COMP90015 Distributed Systems

Assignment 1: Multi-threaded Dictionary Server

Yun Chen 1412174
The University of Melbourne
April 6, 2024

In the beginning

Because I started the design of the system very early, I added many out of scope elements, especially in the GUI part, where I added some extra designs. However, after clarifying the requirements through a few tutorials, I focused on the core needs, so the additional parts I designed earlier may have deficiencies or vulnerabilities.

Problem Context

In this project, I adopted a client-server architecture to design and implement a multi-threaded dictionary client-server system. The server of the system allows concurrent processing of multiple requests sent by clients, including adding, deleting, modifying, and querying the dictionary. This system uses the TCP protocol to provide a reliable communication connection between the server and clients. Also, Request and Response classes were built in the system, with information transfer in JSON format serving as the message exchange protocol between the server and clients. For fault handling, errors, including network connection errors and parameter mistakes, are properly managed on the client side. Additionally, the server can detect and correctly handle illegal requests from clients (such as adding words already in the dictionary, or deleting or querying non-existent words) and display user-friendly prompt messages on the client side.

System Components

This part will describe the components of the server and client, as well as the overall design goals and approach.

Server

The server uses a concurrent processing mechanism, where each network request is handled by a single execution thread, and it can detect and manage various types of error. When the client makes a request, the server creates a new thread to process and respond. Due to the use of multithreading technology, the server can concurrently handle requests from different clients. the server contains a *DictionaryService* class, which is used to manage dictionary data, including adding, updating, or deleting words and their meanings. All operations related to data modification are synchronized through a ConcurrentHashMap object within the *DictionaryService* class, ensuring that modifications to the dictionary data are out of conflict.

The *ClientHandler* class is used for handling connections and communication with the client, with each connection being processed in its own thread. The *MyThreadPool* class is used to manage the thread pool, optimizing resource usage and enhancing server performance. The *ServerMain* class serves as the entry point of the server, responsible for initializing the server and setting up the database connection. Through the *DatabaseConnection* class, the server can establish a connection with the database and perform database operations.

Client

The client primarily consists of three parts: the GUI, the GUI controller, and the network connection component.

The GUI is implemented using JavaFX, providing a GUI for the user. Besides accommodating the user's query and modification operations, it also guides users on how to use the client, such as asking for confirmation when they click the add or remove button, thereby enhancing user experience.

The GUI controller is responsible for processing the operational demands captured by the GUI. For add, delete, and query operations initiated by the user, the controller establishes a new TCP connection with the server and closes it after receiving a response.

The client also includes a ClientTask class responsible for specific network communication tasks, such as exchanging messages with the server.

Design Details

This part will delve into the detailed design aspects of the program, including the implementation logic of core functionalities, as well as the static and dynamic design models.

Server

When the server starts, it can accept two parameters: the port and the database file path. If these parameters are not provided, it will use the default values set in the project. Then, it verifies the format of the parameters. If the port number is out of range or the file path does not exist, it will throw an error.

To ensure data consistency and security, each time an add, delete , or update operation is performed, memory is locked first. I added the synchronized keyword to each CRUD operation to ensure that access to shared resources is mutually exclusive. Then, the data is updated in memory first, because memory operations are fast. After a successful memory update, the database is updated. The database synchronized storage ensures that data will not be lost even if the server shuts down unexpectedly. Upon the next startup, all data can be restored simply through the databas.

When the server receives a request from the client, it creates a new thread to handle each request. Depending on the command, the server processes the request in different ways and sends various statuses and feedback to the client. The JSON format for responses is defined in the following table.

Table 1: Server response format (JSON fields)

status	message

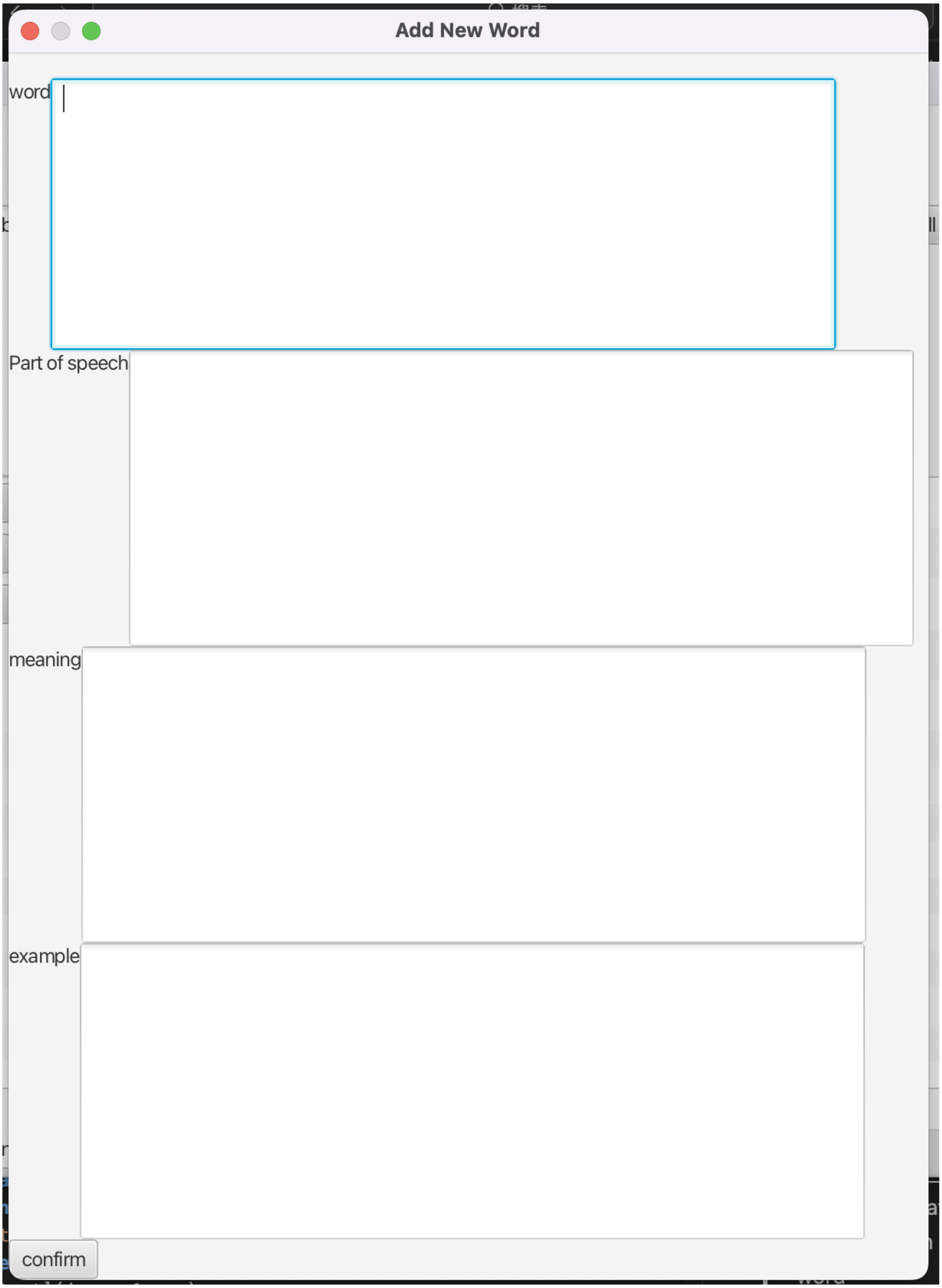
status	message
A string indicating the state	The meaning of a word(Only need in QUERRY mode, the other mode will return some string message)

