COMET-RIDE

Logo, company name

Description automatically generated

**MIS-6308**

**Professor Srinivasan Raghunathan**

**Group 28**

**Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju**

Table of Contents

[1. Executive Summary 3](#_Toc134455353)

[2. Problem Statement 3](#_Toc134455354)

[3. Business Need 3](#_Toc134455355)

[4. Objectives 3](#_Toc134455356)

[5. Scope 4](#_Toc134455357)

[6. Proposed Solution 4](#_Toc134455358)

[7. Business process model using BPMN for the key business processes. 5](#_Toc134455359)

[8. Context Diagram for the proposed system 6](#_Toc134455360)

[9. Use Case Diagram 7](#_Toc134455361)

[10. Use Case Description 7](#_Toc134455362)

[11. Data dictionary 12](#_Toc134455363)

[12. Class Diagram 14](#_Toc134455364)

[13. Sequence Diagram 16](#_Toc134455365)

[14. Functional Specification Document for the proposed system 16](#_Toc134455366)

[15. Interface Design 17](#_Toc134455367)

[16. Database Design 20](#_Toc134455368)

[17. Complete Class Diagram 21](#_Toc134455369)

[18. Software Design 21](#_Toc134455370)

[19. Project Management 24](#_Toc134455371)

[i. Project Activities 24](#_Toc134455372)

[ii. Allocation of Activities to Team Members 26](#_Toc134455373)

[iii. Planned Timeline 26](#_Toc134455374)

[iv. Minutes of Meeting 27](#_Toc134455375)

# Executive Summary

This project allows team members to incorporate collecting problems and solving them in a system analysis design methodology. It gave us a chance to collect raw data, the requirements of the customer, plan the framework of the structure, and have a deeper understanding of the system analysis and project management.

The project is to focus on UTD Services application, one of the most important and popular applications of the students’ daily life in UTD. By adding a new feature, the comet bike, will greatly increase the convenience of students travelling inside the campus from one building to another within the 2-mile radius.

This application provides the live availability of bikes at zones, unlocking and locking the bike from the zone and it also shows the live tracking of bike to monitor it. The project focuses on a service to all the comets that students lack in having intra transport service.

Below are some of the important features offered by UTD Services:

1. Maps: Find classroom, an office, or a meeting location, and look up building names and abbreviations.
2. Parking: Check real-time availability for Parking Structures.
3. Dining: Show open restaurants and food trucks, order food Online, find vending services.
4. Transit: Display the bus route and arrangement information.

# Problem Statement

Students at UT Dallas have recently reported difficulty transiting the campus. The extreme temperatures like intense heat for much of the year and cold for few months, longer distances between buildings, and decreased productivity due to transit time are just a few of the significant issues we discuss and intend to solve with a simple solution.

# Business Need

CometRide is a unique concept developed by our team to address the issue. It will have multiple functionalities built into it at first, with more and more features being added in subsequent months. There are numerous benefits to it, all aimed at improving the students' lives and productivity.

We hope that this project will be fully functional and implemented as soon as possible because UTD lags its competitors in this space who already have an existing system like this for their students and staff.

# Objectives

Objectives of the proposed system are as follows.

* Check the bike availability and display them to the users.
* Help User to book a ride.
* Display available parking locations to the user.
* Issue and inventory management system.

# Scope

This application tracks and displays, via GPS, the bikes available nearby or at bike Zones around the campus within 2 miles radius. Comets can search for their destination/location and see the estimated time to reach the designated bike zone. Comets can scan the QR code given on the vehicle to begin or end the trip and can park their bike at zones nearest to their destination. Upon this, the added service will be released from the comet account. For the service, to be available to everyone, the minimum service fee will be included in the tuition fee.

The fundamental components of this project are:

HTML, CSS, JavaScript, Python, Raspberry Pi - We will be using the mentioned programming languages to develop the project.

HTML - To create the structure of the CometRide website.

CSS - Styling the CometRide website.

JavaScript - To create dynamic webpages.

Python - To create backend software.

# Proposed Solution

1. Conduct user research: Collect feedback from UTD students regarding their transportation needs and preferences. Determine if there is demand for an intra-campus transportation service like Comet bike.

2. Define requirements: Based on the user research, define the requirements for the Comet bike feature, including the ability to view bike availability at zones, unlock and lock bikes from the zones, and live tracking of bikes.

3. Plan the structure: Plan the technical structure of the Comet bike feature, including the necessary hardware and software components, and how they will be integrated with the existing UTD Services app.

