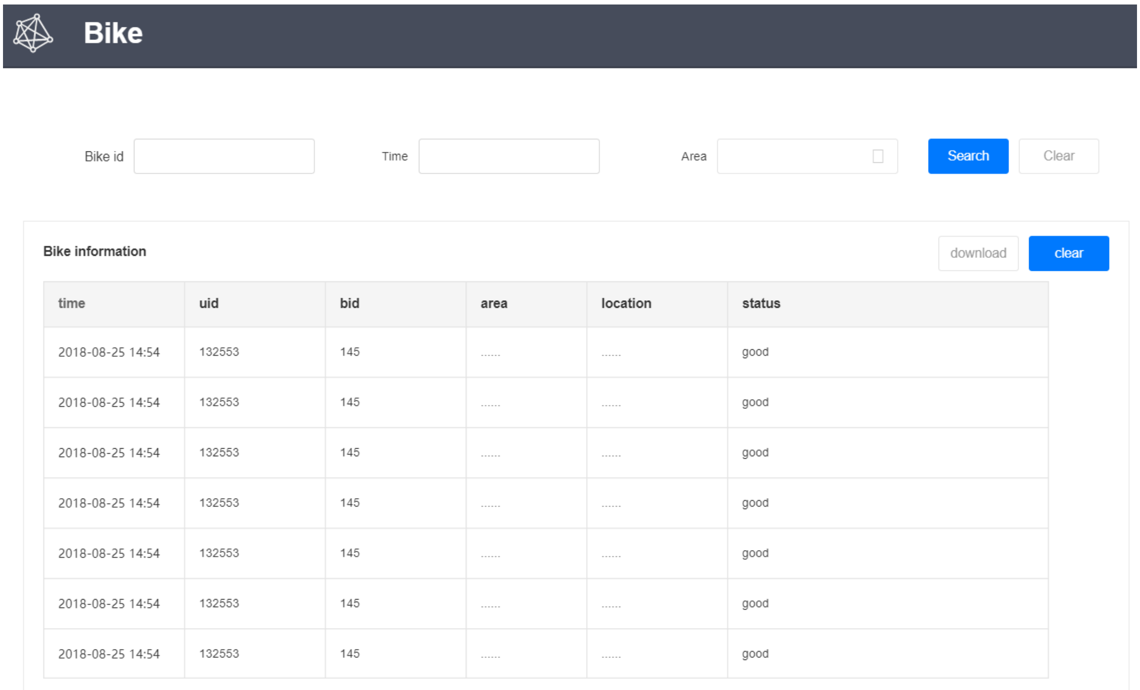


Figure A.6: Prototype - Operator Mode

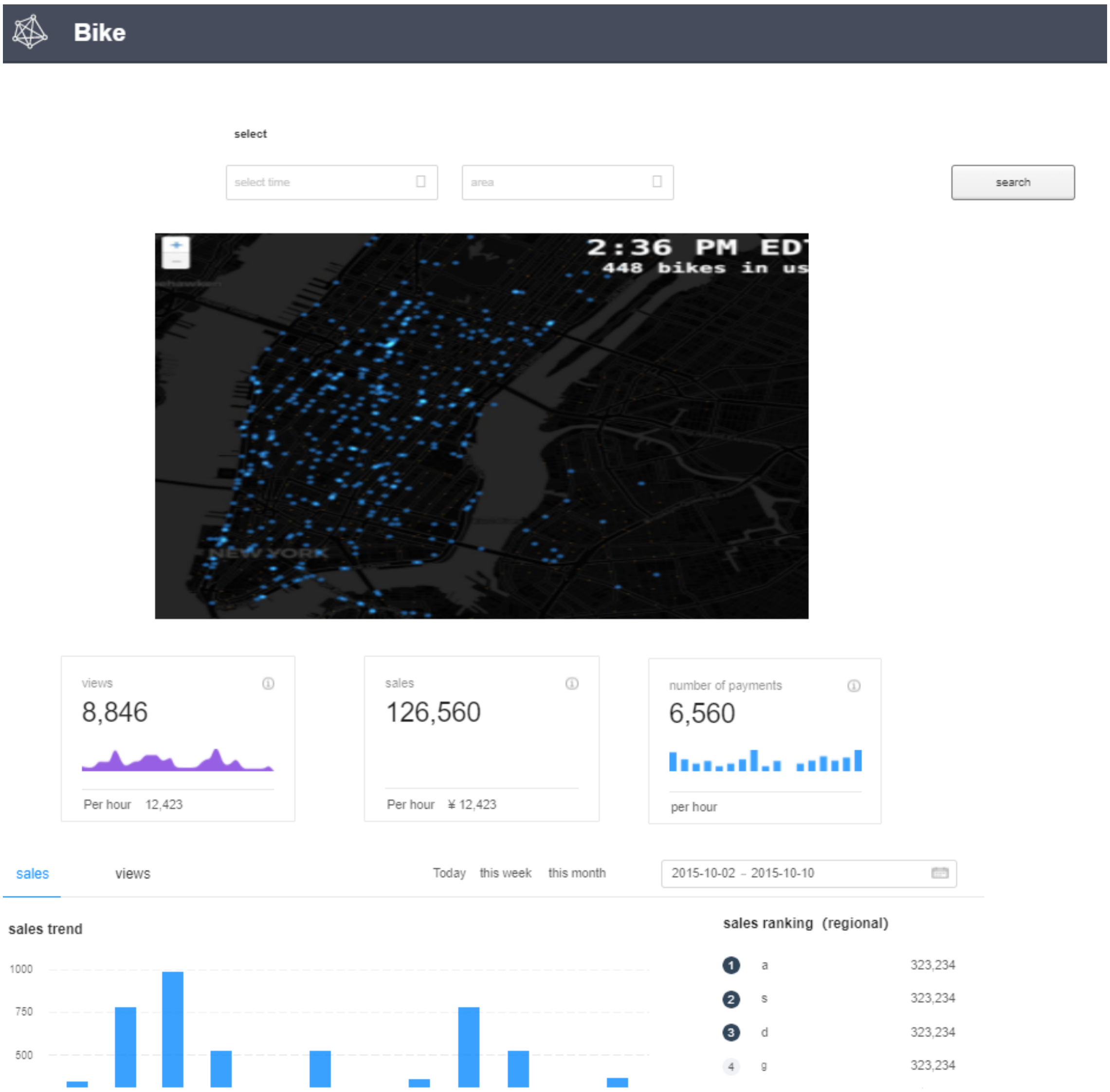


Figure A.7: Prototype - Manager Mode, Part 1

Figure A.8: Prototype - Manager Mode, Part 2

Appendix B

Implementation

This appendix contains additional figures relating to the implementation section. The figures show various features at the development stage.

