Project document

Exercise Notebook

Name: Nguyen Thu Trang
Topic: 334 - Exercise notebook
Student number: 100570029
Degree program: Data Science - Bachelor of Science and Technology
Year of studies: 2022 – 2025

1. General description

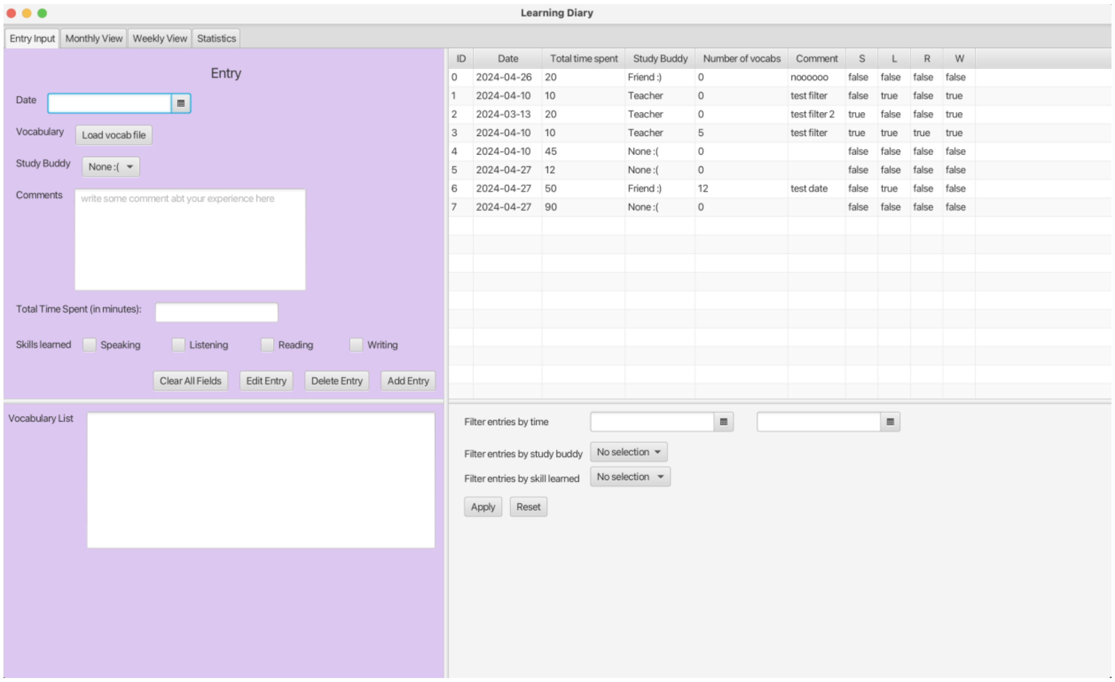
This will be a language learning diary application. The application's aim is to provide language learners with a diary of some sort to take notes and keep track of their progress.

Users can add an entry, delete an entry or edit an entry. For saved entries, users can also look up entries that have specific properties regarding date, study buddy, and skills learned. Users can also browse the entries in a calendar, and view some basic statistics.

2. User interface

Running the program:
Run the '**Main.scala**' file

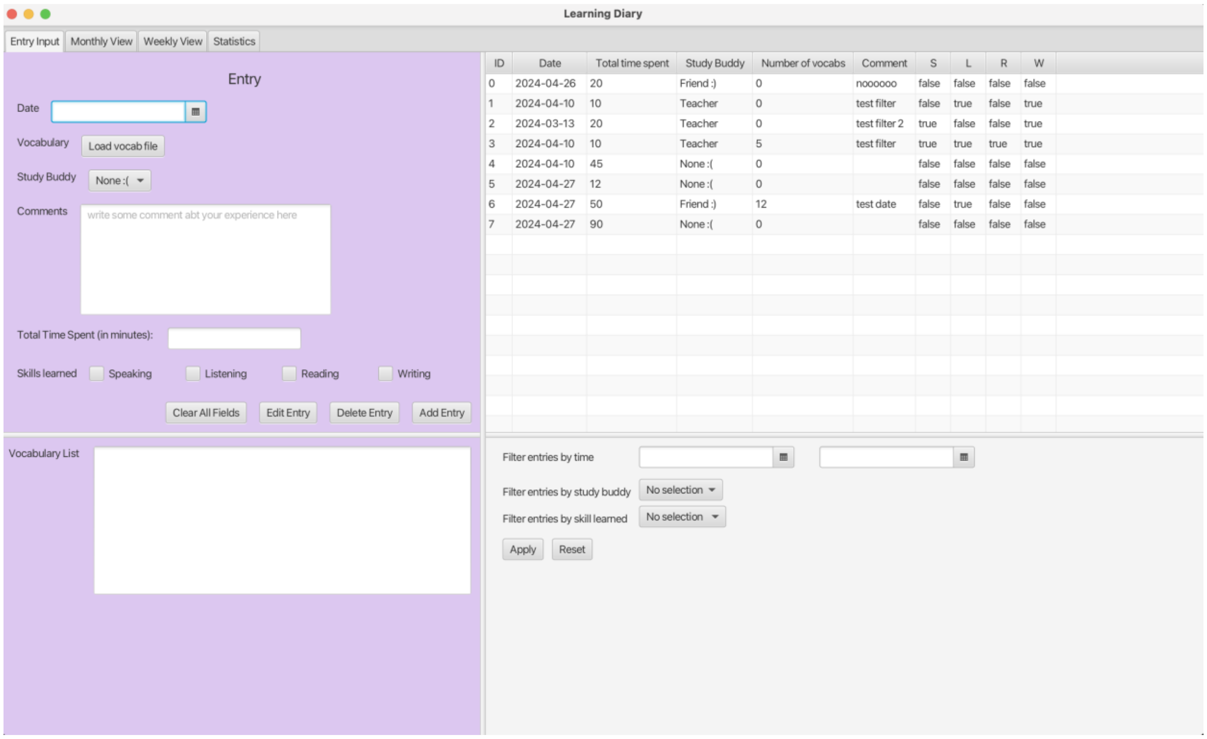
This is how the application looks on opening the application upon running the '**Main.scala**' file:



ID	Date	Total time spent	Study Buddy	Number of vocabs	Comment	S	L	R	W
0	2024-04-26	20	Friend :)	0	noooooo	false	false	false	false
1	2024-04-10	10	Teacher	0	test filter	false	true	false	true
2	2024-03-13	20	Teacher	0	test filter 2	true	false	false	true
3	2024-04-10	10	Teacher	5	test filter	true	true	true	true
4	2024-04-10	45	None :(	0		false	false	false	false
5	2024-04-27	12	None :(	0		false	false	false	false
6	2024-04-27	50	Friend :)	12	test date	false	true	false	false
7	2024-04-27	90	None :(	0		false	false	false	false

There are 4 tabs: Entry Input, Monthly View, Weekly View, and Statistics.

a) Entry Input



ID	Date	Total time spent	Study Buddy	Number of vocabs	Comment	S	L	R	W
0	2024-04-26	20	Friend :)	0	noooooo	false	false	false	false
1	2024-04-10	10	Teacher	0	test filter	false	true	false	true
2	2024-03-13	20	Teacher	0	test filter 2	true	false	false	true
3	2024-04-10	10	Teacher	5	test filter	true	true	true	true
4	2024-04-10	45	None :(	0		false	false	false	false
5	2024-04-27	12	None :(	0		false	false	false	false
6	2024-04-27	50	Friend :)	12	test date	false	true	false	false
7	2024-04-27	90	None :(	0		false	false	false	false

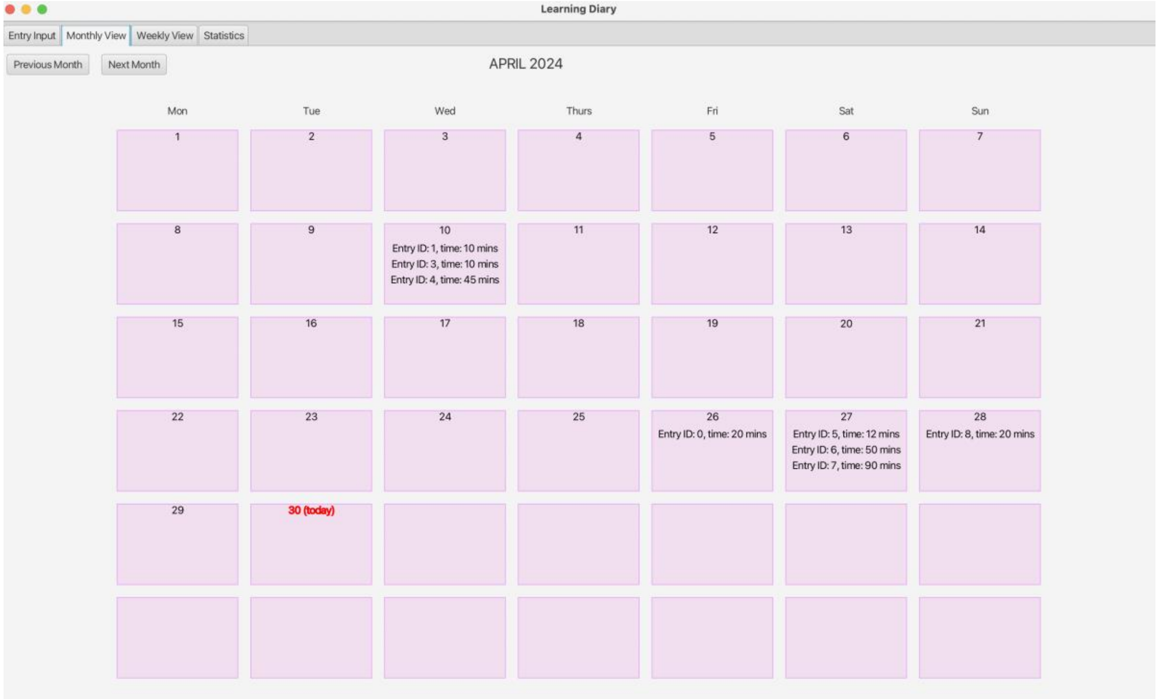
Most of the usage of the application happens here.

+ On the left side, there is an Entry Input form from which the user can pick the date and study buddy, type in comments, total time spent, and tick the skills learned that day. Furthermore, the users can also upload a vocabulary text file (the format will be shown below) if they want. Two mandatory input fields are date and total time spent.

+ On the right side, there is an Entry Table where all the entries are recorded. When users click on an entry, all the information will be shown on the left (including the vocabulary list). The users can make changes to an already saved entry this way, but they cannot change the vocabulary once saved. After making changes, users may click on 'Edit Entry' for the changes to be saved. Users can also delete an entry by choosing an entry from the table, and then click on 'Delete Entry'.

