IT314: Software Engineering

Course Project Lab 6

Group: 32 Central Mess Management System

202001444	PARMAR GAURANG GAJENDRABHAI
202001454	KANANI DEEP MUKESHBHAI
202001075	RAJPUT VRUND ASHWINKUMAR
202001418	RAHIL HARDIK SHUKLA
202001053	KRISH GULABANI
202001443	LIMBACHIYA OM
202001232	PRAJAPATI PARTH KANUBHAI
202001427	MOHAL RAJYAGURU
202001435	MEHTA NEELKUMAR HIRENKUMAR
202001451	VIVEK GODHASARA
202001417	DHANANJAY A VORA

Domain Analysis Modelling

Boundary Objects:

1. Details Interface:

   Display recent Menu.These details are visible even without authentication.

2. Authentication / Registration Interface:

2.1. User Authentication:

2.1.1. The interface where the user can login and SignUp to the site to access all the functionalities. Without login, users can only see the menu.

2.2. Manager Authentication:

2.2.1. The Interface where the manager can login and SignUp to the site and manage their details.

2.2.2. After Authenticating, the manager can create or edit their details.

Entity Objects:

1. **User:** It contains information about the users roles and responsibilities. User can be Manager, customer and cadet. Information includes name, username, birthdate, password, gender and role, etc. It also includes operations that can be done by user which includes Login, Signup, viewUserdetail, etc.

2. **Manager**: It contains the information about the operations that manager can do.

3. **Cadet** : It contains information about the cadet and operations that can be done by cadets.

4. **Customer**: It contains information about the customer and operation that can be done by the customer.

Control objects:

1. **Registration as user:**

   It controls the flow of registration data to the database.There will be three types of user Manager, Customer and Cadet. Every registered user information will be stored in this database.

2. **Daily Checklist:**

   The Daily Checklist object is a control object as it is responsible for checking and ensuring that daily tasks are performed as required. The Daily Checklist object can access the system functionalities related to daily tasks.

3. **Payment:**

   The Payment object is a control object as it is responsible for handling for storing the payment transactions between the system and the customer. The Payment object can access the system functionalities related to payment processing.

