BACKEND DEVELOPER USING FLASK AT BHAWANI INFOTECH

Tribhuvan University

Institute of Science and Technology

A Final Year Internship Report Submission in
Partial Fulfillment of the Requirement for the Degree of
Bachelor of Science in Computer Science and Information Technology

Submitted To

National College of Computer Studies
Department of Computer Science and Information Technology
Paknajol, Kathmandu

Submitted by

Sudip Upreti (23855/076)

Under the Supervision of

Mr. Chhetra Bahadur Chhetri

3rd July 2024

DECLARATION

The internship report entitled "**BACKEND DEVELOPER AT BHAWANI INFOTECH**" is being submitted to the Department of Computer Science and Information Technology, Paknajol, Kathmandu, Nepal, under the supervision of **Mr. Chhetra Bahadur Chhetri**. This project is original and has not been submitted earlier in part or full in this or any other form to any university or institute, here or elsewhere, for the award of any degree.

By
Sudip Upreti (23855/076)

MENTORS' RECOMMENDATION FROM COMPANY

I hereby recommend that this report is prepared under my mentorship by **Mr. Sudip Upreti** entitled **"Backend Development Intern"** in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Information Technology (BSc.CSIT) to be processed for the evaluation.

Mr. Srijan Pokharel
Internship Mentor

Bhawani InfoTech Pvt. Ltd

Basundhara, Kathmandu, Nepal

SUPERVISORS' RECOMMENDATION

I hereby recommend that this report is prepared under my mentorship by **Mr. Sudip Upreti** entitled "**Backend Development Intern**" in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Information Technology (BSc.CSIT) to be processed for the evaluation.

____

Mr. Chhetra Bahadur Chhetri
Supervisor
National College of Computer Studies (NCCS)
Paknajol, Kathmandu

EXAMINERS' APPROVAL LETTER

This is to certify that this internship report carried out by **Mr. Sudip Upreti** entitled **"Backend Development Intern"** under the guidance and supervision is a report for the degree of Bachelor of Science in Computer Science and Information Technology (BSc.CSIT) at National College of Computer Studies (NCCS).

Mr. Srijan Pokharel	**Mr. Chhetra Bahadur Chhetri**
Internship Mentor	Supervisor
Bhawani InfoTech Pvt. Ltd	National College of Computer Studies
Basundhara, Kathmandu, Nepal	Paknajol, Kathmandu

External Examiner Date:

ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to everyone who supported me throughout my internship journey at Bhawani Infotech. First and foremost, I am deeply thankful to **Mr. Kapil Pathak**, CEO of **Bhawani Infotech**, for providing me with the opportunity to be part of this esteemed organization and for fostering an environment conducive to learning and growth.

I am immensely grateful to my mentor, **Mr. Srijan Pokharel**, whose guidance, expertise, and encouragement were invaluable throughout the internship. His mentorship not only enriched my technical skills but also inspired me to strive for excellence in every aspect of my work.

I would also like to extend my appreciation to my college mentor, **Mr. Chhetra Bahadur Chhetri**, for his unwavering support, constructive feedback, and continuous encouragement during the course of my internship.

Additionally, I am thankful to all the colleagues at Bhawani Infotech who generously shared their knowledge and experiences, making my internship experience enriching and fulfilling.

Thank you all for believing in me and for contributing to my growth and development as a professional.

ABSTRACT

This project involves the development of two distinct websites: a blog site and an e-commerce site, both built using the Python Flask framework with SQL Alchemy as the database management system. The blog site enables users to post, edit, and like on articles, while the e-commerce site provides features such as product browsing, adding items to the cart, user ratings, and a seamless checkout process. Key functionalities implemented include user authentication to secure access, a notification system to keep users informed, CRUD operations, integration of the Khalti payment gateway for secure transactions and advanced features like rating, like etc. These projects provided practical insights into software development practices, enhanced technical skills in Flask, SQL Alchemy, and frontend technologies, and offered valuable lessons in teamwork and problem-solving during the internship at Bhawani Infotech.

Keywords: *Flask, SQL Alchemy, Backend, Bhawani Infotech*

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

API	Application Program Interface
CRUD	Create, Read, Update, Delete
CSS	Cascading Style Sheets
DB	Database
ER	Entity-Relationship
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IDE	Integrated Development Environment
IT	Information Technology
JS	JavaScript
ORM	Object-Relational Mapping
SQL	Structured Query Language
URL	Uniform Resource Locator

Chapter 1. INTRODUCTION

1.1. Introduction

During my internship, I had the opportunity to work on two distinct web development projects: an e-commerce site and a blog site. These projects provided a comprehensive experience in both front-end and back-end development, leveraging modern technologies to build functional and user-friendly web applications.

The Blog Site Project is an innovative platform designed to provide users with a comprehensive and interactive blogging experience. This project integrates essential features such as user authentication, post creation, liking posts, and social networking functionalities like sending friend requests. The aim is to create a community-driven site where users can share their thoughts, interact with others, and build connections. The Blog Site Project is designed to go beyond traditional blogging by incorporating social networking features, thus enhancing user interaction and community building. With a focus on robust functionality and user experience, this platform will cater to both new and seasoned bloggers, providing them with the tools to create, share, and connect.

In the ever-expanding digital landscape, e-commerce has become an indispensable facet of modern business, facilitating seamless transactions and unparalleled convenience for consumers worldwide. Our e-commerce website represents a culmination of robust features designed to enhance user experience and streamline operations for both administrators and customers alike.

From comprehensive user authentication protocols to intuitive product management functionalities such as adding products and categories, our platform ensures a seamless process from browsing to checkout. Customers can effortlessly navigate through our diverse catalog, add items to their cart, and proceed to securely place orders. The system further includes advanced features like order management, invoice generation, and product filtering to enhance usability and efficiency.

Additionally, our platform fosters customer engagement through interactive features such as product ratings and likes.

1.2. Problem Statement

Running e-commerce and blog websites comes with its share of challenges. E-commerce sites often struggle with user-friendly designs that make it hard for customers to browse and buy products smoothly. This leads to lower satisfaction and inefficiencies in managing orders and inventory. Security is also a big concern, with risks of data breaches due to weak protections. Similarly, blog sites face usability issues with interfaces that make it difficult for users to create and manage posts easily. Security problems also affect them, potentially compromising user information. Both types of sites lack effective ways to gather feedback from users, which makes it hard to improve and engage customers effectively.

1.3. Objectives

The main objectives of this project are as follows:

Blog Site

- To ensure secure user registration, login, and authentication.
- To enables users to search and sort blog posts and includes social media features for managing friendships and notifications.

Ecommerce Site

- Implement comprehensive product management and seamless shopping cart functionality.
- Develop a secure checkout system with Khalti API integration and personalized user accounts.

1.4. Scope and Limitation

1.4.1. Scope for Blog Site

The blog site seeks to engage users with a wide array of content topics, supported by user-friendly features for content creation, editing, scheduling, and categorization. It will encourage user interaction through comments, social sharing, and subscriptions, while leveraging analytics tools to monitor visitor demographics and content performance. The platform aims to foster a vibrant community around its content and provide valuable insights for content strategy refinement.

The limitation of the project:

- Potential usability challenges with the blog's interface affecting user experience.
- Security vulnerabilities related to user data storage and platform integrity.
- Technical constraints impacting scalability and performance during peak traffic.
- Dependence on user engagement for feedback and content interaction.

1.4.2 Scope for Ecommerce Site

The e-commerce site aims to offer a comprehensive online shopping experience with a diverse range of products, robust user features including secure checkout and order tracking, and efficient administrative tools for inventory management and sales analytics. It will integrate with secure payment gateways to facilitate seamless transactions and provide a scalable platform capable of handling increased user traffic and product offerings over time.

The limitation of the project:

- Limited scalability for handling exceptionally high user traffic and large product catalogs may require further optimization.
- Dependency on third-party payment gateways could lead to potential disruptions if those services experience issues.

1.5. Report Organization

The report is structured as follows:

Chapter 1 provides an introduction to the internship report, detailing the project's focus, problem statement, objectives, scope, limitations, and the overall organization of the report.

Chapter 2 presents an overview of the host organization, including its hierarchy, operational domains, and the specific department where the internship took place. It may also feature a literature review if applicable.

Chapter 3 delves into the core of the internship experience, describing the roles and responsibilities, offering a weekly log of activities, outlining project descriptions, and detailing specific tasks completed.

Chapter 4 concludes the report by summarizing the internship experience, highlighting key takeaways, achievements, challenges, and learning outcomes, and providing a reflective conclusion.

