Food Ordering website

Revision History

|  |  |
| --- | --- |
| Version No. | Prepared by / Modified by |
| 1 | Team E |

Table of Contents

[1 Introduction](#_Toc379364881)

[2 Scope of Change](#_Toc379364882)

[3 List of impacted modules](#_Toc379364883)

[4 Design and Detailed technical updates](#_Toc379364884)

[4.1 Proposed uservInter face design](#_Toc379364885)

[4.1.1 Use case Model](#_Toc379364886)

[4.1.2 Sequence diagram](#_Toc379364887)

[5 Details of Alternative Design Approach](#_Toc379364892)

[5.1 CI/Build relates tasks](#_Toc379364894)

[6 References](#_Toc379364900)

# Introduction

The theme of project is the online restaurant application it is developed using HTML, JAVA script, CSS. The app is very useful where the customer can easily order items or dishes from home through mobile or system.

1. Admin:

Admin can add or remove food items or dishes to app. Admin can makes updates or any changes in app.

1. User:

User can order the items from the app. Filtering can be done with respect to price or brand of the product.

The user will be provided with features like sort, filter, on the home page, to ease the search of the product. Sorting can be done either alphabetically on the basis of product name or according to the product price (Low to High and High to Low).

Our application reduces lot of work load for customer as well as owner. The transaction of money is completed in real time system.

# Scope of Change

1. Customer can able to purchase the food items from the application and admin can able to modify the application.
2. The system has email validation through mail.
3. The system recommends a facility to accept the orders 24\*7 and a home delivery system which can make customers happy
4. Customer can choose door delivery option
5. Our App features are very easy to access and it’s very easy to use it.