4. Develop the feature: Develop the Comet bike feature, including the hardware components such as bike locks and the software components such as the bike availability display and live tracking.

5. Test the feature: Test the Comet bike feature to ensure it is functioning properly and meets the defined requirements.

6. Deploy the feature: Deploy the Comet bike feature to the UTD Services app and make it available to students.

7. Collect feedback: Collect feedback from students on the Comet bike feature to determine if any improvements are needed or if additional features should be added in the future.

# Business process model using BPMN for the key business processes.

Diagram

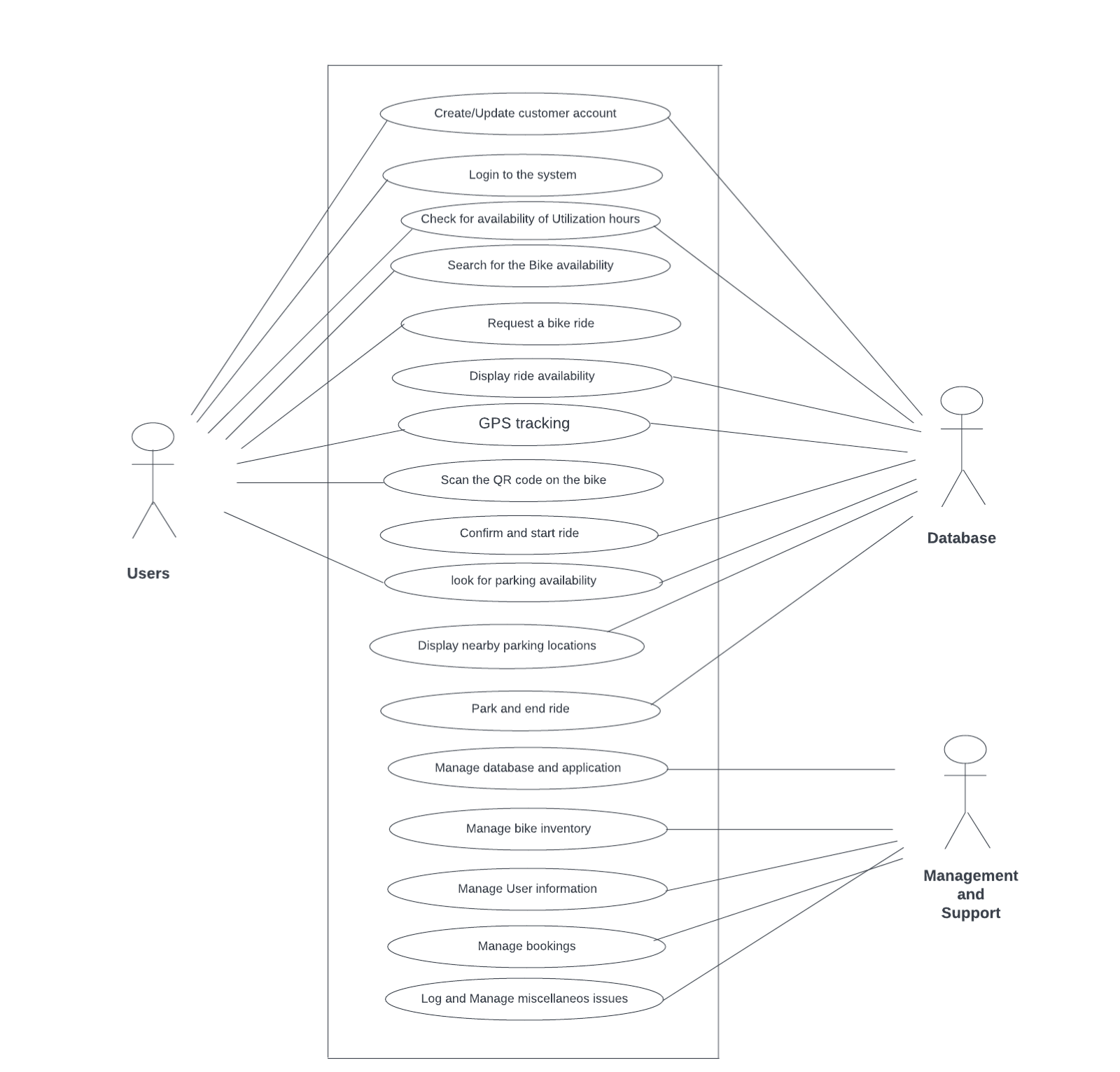
Description automatically generated

# Context Diagram for the proposed system

Diagram

Description automatically generated

# Use Case Diagram



# Use Case Description

**Use Case Description 1:**

|  |  |
| --- | --- |
| **Use Case Name** | **Create/Update customer Account** |
| **Primary Actor:** | User |
| **Brief Description** | User wants to create/Update the Account |
| **Trigger** | Passenger will be able to create/update the account with UTD ID and Password |
| **Normal Flow of events:** | 1. The user navigates to the system.  2. User selects account update/create option.  3. Passenger enters the username (UTD ID), password as inputs.  5. If verified, login and allow the passenger to use the system. |
| **Exception Flow** | 1. If the required inputs are provided and the credentials match with the credentials stored in DB then redirect the passenger to main page. If the credentials do not match, then prompt the passenger to reenter the credentials. |

**Use Case Description 2:**

|  |  |
| --- | --- |
| **Use Case Name** | **User Log In** |
| **Primary Actor:** | User |
| **Brief Description** | User wants to log in to the System |
| **Trigger** | User will be able to log in to the System |
| **Normal Flow of events:** | 1. The user navigates to the system.  2. User selects sign in option.  3. User enters the username (UTD ID), password as inputs.  4. Checks if the User ID and password matches with the ID and password stored in the database.  5. If verified, login and allow the user to use the system. |
| **Exception Flow** | 1. If the required inputs are provided and the credentials match with the credentials stored in DB then redirect the user to main page. If the credentials do not match, then prompt the user to reenter the credentials. |

**Use Case Description 3:**

|  |  |
| --- | --- |
| **Use Case Name** | **Check Availability of Utilization Hours** |
| **Primary Actor:** | User |
| **Brief Description** | User wants to check the limit for the utilization of Hours |
| **Trigger** | User will be able to check the hours limit into the System |
