PHASE -4 DEVELOPMENT Part-2

OBJECTIVE:

To build the traffic information platform and mobile apps for iOS and Android, we'll need a combination of web development technologies and mobile app development tools. Here's a high-level overview of the steps to achieve this:

Traffic Information Platform (Web):

❖ Web Development:

Create a web application using HTML, CSS, and JavaScript for displaying real-time traffic information.Utilize popular web frameworks like React, Angular, or Vue.js for front-end development to create an interactive and responsive user interface.

❖ Real-Time Traffic Data:

Integrate with your IoT-based traffic management system to receive real-time traffic data.Use JavaScript libraries or frameworks like Axios or Fetch API to make API requests to your backend server for data updates.

❖ Data Presentation:

Display traffic data using interactive maps, charts, and tables.Provide information on traffic flow, congestion, road closures, and any accidents or incidents.Implement features like zoom, pan, and filter options for users to explore the data.

❖ User Authentication:

Implement user authentication and authorization to allow registered users to personalize their experience.Consider using technologies like JWT (JSON Web Tokens) for secure authentication.

❖ Cross-Platform Development:

Consider using cross-platform app development frameworks like React Native or Flutter to build mobile apps for both iOS and Android simultaneously, saving time and effort.

❖ App Features:

Create mobile apps that provide users with access to real-time traffic updates and route recommendations.Implement user-friendly interfaces with features such as:

- Real-time traffic maps with markers and overlays.
- GPS-based navigation and route planning.
- Alerts for traffic incidents and congestion.
- User preferences and saved routes.

❖ API Integration:

Utilize the same APIs used for the web platform to fetch real-time traffic data for mobile apps.Ensure that the mobile apps can display this data seamlessly and in a mobile-friendly format.

❖ Push Notifications:

Implement push notifications to alert users about critical traffic events, such as accidents or road closures.

❖ Testing and Deployment:

Thoroughly test the mobile apps on iOS and Android devices to ensure compatibility and functionality.Deploy the apps to their respective app stores (Apple App Store and Google Play Store).

Html code:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Traffic Information</title>
    <link rel="stylesheet" type="text/css" href="styles.css" />
  </head>
  <body>
    <header>
      <h1>Real-Time Traffic Information</h1>
    </header>
    <div class="traffic-info">
      <div class="traffic-light-status">
        <h2>Traffic Light Status</h2>
        <p id="red-light-status">Red Light: Off</p>
        <p id="yellow-light-status">Yellow Light: Off</p>
        <p id="green-light-status">Green Light: Off</p>
      </div>
      <div class="pedestrian-crossing">
        <h2>Pedestrian Crossing</h2>
        <p id="pedestrian-status">Pedestrian: Waiting</p>
      </div>
      <div class="ambulance-alert">
        <h2>Ambulance Alert</h2>
        <p id="ambulance-status">Ambulance: No Alert</p>
      </div>
    </div>
    <script src="script.js"></script>
  </body>
</html>
```

Css code:

```css
body {
    font-family: Arial, sans-serif;
    background-color: #f3f3f3;
    margin: 0;
    padding: 0;
}

header {
    background-color: #333;
    color: #fff;
    text-align: center;
    padding: 10px;
}

.traffic-info {
    max-width: 800px;
    margin:auto;
    background-color: #fff;
    border: 1px solid #ccc;
    padding: 40px;
    border-radius: 5px;
    display: flex;
    flex-direction: column;
    margin-top: 40px;
}

.traffic-light-status, .pedestrian-crossing{
    margin-bottom: 35px;
    padding: 0 20px 20px 20px;
    background-color: #f9f9f9;
    border: 1px solid #ddd;
    border-radius: 5px;
}

.ambulance-alert{
    padding: 0 20px 20px 20px;
    background-color: #f9f9f9;
    border: 1px solid #ddd;
    border-radius: 5px;
}

#red-light-status, #yellow-light-status, #green-light-status,
#pedestrian-status, #ambulance-status {
    margin: 0;
}
```

```
#red-light-status.on, #yellow-light-status.on, #green-light-status.on,
#pedestrian-status.on, #ambulance-status.on {
    color: green;
}

#red-light-status.off, #yellow-light-status.off, #green-light-
status.off, #pedestrian-status.off, #ambulance-status.off {
    color: red;
}
```

Js code:

```
function updateTrafficStatus() {
    const redLightStatus = document.getElementById("red-light-status");
    const yellowLightStatus = document.getElementById("yellow-light-
status");
    const greenLightStatus = document.getElementById("green-light-
status");
    const pedestrianStatus = document.getElementById("pedestrian-
status");
    const ambulanceStatus = document.getElementById("ambulance-status");
    const isRedOn = true; // Replace with actual data
    const isYellowOn = false; // Replace with actual data
    const isGreenOn = true; // Replace with actual data
    const isPedestrianWaiting = true; // Replace with actual data
    const isAmbulanceAlert = false; // Replace with actual data
    // Update traffic light status
    redLightStatus.textContent = `Red Light: ${isRedOn ? "On" : "Off"}`;
    yellowLightStatus.textContent = `Yellow Light: ${isYellowOn ? "On" :
"Off"}`;
    greenLightStatus.textContent = `Green Light: ${isGreenOn ? "On" :
"Off"}`;
    // Update pedestrian crossing status
    pedestrianStatus.textContent = `Pedestrian: ${isPedestrianWaiting ?
"Waiting" : "Crossing"}`;
    // Update ambulance alert status
    ambulanceStatus.textContent = `Ambulance: ${isAmbulanceAlert ?
"Alert" : "No Alert"}`;
    // Add or remove 'on' class to highlight active status
    isRedOn ?
(redLightStatus.classList.add("on"),redLightStatus.classList.remove("of
f")) :
(redLightStatus.classList.remove("on"),redLightStatus.classList.add("of
f"));
    isYellowOn ?
(yellowLightStatus.classList.add("on"),yellowLightStatus.classList.remo
ve("off")) :
```

```
(yellowLightStatus.classList.remove("on"),yellowLightStatus.classList.a
dd("off"));
    isGreenOn ?
(greenLightStatus.classList.add("on"),greenLightStatus.classList.remove
("off")) :
(greenLightStatus.classList.remove("on"),greenLightStatus.classList.add
("off"));
    isPedestrianWaiting ?
(pedestrianStatus.classList.add("on"),pedestrianStatus.classList.remove
("off")) :
(pedestrianStatus.classList.remove("on"),pedestrianStatus.classList.add
("off"));
    isAmbulanceAlert ?
(ambulanceStatus.classList.add("on"),ambulanceStatus.classList.remove("
off")) :
(ambulanceStatus.classList.remove("on"),ambulanceStatus.classList.add("
off"));
}
// Initial update
updateTrafficStatus();
// Simulated real-time updates
setInterval(() => {
    // Fetch and update data from Python IoT code
   updateTrafficStatus();
}, 5000); // Update every 5 seconds (adjust as needed)
```