Version: 1.2.1

Created: 10/24/2019

LAST UPDATED: 10/29/2019

Car Maintenance Calendar

**CMC**

Presented by: Michael Mobilio

LUWAI WHEAGAR

MOHAMED SANGARE

KRUSHNA PATEL

Contents

[**README FILE** 2](#_Toc23258292)

[***DESCRIPTION*** 2](#_Toc23258293)

[***PREREQUISITES*** 2](#_Toc23258294)

[**Team Management tools** 2](#_Toc23258295)

[**Require Softwares** 2](#_Toc23258296)

[**Require Operating System** 2](#_Toc23258297)

[**Require Physical Network** 2](#_Toc23258298)

[**Require APIs’** 2](#_Toc23258299)

[**Project Deliverables** 2](#_Toc23258300)

[***INSTALLATION SETUP*** 3](#_Toc23258301)

[***USER GUIDE*** 4](#_Toc23258302)

[**CODE DOCUMENTATION** 8](#_Toc23258303)

[***Project Tree*** 8](#_Toc23258304)

[**TEAM 2 INFORMATION** 12](#_Toc23258305)

[***Rolls*** 12](#_Toc23258306)

[***Responsibilities*** 12](#_Toc23258307)

[***Content*** 12](#_Toc23258308)

[**CITATION** 13](#_Toc23258309)

# **README FILE**

## ***DESCRIPTION***

Our project is on "Car Maintenance Calendar." CMC is created for the user to get their next schedule reminder of maintenance services for their cars. For instance, CMC will tell them when their next oil change, oil filter, air filter, fuel filter replacement is going to be, and so forth. Also, there will be a YouTube playlist provided according to their maintenance services for them to see how to fix of their own if they desire.

## ***PREREQUISITES***

### **Team Management tools**

1. Trello ([**click here**](https://trello.com/?&aceid=&adposition=1t3&adgroup=77960762125&campaign=1407850971&creative=389415002626&device=c&keyword=trello&matchtype=e&network=g&placement=&ds_kids=p48959053108&ds_e=GOOGLE&ds_eid=700000001557344&ds_e1=GOOGLE&gclid=CjwKCAjw3c_tBRA4EiwAICs8CgOHhOKXAC5wYaJQ8W0o4hHm89ySLckfomWFGqrt-Eg_eJ6UT5eVZRoCtt0QAvD_BwE&gclsrc=aw.ds))
2. Discord ([**click here**](https://discordapp.com/))
3. GitHub ([**click here**](https://github.com))

### **Require Softwares**

1. Virtual Box ([**click here**](https://www.virtualbox.org/wiki/Downloads))
2. RabbitMQ

### **Require Operating System**

Ubuntu 18.04.3 LTS ([**click here**](https://ubuntu.com/download/desktop))

### **Require Physical Network**

1. One Physical Router with four Ethernet ports
2. Four Ethernet cables

### **Require APIs’**

1. Google Calendar API ([**click here**](https://developers.google.com/calendar))
2. YouTube API ([**click here**](https://developers.google.com/youtube))
3. CarMD API ([**click here**](https://api.carmd.com/member/docs))

### **Project Deliverables**

1. Authentication
2. RMQ between VMs
3. Trello Populated and in use
4. Logging
5. 4 servers cluster with dedicated front-end, back-end, and DMZ servers
6. Functioning Web based front-end
7. Automatic Data Collection from third party source
8. A functioning communication layer between your servers
9. A user authentication system (login page)
10. Distributed Logging system (error, warning, logs)
11. Centralizing Logging
12. PM Documentation
13. Project Documentation (changelog, readme, topology, etc)
14. Security Features
15. Team Specific Features
16. Save my cars
17. Get scheduled maintenance calendar for each make model year
18. Schedule reminders based on weekly mileage and current mileage
19. Get alerts for recalls
20. YouTube search for Maintenance procedure
21. Generate YouTube playlist for my car and upcoming maintenance
22. Ec oauth

## ***INSTALLATION SETUP***

1. Install Virtual Box
2. Download Ubuntu image (while 1 & 2 downloading do 3 to 5)
3. Install & setup Trello
4. Install & setup Discord
5. Install & setup GitHub
6. Setup Ubuntu image on Virtual box ([**click here**](https://www.youtube.com/watch?v=QbmRXJJKsvs) for how)
7. Install RabbitMQ on Ubuntu Desktop

