VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELGAUM-590 014

A Mini-Project Report
on

MEDICAL SUPPLY MANAGEMENT

Submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

Submitted by
ABHISHEK PANDEY, 1VE21CS004
ATUL YADAV, 1VE21CS024

Under the Guidance of
Mrs. Kulkarni Varsha
Assistant Professor
Department of Computer Science and Engineering
Sri Venkateshwara College of Engineering, Bangalore-562 157

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SRI VENKATESHWARA COLLEGE OF ENGINEERING
BANGALORE - 562 157

2023-2024

SRI VENKATESHWARA COLLEGE OF ENGINEERING,
Vidyanagar, Bangalore – 562 157
Department of Computer Science and Engineering

Accredited by NAAC & NBA*
SVCE BENGALURU
SRI VENKATESHWARA COLLEGE OF ENGINEERING
— Affiliated to VTU, Approved by AICTE, Recognised by UGC u/s 2(f) & 12(B)—

CERTIFICATE

This is to certify that the Mini-Project entitled **"Medical Supply Management"** carried out by **Mr. Abhishek Pandey 1VE21CS004 and Mr. Atul Yadav 1VE21CS024** of V Semester students of Sri Venkateshwara College of Engineering, in partial fulfillment for the award of Bachelor of Engineering in Computer Science and Engineering of **Visvesvaraya Technological University, Belgaum** during the academic year 2023-2024. The Mini-Project report has been approved as it satisfies the academic requirements in respect of DBMS Laboratory with Mini-Project (21CSL55) work prescribed for the said Degree.

Signature of the Guide	**Signature of the HOD**
Mrs. Kulkarni Varsha	**Dr. Hema M S**
Asst. Professor, Dept. of CS&E	HOD, Dept. of CS&E
SVCE, Bangalore	SVCE, Bangalore

Name of the Examiners:	**Signature with Date**
1.	
2.	

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without complementing those who made it possible, whose guidance and encouragement made our efforts successful.

My sincere thanks to highly esteemed institution **SRI VENKATESHWARA COLLEGE OF ENGINEERING** for grooming up me in to be software engineer.

I express our sincere gratitude to **Dr. Nageswara Guptha**, Principal, **SVCE**, Bengaluru for providing the required facility.

I am extremely thankful to **Dr. Hema M S**, **HOD of CSE**, SVCE for providing support and encouragement.

I am grateful to **Mrs. Kulkarni Varsha**, Asst. Professor, Dept. of **CSE**, **SVCE** who helped me to complete this project successfully by providing guidance, encouragement and valuable suggestion during entire period of the project. I thank all my computer science staff and others who helped directly or indirectly to meet my project work with grand success.

Finally, I am grateful to my parents and friends for their invaluable support guidance and encouragement.

ABHISHEK PANDEY [1VE21CS004]

ATUL YADAV[1VE21CS024]

ABSTRACT

Pharmacy Supply Management in healthcare systems is evaluated with a particular focus on the distribution of medicines from a wholesaler to clinics. Currently, there are issues with service levels to clinics that need addressing. The value of the paper arises from providing a detailed analysis of a healthcare supply chain in the developing world and diagnosis the parameters involved in inventory.

Pharmaceutical practices have evolved over time to become fully encompassed in all aspects of pharmacy itself. Such practices include: dispensing of drugs, consultation, drug regulation, and the sale of these drugs. Creating an Online Pharmaceutical Management System would help in pharmaceutical practices for all parties involved. It is eminent that the system provides a safe, secure and verified platform for all parties which help to bridge the communication gap and provide legitimate drugs. Therefore, if all recommendations are strictly adhered to, there will be strict monitoring and regulation of how drugs are circulated and a decrease in the spread of fake drugs.

Contents

CHAPTER-1

INTRODUCTION

1.1 OBJECTIVES:

- The main objective of the project is to design and develop a user friendly-system
- Easy to use and an efficient computerized system.
- To develop an accurate and flexible system, it will eliminate data redundancy.
- To study the functioning of pharmacy supply management System.
- To make a software fast in processing, with good user interface.
- To make software with good user interface so that user can change it and it should be used for a long time without error and maintenance.
- To provide synchronized and centralized farmer and seller database.
- Computerization can be helpful as a means of saving time and money.
- To provide better Graphical User Interface (GUI).
- Less chances of information leakage.
- Provides Security to the data by using login and password method.
- To provide immediate storage and retrieval of data and information.
- Improving arrangements for medicines coordination.
- Reducing paperwork.

1.2 LIMITATIONS:

- Time consumption in data entry as the records are to be manually maintained consumes a lot of time.
- Lot of paper work is involved as the records are maintained in the files and registers.
- Storage Requires as files and registers are used the storage space requirement is increased.
- Less Reliable use of papers for storing valuable data information is not at all reliable.
- Aadhar linkage with the official aadhar database has not been done.

CHAPTER-2

STUDY OF EXISTING SYSTEM

2.1 CASE STUDY

Rising debt, cost-cutting, and layoffs in health care-delivery facilities, alluded to earlier. Models for the design and operation of supply chain networks may be steady state or dynamic and may be deterministic or deal with uncertainties (particularly in product demands). Research in this field started very early on, with location-allocation problems forming part of the earlier set of ‘‘classical’’ operations research problems. The gap between the growing demand and available supply of high-quality, cost effective, and timely health care continues to be a daunting challenge not only in developing and underdeveloped countries, but also in developed countries. Further, the issues involved with the supply chain design in 3developing countries are prevalent in developed countries, especially with the rising number of uninsured and jobless among the patient populations and with the budget deficits. Thus the project is a sincere effort in simplifying the tasks of administrators in an easily usable format.

2.2 PROPOSED SYSTEM

While there has been no consensus on the definition of Pharmacy Supply Management in the literature, they have proposed that researchers adopt the below definition to allow for the coherent development of theory in the area. In order to have a successful supply management, we need to make many decisions related to the flow of information, product, and funds. Each decision should be made in a way to increase the whole supply chain profitability. Supply management is more complex in healthcare and other industries because of the impact on people’s health requiring adequate and accurate medical supply according to the patient’s need.

CHAPTER 3. DATABASE DESIGN

3.1 SOFTWARE REQUIREMENTS SPECIFICATION

3.1.2 SOFTWARE REQUIREMENTS:

Frontend- HTML, CSS, Java Script, Bootstrap

Backend-Python flask (Python 3.7) , SQLAlchemy,

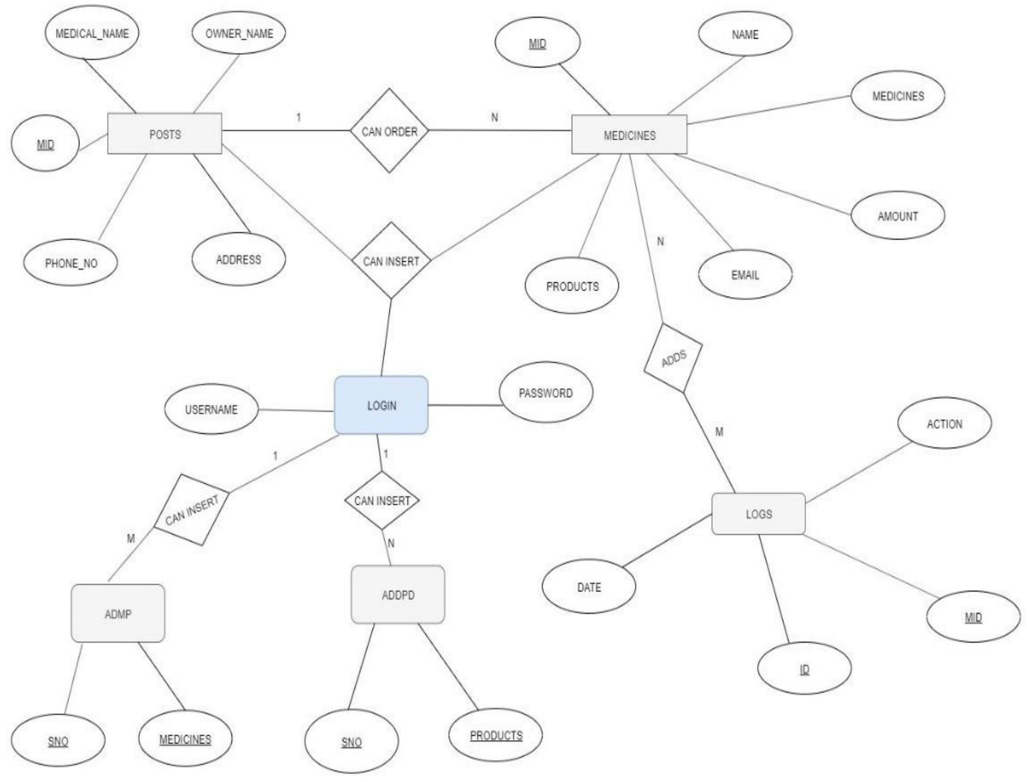
- Operating System: Windows 10
- Google Chrome/Internet Explorer
- AMPPS (Version-3.7)
- Python main editor (user interface): PyCharm Community
- workspace editor: Sublime text 3

