Chapter 1

Methodology

3.1 Entry and Exit Logic Using Two-Zone UWB Detection

The development of a robust and intelligent ticketing system begins with defining how a passenger is recognized within a transport terminal during both entry and exit. This section describes a two-zone Ultra-Wideband (UWB) detection model designed to ensure secure, contactless, and logic-bound movement validation at access control points. The system architecture uses spatial verification combined with sequential boolean logic to confirm whether a passenger has legitimately entered or exited the paid area. This model operates across metro stations, terminal hubs, and other fixed-location transport infrastructures.

The design is based on a modular zone arrangement. Each station's access path is divided into two logical regions: one before and one after the gate. Both regions are instrumented with UWB anchors that track movement and tag presence using time-of-flight techniques. A physical gate placed between the zones enforces directional traversal and ensures transactional integrity.

The decision to consider an entry or exit valid depends on the correct presence of the UWB tag in both zones. The logic evaluation is performed through a boolean AND condition, and fare deduction occurs only upon successful validation at exit. This ensures that fare processing is tied directly to legitimate passenger behavior and not just sensor triggering.

3.1.1 Fundamentals of the Two-Zone Structure

The core innovation in this system lies in the intelligent subdivision of passenger flow paths into two logical spatial zones around the gate:

- **Zone 1:** Positioned at the approach side of the entry or exit pathway. Detection in this zone indicates intent to travel or arrival at an exit point.
- **Zone 2:** Located immediately after the gate barrier. Detection here signifies confirmation of movement into or out of the paid area.

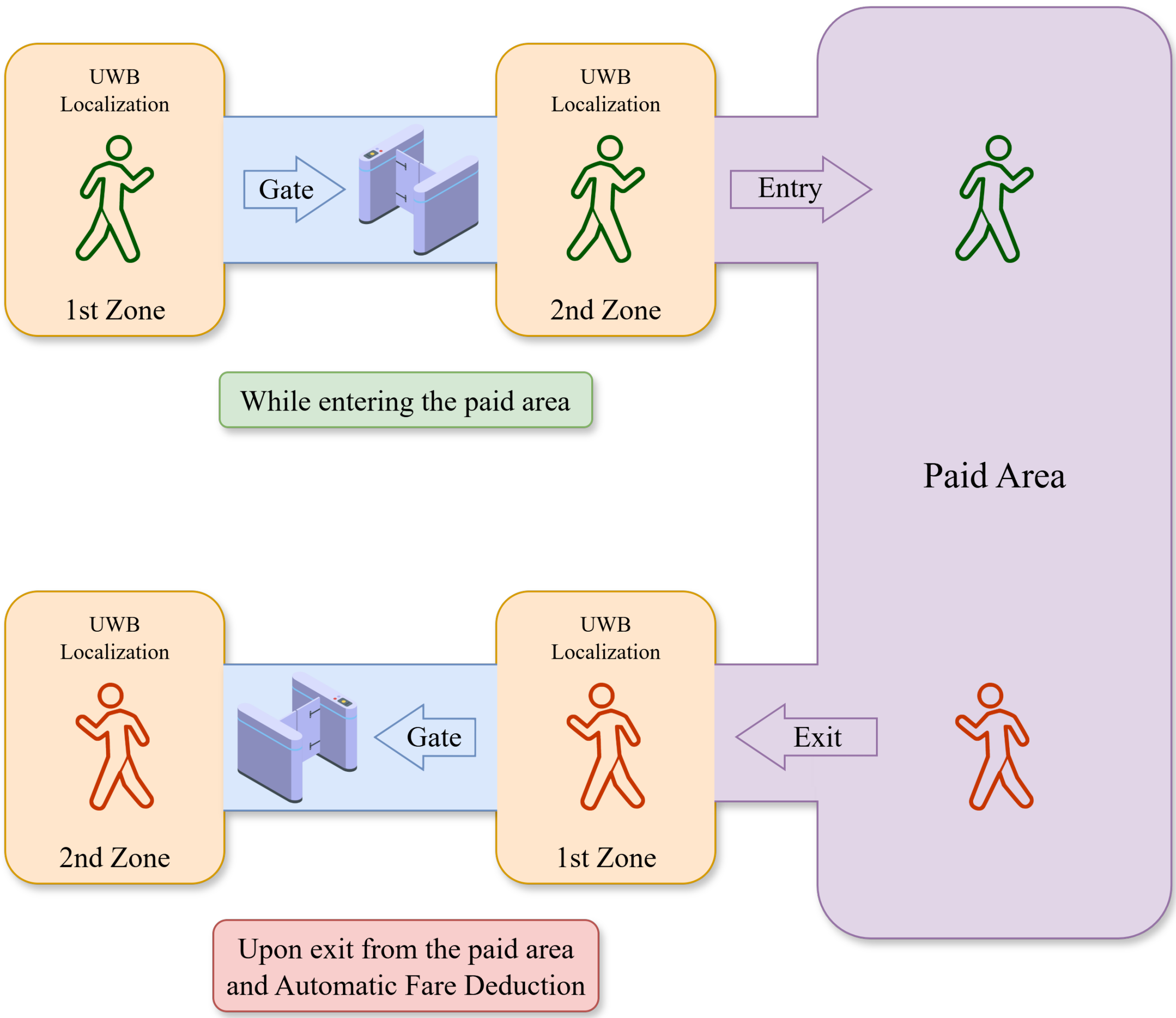

Figure 3.1: Passenger flow structure from unpaid area to paid area during entry and back during exit

Each zone is covered by a set of four UWB anchors configured to triangulate the tag location of passengers. The anchors are fixed in the four corners of each zone to maximize spatial accuracy and reduce blind spots. The presence of a UWB tag is validated within a defined time window to maintain temporal relevance and to prevent bypass manipulation.

3.1.2 Entry Evaluation Using Boolean Conditions

A valid entry is only confirmed when both Zone 1 and Zone 2 register the UWB tag in proper sequence. The logic follows an AND condition that ensures both spatial zones have been passed within a valid window.

The truth table for entry validation is shown below:

Zone 1	Zone 2	Entry Status
FALSE	FALSE	Rejected
FALSE	TRUE	Invalid
TRUE	FALSE	Incomplete
TRUE	TRUE	Accepted

Table 3.1: Entry Status Based on Zone Conditions

Entry Case 1: FALSE FALSE

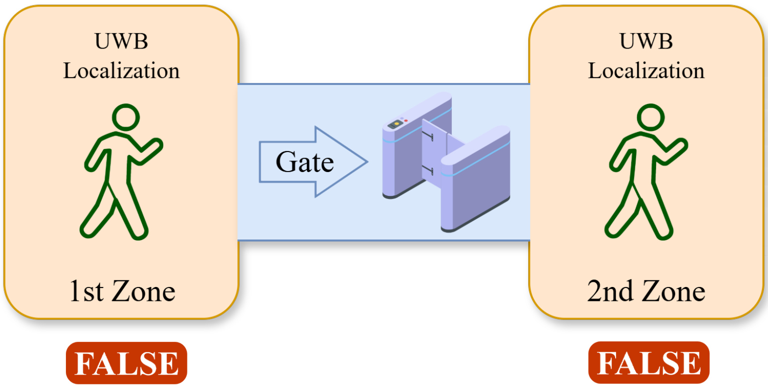

Figure 3.2: Entry rejected due to absence in both zones

No detection occurs in either zone. The system ignores the event. This condition may arise due to non-participant presence near the access point, such as maintenance personnel, loiterers, or passengers without UWB tags.