| **Normal Flow of events:** | 1. The user navigates to the system.  2. User selects sign in option.  3. User enters the username (UTD ID), password as inputs.  4. Checks if the User ID and password matches with the ID and password stored in the database.  5. If verified, login and allow the user to use the system.  6. Check for the availability of the utilization hours |
| **Exception Flow** | 1. If there is no limit for the utilization hours, display the usage after a minimum number of hours. |

**Use Case Description 4:**

|  |  |
| --- | --- |
| **Use Case Name** | **Search for ride** **availability** |
| **Primary Actor:** | User |
| **Brief Description** | User wants to search for a ride |
| **Trigger** | User enters from and to locations and searches for a ride |
| **Normal Flow of events:** | 1. The passenger opens the app.  2. User signs in the app.  3. User enters source and destinations  4. User searches for available rides |

**Use Case Description 5:**

|  |  |
| --- | --- |
| **Use Case Name** | **Request ride** |
| **Primary Actor:** | User |
| **Brief Description** | User wants to request a ride |
| **Trigger** | User enters from and to locations and requests ride |
| **Normal Flow of events:** | 1. The User opens the app.  2. User signs in the app.  3. User enters source and destinations  4. User request ride for the source and destination |

**Use Case Description 6:**

|  |  |
| --- | --- |
| **Use Case Name** | **Display ride** **availability** |
| **Primary Actor:** | Database |
| **Brief Description** | Database displays ride availability |
| **Trigger** | User enters source and destination and database displays ride availability |
| **Normal Flow of events:** | 1. The user opens the app.  2. User signs in the app.  3. User enters source and destinations  4. User searches for available rides  5. Database searches and displays the ride availability. |

**Use Case Description 7:**

|  |  |
| --- | --- |
| **Use Case Name** | **GPS tracking** |
| **Primary Actor:** | Database, User |
| **Brief Description** | Database or User tracks the bike |
| **Trigger** | User or Database triggers the tracking of the bike |
| **Normal Flow of events:** | 1. The user opens the app.  2. User signs in the app.  3. User enters source and destinations  4. User searches for available rides  5. Database searches and displays the ride availability.  6. Database or User can track the bike with the bike id. |

**Use Case Description 8:**

|  |  |
| --- | --- |
| **Use Case Name** | **Scan the QR code on bike** |
| **Primary Actor:** | User |
| **Brief Description** | To start the bike, user has to scan the code |
| **Trigger** | Scanning of the QR code. |
| **Normal Flow of events:** | 1. The user opens the app.  2. User signs in the app.  3. User enters source and destinations  4. User searches for available rides  5. Database searches and displays the ride availability.  6. User scans the QR code to start the bike. |

