VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

Software Engineering (CO3001)
The Dining Philosophers

Report

Restaurant POS 2.0

Advisors: Assoc Prof. Bui Hoai Thang
Mr. Bang Ngoc Bao Tam

HO CHI MINH CITY, NOVEMBER 2021

Member list

No.	Full name	Student ID	Percentage of work
1	Nguyễn Hoàng	1952255	100%
2	Trần Nguyễn Phước Nhân	1952893	100%
3	Nguyễn Chính Khôi	1952793	100%
4	Đào Tiến Tuấn	1953069	100%
5	Đinh Huy Thiện	1952462	100%

Contents

1 Context, stakeholders, expectations and scope

1.1 Context

In restaurant business, the POS systems include many services such as table reservation, food ordering, alerts, billing (credit card or cash), catalogue system and processing user database During the COVID-19 pandemic, the need of using POS systems raised dramatically because of its portability, business intelligence and reducing wasted effort and opportunity to scale a large business Moreover, it can be easily extended for further scale of the business.

1.2 Stakeholders

Managers — Either the restaurant owners managing the restaurant themselves or hired managers The requirements provided by them will be directly delivered to the developers They can provide about the feature or the UI they want to use when managing their employees.

Clerks — The environment for clerks should be professional for them to deliver the convenience and time-efficiency to the customers, which is important They should be able to tell the developers what is comfortable to use and does not waste too much time.

Customers — Despite not working directly with developers, they can still contribute to the development of the software The software will gather information from customers (with their permission) to improve the experience Feedback features and occasional surveys will also be included to help even further.

1.3 Expectation

For optimizing the workflow of the restaurant and its service quality, there should be an intermediate and automated worker to handle repetitive tasks with time-efficiency and high precision This software is expected to be a web-based Point of sale (POS) system that solves the fore-mentioned problem and it follows the business flow as described in Figure 1.

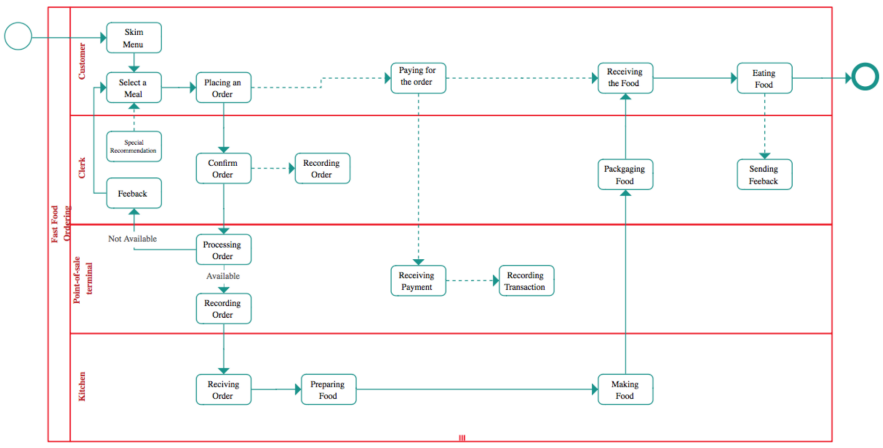

Figure 1: Customer-drawing workflow

Besides the previous requirement there are customized requests that have been

made by the owners of the restaurant These will be taken into consideration and implemented into the software:

- The Customers should be able to interact with the Clerks without direct contact.
- Without having to install apps, there should be QR code and some other Web features for the customers.
- Customers can access this software on most platforms like mobile devices, tablets, laptops or even personal computers.
- The flexibility should be considered for multiple restaurants can be used if there is an expansion of quantity.
- The transaction rate measured recently is approximately 300 orders/day.

The eye-catching features that defines the software quality is the graphic design, which is also provided by the client This software User Interface (UI) will mainly be based on these Figures below if there are no changes in the future:

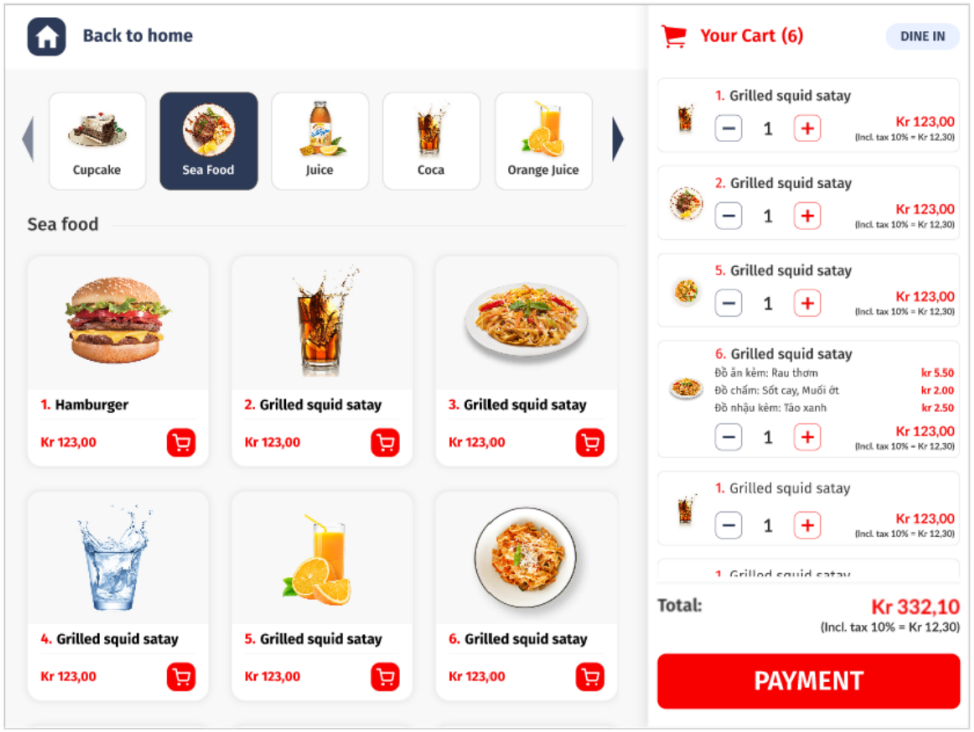

Figure 2: The menu screen

This is the main menu for both take-away and dine-in customers.

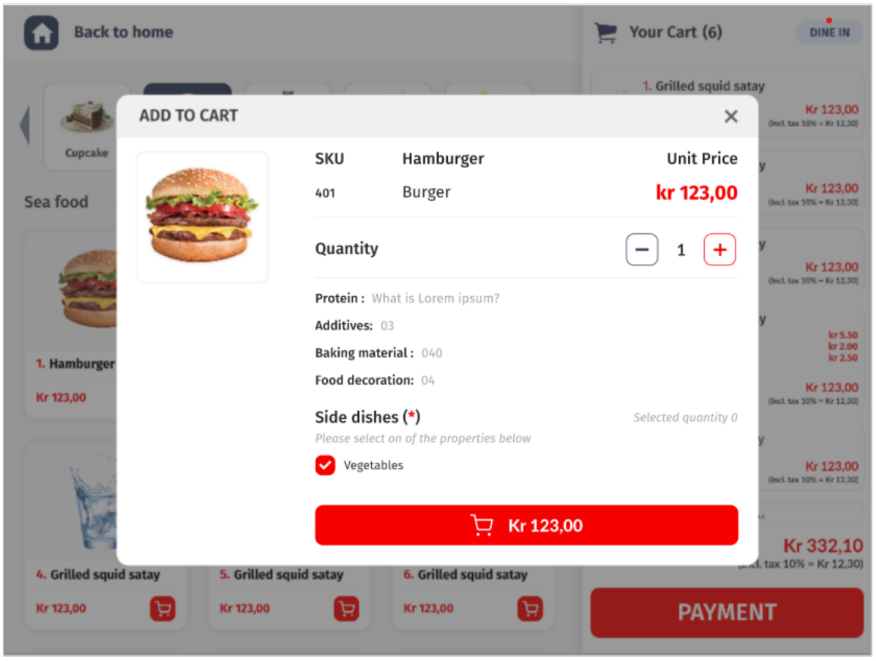

Figure 3: The detail item screen

This is the detail item screen when clicking to select an item in Figure 2 When changing the Quantity or click on the shopping button, returning to the Menu screen and update the sum-up Your Cart in the right hand side.