The system logs the event for statistical purposes but does not open the gate or record any journey. No backend activity is triggered.

Entry Case 2: FALSE TRUE

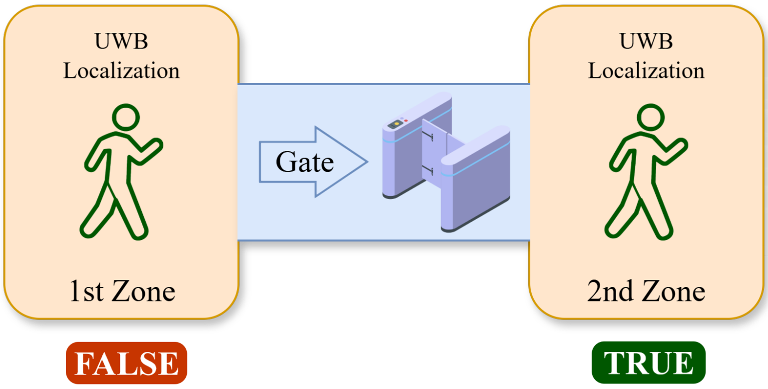

Figure 3.3: Invalid entry triggered in post-gate zone without prior detection

Zone 1 does not detect the tag, but Zone 2 does. This may indicate a passenger entering from the wrong direction or attempting to bypass the system. It could also be caused by an undetected Zone 1 read due to tag obstruction or anchor failure.

This condition is flagged for auditing. Gate access remains locked unless overridden manually. Fare processing remains inactive.

Entry Case 3: TRUE FALSE

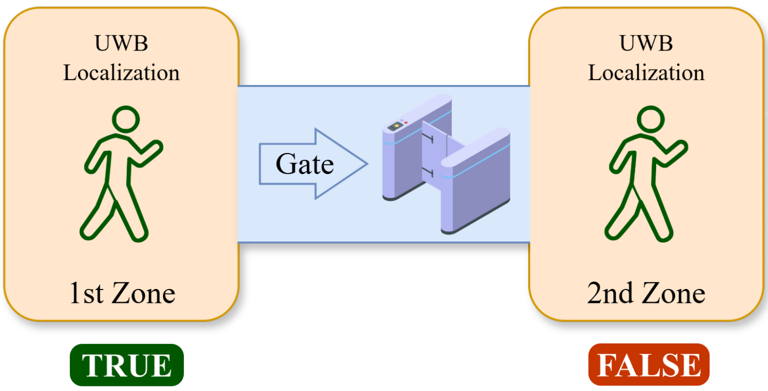

Figure 3.4: Incomplete entry detected in Zone 1 only

The passenger enters Zone 1 and is detected. However, they fail to reach Zone 2. This may occur if they change their mind, stop short of the gate, or are blocked due to insufficient balance or authentication failure.

The system initiates a timeout counter. If Zone 2 is not activated within a defined interval, the entry attempt is cancelled and removed from the live session log.

Entry Case 4: TRUE TRUE

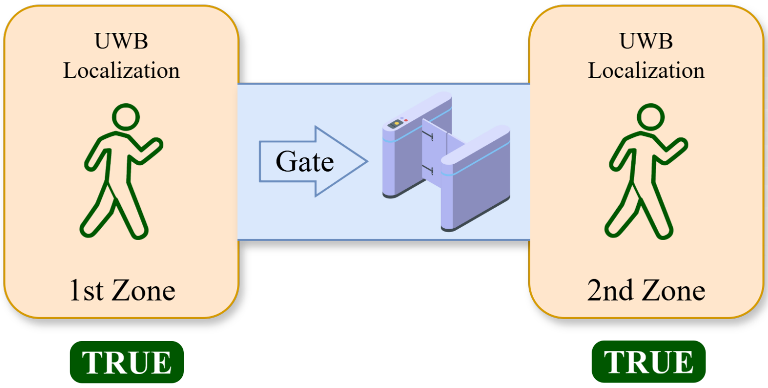

Figure 3.5: Valid entry through both zones with full detection

This is the only valid condition for recognizing entry. The tag is detected in Zone 1, the gate is passed, and Zone 2 confirms the continuation. The gate opens automatically, and a transit session is created. Entry station code and timestamp are recorded.

The fare system marks the user as inside the paid area. Any further behavior such as zone wandering, idle time, or transit path is tracked accordingly.

3.1.3 Exit Evaluation Using Boolean Conditions

Similar to entry, exit detection also uses an AND condition logic applied to the two zones. However, the flow direction is reversed and includes the additional action of fare deduction.

The truth table for exit validation is shown below:

Zone 1	Zone 2	Exit Status
FALSE	FALSE	Rejected
FALSE	TRUE	Invalid

Zone 1	Zone 2	Exit Status
TRUE	FALSE	Incomplete
TRUE	TRUE	Accepted + Fare Deducted

Table 3.2: Exit Status Based on Zone Conditions

Exit Case 1: FALSE FALSE

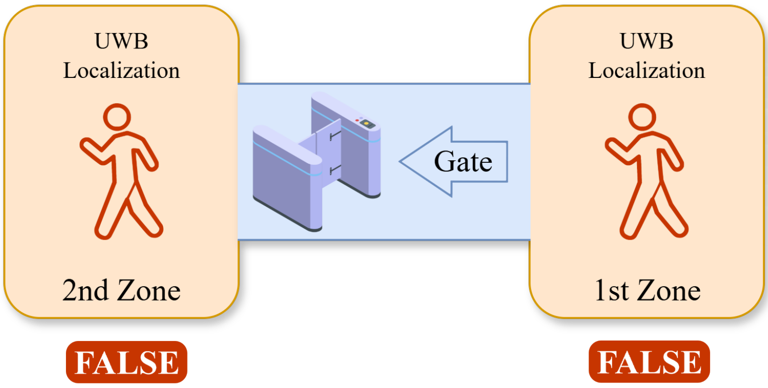

Figure 3.6: Exit rejected due to no tag detection

Neither zone detects the tag. This may result from users hovering near the gate or false motion artifacts. No trip closure or fare processing occurs.

