Smart parking using IoT

**Prologue:**

The integration of Google Maps and MIT App Inventor enables the creation of a Smart Parking App that offers real-time parking availability information. This app addresses urban parking challenges, alleviating traffic congestion and environmental issues. By leveraging MIT App Inventor's user-friendly interface and Google Maps' geospatial features, users can easily find nearby parking slots, enhancing their urban parking experience. This document outlines the development of this Smart Parking App, which optimizes parking guidance using Google Maps.

**App overview:**

* The Smart Parking App, created with MIT App Inventor and Google Maps, simplifies urban parking. It addresses parking challenges by offering real-time slot availability information, reducing congestion, and minimizing environmental impact. Users can locate and reserve parking slots, enhancing their parking experience.
* The app leverages the capabilities of Google Maps to provide real-time information about parking slot availability in nearby locations. Users can easily locate and reserve parking slots, thereby streamlining their parking experience.
* The primary objective of this app is to improve convenience for drivers and optimize parking resource management for parking facility operators.

**Key Features of the Application:**

* The Smart Parking App developed using Google Maps in MIT App Inventor offers a range of user-centric features designed to enhance the urban parking experience.
* **Real-time Parking Availability**:The Smart Parking App offers users up-to-the-minute data on available parking slots, reducing the time and stress associated with finding a parking spot.
* **Google Maps Integration:** By leveraging Google Maps, the app delivers an intuitive and interactive interface, allowing users to view parking locations and navigate to their chosen slots effortlessly.
* Utilizes the device's location services to identify the user's current position and the closest available parking slots.
* Helps users navigate to their selected parking location.
* Permits users to search for parking slots based on specific criteria such as location, price, and availability.
* Filters and sorts results for user convenience.

**App Development:**

**Developing a smart parking app using MIT App Inventor involves several steps. Here's a high-level step-by-step process:**

**Step 1: Design the User Interface (UI):**

* Define the specific features and functionalities your smart parking app will have.
* Identify the IoT devices and sensors you'll be using for parking detection and data collection.
* Visit the MIT App Inventor website and create an account if you don't already have one.
* Use MIT App Inventor's drag-and-drop interface to design the app's user interface.
* Include buttons, labels, and input fields for user interaction.
* Design screens for different app sections, such as parking availability, booking, and payment.

**Step 2: Connection to IoT Devices**

* Set up the communication between our app and IoT devices using Raspberry Pi pico W in wokwi and also other suitable connectivity options with the IoT sensors that detect parking availability.
* Log in to Wokwi account or sign up for one if you don't have an account.
* Create a new project in Wokwi.< Selected the Raspberry Pi Pico as your microcontroller.
* Add virtual components for sensors or actuators you want to use in your smart parking system. For example, you can add virtual ultrasonic distance sensors to simulate parking spot occupancy detection..

**Step 3: Cloud Data processing**

* Write the Python code for your Raspberry Pi Pico that interacts with the virtual sensors and sends data to ThingSpeak. Use the urequests library to make HTTP requests to ThingSpeak.
* In your Python code, include the ThingSpeak channel ID and write API key for data transmission.
* To Set Up the ThingSpeak Channel,Log in to ThingSpeak and create a new channel for your smart parking system. Note the channel ID and write API key, which you'll use in your Python code.

