Software Requirements Specification

for

The Answer Management System

Version 1.0 approved

Prepared by

GROUP 3

April 11, 2025

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
	01/21/25	Completed section 2.3 and 2.4, got some progress on 2.1 and 2.2. If anyone wants to modify any 4 of these sections, that would be appreciated	1
	01/22/25	Added 1.1, amended 2.2, 2.3, 2.4, 2.6, 2.7, 3.1,3.2 and 3.3. Please delete the para in < > before submission.	2
	01/29/25	Added 5.1, 5.2, 5.3, 5.4 and 5.5.	3
	01/30/25	Added 4.1 to 4.7.3	4
	01/30/25	Added 1.5, 3.4 and 6	5
	01/31/25	Added 4.8 to 4.10, added Appendix D, amended content page	6

1. Introduction

1.1 Purpose

This software will manage the day to day functions of a clothing storefront including inventory, employees, record keeping, and marketing. Several key functions, such as a PoS system, accounting, data management, will be offloaded onto other services using their appropriate APIs.

1.2 Document Conventions

DB	Database
OS	Operating System
SQL	Sequential Query Language
PoS	Point of Sales
T&A	Time and Activity

1.3 Intended Audience and Reading Suggestions

This document is mandatory reading for owners of the business, and the developers implementing the features and functions of the software described. It is recommended reading for managers and document writers. Sequential reading is recommended to get a good understanding of the software described.

1.4 Product Scope

This software will be used to manage the day to day operations of “The Answer” clothing store. It will primarily be used to tie in disparate services for accounting, PoS system, and possibly more under one management system. It will handle its own record keeping for sales, and inventory, and also manage employees. It will also have features to manage promotional marketing for sales events.

Below is a list of what will be handled by the software itself, and what services it will consume.

Management Software	**Services**
Inventory	PoS
Sales records	Accounting
Employee management	Supply
Marketing	

1.5 References

Oracle Cloud Infrastructure (OCI). (n.d.). *Explore the Features of Oracle Database 23ai.* https://www.oracle.com/database/23ai/#security

2. Overall Description

2.1 Product Perspective

This is a new self contained software that utilizes other services through their respective APIs to enhance its own features.

2.2 Product Functions

This software's primary functions are as defined below:

- Collect and store data about the inventory of the store
- Archive records made through transactions
- Utilize other services through APIs for supply management, accounting, and PoS systems

2.3 User Classes and Characteristics

Employees

Will use this software to input data, keep track of inventory and generate inventory reports and market the product

Managers

Review and approve inputs or amendments made by employees, manage employees such as employee list, attendance,clock in time, shift plans, etc)

Developers

Will be able to modify, test and program features for the software

2.4 Operating Environment

The application will exist on an Arch Linux based cloud server infrastructure. It will serve HTTP through Nginx processed by Python, and using Oracle as a database engine, along with Amazon S3 for long term data storage and backups.

2.5 Design and Implementation Constraints

1. Integration with Third-Party API's
2. Regulatory and Compliance Requirements
3. Database and Storage
4. Programming Language and Frameworks
5. Security Considerations
6. Scalability
7. Performance Requirements
8. Design Conventions and Standards

2.6 User Documentation

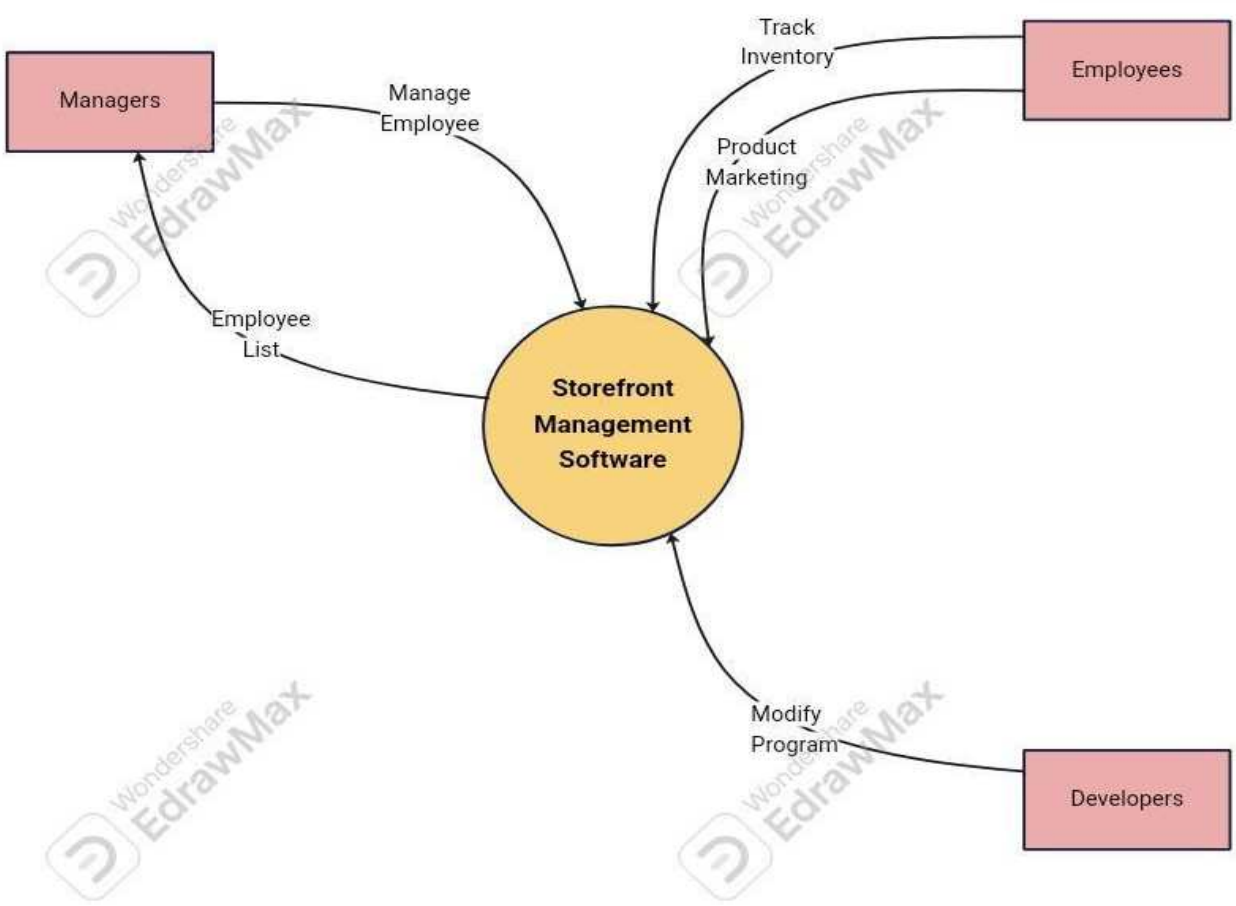
A user manual will be delivered in a word document, available online, and integrated into software.

2.7 Assumptions and Dependencies

The inventory component of the software is a perpetual inventory system which will be achieved by linking to a POS system. All data in the software process will be stored in a SQL server. The employee component of the software will be connected to a T&A system to collect employee attendance date and time.

3. External Interface Requirements

3.1 User Interfaces



3.2 Hardware Interfaces

End-users will need a computer with internet capability to operate the software.

