Smart Parking System

Made By:
Mohamed Abdo
MD Ruhul Saleh
MD Helal Uddin
Moeez Mufti

Group 17

19.06.2025

System Overview

- Smart, automated parking system using RFID and slot sensors
- Controls vehicle access, tracks parking time, and automates fee collection
- Integrates:
  - RFID access control
  - Real-time slot detection
  - Payment processing
- Ensures only authorized users enter and park
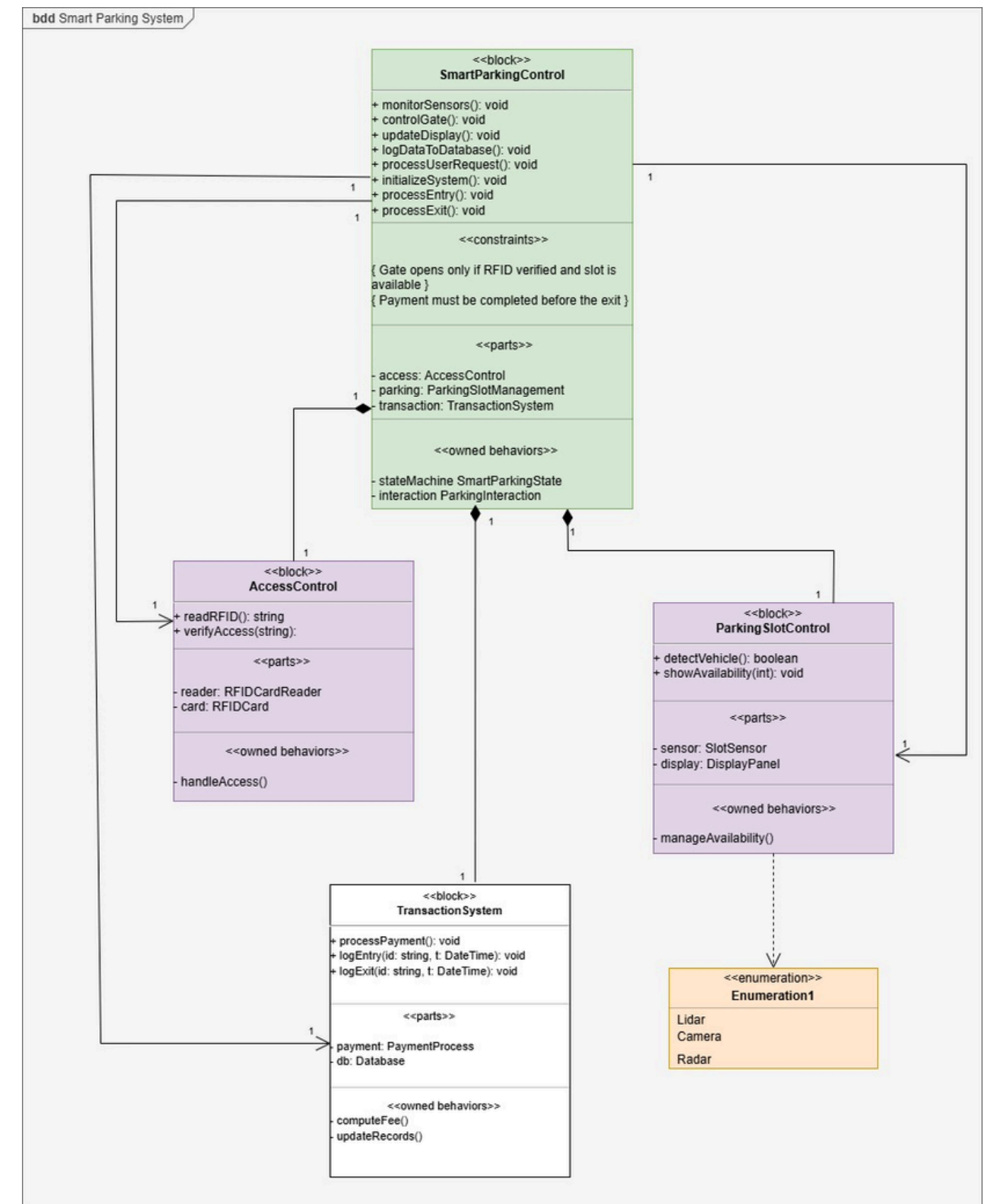- Provides live availability display and smooth entry/exit flow

The main goal is to model the entire parking system, including the RFID access control, vehicle tracking logic, parking slot monitoring, and integrated payment mechanism.

