**Dazzle Digital**

# Google Sheets

We've used google-sheets+python+mongodb+nodejs+express combo as CMS for Intrigd app.

# Storyboard

1. **Server Setup**
   * **Installations**
     + **Python**
     + **Mongodb**
     + **NodeJS**
   * **Code Deployment**
     + **Git Setup**
     + **Running Services**
2. **Architecture**
3. **Code Explanation**
   * **Generic Functions**
   * **Script wise Explanation**
4. **Troubleshooting guide**
5. **Storyboard**
   * User creates content like adding topics, articles, images and logos in google sheets. Data gets synced to database, which is further consumed by middleware api.

# Server Setup Installations

* + Create an aws EC2 instance and make an ssh connection, with following command on Terminal.
    - ssh -i /path/to/pemfile.pem [ubuntu@3.7.45.91](mailto:ubuntu@3.7.45.91)

# Python

* + $ sudo apt-get update
  + $ sudo apt-get install python3.6
  + $ sudo apt install python3-pip

# Node

* + $ sudo apt-get update
  + $ sudo apt-get install nodejs
  + $ sudo apt install express
  + Or install only (npm i) --- only one command can install all dependencies to our directory.

# MongoDB

* + wget -qO [- https://www.](http://www.mongodb.org/static/pgp/server-4.2.asc)mongodb.[org/static/pgp/server-4.2.asc](http://www.mongodb.org/static/pgp/server-4.2.asc) | sudo apt-key add -
  + sudo apt-get install gnupg
  + wget -qO [- https://www.](http://www.mongodb.org/static/pgp/server-4.2.asc)mongodb.[org/static/pgp/server-4.2.asc](http://www.mongodb.org/static/pgp/server-4.2.asc) | sudo apt-key add -
  + echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu bionic/mongodb-org/4.2 multiverse" | sudo tee

/etc/apt/sources.list.d/mongodb-org-4.2.list

* + sudo apt-get update
  + lsb\_release -a
  + sudo apt-get install -y mongodb-org
  + echo "mongodb-org hold" | sudo dpkg –set-selections
  + echo "mongodb-org-server hold" | sudo dpkg –set-selections
  + echo "mongodb-org-shell hold" | sudo dpkg –set-selections
  + echo "mongodb-org-mongos hold" | sudo dpkg –set-selections
  + echo "mongodb-org-tools hold" | sudo dpkg --set-selections

# GIT Setup

* + Code is available on git, we should make a git pull from master banch
  + Navigate to code directory where you want to store your code
    - cd /path/to/dir
    - create an ssh keygen
      * ssh-keygen -t rsa -b 2048 -C "[email@example.com"](mailto:email@example.com)
      * ssh-keygen -o -f ~/.ssh/id\_rsa
      * xclip -sel clip < ~/.ssh/id\_rsa
      * Navigate to https://gitlab.com and sign in.
      * Select your avatar in the upper right corner, and click Settings
      * Click SSH Keys.
      * Paste the public key that you copied into the Key text box.
      * Make sure your key includes a descriptive name in the Title text box, such as Work Laptop or Home Workstation.
      * Include an (optional) expiry date for the key under “Expires at” section. (Introduced in GitLab 12.9.)
      * Click the Add key button.
      * git config --global user.name "Username"
      * git config --global user.email "[email@intrigd.co"](mailto:email@intrigd.co)
      * git clone [git@gitlab.com](mailto:git@gitlab.com):Intrigd.Vishesh/google\_sheets.git