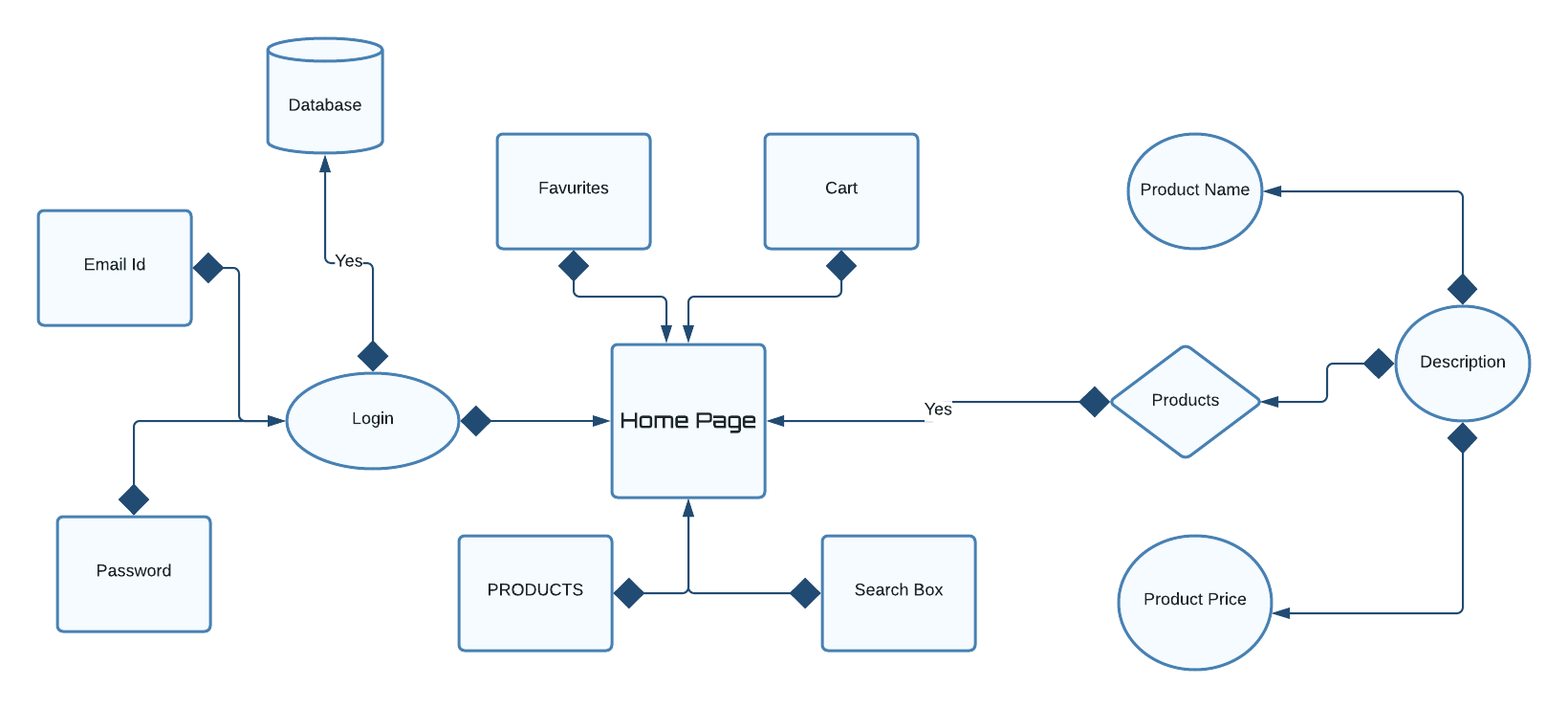
# List of modules

These are the modules used in this app:

1. **Home:** It’s a dashboard look for the website.
2. **Sign-In:** Users should enter their personal details like name, email id, address, mobile number. The user must choose a password for himself which will be used for future logins. The password will be encrypted before being fed in the DB. After successful registration, the user can login to the application.
3. **Log-In:** The user can login using email and password. If the user fails to enter correct email and password, the Forgot Password option can be clicked which will allow the user to reset the password. The user can purchase a product, compare or add to wish list only when he/she logins.
4. **User:** The user will be provided with features like sort, filter, on the home page, to ease the search of the product. Sorting can be done either alphabetically on the basis of product name or according to the product price (Low to High and High to Low). Filtering can be done with respect to price or brand of the product.
5. **Reviews**: Reviews sections shows customers feedback on our restaurant.
6. **Cart**: You can see items which you ordered in this cart.
7. **Customer Services:** you can contact if any doubts regarding to app or any problem occurred by ordering.
8. **Mongodb Database(Test):**
   1. **User collection:** whenever the new user signed up the data was stored in users collection and whenever the user tries to login it will be retrieved
   2. **Product collection:** whenever admin enter the product details it will stored in products collection and admin can done CRUD operations with this collection and it will be displayed in menu
   3. **Order collection:** whenever the user ordered product it will be stored in orders collection
   4. **Sessions collection:** whenever user login, one unique session will be created and stored in sessions collection

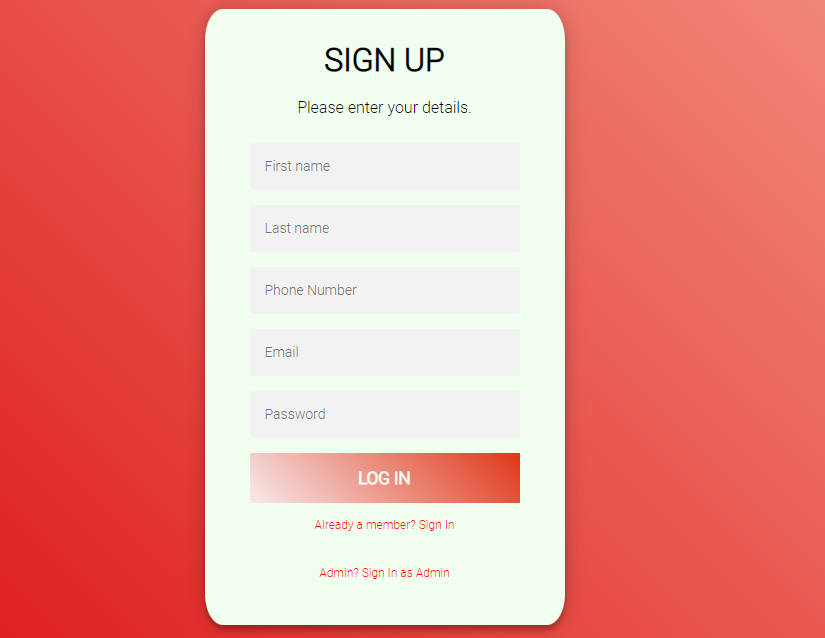
# 4 Design and Detailed technical updates