3.3 Software Interfaces

- *a POS system*
- *a SQL server*
- *a T&A system*

3.4 Communications Interfaces

- HTTP - End-user can check inventory, HR administrators can update employee information Supervisors can update staff work schedule, attendance record on web browsers.

- FTP - Employee time punch record is recorded by external system and stored in FTP. This software has time attendance and exceptions inbound interface will read time punch record, then record employee's attendance and exceptions.

4. System Features

4.1 Real-Time Stock Update(FR01)

4.1.1 Description and Priority

This feature automatically updates and synchronizes stock levels with data from the POS system in real-time.

Priority: High (accurate, real-time inventory levels are critical for sales and restocking operations).

Requester: Business owner

4.1.2 Stimulus/Response Sequences

- Stimulus: The POS system reports a sale or return from a customer.

  Response: The system immediately deducts or adds the relevant quantity to the inventory and logs the change.

- Stimulus: The user initiates a manual synchronization request when the user finds the data is not updated.

  Response: The system requests the latest sales and return data from the POS and updates any out-of-sync items in the inventory.

4.1.3 Functional Requirements

REQ-1: The system shall automatically retrieve the latest sales and return data from the POS at a configurable interval.

REQ-2: The system shall log each inventory change with a timestamp and a user ID.

REQ-3: In the event of communication failure with the POS, the system shall retry at a configurable interval until it successfully synchronizes.

REQ-4: The system shall generate a warning notice when synchronization fails repeatedly after a configurable threshold.

4.2 Inventory Management(FR02)

4.2.1 Description and Priority

This feature allows employees to manually manage the inventory by adding new items, deleting items, and editing existing item details.

Priority: High (Essential for keeping product information accurate and updated).

Requester: Business owner

4.2.2 Stimulus/Response Sequences

- Stimulus: An authorized user selects “Add Item” and inputs item’s details.

  Response: The system records the inputs and creates a new inventory record.

- Stimulus: An authorized user selects an existing item for editing.

  Response: The system retrieves the current item details, allowing the user to amend the record upon confirmation.

- Stimulus: An authorized user deletes an existing item.

  Response: The system prompts for confirmation and removes the item or moves it to an archive once confirmed.

4.2.3 Functional Requirements

REQ-1: The system shall provide a user interface for adding,deleting and editing inventory items with fields.

REQ-2: The system shall maintain backups of previous versions for authorized users to recover.

REQ-3: The system shall log each inventory change with a timestamp and user ID.

REQ-4: In the event of communication failure with the POS, the system shall queue stock updates and retry until successful synchronization.

REQ-5: The system shall generate an error notification if the synchronization fails repeatedly beyond a specified threshold.

4.3 Supplier Profile Management(FR03)

4.3.1 Description and Priority

This feature enables storing and managing supplier information, including contact details, purchase histories, and other relevant data.

Priority: Medium (Crucial for purchasing and supplier relationship management).
Requester: Business owner

4.3.2 Stimulus/Response Sequences

- Stimulus: An authorized user creates a new supplier profile.

  Response: The system records supplier details and stores them in a supplier database.

- Stimulus: A user updates purchase information in the system.

  Response: The system records the transaction details, updates the supplier's purchase history, and calculates any relevant metrics.

4.3.3 Functional Requirements

REQ-1: The system shall allow authorized users to create, view, and update supplier profiles.

REQ-2: The system shall store purchase details linked to each supplier.

4.4 Inventory Report and Analysis(FR04)

4.4.1 Description and Priority

This feature generates custom inventory reports to analyze stock aging, cost, stock levels, warehouse locations, and other key metrics. It also provides alerts for low stock levels.

Priority: medium (Timely and accurate reporting is essential for decision-making).
Requester: Store Manager

4.4.2 Stimulus/Response Sequences

- Stimulus: An authorized user request for an “Aging Report” from the system.

  Response: The system calculates how long items have been in stock by its stock in date and displays a report sorted by aging tiers (e.g., under 30 days, 30–60 days).

- Stimulus: Stock level of a product dropped below a configurable level.

  Response: The system automatically gives a low stock level alert to authorized users after logging in.

4.4.3 Functional Requirements

REQ-1: The system shall allow users to generate customizable reports.

REQ-2: The system shall compute inventory aging, cost based on purchase details.

REQ-3: The system shall provide cost analysis, including total stock value, cost of goods sold, and profit margins.

REQ-4: The system shall allow reports to be exported in various formats.

4.5 Employee Profile Management(FR05)

4.5.1 Description and Priority

This feature manages employees’ personal details, roles, salaries, and sets authorities for different actions within the software.

Priority: High (Crucial for controlling access and storing sensitive employee data).
Requester: HR manager

4.5.2 Stimulus/Response Sequences

- Stimulus: An authorized user creates a new employee profile.

  Response: The system records the employee's personal details, role, starting salary and access right.

- Stimulus: An authorized user updates an employee's profile.

  Response: The system updates the employee's profile and access rights.

4.5.3 Functional Requirements

REQ-1: The system shall allow authorized users to add new employee profiles with fields.

REQ-2: The system shall enable authorized users to edit or deactivate employee profiles.

REQ-3: The system shall provide role-based access control, each role has its specific permissions.

REQ-4: The system shall encrypt sensitive information.

4.6 Work Schedule and Shift Management(FR06)

4.6.1 Description and Priority

This feature creates and stores employees' work schedules and shifts. It suggests schedules based on employees' working times, availability.

Priority: Medium (Efficient scheduling improves workforce management and daily operations).
Requester:Store Manager

4.6.2 Stimulus/Response Sequences

- Stimulus: An authorized user generates a weekly schedule.
  Response: The system suggests an optimal schedule based on employees' availability and roles.

- Stimulus: An employee's availability changes or adjusts a shift.
  Response: The system updates the schedule and provides notifications if it contradicts others shifts.

4.6.3 Functional Requirements

REQ-1: The system shall allow authorized users to create, edit, and publish weekly or monthly schedules.

REQ-2: The system should provide schedule suggestions based on employee role, working hours, preferred and available time slots.

REQ-3: The system should notify overlapping or conflicting schedules to users and request for manual resolution.

4.7 Employee Time Tracking and Attendance(FR07)

4.7.1 Description and Priority

This feature records employees' clock-in/clock-out times, and manages log of leave or absence.

Priority: High (attendance and time-tracking are essential for payroll and daily operations).
Requester: Business owner

4.7.2 Stimulus/Response Sequences

- Stimulus: An employee clocks in or clocks out via logging in or logging out the system.

  Response: The system records the timestamp and attendance for the employee.

- Stimulus: An employee applies for annual/sick leave.

  Response: The system shall prompt for approval of a higher role. Upon approval, it shall update the attendance record and shift schedule, and calculate remaining leave balance.

4.7.3 Functional Requirements

REQ-1: The system shall log all users' clock-in and clock-out time with timestamps.

REQ-2: The system shall allow authorized users to edit on time punches.

REQ-3: The system shall produce reports with attendance details for further payroll calculation.

REQ-4: The system shall manage and update leave balances upon leave approvals.

REQ-5: The system should provide warning messages and log if an employee's clock in time passes his/her scheduled shift time.

4.8 Review and Approval Feature(FR08)

4.8.1 Description and Priority

This feature enforces a review and approval process whenever specific actions occur (e.g. amending inventory quantity). When a user submits a change, the system places it in a pending state until it is reviewed and approved by one or more employees with access rights to review and approve.

