12/27/2021

Software Engineering

Virtual Market Application

Architectural Model - Report

Group Members

Michael Derece Kebede
Morteza Yosefy
Atiqurahman Mayar
Kibru Joba Kuture

Table of Contents

Introduction

A quick **recap**! We were asked to develop an Online Virtual Market Application (VMA) that lets users shop for required products and have these products delivered to their addresses. Some of the capabilities of this application are expected but not limited to:

VMA offers various product categories such as “Fruits & Vegetables”, “Dairy Products”, “Cleaning Products”, “Electronics” and so on.

Some of the tasks that **Customers** can do are given below:

- Search and filter products
- Add desired products into their shopping cart
- Benefit from coupons and discounts
- Specify some of the products as their favorites,
- Set and update their personal information such as address details and payment information
- Place their order, etc.

Some of the tasks that **Market/Store Managers** can do are given below:

- Set/update a physical store location
- Define/update new products
- Keep track of the stock
- Define/update coupons and discounts
- Fulfill orders
- Get statistics & reports about orders, etc.

The developed VMA will be accessible from different platforms such as web browsers, smartphones/tablets etc. (It’s **responsive**)

The use case to be implemented is placing an order by the customer, it is assumed that the use cases that the store manager must perform before this use case are already completed.

Example: products are already defined

As you know, we already went through project plan and requirement phase. Therefore, we have collected the specifications regarding to this project.

In this section, we're going to talk about the architectural model. First thing first, we will define the architectural model we're going to use, then we will determine and explain the main components and sub-systems, and the relationships between them which all included in our project.

After all, we will demonstrate the architectural model's block diagram.

Definition of Architecture

Architectural design is a creative process where you design a system organization that will satisfy the functional and non-functional requirements of a system. Although each software system is unique, systems in the same application domain often have similar architectures that reflect the fundamental concepts of the domain. So, it leads reuse of existed system architectures.

Virtual Marketing Application is based on the client-server architecture in which the client can be an application, which uses a Graphical User Interface (GUI) that sends request to a server for certain services and the server is the provider of the services requested by the client. Therefore, we preferred to use client-server architecture, and it's really common architecture for this project type.

The VMA web server is the computer program that provides services to other computer programs and serves requested Hyper Text Mark-up Language (HTML) pages or files. A client refers to a customer who requests for certain services and the server refers to the business application through which the services are provided. A machine can be both a client as well as a server.

There are two types of client server architecture that VMA follows including two-tier and three-tier architectures. In our project, we use the three-tier architecture.

In three-tier architecture, the user interface and the business application logic known as **business rules**. The data storage and access, are developed and maintained as independent modules. The three-tier architecture includes three tiers named as top tier, middle tier and third tier. The top tier includes a **user interface** where user services such as session, text input, and dialog and display management reside and the middle tier provides **process management** services such as process development, process monitoring and process resourcing that are shared by the multiple applications.

The third tier provides **database management functionality** in which the data management component ensures that the data is consistent throughout the distributed environment. Also the centralized process logic in this architecture, which makes administration easier by localizing the system functionality, is placed on the middle tier. The client-server architecture provides standardized and abstract interfaces to establish communication between multiple modules. When these modules are combined, they become an integrated business application and here each module is a shareable and reusable object that can be included in another business application.

a) Components of Architecture

Three-tier architecture is a well-established software application architecture that organizes applications into three logical and physical computing tiers: the presentation tier, or user interface; the application tier, where data is processed; and the data tier

We have basically three components (tiers) in this project.

I. User Interface / Presentation Tier

Occupies the top level and displays information related to services commonly available on a web browser or web-based application in the form of a graphical user interface (GUI). It constitutes the front-end layer of the application and the interface with which end-users will interact directly.

This tier is usually built on web development frameworks, such as CSS or JavaScript (ReactJS), and communicates with other tiers by sending results to the browser and other tiers in the network through API calls. It must be user friendly and therefore needs to be designed differently for each user group (customer, store-manager, and system admin).

II. Application

Also called the middle tier, logic tier, business logic tier. It is pulled from the presentation tier. It controls the application's core functionality by performing detailed processing and is usually coded in programming languages, such as Python, Java, JS, etc.

III. Database

Where information is stored and retrieved. Data in this tier is kept independent of application servers or business logic, and is managed and accessed with programs, such as MongoDB, Oracle, MySQL, and Microsoft SQL Server.

b) Sub-systems of Components

Note: As far as anyone knows, we are not obligated to explain all the sub-systems with their features and relationships here; but we will demonstrate them in detail later in the next section.