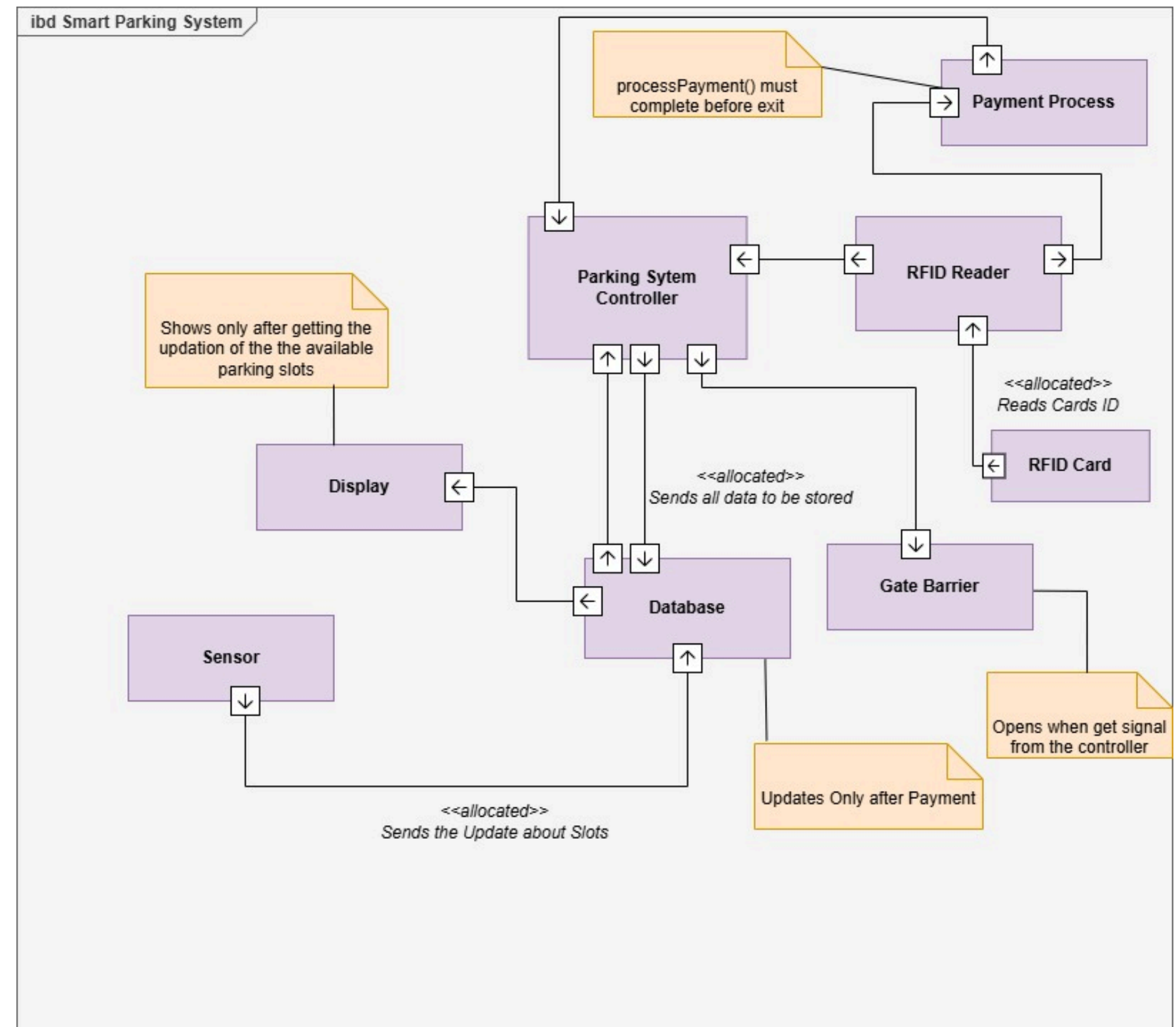# 1. Structure Diagrams

Block Definition Diagram

1. Outlines the structural foundation of the Smart Parking System.
2. Central block: SmartParkingControl, which connects to AccessControl, ParkingSlotControl and TransactionSystem.
3. Captures logical relationships and constraints between system blocks.
4. Supports function-to-component mapping and defines how structural elements enable system behavior.

Internal Block Diagram

1. Shows how system components interact during real-time operation.
2. Highlights internal communication between the controller, RFID reader, sensors, database, payment process, display and gate barrier.
3. Database updates live slot availability and display shows that.
4. Diagram connections represent functional links with responsibilities (e.g., "read ID", "update slot").
5. Ensures actions (like gate opening or logging) occur only when proper conditions are met.

Package Diagram

1. Requirement Package:
   a. Captures functional and non-functional requirements.
2. Behavior Package:
   a. Use case diagrams
   b. Sequence diagrams
   c. Activity diagrams
   d. State machine diagrams
3. Structure Package:
   a. Represents system components and connections:
      i. Block definition diagrams
      ii. Internal block diagrams
      iii. Parametric diagrams

