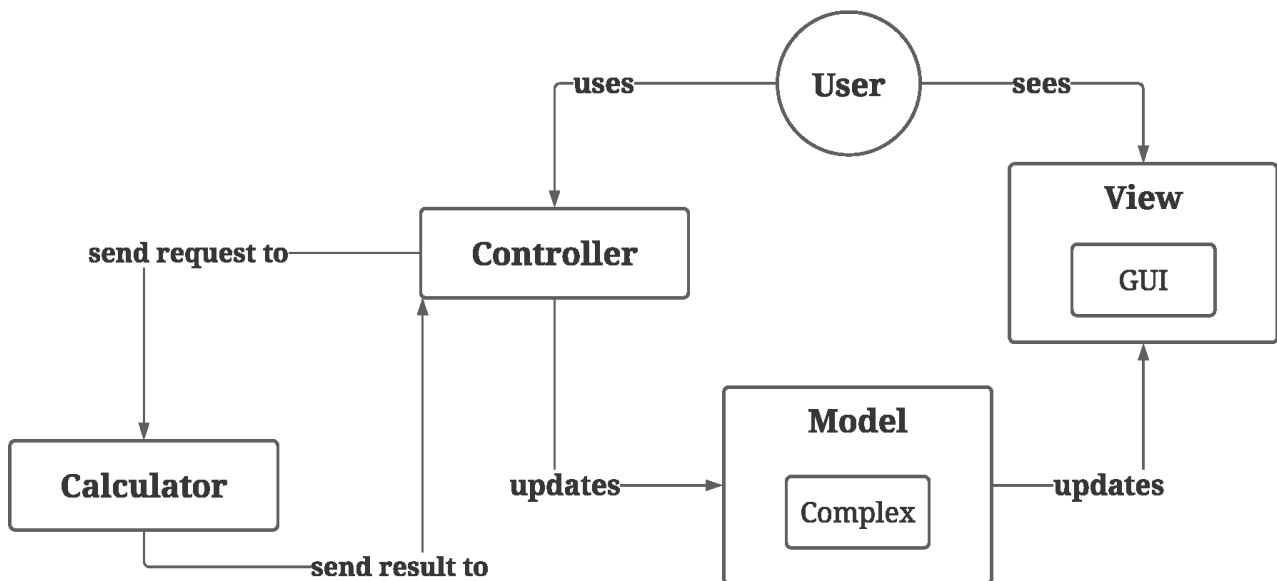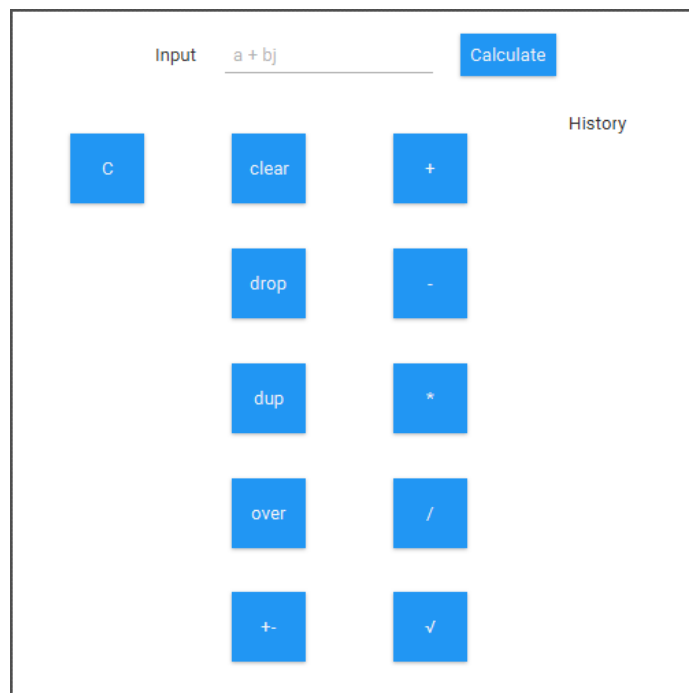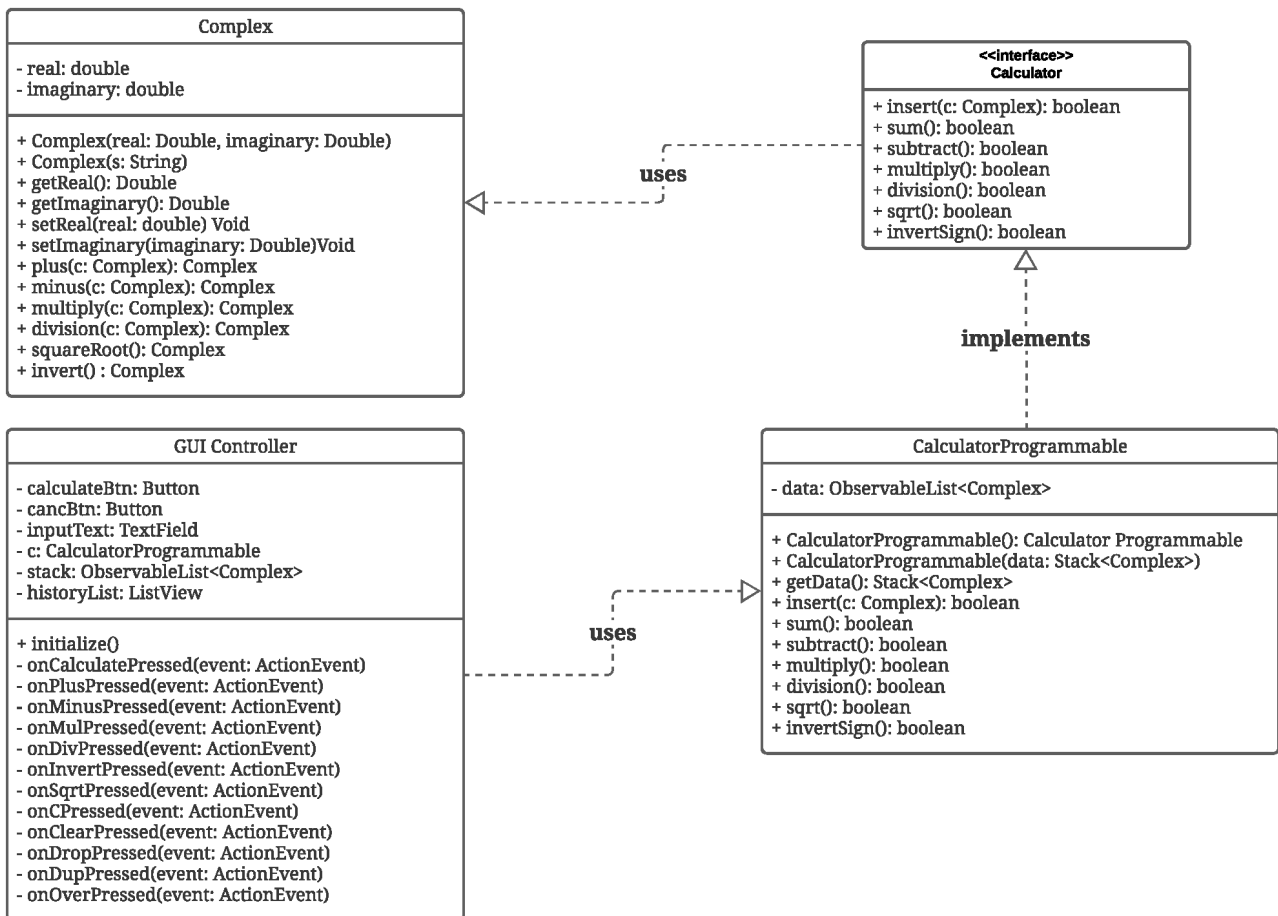# Design