I. Sub-systems of UI Tier

Customer UI

It is the panel where the customer enters. From this panel, the customer can view the products, order, do payment, track their orders, etc.

Store-manager UI

This is a sub-system that provides easy control for store-managers. Store-managers can add, delete, and update products and their prices.

Admin UI

It is an admin panel. It is where system settings are viewed and manipulated.

II. Sub-systems of Application Tier

Login/Register System

It is the section where user's entry and registration are checked. It performs operations such as adding the user to the system and checking their presence in the system.

Authentication System

It is the section where the user identity is verified and checks his/her permissions.

Data Validation System

It is a sub-system which checks all input data that come from users for the system. For example, e-mail, phone number, etc.

Payment System

It is the section where the users pay for their orders.

Notification System

It is a sub-system that notifies users such as flash sales.

Courier System

It takes orders from store managers and delivers to customers, and vice versa in case customer does return.

Tracking System

In this section, the customer tracks their orders.

Return System

Customers can file some return request for the products they want to return.

Coupons System

That's where the coupons defined by the store managers display, customers will be able to use it.

III. Sub-systems of Database Tier

Virtual Market Application Database

In this project, data is the most significant part, which storing and manipulating them really matters. It is the database where the data belonging to the VMA system is stored. All the necessary data that keeps the system alive are stored right here. It is a relational database regarding to normalization rules. This is the main database that oversees storing and sharing data with related databases.

