Examination Report – Industrial Software Engineering

M. Sc. - Information Technology

OWL University of Applied Sciences

Project Group

Juhi Soni
Maram Kamasheh
Sai Srujana Kadambari
Supriya Sindigere Kumaraswamy

I. Introduction

Human life become comfortable and easy with technological evolution. However, this technological evolution has created many problems also, and one of these problems is global warming. Today's world is facing global warming issues. The world has faced many catastrophic events because of climate change. Now it is our duty to take steps that reduce these effects and try to use solutions that are not harmful to our environment. A bike-sharing system is a small attempt to save our environment by providing an eco-friendly solution for everyday commuting. A bike-sharing system is a microservice architecture-based mobile application. It is affordable and easy to use. Any person who is in needs to use a bike and above the age of 14 can use this application. The application will provide features like login/registration, a user wallet to pay, a start trip, an end trip, park/unpark bike. Apart from these features application also provides bike issue reporting functionality and gives information such as burned calories, traveled distance, and average speed during rides after the end trip. A third-party payment gateway is integrated into the application for payment processing. For the fast retrieval of the data during the start and end trip, an application will store the customer's payment information in the application database so that every time it doesn't need to call a third-party payment API. A user cannot start a trip without a sufficient wallet balance, a user has to add the pre-defined minimum amount in the wallet. The GPS tracker is used for continuous bike location tracking. An admin portal will be used for user management, bike management, bike station management, payment management, and bike ride management.

Agile practices will be followed in project management and software development [2]. We will follow the microservice architecture pattern because microservice is easy to test, deploy and scale [1]. It reduces complexity and provides agility [1]. It also allows us to upgrade it independently in the future [1]. The test-driven development process will be followed in software development [2]. Further information about system architecture is given in section VII. The application business process is defined in section II. The use cases are described in section III and user stories in section IV. The application domain model is described in section V and the implementation processes in section VI.

II. Business Process Model

I. Business Process

The application home page will display nearby bike stations based on the user's location. In order to take a ride, a user needs to scan the bike QR code that is located on the bike lock. Alternatively, the user can also enter a bike code from a bike lock if he faces some issue with the QR code scanner. Upon receiving the start trip signal, the application checks user authentication status, wallet balance, and user status (active, inactive, black listed). If the application will encounter any issue during these checks; it will notify the user and also gives information about necessary steps which need to be taken care of from the customer side. For example, the application will redirect the user to add wallet balance screen if the user doesn't have a minimum wallet balance.   If the user passes all the checks application will unlock the bike and start a timer. The bike rental amount will be calculated based on the total time user has spent on his ride. During the end trip process, the rental amount will be deducted from the user's wallet. The application also provides a bike park/unpark feature. The user can use this feature if, during the bike riding, a user wants to stop at some place (for example, for grocery shopping), the user doesn't have to end a trip; he can use the bike park feature in this situation. Bike park feature implementation uses the same bike lock/unlock service. This service will receive input during the start/end trip and park/unpark. It does parallel processing. On the bike park action, the application will lock the bike and stop the timer. It will be free for the first 20 min, but after that application starts a timer because during parking, this bike will not be available for other customers, and an idle bike will not generate revenue for a business. After 20 min, the customer can unpark the bike and continue riding or end a trip if a station is nearby. On unpark action, the application will unlock the bike and start the timer. During the end trip process, the application checks whether a bike is at a station or not. The application will reject the end trip request if a bike is not at the correct location. The user can't place a bike at a random spot. If a bike is at the station, the application will calculate the rental amount, deduct it from the wallet, stop the timer and lock the bike. If the rental amount is more than a wallet balance, the application will deduct all money and add the remaining amount as an outstanding amount in the user's wallet. So next time a user has to pay the outstanding amount plus the required minimum amount during the start trip process.

II. Business Process Model

The business process model is described in figure.2. The business process model has three pools [2] and four lanes [2], out of three pools, the first pool indicates the user interface (mobile application), the second pool application backend, and the third pool payment gateway. The first pool describes features available in the application like start trip, bike park/unpark, end trip, etc. The second pool has a bike lock/unlock and payment management lane. It contains a number of checks and action that needs to be done during a particular user request. The third pool describes the application backend and payment gateway integration. Message flow [2], Sequence flow [2], and Association flow [2] describe the communication between applications' frontend, backend, and third-party payment gateway. Information about BPMN modeling elements [2] is added in Table.1.

