Table of Contents

Documentation

Wordle Clone

2022

2. Login

3. Registration

4. FetchScore

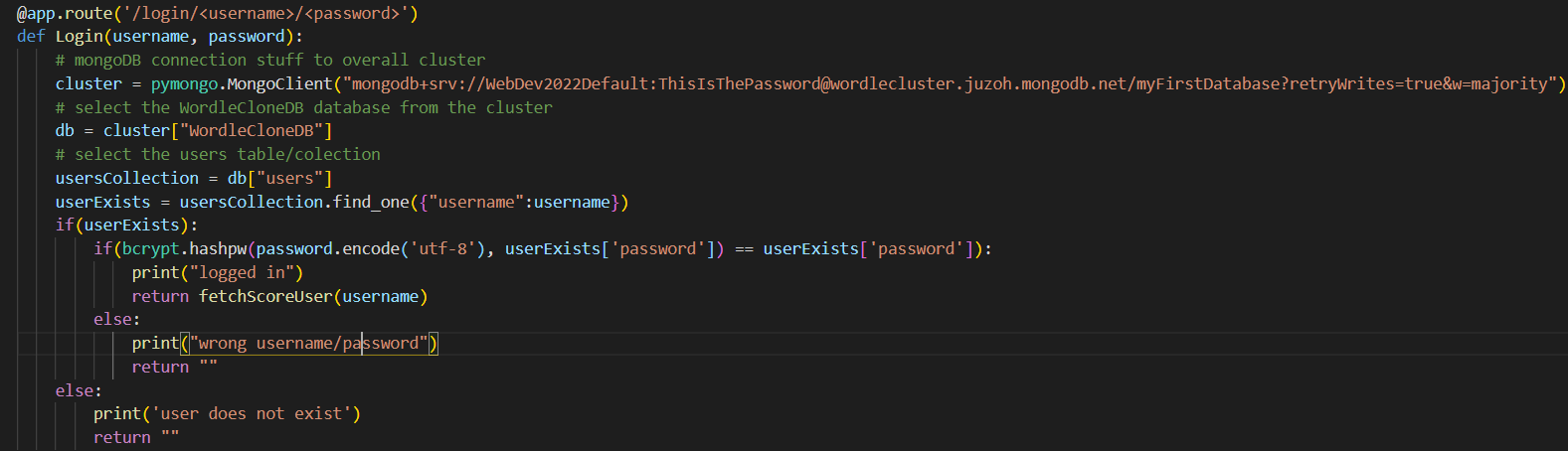
5. FetchScoreDate

6. FetchScoreUser

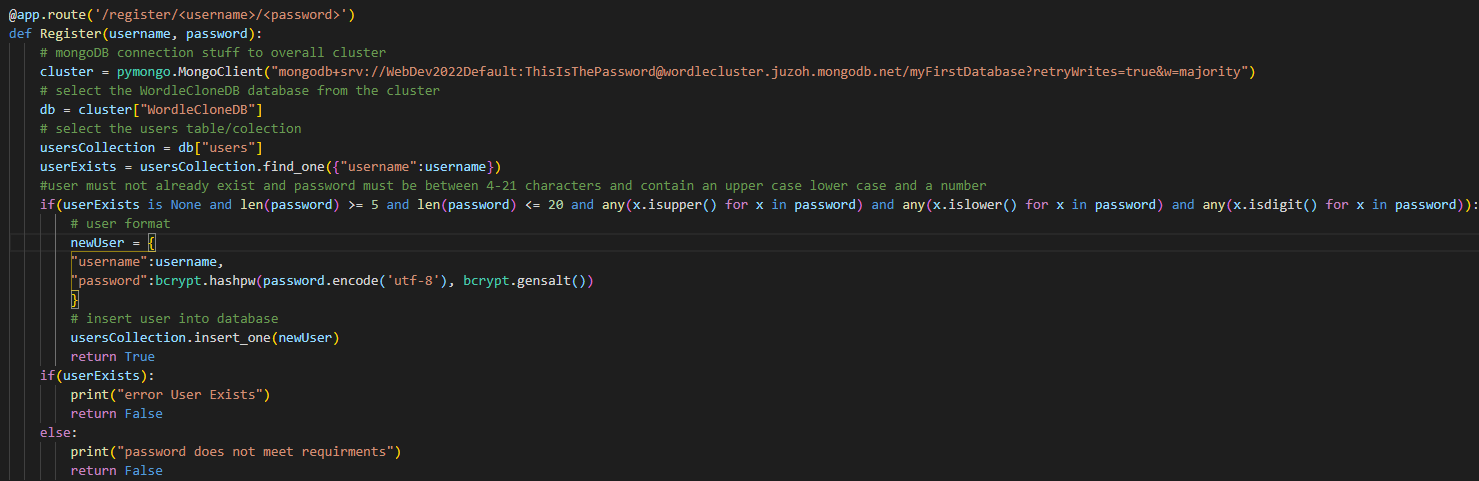
7-9. DataBase

10-11. DailyWord

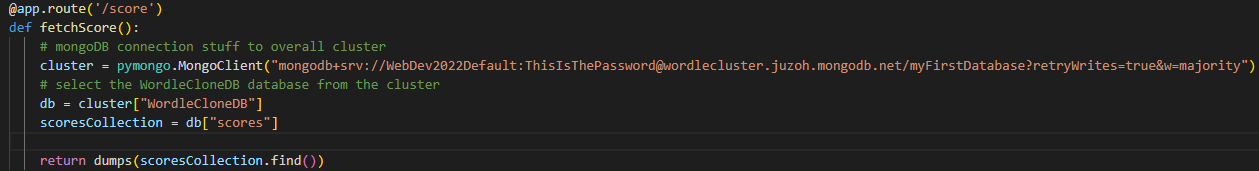
|  |  |
| --- | --- |
| Name: Login | Description: Logs the user in |
| API route:  @app.route('/login/<username>/<password>')  API Call:  “<http://127.0.0.1:5000/server/login/><username>/<password>” | <username> : username string  <password>: password string    Example: “[http://127.0.0.1:5000/server/login/peter/Peter1”](http://127.0.0.1:5000/server/login/peter/Peter1%E2%80%9D) |
| API function:  def Login(username, password)  params:  (String) username: Must be 4 - 25 characters long, and a string.  (String) password: Must be 4 – 25 characters, cannot contain special  returns:  Dict of user scores | Takes in user name and password request from frontend and confirms the information is correct then returns the users list of scores to the frontend |
| Requirements:  Pymongo: pymongo.readthedocs.io links to mongoDB also used to convert the mongoDB cursor to JSON  Bcrypt: <https://www.npmjs.com/package/bcrypt> used to encrypt the password |  |



