

PUNJAB VERSE

MINOR PROJECT REPORT

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE
AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY

(Computer Science & Engineering)



Manjot Kaur(2203497)

Jasvir Kaur(2203474)

Jasmin Kaur Gahlot(2203470)

UNDER THE GUIDANCE OF

Prof. Palak Sood

Department of Computer Science and Engineering
GURU NANAK DEV ENGINEERING COLLEGE,
LUDHIANA, 141013

ABSTRACT

Proverbs are deeply rooted in cultural contexts and often lose their meaning when translated literally. Existing translation tools fail to provide contextually accurate translations of Punjabi proverbs, making it difficult for users to understand their true essence. This project aims to develop a Flask-based proverb Translation System that enables users to search for Punjabi proverbs and receive contextual English translations along with equivalent English idioms. The system incorporates a searchable database that categorizes idioms based on themes such as wisdom, humor, and relationships. Additionally, it features a fuzzy search mechanism to accommodate minor spelling errors and variations, ensuring that users can find relevant idioms even if they do not enter the exact wording.

The system is structured into multiple modules, including a user-friendly interface for easy navigation, a search module for retrieving matching proverbs, a database module to store and manage proverbs. The project utilizes Flask (Python) for the backend, SQLite for database management, and HTML, CSS for the frontend.

By offering a structured and efficient way to translate Punjabi proverbs, this system addresses the limitations of existing translation tools and enhances language learning and cross-cultural understanding. It serves as a valuable resource for students, researchers, and language enthusiasts, promoting better accessibility to linguistic heritage.

ACKNOWLEDGEMENT

We are highly grateful to the Dr. Sehijpal Singh, Principal, Guru Nanak Dev Engineering College (GNDEC), Ludhiana, for providing this opportunity to carry out the minor project work.

The constant guidance and encouragement received from Dr. Kiran Jyoti H.O.D. CSE Department, GNDEC Ludhiana has been of great help in carrying out the project work and is acknowledged with reverential thanks.

We would like to express a deep sense of gratitude and thanks profusely to Prof. Palak Sood, without her wise counsel and able guidance, it would have been impossible to complete the project in this manner.

We express gratitude to other faculty members of computer science and engineering department of GNDEC for their intellectual support throughout the course of this work.

Finally, we are indebted to all whosoever have contributed in this report work.

Jasmin Kaur Gahlot

Jasvir Kaur

Manjot Kaur

LIST OF FIGURES

Fig. No.	Figure Description	Page No.
2.1	Agile Model	15
3.1	System Architecture	18
3.2	Flowchart of proposed work	20
3.3	Sequence Diagram of proposed work	23
4.1	Test Case 1	32
4.2	Test Case 2	32
4.3	Test Case 3	33
4.4	Test Case 4	33
4.5	Test Case 5	34
4.6	Unit Testing	34
4.7	Integration Testing	35
4.8	Acceptance Testing	35
5.1	Searching Bar	40
5.2	About Us	40
5.3	Features	41
5.4	Steps	42
5.5	Home Page	43

5.6	User Interaction: Punjabi Phonetics	43
5.7	User Interaction: Romanized Phonetics	44
5.8	Keyword Searching: Gurmukhi	45
5.9	Keyword Searching: Romanized Punjabi	45
5.10	Error Message	46
5.11	Database	47

LIST OF TABLES

Table No.	Table Description	Page No.
4.1	Black Box Testing	31

TABLE OF CONTENT

Contents	Page No.
Abstract	<i>i</i>
Acknowledgement	<i>ii</i>
List of Figures	<i>iii</i>
List of Tables	<i>iv</i>
Table of Contents	
Chapter 1: Introduction	1
Chapter 2: Requirement Analysis and System Specification	9
Chapter 3: System Design	17
Chapter 4: Implementation and Testing	27
Chapter 5: Results and Discussions	38
Chapter 6: Conclusion and Future Scope	48
References	51

Chapter 1 Introduction

1.1 Introduction to Project

Language stands as a pillar of human culture serving as a way to convey thoughts, spread knowledge, and hand down traditions. Every language contains a wealth of sayings that hold deep meanings, insights, and shared wisdom. Punjabi, a lively and culturally diverse language that millions speak around the world, has a huge set of sayings that give us a peek into the customs, values, and beliefs of Punjabi speakers. These sayings, which come from people's everyday lives and experiences, have been handed down from one generation to the next keeping the heart of Punjabi heritage alive. Yet, as we focus more on bringing the world together and talking across cultures, it's become more important to translate and explain these sayings to a broader audience.

Proverbs often present challenges during translation because their meanings are not always literal and are deeply tied to the cultural context in which they originate. A direct word-for-word translation frequently fails to convey the true intent, humor, or wisdom embedded in the proverb. The Proposed work addresses the gap between languages by developing a comprehensive system for translating Punjabi proverbs into English while maintaining most of the accuracy, contextual relevance, and cultural depth.

The primary objective of this project is to compile a robust dataset of Punjabi proverbs, complete with their contextual English equivalents and English contextual meaning. By compiling a collection of proverbs, this project aims to preserve and digitize the cultural wisdom of the Punjabi language while making it accessible to a global audience.

To enhance usability, this project will incorporate a user-friendly interface designed for simplicity and functionality. The interface will allow users to easily input Punjabi proverbs in their original script and retrieve translations in English with proper formatting and readability.

This design ensures that the system is inclusive and accessible, regardless of the user's linguistic background.

A critical aspect of this project is ensuring the linguistic accuracy of translations. Proverbs are not merely phrases but can be used to represent cultural and any inaccuracy in translation risks diminishing their essence.

This project is significant not only for its linguistic and cultural contributions but also for its potential to bridge gaps between languages and foster cross-cultural understanding. In today's interconnected world, effective communication and cultural appreciation are more important than ever. By providing a reliable resource for translating Punjabi proverbs into English, this project will enable people from diverse backgrounds to understand and appreciate Punjabi culture, thereby promoting cross-cultural dialogue and mutual respect.

Moreover, the project addresses the urgent need for digital tools that preserve endangered linguistic and cultural artifacts. As younger generations increasingly adopt dominant global languages like English, there is a risk of losing the richness of regional languages and their associated cultural elements, such as proverbs. This project will serve as a digital repository, safeguarding Punjabi proverbs for future generations while integrating them into the global digital landscape. By combining the traditional wisdom of Punjabi proverbs with modern technological solutions, the project ensures that this vital aspect of Punjabi culture remains relevant and accessible in the digital age.

In conclusion, this project is a multifaceted effort to preserve, digitize, and promote Punjabi proverbs while facilitating their accurate translation into English. It not only aims to provide a practical tool for translation but also seeks to celebrate and share the cultural richness of Punjabi heritage with a global audience. Through its comprehensive dataset, user-friendly interface, and commitment to linguistic accuracy, this project aspires to create a bridge

between languages and cultures, enabling deeper appreciation and understanding of Punjabi proverbs in an increasingly interconnected world.

1.2 Project Category

The proposed work belongs to Application Based project category.

Application based projects focus on designing, developing and delivering a working product or system that users can interact with. Application-based projects involve the creation of practical software or systems that serve a specific purpose and are intended for real-world use. These projects focus on the full development lifecycle, including planning, designing user-friendly interfaces, coding the core functionality, and testing to ensure reliability. The end goal is to deliver a working product that users can interact with, such as a website, mobile app, or desktop software, often solving a particular problem or fulfilling a specific need. These types of projects emphasize functionality, usability, and user experience, and are commonly used in academic, industrial, or entrepreneurial settings to demonstrate applied technical skills.

1.3 Problem Formulation

In the context of the Punjabi Proverb Translation System, the goal is to create an effective and user-friendly tool for translating Punjabi proverbs into English, while maintaining the cultural and contextual meaning. Users often face difficulties in translating Punjabi proverbs, particularly due to the complexities of language and script. Additionally, there is a need to accommodate various user inputs, including Romanized Punjabi, which is commonly used by speakers who are not familiar with the Gurmukhi script. The problem can be formulated as follows:

- 1) **Language Barrier:** Punjabi proverbs are rich in cultural meaning and often do not have direct English equivalents. Translating them accurately is a challenge, as the meaning needs to be conveyed, not just the words.

- 2) **Romanized Input:** Many users might be unfamiliar with the Gurmukhi script but are familiar with Romanized Punjabi. There is a need to support Romanized Punjabi input to make the system more accessible.
- 3) **Spellings and Typo Variations:** Users may input proverbs with spelling mistakes or slight variations, and the system should still be able to find the most accurate match. This requires handling fuzzy matching to find relevant proverbs.
- 4) **User Experience:** The system should provide a user-friendly interface for searching, viewing the proverb, and receiving suggestions for potential matches even when the input is **imprecise**.

1.4 Recognition of Need

The proposed has following reason for recognition of needs:

1. Language Barrier and Lack of Accurate Translations:

Punjabi proverbs are deeply rooted in the culture and traditions of the Punjabi-speaking community. Many proverbs don't have direct English equivalents, which makes it difficult for non-Punjabi speakers to understand their true meaning. English speakers or learners of Punjabi need a tool that translates both the literal and contextual meanings of these proverbs to bridge the language gap.

2. Romanized Punjabi:

In many cases, users are more familiar with Romanized Punjabi rather than the Gurmukhi script. This is particularly common among younger generations or those who have limited exposure to the Gurmukhi script. A Romanized input search feature is needed to make the tool more inclusive and user-friendly.

3. Typos and Input Variations:

Users may enter incorrect spellings or slightly varied versions of a proverb, especially in Romanized form. Traditional search systems may fail to recognize these variations, leading to poor user experience. The fuzzy matching feature is essential to handle these cases, ensuring that users are still able to find the correct proverbs even if their input is incorrect.

4. Accessibility and User Experience:

The user interface for proverb search and translation must be intuitive and user friendly. Users should not need to be linguistically proficient in Punjabi to access the meanings of proverbs. There is a need for a web-based solution or platform where users can easily input a proverb, view the translation, and understand the contextual meaning in English.

1.5 Existing System

1. Google Translate

Google Translate allows users to translate text between multiple languages, including Punjabi. Features:

- 1) Language translation between Punjabi and English.
- 2) Suggestions and context-based results for common idioms and proverbs.
- 3) Romanized Punjabi support.
- 4) Real-time translations and suggestions.

Google translator is not accurate while translating proverbs. It provides literal meaning of proverbs.

