# C:\Users\LEVEL51PC\Desktop\tribhuvan-university-tu-logo-AD67A28902-seeklogo.com.png

# 

# Tribhuvan University Faculty of Humanities and Social Sciences Triton International College

# Supervisor’s Recommendation

I hereby recommend that this project prepared under my supervision by **Aashutosh Kattel** entitled “**Natural Language Processing based Recommendation Engine**” in partial fulfillment of the requirements for the degree of Bachelor of Computer Application is recommended for the final evaluation.

**SUPERVISOR   
Yogesh..Deo  
BCA Department  
Triton International College, Koteshwore**

# C:\Users\LEVEL51PC\Desktop\tribhuvan-university-tu-logo-AD67A28902-seeklogo.com.png

# Tribhuvan University Faculty of Humanities and Social Sciences Triton International College

# LETTER OF APPROVAL

# This is to certify that this project prepared by AASHUTOSH KATTEL entitled “Natural Language Processing based Recommendation Engine” in partial fulfillment of the requirements for the degree of Bachelor in Computer Application has been evaluated. In our opinion it is satisfactory in the scope and quality as a project for the required degree.

|  |  |
| --- | --- |
| Supervisor, BCA Department  Triton International College | HOD, BCA Department  Triton International College |
| Internal Examiner | External Examiner |

# ABSTRACT

**Natural Language Processing based Recommendation Engine,** a sophisticated recommendation system leveraging advanced natural language processing techniques to deliver intelligent, contextually-aware content suggestions. By analyzing semantic relationships, textual features, and latent representations of user interactions, the engine employs deep learning models and embedding techniques to capture nuanced semantic meanings. Utilizing techniques such as word embedding, transformer architectures, and contextual analysis, the system generates personalized recommendations that transcend traditional collaborative and content-based filtering approaches. This system is backed by PHP for server-side processing and Python for algorithm for implementation. The recommendation framework integrates advanced NLP methodologies like BERT, word2vec, and semantic similarity metrics to create a dynamic, adaptive recommendation mechanism that understands user preferences at a deeper linguistic and contextual level. This system can be used as a stand-alone service as well as an integrated feature.

**Keywords: Natural Language Processing, Recommendation Systems, Semantic Embedding, Deep Learning, Contextual Recommendations***.*

# ACKNOWLEDGEMENT

In the pursuit of this project, I find myself compelled to recognize the most critical contributor to its success: myself. Throughout this challenging journey, I have been my own most steadfast supporter, advocate, and motivator.

I acknowledge the countless late nights, the moments of intense concentration, and the relentless pursuit of excellence that defined this project. My dedication to meticulous research, innovative problem-solving, and continuous self-improvement has been nothing short of extraordinary. I am grateful for my ability to transform challenges into opportunities, to learn from setbacks, and to emerge stronger and more knowledgeable. This achievement is a testament to the power of self-belief, hard work, and the extraordinary potential that resides within.

In recognizing my own efforts, I am reminded that success is not merely about the end result, but about the journey of personal growth, resilience, and unwavering self-trust.

**Aashutosh Kattel**

Contents

[Supervisor’s Recommendation i](#_Toc183099986)

[LETTER OF APPROVAL ii](#_Toc183099989)

[ABSTRACT iii](#_Toc183099995)

ACKNOWLEDGEMENT………………………………………………………………..iv

[List of Figures vii](#_Toc183099996)

[List of Tables viii](#_Toc183099997)

[List of Abbreviations ix](#_Toc183099998)

[Chapter 1: Introduction 1](#_Toc183099999)

[1.1 Introduction 1](#_Toc183100000)

[1.2 Problem Statement 2](#_Toc183100001)

[1.3 Objectives 2](#_Toc183100002)

[1.4 Scope and Limitations 3](#_Toc183100003)

[1.4.1 Scope 3](#_Toc183100004)

[1.4.2 Limitations 3](#_Toc183100005)

[1.5 Development Methodology 4](#_Toc183100006)

[1.6 Report Organization 5](#_Toc183100007)

[Chapter 2: Background Study and Literature Review 6](#_Toc183100008)

[2.1 Background Study 6](#_Toc183100009)

[2.2 Literature Review 7](#_Toc183100010)

[Chapter 3: System Analysis and Design 7](#_Toc183100011)

[3.1 System Analysis 8](#_Toc183100012)

[3.1.1 Requirement Analysis 8](#_Toc183100013)

[3.1.2 Feasibility Study 9](#_Toc183100014)

[3.1.3 Data modelling: ER Diagram 10](#_Toc183100015)

[3.1.4 Process Modelling: DFD 12](#_Toc183100016)

[3.2 System Design 13](#_Toc183100017)

[3.2.1 Database Schema Design 13](#_Toc183100018)

[3.2.2 Physical DFD 14](#_Toc183100019)

[3.3 Algorithm details 15](#_Toc183100020)

[3.3.1 Natural Language Processing 15](#_Toc183100021)

[Chapter 4: Implementation and Testing 17](#_Toc183100022)

[4.1 Implementation 17](#_Toc183100023)

[4.1.1 Tools Used 17](#_Toc183100024)

[4.1.2 Implementation details of modules 18](#_Toc183100025)

[4.2 Testing 19](#_Toc183100026)

[4.2.1 Test cases for Unit Testing 19](#_Toc183100027)

[4.2.1 Test cases for System Testing 19](#_Toc183100028)

[Chapter 5: Conclusion and Future Recommendations 24](#_Toc183100029)

[5.1 Conclusion 24](#_Toc183100030)

[5.2 Lesson learnt/Outcome 24](#_Toc183100031)

[5.3 Future Recommendations 25](#_Toc183100032)

[Appendices 29](#_Toc183100033)

[References …31](#_Toc183100034)

# List of Figures

[Figure 1: Iterative Waterfall Model 4](#_Toc178154478)

[Figure 2: Use Case Diagram 8](#_Toc178154479)

[Figure 3: ER diagram 11](#_Toc178154480)

[Figure 4: Process Modeling 12](#_Toc178154481)

[Figure 5: Database Schema 13](#_Toc178154482)

[Figure 6: Context Diagram 14](#_Toc178154483)

[Figure 7: Level 1 DFD 14](#_Toc178154484)

# List of Tables

[Table 1: Report Organization 5](#_Toc178154492)

[Table 2: Sign un Testing 22](#_Toc178154493)

[Table 3: Sign in Test 23](#_Toc178154495)

# List of Abbreviations

CSS Cascading Style Sheets

DOM Document Object Model

DFD Data Flow Diagram

ER Entity Relationship

API Application Programming Interface

JSON JavaScript Object Notation

DB Database

URL Uniform Resource Locater

NLP Natural Language Processing

UI User Interface

UX User Experience

CASE Computer Aided Software Engineering

# Chapter 1: Introduction

## 1.1 Introduction

In the rapidly evolving digital marketplace, consumers increasingly rely on product reviews and recommendations to guide their purchasing decisions. The exponential growth of online platforms and e-commerce ecosystems has generated an unprecedented volume of user-generated content, creating both opportunities and challenges for intelligent recommendation systems. Natural Language Processing (NLP) emerges as a transformative technology that bridges the gap between raw textual data and actionable consumer insights, offering a sophisticated approach to understanding and interpreting the complex landscape of product reviews.

Traditional recommendation engines often struggle with the nuanced complexity of human language, frequently missing critical contextual and sentiment-driven information. An NLP-powered recommendation engine represents a revolutionary solution that transcends conventional filtering techniques by comprehending the semantic meaning, emotional undertones, and contextual subtleties embedded within product reviews. By leveraging advanced linguistic analysis techniques, these intelligent systems can extract profound insights from unstructured text, transforming ambiguous user feedback into precise, actionable recommendations. Machine learning models trained on extensive corpora of review data can identify intricate patterns, assess review credibility, and generate nuanced recommendations that reflect genuine user experiences with unprecedented accuracy.

