



Inspiring Excellence

**CSE370 : Database Systems  
Project Report  
Project Title : DrugWeb**

| <b>Group No : 13, CSE370 Lab Section : 19, FALL 2025</b> |                     |   |
|--|---------------------|---|
| <b>ID</b>  | <b>Name</b>         | <b>Contribution</b>   |
| 22201512   | SWAGATA DEY         | Customer dashboard, Search, add to cart, review.  |
| 22301298   | SAJID MUHAMMOD      | Admin dashboard, assign deliveryman, customer's requested medicine, handle request.             |
| 21301269   | MUSHFIRAT CHOWDHURY | Deliveryman dashboard, Deliverymen confirm payment, customer payment procedure and earn points. |

## Table of Contents

| Section No | Content                | Page No |
|------------|------------------------|---------|
| 1          | Introduction           | 3       |
| 2          | Project Features       | 3       |
| 3          | ER/EER Diagram         | 4       |
| 4          | Schema Diagram         | 5       |
| 5          | Normalization          | 5       |
| 6          | Frontend Development   | 6       |
| 7          | Backend Development    | 9       |
| 8          | Source Code Repository | 14      |
| 9          | Conclusion             | 15      |
| 10         | References             | 15      |

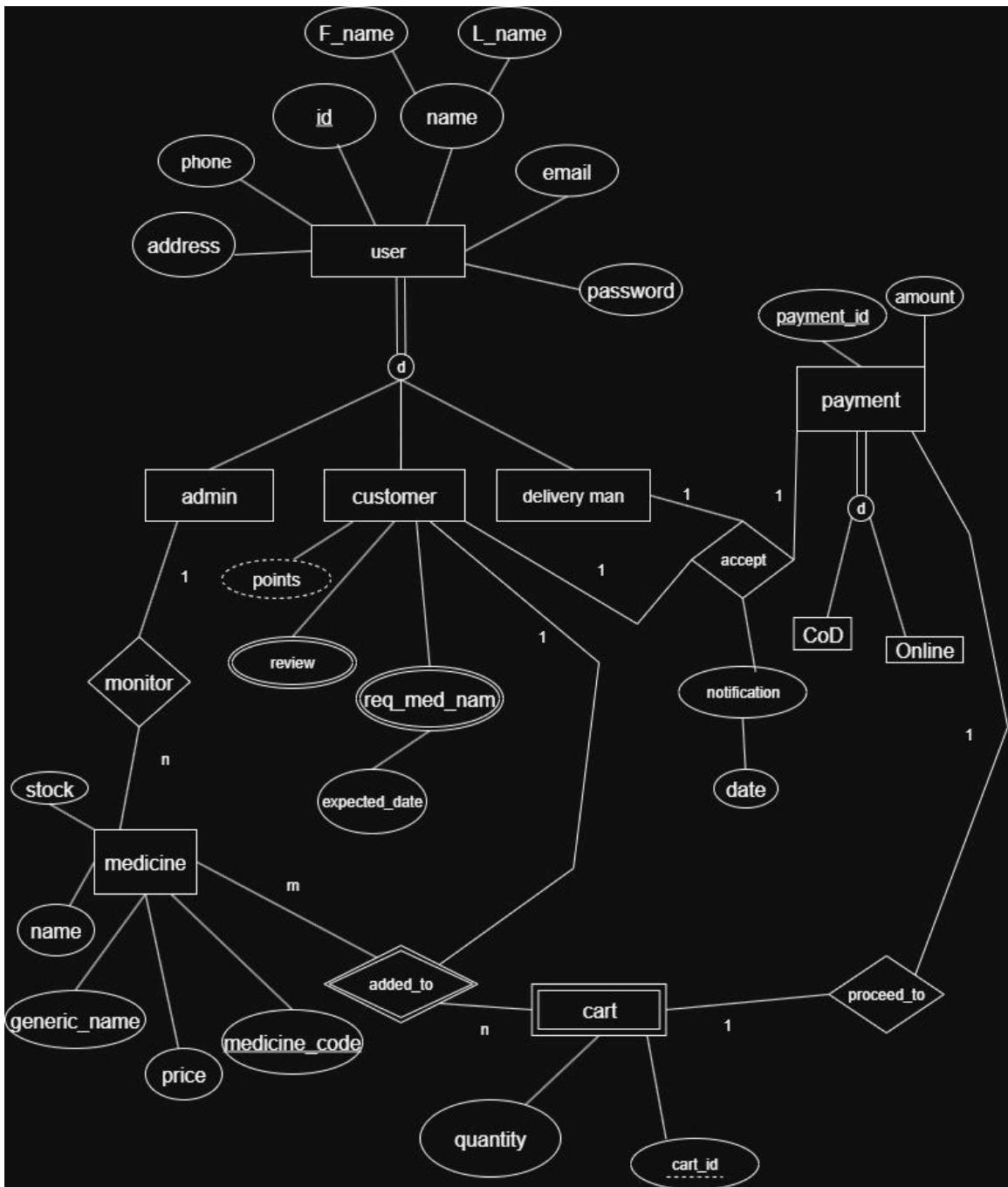
## Introduction

“DrugWeb” is an online pharmacy where a customer can buy medicines according to their needs. Our website has three types of users such as customer, admin, deliveryman. Firstly, a customer can easily search and buy medicines which are in our stock. But in case a medicine which is not in our stock, then customers can request the name of the medicines with the expected date. Then our admin can accept or reject the request according to the availability of that medicine. Again, customers can give reviews and can see other customers reviews of our previous services. Customers can add to cart any medicines and after a successful payment they can earn points. Secondly, admin can see all reviews, monitor the stock of the medicines, handle the requested medicines and payments by assigning a deliveryman. Thirdly, a deliveryman can see the assigned order and can choose a date for the delivery. After confirmation by the deliveryman, a notification will notify the customers with the delivery date.

