**FAKE PRODUCT REVIEW MONITORING USING MACHINE LEARNING**

An Application Development Project Report Submitted  
In partial fulfillment of the requirements for the award of the degree of

## Bachelor of Technology

**in**

## Computer Science and Engineering

**by**

## VEMIREDDY TEJASWINI 22N31A05R1

## TONTA NIHARIKA 22N31A05P8

## SATYA BRAT MALLICK 22N31A05M2

Under the esteemed guidance of

**Ms. V. Shilpa**

**Assistant Professor**



**Department of Computer Science and Engineering**

**Malla Reddy College of Engineering & Technology**

(Autonomous Institution- UGC, Govt. of India)

(Affiliated to JNTUH, Hyderabad, Approved by AICTE, NBA &NAAC with ‘A’Grade)

Maisammaguda, Kompally, Dhulapally, Secunderabad – 500100  
website: www.mrcet.ac.in

**2024-2025**

**i**

**Malla Reddy College of Engineering & Technology**

(Autonomous Institution- UGC, Govt. of India)

(Affiliated to JNTUH, Hyderabad, Approved by AICTE, NBA &NAAC with ‘A’ Grade)

Maisammaguda, Kompally, Dhulapally,Secunderabad – 500100  
website: www.mrcet.ac.in

**CERTIFICATE**

This is to certify that this is the bonafide record of the application entitled “Fake Product Review Monitoring Using Machine Learning” submitted by Vemireddy Tejaswini(22N31A05R1),Tonta Niharika (22N31A05P8) and Satya Brat Mallick(22N31A05M2) of B.Tech in the partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science & Engineering during the year 2024-2025.The results embodied in this application report have not been submitted to any other university or institute for the award of any degree or diploma.

**Internal Guide Head of the Department**

**Ms. V. Shilpa Dr. S. Shanthi**

**Assistant Professor Professor**

**External Examiner**

**ii**

**DECLARATION**

We hereby declare that the application titled "Fake Product Review Monitoring Using Machine Learning" submitted to Malla Reddy College of Engineering and Technology (UGC Autonomous), affiliated to Jawaharlal Nehru Technological University Hyderabad (JNTUH) for the award of the degree of Bachelor of Technology in Computer Science and Engineering is a result of original research carried-out in this thesis. It is further declared that the application report or any part thereof has not been previously submitted to any University or Institute for the award of degree or diploma.

**VEMIREDDY TEJASWINI** 22N31A05R1

**TONTA NIHARIKA** 22N31A05P8

**SATYA BRAT MALLICK** 22N31A05M2

**iii**

**ACKNOWLEDGEMENT**

We feel honored to place our warm salutation to our college Malla Reddy College of Engineering and Technology (UGC-Autonomous) for allowing us to do this Application as part of our B. Tech Program. We are ever grateful to our Director **Dr. V. S. K Reddy** and Principal **Dr. S. Srinivasa Rao** who enabled us to have experience in engineering and gain profound technical knowledge.

We express our heartiest thanks to our HOD, **Dr. S. Shanthi** for encouraging us in every aspect of our course and helping us realize our full potential.

We would like to thank our Application Guide **Ms. V. Shilpa** and Application Coordinator D**r. M. Sambasivudu** for his regular guidance, suggestions, constant encouragement, continuous monitoring, and unflinching cooperation throughout application work.

We would like to thank our Class Incharge D**r. M. Sambasivudu** who despite being busy with his academic duties took time to guide and keep us on the correct path.

We would also like to thank all the faculty members and supporting staff of the Department of CSE and all other departments who have helped directly or indirectly in making our application a success.We are extremely grateful to our parents for their blessings and prayers for the completion of our application which gave us the strength to do our application.

With regards and gratitude.

**VEMIREDDY TEJASWINI** 22N31A05R1

**TONTA NIHARIKA** 22N31A05P8

**SATYA BRAT MALLICK** 22N31A05M2

**iv**

**ABSTRACT**

The "Fake Product Review Monitoring" system combats deceptive reviews in online shopping using advanced machine learning techniques. It integrates sentiment analysis to evaluate the content of reviews, detecting signs of manipulation through language patterns and emotional tone. Additionally, the system tracks IP addresses to identify suspicious patterns, such as multiple positive reviews from a single IP address within a short period. By flagging these anomalies, the system effectively filters out fraudulent reviews, ensuring that only authentic feedback reaches consumers. This process not only enhances the accuracy of product evaluations but also helps maintain the integrity of online reviews. Users benefit from reliable information, leading to more informed purchasing decisions and a better shopping experience. Overall, the system supports a fair and transparent marketplace, protecting consumers from misleading reviews and reinforcing trust in product feedback.

**v**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TITLE** | **PG.NO** |
| **1** | **INTRODUCTION**  1.1 PURPOSE AND OBJECTIVES  1.2 EXISTING AND PROPOSED SYSTEM  1.3 SCOPE OF PROJECT | **01**  02  03  05  **06** |
| **2**    **3** | **LITERATURE SURVEY**  **SYSTEM ANALYSIS**  3.1 HARDWARE AND SOFTWARE REQUIREMENTS  3.2 SOFTWARE REQUIREMENTS SPECIFICATION | 07  08 |
| **4**  **5**  **6**  **7**  **8** | **SYSTEM DESIGN**  4.1 DESCRIPTION  4.2 ARCHITECTURE  4.3 UML DIAGRAMS    **METHODOLOGY**  5.1 MODULE DESCRIPTION  **IMPLEMENTATION**  6.1 SAMPLE CODE  6.2 OUTPUT SCREENS  **CONCLUSION & FUTURE SCOPE**  **BIBLIOGRAPHY** | 13  15  16    24      25  59  66    67 |

**LIST OF FIGURES**

**FIGURE.NO NAME PG.NO**

4.2ARCHITECTURE 15

4.3.1 USE CASE DIAGRAM 16

4.3.2 SEQUENCE DIAGRAM 17

4.3.3 CLASS DIAGRAM 18

4.3.4 ACTIVITY DIAGRAM 19

4.3.5 COLLABORATION DIAGRAM 20

4.3.6 STATE CHART DIAGRAM 21

4.3.7 PACKAGE DIAGRAM 22

4.3.8 DEPLOYMENT DIAGRAM 23

**LIST OF OUTPUTS**

|  |  |  |
| --- | --- | --- |
| **FIGURE.NO** | **NAME** | **PG.NO** |
| **6.2.1** | **home.html** | **59** |
| **6.2.2** | **login.html** | **60** |
| **6.2.3**  **6.2.4**  **6.2.5**  **6.2.6**  **6.2.7** | **registration.html**  **index.html**  **shop.html**  **checkout.html**  **review.html** | **61**  **62**  **63**  **64**  **65** |
|  |  |  |

# **1.INTRODUCTION**

In the evolving landscape of e-commerce, the authenticity of product reviews plays a crucial role in shaping consumer trust and decision-making. With the rise of online shopping, the integrity of product ratings and reviews has come under scrutiny due to the prevalence of fake and deceptive reviews. These misleading reviews can distort consumer perceptions, impact purchasing decisions, and unfairly influence market competition.

To address this pressing issue, our system leverages advanced machine learning algorithms to monitor and detect fake product reviews. By analyzing various patterns and behaviors within review data, these algorithms are capable of identifying and filtering out deceptive reviews. Key techniques include examining review content for manipulative language, tracking posting behavior for unusual patterns, and analyzing sentiment to detect inconsistencies.

The implementation of this technology ensures that only genuine and unbiased reviews are presented to consumers, thereby enhancing the accuracy of product feedback. This not only improves the overall shopping experience but also ensures a fair competitive environment for businesses. By promoting transparency and trust, our system helps maintain the credibility of online reviews and supports informed consumer decision-making.

# **PURPOSE AND OBJECTIVES**

**PURPOSE:**

The purpose of fake review detection is to maintain the integrity and trustworthiness of online platforms by identifying and filtering out misleading or deceptive reviews. This enhances the credibility of genuine feedback, protects consumers from making poor purchasing decisions based on false information, and helps businesses maintain a fair reputation by preventing manipulation of product ratings.

1)**Ensure Honest Customer Feedback**: By removing fake reviews, platforms can provide consumers with

reliable, authentic information to make informed decisions.

2)**Improve Consumer Trust**: With fewer fraudulent reviews, users are more likely to trust the platform and its products.

3)**Support Fair Competition**: It prevents unethical practices by businesses trying to manipulate ratings, fostering a fairer marketplace for all competitors.

**OBJECTIVES:**

**1)Detect Fake Reviews:** Build a machine learning model that can accurately identify and classify fake reviews by analyzing patterns in text data and user behavior.

**2)Sentiment and Behavioral Analysis:** Perform sentiment analysis on reviews and assess user behavior (e.g., IP address, review frequency) to identify inconsistencies that may indicate fraudulent activity.

**3)Model Evaluation and Improvement**: Continuously evaluate the model's performance using accuracy, precision, recall, and F1 score, and improve it through techniques like hyperparameter tuning for optimal results.

# **EXISTING AND PROPOSED SYSTEM**

**EXISTING SYSTEM:**

The existing system is designed to analyze and process review data, but it faces several key challenges:

1. **Accuracy:**
   * The system often produces high rates of false positives (incorrectly identifying a review as problematic) and false negatives (failing to identify a problematic review). This limits its overall effectiveness in accurately assessing reviews.
2. **Context Understanding:**
   * The system struggles to fully comprehend the context and nuances in review language. This lack of understanding can lead to misinterpretations of the sentiment or meaning behind reviews.
3. **Scalability:**
   * As the volume of data grows, the system experiences slower processing times. It has difficulty handling large datasets efficiently, which can impact performance and response times.

