Software Requirements Specification

<Bike-Sharing-System>

Date of Submission: 02.02.2025

Group Members		
	Name	ID
1.	**MD. SAKIF ULLAH RAFI**	**20-43945-2**
2.	**SAGOR SAHA**	**21-45074-2**
3.	**MD. TANVIR SAJIB**	**22-46485-1**
4.		

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

- This document specifies the software requirements for a Bike Sharing System. It focuses on the initial version (Version 1.0) of the system that will be implemented as a pilot project in Dhaka, Bangladesh.

- The document covers the entire Bike Sharing System, including the user interface, backend database, and mobile app for users to rent and return bikes. It also includes the management system for operators to monitor and maintain the bikes. This SRS focuses on the digital aspect of the system, not on physical infrastructure like bike lanes.

- The Bike Sharing System is an affordable and eco-friendly solution to promote sustainable transportation in Dhaka. It aims to provide an alternative to cars and motorbikes, reducing traffic congestion and pollution. The system will benefit tourists, students, and city dwellers by offering a convenient way to move around the city. This aligns with broader goals like supporting student mobility, promoting tourism, and fostering an environmentally sustainable transportation network.

- This system supports corporate goals like promoting green initiatives by reducing carbon emissions, enhancing urban mobility and making transportation affordable for all which encourages a healthier lifestyle by promoting cycling.

- Business Requirements of the product:
  - The system will offer low-cost bike rentals for students, tourists, and daily commuters.
  - Encourages the use of non-motorized transport to reduce environmental impact.
  - Provides tourists with a convenient way to explore the city.
  - Implement features like GPS tracking and smart locks to prevent bike theft.
  - Offers a mobile app with features like booking, payment, and route recommendations.

1.2 Document Conventions

- Standards or typographical conventions that were followed:

Formatting and Style:
- Standard font: Times New Roman, size 12 for the main text.
- Headings: Bold, larger font size (size 14 for main headings).

Highlighting:
- Critical points or warnings are written in bold or highlighted.
- Diagrams and tables are used where needed for clarity.

Prioritization:
- Each specific requirement is labeled with a priority (High, Medium, Low) to indicate its importance.

1.3 Intended Audience and Reading Suggestions

- This document is designed for various stakeholders of the Bike Sharing System. Developers will use it to understand technical requirements for building the app, backend database, and docking station integration. Project Managers will rely on it to oversee timelines, allocate resources, and ensure alignment with business goals. Testers will reference it to identify features for testing and validate system quality. Marketing Staff will use the document to learn key features for promoting the system, while Users and Documentation Writers will refer to it to understand the system's functionality and prepare user manuals or guides.

- The document should be read in a sequence suited to each audience. Everyone should begin with the Introduction section (1.1–1.4) for an overview of the system. Developers and Testers should then focus on the System Requirements (3.1) and Design and Interface Requirements (4.1–4.3) for technical details. Project Managers should prioritize the Overall Description (2.1–2.6) and Project Requirements (3.3) to plan and manage the project. Marketing Staff can benefit from the Product Perspective and Business Objectives in the Overall Description (2.1). Finally, Users and Documentation Writers should review the System Features and UI/UX Design Specification to understand functionality and user experience.

1.4 References

- Draw.io (website) was used to make diagrams, Balsamiq was used for UI

2. Overall Description

2.1 Product Perspective

- Dhaka, Bangladesh, faces severe traffic congestion, air pollution, and a lack of affordable, eco-friendly transportation options. As there are no existing bike-sharing systems or dedicated cycle lanes, there is a need for a solution that promotes sustainable and cost-effective mobility for students, tourists, and daily commuters.

- The Bike Sharing System is a new, independent product, designed from scratch to address these issues. It is not part of any existing system or product family. The system integrates multiple components which includes a mobile app for users, a backend database for managing

operations, and docking stations for bike storage and security. These components work together to provide a seamless experience, ensuring safe and efficient transportation.

- The business objectives of the Bike Sharing System are to provide an affordable and eco-friendly transportation option for students, tourists, and commuters in Dhaka. The system aims to reduce traffic congestion and air pollution while promoting a sustainable and healthy lifestyle through cycling. It also supports tourism by offering an easy way for visitors to explore the city and ensures security with features like GPS tracking, smart locks, and geofencing to create a reliable and user-friendly service.

2.2 Product Functions

- **Major Functions Summary:**

  - **User Registration and Login**: Allows users to create accounts, log in, and manage their profiles.

  - **Bike Rental and Return**: Users can rent bikes by selecting them from the app, and mark bikes as returned in designated virtual docking areas (geofenced locations).

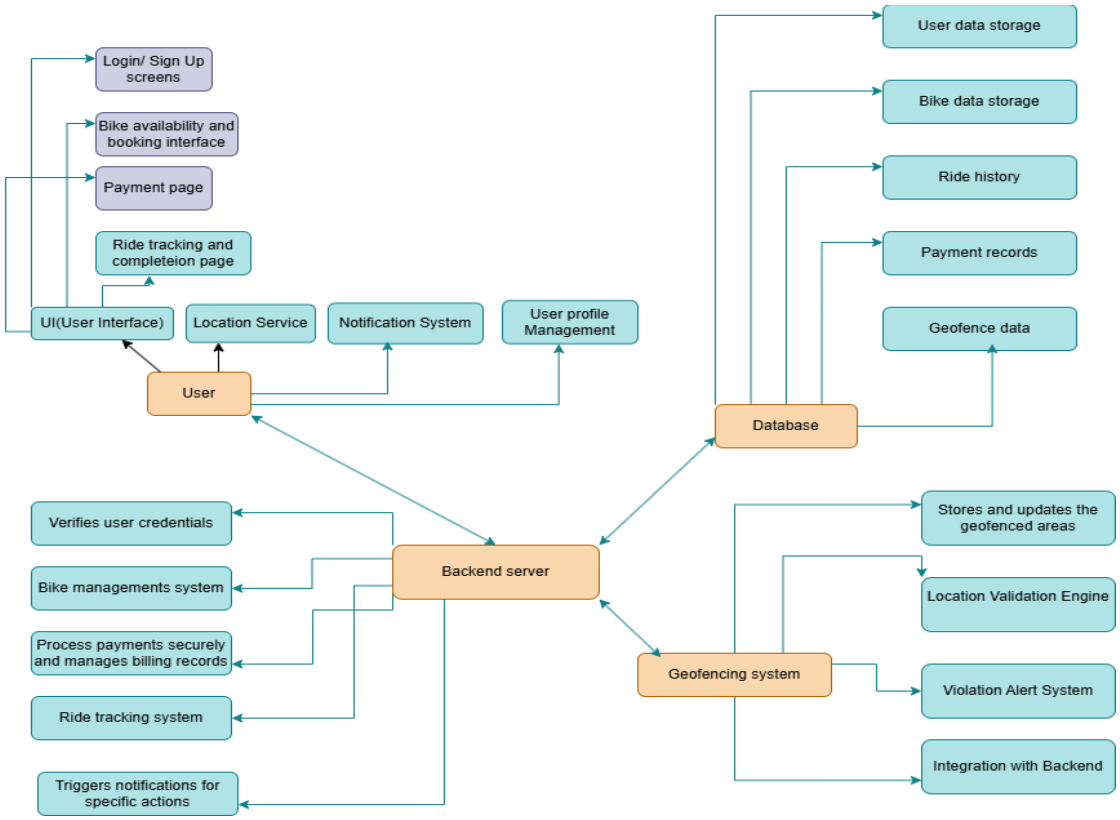
  - **Real-Time Bike Availability**: Displays the number of available bikes and free return slots at nearby docking locations.

