



POLSKO-JAPOŃSKA AKADEMIA TECHNIK KOMPUTEROWYCH

PalaTOS **Tablet Ordering System**

Daryna Drabysheuskaya
s25580

Modeling and analysis of information systems

Final project documentation

Tutor: Mariusz Trzaska,, prof. PJATK

SOFTWARE REQUIREMENTS SPECIFICATION

1.0 INTRODUCTION

1.1 Purpose

PalaTOS is a tablet-based ordering system designed to streamline in-restaurant service operations. It enables customers to place orders and process payments directly from their tables, minimizing waiter intervention. The system improves service efficiency, reduces staff workload, and accelerates order processing.

1.2 Scope

- 1) Development of a web-based application for in-restaurant ordering and service coordination.
- 2) Implementation of a structured database for data persistence, retrieval, and administrative control.

1.3 Definitions, Acronyms and Abbreviations

- 1) **Guest** – An unauthenticated actor who may only access the system’s login and registration functionality. No application features beyond authentication are available to this actor.
- 2) **Customer** - A registered and logged-in user with full access to menu browsing, cart modification, order submission, payment processing, and feedback features.
- 3) **Employee** – An authenticated staff member responsible for managing orders, handling product data, confirming payments, and analyzing feedback through employee-specific functions.
- 4) **User** – An abstract base class used in the system’s analytical model to define shared attributes and behaviors of Customer and Employee. It does not represent a standalone system actor.
- 4) **PalaTOS** – The name of the system, it originates from the Latin term *palatus* (meaning “palate”) and appears in its accusative plural form. It also functions as an acronym, where “TOS” stands for **Tablet Ordering System**.

1.4 References

- 1) Wiegers, K., & Beatty, J. (2013). *Software Requirements* (3rd ed.). Microsoft Press.

2.0 OVERALL DESCRIPTION

Overview

PalaTOS (Palate Tablet Ordering System) is a role-based restaurant automation platform designed to streamline the ordering and service process. The system enables authenticated customers to place and manage orders directly from their tables via tablet devices, eliminating the need for waiter intervention. Employees access a dedicated interface to manage incoming orders, update product offerings, confirm payments, and track service status. The platform integrates structured role hierarchy, dynamic order state transitions, and service-based pricing strategies (e.g., holiday or regular discounts). PalaTOS improves operational efficiency, reduces service time, and enhances the dining experience through seamless digital interaction.

Features

Authentication and User Management

Login and Registration: Guests can create accounts or log in to the system. Once authenticated, users gain access to features based on their assigned role (Customer or Employee).

Profile Management: Customers and Employees are granted access to modify their personal profile data, which may include adjusting system-stored contact and identification information.

Customer Functionalities

Menu Browsing: Customers can browse menu items based on selected category. Upon choosing a product category, the system displays a list of corresponding items, including their names, prices, and images.

View Product Details: Customers can open a dedicated product details page from the menu. This view includes extended information such as product name, description and other information. From this page, items can be added directly to the cart.

Cart Operations: Customers can add selected products to a virtual cart, specify the table number for the order, adjust item quantities, and optionally include additional notes. The system automatically calculates the total price based on the selected items and quantities.

Order Tracking and Modification: Once an order is placed, it appears with the status NEW. Customers can view a detailed summary and access order-related actions. Cancellation is only permitted while the order remains unaccepted by the employee. When cancellation is triggered, the system prompts the user to either **reactivate** or **delete** the order:

- **Reactivation** restores the order to its previous state.
- **Deletion** permanently removes the order from both the customer's view and the system database.

Payment Processing: Customers can initiate payment at any order status, using available options (e.g. card, cash). Payment confirmation is handled by the employee. Upon successful confirmation, the system updates the order's payment status to PAID.

Feedback Submission: The option to leave feedback becomes available only when the order has been both **served** and **paid**. Feedback is linked to the completed order and is accessible to employees for further service evaluation.

Employee Functionalities

Order Management: Employees handle the processing of customer orders throughout their lifecycle. They can accept or cancel orders, update the order status (e.g., set to IN_PROGRESS or SERVED), and manage payment confirmation, including setting the payment status to PAID. Status-related actions apply once the order has been accepted.

Product Management:

Employees can add new products to the menu, as well as edit or remove existing entries.

