AOA for User Interface/Platform

Factors (ordered by Priority)

- Ease of Development: How easy it is to accomplish the same goal
- User Experience: How easy it is to operate the application
- Cross-Platformness: How many platforms are supported/easy to be ported

Choices

- TkInter(native)
- PyQt (Industrial Grade UI Library)
- Web Frontend (via RESTful API,e.g. Bottle, Flask)

Body

Ease of Development

UI Designing:

Example code for each library in order to accomplish the same layout:

TkInter

```
def demo():
  lbl.set("demo!")
```

```
root = Tk()
root.title("Title")
root.geometry("200x50")
mainframe = ttk.Frame(root)
mainframe.pack()
lbl = StringVar()
l = ttk.Label(mainframe, textvariable=lbl)
l.pack()
b = ttk.Button(mainframe, text="Demo Button", command=demo)
b.pack()

root.mainloop()
```

PyQt

```
import sys
from PyQt5.QtWidgets import (QWidget, QToolTip,
    QPushButton, QApplication, QLabel)
from PyQt5.QtGui import QFont

def demo():
    lbl.setText("demo")
    lbl.resize(lbl.sizeHint())

app = QApplication(sys.argv)

w = QWidget()
w.resize(200, 50)
w.setWindowTitle('Title')
```

```
lbl = QLabel('', w)
lbl.move(60,5)
btn = QPushButton('Demo Button', w)
btn.resize(btn.sizeHint())
btn.move(60,25)
btn.clicked.connect(demo)
w.show()

sys.exit(app.exec_())
```

Web Frontend

```
<!DOCTYPE html>
<head>
  <title>Title</title>
</head>
<body>
<style>
     #label {
       height: 18px;
     }
     .container {
         width: 200px;
         height: 50px;
     }
  </style>
  <div class="container">
  <div id='label'></div><br />
  <button onclick="demo()">Demo Button</button>
```

```
  </div>
  <script>
    function demo() {
      document.getElementById('label').innerText = "demo";
    }
  </script>
</body>
```

Summary

The TkInter and PyQt approach is pretty similiar in nature, both requires UI designing through Python code. While this approach is native to the rest of the codebase and greatly supports dynamic generation of UI component, the ease of styling the components is pathetic compared to styling using CSS. The above demostration shows that the desired UI can be accomplished by all three approaches. On top of that, web frontend can be debugged easily and developed with aid using developer-targeted browser like Firefox, au contraire the IDEs for TkInter and PyQt are less easy to configure and use. Considering the our team's familiarity with the languages, Web Frontend is the best choice as the alternatives does not provide obvious strength and perks for using them.

Function:

TkInter, PyQt

```
'''
this is a minified example, showing the most basic outline of the application flow,
where inspektor is our connector to make use of gitinspector's code,
uicore is our implementation of UIs functions
'''
from GitRekt import inspektor, uicore
import urllib.request


def handler(gdocid):
```

```
  # query google's API

  data = urllib.request.urlopen("api.google.com/example/api/placeholder?id=%s" %
    gdocid).read()

  # process returned data using our own function (written natively in Python)

  output = inspektor.processData(data)

  # post data to output handler (similiar to corresponding example above)

  uicore.showOutput(output)
```

Web Frontend

```
from flask import Flask

app = Flask(__name__)


@app.route("/request/<gdocid>")

def inspect(gdocid):

  data = urllib.request.urlopen("api.google.com/example/api/placeholder?id=%s" %
    gdocid).read()

  output = inspektor.processData(data)

  return render_template('result.html', result=output)
```

Summary

Both approaches show similiar way in getting things done, therefore this factor can be neglected while considering the alternatives.

User Experience

TkInter and PyQt utilizes respective OS native UI library, which works fine and fair. However, most of the users nowadays are more used to the web interface (i.e. heavy usage of web applications), the navigation of web frontend would be more straightforward to them.