  - **Payment and Billing:** Facilitates secure payment processing for bike rentals through mobile wallets or cards.

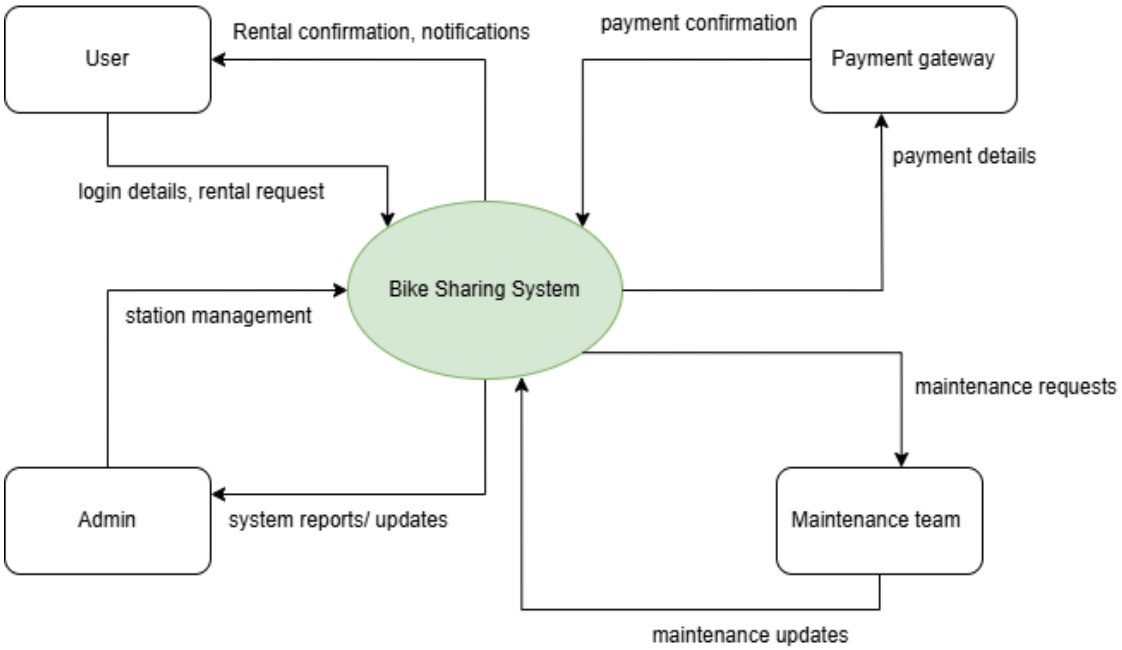
  - **Ride Tracking:** Tracks the user's bike ride, including duration, distance, and route taken.

  - **Notifications and Alerts**: Sends alerts for incomplete returns, out-of-bound rides, or updates.

- **System architecture diagram:**



- **Top level data flow diagram:**



2.3 User Classes and Characteristics

- Identifying various user classes:

a) Casual Users:
- Individuals who use the bike-sharing system occasionally, like tourists or first-time users.
- These users are less frequent and typically use the system for short trips.

b) Regular Users:
- Individuals who use the system frequently, such as daily commuters or students.
- They are likely to have a subscription or membership.

c) System Administrators (Admins):
- Staff responsible for managing the system, including bikes, stations, and user data.
- They have higher privileges and access to backend functionalities.

d) Bike Maintenance Team:
- Technicians responsible for maintaining and repairing bikes.
- They interact with the system to check maintenance requests and update repair statuses.

e) Payment Gateway (Third-Party Entity):
- External service handling payment transactions for rentals.
- It plays an essential role in the workflow.

- **Pertinent Characteristics of Each User Class:**

a) Casual Users:

- Frequency of Use: Low, occasional usage.
- Functions Used: Register/login, rent bikes, return bikes, and make payments.
- Technical Expertise: Basic, the system must be user-friendly and intuitive.
- Security/Privilege Levels: Minimal, limited to their own account activities.
- Educational Level: No specific requirement, designed for the general public.

b) Regular Users:

- Frequency of Use: High, frequent daily or weekly usage.
- Functions Used: May use advanced features like subscription management.
- Technical Expertise: Moderate, familiar with the system over time.
- Security/Privilege Levels: Standard, limited to their own account activities.
- Educational Level: No specific requirement, designed for regular commuters and student.

c) System Administrators:

- Frequency of Use: High, use the system daily for management tasks.
- Functions Used: Add/remove bikes, manage stations, monitor user activities, and view reports.
- Technical Expertise: High, requires familiarity with backend systems and tools.
- Security/Privilege Levels: High, full access to system management and data.
- Educational Level: Trained personnel with technical knowledge.

d) Bike Maintenance Team:

- Frequency of Use: Medium; interact with the system as needed for maintenance tasks.
- Functions Used: View maintenance requests, update bike statuses, and schedule repairs.
- Technical Expertise: Moderate; basic knowledge of system operations.
- Security/Privilege Levels: Medium; limited to maintenance-related data.
- Educational Level: Technicians with practical expertise.

e) Payment Gateway:

- Frequency of Use: Continuous; processes payments for all transactions.
- Functions Used: Processes payment details and provides payment confirmations.
- Technical Expertise: Fully automated; no direct human interaction.
- Security/Privilege Levels: High; handles sensitive payment data.

- **Most Important User Classes:**

The most important user classes for the bike-sharing system are Regular Users and System Administrators. Regular users are the primary drivers of the system, providing consistent usage and revenue. Their frequent interactions with the system make their satisfaction critical for long-term success. They rely on the system for daily commutes, requiring seamless functionality, reliable bike availability, and a user-friendly interface. System administrators are equally important as they ensure the system operates smoothly. They manage bike inventories, monitor station activities, handle user accounts, and address technical or operational issues, directly influencing the quality of service and user satisfaction.

Less Important User Classes:

The less important user classes include Casual Users, Bike Maintenance Team, and the Payment Gateway. Casual users interact with the system occasionally, contributing to its revenue but not as significantly as regular users. While their experience is important, they are not as critical to the system's long-term success. The bike maintenance team plays a crucial operational role by ensuring bikes are functional and stations are well-maintained, but their work is primarily behind the scenes and does not directly impact the end-user experience. The payment gateway, although essential for processing transactions, functions as an automated external service rather than an active user, making it less significant in terms of user interaction.