* Only one of the team members require to do this step whomever later going to handle role of RabbitMQ; however, it is good practice if all the team do this.
* Open terminal on Ubuntu and do the below commands:
  + $sudo apt-get update (enter)
  + $sudo apt-get upgrade (enter)
  + $sudo apt-get install erlang (enter)
  + $sudo apt-get install rabbitmq-server (enter)
  + $sudo systemctl enable rabbitmq-server (enter)
  + $sudo systemctl start rabbitmq-server (enter)
  + $sudo systemctl status rabbitmq-server (enter)
  + $sudo rabbitmq-plugins enable rabbitmq\_management (enter)
  + $sudo rabbitmq add\_user admin admin (enter)
  + $sudo rabbitmq set\_user\_tags administrator (enter)
  + $sudo rabbitmq set\_permissions -p / admin “ .\*” “ .\*” “ .\*” (enter)
  + Open web browser on Ubuntu and search (put your ubuntu ipv4 add) 192.168.7.153: 15672 (enter)
  + Login with username admin and pass admin or guest (enter)
* Now time to test if everything is working so do “Hello World”
* Open terminal on Ubuntu and do the below commands:
  + $sudo python3 -m pip install pika –upgrade (enter)
  + Now you create two .py files send.py and reciev.py ([**click here**](https://www.rabbitmq.com/tutorials/tutorial-one-python.html) for the code)
  + Open another terminal and run both files in separate terminal side by side
  + $python3 send.py (enter) (same goes for another file on other terminal)

1. Time to Install vagrant on your actual operating system (not on the virtual box operating system)

* Only for who are handling Backend, DMZ, and RabbitMQ rolls
  + Remain your virtual box (VB) open
  + Open command line on your operating system
  + go to your GitHub directory for example mine is “C:\Users\Krush\Documents\GitHub\it490-car-calendar>”
  + >vagrant up broker (enter) (to delete this generated VB broker enter this command >vagrant destroy broker)
  + >vagrant ssh broker (enter) (to get into the VB broker terminal)
* Only for who are handling Frontend roll
  + Remain your virtual box (VB) open
  + Open command line on your operating system
  + go to your GitHub directory for example mine is “C:\Users\Krush\Documents\GitHub\it490-car-calendar>”
  + >vagrant up web (enter) (to delete this generated VB web enter this command >vagrant destroy web)
  + >vagrant ssh web (enter) (to get into the VB web terminal)
* Only for who are handing Database roll
  + Remain your virtual box (VB) open
  + Open command line on your operating system
  + go to your GitHub directory for example mine is “C:\Users\Krush\Documents\GitHub\it490-car-calendar>”
  + >vagrant up web (enter) (to delete this generated VB db enter this command >vagrant destroy db)
  + >vagrant ssh db (enter) (to get into the VB db terminal)