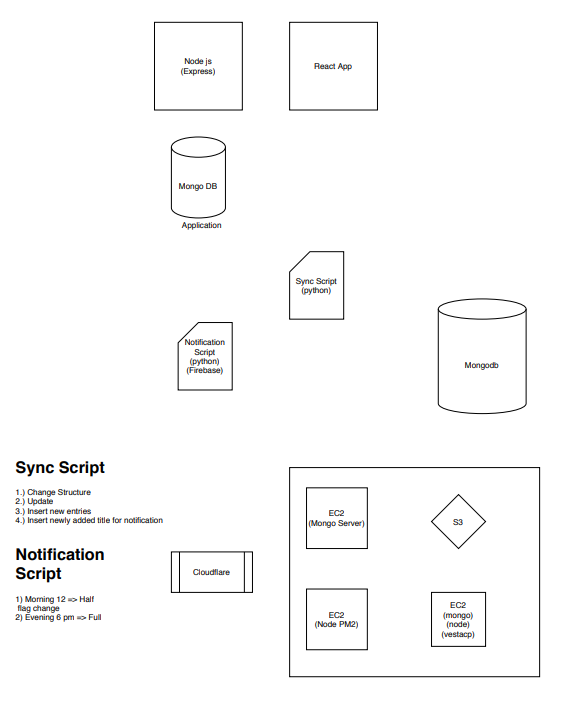
Install all the python dependencies with the following command

* + - pip3 install -r requirements.txt

# Running Services

* Starting MongoD
  + sudo systemctl start mongod
* Starting Mongo Client (Not needed though – Only for testing purposes)
  + mongo
* Navigate to git directory
  + cd google\_sheets
* Starting Rest API Server
  + python3 scripts\_api.py

Architecture Flow Diagram



# Code Explanation Generic Functions

These are the reusable common functions used across all the scripts.

def load\_env(var\_name = None):

"""

loads environmental variable, var\_name -> key

:param var\_name: DEV\_LOCAL/DEVELOPMENT/PRODUCTION

:return: current environment declared at OS level """

def load\_configs(env):

"""

loads config file content into memory for further processing

:param env: Environment variable

:return: file content in JSON Dict format """

def postCardOnTeams(title, dict = {}, abort\_module=False): """

Posts content on teams in card format

:param title: Title in Header format

:param dict: each key-value pair gets printed in each line

:param abort\_module: script stops immediately if set to true

:return: nothing """

def get\_live\_sheetdata(SCOPES, SAMPLE\_SPREADSHEET\_ID, SAMPLE\_RANGE\_NAME): """

reads google sheet and loads content in the list of dictionaries format

:param SCOPES: Read/Write

:param SAMPLE\_SPREADSHEET\_ID: ID extracted from sheet url

:param SAMPLE\_RANGE\_NAME: number of rows to be taken into account

:return:

"""

def Create\_Service(client\_secret\_file, api\_service\_name, api\_version, \*scopes): """

function for updating google sheet

pls refer python google api docs for more information, picked up from google docs """

def build\_dict\_list(values):

"""

Transforms list of lists to list of dictionaries

:param values: list of lists

:return: list of dictionaries """

def check\_for\_blank(input\_dictionary={}):

"""

checks if entire row is empty how it works -

- basically checks each of value in dictionary is empty, if all are empty, returns true

:param input\_dictionary: input row of google sheet or

:return: True/False """

def getIndianTime():

"""

:return: current indian time """

def write\_spread\_sheet(res\_list, SAMPLE\_SPREADSHEET\_ID, SAMPLE\_RANGE\_NAME): """

:param res\_list: list of dictionaries

:param SAMPLE\_SPREADSHEET\_ID: sheet id

:param SAMPLE\_RANGE\_NAME: range

:return:

"""

def get\_data\_from\_mongo():

"""

loads data from mongo

:return: list of dictionaries """

def upload\_files(files, path, bucket):

"""

uploads processed files from local machine to AWS S3 bucket

:param files: list of files needs to be pushed to S3

:param path: file path in S3 bucket

:param bucket: Name of the bucket which comes from config file

:return: uploaded files """

def resize(src, dest, width, height): """

resize a single image

:param src: source image path

:param dest: destination image path

:param width: width of resized image

:param height: height of resized image

:return: resized image in dest directory """

def get\_existing\_s3\_objects(prefix, bucket):

"""

retieves file list from a specified S3 bucket

:param prefix: file path inside bucket

:param bucket: bucket name

:return: file list in S3 bucket """