2.4 Hardware and Operating Environment

- **Hardware Platform:**

  - The system will use smartphones (Android and iOS) for user interactions, as the primary interface is a mobile application.
  - Docking stations will be equipped with hardware such as touchscreens, card readers, QR code scanners, and locking mechanisms for bikes.
  - The backend system will run on cloud servers or dedicated servers with sufficient storage and processing power to handle user data, transactions, and real-time operations.

Operating System and Versions:

- The mobile application will support Android (version 8.0 and above) and iOS (version 13 and above) to cover a wide range of devices.
- Docking stations will run on lightweight operating systems such as Linux or for performance and security.
- Backend servers will operate in a reliable and scalable environment, such as Linux or Windows.

Other Software Components:

- The system will integrate with payment gateways (Bkash, Nogod) for secure transaction processing.

- It will require a database management system (MySQL) for storing user, bike, and transaction data.

- APIs will facilitate communication between the mobile app, docking stations, and backend system.
- Security software like SSL protocols will be implemented for data encryption during user interactions and transactions.

2.5 Design and Implementation Constraints

- The system must comply with local transportation regulations, such as those for shared vehicles and public safety. Any environmental policies regarding the use of energy-efficient equipment at docking stations may also apply.
- Docking stations may have limited processing power, requiring lightweight software. Memory and storage in embedded systems must be optimized to ensure smooth operation without frequent upgrades. Battery-operated components in bikes, such as GPS trackers, will have power constraints, necessitating energy-efficient design.
- The system must integrate seamlessly with payment gateways (Bkash, Nogod) for secure transactions. Compatibility with third-party mapping services (Google Maps) for navigation and station location must be ensured.
- The project may be constrained to use specific technologies like MySQL for databases and Java (Android) and Swift (iOS) for mobile application development.
- The system must support simultaneous operations, such as multiple users renting bikes or returning them at the same time, without performance issues.
- The system must implement strong user authentication methods, such as passwords, OTPs, or biometric options.

2.6 User Documentation

- The system will provide a User Manual for regular and casual users, explaining how to register, rent bikes, return them, and make payments. This manual will include step-by-step instructions with screenshots and visuals for better understanding. For system administrators and the maintenance team, a Technical Guide will be delivered, covering system management, bike maintenance, and troubleshooting common issues. Additionally, an Online Help System will be integrated into the mobile application and kiosk interface, offering context-sensitive assistance for tasks like resolving payment issues or locating nearby stations. Frequently Asked Questions (FAQs) will also be included in the app and website, addressing common queries and issues.
- The user manual and technical guide will be shared as PDF files so that users can easily download, view offline, or print them. The online help system will be available on the web, making it work smoothly on all types of devices like phones, tablets, and computers. All the documents and guides will be written in simple, clear language with pictures and easy steps to make them helpful and easy to understand for everyone.

3. System Requirements

3.1 System Features

- **User Registration and Account Management:**
  Provides user accounts with features like sign-up, login, profile management, and ride history.
  Priority: High
  Benefit: 8 (enhances user engagement and experience)
  Penalty: 7 (limited functionality without user accounts)
  Cost: 5 (standard feature with moderate implementation cost)
  Risk: 4 (minor risk in ensuring data security)

- **Bike Rental and Return System:**
  Allow users to rent and return bikes via the mobile app or kiosks at docking stations. This includes real-time bike availability and reservation options.
  Priority: High
  Benefit: 9 (essential for core functionality)
  Penalty: 9 (system fails without it)
  Cost: 7 (requires backend support and user interface development)
  Risk: 6 (risks with bike tracking and availability issues)

- **Real-Time Bike Tracking and GPS Navigation:**
  Tracks the location of bikes in real-time and provides GPS navigation to nearby docking stations.
  Priority: Medium
  Benefit: 7 (improves user convenience)
  Penalty: 6 (affects user experience without it)
  Cost: 7 (higher cost due to IoT device integration)
  Risk: 6 (reliance on network and GPS accuracy)

- **Map Integration:**
  Provides an interactive map within the mobile app to help users locate nearby docking stations, check bike availability, and navigate to their destinations.
  Priority: High
  Benefit: 8 (improves user convenience and accessibility).
  Penalty: 7 (users may face difficulties without map functionality).
  Cost: 6 (requires integration with mapping APIs like Google Maps).
  Risk: 5 (risks include dependency on real-time data accuracy).

- **Payment Integration:**
  Enables secure payment through multiple options, including credit/debit cards, mobile banking, and online payment gateways.
  Priority: High
  Benefit: 8 (critical for revenue generation)
  Penalty: 8 (users will leave without smooth payments)
  Cost: 6 (moderate cost to integrate payment APIs)
  Risk: 5 (security concerns with transactions)

- **Maintenance and System Alerts:**
  Send alerts for maintenance needs, such as low battery bikes or damaged components, to the maintenance team.
  Priority: Medium
  Benefit: 6 (ensures system reliability)
  Penalty: 5 (reduces operational efficiency without it)
  Cost: 5 (moderate cost for backend monitoring)
  Risk: 4 (low risk as alerts are automated)

- **Privacy and Security:**
  Implement security measures to protect user data, including personal information, payment details, and ride history. Features include data encryption, two-factor authentication.
  Priority: High
  Benefit: 9 (essential for user trust and system credibility).
  Penalty: 9 (loss of user trust and potential legal consequences without it).
  Cost: 7 (requires investment in encryption technologies and compliance measures).
  Risk: 6 (risks include potential breaches and data leaks).

- **QR Code Scanning:**
  Enables users to unlock bikes by scanning a QR code placed on the bike using the mobile app.
  Priority: High
  Benefit: 8 (smooth user experience and reduces friction in the rental process).
  Penalty: 7 (users may face delays or confusion without this feature).
  Cost: 6 (requires integration of QR code generation and scanning functionality).
  Risk: 4 (low risk).

- **Premium Subscription:**
  Offers exclusive benefits to users who subscribe to a premium plan, such as unlimited rides, priority access to bikes during peak hours, discounted rates.
  Priority: Medium

Benefit: 7 (increases revenue).
Penalty: 5 (May reduce user satisfaction for non-premium users).
Cost: 5 (requires additional development to manage subscription features).
Risk: 4 (low risk but may require effort to balance premium and free user satisfaction).

- **Ride History:**
  Allow users to view a detailed log of all their previous rides. Users can access and review their ride history through their account in the mobile app.
  Priority: Medium
  Benefit: 7 (enhances user experience by providing a record of usage).
  Penalty: 5 (users may feel disconnected without the ability to track their rides).
  Cost: 5 (requires database storage and backend support).
  Risk: 4 (low risk, but potential issues with data syncing or accuracy).