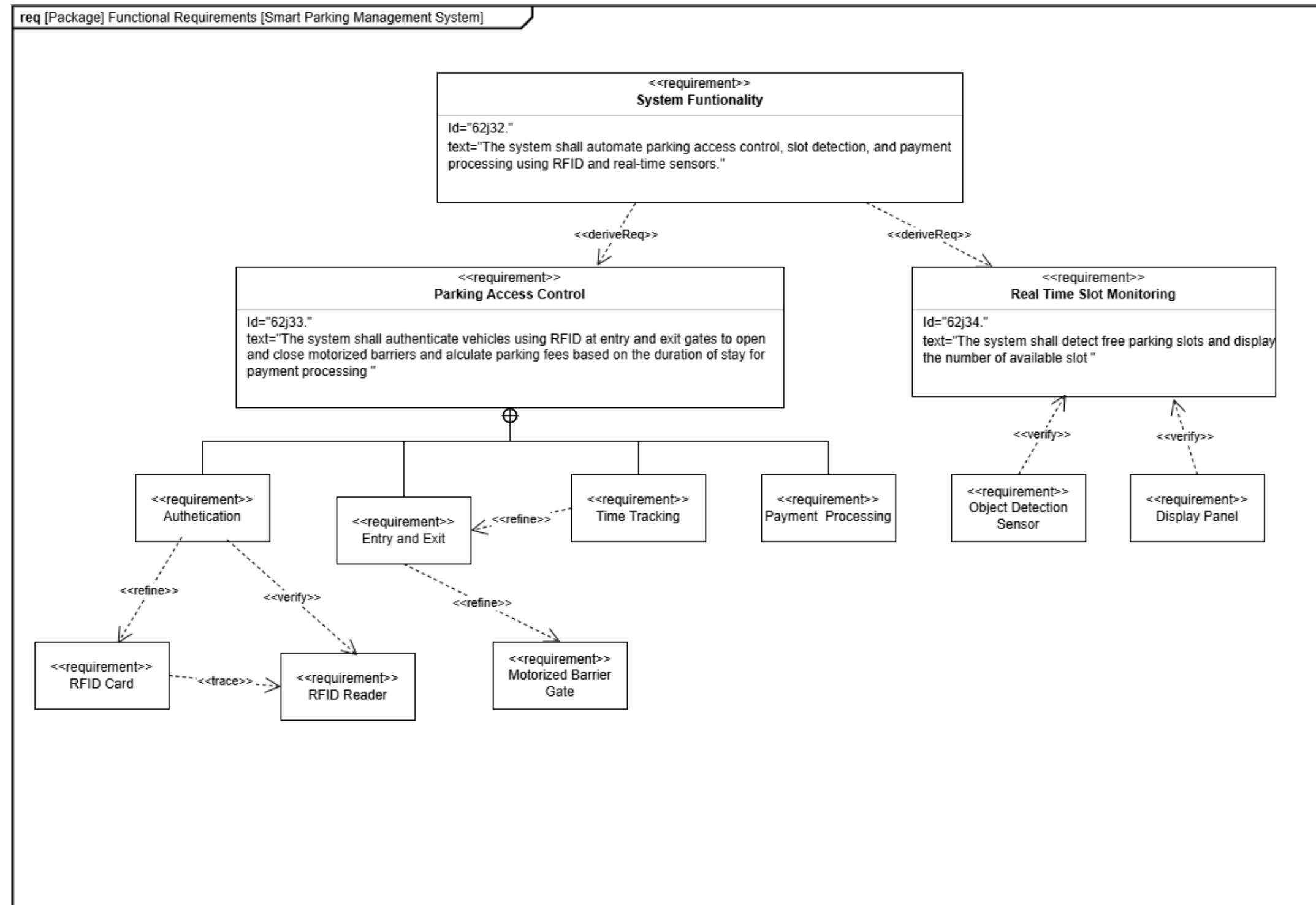pkg Smart Parking System [Model Organization]
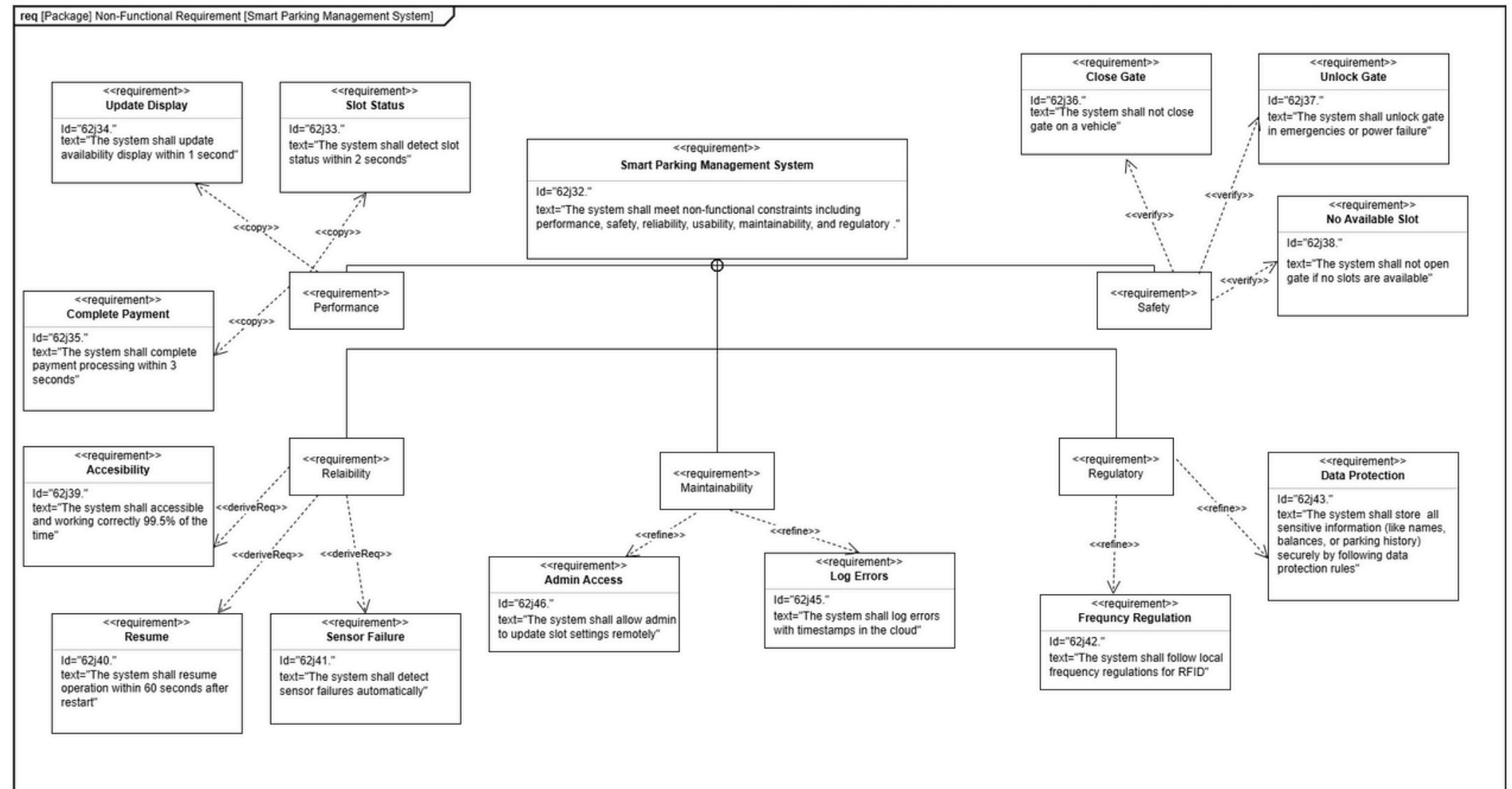
2. Requirement Diagrams

functional requirements

1. Focuses on key system functions
2. Central requirement: System Functionality
3. Includes verification and traceability links for validation.
4. Ensures all components align with smart automation goals.