Chapter 2. ORGANIZATION DETAILS AND LITERATURE REVIEW

2.1. Introduction to Organization

Bhawani Infotech is a distinguished IT company celebrated for its seamless integration of cutting-edge innovation and unwavering adherence to its guiding principles. Founded with a vision to transform industries and enrich lives on a global scale, Bhawani Infotech consistently delivers pioneering and high-quality solutions to its clients.

At the core of Bhawani Infotech's success is a team of highly skilled professionals dedicated to staying ahead of the curve. By anticipating and meeting the ever-evolving needs of its customers, the company ensures that its offerings remain relevant and impactful. The expertise of Bhawani Infotech spans a wide range of services, including web development, application development, cloud and web hosting, and digital marketing, making it a comprehensive provider of innovative business solutions.

Driven by a relentless pursuit of excellence, Bhawani Infotech continually strives to stay at the forefront of technological advancements, providing practical solutions that cater to diverse needs. In the fast-paced and competitive IT industry, Bhawani Infotech stands as a beacon of quality, innovation, and timely delivery. Its mission is to empower businesses to harness the full potential of technology, fostering their growth and success.

Table 2.1: Contact Details of the Organization

Location:	Basundhara, Kathmandu, Nepal
Opening Hours	9 AM to 6 PM
Contact No:	+977 9802359174 +977 9802359175
Email Address:	bhawaniinfotech.com info@bhawaniinfotech.com
Website:	https://www.bhawaniinfotech.com/

2.2. Organization Hierarchy

Bhawani infotech comprises an administrative team, interns, junior, senior programmers and web designers.

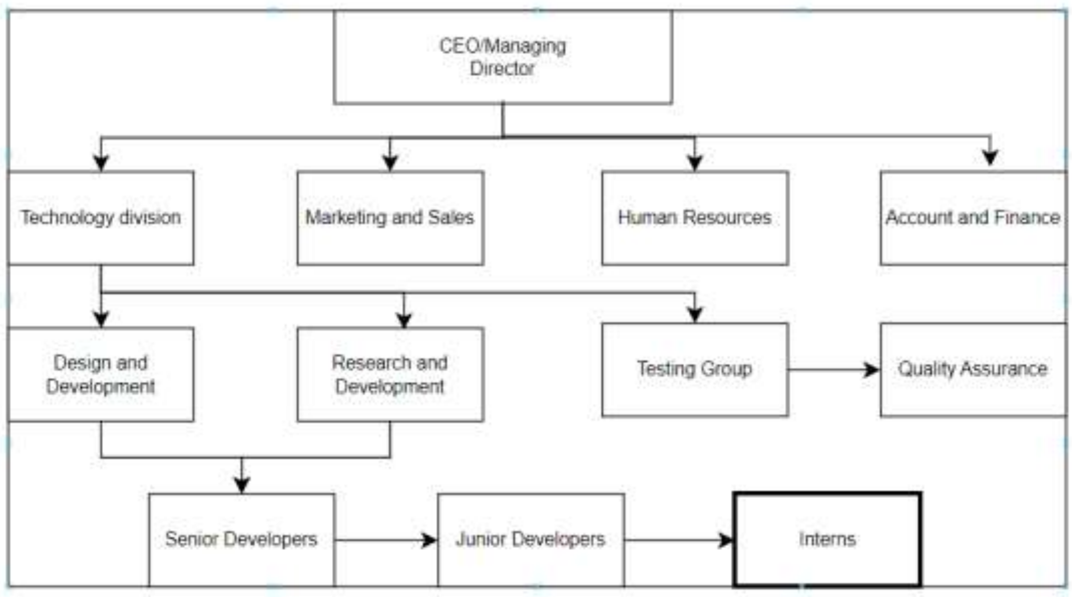


Figure 2.1: Organization Hierarchy

2.3 Working Domains of Organization

Bhawani infotech provides multiple services in websites design and development. The areas of expertise of the organization are:

- Webapp Development
- Mobile app development
- Branding

2.4 Description of Intern Department

2.4.1 Internship Period Details

The internship period is three months at Bhawani infotech Pvt. Ltd. The internship details in the organization are summarized in the following table.

Table 2.2: Internship Period Details

Internship Start Date:	17th March 2024
Internship End Date:	18th June 2024
Office Hour:	9 Am to 6 PM
Working Days:	Monday to Friday
Position:	Backend Developer
Supervisor:	Srijan Pokhrel

2.5 Literature Review

This paper reviews the benefits and challenges of developing blog sites and e-commerce platforms. The author argues that the benefits outweigh the challenges and that these platforms can significantly enhance user engagement and business operations. The paper also provides a list of the objectives for the proposed blog site and e-commerce site projects, which include developing a user-friendly content management system, integrating secure payment gateways, optimizing for search engines, and ensuring responsive design for mobile devices (Smith, 2020).

The blog site described in this paper is a comprehensive content management system (CMS) designed to facilitate content creation, management, and distribution. The system allows users to create multimedia-rich posts, manage comments, and interact with social media platforms directly from the site. This can enhance user engagement and retention, providing a seamless experience for both content creators and readers. The system also incorporates search engine optimization (SEO) features to improve visibility and attract organic traffic (Johnson, 2019).

The e-commerce site described in this paper is an advanced online store that uses modern technology to improve the shopping experience and streamline business operations. The system includes a secure payment gateway to ensure safe transactions, an inventory management system to track stock levels, and a customer relationship management (CRM) system to manage customer interactions and improve service quality. This can enhance efficiency and accuracy, particularly during high-traffic periods. The system also uses responsive design techniques to ensure that the site is accessible and user-friendly on both desktop and mobile devices, catering to the increasing number of mobile shoppers (Davis, 2018).

The system described by Brown adopts a modular approach to website development, allowing for easy customization and scalability. Once the basic site is set up, additional features such as customer reviews, wish lists, and personalized recommendations can be added to enhance the shopping experience. This modular approach ensures that the site can grow and evolve with the business, providing a flexible and sustainable solution for e-commerce operations (Brown, 2017).

Chapter 3. INTERNSHIP ACTIVITES

3.1. Roles and Responsibilities

The author's role at Bhawani infotech was as a Backend Developer Intern, contributing significantly to the web development team. Utilizing the Flask Python framework, the intern played a crucial role in developing and maintaining web applications for both an e-commerce site and a blog site.

The responsibilities of the Flask Developer Intern were as follows:

- Collaborating with the development team to design and implement backend solutions using Python and Flask.
- Working with SQLAlchemy to interact with databases SqlAlchemey
- Collaborating with frontend developers to ensure seamless integration of frontend components with the Flask backend.
- Actively participating in problem-solving activities to identify and address technical issues, ensuring smooth and efficient application operation.
- Implementing security measures to protect user data and maintain application integrity.
- Writing clean, maintainable, and well-documented code to support future development and maintenance.

3.2. Weekly Log

Table 3.1: Weekly Log

Week	Areas of Activities
1	• Orientation and Introduction about company • Python Basic
2	• Learn about git and GitHub • Er diagram for blog site • Setup blog site project
3	• User authentication • Mail api for forgot password • update email and profile picture

4	• view friend blog • Notification system
5	• Testing and debugging
6	• Er diagram for E-commerce site • Setup E-commerce site project • Start templating
7	• User Authentication(registration, login, change password)
8	• Forgot password through mail api using otp • Role based user database(admin, user)
9	• Rating functionality using fetch Api • Filter product through raring, price, category
10	• Add to cart(minus quantity, plush quantity, remove cart) • view cart
11	• Khalti integration
12	• Testing and debugging

3.3 Description of the Projects Involved During Internship

3.3.1 Blog Site

The blog site is a dynamic content management platform aimed at providing users with a seamless experience in creating, sharing, and interacting with blog posts.

- The project is developed using Flask for the backend, SQLAlchemy for database interactions, and HTML/CSS for the frontend.
- Key features include user registration and login, creating and editing blog posts, commenting on articles, and a robust categorization system to organize content.
- The site also allows users to manage their profiles and track their activity within the platform.

- Designed with user engagement in mind, the system is both user-friendly and responsive, ensuring accessibility and ease of use across various devices.

