This is why I commit to Python and rely on it to help me succeed.

We have chosen Python because it is a language that both the contributors of this project have previous experience working with Python and are confident that it is a good fit for the project. The following requirements will be met:

1. *UNIX Support*

All UNIX machines come with a Python interpreter and therefore do not need any code changes for running programs

For the following package requirements we will be using Python's import which allows a program to invoke a module and gain access to any accessible functions that are inside the module.

2. *Support for UNIX-style STDIN, standard I/O and TCP/IP sockets;*

Python's sys library gives the program the ability to access variables used by the interpreter such as **`sys.stdin`** for UNIX-style standard input (STDIN).
Standard I/O is made accessible to the user directly to the command line through the print function and for file I/O through methods attached to the file object.
Python's socket library allows for UNIX-style TCP/IP socket programming.

3. *Modular programming (think modules, functors, packages etc.)*

Python's module system allows for the creation of modules by saving a file. The basic format allows for any file within a directory to be invoked through the **`import`** keyword (as explained above). A package can be created and made available outside of a directory by appending the path to a set of modules that contain an **`__init__`** file within the base directory. More detailed information about modularity can be found within the documentation.

4. *Reading and writing JSON*

Python's json library provides support for reading from and writing to JSON files. For example, **`json.dump()`** will write a JSON object to a file. **`json.load()`** will read from JSON files and deserialize the data into a Python object.

5. *Loading code dynamically*

Python's magic import (**`__import__`**) allows for dynamic loading of Python modules (and packages by extension)

6. *Automatic unit testing and test coverage*

PyCharm provides a unit test framework unittest modelled after JUnit. Unittest allows users to thoroughly test their code without having to modify it. PyCharm also allows users to enable code coverage measurements.

7. *an IDE with support for exploratory programming.*

We will be using the IDE PyCharm for the project. PyCharm supports exploratory programming by allowing users to assess code quality, problem solve and debug. PyCharm provides an interface for users to write correct and well-tested code.