**Use Case Description 9:**

|  |  |
| --- | --- |
| **Use Case Name** | **Scan the QR code on bike** |
| **Primary Actor:** | User |
| **Brief Description** | To start the bike, user has to scan the code |
| **Trigger** | Scanning of the QR code. |
| **Normal Flow of events:** | 1. The user opens the app.  2. User signs in the app.  3. User enters source and destinations  4. User searches for available rides  5. Database searches and displays the ride availability.  6. User scans the QR code to start the bike. |

**Use Case Description 10:**

|  |  |
| --- | --- |
| **Use Case Name** | **Confirm and start ride** |
| **Primary Actor:** | User |
| **Brief Description** | To start the bike, user has to scan the code and start ride |
| **Trigger** | Scanning of the QR code and click the start ride option |
| **Normal Flow of events:** | 1. The user opens the app.  2. User signs in the app.  3. User enters source and destinations  4. User searches for available rides  5. Database searches and displays the ride availability.  6. User scans the QR code to start the bike. |

**Use Case Description 11:**

|  |  |
| --- | --- |
| **Use Case Name** | **Search for parking** **availability** |
| **Primary Actor:** | User |
| **Brief Description** | User wants to park the ride |
| **Trigger** | User searches for a parking the ride. |
| **Normal Flow of events:** | 1. The passenger opens the app.  2. User signs in the app.  3. User enters source and destinations  4. User searches for available rides  5. After ride completion user searches for parking availability. |

**Use Case Description 12:**

|  |  |
| --- | --- |
| **Use Case Name** | **Display parking availability** |
| **Primary Actor:** | Database |
| **Brief Description** | Database displays parking availability |
| **Trigger** | User enters source and destination and database displays parking availability |
| **Normal Flow of events:** | 1. The user opens the app.  2. User signs in the app.  3. User enters source and destinations  4. User searches for available rides  5. Database searches and displays the ride availability.  6. After ride end user searches for parking availability and database displays parking availability |

**Use Case Description 13:**

|  |  |
| --- | --- |
| **Use Case Name** | **Manage database and application** |
| **Primary Actor:** | Management and support system |
| **Brief Description** | Database is managed by Management and Support system |
| **Trigger** | Database maintenance |
| **Normal Flow of events:** | 1 Management and support team maintains the database.  User data, bike data etc. |

**Use Case Description 14:**

|  |  |
| --- | --- |
| **Use Case Name** | **Manage bike inventory** |
| **Primary Actor:** | Management and support system |
| **Brief Description** | Bike inventory is managed by Management and Support system |
| **Trigger** | Inventory requests |
| **Normal Flow of events:** | 1 Inventory management requests raised by user or anyone.  2. It is investigated and resolved by the management support system. |

**Use Case Description 15:**

|  |  |
| --- | --- |
| **Use Case Name** | **Manage user information** |
| **Primary Actor:** | Management and support system |
| **Brief Description** | User information is managed by Management and Support system |
| **Trigger** | User requests |
| **Normal Flow of events:** | 1 User requests or any update for user information.  2. Management and support teams updates or maintains the user information. |

**Use Case Description 16:**

|  |  |
| --- | --- |
| **Use Case Name** | **Manage bookings** |
| **Primary Actor:** | Management and support system |
| **Brief Description** | Manage bookings managed by Management and Support system |
| **Trigger** | Booking requests |
| **Normal Flow of events:** | 1 Booking requests raised by user or anyone.  2. It is investigated and resolved by the management support system. |

**Use Case Description 17:**

|  |  |
| --- | --- |
| **Use Case Name** | **Log and manage miscellaneous issues** |
| **Primary Actor:** | Management and support system |
| **Brief Description** | Issues are managed by Management and Support system |
| **Trigger** | Issue requests |
| **Normal Flow of events:** | 1 Issues are logged by the user.  2. It is investigated and resolved by the management support system. |

# Data dictionary

Below consists of the data associated with different Use-Cases:

**Create/Update Account:**

Create\_Account = loginId +password+ userName+ userType+ mail+ mobileNo

loginId = studentID

Password = Data Element

userType = [Student | Visitor | Alumni | Faculty]

userName = First Name + (Middle Initial) + Last Name

mail = UTD mail ID

**Log In:**

Log In = loginId + Password

loginId = studentID

Password = Data Element

**Check for availability of utilization hours:**

Utilization Hours = loginId + Password + utilizationHours

loginId = studentID

Password = Data Element

Utilizationhours = No of hours remaining for the bike utilization

**Search for Bike availability:**

Search for Ride = Current Location + Destination + Pickup Location

Pickup Location = Data Element

Current Location = [GPS Location | Custom Location]