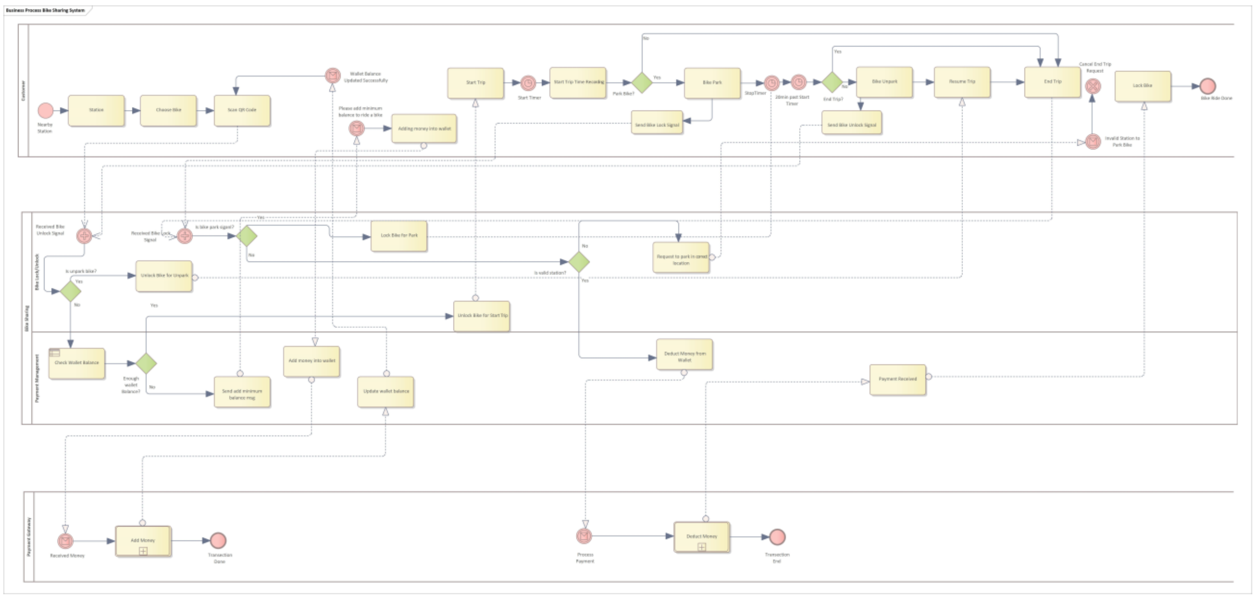


Figure.2: BPMN Diagram

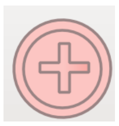
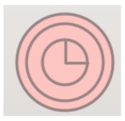
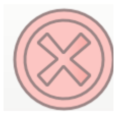
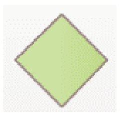
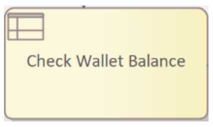
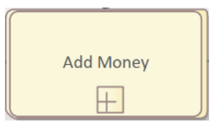
	The bike locks and unlocks event is presented as multiple parallel intermediate events. Because in bike unlock event is called when the user takes unpark and start trip action and bike lock event is called when the user takes park and end trip action.
	Timer intermediate event is used to record trip timing. The application will start the timer during the start trip and bike unpark action. It will stop the timer during the end trip and park action. The application will also start the timer after 20 min during bike park action.
	Cancel intermediate event is used to describe the rejection of an end trip request. The application won't allow a user to end a trip if a user tries to end it at a random location, not at a company's station.
	Exclusive gateway is used for conditional actions when sequence will flow to exactly one of the outgoing branch [2]. Like bike lock or unlock, end trip or park, start trip or park, etc.
	For a check wallet balance, a business rule task is used because to start a trip minimum wallet balance is needed in the user wallet. The application won't allow the user to start a trip without having a sufficient wallet balance.
	Transaction task is used to describe the payment-related operation in the payment gateway pool.

