Automation of J’s

Software Requirements Specification

Group 3

Ethan Butler

Tuyen Le

Ryan Mobley

Jason Tang

Justin Wade

Table of Contents

[1. Requirements Definition 1](#_Toc56026193)

[2. Use Cases 3](#_Toc56026194)

[Use Case Diagram 3](#_Toc56026195)

[Use Case Flow of Events 4](#_Toc56026196)

[3. Class Structure 10](#_Toc56026197)

[Class Diagram 10](#_Toc56026198)

[Class Documentation 11](#_Toc56026199)

[4. Data Model: Entity-Relationship Diagram 13](#_Toc56026200)

[5. Decision Tables 14](#_Toc56026201)

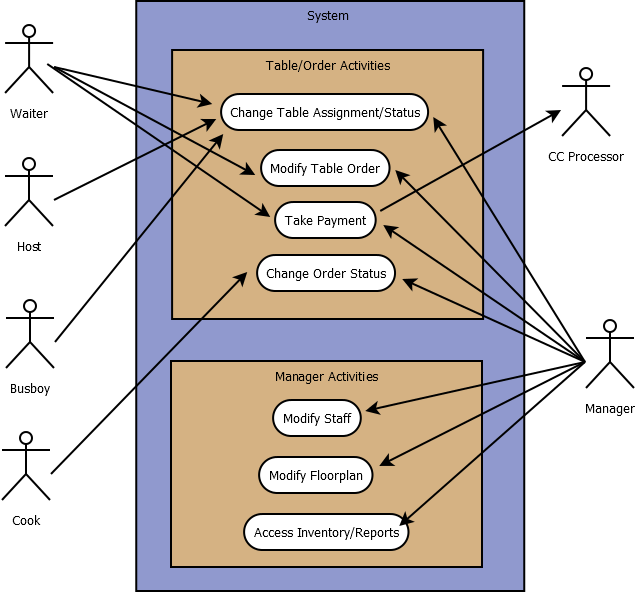
# 1. Requirements Definition

*Input-Process-Output*

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement Number | Input | Process | Output |
| 1.Login | Digits from keypad | Compare with information in system | * If matches, display menu * If not match, display message and let retry |
| 2. Menu options | Click on icons | Event handler is called | * Floor view: open the floor view * Orders: orders view * Manager: manager view * Logout: Return to login screen |
| 3. Floor view | * Table navigation * Toggle table numbers | Accept changes made | View of the floor after changes |
| 4. Customer order entry | * Item by names and quantity * Notes on order * Submit request on order | Accept order or reject order | * Display confirmation message if accept * Show reason of rejection if occurs |
| 5. Payment | * Orders’ detail * Customer payment information | Calculate and receive payment | Receive payment and put in system or show if there is error occurs |
| 6. Making orders | * Order’s detail | Kitchen’s staffs receive order  Prepare order by status | Order ready to be served |
| 7. Manager view | Click on icons | Event handler is called | * Floorplan editor : open floorplan editor screen * Inventory: Inventory view * Employees: Display staff * Reports: Display report |
| 8. Inventory | * Amount in stock, amount need, and price per unit * Submit changes | Calculate the amount to purchase and the cost | Display the changes and cost |
| 9. Floorplan editor | * Numbers of table and their position/size | Staff make change based on system | Changes made and display floorplan |
| 10. Shift | * Make schedule for shifts * Add/remove employee * Edit position | Changes are saved into the system | * Send message informing their schedule * Display changes |
| 11. Staff profile | * Edit information * Delete staff | Changes are saved into the system | Display changes |
| 12. Reports | * All orders detail * Income, expenses | Calculate profit | Display report |

# 2. Use Cases

## Use Case Diagram



## Use Case Flow of Events

**1.0 Flow of Events for Change Table Assignment/Status Use Case**

**1.1 Preconditions**

A shift is currently in progress in the restaurant, and employees are assigned to the shift.

**1.2 Main Flow**

The use case begins when a host, waiter, or busboy logs in to the system. The system verifies that the login is valid (E-1). In the Floor View, the employee selects a table, receiving relevant options to modify the table status (E-2). Based on the assigned role of the employee, the system prompts them to select the desired activity: Host – Assign Table, Modify Assignment; Waiter – Mark Dirty; Busboy – Mark Clean.

If the activity selected is “Assign Table”, the S-1: Assign to waiter subflow is performed.

If the activity selected is “Modify Assignment”, the S-2: Assign to new waiter subflow is performed.

If the activity selected is “Mark Dirty”, the status of the table is changed to “Dirty”.