# Troubleshooting guide

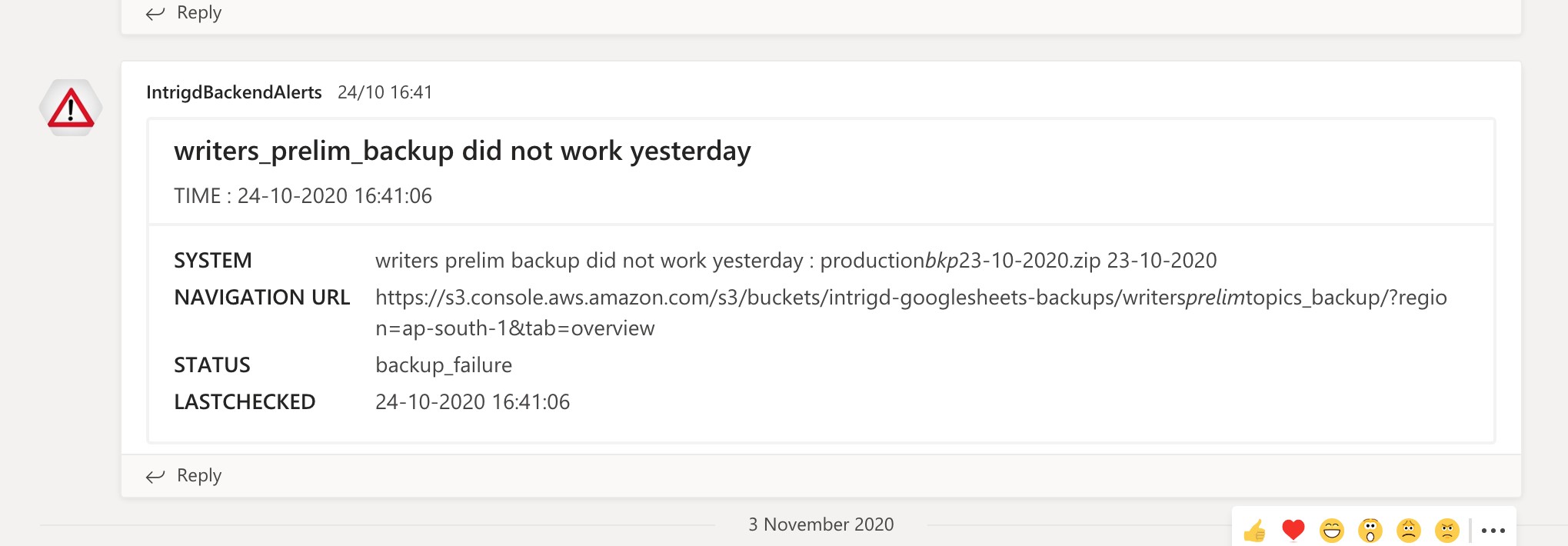
We've setup alerting system to report backend errors on microsoft teams, here are the list of alerts along with remedies

|  |  |
| --- | --- |
| Alert | Error in reading env variable |
| Description | this alert is thrown when env variable is not set on machine |
| Remedy | Please set env variable using cmd, “export INTRIGD\_GS\_ENV=PRODUCTION” |

|  |  |
| --- | --- |
| Alert | Error in reading s3 file, pls check bucket permissions.. |
| Description | This alert is throw when a file cant be uploaded to S3, due to bucket permissions |
| Remedy | Ask vishesh to give write permissions for s3\_user |

|  |  |
| --- | --- |
| Alert | Data Inconsistency Alert - TopicMapping |
| Description | This alert is thrown when topic is available in topic\_mapping and not in Images |
| Remedy | Alert vishesh about this inconsistency |

|  |  |
| --- | --- |
| Alert | Data Inconsistency Alert - Logos |
| Description | This alert is thrown when domain is not available in logo sheet for an article\_url in article sheet |
| Remedy | Please check for code logic |



|  |  |
| --- | --- |
| Alert | writers\_prelim\_backup did not work yesterday |
| Description | This alert is thrown when writers\_prelim\_backup fails to take backup |
| Remedy | Check if script is disabled on crontab and check bucket permissions |

|  |  |
| --- | --- |
| Alert | collection\_backup did not work yesterday |
| Description | This alert is thrown when collection\_backup doesn't work |
| Remedy | Check if script is disabled on crontab and check bucket permissions |

|  |  |
| --- | --- |
| Alert | Error in Db Connection |
| Description | This alert is thrown when database goes offline |
| Remedy | Starting mongod :  sudo mongod --auth --port 27017 --bind\_ip\_all --dbpath /data/db |

|  |  |
| --- | --- |
| Alert | Critical Infrastructure Alert – API Server |
| Description | This alert is thrown when API server goes offline |
| Remedy | Start API server  Python3 scripts\_api.py |

|  |  |
| --- | --- |
| Alert | missing config.json file |
| Description | This alert is thrown when config.json file is unavailable |
| Remedy | Please include config.json file inside project folder |