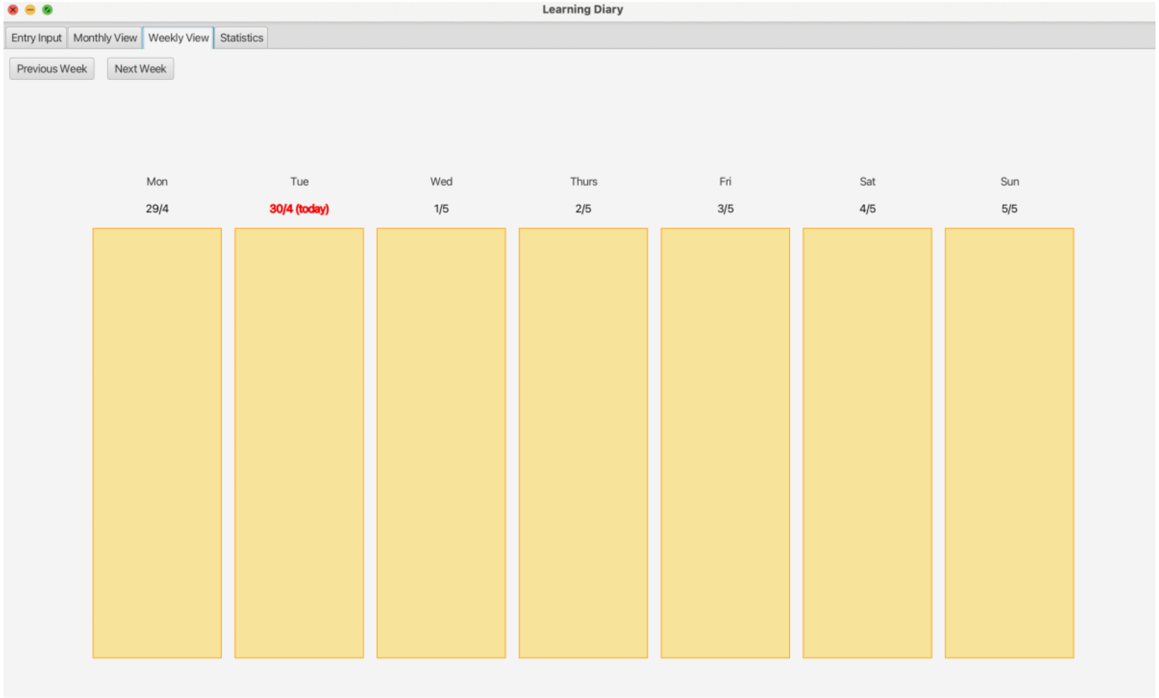
+ In the bottom right corner, there is a look-up area. Users can filter entries by date intervals, study buddy, and skills; or by any combination of those three properties. After choosing the desired properties, users may click on 'Apply' for the filtered entries to show up in the table, and then if they want to reset all the entries, they may click on 'Reset'.

b) Monthly View and Weekly View



APRIL 2024

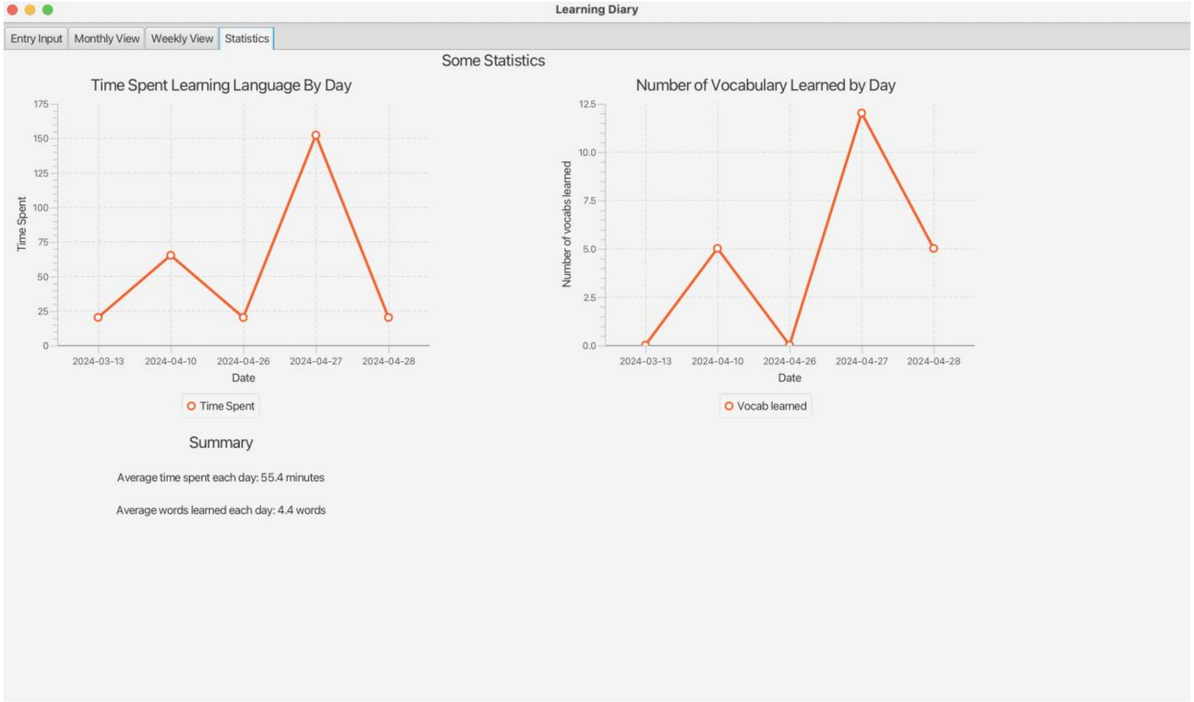
Mon	Tue	Wed	Thurs	Fri	Sat	Sun
1	2	3	4	5	6	7
8	9	10 Entry ID: 1, time: 10 mins Entry ID: 3, time: 10 mins Entry ID: 4, time: 45 mins	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26 Entry ID: 0, time: 20 mins	27 Entry ID: 5, time: 12 mins Entry ID: 6, time: 50 mins Entry ID: 7, time: 90 mins	28 Entry ID: 8, time: 20 mins
29	30 (today)					



Mon	Tue	Wed	Thurs	Fri	Sat	Sun
29/4	30/4 (today)	1/5	2/5	3/5	4/5	5/5

Both Views allow users to browse the entries chronologically, by month and by week. Basic information about the entries, including ID and time spent, are shown in the calendar. Users can navigate through different months/weeks by clicking on the Previous Month/Next Month or Previous Week/Next Week buttons on the top left of the application.

c) Statistics



Summary

Average time spent each day: 55.4 minutes

Average words learned each day: 4.4 words

This tab shows some basic brief statistics on the learning journey. There are two line graphs that respectively show time spent learning by day, and number of vocabularies learned by day. Furthermore, average time spent, and average words learned each day is also presented in the 'Summary' section. I have also decide to remove skills as each class, instead only implementing the skills list as a list of String.

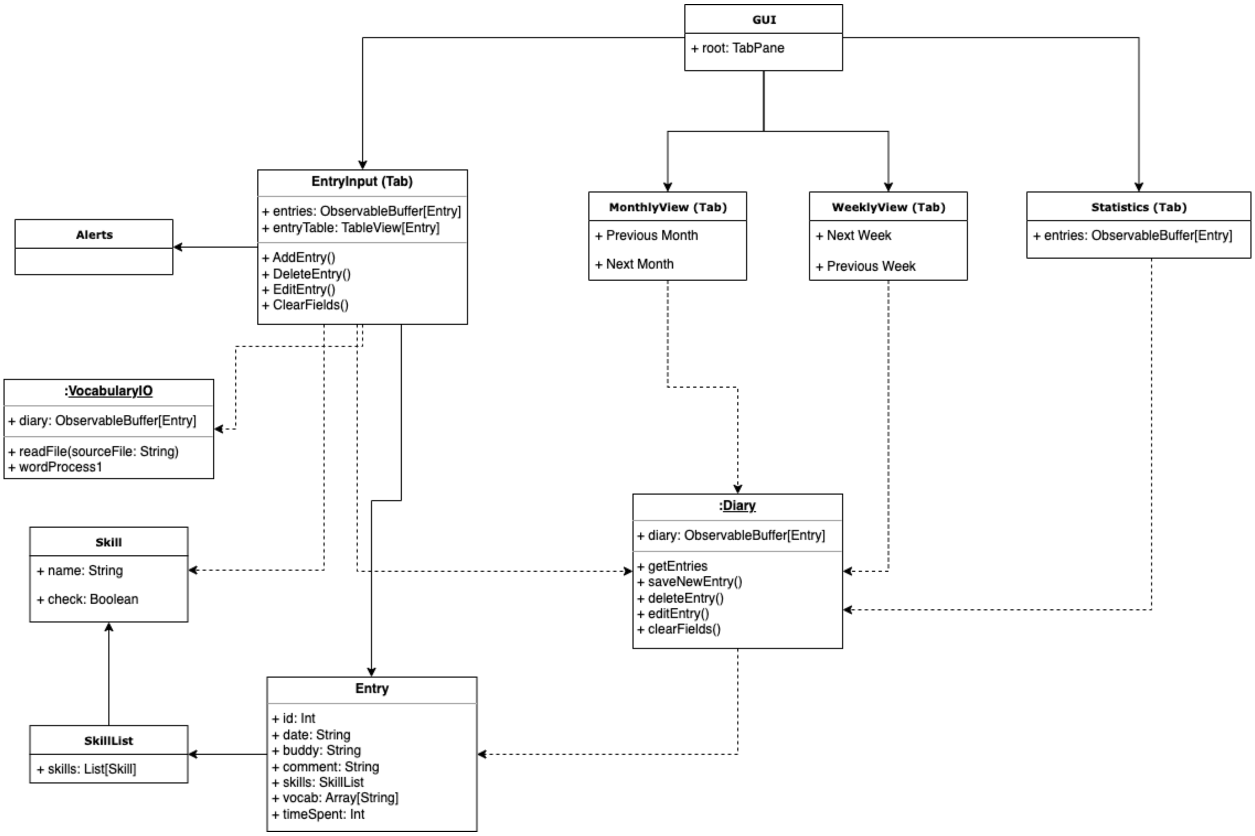
3. Program structure

The program is split into 2 main subparts: the GUI and the internal workings. The GUI is comprised of 4 classes: EntryInput, MonthlyView, WeeklyView, and Statistics. Each class is an extension of the Tab class, since the whole GUI is a TabPane.

The internal workings consist of the object Diary, and case classes Entry, SkillList, Skill that support the JSON file reading/saving functionality. Furthermore, there is the object VocabularyIO used by the EntryInput class to read the vocabulary file.

There have been many changes to the UML compared to the start of the project. For example, there's no Calendar class anymore, as I utilized the java.time library, and instead of individual class for each skill, they are now represented as a list of Skill (with different name) in SkillList. Further changes to classes and other GUI components are mentioned below.

UML Diagram



The class Skill doesn't have timeSpent and comment, instead there's a timeSpent variable for the whole entry. I have also decided to remove test/exercise scores due to time constraints on my part. The GUI also becomes simpler, with most of the functions on the EntryInput Tab.

4. Algorithms

a) Monthly Calendar View

I had much difficulty at first trying to implement the monthly calendar view, since it needs some algorithm for the days in a month to be displayed correctly. First, since a month has at most 31 days, and a week is 7 days, I created a gridpane and loop through each child node by row (i) and then column (j) from 0 to 6. Then I calculated the offset, which is the number of days that the first day of the month is away from Monday. For example, if the 1st falls on Tuesday, the offset would be 1, if it falls on Wednesday, the offset would be 2, et cetera.

Next, for each day, I calculate the date by using this formula: **date = (j+1) + (7*i) - 7**. Since the first row is reserved for naming the days, I must take 7 away from the whole calculation for the correct date. If date is larger than the offset, the current date would be calculated as follows:
current date = date – offset

Finally, since the current date must be smaller than the current month’s number of days, I need to check that condition before printing the date onto the calendar.

b) Statistics

For ‘average time spent each day’: Take the sum of total time spent of all entries and divide by the number of days. If there are multiple entries in a day, it’s still only count as 1 day.

For ‘average words learnt each day’: Take the sum of total words learned and divide by the number of days.

For the graphs, first I create a Map with date as key, and loop through the entries for the time/vocab values. Then, I sort the Map by date, and calculate the sum for each date.

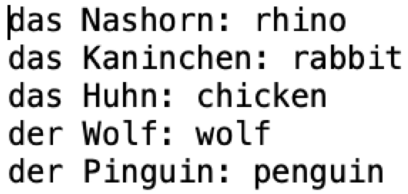
5. Data Structures

In the technical plan, I mentioned using dynamic structures like Arrays and Buffers, and I did use them most of the time. Furthermore, I also use ObservableBuffer to store entries in the GUI for easier manipulation within the EntryInput Tab, and for easier file reading/saving with upickle.