If the activity selected is “Mark Clean”, the status of the table is changed to “Vacant”.

**1.3 Subflows**

S-1: Assign to waiter

The system displays a list of waiters in the current shift. The host selects a waiter to assign them to a table. The use case is finished.

S-2: Assign to new waiter

The system displays the name of the current waiter assigned to the table and a list of the other waiters in the current shift. The host selects a new waiter to assign to the table. The use case is finished.

**1.4 Alternative Flows**

E-1: An invalid login number is entered. The user is notified with error text above the screen. The user can re-enter their login number or terminate the use case.

E-2: The client application is not able to connect with the system. An error message is displayed. The use case begins again.

**2.0 Flow of Events for Modify Table Order Use Case**

**2.1 Preconditions**

The assign table use case must be performed so that the table is assigned to the appropriate waiter.

**2.2 Main Flow**

This use case begins when a waiter logs in to the application. The system verifies that the login is valid (E-1). The waiter selects a table assigned to them, which is verified by the system (E-2). The system prompts the waiter to select the desired activity: Add Item or Edit.

If the activity selected is “Add Item”, the S-1: Add menu item subflow is performed.

If the activity selected is “Edit”, the S-2: Edit item options subflow is performed.

**2.3 Subflows**

S-1: Add menu item

The system displays a screen with the categories of menu items. The waiter selects a category. The system displays the menu items within the category. The waiter selects a menu item. The system displays the options relevant to the menu item. The waiter adds the item to the order, and the use case begins again.

S-2: Edit item options

The system displays the options relevant to the menu item with the current selections for the item. The waiter modifies the options and saves the changes. The use case begins again.

**2.4 Alternative Flows**

E-1: An invalid login number is entered. The user is notified with error text above the screen. The user can re-enter their login number or terminate the use case.

E-2: The waiter selects a table that is not assigned to them. The user is notified with an error message. The waiter logs out the current user and logs in with their correct number to reattempt the use case or terminates the use case.

**3.0 Flow of Events for Change Order Status Use Case**

**3.1 Preconditions**

The assign table use case must be performed so that the table is assigned to a waiter. The modify table order use case must be performed so that an order is received by the kitchen staff.

**3.2 Main Flow**

This use case begins when a cook logs in to the application. The system verifies that the login is valid (E-1). The cook selects an order in the kitchen, and the system displays the order detail popup. Based on the order status, the system prompts the cook to select the appropriate activity: Mark Ready or Not Ready.

If the activity selected is “Mark Ready”, the status of the order is updated to ready. Other relevant actors will see an indicator that a table has an order ready.

If the activity selected is “Not Ready”, the status of the order is reverted to in progress. The order ready indicator is no longer present.

**3.3 Subflows**

No relevant subflows.

**3.4 Exceptional Flows**

E-1: An invalid login number is entered. The user is notified with error text above the screen. The user can re-enter their login number or terminate the use case.

**4.0 Flow of Events for Take Payment Use Case**

**4.1 Preconditions**

The assign table use case and modify table order use cases must be performed so that the table is assigned and has a bill.

**4.2 Main Flow**

This use case begins when a waiter logs in to the application. The system verifies that the login is valid (E-1). The waiter selects a table assigned to them, which is verified by the system (E-2). The system shows the table’s bill, including subtotal, tax, and total. The system prompts the waiter to select the desired activity: Print Receipt, Add Discount, Pay with Cash, or Pay with Card.

If the activity selected is “Print Receipt”, the receipt is printed, and the use case may begin again.

If the activity selected is “Add Discount”, the S-1: Add discount subflow is performed.

If the activity selected is “Pay with Cash”, the S-2: Accept cash payment subflow is performed.

If the activity selected is “Pay with Card”, the S-3: Take card payment subflow is performed.

**4.3 Subflows**

S-1: Add discount

The system displays valid discount options for the table. The waiter selects a discount or manually inputs a percent discount. The system applies the discount to the table’s tab (E-3).

S-2: Pay with Cash

The system displays a number pad to accept the cash total. The waiter inputs the total paid (E-4). The system calculates the change required, displays this to the user, and opens the cash drawer. The use case is completed.

S-3: Pay with Card

The system prompts the user to swipe a card. The user swipes the card. The system interacts with the credit card processor to read the card (E-5). The user inputs any relevant information and finalizes the transaction.

**4.4 Exceptional Flows**

E-1: An invalid login number is entered. The user is notified with error text above the screen. The user can re-enter their login number or terminate the use case.

E-2: The waiter selects a table that is not assigned to them. The user is notified with an error message. The waiter logs out the current user and logs in with their correct number to reattempt the use case or terminates the use case.

