**Darren Wong - 25147064**

**Trieu Huy Vu - 26232952**

**Design Documentation**

**FIT3140 Assignment 3 – Spiking and Analysis of Alternatives**

# SCREEN MOCKUPS

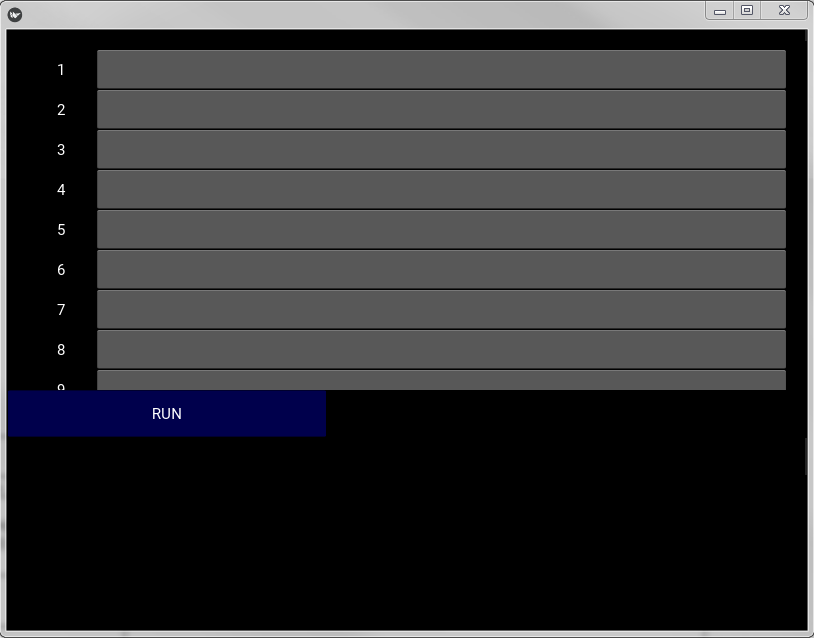


Image 1.1 GUI – Main page of the application

As we can see here in Image 1.1, the Main Page of this application consists of a RUN button, responsible for executing the function, a scrollable interactive workspace, which allows users to code with and an Output Console, currently left blank under the RUN button. The Output Console is also scrollable to allow users to see results that span bigger than the given intended space.

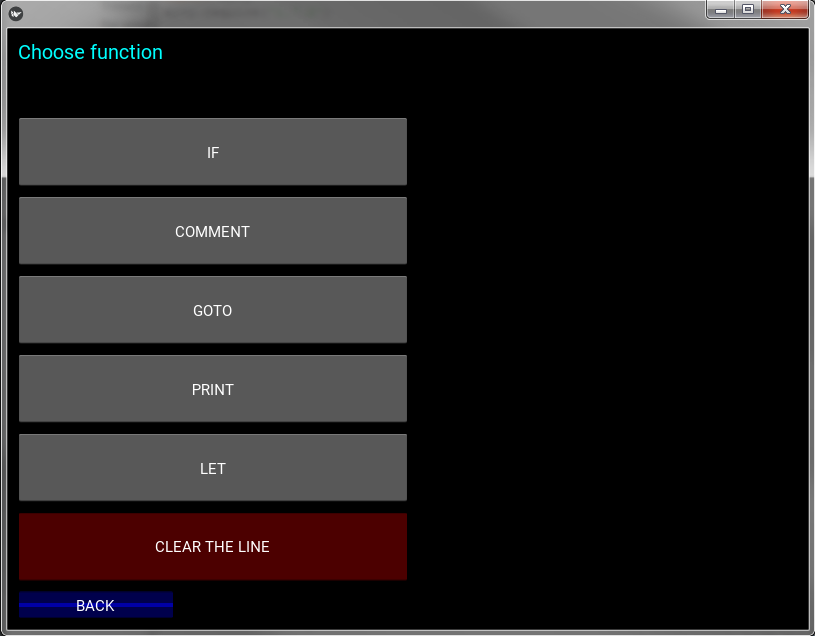


Image 1.2 GUI – List of functions available

Image 1.2 represents what happens when the user clicks on an empty line. It asks the user which function they want to include in their current program.