The technological sophistication of NLP-based recommendation engines lies in their ability to perform multidimensional analysis through advanced text preprocessing, sentiment classification, word embedding techniques, and deep learning architectures. These systems employ complex algorithms that convert unstructured textual data into structured, analyzable information, enabling a comprehensive understanding of product attributes, user satisfaction levels, and potential limitations. By integrating cutting-edge artificial intelligence with linguistic analysis, these recommendation engines empower consumers to make informed decisions, providing transparent and personalized product guidance.

## 1.2 Problem Statement

In the contemporary digital marketplace, consumers face significant challenges in navigating the overwhelming abundance of product information and user-generated reviews across multiple platforms. The complexity of extracting meaningful insights from diverse and unstructured textual data creates substantial barriers to informed decision-making. Existing recommendation systems often rely on simplistic approaches that fail to capture the nuanced semantics, contextual subtleties, and genuine user experiences embedded within product reviews. Moreover, the lack of sophisticated natural language processing techniques results in recommendations that are frequently irrelevant, generic, or disconnected from actual user sentiments

* Current recommendation engines struggle to effectively analyze sentiment, extract meaningful features, and generate personalized suggestions that truly reflect the diverse and complex nature of user experiences.
* Traditional filtering methods predominantly focus on surface-level metadata or basic rating systems, overlooking the rich linguistic information that could provide deeper product understanding.

## 1.3 Objectives

The objectives of NLP based Recommendation Engine are:

* To develop a platform that provides recommendations based on attributes such sentimental keywords, reviews, and tags, rather than relying solely on user ratings or personalized.
* To offer users a intuitive interface where they can easily read reviews, explore comments, and receive suggestions for product without the need for personalized data.

## 1.4 Scope and Limitations

### 1.4.1 Scope

The proposed NLP-powered recommendation engine encompasses a comprehensive approach to transforming product review analysis and recommendation strategies. Its scope spans advanced natural language processing techniques, including sentiment analysis, semantic embedding, and contextual feature extraction across diverse product categories. The system will integrate machine learning models capable of processing multilingual review data, understanding complex linguistic nuances, and generating personalized recommendations. By leveraging deep learning architectures and sophisticated text analysis algorithms, the recommendation engine aims to provide users with intelligent, context-aware product insights. The scope extends to developing a flexible, scalable framework that can adapt to evolving digital marketplaces, offering enhanced user experiences through precise, data-driven recommendation methodologies.

### 1.4.2 Limitations

While NLP based Recommendation Engine aims to simplify the resume creation process, certain limitations are inherent:

* **Limited to Sentiment-Based Recommendations:** Since the app does not rely on user ratings or personalized viewing history, recommendations are purely based on sentiments, which may not always align with individual user preferences.
* **No Personalization:** The app does not offer personalized recommendations based on individual user preferences or past behavior, which may limit its appeal to users looking for tailored suggestions.
* **Internet Dependency:** Users must be connected to the internet to access the recommendations and review features, as the app relies on an online database for admin privileges.

## 1.5 Development Methodology

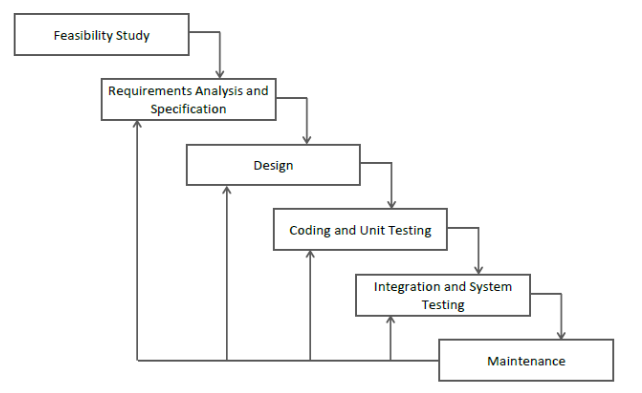


Figure : Iterative Waterfall Model

The development of the NLP based Recommendation Engine is developed with the Iterative Waterfall methodology, which is ideal for solo projects where requirements are clearly defined from the outset and there is limited need for iterative user feedback. This approach involves sequential phases, including requirements gathering, design, implementation, testing, and deployment, allowing for a structured progression through each stage of development. In the initial phase, comprehensive requirements will be documented to guide the design and functionality of the app. The design phase will focus on creating an intuitive user interface and a seamless user experience. Following this, the implementation phase will involve coding the app using established technologies such as PHP and Python. Once the development is complete, thorough testing will be conducted to ensure that the app operates as intended and meets all specified requirements. The Waterfall methodology's linear structure facilitates clear documentation and project management, ultimately resulting in a robust and cohesive app that fulfills its intended purpose without the need for ongoing user feedback.

## 1.6 Report Organization

This is the report organization for the system which also includes charts/diagrams to illustrate the system architecture and design. Furthermore, it contains information regarding the tools and technologies used to build the system.

Table : Report Organization

|  |  |
| --- | --- |
| **Chapter 1:** | **INTRODUCTION:**  Provides an overview of the project, problem statement, objectives, scope and limitations and development mythology. |
| **Chapter 2:** | **BACKGROUND STUDY & LITERATURE REVIEW:**  Provides an overview of background study and literature review done during the project. |
| **Chapter 3:** | **SYSTEM ANALYSIS & DESIGN:**  Provides an overview of the analysis carried out, design implemented and algorithm used within the project. |
| **Chapter 4:** | **IMPLEMENTION & TESTING:**  Provides the overview of the tools used and testing carried out for the system. |
| **Chapter 5:** | **CONCLUSION & FUTURE RECOMMENDATION:**  Provides a summary of the project, outcome of the project and recommendation that can be made. |

# Chapter 2: Background Study and Literature Review

## 2.1 Background Study

The evolution of recommendation systems has been intrinsically linked to the exponential growth of digital platforms and the increasing complexity of consumer decision-making processes. Early recommendation approaches emerged in the late 1990s and early 2000s, predominantly utilizing collaborative filtering techniques that relied on user-item interaction matrices. These initial systems focused on identifying similarities between users or items based on historical interaction data, providing rudimentary suggestions that often lacked depth and personalization.

As digital ecosystems expanded, researchers began exploring content-based filtering methods that incorporated product attributes and metadata to enhance recommendation accuracy. These approaches sought to overcome the limitations of collaborative filtering by analyzing intrinsic characteristics of items and user preferences. However, these methods still struggled to capture the nuanced contextual information embedded within user-generated content, particularly in textual reviews and feedback.

The advent of advanced natural language processing techniques marked a significant paradigm shift in recommendation system design. Machine learning algorithms, particularly deep learning models like recurrent neural networks and transformer architectures, introduced unprecedented capabilities in understanding semantic relationships and extracting meaningful insights from unstructured textual data. Pioneering research in word embeddings, such as Word2Vec and GloVe, demonstrated the potential of representing linguistic contexts mathematically, enabling more sophisticated semantic analysis. Sentiment analysis techniques further enhanced recommendation systems by incorporating emotional and subjective dimensions into the recommendation process.

The continuous evolution of recommendation technologies reflects the growing complexity of digital consumer interactions and the increasing demand for more intelligent, nuanced decision-support systems that can effectively navigate the vast landscape of user-generated content.

## 2.2 Literature Review

Numerous studies have indicated that ratings alone are insufficient to capture the complexity of individual preferences. According to Adomavicius and Tuzhilin [1], traditional collaborative filtering approaches often face issues like the cold-start problem, where new users or items lack sufficient interaction data, resulting in suboptimal recommendations. These methods also tend to reinforce existing preferences rather than explore new ones, leading to a homogenized viewing experience [2].

