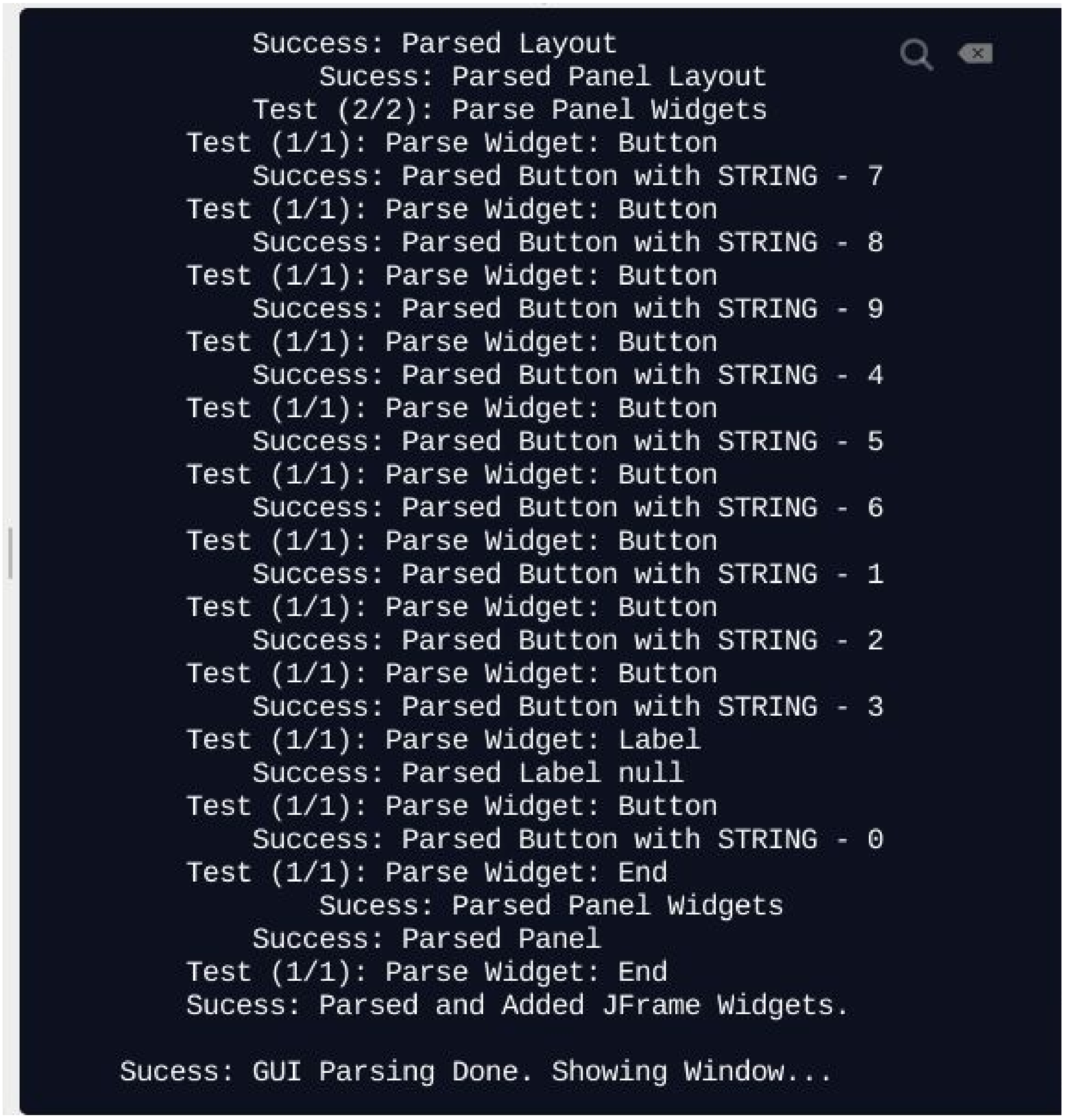
Kyra Samuel

Project1

CMSC330

Guide:

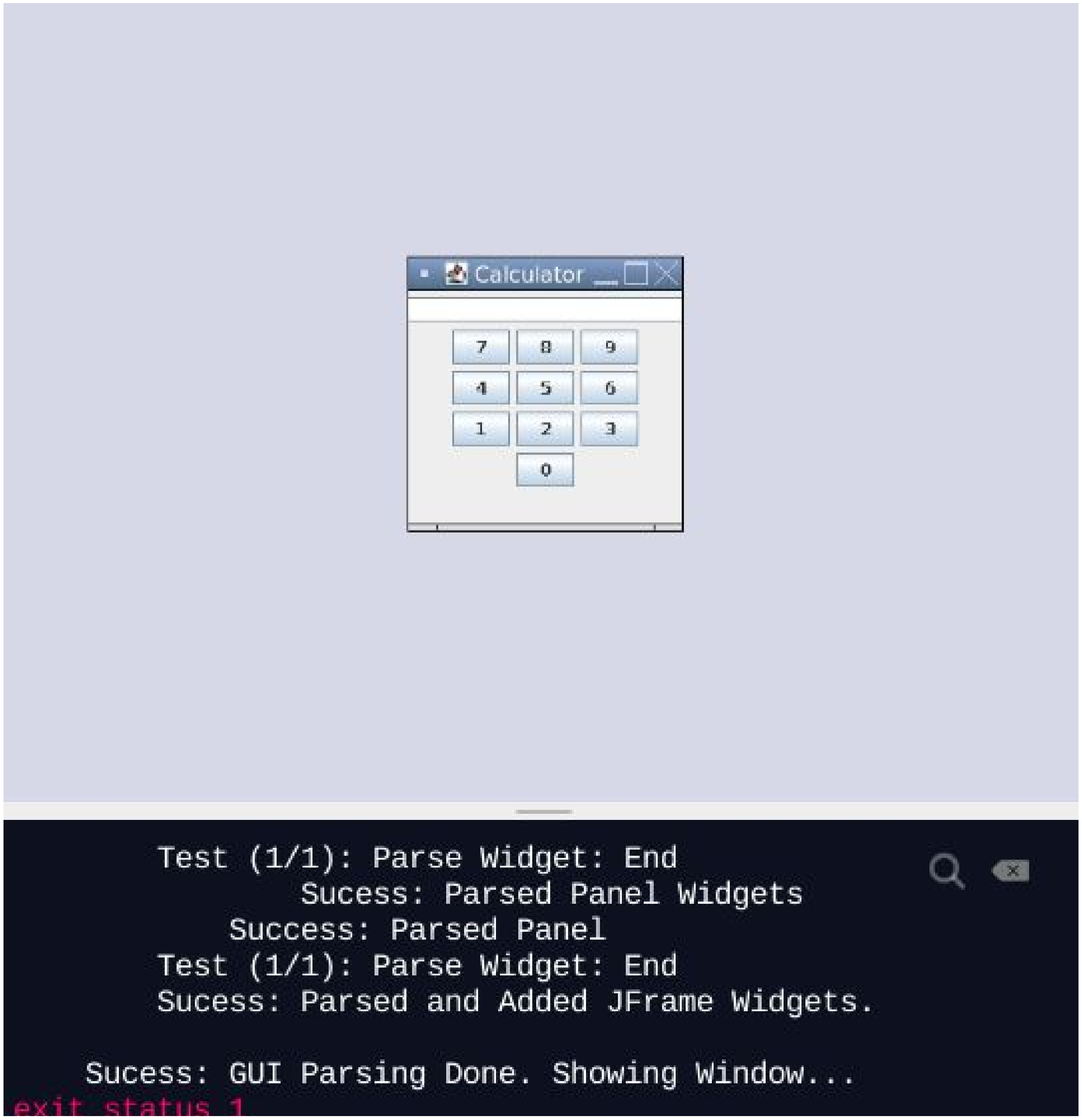
1. Following the Grammar Rules in Project1.pdf write a .txt file to build a GUI
2. Save the txt file to your local directory or Project1 folder
3. Run Main.java
4. You will be prompted to select a text file only.
5. Analyze the console.log for success or failure cases.



```
            Success: Parsed Layout
                Sucess: Parsed Panel Layout
            Test (2/2): Parse Panel Widgets
        Test (1/1): Parse Widget: Button
            Success: Parsed Button with STRING - 7
        Test (1/1): Parse Widget: Button
            Success: Parsed Button with STRING - 8
        Test (1/1): Parse Widget: Button
            Success: Parsed Button with STRING - 9
        Test (1/1): Parse Widget: Button
            Success: Parsed Button with STRING - 4
        Test (1/1): Parse Widget: Button
            Success: Parsed Button with STRING - 5
        Test (1/1): Parse Widget: Button
            Success: Parsed Button with STRING - 6
        Test (1/1): Parse Widget: Button
            Success: Parsed Button with STRING - 1
        Test (1/1): Parse Widget: Button
            Success: Parsed Button with STRING - 2
        Test (1/1): Parse Widget: Button
            Success: Parsed Button with STRING - 3
        Test (1/1): Parse Widget: Label
            Success: Parsed Label null
        Test (1/1): Parse Widget: Button
            Success: Parsed Button with STRING - 0
        Test (1/1): Parse Widget: End
                Sucess: Parsed Panel Widgets
            Success: Parsed Panel
        Test (1/1): Parse Widget: End
        Sucess: Parsed and Added JFrame Widgets.

    Sucess: GUI Parsing Done. Showing Window...
```

