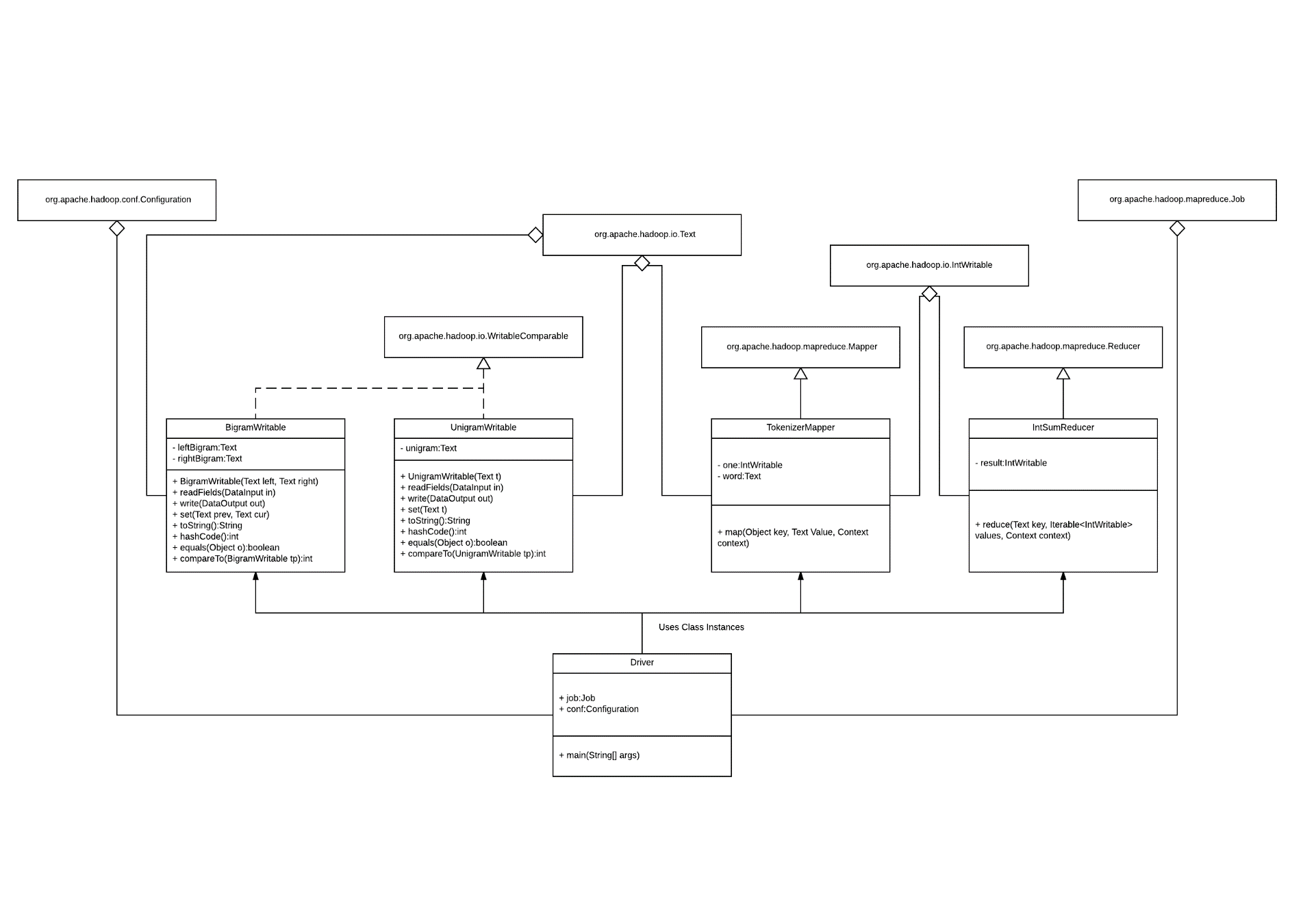
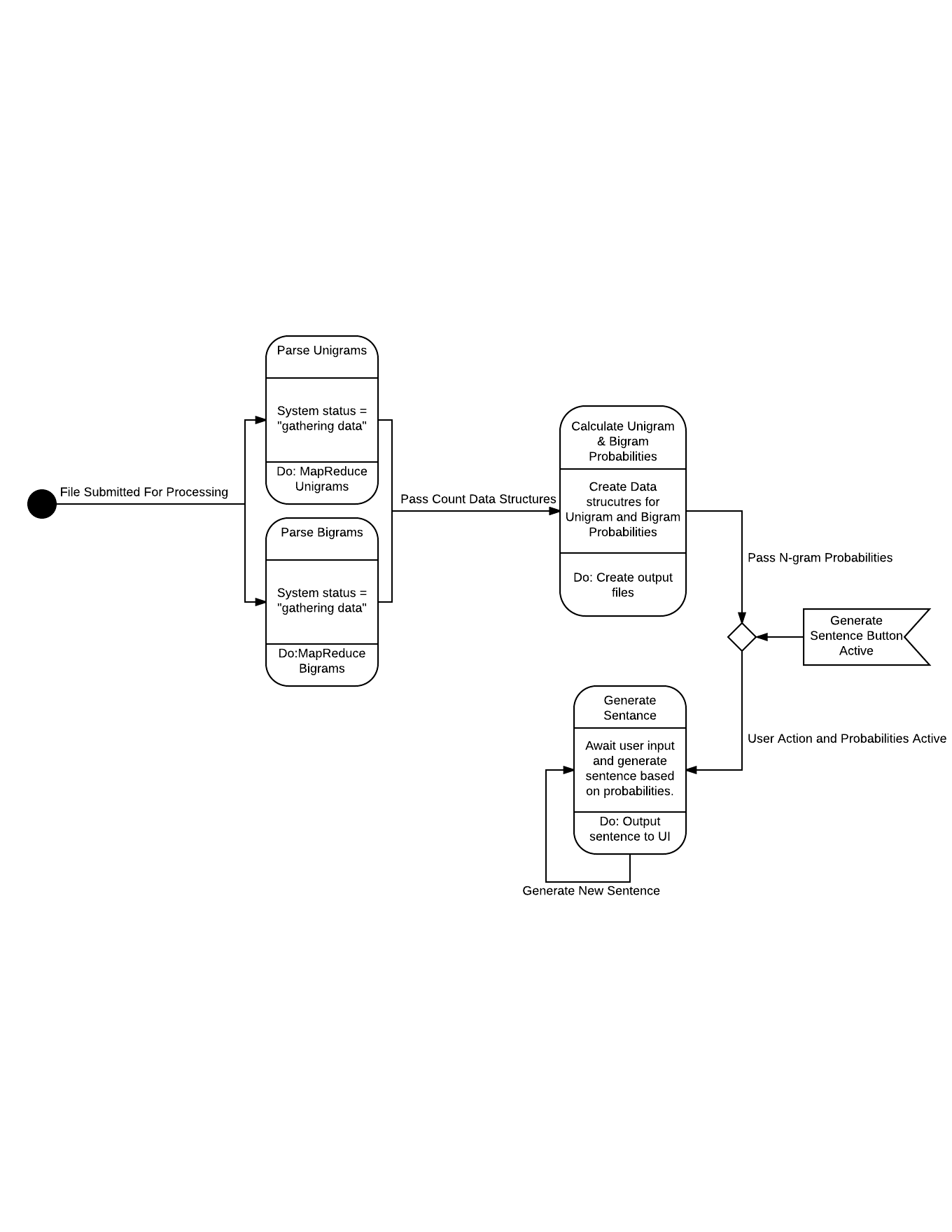