Destination = [GPS Location | Custom Location]

**Request a bike ride:**

Bike Booking = bookingID+bikeID+bikeStatus+bikeLocation

bookingID = ID of the Booking

bikeID = ID of the bike booked

bikeStatus = condition of the bike

bikeLocation = Current Location

**Display ride availability:**

Ride availability = Location+ bikeID

bikeID = Data Element

Location = Location of the bike

**Looking for parking availability:**

Search for parking = parking Location

parking Location = [GPS Location | Custom Location]

Rating = Star Rating

**Log and manage miscellaneous issues:**

Issue logging = bikeid+issueInfo+issueStatus

bikeiD = Data Element

issuseInfo =issue description

issueStatus = [Resolved | Not Resolved]

**Data Elements example:**

• Date Element Name: bikeId

• Alias: Rider Id

• Type: Numeric

• Length: 5

• Picture: 99-999

•      Source: New bikeId creation process

•      Update: cannot be updated …

# Class Diagram

Diagram

Description automatically generated  
**Assumptions:**

• A user can book one bike at a time.

• A user may log issues about the bike.

• System handles multiple bookings.

• Each booking must have a single user.

• System can track multiple bikes ( TrackingSystem Class).

•      System can check multiple parking availabilities ( ParkingSystem Class).

• Users can track the bikes.

**User Table:**

* Primary Key Constraints/Integrity Constraint

Primary Key ‘ loginID’ and attribute ‘userName’ should not be NULL.

* Uniqueness Constraint

o Primary Key ‘loginID’ and attribute ‘mail‘ should be unique.

* Domain Specific / Context Specific Constraints

o Attributes ’, ‘mail and ‘userName’ should not have null value.

**System Table:**

· Uniqueness Constraint

o Primary Key ‘loginID’, ‘searchID’, ‘BookingID’ and ‘TrackingID’ should be unique and should be the foreign keys of respective table user, tracking, booking, Parking tables.

**Booking Table:**

* Primary Key Constraints/Integrity Constraint

o Composite Primary Key ‘bookingID’ should not be NULL.

* Referential Integrity Constraint

o ‘UserID’ which is foreign key in Reservation table should exist as the primary key in User respectively.

**Tracking System Table:**

* Primary Key Constraints/Integrity Constraint

o Composite Primary Key ‘trackingID’ should not be NULL.

**Parking System Table:**

* Primary Key Constraints/Integrity Constraint

o Composite Primary Key ‘searchID’ should not be NULL.

**Issue Table:**

* Primary Key Constraints/Integrity Constraint

o Composite Primary Key ‘IssueID’ should not be NULL.

* Referential Integrity Constraint

o ‘UserID’ which is foreign key in Reservation table should exist as the primary key in User respectively.

**Management and Support systems:**

· Uniqueness Constraint

o Primary Key ‘issueID’ should be unique and should be the foreign keys of respective table Issue.

# Sequence Diagram

Graphical user interface, application

Description automatically generated

# Functional Specification Document for the proposed system

* The proposed system will make it easy for the students at UTD to commute inside the campus.
* The proposed system will create a common platform for the riders who wish to take a ride to college or the nearby areas around the campus. They can simply login to the portal as a rider and submit their UTD ID as proof that they belong to the university.
* Once confirmed by the administrator, they will be able to put their ride details, including the source and the destination and the time at which they wish to receive the ride or if they are flexible with the ride timings.
* If they can find a similar ride, they can send a request to the driver of the ride. If not, a new ride request will be created in the system, which can be referred to in future until the desired ride period.
* The Proposed system will similarly provide a platform for drivers where they can enter the ride details of the ride, they are planning to take with source destinations, time, and route options to be able to help some rider who is on the same way.
* The driver’s riders can cancel their rides or change their schedule as per the current rider request in the system.
* The proposed system also offers admin support in case of any issues or concerns raised by riders to as to ensure great experience for the riders.
* If there is a problem with the bike, the riders can report it on the website. The issue will be taken care of by the support team.

