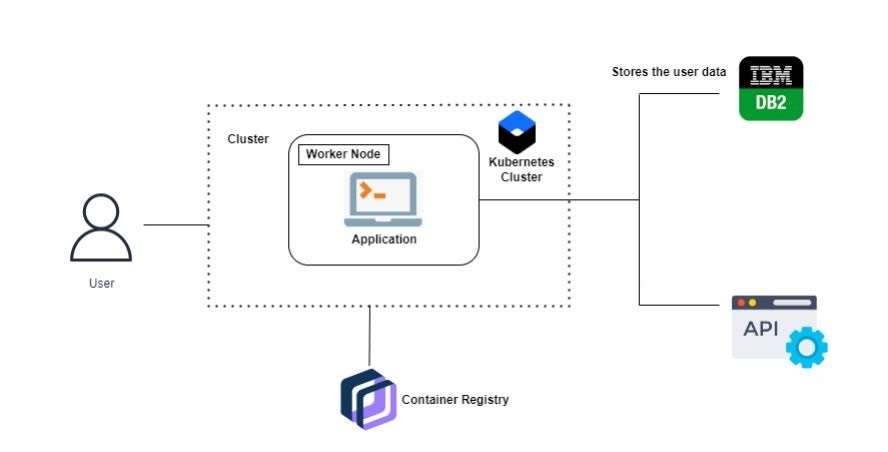
**PHARMCITY**

**INTRODUCTION:**

Online Pharmacy shopping web application is fast gaining ground as an accepted and used business paradigm. More and more business houses are implementing internet sites providing functionality for performing online medicine shopping over the online. It is reasonable to mention that the method of shoppingontheonlineisbecomingcommonplace.Thiswebapplicationistoreducehardshipsfaced by this existing system, also offers low cost ownership. Moreover, this system is designed by the particularneedofthealluserstocarryoutoperationsinasmoothandeffectivemanner.Prescription willbeanalysedbythedoctorthenonlyconsumerwillbeabletoplaceorder,alsoallthemedicines mustbeverifiedandcertifiedbytheregisteredpharmacistbeforedelivery.Ithasdatastoragefacility aswellaseasyandinteractiveretrievalofdatasuchasorderhistoryofprescriptionforfutureuselike insomecasesconsumerneedsmonthlyrefillofsomeprescriptions.Itisuserfriendlyreliableand secure with efficient interface design.

**User**: They can register for an account. After the login user they search for medical products.

**Technical Architecture:**

****

**Pre-Requisites:**

For Completing this project, these are some of the prerequisites needed

* A system with a minimum of 8 GB RAM and 512 GB Hard Disk
* Good Internet Connection
* HTML, CSS , Bootstrap
* IBM Cloud Account
* Python IDE :<https://code.visualstudio.com/download> ,<https://www.anaconda.com/products/distribution>
* Docker Desktop: <https://www.docker.com/>
* IBM CLI for container registry and Kubernetes cluster :<https://cloud.ibm.com/docs/cli?topic=cli-install-ibmcloud-cli>

**Prior-Knowledge:**

To Complete this project, one must understand the below concepts and be able to work with the tools

* HTML, CSS,Bootstrap: <https://www.w3schools.com/html/html_scripts.asp>

* SQL: <https://www.w3schools.com/sql/sql_intro.asp#:~:text=SQL%20stands%20for%20Structured%20Query,for%20Standardization%20(ISO)%20in%201987>

* Python: <https://www.javatpoint.com/python-tutorial>

* IBM DB2: <https://www.ibm.com/docs/en/db2/11.5?topic=framework-application-development-db>
* Cloud Object Storage: <https://cloud.ibm.com/docs/cloud-object-storage?topic=cloud-object-storage-getting-started-cloud-object-storage#gs-add-objects>
* Docker:<https://cloud.ibm.com/docs/Registry?topic=Registry-getting-started>, <https://docs.docker.com/get-docker/>

Kubernetes:[https://cloud.ibm.com/docs/containers?topic=containers-getting- sarted#getting- started](https://cloud.ibm.com/docs/containers?topic=containers-getting-      sarted#getting-      started)

