Phase 1

Load_json.py creates the database named "291db" and loads the data provided by the json file to a collection named "dblp" in the database "291db". The file uses mongoimport command to load the json files.
We also create text indexes in this file to make queries run faster in the phase 2.

Phase 2

Author_search:
The user inputs keywords and is able to see all the authors accordingly. We search the database by using an aggregation function. Finding all the authors that match the keywords.
We use the following function and its logic:
Unwind: Separates the keywords to find the correct authors
Match - Find matches the right column
Grouping by the author at the end
Printing out everything

Article Search:
The user is able to provide keywords and the function retrieves all articles that match the keywords. We start by creating an empty list, and appending the input from user into separate words and add to the list. Then we find the matches use find() command and display the results to the user.
We ask the user to enter the id of the selection, and display all the corresponding fields, followed by all the references that refer to the article.

List Venue:
We ask the user to input the number of top venues they would like to see. We then use an aggregate function to group by id and to count the number of venues. We sort descending by the count of venues and limit the returned result to the number input by the user. We could not implement the functionality to display the number of articles that reference a paper in a venue.

Add_article:
Asks to provide a unique id. And once inputted we check if the id is unique. We use the function count_document to count if any id input matches any stored id. We want the count equal to zero as that would be interpreted as unique. Once the id is checked, the user is asked to input title, authors, and year of the article. We then try to add the information into the collection.

Quitting the program:
The user is able to quit the program at any time by hitting ctrl+c key.

Testing strategy:
We loaded the json file and did incremental testing at every stage of development and compared results to expectations.

Work-distribution:
Arvinder took on phase 1 and creation of the document.
Nicolas worked on author search and article search for phase 2.
Khushi worked on list venue and add article for phase 2.
Everyone helped with testing and debugging.