HARDWARE REQUIREMENTS:

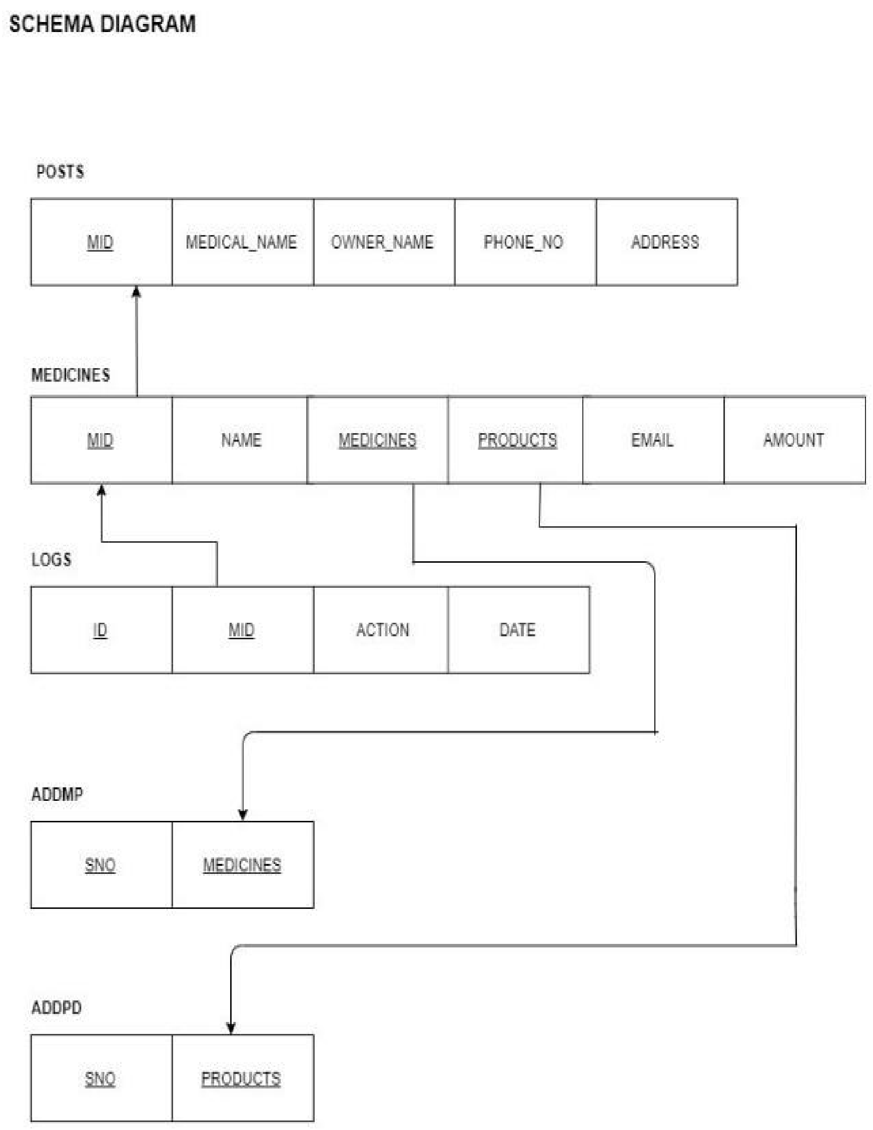
- Computer with a 1.1 GHz or faster processor
- Minimum 2GB of RAM or more
- 2.5 GB of available hard-disk space
- 5400 RPM hard drive
- 1366 × 768 or higher-resolution display
- DVD-ROM driveE

3.2 CONCEPTUAL DESIGN:

3.2.1 E-R DIAGRAM:



3.2.2 SCHEMA DIAGRAM:



3.3 IMPLEMENTATION:

PYTHON:

An "implementation" of Python should be taken to mean a program or environment which provides support for the execution of programs written in the Python language, as represented by the Python reference implementation.

There have been and are several distinct software packages providing of what we all recognize as Python, although some of those are more like distributions or variants of some existing implementation than a completely new implementation of the language.

DBMS: (MySQL)

A Database Management System (DBMS) is computer software designed for the purpose of managing databases, a large set of structured data, and run operations on the data requested by numerous users. Typical examples of DBMSs include Oracle, DB2, Microsoft Access, Microsoft SQL Server, Firebird, PostgreSQL, MySQL, SQLite, FileMaker and Sybase Adaptive Server Enterprise. DBMSs are typically used by Database administrators in the creation of Database systems.

A DBMS is a complex set of software programs that controls the organization, storage, management, and retrieval of data in a database. A DBMS includes:

- A modeling language to define the schema of each database hosted in the DBMS, according to the DBMS data model.
- The dominant model in use today is the ad hoc one embedded in SQL, despite the objections of purists who believe this model is a corruption of the relational model, since it violates several of its fundamental principles for the sake of practicality and performance. Many DBMSs also support the Open Database Connectivity API that supports a standard way for programmers to access the DBMS.

- Data structures (fields, records, files and objects) optimized to deal with very large amounts of data stored on a permanent data storage device (which implies relatively slow access compared to volatile main memory).A database query language and report writer to allow users to interactively interrogate the database, analyze its data and update it according to the users privileges on data.

- Data security prevents unauthorized users from viewing or updating the database. Using passwords, users are allowed access to the entire database or subsets of it called sub schemas. For example, an employee database can contain all the data about an individual employee, but one group of users may be authorized to view only payroll data, while others are allowed access to only work history and medical data.
- If the DBMS provides a way to interactively enter and update the database, as well as interrogate it, this capability allows for managing personal databases. However, it may not leave an audit trail of actions or provide the kinds of controls necessary in a multi-user organization. These controls are only available when a set of application programs are customized for each data entry and updating function.

 A transaction mechanism, that ideally would guarantee the ACID properties, in order to ensure data integrity, despite concurrent user accesses (concurrency control), and faults (fault tolerance).

  - It also maintains the integrity of the data in the database.
  - The DBMS can maintain the integrity of the database by not allowing more than one user to update the same record at the same time. The DBMS can help prevent duplicate records via unique index constraints; for example, no two customers with the same customer numbers (key fields) can be entered into the database. See ACID properties for more information (Redundancy avoidance).

When a DBMS is used, information systems can be changed much more easily as the organization's information requirements change. to the Organizations may use one kind of DBMS for daily transaction processing and then move the detail onto another computer that uses another DBMS better suited for random inquiries and analysis. Overall systems design decisions are performed by data administrators and systems analysts. Detailed database design is performed by database administrators.