## Project Objectives:

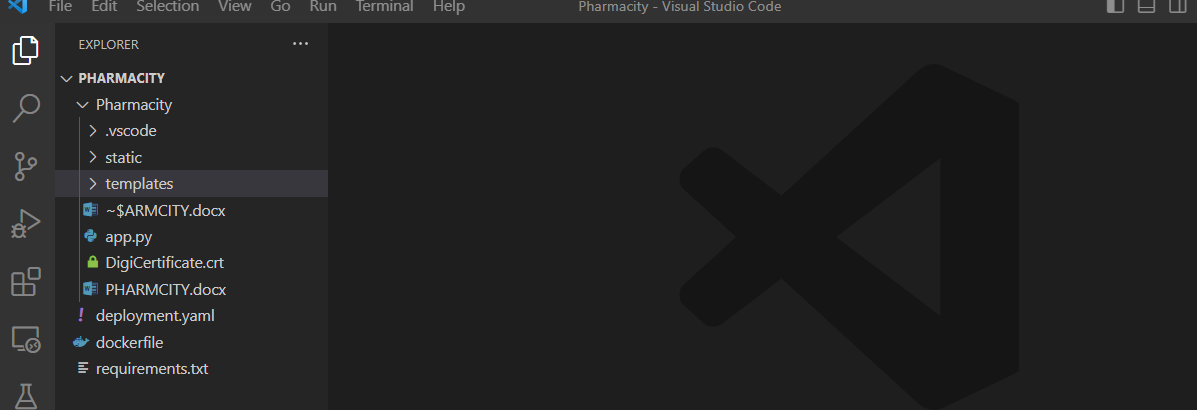
By the end of this project, you will:

* You’ll be able to work with HTML, CSS, JavaScript, and Bootstrap. Build a web application
* Integrate that web application with a Python web framework named FLASK.
* Connect with IBM DB2 for storing the data and Cloud Object Storage for storing the multimedia data.
* Deploying the application in Cloud by making it a container using Docker and managing it with Kubernetes.

**Project Flow:**

To accomplish this, we have to complete all the activities listed below,

* Creating User Interface
  + Create a User Interface using HTML, CSS/Bootstrap and JavaScript, where the user will interact and give input.
* Database Connection
  + Connecting your Application with Database
  + Creating necessary Schemas and Tables
  + Create a Cloud Object Storage to store multimedia data.
* Flask Application
* Creating python’s flask web framework to connect the front end with the back end services.
* Containerization
  + Containerize your application as a docker image and push it into Docker Desktop
  + Creating a Container Registry and pushing the image into the container using Docker Desktop.
* Kubernetes
  + Creating Kubernetes services and deploying the containerized image into the Kubernetes cluster.