6. Files and Internet access

a) Vocabulary file

Whenever the user wants to include a vocabulary file, they can click on the button ‘Load vocab file’ and choose a text file with the following format: on each line, the words they want to save will be on the left-hand side of the colon, and anything else would be on the right side. An example of a vocabulary text file would be as follows:

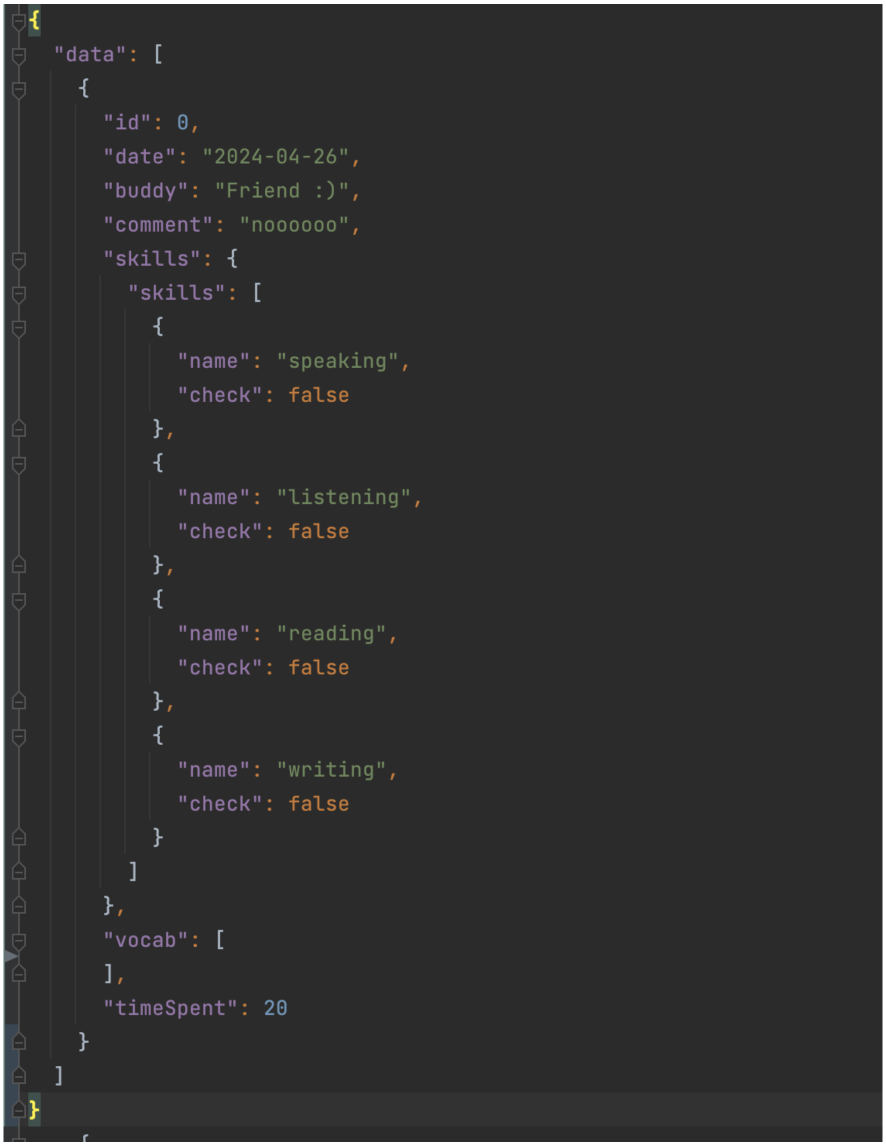


```
das Nashorn: rhino
das Kaninchen: rabbit
das Huhn: chicken
der Wolf: wolf
der Pinguin: penguin
```

In the end, all the words on the left-hand side of the colon will be saved into an array, allowing users to revise later. Two example text files are already created in *src/main/scala/resources*

b) Data

The resources folder consists of the ‘diary.json’ file, where all the entries' data is saved. The program reads the data and writes the data from/to this file. The structure is a single array which contains the entries as objects with all the required properties. An example of the planned file structure is as follows.



```
{
  "data": [
    {
      "id": 0,
      "date": "2024-04-26",
      "buddy": "Friend :)",
      "comment": "noooooo",
      "skills": {
        "skills": [
          {
            "name": "speaking",
            "check": false
          },
          {
            "name": "listening",
            "check": false
          },
          {
            "name": "reading",
            "check": false
          },
          {
            "name": "writing",
            "check": false
          }
        ]
      },
      "vocab": [
      ],
      "timeSpent": 20
    }
  ]
}
```

7. Testing

I mainly tested the program through different inputs through the GUI, since the application depends heavily on different users' inputs. Whenever I try to save the entry, I will investigate the Entry Table on the right-hand side, since the table will change the data accordingly to whenever I try to save, edit, or delete an entry. This method also applies to when I tested for the filtering functionality, since the filtered entries also appear in the table.

I didn't implement any unit tests, and I think this is the biggest shortcoming that this project has.

8. Known bugs and missing features

Saving Data

While testing on my own computer, I have found out that the program doesn't save the data immediately, and sometimes even when I restart the program the changes wouldn't show in the Entry Table. This is resolved by manually choosing the option 'reload from disk' for the 'diary.json' file before opening the application. Furthermore, this applies for both monthly and weekly calendar view, along with the statistics, since all these tabs only read the data at the start of the application.

However, when I tested my application on the school's computer (on both Linux and Windows), I don't have to choose 'reload from disk' upon restarting the application for the changes to be saved, so I don't know if it's a Mac OS problem with the IntelliJ IDE or not. All in all, I suggest doing the reloading from disk for the file 'diary.json' for utmost certainty.

DatePicker

While browsing through the documentation for DatePicker, I found out that it doesn't have any methods equivalent to clearing the inputs. However, DatePicker can only hold null values and/or text values at the start, and if users have picked a date, the date will remain the same unless users pick another date, regardless of the text in the input field of the DatePicker. This makes handling different inputs for datepicker quite difficult, and there may be some problems while editing date with already saved entries.

9. 3 best sides and 3 weaknesses

3 best sides

1. The UI is easy to use
2. Users can filter the entries by time interval, study buddy, and skills; any combination of those three properties are also doable.
3. All the information can be entered on a form.