2. Punjabi to English Idioms Dictionary (Mobile Apps)

There are several mobile apps that function as Punjabi-to-English idiom dictionaries, offering translations of common idioms, proverbs, and sayings.

Features:

- 1) Dictionary of Punjabi proverbs with English meanings.
- 2) Option to search for idioms and their translations.

These system typically lack advanced fuzzy matching features.

1.6 Objectives

The proposed work has following objectives:

1. To compile a comprehensive dataset of Punjabi proverbs with their contextual English meanings.
2. To create a user-friendly interface for translating Punjabi proverbs into English.
3. To ensure language accuracy and interface functionality.

1.7 Proposed System

The proposed work addresses the gap between languages by developing a comprehensive system for translating Punjabi proverbs into English while maintaining most of the accuracy, contextual relevance, and cultural depth.

1.7.1 Key features

The proposed work has following key features:

- 1) **Romanized and Punjabi Script Input:** The system allows users to input Punjabi proverbs in both Romanized as well as Gurmukhi Script.
- 2) **Fuzzy Search:** Fuzzy searching is used to match proverbs partially. It will ensure that even if the user makes mistakes in spelling, the system can still return output and provides right spelling also. As for Romanized Punjabi, every user use different spelling. Fuzzy searching also deals with this problem also.
- 3) **Contextual meaning and English equivalent:** The system provides English contextual meaning as well as most accurate English equivalent if exist with Punjabi proverb in Gurmukhi as well as Romanized Punjabi.

- 4) **User friendly interface:** the proposed work provide a user friendly interface that allows user to search and display the output on screen. The web page is responsive so it can use on phone as well as desktop.

1.7.2 Advantages

The proposed work has following advantages:

- 1) **Cross language understanding:** The proposed system bridge the gap between Punjabi and English, and help user to understand Punjabi proverbs.
- 2) **Enhancing user experience:** system provides Romanized as well as Punjabi script for different users. The system ensure that even if there is variation in the spelling of input still system will provides output.
- 3) **Cultural preservation:** The system help to preserve and share Punjabi cultural expression by making proverbs accessible to people who may not familiar with language.

1.8 Unique Features of the Proposed System

The proposed work has following unique features of the proposed work:

1. **Contextual Meaning:** The proposed work provides the contextual meaning of Punjabi proverbs instead of their literal translations.
2. **English equivalent of proverb:** Finding the English equivalent of any Punjabi proverb is not easy and time wasting process. The database of the project contains Punjabi proverbs with their closest English equivalents if exist.
3. **Centralized database:** The database of the project stores a structured collection of Punjabi proverbs with their contextual meaning and English equivalent if exist. In the dataset the data is organized and easily retrievable.

4. **Web-Based Interface:** The proposed work is a web application which translate with searching from database rather than using any AI, maintain the accuracy and humor of the language.
5. **Efficient Input:** The proposed system provide the efficient input. User can search with in script Punjabi typing, phonetics Punjabi typing as well as Romanized Punjabi.

Chapter 2: Requirement Analysis and System Specification

2.1 Feasibility Study

A feasibility study is an assessment of the practicality of a proposed plan or project. It analyzes the viability of a project to determine whether the project or venture is likely to succeed. The study is also designed to identify potential issues and problems that could arise while pursuing the project.

2.1.1 Technical Feasibility

Technical feasibility is used to ensure that the project can be developed using existing technologies.

- 1) The system use widely available and well supported technologies such as flask, sqlite, html/ css and javascript.
- 2) The project does not require specialized hardware, but only internet access is sufficient.
- 3) Required libraries and tools are open source and freely available.

2.1.2 Economic Feasibility

Economic feasibility evaluate whether the project is financially affordable or not.

- 1) The development cost is minimal because all technologies and tools are free and open-sourced
- 2) There is no requirement of an expensive server or paid AI.
- 3) Future expansion, like hosting online or creating a mobile application can be done low cost.

2.1.3 Operational Feasibility

Operational feasibility used to check the system's functionality and correctness.

- 1) The system is easy to use with user friendly interface, allowing users to search for Punjabi proverbs without requiring technical knowledge.

- 2) It supports Romanized Punjabi input and gives search suggestions, making it user-friendly for different users.

2.2 Software Requirement Specification

2.2.1 Data Requirement

The proposed work relies on a well-defined, organized and accurate dataset. The following type of data is required for proposed work:

- 1) **Punjabi Proverbs:** The traditional Punjabi proverbs were required for proposed work which are the main entries that user search to get translation. The Punjabi proverbs are taken from Punjabi grammar book of PSEB, tenth class, which ensure accurate meaning of the Punjabi proverbs.
- 2) **Proverbs in Romanized Punjabi:** The Romanized Punjabi version of each Punjabi proverb required so that user can search with Romanized Punjabi version.
- 3) **English contextual meaning:** The English contextual meaning of Punjabi proverbs are required so that user can get most accurate answer. Many types of books and internet is used to find English contextual meaning of Punjabi proverbs.
- 4) **English equivalent:** The English equivalent of Punjabi proverb if exists so that user can easily relate Punjabi proverb.

2.2.2 Functional Requirement

Functional requirement define the core features and behavior of the system that meet user's need.

- 1) **User Input Handling:** The system should allow users to provide input in either Gurmukhi or Romanized Punjabi. The system should handle incomplete or slightly misspelled input using fuzzy search.

- 2) **Search Functionality:** The system should search the database for matching Punjabi proverbs based on user input.
- 3) **Fuzzy Matching and Suggestions:** The system should provide real-time suggestions to users for partial searching. Suggestions should appear based on partial matching and threshold similarity scores.
- 4) **Error Handling:** If no matching result is found, the system should display an appropriate message.
- 5) **User Interface:** The system should provide a user-friendly and responsive interface for users.

2.2.3 Performance Requirement

The performance requirements define the performance of the system. It define how fast, reliable, and efficient the system need to be while delivering services to the users.

For the Proposed work, the key performance requirements are:

1. **Response Time:** The system should retrieve and display search results within 2-3 seconds after the user submits a query. Suggestions should appear instantly as the user types.
2. **Search Efficiency:** The fuzzy search algorithm should efficiently handle partial matching without causing noticeable delays. The system should prioritize relevant and highly similar results first.
3. **Load Handling:** The system should be able to handle multiple users performing searches simultaneously without crashing or becoming slow.
4. **Database Performance:** The SQLite database queries should execute quickly and efficiently with optimized fetching

2.2.4 Dependability Requirement

Dependability requirements define reliability, consistency, and fault-tolerant of the system

.For the proposed work, the key dependability requirements are:

1. **System Reliability:** The system should consistently provide accurate search results and correct translations for all queries.
2. **Fault Tolerance:** The system should be able to handle unexpected errors or crashes gracefully. If a problem occurs, it should display a friendly error message and prevent the application from freezing or crashing.
3. **Backup and Recovery:** The proverb database should be backed up regularly to avoid data loss in case of system failure.
4. **Consistency:** The search results and suggestions provided by the system should be consistent across different users, devices, and sessions. Data entries in the database should maintain consistency, with no contradictory information being presented.
5. **High Availability:** The system should be designed to ensure high availability, ensuring minimal downtime even in the event of minor errors or disruptions.
6. **Data Integrity:** The system should guarantee the integrity of data in the database. No data should be lost or corrupted due to user actions or system failures.

2.2.5 Maintainability Requirement

Maintainability requirements define updating and improving the system over time.

For the proposed work, the maintainability requirements are:

1. **Easy Database Updates:** The system should allow easy addition, editing, or deletion of proverbs in the SQLite database without needing major changes in code. The database structure must remain simple and organized for future expansion.

2. **Modular Code:** The Flask backend should have separated functions with different tasks like data retrieval, searching, and suggesting. If any part of the logic needs to be updated it should require changes only in specific functions, not the entire system.
3. **Scalability:** The system should support scaling easily with minimum effort.
4. **Documentation:** The project should be well-documented, with Database schema. Code comments explaining major parts of the code.
5. **Error Logging:** Proper error messages and logs should be maintained to make debugging and maintenance faster in case of future problems.
6. **Updating the UI:** The frontend (HTML/CSS) should be organized so that design improvements or adding new pages can be done easily without affecting the search functionality.

2.2.6 Security Requirement

Security requirements ensure that the system, its data, and user interactions are safe from misuse, errors, or malicious actions. For proposed work the main security requirements are:

1. **Data Integrity:** The system should ensure that the proverb database remains accurate and consistent. Unauthorized users should not be able to modify, delete, or corrupt proverb entries.
2. **Input Validation:** All user inputs should be properly validated and sanitized to prevent SQL Injection attacks. Only clean and safe data should be processed by the server.
3. **Database Protection:** The SQLite database file should be securely stored. Only authorized access should be allowed.
4. **Server Security:** Flask server should be properly configured to handle HTTP requests securely.

5. Error Handling: In case of errors, the system must display generic error messages.

The system should not expose internal system information like database structure, code errors, or server paths to the user.

2.2.7 Look and Feel Requirement

The look and feel requirements focus on the system's visual design, user interface, and overall user experience to ensure the platform is intuitive, appealing, and user-friendly.

- 1. Simple and Clean Interface:** The system should provide a clean and clutter-free interface that allows users to focus on the search functionality without distractions. The layout should be simple, with clearly labeled sections for searching, displaying results, and showing suggestions.
- 2. Responsive Design:** The platform should be mobile-friendly and should automatically adjust to various screen sizes .Buttons and forms should be easy to interact with on both desktop and mobile devices.
- 3. Interactive Elements:** The suggestions for Romanized Punjabi input should appear dynamically as user's type, creating a seamless, interactive experience. Search results should be clearly distinguishable, with clickable proverbs leading to further details or translations.
- 4. Language Accessibility:** The system should ensure that Punjabi script and Romanized input are both displayed clearly.
- 5. Feedback and Error Messages:** When users make an error , the system must provide a friendly, non-technical error message

2.3 SDLC Model

The proposed system is following Agile SDLC model.

2.3.1 Agile SDLC Model:

Agile SDLC model is a combination of iterative and incremental process models with focus on process adaptability and customer satisfaction by rapid delivery of working software product. Agile Methods break the product into small incremental builds. These builds are provided in iterations.