E-3: The waiter does not have the authorization to add the requested discount or enters an invalid discount amount. The user can attempt to try again or terminate the use case.

E-4: The cash total is not sufficient to pay for the order. The user can attempt to re-enter the total or terminate the use case.

E-5: The card is declined or invalid, or the credit card processor cannot be reached. The user is notified of the error, and the use case begins again.

**5.0 Flow of Events for Modify Staff Use Case**

**5.1 Preconditions**

System setup has been performed, and a manager has a login to the system.

**5.2 Main Flow**

This use case begins when a manager logs in to the application. The system verifies that the login is valid (E-1). The manager selects Staff from the Manager screen. The system shows a list of staff members, and the manager performs the desired activity: Add Employee, View Staff Member, or Edit Staff Profile.

If the activity selected is “Add Employee”, the S-1: Add employee subflow is performed.

If the activity selected is “View Staff Profile”, the S-2: View employee subflow is performed.

If the activity selected is “Edit Staff Profile”, the S-2: Edit employee subflow is performed.

**5.3 Subflows**

S-1: Add employee

The system displays blank fields relating to an employee profile. The manager inputs data for each option and saves the new employee (E-2).

S-2: View employee

The system displays data relevant to the selected employee. The manager may choose to review the employee’s activity during the shift or select a table assigned to the employee (E-3).

S-3: Edit employee

The system displays editable fields with the existing data for the selected employee. The manager modifies the necessary data and saves the changes (E-2).

**5.4 Exceptional Flows**

E-1: An invalid login number is entered. The user is notified with error text above the screen. The user can re-enter their login number or terminate the use case.

E-2: Improper employee data has been entered. The user is notified about which fields need to be corrected. The user can correct the errors and attempt to save again or terminate the use case.

E-3: The table details cannot be viewed, as it may have been marked vacant. The user is notified of the error and the employee profile is refreshed. The user may reattempt or terminate the use case.

# 3. Class Structure

## Class Diagram

Diagram

Description automatically generated

## Class Documentation

* Application
  + Methods
    - Login(ID)
    - getReport(report)
    - getInventory()
* Employee
  + Attributes
    - active
    - name
    - ID
    - phone
    - email
    - address
    - rate
    - hours
    - authorization
    - note
    - shiftActivity
* Waiter (inherits Employee)
  + Attributes
    - assignedTables
  + Methods
    - assignTable(table)
    - getAssignedTables()
    - closeTable(table)
* Busboy (inherits Employee)
* Host (inherits Employee)
* Cook (inherits Employee)
* Manager (inherits Employee)
* Table
  + Attributes
    - number
    - state
    - order
    - position
    - size
  + Methods
    - changeState(state)
    - addOrder(order)
    - removeOrder()
* Order
  + Attributes
    - number
    - contents
    - isReady
  + Methods
    - addProduct(product)
    - removeProduct(product)
* Product
  + Attributes
    - name
    - options
    - notes
* Report
  + Attributes
    - reportType
    - date
  + Methods
    - generateReport()
    - formatReport()
* Item
  + Attributes
    - name
    - currentStock
    - neededStock
    - amountToPurchase
    - pricePerUnit
    - totalCost
  + Methods
    - updateDBEntry()

# 4. Data Model: Entity-Relationship Diagram



# 5. Decision Tables

**Decision Table for Case 1.0**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Conditions for Flow 1.0 | R1 | R2 | R3 | R4 | R5 | R6 | R7 | R8 |
| 1.1\* Is restaurant open? | No\* | --------------------------------------Yes---------------------------------------- | | | | | | |
| 1.2.S-1/S-2\*\*  Is waiter assigned? | No | Yes | No | Yes | Yes | No | Yes | No |
| 1.2.3  Is table clean? | No | Yes | No | No | Yes | Yes | No | Yes |
| 1.4.E-1/2  Valid login? | No | Yes | No | No | No | No | Yes | Yes |
| Ready to Seat? | No | Yes | No | No | Yes | Yes | No | Yes |

**Notes:**

\*: when restaurant is closed, seating a customer is impossible  
\*\*: Waiters can be delayed or replaced by another staff member

Rule 1: Whenever 1.1 “Is restaurant open?” is false or ‘No’, then the decision of seating is false regardless of other decisions. It is impossible to seat a customer when the restaurant is closed\*.

Rule 2: The ideal scenario where all conditions are met and lets a client ready to be seated.

Rule 3: 1.1 is open however due to conditions of 1.2.S-1/S-2, 1.2.3, 1.4.E-1 all being false a table is not ready to be seated.