Priority: High (It ensures data integrity and prevents unauthorized editing or amendment).
Requester: Business owner

4.8.2 Stimulus/Response Sequences

- Stimulus: An employee attempts to edit a field (e.g. quantity, cost, description) an inventory item.

  Response: the system creates a request to a manager role and flagged for approval. It remains unchanged until approval is granted.

- Stimulus: A manager approves a pending request of editing a field (e.g. quantity) an inventory item .

  Response: The system applies the change to the inventory, logs the approval event, and notifies the requester on the approval.

4.8.3 Functional Requirements

REQ-1: The system shall provide check boxes to mark specific fields or actions requiring approval.

REQ-2: The system shall create an approval request whenever a user without approval authority attempts to perform restricted actions or amend restricted fields.

REQ-3: The system shall record approval and rejection along with a timestamp and remarks.

REQ-4: The system shall notify employees when a request has been approved or rejected with remarks .

REQ-5: The system shall apply the approved changes upon manager approval.

REQ-6: The system shall allow multiple approval levels for specific actions.

4.9 Recruitment Management(FR09)

4.9.1 Description and Priority

This feature allows HR to oversee the entire recruitment lifecycle.

Priority: low (It makes recruitment process more efficient and ensures the vacancies are filled in a timely manner).

Requester: HR manager

4.9.2 Stimulus/Response Sequences

- Stimulus: An HR staff member creates a new candidate record in the system, entering personal details and acquired skills.

  Response: The system stores the new candidate personal details and acquired skills.

- Stimulus: The hiring manager schedules an interview and updates an existing candidate's interview results.

  Response: The system records the scheduled interview details, interview results, and any comments from the interviewer(s).

- Stimulus: The candidate is hired.
  Response: The system will create a new employee record, transfer the candidate's personal details into the employee profile module, and update the candidate's status to "Hired".

4.9.3 Functional Requirements

REQ-1: The system shall provide a form or interface for HR staff to input and manage candidate personal details.

REQ-2: The system shall store interview schedules, results, and comments in a structured format for easy reference.

REQ-3: The system shall maintain a hiring stage indicator for each candidate .

REQ-4: Upon a successful hire, the system shall transfer the candidate's personal information to the employee profile module, creating a new employee record without needing redundant data entry.

4.10 Employee Self-Service Portal(FR10)

4.10.1 Description and Priority

This feature provides a user-friendly interface where employees will be able to view pay slips, update personal information, submit leave requests, check leave balance, leave internal messages, visit company policy, etc.

Priority: Low (Employees are able to handle HR related tasks and reduce workload in administration).

Requester: Sales associate

4.10.2 Stimulus/Response Sequences

- Stimulus:An employee logs into the portal to view their latest pay stub.

  Response: The system records employee's payroll information and displays it in a secure(e.g. password required).

- Stimulus: An employee applies for annual/sick leave.
  Response: An employee submits a leave request (e.g., vacation, sick leave).
  Response: The system temporarily deducts the requested days from the leave balance and mark as "pending", the system will then notify the relevant manager for approval. Upon approval, the system will notify the HR department and update the employee's leave record accordingly.

- Stimulus: An employee checks his/her work schedule or assigned shifts.
  Response: The system displays the latest schedule information, including any approved shift swaps or changes.

4.10.3 Functional Requirements

REQ-1: The system shall allow employees to securely log in and view personal information, pay slips, working schedule, leave balances and company policy.

REQ-2: The system shall allow employees to update their personal information.

REQ-3: The system shall allow employees to apply for leave and update leave balances upon approval.

REQ-4: The system shall allow employees to leave messages to individual employee as well as multiple employees for important announcements.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Requirement ID	Requirement title	Short Description	Priority	Requester
NFR-1	Performance Requirements	The system should respond to user actions (e.g., searching for an item, adding an item to the cart) within 2 seconds under normal load conditions (up to 100 concurrent users). Under peak load (up to 200 concurrent users), response times should not exceed 5 seconds.	High	Developers
NFR-2	Database Optimization	Database queries should be optimized to retrieve inventory data within 1 second.	High	Developers

5.2 Safety Requirements

Requirement ID	Requirement title	Short Description	Priority	Requester
NFR-3	Safety Requirements	The system should ensure data integrity by implementing appropriate data validation and backup/recovery mechanisms, minimizing data loss due to system failures.	Medium	Developers
NFR-4	Error Handling & User Messages	The system should handle errors gracefully and provide informative error messages to users without revealing sensitive system information.	Medium	Developers
NFR-5	Role-Based Access Control	The system should implement role-based access control to prevent unauthorized access to sensitive data and functionalities. Different user roles (e.g., employee, manager) should have different levels of access.	High	Managers

5.3 Security Requirements

Requirement ID	Requirement title	Short Description	Priority	Requester
NFR-6	User Authentication Requirements	All users should be required to authenticate themselves using strong passwords (minimum 12 characters, including uppercase and lowercase letters, numbers, and symbols) or multi-factor authentication.	High	Developers
NFR-7	Data Encryption Requirements	Sensitive data, such as customer payment information and employee personal data, should be encrypted both in transit (using HTTPS) and at rest (using appropriate encryption algorithms).	High	Developers
NFR-8	Data Privacy Compliance	The system should comply with all applicable data privacy regulations, such as GDPR.	High	Managers

5.4 Software Quality Attributes

Requirement ID	Requirement title	Short Description	Priority	Requester
NFR-9	User Interface	The system should be intuitive and easy to use, even for users with limited technical skills. A consistent user interface should be maintained across all modules of the system.	Medium	Employees
NFR-10	Maintainability and Modifiability	The system should be designed for easy maintenance and modification. Modular design, well-documented code, and automated testing should be employed.	Medium	Developers

5.5 Business Rules

Requirement ID	Requirement title	Short Description	Priority	Requester
NFR-11	Real-Time Inventory Update	Inventory levels should be updated in real-time after each sale. Automatic reordering should be triggered when inventory levels fall below a predefined threshold.	High	Managers
NFR-12	Employee Data Security	Employee data should be stored securely and accessed only by authorized personnel. Employee time and attendance data should be integrated with the payroll system.	Medium	Managers
NFR-13	Sales Record Retention	Sales records should be maintained for at least 5 years for auditing purposes.	Medium	Managers

6. Other Requirements

Oracle Database 23ai

The latest version of Oracle Database which can handle large amounts of data. It introduces a new feature, Oracle SQL Firewall, decreasing the risk of SQL injection and blocking the misuse of stolen credentials. Apart from existing object, system, and administrative privileges, it also supports schema privileges which import security by simplifying authorization for database objects to better implement the principle of least privilege.

Date Time Display

The datetime display should adopt the user's timezone. For example, the employees who work in Vancouver should check their schedule in local time (UTC-8 or daylight time UTC-7). When the administrator who works in Toronto checks the schedule of Vancouver staff the date time shown should be in UTC-5 or daylight time UTC-4.

Minimum Wage

Minimum wage with effective date is administered in the system to check the staff basic salary is not lower than the minimum wage. The system will prompt administrators when the basic salary of staff does not accord with minimum wage policy.

Termination Process

The system keeps the date of notice of termination or resignation, payment date of the last month's salary, and the termination reason of employees.

