FileType Identification

Group Number: 5
Group Members:

- Devipriya Sarkar
- Akshata Nadkarni

Deliverable 1:

The relevant data sources found:

1. https://github.com/github/linguist/blob/master/lib/linguist/languages.yml
   It lists all languages known to the popular version control hosting system GitHub. It can be used to match the given file extensions to that mentioned in this source and extract “category type” and “language name”. [USED IN PROJECT]
2. https://en.wikipedia.org/wiki/List_of_filename_extensions
   Wikipedia page hosting the file name extensions. It can be used to match the given file extensions to that mentioned in this source and extract “short description” and “applications that use the given extension”. [USED IN PROJECT]
3. https://medium.com/web-development-zone/a-complete-list-of-computer-programming-languages-1d8bc5a891f
   Lists all programming paradigms and the programming languages that use them. It can be scraped to extract the ”paradigm” of the language of the given file (after extracting language from extension. [USED IN PROJECT]
4. https://github.com/github/linguist/blob/master/lib/linguist/documentation.yml
   This YAML file can be used to check whether the given file is a “documentation file”.
5. https://github.com/github/linguist/blob/master/lib/linguist/vendor.yml
   This YAML file can be used if the given files are “vendor files” (eg. firmware files etc) and libraries.
6. https://github.com/grosser/language_sniffer
   A Ruby library to detect programming languages by extension or #! In the first line (interpreter information) or content from files and texts.
7. https://www.file-extensions.org/
   This website lists “extensions”, their “type”, “description” and “associated applications”. It can be scraped to get extract the above data of the given file extension.

Deliverable 2:

IdMyFile

A web app to identify, categorize and analyze multiple files based on their file extensions.
Project "Filetype Identification" for BlueOptima by Group 5 - Devipriya Sarkar and Akshata Nadkarni.
The data sources used in the project are Source 1, Source 2, Source 3 listed in deliverable 1.

Setup Instructions

- ## Using PyCharm

  1. Make sure python 2.7 is installed on your system.
  2. Un-zip the directory.
  3. Open PyCharm. Browse to the root project directory (immediate parent of "`IdMyFile.py`") and open it.
  4. Install all requirements in "`requirements.txt`".
  5. Run the project.
  6. Go to "http://127.0.0.1:5000/" or "http://localhost:5000/" to check the web app.
     Note: By default, flask runs on port 5000.
  7. Upload a "plain text file" containing the list of file names and extensions in the format "`<file_name>.<file_extension>`" . A sample test file "`sample_input_file.txt`" is there in the project directory for reference.
  8. The result page shows a table with file information and the invalid lines in the input file are shown in the "Invalid Lines" section.
     Quick Information: The regular expression used for matching the valid file name is "^([\w_\.]+)(\.)([\w])$".*

- ## Using command line

  1. Make sure python 2.7 is installed on your system.
  2. Un-zip the directory.
  3. Install all the packages in "`requirements.txt`". To install using pip, run the following command from the project directory:

     ```
     $ pip install -r requirements.txt
     ```

     Check if you have pip installed by using "`pip -V`" on your command line.
     If you do not have pip installed, install pip by securely downloading get-pip.py.
     Then run the command

     ```
     $ python get-pip.py
     ```

4. To run the application from the project directory,
   On Linux,

```
$ export FLASK_APP=IdMyFile.py
$ flask run
        * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

   or

```
$ export FLASK_APP=IdMyFile.py
$ python -m flask run
        * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