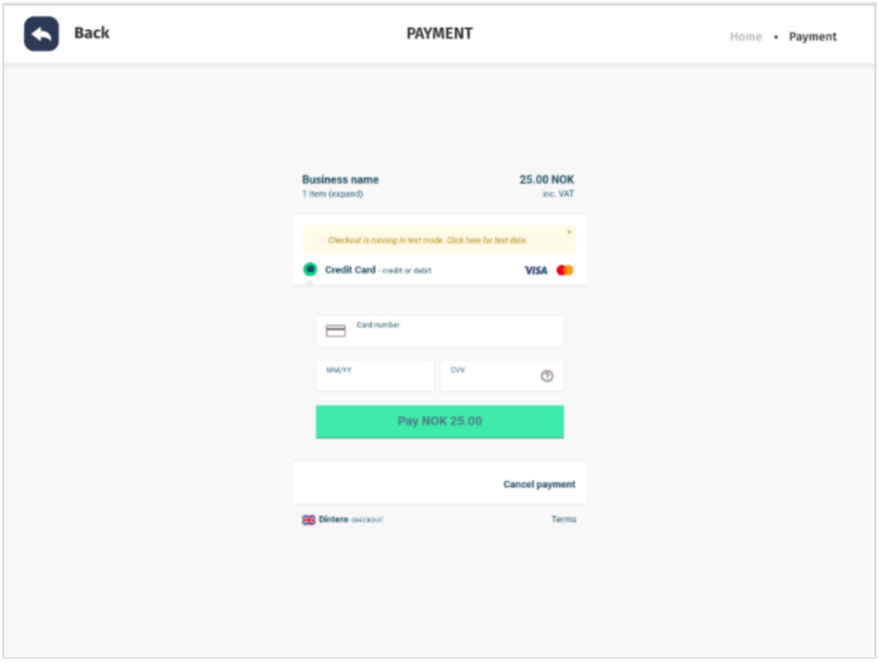

Figure 4: The payment page

1.4 Scope

Specifically, the system provides an automated price list and allows creation and printing of receipts based upon transactions of clients. The system also enables creation and automatic updates of inventory whenever new entries or deletion of entries are made. Only the management has access on the database for security reasons. User accounts will be created to handle the daily transactions of the business, like payments and inventory. Only those with such accounts can be able to access the system. Also, the system will handle credit card transactions, support take-away options and also minimize work efforts across many work phases.

After hitting payment in Figure 2 will take the customer to this page for choosing the payment method or inserting credit card information.

2 Requirements

2.1 Functional Requirement

The system should not allow direct contact between Clerks and Customers so there are separable interfaces for each actors.

1. Customers

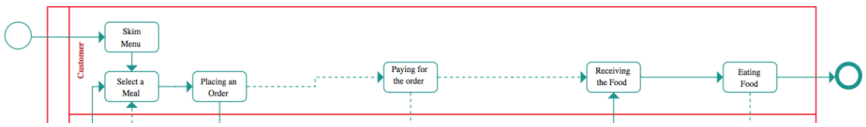

Figure 5: The customer flow

According to the workflow, we can identify some functional requirements that meet the demand of the customers.

- Customers can order food on the menu.
- Customers can pay their bill.
- Customers can give special requirements for their order which can be seen by the cook.
- Customers can give feedback.
- Customers can view their transaction history.

2. Clerks

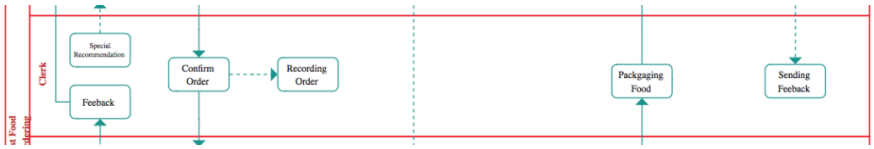

Figure 6: The clerk workflow

Based on the workflow of the clerks, we can define some functional requirements for the system.

- The manager can customize their QR code.
- The clerks can give special recommendations to the customers.

2.2 Non-functional Requirement

According to the expectation of the restaurant owner about the system, we can define the non-functional requirements.

- The system can be extendable in be use in multiple restaurants.
- The system should be usable from a mobile device, a laptop or a tablet device.
- The system should be implemented on the web.
- Customers can get access to the website via the QR code.
- The system should allow non-direct contact between Clerks and Customers.

2.3 Full system use-case diagram

From the workflow chart and the other party's requirements, we obtain the following use-case diagram.

