**REGISTER NUMBER: 71772117152**

**NAME : ABHIRAJ SINGH**

**Travel Blog Project Documentation**

**Objective**

The objective of this project is to create a dynamic and interactive travel blog website using Flask, a Python web framework. This website will allow users to explore various travel destinations, read travel articles, view images, and leave comments. The project aims to provide an engaging platform for travel enthusiasts to share their experiences and discover new places.

**Design Thinking Process**

**1. Research and Analysis**

- We conducted research on existing travel blogs to understand their features and user experience.

- Identified key features such as user registration, article creation, comments, and image uploads.

- Developed user personas to better tailor the website to the target audience.

**2. Ideation**

- Brainstormed website design and structure.

- Discussed color schemes, typography, and visual elements to create an appealing user interface.

- Decided to use a card-based layout for articles and a user-friendly navigation menu.

**3. Prototyping**

- Created wireframes and mockups to visualize the website's layout and functionality.

- Designed the database schema for storing user information, articles, comments, and images.

**4. Development Phases**

**Phase 1: Setting Up Flask Application**

- Initialized a Flask application.

- Defined routes for home, articles, article detail, user registration, and user login.

**Phase 2: Database Integration**

- Integrated a relational database (e.g., SQLite or PostgreSQL) to store data.

- Created models for users, articles, and comments.

- Implemented user authentication and authorization.

**Phase 3: Front-End Development**

- Created HTML templates for rendering pages.

- Designed the user interface using CSS and JavaScript.

- Implemented features like user registration, article creation, image upload, and comment submission.

**Phase 4: Testing and Debugging**

- Conducted unit testing for critical functions.

- Fixed bugs and ensured website responsiveness.

**Phase 5: Deployment**

- Deployed the Flask application to a web hosting service (e.g., Heroku or IBM Cloud).

- Set up a domain name and secured the website with HTTPS.

**Website Structure**

The website consists of the following main components:

- Home Page: Displays featured articles and a navigation menu.

- Articles Page: Lists all available articles with a brief description.

- Article Detail Page: Displays a single article with comments and images.

- User Registration and Login: Allows users to create accounts and log in.

- Article Creation: Registered users can create and publish articles.

- Image Upload: Users can add images to their articles.

- Comment Section: Users can leave comments on articles.

**Technical Implementation Details**

- **Flask**: The web framework used for the project.

- **Database**: A relational database (e.g., SQLite or PostgreSQL) for data storage.

- **HTML/CSS/JavaScript**: Front-end development for rendering pages and user interface.

- **User Authentication**: Implemented using Flask-Login for session management.

- **Image Upload**: Supported using a file upload feature.

- **Comment System**: Users can leave comments on articles.

- **Deployment**: The project can be deployed to a web hosting service, such as IBM Cloud.

**GitHub Repository**

You can find the project's code and files in the GitHub repository:

[GitHub Repository Link] => https://github.com/boopathihari/Travel-Blog-Source-code

**Deployment Instructions**

To deploy the blog using IBM Cloud Static Web Apps, follow these steps:

1. Clone the GitHub repository to your local machine.

2. Create an IBM Cloud account if you don't have one.

3. Set up an IBM Cloud Static Web Apps project.

4. Configure the project to use the GitHub repository as the source.

5. Set the build command and output directory for the Flask application.

6. Deploy the project on IBM Cloud.

**README File**

The README file in the GitHub repository provides detailed instructions on how to navigate the website, update content, and manage dependencies. It includes information on setting up the development environment, configuring the database, and deploying the website.

**Screenshots**

Here are some screenshots of the project's user interface:

