**Twitter Documentation**

**Front-end (web app) used:** Angular 8

**Back-end (server app) used:** Node 10

**Framework used:** Express

**DB used:** MongoDB

**Additional Utils used:**

1. request - to make http calls.
2. oauth - authenticate against OAuth providers.
3. mongoose - mongodb object tool.
4. express - small, robust tool.
5. body-parser - parse incoming request under req.body property.
6. dotenv - for loading environment variables from a .env file into process.env.

**Configs:**

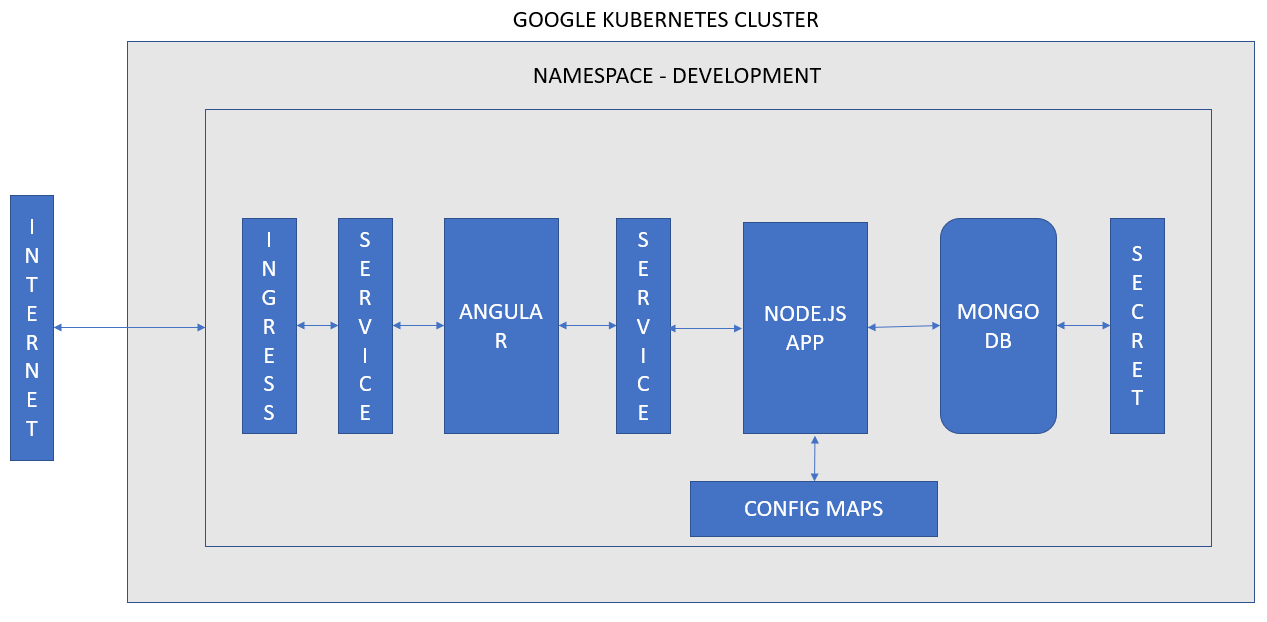
Secrets – contains database username and password from env file of node.js (process.env.keyName)

Config Maps – contains other properties such as Port, DB\_Name, Twitter Owner Dev Keys etc from env file of node.js (process.env.keyName)