3 weaknesses

1. There are no unit tests.
2. The calendar views and statistics don't update immediately whenever the data is changed.
3. No changes to vocabulary list once saved.

10. Deviations from the plan, realized process and schedule

I didn't manage to follow the schedule I set at first at all. Instead of implementing the logic behind the application first, I have implemented GUI first, since this is a text-based program, and it would be easier to test that way. The UI took considerably longer than I expected, especially the Entry Table and the Monthly Calendar View. Especially for the Monthly Calendar, I think I have tried at least three ways before heading the correct direction. Another process that took considerably long is the JSON reading/saving functionality. Since I didn't understand the upickle document on creating customized keys for case classes that the library doesn't support, I also spent much time figuring out how to do it.

After these three have been implemented, the rest come rather smoothly. Overall, I spent quite some time trying to familiarize myself with the scalafx and javafx documentation, since I used both libraries during implementation and there are quite some inconsistencies between those two, and then started seriously implementing the application around late March.

11. Final evaluation

Overall, I'm satisfied with the effort that I have put into this project. Even though the application is rather simple, I think it is very intuitive and easy to navigate and use, with most of the basic functions of a normal diary: Save/Edit/Delete an Entry, looking up entries based on certain properties, and some statistics.

In retrospect, I could have differentiated between the GUI and the internal workings more clearly, instead of implementing almost everything right inside the GUI. Some of the implementations could have been more efficient, and the case classes could have been more clearly divided.

The Teaching Assistant, Aino, was really helpful, and I am eternally grateful for her guidance.

12. References

Haoyi, Li (2023) uPickle 3.3.0. [online] Available at: https://com-lihaoyi.github.io/upickle/

Jenkov, Jacob (2020) JavaFX Tutorial. [online] Available at: https://jenkov.com/tutorials/javafx/index.html

Lewis, M.C. and Lacher, L.L. (2016). Introduction to programming and problem-solving using Scala. Boca Raton: Chapman & Hall/CRC.

Package java.time. [online] Available at: https://docs.oracle.com/javase/8/docs/api/java/time/package-summary.html

Scalafx [online] Available at: https://javadoc.io/doc/org.scalafx/scalafx_3/latest/index.html

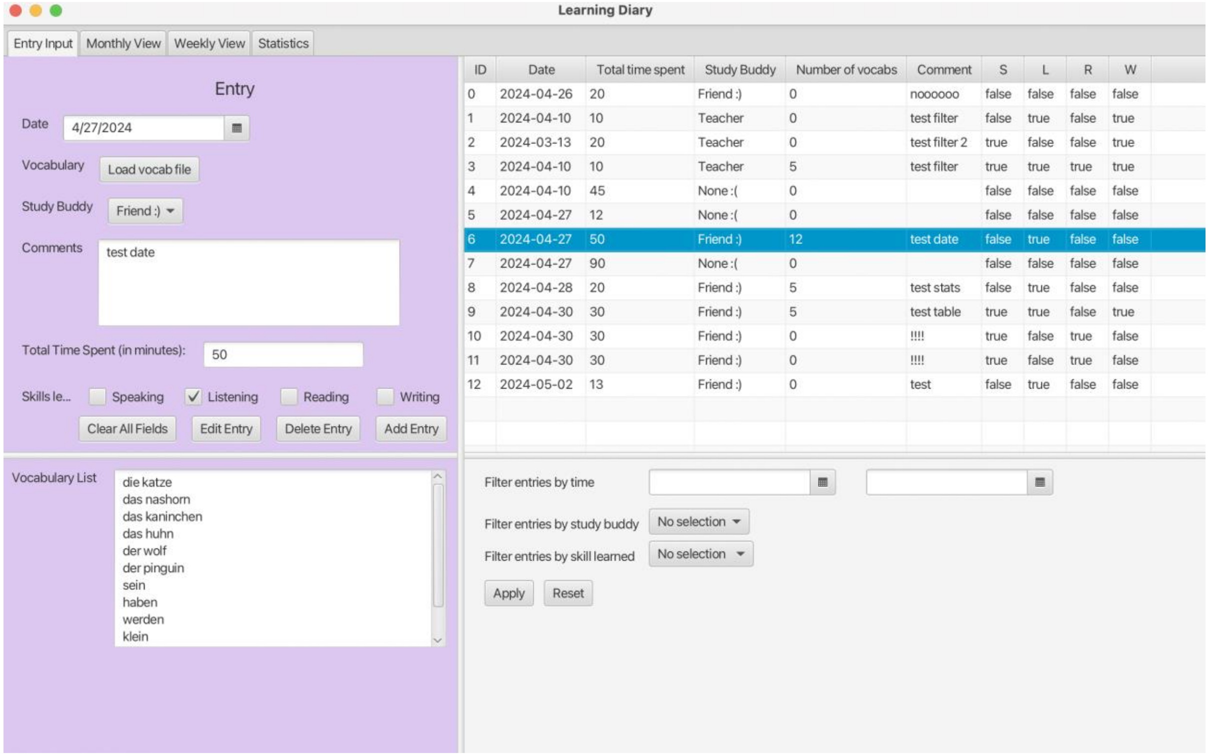
13. Appendices

[1] Project source code: https://version.aalto.fi/gitlab/nguyent119/language-diary

[2] Examples of some functionalities

a) Editing entries

Start by choosing an entry from the EntryTable. For instance, here we choose entry with ID #6. The information will show up on the left.



