About this Project

Purpose

The purpose of this project is to develop an automated smart parking system that integrates IoT technology, license plate recognition, and a mobile app to provide a seamless parking experience for users. The system allows users to view available parking spots, reserve them, and control barriers remotely through the app. It also enables users to park without a reservation, automatically detecting their entry, tracking the time spent in the parking lot, and notifying them to pay upon exit. By utilizing a combination of sensors and cameras, the system accurately manages parking availability, reduces power consumption, and simplifies the billing process, ensuring an efficient and user-friendly parking solution.

In busy urban areas like Colombo, finding parking can be a frustrating and time-consuming experience. A study by the University of Moratuwa found that drivers in Colombo spend an average of 20 minutes searching for a parking spot during peak hours. For regular commuters, this adds up to about 80 hours per year—more than three full days spent just looking for parking! This smart parking system addresses this challenge by offering real-time parking availability and reservation features, significantly reducing the time spent searching for parking and improving the overall parking experience.

Workflow

1. **User Registration and Login**:
    - Users download the mobile app and create an account to access the parking system.
    - After logging in, they can view available parking spots in real-time.
2. **Viewing Available Spots**:
    - The mobile app sends a request to the cloud server via the Wi-Fi router to retrieve the current status of parking spots.
    - The cloud server queries the database and returns the list of available spots to the app.
3. **Reserving a Parking Spot**:
    - Users can select a parking spot from the available options and reserve it through the app.
    - The app sends a reservation request to the cloud server, which updates the database to mark the selected spot as reserved.
    - The cloud server sends a confirmation back to the app.
4. **Controlling the Barrier**:

    - Once the reservation is confirmed, users can control the barrier at their reserved parking spot using the app.
    - The app sends a command to the cloud server, which communicates with the parking spot’s microcontroller to open the barrier for the user.

5. **Vehicle Detection**:
    - At each parking spot, an IR sensor detects the presence of a vehicle.
    - If a vehicle parks in a spot, the microcontroller sends this information to the cloud server, updating the status of the spot to "occupied."
6. **Entrance and Exit Management**:
    - At the entrance and exit points of the parking lot, a license plate camera captures the license plates of vehicles entering and exiting.
    - The camera sends the captured plate data to the cloud server, which tracks the duration of parking for each vehicle.
7. **Handling Unreserved Vehicles**:
    - If a vehicle enters the parking lot without a reservation, it can park in an available spot.
    - The IR sensor detects the vehicle's presence, and the cloud server updates the database to mark the spot as occupied.
    - Once the unreserved vehicle leaves, the system calculates the parking duration and sends a notification to the user's app to process payment.
8. **Payment Processing**:
    - Users receive a notification through the app with the amount due based on their parking duration.
    - The app processes the payment, and the cloud server updates the user’s account accordingly.
9. **Monitoring and Reporting**:
    - The cloud server continuously monitors the status of all parking spots, updating the database and sending real-time information to the mobile app.

Components

Sensors

1. **Ultrasonic Sensors (Entry/Exit Gates)**
    - **Purpose**: Detect vehicles approaching the parking lot gates.
    - **How it Works**: These sensors are installed at the entrance and exit gates to detect when a vehicle is near. When a car approaches, the ulltrasonic sensor triggers the camera to turn on, saving power by only activating the camera when necessary.
    - **Benefit**: Reduces power consumption by activating the camera only when vehicles are present, ensuring efficient use of resources.
2. **Cameras with License Plate Recognition (Entry/Exit Gates)**

    - **Purpose**: Automatically recognize and track vehicles entering and exiting the parking lot.
    - **How it Works**: Cameras at the gates capture images of vehicle license plates. The license plate recognition software identifies and logs the vehicle's entry or exit time. For users without reservations, this data is used to calculate the duration of their stay and trigger a payment notification through the app.
    - **Benefit**: Automates vehicle tracking and ensures accurate billing for users who enter the parking lot without a reservation.
3. **Infrared (IR) Sensors (Parking Spots)**
    - **Purpose**: Detect whether a parking spot is occupied.
    - **How it Works**: IR sensors are installed at each parking spot to detect the presence of a vehicle. When a car parks in a spot, the sensor marks the spot as unavailable in the system. This happens whether the user has reserved the spot in advance or not.
    - **Benefit**: Provides real-time monitoring of parking spot occupancy, ensuring that the app reflects accurate availability.