Recent research emphasizes the importance of incorporating additional data sources to enhance recommendation accuracy. For instance, attributes such as tags can provide a more nuanced understanding of user preferences [3]. Attribute-based recommendations allow users to discover films that align with specific interests beyond numerical ratings. This approach addresses the limitations highlighted by Hu et al. [4], who noted that relying solely on user ratings can overlook crucial contextual factors influencing buying choices.

NLP algorithms have gained traction in the field of recommendation systems due to their effectiveness in measuring the polarity between items based on their attributes. Research by Sarwar et al. [5] shows that tokenizing and polarity can effectively analyze relationships between items, leading to improved recommendation quality. By leveraging this algorithm, the Recommendation Engine can provide personalized suggestions that resonate with individual user preferences rather than solely relying on aggregate ratings.

Additionally, allowing users to read detailed reviews fosters a richer data environment that can enhance the recommendation process. User-generated content has been shown to significantly impact consumer decisions and preferences [6]. Integrating qualitative data from reviews can enrich the understanding of user interests, ultimately leading to more relevant verdicts.

In conclusion, the shift from traditional rating-based systems to attribute-based recommendations presents an opportunity to enrich the movie discovery experience. By utilizing NLP algorithm and incorporating detailed user reviews, the Recommendation Engine aims to address the limitations of existing systems and provide a more personalized and engaging platform for movie enthusiasts.

# Chapter 3: System Analysis and Design

## 3.1 System Analysis

### 3.1.1 Requirement Analysis

The requirement analysis for the Recommendation Engine identifies essential functionalities and quality attributes that will drive the development of an engaging and user-centric platform. This analysis encompasses both functional requirements, such as user registration, and a personalized recommendation system, and non-functional requirements, which focus on performance, usability, security, and scalability. By thoroughly outlining these requirements, the app aims to provide an intuitive interface for receiving verdicts, discover personalized recommendations, and enhance their overall experience while ensuring robust system performance and user satisfaction.

1. **Functional Requirements**

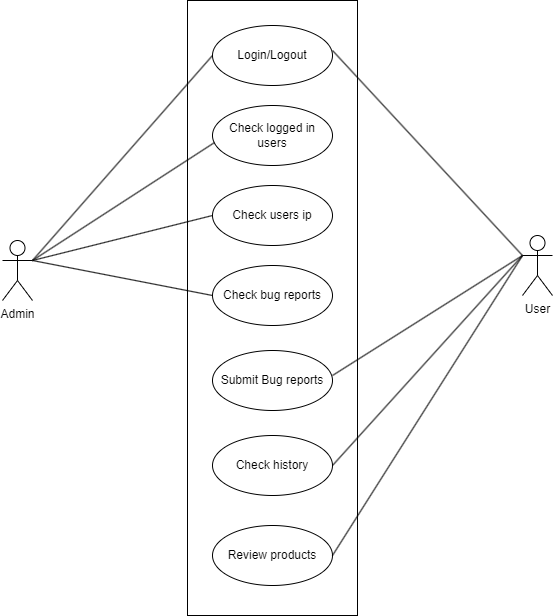


Figure : Use Case Diagram

* **User Login and Authentication:** Users should be able to create accounts securely and log in using google credentials.
* **User Management:** Admins should manage the user accounts and view user IPs.
* **Recommendation System:** The system should generate product verdict based on user interactions using NLP.
* **Search and Browse Functionality:** Users should be able to search and review for products.

**Non-Functional Requirements:**

* **Performance:** Ensure the application can handle concurrent users efficiently, with fast response times for resume generation and data extraction tasks.
* **Security:** Implement data encryption for sensitive user information, secure storage of credentials, and robust authentication mechanisms to protect user accounts.
* **Scalability:** Design the system to scale seamlessly with increasing user base and data volume, supported by scalable infrastructure and efficient database management.
* **Reliability:** Minimize downtime through reliable hosting solutions and regular maintenance updates to address bugs and enhance application stability.

### 3.1.2 Feasibility Study

The feasibility study indicates that the NLP based Recommendation Engine is technically feasible with current technologies, operationally viable with streamlined processes and an engaging user interface, and economically feasible through diverse revenue streams and efficient management practices. By thoroughly examining these feasibility aspects, the engine aims to provide a compelling platform for users seeking to enhance their shopping experience through personalized recommendations and summarized reviews.

1. **Technical**

The NLP based Recommendation Engine will utilize established technologies and frameworks, ensuring seamless integration and robust performance. The implementation of a NLP algorithm for the recommendation system poses some technical challenges, particularly in ensuring accuracy and efficiency in generating suggestions.

1. **Operation**

Operationally, the NLP based Recommendation Engine will streamline the process of online shopping and review submission, minimizing the time and effort required for users to engage with the platform. The user interface will be designed to be intuitive, catering to users of various technical backgrounds. Continuous feedback mechanisms will be integrated to gather user insights and ensure any operational issues are promptly addressed, thereby enhancing overall user satisfaction and encouraging higher adoption rates.

1. **Economic**

The project's economic feasibility involves an analysis of initial development costs alongside ongoing operational expenses. Development costs will encompass software development, database setup, and the integration of the recommendation system. Revenue generation strategies, such as advertising partnerships, premium features, or subscription models for advanced functionalities, will be explored to ensure sustainable operations and fund future updates and enhancements

3.1.3 Data modelling: ER Diagram

The ER diagram effectively captures the essential entities and relationships within the NLP based Recommendation Engine, enabling efficient database design and ensuring that all necessary data relationships are clearly defined for the application’s functionality. This foundational data modeling will facilitate the implementation of features related to user reviews, product information, and tagging, enhancing the overall user experience.

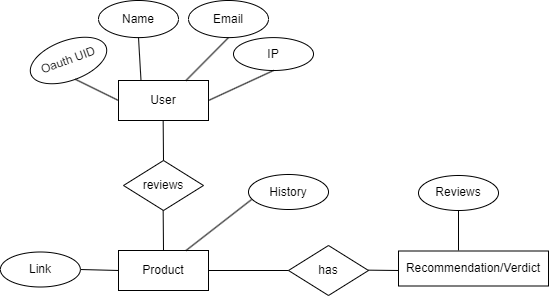


Figure : ER diagram

### 

### **3.1.4** Process Modelling: DFD

Process modeling involves defining the workflows and processes that will be used in the project to facilitate user interactions and system functionality. This modeling will outline how data flows through the application, detailing the various operations that can be performed by users and the system.

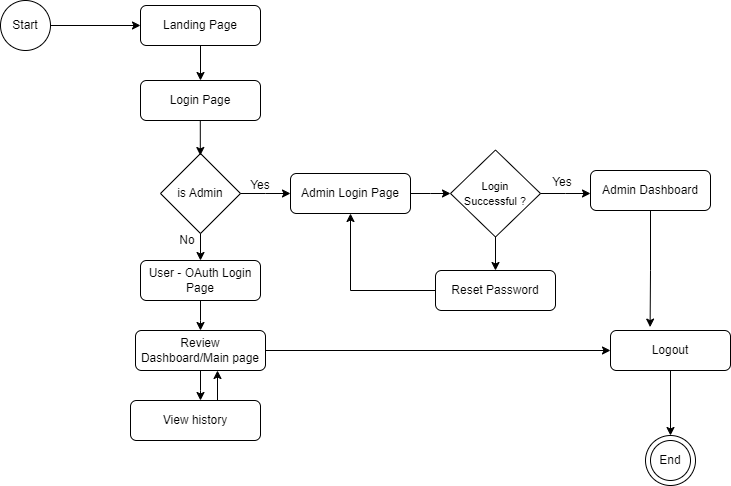


Figure : Process Modeling

## 3.2 System Design

### 3.2.1 Database Schema Design

In the context of the NLP based Recommendation Engine using MySql, the database schema design outlines the structure of collections and documents, ensuring efficient data storage and retrieval. Below is a detailed schema design that corresponds to the entities identified.