Client

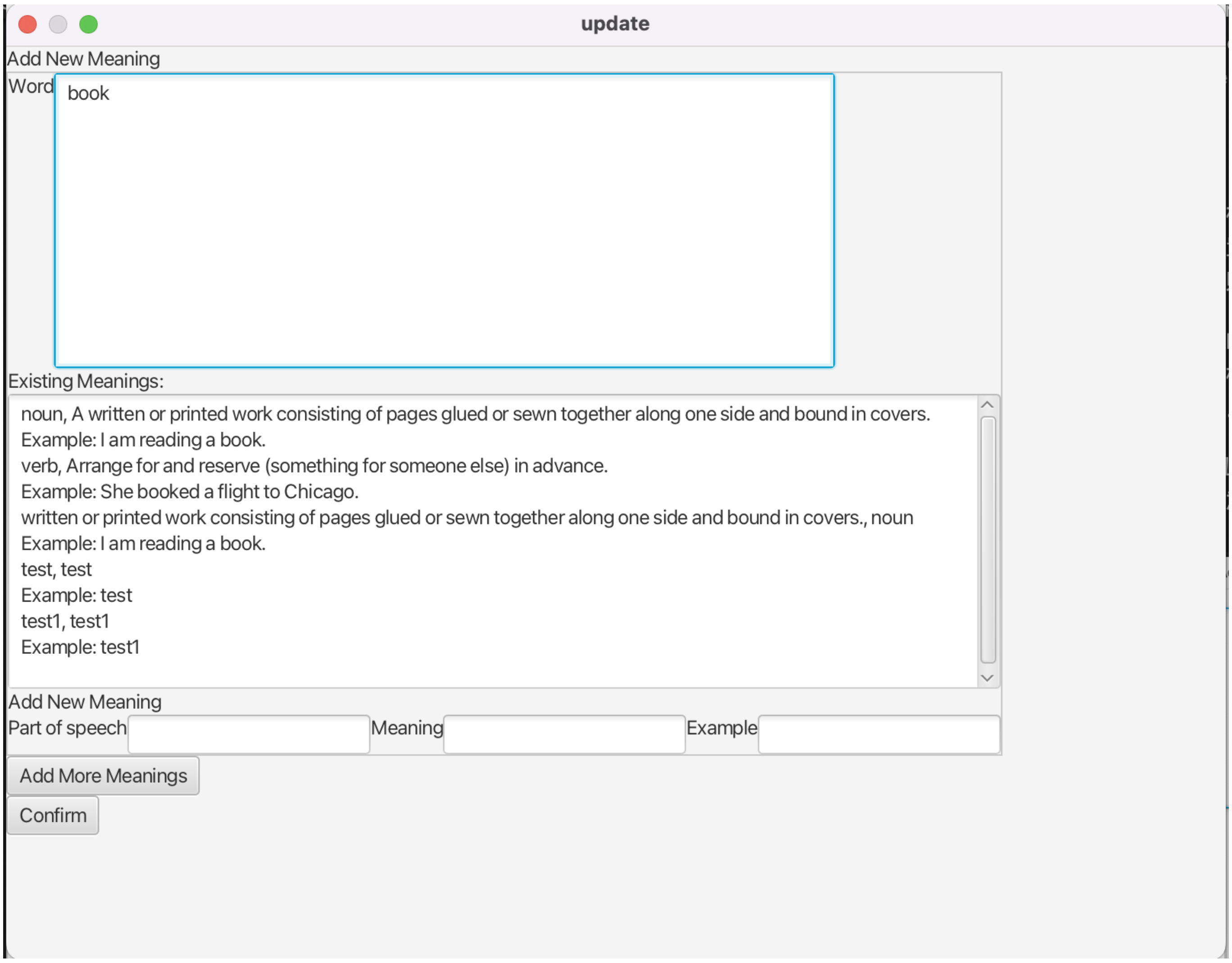
When the client starts, it can accept two parameters: the hostname and the port. If these parameters are not provided, it will use the default values set in the project. Then, it verifies the format of the parameters. If the port number is out of range, it will throw an error.