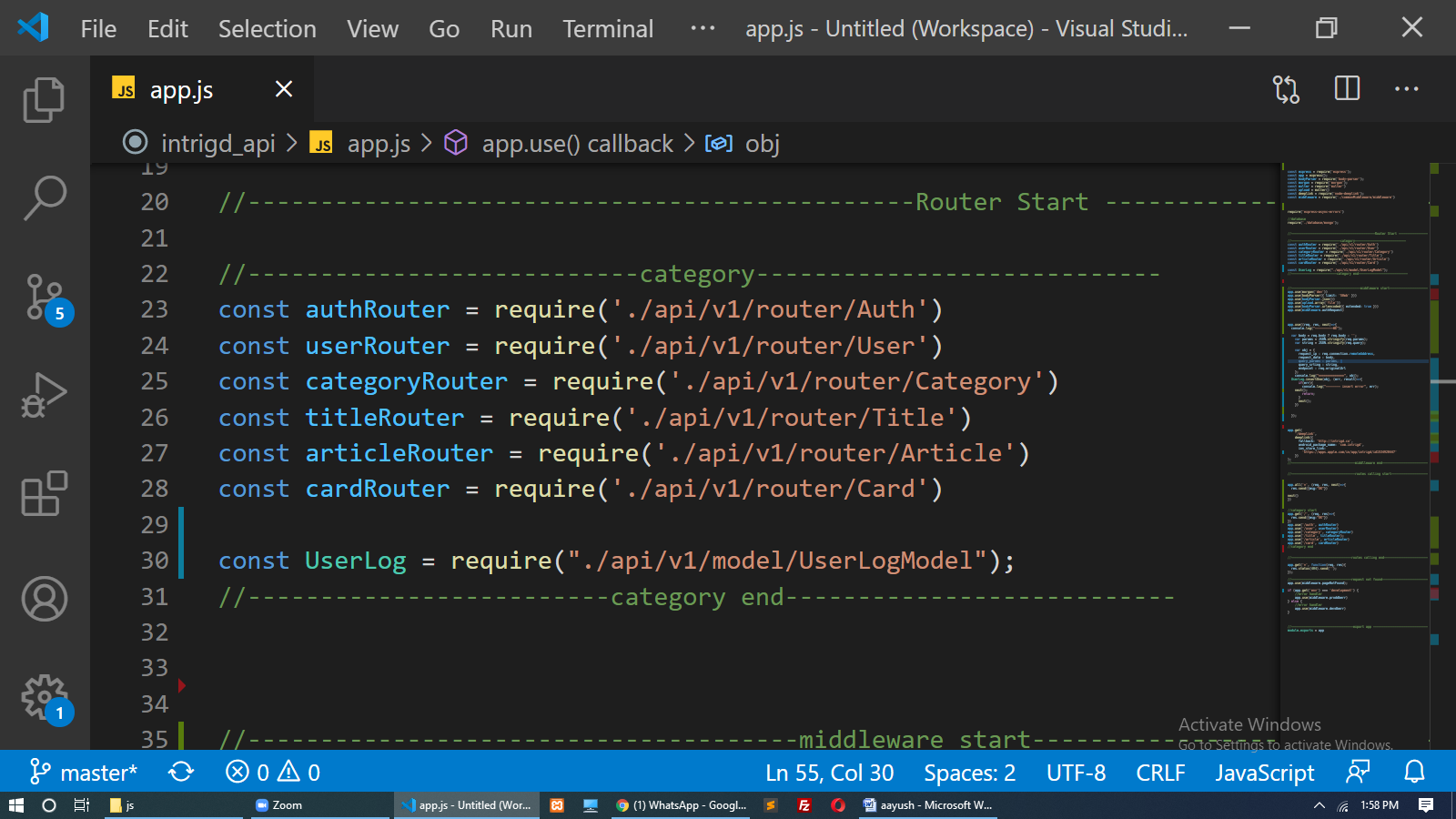
|  |  |
| --- | --- |
| Alert | No data in articlesapp-final sheet |
| Description | This alert is thrown when there is no data in input article sheet |
| Remedy | Check fileid in config.json file or alert vishesh |