Actuators

1. **Barrier Gate Motors (Parking Spots)**
    - **Purpose**: Control the raising and lowering of barriers at each parking spot.
    - **How it Works**: Each parking spot has a barrier controlled by a motor (usually a servo or DC motor). When a user reserves a spot via the app, the system allows them to raise and lower the barrier from the app to access the reserved spot.
    - **Benefit**: Provides secure, app-based control over parking spots, ensuring only reserved users can access them.
2. **LEDs (Parking Spots)**
    - **Purpose**: Display the parking availability for users.
    - **How it Works**: LEDs can be installed near the parking spots show the status of each spot (available, reserved, or occupied with a respective color)
    - **Benefit**: Improves user experience by providing clear visual indicators of parking spot status.

Communication Method

a. Mobile App to Server Communication:

- **Type**: HTTP/HTTPS requests
- **Purpose**: The mobile app communicates with a backend server to retrieve data about available parking spots, manage reservations, and handle payments.
- **How It Works**: Users interact with the app to make reservations or view available spots. The app sends requests to the server using RESTful APIs. The server responds with updated information about parking availability and reservation statuses.

- **Security**: Using HTTPS ensures that data transmitted between the app and the server is encrypted, protecting user information and payment details.

b. Backend Server to Sensors/Actuators Communication:

- **Type**: MQTT (Message Queuing Telemetry Transport)
- **Purpose**: The server communicates with various sensors and actuators deployed throughout the parking lot, ensuring real-time updates and actions based on user interactions.
- **How It Works**: This lightweight messaging protocol is ideal for IoT applications. Sensors send messages (e.g., vehicle detected, parking spot occupied) to the server, which processes these messages and updates the system accordingly. MQTT is efficient for low-bandwidth scenarios and supports a large number of devices.
- **Benefits**: Both protocols facilitate low-latency communication, which is crucial for user satisfaction and system responsiveness.

c. Mobile App to User Notifications:

- **Type**: Push Notifications
- **Purpose**: Notify users about parking status, payment reminders, and other important updates.
- **How It Works**: Once a user reserves a spot or parks without a reservation, the server triggers push notifications to the mobile app. This is typically achieved using services like Firebase Cloud Messaging (FCM) or Apple Push Notification Service (APNs).
- **Benefit**: Keeps users informed in real time about their parking situation and any necessary actions they need to take (e.g., payment notifications).

Mobile App Development

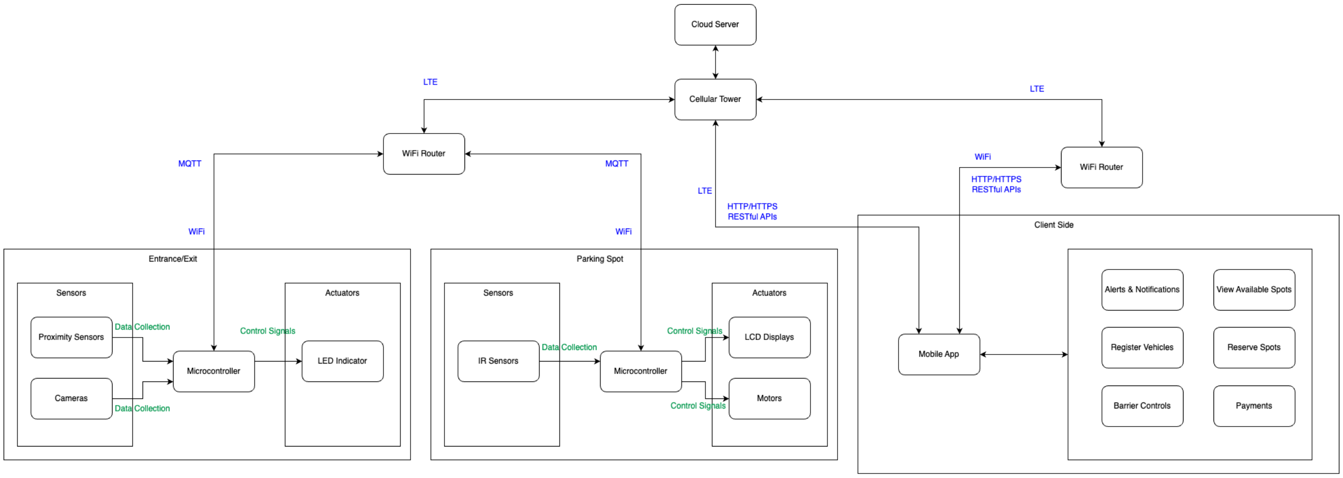
1. **Swift**:
    - **Overview**: Swift is a powerful and intuitive programming language developed by Apple for building apps for iOS, macOS, watchOS, and tvOS.
    - **Advantages**:
        - **Performance**: Swift is designed for high performance and offers modern language features, resulting in fast execution and efficient memory management.
        - **Safety**: Swift's syntax encourages safe programming practices, helping to eliminate common programming errors and enhancing overall code quality.
        - **Rich Ecosystem**: Swift has a robust ecosystem, including numerous libraries and frameworks that streamline development processes.
2. **SwiftUI**:
    - **Overview**: SwiftUI is a declarative framework for building user interfaces across all Apple platforms.

