COLLEGE OF ENGINEERING AND COMPUTER SCIENCE

Fall 2020

CPSC 449 Web-Backend

Project 2 - Documentation

Report Prepared By:

CWID	Name
887480747	Venkata Pranathi Immaneni

Table of Contents

1 How to run the Project?

Step 1: Run the below command to initialise the database

```
flask init
```

Step 2: Run the below command to start the two processes in different ports. Normally one service runs in 5000 port and other runs in 5100 port.

```
foreman start
```

2 Users Microservice

File Name: app.py

NOTE: Users Microservice Runs in 5000 PORT

2.1 createUser API:

createUser API is used to create a new user account. If the user already exits, it returns the error response.

- This is a POST Method – which requires username, email and password.
- Here the request should be in JSON Format (as shown in the sample request). Response is also received in JSON Format.

API Endpoint:	/v1/createUser
Method:	POST
Request Content Type:	Application/JSON
Response Content Type:	Application/JSON

Working:

Send Request in the below format:

```
curl localhost:5000/v1/createUser -d '{"username": "Pranathi", "password": "Pass@123", "email": "ivpranathi@csu.fullerton.edu"}' -H 'Content-Type: application/json'
```

Success Response: You will receive the below response, when the user is successfully created.

Response

```
{
  "ContentLanguage": "en-US",
  "ContentType": "application/json",
  "Message": "User Created Successfully",
  "StatusCode": 200
}
```

Error Response: You will receive the below error response, when user already exists.

Response


```
{
  "ContentLanguage": "en-US",
  "ContentType": "application/json",
  "Message": "User Already Exists",
  "StatusCode": 409
}
```


Error Response: You will receive the below error response, when any of the required parameters – username, email or password are missed.

Response


```
{
  "ContentLanguage": "en-US",
  "ContentType": "application/json",
  "Message": "Missing Username or Password or email fields",
  "StatusCode": 400
}
```


2.2 authenticateUser API:

authenticateUser API is used to authenticate user by checking whether the user has passed valid username and password. It returns true if the given password matches with the hashed password stored in the database

- This is a POST Method – which requires username and password.
- Here the request should be in JSON Format (as shown in the sample request). Response is also received in JSON Format.

API Endpoint:	/v1/authenticateUser
Method:	POST
Request Content Type:	Application/JSON
Response Content Type:	Application/JSON

Working:

Send Request in the below format:

```
curl localhost:5000/v1/authenticateUser -d '{"username": "Pranathi", "password": "Pass@123"}' -H 'Content-Type: application/json'
```

Success Response: You will receive the below response, when the user is authenticated successfully. It returns **True** in the message if the password matches the hashed password in the database, else the message is returned as **False**

```
{
  "ContentLanguage": "en-US",
  "ContentType": "application/json",
  "Message": true,
  "StatusCode": 200
}
```

Failure Response: You will receive the below response, when the user authentication fails. It returns **False** in the message if the password does not match the hashed password in the database.

```
{
  "ContentLanguage": "en-US",
  "ContentType": "application/json",
  "Error": " Authentication Failed",
  "Message": false,
  "StatusCode": 401,
}
```

2.3 addFollower API:

addFollower API enables to start follow a particular user, with the specified username.

- This is a POST Method – which requires username and usernameToFollow
- Here the request should be in JSON Format (as shown in the sample request). Response is also received in JSON Format.

API Endpoint:	/v1/addFollower
Method:	POST
Request Content Type:	Application/JSON
Response Content Type:	Application/JSON

Working:

Send Request in the below format:

```
curl localhost:5000/v1/addFollower -d '{"username": "Pranathi",
"usernameToFollow": "Dinesh"}' -H 'Content-Type: application/json'
```

Success Response: You will receive the below success response, once the follower is added successfully.

```
{
  "ContentLanguage": "en-US",
  "ContentType": "application/json",
  "Message": "Follower Added Successfully",
  "StatusCode": 200

}
```

Failure Response: You will receive the below response, when the username or username to follow does not exists

```
{
  "ContentLanguage": "en-US",
  "ContentType": "application/json",
  "Message": "User to Follow not found",
  "StatusCode": 404

}
```


2.4 removeFollower API:

removeFollower API enables the user to stop following a particular user.

- This is a DELETE Method – which requires username and usernameToRemove
- Here the request should be in JSON Format (as shown in the sample request). Response is also received in JSON Format.