## ***USER GUIDE***

1. User Registration page



1. User Registration Conformation



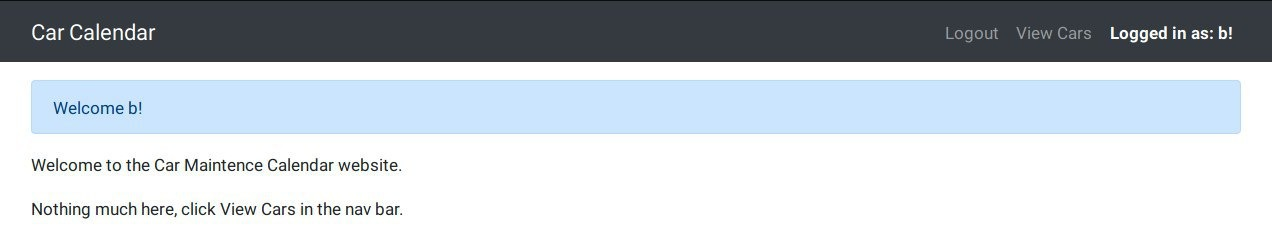
1. User Login page



1. When User Enters Invalid Credentials



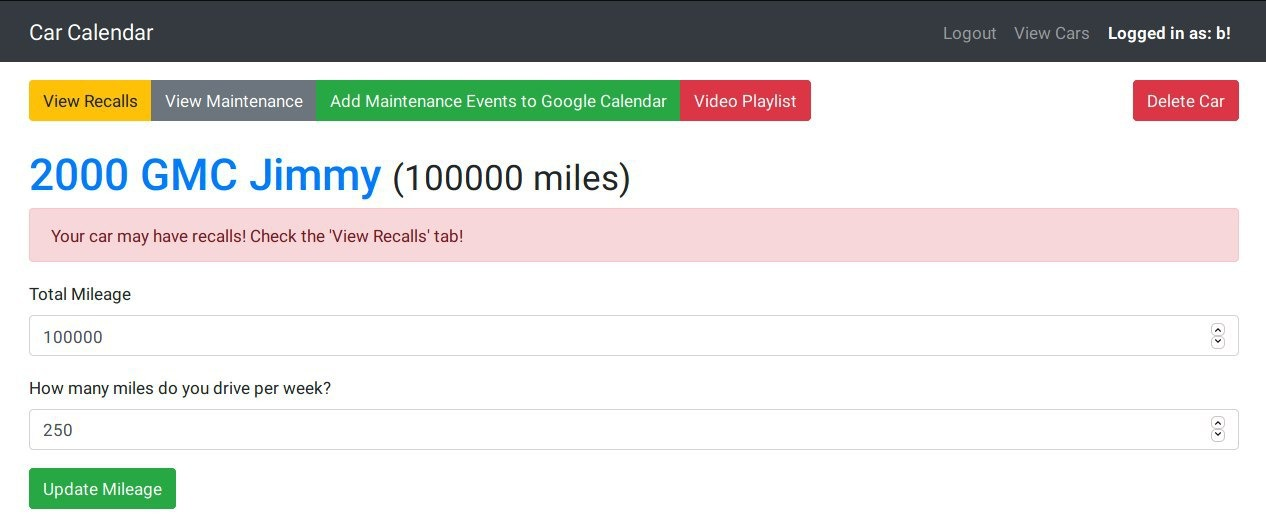
1. User Login Conformation



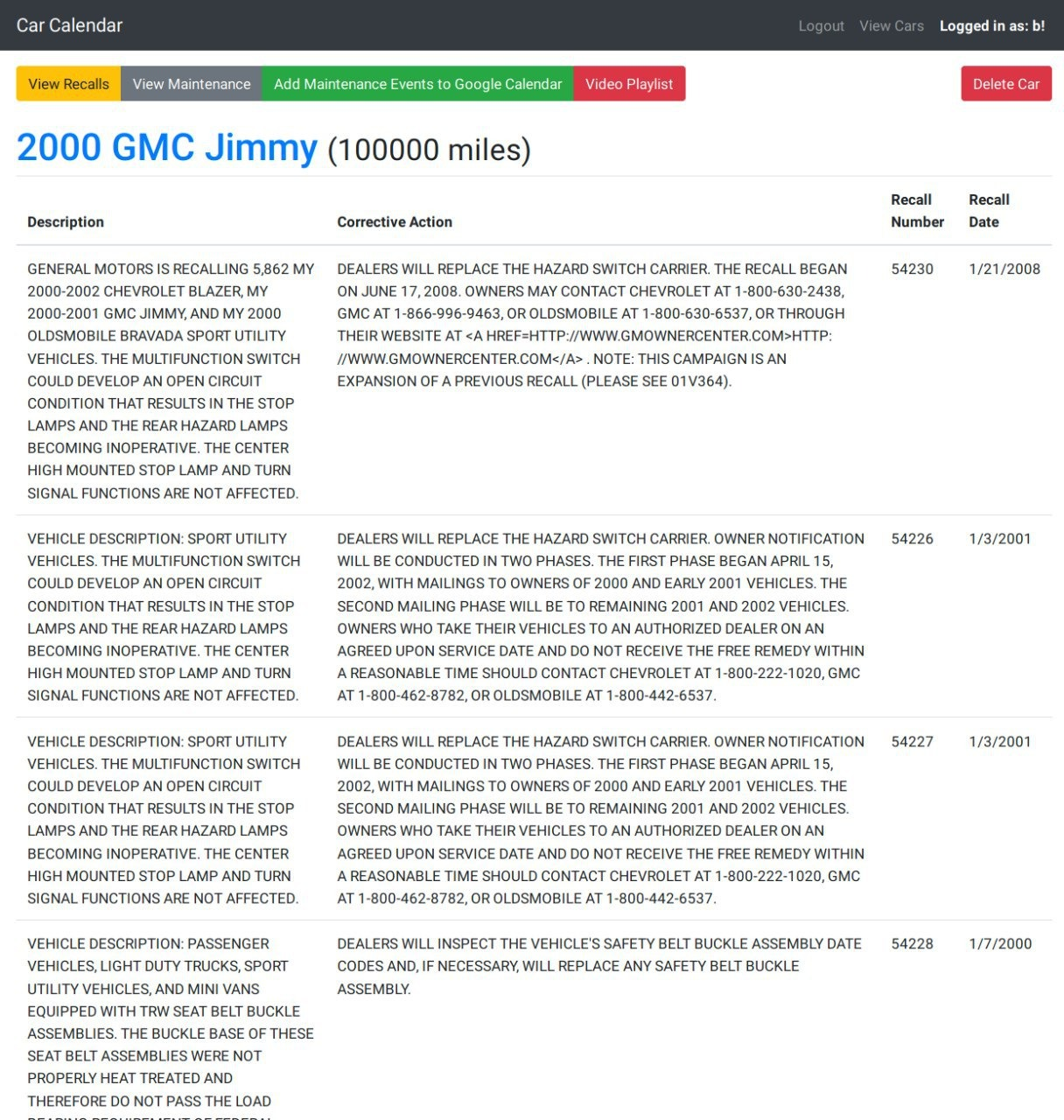
1. User b added is car