SQL:

Structured Query Language (SQL) is the language used to manipulate relational databases.

SQL is tied very closely with the relational model.

- In the relational model, data is stored in structures called relations or tables.
- MySQL is a database server, ideal for both small and large application
- Data definition: Defining tables and structures in the database (DDL used to create, alter and drop schema objects such as tables and indexes).

4.2: Stored Procedure

Routine name: proc 1, post proc

Type: procedure

Definition: Select * from posts;

Select * from medicines;

4.3: Triggers

It is the special kind of stored procedure that automatically executes when an event occurs in the database. Triggers used:

1: Trigger name: on insert

Table: medicines

Time: after

Event: insert

Definition: INSERT INTO logs VALUES (null, new.mid, ’inserted’, NOW());

2: Trigger name: on delete

Table: medicines

Time: after

Event: delete

Definition: INSERT INTO logs VALUES(null, old.mid, ’deleted’, NOW());

BACKEND PYHTON WITH MYSQL CODE

```
from flask import Flask, render_template, request, session, redirect, flash
from flask_sqlalchemy import SQLAlchemy
import json
with open('project\config.json','r') as c:
    params = json.load(c)["params"]
local_server = True
app = Flask(__name__)
app.secret_key = 'super-secret-key'
if(local_server):
    app.config['SQLALCHEMY_DATABASE_URI'] = params['local_uri']
else:
    app.config['SQLALCHEMY_DATABASE_URI'] = params['proud_uri']
db = SQLAlchemy(app)
class Medicines(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    amount = db.Column(db.Integer, nullable=False)
    name = db.Column(db.String(500), nullable=False)
    medicines= db.Column(db.String(500), nullable=False)
    products = db.Column(db.String(500), nullable=False)
    email = db.Column(db.String(120), nullable=False)
    mid = db.Column(db.String(120), nullable=False)
class Posts(db.Model):
    mid = db.Column(db.Integer, primary_key=True)
    medical_name = db.Column(db.String(80), nullable=False)
    owner_name = db.Column(db.String(200), nullable=False)
    phone_no = db.Column(db.String(200), nullable=False)
    address = db.Column(db.String(120), nullable=False)
class Addmp(db.Model):
    sno = db.Column(db.Integer,  primary_key=True)
    medicine = db.Column(db.String, nullable=False)
class Addpd(db.Model):
    sno = db.Column(db.Integer,  primary_key=True)
    product = db.Column(db.String, nullable=False)
class Logs(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    mid = db.Column(db.String, nullable=True)
    action = db.Column(db.String(30), nullable=False)
    date = db.Column(db.String(100), nullable=False)
@app.route("/")
def hello():
    return render_template('index.html', params=params)
@app.route("/index")
def home():
    return render_template('dashbord.html', params=params)
@app.route("/search",methods=['GET','POST'])
def search():
    if request.method == 'POST':
        name = request.form.get('search')
        post = Addmp.query.filter_by(medicine=name).first()
        pro = Addpd.query.filter_by(product=name).first()
        if (post or pro):
            flash("Item Is Available.", "primary")
        else:
            flash("Item is not Available.", "danger")
    return render_template('search.html', params=params)
@app.route("/details", methods=['GET','POST'])
def details():
    if ('user' in session and session['user'] == params['user']):
        posts =Logs.query.all()
        return render_template('details.html', params=params, posts=posts)
@app.route("/aboutus")
def aboutus():
    return render_template('aboutus.html', params=params)
@app.route("/insert", methods = ['GET','POST'])
def insert():
    if (request.method == 'POST'):
        '''ADD ENTRY TO THE DATABASE'''
        mid=request.form.get('mid')
        medical_name = request.form.get('medical_name')
        owner_name = request.form.get('owner_name')
        phone_no = request.form.get('phone_no')
        address = request.form.get('address')
        push = Posts(mid=mid,medical_name=medical_name, owner_name=owner_name,
phone_no=phone_no, address=address)
        db.session.add(push)
        db.session.commit()
        flash("Thanks for submitting your details","danger")
    return render_template('insert.html',params=params)
```

```
@app.route("/addmp", methods = ['GET','POST'])
def addmp():
    if (request.method == 'POST'):
        '''ADD ENTRY TO THE DATABASE'''
        newmedicine = request.form.get('medicine')
        push=Addmp(medicine=newmedicine,)
        db.session.add(push)
        db.session.commit()
        flash("Thanks for adding new items", "primary")
    return render_template('search.html', params=params)
@app.route("/addpd", methods = ['GET','POST'])
def addpd():
    if (request.method == 'POST'):
        '''ADD ENTRY TO THE DATABASE'''
        newproduct = request.form.get('product')
        push=Addpd(product=newproduct,)
        db.session.add(push)
        db.session.commit()
        flash("Thanks for adding new items", "primary")
    return render_template('search.html', params=params)
@app.route("/list",methods=['GET','POST'])
def post():
    if ('user' in session and session['user'] == params['user']):
        posts=Medicines.query.all()
        return render_template('post.html', params=params, posts=posts)
@app.route("/items",methods=['GET','POST'])
def items():
    if ('user' in session and session['user'] == params['user']):
        posts=Addmp.query.all()
        return render_template('items.html', params=params,posts=posts)
```

```
@app.route("/items2", methods=['GET','POST'])
def items2():
    if ('user' in session and session['user'] == params['user']):
        posts=Addpd.query.all()
        return render_template('items2.html',params=params,posts=posts)
@app.route("/sp",methods=['GET','POST'])
def sp():
    if ('user' in session and session['user'] == params['user']):
        posts=Medicines.query.all()
        return render_template('store.html', params=params,posts=posts)
@app.route("/logout")
def logout():
    session.pop('user')
    flash("You are logout", "primary")
    return redirect('/login')
@app.route("/login",methods=['GET','POST'])
def login():
    if ('user' in session and session['user'] == params['user']):
        posts = Posts.query.all()
        return render_template('dashbord.html',params=params,posts=posts)
    if request.method=='POST':
        username=request.form.get('uname')
        userpass=request.form.get('password')
        if(username==params['user'] and userpass==params['password']):
            session['user']=username
            posts=Posts.query.all()
            flash("You are Logged in", "primary")
            return render_template('index.html',params=params,posts=posts)
        else:
            flash("wrong password", "danger")
    return render_template('login.html', params=params)
@app.route("/edit/<string:mid>",methods=['GET','POST'])
def edit(mid):
    if('user' in session and session['user']==params['user']):
        if request.method =='POST':
            medical_name=request.form.get('medical_name')
            owner_name=request.form.get('owner_name')
            phone_no=request.form.get('phone_no')
            address=request.form.get('address')
            if mid==0:

posts=Posts(medical_name=medical_name,owner_name=owner_name,phone_no=phone_no,a
ddress=address)
                db.session.add(posts)
                db.session.commit()
            else:
                post=Posts.query.filter_by(mid=mid).first()
                post.medical_name=medical_name
                post.owner_name=owner_name
                post.phone_no=phone_no
                post.address=address
                db.session.commit()
                flash("data updated ", "success")
                return redirect('/edit/'+mid)
        post = Posts.query.filter_by(mid=mid).first()
        return render_template('edit.html',params=params,post=post)
#        if user is logged in
#delete
@app.route("/delete/<string:mid>", methods=['GET', 'POST'])
def delete(mid):
    if ('user' in session and session['user']==params['user']):
        post=Posts.query.filter_by(mid=mid).first()
        db.session.delete(post)
        db.session.commit()
        flash("Deleted Successfully", "warning")
    return redirect('/login')
@app.route("/deletemp/<string:id>", methods=['GET', 'POST'])
def deletemp(id):
    if ('user' in session and session['user']==params['user']):
        post=Medicines.query.filter_by(id=id).first()
        db.session.delete(post)
        db.session.commit()
        flash("Deleted Successfully", "primary")
    return redirect('/list')
@app.route("/medicines", methods = ['GET','POST'])
def medicine():
    if(request.method=='POST'):
        '''ADD ENTRY TO THE DATABASE'''
        mid=request.form.get('mid')
        name=request.form.get('name')
        medicines=request.form.get('medicines')
        products=request.form.get('products')
        email=request.form.get('email')
        amount=request.form.get('amount')

entry=Medicines(mid=mid,name=name,medicines=medicines,products=products,email=email
,amount=amount)
        db.session.add(entry)
        db.session.commit()
        flash("Data Added Successfully","primary")
    return render_template('medicine.html',params=params)
app.run(debug=True)
```