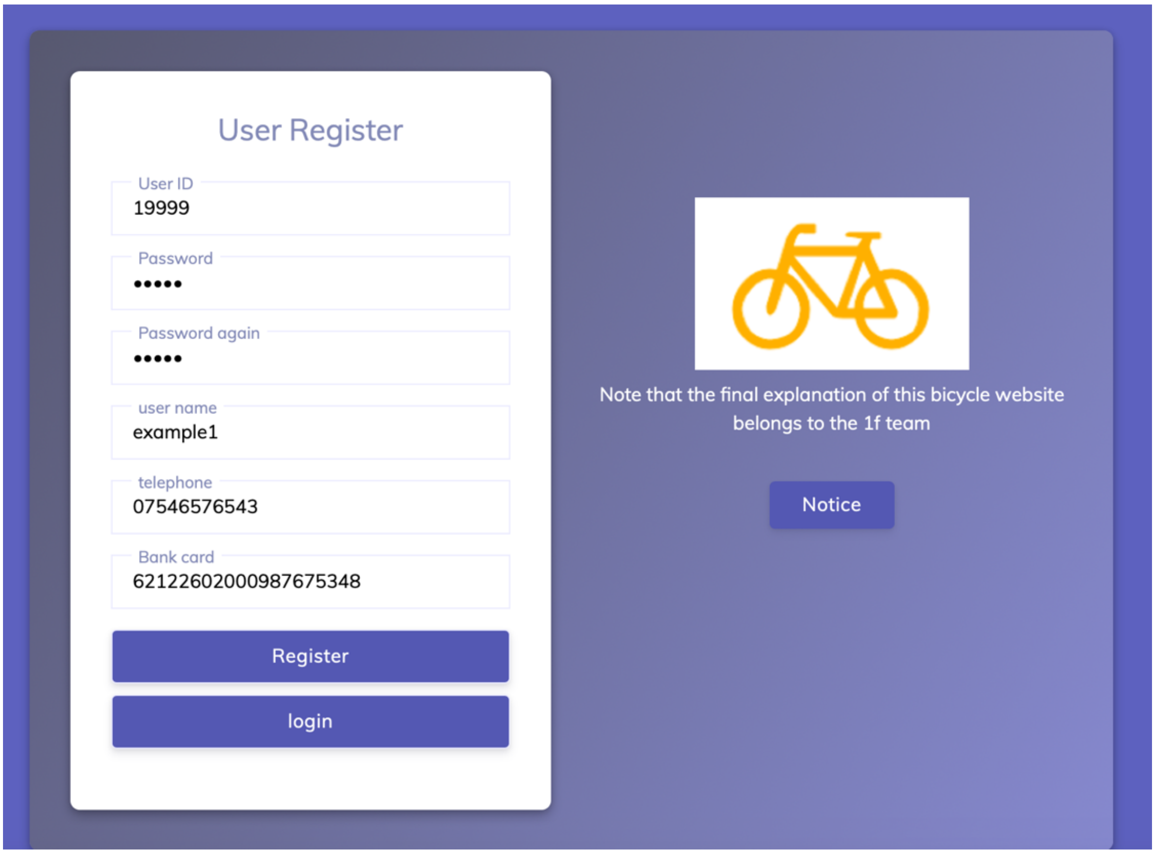


Figure B.1: Registration page

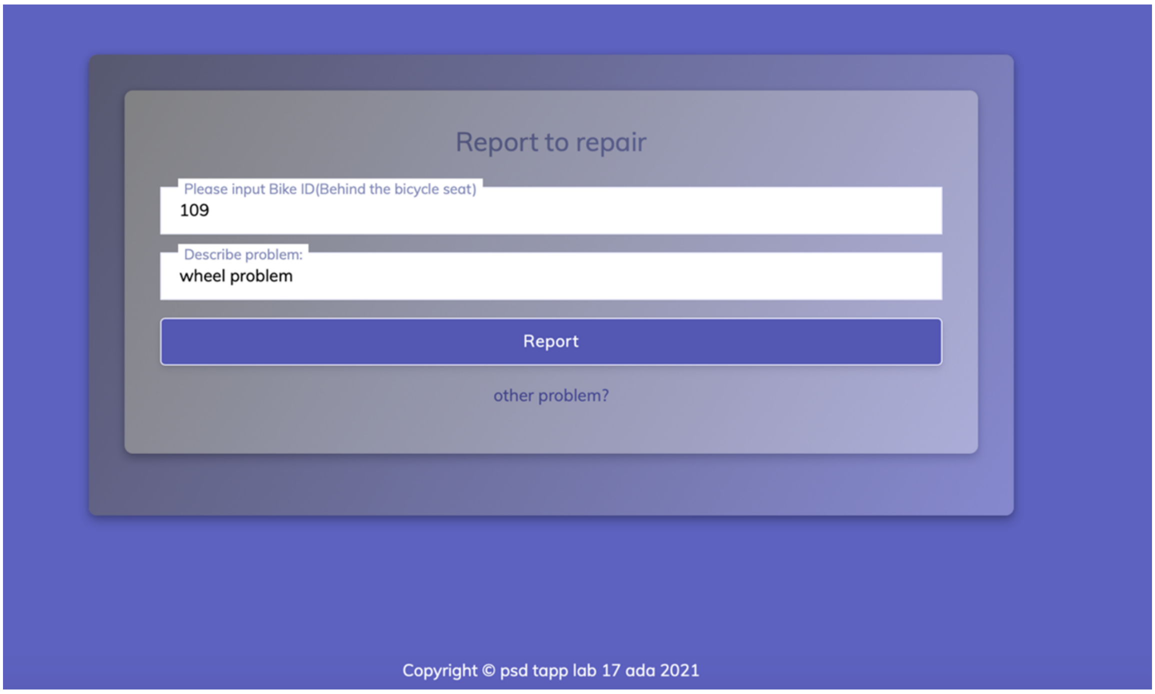


