

MINI DICTIONARY

Data Structures – CS163

23APCS2 | Group 3



Last Modified Day:

August 22, 2024

Table of Contents

[A. Abstract 2](#_Toc175244450)

[B. Introduction 2](#_Toc175244451)

[1. Overview: 2](#_Toc175244452)

[2. Key features: 2](#_Toc175244453)

[C. Group Information 3](#_Toc175244454)

[Group ID: 3 3](#_Toc175244455)

[Member: 3](#_Toc175244456)

[1. Le Vinh Thuan (Team Leader) – 231250\_\_ (ANY%) 3](#_Toc175244457)

[2. Tu Cong Thanh – 231250\_\_ (ANY%) 4](#_Toc175244458)

[3. Le Thi Tuyet Tram – 231250\_\_ (ANY%) 4](#_Toc175244459)

[4. Nguyen Thu Uyen – 23125048 (ANY%) 4](#_Toc175244460)

[5. Ngo Hoang Bao Thach – \_\_1250\_\_ (ANY%) 5](#_Toc175244461)

[D. Data Storage 5](#_Toc175244462)

[E. Project Architecture 6](#_Toc175244463)

[F. Implementation Detail 7](#_Toc175244464)

[G. Technical Problems and Solutions: 8](#_Toc175244465)

[Back-end: Data Structure 8](#_Toc175244466)

[UI: Multi-lined table 8](#_Toc175244467)

[H. Feature Demonstration: 9](#_Toc175244468)

[I. Conclusion 10](#_Toc175244469)

[References 10](#_Toc175244470)

# Abstract

In this project, we focus on implementing a dictionary application using the programming language C++, Ternary Tree and Binary Tree data structure, Cross-Platform GUI Library wxWidgets and CMake. It is thanks to the help of CS163 course’s lecture notes, Dr. Dobb’s article about Ternary Search Tree, and Le Minh Hoang’s Suffix Tree article.

The main objective of this project is to use Ternary Tree to develop a user-friendly and responsive mini dictionary application. The primary aim is to demonstrate the use of data structure in making efficient word searches, additions, edits, and deletions, help users look up the meanings of words and allowing users to switch between various types of dictionaries, such as English-to-English, English-to-Vietnamese, and Vietnamese-to-English.

The report outlines the project’s objectives, key features, and the algorithms employed to optimize performance. We have developed and implemented all the functionality required, albeit not able to fully preprocess data and clean code.

# Introduction

## Overview:

The mini dictionary group project is a practical application of the data structures and algorithms learned in the CS163 course. We are required to manage tasks in Trello and source code in Git. Only array and vector are allowed.

Dictionaries are indispensable tools in language learning and communication, and their digital versions offer enhanced accessibility. The project’s motivation stems from the need to create an efficient dictionary application capable of handling large datasets, such as large number of words, slang terms, and emojis.

By implementing this project, the group aims to demonstrate the effective and suitable use of data structures in managing and manipulating large datasets.

## Key features:

* Switch between data sets, including:
  + English-to-English
  + English-to-Vietnamese
  + Vietnamese-to-English
  + Emoticon
  + English slang
* Search:
  + Using keywords
  + Using definitions
* View Search History
* View, add or remove words from Favorite list
* Modifying dictionary entries:
  + Add/Edit/Remove words
  + Reset dictionary
* Additional features:
  + View a random word and its definition.
  + Play Guess the meaning game.
  + Play Guess the word game.

# Group Information

Include the group ID and a list of group members along with their names and roles in the project. It's also important to mention the tasks assigned to each member and their individual contribution percentages.

## Group ID: 3

## Member:

### Le Vinh Thuan (Team Leader) – 23125019 (ANY%)

* Planning meetings and manage coding progress
* Build Ternary Tree structure
* Gathering Data Set
* UI implementation:
  + Home Page
  + Gameplay Page
* Feature:
  + Search using keywords.
  + Display words of the day.
  + Provide a word with four definitions, and users guess its meaning.
  + Provide a random definition with four keywords, and users choose the correct word.

### Tu Cong Thanh – 23125018 (ANY%)

* Build Ternary Tree structure
* Build Suffix Array structure
* Build Red-Black Tree structure
* Fine-tuning and wrapper
* Gathering Data Set
* UI implementation:
  + Base frame
  + Search Page
  + More Page
  + Game Page
* Feature:
  + Switching Data Sets
  + Game algorithms
  + Search with definition
  + Reset dictionary to its original state.

