**Smart Gate Access Control System**

**1. Project Overview**

The Smart Gate Access Control System is an IoT-based prototype developed using a Raspberry Pi 3B+ designed to automate and secure entry at residential and commercial gates. It integrates stepper-motor-based gate control, license plate recognition through the Google Cloud Vision API, motion-based automation, and OTP-based guest access. This system offers multiple modes of access control and real-time interaction via MQTT and Firebase, supported by a functional mobile app interface.

This DIY platform simulates a real-world smart gate scenario using low-cost components and open-source software to make the technology accessible and replicable. A working miniature gate model was designed using acrylic sheets, hinges, and a stepper motor mounted to demonstrate live gate automation during the final presentation.

**2. Primary Functionalities**

* **Automated Gate Control:**  
  When a valid license plate or OTP is detected, the system opens the gate automatically. The gate remains open as long as motion is detected and closes automatically once no activity is observed.
* **License Plate Recognition:**  
  A Pi Camera captures an image of the vehicle, processes it using OpenCV, and sends it to the Google Cloud Vision API for OCR. The extracted plate number is checked against registered family members in Firebase.
* **OTP Access for Guests:**  
  The admin can generate one-time passwords for guest access. Guests receive the OTP via the app and enter it at the gate. The system validates the OTP and opens the gate if correct and unexpired.
* **Motion Sensor Integration:**  
  A PIR sensor continuously monitors motion. This ensures that the gate remains open when someone is in the vicinity and closes it once the area is clear.
* **Firebase Realtime Database Integration:**  
  Stores user data, license plates, guest entries, and activity logs. Weekly reports are generated and uploaded to Firebase for analytics.
* **MQTT-Based Communication:**  
  Commands and status updates are transmitted using the HiveMQ cloud broker. MQTT ensures lightweight, reliable two-way communication between the Raspberry Pi and the mobile app.
* **Mobile App with Role-Based Access:**  
  Built with Flutter, the app includes Admin, Family, and Guest dashboards. Admins manage users, generate OTPs, view logs, and access live stream. Family members receive access via plate recognition, and guests via OTP.
* **Google Assistant Integration (Admin Feature):**  
  The admin can open the gate using voice commands linked to a Google Assistant action.
* **Weekly Reporting:**  
  Entry logs are analyzed to generate a JSON report detailing the number of entries, access methods, time slots, and user-specific stats.

**3. Project Management Process**

* **Planning:**  
  Gathered requirements, selected hardware components (Pi 3B+, Grove sensors, stepper motor), defined user roles, and brainstormed automation flow.
* **Design:**  
  Designed gate simulation using acrylic materials, defined wiring layout on a breadboard, planned Firebase data structure, and created system architecture diagrams.
* **Implementation:**  
  Developed modular Python classes: StepperMotorManager, OTPManager, FirebaseManager, PlateRecognizer, and MQTTClient. Flutter front end included role-based dashboards and real-time control features.
* **Testing & Debugging:**  
  Verified GPIO responsiveness, motor torque control, image recognition accuracy, MQTT stability, and Firebase communication.
* **Integration:**  
  Merged MQTT logic with mobile app input, integrated Firebase with license verification and OTPs, and deployed camera streaming using mjpg-streamer.
* **DIY Platform Fabrication:**  
  Built a physical gate model using laser-cut acrylic, foamboard base, 3D-printed hinges, and cable management for a clean look.

**4. Team Roles & Responsibilities**

* **Elio Nader (Team Lead):**
  + Core Python developer for sensor logic and gate control
  + Plate recognition and Google Cloud integration
  + MQTT protocol handler
  + Firebase setup and data structure
  + DIY physical model design and presentation logic
* **Georges Haddad:**
  + Wiring, power delivery, and breadboard assembly
  + Sensor and motor testing
  + Flutter UI design and mobile dashboard
  + Video editing and project documentation

**6. Deliverables**

* **Code:**  
  Full Python backend (modular MQTT-based sensor control).
* **README File:**  
  Setup instructions, wiring guide, MQTT topics, installation steps, and Google/Firebase configuration.
* **Pitch Video:**  
  2-minute summary covering features, live gate simulation, mobile interface, and plate recognition demo.

**7. Conclusion**

The Smart Gate Access Control System is a well-integrated, full-stack IoT solution showcasing practical applications of embedded systems, cloud APIs, and automation. It blends security, convenience, and scalability while promoting the use of open-source tools and low-cost components.

The successful execution of this project demonstrates not only technical competence in software and hardware but also project management, teamwork, and presentation skills vital for real-world engineering deployment.