Architecture Design

The software's architecture design starts from MVC pattern with an addition.

A **User** using the software interacts with the **View** (see figure below) module which displays the data of interest through the **GUI**. The **Model** component, on the other hand, provides methods for managing data which are described by the **Complex** sub-module. Each request made by the user is taken over by the **Controller** and forwarded to the **Calculator** module; the latter processes and returns a response to the caller, in order to make it visible to the user.

Detailed Design

Complex

- real: double
- imaginary: double

+ Complex(real: Double, imaginary: Double)
+ Complex(s: String)
+ getReal(): Double
+ getImaginary(): Double
+ setReal(real: double) Void
+ setImaginary(imaginary: Double)Void
+ plus(c: Complex): Complex
+ minus(c: Complex): Complex
+ multiply(c: Complex): Complex
+ division(c: Complex): Complex
+ squareRoot(): Complex
+ invert() : Complex

<<interface>> Calculator

+ insert(c: Complex): boolean
+ sum(): boolean
+ subtract(): boolean
+ multiply(): boolean
+ division(): boolean
+ sqrt(): boolean
+ invertSign(): boolean

uses

implements

GUI Controller

- calculateBtn: Button
- cancBtn: Button
- inputText: TextField
- c: CalculatorProgrammable
- stack: ObservableList<Complex>
- historyList: ListView

+ initialize()
- onCalculatePressed(event: ActionEvent)
- onPlusPressed(event: ActionEvent)
- onMinusPressed(event: ActionEvent)
- onMulPressed(event: ActionEvent)
- onDivPressed(event: ActionEvent)
- onInvertPressed(event: ActionEvent)
- onSqrtPressed(event: ActionEvent)
- onCPressed(event: ActionEvent)
- onClearPressed(event: ActionEvent)
- onDropPressed(event: ActionEvent)
- onDupPressed(event: ActionEvent)
- onOverPressed(event: ActionEvent)

CalculatorProgrammable

- data: ObservableList<Complex>

+ CalculatorProgrammable(): Calculator Programmable
+ CalculatorProgrammable(data: Stack<Complex>)
+ getData(): Stack<Complex>
+ insert(c: Complex): boolean
+ sum(): boolean
+ subtract(): boolean
+ multiply(): boolean
+ division(): boolean
+ sqrt(): boolean
+ invertSign(): boolean

uses

The Complex Class represents the set of complex numbers; it allows you to perform the operations of addition, subtraction, multiplication, division and square root and inversion of the sign.

The Calculator consists of:

- an interface that contains the signature of the methods of a calculator;
- a class that implements the interface, the latter will contain an attribute, of type Stack, that will manage instance objects of the Complex class.

Finally, the Controller takes care of acquiring and analyzing the user command inserted in a TextField and requested through a Button, it has an observable attribute to show the contents of the Stack on the GUI through a ListView.