### Le Thi Tuyet Tram – 23125093 (ANY%)

* UI Design:
  + Light Mode
* UI implementation:
  + Game Page
  + More Page
* Feature:
  + Users can add a new word and its definition.
  + Users can edit the definition of an existing word.
  + Users can remove a word from the dictionary.

### Nguyen Thu Uyen – 23125048 (ANY%)

* Documentation:
  + Report
  + README.md
* UI Design:
  + Light Mode (Home Page, Search Page, Dialogs, Add/Edit, Game Page, History and Favorite Page)
  + Dark Mode
* UI Implementation:
  + History Page
  + Favorite Page
  + Dark Mode
* Features:
  + View the history of search words again.
  + Users can view a random word and its definition.
  + Users can view their favorite list.
  + Users can remove/add a word from their favorite list.

### Ngo Hoang Bao Thach – 17125001 (ANY%)

* Documentation:
  + Report
* Gather Slang dataset
* Features:
  + Refactor get random words
* UI Implementation:
  + Dark mode: (Main frame)
  + Dark mode: Search page
  + Dark mode: Game setting page
  + Dark mode: Game play page

# Data Storage

Discuss how data is stored in the project. Explain the rationale behind the chosen data storage approach.

## Data Storage Overview

We organize data in a structured format to efficiently manage and update dictionary entries. The data is stored in a hierarchical directory structure within a main folder named "Data". This folder contains:

* **History**: history.txt – Stores words that users have searched.
* **Favorites**: favorite.txt – Contains words that users have marked as favorites.
* **Datasets**: Five subfolders corresponding to different datasets: EngtoEng, EngtoViet, ViettoEng, Emotion, and Slang words.

Each dataset subfolder includes:

* **Data.txt**: Contains the initial set of words and their definitions.
* **Insert.txt**: Stores new words added by users.
* **Delete.txt**: Contains words that users have removed from the dataset.

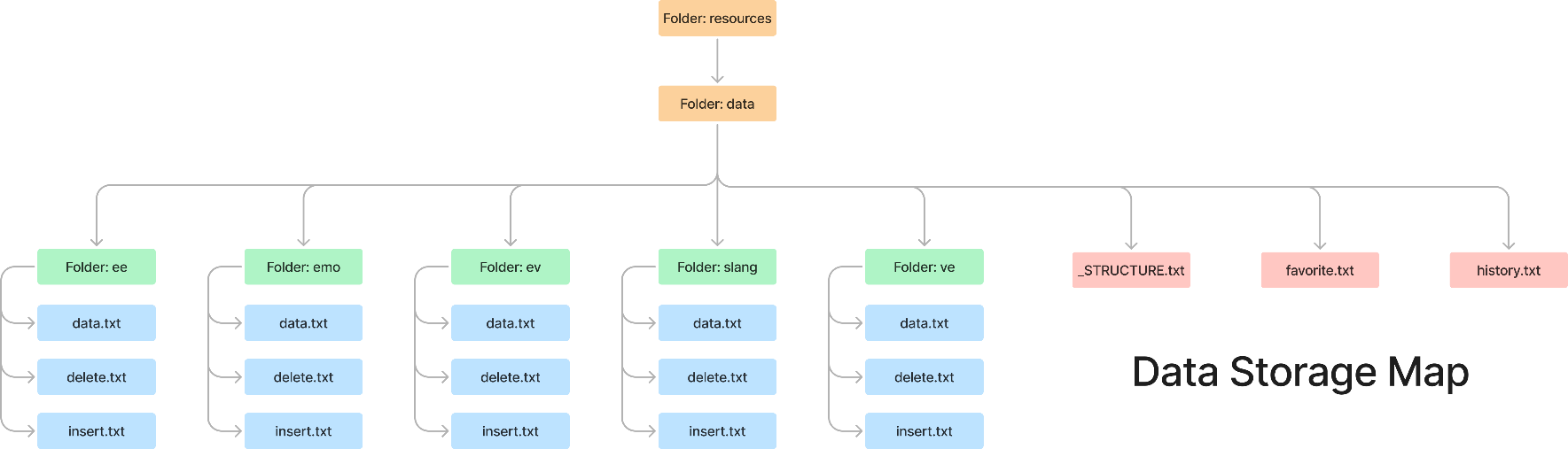


Figure 1. Data Storage Map

## Data Loading and Management Approach

When a dataset is loaded, the application follows a systematic approach to build and maintain the data structures:

* 1. **Loading Initial Data**: The application first reads data.txt to build a primary data structure, a Ternary Search Tree, that contains all the existing words and their definitions. This forms the foundation of the dataset.
  2. **Incorporating User-Added Words**: Next, the application processes insert.txt to integrate any new words added by users into the tree. This allows the dictionary to be dynamically updated with user contributions.
  3. **Removing Words**: Finally, the application reads delete.txt to remove specific words from the data structure. This ensures that the dataset remains current and reflects user deletions.

## Rationalize

**Separation:** By segregating data into different files (data.txt, insert.txt, delete.txt), the approach isolates different types of operations (initial data load, additions, deletions, and reset). This separation simplifies the process of updating the dictionary and enhances maintainability.

**Efficient Updates:** This method allows for efficient updates to the data structure. Loading the entire dataset from data.txt initially and then applying incremental changes from insert.txt and delete.txt avoids the need to rebuild the data structure from scratch every time a user makes a change. This will also allow users to reset the dictionary with ease due to the separative nature of this approach.

**User Interaction Handling:** The separate history.txt and favorite.txt allows the application to track user behavior and preferences without affecting the core dictionary data and simplify the process of data mining.

**Scalability:** The approach supports scalability by managing updates through discrete files. This modular method makes it easier to handle large datasets and multiple user interactions without compromising performance.

**Data Integrity:** Using separate files for different operations ensures that the core dataset remains consistent and correct. This approach helps prevent errors that could arise from mixing data insertion and deletion operations.

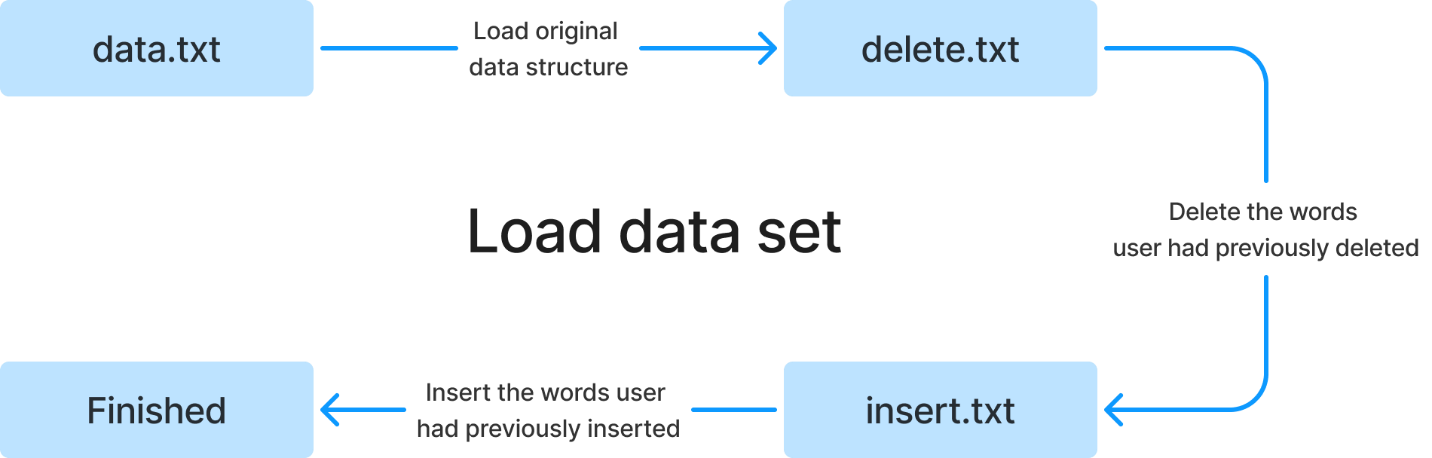


Figure 2. Initial Data Set Loading

# Project Architecture

Describe the overall architecture of the project, including the meaning and structure of project folders. Explain the main data structures employed in the project and their alternatives, highlighting why certain choices were made.

