README.MD

What is MUGHA

MU GitHub Analyzer is a program that accesses GitHub projects to find data that can be used (by other programs) to find trends and their effects in Software Development.

A summary of MUGHA’s capabilities as of May 2017

* Takes lists of GitHub usernames and a user’s login information.
* Retrieves all repositories from the users in the lists.
* Retrieves attributes about those repositories.
* Imports those repositories to the user’s machine.
* Can perform static analysis on those imported repositories if they’re in Java.
* Stores that information in a MySQL database.
* Shows the information from the database in the UI.
* Can delete that information from the database via the UI.

Please consult the requirements document if you have access to see what improvements are expected for it.

**This requires mysql**

1. Make sure you have MySQL installed on your system. You can follow the instructions to do so here: <https://dev.mysql.com/doc/refman/5.7/en/installing.html> .
2. MySql MySql should have a userid configured as ‘root’ with password as ‘’.

Mugha software expects the MySql instance service to be available in the localhost at port number 3306.

**How to install MUGHA**

1. Get the code from this repository <https://github.com/sharkfan/MU-GitHub-Analyzer>
2. Make sure you have Java installed on your system. You can follow the instructions to do so here: <https://java.com/en/download/help/download_options.xml> .
3. Open the application by running the code as a Java application in Eclipse or exporting it as a Jar file which you can also do in Eclipse by clicking File → Export and select Runnable Jar File and inputting settings like these (MU22 should instead be the name of the project you imported into Eclipse)

**Suggestions for future work to do**

Probably break up some of the code in MainController.java into smaller files.

Figure out how to get data from a repository across its commit history instead of the current state mugha currently gets.

Add more metrics to the search tab as the requirements state.

Add more features to the candidate pool tab as the requirements state.

Separate static analysis into its own method ran by its own button as requirements state.

Static analysis is currently limited to Java and embedded into the act of searching and so we made the decision to ignore non-Java projects to show off this function, though it used to not ignore Java projects, and it should go back that way.

Let users select more than one item on the UI table at a time.

Change the requirements to have it so that the database only stores the key and the static analysis metrics and retrieves the rest of the metrics on startup.

The customer proposed that there can be a button that adds 30 random usernames to the username input. That can be a requirement. Also there’s a site that randomly finds repositories that could be where to start: <https://gitrandom.com> <https://github.com/gkrizek/gitrandom>