C++ Foundation, Assignment 1

Task

Write a C++ program that when run does the following:

1. Asks a user for a name of a text file, e.g. *filename.txt*.
2. Reads the text file with the name provided by the user (or prints an error message if the file doesn't exist).
3. Counts the letter frequency in this file, ignoring punctuation and case.
4. Prints the letter frequencies from the highest to the lowest on the screen.
5. Saves the letter frequency in text format to file *filename.out*.
6. Loops back to step 1.

For instance, an interaction with this program might look like this:

```
> ./count_letters

Please enter a file name. Empty file name quits the program.
File name: bob.txt
The file "bob.txt" doesn't exist.

Please enter a file name. Empty file name quits the program.
File name: alice.txt
Opening "alice.txt"

e: 1116
a: 849
r: 758
i: 754
o: 716
t: 695
(here more lines are printed)
x: 15
q: 9
z: 7

Results saved to file "alice.out"

Please enter a file name. Empty file name quits the program.
File name:
Bye!
```

Requirements

1. Separate your program into multiple files - your main program file (*main.cpp*) should ideally contain only the `main` function.
2. Use header and implementation files.
3. Check input and output operations for errors by inspecting the states of streams.
4. Use C++ functions, not C.
5. Use type inference (`auto`) whenever possible.
6. Use one naming convention.
7. No global state variables.

Tips and hints

1. Opening a file and checking if this operation succeeded is two lines of code:

   ```
   std::ifstream input{file_name};
   if (!input){
        std::cout << "Something went wrong...\n";
   }
   else{
        // all ok - proceed
   }
   ```

   or better yet using if with initialiser:

   ```
   if (std::ifstream input{file_name); !input){
        std::cout << "Something went wrong...\n";
   }
   else{
        // all ok - proceed
   }
   ```

2. Reading a text file line by line is similarly easy:

   ```
   std::string line{};
   while(std::getline(input, line){...}
   ```

   or using a `for` loop:

   ```
   for(std::string line{}; std::getline(input, line);){...}
   ```

3. You can use `std::isalpha` `std::islower`, `std::isupper`, `std::tolower` and `std::toupper` from the `cctype` header for processing characters.