8

non-functional requirements

1. Focuses on key system qualities
2. Central requirement: Smart Parking Management System
3. Each quality is linked to refined or verifiable sub-requirements.
4. Ensures the system meets essential quality standards beyond functionality
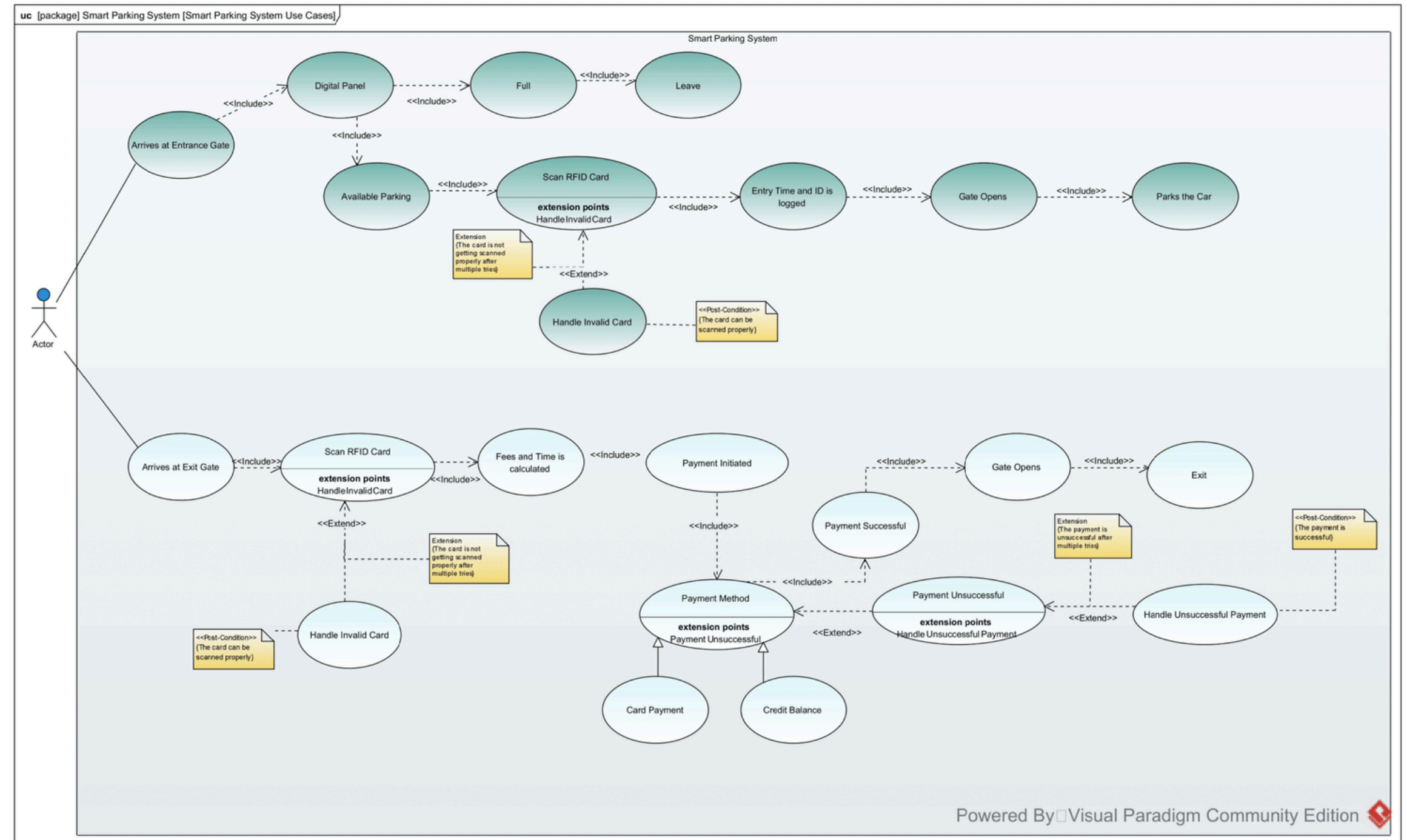

req [Package] Non-Functional Requirement [Smart Parking Management System]

Update Display — <<requirement>> Id="62j34." text="The system shall update availability display within 1 second"

Slot Status — <<requirement>> Id="62j33." text="The system shall detect slot status within 2 seconds"

Smart Parking Management System — <<requirement>> Id="62j32." text="The system shall meet non-functional constraints including performance, safety, reliability, usability, maintainability, and regulatory ."

Close Gate — <<requirement>> Id="62j36." text="The system shall not close gate on a vehicle"

Unlock Gate — <<requirement>> Id="62j37." text="The system shall unlock gate in emergencies or power failure"

No Available Slot — <<requirement>> Id="62j38." text="The system shall not open gate if no slots are available"

Complete Payment — <<requirement>> Id="62j35." text="The system shall complete payment processing within 3 seconds"

Performance — <<requirement>>

Safety — <<requirement>>

Accesibility — <<requirement>> Id="62j39." text="The system shall accessible and working correctly 99.5% of the time"

Reliability — <<requirement>>

Maintainability — <<requirement>>

Regulatory — <<requirement>>

Data Protection — <<requirement>> Id="62j43." text="The system shall store all sensitive information (like names, balances, or parking history) securely by following data protection rules"

Resume — <<requirement>> Id="62j40." text="The system shall resume operation within 60 seconds after restart"

Sensor Failure — <<requirement>> Id="62j41." text="The system shall detect sensor failures automatically"

Admin Access — <<requirement>> Id="62j46." text="The system shall allow admin to update slot settings remotely"

Log Errors — <<requirement>> Id="62j45." text="The system shall log errors with timestamps in the cloud"

Frequncy Regulation — <<requirement>> Id="62j42." text="The system shall follow local frequency regulations for RFID"