FRONT END CODE

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <meta name="description" content="">

  <meta name="author" content="">

  <title>{{params['blog_name']}}</title>

  <!-- Bootstrap core CSS -->

  <link                href="{{url_for('static',filename='vendor/bootstrap/css/bootstrap.min.css')}}"
rel="stylesheet">

  <!-- Custom fonts for this template -->

  <link              href="{{url_for('static',filename='vendor/fontawesome-free/css/all.min.css')}}"
rel="stylesheet" type="text/css">

  <link              href='https://fonts.googleapis.com/css?family=Lora:400,700,400italic,700italic'
rel='stylesheet' type='text/css'>

  <link
href='https://fonts.googleapis.com/css?family=Open+Sans:300italic,400italic,600italic,700ital
ic,800italic,400,300,600,700,800' rel='stylesheet' type='text/css'>

  <!-- Custom styles for this template -->

  <link href="{{url_for('static',filename='css/clean-blog.min.css')}}" rel="stylesheet">

</head>
```

```
<body>

 <!-- Navigation -->

 <nav class="navbar navbar-expand-lg navbar-light fixed-top" id="mainNav">

  <div class="container">

   <a class="navbar-brand" href="#">{{params['blog_name']}}</a>

  <!--      <button class="navbar-toggler navbar-toggler-right" type="button" data-toggle="collapse" data-target="#navbarResponsive" aria-controls="navbarResponsive" aria-expanded="false" aria-label="Toggle navigation">

    Menu

    <i class="fas fa-bars"></i>

   </button> -->

   <div class="collapse navbar-collapse" id="navbarResponsive">

    <ul class="navbar-nav ml-auto">

     <li class="nav-item">

      <a class="nav-link" href="/login">home</a>

     </li>

     <li class="nav-item">

      <a class="nav-link" href="/insert">add medical information</a>

     </li>

     <li class="nav-item">

      <a class="nav-link" href="/list">view ordered list</a>

     </li>

     <li class="nav-item">

      <a class="nav-link" href="/medicines ">Order medicines/products</a>
```

```
        </li>

        <li class="nav-item">

          <a class="nav-link" href="/details ">Details</a>

        </li>

        <li class="nav-item">

          <a class="nav-link" href="/search ">add/search items</a>

        </li>

        <li class="nav-item">

          <a class="nav-link" href="/aboutus">about us</a>

        </li>

        <li class="nav-item">

          <a onclick="return confirm('Are you sure to logout?');" class="nav-link"
href="/logout">logout</a>

        </li>

              </ul>

      </div>

    </div>

  </nav>

  {% block body %}  {% endblock %}

   <hr>

  <!-- Footer -->

  <footer>

    <div class="container">
```

```
<div class="row">
  <div class="col-lg-8 col-md-10 mx-auto">
    <ul class="list-inline text-center">
      <li class="list-inline-item">
        <a href="{{params['ins_url']}}" target=_blank>
          <span class="fa-stack fa-lg">
            <i class="fas fa-circle fa-stack-2x"></i>
            <i class="fab fa-instagram fa-stack-1x fa-inverse"></i>
          </span>
        </a>
      </li>
      <li class="list-inline-item">
       <a href="{{params['fb_url']}}" target=_blank>
          <span class="fa-stack fa-lg">
            <i class="fas fa-circle fa-stack-2x"></i>
            <i class="fab fa-facebook-f fa-stack-1x fa-inverse"></i>
          </span>
        </a>
      </li>
      <li class="list-inline-item">
        <a href="{{params['gh_url']}}" target=_blank>
          <span class="fa-stack fa-lg">
            <i class="fas fa-circle fa-stack-2x"></i>
```

```
                <i class="fab fa-github fa-stack-1x fa-inverse"></i>

              </span>

            </a>

          </li>

        </ul>

        <p class="copyright text-muted">Copyright &copy; Your DBMS website 2024</p>

      </div>

    </div>

  </div>

 </footer>

 <!-- Bootstrap core JavaScript -->

 <script src="{{url_for('static',filename='vendor/jquery/jquery.min.js')}}"></script>

 <script
src="{{url_for('static',filename='vendor/bootstrap/js/bootstrap.bundle.min.js')}}"></script>

 <!-- Custom scripts for this template -->

 <script src="{{url_for('static',filename='js/clean-blog.min.js')}}"></script>

</body>

</html>
```

Login page

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8 " >
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <meta name="description" content="">
    <meta name="author" content="Mark Otto, Jacob Thornton, and Bootstrap contributors">
    <meta name="generator" content="Jekyll v3.8.5">
  <title>loginpage</title>
  <link rel="canonical" href="https://getbootstrap.com/docs/4.3/examples/sign-in/">
  <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
  <link href="/docs/4.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
     <link href="{{url_for('static',filename='css/signin.css')}}" rel="stylesheet">
   {% with messages=get_flashed_messages(with_categories=true) %}
  {% if messages %}
  {% for category, message in messages %}
  <div class="alert alert-{{category}} alert-dismissible fade show" role="alert">
   {{message}}
  <button type="button" class="close" data-dismiss="alert" aria-label="Close">
   <span aria-hidden="true">&times;</span>
  </button>
</div>
{% endfor %}
  {% endif %}
  {% endwith %}
  </head>
  <body class="text-center">
   <form class="form-signin" action="/login" method="post">
    {% set fname='img/' +params['login_image']  %}
  <img class="mb-2" src="{{url_for('static',filename=fname)}}" alt="" width="180"
height="180">
  <h3><span class="badge badge-dark mb-2">LOGIN</span></h3>
  <label for="uname" class="sr-only">USER</label>
  <input type="text" id="uname" name="uname" autocomplete="off" class="form-control"
placeholder="enter username" required autofocus>
  <label for="inputPassword" class="sr-only">PASSWORD</label>
    <br>
   <input type="password" id="password" name="password" class="form-control"
placeholder="enter password" required>
  <div class="checkbox mb-3">
   <label>
    <input type="checkbox" value="remember-me"> Remember me
   </label>
  </div>
  <button class="btn btn-lg btn-dark btn-block" type="submit">Sign in</button>
  <p style="color:rgb(0, 0, 0)",class="mt-3 mb-2">DONE BY: ABHISHEK PANDEY &
ATUL YADAV
```