- **Settings:**
  Provides users with the ability to customize their account such as notification settings, payment methods, language preferences, and privacy options. Users can manage their subscription plans and change their personal details, such as email or password.
  Priority: Medium
  Benefit: 6 (improves user experience).
  Penalty: 4 (users may feel limited if they cannot personalize their experience).
  Cost: 5 (requires development for user account management and settings interface).
  Risk: 3 (low risk).

- **Loyalty and Reward Programs:**
  Offers discounts or points for frequent users to encourage regular usage.
  Priority: Low
  Benefit: 5 (boosts user engagement)
  Penalty: 4 (does not affect core functionality)
  Cost: 4 (requires minor development effort)
  Risk: 3 (low risk due to optional nature)

- **Help and FAQs:**
  Offers a dedicated help section and a list of frequently asked questions (FAQs) within the app.
  Priority: Medium
  Benefit: 7 (enhances user satisfaction by providing quick support).
  Penalty: 5 (users may experience confusion without it).
  Cost: 4 (requires content creation and basic integration).
  Risk: 3 (low risk as it's primarily informational).

3.1.1 Software Login

Functional Requirements (FRs)

1.1 The software shall allow users to log in using their registered email address and password.

1.2 The login credentials (email and password) will be validated against records stored in the system's database.

1.3 Upon successful login, the user's personalized dashboard will be displayed, showing options like renting a bike, viewing ride history, and account settings.

1.4 If the entered email and/or password are incorrect, the system will display an error message and prompt the user to retry.

1.5 If the user exceeds the login attempt limit (e.g., 3 failed attempts), the system shall temporarily lock the account for 15 minutes to enhance security.

1.6 The system shall provide a "Forgot Password" option, allowing users to reset their password by verifying their identity through a verification code sent to their registered email.

Priority Level: High
Precondition: The user must be registered with the system and possess valid login credentials.
Cross-references: QA3, QA5 (example)

3.1.2 Bike Rental and Return System

Functional Requirements (FRs)

1.1 The system shall allow users to view available bikes at nearby docking stations.

1.2 The system shall allow users to rent bikes by scanning a QR code or selecting the bike through the mobile app.

1.3 The system shall record the bike rental time and user ID for tracking purposes.

1.4 The system shall provide a confirmation notification to the user upon successful bike rental.

1.5 The system shall allow users to return bikes to available docking stations and update the bike's availability status.

1.6 The system shall update the user's ride history and calculate the rental cost at the time of return.

Priority Level: High
Precondition: User must be logged in, and bikes must be available at the docking stations.
Cross-references: QA3, QA5 (example)

3.1.3 Payment Integration

Functional Requirements (FRs)

1.1 The system shall allow users to make payments through multiple methods, including credit/debit cards, mobile wallets, and online payment gateways.

1.2 The system shall calculate the rental cost based on duration and distance of the ride.

1.3 The system shall process payments securely and store transaction records.

1.4 The system shall send a payment receipt to the user's email after each successful transaction.

Priority Level: High
Precondition: User must have valid payment details and an active internet connection.
Cross-references: QA3, QA5 (example)

3.1.4 User Registration and Account Management

Functional Requirements (FRs)

1.1 The system shall allow users to register for an account using their email address or phone number.

1.2 The system shall validate the registration information to ensure uniqueness (e.g., no duplicate email addresses).

1.3 The system shall allow users to log in using their registered username and password.

1.4 The system shall allow users to update personal details, including name, email, phone number, and profile picture.

1.5 The system shall allow users to change their password or deactivate their account.

Priority Level: High
Precondition: User must provide valid registration information.
Cross-references: QA3, QA5 (example)

3.1.5 Real-Time Bike Tracking and GPS Navigation

Functional Requirements (FRs)

1.1 The system shall provide real-time location updates for available bikes and docking stations using GPS.

1.2 The system shall display bike availability on a map within the mobile app.

1.3 The system shall allow users to view nearby docking stations and navigate to them using integrated maps.

1.4 The system shall send notifications to the user if their rented bike is approaching its return deadline.

1.5 The system shall update the bike's location in real-time as it is in use.

Priority Level: Medium
Precondition: The bike must be equipped with GPS tracking, and the user must have access to the mobile app and an active internet connection.
Cross-references: QA3, QA5 (example)

3.1.6 Maintenance and System Alerts

Functional Requirements (FRs)

1.1 The system shall monitor bike status for low battery, damage, or required maintenance.

1.2 The system shall send automated alerts to the maintenance team if a bike requires attention.

1.3 The system shall notify the user if a bike is unavailable due to maintenance or other issues.

1.4 The system shall allow the maintenance team to mark a bike as repaired and return it to service.

Priority Level: Medium
Precondition: The bike must be equipped with sensors for maintenance monitoring.
Cross-references: QA3, QA5 (example)

3.1.7 Map Integration

Functional Requirements (FRs)

1.1 The system shall integrate with a map service (e.g., Google Maps) to display available docking stations and bike locations.

1.2 The system shall allow users to zoom in/out and interact with the map to find nearby bikes.

1.3 The system shall provide directions to the selected docking station.

1.4 The system shall update the map in real-time to reflect bike availability and station status.

Priority Level: High
Precondition: User must have internet access and the necessary permissions for location services.
Cross-references: QA3, QA5 (example)

3.1.8 Premium Subscription

Functional Requirements (FRs)

1.1 The system shall allow users to subscribe to a premium membership for additional benefits.

1.2 The system shall provide premium users with features such as unlimited rides, priority access during peak times, and discounted rates.

1.3 The system shall manage premium subscription details, including payment, validity, and renewal.

Priority Level: Medium
Precondition: User must have a valid payment method and an active internet connection.
Cross-references: QA3, QA5 (example)

3.1.9 Privacy and Security

Functional Requirements (FRs)

1.1 The system shall encrypt user data (personal details, payment information) during transmission and storage.

1.2 The system shall provide two-factor authentication for user accounts for added security.

1.3 The system shall comply with data protection regulations such as GDPR and ensure user privacy.

1.4 The system shall implement user access control, allowing different user roles (e.g., admin, user, maintenance) with different permissions.