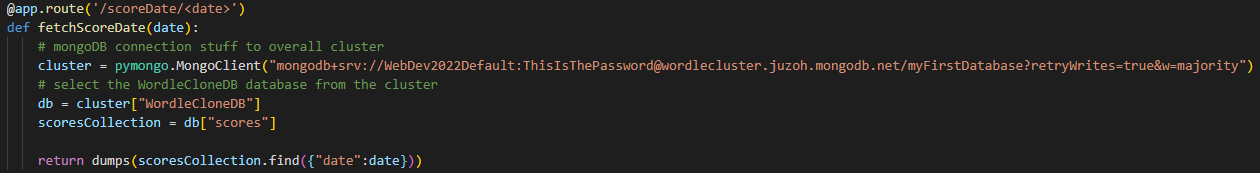
|  |  |
| --- | --- |
| Name: Register | Description: creates an account for the user |
| API route:  @app.route('/register /<username>/<password>')  API Call:  “<http://127.0.0.1:5000/server/register/><username>/<password>” | <username> : username string  <password>: password string    Example: “[http://127.0.0.1:5000/server/register/peter/Peter1”](http://127.0.0.1:5000/server/register/peter/Peter1) |
| API function:  def Register(username, password)  params:  (String) username: Must be 4 - 25 characters long, and a string.  (String) password: Must be 4 – 25 characters, cannot contain special  returns:  a string saying either true or false to the front end letting them know if the registration was a success or not | Takes in user name and password request from frontend and confirms the username does not exist and the password meets the requirements then sends back a string saying true if the user has been added to the database or false if not |
| Requirements:  Pymongo: pymongo.readthedocs.io links to mongoDB also used to convert the mongoDB cursor to JSON  Bcrypt: <https://www.npmjs.com/package/bcrypt> used to encrypt the password |  |



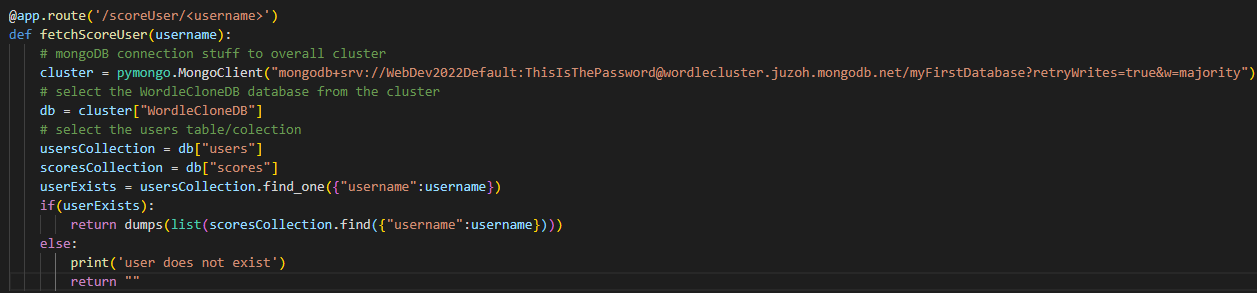
|  |  |
| --- | --- |
| Name: fetchScore | Description: returns all scores in database |
| API route:  @app.route('/score ')  API Call:  “[http://127.0.0.1:5000/server/score”](http://127.0.0.1:5000/server/score) | Example: “[http://127.0.0.1:5000/server/score”](http://127.0.0.1:5000/server/score) |
| API function:  def fetchScore()  params:  none  returns:  dict of all scores in database | When called returns a dict of all scores in the database |
| Requirements:  Pymongo: pymongo.readthedocs.io links to mongoDB also used to convert the mongoDB cursor to JSON |  |



|  |  |
| --- | --- |
| Name: fetchScoreDate | Description: returns all scores in database |
| API route:  @app.route('/scoreDate/<date>')  API Call:  “[http://127.0.0.1:5000/server/scoreDate/<date>/”](http://127.0.0.1:5000/server/scoreDate/%3cdate%3e/) | <date> string of the date you are looking for  Example:  “[http://127.0.0.1:5000/server/scoreDate/04-01-2022”](http://127.0.0.1:5000/server/scoreDate/04-01-2022) |
| API function:  def fetchScoreDate(date)  params:  (String) date must be formatted "%d-%m-%Y"  returns:  dict of all scores from date | When called returns a dict of all scores in the database from the date passed |
| Requirements:  Pymongo: pymongo.readthedocs.io links to mongoDB also used to convert the mongoDB cursor to JSON |  |



|  |  |
| --- | --- |
| Name: fetchScoreUser | Description: returns all scores in database |
| API route:  @app.route('/scoreUser/<username>')  API Call:  “[http://127.0.0.1:5000/server/scoreUser/<username>/”](http://127.0.0.1:5000/server/scoreUser/%3cusername%3e/) | <date> string of the date you are looking for  Example:  “[http://127.0.0.1:5000/server/scoreUser/peter”](http://127.0.0.1:5000/server/scoreUser/peter) |
| API function:  def fetchScoreUser(username)  params:  (String) date must be a valid username"  returns:  dict of all scores from username | When called returns a dict of all scores in the database from the username passed |
| Requirements:  Pymongo: pymongo.readthedocs.io links to mongoDB also used to convert the mongoDB cursor to JSON |  |