Exit Case 2: FALSE TRUE

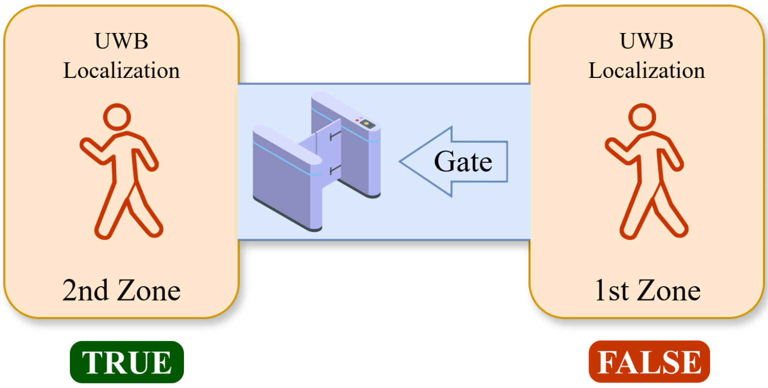

Figure 3.7: Invalid exit triggered post-gate without pre-gate detection

This condition may suggest backward traversal or forced gate access. The system logs this as suspicious. The gate remains locked, and alerts may be sent to the control dashboard.

Exit Case 3: TRUE FALSE

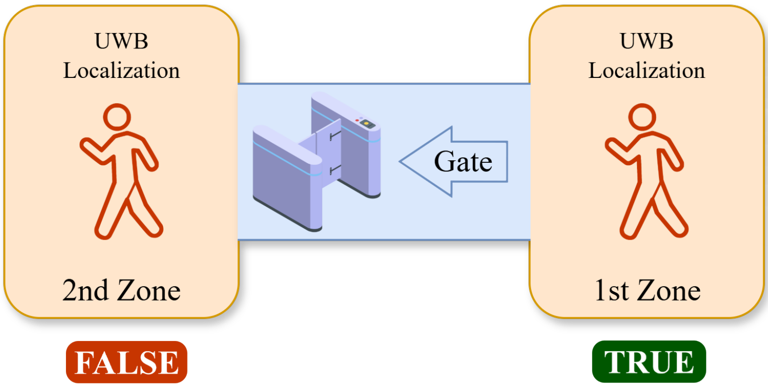

Figure 3.8: Incomplete exit due to tag missing after gate

This scenario may represent an interrupted exit where the passenger is detected in Zone 2 but fails to continue beyond the gate. Gate remains closed. System maintains the session open, and fare computation is paused.

Exit Case 4: TRUE TRUE

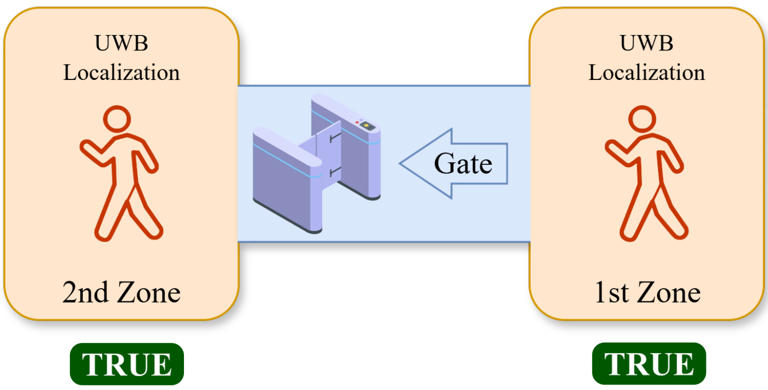

Figure 3.9: Valid exit confirmed in both zones, fare successfully deducted

This is the only valid exit condition. Zone 2 detects intent, the gate is crossed, and Zone 1 confirms exit. The system finalizes the trip, calculates the fare using entry-exit station data, and deducts the appropriate amount. The session is marked as closed.

3.2 Zone Mapping and Trip Logic Resolution

Accurate trip validation in a UWB-based ticketing system depends not only on spatial detection but also on the precise identification of zones across multiple transport terminals. This section focuses on how zone identifiers, anchor configurations, and station codes form an integrated framework for trip tracking and fare enforcement. It also addresses the behavioral corner cases that threaten system integrity and discusses how the architecture handles each through logical time tracking, state transitions, and detection flow validation.

3.2.1 Zone and Station Identification Schema

Each UWB-enabled station is partitioned into two detection zones, Zone 1 and Zone 2, used during both entry and exit procedures. These zones are linked to their respective terminals using a structured identifier format. Each zone is assigned a Zone ID constructed from:

- Station code (e.g., STA01 for Station A)
- Zone tag (Z1 or Z2)
- Direction label (IN or OUT)

For example:

- STA01-Z1-IN (Zone 1 for entry at Station A)
- STA03-Z2-OUT (Zone 2 for exit at Station C)

This structure ensures the backend can reconstruct full passenger journeys using only UWB transition data. It also simplifies multi-station fare table indexing.

Each zone is formed by four or more UWB anchors with unique IDs and fixed coordinates. Anchors are calibrated during setup and revalidated periodically to ensure localization consistency. These anchors operate in synchronous mode to enable time-of-flight (ToF) triangulation.