1.5 The system shall regularly audit security logs to detect and prevent unauthorized access.

Priority Level: Medium
Precondition: User must have an active account and valid credentials.
Cross-references: QA3, QA5 (example)

3.1.10 QR Code Scanning

Functional Requirements (FRs)

1.1 The system shall generate a unique QR code for each bike, which can be scanned by users to unlock the bike.

1.2 The system shall allow users to scan the QR code with the mobile app to rent a bike.

1.3 The system shall validate the QR code to ensure it corresponds to a real bike and is not expired.

1.4 The system shall send a notification to the user confirming the bike rental after a successful scan.

Priority Level: High
Precondition: The bike must have a QR code, and the user must have access to the mobile app.
Cross-references: QA3, QA5 (example)

3.1.11 Help and FAQs

Functional Requirements (FRs)

1.1 The system shall provide a comprehensive FAQ section accessible via the mobile app or website.

1.2 The system shall allow users to search for answers to common questions (e.g., bike rental process, payment issues).

1.3 The system shall provide a contact option for users to get in touch with customer support for unresolved issues.

1.4 The system shall update the FAQ section periodically based on user feedback and common issues.

Priority Level: Medium
Precondition: User must have access to the app or website.
Cross-references: QA3, QA5 (example)

3.1.12 Ride History

Functional Requirements (FRs)

1.1 The system shall allow users to view their past rides, including rental times, distance traveled, and total cost.

1.2 The system shall update the ride history in real-time after each bike rental and return.

1.3 The system shall allow users to filter their ride history by date, duration, or cost.

1.4 The system shall allow users to download or email their ride history for personal reference.

Priority Level: Medium
Precondition: User must have an active account and completed at least one ride.
Cross-references: QA3, QA5 (example)

3.2 Non-Functional/Quality Requirements

QA1: Usability

The system shall ensure that a user can successfully rent a bike and complete the process, including payment and unlocking the bike, in an average of three minutes and a maximum of five minutes.

Priority Level: Medium
Precondition: N/A
Cross-references: QA2 (Performance), QA4 (Reliability)

QA2: Performance

The system shall process and respond to user actions, such as searching for nearby bikes or completing a payment, within two seconds for 90% of the transactions.

Priority Level: High
Precondition: N/A
Cross-references: QA1 (Usability), QA3 (Security)

QA3: Security

The system shall use encryption to secure user data, such as login credentials and payment information, to prevent unauthorized access.

Priority Level: High
Precondition: N/A
Cross-references: QA2 (Performance), QA4 (Reliability)

QA4: Reliability

The system shall maintain 99.9% uptime to ensure users can access the service without interruptions.

Priority Level: High
Precondition: Servers and network infrastructure are operational.
Cross-references: QA2 (Performance)

QA5: Accessibility

The system shall be compatible with both Android and iOS devices, supporting screen readers and other accessibility tools to assist users with disabilities.

Priority Level: Medium
Precondition: Devices meet the system's minimum requirements.
Cross-references: QA1 (Usability)

3.3 Project Requirements

- **Time, Budget, Human Resource:**
  The bike-sharing system is categorized as a Semi-Detached Project due to its moderate complexity, involvement of experienced developers, and some innovative features.

  The basic effort equation is: $E = a * (KLOC)^b$

  E = Effort in person-months
  KLOC = Estimated lines of code (in thousands)
  a, b = Constants based on the project type
  For a Semi-Detached Project, a=3.0, b=1.12

  Assuming the system requires 20,000 lines of code (20 KLOC):
  $E = 3 * (20)^{1.12}$
  $E = 81.42$ person -months

  The development time equation is: $T = c * (E)^d$
  T = Development time in months
  c, d = Constants based on the project type
  For a Semi-Detached Project: c=2.5, d=0.35

  $T = 2{,}5 * (81.42)^{0.35} = 11.7\ months$
  Team Size = $\frac{E}{T} = \frac{81.42}{11.7} \sim 7\ people$

  Effort: 81.42 person-months
  Development Time: 11.7 months
  Team Size: 7 people

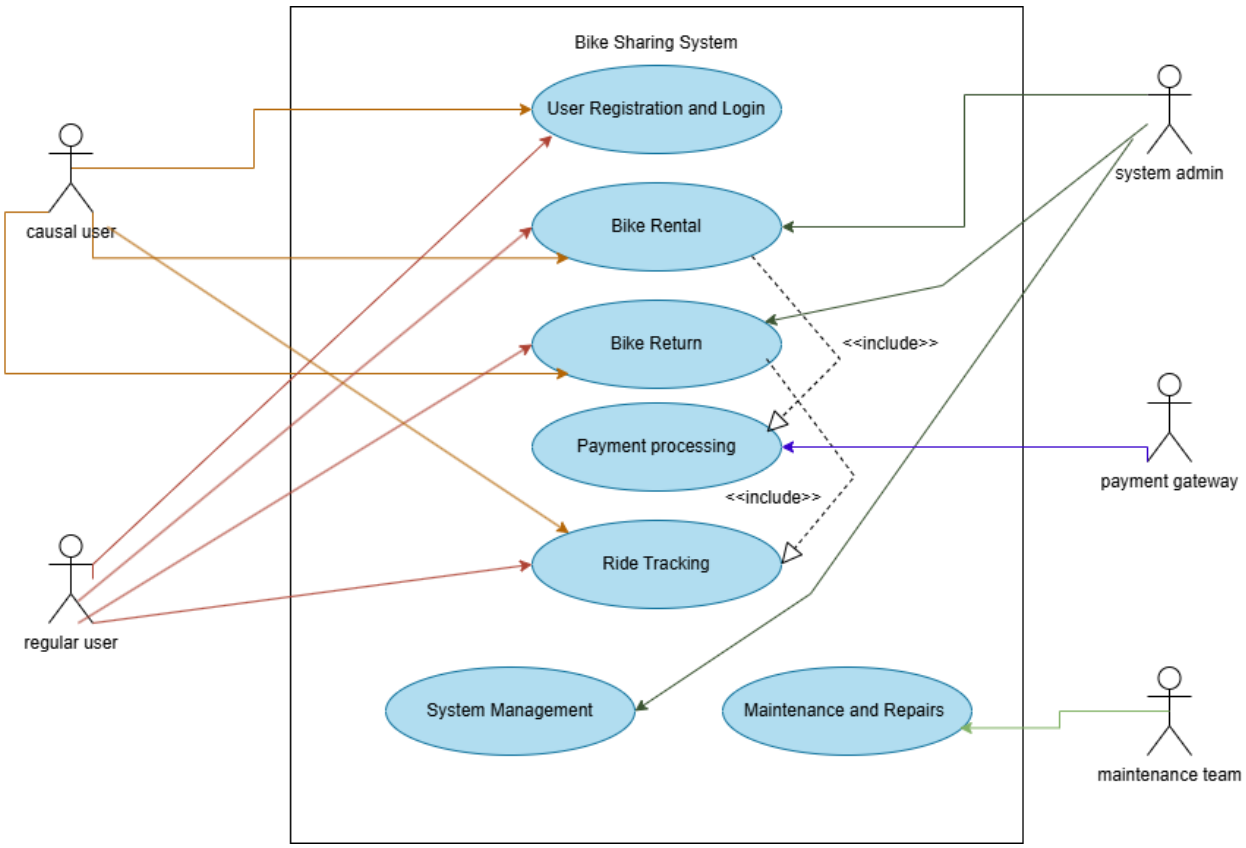
- **Tools:**
  - The system developer needs selenium tools in perform testing activities in week 6(Testing tool)
  - IDEs like Visual Studio Code and PyCharm enable efficient coding also GitHub ensures smooth collaboration, while MySQL handle database management.
  - Figma or Adobe XD or Balsamiq support UI/UX design.