Table.1: BPMN modeling elements

III. Use Cases

I. Use Case Diagram

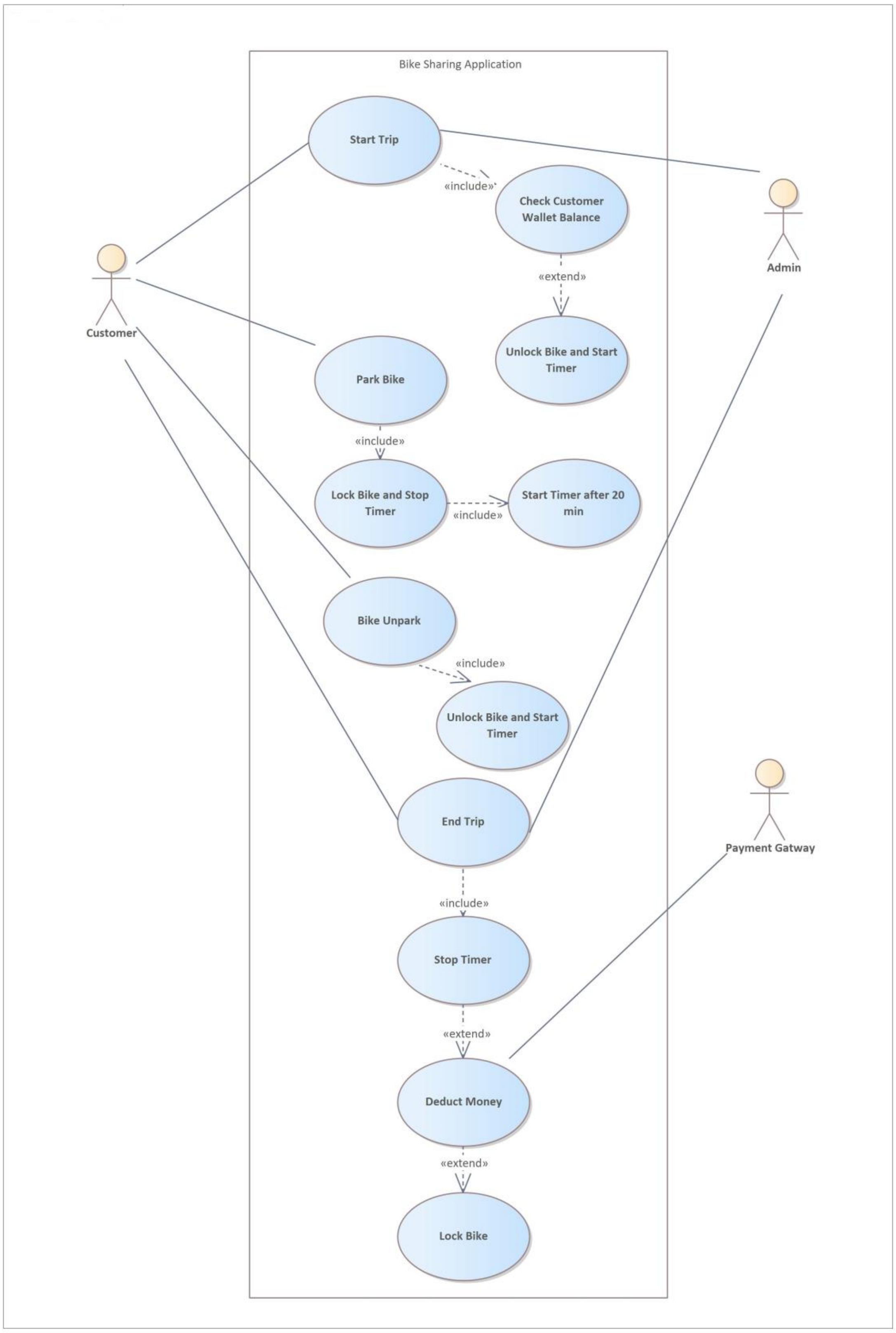


Figure.2: Use case Diagram

II. Use case – Make Trip

Name: Make a trip with a bike
Scope: NextGen Bike Sharing Application
Level: user goal
Primary Actor: Customer
Stakeholders and Interest: Customer: wants to reach the nearest bike and make a trip using it with minimal effort.

- Bike sharing system: wants to provide a service and satisfy customer interest. Wants to check user status. Wants to record the ride information and provide an accurate payment information.
- GPS system: wants to provide critical positioning information about user, bikes, and movement steps location and trac the user location during the trip.
- Tax agencies: want to collect tax from every trip. Maybe multiple agencies, such as national, state and country.
- Payment Authorization Service: wants to receive digital authorization requests in the correct format and protocol. Wants to accurately account for their customers to the application.

Preconditions: A smart phone enabled with GPS services. Location permissions. Registered user in Bike Sharing Application. Minimum wallet balance.
Success Guarantee (or Postconditions): User reach his destination. Bike is returned to an anchor point in bike station. Trip is saved. Tax is correctly calculated. Accounting and database are updated. Payment authorization approvals are recorded.
Main Success Scenario (or Basic Flow):

1. User register to the Bike Sharing application.
2. User gives location permission to the app.
3. The user start searching for the near bikes using the application.
4. The application views the bikes near to user in a map.
5. User picks a bike and scan its QR code with the application.
6. The bike sharing system get the request and check the user status and wallet.
7. The system approve that the user is in valid situation to start the trip.
8. The system view information about the bike (charge if e-bike and how kms can serve).
9. The user approves to use the bike.
10. The system send unlock signal to the bike.
11. The user starts to use the bike.
12. The system records the trip steps and status.

Steps 11 and 12 repeat until the user reach his destination.

13. The user clicks in the application that he ends his trip.
14. The system asks the user to put the bike in an anchor point in bike station and to lock it.
15. The user adds the bike in the station and lock it.
16. The system calculates the total amount of the trip based on trip timing.
17. The system presents total with taxes calculated.
18. The user approves the payment and the system handle the payment.
19. The system records the trip and update the trip, bike, and user data in the database.
20. The system sends report about the trip and payment to user by email.
21. The system asks user to rate the trip and give feedback.
22. The user enters rating and feedback.
23. The system records the data.

Extensions (or Alternative Flow):
*a. At any time, System fails:

1. System will restart automatically or by the user.
2. System will retrieve the last recorded location point and continue from it.

*b. At any time, System detect that the trip value exceeded the minimum wallet balance:
1. System will send a notification to the user about the issue.

4a. The application doesn't find bikes near me:
1. The System notify user that he can modify the constrains of the search.
2. User will respond to the issue:
   2a. The user will expand the search area.
   2b. The user will change the bike filters of the search.

7a. The user is listed in blacklist:
1. The system will reject the user request to use the bike.
2. The system will send a report to the user about his situation.
3. User will respond to the issue:
   3a. The user will fill some forms to fix the issue.
   3b. The user will visit the company to fix the issue.

7b. The minimum wallet balance is not enough:
1. The system will reject the user request to use the bike.
2. The system will send a report to the user about his situation.
3. User will respond to the issue:
   3a. The user will refill his wallet and rescan the QR code.
   3b. The user will make a request to add the trip value to the next trip.

9a. The user doesn't approve to use the bike for an issue:
1. User will reject the bike.
2. User will pick another near bike.

11a. The user finds some issues with the bike:
1. The user performs Report Bike Issues to report about the bike problem.
2. The user returns the bike to its location and lock it.
3. The user retrieves his fees.

12a. The system can't record the bike location for a while:
1. System will retrieve the last recorded location point and continue from it/ get the user location from GPS system.

15a. The user doesn't add the bike in the station and doesn't lock it and leaves it in the street:
1. The system will record an issue about this bike.
2. The system will send support team member to the bike location.
3. The system will update use data and add him to blacklist users.

Special Requirements:
- Bikes with GPS sensors and special locks.
- Bike stations with anchor points and power supply.
- Mini display to display the QR code on the bike and change it automatically.
- Smart phone to install the application.

Technology and Data Variation List:
5a. The QR code may be viewed using mini display devices or on paper that covered with plastic for protection.
5b. User can scan the QR code with the application or he can enter the number that existed under the QR code.

Frequency of Occurrence: Could be nearly continues.

Open Issues:

- What are the tax law variations?
- Must a user return the bike to a bike station and lock it every ride?
- What customization is needed for using new transportation vehicle (ex. e-scooter).
- What customization is needed to give user ability to buy cash (ex. Ticket machine).

III. Use case – Bike Park/Unpark

The application provides a bike park/unpark feature if a user wants to stop at some place, then continue his ride without ending a trip. This use case is derived from the business process model (first lane).

