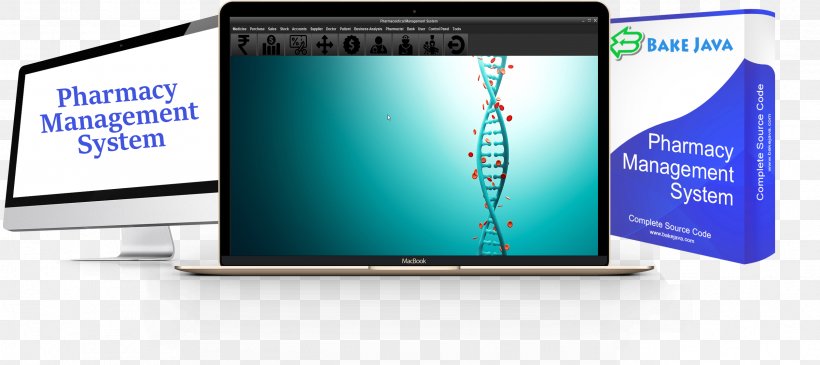
**PharmacyManagementSystem**  
**Low Level Design (LLD)**



DOCUMENT APPROVAL

**Approvers of this document**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Department** | **Role** | **Signature** | **Date** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Document Change History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Document Version #** | **Author** | **Date** | **Description** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1.0 Document Purpose 4](#_Toc94636300)

[2.0 Intended Audience 4](#_Toc94636301)

[3.0 Project Background, Objective(s) 4](#_Toc94636302)

[3.1 Project Background 4](#_Toc94636303)

[3.2 Project Objective 4](#_Toc94636304)

[4.0 Design Pattern 5](#_Toc94636305)

[5.0 Solution Diagram 5](#_Toc94636306)

[6.0 Solution Steps 5](#_Toc94636307)

[7.0 Classes/function name 6](#_Toc94636308)

[8.0 Validations 6](#_Toc94636309)

[9.0 Data model/Tables 6](#_Toc94636310)

[10.0 API Canvas 6](#_Toc94636311)

[11.0 HTTP Status Code 7](#_Toc94636314)

[12.0 Request 7](#_Toc94636314)

[13.0 Response 7](#_Toc94636314)

# Document Purpose

This document describes the solution architecture for Pharmacy Management System.

# Intended Audience

This document is intended as a reference for the following roles who are interested in the Pharmacy management system architecture.

|  |  |
| --- | --- |
| Role | Nature of Engagement in WB Classics Portal Technical Architecture |
| Product Owners/SME | Key stakeholder to ensure that the architecture is aligned with business goals. |
| Business Analysts | Business analysts are one of the stakeholders who are informed with the key architectural decisions. |
| Enterprise Architects | To enforce Customer management Platform Architecture is aligned to business goals and architecture, architectural guidelines. |
| Solution Architects | To ensure solution design and architecture is aligned to business requirements, architectural guidelines. |
| Developers | Use Technical Architecture Document as the guiding document for detail design and implantation approach to align with Customer management Microservice |

# Project Background, Objective(s)

## Project Background

Pharmacy management system enables better workflow for both the admin and the doctor as everything is available on the website and they no need to talk one on one for ordering the drugs, it saves the time of both doctor and admin. It increases their productivity.

## Project Objective

Pharmacy Management has various operations like listing drugs, creation, updating and deletion of drug Details as well as the supplier details. Doctor can buy the drugs.

Both Admin and the doctor can first register themselves and then they can perform all the operations.

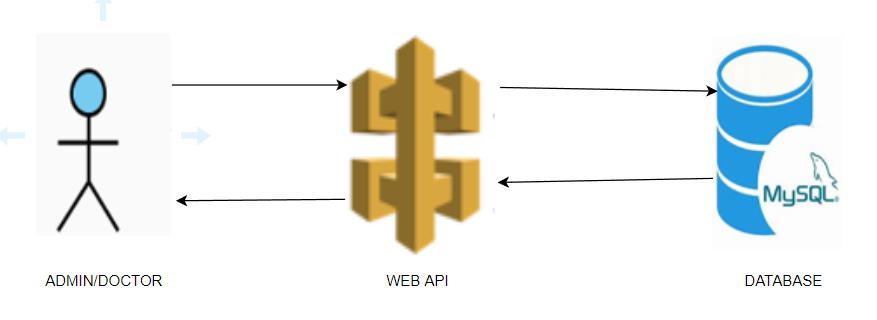
**There are two Actors in this project – Admin and Doctor**