2.3.2 Steps in Agile Model:

The agile model has following steps:

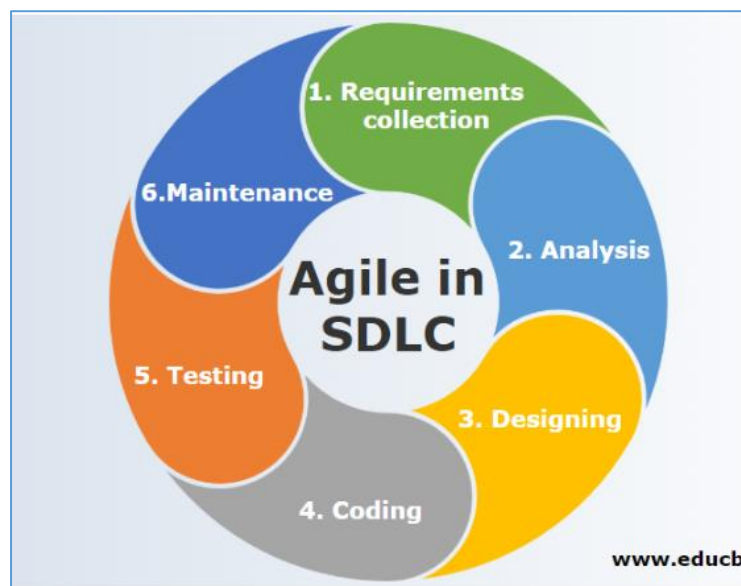


figure 2.1 Agile Model

1) Requirement Gathering

Development team gather the requirement of the project and plan the schedule, time and effort to build the project. Based on the gather requirement developers evaluate technical and economic feasibility.

2) Analysis the Requirements

Development team use user-flow-diagram or high-level UML Diagrams to show the working of the new features and show how they will apply to the existing software. Wire framing and designing user interfaces are done in this phase.

3) Construction / Iteration

Development team members start working on project, which aims to deploy a working product. Each cycle typically consist between 1-4 weeks, and at the end, the team delivers a working version of the software.

4) Testing / Quality Assurance

Testing involves Unit Testing, Integration Testing, and System Testing. Unit testing is used to test individual blocks (units) of code. Integration testing is used to identify and resolve any issues that may arise when different units of the software are combined. Goal is to ensure that the software meets the requirements of the users and that it works correctly in all possible scenarios.

5) Deployment

Development team will deploy the working project to end users. Once an iteration is finished and fully tested, the software is ready to be released to the end users. In Agile, deployment isn't a one-time event—it's an ongoing process. Updates and improvements are rolled out regularly, making sure the software keeps evolving and getting better with each release.

6) Feedback

The team receives feedback about the product and works on correcting bugs based on feedback provided by the customer.

Chapter 3 System Design

3.1 Design Approach

3.1.1 Functional Oriented

The proposed system is using functional based design even though it may align with some OOP concepts like modularity and separation of concern.

3.2 Detail Design

3.2.1 System Architecture of Proposed Work

System architecture refers to the overall structure and design of a software system, including its components, modules, data flow, and interactions between different parts. It serves as a blueprint that defines how the system functions and how different components work together. The proposed work follow a Three-Tier Architecture ensuring the balance between performance, scalability and security:

1. Presentation Tier:

The presentation tier is the user interface and communication layer of the application, where the end user interacts with the application.

In the Proposed work Presentation tier provides the search form to the user and provides the required output. It provides the responsive design that can run on desktop as well as mobile.

- a. **Function:** The main purpose of Presentation layer is to take input of Punjabi proverb from user and display English contextual meaning and English equivalent of searched Punjabi Proverb.
- b. **Technology used:** HTML and CSS is used to design the responsive and user-friendly interface in proposed work.

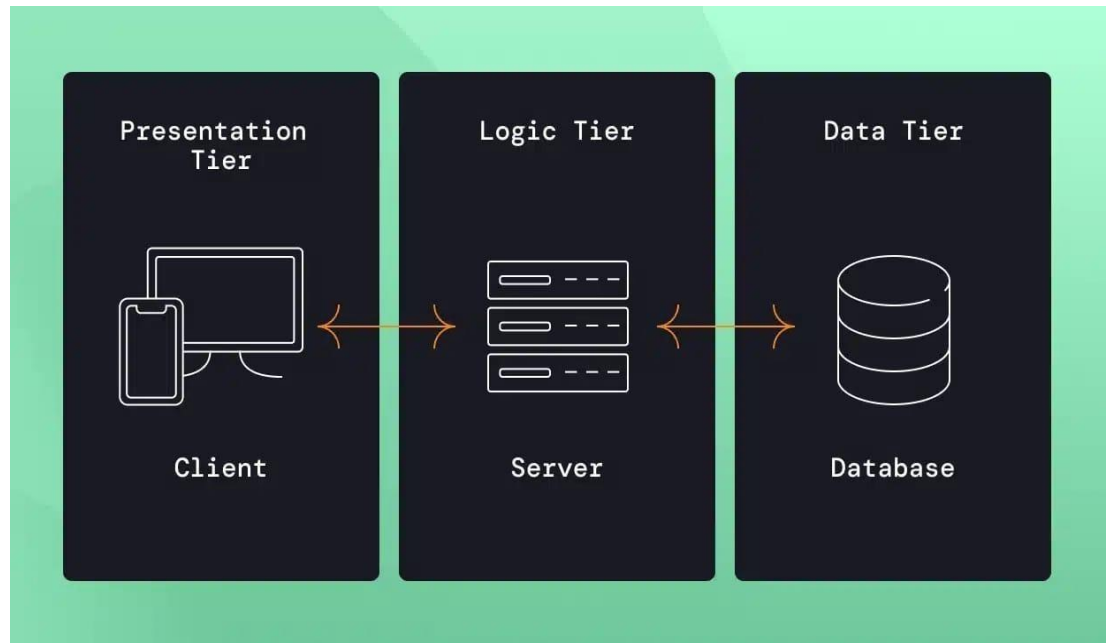


figure 3.1 System Architecture

2. Logic Tier:

The application tier, also known as the logic tier or middle tier, is the heart of the application. In this tier, information that is collected in the presentation tier is processed. This tier interact with data tier to process the query of a user to provide the result.

In the proposed work Logic Tier takes the input as Punjabi proverb from presentation layer and interact with data tier to match the input with database entries to fetch the English contextual meaning of input Punjabi proverb.

- a. **Function:** The main purpose of the Application layer is to controls system logic and database interactions.
- b. **Technology used:** FLASK will be used to handle request and logic and flask will be used to connect FLASK to the database.

3. Data Tier

The data tier, sometimes called database tier, data access tier or back-end, is where the information that is processed by the application is stored and managed. The logic tier interact with data tier to search the data entries in database.

In the Proposed work Data Tier stores a structured Punjabi proverb data with their English contextual meaning and English equivalent if possible.

- a. **Function:** The main purpose of the data tier is to stores the Punjabi proverb with their English contextual meaning and manage proverb translation.
- b. **Technology used:** SQLite is used to store dataset of Punjabi proverbs and their English contextual meaning.

3.2.2 Flow Chart of Proposed Work

Flowchart is a diagrammatic representation of sequence of logical steps of a system. Flowcharts use simple geometric shapes to depict processes and arrows to show relationships and process or data flow.

The flow chart of the proposed work is designed to ensure fast searching and providing accurate English contextual meaning of the searched Punjabi proverb while maintaining the reliability of the language. Below is a step by step breakdown of the flow chart based on reference image:

1. User input:

- i. The user enter the Punjabi proverb partially or fully in the search form.

2. Flask receives the request:

- i. The input is sent to the FLASK backend using an HTTP request with POST or GET method.
- ii. The FLASK extracts the search query and prepare it for database processing.

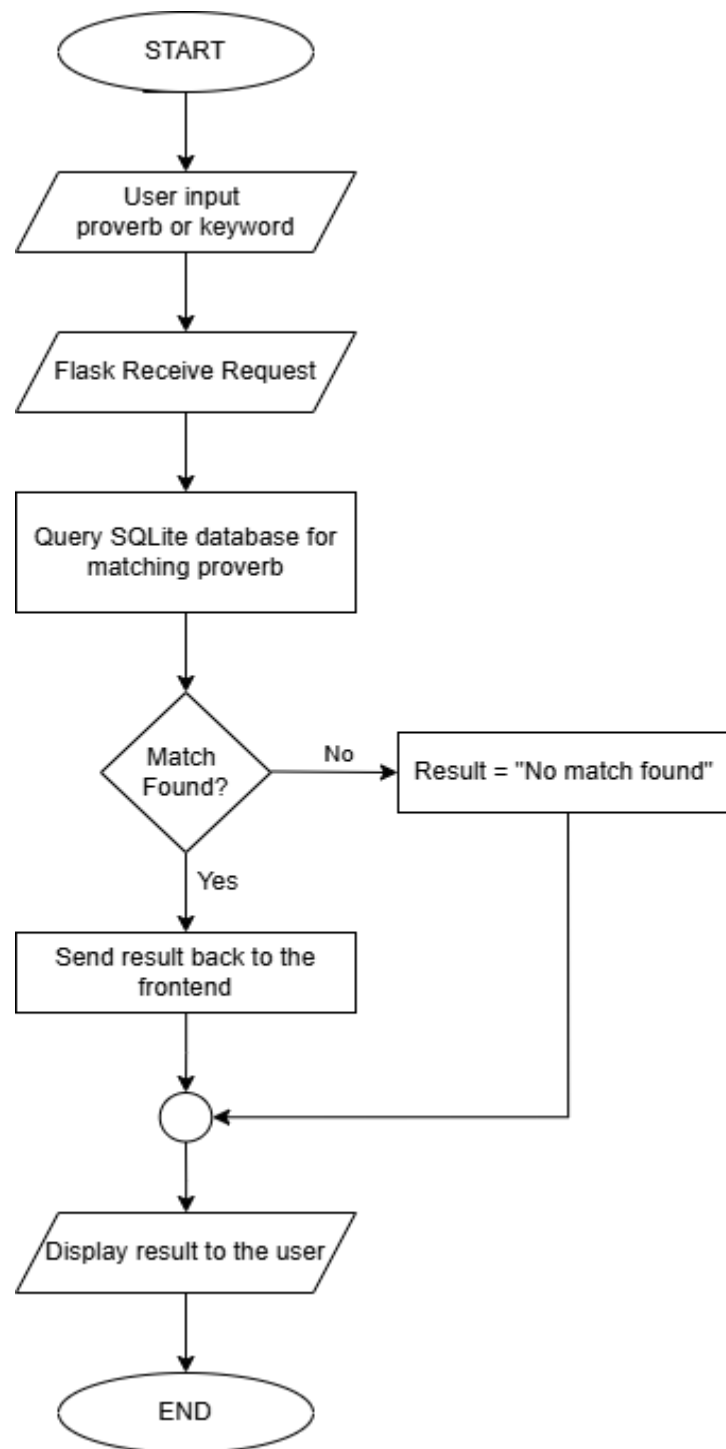


Figure3.2 Flow chart of proposed work

3. SQLite query

- i. The system queries the SQLite database to find matching entries with the searched input.
- ii. The database contains Punjabi proverbs, their contextual meaning and English equivalent.