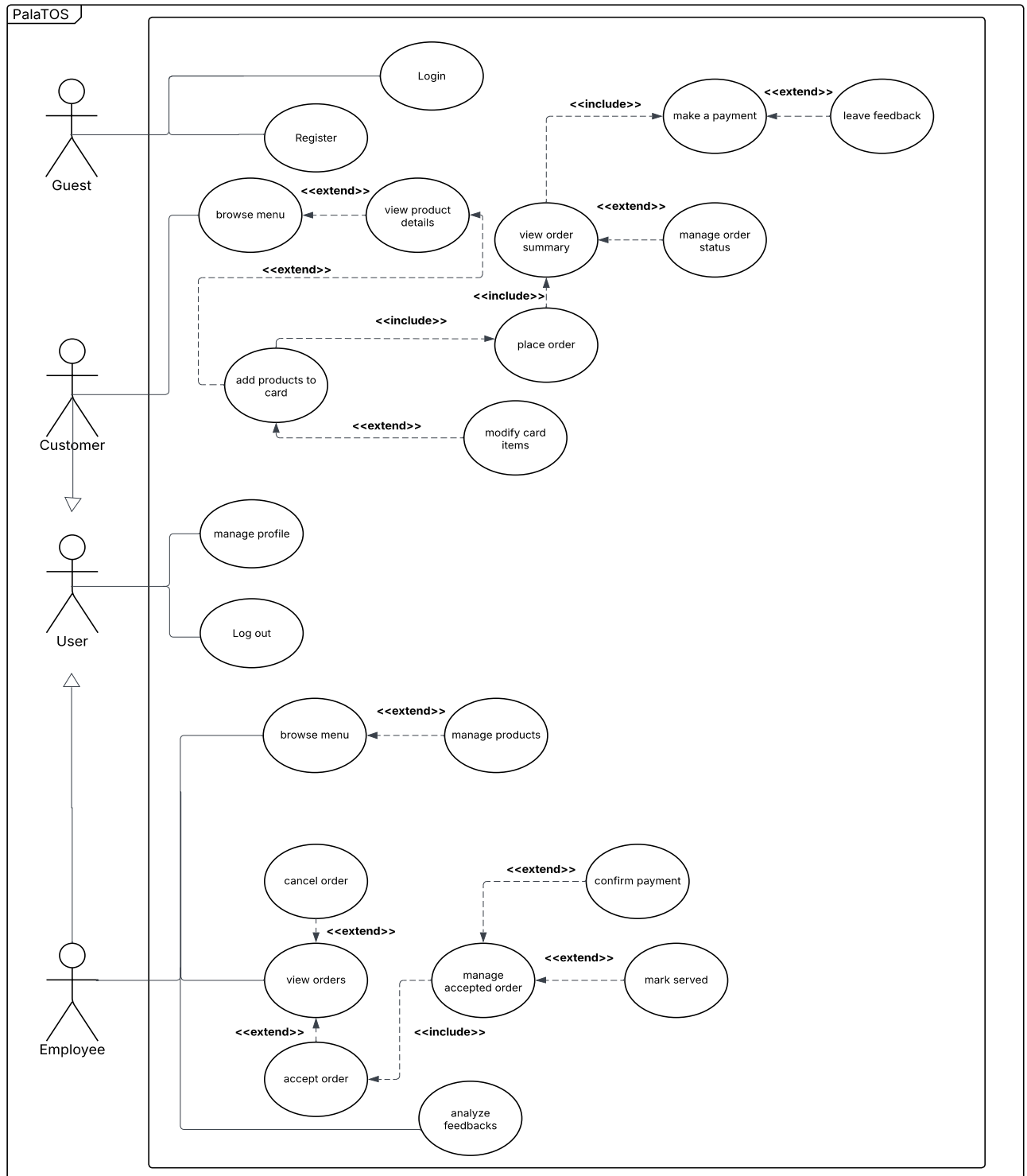
Feedback Analysis:

All submitted feedback can be accessed and evaluated by employees in order to improve service quality.

Conclusion

PalaTOS is a functional prototype designed to support basic restaurant operations through digital ordering and service tracking. While the system provides core features for customer and employee interaction, it remains an early-stage solution intended for limited environments. Further development can extend its capabilities and adapt it to the needs of larger or more complex restaurant settings.

