**Project Report**

**On**

**E-COMMERCE WEBSITE**

**CSC Computer Education**

**FULL STACK PYTHON**

**GUIDED BY: PREPARED BY:-**

**MR. Mr. R DEIVASIGAMANI, MSc., P BALAGANESH**

****

**YEAR-20xx-xx**

**SUBMITTED TO**

**CSC COMPUTER EDUCATION**

**Branch : PERUNDURAI**

**64 First Floor, Amman Complex, Opp New Bus Stand,**

**Perundurai. Erode(district). 638052.**

**Email:** [**cscperundurai@gmail.com**](mailto:cscperundurai@gmail.com)**,** [**perunduraicsc@gmail.com**](mailto:perunduraicsc@gmail.com)

**Phone: 7200001338,9965553027**

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Chapter** | **Title** | **Page No.** |
| **1** | **Introduction** |  |
|  |  |  |
|  |
|  |
|  |
|  |
| **2** |  |  |
|  |  |  |
|  |
|  |
| **3** |  |  |
|  |  |  |
|  |
|  |
| **4** |  |  |
|  |  |  |
|  |
|  |
| **5** |  |  |
|  |  |  |
|  |
|  |
| **6** |  |  |
| **7** |  |  |
| **8** |  |  |

**1. Project Overview**

**1.1. Project Title**

**Title:** E-Commerce Website

**1.2. Project Description**

**Description: Welcome to Art Haven**

Explore a curated collection of exquisite drawings at Art Haven, your premier online destination for art enthusiasts and collectors. Whether you’re looking for contemporary pieces, classic sketches, or unique illustrations, our store offers a diverse range of artworks to suit every taste and style.

**Why Choose Us?**

**Exclusive Collection**: Discover a wide variety of drawings from talented artists around the world.

* **Authenticity Guaranteed**: Each piece is verified for authenticity, ensuring you receive genuine artwork.
* **Secure Transactions**: Shop with confidence using our secure payment methods and data protection measures.
* **Fast Delivery**: Enjoy prompt and reliable shipping to get your artwork delivered safely to your doorstep.
* **Customer Support**: Our dedicated team is here to assist you with any questions or concerns.

**How It Works**

1. **Browse**: Navigate through our user-friendly website to explore our extensive collection of drawings.
2. **Select**: Choose your favorite pieces in cart and add them to your cart.
3. **Purchase**: Proceed to checkout, provide your shipping details, and select your preferred payment method.
4. **Receive**: Sit back and relax while we carefully package and ship your artwork to you.

Join our community of art lovers and start building your collection with Art Haven. Happy shopping!

**1.3. Objectives**

* To develop a user-friendly web application using Flask.
* The customer can login in web page.
* The customer data is store in database and before login run the python file of user.
* To buy the products click cart icon in the navigation bar.
* Select the products to buy in cart.
* After selecting the products in the cart, click pay now button.
* Fill the details in payment form to buy the products.
* After completing the payment form, click order now button to place the order.
* The customer order is confirmed.
* The customer will receive the product within 2 or 3 days.
* The customer can contact us by filling the contact us details in web page.

**1.4. Target Audience**

**Audience:** Who loves and interested in the art work. They will buy our products. Everyone can buy our products, our products are made for every customers to satisfy them.

**2. Requirements**

**2.1. Functional Requirements**

* User registration and login functionality.
* Shopping cart, Checkout and Order.
* Customer support, return and exchanges.
* Integration with payments and shipping charge is free.

**2.2. Non-Functional Requirements**

* Performance: The application should handle up to 1000 simultaneous users.
* Security: User data must be protected, and password hashing should be implemented.
* Usability: The application should be intuitive and user-friendly.

**2.3. Technical Requirements**

* Programming Language: Python as a can simply be referred to as a multi-functionality having capabilities to operate in multiple ways. Surprisingly a developer can do wonders and can easily develop software by implementing easy methods. Being an interpreted language (smoothness in dev. process), this platform is absolutely free and open for all, it also offers platform independency, meaning their (Python’s) code can run on any platform without making any changes (such as Linux, macOS, etc.)
* Framework: Flask is a lightweight backend framework with minimal dependencies. Flask is easy to learn because its simple and intuitive API makes it easy to learn and use for beginners. Flask is a flexible Framework because it allows you to customize and extend the framework to suit your needs easily. Flask can be used with any database like:- SQL and NoSQL and with any Frontend Technology such as React or Angular. Flask is great for small to medium projects that do not require the complexity of a large framework.
* Database: A database is an organized collection of data stored in a computer system and usually controlled by a database management system (DBMS). The data in common databases is modeled in tables, making querying and processing efficient. Structured query language (SQL) is commonly used for data querying and writing. The Database is an essential part of our life. We encounter several activities that involve our interaction with databases, for example in the bank, in the railway station, in school, in a grocery store, etc. These are the instances where we need to store a large amount of data in one place and fetch these data easily.
* Frontend: HTML, CSS, JavaScript

**3. System Architecture**

**3.1. High-Level Architecture**

A diagram or description of the system architecture, including:

* **Client-side:** The user interface in the browser.
* **Server-side:** Flask application handling requests and responses.
* **Database:** Storage for user data and tasks.
* **APIs:** Any external services or APIs used.

**3.2. Components**

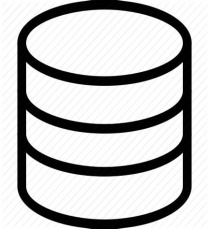
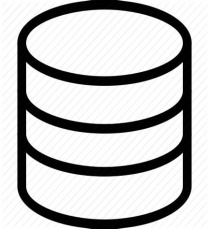
* **Frontend:** HTML/CSS/JavaScript templates for user interaction.
* **Backend:** Flask routes and handlers for business logic.
* **Database:** Schema and models for user data and tasks.

Home

User

Contact Us

Cart

** **

Payment

Order

Database

Database

**4. Design**

**4.1. Database Design**

* **Tables:** Users, Tasks, etc.
* **Relationships:** How tables relate to one another (e.g., one-to-many relationship between users and tasks).
* **Schema:** Detailed schema definition for each table.

**4.2. Application Flow**

* **User Flow:** Diagram or description of how users interact with the application.
* **Request-Response Flow:** How the application processes requests and sends responses.

**5. Implementation**

**5.1. Project Setup**

* **Dependencies:** List of required libraries and tools (e.g., Flask, SQLAlchemy).
* **Installation Instructions:** Steps to set up the development environment.

**5.2. Code Structure**

* **File Organization:** Description of the file structure and organization.
* **Key Modules:** Explanation of important modules and their responsibilities.

**5.3. Example Code**

Provide snippets of critical code for key functionalities, such as user authentication or task management.

**6. Testing**

**6.1. Testing Strategy**

* **Unit Testing:** Tests for individual components or functions.
* **Integration Testing:** Tests for interactions between components.
* **End-to-End Testing:** Tests for complete workflows from start to finish.

**6.2. Tools**

* **Testing Frameworks:** pytest, unittest, etc.
* **Mocking:** Use of mock objects or services for testing.

**7. References**

List any resources, libraries, frameworks, or tools that were used or referenced during the project.