4. Matching the query

- i. The system checks whether a matching proverb exists in the database.
- ii. If the match found the system retrieves the proverb, its English equivalent and contextual meaning.
- iii. If match not found the system display a message for no matching proverb found.

5. Display Result

- i. The retrieved proverb and its contextual meaning are send back to the frontend
- ii. The user sees the translated proverb on their screen.

3.2.3 Sequence Diagram of Proposed Work

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

A Sequence Diagram of Proposed Work shows the interaction between different components of the system.

3.2.1.1 The different Components in the sequence diagram:

Below are the different component of Sequence Diagram of Proposed work based on the reference image.

1. **User:** The user is the person that interact with the system to search the contextual meaning of Punjabi proverb.
2. **Web interface (front end):** Web interface is a HTML form where user enter the proverb and that interact with the backend to process the query.
3. **FLASK (back end):** The back end is used to process user request and communicate with the database.
4. **SQLite database:** The SQLite database is used to store and retrieve the proverb translation.

3.2.1.2 Flow of sequence diagram:

Below given is the step by step breakdown of the sequence diagram of the proposed work based on the reference image:

1. Enter the proverb:

- i. User enter the Punjabi proverb as an input in the search form.
- ii. User can enter complete proverb or can search partially also.

2. Send request:

- i. Front end receive the input from the user.
- ii. Front end sends the request to the backend.

3. Query Database:

- i. FLASK backend receive the request of front end.
- ii. The backend process the request and send the queries to the Database.

4. Return data:

- i. The database receive the queries from back end.
- ii. Database search for a match with input.

- iii. If database founds the match then it return the contextual meaning and equivalent of the Punjabi proverb.
- iv. If database can't found the match then it return the message "no result found".

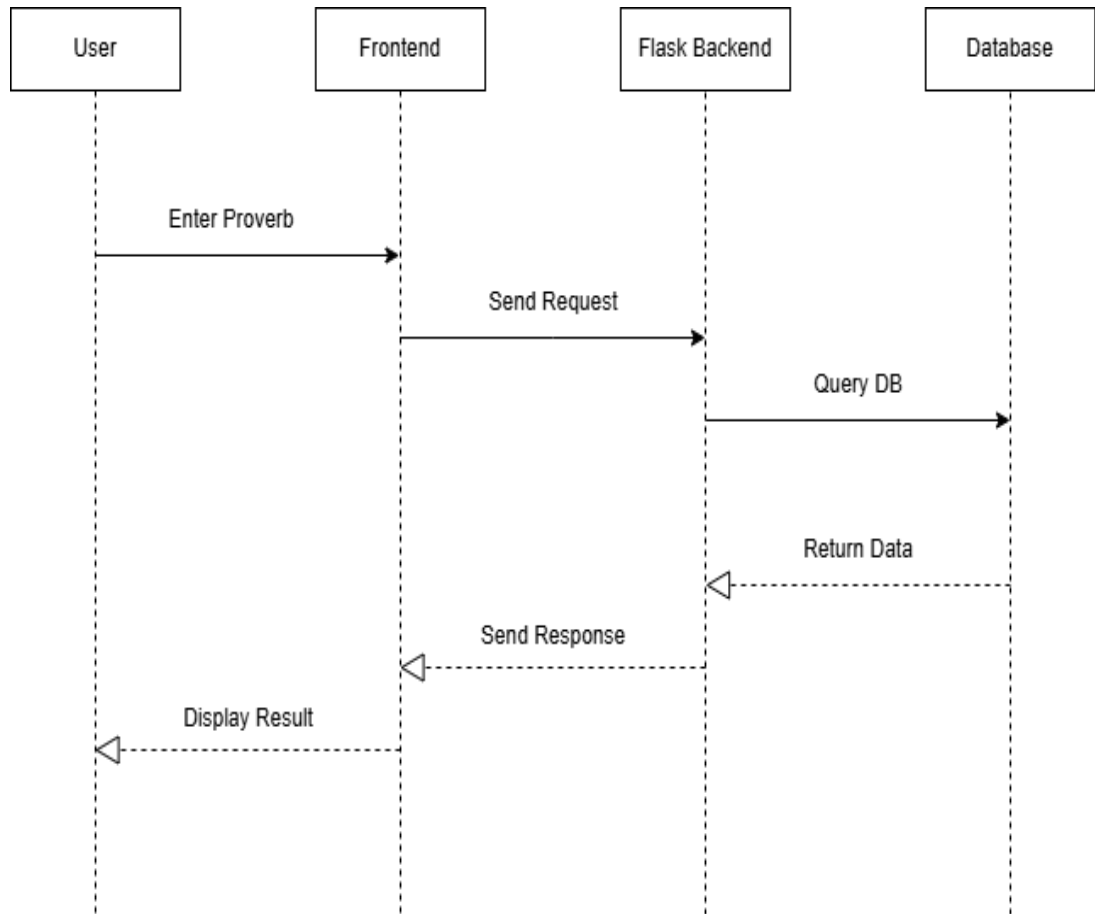


Figure 3.3 Sequence diagram of proposed work

5. Send response:

- i. Backend receive the response of the queries from the database.
- ii. Backend sends the response back to the frontend.

6. Display result:

- i. Front end display the result of searching of the user.

3.3 User Interface Design

1. Homepage layout:

The homepage contains search box where user can type a Punjabi proverb in Romanized or Punjabi inscript. It contains a search button to submit input to system. If the results are found, output is display on same page but if results are not found system provides error message.

2. Suggestion feature:

The homepage provides suggestion features for user. As user start typing, system provides suggestion list that match with input keyword based on Punjabi and Romanized Punjabi. This suggestion feature is auto suggest dropdown appear under searching box.

3. Display of Search result:

After user provides input to system, results are display in the screen which includes Punjabi idiom, Romanized Punjabi, English contextual meaning and English equivalent if input is correct but if input is wrong system provides error message.

3.4 Methodology

Methodology of the proposed system is given below:

1) Database creation

For proposed work database is created which has 200 punjabi proverbs. The database contains “Proverbs” table which contains:

- a) **Punjabi Idiom:** Punjabi idiom column contains 200 Punjabi idioms with ID.
- b) **English contextual Meaning:** This column contains the English Contextual meaning of Punjabi proverbs

- c) **English Equivalence:** This column contains English equivalence of Punjabi Proverbs if exist.
- d) **Romanized Punjabi:** This column contains Romanized Punjabi for Punjabi Proverbs
- e) **Technology used:** SQLite is used to store proverbs as it is lightweight, fast and easy to set up for 200 proverbs. It is compatible with flask also.

2) Backend development

In the backend of the proposed work a connection is establish with database and search function is implement to get English contextual meanings.

A. **Technology used:** Flask is used to create back end of the proposed work to handle server side logic.

B. Task of backend:

- a) **Connection with database:** flask create a connection with database for searching function.
- b) **Data retrieval:** Backend fetch proverbs from SQLite database using SQL queries. It use fuzzy search to find match between the user's input and proverbs database.

3) Frontend development

The proposed work includes user friendly interface for input and displaying results.

A. **Technology used:** HTML, CSS and Javascript is used for frontend development.

B. Task of frontend:

- a) **Search bar:** The front end provides search bar to user to input proverbs.
- b) **Display result:** After searching front end provides result of search based on if its correct or not, system provides translation or error.

- c) **Suggestion dropdown:** The front end provides auto suggestion dropdown based on the input of user for partially searching.

4) Testing and debugging

Different testing techniques and cases are used to ensure accuracy and performance of the system.

- a) **Unit testing:** unit testing is used to ensure that core functionalities are working correctly or not. Every function is tested differently with different test case for unit testing.
- b) **Integration testing:** integration testing is used to ensure that system work correctly after integration. Integration testing is done with different test cases.
- c) **User testing:** the system was tested with few user to ensure that the interface was intuitive and the result are accurate.
- d) **Performance testing;** performance testing is done to ensure that the search works smoothly even with increasing size of the proverb or with multiple users.

Chapter 4 Implementation and Testing

4.1 Introduction to Language and Various tool used in Proposed Work

4.1.1 Languages Used for Proposed Work:

1. Python

Python is used for back end system for a proposed work.

Python is an interpreter, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. It offers extensive support through libraries and framework like Flask, help to build web applications efficiently. Python's flexibility allow easy integration with database like SQLite, enabling efficiently retrieval of data.

2. SQLite:

SQLite is used for Database management in a proposed work.

SQLite is a lightweight, highly efficient, server less, and self-contained SQL database engine that stands out for its simplicity and ease of integration. Designed to be embedded within applications, SQLite eliminates the need for separate database server processes and complex configurations. It is used for small to medium projects to store entire data in a single file. It is fast, reliable and compatible with python through libraries like SQLite3 and web frameworks like Flask.

3. HTML and CSS

HTML is the standard markup language used to create web pages. It structures the content by using elements such as headings, paragraphs, lists, links, images, and more.

HTML elements are the building blocks of web pages, allowing developers to embed multimedia, create forms, and design the overall layout.

CSS stands for Cascading Style Sheets and it is used to style web documents. It is used to provide the background color and is also used for styling. It controls the layout, colors, fonts, and overall look of a web page. CSS is also recommended by World Wide Web Consortium (W3C). It can also be used along with HTML and Javascript to design web pages.

4. **JavaScript**

JavaScript is a dynamic, high-level programming language mainly used to create interactive and responsive elements on websites. JavaScript can enhance user experience by handling features such as input suggestions, form validation, and real-time updates without reloading the page. It works alongside HTML and CSS to make web applications more engaging and user-friendly. JavaScript's flexibility and wide browser support make it an essential tool for adding dynamic functionality to any web project.

4.1.2 Framework and Libraries Used for Proposed Work

1. **Web Frameworks: Flask** Flask is a lightweight and flexible web framework for Python, widely used to build web applications quickly and easily. Flask helps manage user inputs, connect to databases like SQLite, and display results through simple and organized routes. It follows a minimalistic approach, giving developers the freedom to structure their applications as needed while still offering powerful tools and extensions. Flask's simplicity, combined with its ability to handle backend operations smoothly, makes it an ideal choice for creating user-friendly and efficient web applications.

2. Python library: RapidFuzz

RapidFuzz is a fast and efficient Python library used for fuzzy string matching. RapidFuzz can help implement features like partial searching and providing suggestion. It helps the user to enter any type of input like Inscript or phonetics Punjabi as well as Romanized Punjabi. It is designed to be much faster than traditional fuzzy matching libraries, making it ideal for real-time applications where quick responses are important. RapidFuzz's accuracy and speed make it a powerful tool for improving search and input correction features in web applications.

4.1.3 IDE's Used for Proposed Work

1. Visual Studio Code

Visual Studio Code (VS Code) is a free, lightweight, and powerful code editor developed by Microsoft, widely used for programming in languages like Python, JavaScript, and more. VS Code provides a simple and organized environment to write, edit, and debug code for the Flask backend, SQLite database, and frontend design. It supports many helpful extensions like Python tools, database explorers, and live server previews, making development faster and more efficient. Its user-friendly interface and strong customization options make VS Code a popular choice among developers.

2. DB browser

DB Browser for SQLite is a free, open-source tool that allows users to easily create, design, and manage SQLite database files through a visual interface. DB Browser helps in building and editing the database where all the proverbs, their meanings, and Romanized forms are stored. It makes tasks like inserting, updating, and viewing data simple without needing to write SQL commands manually. By providing a user-

friendly way to handle database operations, DB Browser speeds up development and ensures the database is well-organized and accurate.

4.2 Algorithm used

The proposed work uses Naive Search combined with Fuzzy String Matching based on Levenshtein Distance.

4.2.1 Naïve search combined:

1. Loop over every possible starting index i in the text where the pattern could match
2. For each position i , compare the substring $T[i \dots i+m-1]$ with the pattern $P[0 \dots m-1]$.
3. If all characters match, record the index i as a valid match.
4. Continue until all positions are checked.

4.2.2 Levenshtein distance:

1. Create a matrix dp of size $(m+1) \times (n+1)$ where $dp[i][j]$ represents the minimum edit distance between $str1[0 \dots i-1]$ and $str2[0 \dots j-1]$.
2. Initialize: $dp[0][j] = j$ (converting empty $str1$ to first j chars of $str2$)
3. $dp[i][0] = i$ (converting first i chars of $str1$ to empty $str2$)
4. Fill the matrix:
5. If $str1[i-1] == str2[j-1]$, then $dp[i][j] = dp[i-1][j-1]$
6. Else $dp[i][j] = 1 + \min(dp[i-1][j], // \text{delete } dp[i][j-1], // \text{insert } dp[i-1][j-1]) //$
substitute
7. The answer is $dp[m][n]$

4.3 Testing Techniques

4.3.1 Black Box Testing

Black box testing is a software testing method where the internal structure, design, or implementation of the system being tested is not known to the tester. The focus is purely

on inputs and expected outputs without considering how the system processes those inputs.

Table 4.1 Black Box Testing

Test case ID	Test case	input	Expected output	Actual output	Fail/pass
1	Exact Punjabi proverb	ਸਾਹ ਸੁੱਕ ਜਾਣਾ	To be extremely scared or anxious	To be extremely scared or anxious	Pass
2	Romanized Punjabi proverb	Sah suk jana	To be extremely scared or anxious	Romanized Punjabi : Saah sukk jaana Meaning: To be extremely scared or anxious	pass
3	User enter input with small typo	ਉੱਚਾ	Provide auto suggestion dropdown with similar proverbs	Auto suggestions shows similar proverbs	pass
4	User enter unrelated input	xyzabc	Error message	Error message	pass
5	Empty input	“ “	Please fill the search bar	Show please fill the search bar	pass

Test case 1: User provides exact Punjabi proverb

The screenshot shows a web application interface with a light blue background. At the top, there is a text input field labeled "Enter Punjabi Muhavra (Idiom):" containing the text "ਸਾਹ ਸੁੱਕ ਜਾਣਾ". Below the input field are two buttons: "Search" and "Clear". Below the buttons is a large, rounded rectangular box with a light orange background. Inside this box, the text "ਪੰਜਾਬੀ: ਸਾਹ ਸੁੱਕ ਜਾਣਾ" is displayed in bold. Below this, the text "Romanized Punjabi: saah sukk jaana" is displayed. Further down, the text "Meaning:" is followed by "To be extremely scared or anxious". At the bottom, the text "Equivalent English Idiom:" is followed by "Scared out of one's wits".

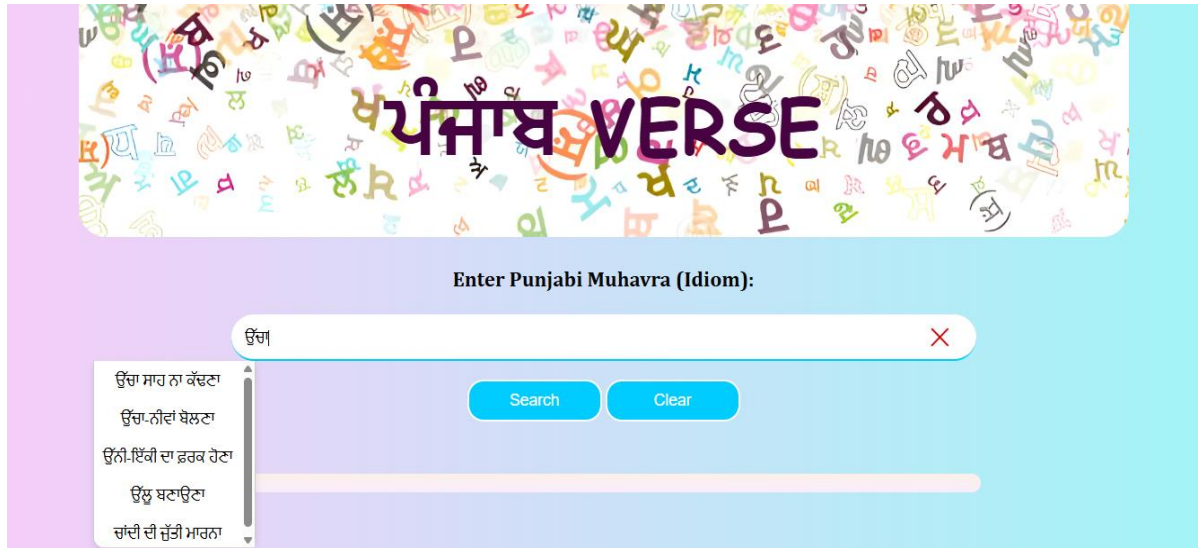
Figure 4.1 Test Case 1

Test case 2: User provides Romanized Punjabi proverb

The screenshot shows a web application interface with a light blue background. At the top, there is a text input field labeled "Enter Punjabi Muhavra (Idiom):" containing the text "saah sukk jaana". Below the input field are two buttons: "Search" and "Clear". Below the buttons is a large, rounded rectangular box with a light orange background. Inside this box, the text "ਪੰਜਾਬੀ: ਸਾਹ ਸੁੱਕ ਜਾਣਾ" is displayed in bold. Below this, the text "Romanized Punjabi: saah sukk jaana" is displayed. Further down, the text "Meaning:" is followed by "To be extremely scared or anxious". At the bottom, the text "Equivalent English Idiom:" is followed by "Scared out of one's wits".

Figure 4.2 Test Case 2

Test case 3: User enter input with small typo



ਪੰਜਾਬ VERSE

Enter Punjabi Muhavra (Idiom):

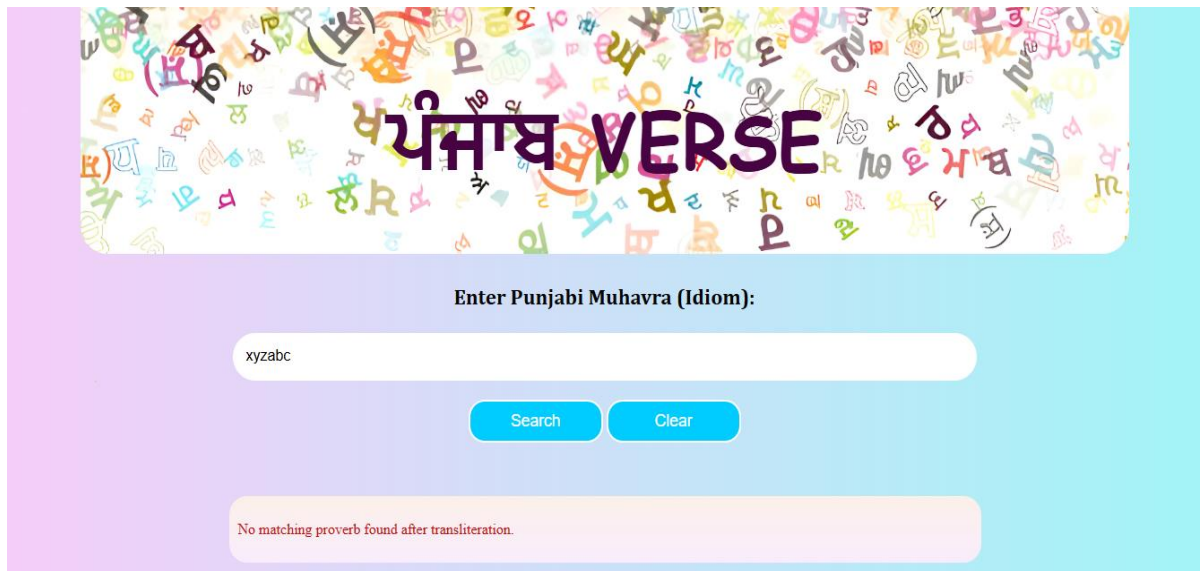
ਉੱਚਾ

ਉੱਚਾ ਸਾਹ ਨਾ ਕੱਢਣਾ
ਉੱਚਾ-ਨੀਵਾਂ ਬੋਲਣਾ
ਉੱਨੀ-ਇੱਕੀ ਦਾ ਫ਼ਰਕ ਹੋਣਾ
ਉੱਲੁ ਬਣਾਉਣਾ
ਚਾਲੀ ਦੀ ਜੁੱਤੀ ਮਾਰਨਾ

Search Clear

Figure 4.3 Test Case 3

Test case 4: User enter unrelated input



ਪੰਜਾਬ VERSE

Enter Punjabi Muhavra (Idiom):

xyzabc

Search Clear

No matching proverb found after transliteration.