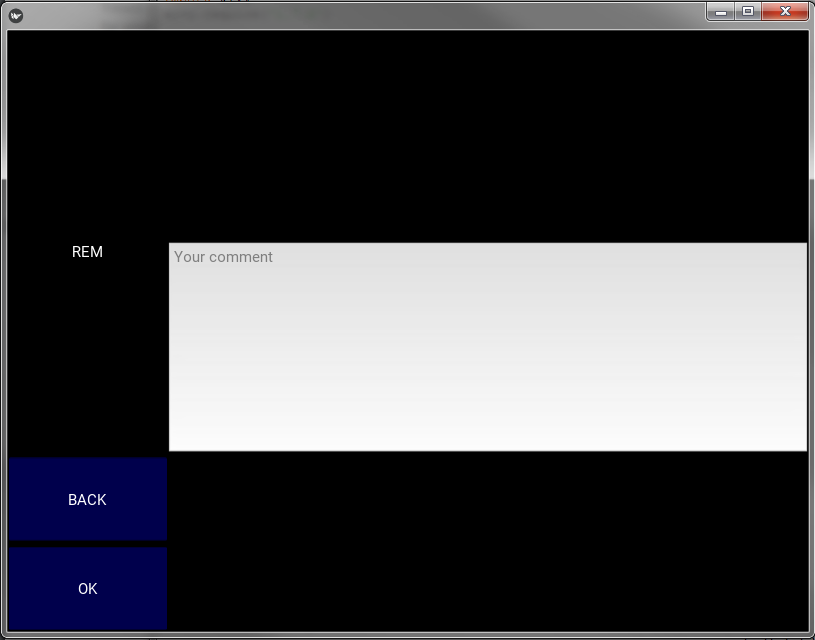


Image 1.3 GUI – User input screen

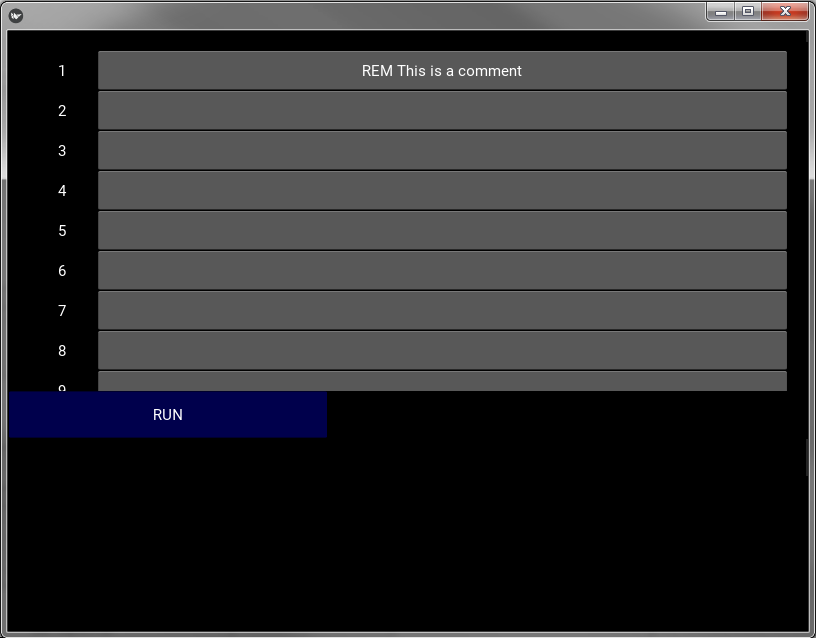
Image 1.3 shows the interface of when a function is chosen and a user input is required. Each function will have its own separate screen to accommodate for their differences.

Image 1.4 GUI – Update Main Page of application with user’s code

Image 1.4 shows when a line of code is updated and incorporated into the program.

# UML CLASS DIAGRAM

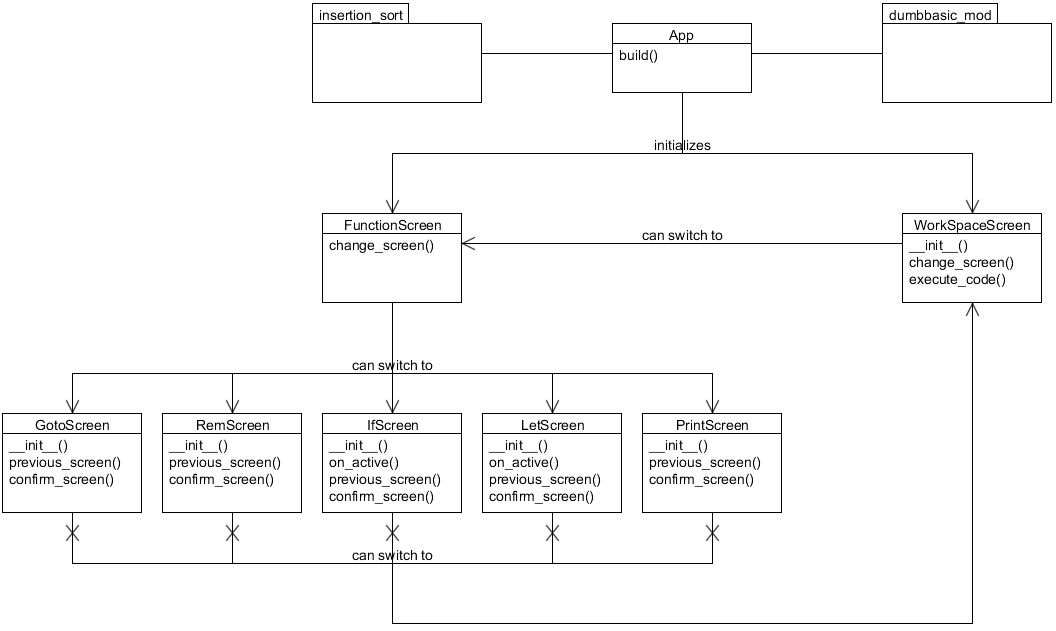


Image 1.5 Conceptual Glass Diagram of the system

In this diagram, the App class is responsible for initializing and starting ScratchBasic. This allows the flexibility to add more screens and extend functionality and allowing classes to be of high cohesion. The two packages that are imported are an insertion\_sort, sorting algorithm designed to sort a list and dumbbasic\_mod, a module that is responsible of the smooth running of the code entered by the user.

The WorkSpaceScreen class is responsible for allowing a user to enter their desired coding. When a code is being entered, they are taken to the FunctionScreen class.

The FunctionScreen class is responsible for holding all the available functions of ScratchBasic and allows users to select a specific function, bringing them to the corresponding function screen i.e. GotoScreen, RemScreen, IfScreen, LetScreen and PrintScreen.

The GotoScreen, RemScreen, IfScreen, LetScreen and PrintScreen classes is responsible for getting the user’s input which Is stored into a dictionary.

However, in this diagram, GotoScreen, RemScreen, IfScreen, LetScreen and PrintScreen can all navigate to the WorkSpaceScreen but not the other way around. This was implemented with the assumption that if the user does select a wrong function, the back button would take them to the FunctionScreen class and not the WorkSpaceScreen class, to allow the user to reselect another function.