Backend Documentation

In order to run the application - the following 2 dependencies need to be added.

1. Flask
2. Flask-CORS

These are mentioned in the setup.py file in the backend directory.

Filename: app.py

Functions -

1. ***signup()***

Route: /signup Method: POST

There are 5 parameters in the JSON sent to this API - first name (**firstName**), last name (**lastName**), email (**email**), contact (**contact**) and password (**password**).

Response -

a. If there already exists a row in the users table with the same email address, the message returned is “An account with this email already exists”.

b. If there already exists a row in the users table with the same contact information, the message returned is “An account with this contact already exists”.

c. In the case when neither of the above two hold, a new row is created in the users table with the information from the JSON and the message returned is “Added successfully”.

2. ***login()***

Route: /login

Method: POST

There are 2 parameters in the JSON sent to this API - email (**email**) and password (**password**).

Response -

a. If the email does not exist in the users table, the message returned is “Please create an account!”.

b. If the email exists, but the password entered is wrong, the message returned is “Invalid credentials!”.

c. If the email and password pair is correct, the message returned is “Logged in successfully”.

3. ***create_bid() Route: /bid/create***

Method: POST

There are 3 parameters in the JSON sent to this API - productId (**prodId**), email (**email**) and amount (**bidAmount**).

From the product table, we find the minimum amount of the bid that the seller had mentioned.

Response -

a. If the bid amount is less than the minimum amount, the message returned is “Amount is lesser than initial price”.

b. If the bid amount is greater than the minimum amount, the bid is created/updated in the bids table and the message returned is “Saved bid”.

4. ***create_product() Route: /create/product***

Method: POST

There are 6 parameters in the JSON sent to this API - productName (**productName**), sellerEmail (**sellerEmail**), initialPrice (**initialPrice**), increment (**increment**), photo (**photo**) and description (**description**).

Response -

This API creates a new product in the product table with the aforementioned details and returns the message “Added product successfully”.

5. ***get_all_products()***

Route: /product/listAll Method: GET

Response -

It returns the details of all the products that are present in the product table.

6. ***get_product_image()***

Route: /product/getImage

Method: POST

There is 1 parameter in the JSON sent to this API - productId (**productID**). Response -

It gets the image corresponding to the productId and returns it in the response.

7. ***get_product_details()***

Route: /product/getDetails

Method: POST

There is 1 parameter in the JSON sent to this API - productID (**productID**).

Response -

The first name, last name and bid amount of the top 10 bids on this particular productID are extracted by making use of users, product and bids tables and returned as the response.

8. ***update_product_details()***

Route: /product/update

Method: POST

There are 6 parameters in the JSON sent to this API - productId (productID), productName (productName), deadlineDate (deadlineDate), description (description) and increment (increment).

The previous initialPrice is extracted from the row in the product table which corresponds to the same productId.

The row is updated according to the details sent in the JSON.

Response -

The message returned in the response is “Updated product successfully”.

9. ***get_landing_page()***

Route: /getLatestProducts

Method: GET

This API gets the details of the 10 latest products that have been added to the inventory. It finds the maximum bid amount (if any bids have been made) from the bids table for each of these products. Further it finds the first name and last name of the people who made these bids.

Response -

The aforementioned details are returned as the response.

10. ***get_profile():***

Route: /profile

Method: POST

This API fetches the profile details of all its customers. It finds the personal information for every customer as well as the products the customer has places for sale and to place the bids. For each customer the following details are displayed:

a. First name
b. Last Name
c. Email Id
d. Contact Number
e. Number of products on sale
    - Link to each of these products
f. Number of products for bid
    - Link to each of these products