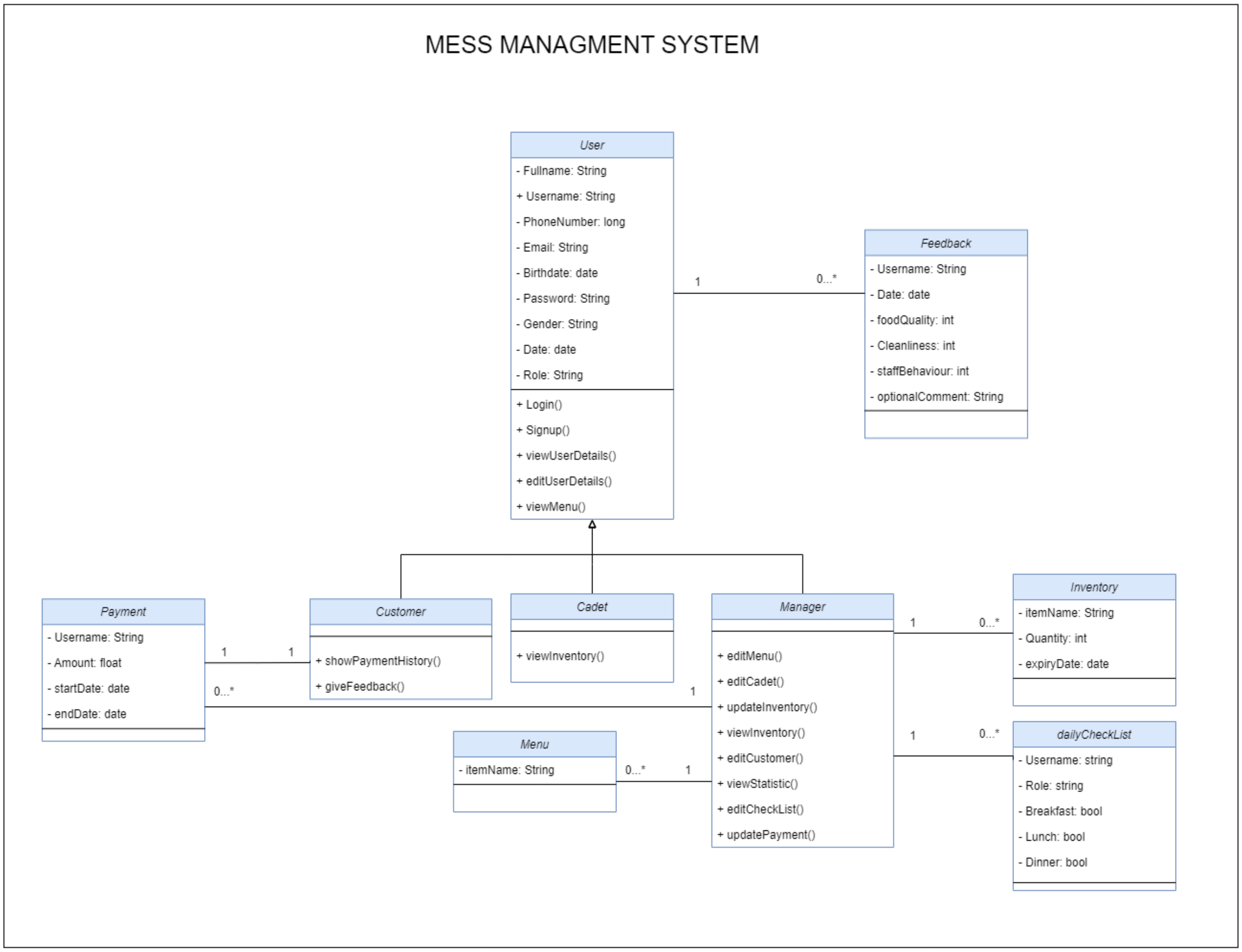
4. **Login as user:**

   It controls the validation of the user information from the database.

5. **Inventory:**

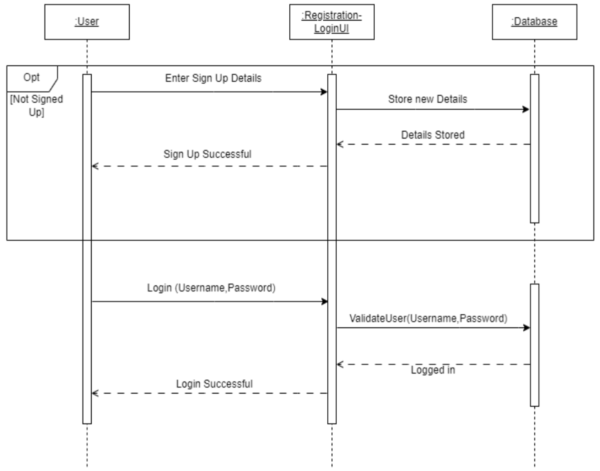
   The Inventory object is a control object as it is responsible for managing the stock of items available for use.