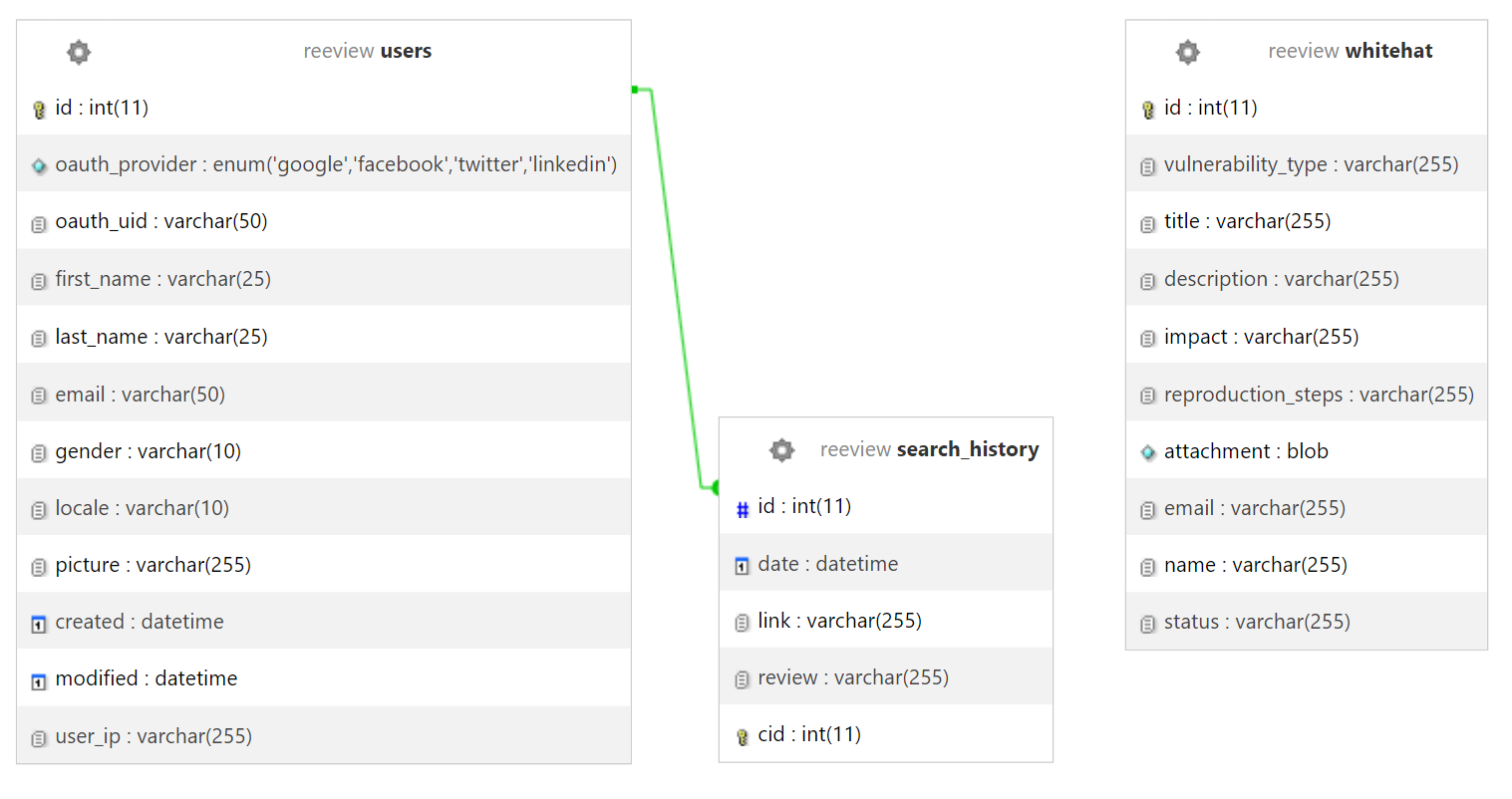


Figure : Database Schema

### 3.2.2 Physical DFD

**i) Context Diagram**

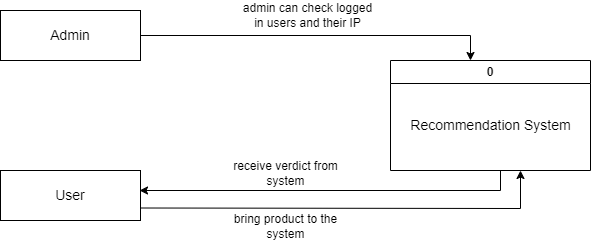


Figure : Context Diagram

1. **Level 1 DFD**

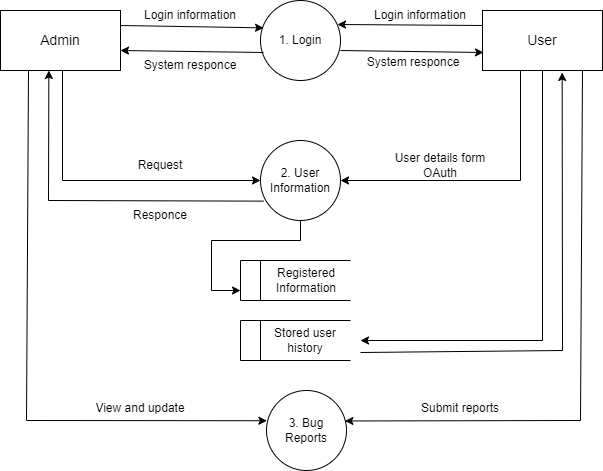


Figure : Level 1 DFD

## 3.3 Algorithm details

### 3.3.1 Natural Language Processing

In the project, the **NLP algorithm** is employed to provide recommendations. This algorithm is particularly effective in analyzing the sentiments.

## 1. Sentiment Analysis Function

Let R = {r₁, r₂, ..., rₙ} be the set of n reviews, and S be the set of sentiment scores.

S = { s\_i | s\_i = p(r\_i), i = 1, 2, ..., n }

Where p(r) is the polarity function that maps a review r to a sentiment score s.

The average sentiment can be represented as:

**s̄ = (1/n) \* Σ(i=1 to n) s\_i**

## 2. Sentiment Categorization Function

Let S = {s₁, s₂, ..., sₙ} be the set of n sentiment scores, and C be the set of categories.

C = { c\_i | c\_i = f(s\_i), i = 1, 2, ..., n }

Where f(s) is defined as:

f(s) = { 'Outstanding: Highly Recommended', if s > 0.75 'Excellent: Top-tier Choice', if 0.5 < s ≤ 0.75 'Good: Recommended', if 0 < s ≤ 0.5 'Acceptable: In your consideration', if -0.5 < s ≤ 0 'Poor: Not Recommended', otherwise }

The count of reviews in each category can be represented using indicator functions:

**count(category) = Σ(i=1 to n) 1{f(s\_i) = category}**

Where 1{condition} is the indicator function that equals 1 when the condition is true and 0 otherwise.

## 3. Keyword Extraction Function

Let R be a review, and K be the set of extracted keywords.

K = {k | k = g(R)}

Where g(R) is the RAKE algorithm function that extracts keywords from the review R.

## 4. Keyword Sentiment Categorization Function

Let K = {k₁, k₂, ..., kₘ} be the set of m extracted keywords, and C be the set of sentiment categories.

C = { c\_j | c\_j = h(k\_j), j = 1, 2, ..., m }

Where h(k) is defined as:

h(k) = { 'positive', if p(k) > 0.2 'negative', if p(k) < -0.2 'neutral', otherwise }

Where p(k) is the polarity function that maps a keyword k to a sentiment score.