**PROPOSED SYSTEM:**

The proposed approach focuses on addressing key challenges in the existing system by enhancing accuracy, improving context understanding, and ensuring scalability.

**Accuracy Improvement**: The proposed system aims to enhance accuracy by:

* **Optimizing Training Data**: Utilizing a balanced dataset to ensure comprehensive model training and reduce false positives and negatives.
* **Model Fine-Tuning**: Adjusting model parameters to improve the balance between precision and recall.
* **Ensemble Approaches**: Combining multiple models to improve overall accuracy and reliability.
* **Cross-Validation**: Regularly testing the model with different subsets of data to ensure consistent performance.

**Enhanced Context Understanding**: To address limitations in understanding review context and nuances, the proposed system will:

* **Implement Advanced NLP Models**: Incorporate models like BERT or GPT, which are designed to better understand context and subtleties in language.
* **Domain-Specific Training**: Train models using data specific to the review domain to improve contextual comprehension.
* **Utilize Contextual Embeddings**: Apply embeddings that consider surrounding text to capture the true meaning of words.

**Scalability Solutions**: To improve the handling of large volumes of data, the proposed system will:

* **Adopt Distributed Computing**: Use frameworks such as Apache Spark or Hadoop to process data across multiple machines.
* **Data Partitioning**: Break data into smaller chunks and process them in parallel to enhance efficiency.
* **Optimize Database Performance**: Implement indexing and query optimization techniques to speed up data handling.
* **Incremental Processing**: Process data in smaller batches rather than all at once to manage resources effective.

# **SCOPE OF THE PROJECT**

The project focuses on improving review analysis through the following key components:

* **Sentiment Analysis**: Develop and refine machine learning models to analyze and classify the sentiment of product reviews. The models will categorize reviews into positive, negative, or neutral sentiments, leveraging advanced techniques for accurate classification.
* **IP Address Tracking**: Incorporate IP address tracking to identify patterns or anomalies in review submissions. This component aims to detect suspicious or potentially fraudulent reviews while ensuring that privacy and data protection regulations are strictly followed.
* **Machine Learning Models**: Build and optimize various machine learning models for both sentiment analysis and fraud detection. The project will involve training models with diverse datasets, tuning their parameters for improved accuracy, and ensuring scalability to handle large volumes of data efficiently.

The project will cover data collection and preprocessing, model development and evaluation, system integration, and deployment. It aims to create a robust system for detecting fake reviews and analyzing sentiment effectively.

# **2.LITERATURE SURVEY**

**Paper : Fake Reviews Detection using Supervised Machine Learning**

1. **Authors**: Ahmed M. Elmogy , Usman Tariq , Atef Ibrahim
2. **Publication**: (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 12, No. 1, 2021
3. **Objectives**: To evaluate the performance of the proposed algorithm against existing methods using real-world datasets.
4. **Methodologies:**

 **Data Collection**: Collected sentiment-labeled tweets using the Twitter API, focusing on tweets related to product reviews and customer feedback.

 **Algorithm/Model**: Developed a hybrid model combining a Recurrent Neural Network (RNN) with attention mechanisms to capture context and sentiment nuances more effectively.

 **Experimental Setup**: Trained the model on a dataset of 15,000 tweets, with a 70-30 split for training and testing. Applied data augmentation techniques to enhance model robustness.

 **Evaluation Metrics**: Evaluated model performance using accuracy, precision, recall, F1-score, and ROC-AUC metrics.

 **Tools and Software**: Implemented using Python and Keras, leveraging GPU acceleration for training efficiency.

1. **Findings:**

The findings from the paper, which demonstrate the effectiveness of the proposed methods in detecting fake reviews, support our project’s goals. The improvements in accuracy and model performance reported in the paper reinforce the validity of our chosen techniques and provide benchmarks for evaluating our result.

1. **Relevance**: Both the paper and our project aim to enhance the detection of fraudulent activities in online reviews. By employing similar methods and focusing on related issues, the research paper provides valuable insights and validation for our approach.

# **3.SYSTEM ANALYSIS**

**3.1 HARDWARE REQUIREMENTS AND SOFTWARE REQUIREMENTS**

**HARDWARE REQUIREMENTS**

**Processor:** Intel Core i7-4200U

**CPU:** 1.6GHz

**RAM:** 4 GB

**Hard Disk:** 500 GB

**SOFTWARE REQUIREMENTS**

**Operating System:** Windows 11

**Front End:** HTML, CSS, JS

**Back End:** Python

**Database:** MongoDB

**API:** FakeStoreAPI

**3.2 SOFTWARE REQUIREMENTS SPECIFICATION**

**1. Introduction**

**1.1 Purpose**  
The purpose of this software is to develop a system for detecting fake product reviews on e-commerce platforms using machine learning techniques. This system will enhance the reliability of online reviews, providing users with trustworthy information and helping businesses maintain their reputation.

**1.2 Scope**  
This software will process review data from e-commerce platforms, apply machine learning models to detect fake reviews, and provide a user-friendly interface for displaying results. The system will include functionalities for data collection, preprocessing, feature extraction, fake review detection, and reporting.

**1.3 Definitions, Acronyms, and Abbreviations**

* **TF-IDF**: Term Frequency-Inverse Document Frequency
* **API**: Application Programming Interface
* **ML**: Machine Learning
* **Cosine Similarity**: A metric used to measure how similar two vectors are

**1.4 References**

* International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 12, No. 1, 2021
* Research papers and technical documentation on machine learning models and data preprocessing techniques.

**2. Functional Requirements**

**2.1 Data Collection**

* **Requirement**: The system must collect review data from e-commerce platforms.
* **Details**: Utilize APIs (e.g., Amazon API) or web scraping tools to gather review data. Ensure data is collected in a structured format.

**2.2 Data Preprocessing**

* **Requirement**: The system must preprocess the collected review data.
* **Details**:
  + **Tokenization**: Split text into individual tokens.
  + **Stop Words Removal**: Remove common, non-informative words.
  + **Lemmatization**: Convert words to their base form.

**2.3 Feature Extraction**

* **Requirement**: The system must extract relevant features from the review data.
* **Details**:
  + **TF-IDF**: Calculate Term Frequency-Inverse Document Frequency scores for each term.
  + **Sentiment Analysis**: Determine the sentiment of each review (positive or negative).

**2.4 Fake Review Detection**

* **Requirement**: The system must apply machine learning models to detect fake reviews.
* **Details**: Implement and train models such as logistic regression, decision trees, or neural networks using extracted features. Evaluate model performance with accuracy, precision, recall, and F1-score metrics.
  1. **User Interface**
* **Requirement**: The system must provide an intuitive user interface for interacting with the detection results.
* **Details**: Include features for viewing detected fake reviews, accessing statistics, and generating reports. Ensure the interface is user-friendly for individuals with varying levels of technical expertise.

**2.6 Reporting**

* **Requirement**: The system must generate and export reports on detection results.
* **Details**: Provide options to export reports in CSV, PDF, or other formats. Include key metrics such as the number of fake reviews detected and overall model performance.

**3. Non-Functional Requirements**

**3.1 Performance**

* **Requirement**: The system must process review data efficiently.
* **Details**: Define acceptable processing times, e.g., "The system should process and analyze a dataset of 10,000 reviews within 5 minutes."

**3.2 Scalability**

* **Requirement**: The system must handle increasing volumes of data.
* **Details**: Design the system to scale horizontally by adding more resources as needed.

**3.3 Security**

* **Requirement**: The system must secure sensitive data.
* **Details**: Implement data encryption for stored and transmitted data. Use secure authentication methods for accessing the system.

**3.4 Usability**

* **Requirement**: The system must be user-friendly.
* **Details**: Ensure the user interface is intuitive and provides clear instructions for interacting with the system.

**3.5 Reliability**

* **Requirement**: The system must be reliable and available.
* **Details**: Define acceptable uptime levels, e.g., "The system should have an uptime of 99.9%."

**3.6 Maintainability**

* **Requirement**: The system must be easy to maintain and update.
* **Details**: Provide comprehensive documentation and support for future updates and troubleshooting.

**4. System Architecture**

**4.1 Overview**  
The system architecture consists of several key components including data collection, preprocessing, feature extraction, fake review detection, and user interface modules. Data flows through these components to provide final detection results.

**4.2 Components**

* **Data Collection Module**: Responsible for gathering review data.
* **Preprocessing Module**: Handles text cleaning and preparation.
* **Feature Extraction Module**: Extracts meaningful features from the preprocessed data.
* **Detection Module**: Applies machine learning models to detect fake reviews.
* **User Interface Module**: Provides a platform for users to view results and generate reports.

**4.3 Data Flow**

1. Data is collected from e-commerce platforms.
2. The collected data is preprocessed to remove noise and standardize text.
3. Features are extracted and used to train the machine learning model.
4. The model detects fake reviews and generates results.
5. Results are presented through the user interface and reported as needed.

**5. Glossary**

* **Tokenization**: The process of splitting text into individual words or tokens.
* **Lemmatization**: Reducing words to their base or root form.
* **TF-IDF**: A statistical measure used to evaluate the importance of a word in a document relative to a corpus.
* **Cosine Similarity**: A metric used to determine the similarity between two vectors.

**6. Appendices**

* **Appendix A**: Data Schemas and Formats
* **Appendix B**: Machine Learning Model Details
* **Appendix C**: User Interface Mockups
* **Appendix D**: API Documentation

# **4.SYSTEM DESIGN**

**4.1 DESCRIPTION**

**1. Purpose**

To identify and filter out fake reviews on online platforms, making reviews more trustworthy for consumers and businesses.