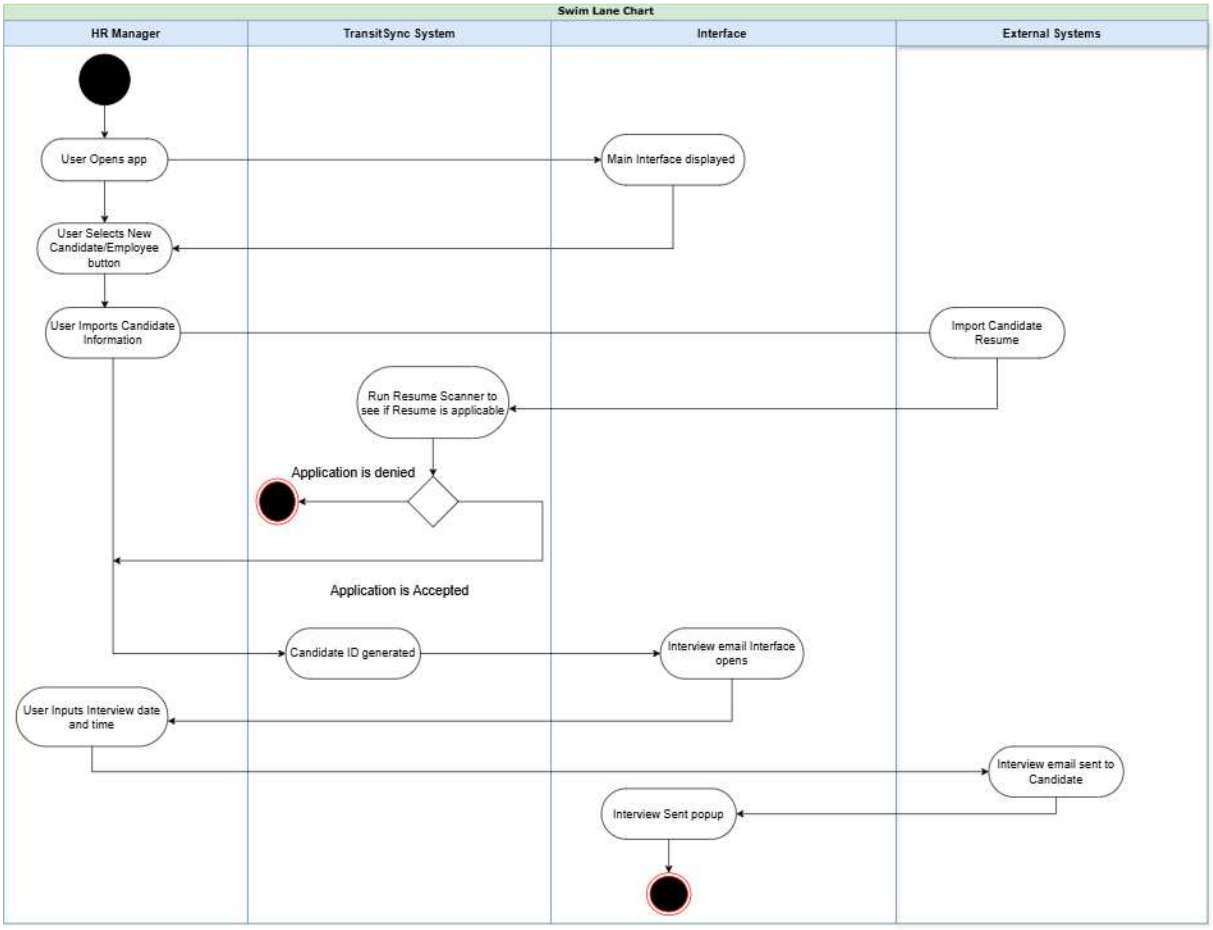
7. Use Cases

Use Case	FR ID	Actors	Description
Update inventory quantity	FR01	Sales Associate	The Sales Associate initiates the checkout process for a customer. The POS system scans the purchased items and transfers the item details to the ERP software. The system then updates the inventory levels accordingly.
Generate inventory reports	FR04	Store Manager	The Store Manager generates an inventory report by filtering stock in/out dates, supplier names, costs, etc. The software displays the filtered results. A button to export the report is available, and upon clicking it, the Store Manager is prompted to select the format and file path for saving the report.
Stock in from a new supplier	FR02 FR03	Sales Associate	The Sales Associate enters supplier details into the software to create a supplier ID and profile. The transaction details, including item name, quantity, and cost, are recorded under this supplier ID. These details are then automatically updated in the inventory records.
Hiring a candidate	FR09 FR05	HR Manager Store Manager	The HR Manager enters details of a potential candidate into the hiring module, which generates a **c**andidate ID under their profile. The HR Manager

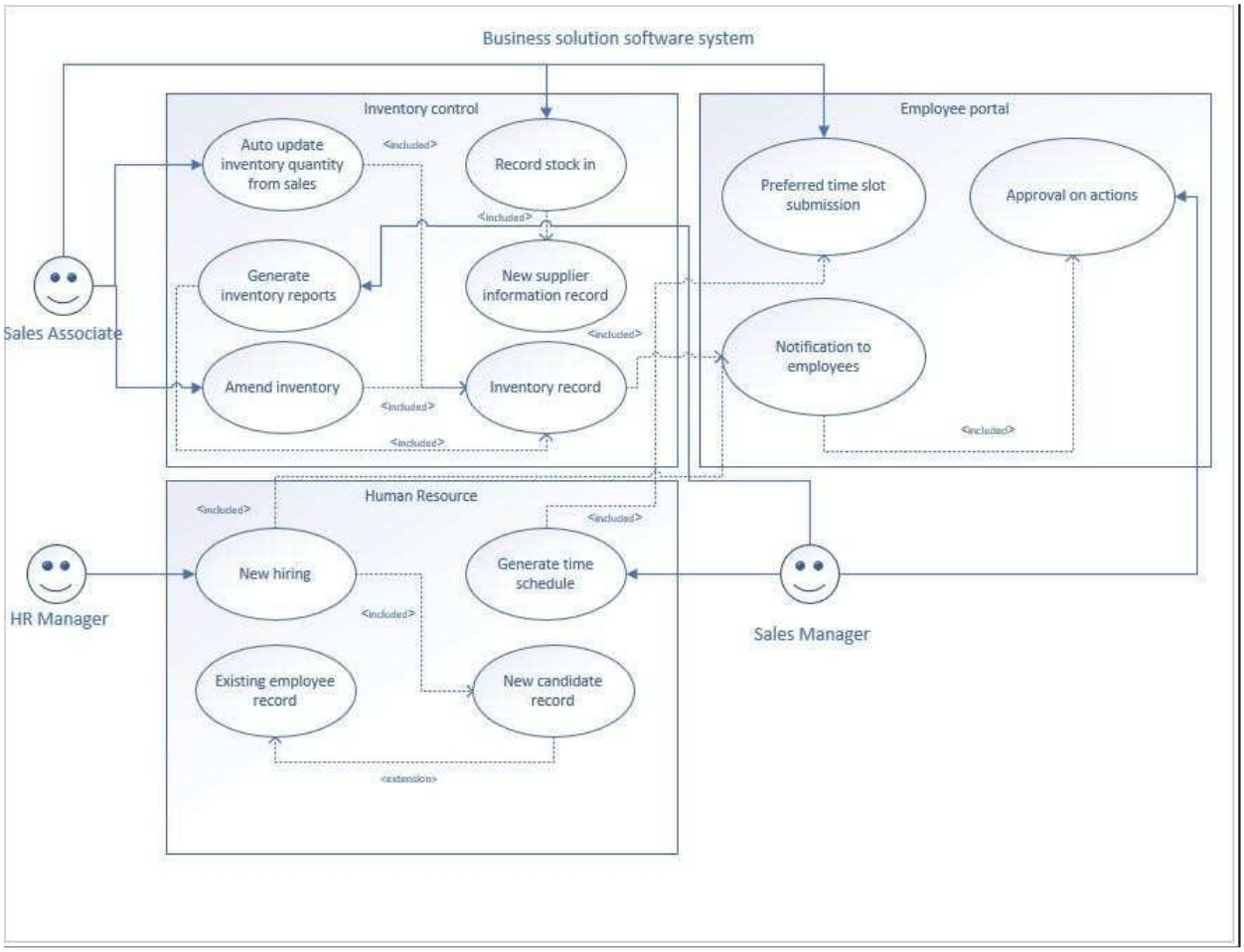
			schedules interviews by setting the interview date, the interviewer's name, and the number of interview rounds. On the interview date, the interviewer receives a notification in their account with a link to update interview comments. Once all interview rounds are completed, the software notifies the HR Manager. The HR Manager then decides whether to extend an offer. If the candidate is not selected, their profile is archived. If the candidate is hired, their profile is transferred to the employee database.
Amend of inventory	FR 08	Sales Associate Store Manager	A Sales Associate discovers a mismatch between the recorded inventory level and the actual stock quantity. The Sales Associate clicks the edit button in the inventory management section to update the stock quantity. They must input the new quantity and provide a reason for the amendment in the remarks section. A notification is sent to the Store Manager for approval. The amendment is applied only after the Store Manager approves it.
Manage employee work schedule	FR06 FR07 FR10	Store Manager Sales Associate	Every Monday, Sales Associates input their weekly availability and preferred shifts. The Store Manager accesses the work schedule section and instructs the software to generate a weekly