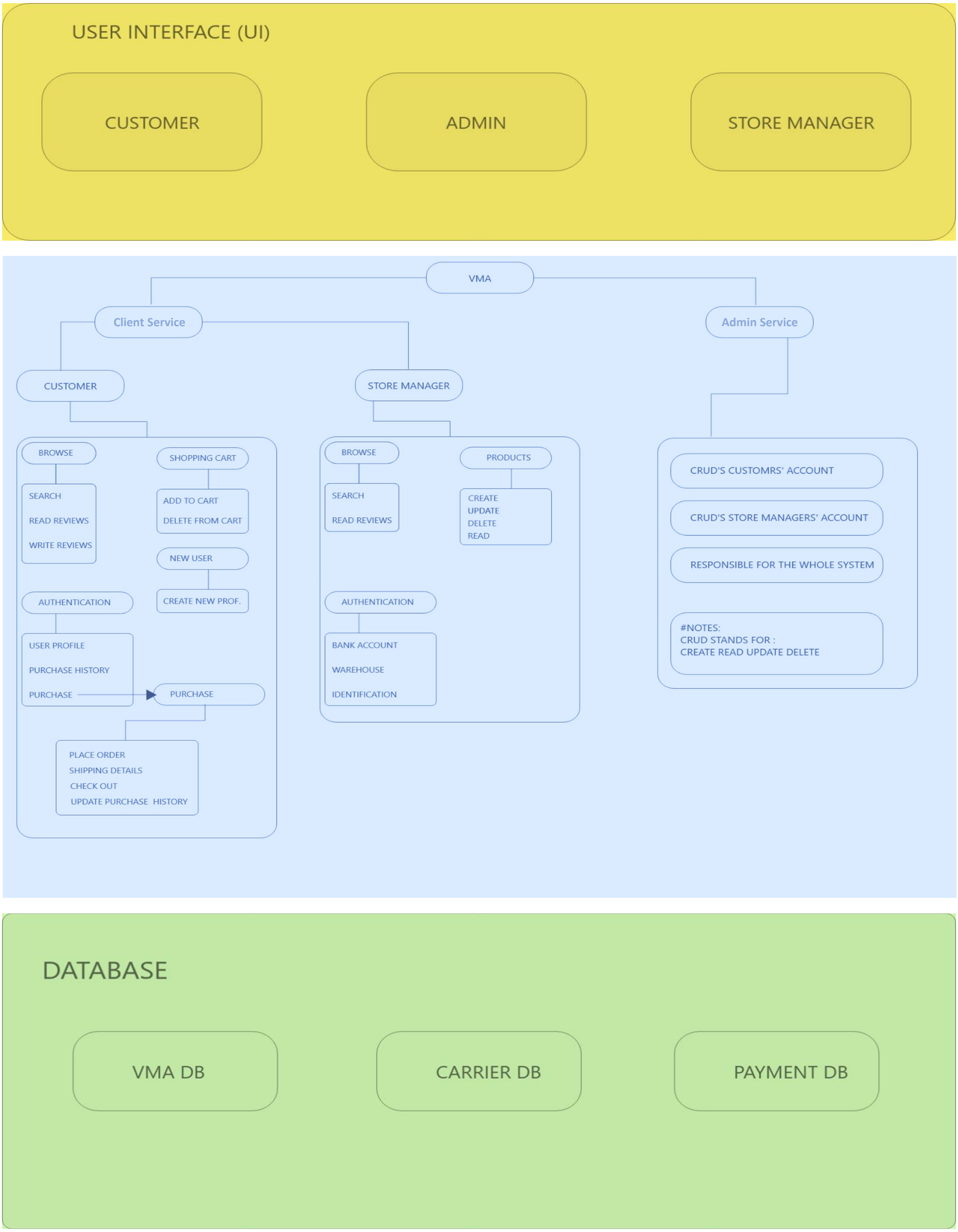
Courier Database

It's the database that is in contact with our main database (VMA Database). All necessary data regarding to the products delivery are acquired and stored here.

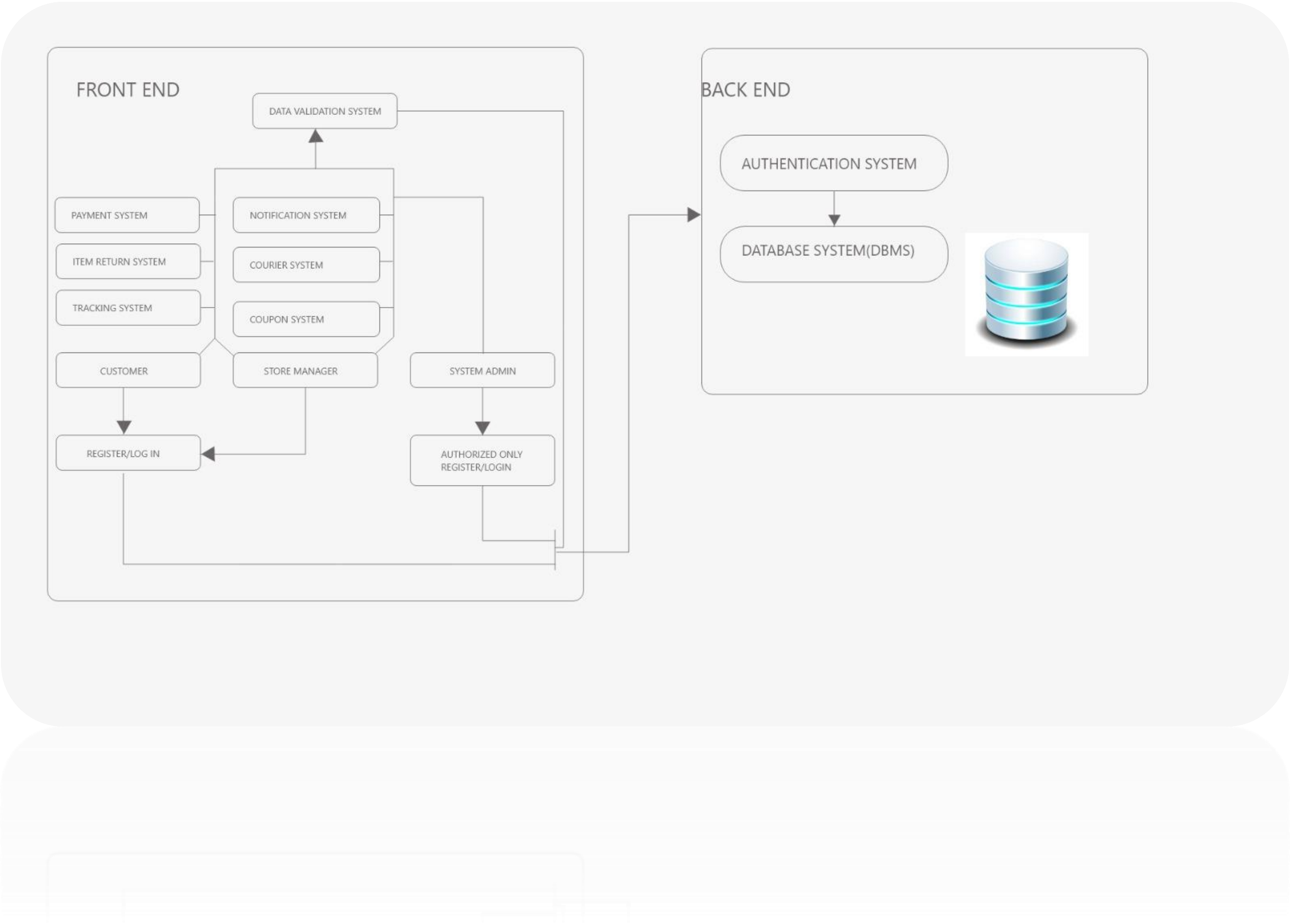
Payment Database

That's where the transactions related to each order are stored, and it is also connected with main database which is our VMA database.