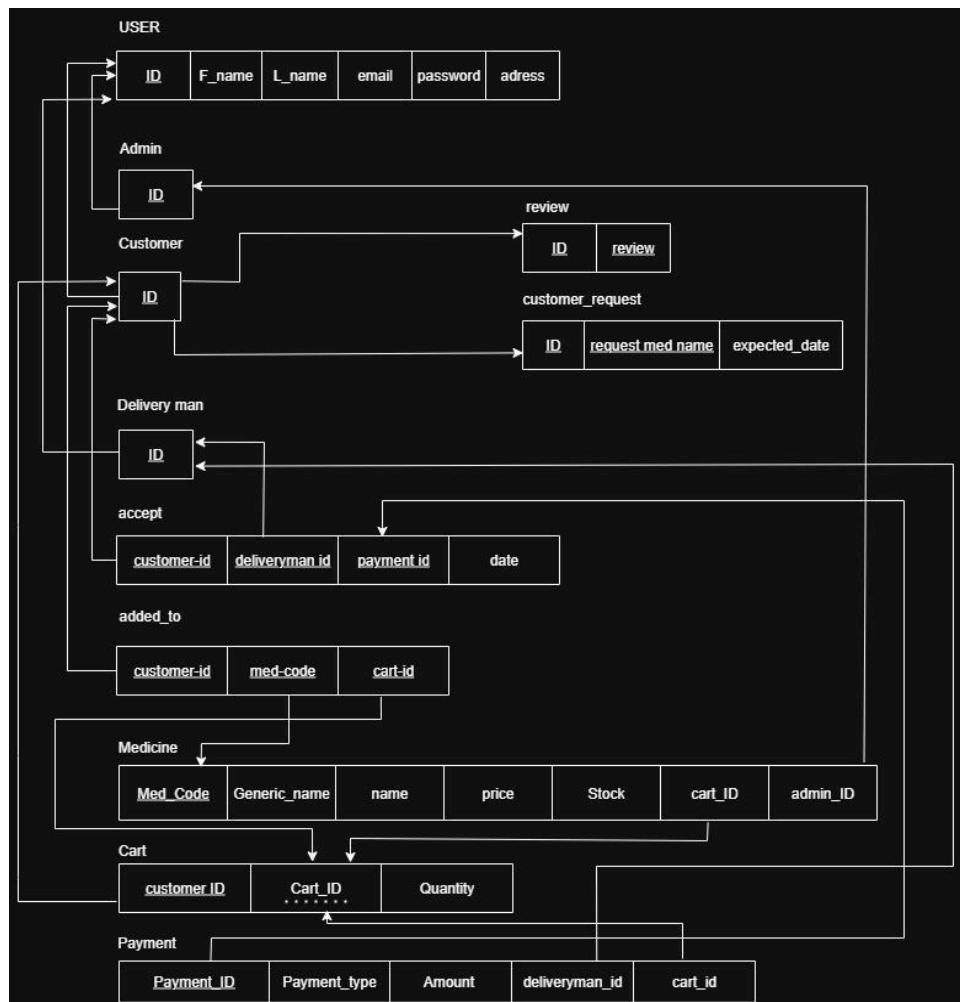
## Project Features

| <b>ID,<br/>Name</b>                        | <b>Features [3 per member]</b> |   |
|--|--------------------------------|---|
| 22201512<br><b>SWAGATA<br/>DEY</b>         | Ft 1                           | Customer dashboard and notification                       |
|  | Ft 2                           | Search by different order                                 |
|  | Ft 3                           | Add to cart and review                                    |
| 22301298<br><b>SAJID<br/>MUHAMMOD</b>      | Ft 1                           | Admin dashboard   |
|  | Ft 2                           | Assign deliveryman  |
|  | Ft 3                           | Customer's requested medicine and handle request by admin |
| 21301269<br><b>MUSHFIRAT<br/>CHOWDHURY</b> | Ft 1                           | Deliveryman dashboard                                     |
|  | Ft 2                           | Deliveryman's confirm payment                             |
|  | Ft 3                           | Customer payment procedure and earn points                |

## ER/EER Diagram



## Schema Diagram



## Normalization

- Explain if your converted Schema is in 1NF or not. If not, decompose it to 1NF.
- Explain if your converted Schema is in 2NF or not. If not, decompose it to 2NF. Can there be any partial functional dependencies in your relational schema?
- Explain if your converted Schema is in 3NF or not. If not, decompose it to 3NF. Can there be any transitive dependencies in your relational schema?

- Our schema diagram is already in 3NF. So, we didn't have to convert our schema into 3NF.

## Frontend Development

In our frontend, we used HTML and CSS to show the data and basic templates.

### Contribution of ID : 22201512, NAME: SWAGATA DEY

- Customer homepage design where a customer can find reviews, request medicine, profile, cart, notification section along with the search option.
- Login and signup page where a user has to login according to his role.

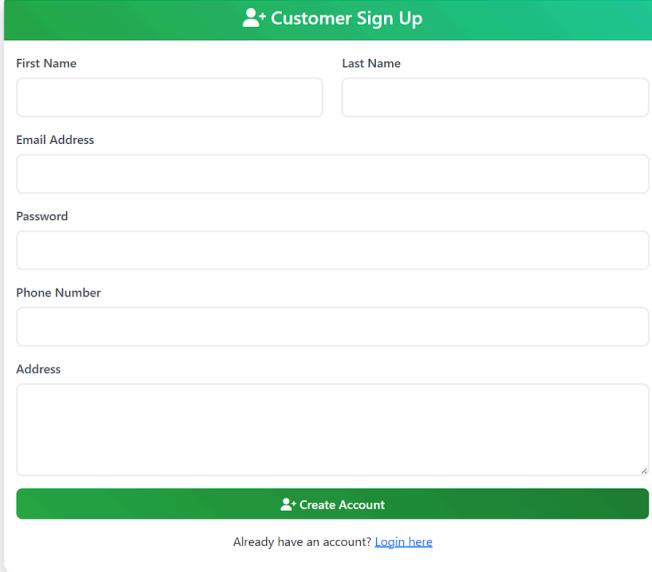
The screenshot displays two main sections of the DrugWeb application:

**Customer Homepage (Top Section):**

- Header:** DrugWeb logo, Dashboard, Profile, Cart, Notifications, Reviews, Request Medicine, Logout.
- Welcome Message:** Welcome, 115 Points, Delivery Updates, Customer Dashboard.
- Search Bar:** Search Medicines, Search by name or generic name..., Price: Low to High, Search button, Show All button.
- Popular Medicines Grid:** Shows 9 items found, sorted by Price (Low to High).
  - Napa 500 mg Tablet (10 pcs)**: Generic: Paracetamol, Price: \$12.00, In Stock, Details, Add to Cart.
  - Calpol 500 mg Tablet (10 pcs)**: Generic: Paracetamol, Price: \$12.50, In Stock, Details, Add to Cart.
  - Ace 500 mg Tablet (10 pcs)**: Generic: Paracetamol, Price: \$13.00, In Stock, Details, Add to Cart.
  - Tussin Syrup (100 ml)**: Generic: Dextromethorphan + Guaifenesin, Price: \$35.00, In Stock, Details, Add to Cart.
  - Tusca Syrup (100 ml)**: Generic: Dextromethorphan + Guaifenesin, Price: \$38.00, In Stock, Details, Add to Cart.
  - T-Day 10 mg Tablet (10 pcs)**: Generic: Cetirizine, Price: \$38.00, In Stock, Details, Add to Cart.

**Login/Signup Form (Bottom Section):**

- Header:** DrugWeb, Login, Sign Up.
- Login Form:**
  - Email Address: Input field.
  - Password: Input field.
  - Login As: Select User Type dropdown menu.
    - Select User Type
    - Customer (selected)
    - Admin
    - Delivery Man
- Footer:** © 2025 DrugWeb. All rights reserved.



The image shows a 'Customer Sign Up' form with a green header and footer. The form fields include: First Name, Last Name, Email Address, Password, Phone Number, and Address. Below the address field is a 'Create Account' button. At the bottom, a link says 'Already have an account? [Login here](#)'.

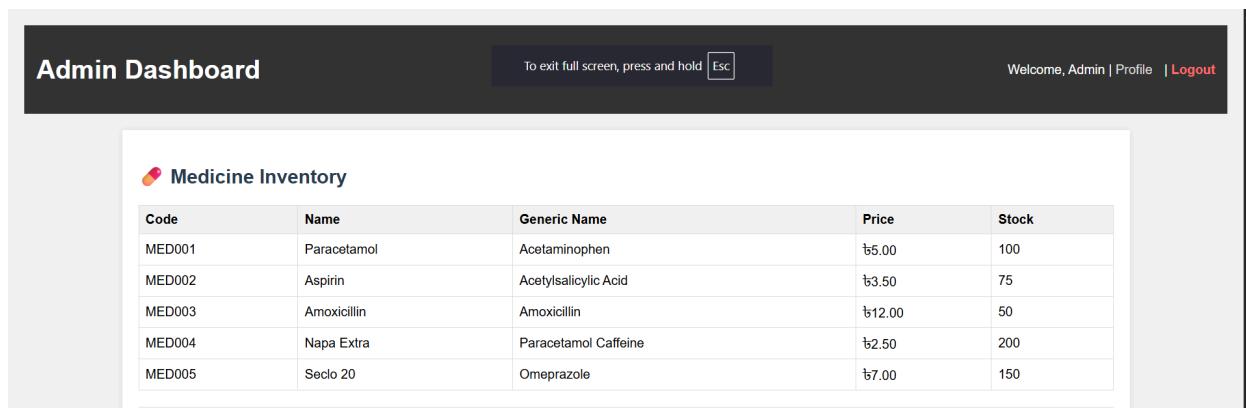
**Contribution of ID : 22301298, NAME: SAJID MUHAMMAD**

## 1. Admin Interface

- Central Dashboard:** Provides real-time tables for monitoring Medicine Stock, Customer Reviews, and Pending Medicine Requests.
- Quick Actions:** "Accept" or "Decline" buttons for medicine requests directly on the dashboard to handle customer needs instantly.
- Order Management:** A specialized "Payments" page to view all orders and a dropdown interface to Assign Deliverymen to specific orders.

## 2. Customer Interface

- Request System:** A dedicated form to request unavailable medicines, complete with a history table that tracks the status (Pending/Accepted) of each request.



The image shows an 'Admin Dashboard' with a dark header bar. It includes a 'Medicine Inventory' section with a table showing medicine details like Code, Name, Generic Name, Price, and Stock. The header also has a 'Welcome, Admin | Profile | Logout' link.

| Code   | Name        | Generic Name         | Price  | Stock |
|--------|-------------|----------------------|--------|-------|
| MED001 | Paracetamol | Acetaminophen        | ₹5.00  | 100   |
| MED002 | Aspirin     | Acetylsalicylic Acid | ₹3.50  | 75    |
| MED003 | Amoxicillin | Amoxicillin          | ₹12.00 | 50    |
| MED004 | Napa Extra  | Paracetamol Caffeine | ₹2.50  | 200   |
| MED005 | Seclo 20    | Omeprazole           | ₹7.00  | 150   |

### medicine Requests

| Customer Name      | Medicine Requested | Expected Date | Status   | Action   |
|--------------------|--------------------|---------------|----------|--|
| Musfirat choudhury | Omeprazole         | 2026-01-23    | Pending  | <button>Accept</button> <button>Decline</button> |
| Musfirat choudhury | Adovas             | 2026-01-30    | Accepted | Accepted   |
| Hello World        | Don A 10 mg        | 2026-01-30    | Accepted | Accepted   |
| Musfirat choudhury | Motrin             | 2026-01-10    | Pending  | <button>Accept</button> <button>Decline</button> |
| Hello World        | Norium 10 mg       | 2026-01-31    | Pending  | <button>Accept</button> <button>Decline</button> |
| Musfirat choudhury | Rolac 10 mg        | 2026-01-07    | Pending  | <button>Accept</button> <button>Decline</button> |
| Hello World        | Rolac 10 mg        | 2026-01-15    | Pending  | <button>Accept</button> <button>Decline</button> |
| Hello World        | Tryplin 10 mg      | 2026-01-06    | Declined | Declined   |