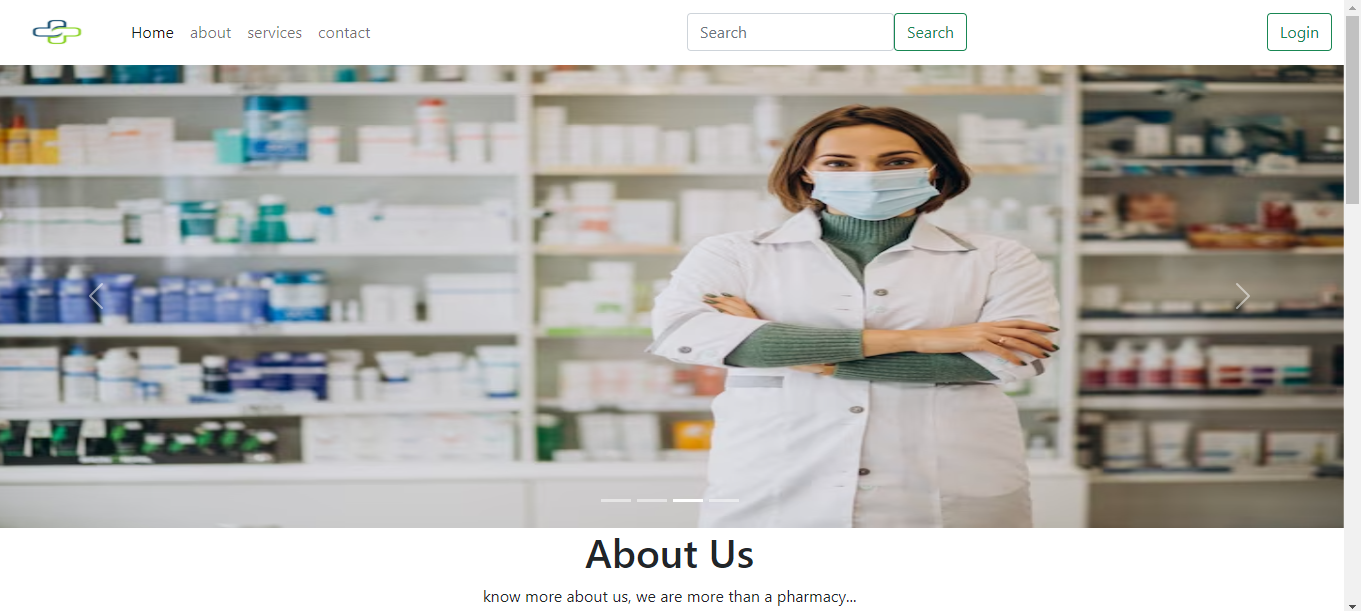
****

**Milestone 1: Creating a User Interface**

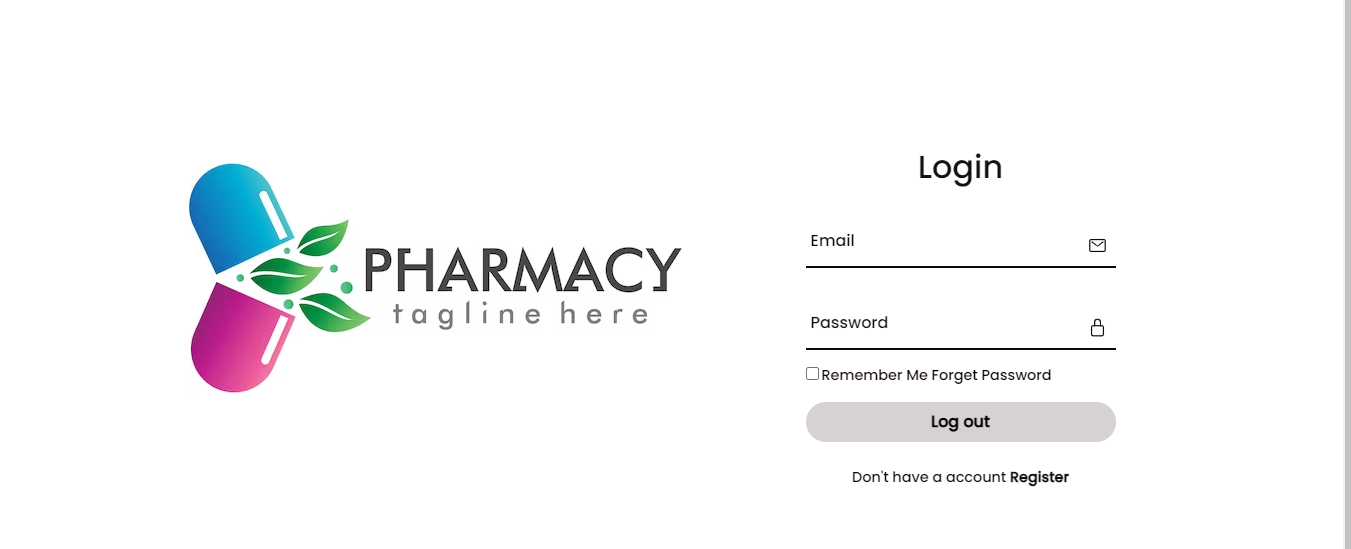
User interfaces (UI) are used to provide a way for users to interact with a system or application. They are designed to make it easy for users to access and control the functionality of a system. User interfaces play a critical role in the overall user experience by providing a way for users to interact with a system in a simple, efficient and effective manner.