Some functions of the client, due to additional design, are detailed in the Creativity elements part. This part provides a brief overview of the core functional design of the client, including adding new words, updating word meanings, and deleting words.

In the GUI, after clicking the "add new word" button, a new window will pop up. In this window, you can sequentially add a word's part of speech, meaning, and example sentence. Note that you can add multiple meanings for a word by using the enter key to create new lines.

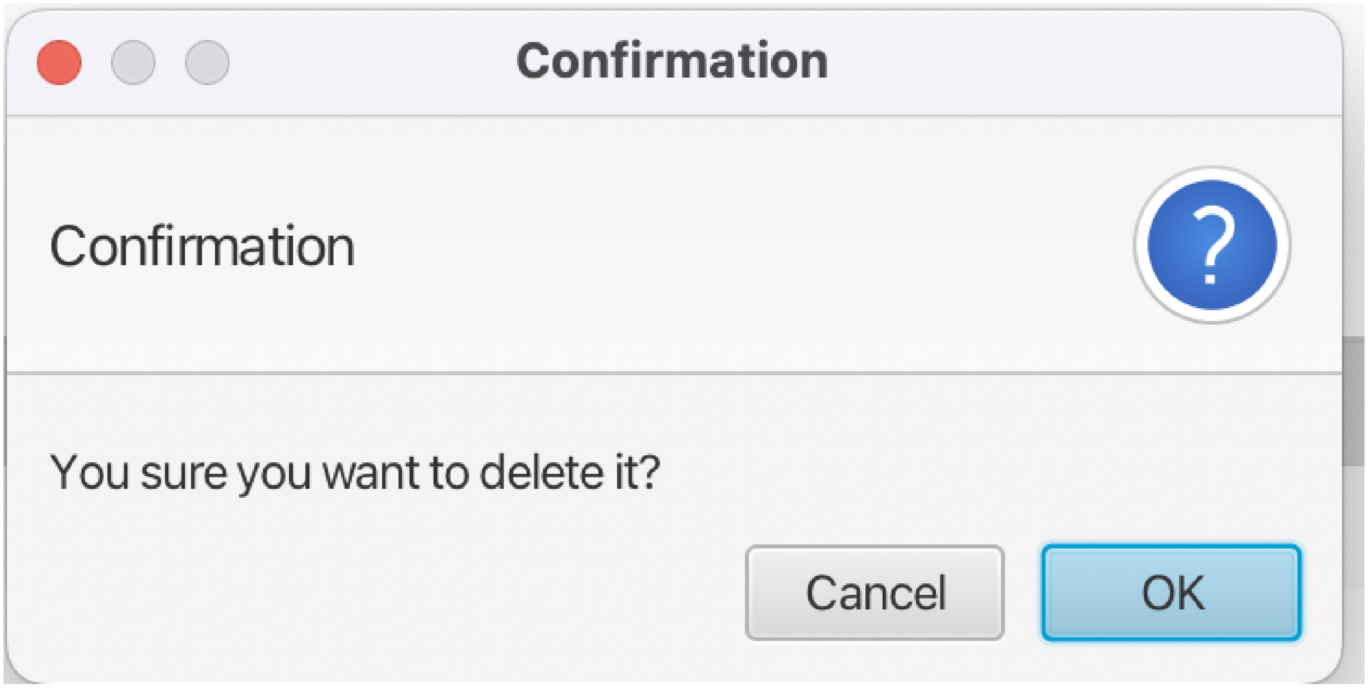


In the word list, there are two buttons behind each word: update and delete. After clicking the update button, the poping up interface will display the meanings that the word already has, to avoid the addition of duplicate meanings as much as possible. And when the delete button is clicked, the user is first asked to confirm the deletion to prevent accidental activation.



book

noun, A written or printed work consisting of pages glued or sewn together along one side and bound in covers.
Example: I am reading a book.
verb, Arrange for and reserve (something for someone else) in advance.
Example: She booked a flight to Chicago.
written or printed work consisting of pages glued or sewn together along one side and bound in covers., noun
Example: I am reading a book.
test, test
Example: test
test1, test1
Example: test1



Confirmation

You sure you want to delete it?

When user press the button in the GUI, client will send a request to the server. The JSON format for request is defined in the following table.

Table 2: Client request format (JSON fields)

command	word

command	word
A string indicating different command	A word(structure) for operation including every message of this word