- End –

Code Explanation :

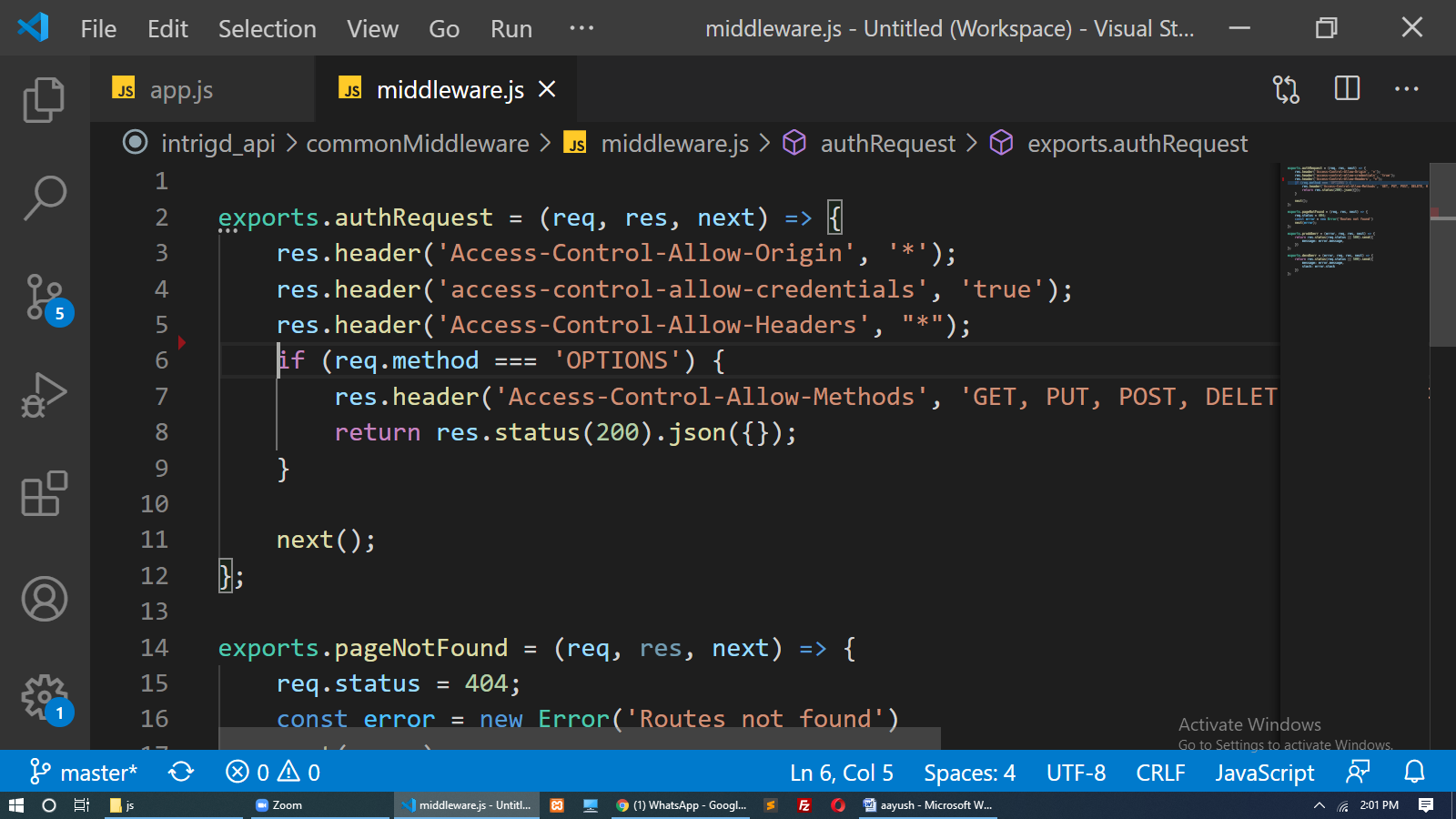
The coding flow as fallows :

