Project Proposal

1. Project Title

SmartPark: A Cloud-Based University Parking Reservation & Analytics System

Project Type: Web Application

POC for Senior Project: NO

2. Team Members

Member 1: Nyi Min Htet - 6622132

Member 2: Zaw Lin Aung- 6622137

3. Problem Statement & Motivation

Many universities experience significant parking congestion due to limited parking capacity and the absence of structured reservation systems. Students and staff often spend excessive time searching for available parking spaces, resulting in delays, frustration, and reduced productivity.

Currently, parking allocation is typically managed manually or through first-come-first-serve approaches without real-time visibility or data-driven analysis.

The SmartPark system provides real-time parking availability, online reservation management, administrative control over zones and slots, and analytical insights for decision-making.

4. Project Scope & Features

Core Functional Features

- Role-based authentication (Admin / Staff / Student)
- Secure login using JWT
- CRUD management of Parking Zones
- CRUD management of Parking Slots
- CRUD management of Reservations
- Real-time parking slot availability tracking
- Reservation creation (time-based booking)
- Reservation cancellation and updates
- Capacity validation to prevent over-booking

Analytics & Dashboard Features (Admin)

- Daily / Weekly / Monthly reservation statistics

- Most reserved parking zone (ranking analysis)
- Peak reservation hour detection
- Occupancy rate per zone
- Total active reservations overview
- Reservation trend analysis (chart visualization)

5. Data Models

Entity: User

- name
- email
- password (hashed)
- role (Admin / Staff / Student)

Operations: Create, Read, Update, Delete

Entity: ParkingZone

- zoneName
- location
- totalSlots

Operations: Create, Read, Update, Delete

Entity: ParkingSlot

- slotNumber
- zoneId (reference to ParkingZone)
- status (Available / Occupied)

Operations: Create, Read, Update, Delete

Entity: Reservation

- reservationDate
- startTime
- endTime
- userId (reference to User)
- slotId (reference to ParkingSlot)
- status (Active / Cancelled / Completed)

Operations: Create, Read, Update, Delete

6. System Architecture

The application follows a full-stack web architecture where the frontend communicates with RESTful API endpoints. The backend processes business logic and validation, while MongoDB stores structured data using references. JWT handles authentication and authorization. Nginx acts as a reverse proxy and PM2 manages the Node.js process.

7. Technology Stack

- Frontend: Next.js (React-based framework)
- Backend: Next.js RESTful API Routes
- Database: MongoDB (Self-managed on Ubuntu VM)
- Authentication: JSON Web Token (JWT)
- Deployment: Azure Virtual Machine (Ubuntu 24.04)
- Process Manager: PM2
- Reverse Proxy: Nginx

8. Expected Outcomes

- Functional cloud-hosted web application
- Fully implemented RESTful API
- Role-based access control
- Real-time reservation logic
- Analytical dashboard with aggregation queries
- Production-ready deployment environment