# Design Pattern

|  |  |  |
| --- | --- | --- |
| # | Name | Description |
| 1 | Angular | Angular is a platform and framework for building single-page client applications using HTML and TypeScript. |
| 2 | API | Using HTTP requests, we will use the respective action to trigger various operations |
| 3 | SQL | It's used for relational databases. A SQL database is a collection of tables that stores a specific set of structured data |

# Diagrams

5.1 SOLUTION DIAGRAM



**Solution diagram**

5.2 ER DIAGRAM

Diagram

Description automatically generated

**Er Diagram**

5.3 USE CASE DIAGRAM

Diagram

Description automatically generated

**Use case diagram**

# 6.0 Solution Steps

**Doctor Registration**

1. Doctor User will enter the required details such as email, password to login.
2. He can register if he doesn’t have an account. These are the details needed to be provided during registration Name, contact, email, password after providing all the details he can proceed with register.

**Admin Login**

1. Admin need to enter the following details to login such as email and password.
2. On successful login, he Can perform various tasks.

**Doctor Functionalities**

1. After successful login, doctor has the following functionalities like view/search the drugs and buy them.
2. Initially there will a search box, where the doctor can search the drugs using the name of the drug and the respective results will be shown.
3. In the results the details like drug name, price, available quantity etc is shown.
4. He can select the drug he wants and proceed for the buy option.
5. After buying the desired drug, an email will be sent to the doctor.

**Admin Functionalities**

1. Admin has a wide range of functionalities over the website like view drugs, view suppliers, view doctor orders, verify orders, adding orders to Picked up orders after verification, view sales report.
2. **View drugs**: View and check details of each drug available.
3. He has the functionalities like Add/Edit/Delete drugs.
4. **View suppliers**: View and check details of each supplier.
5. He has the functionalities like Add/Edit/Delete Supplier.
6. **View doctor orders**: View orders placed by all the doctors and check if the order is valid or not.
7. **Verify orders**: On validating the order, he can verify the order.
8. **Add order to picked up section**: If the order has been picked up then add that order to picked up orders section.
9. **Views Sales Report**: Admin can have all the detailed view related to sales like what drugs were sold and how much money was received and what are the stocks needed to be updated.
10. Download/Print sales report.

# 7.0 Classes/function

|  |  |  |
| --- | --- | --- |
| **#** | **Class** | **Description** |
| 1 | Doctor.cs | Model holds the doctor schema details |
| 2 | Admin.cs | Model holds the admin schema details |
| 3 | Order.cs | Model holds the admin Doctor Orders schema details |
| 4 | Drug.cs | Model holds the admin Inventory schema details |
| 5 | Supplier.cs | Model holds the Supplier schema details |
| 6 | AdminDAL.cs | This Admin dal is present in dal layer where it contains the functionalities that the doctor performs like adding, deleting, updating drugs, suppliers etc |
| 7 | DoctorDAL.cs | This doctor dal layer will have the functionalities like  Searching drugs, ordering the drugs. |
| 8 | AdminController.cs | This is the controller class that contains the functions linked from admin dal layer and triggers respective functions as per our actions. |
| 9 | DoctorController.cs | This is the controller class that contains the functions linked from doctor dal layer and triggers respective functions as per our request. |

**8.1 Hardware Requirements**

# • Processor: Minimum 1.8 GHz. Recommended 2GHz or more.

# • Ethernet connection (LAN) OR a wireless adapter (Wi-Fi)

# • Hard Drive: Minimum 32 GB; Recommended 64 GB or more.

# • Memory (RAM): Minimum 4 GB; Recommended 8 GB or above.

# • OS: Windows.

# 8.2 Software Requirements

# • Visual studio .

# • SQL Server Management Studio .

# • Visual Studio Code.

# Data model/Table

|  |  |  |
| --- | --- | --- |
|  | **Doctor** |  |
| PK | DoctorId | VARCHAR(25) |
|  | DoctorName | VARCHAR(25) |
|  | Contact | VARCHAR(25) |
|  | Address | VARCHAR(50) |
|  | EmailId | VARCHAR(35) |
|  | Password | VARCHAR(25) |

|  |  |  |
| --- | --- | --- |
|  | **Order** |  |
| PK | OrderId | INT |
| FK | DoctorId | VARCHAR(25) |
|  | DrugId | INT |
|  | DrugPrice | DOUBLE |
|  | DrugName | VARCHAR(25) |
|  | DrugQuantity | INT |
|  | TotalAmount | DOUBLE |
|  | PickUpDate | DATETIME |
|  | IsPicked | VARCHAR(25) |