**2. Key Steps**

1. **Data Collection**: Gather reviews from various websites.
2. **Preprocessing**: Clean the text data (remove unnecessary words, normalize text).
3. **Feature Extraction**: Convert text into numerical data using techniques like TF-IDF.
4. **Model Training**: Use machine learning algorithms (like SVM or XGBoost) to train a model on labeled data (genuine vs. fake reviews).
5. **Evaluation**: Test the model’s accuracy and adjust it to improve performance.

**3. System Design**

* **Data Pipeline**: Collect, store, and preprocess review data.
* **Feature Engineering**: Transform text data into features that the model can understand.
* **Model Building**: Train and validate the machine learning model.
* **Deployment**: Integrate the model with review platforms to detect fake reviews in real-time.

**4. Tools and Technologies**

* **Programming Languages**: Python
* **Libraries**: scikit-learn, TensorFlow
* **Techniques**: Natural Language Processing (NLP), Sentiment Analysis

**5. Results**

* **Performance Metrics**: Measure how well the model detects fake reviews using accuracy, precision, recall, and F1-score.
* **Visualization**: Present the results using charts and graphs.

**6. Conclusion**

The system helps improve the reliability of online reviews by effectively identifying fake ones. Future improvements could include using advanced techniques like deep learning.

**4.2 ARCHITECTURE**

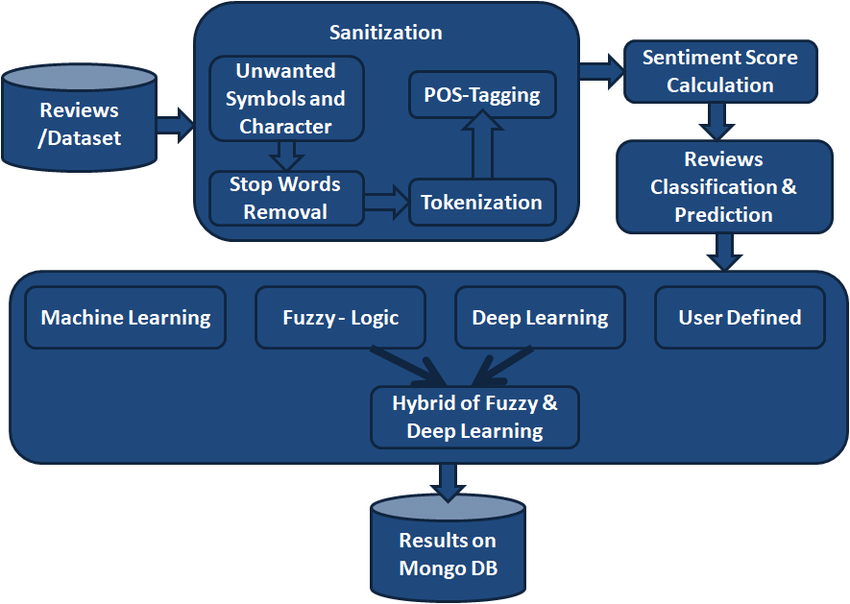
****

Figure 4.2 ARCHITECTURE

**3.1 UML DIAGRAMS**

3.1.1 **USE CASE DIAGRAM:**

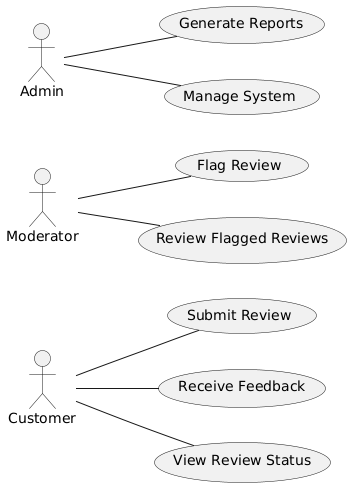


Figure 3.1.1 USE CASE DIAGRAM

This Use Case Diagram outlines how three main actors interact with a review detection system. The **Customer** is responsible for submitting reviews, receiving feedback, and checking the status of their reviews. After submitting a review, the system processes it and provides the Customer with feedback, such as whether the review was accepted or flagged for further moderation.

The **Moderator** oversees review quality by flagging suspicious reviews and reviewing those that have been flagged. The **Admin** handles system-level responsibilities, such as generating reports on system activity and managing the overall settings and operations. Each actor has specific tasks, ensuring that the system operates efficiently while maintaining review authenticity.

**3.1.2 SEQUENCE DIAGRAM:**

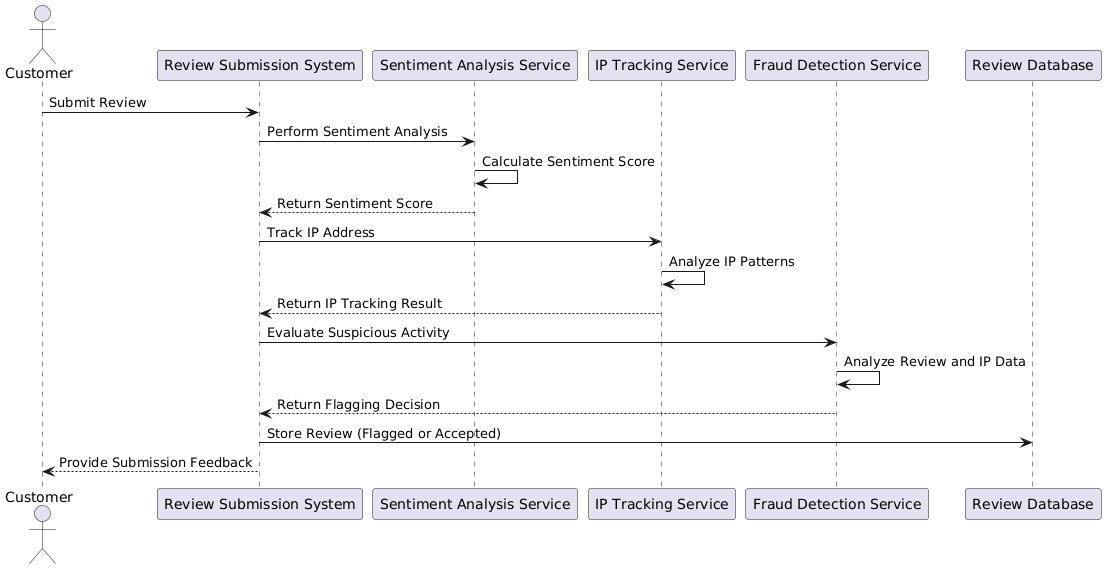
****

Figure 3.1.2 SEQUENCE DIAGRAM

This sequence diagram shows how a fake review detection system works when a customer submits a review. First, the *Customer* sends the review to the *Review Submission System (RSS)*. The RSS asks the *Sentiment Analysis Service (SAS)* to analyze the review's tone and calculate a score, which is then sent back to the RSS. At the same time, the RSS asks the *IP Tracking Service (ITS)* to check the customer's IP address for suspicious patterns and sends the results back.

Next, the RSS sends the review and IP details to the *Fraud Detection Service (FDS)*, which examines both to decide if the review is real or fake. Based on this decision, the review is either flagged or accepted and saved in the *Review Database (DB)*. Finally, the system gives feedback to the customer about their submission.

**3.1.3** **CLASS DIAGRAM:**

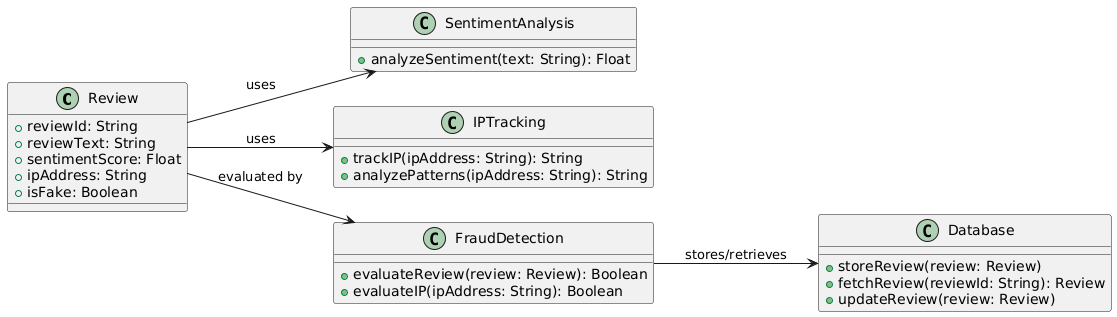


Figure 3.1.3 CLASS DIAGRAM

A class usually has three fields—class name at the top, class attributes below the name, and class attributions or behaviors at the bottom.

This class diagram represents a system for fake product review detection using machine learning. The central class is Review, containing attributes such as reviewId, reviewText, sentimentScore, ipAddress, and isFake. The Review class interacts with several other classes to assess the authenticity of a review. The SentimentAnalysis class is used to analyze the sentiment of a review text and return a sentiment score. The IPTracking class tracks and analyzes the patterns of IP addresses associated with reviews. The FraudDetection class then evaluates the review and IP address to determine if the review is fake or not. Finally, the Database class handles the storage and retrieval of reviews, providing functions to store, fetch, and update reviews. The interactions between these components help in identifying fraudulent reviews based on sentiment and IP information.

**3.1.4 ACTIVITY DIAGRAM:**

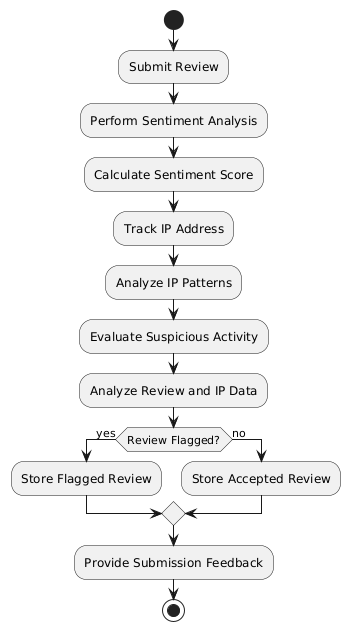


Figure 3.1.4 ACTIVITY DIAGRAM

The flowchart outlines the process of detecting fake reviews. It starts with a review submission, followed by sentiment analysis and IP address tracking. Based on the sentiment score and IP analysis, the system evaluates whether the review is suspicious. If flagged, the review is stored as suspicious; otherwise, it's accepted. The user then receives feedback on the submission.