3.0 USE CASES

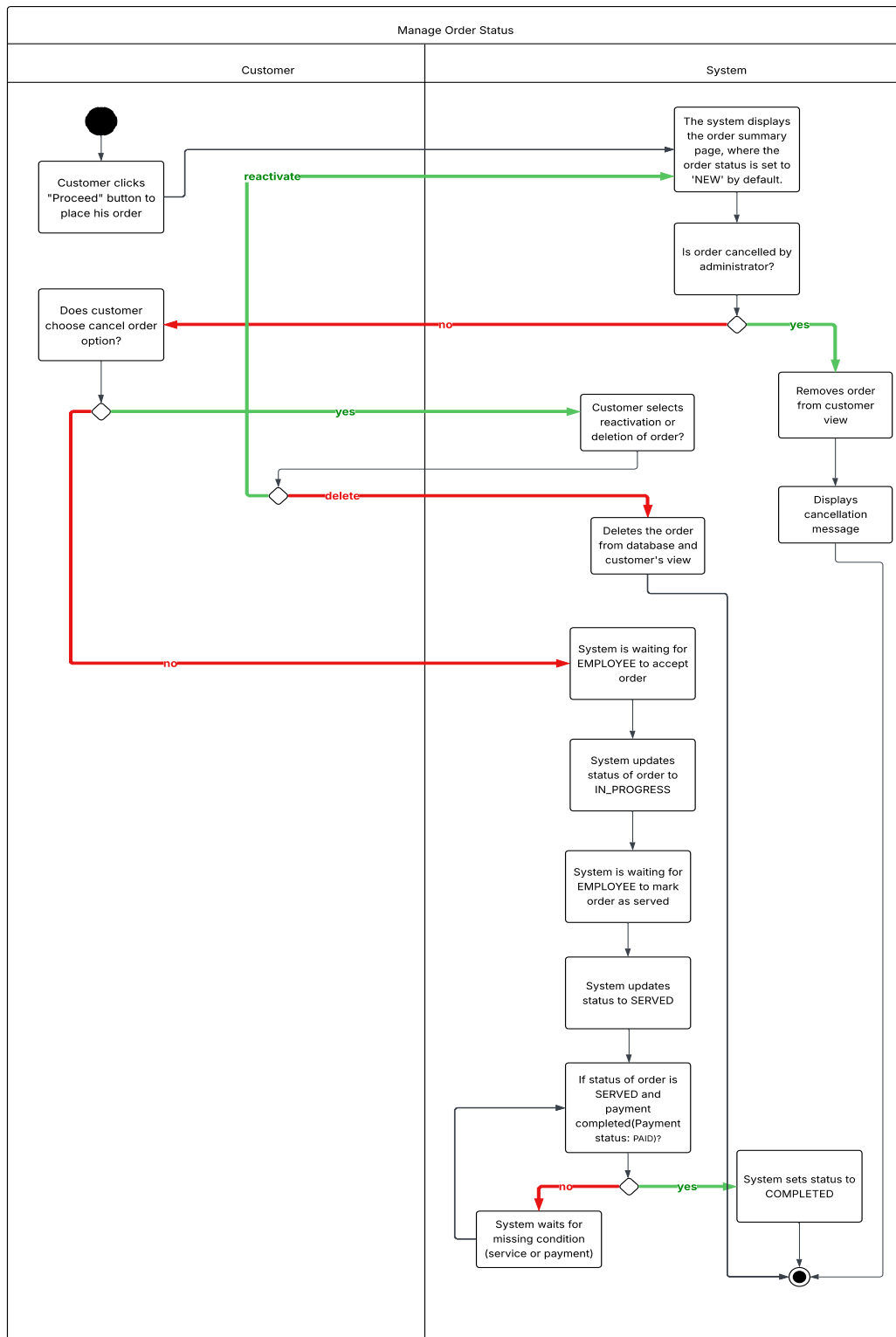


4.0 USE CASE SCENARIO

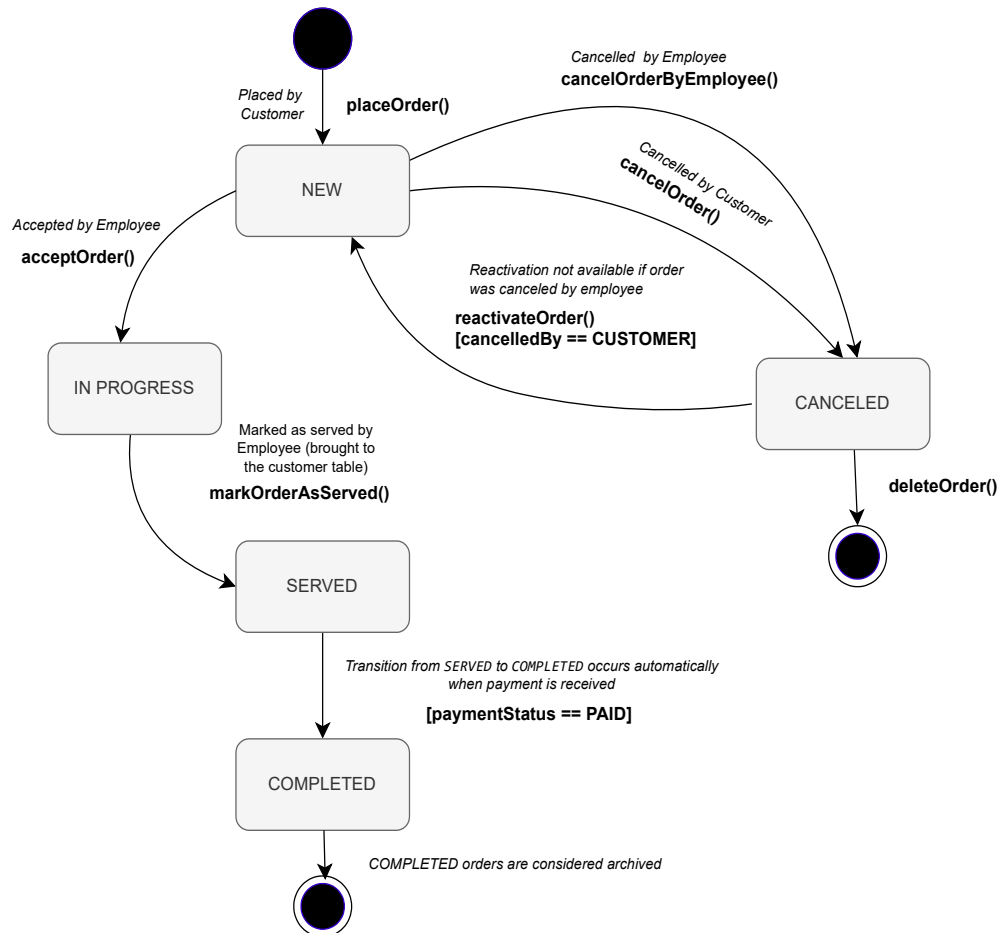
Use Case name	Manage Order Status
Actor	Primary Actor: Customer Supporting Actor: Employee
Purpose	To allow the customer to manage the status of their placed order by performing actions such as canceling, reactivating, or deleting it, and to enable the employee to process order acceptance, service, and payment confirmation in coordination with customer actions.
Trigger	The customer clicked “Proceed” to place an order.
Assumption and pre-conditions	<ul style="list-style-type: none"> • The customer must be logged into the system. • The customer has at least one existing order with status NEW. • The selected order has not yet been accepted by an employee (if cancellation is intended). • The system must be online and operational.
Basic path	<ol style="list-style-type: none"> 1. Customer places an order. 2. System displays the order summary with status NEW. 3. Employee accepts the order. 4. System updates the order status to IN_PROGRESS. 5. Employee marks the order as SERVED. 6. System updates the order status to SERVED. 7. System checks if the order is both SERVED and PAID. 8. If both conditions are satisfied, the system updates the status to COMPLETED.
Alternative flow of events	<p>1) Customer chooses to cancel the order:</p> <p>2a1. System prompts the user to either reactivate or delete the order.</p> <p>2a.2. If Reactivate is selected:</p> <p> 2a.2.1. System restores the order with status NEW.</p> <p> 2a.2.2. System returns to displaying the order summary (step 2).</p> <p>2a.3. If Delete is selected:</p> <p> 2a.3.1. System permanently removes the order.</p> <p> 2a.3.2. Flow ends.</p> <p>2) System detects that the order is not eligible for completion:</p>