			schedule based on employee input. The Store Manager can manually adjust the schedule if needed. If any conflicts arise after adjustments, the system sends an alert. The Store Manager finalizes and publishes the schedule by clicking the publish button. Sales Associates receive a notification in their portal, prompting them to either accept the schedule or request a change. If an employee requests a schedule change, the Store Manager must approve or deny the request.

8. SwimLane Activity Diagram



9. Use Case Diagram



Appendix A: Glossary

DB (Database): A structured collection of data stored electronically and accessed through SQL.

PoS (Point of Sale): A system for processing sales transactions.

SQL: A language for managing and querying databases.

T&A (Time and Attendance): A system for tracking employee hours and attendance.

API (Application Programming Interface): A set of protocols that allows software applications to communicate.

FTP (File Transfer Protocol): A protocol for transferring files over the internet.

HR (Human Resources): A department/system managing employee data.

CRUD: Basic database operations (Create, Read, Update, Delete).

Appendix B: Analysis Models

Entity-Relationship Diagram (ERD)

- **Entities**: Inventory, Employees, Sales, Suppliers.
- **Relationships**: Inventory ↔ Supplier, Employees ↔ Sales.

Data Flow Diagram (DFD)

- **External Entities**: PoS System, Employees, T&A, Suppliers.
- **Processes**: Manage Inventory, Process Sales, Update Time & Attendance, Generate Reports.

Use Case Diagram

- **Roles**: Employee, Manager, Admin.
- **Use Cases**: Manage Inventory, Approve Shifts, Generate Reports.

Appendix C: Stakeholder Register

Stakeholder name	Stakeholder position	External/internal	Stakeholder Contact details	Operational/ Executive	interest (high, medium, low)
Alex Carter	Business Owner	Internal	alex.carter@theawnser.com	Executive	High
Jamie Brooks	Store Manager	Internal	JamieB@gmail.com	Operational	High
Taylor Smith	Software Developer	Internal	taylor.smith@theanswer.com	Operational	High
Riley Johnson	Accountant	Internal	riley.johnson@theanswer.com	Operational	Medium
Jordan Lee	Sales Associate	Internal	jordan.lee@theanswer.com	Operational	Medium
Gordan Brown	HR Manager	Internal	gordan.brown@theanswer.com	Operational	High
Chris Walker	POS System Provider	External	chris.walker@posvendor.com	Operational	High
Pat Green	Marketing Consultant	External	pat.green@marketingpro.com	Executive	Medium
Sam White	Compliance Officer	External	sam.white@compliancecheck.com	Executive	High

Appendix D: Interview Questions

Questions	Stalkholer	Answer
1) How frequently do you need inventory reports, and what key data should they include?	Store Manager	We need to update inventory reports weekly for analysis. Inventory report should include item ID, item name, quantity, date of stock in and location.
2) How do you currently manage shift schedules, and what improvements would you like to see in the new system?	Store Manager	I normally manage our shifts by excel, which is hard to manage and time consuming because everyone has a preferred time slot. The new system should be able to generate a suggested schedule based on the availability of our staff.
3) What level of access control do you need for employees in different roles within the system?	Business Owner	Employees should be able to access data that is relevant to their role only. Sales associates are capable of adjusting inventory quantity and the manager has the right to approve it before the change is implanted. HR managers shall have access to staff details and leave records. I should have full access to the system.
4) What is the major issue in your daily operation that you need the system to solve?	Business Owner	Managing a real-time inventory is hard for me without the help of technology. It is crucial for daily operation as I need to keep track of the stock level. I also need a weekly inventory report to analyze what customers like.
5) What is your biggest challenge in your role as HR manager?	HR manager	Calculating leave balance and tracking of staff clock in time is time consuming. I can focus on hiring talented staff if there is a system that can do these calculations for me.

Appendix E: To Be Determined List

Final UI Design: User interface mockups still being finalized.
PoS API Details: Awaiting specifications from the PoS vendor.
Cloud Hosting: Final decision on hosting provider (AWS, Google Cloud, etc.) pending.
Security Protocols: Specific encryption methods for employee data still under review.
Scalability Requirements: Load testing to determine the number of concurrent users.

Class Responsibility and Collaboration (CRC) Cards

Inventory

Responsibilities: Track inventory, update stock, generate inventory reports.
Collaborations: `SalesAssociate` (updates stock), `StoreManager` (approves changes), `Supplier` (provides inventory details).

Supplier

Responsibilities: Manage supplier details, record new stock transactions.
Collaborations: `Inventory` (updates stock levels), `SalesAssociate` (provides transaction details), `StoreManager` (overseas supplier records).

SalesAssociate

Responsibilities: Process customer sales, update inventory, request inventory amendments.
Collaborations: `Inventory` (updates stock), `StoreManager` (approves changes), `Supplier` (provides supplier details).

StoreManager

Responsibilities: Approve inventory updates, generate inventory reports, manage employee schedules.
Collaborations: `Inventory` (manages and approves stock changes), `SalesAssociate` (reviews inventory adjustments), `Schedule` (manages shift assignments), `HRManager` (hiring and scheduling decisions).