Name: Park or Unpark Bike

Description: A bike park feature will lock the bike and stop the timer. It is only free for 20 min; after 20 min application will start the timer because during this time bike won't be available for other customers and an idle bike is not good for business. The customer can resume his trip using the bike unpark feature. A bike unpark feature will unlock the bike and start the timer. The customer can end the trip if a customer wants to stay more than 20 min and doesn't want to pay an extra amount. An end trip will stop the timer, calculate and deduct the trip amount and lock the bike. For some reason, if a customer is unable to end a trip; he can tell the admin to end the trip on his behalf. The admin will end the trip using the admin portal.

Principal Actor: User

Precondition: A registered user with a minimum wallet balance must have started a bike ride.

Trigger: Any reason like grocery shopping, pharmacy visits, etc.

Postcondition: Lock the bike, stop a timer, and the user went for grocery shopping on the bike park action. Unlock the bike, start a timer, and the user has resumed the trip on bike unpark action.

IV. User Stories

As a tool in Agile software development user stories are used to highlight the features of the solution in the software development process [3]. Our team mainly focused on three questions for creating user stories which helped both in the development and improvements of our solution, they are:

1. Who are the main customers of our solution? (Type of users in form of "Epic story")
2. What are the main goals of the user? (These are defined in our user stories)
3. Why the user needs the feature and how can he/she do it? (Acceptance criteria)

Mainly we focused on INVEST principle [4,5] to create user stories.

- I**ndependent**: Our solution is self-contained in a way that each user story is independent and allows to be released without depending on one another. Any changes in our user story will not affect another one.
- N**egotiable**: Capturing the main essence of our user stories, Our solution will give freedom and flexibility to develop or implement in the working process.
- V**aluable**: Our solution is mainly based on user satisfaction and user needs . So, all the user stories are clearly understandable and valuable as it delivers value to end user.
- E**stimable**: All the user features which are included in our solution are measurable. This allowed us to prioritize which feature we need to work more. Our business model clearly shows Start a trip/ End trip and Bike Park/ Unpark are the most important features. So, we prioritized more value and time to them.
- S**mall**: Our user stories are simple and easily understandable and segregating into small chunks of work allowed us to complete the solution in small time.
- T**estable**: As our user story has a pre-written acceptance criteria how user can avail the main features of our solution. This helps us to test whether the project is on right track or not.

With our initial idea and use cases we need to create a solution where we can share a vehicle on time basis and charges will be applied on how much time user spend riding a bike. As our solution is mainly based on 2-wheeler eco-friendly vehicles (such as bicycle, Electronic E-bike, Battery cycle etc..), most of our users will be above the age of 14. By the above-mentioned points, we created our features and design as per the requirements and also to make the solution user friendly.

By the above questions and decisions, we came up with three user characters (personas) they are:

1. A student
2. A tourist or a visitor
3. An employee

User Story of our personas: Most of the users will travel to nearby locations in their own town or city to commute to the places like their college/university/office, grocery store, a friend's place or for sight-seeing etc. Keeping all user needs in mind we added features which will be beneficial in both times saving and economical to them. The main features are "START A

TRIP" and "BIKE PARK AND UNPARK" features. These two features will help users for searching a bike in nearby locations which will suffice their expenses. We added "GPS service" feature which can find the shortest route to reach his destination, "WALLET" feature where the user can add money to the wallet any time of his choice and travel stress free of payment delay. The features are explained with the user story and how to avail the feature is briefed below in the form of "Acceptance Criteria". All the features are how to avail those features are explained below.

Epic Story	User Stories	Acceptance Criteria
As a user I want to commute to nearby locations without using public transportations So I can save lot of time.	As a student I want to select a bike which will suit my expenses So I can travel cheaply.	Ensure the student can: • Register to the application if he is a new user • Login to the application if he is an existing user. • Access the search button. • trigger a map with the maximum 10 bikes that are 0.1 km near to me by clicking the search button. • change the thresholds to expand the search. • View brief information about the bike, its distance from the location, condition of the bike and charge per hour. • Choose a bike which is close to his location. • Click on the bike so that he can pin the bike which makes the bike invisible for others.
	As a visitor I want an easy payment method and should pay in no time so I can save time and I can avail online banking services for payment.	Ensure the visitor can: • Register to the application if he is a new user • Login to the application if he is an existing user. • User can add wallet balance by navigating to 'My wallet' and clicking add balance. • For adding wallet balance user can choose his credit card, paypal, debit card or net banking options. • By adding minimum wallet balance user can start his ride with the selected bike

		<ul><li>At the end of the ride the bike payment system will calculate the remaining amount of money user need to pay based on his ride time and parking time</li><li>If user didn't pay his remaining ride balance user cannot be able to start a new ride with his</li></ul>
	As an employee I want to end my trip after I reached my destination, So I won't get charged extra.	Ensure the student can:<ul><li>Register to the application if he is a new user</li><li>Login to the application if he is an existing user.</li><li>User can select a bike and scan the QR code to start a ride.</li><li>User can start his/her trip.</li><li>After reaching his destination user should open the bike share application and navigate to my trip to end a ride.</li><li>Application won't allow the user to end the trip if he has parked a bike at a random place.</li><li>If a user has parked a bike at the correct station application will stop the timer and lock a bike.</li><li>By calculating the time of the ride user will get charged accordingly.</li></ul>

V. Domain Model

We made our domain model based on the identified use cases, user stories and the business process that have motioned before.

Our main classes are Customer, Bike and Trip and the rest classes are crucial for the entire processes of the use cases and user stories, to start and make a trip with a bike customer need to find the suitable bike for his trip, so that we need to use bike type and bike status classes, system need to check the user status and user will scan the QR code of the bike so that we also need customer status and QR code classes, all the classes that mentioned before will help to represent the current needed information about either the customer or the bike.

