## horizontal line

Library Manager Software

https://github.com/trevalkov/libsearch

**─**

Rodrigo Ferreira

BSc Computer Science

# Documentation

**A library manager software with a TUI interface that allows for some business logic to be executed by the user along with reading and writing data to the disk.**

For the **design** we have a main application done in a mixed procedural programming and OOP that serves the frontend interface interacting with the user and not validating input inside the menu file, menu.hpp. This data is passed to the domain functions that can be found inside library.hpp and perform the business logic and keeping track of data in memory, saved to disk when the program quits.

A dynamic vector template (Cherno, youtube.com, 2020) was the **data structure** chosen as we can quickly access product data with good performance when we pair the key product id with a structure that holds all the data we need while getting rid of deleted books.

For **program flow** we execute one of the following business actions - by title, add a new book, reduce book amount and delete it when amount is zeroed, increment amount, and search book returning data to user. These functions have the performance of O(n) and an improved space complexity compared to a normal vector as we shrink the vector when delete a book in memory.

**Testing** domain functions was done using Catch2 (stable v2) framework where we tested for basic business logic with arbitrary data, loaded and backed up data between memory and disk.

# Conclusion

I wrote this code months ago but was late for the submission and the code was too much of a mess to fix, there were no tests and the design was bad as everything was mixed up, I tried my best to fix it. Input was not validated so that's something I would do next time but it didn’t seem productive this time. Also could have ordered data and implemented a sorting algorithm and a binary search algorithm reducing time complexity to O(logn) for searching.

In conclusion, next time I will be writing the tests for the domain beforehand and separating the frontend, API and backend integration completely from the beginning.

# README

> This project was developed as an undergraduate Computer Science project.

> It is licensed under GPL.

## Build

$ make

> Compilation with C++11 not a requirement

## Run

$ make run

## Test

$ make tests

> Must be compiled for C++11 (Catch2 v2 testing framework)

> Unix only dependency management script

Extra stuff for university bureaucracy including git screenshots and maybe other useless rubbish that i might opt out including uml design xDDD lol