**Database**

Requirements:

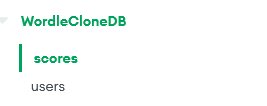
Libraries:

pymongo

datetime

The database was made with MongoDB.

The database contains 2 collections or tables, the scores table and the users table.



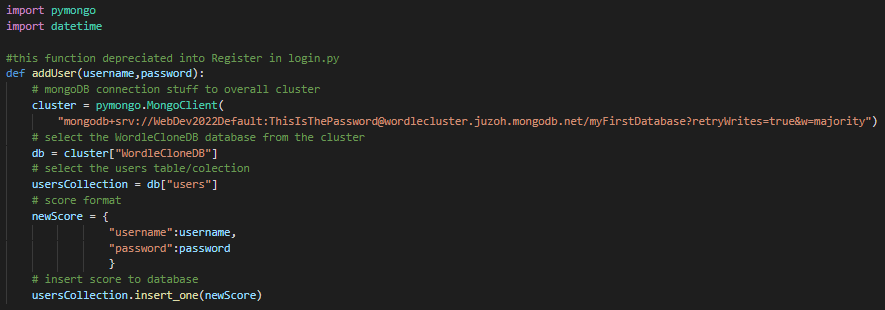
The scores table contains all the scores for each user, with each entry having an ID, the username (string), score (integer) and date (in seconds since epoch UTC). 

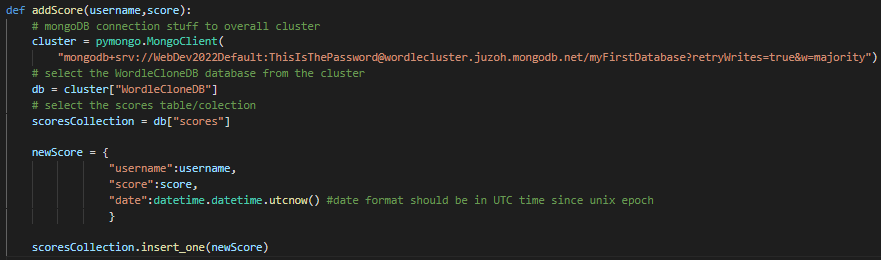
Figure: Example of an entry in the scores table with formatted date

The users table contains all the user's info. This includes the ID, the username (string), and the password (string). 

Figure: Example of an entry in the users table

Below is an example in python of how you can connect and add to the database.





The following links were used to learn how to implement a basic database in MongoDB

<https://www.mongodbtutorial.org/>

<https://www.tutorialspoint.com/mongodb/index.htm>

|  |  |
| --- | --- |
| Name: Daily Word | Description: Fetches the word for the day |
| API route:  @app.route('/dailyWord')  API Call:  “<http://127.0.0.1:5000/server/dailyWord/> |  |
| API function:  def dailyword():  returns:  The result of get\_daily\_word() | Fetches the word for the day |
| get\_daily\_word() | Checks the current date and if the current date is greater than the date the previous time get\_daily\_word() was called then generate a new word for the day using word\_of\_the\_day(). If it is the same date then return the previously generated word for that day. |
| Word\_of\_the\_day() | Uses the random\_word library to generate random words until that word is 5 letters long. It then returns that 5 letter long word. |
| Requirements:  random\_word: <https://pypi.org/project/Random-Word/> Generates a random word to use.  datetime: <https://docs.python.org/3/library/datetime.html> Allows to fetch the current date. |  |

Text

Description automatically generated

Text

Description automatically generated