TStore - System Design Document

Project Name

The Store(TStore)

Team 5 Members

Jack Baxter

Shengrui Chen

Austin Pham

Hao Nguyen

Ryan Sheehan

1. **Introduction**

**1.1 Purpose**

This manual defines the outline design of TStore, and clarifies the detailed design of TStore's functional content, functional boundaries. This is the basis for the entire software development, and it guides the work in the later stages. This article is also the basis for system acceptance after the project is completed.

**1.2 Background**

Since the outbreak of COVID-19, more and more people have been infected with this virus. In order to reduce the number of people going out, our team decided to implement an online C2C platform that allows people to buy all kinds of items they need without going out.

**1.3 Definition**

**TStore**: TStore is the name of our online shopping mall.

**RESTful**: Representational state transfer (REST) is a software architectural style that defines a set of constraints to be used for creating Web services.

**JSON**: JavaScript Object Notation is an open standard file format, and data interchange format, that uses human-readable text to store and transmit data objects consisting of attribute-value pairs and array data types (or any other serializable value).

**JDBC**: Java Database Connectivity (JDBC) is an application programming interface (API) for the programming language Java, which defines how a client may access a database.

**1.4 Reference**

[RESTful](https://en.wikipedia.org/wiki/Representational_state_transfer)

[JSON](https://en.wikipedia.org/wiki/JSON)

[JDBC](https://en.wikipedia.org/wiki/Java_Database_Connectivity)

1. **Design Overview**

**2.1 Project operating environment**

The display part of the website is implemented by HTML, and the database we use PostgreSQL. Both the database and the project are deployed on the server, and users only need to access them through a browser.

**2.2 Basic process**

**Shopping process:**

The basic process for customers to buy something on TStore can be divided into the following four steps:

Step 1: Add to cart

Step 2: Proceed to checkout

Step 3: Prepared for shipment

Step 4: Leave seller feedback

**The detailed operation steps are as follows:**

If the user is already a member, after logging in to the TStore, the user can purchase through any of the following channels:

* Click "Buy Now" to buy directly

Step 1: Click to “Buy Now” button

Step 2: Confirm the delivery address, purchase quantity, payment method and other elements, then click “Place your order” button

Step 3: After the server accepts the user's purchase request, it performs data processing in the background and returns the processing result.

Step 4: The client displays the message of whether the purchase is successful or not.

Step 5: Users can go to my order to check transaction records and transaction status

* Click "Add to Cart" and then "Proceed to checkout"

The purchase method after being added to the shopping cart is similar to the purchase method immediately.

**2.3 Structure**

1. User registration page: Users can register on this page to use the functions of the website.
2. User login page: Provide user login function, allowing users to keep logged in status.
3. User profile page: contains user profile, such as address, payment method, and other information, which can be viewed or modified.
4. Product display page: display all kinds of products that can be purchased on our website, allowing users to search through product name, category, and other information.
5. User order page: display order information.
6. Shopping cart: shopping records of the user's current session.

**3. Interface Design**

**3.1 User interface**

The project uses a B/S structure, and users can access and operate our pages through a browser. Therefore, these pages are the main body of interaction with customers, so the design of the interface should be clear and friendly, with the main purpose of facilitating user operation.

**3.2 Internal interface**

The front-end page uses RESTful architecture to interact with the server, and data is transmitted in JSON format.

The server interacts with the database using JDBC.

**4. Data structure design**

**4.1 Logical structure**

1. User information table

User (UserID, UserName, Password, PhoneNumber, Address, RegistrationTime)

1. Commodity Information table

Commodity (CommodityID, CommodityName, Description, Price, Category)

1. Category classification table

Category (CategoryID, CategoryName, description)

1. Order information table

Order (OrderID, UserID, DeliveryAddress, OrderTime, OrderStatus)

1. Shopping cart table

ShoppingCart (CartID, CommodityID, Price, Quantity, OrderID)

**4.2 Physical structure design**

1. User information table (User)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Range | Req/Opt | Derived/Stored |
| UserID | INTEGER | >0 | Required | Stored |
| UserName | VARCHAR(50) | not empty | Required | Stored |
| Password | VARCHAR(50) | not empty | Required | Stored |
| PhoneNumber | VARCHAR(15) |  | Optional | Stored |
| Address | VARCHAR(100) |  | Optional | Stored |
| RegistrationTime | DATE | not empty | Required | Stored |

1. Commodity Information Table (Commodity)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Range | Req/Opt | Derived/Stored |
| CommodityID | INTEGER | >0 | Required | Stored |
| CommodityName | VARCHAR(50) | not empty | Required | Stored |
| Description | VARCHAR(300) |  | Required | Stored |
| Price | INTEGER |  | Required | Stored |
| Category | INTEGER |  | Required | Stored |

1. Category classification table (Category)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Range | Req/Opt | Derived/Stored |
| CategoryID | INTEGER | >0 | Required | Stored |
| CategoryName | VARCHAR(50) | not empty | Required | Stored |
| Description | VARCHAR(300) |  | Optional | Stored |

1. Order information table (Order)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Range | Req/Opt | Derived/Stored |
| OrderID | INTEGER | >0 | Required | Stored |
| UserID | INTEGER | not empty | Required | Stored |
| DeliveryAddress | VARCHAR(100) |  | Required | Stored |
| OrderTime | DATE | not empty | Required | Stored |
| OrderStatus | VARCHAR(10) | not empty | Required | Stored |

1. Shopping cart table (ShoppingCart)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Range | Req/Opt | Derived/Stored |
| CartID | INTEGER | >0 | Required | Stored |
| CommodityID | INTEGER |  | Required | Stored |
| Price | INTEGER |  | Required | Stored |
| Quantity | INTEGER |  | Required | Stored |
| OrderID | INTEGER |  | Required | Stored |

**4.3 Data structure and project**

This project uses a relational database, and standard SQL statements should be used to interact with the database during the implementation process.

**5. System Error Handling Design**

**5.1 Error Message**

The project adopts an exception handling mechanism, which can deal with exceptions in time to ensure the safety and stability of the program. For the user's wrong input, a relative prompt should be given to tell the customer the reason and solution of the error, so as to minimize the possibility of making the user wrong.

**5.2 Error Remedy**

Remedial measures should be taken as far as possible when the system fails, including:

1. For the database, the security of the data should be guaranteed, and database backups should be performed frequently.
2. If the error is caused by the user's improper operation, the user should be prompted for the reason for the error, and then the user should be allowed to operate again
3. For other unpredictable errors, if the system cannot handle it, then we should do a good of testing to minimize the possibility of unpredictable errors.

**6. System Maintenance Design**

**6.1 Software maintenance**

If the business process or requirements changed, the program should be modified to adapt to the new changes.

**6.2 database maintenance**

During the operation of the system, various data may be generated. These data need to be maintained in a timely manner, including backup and restoration of the database, adjusting the data structure of the database, and modifying the records of the database.