### Customer Reviews

| Customer    | Review                                       |
|-------------|--|
| Hello World | After Taking Aspirin my pain didn't decrease |

[View All Payments](#)

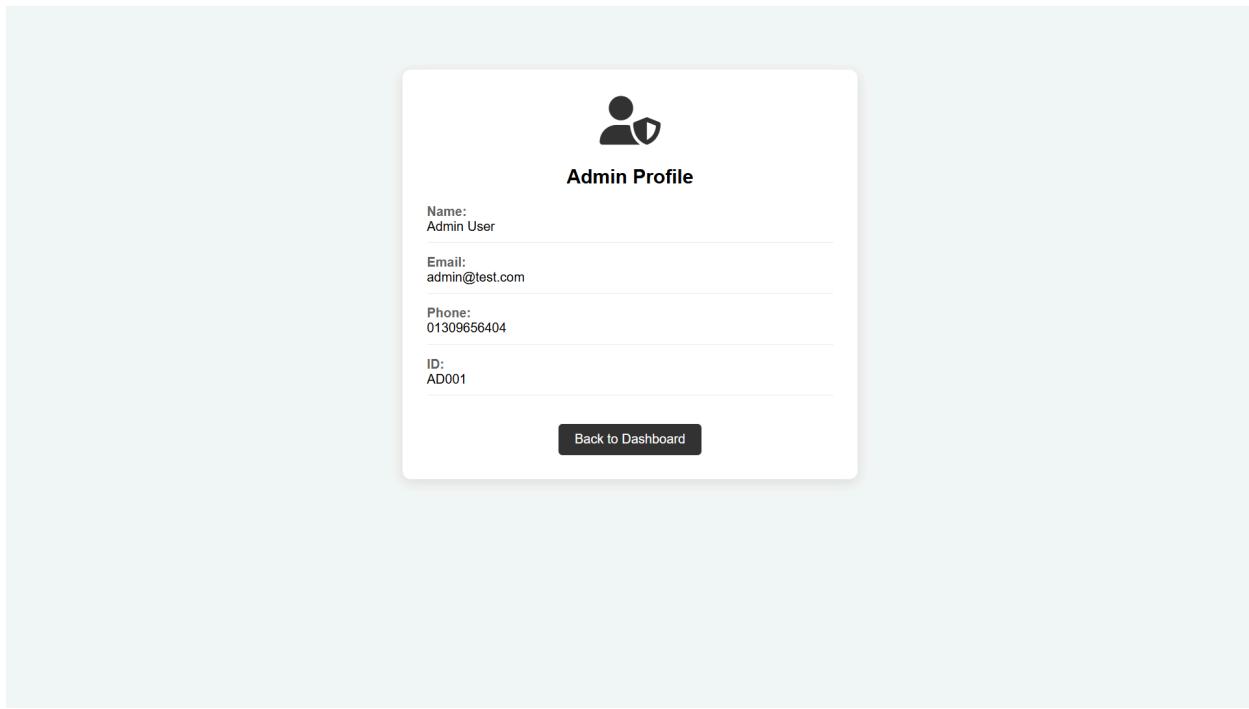
DrugWeb Admin Dashboard Logout

### Customer Orders & Payments

[← Back to Dashboard](#)

| Order ID   | Customer                          | Amount | Payment          | Address   | Delivery Status                 | Action   |
|------------|-----------------------------------|--------|------------------|-----------|---------------------------------|--|
| #PAY954658 | Hello World<br>01309656404        | ₹42.00 | Cash on Delivery | Mirpur 12 | Pikachu Delivery<br>01309656404 | Select Delivery Man... <button>Assign</button> |
| #PAY475781 | Musfirat choudhury<br>01309656404 | ₹27.00 | Bkash            | Mirpur 13 | Pikachu Delivery<br>01309656404 | Select Delivery Man... <button>Assign</button> |
| #PAY372201 | Hello World<br>01309656404        | ₹30.00 | Cash on Delivery | Mirpur 12 | Pikachu Delivery<br>01309656404 | Select Delivery Man... <button>Assign</button> |
| #PAY277301 | Hello World<br>01309656404        | ₹14.50 | Bkash            | Mirpur 12 | Unassigned                      | Select Delivery Man... <button>Assign</button> |
| #1001      | Hello World<br>01309656404        | ₹50.00 | Cash on Delivery | Mirpur 12 | Pikachu Delivery<br>01309656404 | Select Delivery Man... <button>Assign</button> |



**Request a Medicine**

Can't find a medicine? Request it here and we'll try to stock it for you.

Medicine Name: E.g., Napa Extend, Seclo 20

Expected Date Needed: mm/dd/yyyy

**Submit Request**

**Your Request History**

| Medicine Name | Expected Date | Status   |
|---------------|---------------|----------|
| Tryptin 10 mg | 2026-01-06    | Declined |
| Rolac 10 mg   | 2026-01-15    | Pending  |
| Norium 10 mg  | 2026-01-31    | Pending  |
| Don A 10 mg   | 2026-01-30    | Accepted |

### Contribution of ID : 21301269, Name : Mushfirat Chowdhury

- Deliveryman dashboard where he can find his profile option and can see the assigned orders.
- Deliveryman's confirmation of an order with a date.

**DrugWeb**

Delivery man login successful!

## Delivery Dashboard

**Assigned Orders** 11 **Available Orders** 1 **Total Value** ₹3537.50 **Accepted Orders** 10

**Delivery Assignments** 11 Orders

| #   | Customer Name | Status                |
|-----|---------------|-----------------------|
| #37 | Adrita Tanaz  | Assigned              |
| #34 | Shadman Ahmed | Accepted for Delivery |

**Accept Delivery** **Decline**

**DrugWeb**

Delivery man login successful!

## Delivery Dashboard

**Assigned Orders** 11 **Available Orders** 1 **Accepted Orders** 10

**Delivery Assignments** 11 Orders

| #   | Customer Name | Status                |
|-----|---------------|-----------------------|
| #37 | Adrita Tanaz  | Assigned              |
| #34 | Shadman Ahmed | Accepted for Delivery |

**Schedule Delivery**

Select Delivery Date

dd----yyyy

September 2025

**Accept & Schedule**

## Backend Development

For the backend, we used the FLASK framework of python and MYSQL for handling queries and creating tables in the database.