## 4.1 Proposed user Interface design

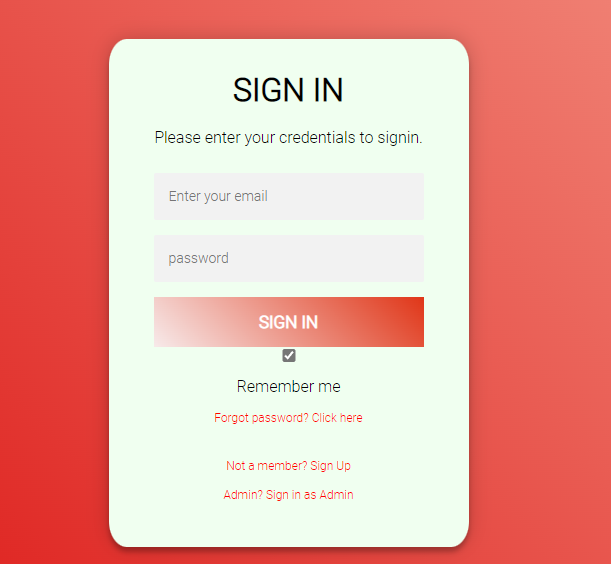


## 4.2.1Use Case Model

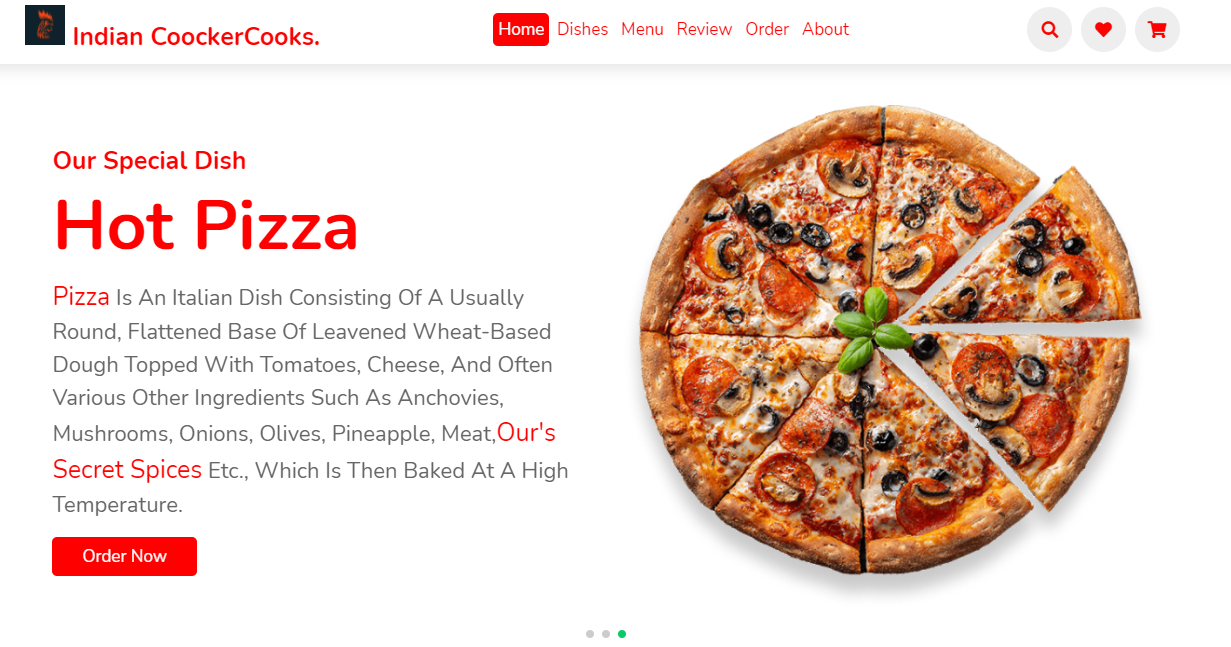
|  |  |
| --- | --- |
| Brief Description | Customer Registration (Sign-Up) |
| Basic Flow | This use case describes how a client register in to the system   1. The client has to register himself into the system. 2. After the successful registration, client will get a success message. 3. The following information is required during registration.  * First Name * Last Name * Email * Mobile Number * Password |
| Alternate Flow | The system will validate the information provided. If any invalid data is found, the input form will be redirected with error message. |
| Validation | 1. First name and Last name must be required. 2. Email should be valid. 3. Mobile number must be in numbers and contain 10 digits. 4. Password must contain at least one number, one uppercase and one lowercase and at least 8 or more characters. |
| Pre-Conditions | User should have network access and Browser with latest updates. |
| Post-Conditions | Success message should be shown. |