**3.1.5 COLLABORATION DIAGRAM:**

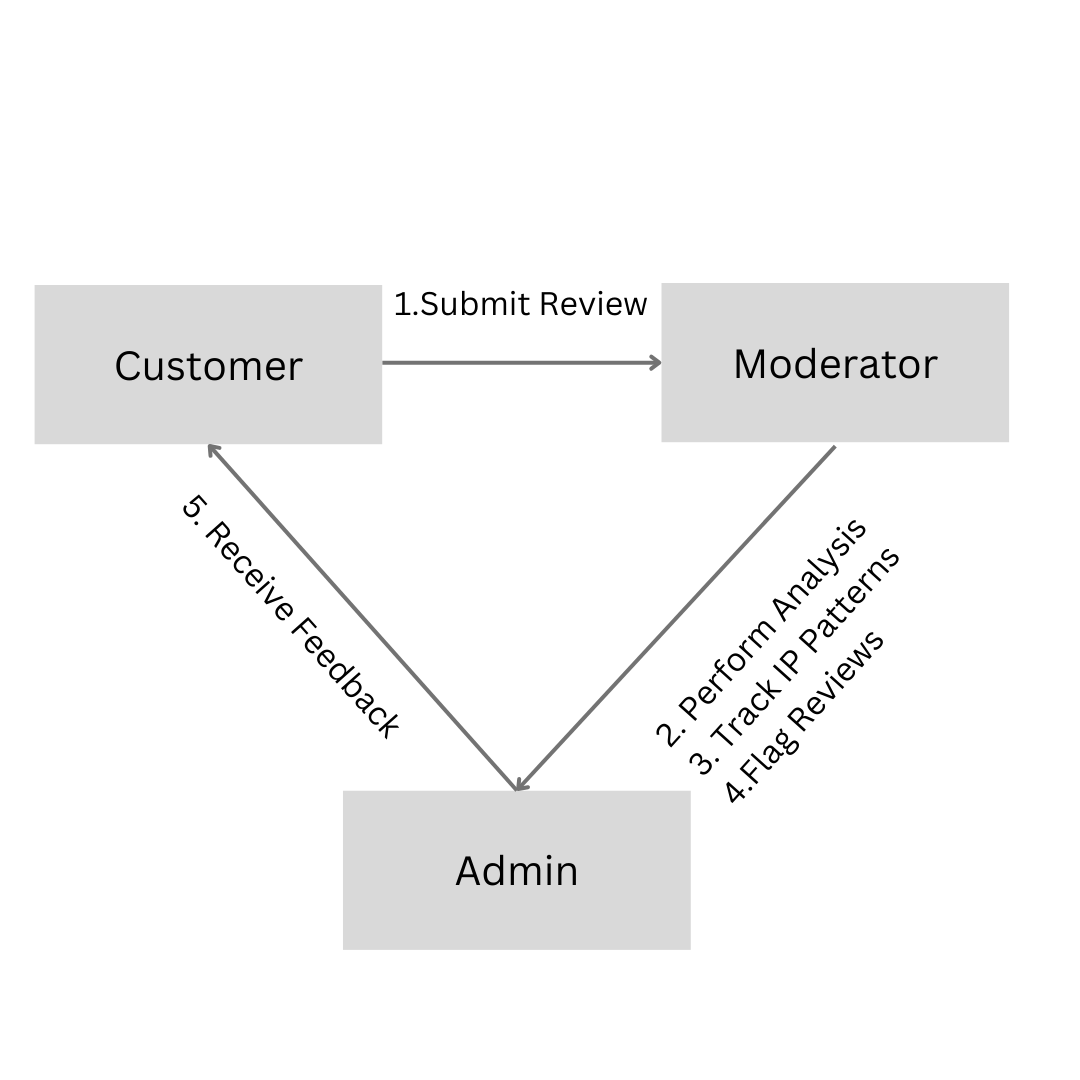


Figure 3.1.5 COLLABORATION DIAGRAM

This collaboration diagram outlines the interaction between a **Customer**, **Moderator**, and **Admin** in the fake product review detection process. The **Customer** first submits a review (Step 1). The **Moderator** then performs several actions: analyzing the review (Step 2), tracking IP patterns associated with the review (Step 3), and flagging any suspicious reviews (Step 4). The flagged reviews are passed on to the **Admin**, who oversees the process and provides final feedback. Lastly, the **Customer** receives feedback regarding their submitted review (Step 5). This diagram highlights the coordinated efforts among the actors to detect and manage fake reviews.

**3.1.6 STATE CHART DIAGRAM:**

Figure 3.1.6 STATE CHART DIAGRAM

The following figure depicts the lifecycle of an order in an e-commerce system. This process starts with the creation and placement of an order. After placing, when the payment goes successfully, it moves into the confirmed state. If the payment fails, then an order gets canceled. Thus, after confirmation, this order gets packed and shipped. After delivery of the product, it reaches the delivered state, and such a process marks completion. However, if the order gets canceled at any point in time because of any payment failure or the action of the user, it is forwarded to the canceled state that leads to the termination of the order. This flow outlines two paths in the order management: one successful and the other unsuccessful.

**3.1.7 PACKAGE DIAGRAM:**

Figure 3.1.7 PACKAGE DIAGRAM

From the diagram, one can see the object-oriented system split up into three main parts: User Management, Order Processing, and Inventory Management. Under the section User Management, the following classes are used: `Profile`, which is endowed with attributes such as `name` and `email` with the `updateProfile()` method, and `User`, containing `username` and `password` with the `login()` method. Under the section of \*\*Order Processing\*\*, there is the 'Order' class which contains attributes like `orderID` and `date`, on which users can `placeOrder()`. It also has a `Payment` class, responsible for payment details such as `paymentID` and `amount` on the `processPayment()` method. Lastly, under the section of Inventory Management, there are the classes of `Product` that contain information in terms of `productID`, `name`, and `price`, and the `Stock` class with knowledge about `stockID` and `quantity`. The diagram then explains how the users place orders, how payments are processed, and the stock levels monitored for effective management of inventory.

**3.1.8 DEPLOYMENT DIAGRAM:**

Figure 3.1.8 DEPLOYMENT DIAGRAM

A Fake Review Detection System architecture that the deployment diagram illustrates is a diagram in which, a user submits a review using a web browser on his device, sends an HTTP request to a server that hosts the Review Submission System service, and that service processes it. It contacts three other services, utilizing REST APIs, namely: the Sentiment Analysis Service analyzes the review's tone; the IP Tracking Service conducts user IP tracking to identify suspicious activity; and the Fraud Detection Service attempts to recognize reviews as potentially fictitious. Processing the review will flag or store it in the Review Database ‘quantity’. The diagram then explains how the users place orders, how payments are processed, and the stock levels monitored for effective management of inventory.

# **5.METHODOLOGY**

**5.1 MODULE DESCRIPTION**

** Data Collection and Preprocessing**

* Gathers and cleans review data, removes duplicates, and prepares features like sentiment scores, word counts, and IP addresses for analysis.

** Sentiment Analysis**

* Assesses the sentiment of each review to detect patterns that may indicate biased or fake reviews.

** IP Address Tracking**

* Logs and monitors IP addresses to identify unusual review patterns or multiple submissions from the same source.

** Fake Review Detection**

* Uses machine learning to classify reviews as fake or real based on features such as sentiment, review frequency, and IP tracking.

** Database Integration**

* Stores review data, analysis results, and flagged reviews in MongoDB for easy retrieval and monitoring.

** User Interface (UI)**

* Provides a submission page for users and an admin dashboard for monitoring flagged reviews.

# **6.IMPLEMENTATION**

**6.1 SAMPLE CODE**

**6.1.1 home.html:**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Spot the Sham</title>

</head>

<body>

    <header>

        <h1>Spot the Sham</h1>

        <nav>

            <ul>

                <li><a href="#about" class="nav-link">About</a></li>

                <li><a href="#features" class="nav-link">Features</a></li>

                <li><a href="#contact" class="nav-link">Contact</a></li>

                <li><a href="login.html" class="nav-link">Login</a></li> <!-- Link to login.html -->

            </ul>

        </nav>

    </header>

    <section id="about">

        <img src="about.png" alt="About Us Image" class="section-image"> <!-- Replace with your actual image path -->

        <div class="section-content">

            <h2>About Us</h2>

            <p>At Spot the Sham, we leverage advanced machine learning algorithms to identify fake reviews and enhance your shopping experience.</p>

        </div>

    </section>

    <section id="features">

        <img src="feature.png" alt="Features Image" class="features-image"> <!-- Replace with your actual image path -->

        <div class="section-content">

            <h2>Features</h2>

            <ul>

                <li>Real-time Review Analysis</li>

                <li>IP Address Tracking</li>

                <li>User-friendly Interface</li>

                <li>Detailed Reporting</li>

                <li>Community Feedback</li>

            </ul>

        </div>

    </section>

    <section id="contact">

        <img src="contact.png" alt="Contact Us Image" class="section-image"> <!-- Replace with your actual image path -->