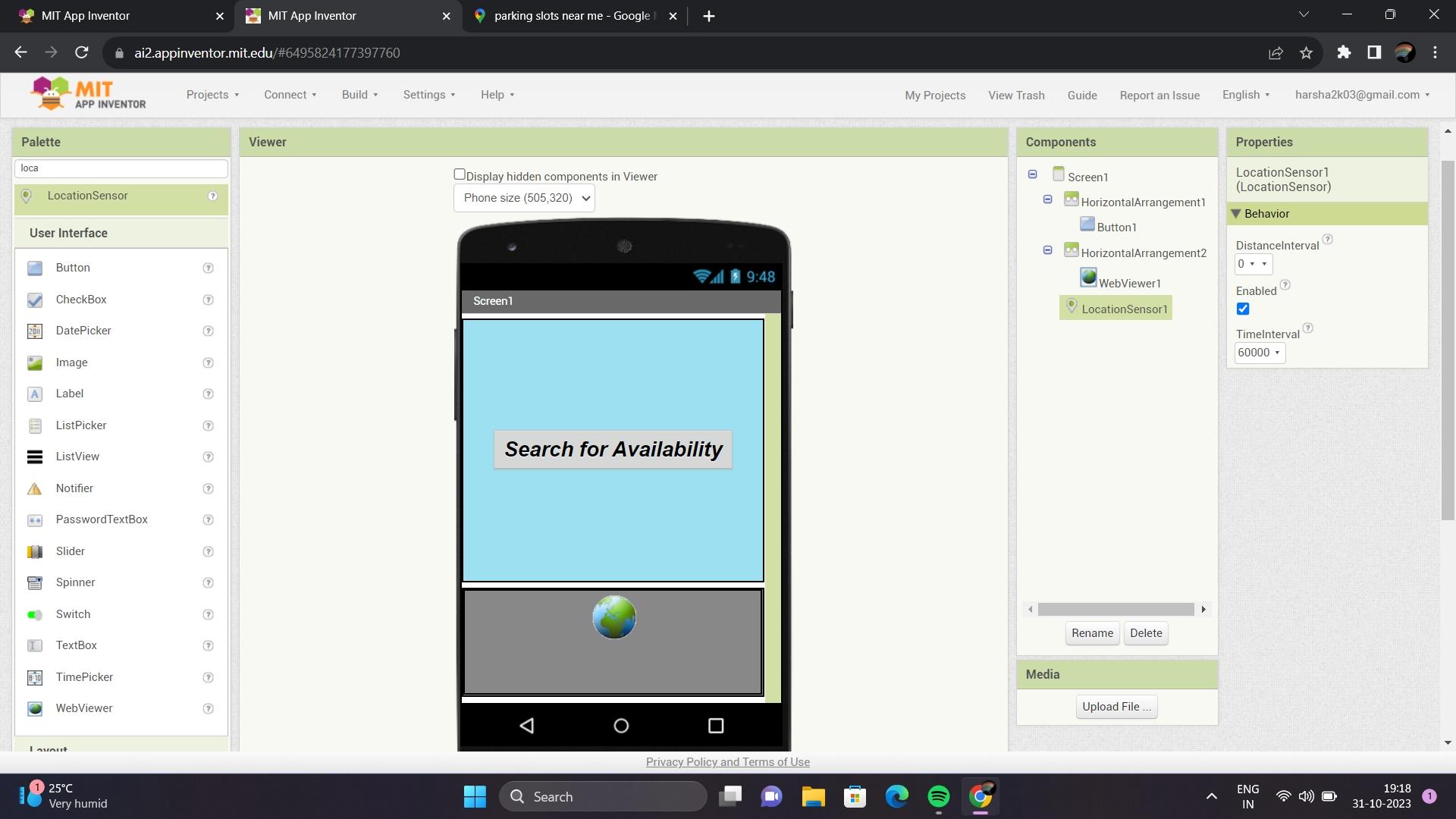
**Step 4: Run the IoT Simulation**

* Run the simulation in Wokwi to test the IoT components. Verify that data is being sent to ThingSpeak and monitor the channel for updates.