Figure 4.4 Test Case 4

Test case 5: User provides empty input

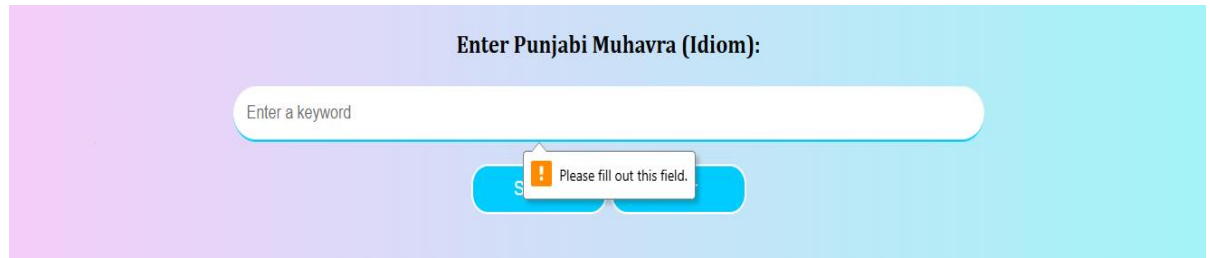


Figure 4.5 Test Case 5

4.3.2 Unit testing

Unit testing is a type of software testing where individual units or components of a software application are tested in isolation to ensure they work as intended.

```
PS C:\Users\HP\Documents\Manu\punjabverse> pytest unit_testing.py -v
===== test session starts =====
platform win32 -- Python 3.10.0, pytest-8.3.5, pluggy-1.5.0 -- C:\Users\HP\AppData\Local\Programs\Python\Python310\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\HP\Documents\Manu\punjabverse
collected 8 items

unit_testing.py::test_exact_match_punjabi PASSED
unit_testing.py::test_exact_match_romanized PASSED
unit_testing.py::test_no_match_gurmukhi PASSED
unit_testing.py::test_no_match_romanized_manual PASSED
unit_testing.py::test_fuzzy_allowed_if_selected PASSED
unit_testing.py::test_suggestion_api_romanized PASSED
unit_testing.py::test_suggestion_api_gurmukhi PASSED
unit_testing.py::test_no_match_romanized_manual_with_error PASSED

===== 8 passed in 0.28s =====
```

Figure 4.6 Unit Testing

4.3.3 Integration testing

Integration testing is a level of software testing where individual units or modules are combined and tested as a group to verify that they work together correctly.

```
pytest integration.py -v
===== test session starts =====
platform win32 -- Python 3.10.0, pytest-8.3.5, pluggy-1.5.0 -- C:\Users\HP\AppData\Local\Programs\Python\Python310\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\HP\Documents\Manu\punjabverse
collected 8 items

integration.py::test_home_page_loads PASSED
integration.py::test_exact_match_gurmukhi PASSED
integration.py::test_exact_match_romanized PASSED
integration.py::test_fuzzy_fallback_works PASSED
integration.py::test_no_match_gurmukhi_message PASSED
integration.py::test_no_match_romanized_message PASSED
integration.py::test_suggestion_gurmukhi_api PASSED
integration.py::test_suggestion_romanized_api PASSED

===== 8 passed in 0.28s =====
```

Figure 4.7 Integration Testing

4.3.4 Acceptance testing

Acceptance testing is the final level of software testing where the system is tested from the user's perspective to verify if it meets the business requirements and is ready for release.

```
PS C:\Users\HP\Documents\Manu\punjabverse> pytest acceptance.py -v
===== test session starts =====
platform win32 -- Python 3.10.0, pytest-8.3.5, pluggy-1.5.0 -- C:\Users\HP\AppData\Local\Programs\Python\Python310\python.exe
cachedir: .pytest_cache
rootdir: C:\Users\HP\Documents\Manu\punjabverse
collected 6 items

acceptance.py::test_acceptance_exact_match_gurmukhi PASSED
acceptance.py::test_acceptance_romanized_selected_suggestion PASSED
acceptance.py::test_acceptance_romanized_manual_input PASSED
acceptance.py::test_acceptance_gurmukhi_no_match PASSED
acceptance.py::test_acceptance_gurmukhi_suggestion_prefix PASSED
acceptance.py::test_acceptance_romanized_suggestion_prefix PASSED

===== 6 passed in 0.34s =====
PS C:\Users\HP\Documents\Manu\punjabverse>
```

Figure 4.8 Acceptance Testing

4.4 Test Cases designed for the project work

The proposed work has following test cases:

4.4.1 Functional Testing

Functional Testing is used to test the core features of the system. Its goal is to ensure the search and translation of proposed work correctly. Given are the test cases that will use for functional testing:

- 1) **Valid proverb search:** Enter all the Punjabi proverbs that are in the dataset to ensure whether system provides correct English contextual translation.
- 2) **Invalid proverb search:** Enter any random text to ensure that system should return no result found as output.
- 3) **Case Insensitivity Test:** Enter the proverb whether using phonetics or inscript Punjabi typing or Romanized Punjabi the search should return the same result regardless of input case.
- 4) **Empty input handling:** Click the search button without entering any text. The system should show an error message prompting the user to enter a search term.

4.4.2 UI/UX Testing

UI/UX Testing is used to test the responsiveness of the user interface. Its goal is to ensure the website works smoothly across devices. Below are some test cases used for UI/UX testing:

- 1) **Mobile compatibility:** Try to open the website on a mobile browser. The layout should be responsive and work correctly.
- 2) **Button and navigation testing:** Try to click all the button. All button should function correctly and not crash the site.

4.4.3 Performance Testing

The Performance testing is used to check the speed and load handling capability of the system.

1. **Response time test:** Measure the time taken for a search request to return result. Response time should be fast

Chapter 5: Result and Discussion

5.1 User Interface Representation

5.1.1 Module

A module is a separate, reusable part of a software system that performs a specific function. It helps organize the code and makes the system easier to manage, test, and update.

The Proposed work has three modules:

- I. Proverb Dataset Management Module
- II. Search Module
- III. Frontend User Interface Module

5.1.1.1 Proverb Dataset Management Module

- I. Functionality:** The Proverb Database Management Module is used to store all proverbs with their contextual meaning in a structured format.
- II. Key features:** This module stores the Punjabi proverb with their contextual meaning and English equivalent. It allows easy retrieval of data for searching
- III. Implementation:** SQLite is used as database system.
- IV. Work flow:**
 - 1) The database stores each proverb with an ID, their contextual meaning and English equivalent.
 - 2) When search module request a proverb it retrieve the correct data.
 - 3) The module sends the result back to the search module for processing.

5.1.1.2 Search Module:

- I. Functionality:** The search module processes the user input and finds the best matching proverb from the database.

- II. **Key features:** This module finds the proverbs based on user input. This module provides the result even if there is spelling mistake.
- III. **Implementation:** z
- IV. **Work flow:**
 - 1) Search module takes user input from UI Module.
 - 2) It queries the database module for relevant result.
 - 3) It return a list of matching proverbs to the UI module

5.1.1.3 Front end user interface module(UI module)

- I. **Functionality:** The UI module provides an easy to use web interface for users to search, browse and view the translation.
- II. **Key features:** UI module provides a search bar where user enter Punjabi proverb and display the search result with their English contextual meaning.
- III. **Implementation:** HTML and CSS is used for user friendly interface.
- IV. **Work flow:**
 - 1) User enter a Punjabi proverb in the search bar.
 - 2) This module sends the input to the search module for processing.
 - 3) It retrieve the result and display it on the webpage.

5.1.2 Layout of the Website:

1. Searching bar: The main page contains a search bar for user to give an input to a system. A search button which is used to give search command to a system. A clear button is used to clear old output on the web page. During providing an input, web page shows cross on the right side of search bar to clear input from search bar.

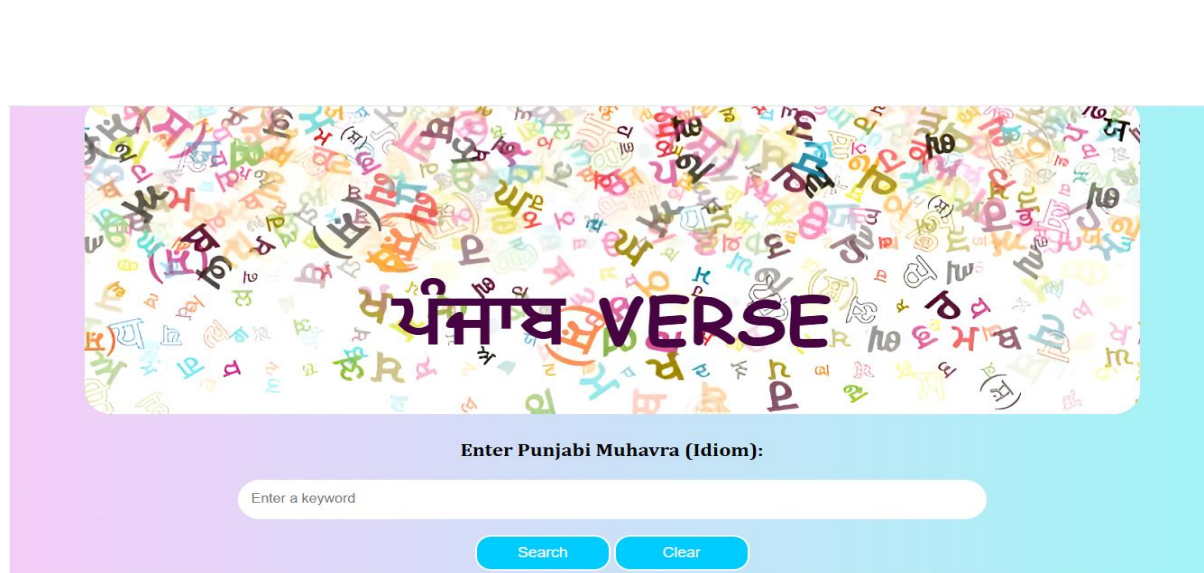


Figure 5.1 Searching bar

2. About us



Figure 5.2 About us

On Main page under searching bar, there is introduction of website and importance of Punjabi proverbs.