Customer status class is associated to customer class with one-to-many association called has where the status can be associated to more than one customer, bike status and bike type classes are also associated to bike class with one-to-many association called has, where the status or type can be associated to more than a bike at the same time, QR code class is associated to bike class with many-to-one association called has because a bike can have more than one QR code by the time but the QR code belongs to only one bike.

When a user starts his trip with a bike a new trip will be created, the system needs to record the trip steps including the location point and the status of the bike in each step in order to record the cases of malfunctions or parking, so that we have trip class which represent the many-to-many relation between trip and customer, trip class is associated to customer class with many-to-one association called made-by where the customer can make many trips but the trip belongs to only one customer, trip class is also associated to bike class with many-to-one association called done-with where the bike can be used to make many trips but the trip is done with only one bike.

As mentioned before, trip consists of many steps which called location point, they represent the path of the trip where every point has location time and status for the bike, so that we have location point class which is associated to trip class with many-to-one association called has because trip has many location points but the location point belongs to only one trip, location point class need to be associated with bike status class with many to one association called has because the same status can be used by many points but the point should use only one status.

At the beginning and the end of each trip user should unlock and take or deliver and lock the bike on an anchor point in bike station, so that the bike can be existed in a bike station when it is needed, in order to achieve that we need to create bike station and anchor point classes which record the previous information for each bike, bike station class is associated with bike class with one-to-many association called existed-in because many bikes can be existed in one bike station but the bike at one time can be existed in only one station, anchor point class is associated with bike class with one-to-one association called existed-in because a bike can be existed at one time in only one anchor point that can’t be used at the same time by any other bike but it can be later.

Bike station and anchor point classes are associated with each other with one-to-many association called has, where the bike station consists of many anchor points and one anchor point can be existed in only one bike station.

At the end of every trip a payment process is achieved, this payment process need to be recorded in the system, so that we have bill class that record the details of the payment process and this class is associated to the trip class with one-to-one association called has because each trip as we mentioned before should have a bill and the bill information of this trip will be used only for this trip record.

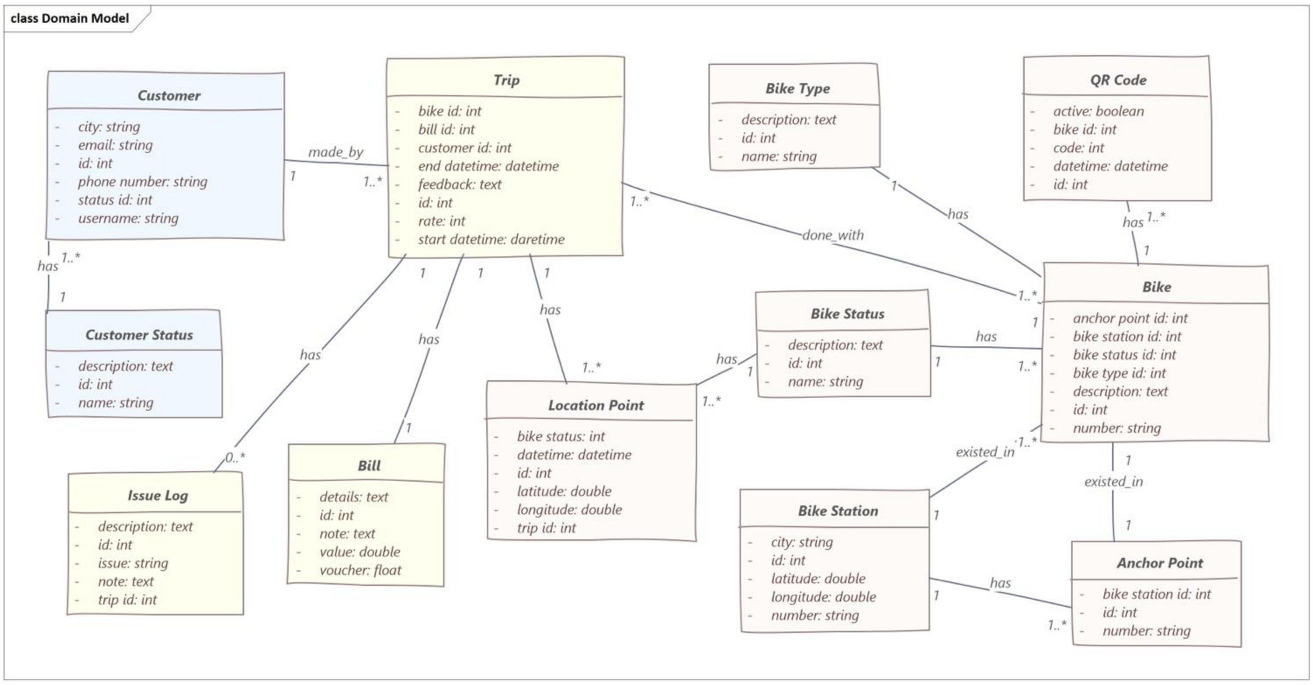


Figure.3: Domain Model V1

During the trip user may need to park the bike for some reasons and then unparck it, this process can be recorded using location point class, that record every step of the trip with the status of the bike in this step.

During the trip user can also face some issues and he can report about, so we add issue log class that is associated to trip class with many-to-one association called has with considering that the trip may not has any issues or multiple issues, and the issue will belong to only one trip.

The domain model was created using Enterprise Architect (EA) tool [10], and after making this version of domain model we thought about some modification and enhancements, so we add extra classes which are issue class, city class and voucher class.

These extra classes only organize the model or help to add extra features to it, the voucher class helps to add voucher feature which allow users to have discounts according to specific criteria, voucher is applied in the payment process so the voucher class will be associated to bill class with one-to-many association called uses, with considering that the bill can use no voucher or multiple vouchers.

Issues class is using to organize the repeated and usual issues that the user could face during his trip, this class is associated to issue log class with one-to-many association called has, where the user can report about one or more issue during his trip.