   - **Advantages**:
     - **Declarative Syntax**: SwiftUI allows developers to describe the UI in a simple, readable manner, making it easier to create and manage complex interfaces.
     - **Live Preview**: SwiftUI offers live previews of the UI, enabling developers to see changes in real-time as they code, which speeds up the design process.
     - **Cross-Platform Compatibility**: With SwiftUI, you can create apps for both iOS and macOS using a single codebase, ensuring consistency across platforms.

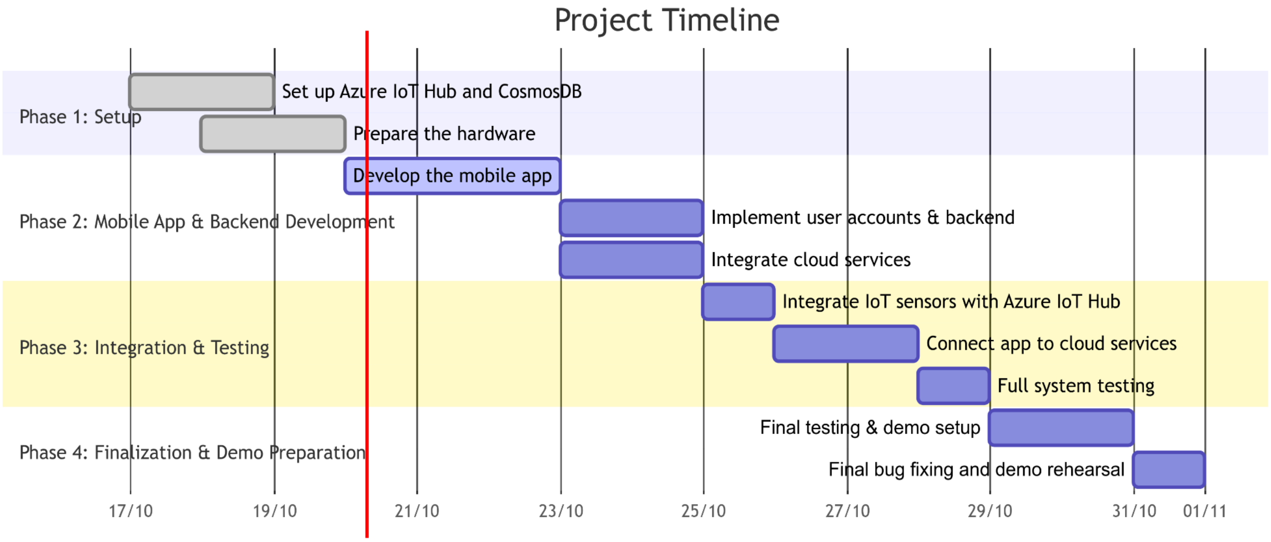
3. **Skip**:
   - **Overview**: Skip is a cross-platform development tool that allows developers to create both Android and iOS apps using Swift and SwiftUI code.
   - **Advantages**:
     - **Single Codebase**: Skip enables the use of a single codebase for both iOS and Android, reducing development time and effort while maintaining feature parity across platforms.
     - **Native Performance**: By leveraging Swift and SwiftUI, apps built with Skip deliver native performance and a seamless user experience on both platforms.
     - **Rapid Development**: Skip simplifies the development process, allowing developers to focus on functionality and design rather than platform-specific requirements.

Using **Swift**, **SwiftUI**, and **Skip** for the app provides a powerful combination that enhances performance, safety, and cross-platform compatibility. These technologies not only streamline the development process but also ensure that the app offers a rich user experience on both iOS and Android devices. By adopting this modern tech stack, you can create a robust, user-friendly application that meets the needs of your users while leveraging the benefits of a unified development approach.

Block Diagram



Timeline


Project Timeline
Phase 1: Setup
Set up Azure IoT Hub and CosmosDB
Prepare the hardware
Phase 2: Mobile App & Backend Development
Develop the mobile app
Implement user accounts & backend
Integrate cloud services
Phase 3: Integration & Testing
Integrate IoT sensors with Azure IoT Hub
Connect app to cloud services
Full system testing
Phase 4: Finalization & Demo Preparation
Final testing & demo setup
Final bug fixing and demo rehearsal
17/10
19/10
21/10
23/10
25/10
27/10
29/10
31/10
01/11