Entry

Date 4/27/2024

Vocabulary

Study Buddy Friend :)

Comments test date

Total Time Spent (in minutes): 50

Skills le... □ Speaking ☑ Listening □ Reading □ Writing

Vocabulary List
die katze
das nashorn
das kaninchen
das huhn
der wolf
der pinguin
sein
haben
werden
klein

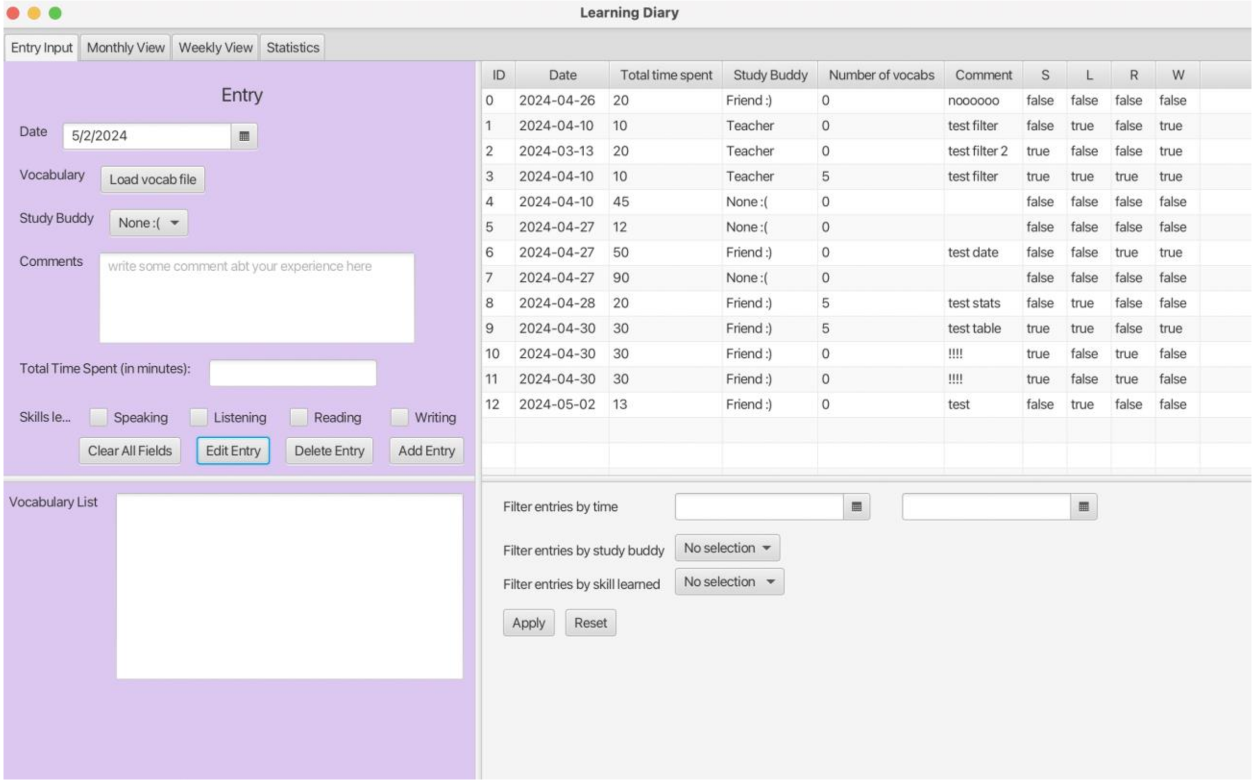
ID	Date	Total time spent	Study Buddy	Number of vocabs	Comment	S	L	R	W
0	2024-04-26	20	Friend :)	0	noooooo	false	false	false	false
1	2024-04-10	10	Teacher	0	test filter	false	true	false	true
2	2024-03-13	20	Teacher	0	test filter 2	true	false	false	true
3	2024-04-10	10	Teacher	5	test filter	true	true	true	true
4	2024-04-10	45	None :(	0		false	false	false	false
5	2024-04-27	12	None :(	0		false	false	false	false
6	2024-04-27	50	Friend :)	12	test date	false	true	false	false
7	2024-04-27	90	None :(	0		false	false	false	false
8	2024-04-28	20	Friend :)	5	test stats	false	true	false	false
9	2024-04-30	30	Friend :)	5	test table	true	true	false	true
10	2024-04-30	30	Friend :)	0	!!!!	true	false	true	false
11	2024-04-30	30	Friend :)	0	!!!!	true	false	true	false
12	2024-05-02	13	Friend :)	0	test	false	true	false	false

Filter entries by time

Filter entries by study buddy No selection

Filter entries by skill learned No selection

We will edit the skills. We remove listening, then add reading and writing. After making the changes that we want, click on “Edit entry” for the entry to be saved with the new information. Here’s how it looks after we click on edit with the modified information:



Date: 5/2/2024
Vocabulary
Study Buddy: None :(
Comments: write some comment abt your experience here
Total Time Spent (in minutes):
Skills le... □ Speaking □ Listening □ Reading □ Writing