**Activity 1: Creating HTML pages with CSS and JS**

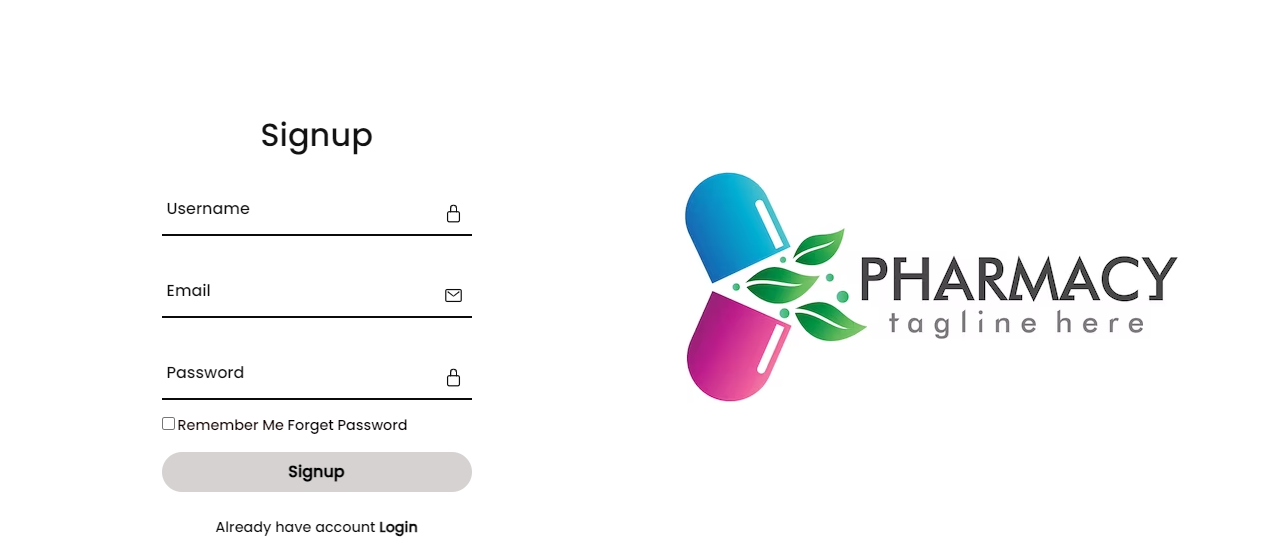
* Home page

****

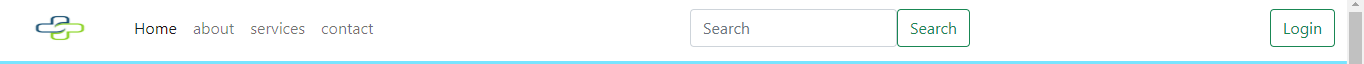
* Login page

****

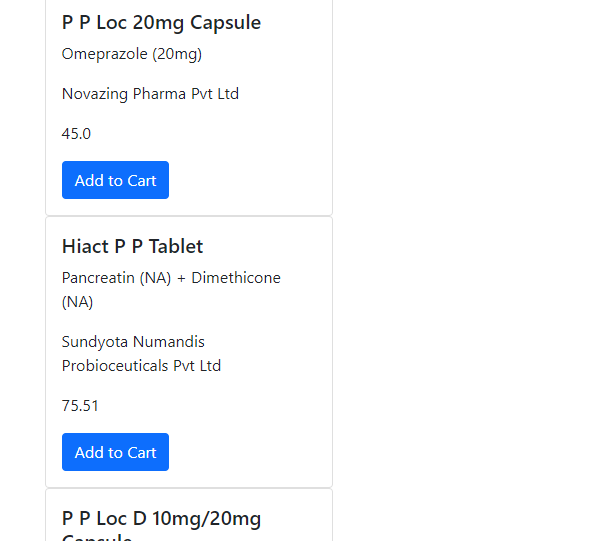
* Register page

****

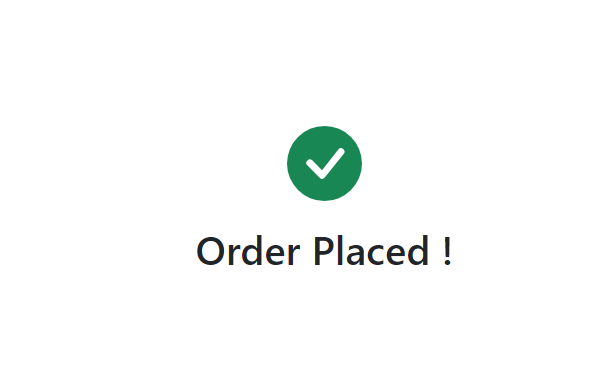
* User search page
* Enter the Name of the medicine product in the search box and click on submit button



* After clicking on the submit button the medical products will will be appear.



* After appearing of the products then add the your selected product to the cart and then that product will be displayed in the cart.
* After product displayed in the cart then click on pay button to buy the product. And enter all the details given in the payment page.
* After payment is done and the message will be shown as order placed.



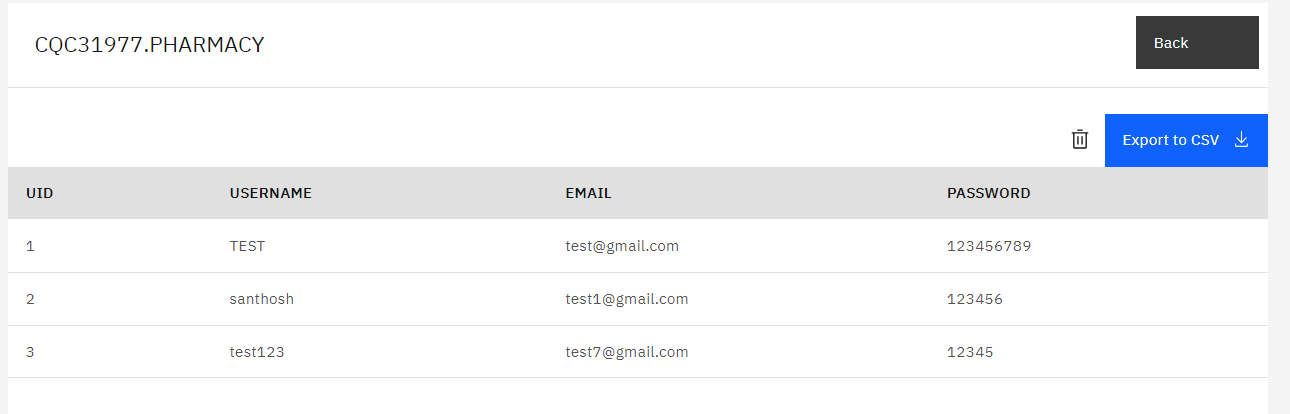
**Milestone 2: Database Connection**

Databases are used to store, organize, and retrieve data efficiently. They provide a way to manage large amounts of data and ensure that it is accurate, consistent, and readily available when needed.

