Name: Banty Kr Manjhi
banty9475@gmail.com

Backend API with required feature : Done

Tech Stack

Used: Flask, Postgresql
Can be used: Node.js, express.js, postgresql

Further Improvement

Current capability: The system currently can easily manage 1:1 traffic for good amount of user but with increase in number of users the speed will be slowed as the db query will take more and more time to return the data, also parsing the tweet data would become more difficult once the user base increases. Since this project is just showing the API design and needs much important changes but not limited to caching, db sharding, load balancer, distributed network overa all a microservice like architecture.

Things that we can do to make this more robust and scalable are:

- ☐ Caching: This is a very important part of a microservice where we can reduce the db call signficatally. In case of twitter like system this is really easy and important as top # tag which most of the users are interested in can be cached as per the location so that each time someone clicks on a hashtag we don't need a db call . Ex redis cache, flask cache etc.
- ☐ Distributed data center / network: As the service would be used by people around the world we need to have a local data center to reduce the time of each request.
- ☐ Database: As the application would deal in huge data we need to develop new mechanisms to query the whole system in the lowest time possible i.e sharding . We need both horizontal and vertical scaling in the DB part.
- ☐ Load balancer: This is really important for huge applications because even a minute downtime may impact the business and user experience. This will help the system in managing traffic and resources effectively and efficiently.
- ☐ Other: There are other thing we can do in the coding part itself to lower done the time complexity and increase the user experience:
    - ☐ Using appropriate data structure to store tweets, pictures, followers list etc.
    - ☐ Using class / module based coding so that the reusability of the code can be increased.
    - ☐ Using different algorithms while sorting, searching, deleting or updating which takes the min time possible.

- [ ] Preventing data leaks by using the concepts of abstraction and encapsulation