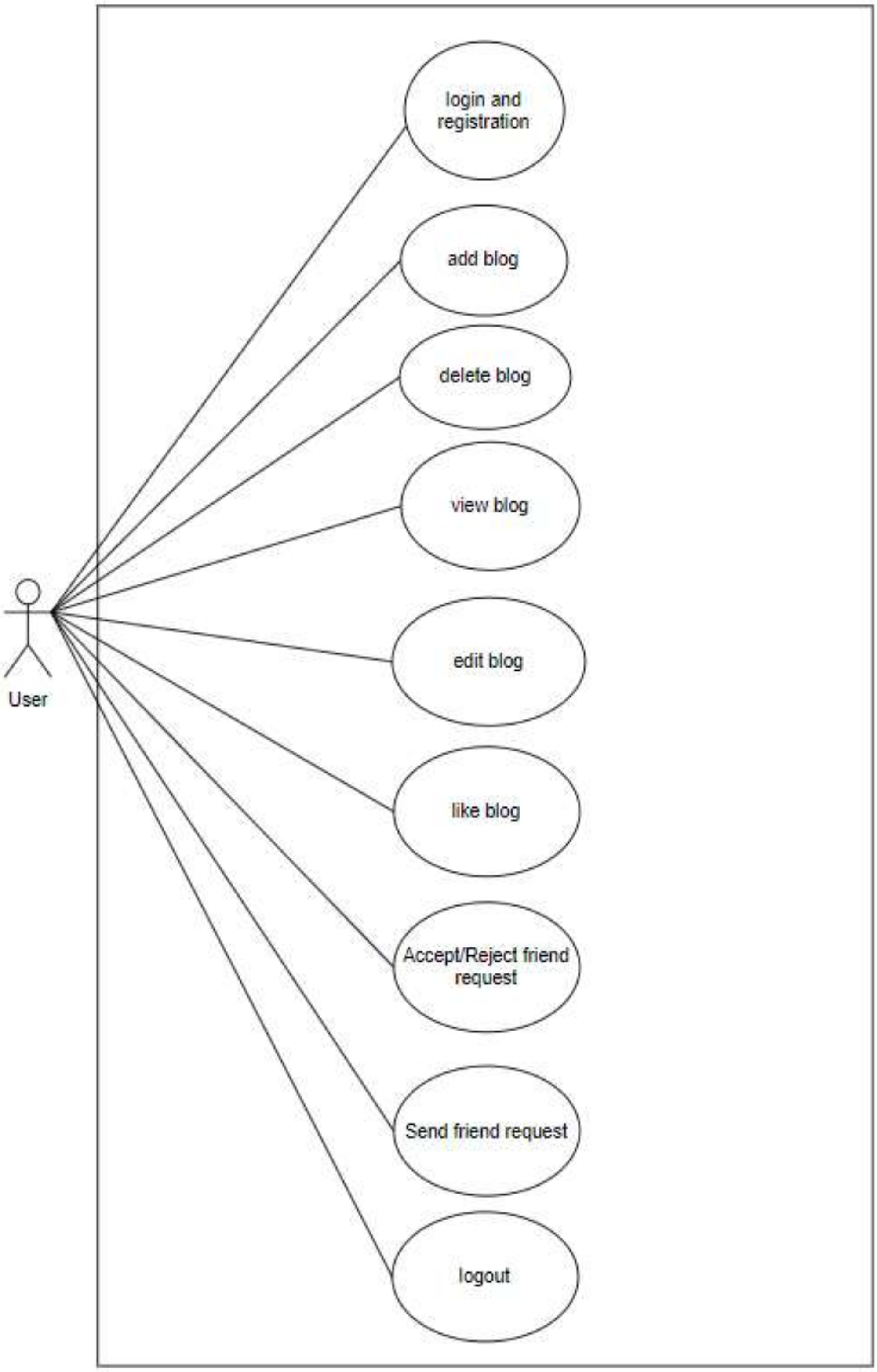


Figure 3.1: Use Case Diagram of User(Blog Site)

The figure 3.1 illustrates the various interactions a user can have with the blog site. Users can log in and register to gain access to the platform. Once logged in, they can add new blog posts, delete existing ones, view and edit their own posts, and like other users' blog posts. The platform also includes social media functionality, allowing users to send and accept or reject friend requests. Finally, users have the option to log out when they are done using the site. These features collectively aim to provide a comprehensive and engaging user experience.

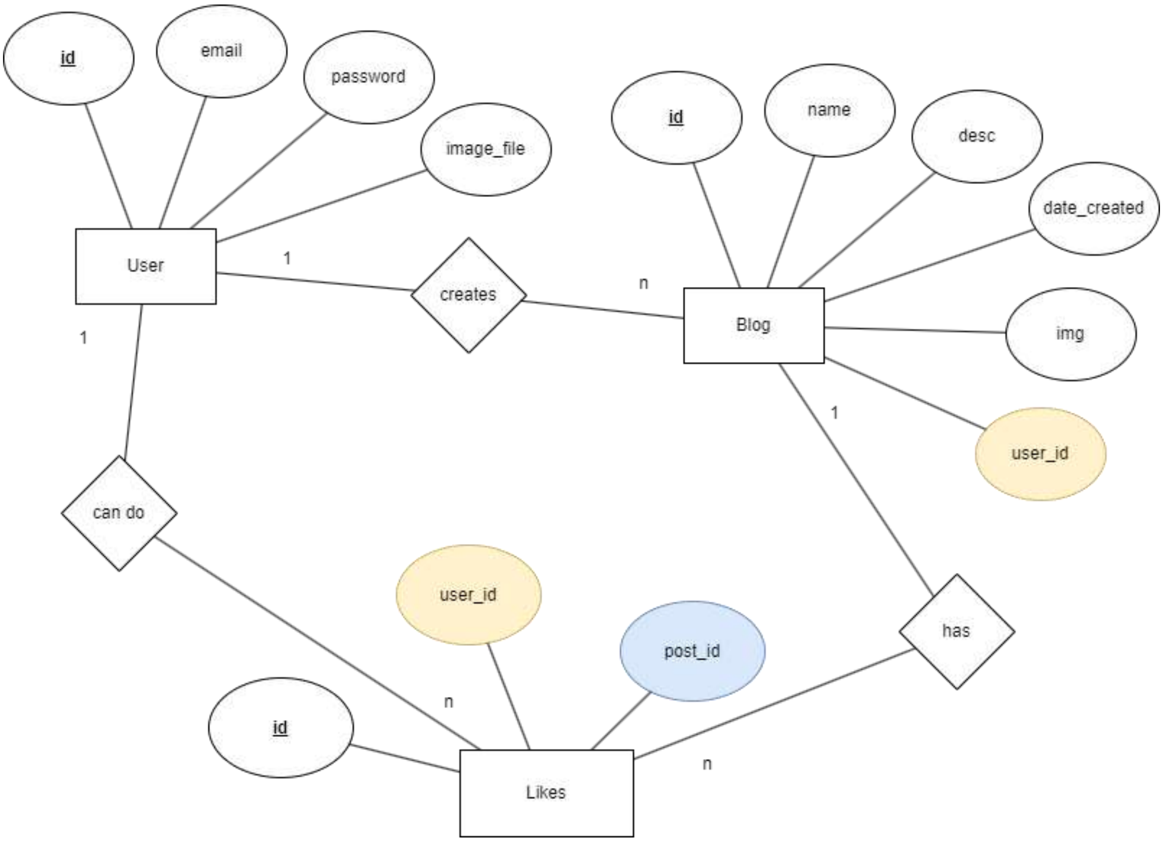


Figure 3.2: Er-Diagram of Blog Site

The figure 3.2 represents a user interaction with a blogging application, illustrating the relationships between the user and various use cases within the system. Users can create blog posts, which include attributes like name, description, date created, image, and user ID. Each blog post is uniquely identified by an ID. Users can perform actions on the blog posts, such as liking them, with each like associated with a user ID and a post ID. This Entity-Relationship (ER) diagram helps in understanding the structure of the database and the interactions between different entities in the blogging system, such as users creating and liking blog posts.

3.3.2 E-commerce Site

The e-commerce site is a robust online shopping platform designed to provide users with an easy and enjoyable shopping experience.

- The project is developed using Flask for the backend, SQLAlchemy for database management, and HTML/CSS for the frontend.
- The website features include browsing products, adding items to the cart, viewing order lists, checking purchase history, managing customer profiles, editing profile details, and an integrated review and rating system.
- The platform also supports secure online payments, ensuring that transactions are safe and reliable.
- The system is designed to be user-friendly and responsive, catering to a wide range of devices and ensuring a smooth user experience.