	<p>2b. System detects that the order is not eligible for completion:</p> <p>2b.1. If the order is in status SERVED but not PAID:</p> <p>2b.1.1. System keeps the order in SERVED status.</p> <p>2b.1.2. System waits until payment is confirmed.</p> <p>2b.1.3. Once payment is completed, system proceeds to check completion conditions (return to the step 6 of the basic flow).</p> <p>3) Employee cancels the order (instead of accepting it):</p> <p>3a1. System updates the order status to CANCELED.</p> <p>3a2. The order becomes locked — no further actions (e.g., payment, deletion, reactivation) are possible.</p> <p>3a3. The order disappears from the customer's view upon page reload.</p> <p>3a4. The order remains stored in the system and visible only to employees for administrative purposes.</p> <p>→ End of flow.</p>
Post condition	<p>The order process ends. If status marked as COMPLETED, the customer gains the option to leave feedback.</p>

5.0 ACTIVITY DIAGRAM

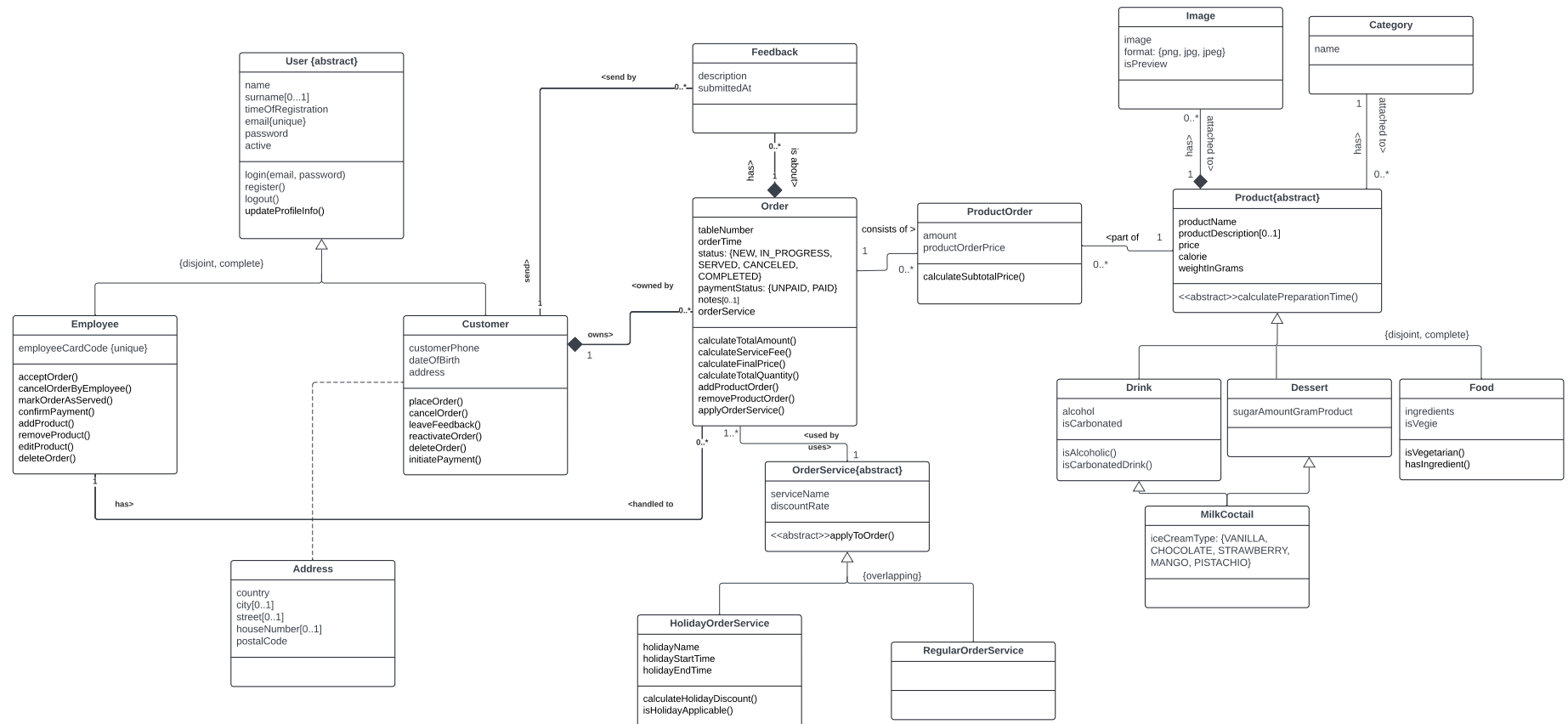


5.0 STATE DIAGRAM

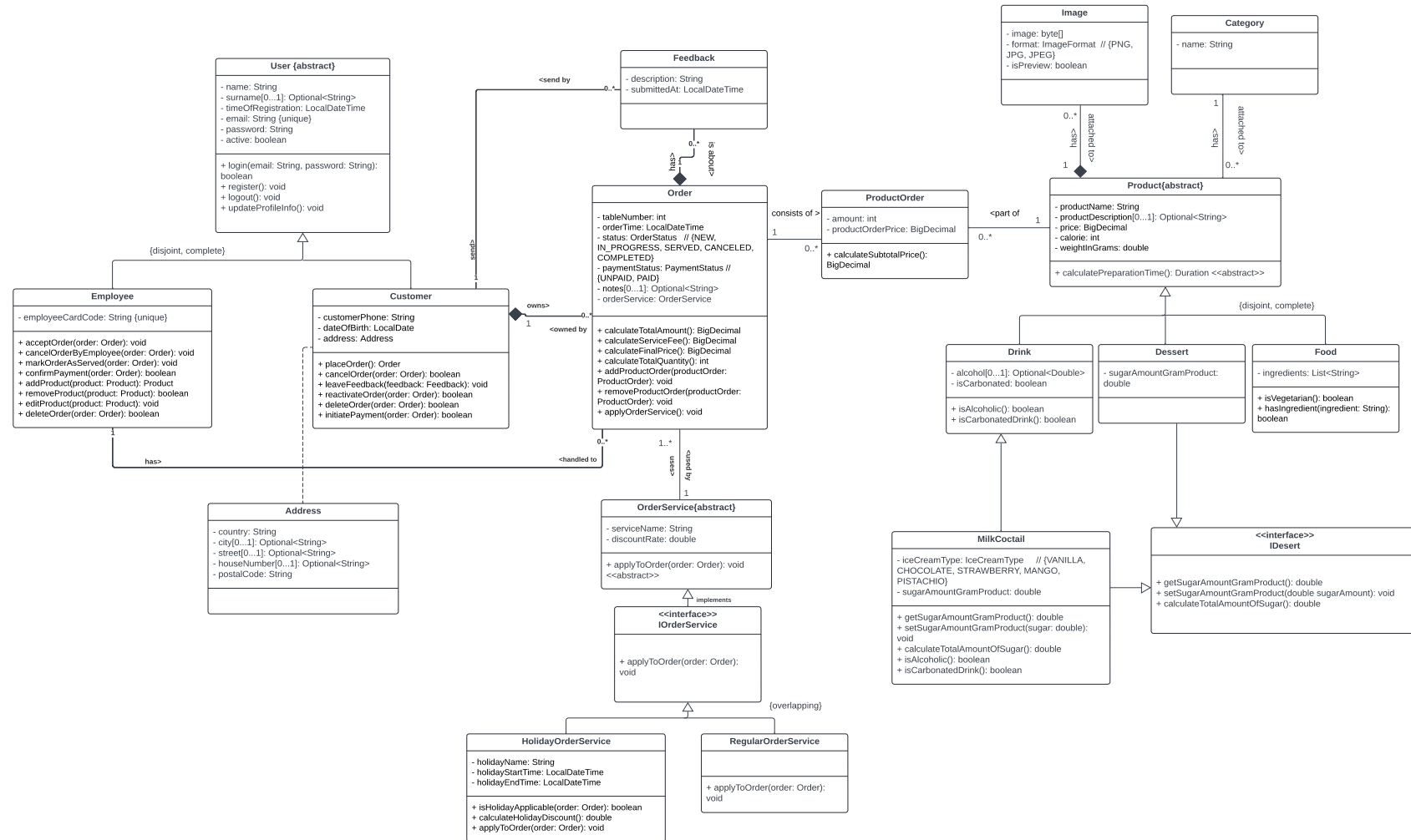


Prepared in Draw.io, <https://www.drawio.com/>

6.0 ANALYTICAL CLASS DIAGRAM



7.0 DESIGN CLASS DIAGRAM



8.0 GUI DESIGN



Navigation Bar Design

The top navigation bar serves as the primary means of system-level navigation for the PalaTOS platform. It includes the brand name **PalaTOS** positioned on the left, and a sequence of interactive buttons on the right: **Login**, **Menu**, **Card**, and **Profile**. These options allow users to access the main system modules. The currently active section is visually highlighted using a contrasting white background with black text, while the inactive buttons appear in a subdued tone to indicate availability without focus. This approach enhances usability by clearly orienting the user within the interface.

The navigation bar features a **deep red background**, intentionally selected for its association with dining and hospitality. From a psychological perspective, red is commonly linked with appetite stimulation and emotional warmth, making it particularly suitable for restaurant environments. The visual style was inspired by Korean tablet-based restaurant ordering systems, where red is frequently used to create a sense of urgency and engagement. The overall layout ensures clarity, functional grouping of navigation elements, and a strong brand presence.

A login form titled 'Authorization' centered on a light gray background. The form is a white rounded rectangle containing two input fields: 'Email' with the placeholder 'Enter your email' and 'Password' with the placeholder 'Enter your password'. Below these fields is a large black button with the text 'Log in' in white. At the bottom of the form, there is a link that says 'Don't have an account? Register'.