City class is used to organize the structure, it is associated with both customer class and bike station class with one-to-many association, the customer-city association called lives-in, and the bike station-city association called existed in, so many users and many stations can be existed in the same city.

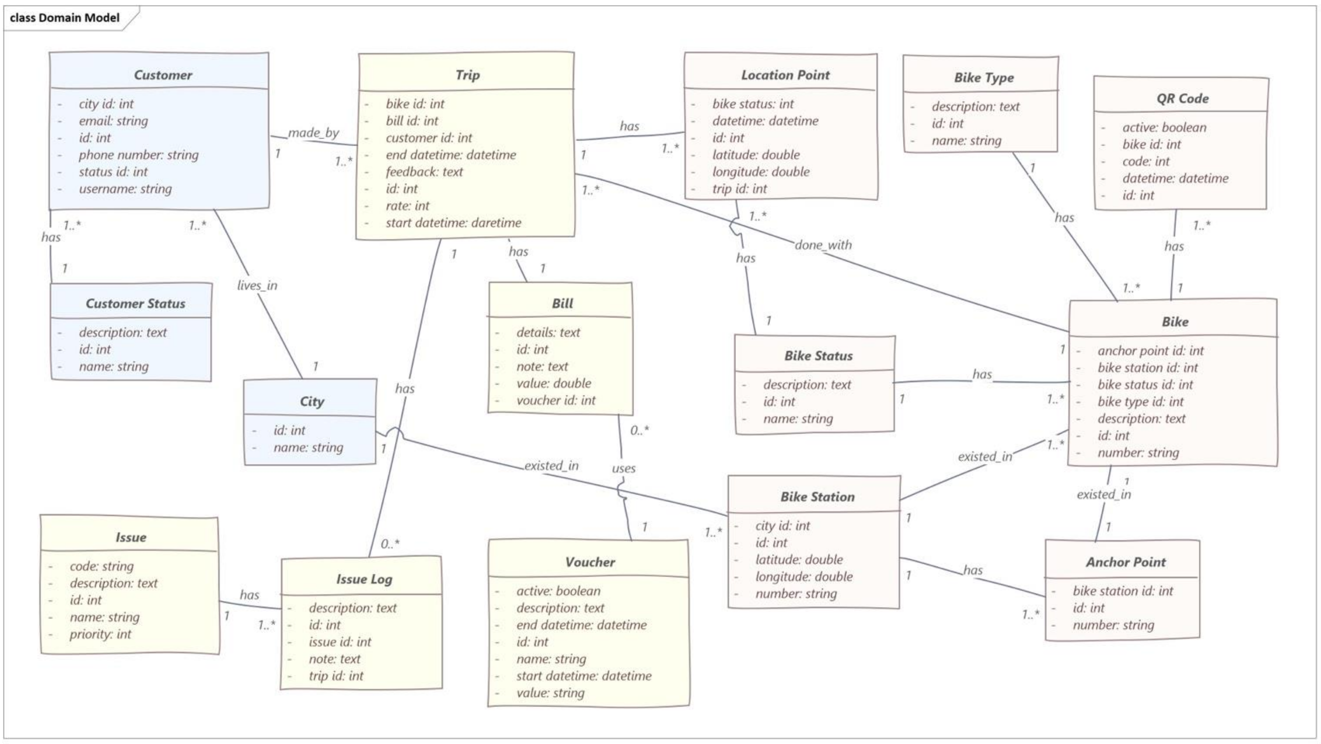


Figure.4: Domain Model V2

VI. Implementation processes

Every business has a process, which is a series of actions and duties bound by predetermined guidelines and intended to help the business achieve its objective. Business processes are another name for these procedures. Each of these processes has a start and a finish, one or more triggers, and produces a certain outcome. The number of steps in the process and the number of systems involved determine how complicated the processes are [11]. Various automation methods, including Business Process Automation (BPA), Robotic Process Automation (RPA), and Custom Automation, can be used to automate all these operations. The speed, precision, and efficiency of corporate processes are all improved by these automated processes. They also cut down on time spent on monotonous tasks [12].

A block of code that executes a particular operation in response to program rules is known as robotic process automation (RPA). These lines of code mimic the work that a person can do. Before using RPA, a subject matter expert (SME) must document the task that needs to be automated. By turning the task's steps into lines of code, one particular task can be mechanized. The code contains all the guidelines and details needed for a bot to carry out the action [12]. These procedures are algorithm-driven, business logic-programmed, and data-supplied [13]. RPA can be utilized for straightforward, simple activities that follow rules, as long as the data is structured and consistent. For instance, call centres in banking, farming robots in agriculture, and robotic assistants in healthcare [12] are some examples.

Business process automation (BPA) is the use of cutting-edge technologies to automate intricate business procedures and tasks that go beyond standard record-keeping and data manipulation. BPA can automate repetitive activities, data gathering and management, linking and integrating data sources and services, and app generation. The BPA gathers structured and de-identified data, as well as unstructured data from social media and customer behaviour patterns, to carry out all the functions [11].

BPA software offers features that enhance overall business process management and can increase task and end-to-end process speed and accuracy. BPA software comes with tools to facilitate analytics and collaboration, process modelling, and orchestration [12].

BPA software can be used to accomplish challenging tasks like streamlining processes, enhancing user experiences, and lowering error rates. Additionally, it can be utilized to streamline procedures, automate routinely modified operations or workflows, and streamline intricate procedures like hiring and onboarding new employees. For commercial customers, BPA also offered low-code automation [12].

By updating and modernizing manual tasks, custom automation is a technique used to partially or automate the manufacturing process. According to the requirements of the organization, automation solutions are relevant [14]. Custom automation offers three key advantages: safety, cost savings, and speed. It enables producers to concentrate more on quality, enhances process accuracy and optimization, and increases production volume adaptability and efficiency. Additionally, it is economical [15].