API Endpoint:	/v1/removeFollower
Method:	DELETE
Request Content Type:	Application/JSON
Response Content Type:	Application/JSON

Working:

Send Request in the below format:

```
curl -i -X DELETE -H 'Content-Type: application/json' -d  '{"username": "Pranathi",
"usernameToRemove": "Dinesh"}' http://localhost:5000/v1/removeFollower
```

Success Response: You will receive the below success response, once the follower is added successfully.

```
{
  "ContentLanguage": "en-US",
  "ContentType": "application/json",
  "Message": "Follower Removed Successfully",
  "StatusCode": 200

}
```


Failure Response: You will receive the below response, when the username or username to remove does not exists in the database


```
{
  "ContentLanguage": "en-US",
  "ContentType": "application/json",
  "Message": "User to remove not found",
  "StatusCode": 404

}
```

3. Timeline Microservice

File Name: timelinesApi.py

NOTE: Timeline Microservice Runs in 5100 PORT

3.1 postTweet API:

postTweet API enables the user to post the new tweet – in the form of text, which is visible to all of his followers.

- This is a POST Method – which requires username and post(text to be posted).
- Here the request should be in JSON Format (as shown in the sample request). Response is also received in JSON Format.

API Endpoint:	/v1/postTweet
Method:	POST
Request Content Type:	Application/JSON
Response Content Type:	Application/JSON

Working:

Send Request in the below format:

```
curl localhost:5100/v1/postTweet -d '{"username": "Pranathi", "post": "Its been a great day!!"}' -H 'Content-Type: application/json'
```

Success Response: You will receive the below response, when the post is successfully posted and saved to the posts table of users.db

```
{
  "ContentLanguage": "en-US",
  "ContentType": "application/json",
  "Message": "Tweet Posted Successfully",
  "StatusCode": 200

}
```

Failure Response: You will receive the below response, when the user does not exists

```
{
  "ContentLanguage": "en-US",
  "ContentType": "application/json",
  "Message": "User Not Found",
  "StatusCode": 400

}
```

3.2 getUserTimeline API:

getUserTimeline API Returns recent 25 tweets from the given username.

- This is a GET Method – which requires username

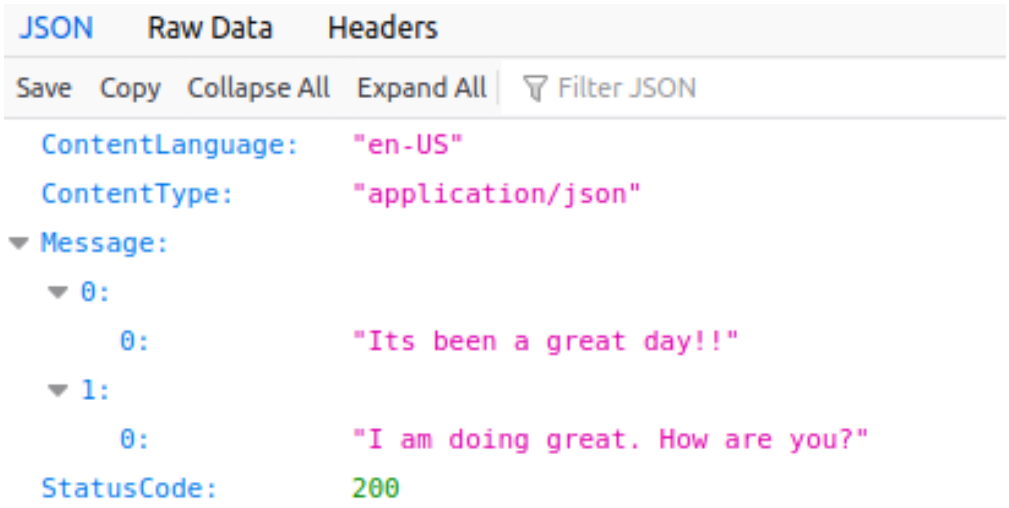
API Endpoint:	/v1/userTimeline/<username>
Example:	/v1/userTimeline/Pranathi
Method:	GET
Response Content Type:	Application/JSON

Working:

Send Request in the below format:

http://127.0.0.1:5100/v1/userTimeline/Pranathi

Success Response: You will receive the below response with the list of posts:



3.3 getPublicTimeline API:

getPublicTimeline API Returns recent 25 tweets from all the users, who posted recently.