# Interface Design

**Graphical user interface, website

Description automatically generated with medium confidence**

**Graphical user interface, application

Description automatically generated**Map

Description automatically generated with medium confidenceQr code

Description automatically generated

# Database Design

Diagram

Description automatically generated

# Complete Class Diagram

Diagram

Description automatically generated

# Software Design

**Signature:**

|  |
| --- |
| MethodName : UserLogin() |
| Client: User |
| Arguments received: StudentID |
| Associated Use case: UserLogin, Verify login |
| Type of value returned: Yes/No (for verification) |
| Post Condition: Allow login if Yes |

**Logic:**

IF Request Type = Login

THEN

IF Request Person = User

THEN

FETCH student ID

FETCH UTD\_Database

IF (COMPARE student ID= student ID in database)

THEN

Yes (Accept Sign In request)

ELSE

No (Decline Sign in request)

**Signature:**

|  |
| --- |
| MethodName: create/update account () |
| Client: User |
| Arguments received: StudentID |
| Associated Use case: user details |
| Type of value returned: Yes/No (updated) |
| Post Condition: Allow login if Yes |

**Logic:**

IF Request Type = create/update account

THEN

IF Request Person! = User

THEN

INSERT New student ID

UPDATE UTD\_Database

IF UPDATED

THEN

Yes (Accept Sign In request)

ELSE

No (Decline Sign in request)

**Signature:**

|  |
| --- |
| MethodName : Search for Bike Availability() |
| Client: User |
| Arguments received: Source, Destination |
| Associated Use case: Search bike |
| Type of value returned: String |
| Post Condition: Display bikes if available else create ride in the system |

**Logic:**

IF Request Type = Search for Bike Availability

THEN

FETCH User input parameters: Source , Destination, Time

FETCH UTDRides Database for same input parameters.

FETCH data from GPS for the same input parameters.

IF

User Input = UTD Ride database rides

THEN Send ride request option.

ELSE

CREATE ride request in system.

**Signature:**

|  |
| --- |
| MethodName : Book A ride() |
| Client: User |
| Arguments received: Source, Destination |
| Associated Use case: Search bike |
| Type of value returned: String |
| Post Condition: Display bikes if available else create ride in the system |

**Logic:**

IF Request Type = Search for Bike Availability

THEN

FETCH User input parameters: Source, Destination, Time

FETCH UTDRides Database for same input parameters.

FETCH data from GPS for the same input parameters.

IF

User Input = UTD Ride database rides

THEN Send ride request option.

IF Booking Confirmation == yes

THEN Confirm ride as Booked

**Signature:**

|  |
| --- |
| MethodName : Search for parking Availability() |
| Client: User |
| Arguments received: Source, Destination |
| Associated Use case: Search parking |
| Type of value returned: String |
| Post Condition: Display bike parking available |

**Logic:**

IF Request Type = Search for Parking Availability

THEN

FETCH User input parameters: Source, Destination, Time

FETCH UTDRides Database for same input parameters.

FETCH data from GPS for the same input parameters.

IF

User Input = UTD parking availability database rides

THEN display parking availability option.

**Signature:**

|  |
| --- |
| MethodName : End ride() |
| Client: User |
| Arguments received: Source, Destination |
| Associated Use case: Search parking |
| Type of value returned: String |
| Post Condition: End ride |

**Logic:**

IF Request Type = Search for Parking Availability

THEN

FETCH User input parameters: Source, Destination, Time

FETCH UTDRides Database for same input parameters.

FETCH data from GPS for the same input parameters.

IF

User Input = UTD parking availability database rides

THEN display parking availability option.

      IF END Ride == ‘yes’

      THEN Bike is parked, and ride is ended.

**Signature:**

|  |
| --- |
| MethodName : CheckIssueRequest() |
| Client: Rider |
| Arguments received: Complaints |
| Associated Use case: HelpCenter, resolveIssue |
| Type of value returned: String |
| Post Condition:  Create tickets and offer resolution |