1. We are using MVC pattern for the API server in Express Framework.
2. We create route for each and every api url.

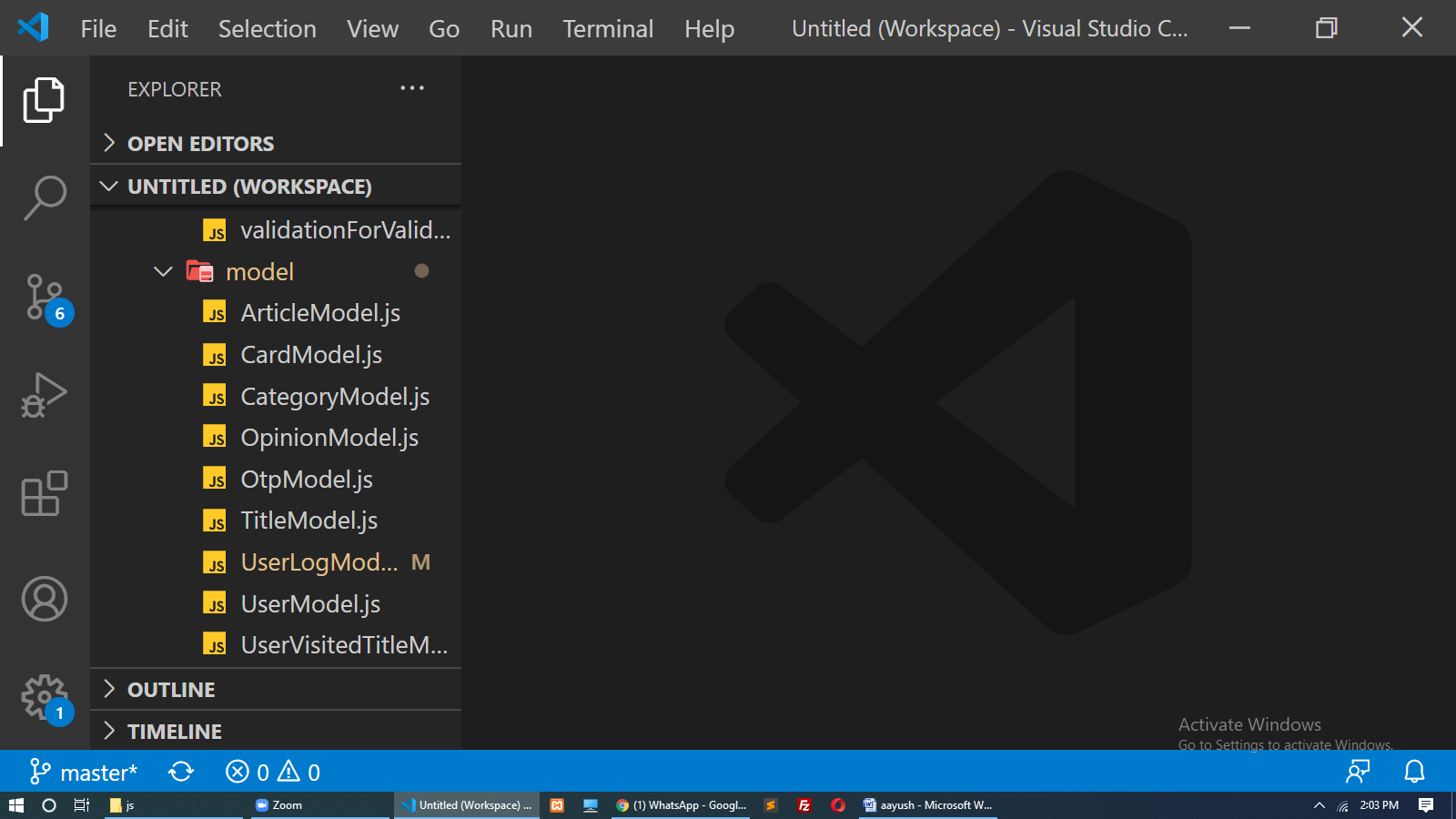


3. we create middleware for middleware logic and functions.