Tests:

Default shown in Projectt1.pdf



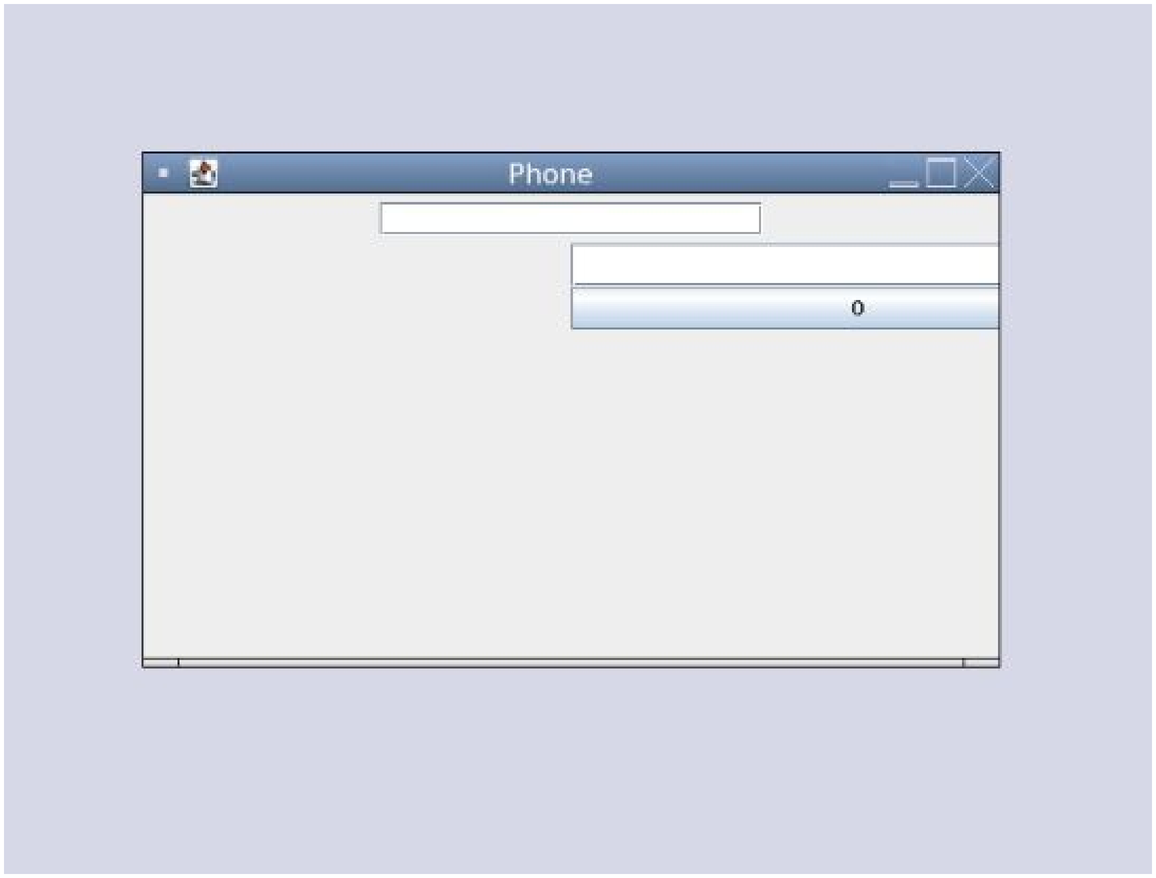
```
      Test (1/1): Parse Widget: End
              Sucess: Parsed Panel Widgets
          Success: Parsed Panel
      Test (1/1): Parse Widget: End
      Sucess: Parsed and Added JFrame Widgets.

  Sucess: GUI Parsing Done. Showing Window...
exit status 1
```

test.txt

Window "Calculator" (200, 200) Layout Flow:
Textfield 20;
Panel Layout Grid(4, 3, 5, 5):
Button "7";
Button "8";
Button "9";
Button "4";

```
 Button "5";
 Button "6";
 Button "1";
 Button "2";
 Button "3";
 Label "";
 Button "0";
 End;
End.
```

