COLLEGE CODE : 9609

COLLEGE NAME : JAYAMATHA ENGINEERING COLLEGE

DEPARTMENT : CSE

STUDENT NM ID : AA8E4G1CC2CC452A5DG4428B8773DE2C

ROLL NO : 960923104010

DATE : 13/10/2025

Completed the project named as

Phase - 5 PROJECT DEMONSTRATION & DOCUMENTATION

Name : IBM-FE- SINGLE PAGE APPLICATION

SUBMITTED BY,

NAME : KIRUTHIKA.M

MOBILE NO : 9080413324

1. FINAL DEMO WALKTHROUGH

- **This is typically a live or recorded demonstration of the final, working project.**
- **Goal: To showcase the project's features and functionality to the audience (e.g., instructors, stakeholders).**
- **Content should cover:**

- **Setup: A brief overview of how to get the project running.**

- **Core Functionality: A step-by-step demonstration of the key features and use cases that the project was designed for.**

- **User Experience: Showing the typical user flow through the application or system.**

- **Conclusion: Briefly summarizing the project's success in meeting the initial objectives.**

2. PROJECT REPORT

- **The project report is a comprehensive written document summarizing the entire project lifecycle, from initial concept to final deployment.**
- **Goal: To formally document the project's scope, methodology, results, and evaluation.**
- **Key Sections often include:**

- **Introduction: Project aims, scope, and objectives.**

- **Literature Review/Background: Contextual information and technologies used.**

- **Design & Methodology: Explanation of the architecture, chosen**

tools/technologies, and implementation steps.

- **Results & Evaluation: Discussion of the achieved results, meeting the requirements, and performance analysis.**

- **Conclusion & Future Work: Summary of findings and potential improvements or next steps.**

3. SCREENSHOTS / API DOCUMENTATION

- **This section covers the technical details necessary for others to understand and potentially use or integrate with the project.**

- **Screenshots: Visual proof of the working application. Include images of the main interfaces, key features, and successful results/outputs.**

- **API Documentation (if applicable): If the project involves an Application Programming Interface (API), this is crucial. It should detail:**

- **Endpoints: The URLs for accessing different functions (e.g., /users, /products).**

- **Methods: The HTTP methods used (GET, POST, PUT, DELETE).**

- **Request/Response Formats: Examples of data structures sent to and received from the API (e.g., JSON payload examples).**

- **Authentication/Authorization: How users/services secure access to the API.**

4. CHALLENGES & SOLUTIONS

- **This is a reflective section where you document the difficulties encountered during the project and how you overcame them.**
- **Goal: To show critical thinking, problem-solving skills, and a transparent**

account of the development process.

- **For each major challenge, document:**

- **The Problem: Clearly describe the technical, design, or conceptual hurdle faced (e.g., "Difficulty integrating the third-party payment gateway," or "Performance degradation with large datasets").**

- **The Solution: Explain the specific steps, code changes, or design decisions implemented to resolve the problem.**

- **Learnings: Briefly state what was learned from the experience.**

5. GITHUB README & SETUP GUIDE

- **The README file is the first thing people see in your code repository and is essential for project visibility and usability.**
- **GitHub README: Should contain a concise, professional overview of the project, including:**

- **Project Title & Description: A brief, compelling summary.**

- **Features: A bulleted list of the main capabilities.**

- **Technologies Used: A list of the tech stack (e.g., React, Python/Flask, MongoDB).**

- **Links: A link to the live deployment (if available).**
- **Setup Guide: Detailed, step-by-step instructions for a new user or developer to clone, install dependencies, configure, and successfully run the project locally. This should be clear enough for someone unfamiliar with your project to get it running without assistance.**

6. FINAL SUBMISSION (REPO + DEPLOYED LINK)

- **This is the delivery mechanism for the project's final artifacts.**

- **Repo (Repository): The final, clean, and complete source code, typically submitted via a link to a GitHub/GitLab repository. The code should be well-organized and include any necessary configuration files.**

- **Deployed Link: A live URL where the final, working application or system can be accessed and tested by the evaluators. This confirms the project is operational in a production or staging environment.**

7.GITHUB REPOSITORY SETUP GUIDE

- **Setting up a new repository is the first step to starting a new project on GitHub.**

- **Steps to Create a New Repository (via GitHub Website)**

- **Log in to GitHub.**

- **In the upper-right corner of any page, click the + sign, then select New repository.**

- **Fill in the details:**

- **Repository name: Choose a short, memorable name for your project (e.g., web-data-scraper).**

- **Description (Optional): Add a brief line explaining what the repository is for.**

- **Visibility: Choose Public (visible to everyone) or Private (only you and collaborators can see it).**

- **Initialize this repository with:**

- **\checkmark Add a README file: Highly recommended. This creates the**

essential README.md file you can edit later.