Rule 4: Condition when floor staff has a waiter ready to be assigned, but there are no clean tables the client would not be ready to be seated.

Rule 5: Condition when a waiter is assigned, and the table is clean but invalid login. This would end up as true due to 1.4.E-1/E-2 being tasks that can be done later in the service.

Rule 6: Condition where a table is clean, but there is no waiter assigned to the table, and a valid login is not done, but 1.2.3 is true. The table will be able to be seated, but service may be delayed due to a waiter being needed at the time, and a login can be verified later in the service.

Rule 7: A condition where the waiter is assigned and a valid login by a host is done, but the table is not clean. The customer will not be seated due to the table not being clean.

Rule 8: Like Rule 6, a condition where a table is clean, but there is no waiter assigned to that customer. However, this time a valid log in is done and only an assigned waiter is needed.

**Decision Table for Case 2.0**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Conditions for Flow 2.0 | R1 | R2 | R3 | R4 | R5 | R6 | R7 |
| 2.1  Valid login for waiter? | Yes | Yes | Yes | No | No | Yes | No |
| 2.1.2\*  Retry login? | Yes | No | No | Yes | Yes | No | Yes |
| 2.2  Add Item? | Yes | Yes | No | Yes | No | Yes | Yes |
| 2.2.S-2\*\*  Edit item? | Yes | Yes | No | Yes | No | No | No |
| Place order? | Yes | Yes | No | Yes | No | Yes | Yes |

**Notes:**\*: When waiter has invalid login, they can attempt login again. This decision table is assuming it succeeds when they try again.   
\*\*: This is only appliable when 2.2 “Add item?” is true; otherwise, this condition is ignored until 2.2 is true.

Rule 1: Condition that is impossible due to 2.1.2 it is impossible to retry a login due to logging in the first time. Aside that the waiter can then proceed to add an item and make edits to it making an order placement possible.

Rule 2: Condition that accept a valid login and adds and item to an order and is editable. This makes an order placement true

Rule 3: Condition that accepts a valid login in the first time but does not place an order due to 2.2 being false.

Rule 4: Condition like rule 2, but user is asked a second time to log in and user succeeds.

Rule 5: Condition like Rule 3 where 2.2 is false therefore making an order placement false, but user logs in successfully after a second attempt.

Rule 6: Condition where login is valid first time and 2.2 is true but 2.2.S-2 is false. The order would still place due to 2.2 being independent from 2.2.S-2 but 2.2.S-2 being dependent on 2.2.

Rule 7: Condition where login is invalid first time but valid on next attempt, and 2.2 is true but 2.2.S-2 is false. The order would still place due to 2.2 being independent from 2.2.S-2 but 2.2.S-2 being dependent on 2.2.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Conditions for Flow 3.0 | R1 | R2 | R3 | R4 | R5 | R6 |
| 3.1  Valid Login? | Yes | Yes | No | Yes | No | No |
| 3.1.2\*  Retry Login? | Yes | No | Yes | No | Yes | No |
| 3.2  Is order finished? | Yes | Yes | Yes | No | No | No |
| Mark Order ready for pickup? | Yes | Yes | Yes | No | No | No |

**Decision Table for Case 3.0**

**Notes:**\*: When chef has invalid login, they can attempt login again. This decision table is assuming it succeeds when they try again.

Rule 1: Condition that is impossible due to 3.1.2 it is impossible to retry a login due to logging in the first time. Aside that the chef proceeds to mark the order finished which makes it ready for pickup by the waitstaff.

Rule 2: Condition where chef successfully log ins first try and marks order as finished to make it ready for pickup.

Rule 3: Condition where login is invalid the first time but succeeds on the next few tries. Marks order as finished and makes it ready for pick up.

Rule 4: Condition where login is successful, but chef marks order as not complete making it that the order is not ready for pickup by wait staff.

Rule 5: Condition where login is not successful the first try but works the next few times, but chef marks order as not complete making it that the order is not ready for pickup by waitstaff.

Rule 6: Condition where chef is unable to log in and therefore cannot check off on orders for pickup.