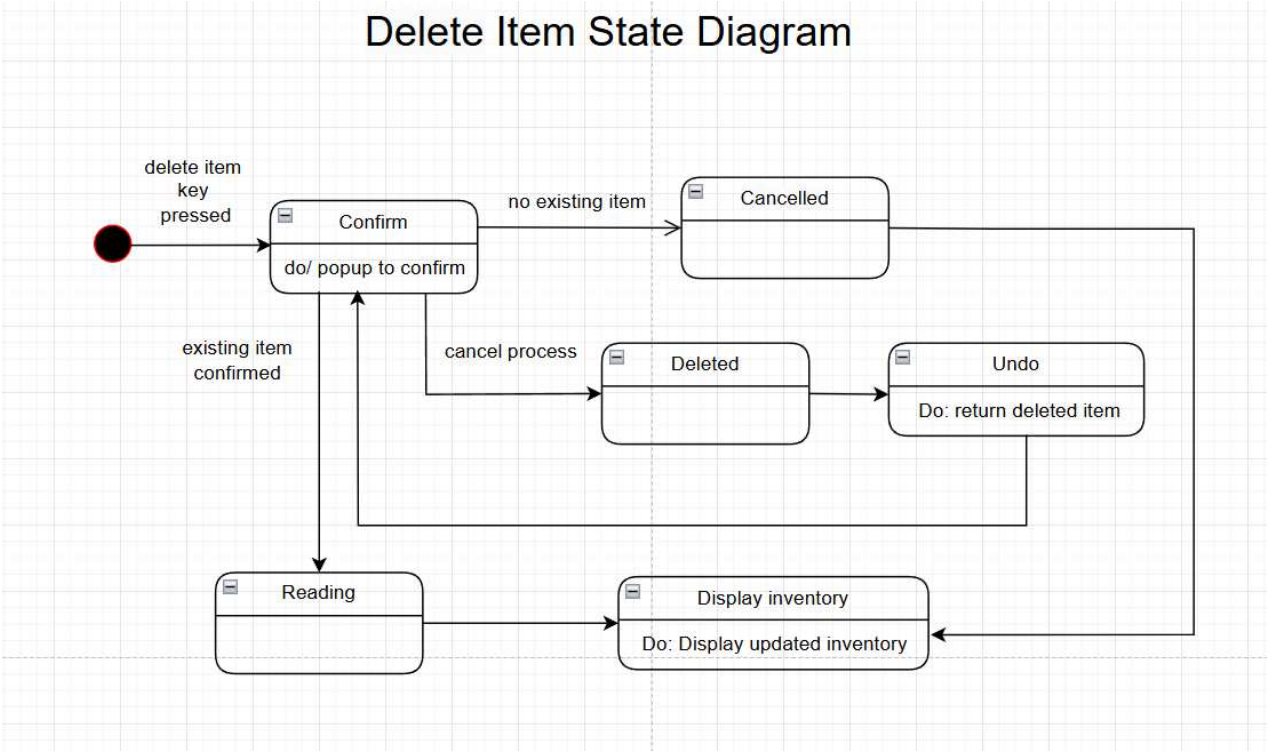
HRManager

Responsibilities: Manage hiring process, maintain employee records, schedule interviews.
Collaborations: `HiringCandidate` (updates and maintains hiring records), `StoreManager` (final hiring decisions), `Schedule` (manages employee schedules).