- **Services:**
  - Charges for using third-party APIs like maps or payment gateways.
  - Regular expenses for updating the system and fixing issues.
  - Payments for maintaining the server infrastructure to handle system operations.

4. Design and Interface Requirements

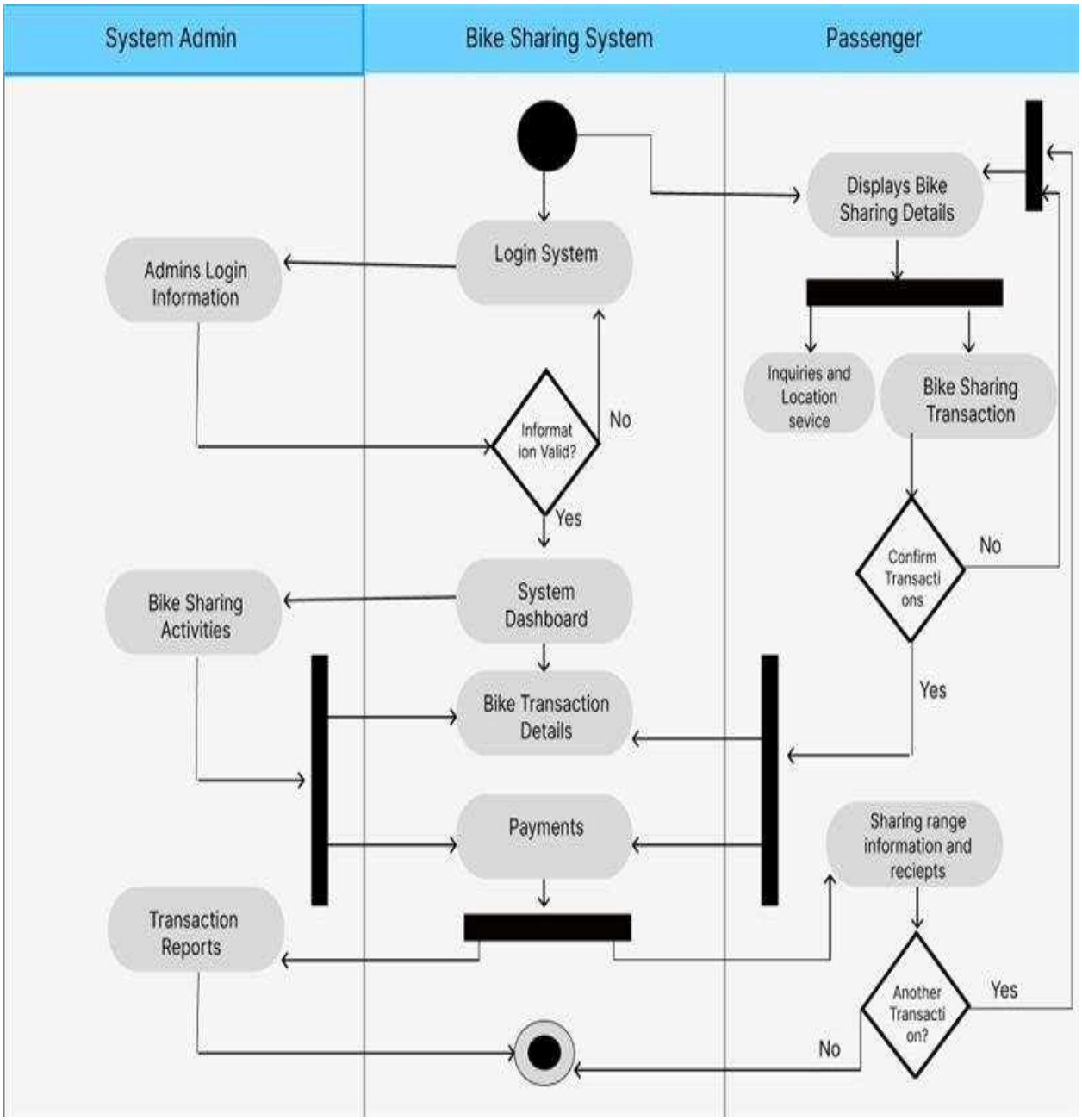
4.1 UML Diagrams

- **Use Case Diagram –**

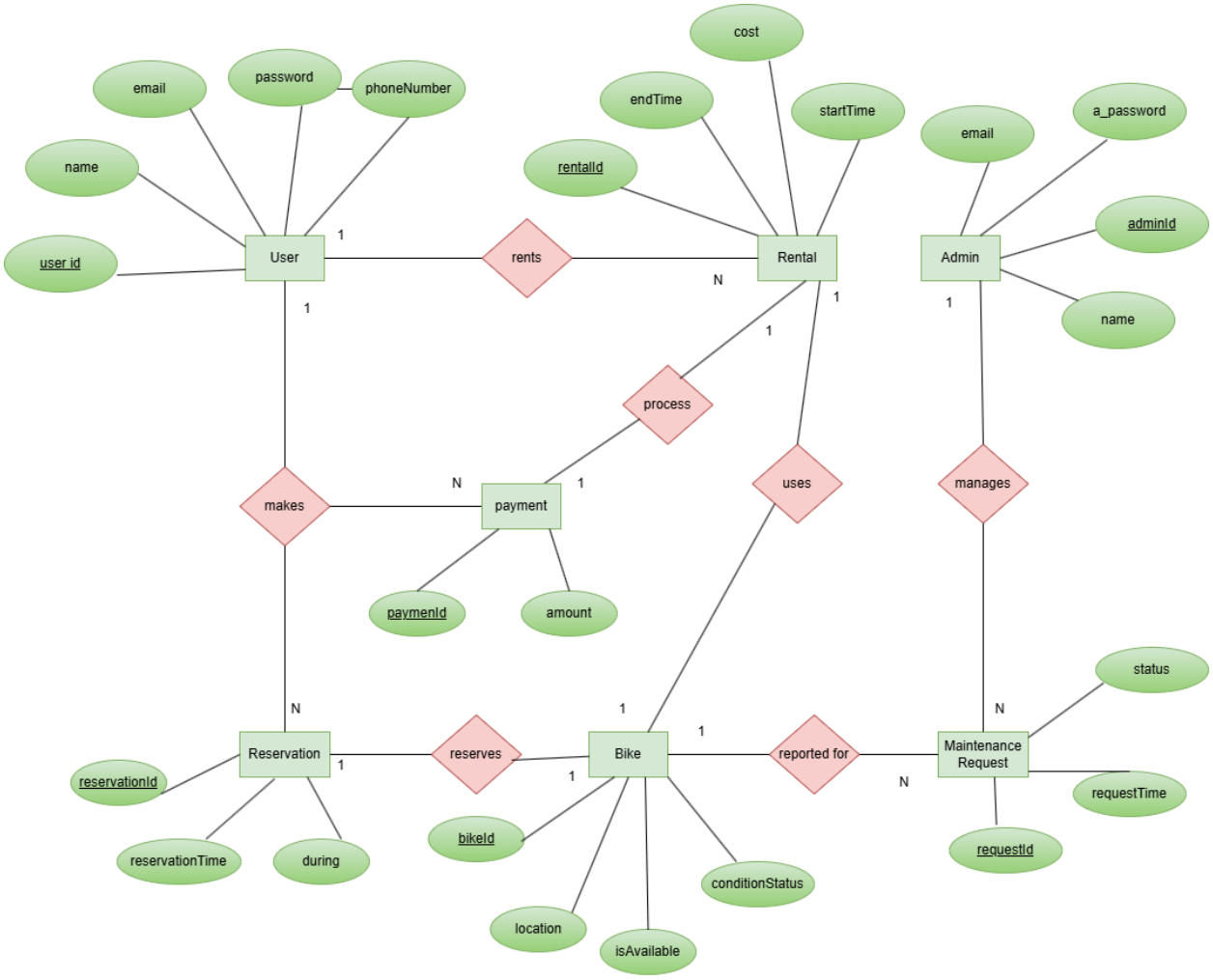


Class Diagram –

Activity Diagram –


System Admin
Bike Sharing System
Passenger
Displays Bike Sharing Details
Login System
Admins Login Information
Inquiries and Location sevice
Bike Sharing Transaction
Informat ion Valid?
No
Yes
Confirm Transacti ons
No
Bike Sharing Activities
System Dashboard
Yes
Bike Transaction Details
Payments
Sharing range information and reciepts
Transaction Reports
Another Transacti on?
Yes
No

E-R Diagram –



4.2 Data Dictionary

1.User Entity

Entity	Attribute	Type/Size	Validation	Key
User	UserId	Number (5)	10000-99999	Primary
User	Name	Text(30)	Required	
User	Email	Text(50)	Valid email format	
User	Phone Number	Text(15)	Valid Phone Number	
User	Password	Text (20)	Minimum 8 Characters	

2. Bike Entity

Entity	Attribute	Type/Size	Validation	Key
Bike	BikeId	Number (5)	Unique and Required	Primary
Bike	Location	Text (50)	Required	
Bike	isAvailable	Boolean	True/False	
Bike	ConditionStatus	Text (20)	Required	

3.Reservation Entity

Entity	Attribute	Type/Size	Validation	Key