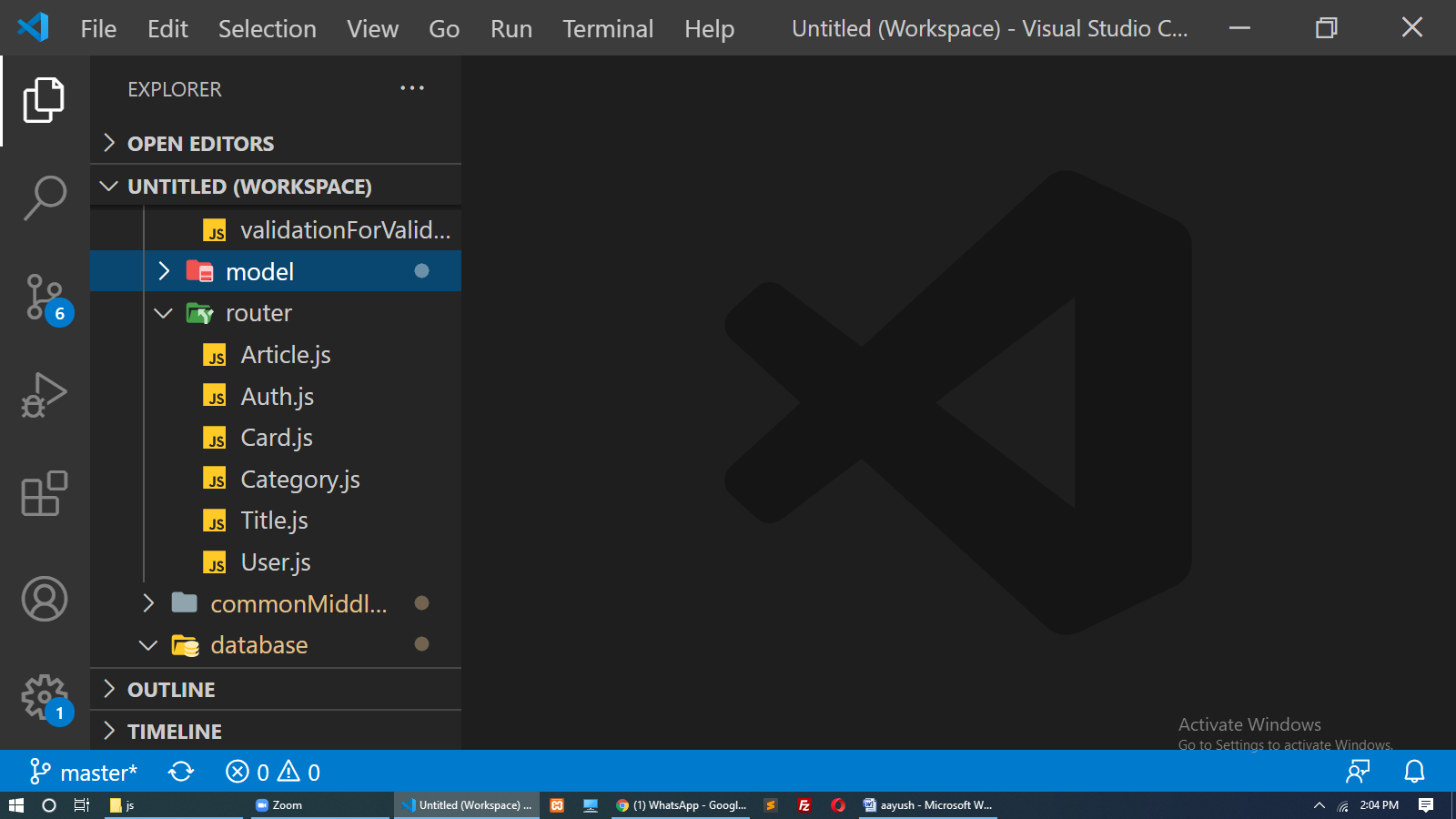
* Authrequest
* Paegnotfound
* Deverror



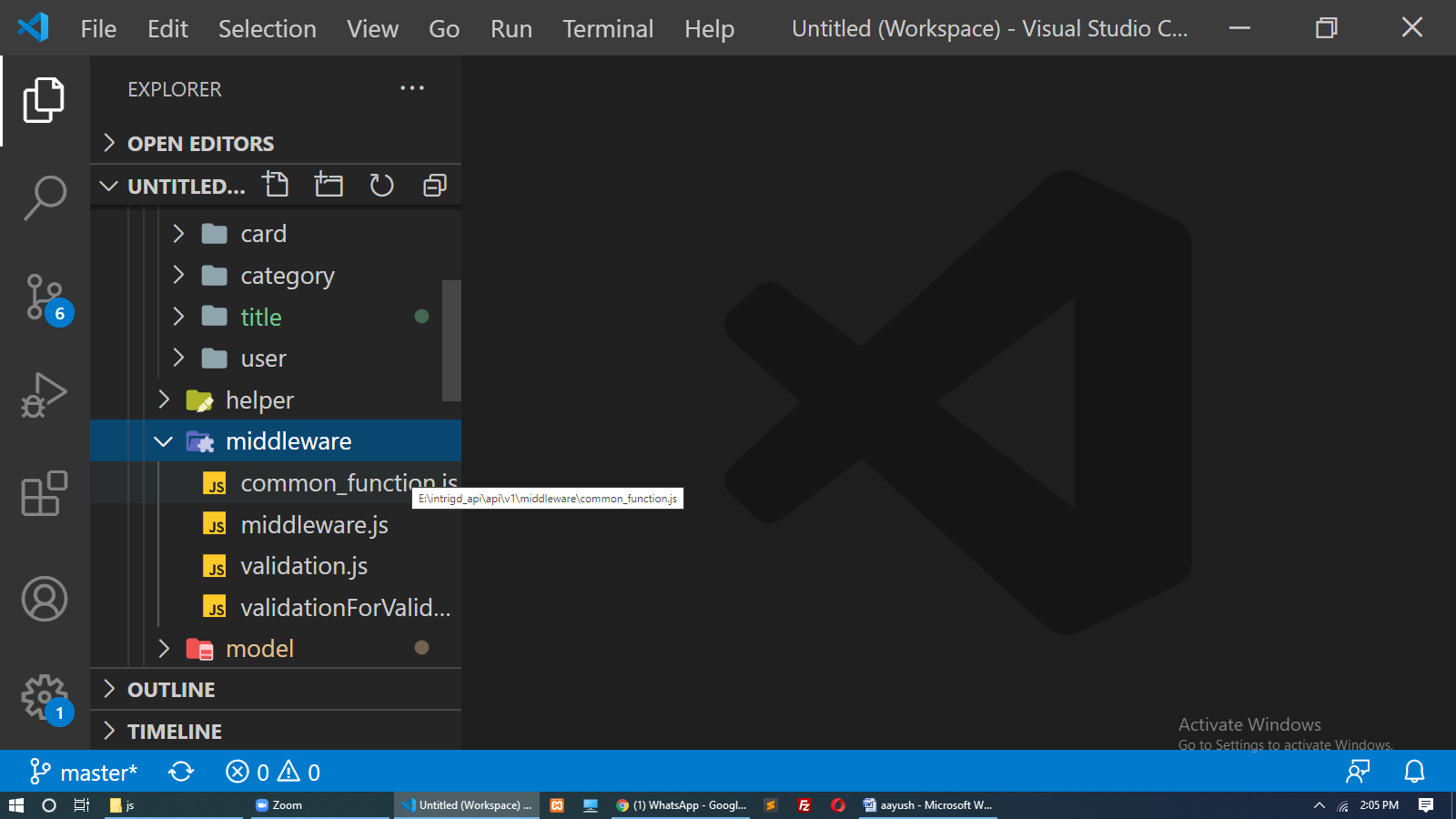
1. We create model for every Collections(tables) for Database my Mongoose ORM.
   * ArticleModel
   * CardModel
   * CategoryModel etc.



1. We also create separate routes for request.



1. We make some middleware for business logic



1. Because of we get data form another server(api) that’s way we make our own logic for make suitable JSON like in separate Controller like this.