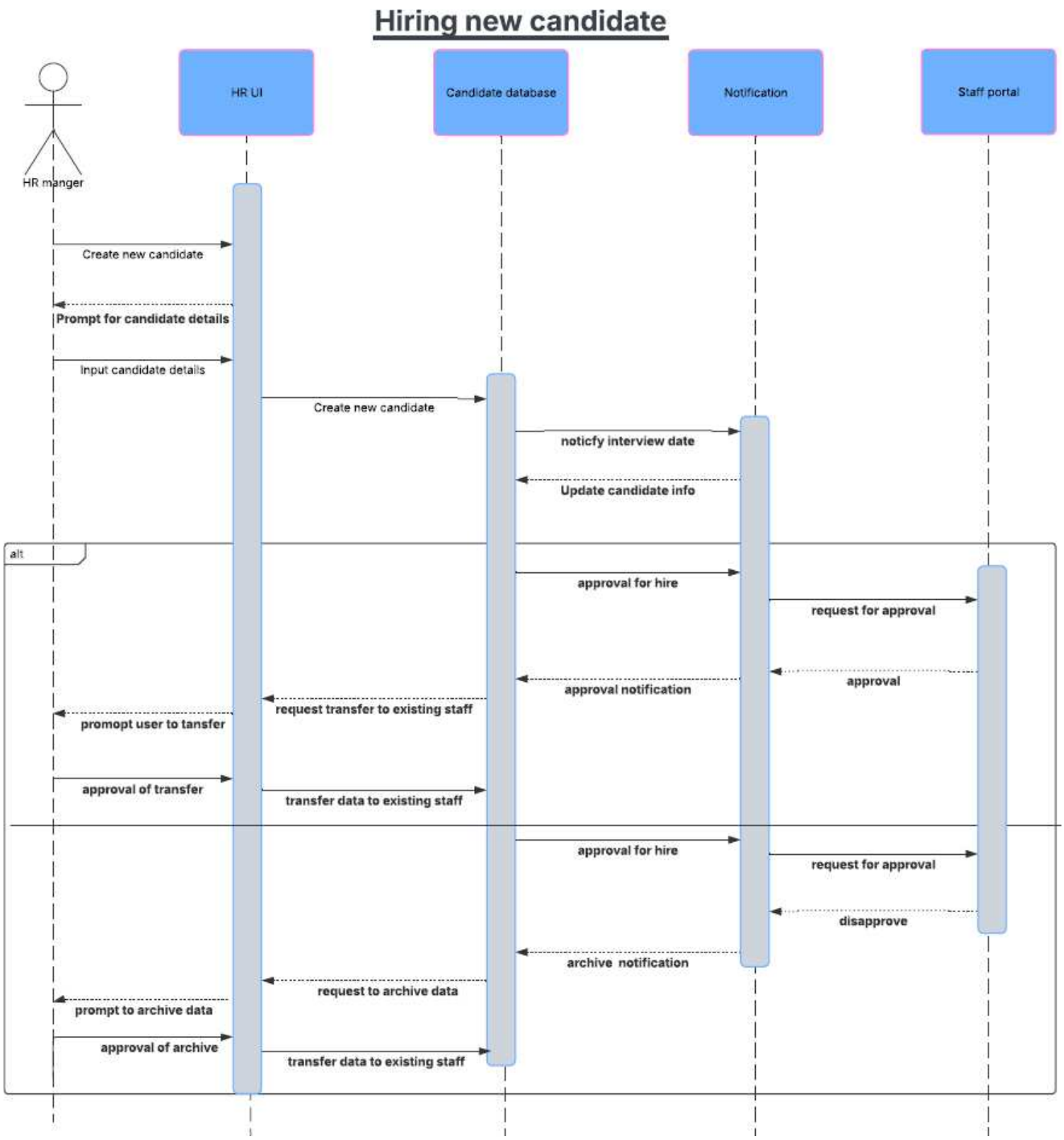
HiringCandidate

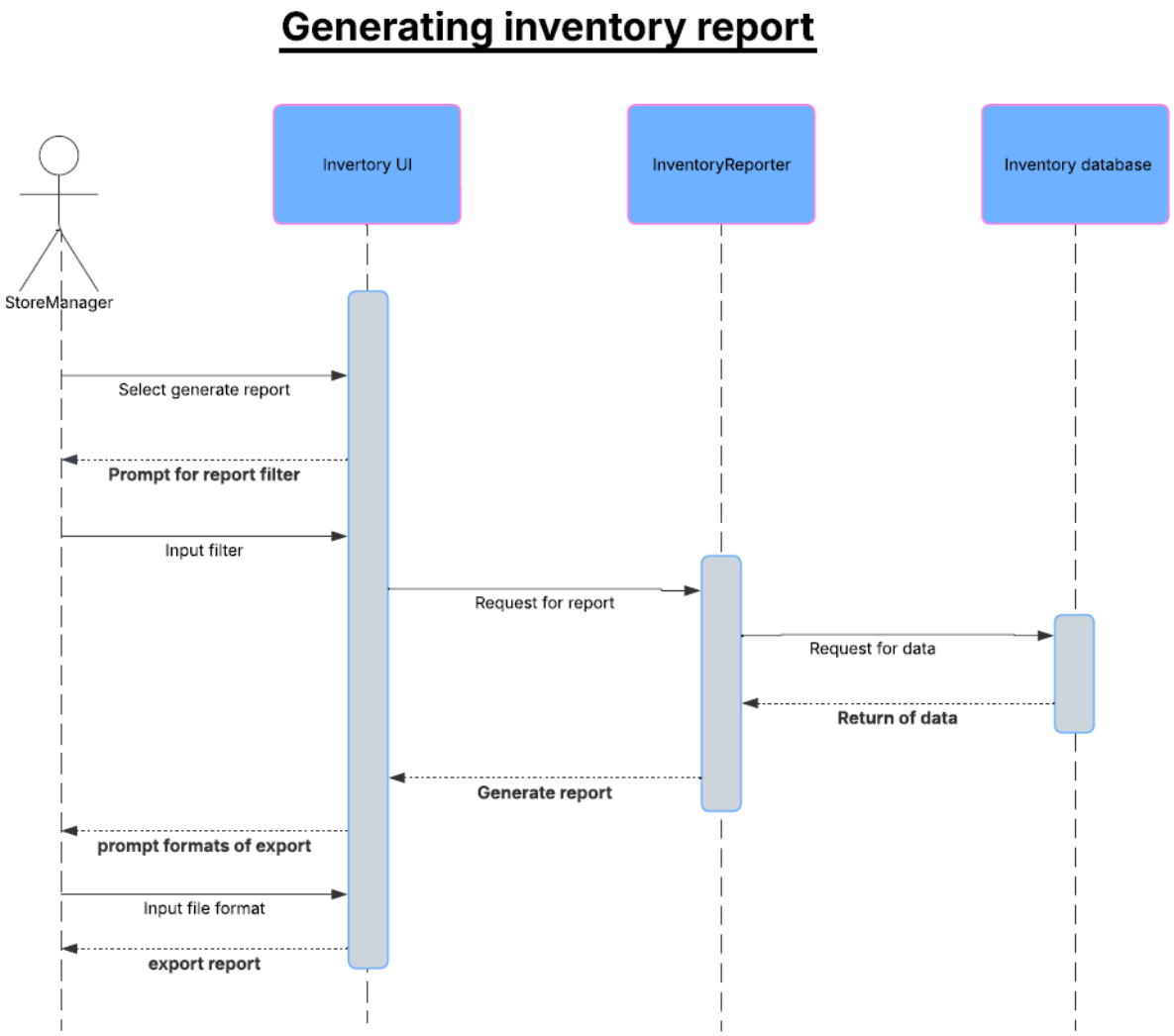
Responsibilities: Store and update candidate details, track hiring status.
Collaborations: `HRManager` (provides candidate information and receives hiring updates).