Cross-Platformness

Both TkInter and PyQt supports all the three major operating systems (Windows, Linux and Mac) and web frontend is cross-platform by its nature. As mobile platform is not targeted (it is inconvenience and unrealistic to use our application on mobile device), these three choices are on par with each other.

Recommendation

As the ease of development is our primary concern, web frontend would be the obvious choice. On top of that, the ease of styling for web frontend would provide a much better and more immersive user experience compared to the other alternatives. The strengths of the other choices do not outweigh their development cost, therefore using a web frontend is highly recommended in order to produce quality software and deliver in time.

AOA for Language

Factors (ordered by priority)

- Suitability for back-end scripting
- Libraries
- Readability/Maintainability
- Conventions

Choices

- Python
- Javascript

Body

Suitability for back-end scripting

Python is a general purpose programming language, and is relatively “stronger typed” than Javascript, in the sense that Python prevents typing errors at runtime. As such, Python can be safely used for back-end programming.

Javascript is mainly a language for front-end web programming. That means it is meant to make web pages interactive, or less static. However, it can also be used for back-end programming, with tools such as node.js and others.

Libraries

Python itself has an extensive standard library. When downloading from the official website, the standard libraries will also be downloaded into one’s machine. These include tools such as encodings, web frameworks and testing among others. Aside from that, the Python community also has plenty of open source libraries that are free to download.

Javascript also has a large list of libraries to choose from that can be found online. The only difference is that a javascript library has to be loaded up by a script tag in a HTML file.

Readability/Maintainability

The following is an implementation of selection sort in Python and Javascript respectively.

Python:

```
def selectionSort(inputList):
    """
    Sorts a given list of numbers in ascending order.
    """
    length = len(inputList)
    for current in range(length):
        minIndex = current
        # Searching for minimum value.
        for comparing in range(current, length):
            if inputList[comparing] < inputList[minIndex]:
                minIndex = comparing
        # Putting the minimum value into its correct place.
        temporary = inputList[current]
        inputList[current] = inputList[minIndex]
        inputList[minIndex] = temporary
    return inputList
```

Javascript:

```
function selectionSort(inputList) {
    /*
    Sorts a given function of numbers in ascending order.
    */
    let length = inputList.length;
    for (let current = 0; current < length; current++) {
        let minIndex = current;
        // Searching for minimum value.
        for (let comparing = current; comparing < length; comparing++) {
            if (inputList[comparing] < inputList[minIndex]) {
                minIndex = comparing;
            }
        }
        // Putting the minimum value into its correct place.
        let temporary = inputList[current];
        inputList[current] = inputList[minIndex];
        inputList[minIndex] = temporary;
    }
    return inputList;
}
```

It can be seen that the Python code has an advantage of being more intuitive to read by comparing the for loops. However, the Javascript code allows for less clutter and hence easier readability with the curly braces. However this can be easily overcome in Python by just adding blank lines.

Conventions

Python has quite a standard when it comes to conventions. On the Python official website, there is the PEP 8 document which contains coding style guide for Python. This document covers a broad range of subjects, including programming recommendations and documentation strings to the trivial details, such as using tabs or spaces and line breaks at binary operators.

Javascript has quite a few style guides to choose from, ranging from Airbnb's style guide and Google's style guide to Javascript Standard and Idiomatic style guide. The existence of these different style guides mainly comes from the philosophy behind each style guide and who is enforcing the style guide. However, for a general pointer, Airbnb is a good place to turn to as it covers almost all aspects of Javascript and is also one of the most popular styles.

Recommendation

Ultimately, the language chosen here will be used to interact with the Google Drive and process its data. As such, back end feasibility carries much weight, which Python has the upper hand by nature. Also, since this is a team-based project, multiple people will work on the same code. As such, Python's easy readability and universally accepted convention also tops Javascript. All the members are also more familiar with the language Python and this shortens the time needed to learn new programming language. As such, Python will be the chosen language to code the features of the project.