**Logic:**

IF Report Concern == True

THEN

CREATE Incident in the system

FETCH Incident Details

SEND to Admin

OFFER resolution to driver/rider

CLOSE Ticket

# Project Management

## Project Activities

|  |  |  |  |
| --- | --- | --- | --- |
| **Task Name** | **Start Date** | **End Date** | **Duration** |
| **SAPM Group 28** | **1/28/23** | **4/7/23** | **69 days** |
| **Project Conception** | **1/28/23** | **2/9/23** | **12 days** |
| Project Conception | 1/28/23 | 2/1/23 | 5 days |
| Project Conception | 2/2/23 | 2/5/23 | 4 days |
| Project Conception | 2/6/23 | 2/9/23 | 3 days |
| **Project Planning** | **2/10/23** | **2/16/23** | **5 days** |
| Executive Summary | 2/10/23 | 2/12/23 | 3 days |
| Problem Statement | 2/13/23 | 2/16/23 | 2 days |
| **UML diagrams** | **2/17/23** | **4/7/23** | **49 days** |
| Business Process Model | 2/17/23 | 2/22/23 | 5 days |
| Context Diagram | 2/23/23 | 2/24/23 | 2 days |
| Use Case Diagram | 2/25/23 | 2/28/23 | 5 days |
| Use Case Descriptions | 3/1/23 | 3/3/23 | 3 days |
| Data Dictionary | 3/4/23 | 3/7/23 | 4 days |
| Class Diagram | 3/8/23 | 3/14/23 | 7 days |
| Sequence Diagram | 3/15/23 | 3/18/23 | 4 days |
| Functional Specifications | 3/19/23 | 3/21/23 | 3 days |
| **Project Design** | 3/22/23 | 4/4/23 | **14 days** |
| Interface Design | 3/23/23 | 3/24/23 | 3 days |
| Database Design | 3/25/23 | 3/29/23 | 5 days |
| Complete Class Diagram | 3/29/23 | 3/31/23 | 3 days |
| Software Design | 4/1/23 | 4/3/23 | 3 days |
| **Project Deployment** | 4/4/23 | 4/7/23 | **4 days** |
| Final Project Deliverable | 4/4/23 | 4/7/23 | 4 days |

## Allocation of Activities to Team Members

|  |  |
| --- | --- |
| NAME | ACTIVITY |
| Sai Sree Kottam | Problem Statement, Context Diagram, Activity Diagrams, Database Design, Function Specification, Report documentation, Proof-read |
| Nitun Panigrahy | Use Case Diagram, Use Case Descriptions, Sequence Diagram, Data Dictionary, Interface Design, Report documentation, Minutes of Meating |
| Mounica Satya Kavya Dakoju | Context Diagram, Use Case Descriptions, Sequence Diagram, BPMN, Function Specification, Class Diagram, Minutes of Meating |

## Planned Timeline

As part of our planned timeline, we decided to meet once every week in-person or over a group call to discuss project ideas and deliverables. We decided to keep work in progress and not keep anything for last minute submission. Below is the timeline chart we decided to follow at the beginning of this project.

|  |  |
| --- | --- |
| **Date** | **Tasks** |
| 01/28/2023 | Formation of Team groups on teams |
| 02/02/2023 | Discussion of Project Ideas |
| 02/5/2023 | Finalizing a Project Idea |
| 02/10/2023 | Write project proposal and executive summary, decide project scope and the functional requirements |
| 02/15/2023 | Divide workload and decide activities that needs to be done, list all topics needs to be focused more on |
| 02/17/2023 | Initiate the context diagram, document the use cases, create use case diagram and finally, divide the project deliverables. |
| 02/27/2023 | Sequence Diagram and BPMN |
| 03/08/2023 | Data Base Design |
| 03/13/2023 | Work completion due date |
| 03/14/2023 | Review of The Work Done |
| 03/15/2023 | Data dictionary |
| 03/19/2023 | Activity Diagram |
| 03/21/2023 | Function Specification and Class Diagram |
| 04/01/2023 | Interface Design |
| 04/04/2023 | Report documentation |
| 04/07/2023 | Final Draft Documentation |

## Execution Timeline:

|  |  |  |
| --- | --- | --- |
| **Estimated Date** | **Completed Date** | **Tasks** |
| 01/28/2023 | 02/02/2023 | Team Group Creation |
| 02/03/2023 | 02/05/2023 | Collaborate on project ideas. |
| 02/06/2023 | 02/10/2023 | Refined the project scope and determined the specific outcomes to be delivered. |
| 02/11/2023 | 02/16/2023 | Developed project proposal and executive summary, establish project scope and requirements. |
| 02/18/2023 | 02/22/2023 | Allocated tasks and determined the essential activities, as well as identified the areas that require more focus. |
| 02/25/2023 | 02/27/2023 | Discussed the context and use case diagrams and divided the project deliverables accordingly. |
| 02/27/2023 | 04/03/2023 | Completed project analysis, discussed risk factors, and finalized the design. |
| 04/04/2023 | 04/07/2023 | Created a Word document and updated individual parts and reported to the team about completed activities. |
| 04/07/2023 | 04/15/2023 | Completed project management part and assisted peers who faced difficulties in understanding any part. Created project presentation ppt and practiced presentation. |

## Minutes of Meeting

Allocation of activities to team members:

The group met and collaborated on every aspect of this project.