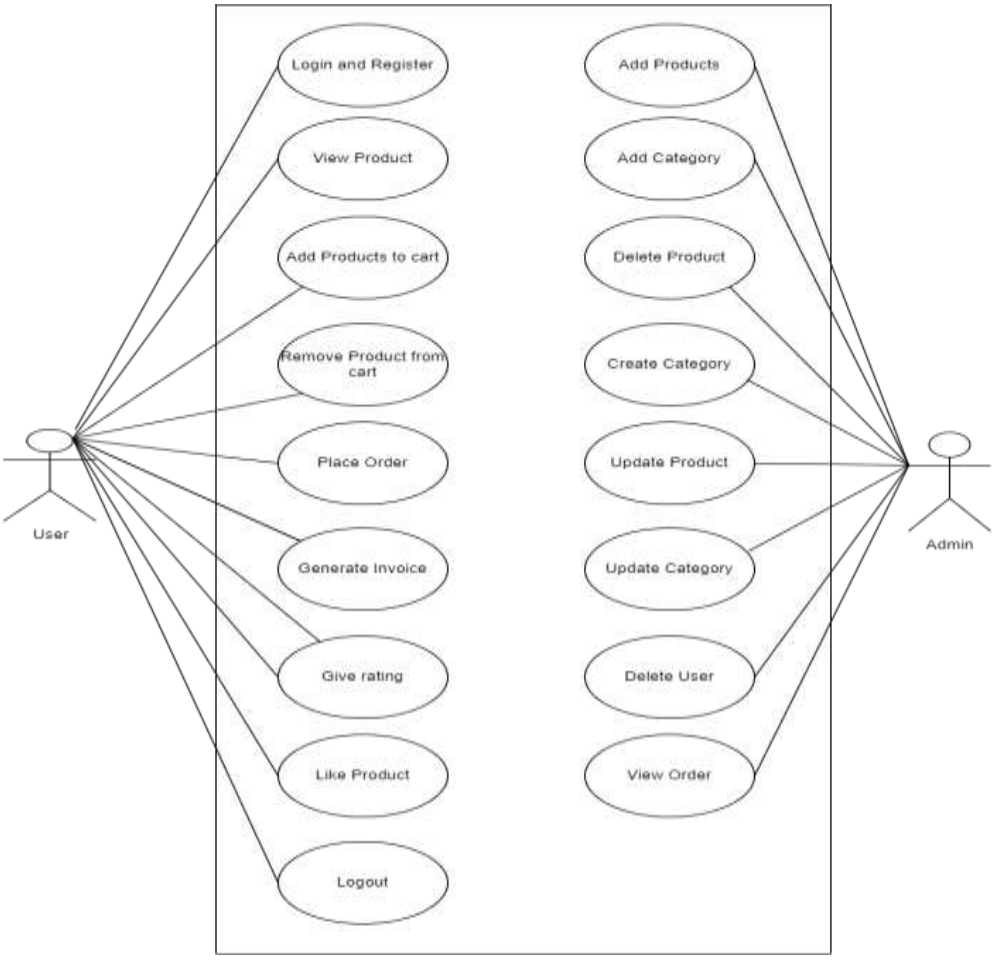


Figure 3.3: Use Case diagram of users (E-Commerce site)

The figure 3.3 outlines the interactions between users and administrators within the e-commerce site. Users can log in and register, view products, add or remove items from their cart, place orders, and generate invoices. They can also give ratings to products, like products, and log out. Administrators have control over the site's inventory and user management. They can add, update, and delete products and categories, delete users, and view orders. This structure ensures that users have a seamless shopping experience, while administrators can efficiently manage the site's operations.

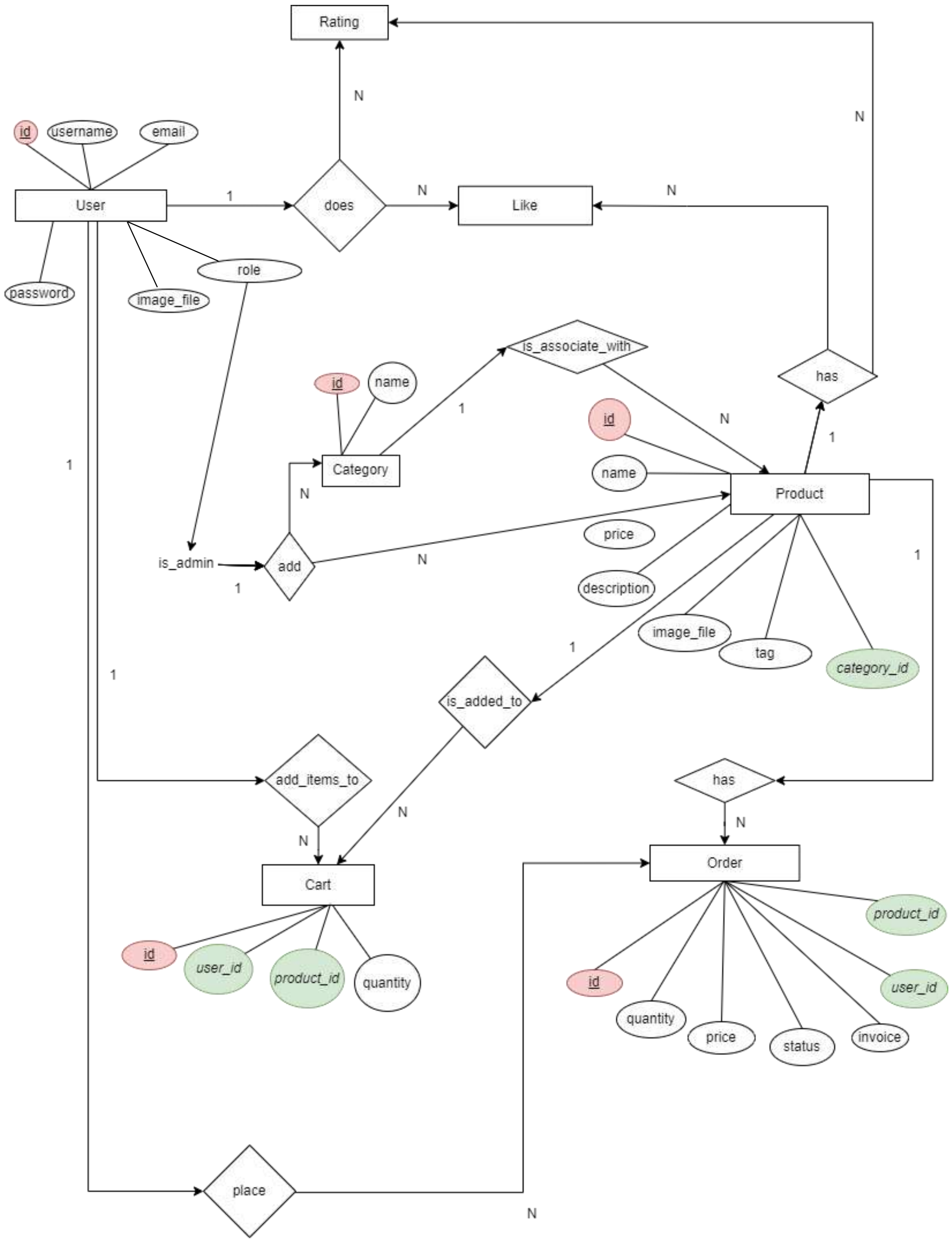


Figure 3.4: Er- Diagram for E-commerce site

The figure 3.4 represents the interactions between users, admins, and various components within an e-commerce application. It depicts the relationships and interactions between entities such as User, Category, Product, Cart, Order, and Rating. Users can perform actions like browsing products, adding items to the cart, and placing orders. They can also like and rate products. Admins have additional capabilities such as managing products by adding, updating, or removing them from categories. The system facilitates the generation of invoices and order tracking for users, with admins having the ability to update the status of orders. This diagram provides a comprehensive overview of the database structure and the interaction flow within the e-commerce system.

3.3.3 Tools Used

i. **Visual Studio Code (VS Code)**

A powerful and versatile code editor used for writing and debugging your project's code. It provides support for Python, JavaScript, and various extensions to enhance productivity.

ii. **Python Flask Framework**

A micro web framework written in Python. It provides the tools and libraries necessary to build your web application, handle routing, manage requests and responses, and integrate with databases.

iii. **JavaScript/Fetch API**

JavaScript is used for client-side scripting to enhance interactivity on the website. The Fetch API is employed for making asynchronous requests to the server, enabling dynamic content updates without refreshing the page.

iv. **GitHub**

A platform for version control and collaboration, allowing you to host your code repositories, track changes, manage versions, and collaborate with others through pull requests and issues.

v. **SQlite**

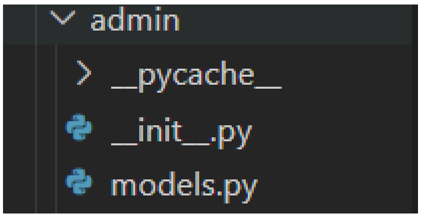
A lightweight, disk-based database used to store and manage the application's data. It's integrated into the Flask app to handle user information, blog posts, and other relevant data efficiently.