- This is a GET Method

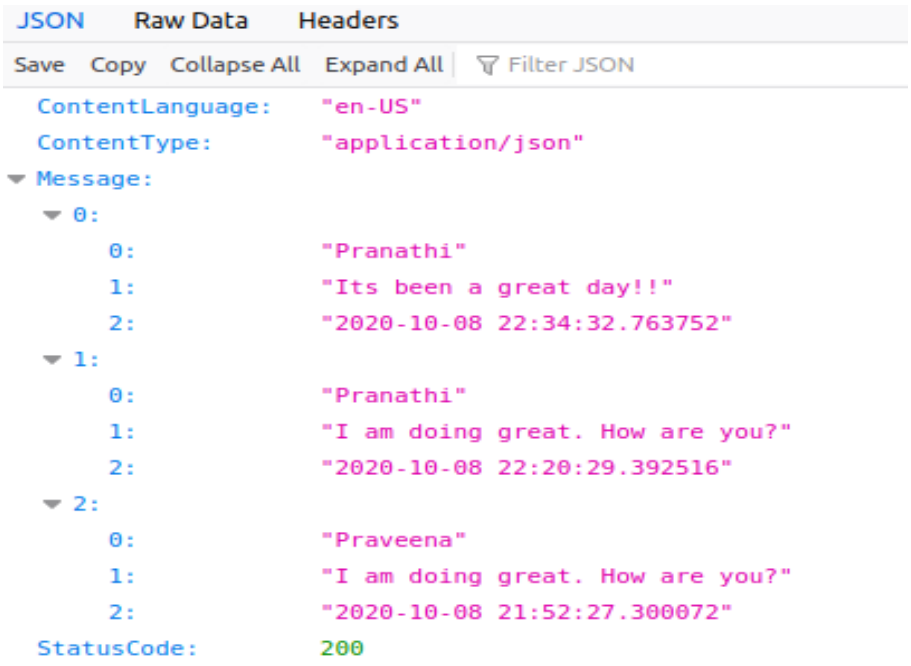
API Endpoint:	/v1/publicTimeline
Method:	GET
Response Content Type:	Application/JSON

Working:

Send Request in the below format:

http://127.0.0.1:5100/v1/publicTimeline

Success Response: You will receive the below response – retrieves all the posts from all the users



3.4 getHomeTimeline API:

getHomeTimeline_API Returns recent 25 tweets from all the users, that this specified user follows.

- This is a GET Method

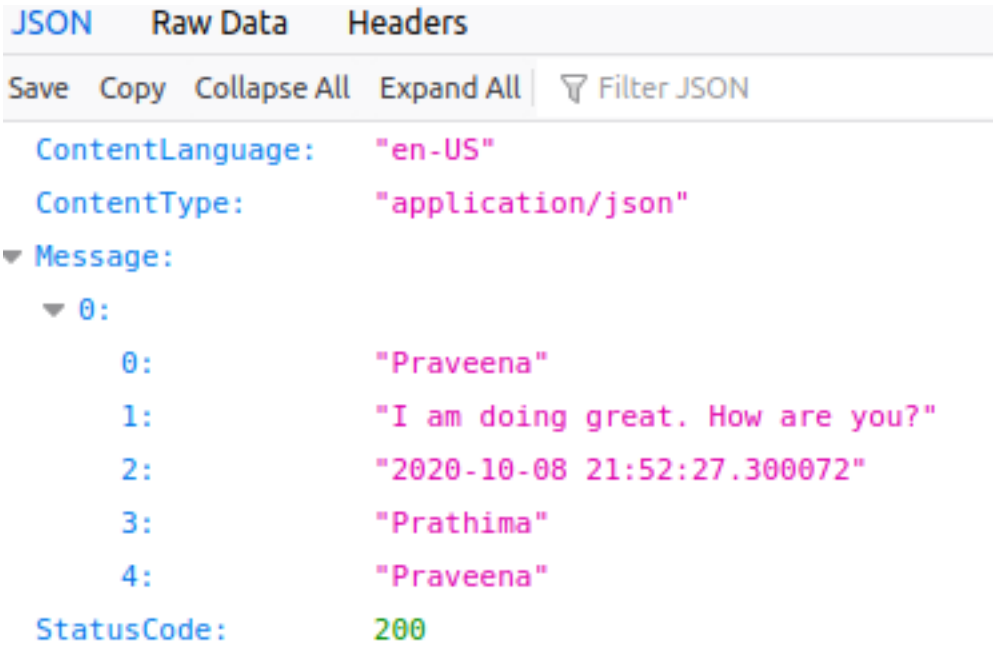
API Endpoint:	/v1/homeTimeline/<username>
Example:	/v1/homeTimeline/Prathima
Method:	GET
Response Content Type:	Application/JSON

Working:

Send Request in the below format:

http://127.0.0.1:5100/v1/homeTimeline/Prathima

Success Response: You will receive the below response – retrieves all the posts from user – this specified user follows


JSON
Raw Data
Headers
Save
Copy
Collapse All
Expand All
Filter JSON
ContentLanguage: "en-US"
ContentType: "application/json"
Message:
0:
0: "Praveena"
1: "I am doing great. How are you?"
2: "2020-10-08 21:52:27.300072"
3: "Prathima"
4: "Praveena"
StatusCode: 200