## Overview

We organize our project into 2 main folders Resources and Sources.

The Resources include all resources for our project.

// Draw the hierarchy of Resources and Sources

## Project Structure

## Main data structure

The project utilizes two primary data structures to handle different aspects of dictionary search functionality: the Ternary Search Tree (TST) and the Suffix Tree. Each of these structures is chosen for its specific advantages in efficiently managing and retrieving dictionary data, catering to different search requirements.

### 3.1. Ternary Search Tree Implementation

The Ternary Search Tree is employed as the main data structure for storing and searching words along with their definitions. A TST is a hybrid between a binary search tree and a trie, making it particularly well-suited for managing dictionary data where prefix-based searches are common.

A computer screen with white text and green text

Description automatically generated with medium confidence

Fig. Instances of Treenode

Each node in the TST includes several data type, including val (a character), eow (to check whether this is the end of a word), defi (if eow = True then defi will store the definition of that word), and 3 pointers to left, mid, and right.

A white and blue dots on a black background

Description automatically generated

Fig. An example visualization of a TST

### 3.2. Suffix Tree Implementation

# Implementation Detail

Provide a list of the structures/classes used in the project and their relevant functions. Choose several main flows or use cases and explain how these structures and methods collaborate with each other to achieve the desired functionality.

# Technical Problems and Solutions:

We encountered some big technical challenges that we must resolve together, include:

## Back-end:

1. Data Consistency Issues

When we prepared datasets for our project, we could not find 5 datasets with the same format, so we have to re-format the dataset into 2 formats:

* For Eng-Viet and Viet-Eng datasets:
* For others datasets:

1. Performance Concerns
2. Take a word randomly
3. Search with definition

## Front-end:

# Feature Demonstration:

Showcase all the features of the project, either through a video clip or screenshots. Provide a step-by-step explanation of each feature and its functionality.

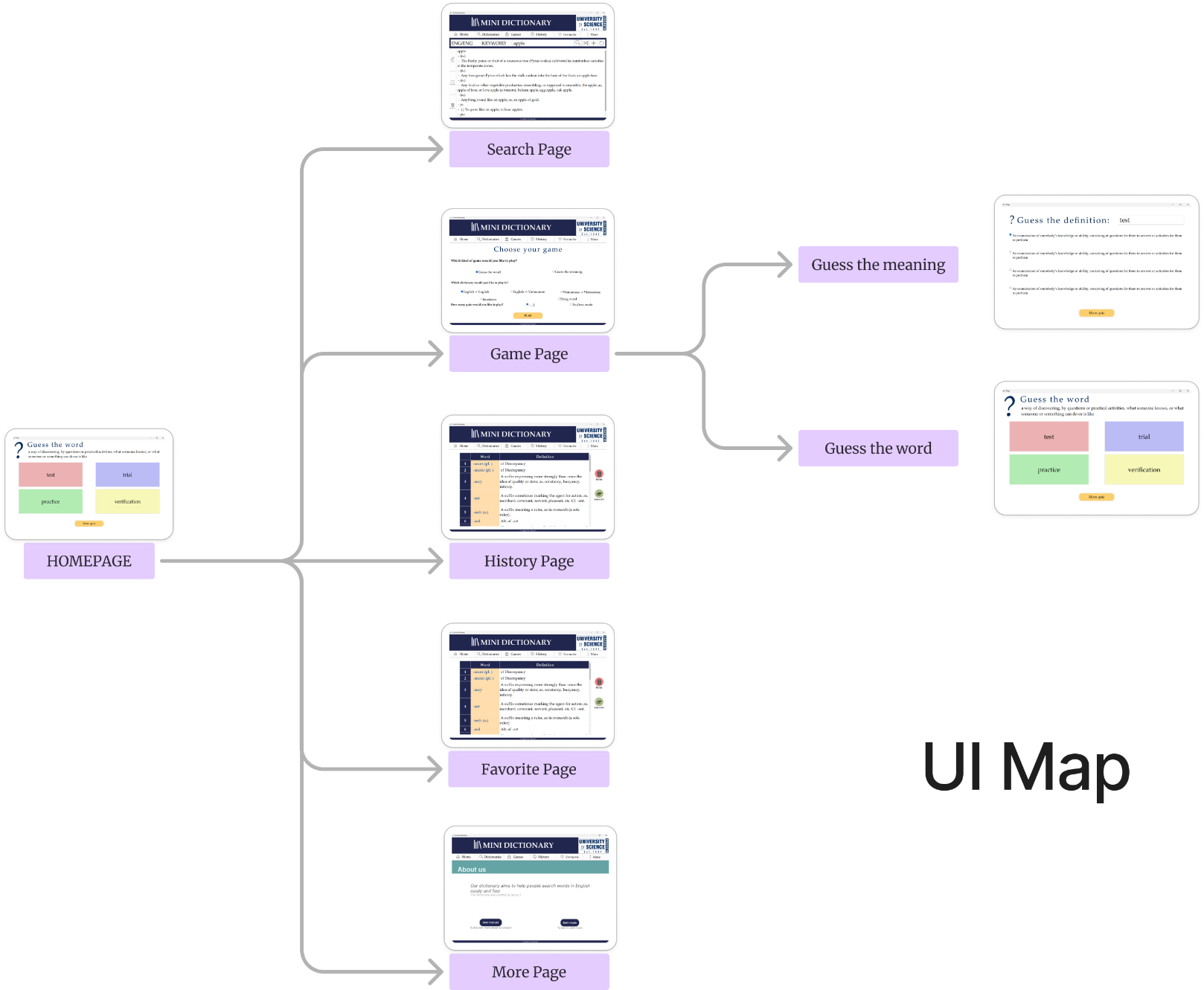


Figure n. UI Map

## Home Page



Figure n. Staff Home Screen

Navigation Bar: The navigation bar will always be on the top, under the Header.

* **Home button**: Return to Home Page (figure n, right)
* **Dictionaries button:** Return to Search Page
* **Games button:** Return to Choosing game Page
* **History button:** Return to History Page
* **Favourite button:** Return to Favourite Page
* **More button:** Return to More Page (contains additional setting of the dictionary and more information)

Word of the day: A random word every time you open the dictionary for a little fun 😊

Explore more: This section encourages you to explore more about the dictionary and its additional functions.

* **Word Quiz**: Return to Game Page
* **Favorite List:** Return to Favorite Page
* **Discovery**: Return to Search Page with a random word displayed
* **Edit your dictionary:** Return to Editing Page

## History/Favorite Page

History/Favorite List: The search word history or the favorite list will be present in a grid, the first column is the words, and the second column its meaning. You can choose one or multiple words, by either using your mouse or SHIFT and CTRL key. If the list is too long, there will be a scrollbar next to the list.



Figure n. Staff Home Screen

* **Select All**: Select all words present in the list.
* **Delete:** Delete all the selected words.

## Dictionary/Search Page

Search Bar: The search bar will always be under the navigation bar. This includes:

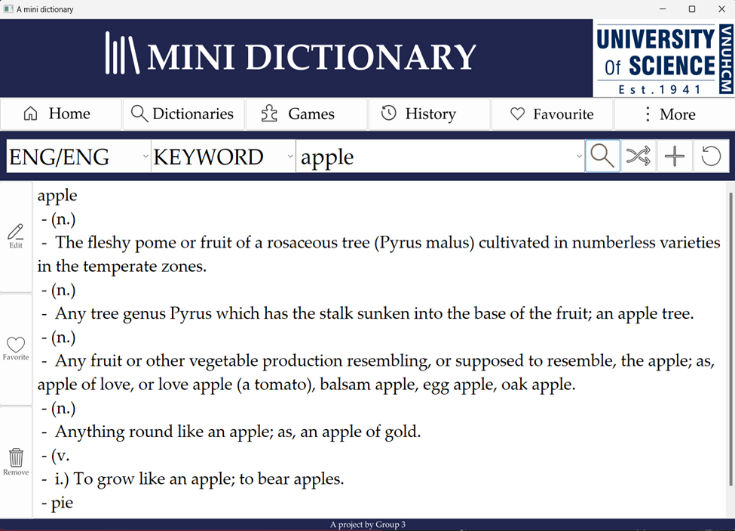


Figure n. Staff Home Screen

* **Choosing dictionary:** You can change your data set by clicking the button on the far left of search bar (default: ENG/ENG). It will drop down a list of dictionaries available. Choose your desired dictionaries by clicking on one of them.