3.4 Tasks/ Activities Performed

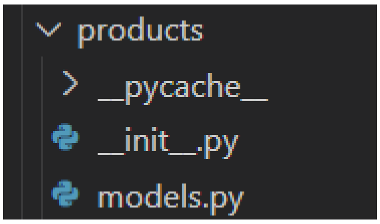
3.4.1 Module Description

The backend system is meticulously designed to ensure that only authorized users have access, thereby maintaining stringent data integrity. Users with appropriate permissions are empowered to manage records securely. The system places a premium on user control, offering a cohesive and user-friendly backend interface for seamless operations. Emphasizing clarity and precision, the backend design enhances overall system usability.

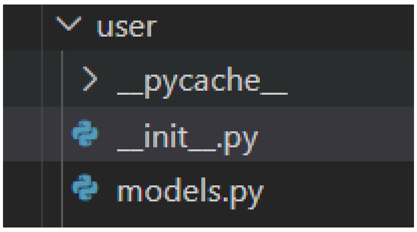
Model: In Flask, the Model represents the data layer and business logic of the application. It interfaces with the SQLite database, enabling operations such as data retrieval, updates, and storage. Models in Flask typically define database tables and relationships, ensuring structured data management.



```
admin
  __pycache__
  __init__.py
  models.py
```

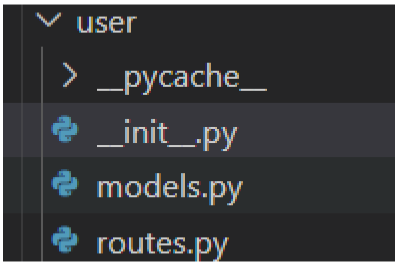


```
products
  __pycache__
  __init__.py
  models.py
```

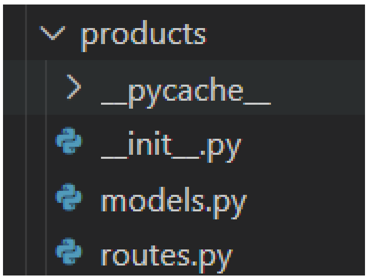


```
user
  __pycache__
  __init__.py
  models.py
```

View and routes: Views in Flask are responsible for rendering HTML templates and presenting data to users. They dictate how information is displayed and ensure a responsive and visually appealing frontend experience. Views receive data from the Controller (if used) or directly from the Model and pass it to the templates for rendering.

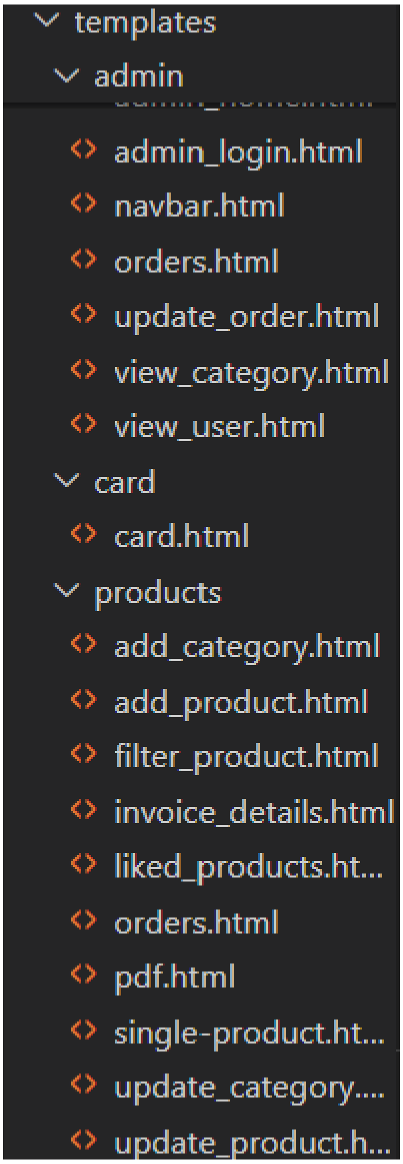


```
user
  __pycache__
  __init__.py
  models.py
  routes.py
```

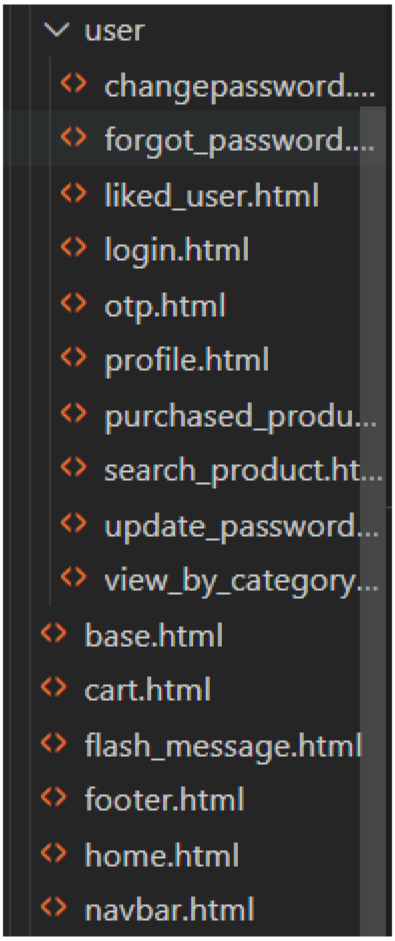


```
products
  __pycache__
  __init__.py
  models.py
  routes.py
```

Template: Templates in Flask (typically using Jinja2 templating engine) are HTML files that include placeholders for dynamic content. They receive data from the View and render it into the final HTML that is sent to the client's browser. Templates facilitate the separation of presentation logic from application logic, promoting maintainability and reusability of code.



```
templates
  admin
    admin_login.html
    navbar.html
    orders.html
    update_order.html
    view_category.html
    view_user.html
  card
    card.html
  products
    add_category.html
    add_product.html
    filter_product.html
    invoice_details.html
    liked_products.ht...
    orders.html
    pdf.html
    single-product.ht...
    update_category....
    update_product.h...
```



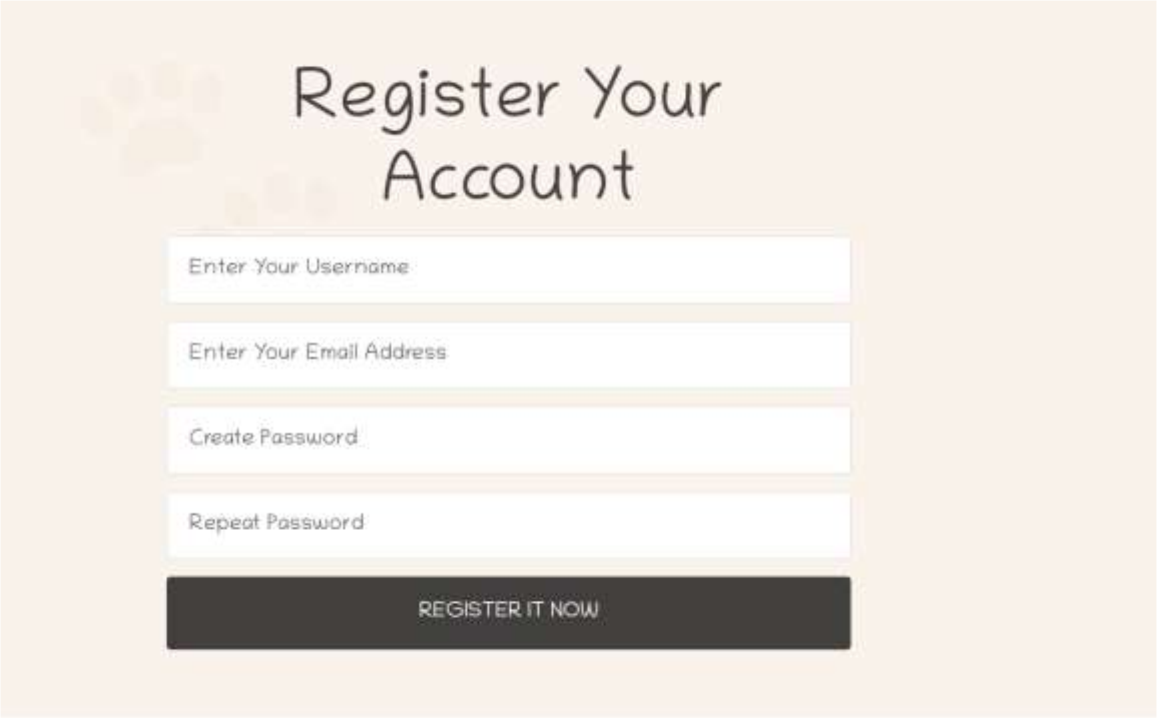
```
user
  changepassword....
  forgot_password....
  liked_user.html
  login.html
  otp.html
  profile.html
  purchased_produ...
  search_product.ht...
  update_password...
  view_by_category...
base.html
cart.html
flash_message.html
footer.html
home.html
navbar.html
```

3.4.2 Creating Sign Up/Login Module(E-commerce site)

The registration module allows new users to create an account by providing their username, email, and password. This module ensures that user information is securely collected, validated, and stored in the database. Successful registration enables users to log in and access personalized features of the website. The login module allows existing users to access their accounts by providing their email and password. It verifies the user's credentials against the stored data and, upon successful authentication, grants access to the system while maintaining session security.