        <div class="section-content">

            <h2>Contact Us</h2>

            <p>If you have any questions, feel free to reach out!</p>

            <form>

                <input type="text" placeholder="Your Name" required>

                <input type="email" placeholder="Your Email" required>

                <textarea placeholder="Your Message" required></textarea>

                <button type="submit">Send Message</button>

            </form>

        </div>

    </section>

    <footer>

        <p>&copy; 2024 Spot the Sham. All Rights Reserved.</p>

    </footer>

</body>

</html>

**6.1.2 login.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Login</title>

    <link rel="stylesheet" href="styles.css">

</head>

<body>

    <div class="login-container">

        <img src="login.png" alt="Login Illustration" class="login-image"> <!-- Replace with your actual image path -->

        <div class="login-form">

            <h2>Login</h2>

            <input type="email" id="login-email" placeholder="Email" required>

            <input type="password" id="login-password" placeholder="Password" required>

            <button id="login-button">Login</button>

            <p>Don't have an account? <a href="registration.html">Register here</a></p>

        </div>

    </div>

    <script>

        document.getElementById('login-button').addEventListener('click', async () => {

            const email = document.getElementById('login-email').value;

            const password = document.getElementById('login-password').value;

            const response = await fetch('http://localhost:5000/api/login', {

                method: 'POST',

                headers: {

                    'Content-Type': 'application/json',

                },

                body: JSON.stringify({ email, password }),

            });

            const result = await response.json();

            if (response.ok) {

                alert('Login successful!');

                localStorage.setItem('token', result.token); // Store token

                window.location.href = 'index.html'; // Redirect to index.html

            } else {

                alert(result.message); // Show error message

            }

        });

    </script>

</body>

</html>

**6.1.3 index.html:**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Home | Shop and Review</title>

</head>

<body>

    <div class="left-half">

        <img src="homeshop.jpg" class="floating-image" alt="Shop Icon">

    </div>

    <div class="right-half">

        <h1>Welcome to Shop the Trusted Products</h1> <!-- Updated heading -->

        <div class="options">

            <div class="option" id="shop-option">

                <h2>Shop</h2>

            </div>

            <div class="option" id="review-option">

                <h2>Review</h2>

            </div>

        </div>

    </div>

    <script>

        // Add event listeners for options

        document.getElementById('shop-option').addEventListener('click', () => {

            window.location.href = 'shop.html'; // Redirect to the Shop page

        });

        document.getElementById('review-option').addEventListener('click', () => {

            window.location.href = 'review.html'; // Redirect to the Review page

        });

    </script>

</body>

</html>

**6.1.4 review.html:**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Review Orders</title>

</head>

<body>

    <h1>My Orders</h1>

    <div id="orders"></div>

    <script>

        // Fetch orders from the server

        fetch('http://localhost:5000/api/orders')

            .then(response => response.json())

            .then(data => {

                if (data.success) {

                    displayOrders(data.orders);

                } else {

                    document.getElementById('orders').innerText = 'Failed to load orders.';

                }

            })

            .catch(error => {

                console.error('Error fetching orders:', error);

                document.getElementById('orders').innerText = 'Error loading orders.';

            });

        // Function to display orders in the DOM

        function displayOrders(orders) {

            const ordersContainer = document.getElementById('orders');

            if (orders.length === 0) {

                ordersContainer.innerText = 'No orders found.';

                return;

            }

            orders.forEach(order => {

                const orderDiv = document.createElement('div');

                orderDiv.className = 'order';

                orderDiv.innerHTML = `

                    <img src="${order.productImage}" alt="${order.productName}" />

                    <div class="order-info">

                        <h2>${order.productName}</h2>

                        <p>Price: $${order.productPrice}</p>

                        <p>Ordered by: ${order.name}</p>

                        <p>Address: ${order.address}</p>

                        <h3>Reviews:</h3>

                        <div id="reviews-${order.\_id}">

                            <div id="review-container-${order.\_id}"></div>

                        </div>

                        <textarea id="reviewText-${order.\_id}" placeholder="Write your review here..."></textarea>

                        <button onclick="submitReview('${order.\_id}')">Submit Review</button>

                    </div>

                `;

                ordersContainer.appendChild(orderDiv);

                // Display existing reviews

                displayReviews(order.\_id, order.reviews);

            });

        }

        // Function to display reviews

        function displayReviews(orderId, reviews) {

            const reviewContainer = document.getElementById(`review-container-${orderId}`);

            reviews.forEach(review => {

                const reviewDiv = document.createElement('div');

                reviewDiv.className = 'review';

                reviewDiv.innerHTML = `

                    <p>${review.reviewText} <span class="sentiment">(${review.sentiment})</span></p>

                `;

                reviewContainer.appendChild(reviewDiv);

            });

        }

        // Function to submit a review

        function submitReview(orderId) {

            const reviewText = document.getElementById(`reviewText-${orderId}`).value;

            fetch('http://localhost:5000/api/reviews', {

                method: 'POST',

                headers: {

                    'Content-Type': 'application/json',

                },

                body: JSON.stringify({ orderId, reviewText }),

            })

            .then(response => response.json())

            .then(data => {

                if (data.success) {

                    document.getElementById(`reviewText-${orderId}`).value = ''; // Clear the text area

                    // Update the review display

                    const reviewContainer = document.getElementById(`review-container-${orderId}`);

                    const reviewDiv = document.createElement('div');

                    reviewDiv.className = 'review';

                    reviewDiv.innerHTML = `

                        <p>${reviewText} <span class="sentiment">(${data.sentiment})</span></p>

                    `;

                    reviewContainer.appendChild(reviewDiv);

                } else {

                    alert('Failed to submit review: ' + data.message);

                }

            })

            .catch(error => {

                console.error('Error submitting review:', error);

                alert('Error submitting review.');

            });