Architectural Model as a Block Diagram



Here you can see the block diagram demonstrating the architectural model:



Elimination of Other Possible Architectural Models

Now Let us explain why we did not pick one of the other possible architectural models for our project.

As far as we concerned, we have discussed about four generic architectural models, once we discovered client-server architectural model, we realized that it is the most suitable architecture model for our project, that's the reason we picked it, we already discussed enough about why we selected this architectural model. Now let's talk about the reasons why we didn't select the other possible architectural models.

As we mentioned before, e-commerce is based on **the client-server** architecture. A client can be an application, which uses a Graphical User Interface (GUI) that sends request to a server for certain services. The server is the provider of the services requested by the client.

Repository Architectural Model

The crucial point is this model is suitable for those projects that work around a main database, and if the software is going to cope with big data in the database. You should use this pattern in case you have a system in which large amount of informations are generated that has **to be stored for a long time**. Each sub-system maintains its own database and passes data explicitly to other sub-systems.

In addition, this project is based on a shared database system where people involving in the system want to access it. The repository is a **single point of failure** so problems in the repository affect the whole system, may be some **inefficiencies** in organizing all communication through the repository, and distributing the repository across several computers may be difficult. Therefore, we decided not to use this architectural model.

Pipe and Filter Architectural Model

This model is commonly used in **data processing application** (both batch and transaction based). In this model, all the system work together in separate stages in order to generate related outputs. This model is **not really suitable for interactive systems**. That kind of system is not an option here. In our project, we're not just processing data, and the data is not sorted.

Layered Architectural Model

In this project, there're a lot of engagements between the sub-systems and different layers. Therefore, as long as **performance** of the systems using the layered architecture model can be a problem because of multiple levels of interpretation of a service request as it is processed at each layer. This model is used when building new facilities on top of existing systems; when development is spread across several teams with each team having responsibility for a layer of functionality, and when there is a requirement for multi-level security.

Glossaries

VMA – Virtual Market Application. An online market place where customers can shop for various categories of products.

Client Service – An application software component which contains customer and store manager.

Admin Service – An application software component which contains the admin.

VMA DB - Database where the data belonging to the VMA system is stored.

Payment DB - Is where the transactions related to each order are stored.

Courier DB - All necessary data regarding to the products delivery are acquired and stored here.

Browse - A function of VMA that lets users search for products, and read and write reviews for products.

Shopping Cart - Lets customers collect and temporarily store products that they want to purchase.

CRUD – CREATE READ UPDATE DELETE.

Customer - Users who login into the VMA with the intention of buying/browsing products.

Admin – Administers the VMA. Has ability to manage customers and store managers alike.

Store Manager - Users who administer, manage or own one store within the VMA.

Data Validation System – It validates any input coming from either customer or store manager.

Courier System – Facilitates the dispatch and return of products from store manager to customer and vice versa throught a courier.

Coupon System – Manages the creation and use of coupons on the VMA.

Notification System – Notifies users with configurable updates.

Item Return System – Allows for the return of products by the user.

Tracking System - Tracks orders from products original location to a certain destination.

Payment System – Facilitates payments between customers, store managers, couriers and admin.

Authentication System – Authenticates users before giving access to the database.

Responsive - Is an approach to web design that aims to make web pages render well on a variety of devices and window or screen sizes from minimum to maximum display size to ensure usability and satisfaction.

Conclusion

In this report, we have discussed about architectural models, which, in this case the client-server architecture model is the best solution. We have explained the main components, sub-systems, and the relationship between them, we’ve also demonstrated the structural representation of our system using block diagrams, which including components, sub-systems and relationship. Then, we listed the reasons why we haven’t picked other possible architectural models in detail, we have printed out a glossary in section 5.

As a short recap, we want to discuss about the definition of the architectural model we used, the client-server architectural model. It is a distributed system model which shows how data and processing is distributed across a range of components. It is a set of stand alone servers which provide specific services such as data management, etc. It is a kind of network which allows clients to access servers. The principle advantage of this model is that servers can be distributed across a network. General functionality can be available to all clients and does not need to be implemented by all services.