CAD Assignment - 3

In the Assignment, you created 3 HTML files (index, profile and log-in), CSS files and stored the registration data in IBM db2 and did login validation.

In this assignment follow these tasks

1. Task 1: You have to containerize the whole application by using the docker
2. Task 2: Upload this docker image to the IBM container registry
3. Task 3: Then orchestrate the docker container using KubernetesLogin and Register html page

login.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Login</title>
  <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
  <header>
    <h1>Login</h1>
  </header>

  <div class="login-form">
    <form>
      <label for="username">Username:</label>
      <input type="text" id="username" name="username" required>

      <label for="password">Password:</label>
      <input type="password" id="password" name="password" required>

      <button type="submit">Login</button>
    </form>
  </div>
</body>
</html>
```

register.html

```
<!DOCTYPE html>
<html>
<head>
  <title>Register</title>
  <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
  <header>
    <h1>Register</h1>
  </header>

  <div class="register-form">
    <form>
      <label for="username">Username:</label>
      <input type="text" id="username" name="username" required>

      <label for="email">Email:</label>
      <input type="email" id="email" name="email" required>

      <label for="password">Password:</label>
      <input type="password" id="password" name="password" required>

      <button type="submit">Register</button>
    </form>
  </div>
</body>
</html>
```

style.css

```
/* CSS for both pages */

body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
}

header {
  background-color: #333;
  color: #fff;
  text-align: center;
  padding: 10px;
}

h1 {
  margin: 0;
}

form {
  max-width: 300px;
  margin: 0 auto;
  padding: 20px;
  background-color: #f5f5f5;
  border: 1px solid #ccc;
}

label {
  display: block;
  margin-bottom: 5px;
}

input[type="text"],
input[type="email"],
input[type="password"] {
  width: 100%;
  padding: 5px;
  margin-bottom: 10px;
  border: 1px solid #ccc;
}

button {
  background-color: #333;
  color: #fff;
  padding: 10px 20px;
  border: none;
  cursor: pointer;
}

button:hover {
  background-color: #555;
}
```

Task 1: Containerize the application using Docker

Create a Dockerfile in the root directory of your application. In the Dockerfile, define the base image, copy the necessary files (HTML, CSS, server code, etc.), and install any dependencies.Configure the Dockerfile to run your application when the container starts.Build the Docker image using the Dockerfile. Run the following command in the terminal:

```
docker build -t your-image-name .
```

Task 2: Upload the Docker image to the IBM Container Registry

Create an account on the IBM Cloud platform (if you haven't already).Install the IBM Cloud CLI and log in to your IBM Cloud account using the CLI.Log in to the IBM Container Registry using the CLI:

```
ibmcloud cr login
```

Tag your Docker image with the IBM Container Registry URL. Replace <region> with the appropriate region (e.g., us.icr.io):

```
docker tag your-image-name <region>/<namespace>/<your-image-name>
```

Push the Docker image to the IBM Container Registry:

```
docker push <region>/<namespace>/<your-image-name>
```

Task 3: Orchestrate the Docker container using Kubernetes

Set up a Kubernetes cluster on the IBM Cloud platform (if you haven't already).Install the IBM Cloud CLI and log in to your IBM Cloud account using the CLI.Set the context for the IBM Cloud CLI to point to your Kubernetes cluster:

```
: ibmcloud ks cluster config --cluster <cluster-name-or-ID>
```

Deploy your Docker container to the Kubernetes cluster using a Kubernetes deployment manifest. Create a YAML file (e.g., deployment.yaml) and define the deployment specifications.

Apply the deployment manifest to deploy the application to the Kubernetes cluster:

```
kubectl apply -f deployment.yaml
```