Table 3: The data structure of Word

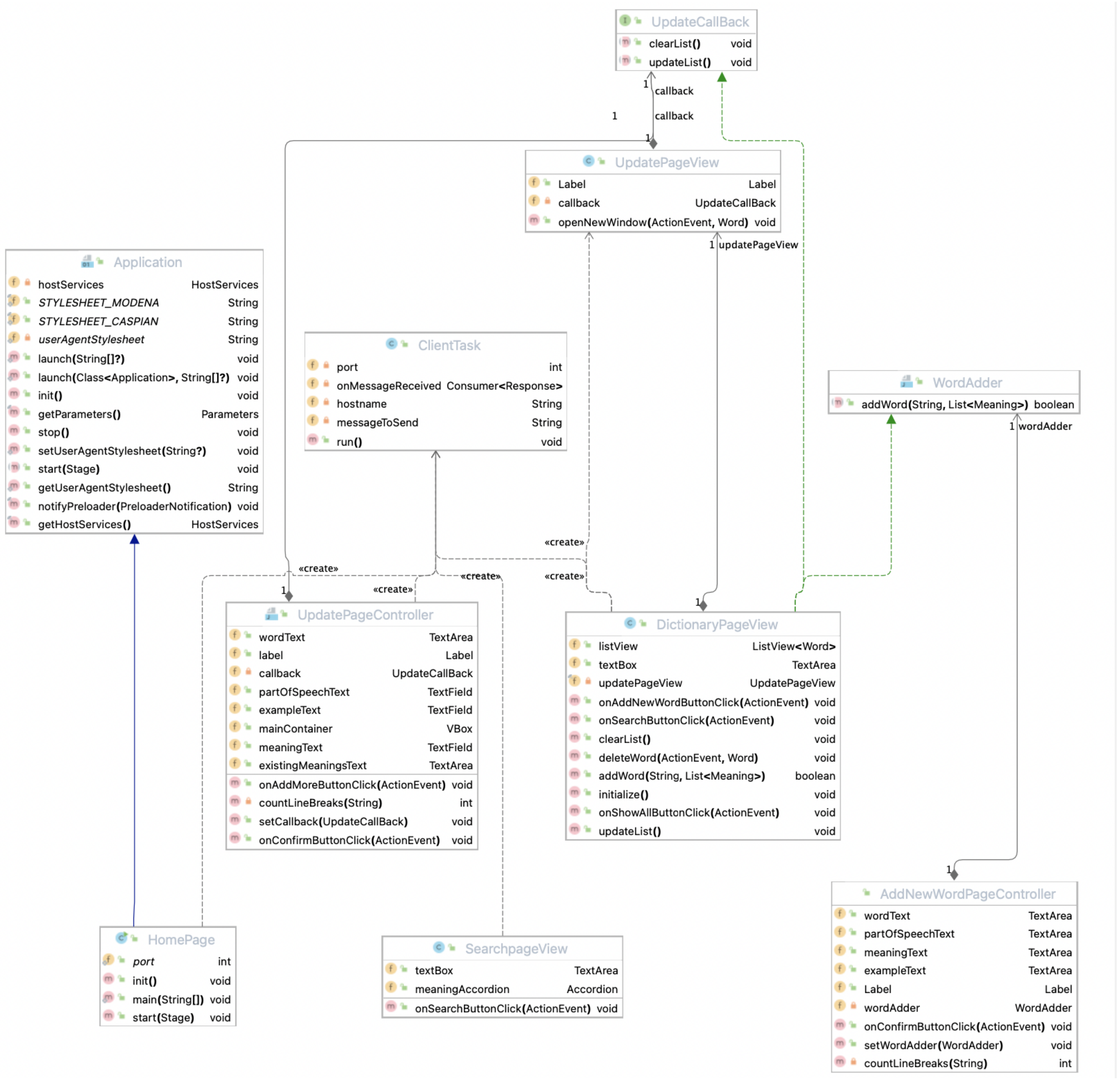
word	meanings
A string indicating name of the word.	A thread-safe list to store the meanings of this word.

Table 4: The data structure of Meaning

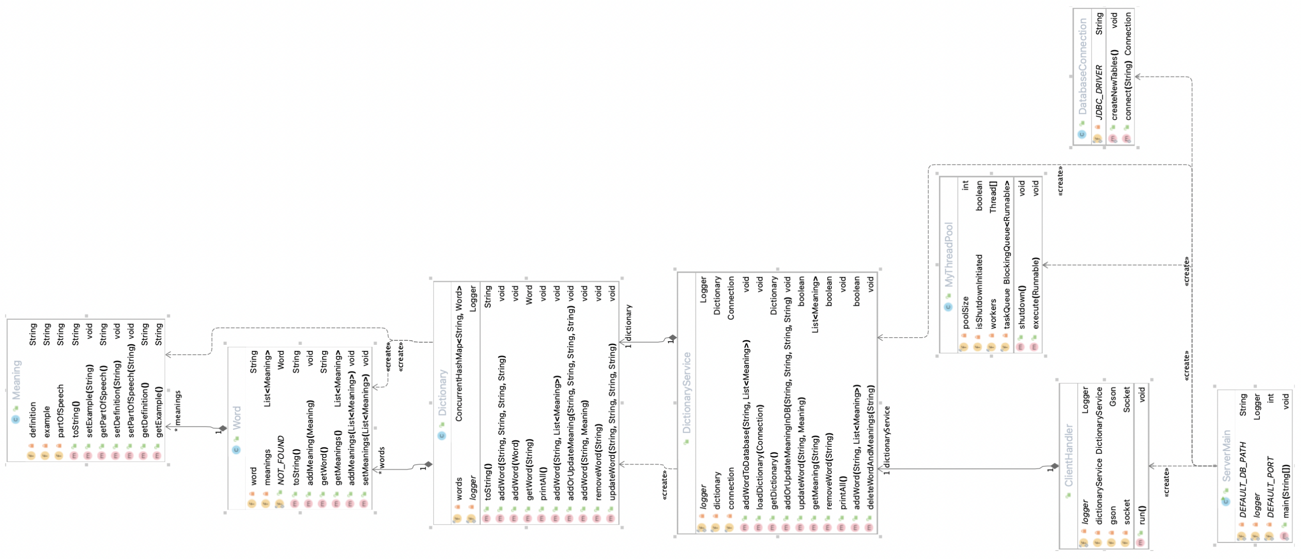
definition	example	partOfSpeech
A string indicating the defination of word	A string indicating the example sentence of word	A string indicating the part of speech of word