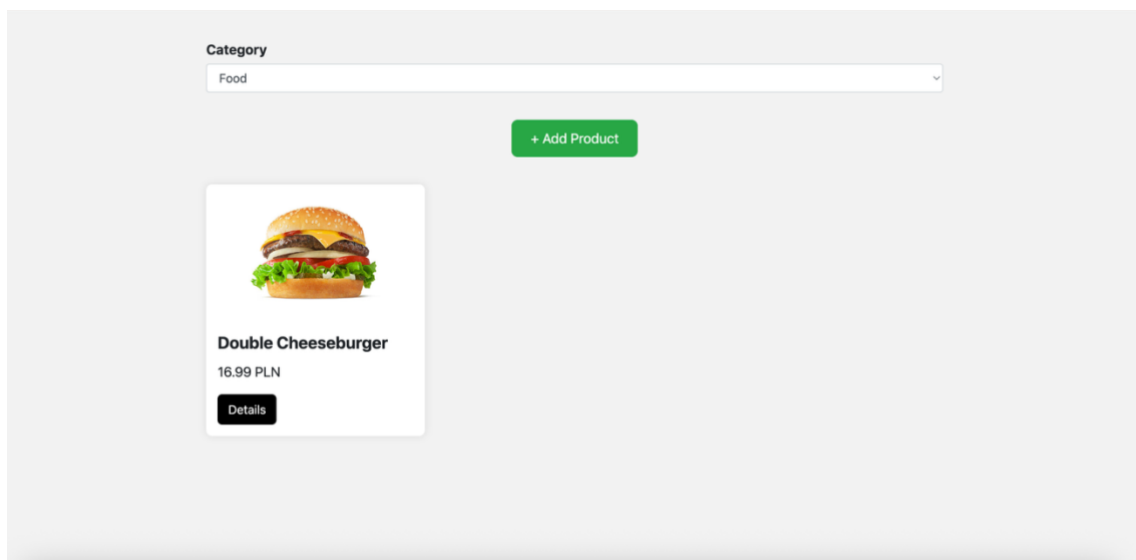
Login Page Design

The login page in the PalaTOS system is designed to be simple and user-friendly. In the center, there's a clean white box where users — either customers or employees — can enter their email and password. Below the login button, there's also a link to register if someone doesn't have an account yet.

The same page is used for both customers and employees, keeping things consistent. Once logged in, the system automatically shows different options depending on the user's role.

At the top, the red navigation bar stays visible, but until the user logs in, the other buttons like **Menu**, **Cart**, and **Profile** are shown but can't be clicked — this helps users know what features are available after login.

This layout makes it easy for anyone to sign in, with a smooth and intuitive experience.

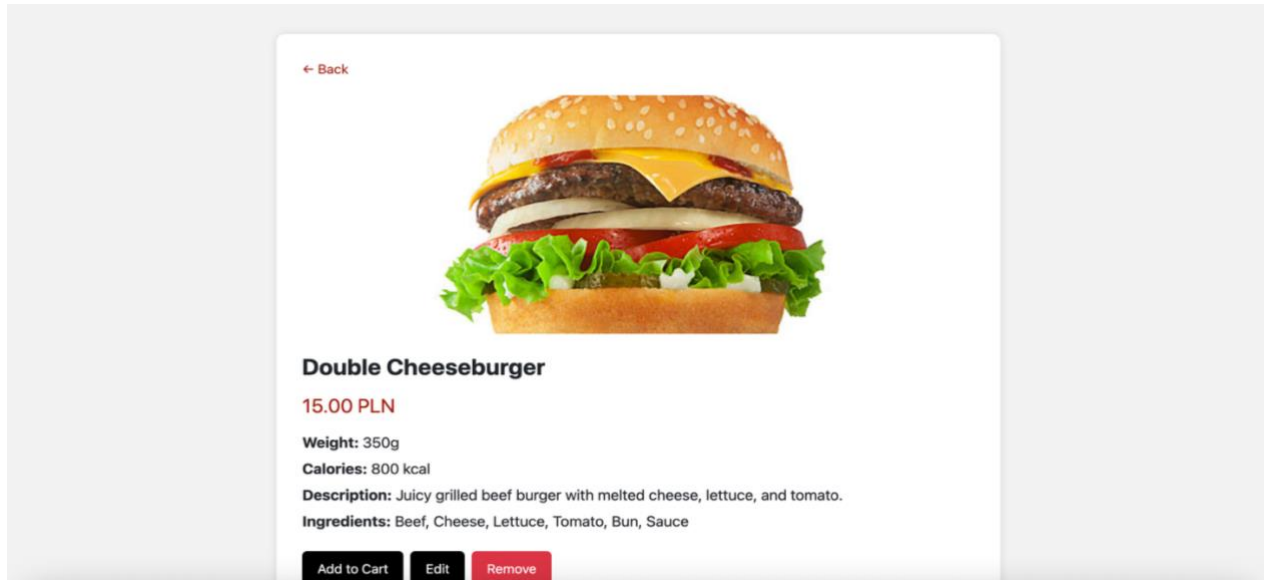


Menu Section Page

This page shows the menu with all available products. At the top, there is a dropdown where the user can choose a category (like Food or Drinks), and the page will show only products from that category. Each product is displayed in a card with a picture, product name, price, and a **Details** button.

For administrators (employees), there's an extra green **Add Product** button that allows them to add new menu items directly from this page. For customers, this button doesn't appear — they only see the products and can check their details.

The design stays the same for both types of users, which makes the system easy to use and familiar no matter the role.