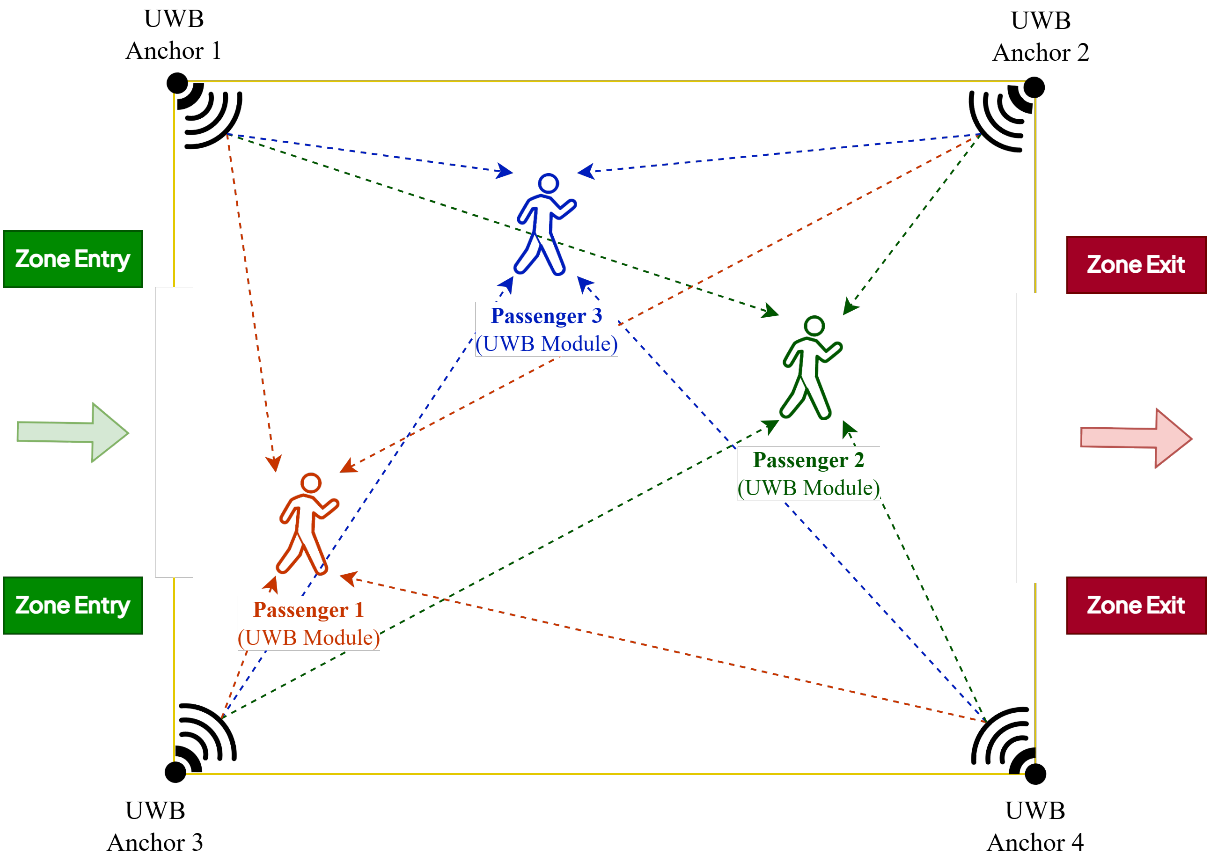

Figure 3.10: UWB anchors forming a spatial detection grid around passengers

Zone detection is confirmed only when ToF measurements satisfy trilateration criteria within zone boundaries defined in the deployment plan.

3.2.2 Fare Computation by Entry and Exit Mapping

Fare is computed based on the station of entry and exit. Once a passenger enters a terminal, their Tag ID, entry timestamp, and Station ID are recorded. The fare is calculated after the exit transition is complete. The fare engine accesses a fare matrix where each row and column corresponds to a station code.

If Entry = STA01 and Exit = STA04, fare is computed using:

$$\text{Fare} = \text{Tariff[STA01][STA04]}$$

Each successful entry-exit pair is archived in the system log with corresponding timestamps and tag identifiers. Time duration is also recorded for additional analytics.

3.2.3 Reentry Without Exit

One misuse scenario occurs when a passenger completes a loop trip without officially exiting. For example:

- Entry at Station A
- Travel to Station C
- Return to Station A
- No exit logged

In this case, no fare is deducted because the system does not detect an exit. This can lead to indefinite ride looping within the paid area.

To prevent this, the system uses a time-based safeguard:

- If `TimeInPaidArea > T_max` (e.g., 90 minutes), the system:
  - Assigns maximum possible fare
  - Sends flag to user account
  - Locks the tag until reset by admin or support

Each zone interaction is timestamped and cross-referenced for temporal consistency. If exit is not logged within the timeframe, the trip is closed with penalty fare.

3.2.4 Interrupted Exit Attempt

Another corner case involves interrupted exit:

- Passenger crosses into Zone 2 at the exit point
- Stops before gate traversal
- Returns into paid area

This partial transition may falsely indicate an exit attempt. If not handled properly, this could either prematurely deduct fare or break the FSM chain.

Resolution involves:

- Fare deduction only occurs after both Zone 2 and Zone 1 at exit are confirmed
- Gate traversal is used as a hard commit
- A 10-second timeout window is enforced. If Zone 1 is not detected within this time, the state is reset

This ensures that incomplete exit attempts do not finalize the trip or deduct fare prematurely.

3.2.5 State Machine for Zone Transition Logic

The system uses a finite state machine (FSM) to model each passenger's movement:

- **IDLE:** No detection
- **Z1_DETECTED:** Zone 1 detection occurred
- **GATE_CONFIRMED:** Gate sensors confirm movement
- **Z2_DETECTED:** Zone 2 detection confirmed
- **IN_TRANSIT:** Passenger is inside paid area
- **EXITED:** Full exit logic complete

Transitions occur based on time, zone status, and gate feedback. In MATLAB simulation, these are modeled with condition-based transitions:

```
if zone1 == 1 && gate == 1 && zone2 == 1
    state = 'IN_TRANSIT';
end
```

This modeling is expanded with timeout functions and fallback states to capture failure cases.

3.2.6 Misuse Cases and Control Strategies

Case	**Failure Condition**	**Mitigation Logic**
No exit detected	Time exceeds threshold	Auto-fare max fare assigned
Partial exit	Zone 2 only, no gate pass	Reset state, wait for complete traversal
Entry without UWB tag	Gate blocks access	Authentication enforced at gate
Duplicate entry at same station	Entry without exit	Deny new entry until session closed

Table 3.3: Failure Cases and Mitigation Logic

This layered approach combines spatial detection, time logic, and mechanical enforcement to secure every passenger interaction against manipulation or error.

3.3 Zone Mapping and Trip Logic Resolution

While static infrastructure such as metro stations benefit from fixed zone definitions, buses introduce complexity due to their mobility. This section presents the implementation of UWB ticketing logic in bus environments, where zone determination is coupled with GPS tracking for dynamic fare calculation. Despite movement, the logical model remains consistent with the fixed-station design by retaining dual-zone detection and logical sequencing. Key enhancements include spatial synchronization between UWB anchors and bus positioning, as well as handling edge behaviors arising from mobile zone transitions.

3.3.1 Dual-Zone Configuration in Buses

Each bus integrates two detection zones:

- **Zone 1:** Installed directly at the door (entry/exit point) to detect physical boarding or alighting.

- **Zone 2:** Configured inside the bus cabin to confirm that the passenger has entered or is within the seating area.