3. Features



Figure 5.3 Features

Under Objective there is “why our website” which helps user to know about features of website.

4. Steps:

At last webpage includes steps to use the website so user can have clear guidance to use the website.



Figure 5.4 Steps

5.2 Snapshots of System

5.2.1 Home Page:

Home page of website includes the search bar with Search and clear button. User can type Punjabi Proverbs as input in the Search bar and use search button to search its Contextual Meaning.

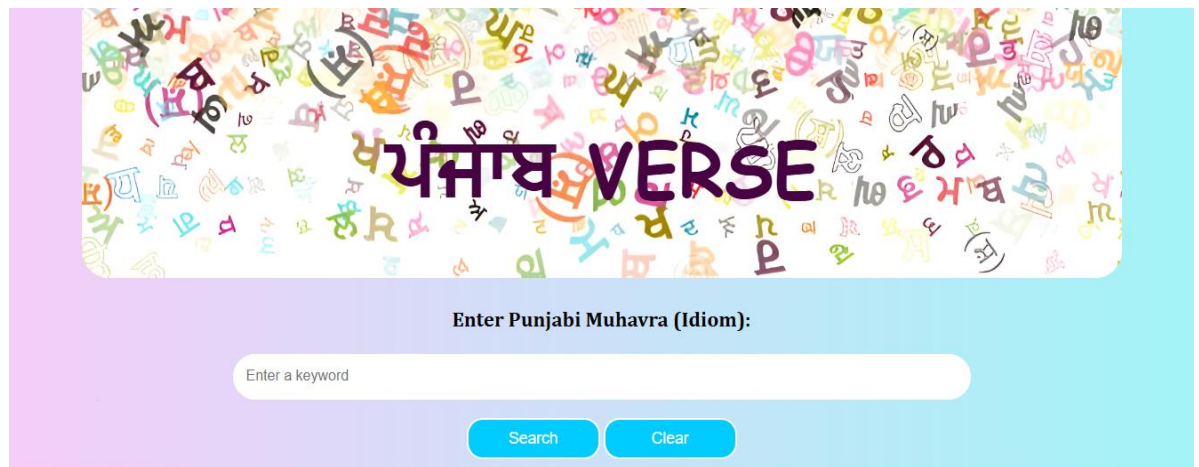


Figure 5.5 Home Page

5.2.2 User interaction: Searching in Punjabi Phonetics or Inscript

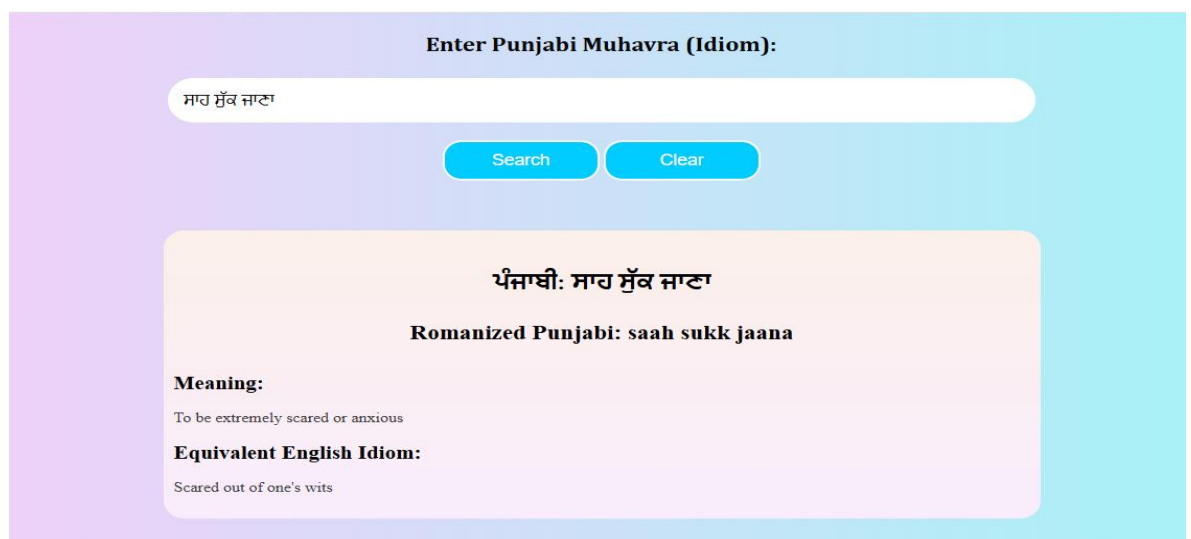
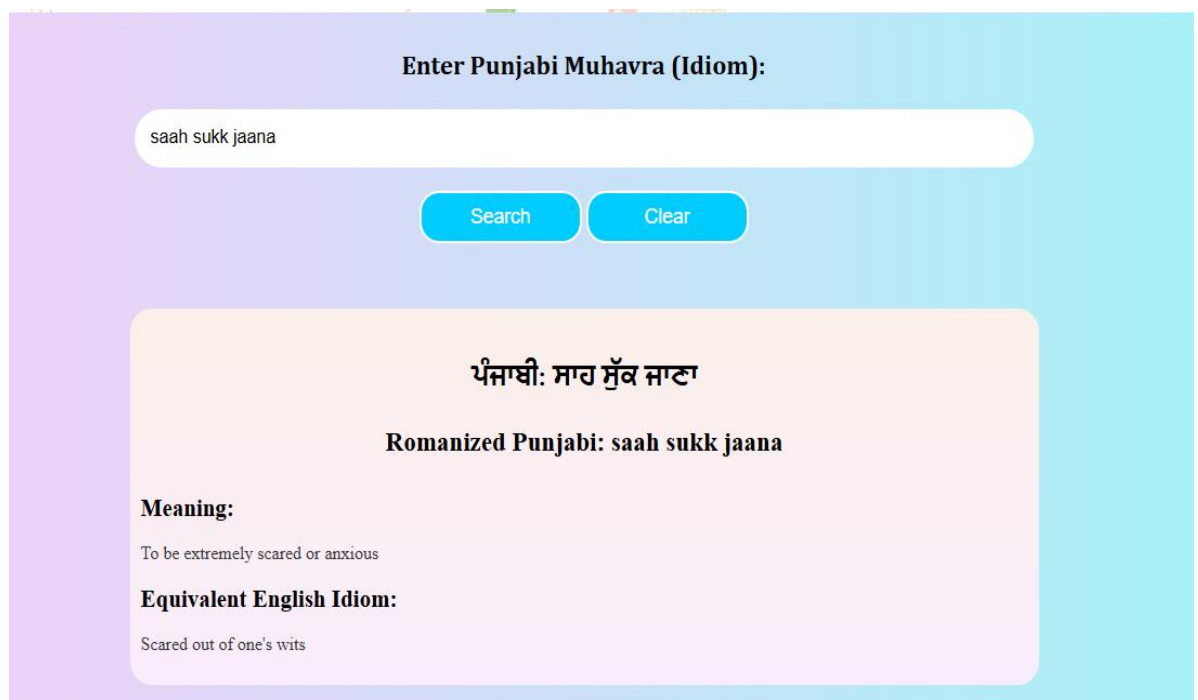


Figure 5.6 User Interaction: Punjabi phonetics

Home page provides the Search bar for user. User can Search with Punjabi Phonetics or Inscript to find Contextual meaning of Punjabi Proverb. The System provides the Contextual Meaning with Its Equivalent English Meaning and Romanized Punjabi of Proverb on the same page. . The system provides fuzzy searching that if user put wrong spelling then the system try to find most accurate match and provide answer to the user.

5.2.3 User interaction: Searching in Romanized Punjabi



Enter Punjabi Muhavra (Idiom):

saah sukk jaana

ਪੰਜਾਬੀ: ਸਾਹ ਸੁੱਕ ਜਾਣਾ

Romanized Punjabi: saah sukk jaana

Meaning:
To be extremely scared or anxious

Equivalent English Idiom:
Scared out of one's wits

Figure 5.7 User Interaction: Romanized Punjabi

Home page provides the Search bar for user. User can Search with Romanized Punjabi to find Contextual meaning of Punjabi Proverb. The System provides the Contextual Meaning with Its Equivalent English Meaning and Romanized Punjabi of Proverb on the same page. The system provides fuzzy searching that if user put wrong spelling then the system try to find most accurate match and provide answer to the user.