### Contribution of ID : 22201512 , NAME: SWAGATA DEY

- Database creation with tables and assigning primary keys and foreign keys.

- Customer dashboard where a customer can see reviews and can give reviews, get notifications.
- Search any medicine and can sort by price or name.
- Add to cart and in the cart increase or decrease quantity of the medicines.

The screenshot shows a code editor with two tabs open, both titled "app.py".

**Top Tab:**

```

703     @app.route('/customer_dashboard')
704     def customer_dashboard():
705         if 'user_id' not in session or session['user_type'] != 'customer':
706             flash('Please login as customer first!', 'error')
707             return redirect(url_for('login'))
708
709         # Get search and sort parameters (removed category)
710         search = request.args.get('search', '')
711         sort_by = request.args.get('sort_by', 'name')
712         show_all = request.args.get('show_all', '0')
713
714         connection = get_db_connection()
715         medicines = []
716         customer_points = 0
717
718         if connection:
719             cursor = connection.cursor(dictionary=True)
720
721             # Get customer's current points
722             try:
723                 cursor.execute("SELECT points FROM customer WHERE Customer_ID = %s", (session['user_id'],))
724                 points_result = cursor.fetchone()
725                 customer_points = points_result['points'] if points_result else 0
726             except Exception as e:
727                 print(f"Error fetching customer points: {e}")
728                 customer_points = 0
729
730             # Build query based on search and sort (no category)
731             base_query = "SELECT * FROM medicine WHERE 1=1"
732             params = []
733
734             # Add search condition
735             if search:
736                 base_query += " AND (Name LIKE %s OR Generic_name LIKE %s)"
737                 params.extend([f'%{search}%', f'%{search}%'])

```

**Bottom Tab:**

```

841     @app.route('/browse_medicines')
842     def browse_medicines():
843         if 'user_id' not in session or session['user_type'] != 'customer':
844             return redirect(url_for('login'))
845
846         search = request.args.get('search', '').strip()
847         sort_by = request.args.get('sort_by', 'name')
848         category = request.args.get('category', '')
849         page = int(request.args.get('page', 1))
850         per_page = 12 # Show 12 medicines per page
851
852         medicines = []
853         total_count = 0
854
855         connection = get_db_connection()
856         if connection:
857             cursor = connection.cursor(dictionary=True)
858
859             # Build the base query
860             base_query = "SELECT * FROM medicine WHERE 1=1"
861             count_query = "SELECT COUNT(*) as total FROM medicine WHERE 1=1"
862             params = []
863
864             # Add search condition
865             if search:
866                 base_query += " AND (Name LIKE %s OR Generic_name LIKE %s OR Category LIKE %s)"
867                 count_query += " AND (Name LIKE %s OR Generic_name LIKE %s OR Category LIKE %s)"
868                 params.extend([f'%{search}%', f'%{search}%', f'%{search}%'])
869
870             # Add category filter
871             if category:
872                 base_query += " AND Category = %s"
873                 count_query += " AND Category = %s"
874                 params.append(category)
875
876             # Add ordering
877             #<--> cursor.execute(base_query, params)

```

```

1202 @app.route('/reviews', methods=['GET', 'POST'])
1203 def reviews():
1204     if 'user_id' not in session or session['user_type'] != 'customer':
1205         flash('Please login as customer first!', 'error')
1206         return redirect(url_for('login'))
1207
1208     connection = get_db_connection()
1209
1210     if request.method == 'POST':
1211         review_text = request.form['review']
1212         customer_id = session['user_id']
1213
1214         if connection:
1215             cursor = connection.cursor()
1216             try:
1217                 cursor.execute("""
1218                     INSERT INTO customer_review (Customer_ID, review)
1219                     VALUES (%s, %s)
1220                     %s, (customer_id, review_text))
1221                     connection.commit()
1222                     flash('Your review has been submitted successfully!', 'success')
1223             except Error as e:
1224                 connection.rollback()
1225                 flash(f'Error submitting review: {e}', 'error')
1226             finally:
1227                 cursor.close()
1228
1229     # Fetch all reviews with customer names
1230     reviews_list = []
1231     if connection:
1232         cursor = connection.cursor(dictionary=True)
1233         cursor.execute("""
1234             SELECT cr.review, u.F_name, u.I_name, cr.Customer_ID
1235             FROM customer_review cr
1236             JOIN user u ON cr.Customer_ID = u.ID
1237             ORDER BY cr.Customer_ID DESC
1238
1239

```

```

1601 @app.route('/add_to_cart', methods=['POST'])
1602 def add_to_cart():
1603     print(f"Session data: {dict(session)}") # Debug
1604
1605     if 'user_id' not in session or session['user_type'] != 'customer':
1606         return jsonify({'success': False, 'message': 'Please login as customer first'})
1607
1608     try:
1609         data = request.json
1610         print(f"Received data: {data}") # Debug
1611
1612         med_code = data.get('med_code')
1613         med_name = data.get('med_name')
1614         quantity = int(data.get('quantity', 1))
1615         price = float(data.get('price', 0))
1616         customer_id = session['user_id']
1617
1618         print(f"Parsed - Code: {med_code}, Name: {med_name}, Price: {price}, Qty: {quantity}, Customer: {customer_id}") # Debug
1619
1620         if not med_code or quantity <= 0:
1621             return jsonify({'success': False, 'message': 'Invalid input data'})
1622
1623         connection = get_db_connection()
1624         if not connection:
1625             return jsonify({'success': False, 'message': 'Database connection failed'})
1626
1627         cursor = connection.cursor(dictionary=True)
1628
1629         # Check if medicine exists and has enough stock
1630         cursor.execute("SELECT Stock FROM medicine WHERE Med_code = %s", (med_code,))
1631         medicine = cursor.fetchone()
1632
1633         if not medicine:
1634             cursor.close()
1635             connection.close()
1636             return jsonify({'success': False, 'message': 'Medicine not found'})
1637

