SURGE System — Assignment Explanation and Technical Roadmap

1. Assignment Overview (Plain-Language Explanation)

The goal of the SURGE project is to design and prototype a **digital, app-free system** that manages passenger congestion in an airport **before bottlenecks form**, rather than reacting after congestion occurs.

Instead of installing new hardware or requiring passengers to download an app, SURGE operates by issuing a **temporary, anonymous digital identity (SURGE ID)** that is represented as a QR code. This QR code can be scanned at key movement points throughout the airport. Each scan generates real-time movement data that is used to:

- Measure congestion and dwell time
- Create live heat maps of airport zones
- Control passenger flow through virtual queues
- Provide real-time guidance to passengers
- Support staff and operational decision-making

The system functions as a **coordination layer only**. It does not collect personal data, does not track GPS location, and does not replace existing airport authorities.

—

2. Official SURGE Zones (Defined by the Document)

The SURGE documentation explicitly defines the points at which a SURGE ID QR code may be scanned. These scan locations represent the **official operational zones** of the system.

Movement Zones

- Terminal entry
- Security access
- Customs corridors
- Boarding gates
- Transfer points
- Amenity access (food, seating, washrooms, retail)

Each zone represents a controlled checkpoint where:

- A QR scan indicates entry into the zone
- Dwell time is calculated using timestamps

- Congestion is inferred from scan frequency and duration

These zones are the **primary inputs** to all congestion intelligence, routing logic, and staffing recommendations in the SURGE system.

—

3. Easiest Possible Tech Stack (Python-First)

To minimize complexity while still meeting the requirements of the assignment, the following tech stack is recommended. This stack prioritizes **clarity, speed of development, and explainability**.

Backend (Core System)

- Python
- FastAPI (REST + WebSockets)
- UUID-based SURGE IDs

Real-Time State & Caching

- Redis
- TTL-based expiration for SURGE IDs
- Zone counters and dwell tracking

Data Processing & Intelligence

- Python
- Pandas / NumPy
- Rule-based congestion scoring

Frontend (Browser-Based, No App)

- Minimal HTML/CSS
- Optional React (not required for MVP)
- QR scan redirects via URL parameters

Visualization

- Chart.js or similar lightweight charting
- Simple heat-map representations

Infrastructure (Optional)

- Docker
- Local or cloud deployment

—

4. Phase-Based Project Roadmap

Phase 0 — System Understanding & Planning

Objective

Establish clear boundaries and definitions before implementation.

Tasks

- Define all SURGE zones
- Define what a scan event contains (ID, zone, timestamp)
- Decide MVP feature set

Deliverables

- System diagram
- Zone definitions
- Data flow outline

—

Phase 1 — SURGE ID Generation

Objective

Create anonymous, temporary digital identities.

Implementation

- Generate SURGE ID using UUID
- Store ID in Redis with TTL
- Encode SURGE ID into QR code

Key Principle

SURGE IDs must be:

- Anonymous
- Time-limited
- Automatically expired

—

Phase 2 — QR Scan Ingestion

Objective

Convert scans into structured movement data.

Implementation

- Create scan endpoint (FastAPI)
- Accept zone and SURGE ID
- Timestamp every scan
- Store zone entry in Redis

Each scan represents:

```
(SURGE_ID, Zone, Timestamp)
```

—

Phase 3 — Dwell Time & Heat Mapping

Objective

Measure congestion using scan behavior.

Core Logic

```
congestion_score = scan_rate × average_dwell_time
```

Implementation

- Track dwell time per zone
- Aggregate scans over rolling windows
- Compute congestion scores

—

Phase 4 — Passenger Guidance

Objective

Actively influence passenger movement.

Implementation

- Browser-based passenger page
- Show wait times
- Recommend hold or proceed
- Suggest low-congestion amenities

—

Phase 5 — Operations Dashboard

Objective

Provide staff with real-time visibility.

Implementation

- Live zone congestion display
- Manual override controls
- Alert thresholds

—

Phase 6 — Workforce Intelligence (Stretch Goal)

Objective

Align staffing with passenger demand.

Implementation

- Generate staffing demand scores
- Recommend redeployment by zone
- Require supervisor approval

—

Phase 7 — Privacy, Hardening & Finalization

Objective

Ensure compliance and stability.

Tasks

- Enforce token expiry
- Remove all personal data
- Aggregate analytics only
- Prepare final demo

—

5. MVP Summary

Minimum Viable System

- SURGE ID issuance
- Zone-based QR scans
- Live congestion scores
- Passenger guidance UI

Optional Extensions

- Predictive modeling
- Workforce optimization
- External data integration

—

Conclusion

This roadmap demonstrates how the SURGE system can be implemented incrementally using a simple, Python-based technology stack. Each phase builds logically on the previous one, ensuring that the core functionality is delivered before advanced optimization features are introduced.