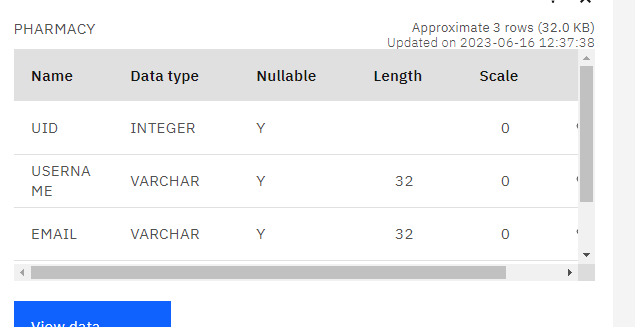
Databases play a crucial role in today's digital world by providing a reliable and efficient way to store, manage, and retrieve data.

Activity 1: Creating tables in the database

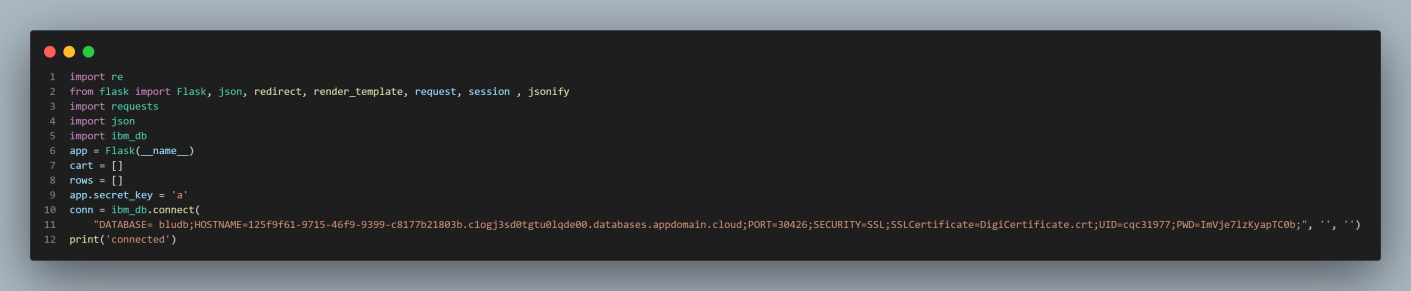
* USER1 as Table name with user & admin login credentials



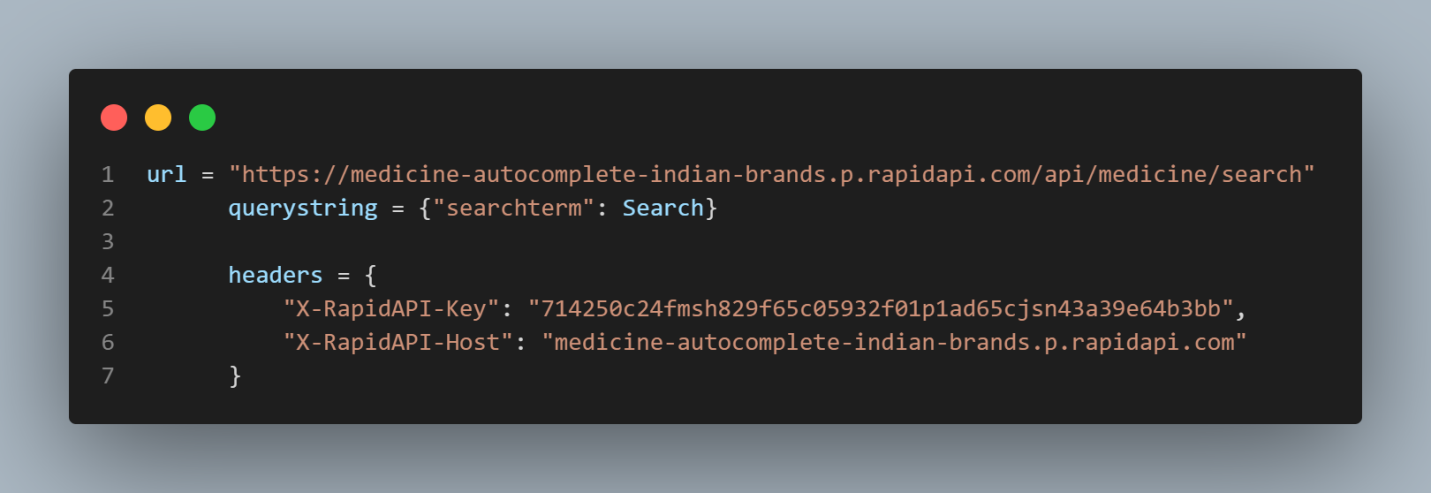
* Table name with all details



* Activity 2: Connecting database with python , importing python modulus and api keys