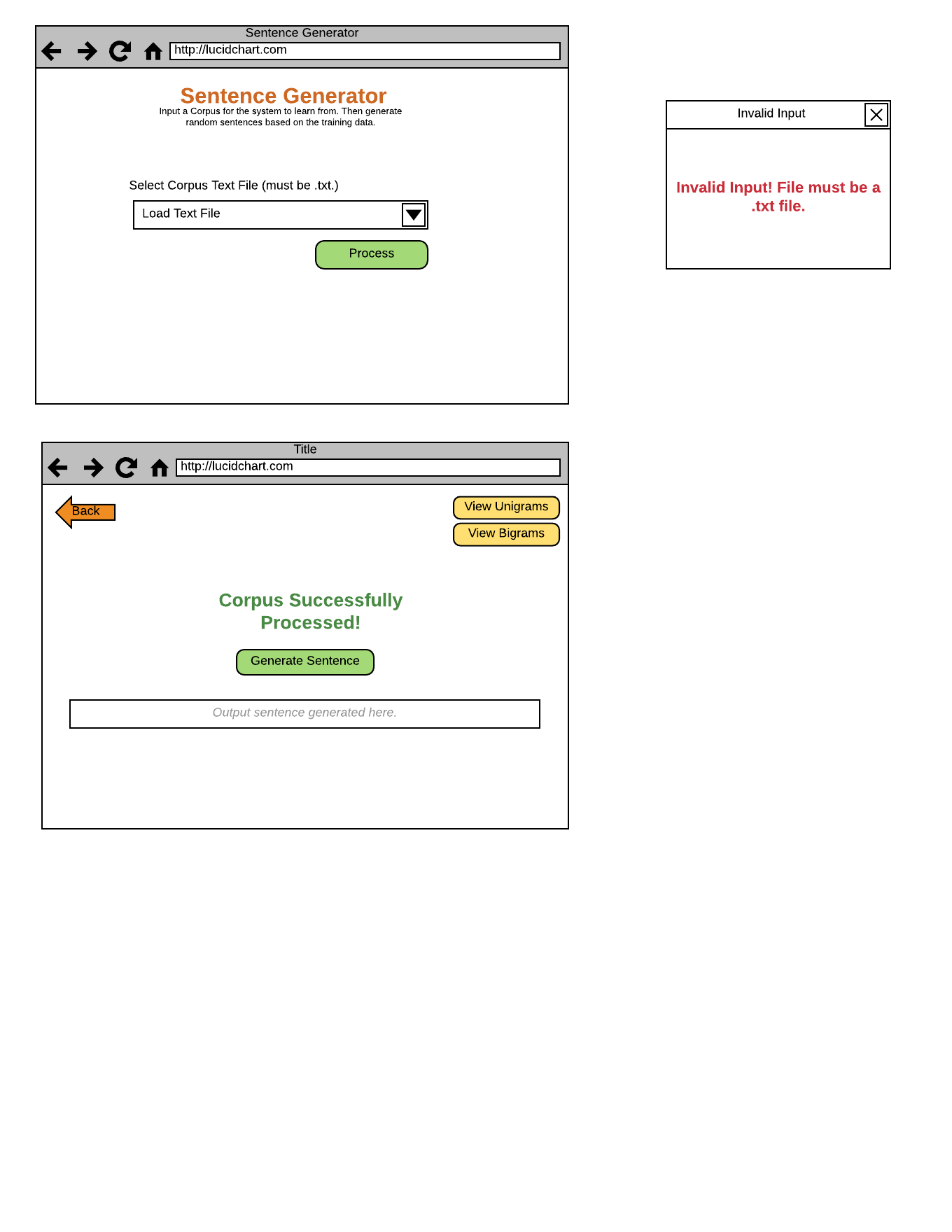
**Use Case Diagram**