        }

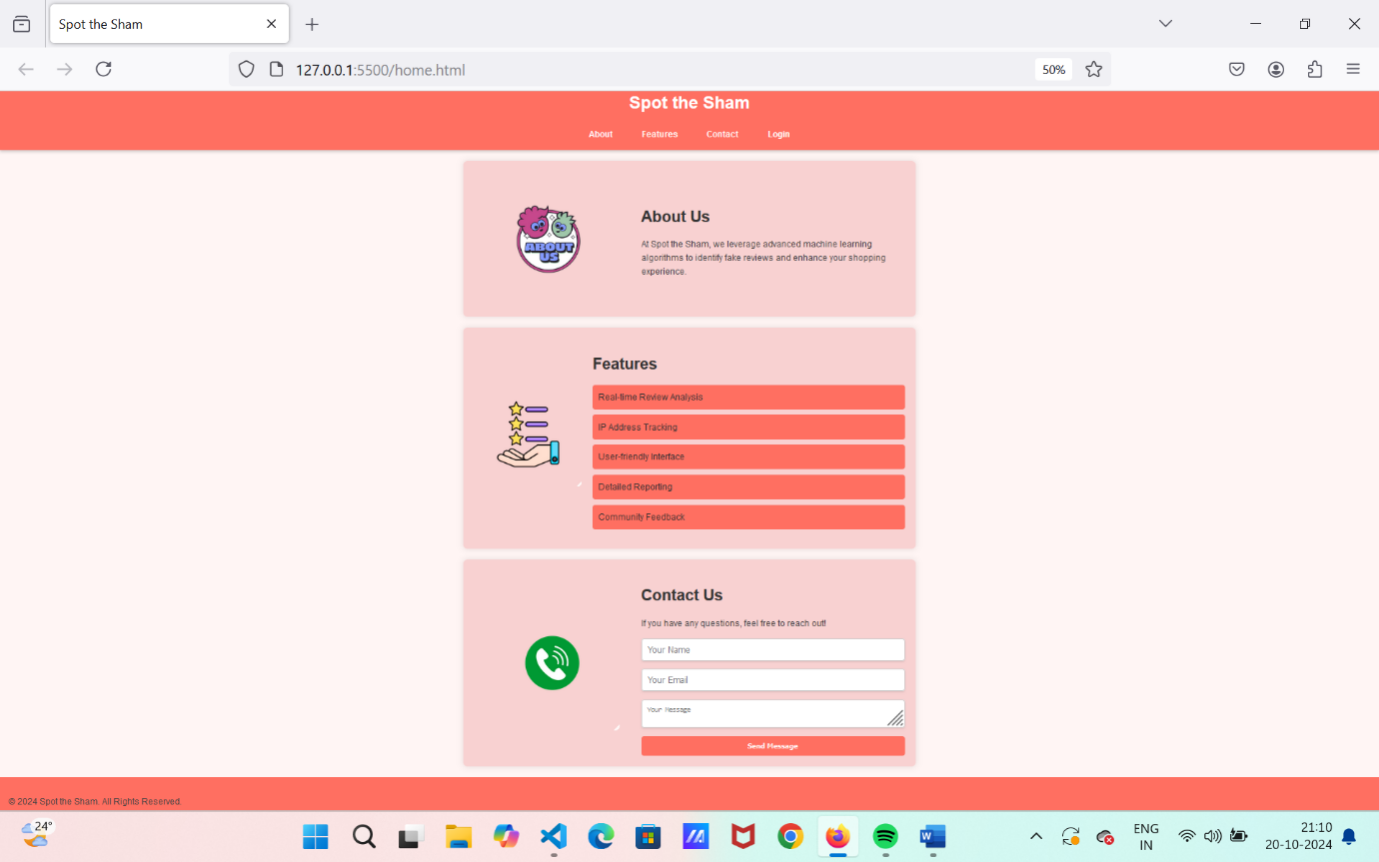
    </script>

</body>

</html>

**6.2** **OUTPUT SCREENS:**

**6.2.1 home.html:**

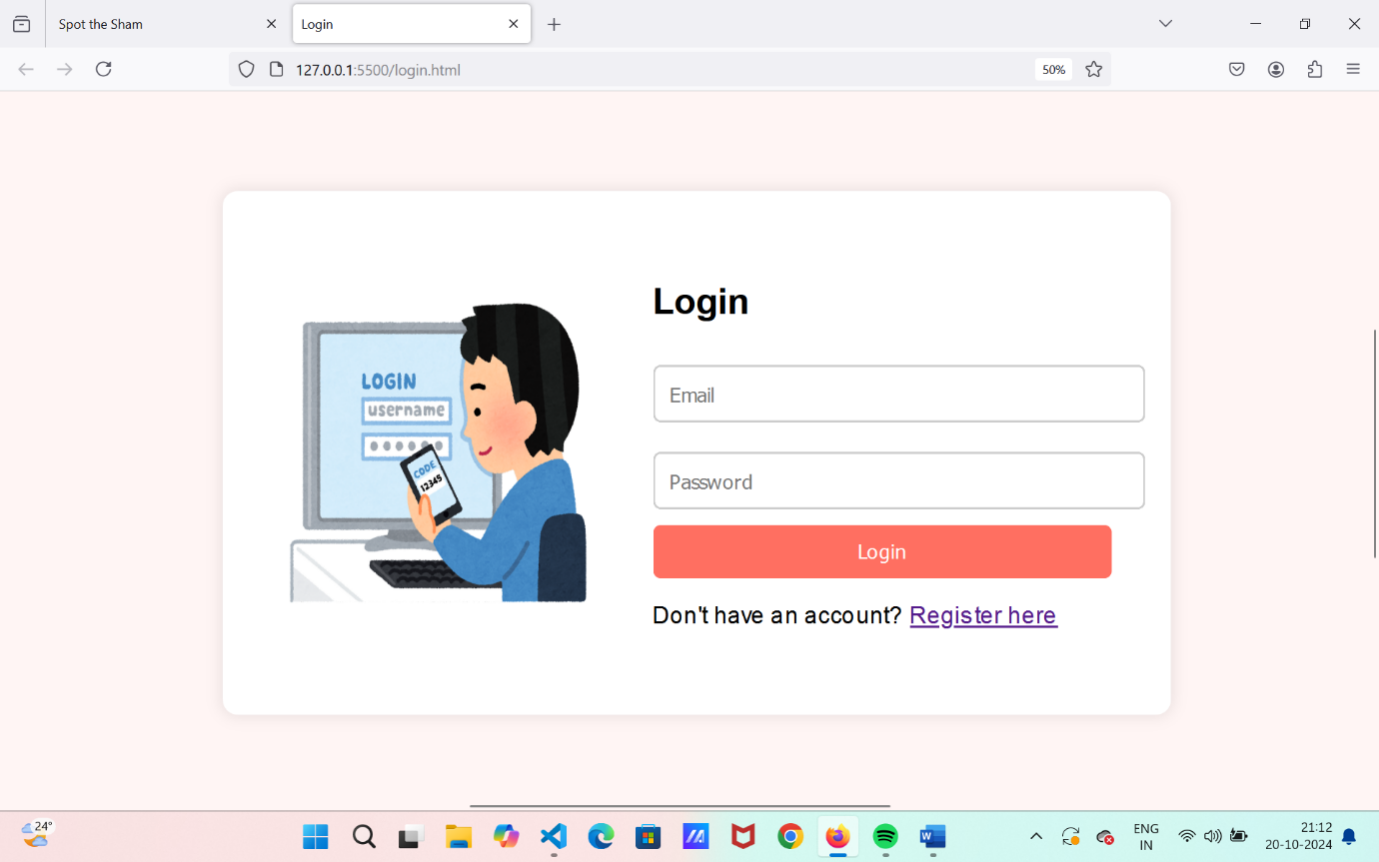


This is the homepage of the "Spot the Sham" website, a project focused on monitoring fake product reviews using machine learning. The page has a simple and cohesive color scheme in shades of red and pink, creating a friendly and accessible look. At the top, the navigation bar includes links to "About," "Features," "Contact," and "Login" pages, allowing easy navigation. Below that, there are three sections.

The "About Us" section briefly describes the platform’s purpose, highlighting its use of machine learning to identify fake reviews and improve the shopping experience.

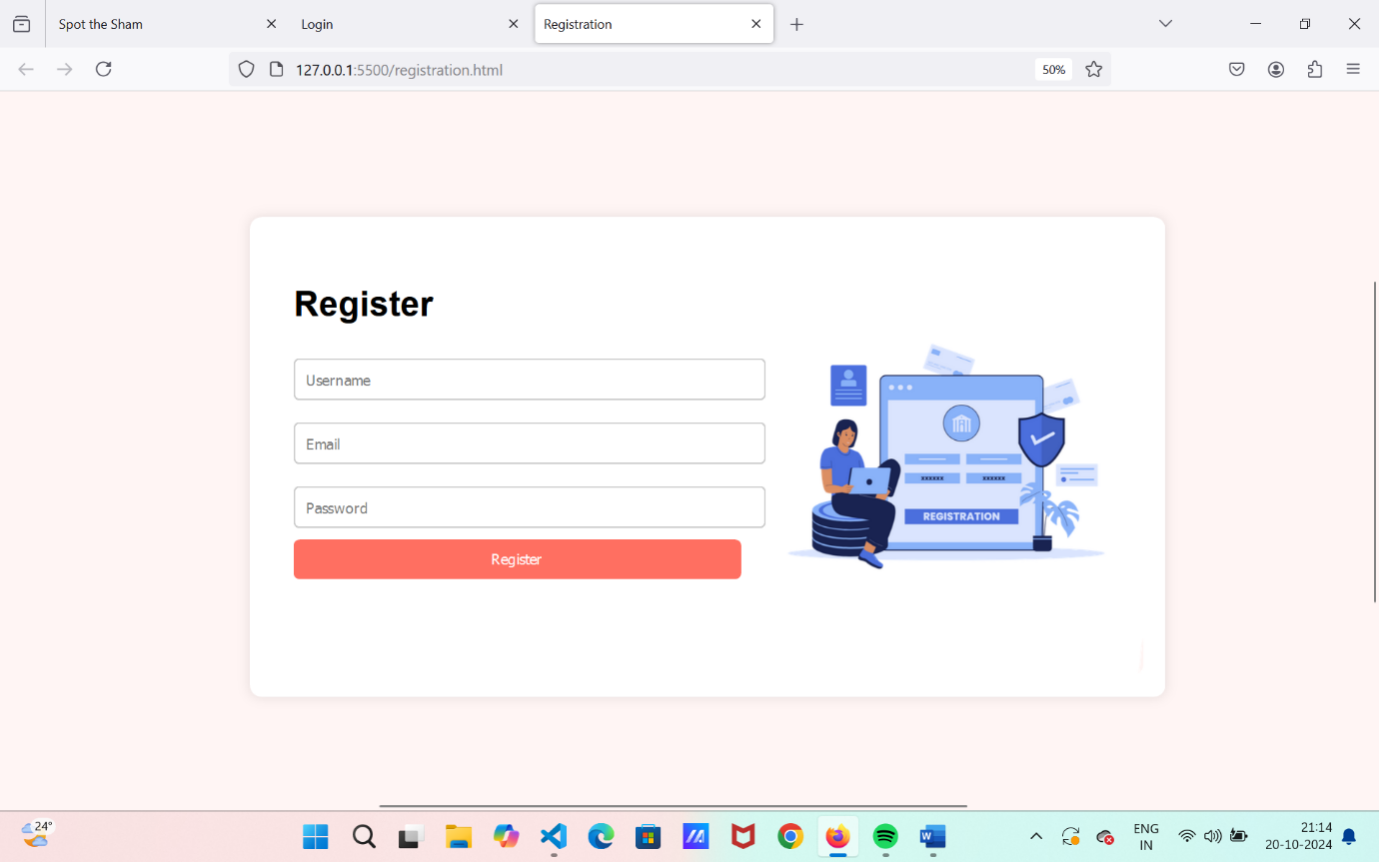
The "Features" section lists key functionalities, such as real-time review analysis, IP address tracking, a user-friendly interface, detailed reporting, and community feedback. The final section, "Contact Us," provides a form where users can enter their name, email, and message to reach out for support or inquiries. The page is designed with clear icons and a footer that states copyright information, giving it a professional touch.

**6.2.2 login.html:**



The login.html page provides a straightforward interface for users to log into the "Spot the Sham" website. It centers a simple form where users can enter their email and password to access their accounts. Upon clicking the "Login" button, a JavaScript function is triggered to send these credentials to the backend API (http://localhost:5000/api/login) using a POST request. If the login is successful, an authentication token is saved in the browser's localStorage, and the user is redirected to the homepage (index.html). If there's an error, an alert displays the error message. The page is also responsive, with a layout that adjusts for smaller screens by hiding the image and stacking form elements vertically to ensure usability across devices. This setup provides a clean, user-friendly login experience.