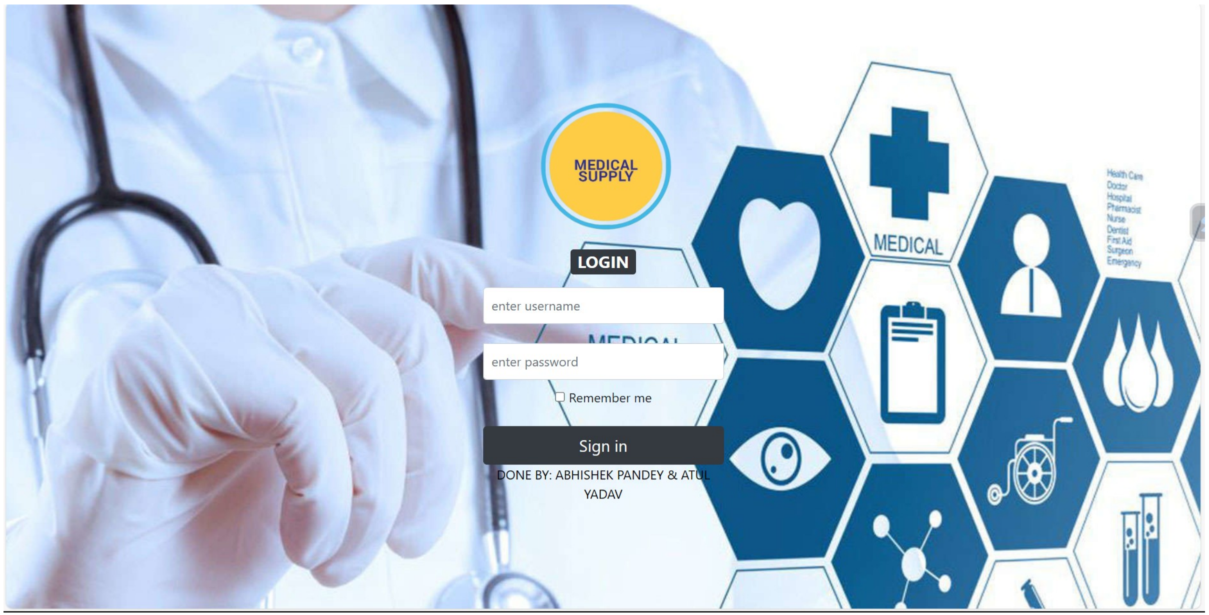
```
  </p>
</form>
</body>
<script          src="https://code.jquery.com/jquery-3.3.1.slim.min.js"          integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
<script           src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
crossorigin="anonymous"></script>
<script                 src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>
</html>
```

CHAPTER 4

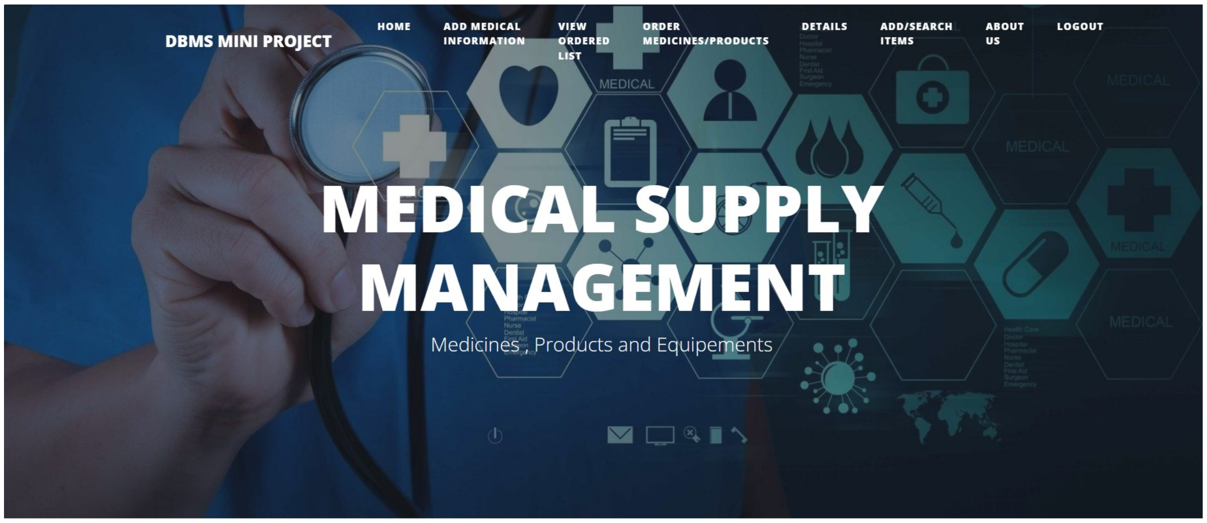
USER INTERFACES

4.1 SCREEN SHOTS

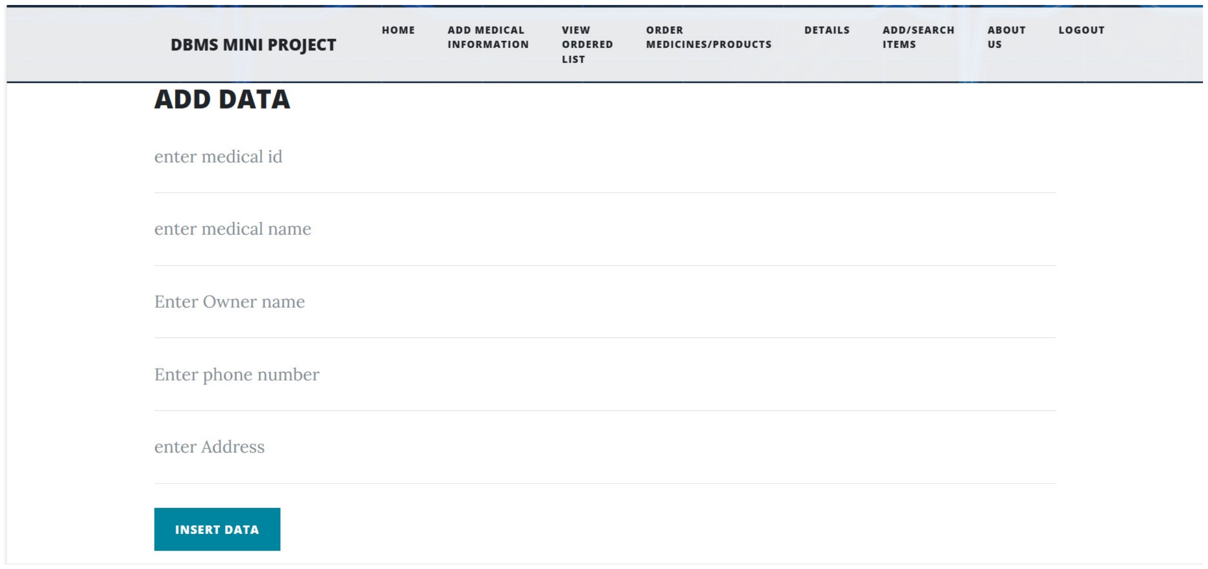
LOGIN PAGE:



After Login:



ADD MEDICAL SHOP INFO



ADD DATA

enter medical id

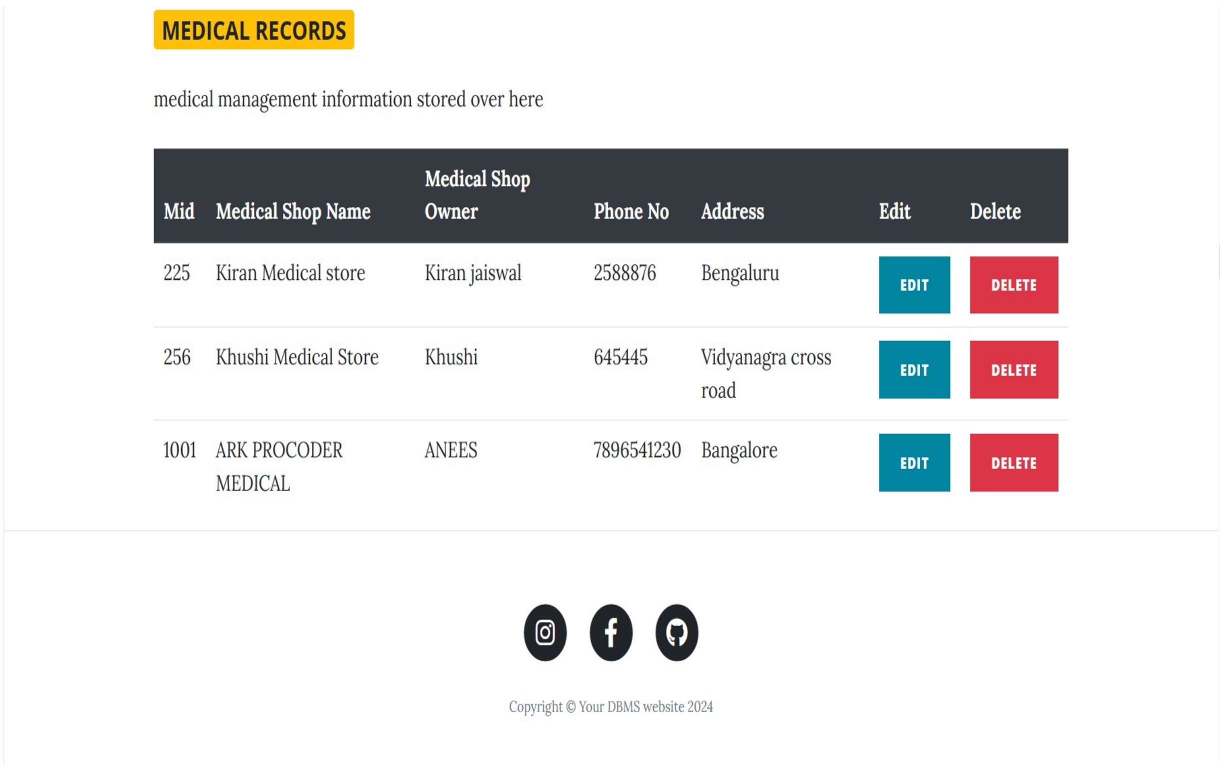
enter medical name

Enter Owner name

Enter phone number

enter Address

MEDICAL RECORDS

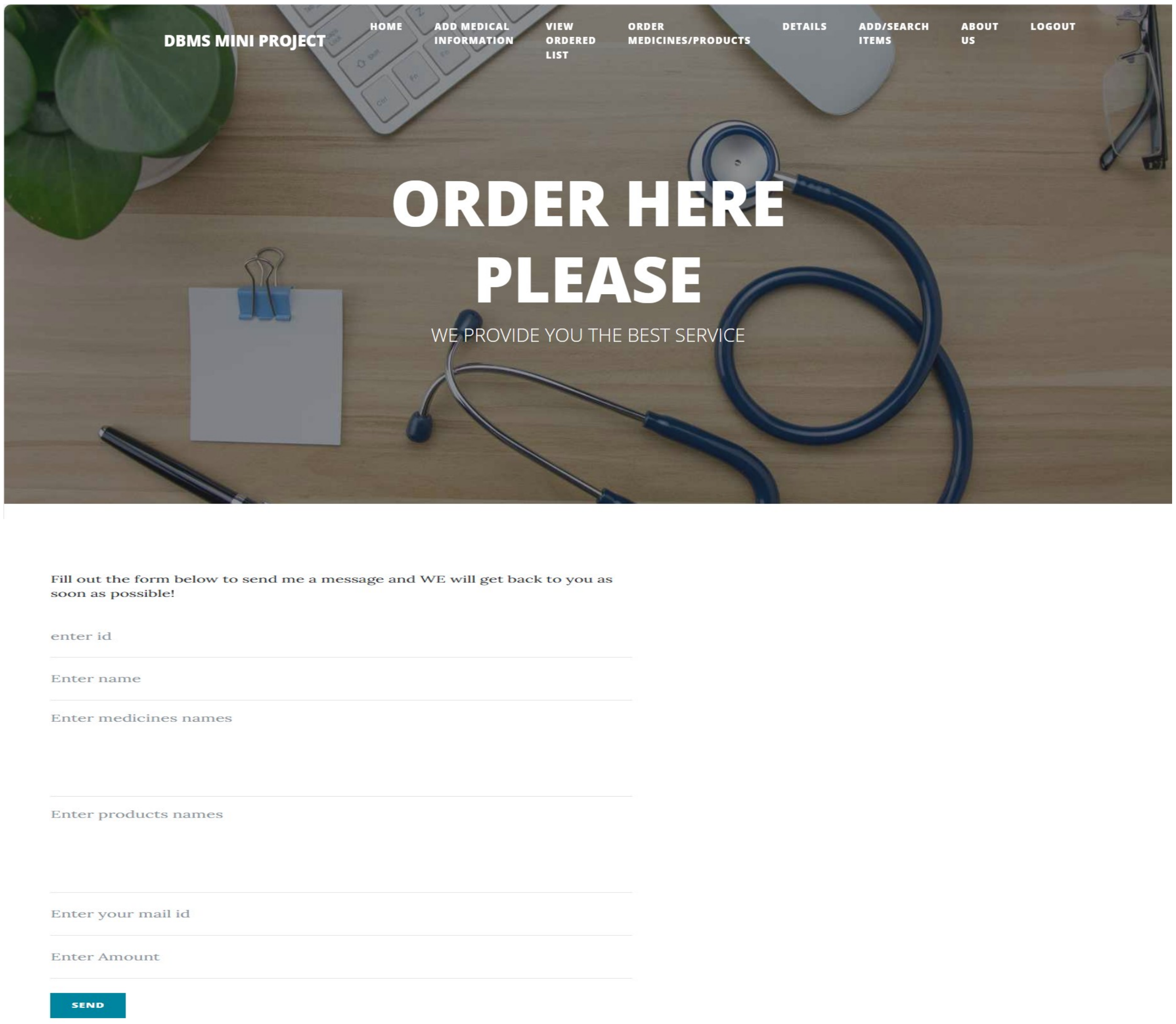

MEDICAL RECORDS

medical management information stored over here

Mid	Medical Shop Name	Medical Shop Owner	Phone No	Address	Edit	Delete
225	Kiran Medical store	Kiran jaiswal	2588876	Bengaluru	EDIT	DELETE
256	Khushi Medical Store	Khushi	645445	Vidyanagra cross road	EDIT	DELETE
1001	ARK PROCODER MEDICAL	ANEES	7896541230	Bangalore	EDIT	DELETE

Copyright © Your DBMS website 2024

ORDERS



ORDER HERE PLEASE

WE PROVIDE YOU THE BEST SERVICE

Fill out the form below to send me a message and WE will get back to you as soon as possible!