The proportion of keywords in each sentiment category can be represented as:

proportion(category) = (1/m) \* Σ(j=1 to m) 1{h(k\_j) = category}

**Benefits of Using NLP Algotithm**

* NLP improves sentiment-based personalization by detecting subtle emotional undertones in text reviews, going beyond simple numerical ratings. This deeper understanding of user feedback helps create more targeted recommendations by recognizing context-specific preferences and adapting to user sentiment patterns. The algorithm can efficiently handle large datasets, making it suitable for scaling as the app's movie database grows.
* The enhanced feature extraction capability automatically identifies key information from unstructured text data. By recognizing and categorizing important attributes from reviews and comments, NLP builds richer user profiles and product descriptions, leading to more accurate recommendations.
* NLP enables scalable processing of massive amounts of user feedback across multiple platforms. This automated approach maintains consistent evaluation criteria while processing thousands of reviews simultaneously, allowing businesses to keep their recommendations current with minimal manual intervention.

# Chapter 4: Implementation and Testing

## 4.1 Implementation

### 4.1.1 Tools Used

1. **Visual Studio Code:** VS Code is a widely used source code editor known for its speed and versatility.
2. **HTML:** Hypertext Markup Language is employed in this web app for UI structure.
3. **CSS:** Cascading Style sheet is employed to make the structure obtained from HTML more attractive.
4. **JAVASCRIPT:** JS is used for DOM manipulation in this web app.
5. **PHP:** PHP has served as a main backend language. Establishing connection with database and routing.
6. **PYTHON:** NLP Algorithm implementation and logic processing.
7. **PANTRY DB:** JSON based storage solution for storing Admin credentials.
8. **MYSQL:** MySql has served as database for storing user’s data.
9. **CASE tools:** These tools are used to represent system component, data and control flow in this project and system structure in graphical form and construct the project document. (Draw.io, MS-Word)
10. **GIT/GITHUB:** Version control system.
11. **OAuth:** Google OAuth 2.0 has been employed for seamless user registration and logging in.
12. **Postman:** API testing tool.

### 4.1.2 Implementation details of modules

**1. User Management Module**

* **Responsibility:** Handles user registration, authentication, and profile management.
* **Implementation:**

**Backend (PHP):**

User data is stored in the Users collection in Mysql, including user credentials and profile information.

Authentication: OAuth for authentic and secure login sessions.

**Frontend:**

Logins: Simple log in process using OAuth 2.0..(Google API)

History Management: Allows users to update their history and view previously reviewed product.

**2. Admin Module**

* **Responsibility:** Admin-only module that allows user monitoring and bug report handling.
* **Implementation:**

**Backend (PHP + Python):**

Admins can view user details received from OAuth.

Data is stored in Mysql.

Admin login credentials rely on Pantry DB (Json based storage solution)

**Frontend :**

User DB table.

**3. Product Review Module**

* **Responsibility:** Allows users to view product reviews and get verdicts from the system..
* **Implementation:**

**Backend (PHP + Python)**

**Frontend:**

Review Form: Allows users to paste URL.

Review Display: Shows summarized reviews.

## 4.2 Testing

### 4.2.1 Test cases for Unit Testing

## Test Case 1: Sentiment Analysis Function

**Description**: Verify the sentiment analysis function correctly maps reviews R to sentiment scores S using polarity function p(r).

**Input**:  
R = {r₁, r₂, r₃} where:

* r₁ = "This product is excellent!"
* r₂ = "Average performance, nothing special"
* r₃ = "Terrible experience, very disappointed"

**Expected Output**:  
S = {s₁, s₂, s₃} where:

* s₁ ≈ 0.8 (highly positive)
* s₂ ≈ 0.0 (neutral)
* s₃ ≈ -0.7 (highly negative)

Average sentiment: s̄ = (1/3) \* Σ(i=1 to 3) s\_i ≈ 0.033

**Test Steps**:

1. Initialize R = {r₁, r₂, r₃}
2. Execute S = analyze\_sentiment(R)
3. Verify ∀i ∈ [1,3]: s\_i = p(r\_i)
4. Calculate s̄ and verify result

## Test Case 2: Sentiment Categorization Function

**Description**: Verify the categorization function f(s) correctly maps sentiment scores to appropriate categories.

**Input**:  
S = {s₁, s₂, s₃, s₄, s₅} where:

* s₁ = 0.8
* s₂ = 0.6
* s₃ = 0.3
* s₄ = -0.2
* s₅ = -0.7

**Expected Output**:  
C = {c₁, c₂, c₃, c₄, c₅} where:

* c₁ = 'Outstanding: Highly Recommended'
* c₂ = 'Excellent: Top-tier Choice'
* c₃ = 'Good: Recommended'
* c₄ = 'Acceptable: In your consideration'
* c₅ = 'Poor: Not Recommended'

**Test Steps**:

1. Initialize S = {s₁, s₂, s₃, s₄, s₅}
2. Execute C = categorize\_sentiment(S)
3. Verify ∀i ∈ [1,5]: c\_i = f(s\_i)
4. Verify count(category) = Σ(i=1 to 5) 1{f(s\_i) = category}

## Test Case 3: Keyword Extraction Function

**Description**: Verify the RAKE algorithm correctly extracts relevant keywords from review text.

**Input**:  
R = "The battery life is excellent but the price is too high for this smartphone"

**Expected Output**:  
K = {k₁, k₂} where:

* k₁ = "battery life excellent"
* k₂ = "price high smartphone"

**Test Steps**:

1. Initialize review R
2. Execute K = extract\_keywords(R)
3. Verify K = g(R)
4. Verify |K| > 0

## Test Case 4: Keyword Sentiment Categorization

**Description**: Verify keyword sentiment categorization function h(k) correctly classifies keyword sentiments.

**Input**:  
K = {k₁, k₂, k₃} where:

* k₁ = "excellent performance"
* k₂ = "average quality"
* k₃ = "poor service"

**Expected Output**:  
C = {c₁, c₂, c₃} where:

* c₁ = 'positive' (p(k₁) > 0.2)
* c₂ = 'neutral' (-0.2 ≤ p(k₂) ≤ 0.2)
* c₃ = 'negative' (p(k₃) < -0.2)

proportion(positive) = 1/3 proportion(neutral) = 1/3 proportion(negative) = 1/3

**Test Steps**:

1. Initialize K = {k₁, k₂, k₃}
2. For each k\_j ∈ K:
   * Execute c\_j = categorize\_keyword\_sentiment(k\_j)
   * Verify c\_j = h(k\_j)
3. Calculate proportion(category) = (1/3) \* Σ(j=1 to 3) 1{h(k\_j) = category}
4. Verify Σ proportions = 1

### 4.2.1 Test cases for System Testing

**1) Sign Up Testing**

Table : Sign un Testing

|  |  |
| --- | --- |
| Objective | To test the user signup |
| **Test I** | |
| Action | User enters through gmail account |
| Expected Result | User gets to access review page and admin receives user info. |
| Actual Result | User got to access review page and admin received user info.. |
| **Test II** | |
| Action | Attempt register a user with an email without fullname. |
| Expected Outcome | Receive an error message requesting for full name. |
| Actual Result | Received an error message requesting for full name. |
| Conclusion | Test Successful |

**2) Sign In Testing**

Table 3: Sign in Test

|  |  |
| --- | --- |
| Objective | To test user sign in. |
| **Test I** | |
| Action | User credentials was provided and sign in initiated. |
| Expected Result | User is signed in and redirected into the system dashboard. |
| Actual Result | User was signed in and redirected into the system dashboard. |

# Chapter 5: Conclusion and Future Recommendations

## 5.1 Conclusion

The NLP based Recommendation Engine successfully addresses the limitations of traditional recommendation systems by focusing on personalized movie discovery based on sentiments, rather than relying solely on user ratings. Through the use of NLP algorithm, the app delivers highly relevant verdict aligned with the individual preferences of each user. The integration of PHP for backend development and Python for algorithm implementation provides a robust, scalable platform, while MySql ensures efficient handling of user and data.