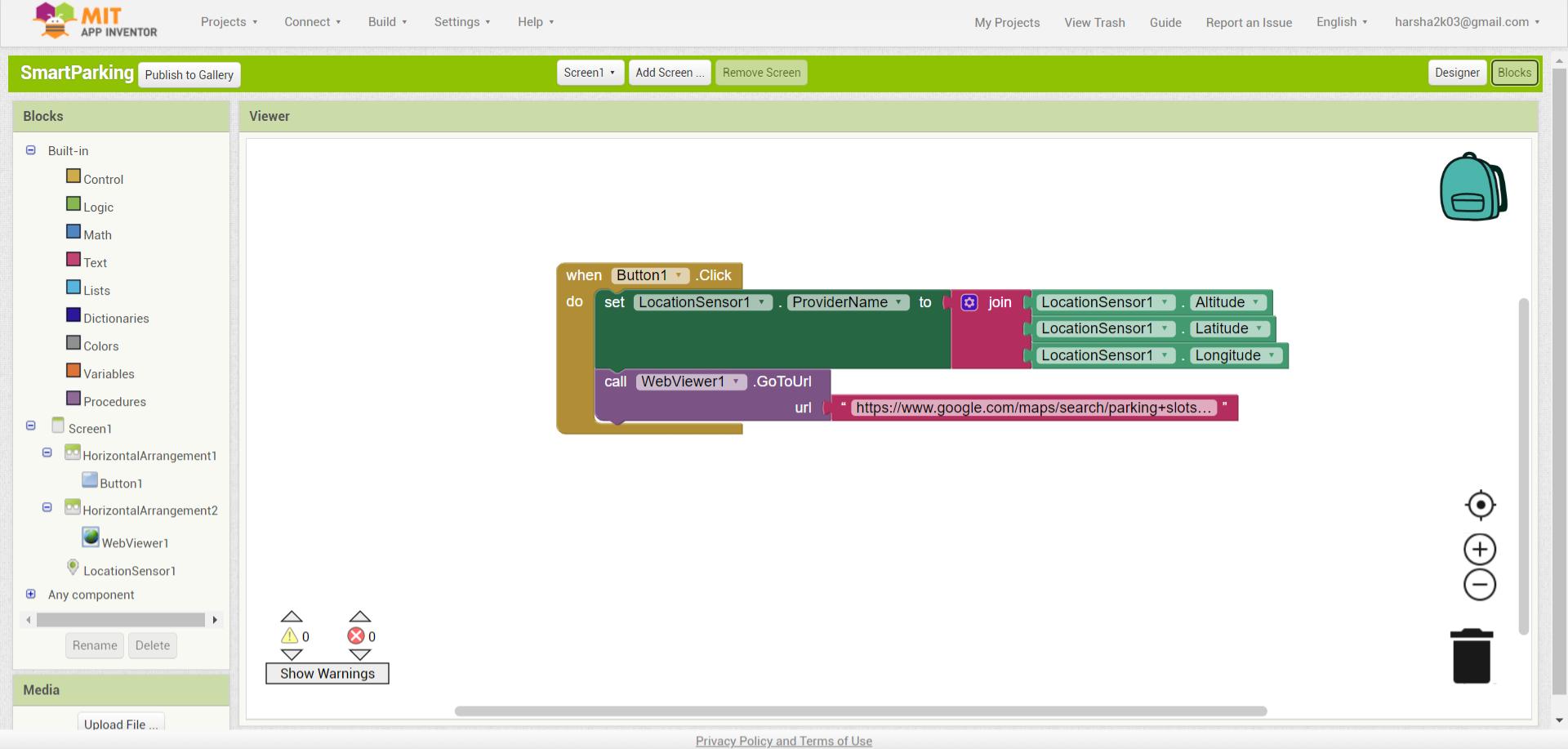
**Step 5: Create the MIT App Inventor App**

* Go to MIT App Inventor(**https://appinventor.mit.edu/)**
* Create a new project in MIT App Inventor.
* Design the user interface for your smart parking app. You might want to include features like displaying parking spot availability and a map.
* Write the app's logic using MIT App Inventor's blocks-based interface. Use the Web component to retrieve data from your ThingSpeak channel.

**Sample Images:**

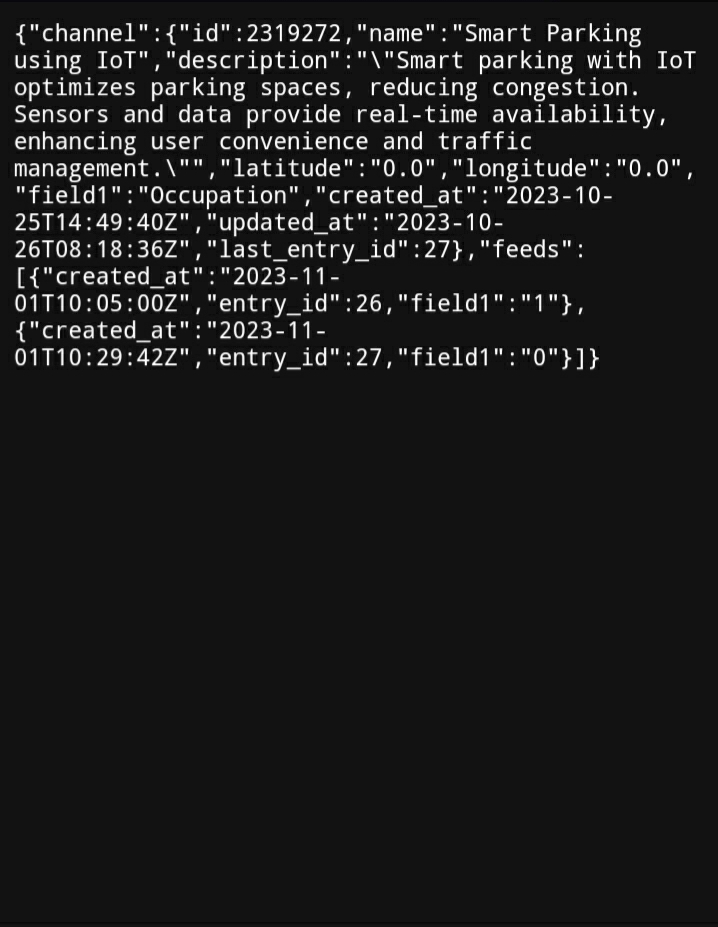
Designer:

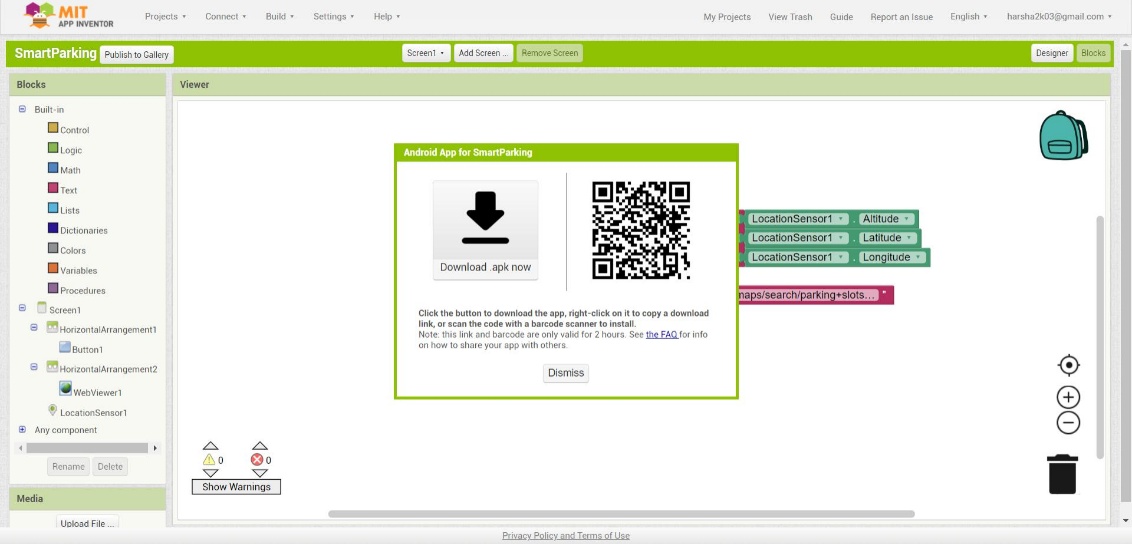
Block:



**Step 6: Connect the App to ThingSpeak**

* In MIT App Inventor, use the Web component to make a GET request to retrieve data from your ThingSpeak channel. Provide the ThingSpeak API endpoint for your channel.
* Parse the response data to extract relevant information such as parking spot availability or occupancy status.
* Update the app's user interface to reflect the parking status based on the data retrieved.

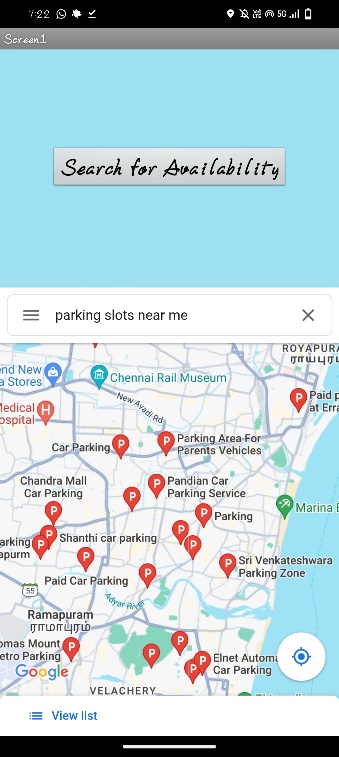


**Step 7: Test and Deploy**

* Make any necessary adjustments and refinements to the app, test your MIT App Inventor app on an Android device or emulator to ensure it works correctly by downloading the app that has been created.
* Once you're satisfied with the app's functionality, you can package it and distribute it to others or publish it on the Google Play Store.

**Final Output:**

**Homepage:**

**Map view (Google Maps):**

**Conclusion:**

The Smart Parking App is not only a valuable tool for drivers seeking efficient parking solutions but also a testament to the potential of technology in creating smarter and more sustainable urban environments. By combining MIT App Inventor's accessibility and Google Maps integration, this project offers a user-friendly solution to a common urban challenge, with the promise of continued improvement and expansion in the future.