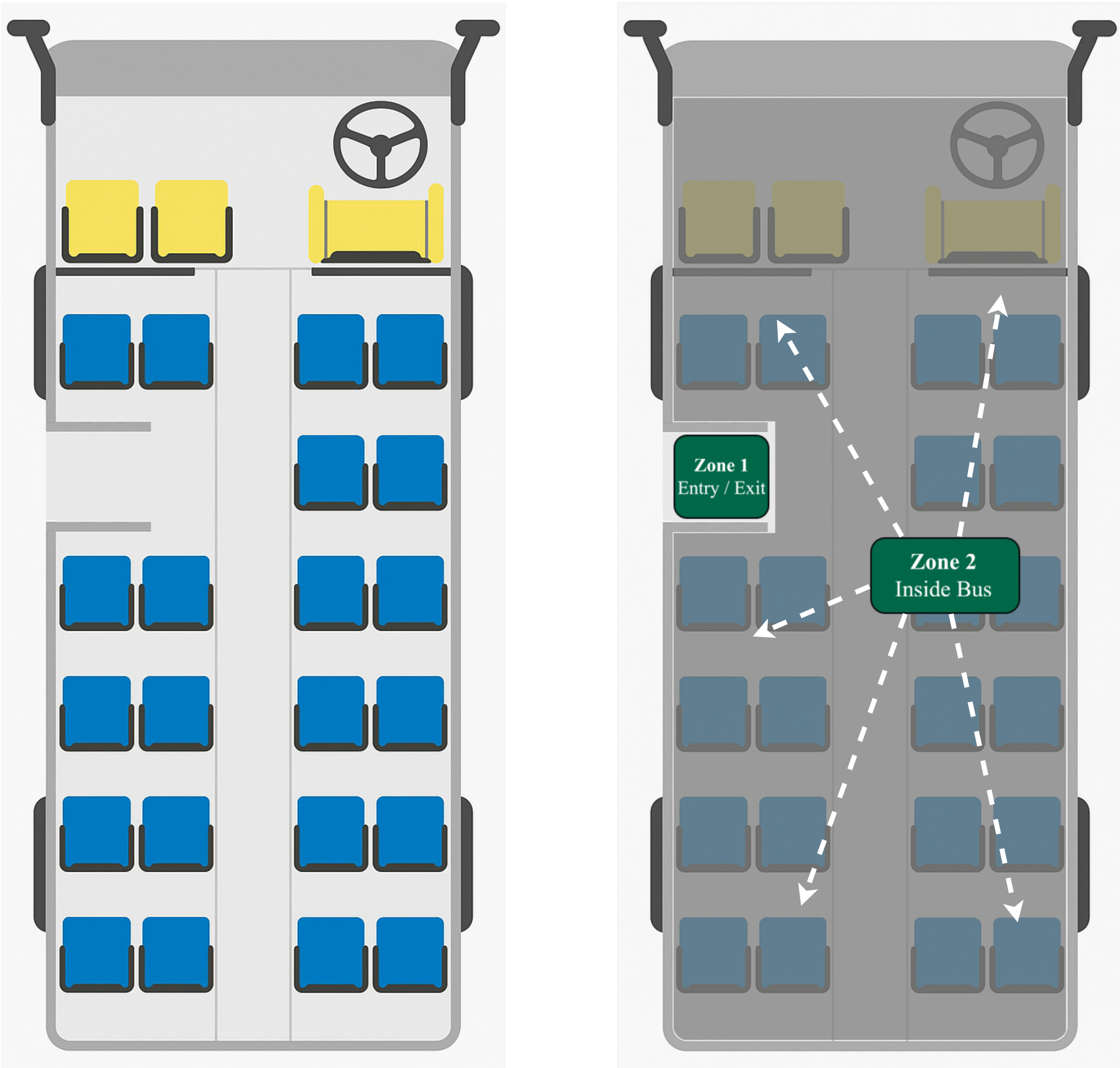

Figure 3.11: Layout of Zone 1 at the bus entrance and Zone 2 covering the interior space

As in terminals, each zone is equipped with four or more UWB anchors. These anchors track the tag's ToF-based position as passengers enter or leave the bus. Zone entry is only validated when detection occurs in correct sequential order.

3.3.2 Entry and Exit Logic in Mobile Context

The two-zone logic remains the same as stationary environments but must operate relative to the movement of the bus. The following logic applies:

Entry Logic in Bus:

- Zone 1 detection (near door) must occur first.
- Zone 2 detection (cabin area) must occur next.
- Gate element is absent, so the sequence serves as logical confirmation.

Exit Logic in Bus:

- Zone 2 detection confirms the passenger is near exit region.
- Zone 1 detection confirms actual departure.

This sequencing confirms a valid entry or exit event in a mobile environment without using gates. The system still relies on the AND logic for TRUE–TRUE combinations.

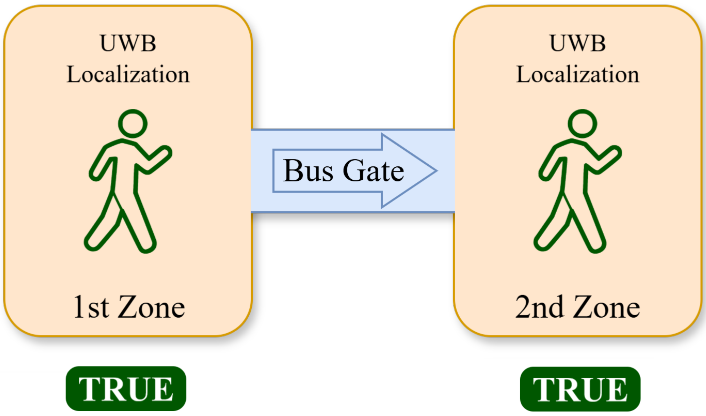

Figure 3.12: A valid bus entry when both Zone 1 and Zone 2 detect the tag sequentially

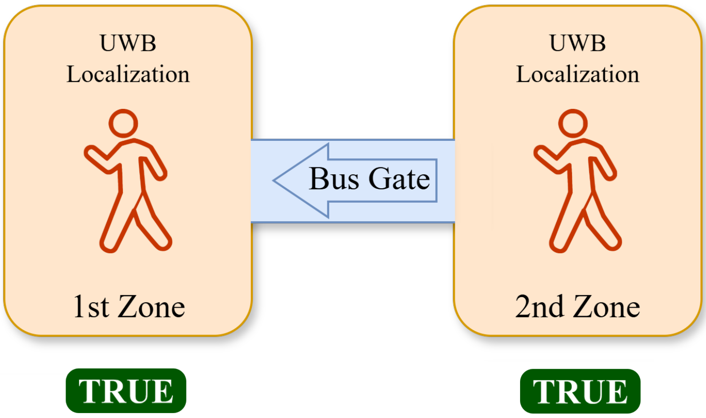

Figure 3.13: A valid bus exit when Zone 2 is detected first and followed by Zone 1

3.3.3 Fare Calculation Using GPS

Unlike metro systems where fare is distance-based between station pairs, bus fare is computed using GPS-coordinated zone transition data. The logic is as follows:

On entry:

- System records current GPS coordinates as boarding point.
- Timestamp and Tag ID are logged.

On exit:

- System records current GPS coordinates as exit point.
- Distance is computed using the haversine formula:

$$\text{Distance} = 2r \cdot \arcsin\left(\sqrt{\sin^2\left(\frac{\Delta\phi}{2}\right) + \cos(\phi_1)\cos(\phi_2)\sin^2\left(\frac{\Delta\lambda}{2}\right)}\right)$$

Where:

- r = radius of Earth
- ϕ_1, ϕ_2 = latitudes
- λ_1, λ_2 = longitudes

Fare is computed as:

$$\text{Fare} = \text{Base Rate} + (\text{Distance} \times \text{Rate per km})$$

This makes the bus implementation scalable across any route and independent of predefined terminal locations.

3.3.4 Zone Traversal Model in MATLAB Simulation

In simulation, each bus movement is modeled as a sequence of GPS points with attached time series data. Passenger tags are represented as entities with position vectors updated per time step. Entry and exit are triggered when those vectors fall within defined zone boundaries.

MATLAB pseudocode representation:

```
if inZone1 && movedToZone2
    entryConfirmed = true;
    entryLocation = gpsCoords;
end

if inZone2 && movedToZone1
    exitConfirmed = true;
    exitLocation = gpsCoords;
end
```

Simulation verifies zone transitions and uses positional data for trip distance calculations. Multiple agents (passengers) can be simulated in parallel using arrayed state machines.

3.3.5 Edge Behavior in Buses and Fail-Safes

Several mobile-specific edge behaviors must be managed:

Passenger enters but remains at the door (Zone 1 only):

- Entry not confirmed, no trip logged.