   On Windows cmd (not PowerShell),

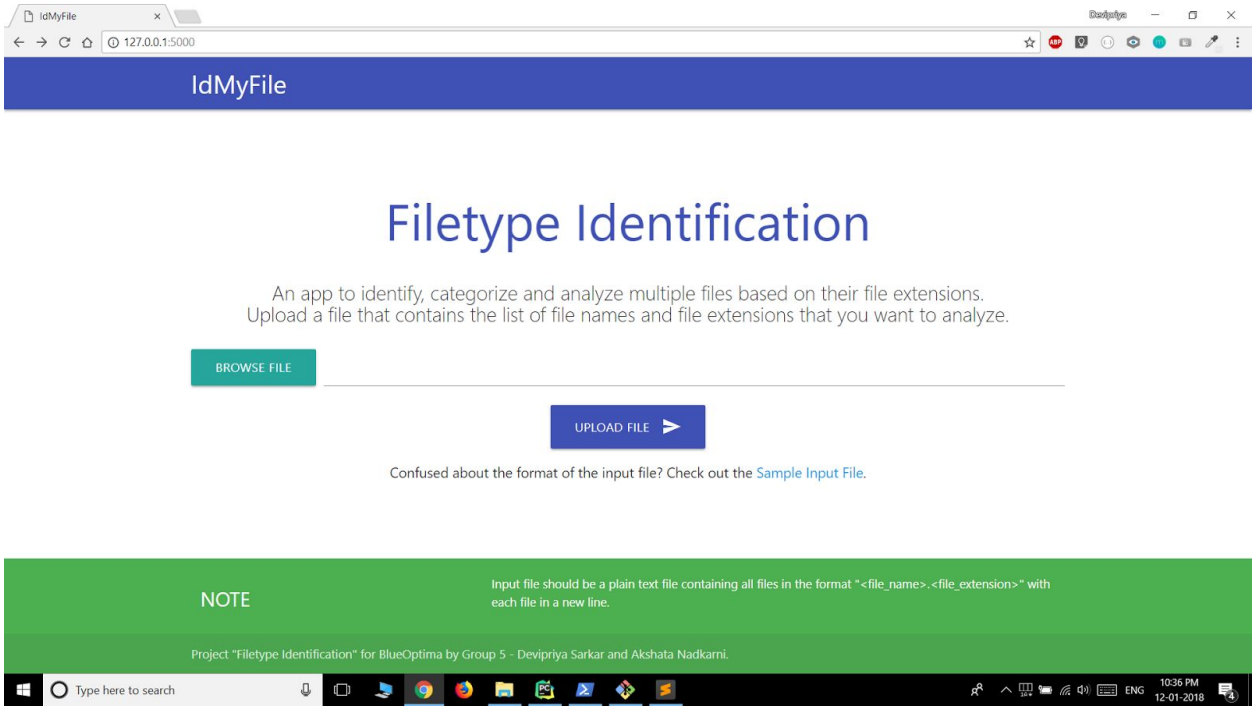
```
$ set FLASK_APP=IdMyFile.py
$ flask run
        * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

5. Go to "http://127.0.0.1:5000/" or "http://localhost:5000/" to check the web app.
   Note: By default, flask runs on port 5000.
6. Upload a "plain text file" containing the list of file names and extensions in the format "`<file_name>.<file_extension>`" . A sample test file "`sample_input_file.txt`" is there in the project directory for reference.