Reservation	ReservationId	Number (5)	Unique and Required	Primary
Reservation	UserId	Number(5)	Valid Foreign Key	Foreign
Reservation	BikeId	Number(5)	Valid Foreign Key	Foreign
Reservation	ReservationTime	DataTime	Required	
Reservation	Duration	Number(3)	Positive Value	

4.Rental Entity

Entity	Attribute	Type/Size	Validation	Key
Rental	RentalId	Number(5)	Unique and Required	Primary
Rental	UserId	Number(5)	Valid Foreign Key	Foreign
Rental	BikeId	Number(5)	Valid Foreign Key	Foreign
Rental	StartTime	DataTime	Required	
Rental	EndTime	DataTime	Required	
Rental	Cost	Decimal(10)	Positive Value	

5. Payment Entity

Entity	Attribute	Type/Size	Validation	Key
Payment	paymentId	Number (5)	Unique and Required	Primary
Payment	userId	Number (5)	Valid Foreign Key	Foreign
Payment	amount	Decimal (10)	Positive Value	

6. MaintenanceRequst Entity

Entity	Attribute	Type/Size	Validation	Key
MaintananceRequest	requestId	Number (5)	Unique and Required	Primary
MaintananceRequest	bikeId	Number (5)	Valid Foreign Key	Foreign
MaintananceRequest	requestTime	DateTime	Required	
MaintananceRequest	status	Text (20)	Pending/Resolved	

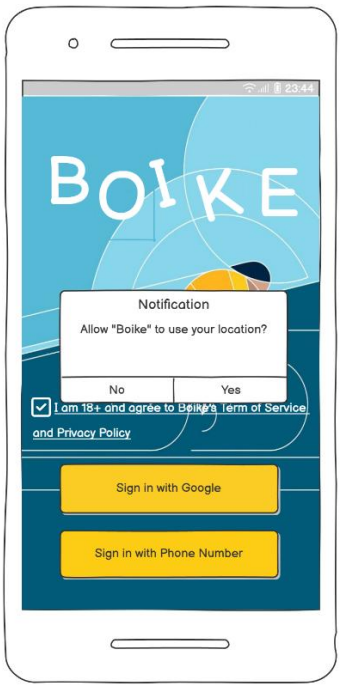
7. Admin Entity

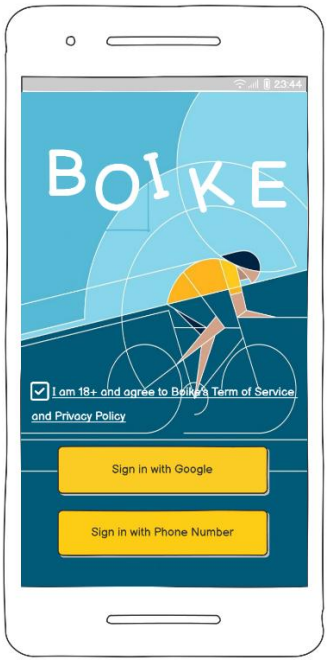
Entity	Attribute	Type/Size	Validation	Key
Admin	adminId	Number (5)	Unique and Required	Primary
Admin	name	Text (30)	Required	
Admin	email	Text (50)	Valid Email Format	
Admin	password	Text (20)	Minimum 8 characters	