The development process, guided by the Iterative Waterfall model, allowed for a structured yet flexible approach, facilitating the refinement of features like reviews, personalized recommendations, and user management. Each module was systematically tested to ensure the smooth operation of critical functionalities, resulting in a user-friendly application that enhances the movie exploration experience.

Overall, the recommendation engine provides users with an intuitive interface to discover, review, and engage with product that resonate with their preference, making the platform a valuable tool for seeking personalized recommendations.

## 5.2 Lesson learnt/Outcome

Throughout the development of the NLP based Recommendation Engine, several important lessons were learned, both technical and non-technical. From a technical perspective, the integration of Python and PHP highlighted the importance of choosing the right stack for scalable, dynamic applications. The project underscored the value of utilizing algorithms like NLP for personalized recommendations, reinforcing the importance of understanding data relationships in creating relevant user experiences.

The use of the Iterative Waterfall model facilitated a structured approach to development while allowing for iterative improvements at each stage. This model proved especially useful in handling feature implementation and testing in a solo development environment, demonstrating the need for continuous testing and refinement to ensure a reliable product.

Additionally, managing both the backend and frontend independently emphasized the importance of efficient API design and communication between systems, teaching the significance of modularity and code reuse. Handling real-world scenarios provided hands-on experience in tackling common challenges in web application development.

Overall, this project deepened the understanding of full-stack development, algorithm implementation, and effective project management in a solo setting, leading to the creation of a user-centric platform enhances user engagement.

## 5.3 Future Recommendations

Looking ahead, several enhancements and features could be introduced to further improve the functionality and user experience of the NLP based recommendation Engine:

* **Enhanced Personalization:** The recommendation system could be expanded to consider additional user behavior, such as watch history, time spent on reviews to further tailor recommendations based on more complex user patterns.
* **Social Features:** Integrating social features, such as allowing users to follow others, view their reviews, and share recommendations, could increase user engagement. Creating a community-based environment would enrich the app's experience by leveraging user interactions.
* **Mobile Application Development:** Expanding the platform to mobile devices with native iOS and Android applications would enhance accessibility and usability, allowing users to interact with the app from any device.