The best strategy for our industry is business process automation (BPA), as our business process is comprised of software that executes several intricate, repetitive operations. Additionally, this involves activities like enhancing user experiences and lowering error rates. These procedures serve as auxiliary procedures that aid in the smooth operation of the system but are not accountable for the result. Additionally, the model makes use of outside programs like GPS and maps. Our domain tends to be intricate, interconnected with numerous corporate IT systems, and specially adapted to an organization's requirements. RPA is not appropriate for our area because it is typically used for straightforward iterative tasks and most industries use custom automation.

VII. System Architecture

To design our system architecture, first we needed to define the main components of it, which are the bike with its sensors and lock, the smart mobile, the database, the API Gateway, the backend micro services which can be considered as management system, and the GPS-system.

Bike sensors and lock, smart mobile and database are the components that produce data that needed to be used as input to the backend micro services component, so we called them data sources.

The API Gateway is needed to handle common tasks that are used across a system of API services, such as user authentication, rate limiting, and statistics.

The backend micro services need to get data from the other components in order to perform its processes and functions and return suitable information to its users, with considering that the GPS-system component is needed as an API to help with some feature and functions like generating maps and detecting locations, the system architecture is illustrated in figure.5.

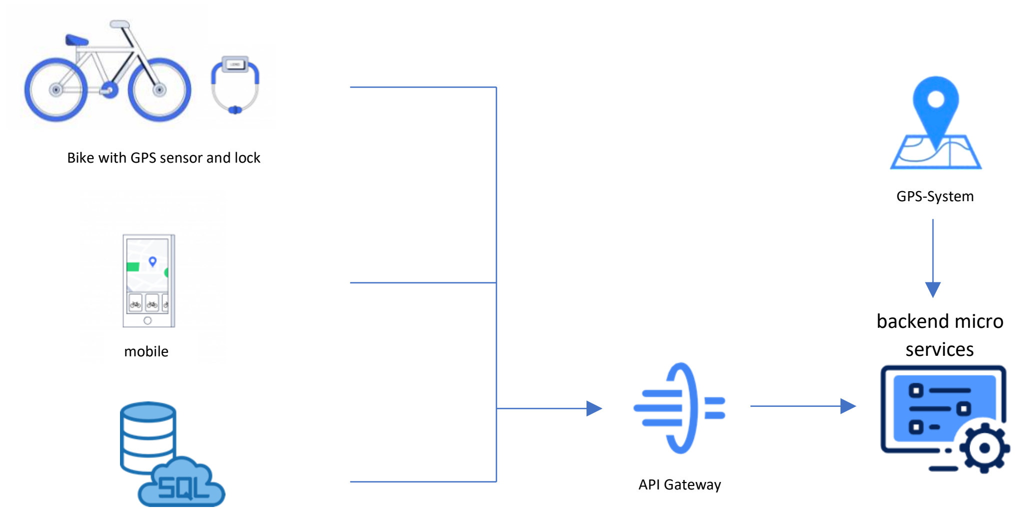


Figure.5: System architecture of both software and hardware main components

After creating the system architecture, we tried to apply the appropriate IoT patterns to it, so we started with the data sources components.

Bike sensors and lock have to be constantly active and connected for the user so we add the pattern Always-On Device [8] to it, they also require fair amount of power which can be provided by rechargeable or replaceable source of energy like batteries, so that we use the pattern Period Energy-Limited Device [6].

The smart mobile also requires fair amount of power which can be provided by rechargeable or replaceable source of energy like batteries, so that we use the pattern Period Energy-Limited Device.

In order to transfer the data from the data sources to the backend micro services we need to connect the data sources components to a network, the smart mobile can be connected to the internet while other sources, for example databases, might be accessed via SQL, so that we will use the Device Gateway pattern [6], this will make the system able to translate different communication technologies so that they can be uniformly accessed by the backend micro services (management system) Component.

Although we have Always-On Devices, there may be situations where they are unavailable, for example during the absence of the Internet for some reasons (such entering a tunnel or the absence of network towers). In this case, a Device Shadow [6] helps as it stores the last known states and desired future states of all devices connected to it.

Thus, it is possible for other components, such backend micro services component to retrieve the last situation of the device and continue from that point.

As can be seen in the middle of figure.6, we added this functionality to the section before the backend micro services and the API gateway with the Relay Service.

Now we turn to the backend micro services component which achieve the processing of the data and contains the main functions and algorithms, in this component a lot of messages and data is going to be received, process and send, so that we add the pattern Rules Engine [6] to this component.

For our bike sharing system we use Remote Processing pattern [9], where the analytics software is hosted remotely in the cloud and the data produced by the devices and other data sources is transferred into the cloud for analysis. To support this scenario, the

Relay service, the API Gateway, and the backend micro services component are packaged as a Smart Service. This Smart Service is then provisioned by the Smart Sharing System Provisioning Engine into a remote environment, as shown in figure.6.