4.3 UI/UX Design Specification

- Balsamiq was used for II/UX design

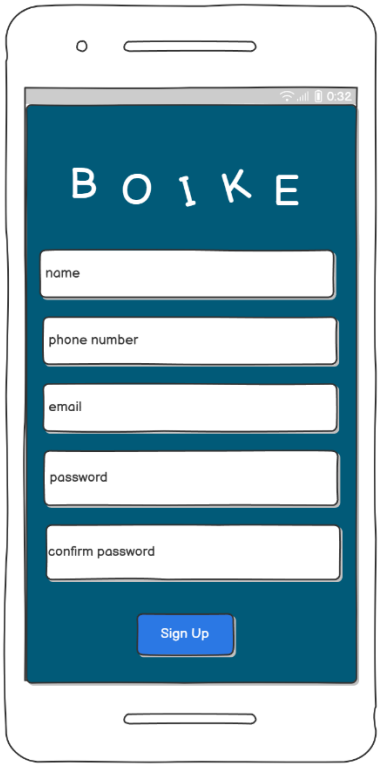
- Homepage-



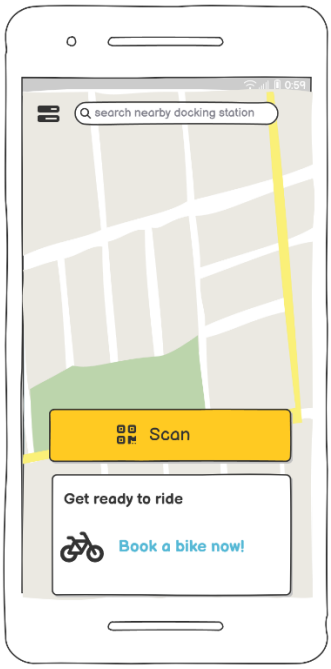


- Login page –

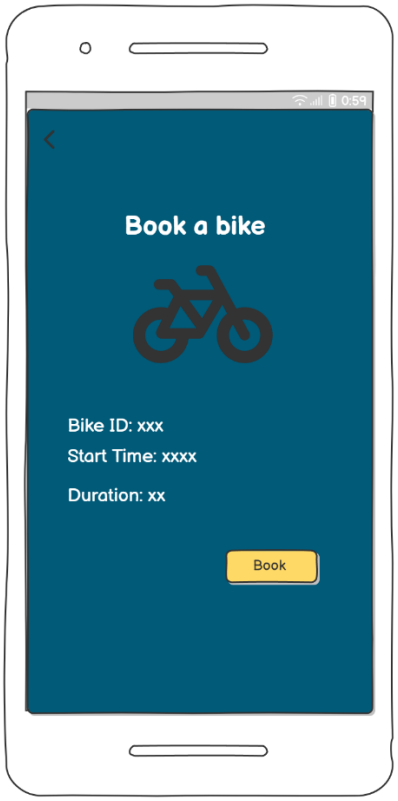
- Registration Page –


BOIKE
name
phone number
email
password
confirm password
Sign Up

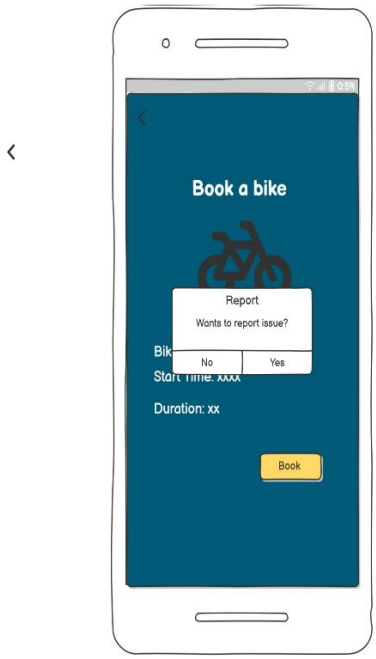
- Dashboard –

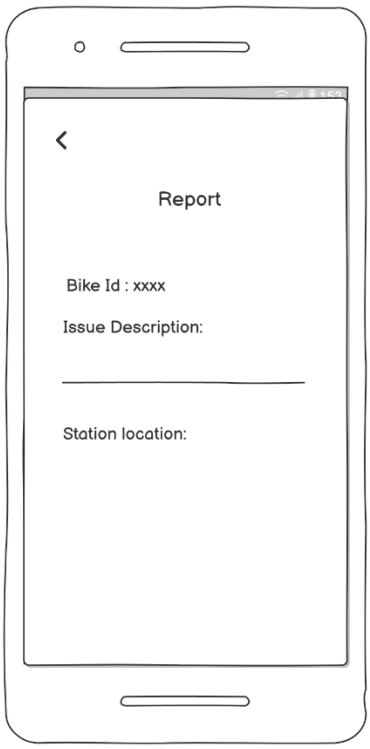

search nearby docking station
Scan
Get ready to ride
Book a bike now!

- Book bike –

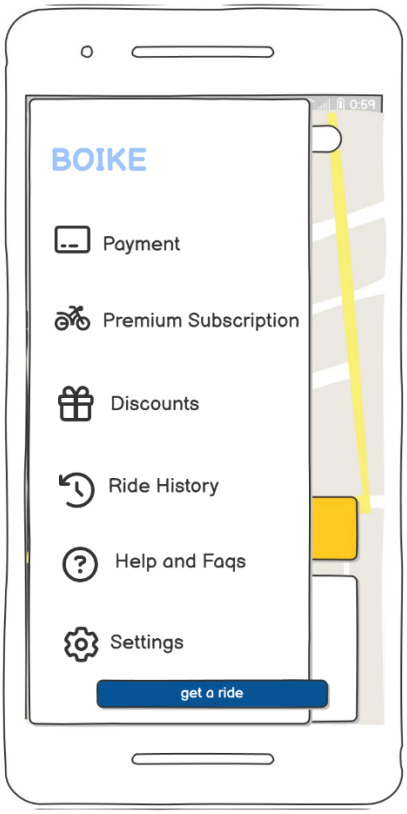


- Report issue –

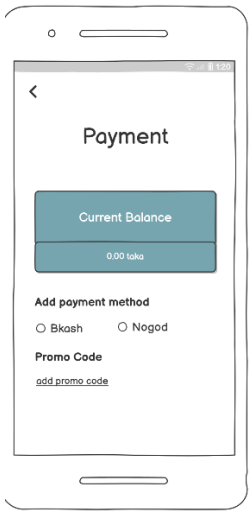



Report
Bike Id : xxxx
Issue Description:
Station location:

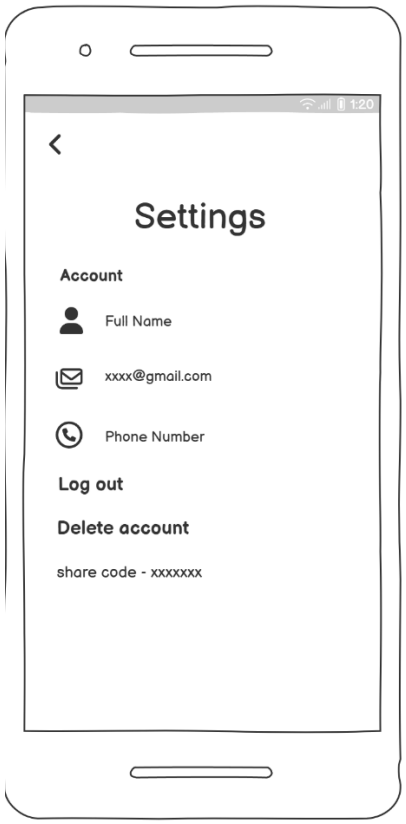
- Sidebar –


BOIKE
Payment
Premium Subscription
Discounts
Ride History
Help and Faqs
Settings
get a ride

- Payment –



- Settings –



Text Format:

- Style: Times New Roman
- Size: 12
- Space: 1.0
- Alignment: Justify