|  |  |  |
| --- | --- | --- |
|  | **Drug** |  |
| PK | DrugId | INT |
|  | DrugName | VARCHAR(25) |
|  | DrugPrice | DOUBLE |
|  | DrugQuantity | INT |
|  | MfdDate | DATETIME |
|  | ExpDate | DATETIME |
| FK | SupplierId | INT |

|  |  |  |
| --- | --- | --- |
|  | **Supplier** |  |
| PK | SupplierId | INT |
|  | SupplierName | VARCHAR(25) |
|  | EmailId | VARCHAR(25) |
|  | PhoneNumber | VARCHAR(25) |
|  | DrugAvailable | VARCHAR(25) |

# 10.0 API Canvas

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Micro Service | Path | Verb | API Description | Role | Auth |
| Doctor  -mgmt-ms | /doctor | POST | doctor registration | Doctor | True |
| Doctor  -mgmt-ms | /doctor/drug | GET | To get the details of the drugs | Doctor | True |
| Doctor  -mgmt-ms | /doctor/buy | POST | To buy the desired drug | Doctor | True |
| Admin-mgmt-ms | /admin | POST | Admin registration | Admin | True |
| Admin-mgmt-ms | /admin/drug | GET | To get the details of the drugs | Admin | True |
| Admin-mgmt-ms | /admin/drug | POST | To Add a new drug | Admin | True |
| Admin-mgmt-ms | /admin/drug | PUT | To edit an existing drug details | Admin | True |
| Admin-mgmt-ms | /admin/drug | DELETE | To delete a drug | Admin | True |
| Admin-mgmt-ms | /admin/supplier | GET | To view a supplier | Admin | True |
| Admin-mgmt-ms | /admin/supplier | POST | To add a new supplier | Admin | True |
| Admin-mgmt-ms | /admin/supplier | PUT | To edit an existing supplier details | Admin | True |
| Admin-mgmt-ms | /admin/supplier | DELETE | To delete a supplier | Admin | True |
| Admin-mgmt-ms | /admin/doctororder | GET | To view a order placed by a doctor | Admin | True |
| Admin-mgmt-ms | /admin/verifyorder | POST | To verify a order placed by a doctor | Admin | True |
| Admin-mgmt-ms | /admin/sales | GET | To view the over all sales | Admin | True |

# 11.0 HTTP Status Code

200 - Request succeeded

400 – Inputs are invalid

404 – Doctor user Not found

502 – Bad gateway

# 12.0 Request

**Register Doctor**

/doctor

{

“name”: “John”,

“password” :”sennna@17”,

“email”:”doctor@gmail.com”

“age” : 22,

“phone\_number” :” 0123456789”,

}

**Register Admin**

/Admin

{

“name”: “Admin”,

“password” :”Admin@17”,

“email”:”Admin@gmail.com”

“age” : 22,

“phone\_number” :”7550245985”,

}

**13.0 Response:**

**Register Doctor**

**If valid details**

{

"message": "doctor registered successfully"

}

status code: 201

**If invalid details**

{

“message”: “Inputs are not valid”

}, status code: 400

**If server encounters unexpected error**

{

“message” :” Internal server error ”

}, status code: 500

**Register Admin**

**If valid details**

{

"message": "Admin registered successfully"

}

status code: 201

**If invalid details**

{

“message”: “Inputs are not valid”

}, status code: 400

**If server encounters unexpected error**

{

“message” :” Internal server error ”

}, status code: 500

**Search drugs**

**If valid ID**

{

Drug­\_name”: “Levoctrizine”,

**If invalid ID**

{

“message”: “Drug not found”

}, status code:404

**If server encounters unexpected error**

{

“message” :” Internal server error ”

}, status code: 500

**Delete Drug**

**If valid ID**

{

“Drug Deleted successfully”

}

status code: 200

**If invalid ID**

{

“message”: “Customer not found”

}, status code:404

**If server encounters unexpected error**

{

“message” :” Internal server error ”

}, status code: 500

**Delete Supplier**

**If valid ID**

{

“Supplier Deleted successfully”

}

status code: 200

**If invalid ID**

{

“message”: “Customer not found”

}, status code:404

**If server encounters unexpected error**

{

“message” :” Internal server error ”

}, status code: 500

**Delete Doctor**

**If valid ID**

{

“Doctor Deleted successfully”

}

status code: 200

**If invalid ID**

{

“message”: “Customer not found”

}, status code:404

**If server encounters unexpected error**

{

“message” :” Internal server error ”

}, status code: 500