Passenger exits but lingers inside (Zone 2 only):

- Exit not confirmed, trip remains active.

Passenger enters and exits within short GPS range:

- Minimum trip distance logic enforces base fare.

Device not detected mid-journey:

- Timeout-based fallback closes trip after N stops or distance.

The FSM is extended with GPS-linked constraints to verify temporal and spatial consistency:

State	Condition
IDLE	Tag not detected
DOOR_DETECTED	Zone 1 triggered
IN_CABIN	Zone 2 confirmed after Zone 1
TRIP_ACTIVE	Entry confirmed, GPS tracking started
NEAR_EXIT	Zone 2 triggered while in motion
EXIT_CONFIRMED	Zone 1 confirmed, fare computed using GPS delta

Table 3.4: State Machine Conditions

This allows precise mapping of both rider behavior and vehicle movement to compute correct ticketing charges and validate travel sequences.

3.4 UWB Positioning Using ToF and Trilateration

Ultra-Wideband (UWB) positioning is a radio-frequency-based method that enables precise location estimation with accuracy up to a few centimeters, even in dense environments. Its application in public transportation relies on the ability to determine whether a user is present in a specific spatial zone and to validate zone transitions based on movement paths derived from signal measurements. This section explains how Time of Flight (ToF) and trilateration are used to determine the user's real-time position in relation to zone boundaries.

3.4.1 Time of Flight Principle

Time of Flight is a method of measuring the time taken for a radio signal to travel from a transmitter to a receiver. In the UWB system, each tag (carried by the passenger) emits or responds to UWB pulses, and multiple anchors (fixed receivers) measure the time taken for the signal to arrive.

$$\text{Distance} = c \times \text{ToF}$$

Where:

- c = speed of light ($\sim 3 \times 10^8$ m/s)

- ToF = time taken by the signal to reach the anchor

Each anchor measures the ToF independently, and these distance values are used to calculate the position of the tag via multilateration.

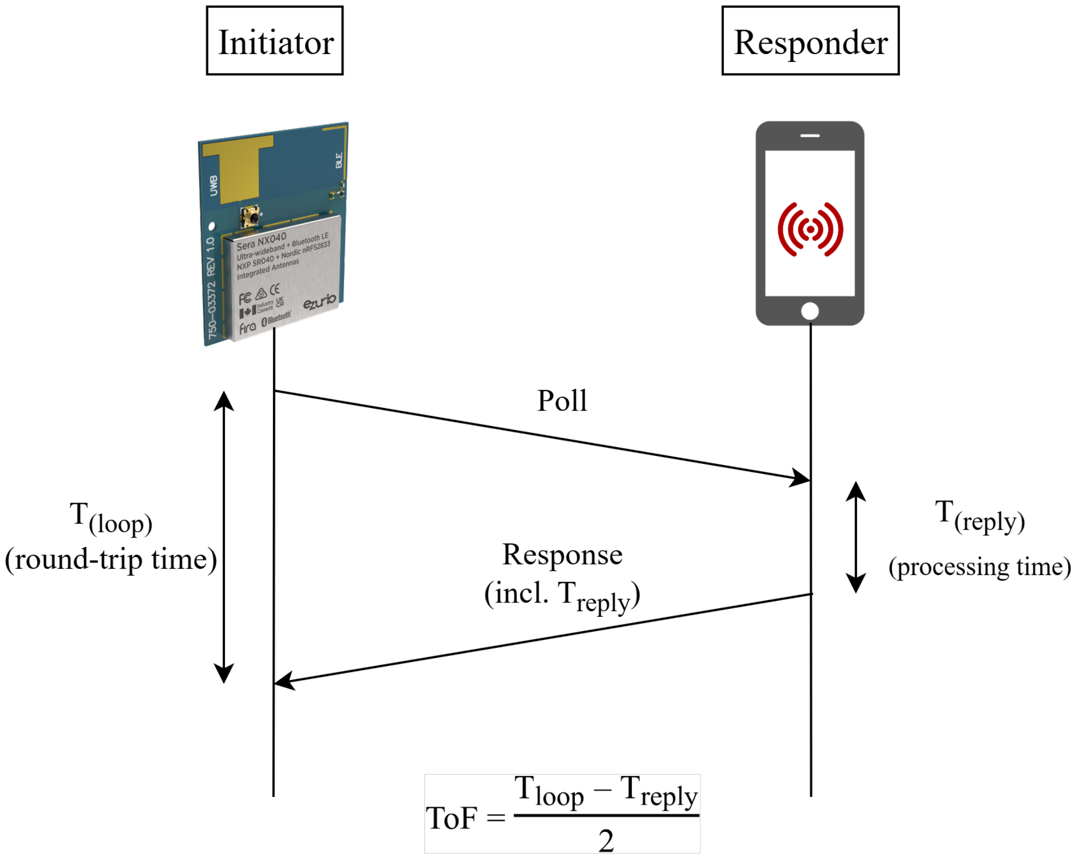

Figure 3.14: Time of Flight measurement between tag and UWB anchor

To mitigate multipath effects or environmental noise, ToF measurements undergo signal calibration, where only the first signal path (direct line-of-sight) is considered for range estimation. Advanced UWB systems apply filtering algorithms to discard reflected paths.

3.4.2 Trilateration-Based Position Estimation

Trilateration refers to the process of calculating a position based on known distances from three or more fixed points. In 2D space, a minimum of three anchors is required; in 3D space, four anchors are necessary for accurate positioning.

Given three anchors at coordinates (x_1, y_1), (x_2, y_2), (x_3, y_3) and corresponding distances d_1, d_2, d_3, the tag's position (x, y) can be estimated by solving the system of equations:

$$(x - x_1)^2 + (y - y_1)^2 = d_1^2$$
$$(x - x_2)^2 + (y - y_2)^2 = d_2^2$$
$$(x - x_3)^2 + (y - y_3)^2 = d_3^2$$

Solving this system yields the precise coordinates of the tag within the defined zone. UWB systems typically use four or more anchors to improve robustness and resolve ambiguity in non-ideal environments.