Snippet for Registration:

```
@app.route('/', methods=['GET', 'POST'])
def home():
if current_user.is_authenticated:
        cart = Cart.query.filter_by(user_id=current_user.id).all()
    if request.method == 'POST':
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        re_password = request.form['repassword']
        existing_user = User.query.filter_by(email=email).first()
        if not existing_user:
            if password==re_password:
                hashed_password = bcrypt.generate_password_hash(password)
                new_user = User(username=username, email=email,
password=hashed_password)
                db.session.add(new_user)
                db.session.commit()
                flash('Your account is register succesfully')
```



Register Your Account

Figure 3.5: Registration form

Snippet for login:

```
@app.route("/login", methods=["GET","POST"])
def login():
    if request.method=='POST':
        email = request.form["email"]
        password = request.form["password"]
        user = User.query.filter_by(email=email).first()
        if user:
            if bcrypt.check_password_hash(user.password,password):
                login_user(user)
                next_page = request.args.get('next')
                return redirect(next_page) if next_page else redirect(url_for("home"))
```

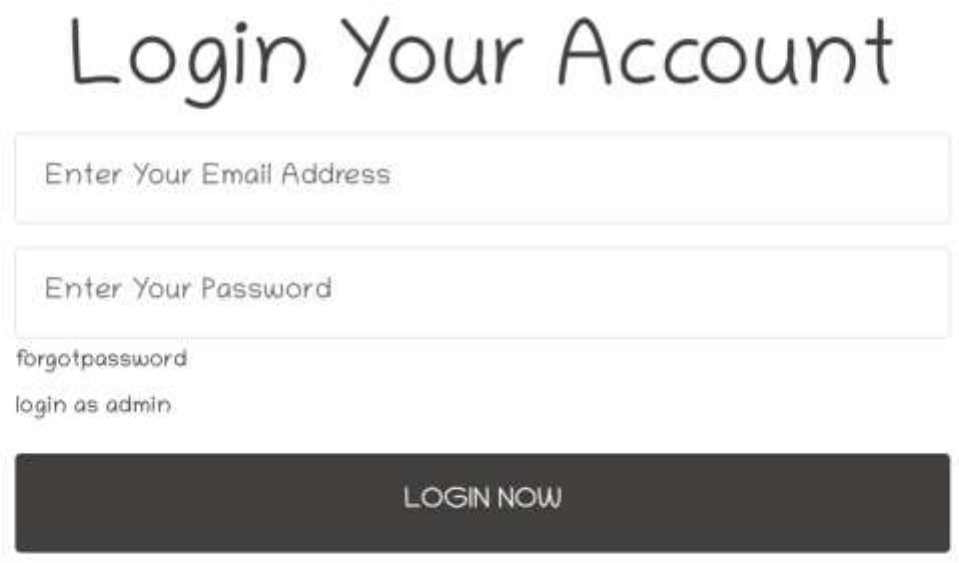


Figure 3.6: Login form

3.4.3 Forgot Password Module

The forgot password module allows users to reset their password securely using an OTP (One-Time Password) sent to their registered email. This module ensures that users can regain access to their accounts if they forget their passwords, while maintaining a high level of security.

Snippet for forgot password:

```
@app.route('/forgot_password', methods=["GET","POST"])
def forgotpassword():
    if request.method == 'POST':
        email = request.form['email']
        user = User.query.filter_by(email=email).first()
        if user:
            session['email'] = email
            otp = ".join(random.sample('0123456789', k=4))
            session['otp'] = otp
            session['otp_timestamp'] = datetime.now(timezone.utc)
            subject = "Forgot Password - OTP Verification"
```

3.4.4 Creating User Module

Different Modules are:

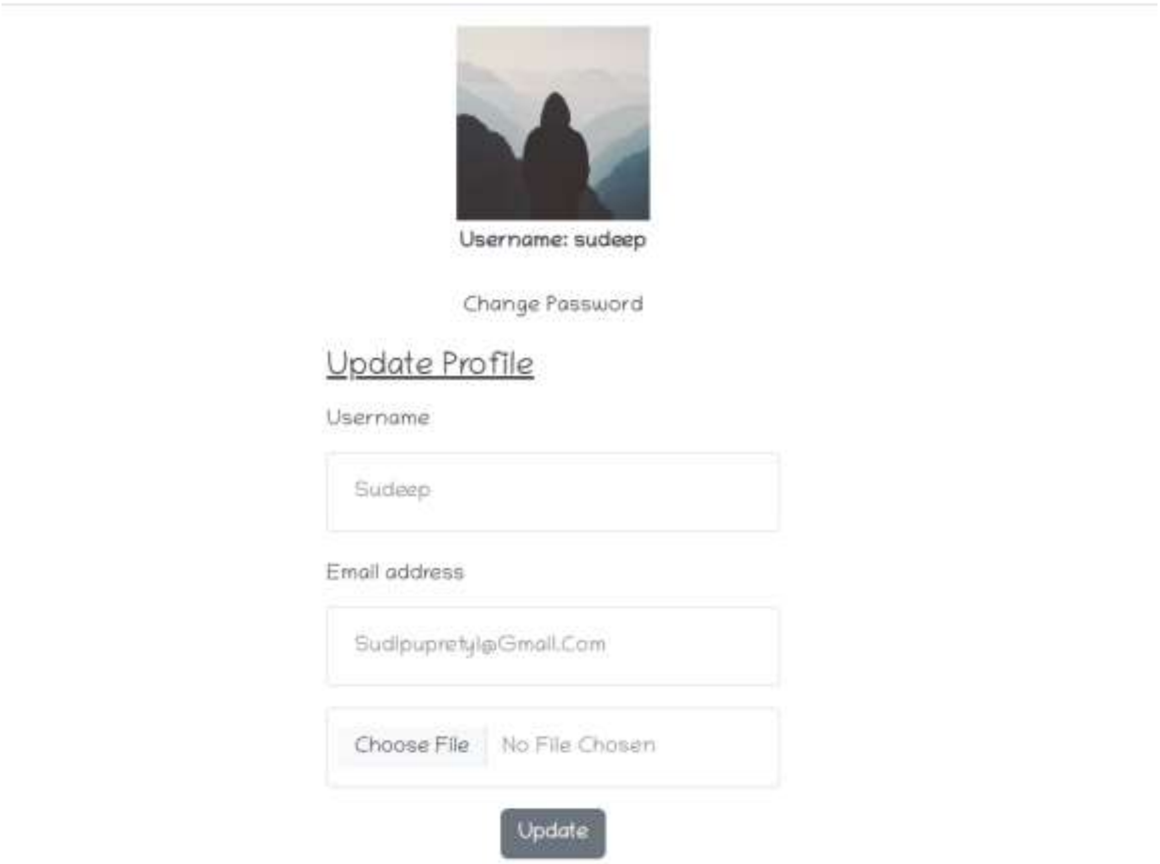
- **Profile Module:**

The Profile module allows users to view and update their personal information.

Users can check their profile details, such as name, email, profile picture. User can update profile details with profile picture and can change password.

Snippet for Profile:

```
@app.route('/profile',methods=["GET", "POST"])
@login_required
def profile():
    if request.method == 'POST':
        current_user.username = request.form['username']
        if 'pic' in request.files:
            pic = request.files['pic']
            if pic.filename != '':
                img_data = pic.read()
                encoded_img = base64.b64encode(img_data).decode('utf-8')
                current_user.image_file = encoded_img
        current_user.email = request.form["email"]
        db.session.commit()
        flash('Your account has been updated!')
```



Username: sudeep

Update Profile

Username

Sudeep

Email address

Sudlpupretyl@Gmall.Com

Figure 3.8: Update Profile form

- **Cart module:**

The shopping cart module allows users to add items to their cart, view the items currently in their cart, and manage their cart contents. This module is essential for enabling users to organize their desired purchases before proceeding to checkout.