Product Details Page

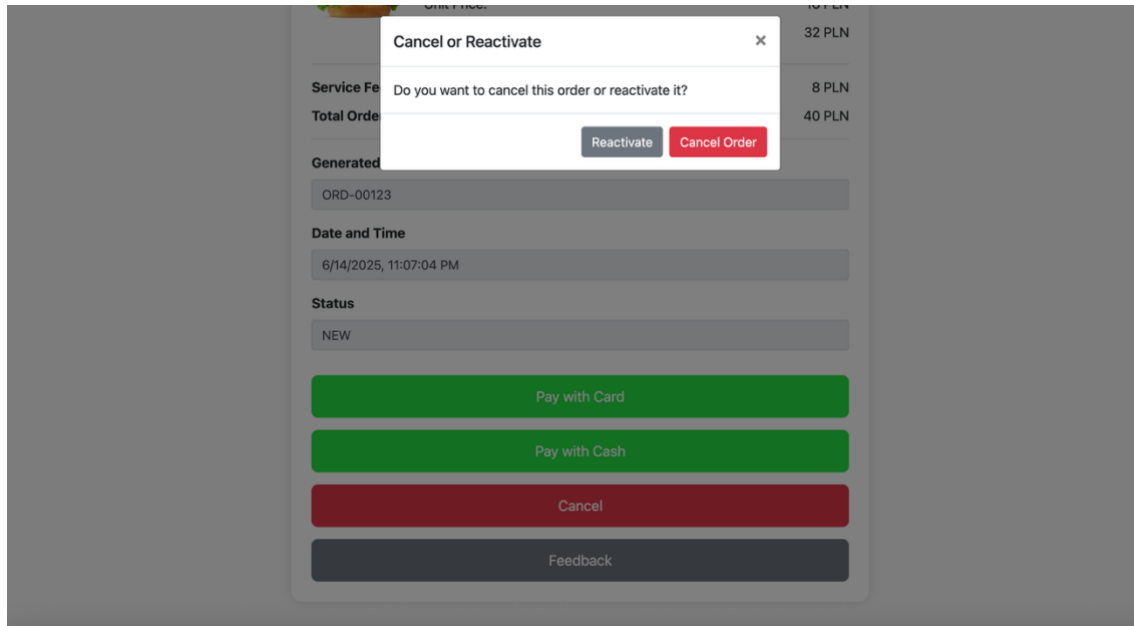
This page is displayed when the user selects **“Details”** on a product from the menu. It shows a large product image along with detailed information such as price, weight, calories, and a context-specific field depending on the type of product — for example, ingredients for food or alcohol content for drinks.

A **“Back”** button at the top allows the user to return to the general menu view.

To simplify presentation and reduce the number of screenshots, this page combines both **customer** and **employee** functionalities:

- For **customers**, only the **“Add to Cart”** button is active and visible, allowing them to add the item directly to their shopping cart.
- For **employees**, **“Edit”** and **“Remove”** buttons are shown, enabling product management directly from the details page.

The interface adapts dynamically based on the user's role, helping keep the layout consistent while ensuring that only relevant actions are accessible. This also provides a more streamlined way to showcase multiple use cases in a single view during demonstrations.



Order Details Page

After a customer places an order from the Cart Summary view, they are redirected to the **Order Details Page**. This interface presents a clear summary of the submitted order along with interactive actions tailored to the order's current state.

Displayed Information:

- **Product Overview:** Includes the image and name of the item originally selected.
- **Generated Order ID and Date & Time:** Automatically filled and displayed, these values cannot be modified.
- **Order Status:** Initially marked as NEW, and automatically updated by the system as the order progresses through acceptance, serving, and payment.
- **Pricing Info:** Reflects total value based on the previous summary (unit price × quantity + service fee).

Available Actions for the Customer:

- **Cancel:** Clicking the **Cancel** button opens a modal with two options:

Reactivate – returns the order to status NEW.

Cancel Order – permanently deletes the order.


- **Payment:**

Customers can choose **Pay with Card** or **Pay with Cash**.

Once either is selected, the system marks the order as PAID, and the buttons become disabled to prevent repeated submissions.

- Feedback:**

The **Feedback** button remains disabled until the order status reaches COMPLETED, ensuring feedback is only collected after the order has been both paid and served.



Cheeseburger

Quantity: 2

Price per Unit: 16 PLN

Total for Product: 32 PLN

Service Fee: 8 PLN

Total Order Price: 40 PLN

Generated Order ID


ORD-00123

Date and Time

2025-06-15, 01:49:06

Accept

Cancel



Cheeseburger

Quantity: 2

Price per Unit: 16 PLN

Total for Product: 32 PLN

Service Fee: 8 PLN

Total Order Price: 40 PLN

Generated Order ID

ORD-00123

Date and Time

2025-06-15, 01:49:06

Status

SERVED

Payment

PAID

Administrator Order Management View (Order Pending Acceptance)

This interface is designed for employees to view and manage individual orders placed by customers. It presents a concise summary of each order, including the item image, quantity, unit price, service fee, and final total price.

The **order ID** and **date/time of placement** are automatically generated and visible, but remain non-editable for consistency and traceability. Two primary action buttons are available:

- **Accept:** This confirms the order and changes its status to *In Progress*.
- **Cancel:** This terminates the order, changing the status to *Canceled*. Canceled orders are locked and become invisible to the customer, though still available to staff.

Once either button is clicked, the interface updates accordingly, hiding the action buttons and replacing them with further administrative fields.

Administrator Order Management View (Order Accepted / In Progress / Served)

After the order is accepted, the action buttons are replaced with dropdown selectors that allow the employee to update the **order status** (e.g., *Served*, *Completed*) and **payment status** (e.g., *Paid*, *Unpaid*).

This gives staff full control over order lifecycle management. The layout remains clear and readable, showing all crucial information (quantity, price, ID, time), while limiting editable fields to those appropriate for staff.