Static Design Model

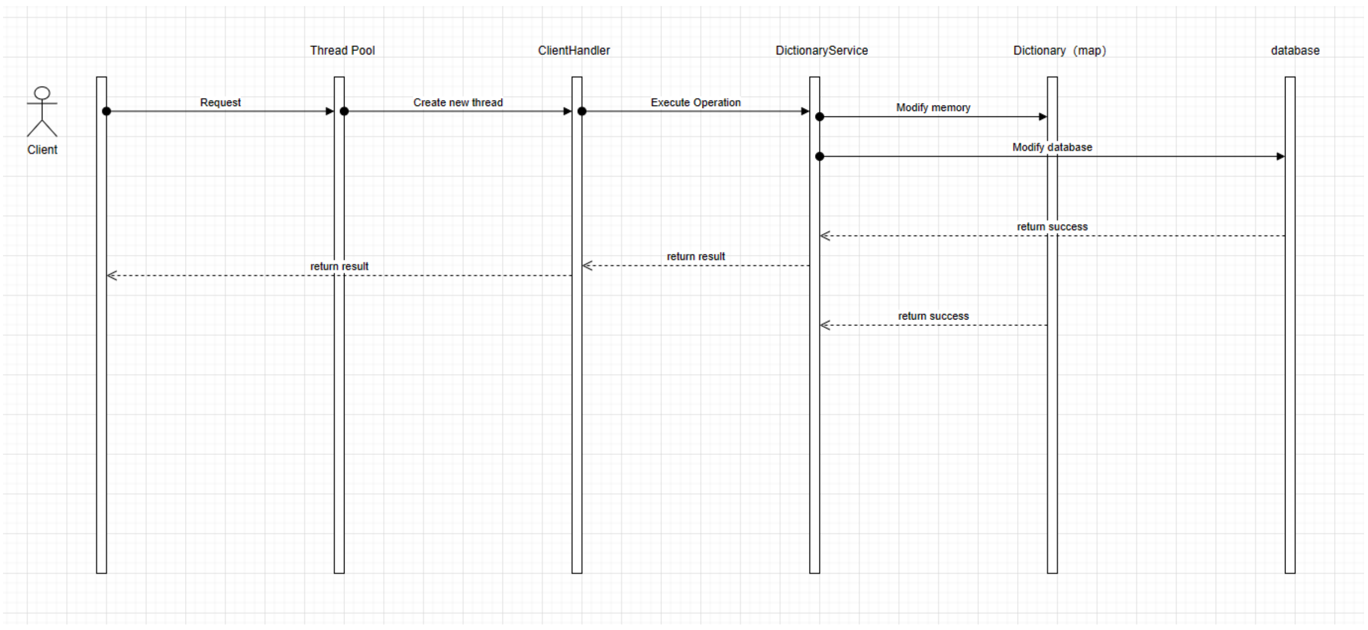
Client


UpdateCallBack
clearList() void
updateList() void
callback
callback
1
1
1
UpdatePageView
Label Label
callback UpdateCallBack
openNewWindow(ActionEvent, Word) void
1 updatePageView
Application
hostServices HostServices
STYLESHEET_MODENA String
STYLESHEET_CASPIAN String
userAgentStylesheet String
launch(String[]?) void
launch(Class<Application>, String[]?) void
init() void
getParameters() Parameters
stop() void
setUserAgentStylesheet(String?) void
start(Stage) void
getUserAgentStylesheet() String
notifyPreloader(PreloaderNotification) void
getHostServices() HostServices
ClientTask
port int
onMessageReceived Consumer<Response>
hostname String
messageToSend String
run() void
WordAdder
addWord(String, List<Meaning>) boolean
1 wordAdder
«create»
«create»
«create»
«create»
«create»
1
1
UpdatePageController
wordText TextArea
label Label
callback UpdateCallBack
partOfSpeechText TextField
exampleText TextField
mainContainer VBox
meaningText TextField
existingMeaningsText TextArea
onAddMoreButtonClick(ActionEvent) void
countLineBreaks(String) int
setCallback(UpdateCallBack) void
onConfirmButtonClick(ActionEvent) void
DictionaryPageView
listView ListView<Word>
textBox TextArea
updatePageView UpdatePageView
onAddNewWordButtonClick(ActionEvent) void
onSearchButtonClick(ActionEvent) void
clearList() void
deleteWord(ActionEvent, Word) void
addWord(String, List<Meaning>) boolean
initialize() void
onShowAllButtonClick(ActionEvent) void
updateList() void
1
AddNewWordPageController
wordText TextArea
partOfSpeechText TextArea
meaningText TextArea
exampleText TextArea
Label Label
wordAdder WordAdder
onConfirmButtonClick(ActionEvent) void
setWordAdder(WordAdder) void
countLineBreaks(String) int
HomePage
port int
init() void
main(String[]) void
start(Stage) void
SearchpageView
textBox TextArea
meaningAccordion Accordion
onSearchButtonClick(ActionEvent) void

Server



Dynamic Design Model



Critical Analysis

This part will describe how the system handles and responds to errors, as well as the system's advantage, disadvantage, and creativity elements.

Failure Handling

In my system, there are primarily two types of errors that can occur. The first is related to network connections, and the second is due to wrong user operations. For both types of errors, our system provides users with adequate error feedback and prompts.

1. Operational errors: These refer to instances where the user performs an incorrect operation. The existing issues description and their solutions are shown in Table 5.