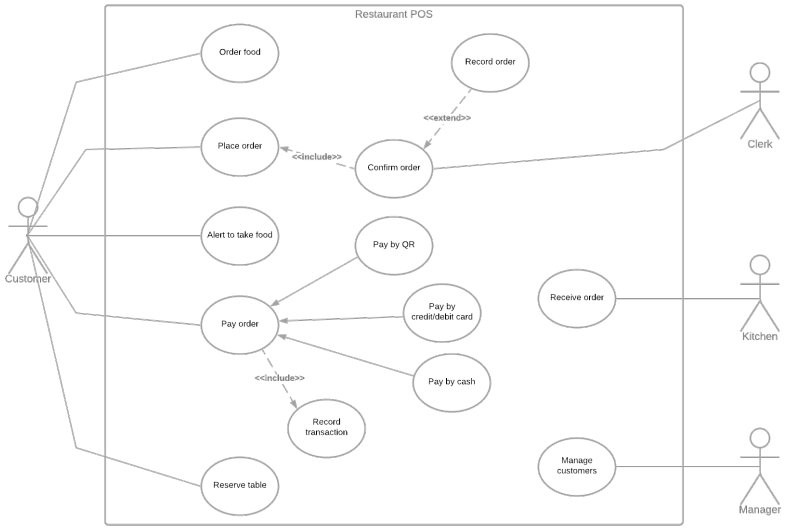

Figure 7: Full system use-case diagram

3 Diagram and table for pay order use-case

The following use-case diagram and table are for the pay order use-case.

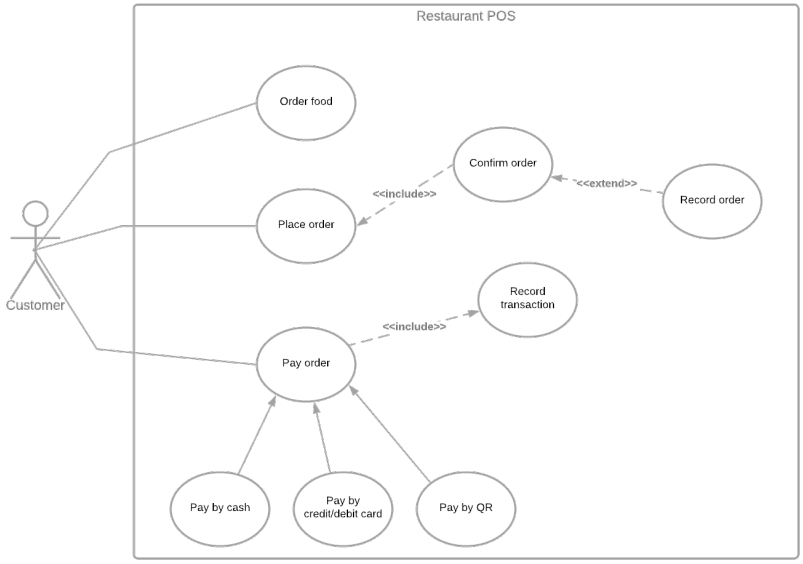

Figure 8: Pay order use-case diagram

Use-case name	Pay order / Pay the bill
Actors	Customer
Pre-condition	An order is confirmed
Trigger	Customer wants to pay the bill
Basic path	1. POS displays bill to customer
	2. Customer chooses a payment method
	3. Customer pays using the chosen method
	4. POS confirms and saves the transaction record or receipt

