**INFSCI 2560 – NETWORK & WEB TECHNOLOGIES**

**FINAL PROJECT REPORT**

**ONLINE PHARMACY APPLICATION MANAGEMENT SYSTEM**

***DeliveRx – Delivering Medicines at your doorstep***

|  |  |
| --- | --- |
| **A picture containing plate, drawing  Description automatically generated**  **A picture containing refrigerator, toy  Description automatically generated** | **A close up of a toy  Description automatically generated** |
|  |  |

**University of Pittsburgh, School of Computing & Information**

**Logo

Description automatically generated**

**Submitted by:**

|  |  |  |  |
| --- | --- | --- | --- |
| **SL. No** | **Name** | **Student ID** | **Pitt Email Address** |
| 1. | David Ball |  |  |
| 2. | Piu Mallick | 4374215 | pim16@pitt.edu |
| 3. | Chuhan Xu |  |  |

Table of Contents

**No table of contents entries found.**

1. **Basic Introduction**

This project represents a **Full-Stack implementation** of a **B2C E-commerce website** (named **DeliveRx**) which handles online delivery of **Over-The-Counter** medicines / medicine-packages to localized customers. This web application is catered to meet immediate medical need(s) of the people who are in need to OTC medicines, especially in the pandemic times (**COVID-19**), when going out in public and availing things are not at all convenient. The online store contains medicines that can be categorized by various diseases/ailment types. The medicines can be browsed and searched by customers on the website and can be added to the cart to make a purchase.

1. **Users / Actors:**

The application has 2 kinds of users/actors. They are stated as follows:

* **Customer** – a person who can purchase medicines.
* **Staff** – a person who can manage the inventory details and view the order details of the customers.

The application has a login page, generic for all types of users, viz. **Customer** and **Staff**. There is a **role-based authentication process**, which means all the users/actors can log-in using the same **Login page**. However, they will be directed to a different portal after log-in, according to their access/roles.

1. **Application Features:**
2. **Generic Features**

* Any user can visit the website and explore – get to know more about the online store and search medicines.
* If the user intends to buy medicine(s), they he/she has to login to the system and continue shopping.

1. **Login / Registration**

* All the users can login to the system if they have valid **username** (**Email Address** in this case) and **Password**. That means, they need to ‘**Sign Up**’ before they can buy medicines.
* If they are not registered in the system, they need to do the registration by providing few details, like **First Name**, **Last Name**, **Email Address**, **Phone Number**, **Address**, **Gender**, **Date of Birth** and **Password**.
* Once done, they can login using the same **Email Address** and **Password** they have used for registration.
* After log-in, they can view a ‘**Welcome Page**’ with features respective to their roles.

1. **Customer Specific Features**

* A customer can visit the website, explore various medicines, and if he decides to purchase medicines, he/she can register (if new) and login to the system and add medicines to the shopping cart and make a purchase.
* Once logged in, the customer is greeted with a ‘**Welcome Message**’.
* The customer would be able to view the features available to him / her, viz:
* View / Update his/her personal information.
* Search medicines / Explore medicines by name/category.
* Add the desired medicines in the shopping cart, and then check out.
* After checkout, a message will be displayed on the screen with order and delivery details.
* The customer can ‘**Log Out**’ when desired.

1. **Staff Specific Features**

* A staff can log in using the ‘**Log In**’ portal.
* Once logged in, the staff is redirected to ‘**Staff Dashboard**’ and greeted with a ‘**Welcome Message**’.
* The staff can ‘**Log Out**’ when desired.
* He / She can navigate between the pages and return to the main dashboard, when desired.
* He / She would be able to view the features available to him / her, viz.:
* **Access to Medicine List**
  + View Medicines
  + Add Medicines
  + Edit / Update Medicine details
  + Delete Medicines
* **Access to Supplier List**
* View Suppliers
* Add Suppliers
* Edit / Update Supplier Details
* Delete Suppliers

1. **Business Rules**

Some of the business rules defined for this web application are listed as follows:

* User can be a customer or a staff.
* One customer can place many orders.
* One order is only from one customer.
* One order can contain many items.
* One category has many medicines.

1. **Technologies Used in building the application**

* **Front-End:** HTML5 / CSS3, JavaScript
* **Back-end:** NodeJS Express
* **Database:** PostgreSQL

|  |  |  |  |
| --- | --- | --- | --- |
| **A close up of a sign  Description automatically generated** | **A close up of a sign  Description automatically generated** | **Icon  Description automatically generated** | **Text  Description automatically generated** |

1. **Project Repository**

The project code repository is available at: <https://github.com/piumallick/DeliveRx>

The database is hosted on **aiven Cloud** (**Aiven for PostgreSQL**): <https://aiven.io/postgresql>

The DB scripts can also be found under the ‘**database**’ folder in the above mentioned **Github repository**.

1. **Web Application Functionalities**
2. **Accessibility 🡪** Efforts have been put in to make sure that the website is friendly to all kinds of users, independent of their physical or mental abilities.

Explanation goes in here.

1. **Responsive Design 🡪 Viewport** and **Bootstrap CDN libraries** have been used to make the website a pleasure to browse and enjoy on any kind of device.

Explanation goes in here.

1. **User Authentication Flow**
2. **Persistent Data Storage 🡪 PostgreSQL** has been used to meet the needs of persistent data storage. The details are discussed in Section 6 of the project report.
3. **REST API Implementation**
4. **Visual Design**
5. **Use of Front-End JavaScript**
6. **Error Handling**
7. **Security**
8. **Database Design**

**PostgreSQL** has been used as the database for this web application.

The following tables have been used:

* **USERS** 🡪 This table contains the user details (customer and staff details).Graphical user interface, application

  Description automatically generated
* **ROLE** 🡪 This table contains the role details (role id and name – whether a staff or a customer).

Table

Description automatically generated

* **ROLE\_MAPPING** 🡪 This table maps the user to its corresponding role.

Table

Description automatically generated

* **MEDICINE \_CATEGORY\_ PRICE** 🡪 This table contains the medicine details – name, category, price, etc.

Graphical user interface, application

Description automatically generated

* **ORDERS** 🡪 This table contains the order details placed by the customer.

**Graphical user interface, application

Description automatically generated**

1. **Application Flow & Screenshots**
2. **Home Page**

**A picture containing graphical user interface

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**

1. **Login Page**

**Graphical user interface, application

Description automatically generated**

1. **Register New User Page**

**Graphical user interface, application

Description automatically generated**

1. **Testing Efforts**

* Unit Testing and Peer testing has been done manually for each and every functionality.

1. **Assumptions**
2. **Future Improvements**
3. **References**

|  |  |  |
| --- | --- | --- |
| SL. No. | Functionality | Reference URL |
|  | Home Page (CSS Template) | [**https://github.com/john-smilga/js-cart-setup/tree/master/js**](https://github.com/john-smilga/js-cart-setup/tree/master/js) |
|  | Login & Registration Functionality | * [**https://medium.com/@timtamimi/getting-started-with-authentication-in-node-js-with-passport-and-postgresql-2219664b568c**](https://medium.com/@timtamimi/getting-started-with-authentication-in-node-js-with-passport-and-postgresql-2219664b568c) * [**https://reallifeprogramming.com/node-authentication-with-passport-postgres-ef93e2d520e7**](https://reallifeprogramming.com/node-authentication-with-passport-postgres-ef93e2d520e7) * [**https://www.youtube.com/watch?v=vxu1RrR0vbw**](https://www.youtube.com/watch?v=vxu1RrR0vbw) |