Test2: Different Layout(Flow), Label with String, Nested Panel

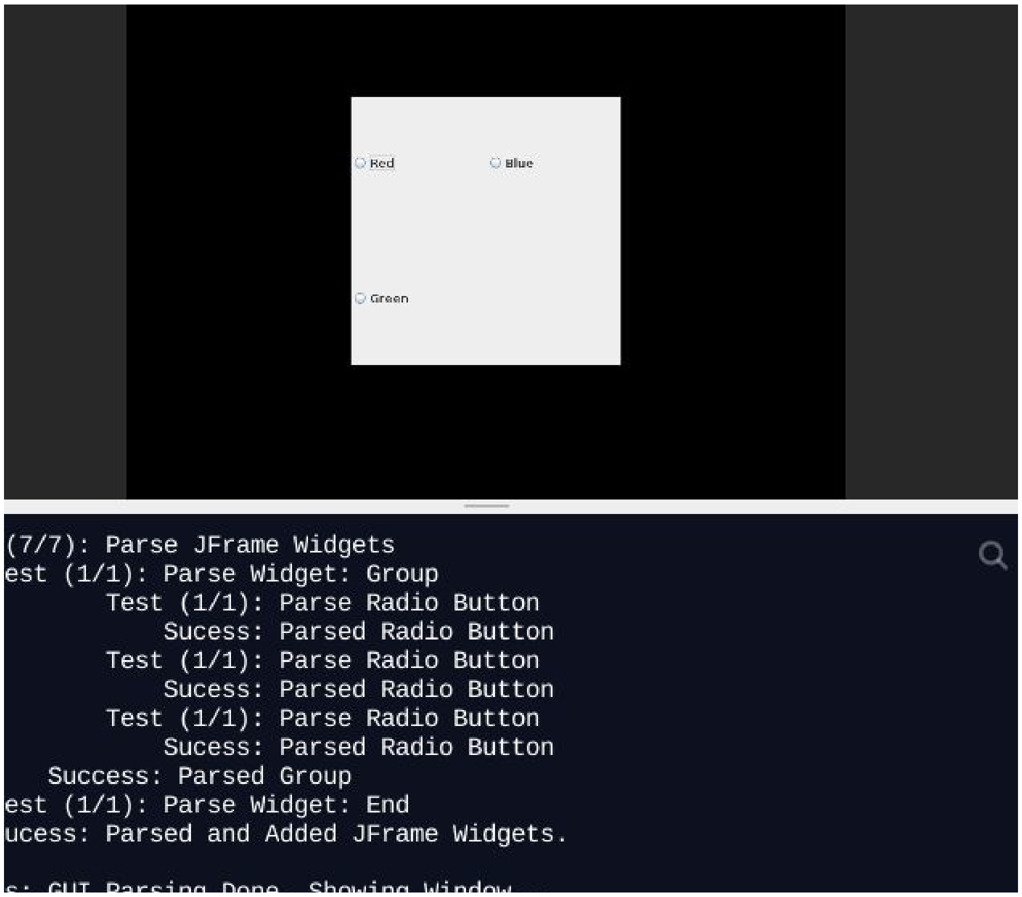


test2.txt

```
Window "Calculator" (200, 200) Layout Flow:
 Textfield 20;
 Panel Layout Grid(4, 3, 5, 5):
 Button "7";
 Button "8";
 Button "9";
 Button "4";
```

```
    Button "5";
    Button "6";
    Button "1";
    Button "2";
    Button "3";
    Label "";
    Button "0";
    End;
   End.
```