4 Activity diagram of major functional requirements

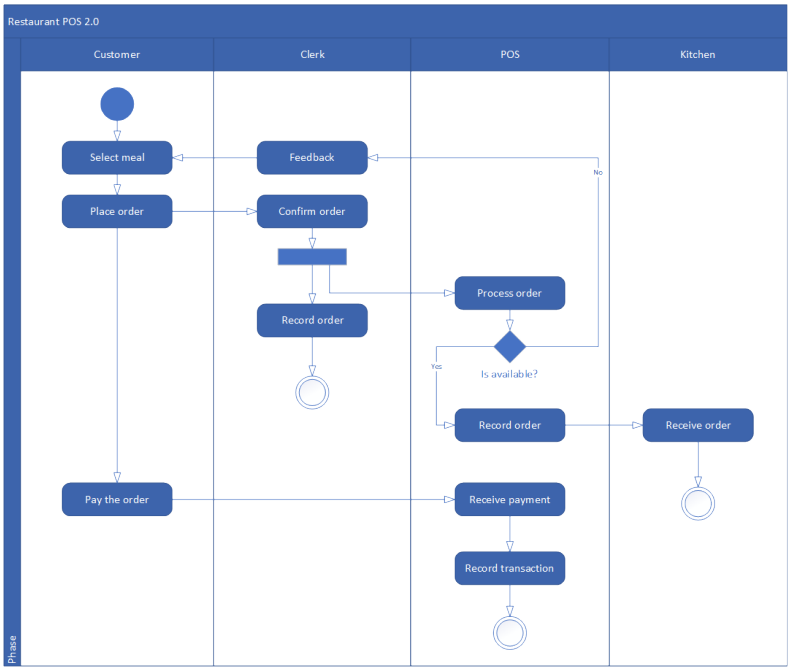

Figure 9: Activity diagram

5 Sequence of diagram of the pay order use-case

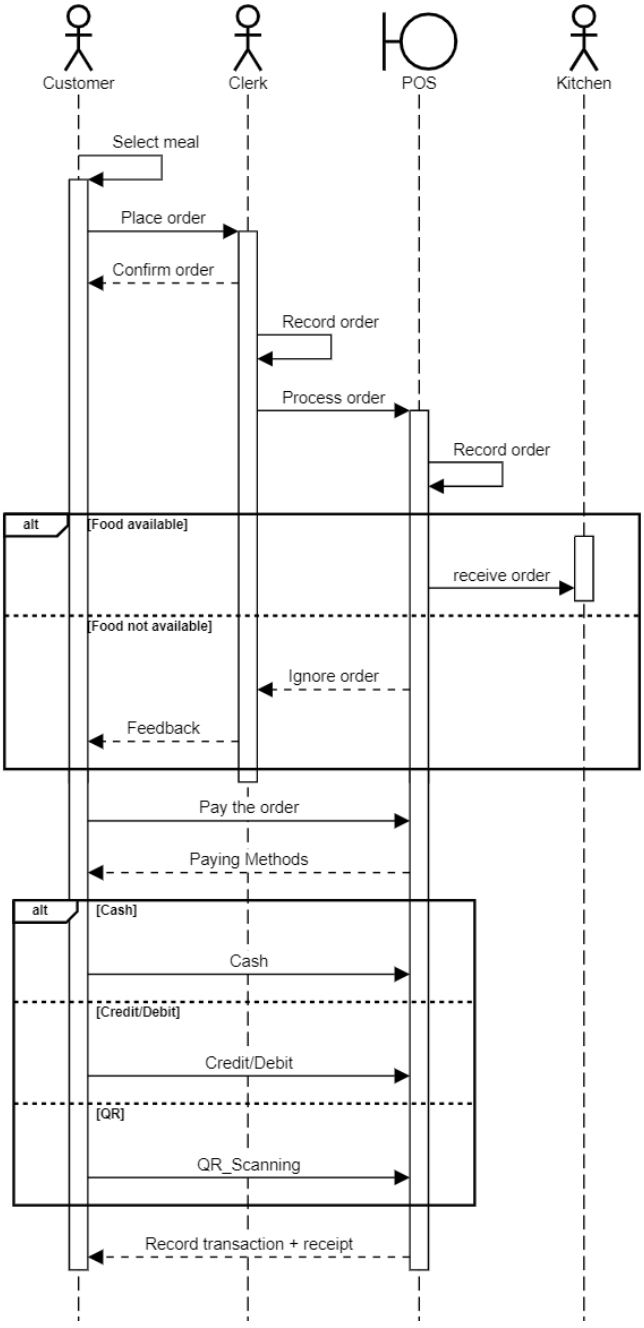

Figure 10: Sequence diagram of given use-case

6 Class diagram of the system

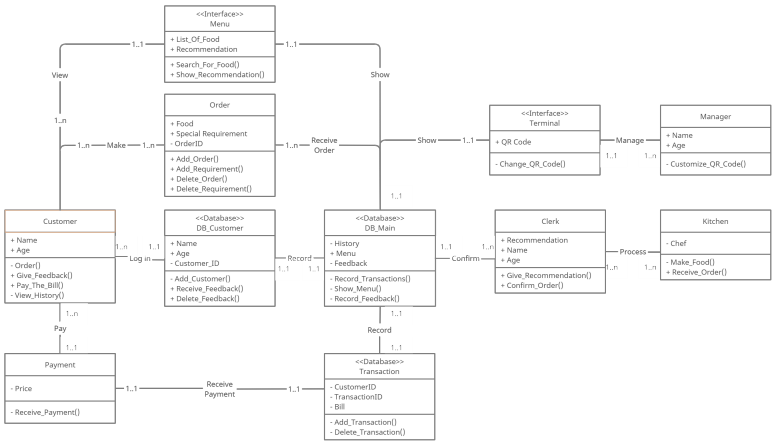

Figure 11: Class diagram

7 Architectural approach description

Model-view-controller (usually known as **MVC**) is a software design pattern commonly used for developing user interfaces that divide the related program logic into three interconnected elements.

This design pattern is considered to be universal when it comes to web application. Because of how it works, most of the webs, in particular it applies well to the idea that an application is accessed in the following way:

- The user asks for some resource
- Some underlying data is retrieved from somewhere.