Class Diagram

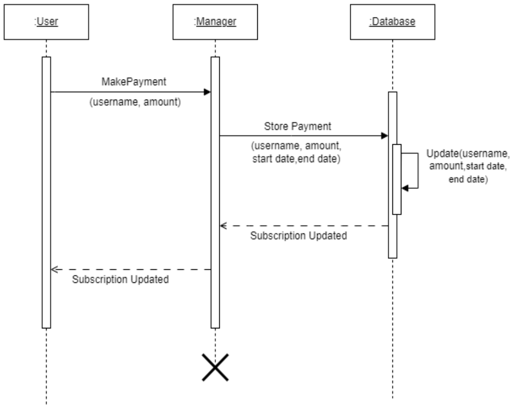


Sequence Diagram

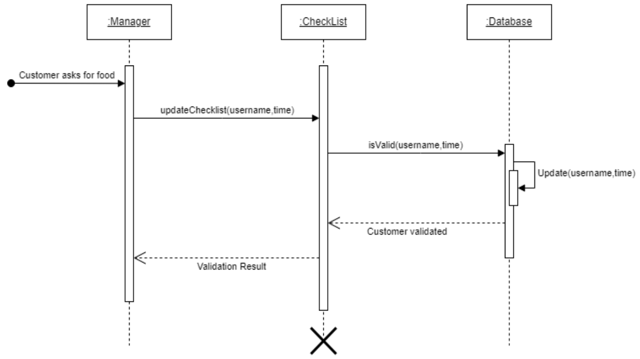
Login/Sign Up



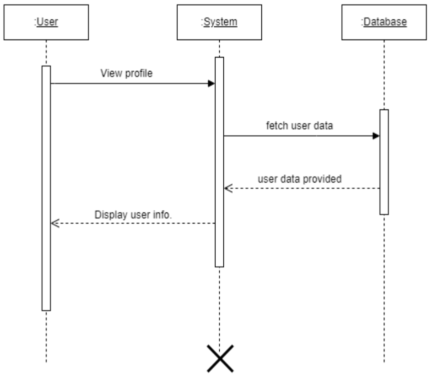
Update Payment



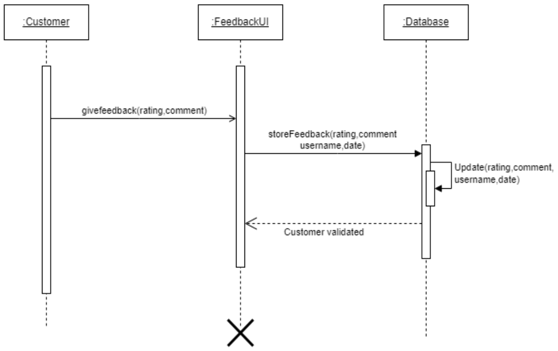
Update Checklist



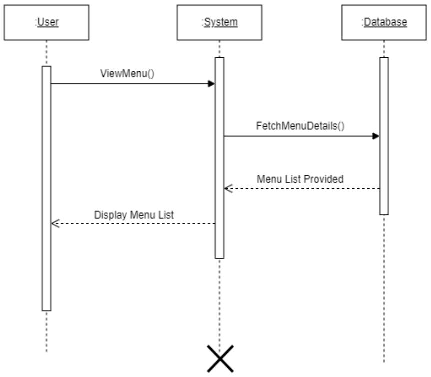
View Profile



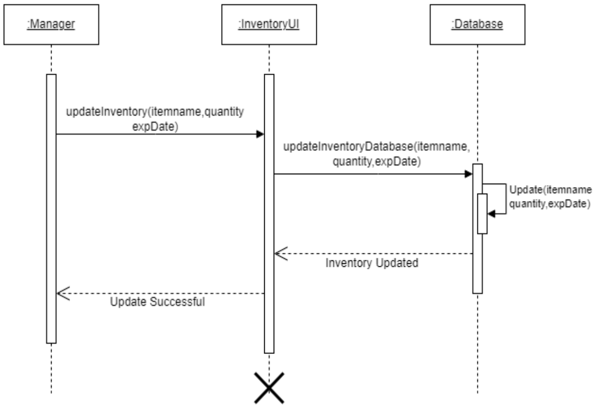
Feedback



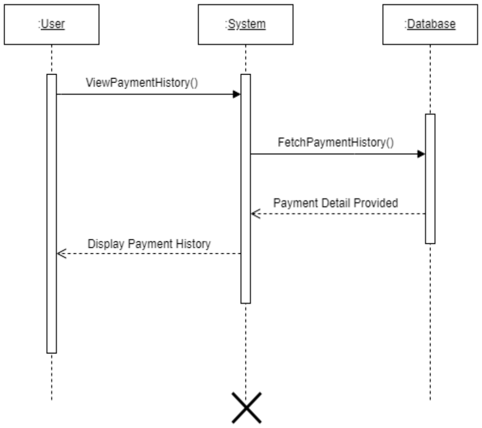
View Menu



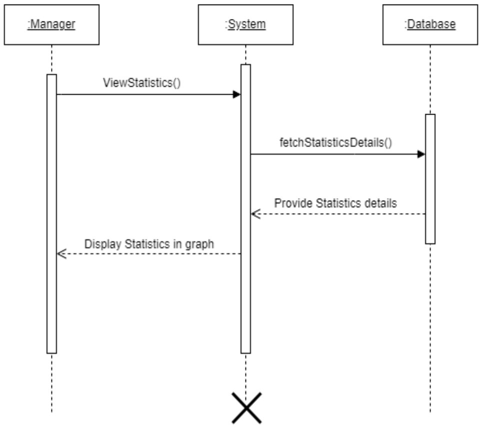
Update Inventory



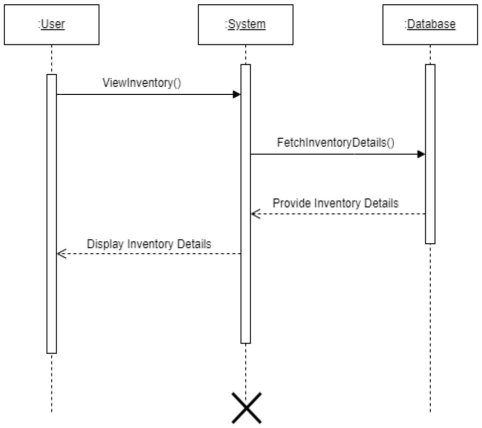
View Payment History



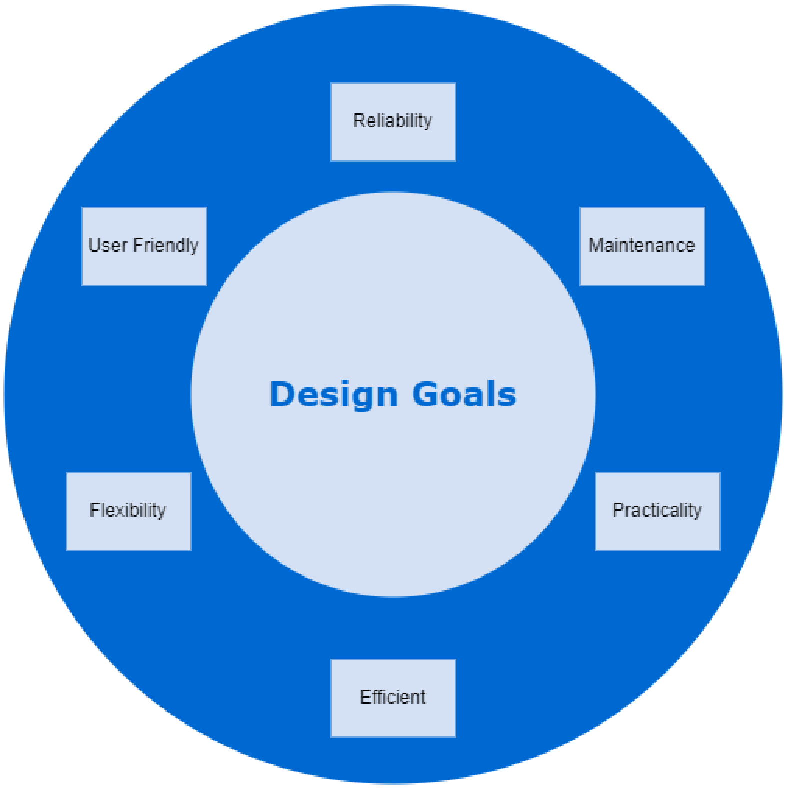
View Statistics



View Inventory



Design Goals:



1. Reliability:

Developers prioritize reliability as a key design goal due to the potential risks associated with a layout that may not function properly under various circumstances.

2. Maintenance:

The purpose of maintenance is to enhance the system's performance, functionality, and reliability, as well as its ability to adapt to changing circumstances and environments, ensuring that the software meets user demands over time.

3. Efficient:

The software must be prompt in providing users with the required information, as a delay in retrieving information can result in a negative user experience and poor design.

4. User-Friendly:

To cater to users unfamiliar with the system, a well-designed interface should be implemented that enables users to understand the system's workings with ease.

5. Flexibility:

The system's flexibility refers to its ability to adapt to unforeseen circumstances while maintaining regular operations.

Software flexibility is a key consideration during design and development, allowing for the program's modification without compromising the system's functioning.

6. Practicality:

Designing a system that caters to its intended audience is essential, ensuring that the software is tailored to the needs of its users.

HIGH LEVEL SYSTEM DESIGN:

ARCHITECTURE:

This application will use a client-server architecture. We will use a 3 tier application architecture that consists of a presentation layer, application layer, and data layer. The data layer stores information, the application layer handles logic, and the presentation layer consists of the front end.