Table 5: Operation errors and solution

Description	Solution
When adding a word, the user needs to correctly input three attributes of the meaning: part of speech, definition, and example sentence. An error will be reported if one is missing.	Pop-up warning window: No Meaning added
When adding a word, it is possible to add multiple meanings at same time. However, if one of the meanings lacks either the part of speech, definition, or example sentence, an error will be reported.	Pop-up warning window: Not a complete meaning structure, you need to check the missing part of speech or meaning or example!
An error is reported when the user adds a word that already exists.	Pop-up warning window: Add repeated words!
An error is reported when the user searches for a word that does not exist.	Pop-up warning window: No word found!
An error is reported when the user adds a new meaning to a word (update) and one of the required elements: part of speech, meaning, or example sentence, is missing.	Pop-up warning window: You can't add the empty meaning!
An error is reported when the user adds a meaning that already exists to a word.	Pop-up warning window: Update failed: The meaning already exists!

2. Network connection errors: In our testing environment, the host we connect to is often localhost, making the network connection very stable. However, in the real world, network connections are often unstable. Sometimes the server may crash, and sometimes there might be issues with the port. The existing problems and their solutions are shown in Table 6.

Table 6: Network connection errors and solution

Description	Solution
Network connection errors, such as the server not starting, connecting to the wrong host or port, or network interruptions	Pop-up error window: Network connection error, please check your network connection.
Unknown host error, using a non-existent domain name "nonexistent-hostname.invalid" in test	Pop-up error window: Unable to resolve the host address, please check if the network connection or server address is correct.
Request timeout, with a 10-second timeout added to a request in test	Pop-up error window: Network request timed out, please check your network connection and try again.

Analysis of the System

Advantage

- Heterogeneity: Using Java for both the client and server addresses the issue of heterogeneity, as the JVM offers fundamental cross-platform capabilities.
- Notification of Error: Most error that might arise during usage have been effectively managed. By providing feedback through the graphical user interface, users can readily identify any errors.
- Well-developed GUI：Ensure an intuitive and efficient experience for users. This GUI design facilitates operation, enhancing overall user satisfaction and productivity.
- Scalability: I designed the core data structure of the dictionary with a three-tier architecture: dictionary-word-meaning. This not only makes the structural hierarchy clearer but also facilitates future expansions and modifications. For example, if we want to add multiple example sentences to the same meaning, we only need to modify the content in the meaning class. Additionally, I created two classes, Request and Response, to serve as protocols for data transmission, which also makes it easier to modify the transmission protocol in the future.

Disadvantage

This system was designed by a student(me) who has just learned about multithreading, so there must have many disadvantage. Therefore, the following mainly describes the potential problems and directions for improvement that I have identified, which have not yet been addressed due to limited technical ability and time constraints.

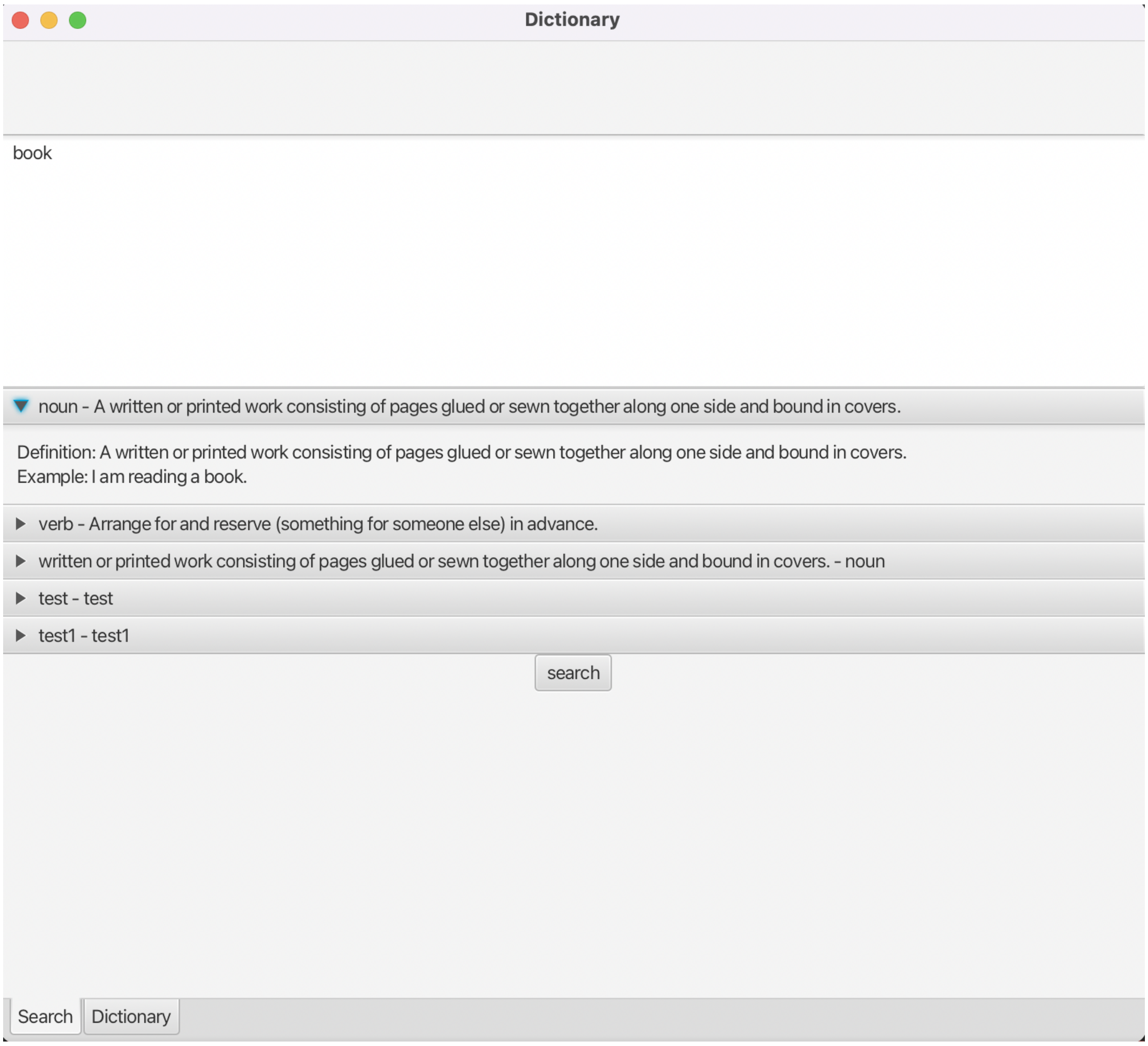
- The thread pool structure I designed is relatively simple, using a BlockingQueue to manage the task queue, which can efficiently handle the queuing and execution of tasks. However, it still lacks several key features. First, it lacks handling for exceptional situations in worker threads. If a worker thread encounters an exception while executing a task, it can cause the thread to terminate, and this thread will not be replaced or restarted. Secondly, there is no implementation for dynamic expansion and contraction of the thread pool. Once created, the size of the thread pool and the number of worker threads remain fixed.
- Security issue: In my system, I just used TCP for transmission, which means information is sent in clear text. This can be read and understood by any entity capable of monitoring network traffic, causing security risks. Also, there's no limit on user input in the frontend. Although I have added anti-injection in SQL statements to prevent attacks on the database, I haven’t designed a proper method to prevent memory leaks in the server.
- For the frontend interaction, there can be improved. Like adapting keyboard operations, such as tab switching and enter to confirm.
- Information synchronization. Although the server's data synchronization has been achieved through a multi-threaded server, the client cannot perceive the update of server data in real-time (needs to perform a query operation to perceive). If a subscribe/publish function could be implemented, it would allow the client's GUI to display new words directly whenever the server data is updated.