Figure B.2: Report a broken bike

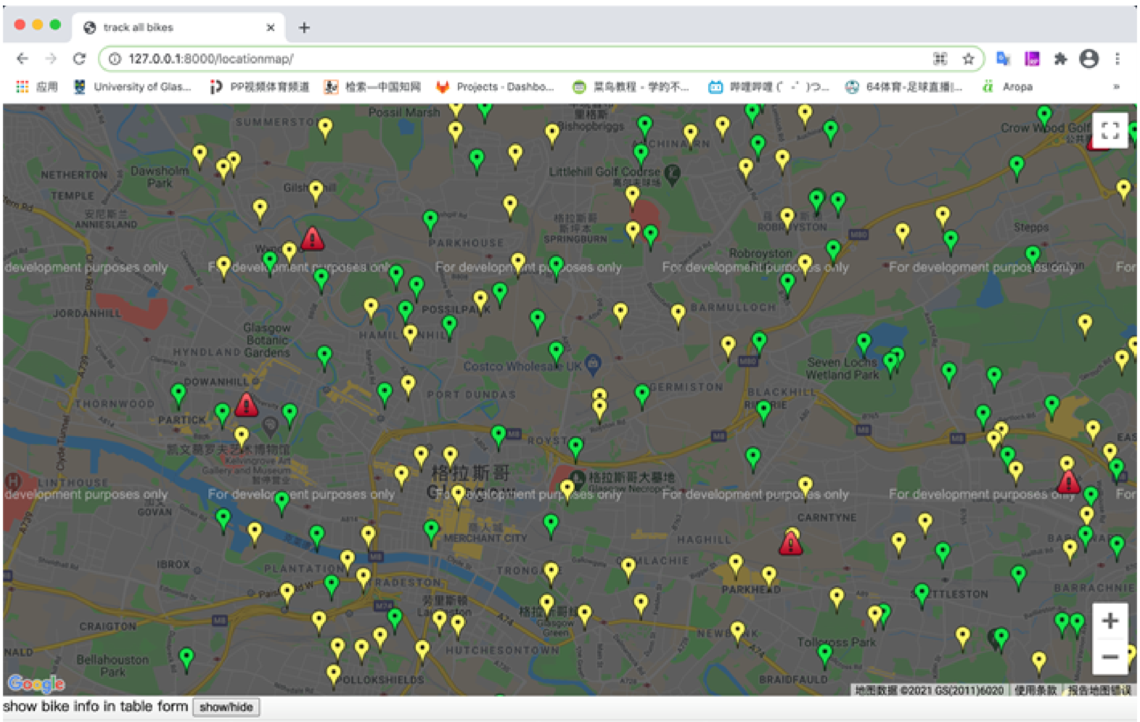