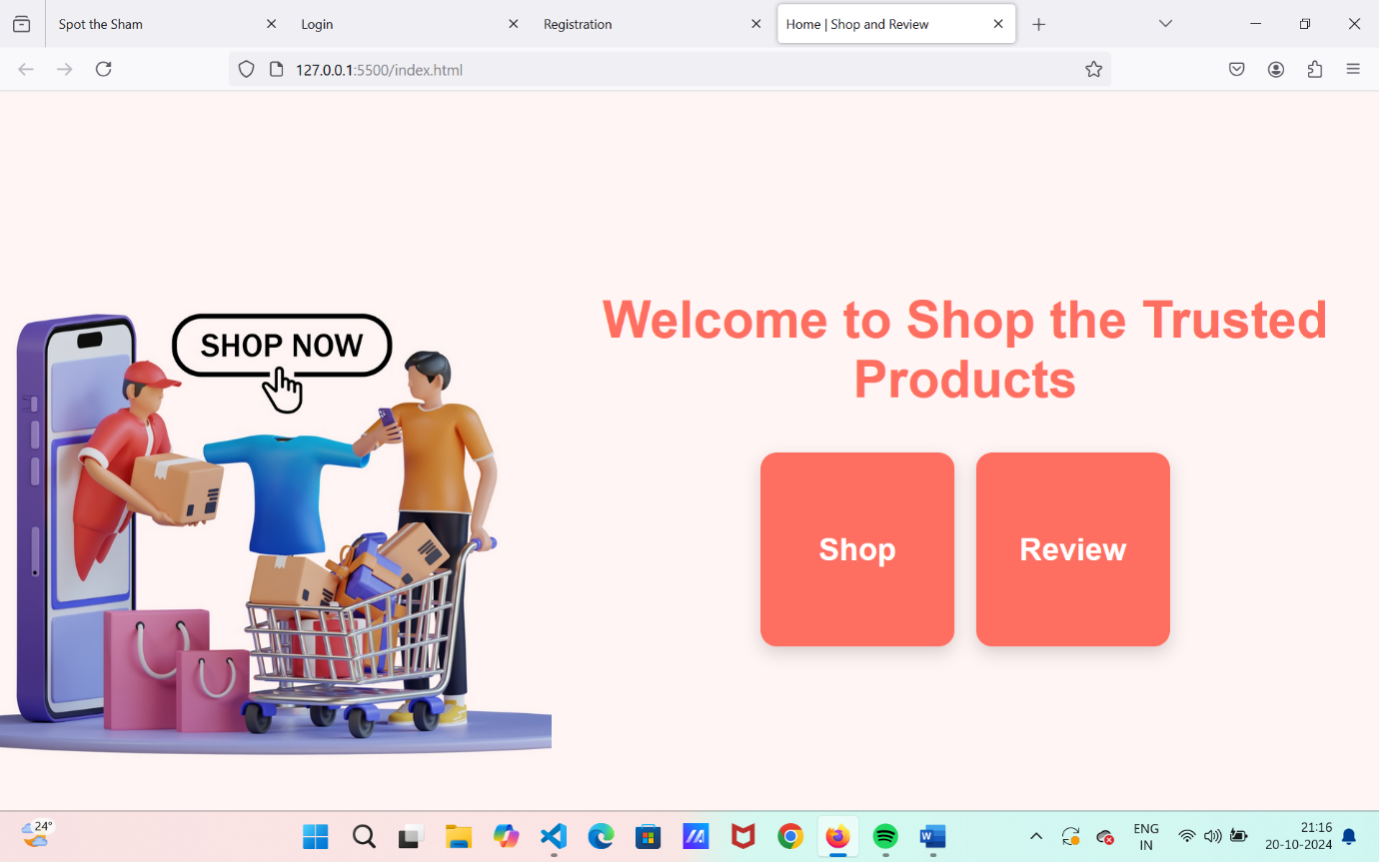
**6.2.3 registration.html:**



The registration.html page allows users to create an account on the "Spot the Sham" website. It includes a registration form where users can enter their username, email, and password. When the "Register" button is clicked, a JavaScript function is triggered to handle the form submission without reloading the page. This function collects the entered data and sends it to the backend API (http://localhost:5000/api/register) via a POST request.

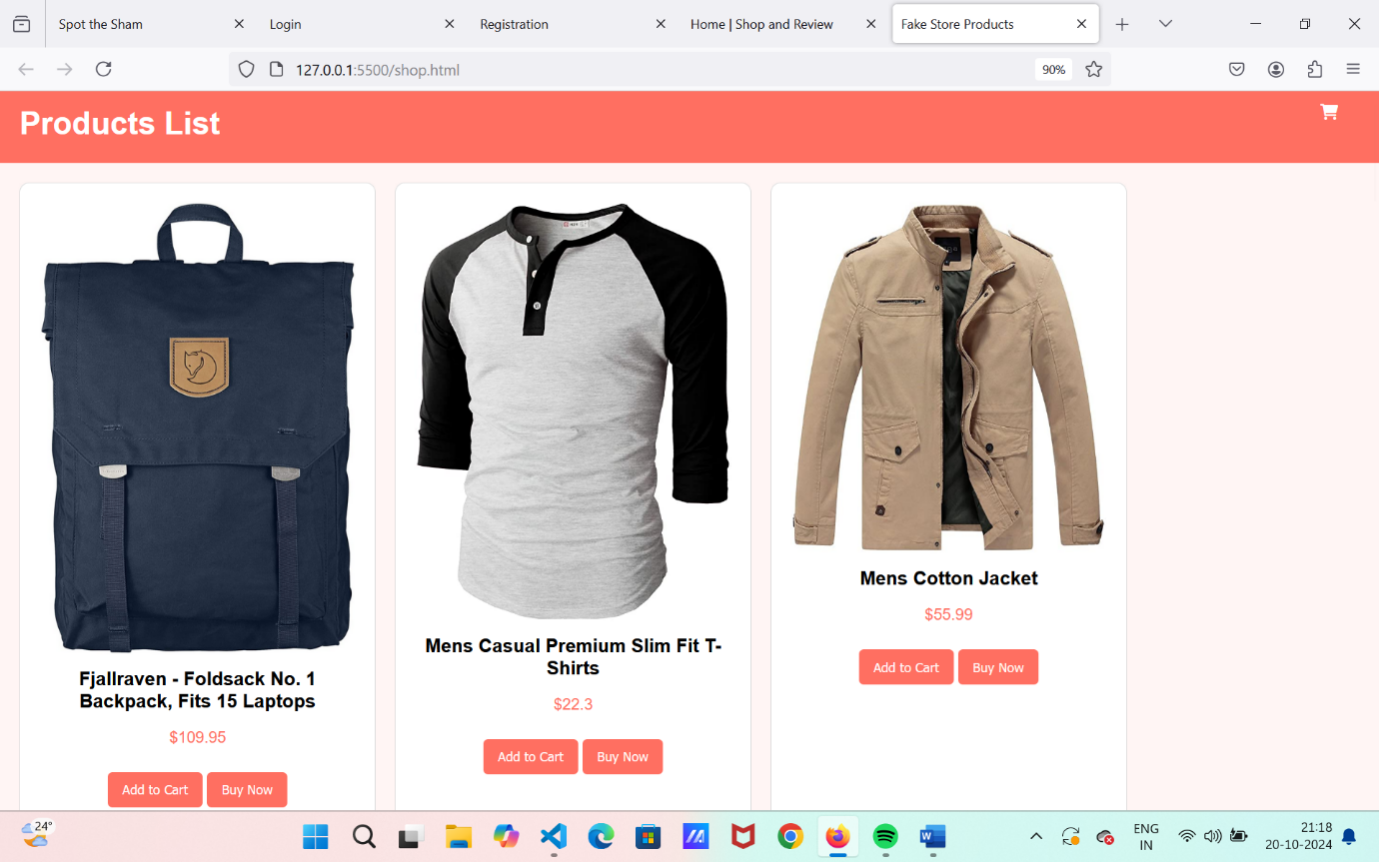
If the registration is successful, the server sends back a message, which is displayed to the user as an alert. The user is then redirected to the login page (login.html). If there is an error, the user is alerted to try again. This setup provides a smooth and user-friendly way to create an account.

**6.2.4 index.html:**



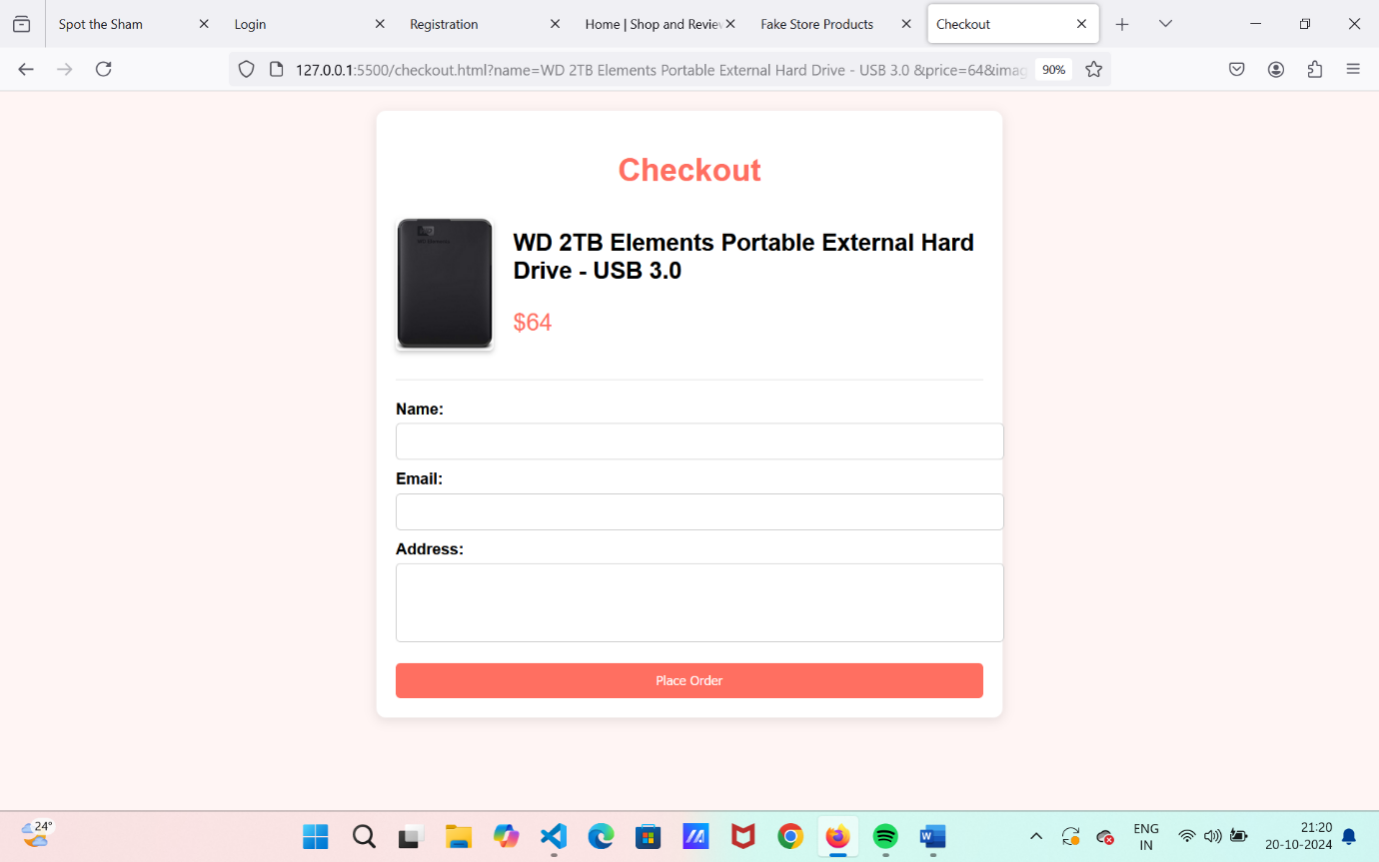
This is the landing page of the "Spot the Sham" website, focused on offering a trusted shopping experience. The background is soft peach, with a prominent welcome message: "Welcome to Shop the Trusted Products." On the left, a 3D illustration shows a smartphone with a delivery person handing out packages, symbolizing e-commerce. To the right, there are two main buttons: "Shop" and "Review," in bold coral, guiding users to browse products or submit reviews. The page design is clean, user-friendly, and visually emphasizes reliability, aligning with the website's purpose of identifying fake reviews for safer shopping.

