To complete your assignment **"Design and deploy a web application in a PaaS environment"** using **Render** (a modern PaaS platform), follow these steps. Render allows you to deploy full-stack applications easily using Git integration.

✅ **Assignment Goal:**

Design and deploy a web application (e.g., simple Node.js/Express or Flask app) using **Render**, showcasing how PaaS simplifies hosting and scaling.

🛠 **Step-by-Step Guide to Use Render for This Assignment**

1. Create a Web App (Locally)

- Choose a stack: e.g.,
    - **Node.js + Express**
    - **Python + Flask**
    - **React for frontend + Express backend**

📝 For a basic example, here's a simple **Node.js (Express)** app:

mkdir render-demo-app && cd render-demo-app

npm init -y

npm install express

Create index.js:

```
const express = require('express');
const app = express();
const PORT = process.env.PORT || 3000;

app.get('/', (req, res) => {
  res.send('Hello from Render-deployed app!');
});

app.listen(PORT, () => console.log(`Server running on port ${PORT}`));
```

Create package.json start script:

```
"scripts": {
  "start": "node index.js"
}
```

---

2. Push to GitHub

Create a GitHub repo and push your app:

git init

git remote add origin https://github.com/your-username/render-demo-app.git

git add .

git commit -m "Initial commit"

git push -u origin master

---

3. Create an Account on Render

- Go to https://render.com
- Sign up (use GitHub for easy integration)

---

4. Deploy the App on Render

- Click **"New Web Service"**
- Connect your GitHub repository
- Select the repository you just pushed

Fill in the deployment details:

- **Environment**: Node
- **Build Command**: npm install
- **Start Command**: npm start
- **Port**: Leave blank (Render uses PORT environment variable automatically)

Click **"Create Web Service"**

Render will now build and deploy your app. After a minute or so, you'll get a **public URL** like:

https://render-demo-app.onrender.com

📸 **Optional: Add a Frontend**

If you want to show a full web experience:

- Add an HTML file (or React frontend)
- Or use Flask/Express to render templates

📝 Assignment Submission Tips

In your report/presentation, include:

- App stack used (Node/Flask/etc.)
- GitHub repo link
- Render deployment link
- Screenshots of:
    - Render dashboard
    - Your live app
    - App code and GitHub

💡 Why Render (PaaS)?

- Zero server configuration
- GitHub integration
- Free tier available
- Auto HTTPS, custom domains, etc.

Would you like me to generate a simple project (Node/Flask/React) or a deployment-ready README.md for submission?

Great! Here's a **simple full-stack project using Node.js (Express) and EJS templating**, which is perfect for deploying on Render and fulfilling your "web application in a PaaS environment" assignment.

📁 Project: "Student Portal"

🔧 Tech Stack:

- Backend: Node.js with Express
- Templating: EJS (for dynamic HTML rendering)
- No database (for simplicity, static content)
- Deploy on Render (PaaS)

🚀 Features:

- Home page
- About page
- Contact form (no backend logic, just a form)

📦 **Folder Structure:**

```
student-portal/
|
├── views/
|   ├── index.ejs
|   ├── about.ejs
|   └── contact.ejs
|
├── public/
|   └── styles.css
|
├── index.js
├── package.json
```

📄 **package.json**

```
{
  "name": "student-portal",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "start": "node index.js"
  },
  "dependencies": {
    "ejs": "^3.1.8",
    "express": "^4.18.2"
  }
}
```

---

index.js (Main Server File)

```
const express = require('express');
const app = express();
const PORT = process.env.PORT || 3000;

app.set('view engine', 'ejs');
app.use(express.static('public'));

app.get('/', (req, res) => {
  res.render('index');
});

app.get('/about', (req, res) => {
  res.render('about');
});

app.get('/contact', (req, res) => {
  res.render('contact');
});

app.listen(PORT, () => {
  console.log(`Server is running on port ${PORT}`);
});
```

---

views/index.ejs

```
<!DOCTYPE html>
<html>
<head>
  <title>Student Portal</title>
  <link rel="stylesheet" href="/styles.css" />
```

```
</head>
<body>
  <h1>Welcome to the Student Portal</h1>
  <a href="/about">About</a> | <a href="/contact">Contact</a>
</body>
</html>
```

views/about.ejs

```
<!DOCTYPE html>
<html>
<head>
  <title>About</title>
</head>
<body>
  <h1>About This Portal</h1>
  <p>This is a student-focused web application built using Node.js and deployed on Render.</p>
  <a href="/">Back</a>
</body>
</html>
```

views/contact.ejs

```
<!DOCTYPE html>
<html>
<head>
  <title>Contact</title>
</head>
<body>
  <h1>Contact Us</h1>
  <form>
    <label>Name:</label><input type="text"><br>
    <label>Email:</label><input type="email"><br>
    <label>Message:</label><textarea></textarea><br>
    <button type="submit">Submit</button>
```

```
  </form>
  <a href="/">Back</a>
</body>
</html>
```

📄 **public/styles.css**

```
body {
  font-family: Arial;
  padding: 20px;
  background: #f2f2f2;
}
h1 {
  color: #333;
}
a {
  margin-right: 10px;
}
```

🧪 **Local Setup**

1. Run:
2. npm install
3. node index.js
4. Open browser: http://localhost:3000

☁️ **Deploying to Render**

Once it's working locally:

1. Push to GitHub
2. Go to https://render.com
3. Create a "Web Service"
4. Connect to GitHub, select your repo
5. Use these settings:

- **Build command**: npm install
- **Start command**: npm start

6. Done!

Would you like me to send this project as a ZIP file or generate a GitHub README.md for you to use?