<<copy>> <<verify>> <<deriveReq>> <<refine>>
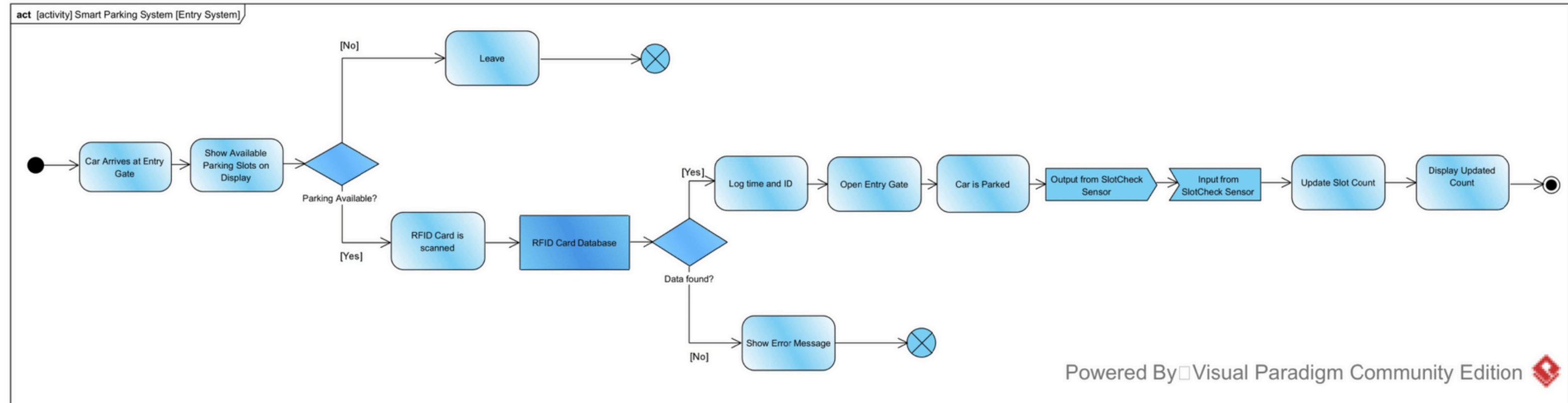
3. Behavioral Diagrams

use case diagram

- Check availability on display
- Scan RFID to enter
- Log time & open gate
- Park vehicle (slot marked)
- Scan RFID to exit
- Calculate fee & pay
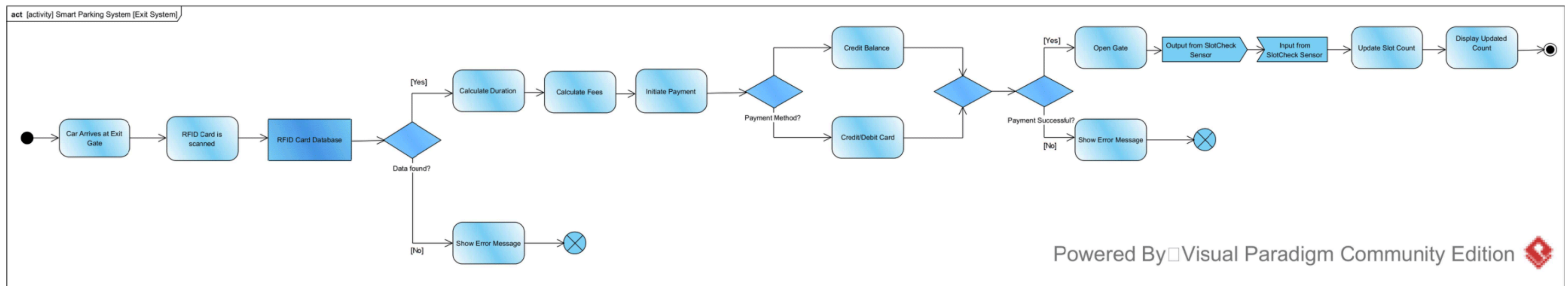- Choose payment method
- Exit gate opens

activity diagram - entry

1. User arrives & checks availability
2. Scans RFID card
3. System validates RFID
4. If valid & slots available →
   a. Log entry time & user ID
   b. Update slot count
   c. Open gate
5. If invalid or full →
   a. Display error/ deny access

act [activity] Smart Parking System [Entry System]

[No] Leave

Car Arrives at Entry Gate → Show Available Parking Slots on Display → Parking Available?

[Yes] RFID Card is scanned → RFID Card Database → Data found?

[Yes] Log time and ID → Open Entry Gate → Car is Parked → Output from SlotCheck Sensor → Input from SlotCheck Sensor → Update Slot Count → Display Updated Count