Excellence

Fancy client GUI design

For make user experience better, I put extra good things in GUI design.

1. I make the most important part of dictionary - the search - by itself on a separate screen. This way, users can use it a lot without other stuff getting in the way. And I add part of speech and example sentence to word meaning in data design, make this dictionary more like real-world dictionary.



noun - A written or printed work consisting of pages glued or sewn together along one side and bound in covers.

Definition: A written or printed work consisting of pages glued or sewn together along one side and bound in covers.
Example: I am reading a book.

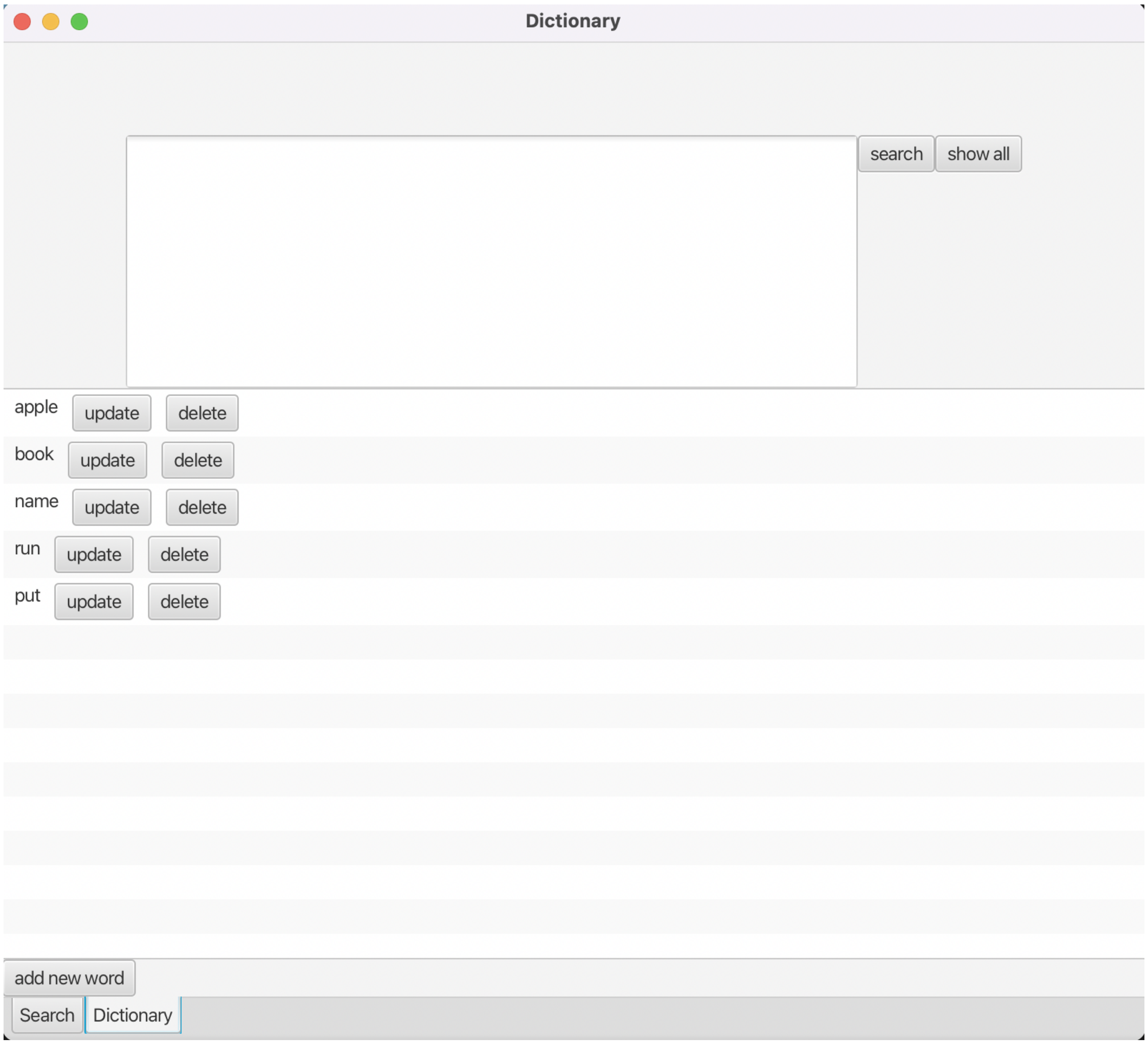
verb - Arrange for and reserve (something for someone else) in advance.