Figure B.3: Types of bike markers

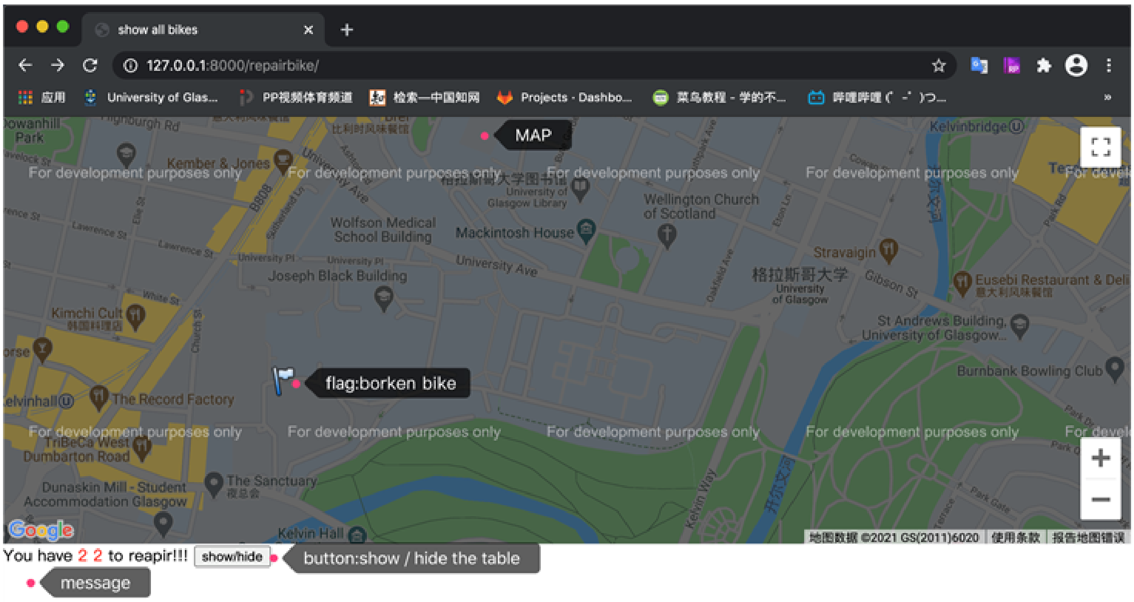
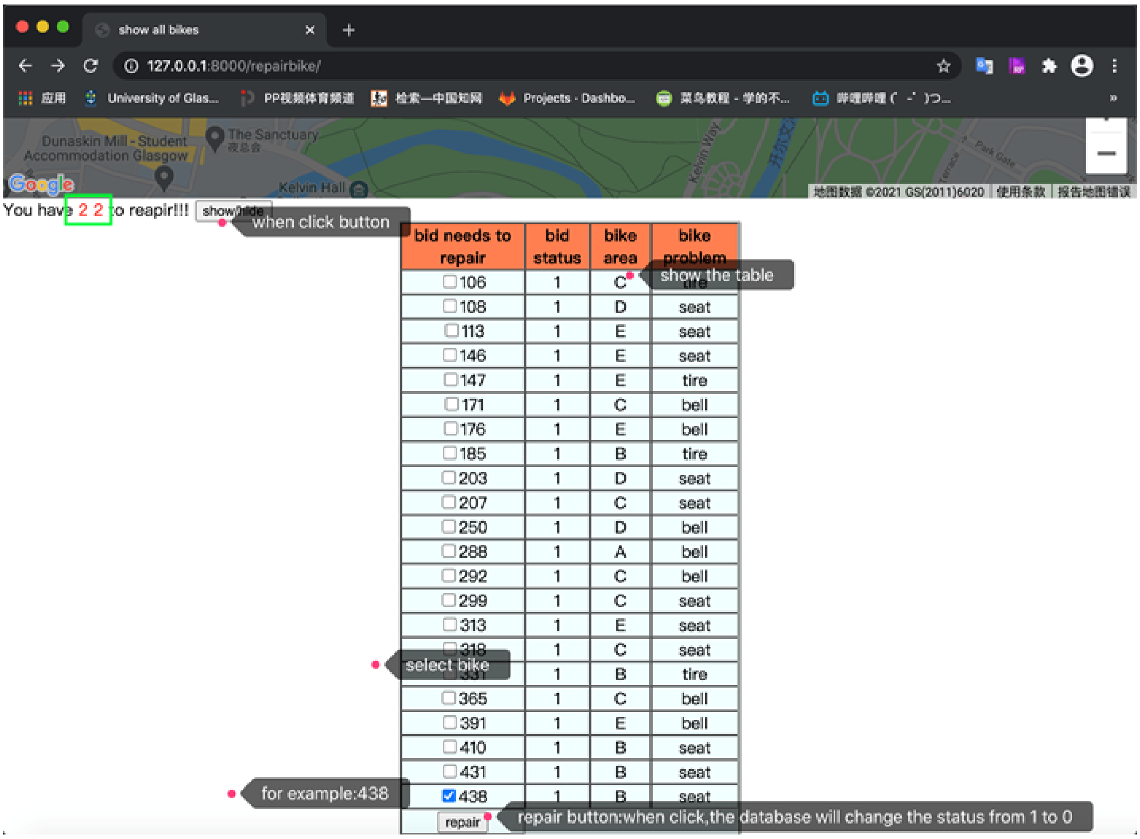