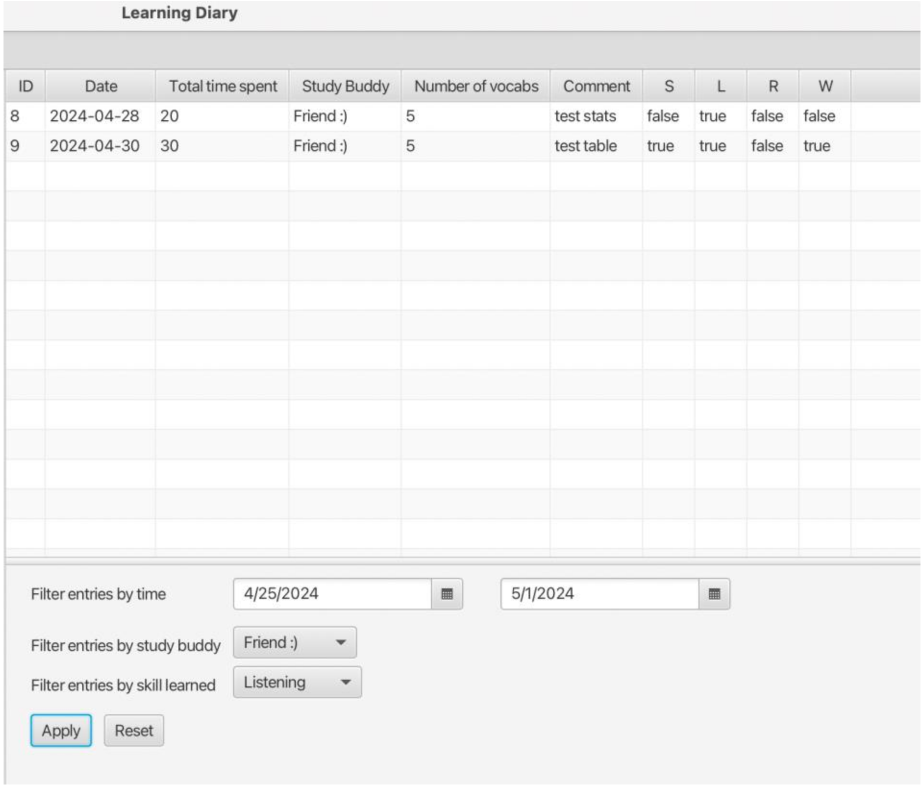
ID	Date	Total time spent	Study Buddy	Number of vocabs	Comment	S	L	R	W
0	2024-04-26	20	Friend :)	0	noooooo	false	false	false	false
1	2024-04-10	10	Teacher	0	test filter	false	true	false	true
2	2024-03-13	20	Teacher	0	test filter 2	true	false	false	true
3	2024-04-10	10	Teacher	5	test filter	true	true	true	true
4	2024-04-10	45	None :(	0		false	false	false	false
5	2024-04-27	12	None :(	0		false	false	false	false
6	2024-04-27	50	Friend :)	0	test date	false	false	true	true
7	2024-04-27	90	None :(	0		false	false	false	false
8	2024-04-28	20	Friend :)	5	test stats	false	true	false	false
9	2024-04-30	30	Friend :)	5	test table	true	true	false	true
10	2024-04-30	30	Friend :)	0	!!!!	true	false	true	false
11	2024-04-30	30	Friend :)	0	!!!!	true	false	true	false
12	2024-05-02	13	Friend :)	0	test	false	true	false	false

Filter entries by time
Filter entries by study buddy: No selection
Filter entries by skill learned: No selection

We can see that in columns ’R’ and ’W’ for entry #6, the values are now both true, while in column ’L’ it is false, just like we want.

b) Filtering entries

Continuing from above, we want to filter out entries from 25/04/2024 to 01/05/2024, with study buddy Friend, and skills learned Listening. We then click on Apply, and the result is shown below:



Learning Diary

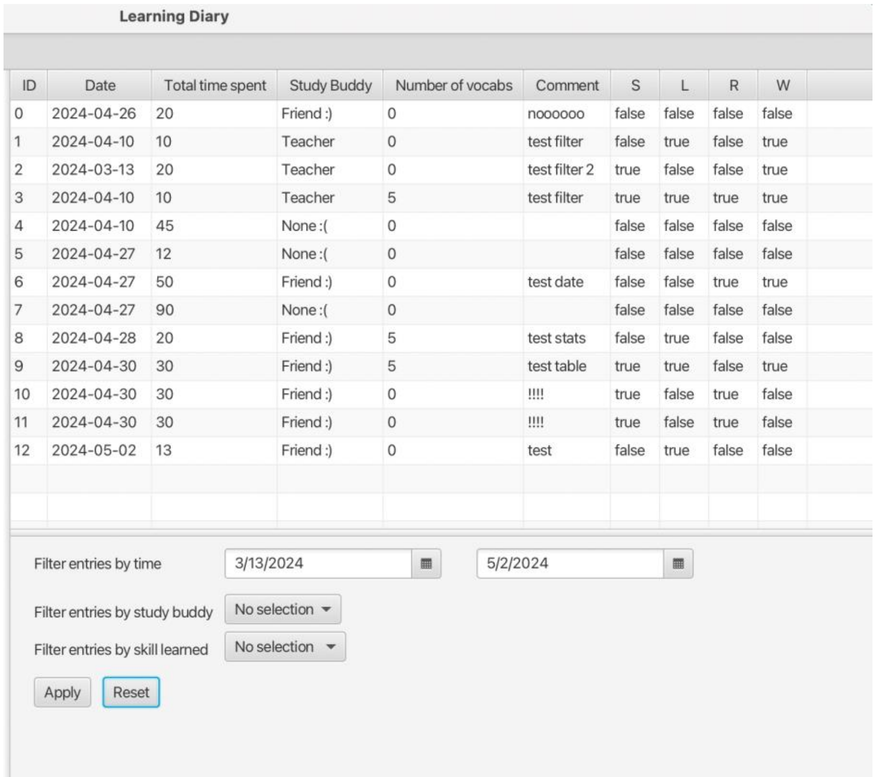
ID	Date	Total time spent	Study Buddy	Number of vocabs	Comment	S	L	R	W
8	2024-04-28	20	Friend :)	5	test stats	false	true	false	false
9	2024-04-30	30	Friend :)	5	test table	true	true	false	true

Filter entries by time 4/25/2024 5/1/2024

Filter entries by study buddy Friend :)

Filter entries by skill learned Listening

After this, to reset the table, we just click on “Reset”. The table will return to its original state.



Learning Diary

ID	Date	Total time spent	Study Buddy	Number of vocabs	Comment	S	L	R	W
0	2024-04-26	20	Friend :)	0	noooooo	false	false	false	false
1	2024-04-10	10	Teacher	0	test filter	false	true	false	true
2	2024-03-13	20	Teacher	0	test filter 2	true	false	false	true
3	2024-04-10	10	Teacher	5	test filter	true	true	true	true
4	2024-04-10	45	None :(	0		false	false	false	false
5	2024-04-27	12	None :(	0		false	false	false	false
6	2024-04-27	50	Friend :)	0	test date	false	false	true	true
7	2024-04-27	90	None :(	0		false	false	false	false
8	2024-04-28	20	Friend :)	5	test stats	false	true	false	false
9	2024-04-30	30	Friend :)	5	test table	true	true	false	true
10	2024-04-30	30	Friend :)	0	!!!!	true	false	true	false
11	2024-04-30	30	Friend :)	0	!!!!	true	false	true	false
12	2024-05-02	13	Friend :)	0	test	false	true	false	false

Filter entries by time 3/13/2024 5/2/2024

Filter entries by study buddy No selection

Filter entries by skill learned No selection