SPNS - IOT project Web application

Work Plan for creating the web application:

# **Design a map:**

Parking spots, parking sections, elevators locations, level locations, exits , etc…  
Options: [Leaflet](https://leafletjs.com/) (open-source).

* Design a multi-level parking where each level is divided into several sections and parking spaces.
* Save as an image and use GIMP to create a compatible map for Leaflet.
* Use the images to make for the parking lot floors and assign them in Leaflet as layers or overlays.
* Assign roads, drivable routes, parking spaces, exits, etc…
* Make the zoom like functionality to act as a way to traverse between the parking lot’s floors.

# **Backend development:**

* **Database:** store parking lot information (same as above).
* A map variable is created for the possible locations the driver can be at.   
  have keys - the string that represents the location,   
  and values - the coordinates on the map.  
  \**Example:* [“F0\_IN, [ 0, [140, 150]]] can point to floor 0 entry point and its floor and coordinates.
* **API:** build an API handle requests from frontend and interact with databases.
  + **Azure IOT HUB:**
    - Assign raspberry-PI as an IOT device.
  + **Azure Event HUB:**
  + **Azure Storage**:
    - Section occupancy - created a table with the following schema:
      * License plate
      * Floor
      * Section
      * Timestamp
      * Unique URL
    - Drivers inside the parking lot.
  + **Azure Functions:**
    - Insert vehicle
    - Update vehicle
    - Remove vehicle
    - Query for occupancy
    - Negotiate
* **Interaction between IOT, web app and Azure:**
  + IOT device web application interaction:
    - QR code scan on IOT screen (front gate camera)
  + IOT Device Azure interaction:
    - Car lot entry + create url (front gate camera + create a new page instance for that user)
    - Car lot exit (Camera + delete the user’s page)
    - Car section location updates (Cameras)
    - Time interval for updating occupancy data and send signalR
  + Web Application Azure interaction:
    - Web app page initialization (front gate camera)
    - Car section location updates (Cameras)
    - Section occupancy (signalR information triggered by IOT device)
* **Users:**
  + **Html token:** read a token given on in the url with the license plate number to determine the vehicle user and accept updates accordingly.  
    \*Example: “https://spnswebapp.azurewebsites.net/?licenseplate=12345678”

# **Frontend development:**

* **HTML/CSS:** design the user interface using HTML/CSS.
* **JavaScript:** handle user interactions and send requests to the backend API using JavaScript:
  + Implement main features for the user interaction:
    - Controls for Changing parking lot floor maps
    - Controls and zoom for the map
    - Button: save location (local cookies storage)
    - Click on the map for selecting the navigation point (or by button).
    - Button: change/fix number plate
    - Button: navigate to low occupancy area
    - Button: navigate to nearest exit
  + Extra buttons for auxiliary sites/ apps:
    - Google maps
    - Waze
    - Pango
* **Map integration:** use the map library from Leaflet to display to the user.
  + User shown on map is static and changes according to cameras (optional to make user able to manually change his location)
* **Location tracking:** set up the mechanism to receive updates from the IOT device.
  + **Azure SignalR:**
    - Azure functions that will react to signalR activated through Azure IOT HUB:
      * Car entered the parking lot.
      * Car went through a parking section.
      * Car went through an exit.

# **Backend-frontend communication:**

* **Flow of communication**:

1. Camera picks up a license plate at one of the entry point
2. The IOT device uses the number captured to create a url for that number and asks azure to create it.
3. The IOT device creates a qr code that will redirect a user to the url created and show it on its screen.
4. Driver will use his phone’s camera to capture the qr code and enter the web application.
5. Everytime the driver runs into a camera inside the parking lot the camera will capture the number plate and the iot device will send it to azure where he saw the vehicle.
6. Azure will update its tables and will send a SignalR to the url that has this drivers plate number.
7. The map in the web application will update according to the new SignalR message.
8. When the driver exits the parking lot through one of the exit points, the IOT device will send this information to azure which will update its tables and remove this vehicle from them as well.

* Create API endpoints to handle requests from the IOT device to the user.
* Develop frontend logic to send the location to the backend.

# **User experience:**

* **Display the user's location on the map dynamically with updates.**
* **Implement other app features**:
  + Find a parking spot or a location - navigate to a low occupancy section in the parking lot, or navigate to a wanted location.
  + Find the nearest exit point - navigate to the closest exit gate from the user’s location.
  + Save parking location - be able to save your parking location on the web application.
  + Fix license plate number - show the number and have a button for setting/ correcting it.
* Add parking payment solution Pango (App and URL)
* Add outside map view for GoogleMaps and Waze (App and URL)

# **Testing, Deployment, Scale:**

* Test on working IOT device.
* Test multiple user connections behavior.