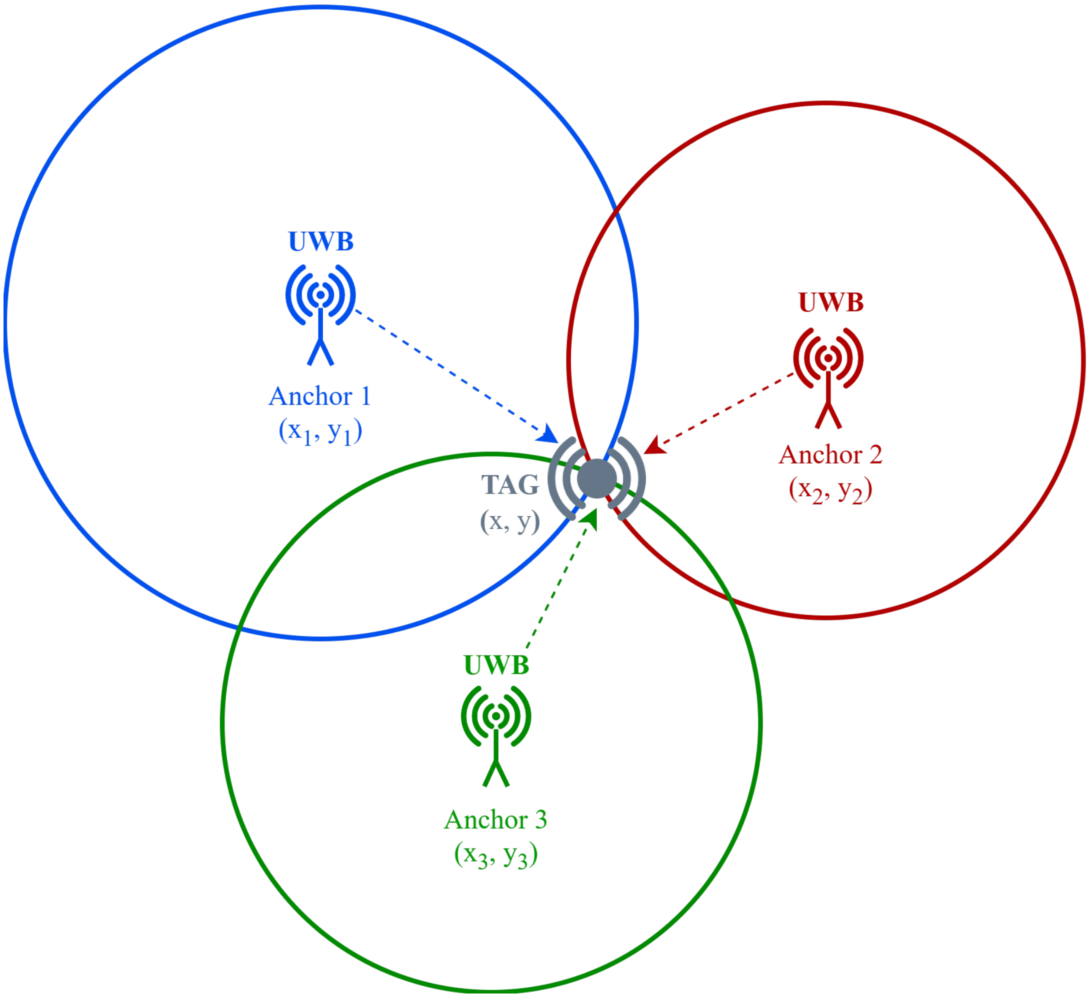

Figure 3.15: Trilateration method using three anchors to calculate the location of a tag in 2D space

In simulation, this is often implemented using least-squares optimization or geometric algebra methods. MATLAB simulation models utilize matrix representations of anchor positions and vectorized distance equations to calculate the tag's estimated position at each timestep.

3.4.3 Spatial Zone Detection Logic

Zones are defined by bounding boxes in the 2D coordinate space of the station or bus. Each zone has spatial limits:

$$x_{\min} \leq x \leq x_{\max}$$

$$y_{\min} \leq y \leq y_{\max}$$

Once a tag's coordinates (x, y) are estimated via trilateration, the system checks whether the tag lies within the predefined boundaries of Zone 1 or Zone 2. This binary result (TRUE or FALSE) is used to determine zone presence.

This method enables:

- Real-time detection of zone entry and exit
- Detection of zone sequence (for valid entry or exit events)
- Precise monitoring in crowded environments (multiple tags)

Simulation environments define each zone with coordinate constraints and map real-time tag positions onto zone occupancy flags.

3.4.4 Practical Considerations in Indoor Environments

This method enables:

- **Multipath Interference:** Caused by signal reflection in enclosed spaces. Mitigated using ToF-first-path filtering.
- **Anchor Placement:** Must ensure at least three anchors maintain line-of-sight visibility to all possible user positions.
- **Anchor Calibration:** All anchors must be precisely aligned to a common coordinate system with fixed offsets accounted for.
- **Signal Congestion:** In high-density passenger areas, the system must handle concurrent tag requests without loss of accuracy.

These considerations are addressed during system planning and reinforced during simulation trials.

3.4.5 MATLAB-Based Position Estimation Logic

A simplified simulation model in MATLAB represents anchors and tag positions as:

```
anchors = [x1 y1; x2 y2; x3 y3; x4 y4];
distances = [d1; d2; d3; d4];

position_estimate = estimatePosition(anchors, distances);
```

The function `estimatePosition()` uses non-linear solvers or least-squares estimation to minimize the difference between computed and actual distances, returning an estimated location vector.

This estimated position is then passed to a zone-checking function:

```
if isWithinZone(position_estimate, zone1_bounds)
    zone1_flag = true;
end
```

Multiple agents (passengers) can be simulated in parallel using vectorized position arrays and loop-based state machines. The simulation outputs include time-series location plots, zone occupancy flags, and inferred entry-exit sequences.

3.5 Security and Fraud Mitigation Architecture

As public transportation systems become increasingly digitized and decentralized, the importance of embedding robust security protocols at every stage of the user journey becomes paramount. The proposed UWB-based ticketing system introduces a high-resolution localization mechanism and contactless fare computation, but it must also address potential vulnerabilities arising from data spoofing, signal replay attacks, unauthorized entry, identity fraud, and backend manipulation. This section outlines the security framework integrated into the system, detailing both proactive and reactive measures for fraud prevention.

3.5.1 Multi-Layer Security Model

The system adopts a multi-layered architecture that integrates protection mechanisms at various critical points:

Layer	**Primary Threats Addressed**	**Core Technologies Used**
Physical Layer	Signal jamming, unauthorized zone entry	UWB STS (Scrambled Timestamp Sequences)
Network Layer	Replay attacks, device impersonation	Nonce-based session authentication
Application Layer	Fake identity tags, ticket manipulation	Encrypted TagID + Secure Trip Log
Backend System Layer	Fare tampering, false transaction logs	Hash-chained fare records + audit trail

Table 3.5: Security Layers, Threats, and Technologies

This layered model ensures that compromise at one level does not propagate system-wide damage. Each component is independently verifiable and cryptographically sealed for auditability.

3.5.2 Secure Tag Authentication

Each passenger is assigned a UWB-compatible tag, either embedded in a smartphone, wearable, or transit card. These tags contain:

- TagID (unique, hashed)
- UserID reference
- Nonce (one-time value) per session

- Encrypted timestamp log

When a tag enters Zone 1, the system initiates a challenge-response handshake with the tag to verify authenticity before updating zone status. If this step fails, the gate remains locked.

To prevent tag cloning or emulation, each device must present a cryptographic signature validated via ECC (Elliptic Curve Cryptography) or ECDSA.

3.5.3 Gate-Level Access Control

Gates serve as enforcement points where decisions are made based on both positional data and cryptographic checks.

Process at the gate includes:

1. Validation of recent zone presence (Zone 1 or Zone 2).
2. Verification of TagID integrity using:
   - Replay protection via timestamp freshness
   - Digital signature verification
3. Fare balance check from backend
4. Final gate unlock signal

If any check fails, the gate remains locked and an access denied event is logged.