##### Meeting 1:

|  |  |
| --- | --- |
| Meeting Date/Minutes | February 10, 2023, 10:00 AM - 12:00 PM |
| Attendees | Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju |
| Discussion | Get to know the team members and project topic idea brainstorming. |
| Conclusion | Project idea brainstorming on different project concepts and agreed on three project topics. |

Meeting 2:

|  |  |
| --- | --- |
| Meeting Date/Minutes | February 17, 2023, 2:00 PM - 4:00 PM |
| Attendees | Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju |
| Discussion | Project topic selection. |
| Conclusion | On further analysis the team decided to finalize on “Comet ride” in the end. |

Meeting 3:

|  |  |
| --- | --- |
| Meeting Date/Minutes | February 25, 2023, 5:00 PM - 7:00 PM |
| Attendees | Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju |
| Discussion | Drafted an Executive Summary and Problem Statement. |
| Conclusion | We defined the project scope, established clear objectives, analyzed any potential constraints, and conceptualized to determine the functional requirements. Executive Summary and Problem statement was also finalized. |

Meeting 4:

|  |  |
| --- | --- |
| Meeting Date/Minutes | March 4, 2023, 9:00 AM - 11:00 AM |
| Attendees | Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju |
| Discussion | Context Diagram |
| Conclusion | Collaborate to create a context diagram and Business Process Model. |

Meeting 5:

|  |  |
| --- | --- |
| Meeting Date/Minutes | March 8, 2023, 3:00 PM - 5:00 PM |
| Attendees | Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju |
| Discussion | Use Case Diagram & Use Case Descriptions |
| Conclusion | Initiated and documented the use cases, followed by the creation of a diagram using Visual Paradigm. Assigned the completion of use case descriptions before the next meeting. |

Meeting 6:

|  |  |
| --- | --- |
| Meeting Date/Minutes | March 15, 2023, 6:00 PM - 8:00 PM |
| Attendees | Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju |
| Discussion | Sequence Diagram and BPMN |
| Conclusion | Allocated the task of completing the Sequence diagram |

Meeting 7:

|  |  |
| --- | --- |
| Meeting Date/Minutes | March 22, 2023, 11:00 AM - 1:00 PM |
| Attendees | Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju |
| Discussion | Review of The Work Done |
| Conclusion | Reviewed and deliberated on the enhancements made to the work. |

Meeting 8:

|  |  |
| --- | --- |
| Meeting Date/Minutes | March 29, 2023, 4:00 PM - 6:00 PM |
| Attendees | Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju |
| Discussion | Data Dictionary |
| Conclusion | Formulated the data dictionary for the project. |

Meeting 9:

|  |  |
| --- | --- |
| Meeting Date/Minutes | April 1, 2023, 1:00 PM - 3:00 PM |
| Attendees | Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju |
| Discussion | Activity Diagrams |
| Conclusion | Deliberated on the sequence diagrams and reviewed the pending tasks before the next meeting. |

Meeting 10:

|  |  |
| --- | --- |
| Meeting Date/Minutes | April 4, 2023, 8:00 AM - 10:00 AM |
| Attendees | Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju |
| Discussion | Database Design |
| Conclusion | Created the tables and their corresponding relationships for the database. |

Meeting 11:

|  |  |
| --- | --- |
| Meeting Date/Minutes | April 5, 2023, 7:00 PM - 9:00 PM |
| Attendees | Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju |
| Discussion | Function Specification and Class Diagram |
| Conclusion | Developed the functionality specifications and constructed the class diagram. |

Meeting 12:

|  |  |
| --- | --- |
| Meeting Date/Minutes | April 6, 2023, 12:00 PM - 2:00 PM |
| Attendees | Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju |
| Discussion | Interface Design |
| Conclusion | Decided how the interface looks and divided the work among ourselves. |

Meeting 13:

|  |  |
| --- | --- |
| Meeting Date/Minutes | April 7, 2023, 2:00 PM - 4:00 PM |
| Attendees | Sai Sree Kottam, Nitun Panigrahy, Mounica Satya Kavya Dakoju |
| Discussion | Report documentation, proof-read, and submission |
| Conclusion | Discussed and completed the Report |