**6.2.5 shop.html:**



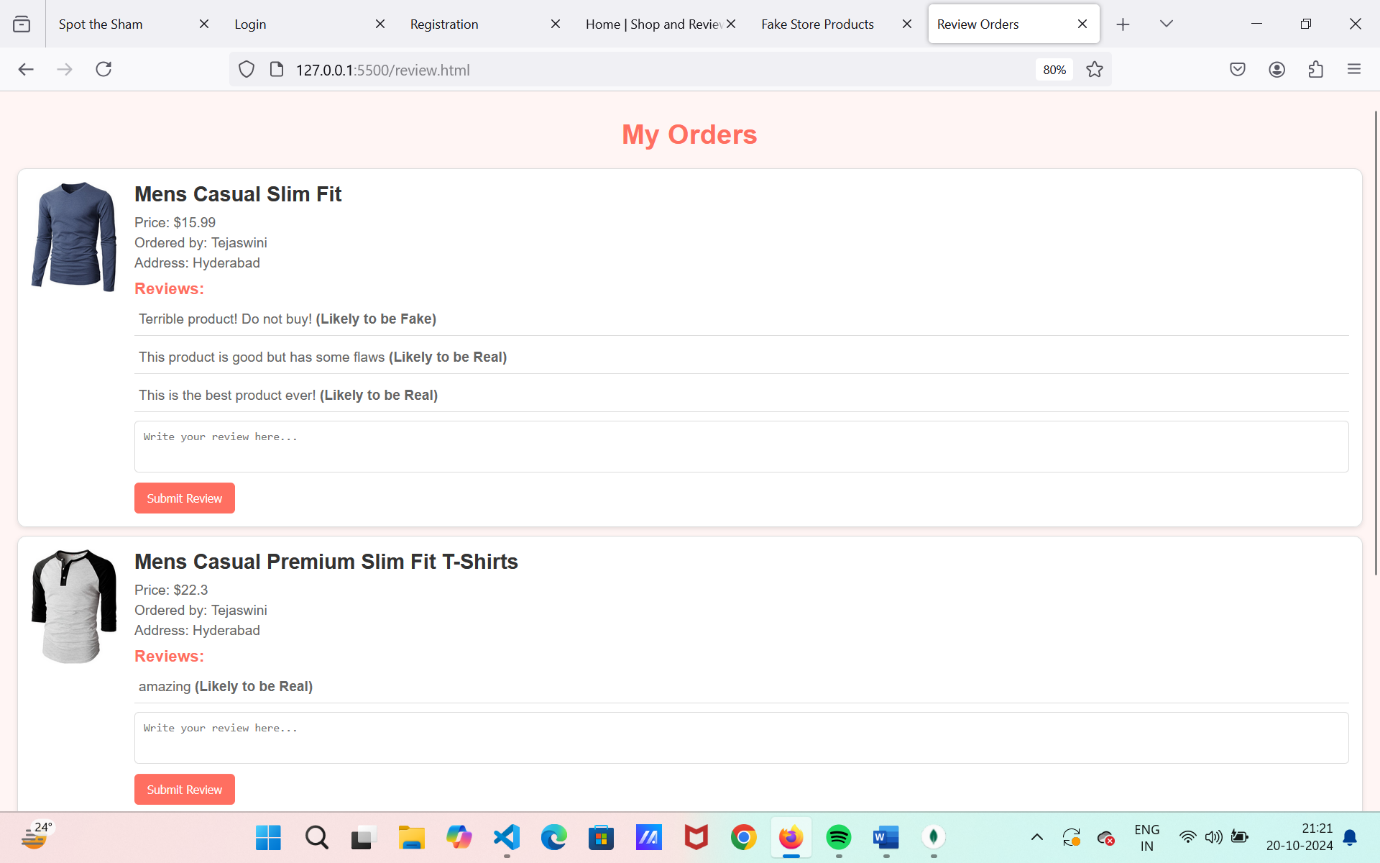
This is the "Products List" page of the "Spot the Sham" website, displaying items available for purchase. The page header is styled in a bold coral banner that reads "Products List." Below, there are cards for each product, showing an image, name, and price. Each product card includes two buttons: "Add to Cart" for adding the item to the shopping cart, and "Buy Now" for immediate purchase. The design is clean, with each product section separated to make browsing easy and straightforward. This layout helps users view multiple items and take quick action on each product.

**6.2.6 checkout.html:**



This HTML document creates a checkout page for an online store. It includes a form where customers can enter their name, email, and address to place an order. The page displays the product's details, including its image, name, and price, which are passed via the URL. The user can see this information, then fill out the form to complete the order. When the form is submitted, the order details are sent to a server using a POST request. If the order is successfully placed, the customer sees an alert and is redirected to the homepage. The page uses CSS for styling, including a fade-in effect when the page loads, and has responsive design elements to ensure the form looks good on various screen sizes.

**6.2.7 review.html:**



his HTML page displays a list of orders and allows customers to leave reviews for each order. The page fetches the order details from the server, showing the product image, name, price, and the customer's information. It also shows any existing reviews for the order. Customers can write and submit new reviews using a textarea. When a review is submitted, it is sent to the server, and the page updates to display the new review along with its sentiment (positive, neutral, or negative). The page also includes styling to make it visually appealing and user-friendly. If there is an issue fetching the orders or submitting a review, appropriate error messages are shown.

# **7.CONCLUSION & FUTURE SCOPE**

**Conclusion:**

The *Fake Product Review Monitoring Using Machine Learning* project successfully identifies and flags potentially fake reviews by leveraging machine learning techniques, sentiment analysis, and IP address tracking. This system enhances the reliability and trustworthiness of product reviews, providing consumers with authentic feedback to make informed purchasing decisions. By filtering out inauthentic reviews, it helps e-commerce platforms maintain credibility and offers a structured approach for administrators to manage reviews efficiently.

**Future Scope:**

1. **Enhanced Model Accuracy**: Further model optimization and experimentation with advanced machine learning algorithms, including deep learning, can improve the accuracy of fake review detection.
2. **Real-Time Monitoring**: Implementing real-time review analysis could allow immediate flagging and alerting of fake reviews as they are posted, providing quicker response times.
3. **User Behavior Analysis**: Incorporating additional behavioral data, such as review timing patterns or user engagement metrics, could increase the detection capability for fake reviews.
4. **Multi-Language Support**: Expanding sentiment analysis to handle multiple languages would broaden the applicability of the system on global platforms.
5. **Integration with External Platforms**: Allowing data exchange and integration with popular e-commerce platforms would make it easier to deploy this solution on a larger scale, extending its reach and impact.
6. **Automated Action on Fake Reviews**: Developing automated responses, such as immediate removal or further verification steps for flagged reviews, could streamline the review moderation process.

This project lays a foundation for creating a robust fake review monitoring system, and future enhancements can further refine its effectiveness and scope across larger and more diverse datasets.

# **8.BIBLIOGRAPHY**

 **Web Sources**

* MongoDB Documentation. (n.d.). "NoSQL Database Documentation." Retrieved from <https://www.mongodb.com/docs>.

 **Online Courses and Tutorials**

* Coursera. (n.d.). "Machine Learning by Andrew Ng." Stanford University, Retrieved from <https://www.coursera.org>.

 **Software and Libraries**

* Pedregosa, F., et al. (2011). *Scikit-learn: Machine Learning in Python.* Journal of Machine Learning Research, 12, pp. 2825–2830.
* *Natural Language Toolkit (NLTK)*. (n.d.). Retrieved from <https://www.nltk.org>.
* *Pandas Documentation.* (n.d.). Retrieved from https://pandas.pydata.org.