7. The result page shows a table with file information and the invalid lines in the input file are shown in the "Invalid Lines" section.
   Quick Information: The regular expression used for matching the valid file name is "^([\w_\.]+)(\.)([\w])$".*

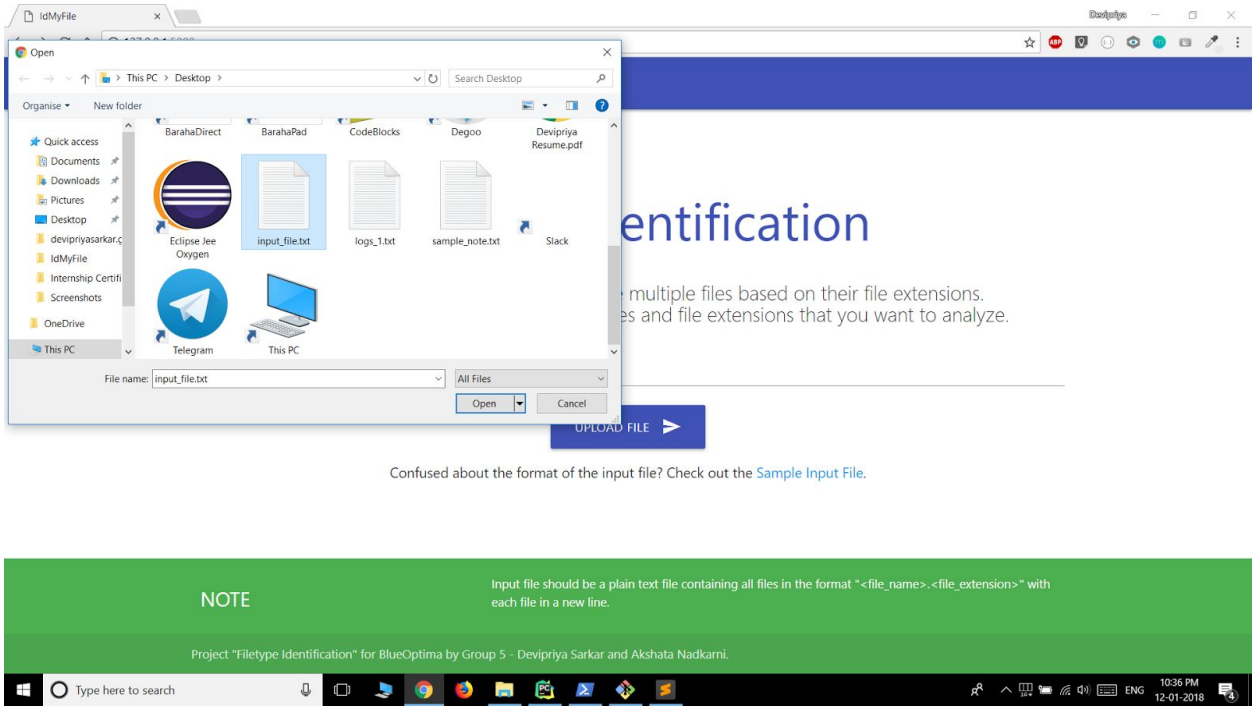
For any setup troubleshooting, refer Flask documentation.

Screenshots

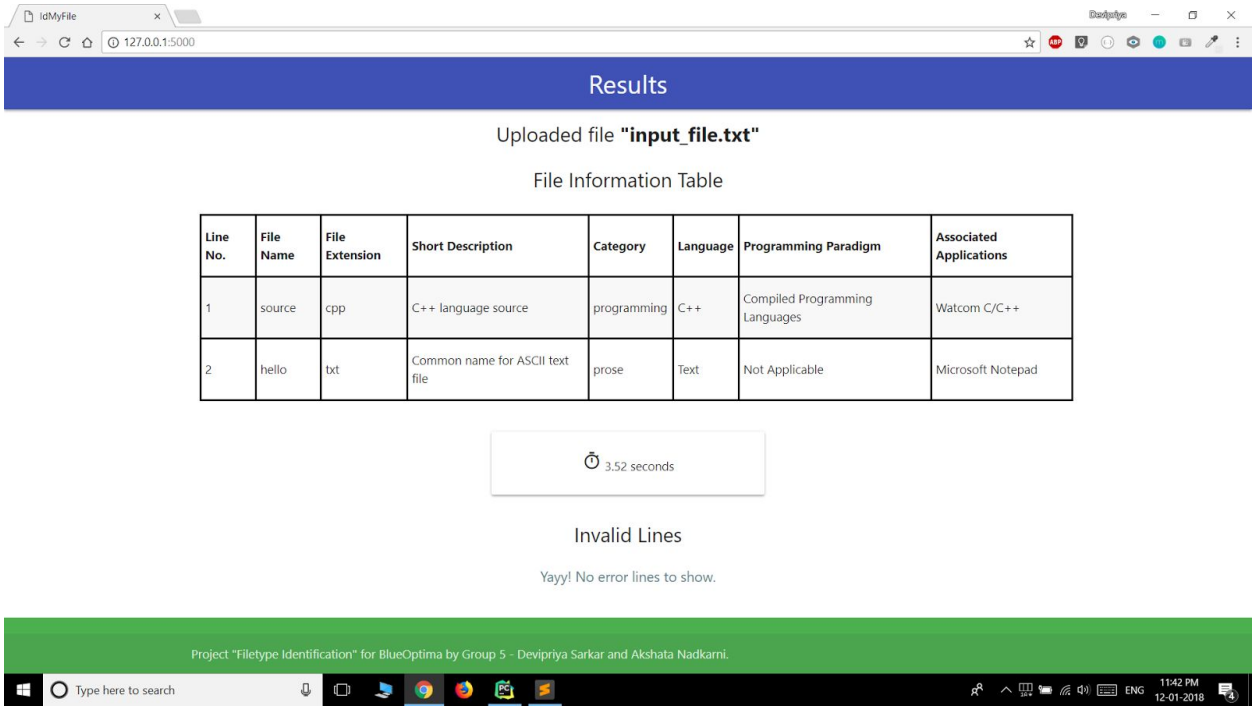
- First Screen



- Browse Input File



- Results Screen



REMARKS: Response time of ~50s for an input file with 50 lines.