Snippet for Cart model:

```
@app.route('/add-to-cart/<int:product_id>', methods=['POST' , 'GET'])
@login_required
def add_to_cart(product_id):
    item_to_add = Product.query.get_or_404(product_id)
    item_exists = Cart.query.filter_by(product_id=product_id,
user_id=current_user.id).first()
```

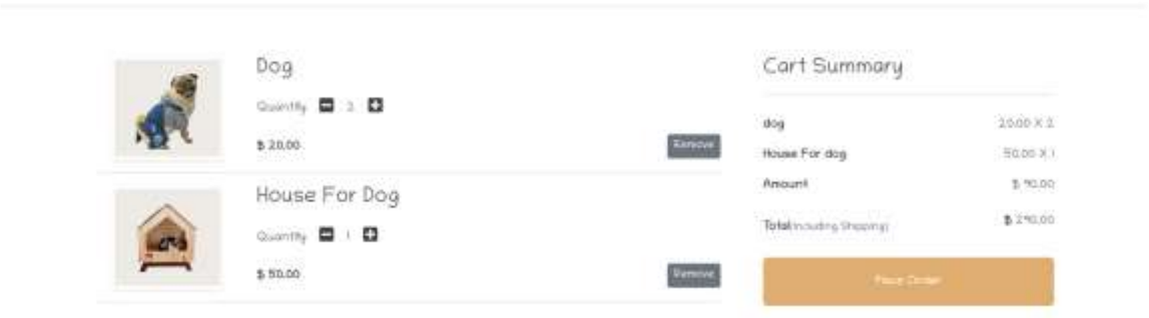


Figure 3.9: Add to Cart/ View cart

- **Rating Module:**

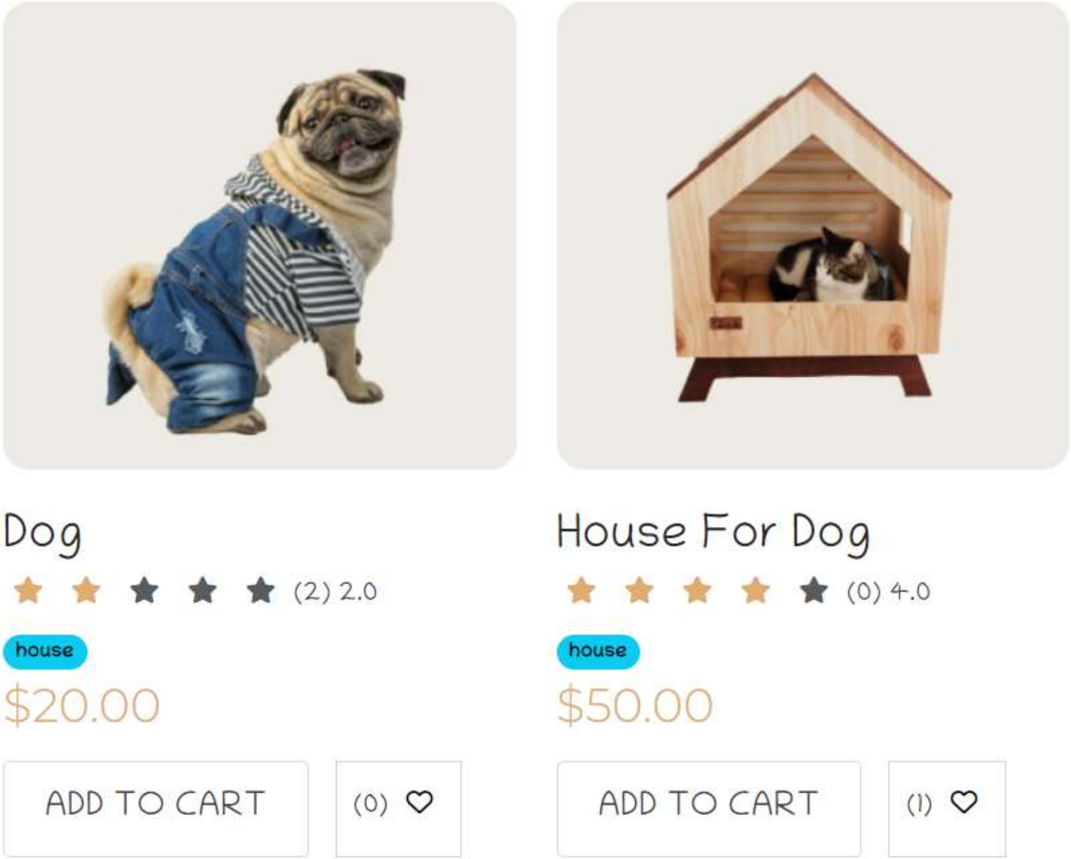
The Rating module enables users to provide reviews and ratings for the website. The value of rating is taken and stored in database to show rating later.

Snippet for rating:

```
@app.route('/rate_product/<int:product_id>/<int:rating_value>', methods=['POST'])
@login_required
def rate_product(product_id, rating_value):
    product = Product.query.get_or_404(product_id)
    purchase = Order.query.filter_by(user_id=current_user.id, product_id=product_id,
status='Delivered').first()

    if not purchase:
        return jsonify({'success': False, 'message': 'You can only rate products you have
purchased.'}), 403

    if rating_value < 1 or rating_value > 5:
        return jsonify({'success': False, 'message': 'Invalid rating value.'}), 400
```



Dog

★★★★★ (2) 2.0

house

$20.00

ADD TO CART

(0) ♡

House For Dog

★★★★★ (0) 4.0

house

$50.00

ADD TO CART

(1) ♡

Figure 3.10: Rating functionality

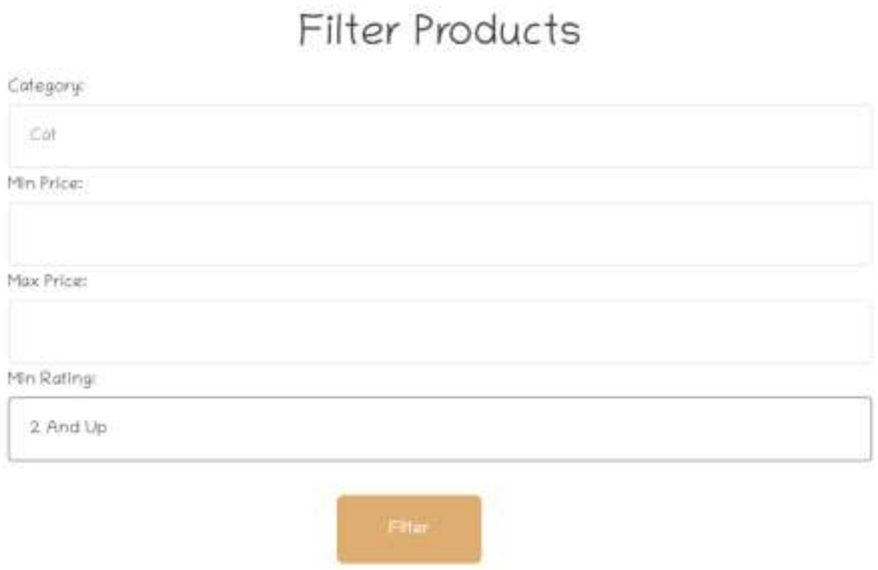
- **Filtering product Module:**

The product filtering module allows users to filter products based on various criteria such as category, rating, and price. This feature enhances the user experience by enabling users to quickly find products that match their preferences and requirements.

Snippet for Filtering product:

```
@app.route('/filter_product', methods=['GET', 'POST'])
def filterproduct():
    categories = Category.query.all()
    filtered_products = Product.query

    if request.method == 'POST':
        category_id = request.form.get('category')
        min_price = request.form.get('min_price')
        max_price = request.form.get('max_price')
```



Filter Products

Category: Cat

Min Price:

Max Price:

Min Rating: 2 And Up

Figure 3.11: Filter Product

- **Online Payment Module:**

The online payment module integrates the Khalti payment gateway to enable secure and seamless payment processing for users. This module allows users to make payments for products or services directly from your Flask-based e-commerce site, enhancing the overall functionality and user experience.