**Decision Table for Case 4.0**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Condition for flow 4.0 | R1 | R2 | R3 | R4 | R5 | R6 | R7 | R8 |
| 4.1\*  Table asks for bill? | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 4.2.1\*\*  Table pays with cash | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes |
| 4.2.2\*\*\*  Table pays with card | Yes | Yes | Yes | Yes | No | No | No | No |
| 4.2.3  Table has discount? | Yes | Yes | No | No | Yes | Yes | No | No |
| 4.2.4  Receipt? | Yes | No | Yes | No | Yes | No | Yes | No |
| Is bill paid? | Yes | Yes | Yes | Yes | Yes |  |  |  |
| Condition for flow 4.0  Continued | R9 | R10 | R11 | R12 | R13 |  |  |  |
| 4.1\*  Table asks for bill? | Yes | Yes | Yes | Yes | Yes |  |  |  |
| 4.2.1\*\*  Table pays with cash | No | No | No | No | No |  |  |  |
| 4.2.2\*\*\*  Table pays with card | Yes | Yes | Yes | Yes | No |  |  |  |
| 4.2.3  Table has discount? | Yes | Yes | No | No | No |  |  |  |
| 4.2.4  Receipt? | Yes | No | Yes | No | No |  |  |  |
| Is bill paid? | Yes | Yes | Yes | Yes | No |  |  |  |

**Notes:**

\*: if this answer is no to the question “is the bill paid?” then it is a no regardless of what happens

\*\*: If there is not enough cash than the bill is “No”

\*\*\*: If a card declines than the bill is “No”

Rule 1: Table asks for bill and pays bill by splitting between both cash and card with a discount and ask for receipt. Bill has been paid.

Rule 2: Table asks for bill and pays bill by splitting between both cash and card with a discount and do not ask for receipt. Bill has been paid.

Rule 3: Table asks for bill and pays bill by splitting between both cash and card without a discount and ask for receipt. Bill has been paid.

Rule 4: Table asks for bill and pays bill by splitting between both cash and card without a discount and do not ask for receipt. Bill has been paid.

Rule 5: Table asks for bill and pays bill by cash with a discount and ask for receipt. Bill has been paid.

Rule 6: Table asks for bill and pays bill by cash with a discount and did not ask for receipt. Bill has been paid.

Rule 7: Table asks for bill and pays bill by cash without a discount and ask for receipt. Bill has been paid.

Rule 8: Table asks for bill and pays bill by cash without a discount and do not ask for receipt. Bill has been paid.

Rule 9: Table asks for bill and pays bill by card with a discount and ask for receipt. Bill has been paid.

Rule 10: Table asks for bill and pays bill by card with a discount and do not ask for receipt. Bill has been paid.

Rule 11: Table asks for bill and pays bill by card without a discount and ask for receipt. Bill has been paid.

Rule 12: Table asks for bill and pays bill by card without a discount and do not ask for receipt. Bill has been paid.

Rule 13: Table asks for bill but do not have a payment method therefor bill is not paid

**Decision Table for Case 5.0**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Conditions for flow 5.0 | R1 | R2 | R3 | R4 | R5 | R6 | R7 | R8 | R9 | R10 |
| 5.1  Valid login? | Yes | Yes | Yes | Yes | Yes | Yes | Yes | Yes | No | No |
| 5.2  View all staff? | Yes | Yes | Yes | Yes | Yes | Yes | Yes | no | No | Yes |
| 5.3.1  View single employee? | Yes | Yes | Yes | Yes | No | No | No | No | No | Yes |
| 5.3.2  Add employee? | Yes | No | Yes | No | Yes | No | No | No | No | Yes |
| 5.3.3  Edit employee? | Yes | Yes | No | No | Yes | Yes | No | No | No | Yes |
| Adjusted staff profiles? | Yes | Yes | Yes | No | yes | Yes | no | no | No | Yes |

**Notes:**

\*: when 5.1 is false or “no” the final result will always be false.

Rule 1: Condition where the manager views all staff and individual staff. Manager adds new staff profiles and edits old staff profiles. Staff profiles have been edited.

Rule 2: Condition where the manager views all staff and individual staff. Manager edits old staff profiles. Staff profiles have been edited.

Rule 3: Condition where the manager views all staff and individual staff. Manager adds new staff profiles. Staff profiles have been edited.

Rule 4: Condition where the manager views all staff and individual staff. Staff profiles have NOT been edited.

Rule 5: Condition where the manager views all staff. Manager adds new staff profiles and edits old staff profiles. Staff profiles have been edited.

Rule 6: Condition where the manager views all staff. Manager edits old staff profiles. Staff profiles have been edited.

Rule 7: Condition where the manager views all staff. Staff profiles have NOT been edited.

Rule 8: Condition where the manager just logs in. Staff profiles have NOT been edited.

Rule 9: Condition where manager does not login making them unable to change or view profiles. Staff profiles have not been changed.

Rule 10: Impossible condition where staff was viewed, added, and edited without the login of the manager. Overall impossible, but staff profiles would be changed.