In simulation environments, this sequence is modeled using stateful checks:

```
if isAuthenticated(TagID) && isZoneValidated(user_pos) && hasBalance(UserID)
   unlockGate();
else
   denyAccess();
end
```

3.5.4 Fare Integrity and Tamper Detection

The fare calculation engine maintains an append-only, hash-linked trip log that includes:

- TagID
- EntryZoneID
- ExitZoneID
- Entry/Exit timestamps
- Fare charged

Each fare transaction is hashed and linked to the previous record, forming a lightweight blockchain-style ledger, which is:

- Immutable (tamper-evident)
- Auditable by transport authority
- ExitZoneID
- Resistant to retroactive fare manipulation

If any mismatch is found (e.g., tag claims a different station), the fare is re-evaluated and a conflict flag is raised.

3.5.5 Fraud Scenarios and Prevention Strategies

Case 1: Tag Cloning

Threat: Malicious actor clones a legitimate user's TagID. Malicious actor clones a legitimate user's TagID.

Mitigation:

- Session-based nonce ensures cloned tags fail on the second use.
- Secure Element hardware in tags stores keys non-extractably.

Case 2: Replay Attack

Threat: Previously captured entry signals are replayed to spoof entry.

Mitigation:

- Timestamp freshness check.
- Zone state validation ensures that Zone 1 → Gate → Zone 2 must happen sequentially within a valid time window.

Case 3: Fare Evasion by Zone Bypass
Threat: A user jumps directly to Zone 2 without passing through the gate.
Mitigation:

- Gate passage required to activate zone transition.
- Absence of gate validation cancels zone state.

Case 4: Backend Manipulation
Threat: Insider attempts to erase trip logs or modify fare calculations.
Mitigation:

- Hash-linked fare records.
- Admin activity logging.
- Role-based access control (RBAC) in backend.

3.5.6 Real-Time Alert and Lockout System

The backend system constantly monitors for anomalous behavior patterns. This includes:

- Multiple failed tag verifications
- Multiple attempts to access exit without entry
- Loop journeys exceeding allowed time
- Exceeding system-defined thresholds for daily travel

Upon detection, the system can:

- Flag the account for investigation
- Automatically apply a penalty fare
- Send alert to user's app
- Lock further usage for a defined cooldown period

3.6 Failure Recovery and Redundancy Management

In any real-world deployment of UWB-based ticketing within high-throughput public transport environments, a multitude of technical or behavioral interruptions can compromise the smooth operation of the system. These range from passenger hesitation, gate malfunctions, or signal blockage due to crowds, to more systemic issues such as anchor desynchronization or tag transmission failure. Therefore, the system must be engineered not only for ideal operation but also for graceful degradation, allowing fallback modes and fail-safes without compromising fare computation or entry-exit validation.

This section outlines the key scenarios of failure, the anticipated system behavior, and the mitigation strategies designed to ensure reliability, continuity, and user trust.

3.6.1 Zone Detection Failure

Scenario: A UWB tag is successfully detected in Zone 1, but due to anchor occlusion, fails to register in Zone 2.
Impact:

- Entry process is left incomplete.
- Passenger may have physically entered the paid area, but the system did not record it.

Recovery Logic:

- Initiate a Zone Re-scan Timer (e.g., 2 seconds) to retry Zone 2 detection.
- If detection fails, the system checks:
  - Whether gate passage was logged.
  - Whether entry intention was valid (e.g., via mobile app interaction).
- If both conditions are met, fallback triggers:
  - Provisional entry status recorded.
  - User flagged for post-entry validation by staff or backend.

This minimizes wrongful denial of service while maintaining backend accountability.

3.6.2 Gate Malfunction During Valid Entry

Scenario: UWB detection completes successfully for both zones, but the gate does not physically open.

Impact:

- Journey is logically initiated but not physically possible.

Mitigation:

- If no gate passage is recorded within a timeout window (e.g., 5 seconds), the system:
  - Cancels the entry attempt.
  - Rolls back the in-transit flag and fare timer.
  - Sends a prompt to the mobile app or audio alert for retry or alternative gate use.
  - Logs the event with gate hardware status for diagnostics.

3.6.3 Unexpected Power Loss or Anchor Failure

Scenario: One or more anchors in a zone stop responding.

Impact:

- Localization accuracy may drop below the required threshold.
- Zone determination becomes ambiguous or incorrect.

Recovery Strategy:

- Use anchor triangulation fallback:
  - Switch to 3-anchor localization instead of 4, reducing accuracy but maintaining continuity.
  - Apply spatial uncertainty buffer in logic model.
- Backend logs affected timestamps and flags fare as "under degraded conditions" for later audit.

This minimizes wrongful denial of service while maintaining backend accountability.

3.6.4 User Behavior Anomalies

Scenario: A user enters Zone 1, but after passing the gate, returns back into Zone 1 or lingers between zones.

Impact:

- System could misinterpret this as a completed entry or canceled trip.
- May cause incorrect fare status or user confusion.

Resolution:

- Temporal logic enforcement:
  - Require valid sequence Zone 1 → Gate → Zone 2 within a specific time range (e.g., 10 seconds).
  - If sequence is broken or reversed, reset zone state and deny entry.
  - Mobile app prompts user to restart entry attempt.

3.6.5 Backend Downtime or Communication Loss

Scenario: Local UWB detection is functional, but backend fare engine or central controller is temporarily offline.

Impact:

- Tag verification and fare computation become unavailable.

Fallback Protocol:

- Local controller logs all zone transitions in an encrypted temporary cache.
- When communication is restored, data is synced to backend with correct timestamps.
- All fare operations marked as "delayed sync", and time offsets adjusted accordingly.

This ensures that no data is lost and users are not penalized due to central outages.

3.6.6 Redundancy Through Anchor Overlap

To handle congestion and dynamic crowd movement, zones are designed with anchor overlap margins:

- Each zone has at least one anchor shared with the adjacent zone.
- This enables soft zone transitions and minimizes the chances of passengers being lost during mid-transition.
- In simulation (MATLAB), this is implemented by allowing transitional coordinates to satisfy either zone's threshold with a buffer region.

3.6.7 Alert Escalation and Manual Intervention Protocol

In cases where automated recovery fails or conflict arises (e.g., repeated failed exits, duplicate tags), the system:

- Flags the journey as "unresolved" in backend
- Notifies:
  - On-site personnel
  - Control center
  - User app (with context-specific instructions)
- Manual override or verification process initiated

The record is locked from further auto-processing until conflict is cleared.

3.6.8 Simulation Strategy for Failure Scenarios

In simulation, these recovery logics are modeled by:

- Random failure injection (e.g., simulate 1 in 50 anchor dropouts)
- State machine branching under incomplete conditions
- Buffer-based rollbacks and retry counters
- Flagging fare status: `valid`, `provisional`, `fallback`, `delayed`

The record is locked from further auto-processing until conflict is cleared.