# Classes and Functionality

|  |  |
| --- | --- |
| **Class name** | **Functionality** |
| MainPage | User authentication and directing to main Application |
| Edit | Adds Processor information in database, provides links to update that information, provides options to query and finally to compare between multiple processors |
| Details | Provides information from the database and also mode to update that information |
| Compare | Provides information about multiple processors for comparison |
| MyProc | Class used for storing object information in the form of objects(Bean Class in Python). |
| MyUser | Class for storing user information in database. |
| ProcList | Class for creating list, used for displaying results |

Figure | Classes in the application

|  |  |
| --- | --- |
| **Class Name** | **Files** |
| MainPage | main.py and main.html |
| Edit | edit.py and edit.html |
| Compare | compare.html and compare.py |
| Details | details.py and details.html |
| MyProc, ProcList, MyUser | snippets.py |
| Other files | app.yaml(manifest file for Google App Engine) |

Figure | information About the Files for each class

# 2. Details information of class methods

**Information about the methods in various classes are listed in the below table. Most classes will have a get and a post method used for communicating with the server which is hosted locally except for the classes that are used to store information in the form of objects.**

**Class MainPage:-**  In this Class we make use of User ID to create an identification for the user and this information is added to the database. This class checks the user ID and then it displays the welcome message biased on the fact that the user has used the application or is using the application for the first time. All this is done using only the ***get*** method as we are pulling the values of the user from the users class. (Generally post method is used to push information in the database or application and get method is used to fetch the information)

**Class Edit:-** In this class, we have the get and the post methods that serve their purpose.

* Post Method: The ***post*** method is used to generate an object for the processor and then put that information in the database. The information is retrieved from the form in the ***edit.html*** file and then we check if that processor is already in the database: if it exists then we discard the new entry and keep the old information.
* Get Method: The ***get*** method here does most of the work since we are displaying all the values and also are including filters in our code that are Queries fired using the NDB commands in python. For displaying all the results we simply make use of the query.fetch() command that provides us all the entries made in the database. To make use of filters, we make use of filter method provided by the NDB. We also pass the values of strings to the compare page when the compare button is pressed. In this case, we first get the names of the strings from the values displayed in the table and then concatenate them in a string and pass it to the Compare class which will use it to display the information. We also pass the name of the processor in the details page so that the Details class can use it to display the information of that processor.

**Class Details:-** In this class, we provide information of the processor and provide means to update the processor information as well.

* post method: The post method here is used in case we try to update the information of the processor (excluding the name, since the name is the key of the processor object that we are creating.)
* get method: The get method here is used for receiving the name of the processor and then passing that information onto the post method for processing the processor information. The get method is used to pass the object to the html template in the template\_values list.

**Class Compare:** In this class, we pull the information about multiple processors and then display that information in a tabular format so that the user can see the key elements in the code.

* post method: The post method is used only used to trigger a button called previous page that is used to redirect to the Edit page.
* get method: The get method receives the names of the processors that were selected in the form of a single string, then the string is split and we obtain all the names of the processors. We then retrieve the objects of those specific strings store them in a StructuredList and then iterate through that list on the html page.

**App.Yaml:**  This is the most important file in the whole code as it tells the Google App Engine about the handlers in the project, which runtime to use and what libraries we should be using. It works the same way as a manifest file works in android.

# 3 Models and Data-Structures

In the application the main data-structure that is used is a ***list***. The list is used to pass information to the jinja2 module. This information is then displayed on the html pages.

Another data-structure that we use is a ***Structured List***. This is a list that stores objects of the type processors. This means that if we iterate through this list we will get all the objects that are stored in it. We use this in the Compare class, where we are comparing between multiple processors. Once we get the names of the processors we want to compare, we get their objects and then store them in the Structured list which is then iterated in the compare.html page.

# 4 UI Decisions

The UI consists of simple designs so that all users would feel ease while trying to use the application. The colours used a light and the contrast level is low so that users don’t feel burning sensation in their eyes while trying to use the application for longer periods of time.

The design considers the fact that human will interact with the application more often and that user will not have any issues while trying to use the site.