```

## Contribution of ID : 22301298, NAME: SAJID MUHAMMAD

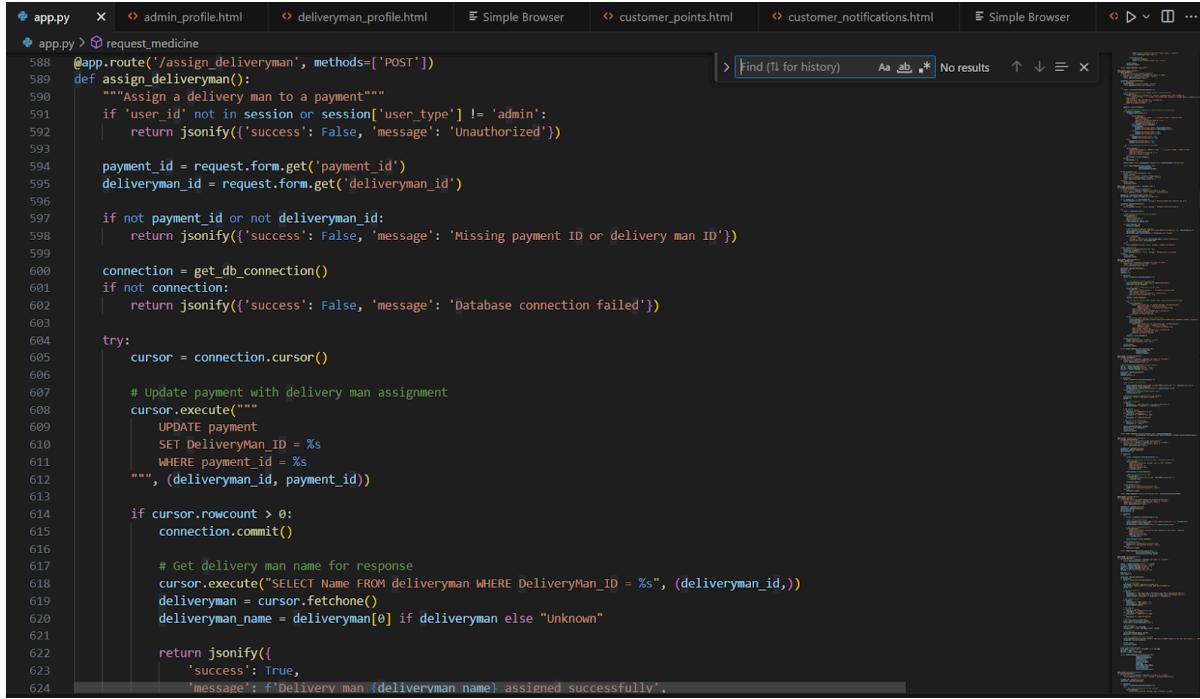
- Admin Dashboard Logic:** Wrote the code to fetch and display the latest medicine stock, reviews, and requests on the admin dashboard.
- Request Handling:** Implemented the functionality that allows the admin to update a request's status to "Accepted" or "Declined" in the database.
- Delivery Assignment:** Created the backend logic that saves the assigned deliveryman to a specific customer order.
- Customer Request Logic:** Wrote the code to save the medicine name and expected date when a customer submits a new request.

```
❶ app.py  x  admin_profile.html  deliveryman_profile.html  Simple Browser  customer_points.html  customer_notifications.html  Simple Browser  ⌂ ⌄ ⌁ ...
```

```
❷ app.py > ⚭ request_medicine
638     @app.route('/admin_dashboard')
639     def admin_dashboard():
640         if 'user_id' not in session or session['user_type'] != 'admin':
641             flash('Please login as admin first!', 'error')
642             return redirect(url_for('login'))
643
644         connection = get_db_connection()
645         medicines = []
646         reviews = []
647         requests = []
648
649         if connection:
650             cursor = connection.cursor(dictionary=True)
651
652             try:
653                 # 1. Get medicines ordered by Med_Code
654                 cursor.execute("SELECT * FROM medicine ORDER BY Med_Code")
655                 medicines = cursor.fetchall()
656
657                 # 2. Get customer reviews with customer names
658                 cursor.execute("""
659                     SELECT cr., CONCAT(u.F_name, ' ', u.L_name) as customer_name
660                     FROM customer_review cr
661                     JOIN customer c ON cr.Customer_ID = c.Customer_ID
662                     JOIN user u ON c.Customer_ID = u.ID
663                 """)
664                 reviews = cursor.fetchall()
665
666                 # 3. Get medicine requests with customer names (handle missing Status column)
667                 try:
668                     cursor.execute("""
669                         SELECT cmr.Customer_ID, cmr.request_med_name, cmr.Expected_date,
670                             IFNULL(cmr.Status, 'Pending') as Status,
671                             CONCAT(u.F_name, ' ', u.L_name) as customer_name
672                         FROM customer_request cmr
673                         JOIN customer c ON cmr.Customer_ID = c.Customer_ID
674                         JOIN user u ON c.Customer_ID = u.ID
675                     """)
676                 except Exception as e:
677                     flash(f'Error submitting request: {e}', 'error')
678             finally:
679                 cursor.close()
680
681             # Fetch customer's previous requests with status
682             requests_list = []
683             if connection:
684                 cursor = connection.cursor(dictionary=True)
685
686                 # First ensure the Status column exists
687                 try:
688                     cursor.execute("ALTER TABLE customer_request ADD COLUMN IF NOT EXISTS Status VARCHAR(20) DEFAULT 'Pending'!")
689                 except Exception as e:
690                     flash(f'Error creating Status column: {e}', 'error')
691
692             return render_template('admin_dashboard.html', medicines=medicines, reviews=reviews, requests=requests_list)
```

```
❶ app.py  x  admin_profile.html  deliveryman_profile.html  Simple Browser  customer_points.html  customer_notifications.html  Simple Browser  ⌂ ⌄ ⌁ ...
```

```
❷ app.py > ⚭ request_medicine
1244     @app.route('/request_medicine', methods=['GET', 'POST'])
1245     def request_medicine():
1246         if 'user_id' not in session or session['user_type'] != 'customer':
1247             flash('Please login as customer first!', 'error')
1248             return redirect(url_for('login'))
1249
1250         connection = get_db_connection()
1251
1252         if request.method == 'POST':
1253             medicine_name = request.form['medicine_name']
1254             expected_date = request.form['expected_date']
1255             customer_id = session['user_id']
1256
1257             if connection:
1258                 cursor = connection.cursor()
1259                 try:
1260                     cursor.execute("""
1261                         INSERT INTO customer_request (Customer_ID, request_med_name, Expected_date)
1262                             VALUES (%s, %s, %s)
1263                         %s, (%s, %s, %s)
1264                     """, (customer_id, medicine_name, expected_date))
1265                     connection.commit()
1266                     flash('Your medicine request has been submitted successfully!', 'success')
1267                 except Error as e:
1268                     connection.rollback()
1269                     flash(f'Error submitting request: {e}', 'error')
1270             finally:
1271                 cursor.close()
1272
1273             # Fetch customer's previous requests with status
1274             requests_list = []
1275             if connection:
1276                 cursor = connection.cursor(dictionary=True)
1277
1278                 # First ensure the Status column exists
1279                 try:
1280                     cursor.execute("ALTER TABLE customer_request ADD COLUMN IF NOT EXISTS Status VARCHAR(20) DEFAULT 'Pending'!")
1281                 except Error as e:
1282                     flash(f'Error creating Status column: {e}', 'error')
```