[No] Show Error Message

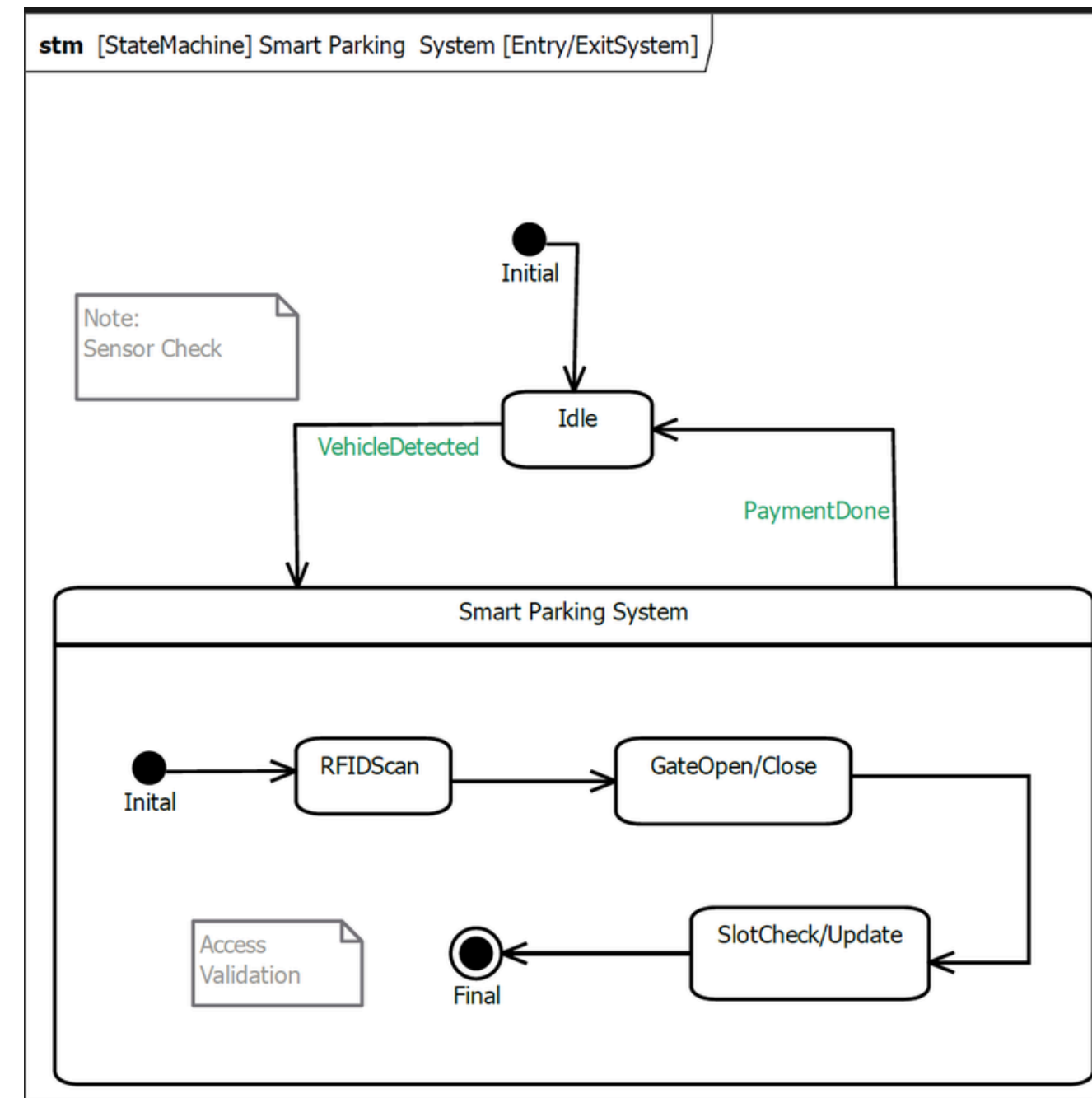Powered By Visual Paradigm Community Edition

activity diagram - exit

1. User scans RFID at exit
2. System validates RFID & fetches entry data
3. Calculate parking duration & fee
4. User selects payment method
   a. Credit balance or card payment
5. If payment successful →
   a. Open gate
   b. Update slot count
6. If failed or insufficient balance →
   a. Display error message

act [activity] Smart Parking System [Exit System]

Car Arrives at Exit Gate → RFID Card is scanned → RFID Card Database → Data found?
[Yes] → Calculate Duration → Calculate Fees → Initiate Payment → Payment Method?
Credit Balance / Credit/Debit Card → Payment Successful?
[Yes] → Open Gate → Output from SlotCheck Sensor → Input from SlotCheck Sensor → Update Slot Count → Display Updated Count
[No] → Show Error Message
Data found? [No] → Show Error Message

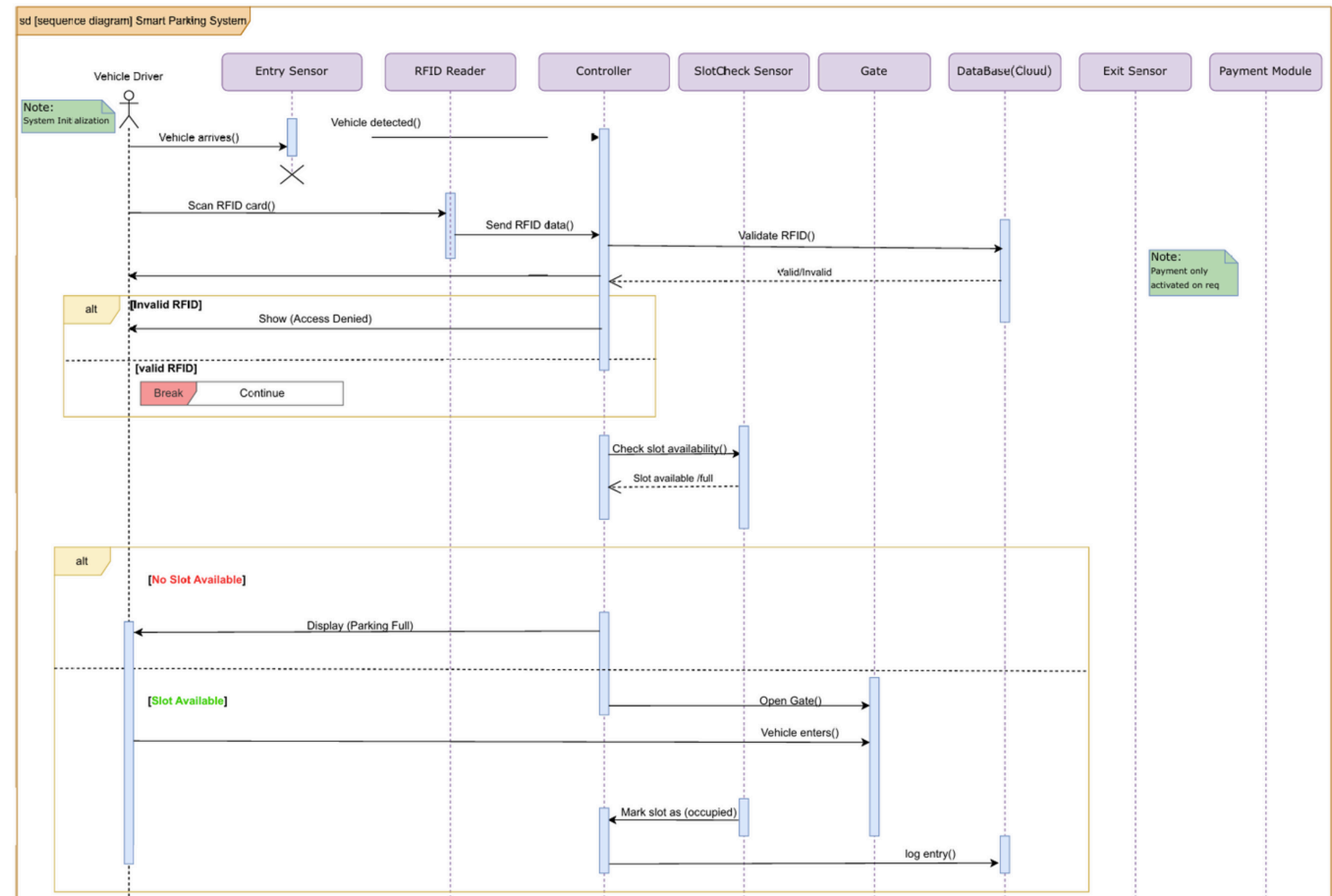Powered By Visual Paradigm Community Edition

