NEA

[Introduction 1](#_Toc208391581)

[Aim 1](#_Toc208391582)

[Research 1](#_Toc208391583)

[Outlook 1](#_Toc208391584)

[Partials solutions 2](#_Toc208391585)

Introduction:

GoCode editor: A modal based coding editor that is written 100% in Golang. This project is an opportunity for me to learn not only how a coding editor works, but also why the best ones are the way they are, along with me expanding my understanding of coding languages by choosing a language that I am not close to mastering and allowing me to learn it through this project in able to expand my portfolio of coding languages for in the future. This editor aims to help coders who want a modal editor such as neovim, but rather than it being in the terminal, it is its own stand-alone app that can provide necessities such as syntax highlighting and some debugging support natively.

## Aim

The overall aim of the project is to give me deeper understanding in how coding editors work and what is required to have a functioning editor. This project will also allow me to learn a new language to allow me to expand my portfolio of coding languages, which will benefit me in the future.

# Research

## Outlook

I believe this project will challenge me to work and understand how coding editors work and what components are needed within an editors to get them to work. This will allow me to use a computational approach to decompose the project down into approachable sections that I can work on and recombine to make a full working prototype. The project will require me to do deeper research into how individual component of a coding editor works and what each piece will need to be functional and how they will work together.

## Pre-requisites

For this project to be successful, I am going to need:

* A window library
* Shell for running/debugging
* User installed languages
* Access to filesystem to open/close/save files

## Features to include

The features I would like to include in my project include:

* Modes – Command/Edit
* GUI for ease of use
* Inbuilt terminal to run/debug files using language servers on users’ machine
* Syntax highlighting to allow users to view code
* Debugging capabilities
* Commands to have functionality
* Finding within files/finding within a directory (fuzzy finder)

## Partials solutions

There are a few partial solution that are similar to what I am trying to make, these include:

* Micro: a text based coding editor

Zachary Yedidia. (Aug 27, 2016). *Micro*. [Online]. github.com. Last Updated: Aug 27, 2024. Available at: <https://github.com/zyedidia/micro> [Accessed 25 August 2025].

* Tide: a TUI terminal based

bethropolis. (April 9th 2025). *Tide*. [Online]. github.com. Last Updated: April 13th 2025. Available at: [https://github.com/bethropolis/tide](https://github.com/bethropolis/tide%20) [Accessed 25 August 2025].

Both of these solutions use the Golang programming language, which is what I am also choosing to use for this project as it will help me to expand my knowledge of coding and allow me to have multiple options for later in life if I choose to code, allowing me to expand my breadth in coding to have multiple options for completing a project. Both solutions are text based, meaning that they only run in the terminal, whereas I would like to have this code editor as a GUI, so the interface is more useable to more people.

## Libraries

To make the program a GUI I have decided to look for multiple window libraries within the Golang ecosystem.