- **Add .gitignore: Choose a template for your main programming language (e.g., Node, Python) to automatically exclude unnecessary files (like build files or dependencies).**

- **Choose a license: Select a license (like MIT, Apache, etc.) if you want to specify how others can use your code.**

- **Click Create repository.**
- **Steps to Link a Local Project to GitHub (Cloning)**
- **If you already have a repository on GitHub, you can bring it to your local machine:**

- **On your GitHub repository page, click the green <> Code button.**

- **Copy the URL (HTTPS or SSH).**

- **Open your terminal or command prompt.**

- **Use the git clone command with the copied URL:**
- **Git clone [THE_COPIED_URL]**

- **Basic Git Workflow Commands**
- **Once you have a repository, here are the commands you'll use most often:**

- **Command Purpose**

- **Git status Shows which files are changed, staged, or untracked.**
- **Git add . Stages all changes for the next commit.**
- **Git commit -m "Your message" Records the staged changes with a descriptive message.**
- **Git push Uploads your local commits to the remote GitHub**

repository.

- **Git pull Downloads and integrates changes from the remote repository to your local machine.**

8.GITHUB README GUIDE

- **The README.md file is the first thing people see. It acts as the front door and instruction manual for your project. It is written using Markdown syntax.**
- **Essential Components of a Good README**
- **| Component | Description | Example Content |**
- **|---|---|---|**
- **| Project Title & Badges | A clear title and optional badges for status (build status, license, version). | # Project Name |**
- **| Description | A one-to-two paragraph explanation of what the project is, what problem it solves, and why it's useful. | ”This project is a Python-based web scraper that pulls daily stock prices…” |**
- **| Table of Contents | (For long READMEs) Links to the main sections for easy navigation. | * Installation |**
- **| Installation | Step-by-step instructions on how to set up the project locally. Include dependencies, prerequisites, and installation commands. | “1. Clone the repo: git clone …\n2. Install dependencies: pip install -r requirements.txt” |**
- **| Usage | Examples of how to run and use the software. Include code snippets or screenshots/GIFs if applicable. | “To run the scraper, use: python main.py –symbol=GOOG” |**
- **| Contributing | Guidelines for how others can contribute to the project (e.g., reporting bugs, submitting pull requests). | “See CONTRIBUTING.md for details.” |**
- **| License | Specify the project’s license (matches what you set in the repository creation). | “Distributed under the MIT License. See LICENSE for more information.” |**
- **| Contact/Links | Who to contact for questions and links to the project’s documentation or live demo. | “Project Link: [Live Demo URL]” |**

- **Markdown Quick Reference**
- **| Feature | Markdown Syntax |**
- **|---|---|**
- **| Main Heading | # Title |**
- **| Sub Heading | ## Subtitle |**
- **| Bold Text | Bold text |**
- **| Italic Text | Italic text |**
- **| Unordered List | * Item 1\n* Item 2 |**
- **| Ordered List | 1. First item\n2. Second item |**
- **| Code Block | \nconsole.log('Hello!');\n |**
- **| Link | Link Text |**
- **| Image | |**
- **Tip: You can often find a good template by searching for "awesome README templates" on GitHub!**

9.LESSONS LEARNED

- **Every technical issue is connected — solving one often reveals the next.**
- **Backend and frontend must be developed in sync for smooth data flow.**
- **Proper Git workflow prevents conflicts during rapid debugging.**
- **Deployment success depends on clean, well-tested local builds.**
- **Real-time testing and teamwork are key to handling chain reactions in large systems.**
- **Documentation and version tracking help prevent repeated mistakes.**

10.GITHUB README & SETUP GUIDE

- **During the project development, several technical, design, integration, deployment, and teamwork challenges were encountered.**

- **Each challenge was interconnected, meaning fixing one helped in understanding or solving another.**

- **This continuous learning cycle made the system more efficient and the team more experienced.**

11. FINAL SUBMISSION (REPO + DEPLOYED LINK)

https://github.com/sakthikaveri23/NM-IBM-Project