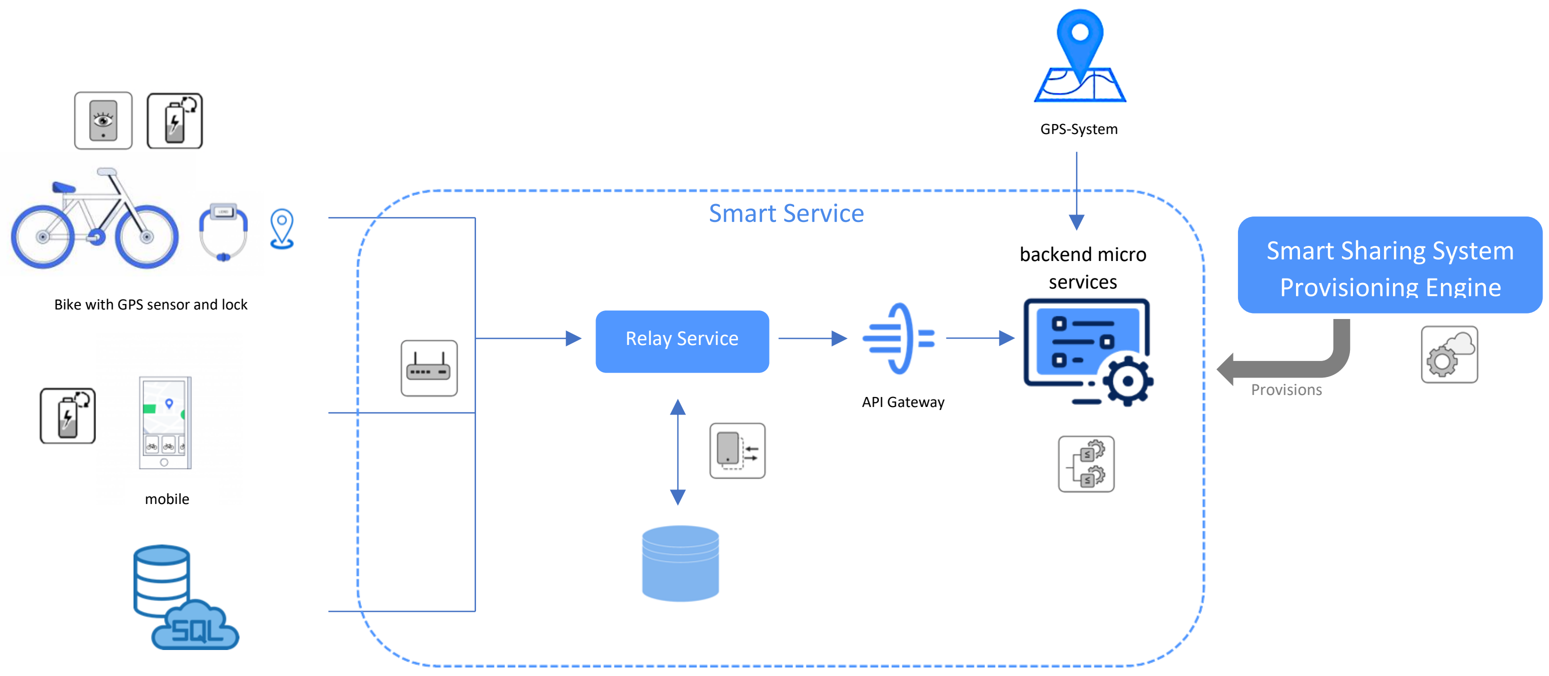


Figure 6 System architecture of both software and hardware components with IoT patterns

VIII. References

[1] Villamizar, Mario, et al. "Evaluating the monolithic and the microservice architecture pattern to deploy web applications in the cloud." 2015 10th Computing Colombian Conference (10CCC). IEEE, 2015.

[2] Prof. Dr. Jessica Rubart & Prof. Dr. Robert Mertens “Industrial Software Engineering Slides” 03.2020

[3] Agile development process: https://https://kissflow.com/project/agile/creating-agile-user-stories/

[4] What is User Story? https://www.visual-paradigm.com/guide/agile-software-development/what-is-user-story/

[5] Kulkarni, A. (2020, October 15): https://www.linkedin.com/pulse/my-understanding-invest-principle-user-stories-amit-kulkarni/

[6] Reinfurt, L., Breitenb¨ucher, U., Falkenthal, M., Leymann, F., Riegg, A.: Internet of things patterns. In: Proceedings of the 21st European Conference on Pattern Languages of Programs (EuroPLoP). ACM (2016)

[7] Reinfurt, L., Breitenb¨ucher, U., Falkenthal, M., Leymann, F., Riegg, A.: Internet of things patterns for communication and management. LNCS Transactions on Pattern Languages of Programming (2017)

[8] Reinfurt, L., Breitenb¨ucher, U., Falkenthal, M., Leymann, F., Riegg, A.: Internet of things patterns for devices. In: Proceedings of the Ninth International Conferences on Pervasive Patterns and Applications (PATTERNS) 2017. pp. 117–126. Xpert Publishing Services (2017)

[9] Reinfurt, Lukas, Michael Falkenthal, Uwe Breitenbücher and Frank Leymann. “de Applying IoT Patterns to Smart Factory Systems.” (2017).

[10].https://sparxsystems.com/enterprise_architect_user_guide/15.2/model_domains/create_a_domain_model.html

[11] “Business Process Automation (BPA): All You Need to Know!” *Master Process Automation Internally with Bots & People*, https://www.botsandpeople.com/blog/business-process-automation-bpa-all-you-need-to-know.

[12] Babb, Benjamin. “RPA vs. BPA: Which Automation Tool Do You Need?” *Pipefy*, 18 Feb. 2022, www.pipefy.com/blog/rpa-vs-bpa/. Accessed 8 Sept. 2022.

[13] “RPA and BPA - Definition, Core Differences, and Similarities.” *Skelia*, 4 Nov. 2021, skelia.com/articles/rpa-and-bpa/. Accessed 8 Sept. 2022.

[14] Patel, Sandeep. “What Is Custom Automation?” *LinkedIn*, LinkedIn, 9 Mar. 2018, https://www.linkedin.com/pulse/what-custom-automation-sandeep-patel/.

[15] Patel, Sandeep. “What Is Custom Automation.” *Nexus Automation*, Nexus Automation, 9 Apr. 2020, https://nexusautomation.com/articles/what-is-custom-automation.