Assignment 1 - Part I
Smart vehicle booking system

==========

For this task, each student will be tasked with crafting a high-level design for a software system. The design will be visualized through a single UML diagram, showcasing the comprehensive class structure of the system. This diagram should encompass class names, primary responsibilities, and the connections between classes, including aspects like inheritance, association, aggregation, and potentially details like cardinality and role names. A dedicated section of the document is required to provide concise descriptions outlining the responsibilities assigned to each class.

It is essential for students to be thorough in this initial design phase, as there **would be a subsequent part** in this assignment that will build upon the foundations laid here. The diligence and clarity exercised in creating this high-level design will significantly influence the success of the subsequent tasks.

==========

Problem Description:

We anticipate the development of a software system for reserving smart bikes intended for on-campus transportation at IIIT-H. The design should support users in booking smart bikes via a mobile app and conducting payment transactions. Users should be able to complete the onboarding process, scan QR codes on bikes located in the parking lot to initiate and conclude trips, and facilitate payments manually or through an auto-deduct feature. While creating this prototype, it is essential to consider various user profiles, ensuring their unique characteristics are clearly reflected in the design, associated documentation, and during the presentation.
For this prototype, you need to incorporate the following:

Smart vehicle:

- The smart vehicles(bike, bicycle or moped) can be used by users(staff, student, teacher) by registering on the application and making payment on the same.
- The vehicle can be docked in docking stations provided on campus and users will be charged according to a defined scheme. The vehicle can be used both inside and outside campus. The payment can be made using the software app.

User Account Management:

- The software product must allow the user to get onboarded. Once the user opens the app, they should be able to:
    - Create an account - You can decide on the login mechanism to be implemented

    - Upload id - You can decide on what ids should be uploaded for various kinds of users.
    - Add money to Wallet - Decide on how the wallet functionality will work with respect to the payment mechanisms, boundations like minimum balance etc.
    - Users History - Users should be able to look at their trip history

Bike reservation rates/charges:

- The payment is such that for the first x kilometers, a base rate of y rupees is taken. After that, it is at z rupees per 100 meters.
- The user books a vehicle using the application. Keep track of the current vehicle, money due, user details, etc. If a bike is not returned to the docking station within 8 hours and the bike is not renewed, a fine of 50 rupees is deducted every day.

Payment Management:

- The user should be able to make payment for the rides they've taken. Payment can be made via in-app wallet. If there is sufficient money in the wallet, the amount can either be auto deducted (if so enabled by the user before) or else, can be done manually via the app.
- If the wallet doesn't have sufficient money, money has to be added to the wallet via other payment options before proceeding for making the payment. Money can be added by existing UPI apps. (Other options are open to interpretation)
- Other options can include deducting from the salary of the user (in case of staff or professors) or adding to the fees (of students). You may add details for adding money to the wallet as per your understanding of payment systems. Your design and/or presentation must make the added details obvious.

Support, Feedback and Ratings:

- Ratings should be provided indicating the satisfaction of the availed service.
- Additionally, user must be able to provide feedback which can improve the app or can help in sustaining the current software
- Support information in the form of documentation should be provided for users' ease of travel and usability.

Trip Management:

- To move the smart bike from the parking lot, the trip must be started. Also, trips can only be started and ended at the designated parking lots of the campus. Trips can be started and ended by using the bike's QR code.
    - For starting the trip
        - Scan the QR code -> bike's details get listed in the app -> Start the Trip
    - For ending the trip:

- Park the bike in the parking lot -> End trip on app screen -> Scan QR code

You may add more details/functionalities pertaining to this. List all the assumptions you make.

Parking Lot Management:

- Track the availability of smart bikes in the parking lot, manage bike locations, and update bike statuses based on user actions.
- It should also monitor overall capacity of the parking lot, should include maintenance status (repairs if any, condition of bikes, etc.) and security features (open-ended). It should also have a data logging system facilitating system analysis and future improvements.

NOTE:

1. Make a single document containing the following:
    a. UML diagrams (both) for the problem statement.
        i. Class diagram for the entire system.
        ii. **At least one** "Object level - Sequence diagram". You are free to choose 2-3 scenarios and implement the same.
    b. Proper explanation for all the considered classes (attributes and functions), inheritance(s), relationship(s), cardinalities. **The explanation must be concise and clear.** Irrelevant explanations will get a penalty.
2. List all your assumptions in the document.
3. **No deadline extensions will be entertained for the assignment.**

Submission Guidelines:

1. The deadline for the submission of Part-I is **Friday, January 19th**
2. Make sure your name and roll number is listed on top of the submitted sheet.
3. The submission format should be **Assignment1_<Roll No>.pdf**.
   Please follow the format strictly otherwise a penalty may be incurred.