- **Presentation Layer:**

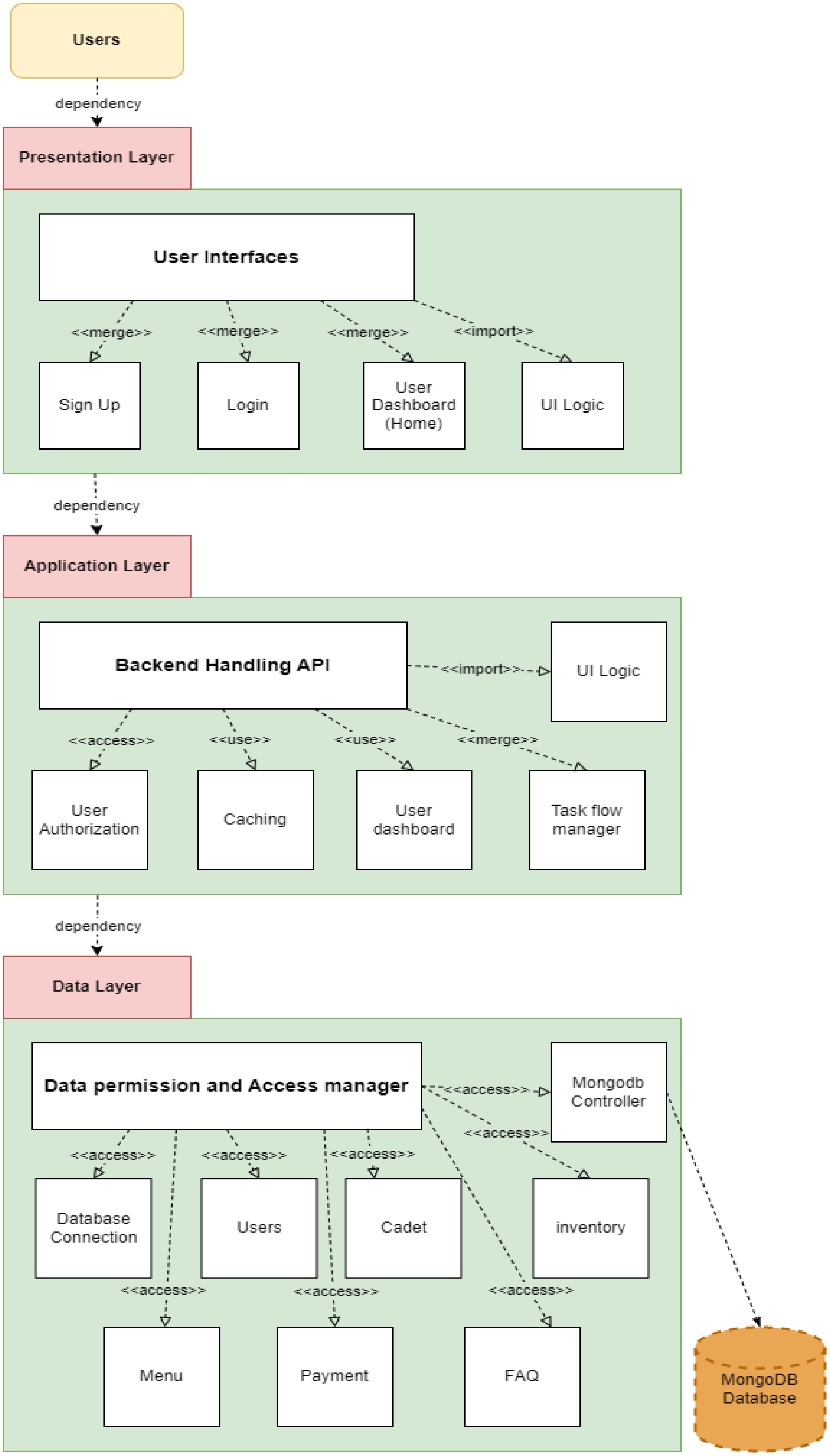
  This layer is distributed to a computing device using a web browser or a web-based application and is constructed with **HTML5**, **CSS3, and Javascript**. Application program interface (API) calls are the primary communication between the user interface and other application layers.

- **Application Layer:**

  This layer is implemented using **Node.js** and contains the business logic that supports the application's core functions. This choice is because it allows the app to be used by many users and provides flexibility for future development.

- **Data layer:**

  The data layer, sometimes called the database layer, data access layer or back-end, is where the information processed by the application is stored and managed. We will be using the **NoSQL Database(MongoDB)** for this layer.


Users
dependency
Presentation Layer
User Interfaces
<<merge>>
<<merge>>
<<merge>>
<<import>>
Sign Up
Login
User Dashboard (Home)
UI Logic
dependency
Application Layer
Backend Handling API
<<import>>
UI Logic
<<access>>
<<use>>
<<use>>
<<merge>>
User Authorization
Caching
User dashboard
Task flow manager
dependency
Data Layer
Data permission and Access manager
<<access>>
Mongodb Controller
<<access>>
<<access>>
<<access>>
<<access>>
Database Connection
Users
Cadet
inventory
<<access>>
<<access>>
<<access>>
Menu
Payment
FAQ
MongoDB Database