Figure B.4: Details of operator's map



bid needs to repair	bid status	bike area	bike problem
106	1	C	tire
108	1	D	seat
113	1	E	seat
146	1	E	seat
147	1	E	tire
171	1	C	bell
176	1	E	bell
185	1	B	tire
203	1	D	seat
207	1	C	seat
250	1	D	bell
288	1	A	bell
292	1	C	bell
299	1	C	seat
313	1	E	seat
318	1	C	seat
331	1	B	tire
365	1	C	bell
391	1	E	bell
410	1	B	seat
431	1	B	seat
438	1	B	seat

Figure B.5: Demonstration on repairing bikes, Part 1

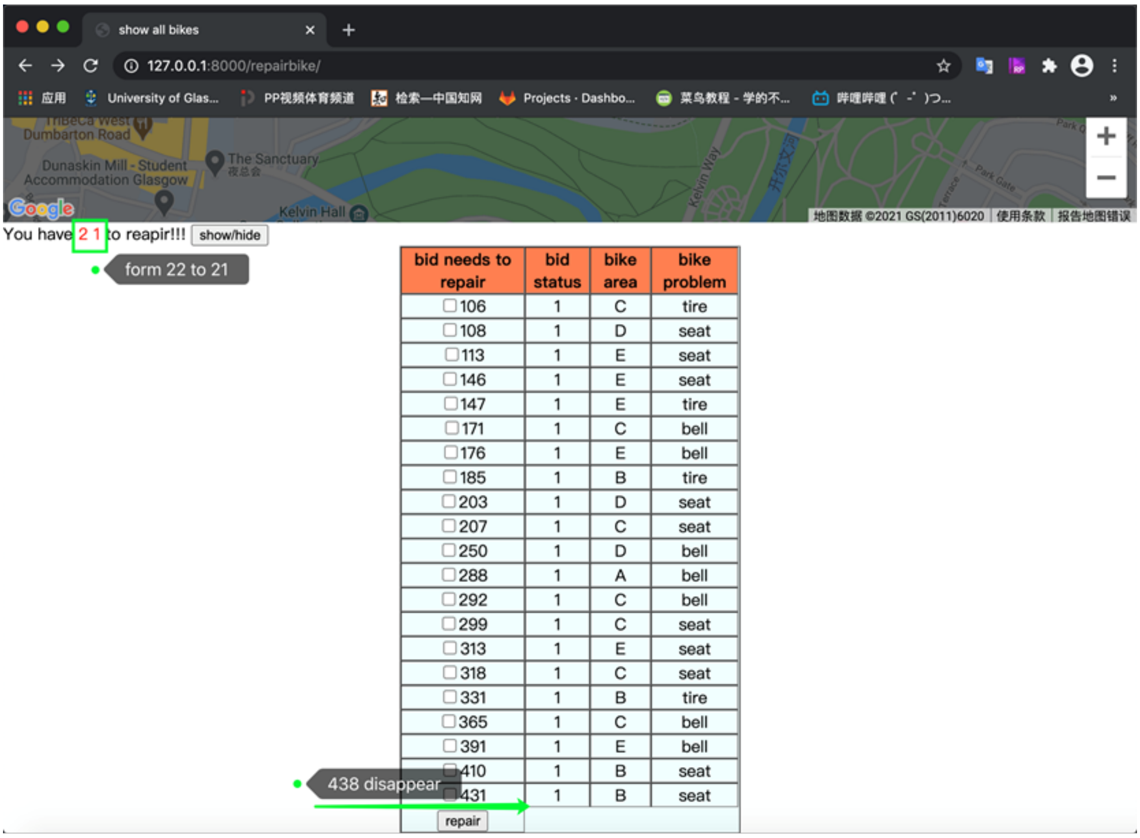


Figure B.6: Demonstration on repairing bikes, Part 2