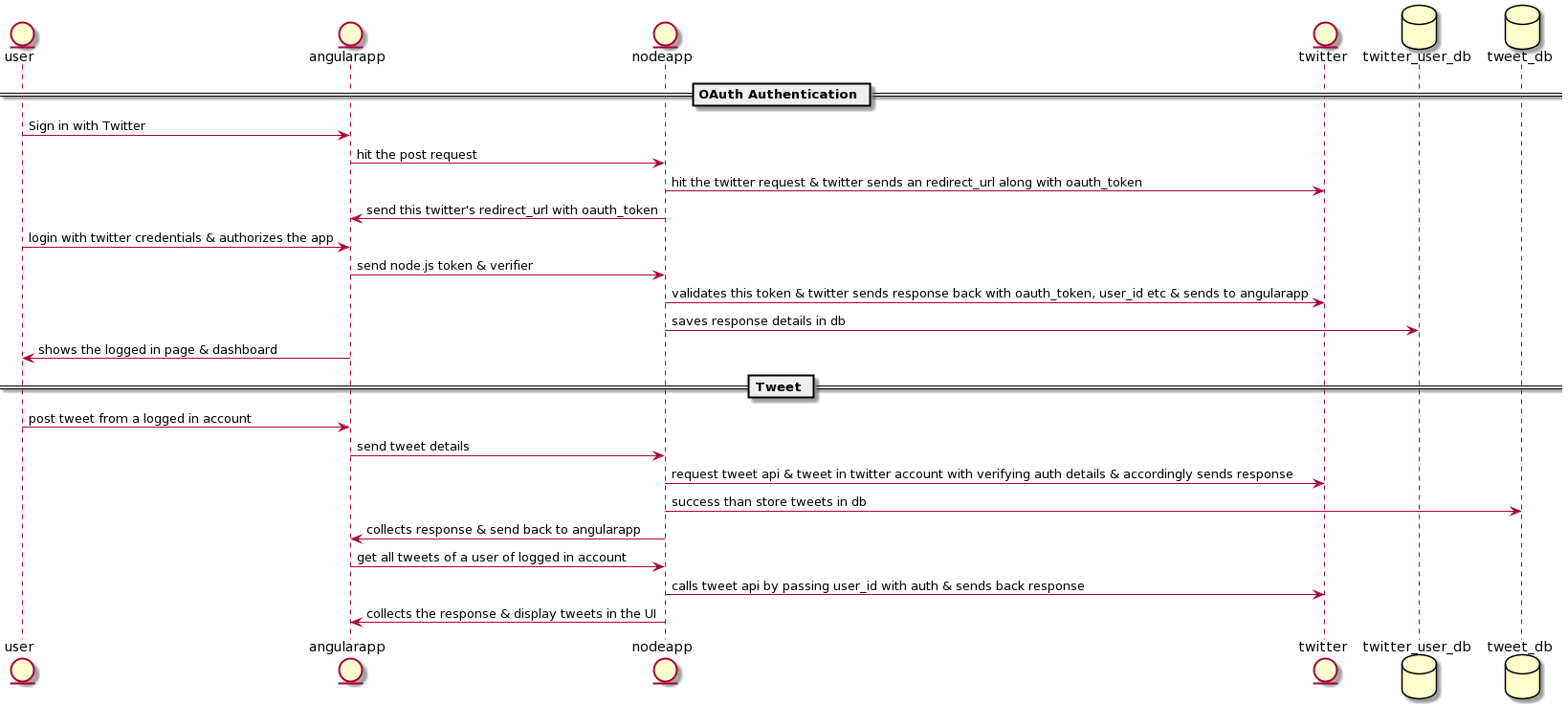
**Test Solution:**

1. User clicks on Sign in with Twitter button.
2. Twitter sends a request token.
3. User enter his twitter credentials and gives consent to the app to authorize it.
4. Twitter validates the credentials and if wrong credentials than return back to user with a message and if correct credentials returns back with oauth\_token and other details in a response to UI.
5. User able to login and can utilize the UI functionality to post the tweet (in text-area-box) and to read the tweets if any.
6. User can logout and can re-login by using credentials.

**High level & Deployment Architecture diagram**



**Sequence Diagram**



**Swagger Documentation**

swagger: "2.0"

info:

title: Node APIs for calling Twitter handlers

version: 1.0.0

host: 34.121.158.218

basePath: /twitter

schemes:

- http

paths:

/login:

post:

summary: For login & requesting token.

responses:

200:

description: OK

/posttweet:

post:

summary: Post a tweet.

responses:

200:

description: OK

/saveaccesstokens:

get:

summary: Save token of authenticated user.

responses:

200:

description: OK

/gettweets:

get:

summary: Returns a list of tweets.

responses:

200:

description: OK