state machine diagram

1. System starts in idle mode.
2. Vehicle is detected by the sensor.
3. RFID card is scanned.
4. Access is validated.
5. Gate opens and closes.
6. Parking slot status is checked and updated.
7. Payment is done.
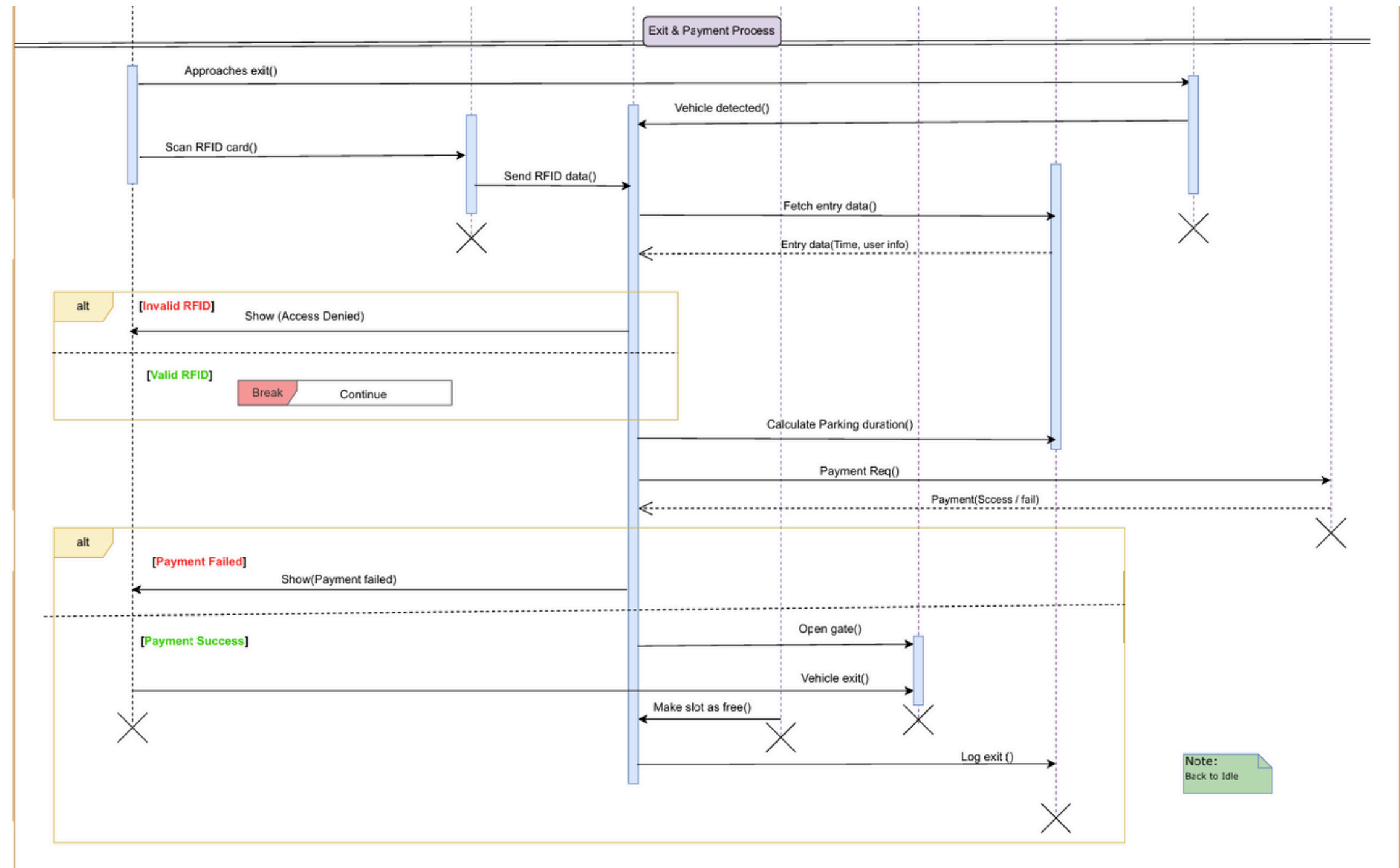8. System returns to idle mode.