* **API KEY**



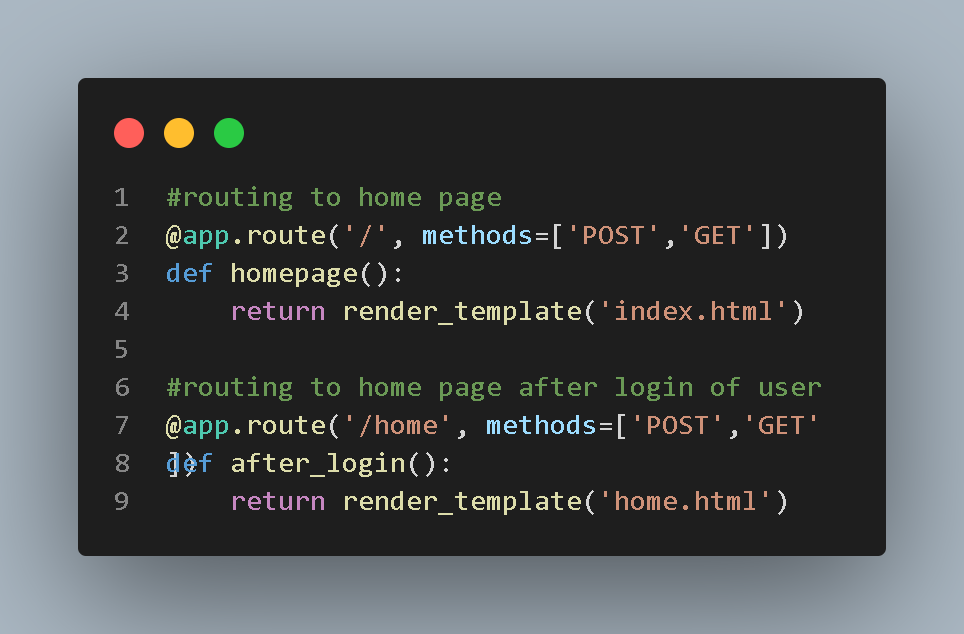
**Milestone 3: Flask Application**

Flask is a lightweight framework, that makes it easy to get started with web development. It is easy to understand and allows for the creation of small to medium-sized web applications.

Flask can be used to create a wide range of web applications, from simple single-page apps to more complex multipage applications with user authentication and data management.

**Activity 1:** Creating sample flask application

Activity 2: Connecting UI and DB with flask

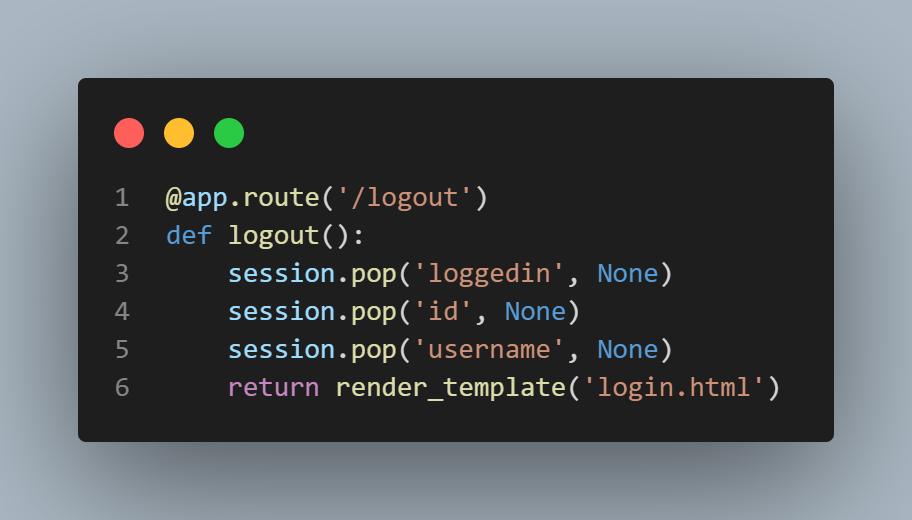
* **Task 1**: Routing to dashboard page
* **Task 2: Routin**g to User login page