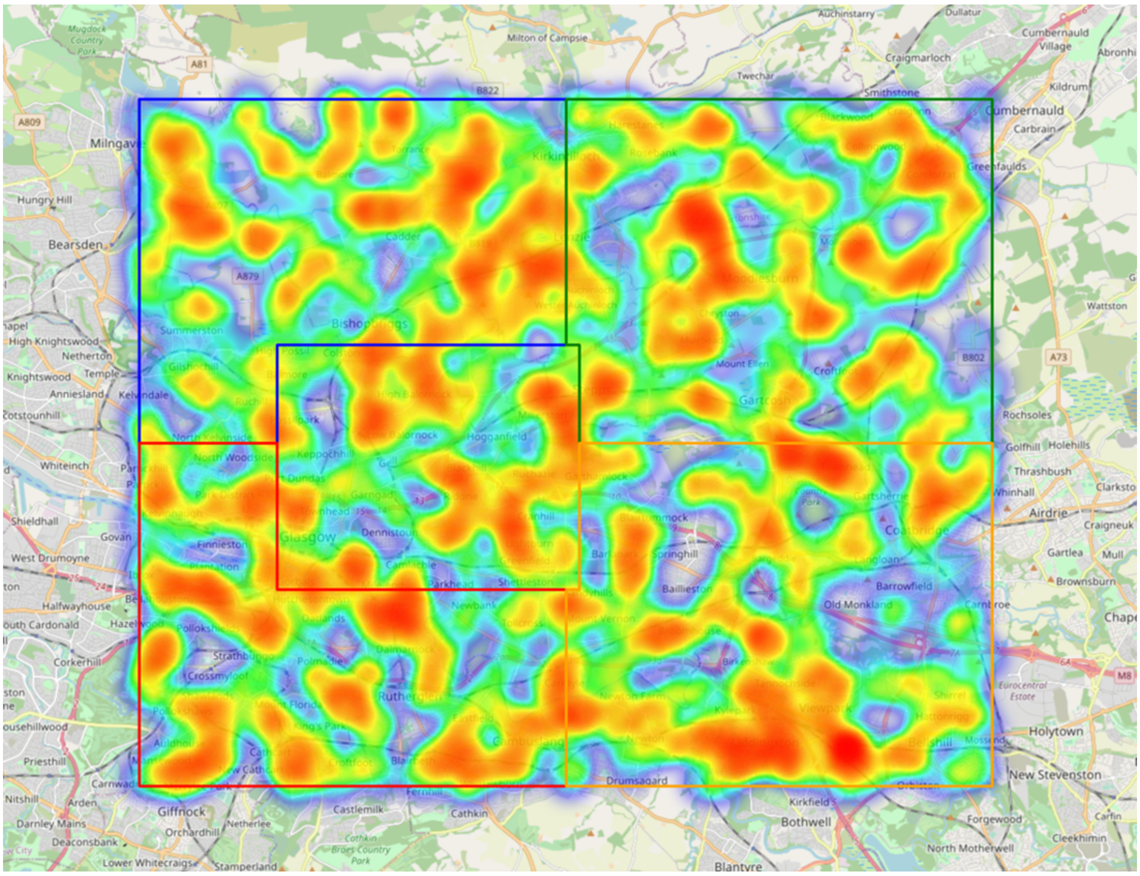


Figure B.7: Heat map for frequency of bike rentals

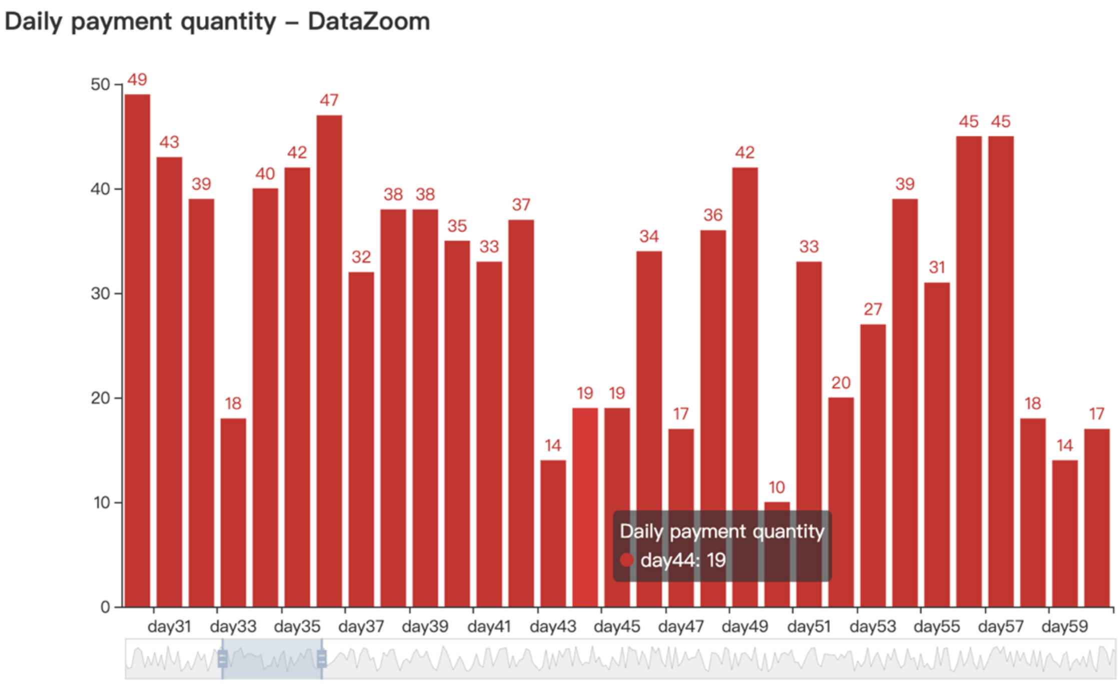


Figure B.8: Records of daily payment totals

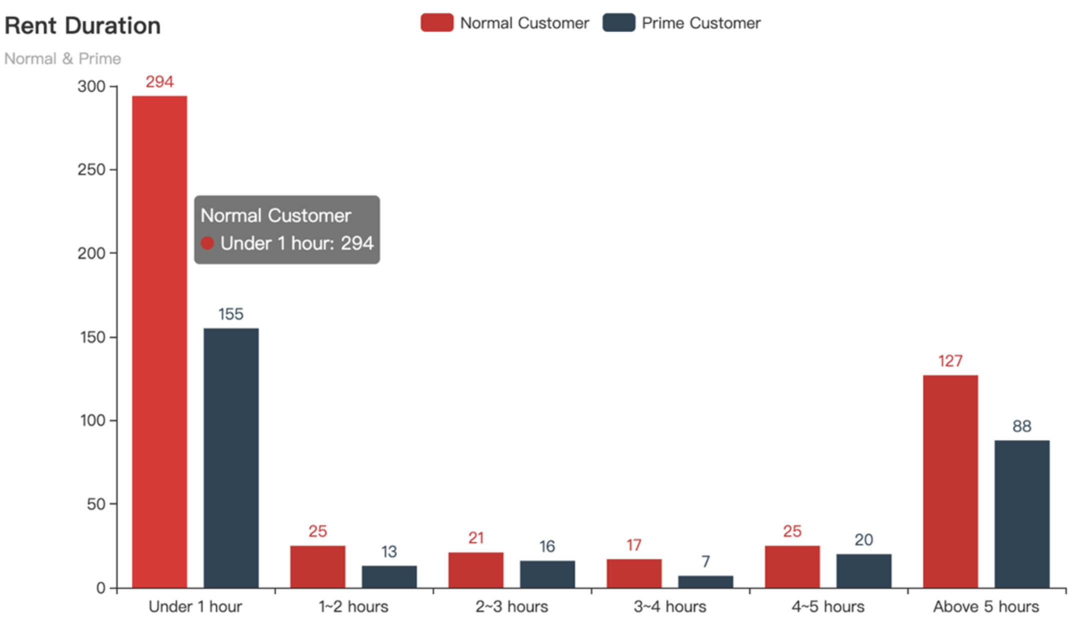


Figure B.9: Records of bike rental duration