Test3: Radio Buttons, Grid Layout with Row/Column Spacing



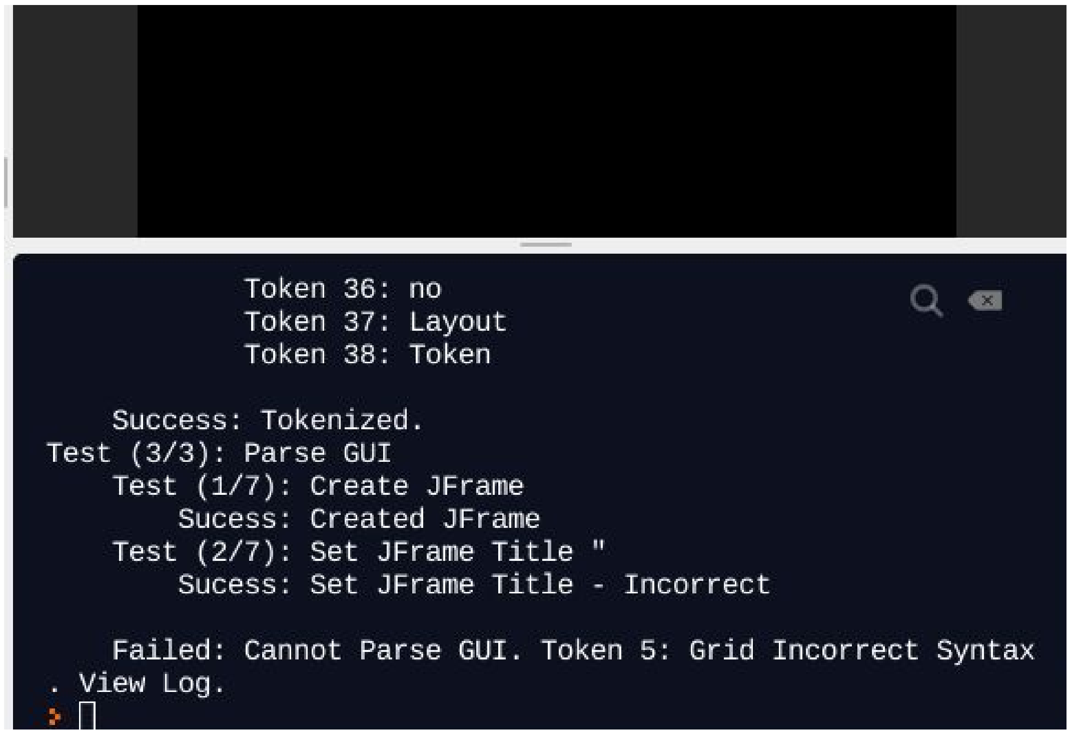
```
(7/7): Parse JFrame Widgets
est (1/1): Parse Widget: Group
       Test (1/1): Parse Radio Button
           Sucess: Parsed Radio Button
       Test (1/1): Parse Radio Button
           Sucess: Parsed Radio Button
       Test (1/1): Parse Radio Button
           Sucess: Parsed Radio Button
   Success: Parsed Group
est (1/1): Parse Widget: End
ucess: Parsed and Added JFrame Widgets.

s: GUI Parsing Done. Showing Window.
```

test3.txt

Window "Calculator" (200, 200) Layout Flow:
 Textfield 20;
 Panel Layout Grid(4, 3, 5, 5):
 Button "7";

```
  Button "8";
  Button "9";
  Button "4";
  Button "5";
  Button "6";
  Button "1";
  Button "2";
  Button "3";
  Label "";
  Button "0";
  End;
End.
```

Incorrect: There is no Layout defined, missing token “Layout”. The console.log shows that “Grid” is the incorrect token.



```
            Token 36: no
            Token 37: Layout
            Token 38: Token

    Success: Tokenized.
Test (3/3): Parse GUI
    Test (1/7): Create JFrame
        Sucess: Created JFrame
    Test (2/7): Set JFrame Title "
        Sucess: Set JFrame Title - Incorrect

    Failed: Cannot Parse GUI. Token 5: Grid Incorrect Syntax
. View Log.
>
```

incorrect.txt

Window "Calculator" (200, 200) Layout Flow:
 Textfield 20;
 Panel Layout Grid(4, 3, 5, 5):
 Button "7";

```
    Button "8";
    Button "9";
    Button "4";
    Button "5";
    Button "6";
    Button "1";
    Button "2";
    Button "3";
    Label "";
    Button "0";
    End;
   End.
```

Reflection:

I learned a lot about how programming languages are written, just by writing out this simple GUI parser (it doesn't cover everything, I would've enjoyed Project2 stemming off of this one to include colors, borders, and paint components: Oval, Circle, Square, Rectangle, etc. I think that this is neat and it allows me to create the product that I'm currently trying to build, the input would be HTML/CSS/Javascript files that would need to parse into files that would match the backend. I would need to update routes, text, animation, etc. For example, a basic HTML5 file parsed into an HTML file that inherits Jinja templates and WTFforms. I used a ton of different methods like overloading, LinkedHashMaps O(1) (faster than ArrayList O(n)). Originally, I wanted to use named groups in Regex, but I had a hard time implementing it in the way the parser video shows. Overall, I learned a lot and I'm happy that I was able to complete this project!