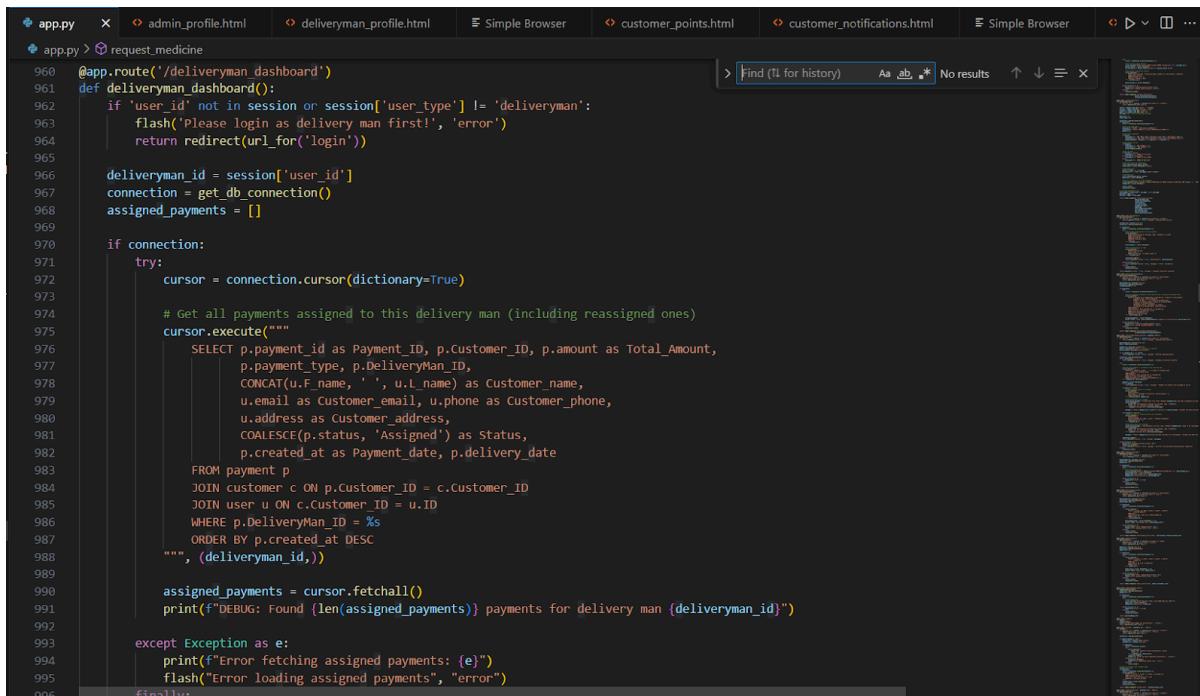
```

❶ app.py x admin_profile.html deliveryman_profile.html Simple Browser customer_points.html customer_notifications.html Simple Browser D v ...
❷ app.py > request_medicine
588 @app.route('/assign_deliveryman', methods=['POST'])
589 def assign_deliveryman():
590     """Assign a delivery man to a payment"""
591     if 'user_id' not in session or session['user_type'] != 'admin':
592         return jsonify({'success': False, 'message': 'Unauthorized'})
593
594     payment_id = request.form.get('payment_id')
595     deliveryman_id = request.form.get('deliveryman_id')
596
597     if not payment_id or not deliveryman_id:
598         return jsonify({'success': False, 'message': 'Missing payment ID or delivery man ID'})
599
600     connection = get_db_connection()
601     if not connection:
602         return jsonify({'success': False, 'message': 'Database connection failed'})
603
604     try:
605         cursor = connection.cursor()
606
607         # Update payment with delivery man assignment
608         cursor.execute("""
609             UPDATE payment
610             SET DeliveryMan_ID = %s
611             WHERE payment_id = %s
612         """, (deliveryman_id, payment_id))
613
614         if cursor.rowcount > 0:
615             connection.commit()
616
617         # Get delivery man name for response
618         cursor.execute("SELECT Name FROM deliveryman WHERE DeliveryMan_ID = %s", (deliveryman_id,))
619         deliveryman = cursor.fetchone()
620         deliveryman_name = deliveryman[0] if deliveryman else "Unknown"
621
622         return jsonify({
623             'success': True,
624             'message': f'Delivery man {deliveryman_name} assigned successfully.'})
625     finally:
626         cursor.close()
627