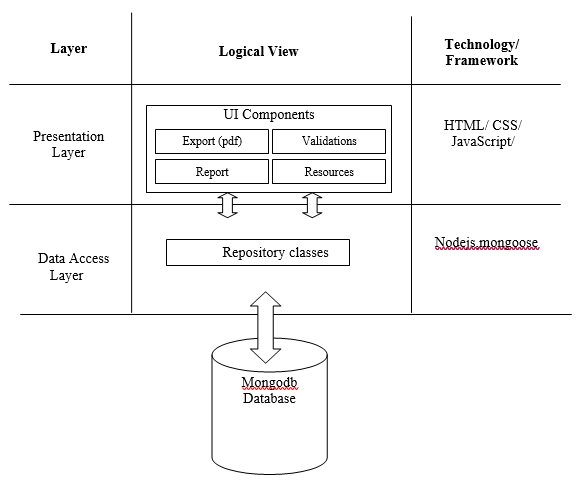
|  |  |
| --- | --- |
| Brief Description | Login |
| Basic Flow | This use case describes how a user log-in in to the system   1. The user has to register himself into the system. 2. After the successful login, user will be taken to the appropriate landing page. 3. The following information is required to login.  * email * password |
| Alternate Flow | The application will validate the credentials provided. If credentials are invalid, login form will be redirected again with error message. |
| Validation | 1. Valid email 2. Valid password |
| Pre-Conditions | User should have network access and Browser with latest updates. |
| Post-Conditions | Landing page has to be displayed. |



|  |  |
| --- | --- |
| Brief Description | Welcome Page |
| Basic Flow | This use case describes how the homepage looks like  1. The user can see our features.  2. He/she can click on the products which they want to order.  3. After the successful login, user will be taken to the appropriate landing page.  4. The following has to be done to move into the home page.   * Sign-Up * Log-In   5.User can able to see reviews, products and categories. |
| Alternate Flow | Not Applicable |
| Validation | Not Applicable |
| Pre-Conditions | User should have already logged into the system. |
| Post-Conditions | Not Applicable |



Architecture Diagram



* Presentation layer: HTML/ CSS/ JavaScript/ Bootstrap/React
* Data Access Layer: MongoDB/NodeJS