**Design Class Diagram**

**State Chart**



**User Interface Mock Up**

**Eight Golden Rules of Interface Design**

1. Strive for consistency.
   1. The interface is designed to be simplistic and intuitive, with a very limited amount of interaction to reduce confusion. This allows consistency, as the interface only has to respond to a small amount of use cases.
2. Enable frequent users to use shortcuts.
   1. A back button is implemented for easy navigation back to the main page. This allows for easy submission of new corpra to generate new sentences with different training data.
3. Offer informative feedback.
   1. If the input data is not in the proper format, a pop-up will display informing them of their error. Otherwise, a new page is shown displaying the words “Corpus Successfully Processed” which informs the user that they are ready to generate sentences.
4. Design dialog to yield closure.
   1. A dialog box is used to warn the user of erroneous and invalid input.
5. Offer simple error handling.
   1. Same as the previous. The only error the user can make when using this system is submission of invalid input. This is easily handled by a dialog box. Input will be sanitized to only allow submission of files.
6. Permit easy reversal of actions.
   1. Once the corpus is processed actions cannot be reversed. However, a back button is located in the lefthand corner to allow easy submission of different corpra. New sentences can be generated without reprocessing the corpra if the user remains on the output page.
7. Support internal locus of control.
   1. The system only responds on user input. Once a correct file path is input into the home screen, the interface awaits action of the user. The user guides the system by clicking the “Process” button when they feel like they have chosen the correct file to be used. Once the corpra is parsed, the next page is loaded, informing the user of its success and displaying a button for generating random sentences. Action is only taken once this button is pressed, outputting a sentence based off the initial input which is used as training data. The user participates throughout the entire process, controlling the two major steps that the system goes through.
8. Reduce short-term memory load.
   1. Buttons are intuitive, with clear description of what they do. A short description is included on the home page to quickly and easily instruct the user on how to proceed. The interface is minimalistic to insure confusion is reduced, and actions to take are obvious.

**Test Case Design**

Unit Testing: The unigram and bigram classes will be tested individually to ensure they work on their own accordance. Their accuracy can be easily tested by observing the output files that are generated by the Hadoop framework, after processing a corpus. Probabilities will also be displayed in output files which will determine if the algorithm is working correctly for unigrams and bigrams individually. Next the interface will be tested to ensure it is properly taking in input from the user, and navigating to the right page or displaying an error depending on the user’s input.

Integration Testing: Once the unigram and bigram classes are determined to be working correctly as well as the interface, testing of these objects working together will commence. The webpage should be able to communicate properly with the Apache Hadoop infrastructure, submitting the right data to it, as well as retrieving the output properly. This will be controlled by a main driver class which will orchestrate the transition and passing of data between modules.

|  |  |  |  |
| --- | --- | --- | --- |
| Functionality Tested | Inputs | Expected Output | Actual Output |
| Submit File | Text file | System either responds with error message or proceeds to next page. |  |
| Download Bigram File | User Action | Bigram file is download, displaying all bigram counts and probabilities. |  |
| Download Unigram File | User Action | Unigram file is download, displaying all unigram counts and probabilities. |  |
| Generate Sentence | User Action | Random sentence is generated. Additionally, this can be used to determine probabilities are being used correctly. |  |

**Requirements:**

Almost all methods are made public for all implemented classes as they are almost all exclusively used by other classes. However, most internal variables are all made private, as they should not be modified by external modules under and circumstances. Modification of these variables most only occur by invoking the appropriate method for that class (usually a set() method), and likewise for retrieving variables.

The use of the Apache Hadoop infrastructure allows for a significantly elegant and efficient design, as it utilizes parallel processing techniques to offer one of the most efficient ways for handling large data, such as corpra.