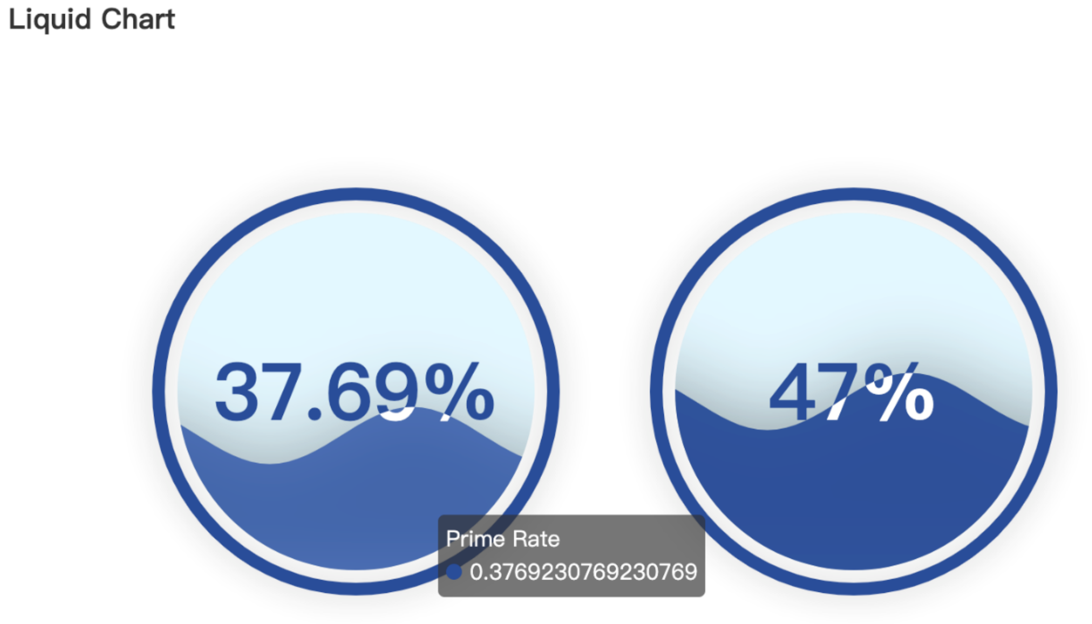


Figure B.10: Percentage of standard and premium customers

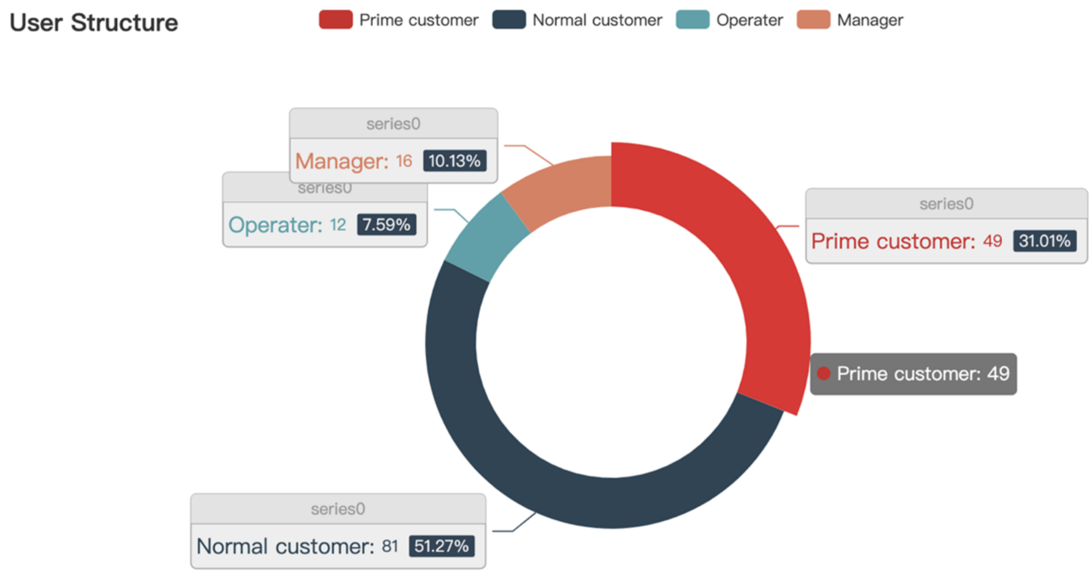


Figure B.11: User structure

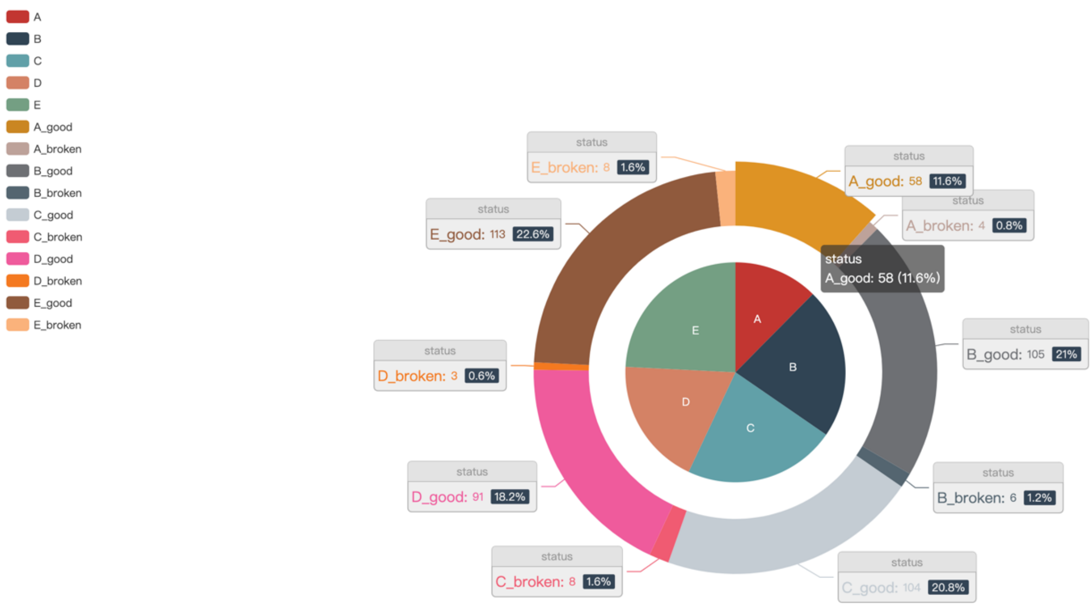


Figure B.12: Relation between areas and operational status