* **Task 3: Routin**g to Register page



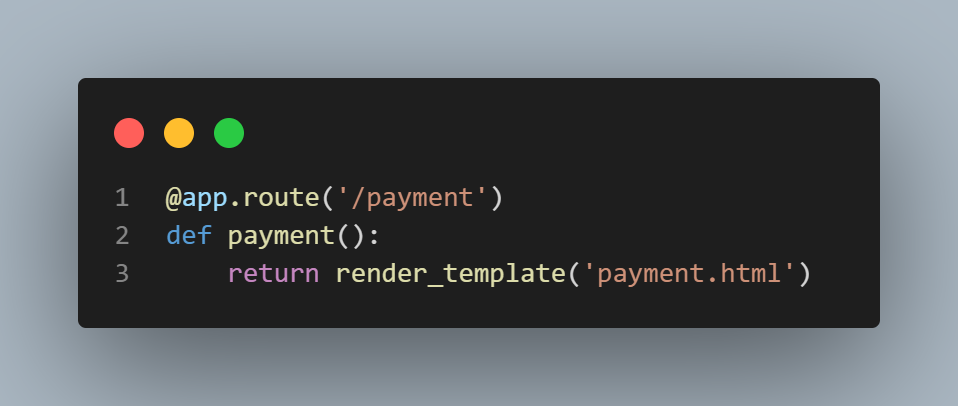
* **Task 4**: Routing to Logout page



* **Task 5:** Routing to cart page



* **Task 6:** Routing to payment page

****

* **Task 7:** Routing to thank you page

****

* **Task 7:** Main function

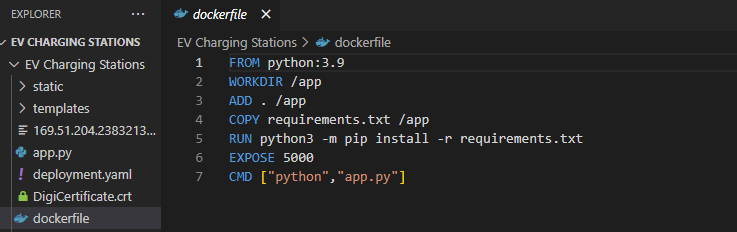


**Milestone 4: Containerization of the application**

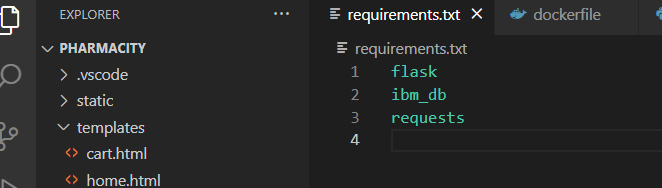
Containerization is a method of packaging and deploying software applications in a way that allows them to run consistently across different environments. Containers are a lightweight alternative to virtual machines and are used to package and deploy applications as self-contained units.

The main advantage of containerization is that it allows for more efficient use of resources and greater flexibility in terms of deployment. Containers are isolated from one another, meaning that they can run on the same host without interfering with each other. This allows for multiple applications to be run on the same machine, which can save on hardware costs.

**Activity 1**: Creating Docker file



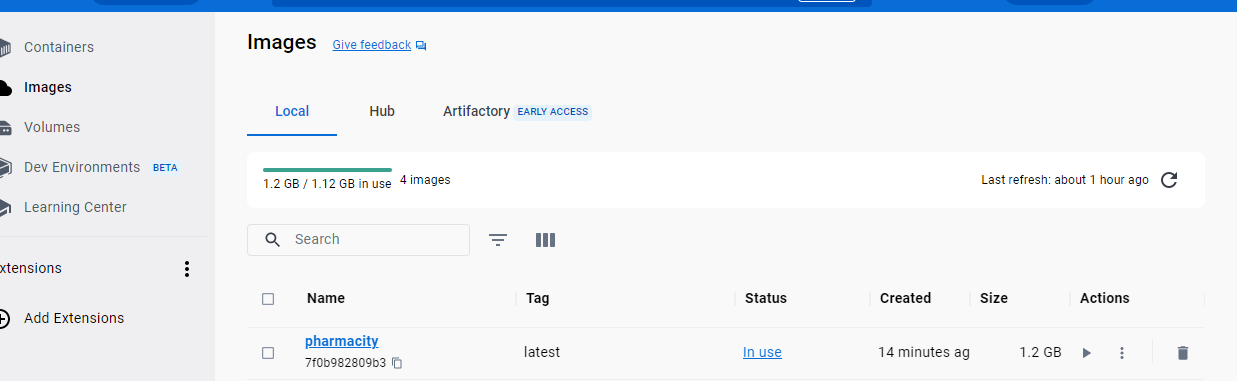
**Activity 2**: Creating requirements file



