Reactor Coding Challenge 01

This challenge is about fixing and improving the given C++ project.

Specification

The application should model the following workflow:

1. It reads a list of words from STDIN, one per line, terminated by the word 'end'.
2. It prints each individual word in alphabetical order, one per line with their number of occurrences.
3. It repeatedly asks the user to enter a word and looks it up in the list of user entries. If it finds the word, it prints its number of occurrences. Otherwise it prints an error message.
4. It terminates when it encounters EOF.

Guidelines

The project has a number of bugs and design deficiencies. Fix as many problems as you can find to conform to the specification. Also make improvements to make the project more maintainable and easier to work with. Ideally you should consider your submission as ready to be delivered to end users in production.

Please preserve and improve the multi-threaded nature of the application and the overall functionality. You may refactor the code into classes and different files if you wish. For any decision you may be making during the development, please justify and explain your reasoning in a changelog.

Some (but not all) the things you should think about when improving upon the project:

- Use a recent C++ version - at least C++17 or later.
- Please ensure that your submission can be built and run on a recent version of Ubuntu (22.04, 24.04, ...).
- Pay attention to producing readable code and an organized directory structure so that your audience can easily understand your project.
- Create automated build scripts so that the project can be built on Ubuntu with minimal human interaction.
- Make sure the code formatting and the style (variable names,

capitalization, ...) are consistent, ideally by using automated tools.
- Add at least unit tests, preferably also integration tests, or fuzz tests.
- Consider setting up other automated tools, such as memory corruption and data race detectors, etc.
- Remove artifacts such as editor configuration files before submitting.

Please do not upload your submission or the original code anywhere public.

What to Include in Your Submission

- A CHANGELOG file outlining your changes to the code and project. Please explain why you made each change.
- A README file containing clear instructions on how to build, execute, and test your program.
- Please zip or tar your submission in a directory named `yourfirst.lastname` and return it via email.
- In your email response please let us know roughly how long you spent on this exercise.