enter id

Enter name

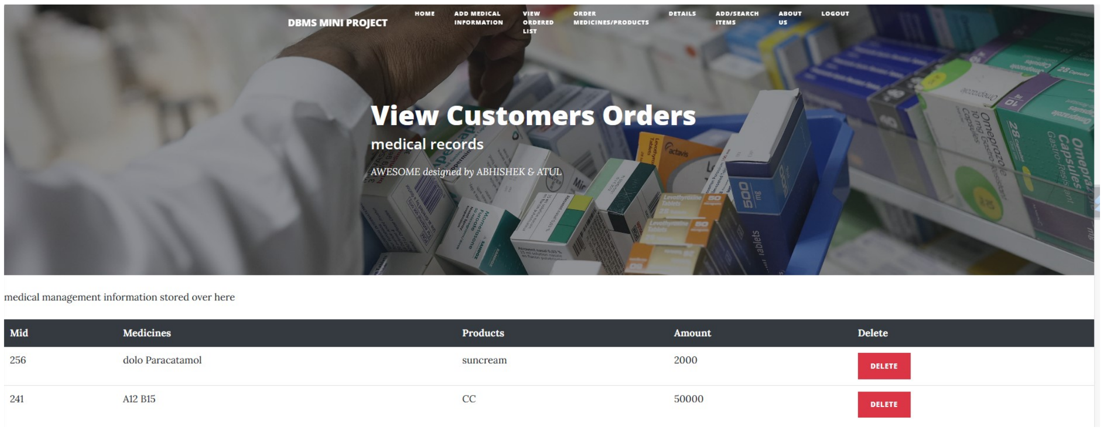
Enter medicines names

Enter products names

Enter your mail id

Enter Amount

ORDERED LIST



View Customers Orders

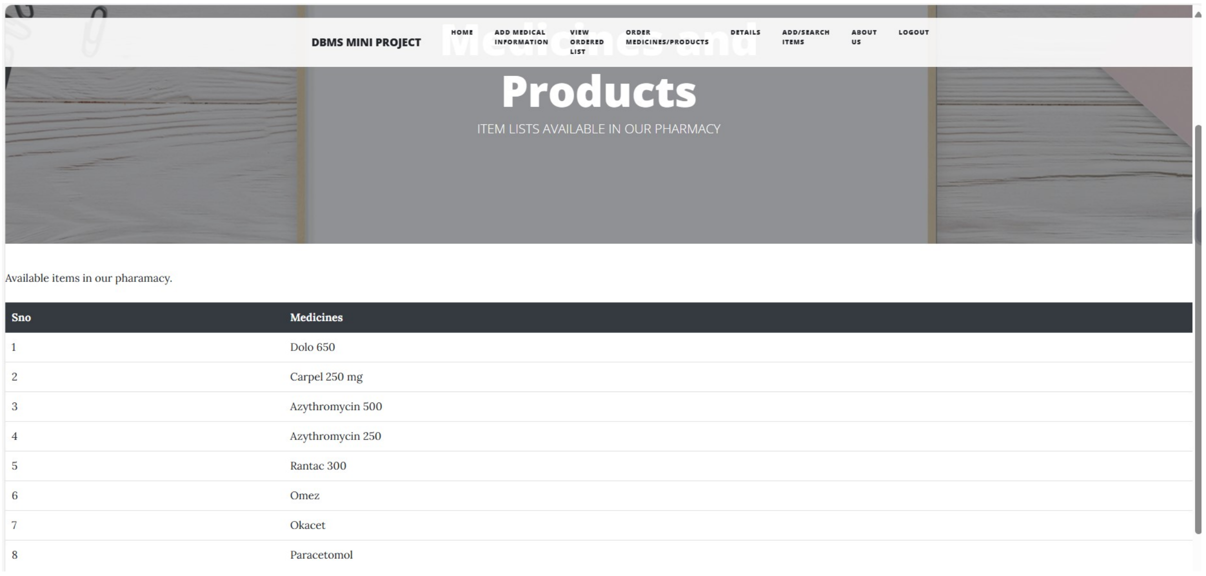
medical records

AWESOME *designed by* ABHISHEK & ATUL

medical management information stored over here

Mid	Medicines	Products	Amount	Delete
256	dolo Paracatamol	suncream	2000	DELETE
241	A12 B15	CC	50000	DELETE

ADDING MEDICINES AND PRODUCTS IN PHARMACY



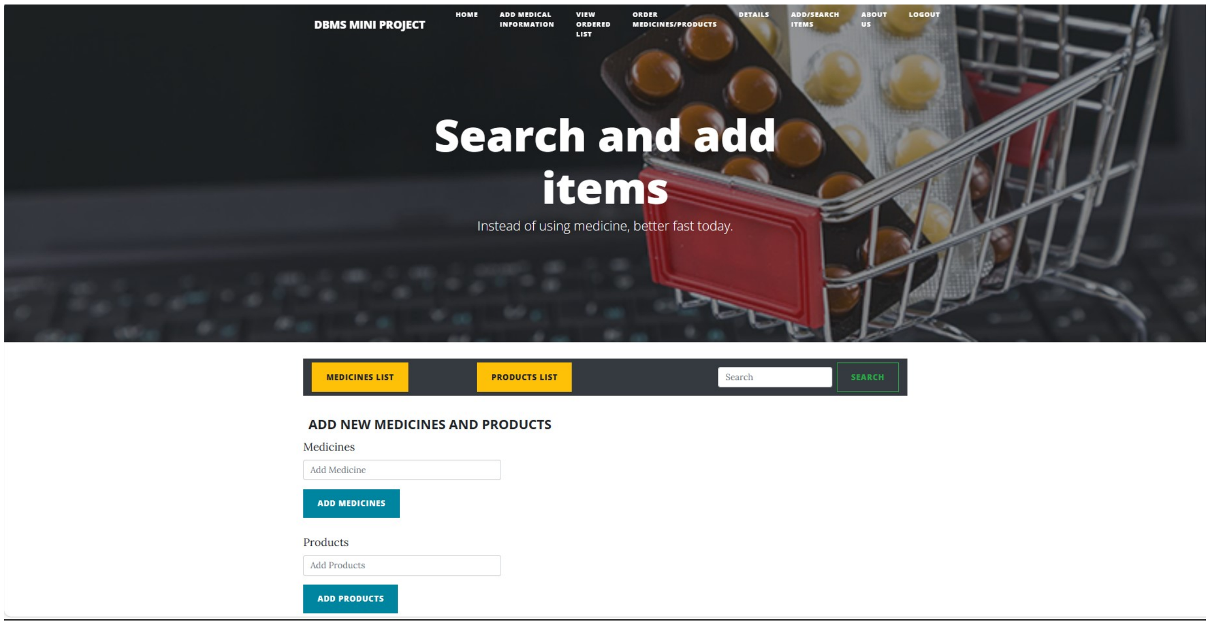
Medicines and Products

ITEM LISTS AVAILABLE IN OUR PHARMACY

Available items in our pharamacy.

Sno	Medicines
1	Dolo 650
2	Carpel 250 mg
3	Azythromycin 500
4	Azythromycin 250
5	Rantac 300
6	Omez
7	Okacet
8	Paracetomol

SEARCH MEDICINS AND PRODUCTS



Search and add items

Instead of using medicine, better fast today.

ADD NEW MEDICINES AND PRODUCTS

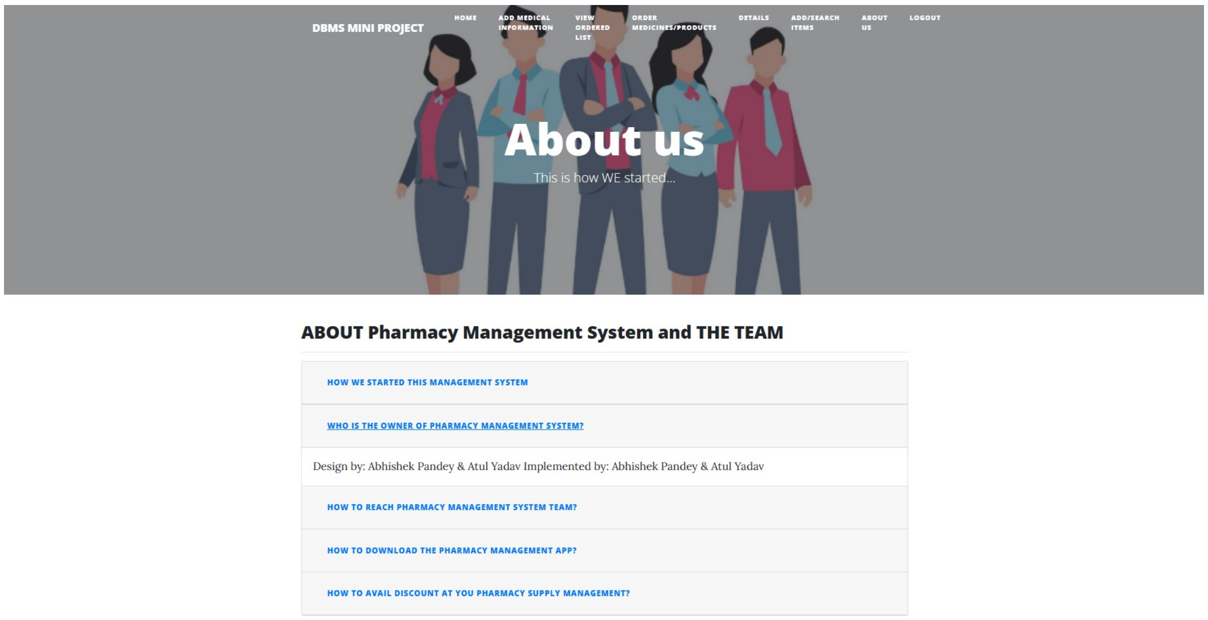
Medicines

Add Medicine

Products

Add Products

ABOUT US



About us

This is how WE started...