- A template is then applied to that data in order to show it to the user.

This is undeniably the way the web works, most Web MVC frameworks start from the assumption that each user produces relatively few requests asking for big chunks of resources at a time. This is exactly what the Restaurant POS project requires when the first requirement is Web-based software serving convenience for the users.

The benefits of of using this approach includes: faster development, the ability to provide multiple views, supports asynchronous technique and avoid model collapsing by only one modification. This is perfect for popular customer-serving web applications, which in this case is our application.

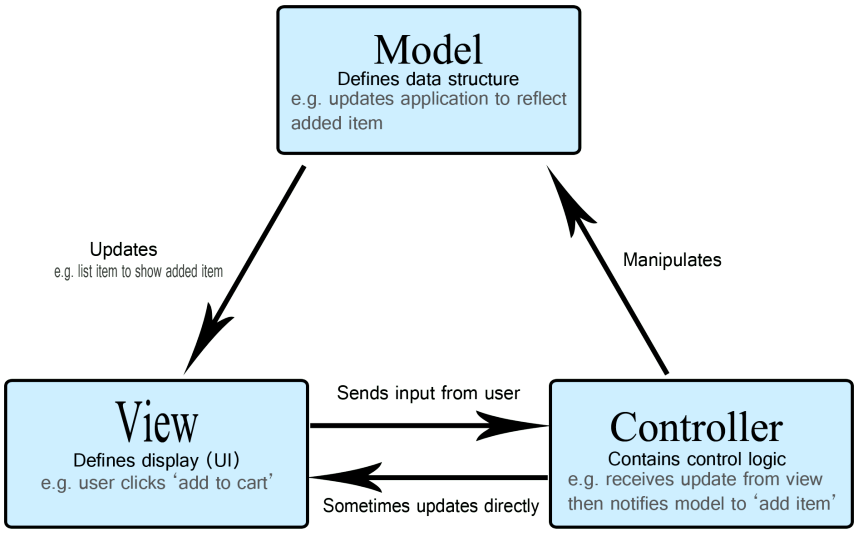

Figure 12: MVC diagram

8 Implementation diagram of major functional requirements

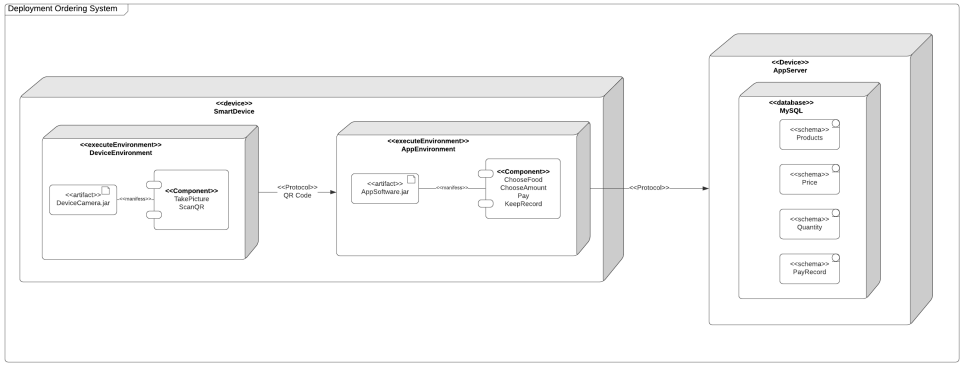

Figure 13: Deployment diagram