# Appendices

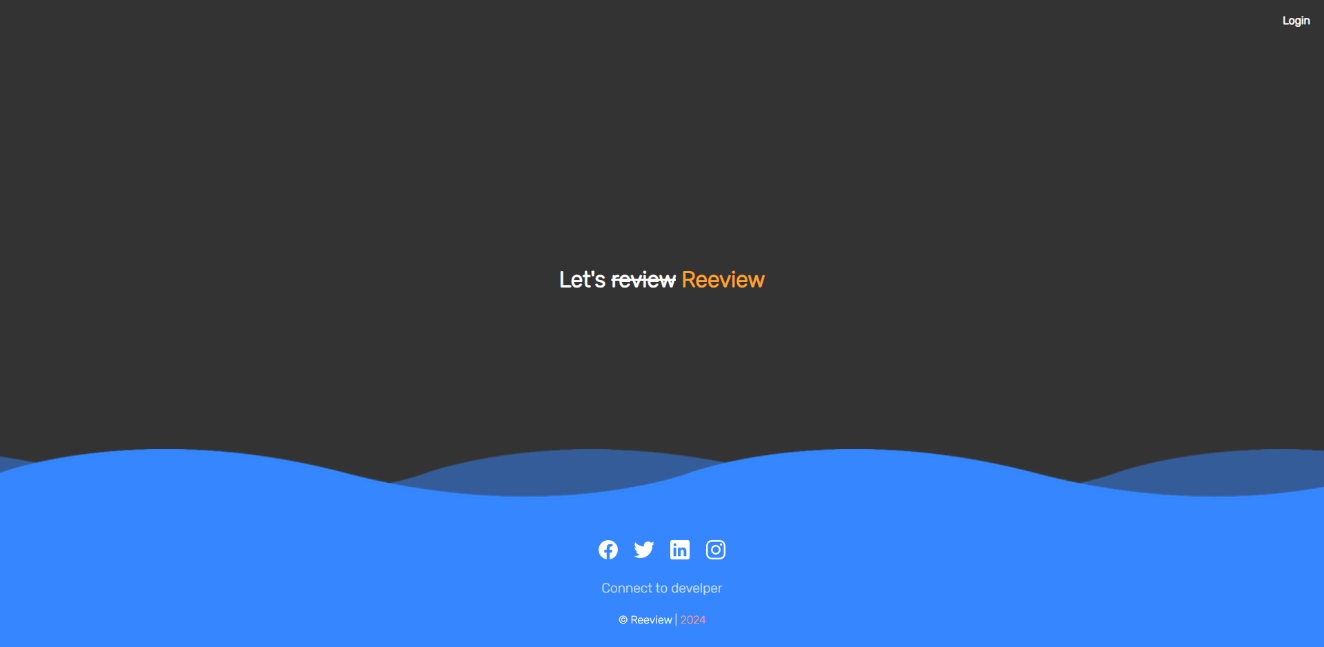
**Screenshots**

Figure : Landing Page

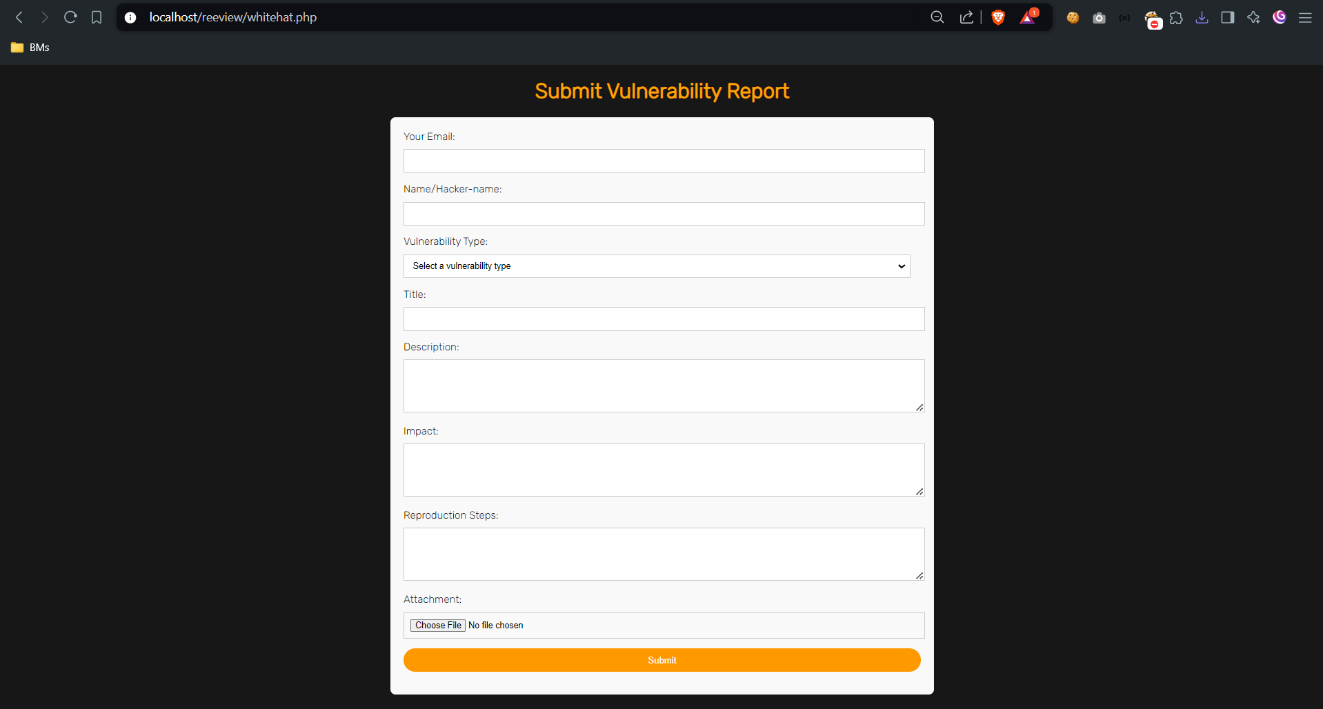


Figure 9: Bug reporting page

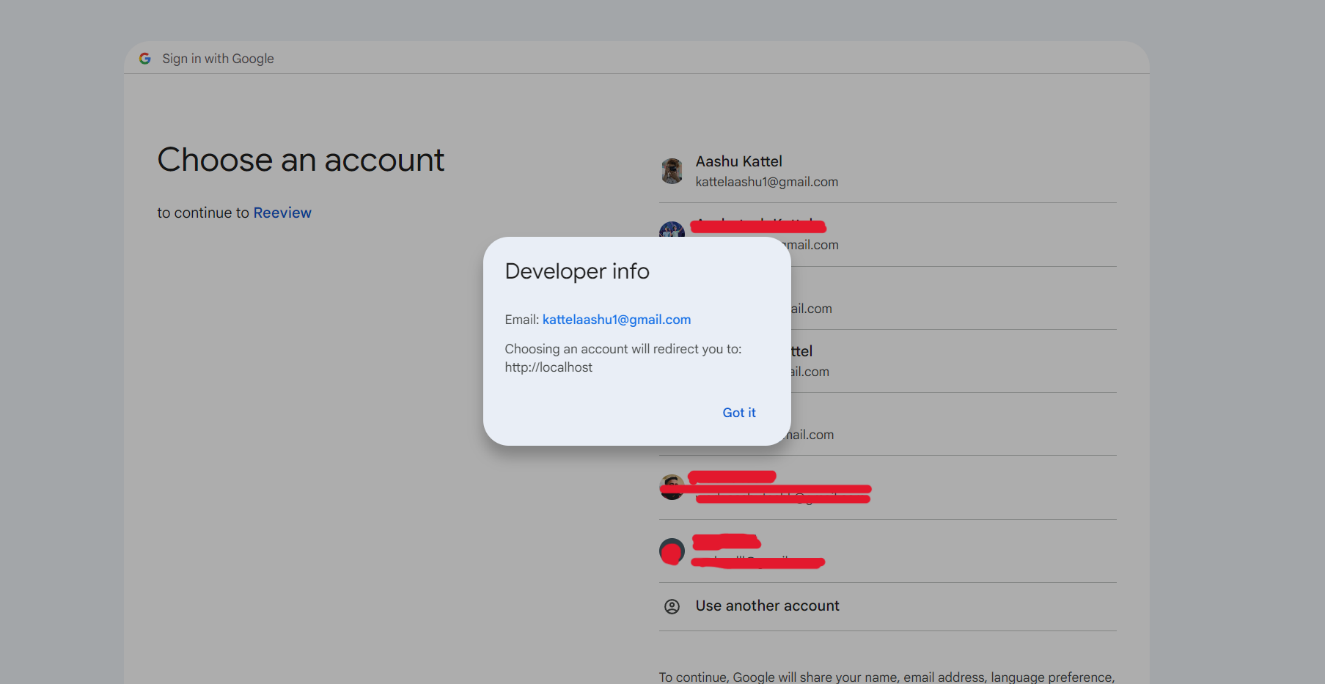


Figure : OAuth Login page

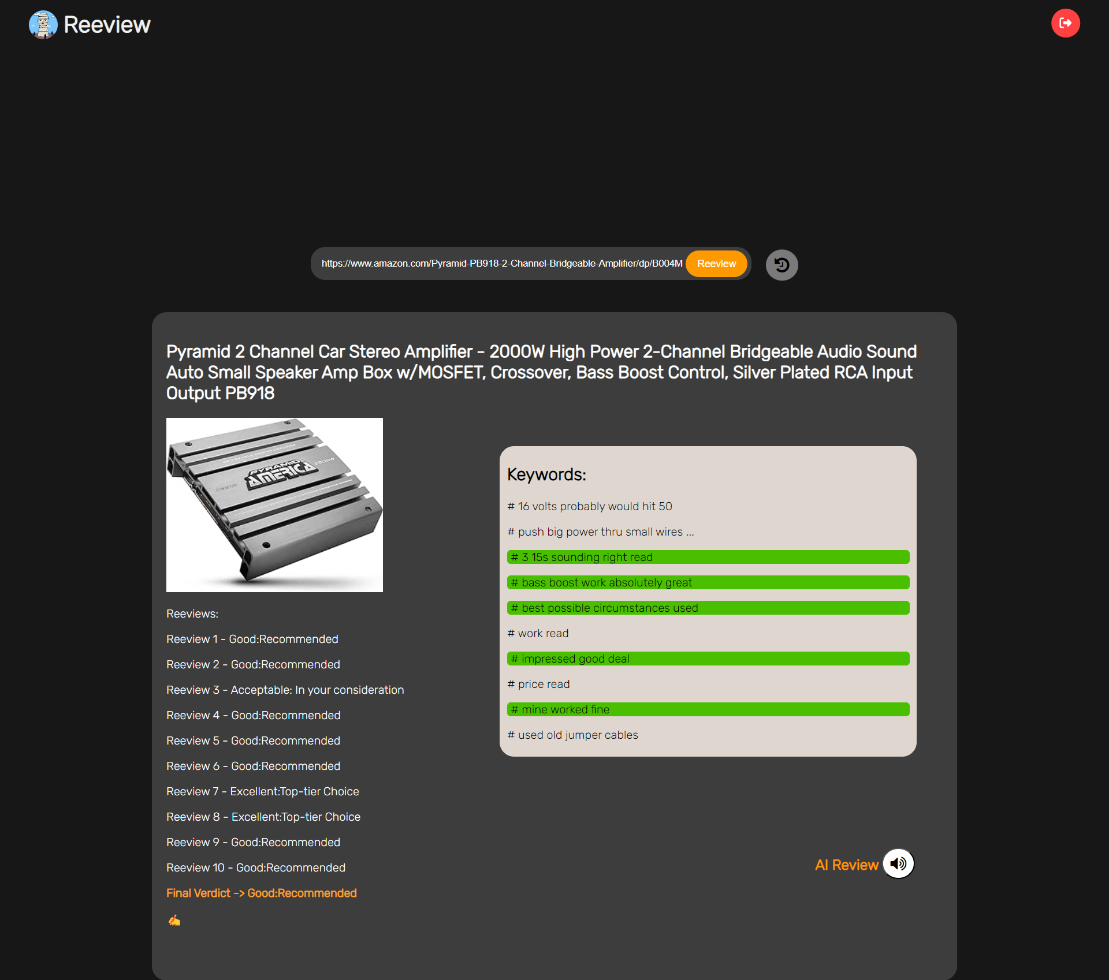


Figure : Main Dashboard

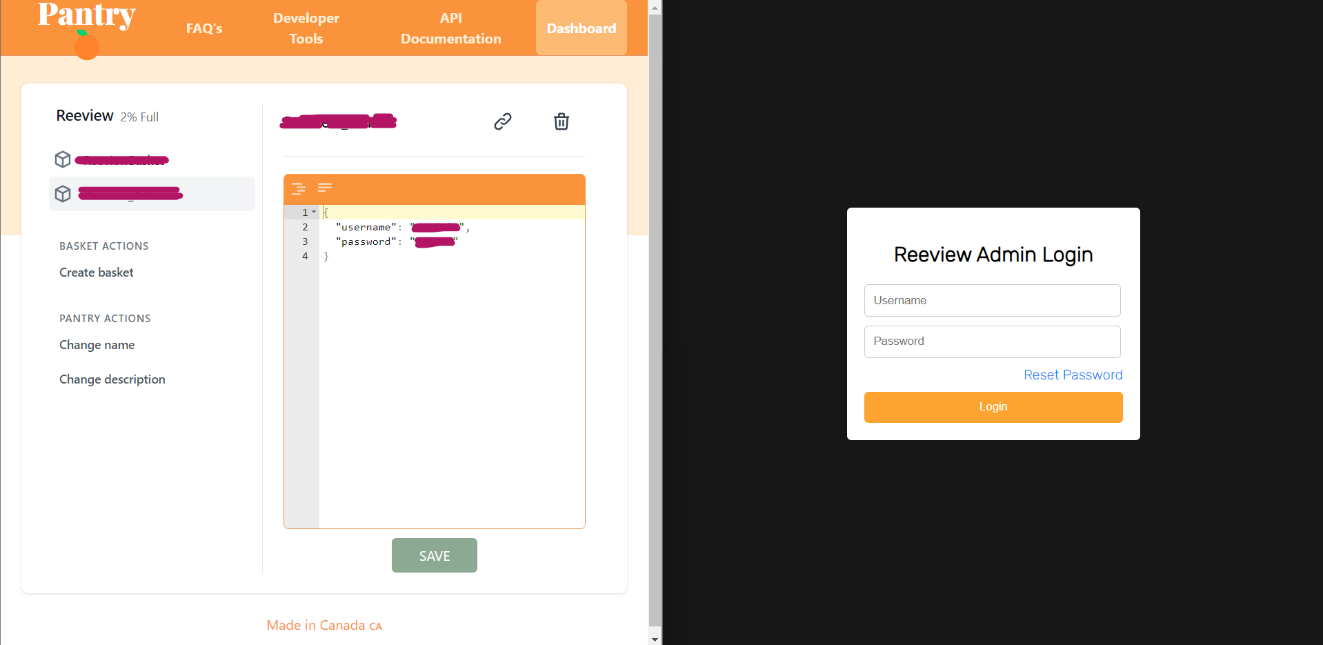


Figure : Admin Login using Pantry DB

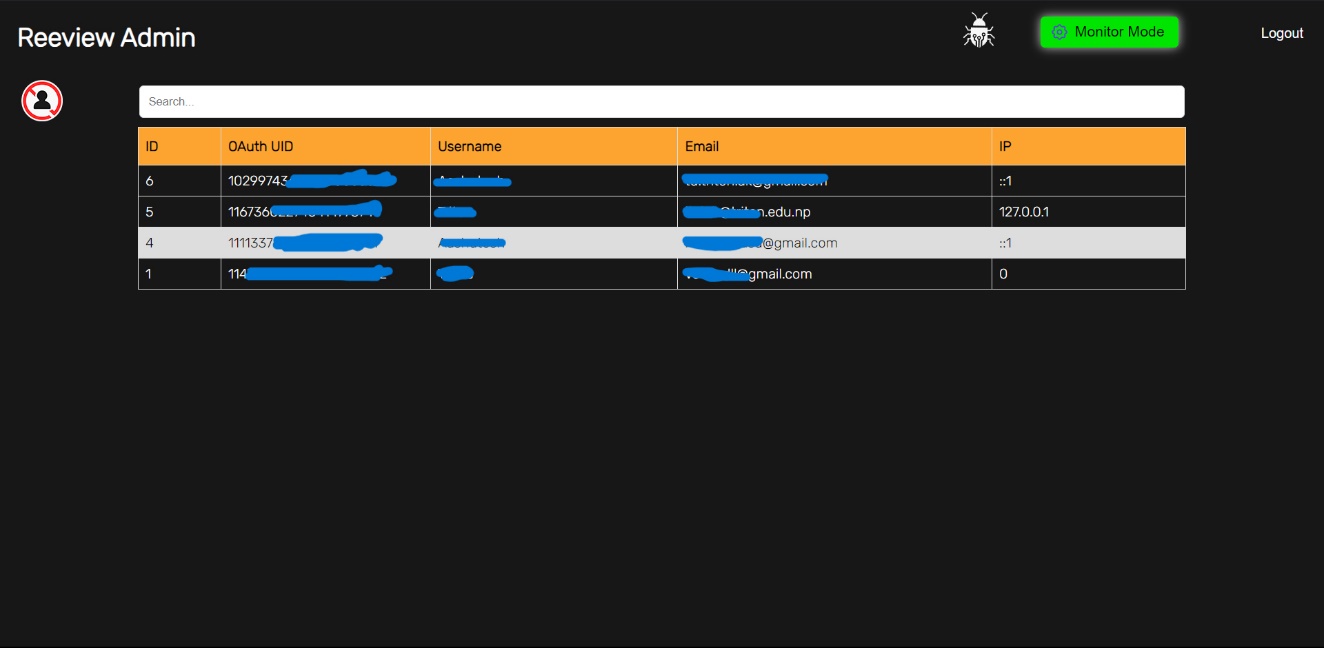


Figure : Admin Panel

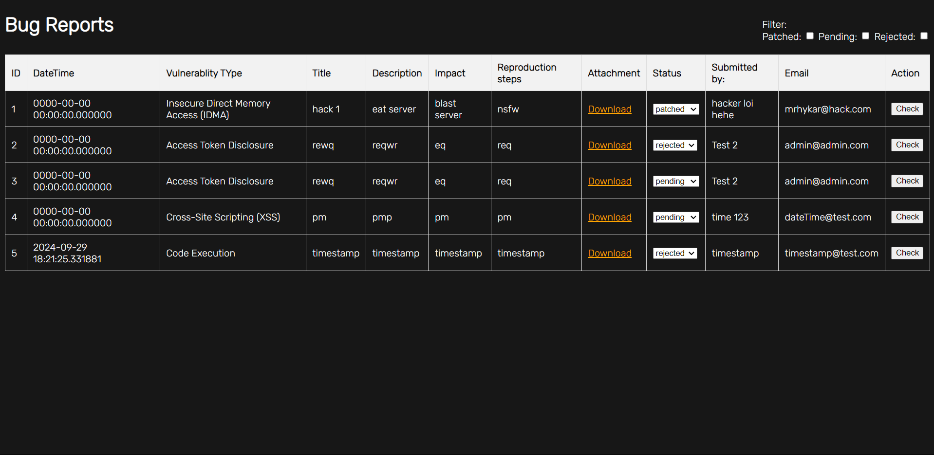
****

Figure : Bug reports

# C:\xampp\htdocs\reeview\ss\Appendices\history record.png

Figure 15: User search History

**Supervisor Log Sheet**

Triton Int’l College  
Koteshwore, Kathmandu  
Bachelors in Computer Application  
**Project Log-Sheet**

Semester: 6th  Project Name: NLP based Supervisor’s name: Yogesh Deo Recommendation Engine  
Student’s Name: Aashutosh Kattel

|  |  |  |  |
| --- | --- | --- | --- |
| **Date 2080** | **Topic/Issue Discussed** | **Comment/Next Target** | **Supervisor’s Signature** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# References

[1] G. Adomavicius and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 6, pp. 734-749, 2005. doi:10.1109/TKDE.2005.99.

[2] F. Ricci, L. Rokach, and B. Shapira, Recommender Systems Handbook. Springer, 2015.

[3] P. Lops, M. Gemmis, and G. Semeraro, "Content-Based Recommender Systems: Two Decades of Progress," in Recommender Systems Handbook, Springer, 2011, pp. 73-105.

[4] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," Computer Science Review, vol. 29, no. 4, pp. 895-924, 2009. doi: 10.1145/1454008.1454022.

[5] J. Bennett and S. Lanning, "The Netflix Prize," in Proceedings of the 2007 ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2007, pp. 3-6.

[6] G. Shani and A. Gunawardana, "Evaluating Recommendation Systems," in Recommender Systems Handbook, Springer, 2011, pp. 257-297.