State Diagram

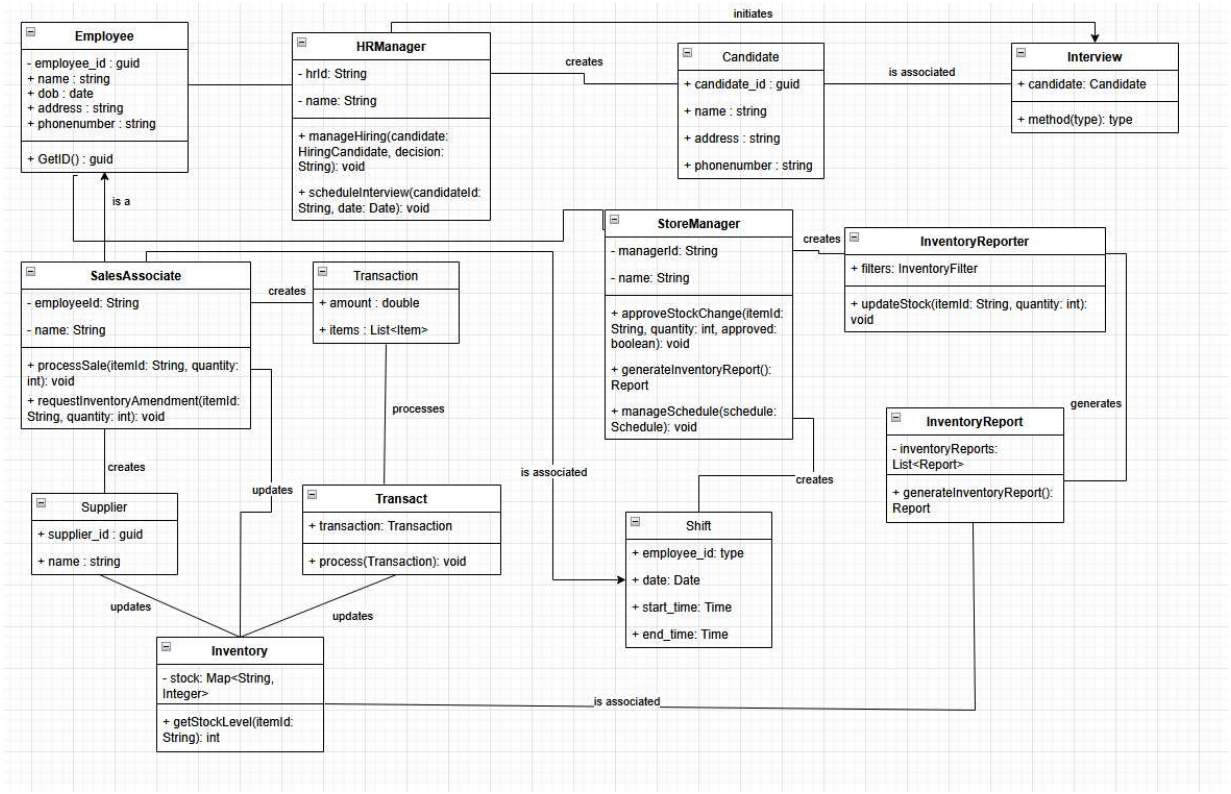


Sequence Diagrams

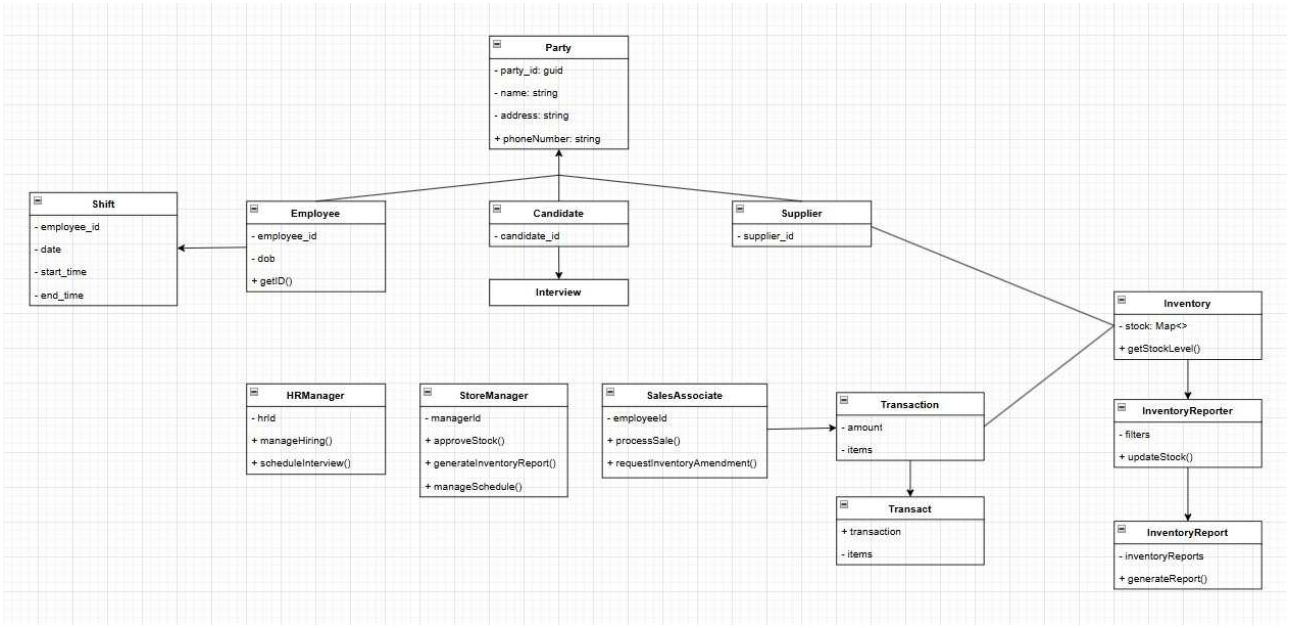



Generating inventory report
Invertory UI
InventoryReporter
Inventory database
StoreManager
Select generate report
Prompt for report filter
Input filter
Request for report
Request for data
Return of data
Generate report
prompt formats of export
Input file format
export report

Domain Class Diagram



Party Class Analysis



The Party Analysis Pattern abstracts the concept of any person or organization that interacts with the system into a single reusable superclass. Instead of separately modeling Customers, Employees, Suppliers, etc. as unrelated classes, you treat them as specializations of a generalized "Party" entity as shown above. Considering the diagram has multiple duplicate attributes such as name, address, phone number, etc. in Employee, Candidate, Supplier classes, it allows them to be defined once in the Party class and reuse them via inheritance. It reduces inconsistencies and duplication of attributes, allowing for better flexibility for the model.

Generated Code for Domain

```
class Employee:
        def __init__(self, name: str, dob: date, address: str, phonenumber: str):
        self.employee_id: uuid.UUID = uuid.uuid4()
        self.name: str = name
        self.dob: date = dob
        self.address: str = address
        self.phonenumber: str = phonenumber

        def GetID(self) -> uuid.UUID:
        return self.employee_id


class SalesAssociate(Employee):
        def __init__(self, name: str, dob: date, address: str, phonenumber: str):
        super().__init__(name, dob, address, phonenumber)
        self.employeeId: str = str(self.employee_id)

        def processSale(self, itemId: str, quantity: int) -> None:
        # Implementation for processing a sale
        print(f"Processing sale for item {itemId}, quantity {quantity}")

        def requestInventoryAmendment(self, itemId: str, quantity: int) -> None:
        # Implementation for requesting inventory amendment
        print(f"Requesting inventory amendment for item {itemId}, quantity {quantity}")


class Supplier:
        def __init__(self, name: str):
        self.supplier_id: uuid.UUID = uuid.uuid4()
        self.name: str = name


class Inventory:
        def __init__(self):
        self.stock: Dict[str, int] = {}

        def getStockLevel(self, itemId: str) -> int:
        return self.stock.get(itemId, 0)


class HRManager:
```

```
        def __init__(self, name: str):
        self.hrId: str = str(uuid.uuid4())
        self.name: str = name

        def manageHiring(self, candidate: 'HiringCandidate', decision: str) -> None:
        # Implementation for managing hiring
        print(f"Managing hiring for candidate {candidate.name} with decision: {decision}")

        def scheduleInterview(self, candidateId: str, date: date) -> None:
        # Implementation for scheduling interview
        print(f"Scheduling interview for candidate {candidateId} on {date}")


class Candidate:
        def __init__(self, name: str, address: str, phonenumber: str):
        self.candidate_id: uuid.UUID = uuid.uuid4()
        self.name: str = name
        self.address: str = address
        self.phonenumber: str = phonenumber


class HiringCandidate(Candidate):
        pass


class Interview:
        def __init__(self, candidate: Candidate, method_type: str):
        self.candidate: Candidate = candidate
        self.method_type: str = method_type


class Item:
        def __init__(self, item_id: str, name: str, price: float):
        self.item_id: str = item_id
        self.name: str = name
        self.price: float = price


class Transaction:
        def __init__(self, amount: float, items: List[Item]):
        self.amount: float = amount
        self.items: List[Item] = items
```

```
class Transact:
        def __init__(self, transaction: Transaction):
        self.transaction: Transaction = transaction

        def process(self, transaction: Transaction) -> None:
        # Implementation for processing a transaction
        print(f"Processing transaction with amount {transaction.amount}")


class StoreManager:
        def __init__(self, name: str):
        self.managerId: str = str(uuid.uuid4())
        self.name: str = name

        def approveStockChange(self, itemId: str, quantity: int, approved: bool) -> None:
        # Implementation for approving stock change
        print(f"Stock change for item {itemId}, quantity {quantity} is {'approved' if approved else
'rejected'}")

        def generateInventoryReport(self) -> 'Report':
        # Implementation for generating inventory report
        return Report()

        def manageSchedule(self, schedule: 'Schedule') -> None:
        # Implementation for managing schedule
        print(f"Managing schedule: {schedule}")


class InventoryFilter:
        def __init__(self, item_category: Optional[str] = None, min_stock: Optional[int] = None):
        self.item_category: Optional[str] = item_category
        self.min_stock: Optional[int] = min_stock


class InventoryReporter:
        def __init__(self, filters: InventoryFilter):
        self.filters: InventoryFilter = filters

        def updateStock(self, itemId: str, quantity: int) -> None:
        # Implementation for updating stock
        print(f"Updating stock for item {itemId} with quantity {quantity}")
```

```
class Report:
        pass


class InventoryReport:
        def __init__(self):
        self.inventoryReports: List[Report] = []

        def generateInventoryReport(self) -> Report:
        # Implementation for generating inventory report
        report = Report()
        self.inventoryReports.append(report)
        return report


class Schedule:
        def __init__(self):
        self.shifts: List[Shift] = []


class Shift:
        def __init__(self, employee_id: uuid.UUID, date: date, start_time: time, end_time: time):
        self.employee_id: uuid.UUID = employee_id
        self.date: date = date
        self.start_time: time = start_time
        self.end_time: time = end_time
```