5.2.4 Keyword Searching

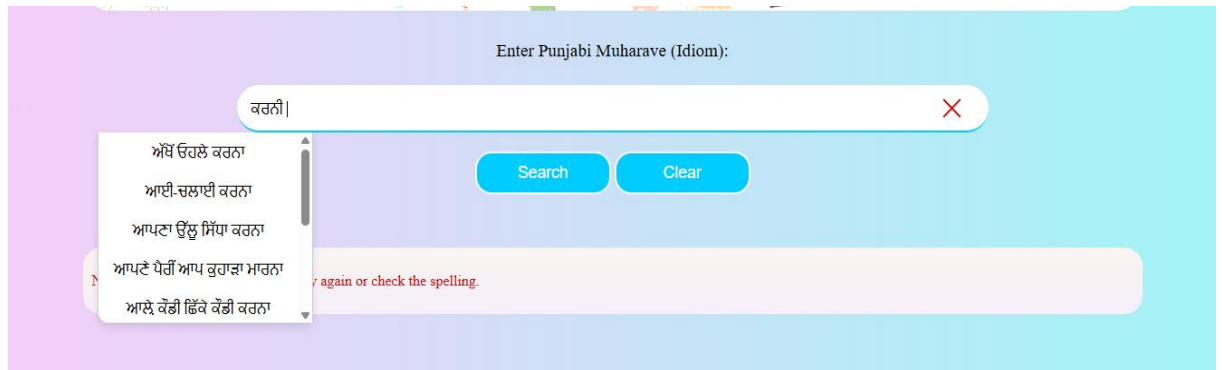


Figure 5.8 keyword searching: Gurmukhi

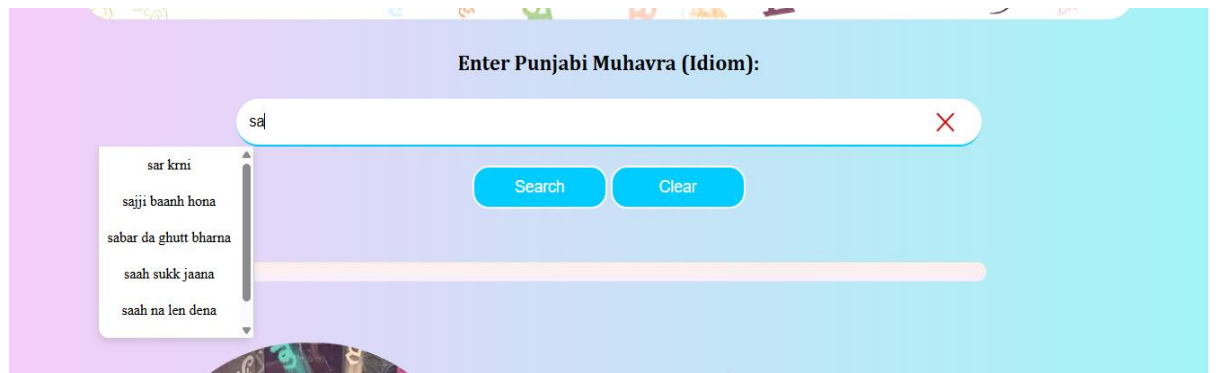


Figure 5.9 keyword searching: Romanized punjabi

Home Page provides the search bar where user can keyword search Punjabi Proverbs with Gurmukhi as well as Romanized Punjabi input. It helps user to find matching proverbs with search word.

5.2.5 Error Message

Enter Punjabi Muharave (Idiom):

ek jut hona

Search Clear

No matching proverb found. Please try again or check the spelling.

Figure 5.10 Error Message

When user search wrong proverb then the home page provides error message to the user.

5.3 Back Ends Representation

The proposed work use Database created by programmer to search English contextual meaning rather than AI to ensure most of the accuracy of the language. The database contains Table “**Proverbs**”, which contains following columns:

- I. Punjabi Idiom:** Punjabi idiom column contains 200 Punjabi idioms with ID.
- II. English contextual Meaning:** This column contains the English Contextual meaning of Punjabi proverbs
- III. English Equivalence:** This column contains English equivalence of Punjabi Proverbs if exist.
- IV. Romanized Punjabi:** This column contains Romanized Punjabi for Punjabi Proverbs.

	ID INTEGER NOT NULL UNIQUE PRIMARY KEY AUTOINCREMENT	Punjabi_Idiom TEXT NOT NULL	English_meaning TEXT NOT NULL	Equivalent_English_Id iom TEXT	English_Transliteration on TEXT	+
1	0			NULL	NULL	
2	1	ਉਸਤਾਦੀ ਕਰਨੀ	To trick or deceive...	Pull a fast one.	ustaadi krni	
3	2	ਉਡੀਕ ਉਡੀਕ ਕੇ ਬੁਝਾ ਹੋ...	waiting for a very ...	waiting till the co...	udeek udeek ke budh...	
4	3	ਉਂਗਲ ਕਰਨਾ	To blame or accuse ...	Point the finger at	ungal krni	
5	4	ਉੱਚਾ ਸਾਹ ਨਾ ਕੱਢਣਾ	To become frightened	Paralyzed with fear	ucha saah na kadna	
6	5	ਉੱਚਾ-ਨੀਵਾਂ ਬੋਲਣਾ	To speak in disresp...	Talk down to someon...	ucha-neeva bolna	
7	6	ਉਡੀਕ-ਉਡੀਕ ਕੇ ਬੁੱਢਾ ਹੋ ...	To wait for a very ...	Wait till the cows ...	NULL	
8	7	ਉੱਨੀ-ਇੱਕੀ ਦਾ ਫ਼ਰਕ ਹੋਣਾ	A very negligible d...	A hair's breadth	unni-ikki da farak ...	
9	8	ਉੱਲੂ ਬਣਾਉਣਾ	To fool someone	Make a fool of some...	ullu banauna	
10	9	ਅੱਖ ਲੱਗਣੀ	To sleep or take a ...	Catch some Z's	akh lagni	
11	10	ਅੱਖ ਖੁੱਲ੍ਹਣੀ	To wake up from sle...	Shake off the sleep	akh khullni	
12	11	ਅੱਖਾਂ 'ਤੇ ਬਿਠਾਉਣਾ	Show extreme care a...	Treat someone like ...	akhan te bithauna	
13	12	ਅੱਖਾਂ ਦਿਖਾਉਣਾ	To scare someone	Give someone the cr...	akhan dikhauna	
14	13	ਅੱਖਾਂ ਅੱਗੇ ਹਨੇਰਾ ਆਉਣਾ	To become scared	Heart skipped a beat	akhan agge hanera a...	
15	14	ਅੱਖਾਂ ਫੇਰ ਲੈਣਾ	To leave someone	NULL	akhan fer laina	
16	15	ਅੱਖਾਂ ਵਿੱਚ ਲਹੂ ਉੱਤਰਨਾ	To get extremely an...	Blow your top	akhan vich lahu uta...	
17	16	ਅੱਖੋਂ ਓਹਲੇ ਕਰਨਾ	To forget someone	NULL	akhon ohle krni	
18	17	ਅੱਗ ਨਾਲ ਖੇਡਣਾ	To take dangerous r...	Play with fire	agg naal khedna	
19	18	ਅੱਗ ਲਾਉਣਾ	To provoke conflict	Stir the pot; Fan t...	agg launa	

Figure 5.11 Database

Chapter 6 Conclusion and Future Scope

6.1 Conclusion:

The Proposed Work, PunjabVerse successfully addresses the key challenges associated with translating and understanding Punjabi proverbs, for users who are familiar with the Gurmukhi script or Romanized punjabi. By integrating fuzzy matching algorithms, Romanized Punjabi input support, and contextual English translations, the system makes the rich cultural heritage of Punjabi proverbs accessible to a much wider audience.

The project focuses on providing a user-friendly, efficient, and accurate platform that overcomes common issues like spelling errors, varied Romanized as well as Gurmukhi input styles, and language barriers. With the help of a user friendly web interface and a robust backend powered by Flask, SQLite, and RapidFuzz, users can easily search for proverbs, explore their meanings, and find English equivalents that preserve the original context.

Overall, the system promotes cultural preservation, language learning, and cross-cultural understanding, offering a valuable resource for students, researchers, and anyone interested in Punjabi literature. The flexibility of the system also allows for future expansion, such as adding more proverbs, supporting more languages, and enhancing search intelligence.

Thus, this project not only fulfills its technical objectives but also contributes meaningfully to bridging the gap between languages and cultures.

6.2 Future Scope

The suggested work sets up a solid base to build on and make better. Down the road, we can grow and boost the system in these ways:

- 1) **Voice Input and image translation:** Add speech-to-text features, so users can say the proverb in Punjabi or show an image to get quick translations without typing. This would make the system easier to use for people who don't like typing in Punjabi.
- 2) **Mobile Application Development:** Create a mobile app version of the system for Android and IOS to give users access on the go. A mobile app could have extra features like voice search, camera input, and alerts for "proverb of the day."
- 3) **Multilingual Support:** Grow the system to handle translations into other languages like Hindi, French, Spanish, and more turning it into a worldwide platform for Punjabi cultural knowledge. Also, add proverbs from other local languages to make a multi-language proverb translation tool.
- 4) **Machine Learning for Smarter Suggestions:** Use machine learning methods to make search results more accurate and offer more personal proverb ideas based on how users behave and what they type. Smart recommendations could suggest proverbs that fit emotions, situations, or topics.
- 5) **User Contribution Feature:** Give users the chance to add new proverbs with their translations and meanings. Set up a system to review and approve submissions to keep the quality high while making the database bigger.
- 6) **Cultural Context Expansion:** Put in more details about cultural background, example sentences, and when to use each proverb. This will help users understand and use the proverbs when they talk to others.

- 7) **Offline Access:** Create a version of the database that works without internet for users who might not always have a connection.
- 8) **Advanced Search Filters:** Add new ways to search, like looking by category (such as wisdom, humor, family, nature) or how hard the proverb is to understand. This will make it easier for users to find proverbs that fit what they need.

References

- [1] A. Punjabi, "9th-10th Adhunik Punjabi vyakran ate lekh Rachna-9.pdf," *Google Docs*, 2019. https://drive.google.com/open?id=1wXHjkmJQwxsiXjzHSNdbDfKJaHVJdw96&authuser=punjabeducare%40ppppjalandhar.org&usp=drive_fs (accessed Apr. 03, 2025).
- [2] Joana Kondo Ismaili, "Problematic Areas in the Translation of Proverbs," *Journal of Cultural and Religious Studies*, vol. 6, no. 11, Nov. 2018, doi: <https://doi.org/10.17265/2328-2177/2018.11.003>
- [3] N. Kothari, C. P. Jain, D. Soni, A. Kumar, A. Dadheech, and H. Sharma, "INSTANT LANGUAGE TRANSLATION APP," *International Journal of Technical Research & Science*, vol. 9, no. Spl, pp. 27–35, Jun. 2024, doi: <https://doi.org/10.30780/specialissue-iset-2024/018>
- [4] I. Jibreel, "Online Machine Translation Efficiency in Translating Fixed Expressions Between English and Arabic (Proverbs as a Case-in-Point)," *Theory and Practice in Language Studies*, vol. 13, no. 5, pp. 1148–1158, May 2023, doi: <https://doi.org/10.17507/tpls.1305.07>.
- [5] T. Spencer, "Proverbial Machine Translation: Translating Proverbs between Spanish and English Using Phrased Based Statistical Machine Translation with the Grammatical Category Based Approach," *The Aquila Digital Community*, 2018. http://aquila.usm.edu/honors_theses/617
- [6] S. Kaur and R. Kaur, "An Efficient Romanization of Gurmukhi Punjabi Proper Nouns for Pattern Matching," *International Journal of Research in Engineering and Technology*, vol. 9, no. 10, pp. 1–6, 2022.
- [7] T. Dhar, S. Sitaram, and M. Shrivastava, "Context-aware Transliteration of Romanized South Asian Languages," *Computational Linguistics*, vol. 50, no. 2, pp. 475–512, Jun. 2024