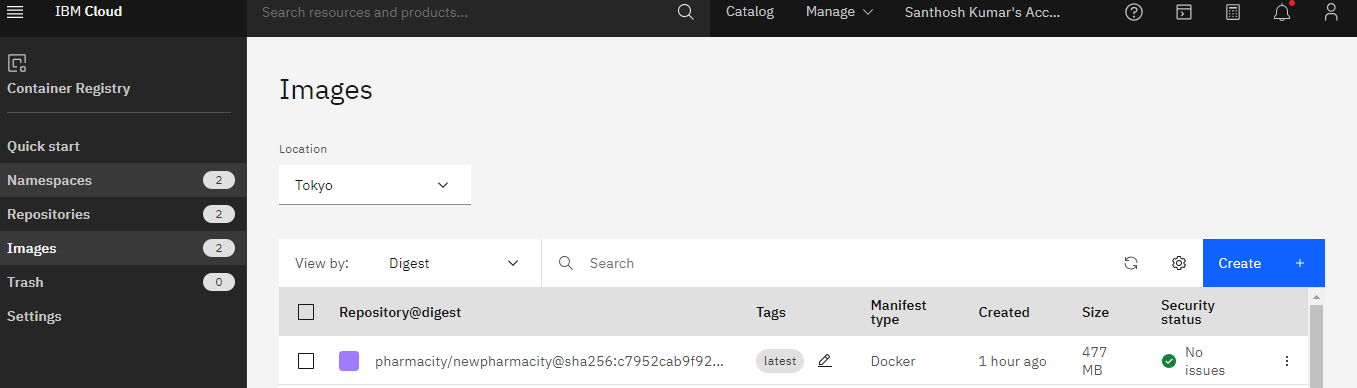
**Activity 3**: Creating requirements file



**Activity 4**: Creating Docker image:



**Activity 5**: Pushing Docker image to Container Registry:



**Milestone 5: Deploying application into Kubernetes Cluster**

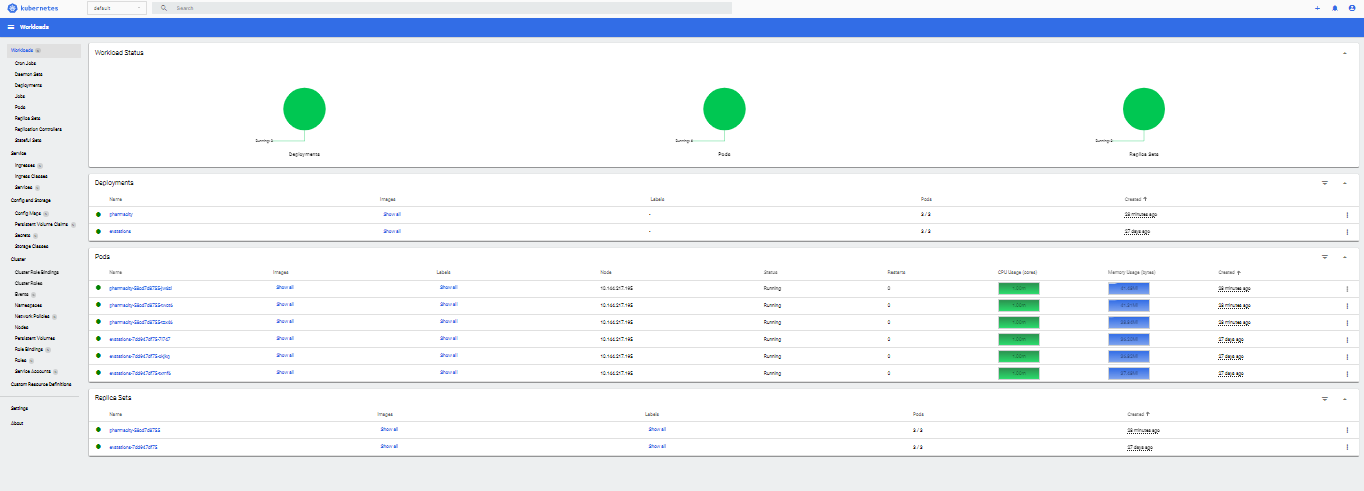
Kubernetes (often abbreviated as "K8s") is an open-source container orchestration system for automating the deployment, scaling, and management of containerized applications. It was originally developed by Google and is now maintained by the Cloud Native Computing Foundation.

Kubernetes also has a rich ecosystem of tools and extensions that can be used to manage and monitor containerized applications, including monitoring, logging, and networking.

**Activity 1**:Creating the YAML file



**Activity 2**: Deploy Image in Kubernetes cluster



**Activity 3**: After Deployment, will get an address like <http://169.51.205.49:31904/>