ABOUT Pharmacy Management System and THE TEAM

HOW WE STARTED THIS MANAGEMENT SYSTEM

WHO IS THE OWNER OF PHARMACY MANAGEMENT SYSTEM?

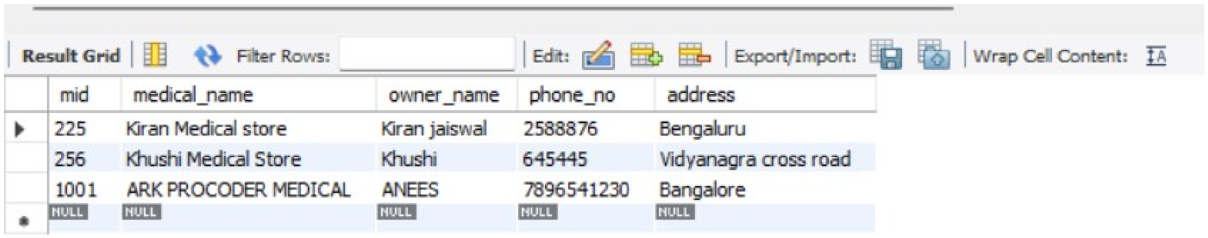
Design by: Abhishek Pandey & Atul Yadav Implemented by: Abhishek Pandey & Atul Yadav

HOW TO REACH PHARMACY MANAGEMENT SYSTEM TEAM?

HOW TO DOWNLOAD THE PHARMACY MANAGEMENT APP?

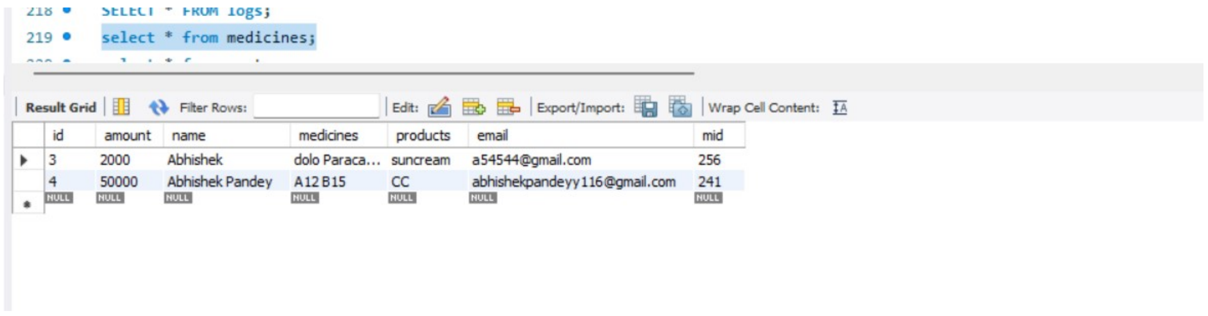
HOW TO AVAIL DISCOUNT AT YOU PHARMACY SUPPLY MANAGEMENT?

MEDICAL SHOP INFO(POSTS)



	mid	medical_name	owner_name	phone_no	address
▶	225	Kiran Medical store	Kiran jaiswal	2588876	Bengaluru
	256	Khushi Medical Store	Khushi	645445	Vidyanagra cross road
	1001	ARK PROCODER MEDICAL	ANEES	7896541230	Bangalore
*	NULL	NULL	NULL	NULL	NULL

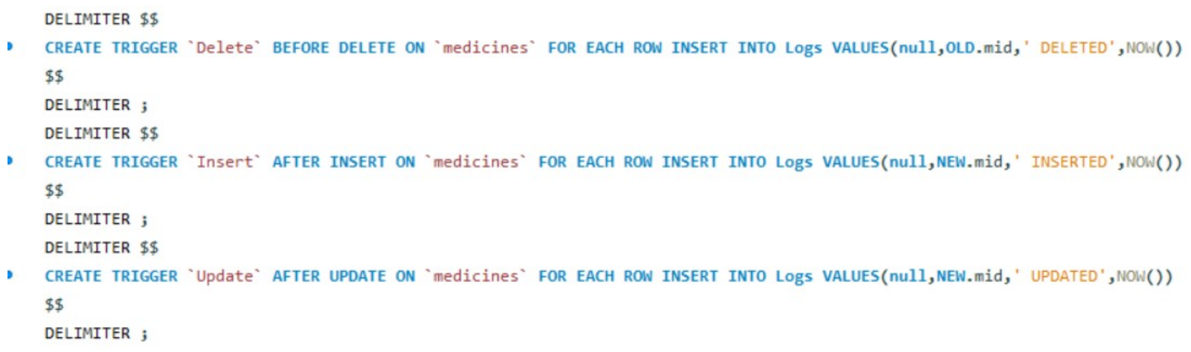
Order Details



```
SELECT * FROM logs;
select * from medicines;
```

	id	amount	name	medicines	products	email	mid
▶	3	2000	Abhishek	dolo Paraca...	suncream	a54544@gmail.com	256
	4	50000	Abhishek Pandey	A12 B15	CC	abhishekpandeyy116@gmail.com	241
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

TRIGGERS



```
DELIMITER $$
CREATE TRIGGER `Delete` BEFORE DELETE ON `medicines` FOR EACH ROW INSERT INTO Logs VALUES(null,OLD.mid,' DELETED',NOW())
$$
DELIMITER ;
DELIMITER $$
CREATE TRIGGER `Insert` AFTER INSERT ON `medicines` FOR EACH ROW INSERT INTO Logs VALUES(null,NEW.mid,' INSERTED',NOW())
$$
DELIMITER ;
DELIMITER $$
CREATE TRIGGER `Update` AFTER UPDATE ON `medicines` FOR EACH ROW INSERT INTO Logs VALUES(null,NEW.mid,' UPDATED',NOW())
$$
DELIMITER ;
```

CONCLUSION

PHARMACY MANAGEMENT SYSTEM successfully implemented offline medicines supply management database which helps us in administrating the data user for managing the tasks performed in medicines supply. The project successfully used various functionalities of SQL Workbench and python flask and also create the fully functional database management system for offline pharmacy. Using MySQL as the database is highly beneficial as it is free to download, popular and can be easily customized. The data stored in the MySQL database can easily be retrieved and manipulated according to the requirements with basic knowledge of SQL. With the theoretical inclination of our syllabus, it becomes very essential to take the at most advantage of any opportunity of gaining practical experience that comes along. The building blocks of this Major Project "Pharmacy Supply Management System" was one of these opportunities. It gave us the requisite practical knowledge to supplement the already taught theoretical concepts thus making us more competent as a computer engineer. The project from a personal point of view also helped us in understanding the following aspects of project development:

- The planning that goes into implementing a project.
- The importance of proper planning and an organized methodology
- The key element of team spirit and co-ordination in a successful project

FUTURE ENHANCEMENT

- Enhanced database storage facility
- Enhanced user-friendly GUI
- More advanced transportation of medicines
- Online Bill payments

REFERENCES

- https://flask.palletsprojects.com/en/3.0.x/
- https://flask-sqlalchemy.palletsprojects.com/en/2.x/models/
- http://www.getbootstrap.com/