If feedback has been submitted by the customer, a **“Feedback”** button will appear at the bottom. Clicking it allows the employee to review any customer comments related to the order. This ensures accountability and offers insight into customer satisfaction after the order has been fulfilled.

***Note:** Due to screen size limitations, the “Feedback” button is not visible in the provided screenshot. However, it is present in the actual interface and becomes available only when the corresponding order has received customer feedback. This ensures that employees can access and review customer input after the order is marked as Completed.*

The screenshot displays a 'Personal Profile' screen with a white background and a light gray border. The title 'Personal Profile' is centered at the top. Below it, various user details are listed in a plain font. At the bottom, there are two dark gray buttons with white text: 'Log out' and 'Update'.

Personal Profile

Name: Daryna
Surname: Drabysheuskaya
Email: darina.d@example.com
Password: *****
Telephone Number: +48 507 333 987
Date of Birth: 1998-07-12
Time of Registration: 2025-03-02 14:45:33
Country: Poland
City: Warsaw
Street: Lochowska
House Number: 38B
Postal Code: 03-757

Log out Update

Personal Profile Screen

The **Personal Profile** screen displays essential information specific to the logged-in user, whether a customer or an employee. While the layout remains consistent, the fields shown depend on the user's role in the system.

This view includes key personal data such as contact details, registration information, and address or work-related attributes. At the bottom of the page, two main actions are available:

- **Update** – allows users to modify their profile information.
- **Log out** – securely ends the session.

The screen ensures that each user sees only the information relevant to their role, while maintaining a clean and accessible interface.

9.0 DISCUSSION OF DESIGN DECISIONS AND THE EFFECT OF DYNAMIC ANALYSIS

Language and Framework Choices

Java

Java was chosen as the primary backend language due to its reliability, object-oriented structure, and strong community support. Its mature ecosystem makes it ideal for building structured and scalable web applications.

Spring Boot

Spring Boot simplifies the development process by offering auto-configuration, built-in server support, and quick project setup. It allows rapid delivery of RESTful services and smooth integration with data layers, making it a strong fit for this restaurant ordering system.

Hibernate

Hibernate is used for object-relational mapping, making it easier to manage complex relationships between entities. Features like lazy loading and automatic table generation help optimize performance during development.

H2 Database

The H2 in-memory database is integrated for development and testing. It is lightweight, fast to configure, and provides easy access via a web console, which is helpful during prototyping and early-stage testing.

HTML, CSS, JavaScript

The frontend is developed using standard web technologies to ensure accessibility, responsiveness, and usability. HTML structures the pages, CSS defines the visual design (including custom styling for navigation, forms, and UI components).

Class Hierarchy and Inheritance

Flattened Class Hierarchy

The product hierarchy is designed using a unified Product superclass with multiple specialized subclasses, such as Drink, Dessert, and a hybrid type like MilkCocktail. It is designed using a flattened structure, where MilkCocktail inherits from both Drink and Dessert, enabling it to represent properties of both.

Composition over Inheritance

MilkCocktail is modeled to exhibit characteristics of both drinks and desserts. Composition is used to combine functionalities.

Use of Enum Types and Optional Attributes

Enumerations are employed for structured values such as Status, PaymentStatus, IceCreamType, ensuring type safety and validation. Fields such as notes, description are marked optional, reflecting real-world variability and improving data robustness.

Analysis and Modifications

Order Lifecycle State Handling:

A status enum was integrated into the Order class to represent its various states, such as NEW, IN_PROGRESS, SERVED, CANCELED, and COMPLETED. This allows the system to track the full lifecycle of an order and support transitions triggered by both customer and employee actions.

Consistency and Validation Logic

Supporting methods were added to enforce valid transitions and ensure correct behavior during operations like confirming payment, marking orders as served, or canceling/reactivating them. These help maintain data integrity in scenarios where multiple roles interact with the same entity.

Enhanced Customer Interaction

Methods like placeOrder, leaveFeedback, etc were introduced to support customer-centric features. These allow users to personalize their experience and provide feedback, which can later be analyzed by employees via the admin panel.

ORM and Class Extent

Object-Relational Mapping (ORM)

The system uses Hibernate to manage persistence and associations across domain entities such as Customer, Order, Product, and Feedback. Associations are clearly defined using annotations like @OneToMany, @ManyToOne, and @ManyToMany to maintain relational integrity between entities (e.g., orders and their items, customers and their feedback).

Efficient Data Access with Lazy Loading

To improve performance and responsiveness, the project takes advantage of Hibernate's lazy loading mechanism. This ensures that related data is only fetched from the database when explicitly required—minimizing unnecessary data transfer and reducing memory usage during session initialization.

Frontend Implementation

Web Interface and Technologies:

The frontend is developed using standard web technologies including HTML, CSS, and JavaScript. This provides a clean and responsive user interface suitable for both customer and employee roles. The design emphasizes clarity, simplicity, and ease of navigation across devices.

REST API Integration:

The frontend communicates with the backend services through RESTful endpoints. This separation of concerns supports scalability and allows the UI to stay decoupled from the business logic, enabling easier maintenance and future expansion.

Client-Side Validation and Interaction:

JavaScript is used to perform validation directly in the browser before form submissions (e.g., during login, registration, or order placement). This improves user experience by providing immediate feedback for missing or invalid fields.

Responsive Layout:

CSS media queries and flexible layouts ensure the application adjusts well to various screen sizes, making the system usable on desktops, tablets, and mobile devices. This is particularly important for in-restaurant tablet ordering or mobile usage by staff and guests.