sequence diagram – entry

1. Vehicle arrives.
2. RFID card is scanned.
3. System checks if RFID is valid.
4. If invalid – access denied.
5. If valid – system checks for empty slot.
6. If no slot – show "Parking Full".
7. If slot available:
8. Gate opens.
9. Vehicle enters.
10. Slot is marked as occupied.
11. Entry is logged

sequence diagram - exit

1. Vehicle reaches exit.
2. RFID is scanned.
3. System checks RFID.
4. If invalid – deny exit.
5. If valid – calculate parking time.
6. Request payment.
7. If failed – show error.
8. If success – open gate.
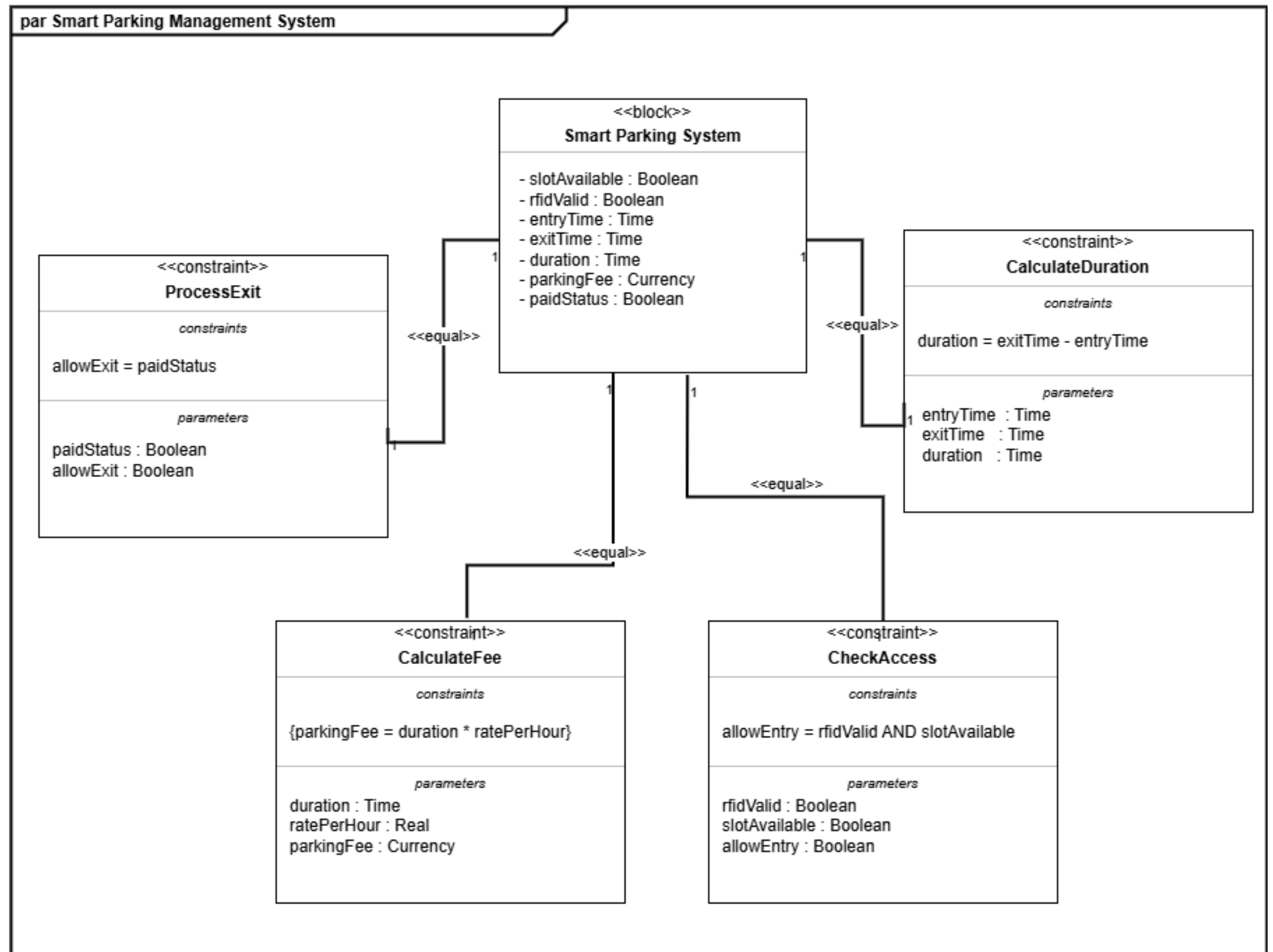9. Vehicle exits.
10. Free up slot.
11. Log exit.

4. Parametric Diagrams

parametric diagram

1. Models logical constraints and relationships within the Smart Parking System.
2. Shows how variables like access, duration, and payment are calculated and enforced.
3. Central block: Smart Parking System
4. Defines the system's internal logic and equations
5. Ensures operations are automated and rule-based

par Smart Parking Management System

<<block>>
Smart Parking System

- slotAvailable : Boolean
- rfidValid : Boolean
- entryTime : Time
- exitTime : Time
- duration : Time
- parkingFee : Currency
- paidStatus : Boolean

<<constraint>>
ProcessExit

constraints

allowExit = paidStatus

parameters

paidStatus : Boolean
allowExit : Boolean

<<equal>>

<<constraint>>
CalculateDuration

constraints

duration = exitTime - entryTime

parameters

entryTime : Time
exitTime : Time
duration : Time

<<equal>>

<<equal>>

<<equal>>

<<constraint>>
CalculateFee

constraints

{parkingFee = duration * ratePerHour}

parameters

duration : Time
ratePerHour : Real
parkingFee : Currency

<<constraint>>
CheckAccess

constraints

allowEntry = rfidValid AND slotAvailable

parameters

rfidValid : Boolean
slotAvailable : Boolean
allowEntry : Boolean

18

thank you