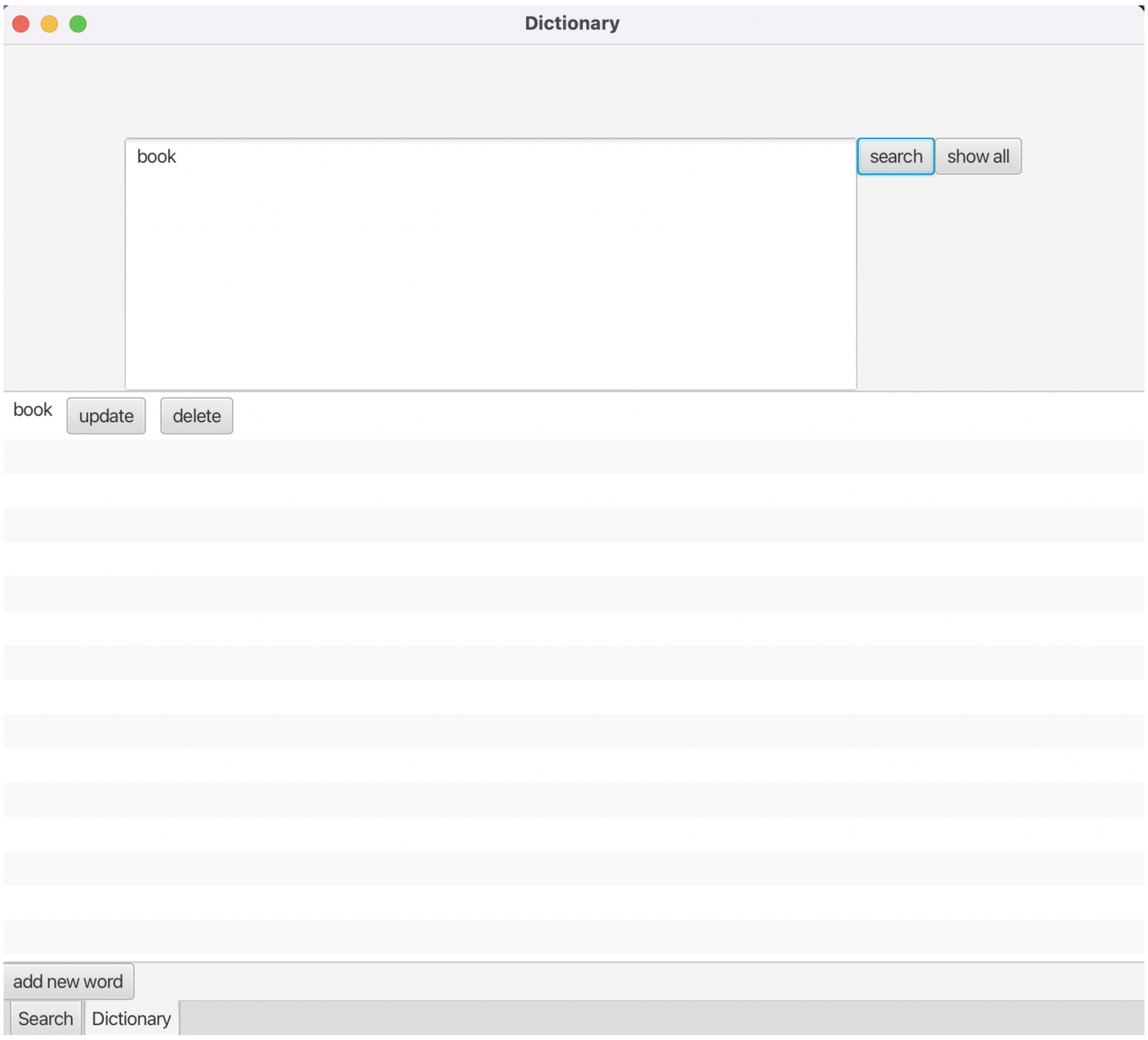
written or printed work consisting of pages glued or sewn together along one side and bound in covers. - noun

test - test

test1 - test1

2. I put all words in a scrollable list, easy for users see all dictionary clear and direct. Also add search thing in list, make users easy find the target word and update or delete it. After find word, can click show all to see all words again.


Dictionary
search
show all
apple update delete
book update delete
name update delete
run update delete
put update delete
add new word
Search
Dictionary



Creativity elements

My own thread pool design

In this system, I made a simple thread pool. This pool can do tasks at the same time and use resources again. It uses LinkedBlockingQueue for waiting tasks to be done, managing tasks that need to run well. When tasks come faster than pool can do, they wait in queue until a thread is free to do them. Also, my thread pool got a good stop way. When shutdown is called, pool finishes all tasks in queue first, then stops all working threads. This makes sure no task is lost when pool shuts down.