There are 5 dictionaries available: English-to-Vietnamese, English-to-English, Vietnamese-to-English, Emoticon, and slang.

* **Choosing search function:** You can change your search function by clicking on the button next to the above button (Default: KEYWORD). It will drop down two options. Choose your desired search function by clicking on either KEYWORD or DEFINITION.
* **Searching:** Type your search into the search box. It will drop down a list of recommended words. Choose your desired word and press the Magnifying Glass button (right of search bar) to search.
* **Randomized word:** If you want to view a random, arbitrary word, press the Shuffle button next to the search button (right of search bar).
* **Add a word:** You can add a custom word to the dictionary by pressing the plus button (right of search bar) to return to Editing Page.
* **Reset the Dictionary:** To reset the dictionary to its original state, you can press the Repeat Button (far right of search bar). This will reset all the data in the dictionary.

Tool bar: The search bar will always be on the left of the screen. This includes:

* **Edit button**: Edit the definition of the currently displayed word.
* **Favorite button:** Add the currently displayed word into the favorite list (can be view in Favorite Page)
* **Delete button:** Delete the currently displayed word from the dictionary.

## Game Page



Figure n. Staff Home Screen

To choose your game: Press the buttons to customize your word game. This includes:

* **Choose the types of game**: We offer you two options: Guess the word (we show you a definition and 4 words, choose the correct answer among them), and Guess the meaning (we show you a word and 4 definitions, choose the correct answer among them)
* **Choose dictionary:** Choose one of the five available dictionaries for our game, you may only pick one.
* **Choose number of rounds:** Choose a number of rounds you one to participate in by typing in or clicking on the box (right of the question) or choose to play in Endless Mode. The game stops when you lose.

## More Page

This page: The search word history or the favorite list will be present in a grid, the first column is the words, and the second column its meaning. You can choose one or multiple words, by either using your mouse or SHIFT and CTRL key. If the list is too long, there will be a scrollbar next to the list.

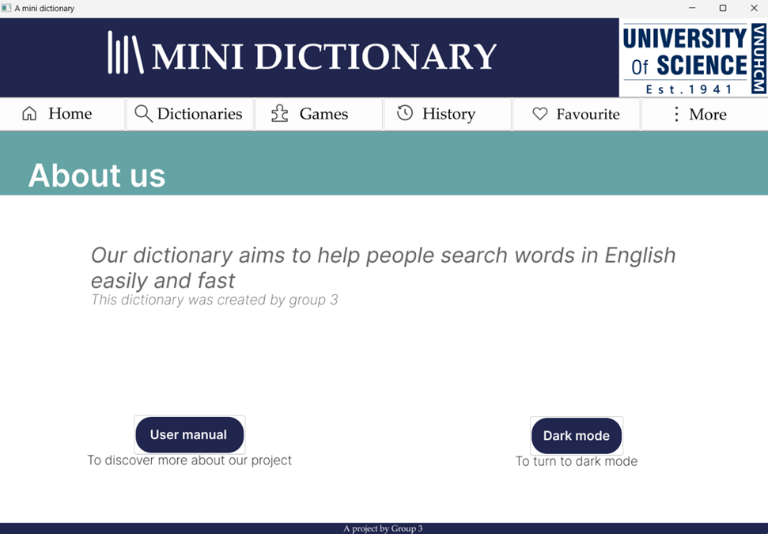


Figure n. Staff Home Screen

* **Select All**: Select all words present in the list.
* **Delete:** Delete all the selected words.

A screenshot of a computer game

Description automatically generated

A screenshot of a quiz

Description automatically generated

# Conclusion

The mini dictionary project successfully achieved its objectives of creating a functional and efficient dictionary application using C/C++. It was an exciting project to work on, and we gained a lot from it, above and beyond its original scope. We were able to study and research many different topics outside of our class.

Although the application is far from perfect, it works as we desired. The mini dictionary showcases the group's proficiency in data structures handling, and all the core and additional functionalities work successfully. While the project has met its initial goals, we believe there are a lot of potential enhancements for this application in the future, including but not limited to transforming the data sets into one unify format that is more easily and effectively processed, and more efficient use of data structures.

# References

[1] [wxWidgets User Manuals](https://docs.wxwidgets.org/3.2/)

[2] Ternary Tree

[3] Data Sets Resources