```

## Contribution of ID : 21301269, Name : Mushfirat Chowdhury

- Deliveryman dashboard with accept or decline payment option.
- Customer payment procedure.
- Customer earn point feature according to customer's total payment.



```

❶ app.py x admin_profile.html deliveryman_profile.html Simple Browser customer_points.html customer_notifications.html Simple Browser D v ...
❷ app.py > request_medicine
960 @app.route('/deliveryman_dashboard')
961 def deliveryman_dashboard():
962     if 'user_id' not in session or session['user_type'] != 'deliveryman':
963         flash('Please login as delivery man first!', 'error')
964         return redirect(url_for('login'))
965
966     deliveryman_id = session['user_id']
967     connection = get_db_connection()
968     assigned_payments = []
969
970     if connection:
971         try:
972             cursor = connection.cursor(dictionary=True)
973
974             # Get all payments assigned to this delivery man (including reassigned ones)
975             cursor.execute("""
976                 SELECT p.payment_id as Payment_ID, p.Customer_ID, p.amount as Total_Amount,
977                 p.payment_type, p.DeliveryMan_ID,
978                 CONCAT(u.F_name, ' ', u.L_name) as customer_name,
979                 u.email as Customer_email, u.phone as Customer_phone,
980                 u.address as Customer_address,
981                 COALESCE(p.status, 'Assigned') as Status,
982                 p.created_at as Payment_date, p.delivery_date
983                 FROM payment p
984                 JOIN customer c ON p.customer_ID = c.customer_ID
985                 JOIN user u ON c.Customer_ID = u.ID
986                 WHERE p.DeliveryMan_ID = %s
987                 ORDER BY p.created_at DESC
988             """, (deliveryman_id,))
989
990             assigned_payments = cursor.fetchall()
991             print(f'DEBUG: Found {len(assigned_payments)} payments for delivery man {deliveryman_id}')
992
993         except Exception as e:
994             print(f'Error fetching assigned payments: {e}')
995             flash("Error loading assigned payments", "error")
996         finally:

```

```
app.py | admin_profile.html | deliveryman_profile.html | Simple Browser | customer_points.html | customer_notifications.html | Simple Browser | D v ... | Find (l for history) Aa ab * No results ↑ ↓ = x |  
app.py > request_medicine  
2532     def process_payment():  
2533         """Process payment and save to database"""  
2534         if 'user_id' not in session:  
2535             return redirect(url_for('login'))  
2536  
2537         payment_type = request.form.get('payment_method')  
2538  
2539         if not payment_type:  
2540             flash("Please select a payment method", "error")  
2541             return redirect(url_for('payment_page'))  
2542  
2543         connection = get_db_connection()  
2544         if not connection:  
2545             flash("Database connection failed", "error")  
2546             return redirect(url_for('payment_page'))  
2547  
2548     try:  
2549         cursor = connection.cursor()  
2550  
2551         print(f"DEBUG: Processing payment for user {session['user_id']}")  
2552         print(f"DEBUG: Payment method selected: {payment_type}")  
2553  
2554         # Calculate total from cart using same method as cart view  
2555         cursor.execute("""  
2556             SELECT SUM(c.total_price) as total  
2557             FROM cart c  
2558             WHERE c.Customer_ID = %s  
2559             """, (session['user_id'],))  
2560  
2561         result = cursor.fetchone()  
2562         total_amount = result[0] if result and result[0] else 0  
2563  
2564         print(f"DEBUG: Calculated total amount: {total_amount}")  
2565  
2566         if total_amount <= 0:  
2567             print("DEBUG: Cart is empty, redirecting to dashboard")  
2568             flash("Cart is empty", "error")  
2569  
2570     
```

```
app.py | admin_profile.html | deliveryman_profile.html | Simple Browser | customer_points.html | customer_notifications.html | Simple Browser | D v ... | Find (l for history) Aa ab * No results ↑ ↓ = x |  
app.py > request_medicine  
2532     def process_payment():  
2533  
2534         # Award points for successful purchase  
2535         # Points calculation: 1 point for every 10 BDT spent (rounded down)  
2536         points_earned = int(total_amount_decimal // 10)  
2537  
2538         if points_earned > 0:  
2539             print(f"DEBUG: Awarding {points_earned} points for purchase of {total_amount_decimal}")  
2540  
2541             # Add points to customer's account  
2542             cursor.execute("""  
2543                 UPDATE customer  
2544                 SET points = points + %s  
2545                 WHERE Customer_ID = %s  
2546             """, (points_earned, customer_id_str))  
2547  
2548             # Log the points transaction  
2549             cursor.execute("""  
2550                 INSERT INTO points_history (customer_id, points_earned, transaction_type, payment_id, description, created_at)  
2551                 VALUES (%s, %s, 'earned', %s, %s, NOW())  
2552             """, (customer_id_str, points_earned, payment_id, f"Purchase reward: {points_earned} points for {total_amount_decimal} purchase"))  
2553  
2554             print(f"DEBUG: Points awarded successfully")  
2555             # Update the flash message to include points earned  
2556             flash(f"Payment successfull Payment ID: {payment_id}. You earned {points_earned} points!", "success")  
2557         else:  
2558             flash(f"Payment successfull Payment ID: {payment_id}", "success")  
2559  
2560         # Clean cart after successful payment  
2561         print(f"DEBUG: Clearing cart for user {session['user_id']}")  
2562         cursor.execute("DELETE FROM cart WHERE Customer_ID = %s", (session['user_id'],))  
2563  
2564         connection.commit()  
2565         print(f"DEBUG: Transaction committed successfully")  
2566  
2567     return redirect(url_for('customer_dashboard'))  
2568  
2569 
```

## **Source Code Repository**

<https://github.com/portfoliosajid/cse370>

## **Conclusion**

In conclusion, our database based website DrugWeb is a place where customers can easily search for medicines, buy medicines, give their reviews and request medicine according to their needs. Additionally, in future one thing which we want to improve in our website is the point system. Now customers can only earn points but we want to upgrade the system such that customers can have the option to redeem their points while paying. Lastly, the most common issue we faced during the backend development is not giving the proper mysql queries which created issues connecting with the database.

## **References**

<https://youtu.be/eDTcSaZcLh4?si=Anr2ggT4MzWl65mx>

(this video helped to understand flask framework and how to make a proper e-commerce website)

<https://www.w3schools.com/MySQL/default.asp>

(This website helped to implement the mysql queries properly)