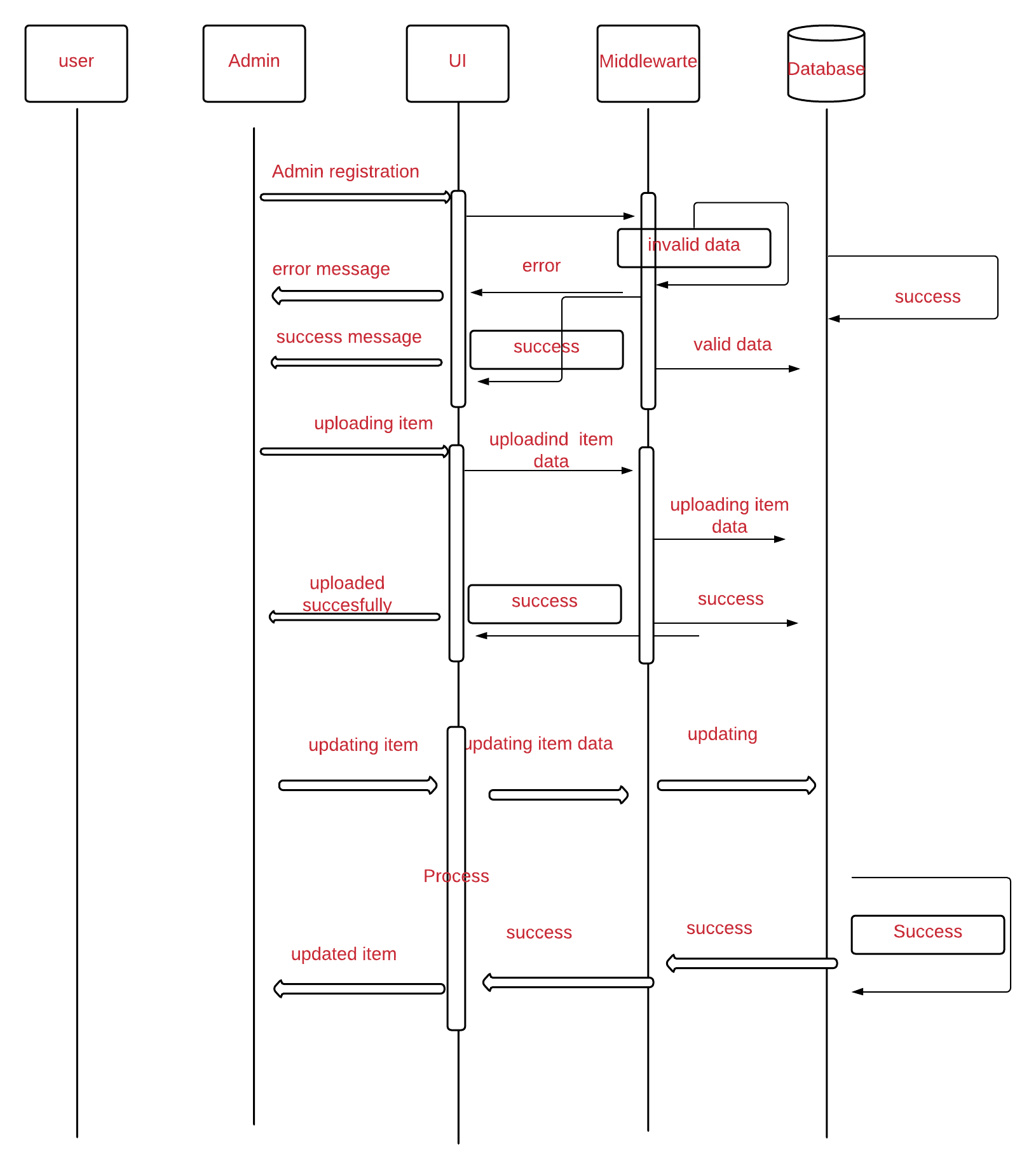
**Presentation Layer:**

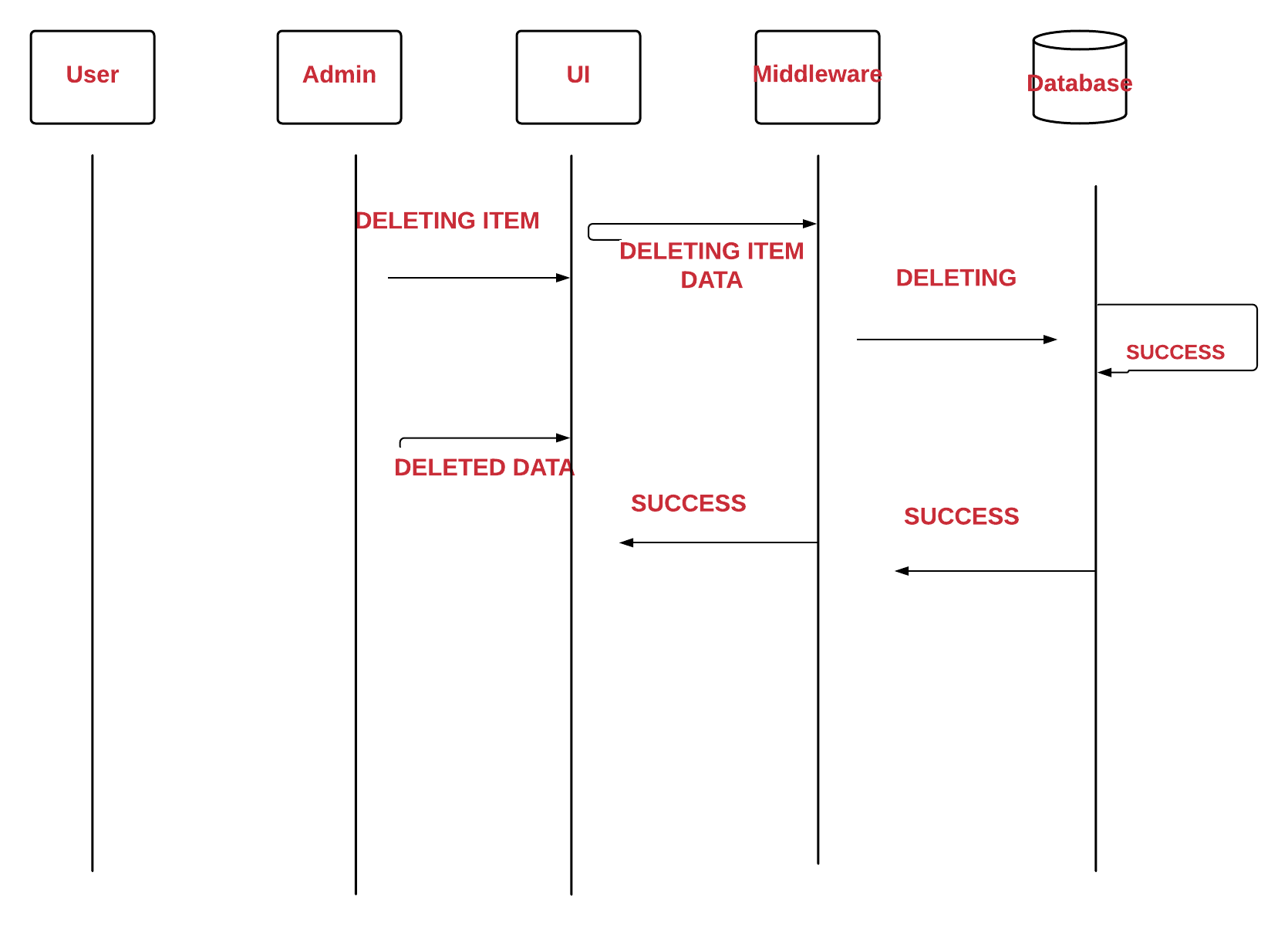
This layer consists of all UI components.

* Validation component: All basic level data validations should be done at UI level.
* Report component: All reports to be generated to the user, will be processed and generated at the browser end.
* Resources: All HTML/CSS/ images, which are required for the page design
* **Data Access Layer:** Using Mongoose and NodeJS we can access the Mongodb database.

### 4.2.2Sequence diagram

### C:\Users\varunsai.paduchuru\Downloads\MicrosoftTeams-image (7).png





# 5 Other Technical changes

## 5.1CI / Build relates tasks

Still have to be done add to cart in local storage of the website

# 6 References

* Google, Github , youtube