UserInfo Portal – Project Summary

Date: March 25, 2025

Candidate: Vamshi Panjala

Project: UserInfo Portal (DevOps + Full-Stack)

This document summarizes the work completed, technologies used, and purpose of each command and folder in the project so far.

# 📘 Project Overview

The UserInfo Portal is a full-stack login and user information display system. It is built using HTML/CSS for the frontend, Node.js + Express for the backend, and a JSON file as a temporary database. The purpose is to simulate a company-style secure login portal that shows user information and can be extended with DevOps automation and deployment on AWS.

# 🗂️ Folder Structure & Purpose

1. frontend/ - Contains HTML and CSS files for the login page UI.

2. backend/ - Contains server.js and handles API logic using Node.js and Express.

3. database/ - Contains users.json file which stores sample user records.

4. scripts/ - Reserved for shell scripts, Dockerfiles, and Ansible playbooks.

5. .github/ - For future GitHub Actions CI/CD workflows.

6. backup/ - Stores daily copies of the project for safety.

7. setup/ - Stores configurations, tokens, and setup files.

# 📄 Key Files Created

✅ index.html - Login UI with email and password input fields.

✅ users.json - Sample user database for login validation.

✅ server.js - Backend server setup with Express and /login API.

✅ test-login.http - Used with REST Client extension in VS Code to test backend APIs.

✅ .gitignore - Prevents node\_modules and other files from being pushed to GitHub.

✅ README.md - Describes the overall purpose of the project.

# 💻 Commands Used (With Explanation)

* cd /c/UserInfoPortal

→ Navigate to main project folder.

* mkdir frontend backend database scripts .github backup setup

→ Create required folders.

* npm init -y

→ Initialize Node.js project and create package.json.

* npm install express

→ Install Express.js for creating backend APIs.

* touch server.js

→ Create main backend file for Express server.

* node server.js

→ Start backend server on localhost.

* git init

→ Initialize Git in the project directory.

* touch .gitignore

→ Create .gitignore to exclude unnecessary files.

* git add .

→ Stage all project files for Git.

* git commit -m 'Initial commit'

→ Commit changes with message.

* git remote add origin <repo-url>

→ Link local project to GitHub repo.

* git push -u origin main

→ Push the project to GitHub.