Snippet for Filtering product:

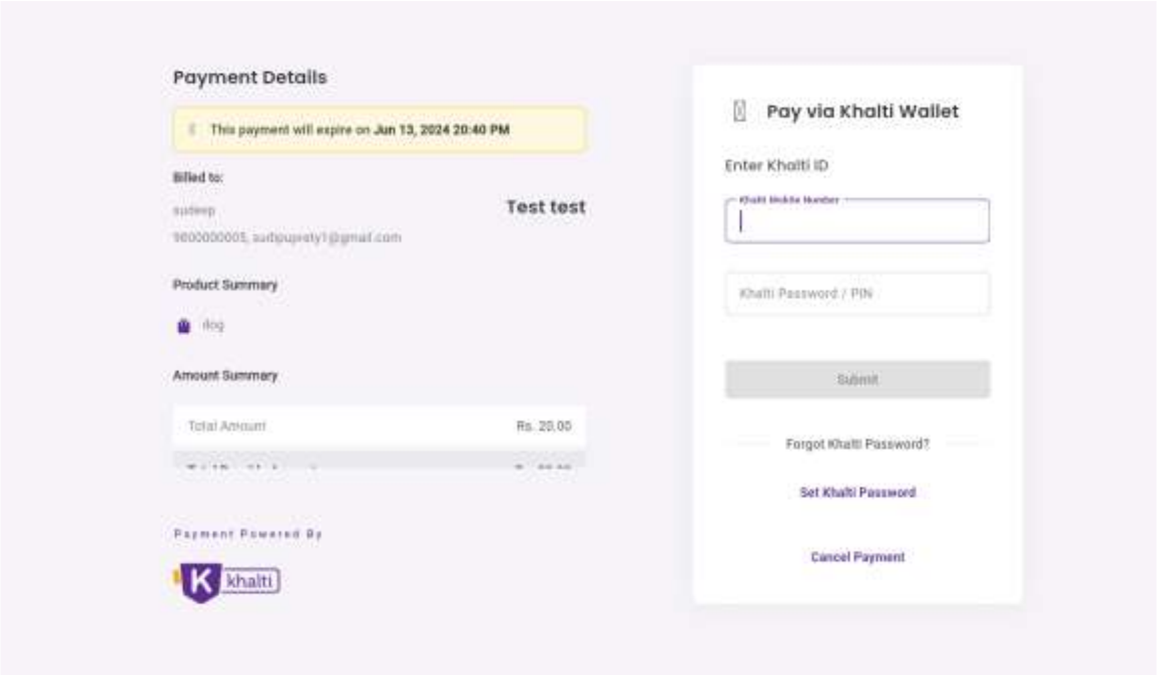
```
@app.route('/initkhalti', methods=['POST'])
@login_required
def initkhalti():
    url = "https://a.khalti.com/api/v2/epayment/initiate/"
    invoice_number = request.form.get('invoice_number')
    orders = Order.query.filter_by(user_id=current_user.id,
invoice=invoice_number).all()

    if not orders:
        return jsonify({'error': 'No orders found for this invoice'}), 400

    amount = sum(order.price for order in orders)
    purchase_order_id = invoice_number
```

```
@app.route('/verify', methods=['GET'])
@login_required
def verify():
    pidx = request.args.get('pidx')
    purchase_order_id = request.args.get('purchase_order_id')

    url = "https://a.khalti.com/api/v2/epayment/lookup/"
    payload = {'pidx': pidx}
    headers = {
        'Authorization': 'Key ******************',
        'Content-Type': 'application/json'
    }
```



Payment Details

This payment will expire on **Jun 13, 2024 20:40 PM**

Billed to:

sudeep

9800000005, sudipuprety1@gmail.com

Test test

Product Summary

blog

Amount Summary

Total Amount Rs. 20.00

Pay via Khalti Wallet

Enter Khalti ID

Khalti Mobile Number

Khalti Password / PIN

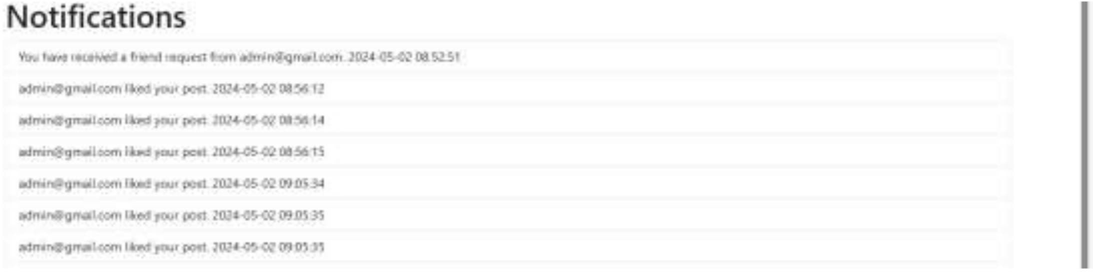
Figure 3.12: Khalti payment gateway

- **Notification module(blog site)**

The notification module allows the system to send various types of notifications to users. This includes alerts for important updates, reminders, or transactional messages such as friend requests.

Snippet for notification:

```
@app.route('/notification/<int:user_id>')
def display_notifications(user_id):
    notifications = get_unread_notifications(user_id)
    return render_template('user/notification.html', notifications=notifications)
```



Notifications

You have received a friend request from admin@gmail.com. 2024-05-02 08:52:51

admin@gmail.com liked your post. 2024-05-02 08:56:12

admin@gmail.com liked your post. 2024-05-02 08:56:14

admin@gmail.com liked your post. 2024-05-02 08:56:15

admin@gmail.com liked your post. 2024-05-02 09:05:34

admin@gmail.com liked your post. 2024-05-02 09:05:35

admin@gmail.com liked your post. 2024-05-02 09:05:35

Figure 3.13: Notification system

3.4.5 Testing

Testing is a crucial aspect of software development to ensure that your Flask-based site functions correctly, meets requirements, and performs reliably under different scenarios. It involves systematically verifying and validating each component of your application to identify bugs, errors, and potential issues early in the development process.

Table 3.2: Unit Testing for Registration (Register Form)

Description	Input	Expected Outcomes	Status
Test if a new user is successfully register	Name: Sudip, Email: sudipuprety1@gmail.com, password: 1111, re-password: 1111	New user is register and stored in user’s table.	Passed
Test registration with only name field	Name: sudip	New user data stored in user’s table	Failed
Test registration with worng re-password	Name: Sudip, Email: sudipuprety1@gmail.com, password: 1111, re-password:	New user data stored in user’s table	Failed

	1112		

Table 3.3: Unit Testing for Login

Description	Input	Expected Outcomes	Status
Test if a user can login with valid credentials	Email: sudipuprety1@gmail.com, password: 1111	User is login to home page	Passed
Test login with an invalid email format.	Email: ram.com	The user is not logged in.	Passed

Table 3.4: Unit testing for Admin login

Description	Input	Expected Outcomes	Status
Test if a Admin can login with valid credentials	Email: admin@gmail.com, password: 1111	Admin is login to admin home page	Passed
Test login with an invalid email format.	Email: adm.com	The admin is not logged in	Passed

Table 3.5: Testing for Making a Payment

Description	Input	Expected Outcomes	Status
Test making payment through test number	Phone number: 9800000001	Payment successful.	Passed
Test making payment through unknown number	Phone number: 98897362382	Payment successful	Failed

Chapter 4. CONCLUSION AND LEARNING OUTCOMES

4.1. Conclusion

In this project, an e-commerce website and a blog site were successfully developed using the Flask framework. The backend development involved designing and implementing a robust and scalable system that supports various functionalities such as user authentication, product management, order processing, notifications, and online payments. The integration of different modules and third-party services ensured a seamless user experience and secure transactions. This project provided a comprehensive understanding of building and deploying full-fledged web applications using Flask, along with best practices in web development, security, and user experience.

4.2. Learning Outcomes

In this project, I successfully developed an e-commerce website and a blog site using the Flask framework. As the backend developer, I was responsible for designing and implementing a robust and scalable backend system that supports various functionalities such as user authentication, product management, order processing, notifications, and online payments. The integration of different modules and third-party services ensured a seamless user experience and secure transactions. This project provided a comprehensive understanding of building and deploying full-fledged web applications using Flask, along with best practices in web development, security, and user experience.

REFERENCES

Smith, A. (2020). *The Future of Blogging: Trends and Technologies*. Retrieved from https://techpress.com/articles/future-of-blogging

Davis, L. (2018). *E-commerce Essentials: Building and Managing Your Online Store*. Retrieved from https://digitalinsights.com/publications/ecommerce-essentials

Brown, T. (2017). *Modular Web Development for E-commerce*. Retrieved from https://codecraft.com/journal/modular-web-development