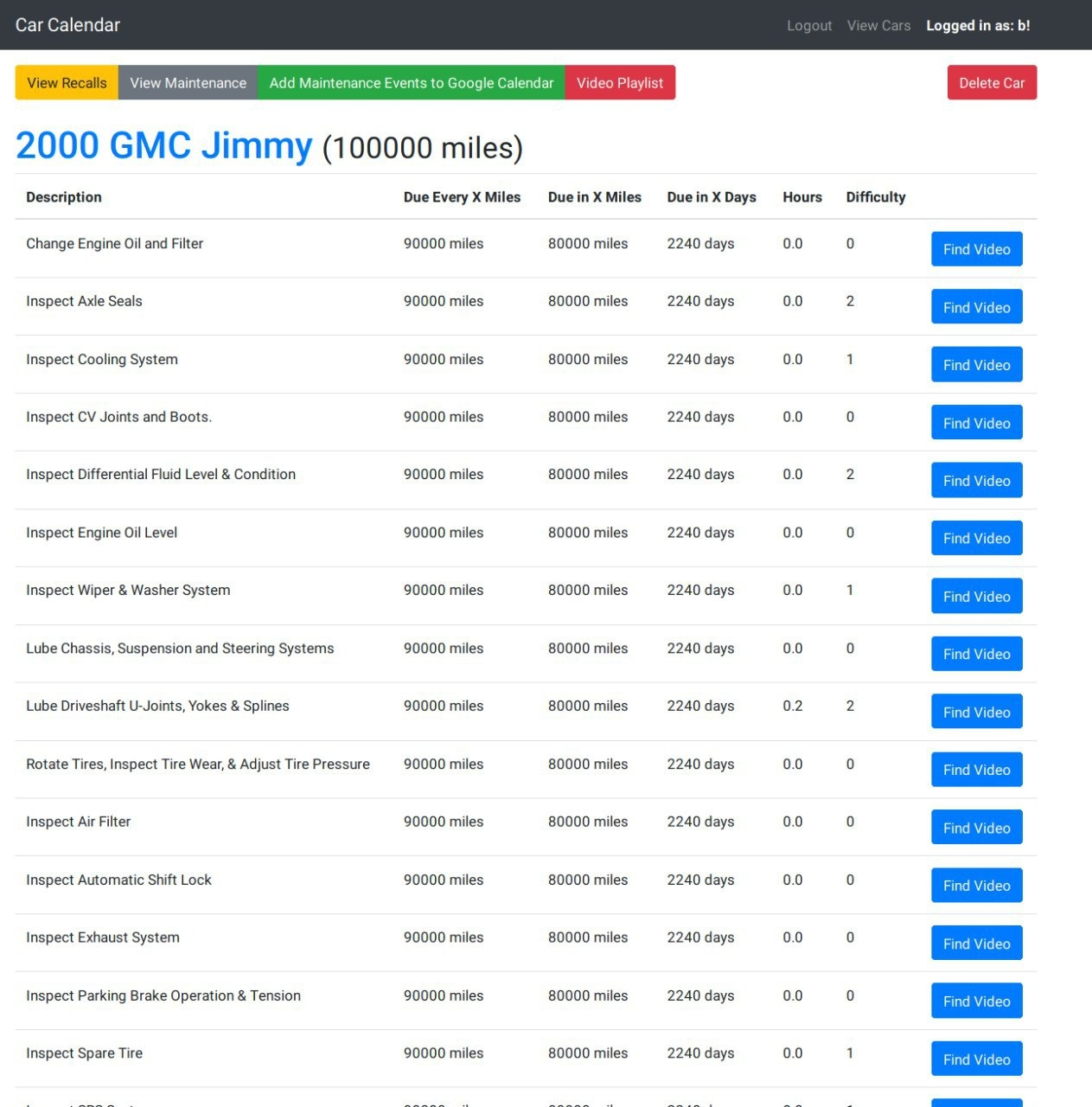
1. User b Car Maintenance Calendar



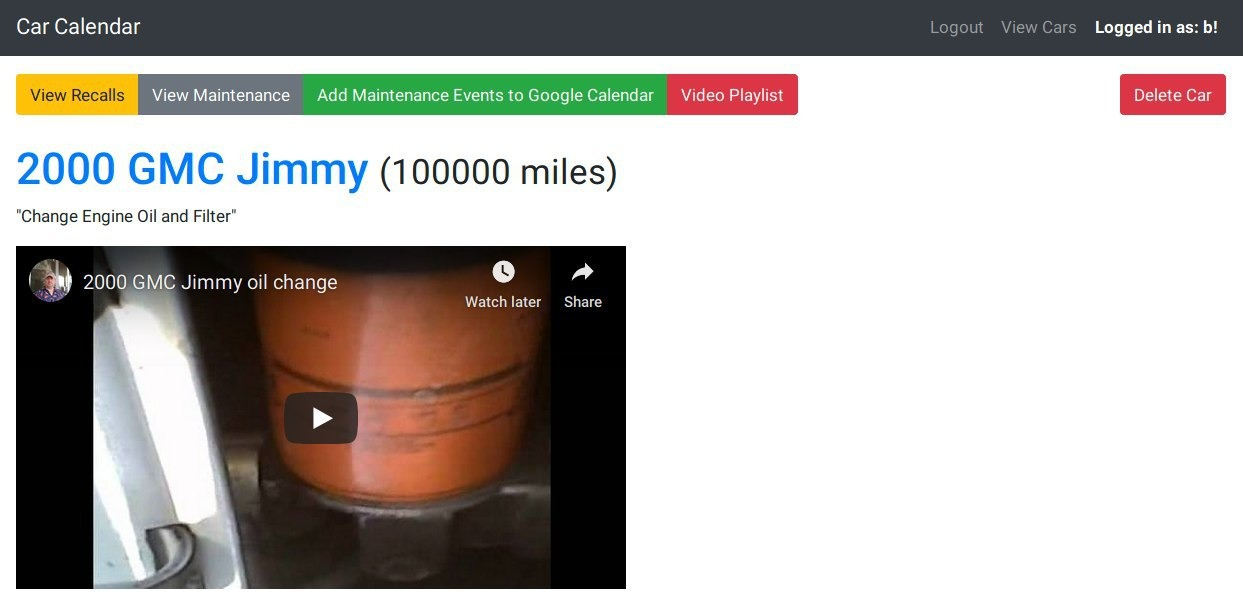
1. Description on user b car recalls



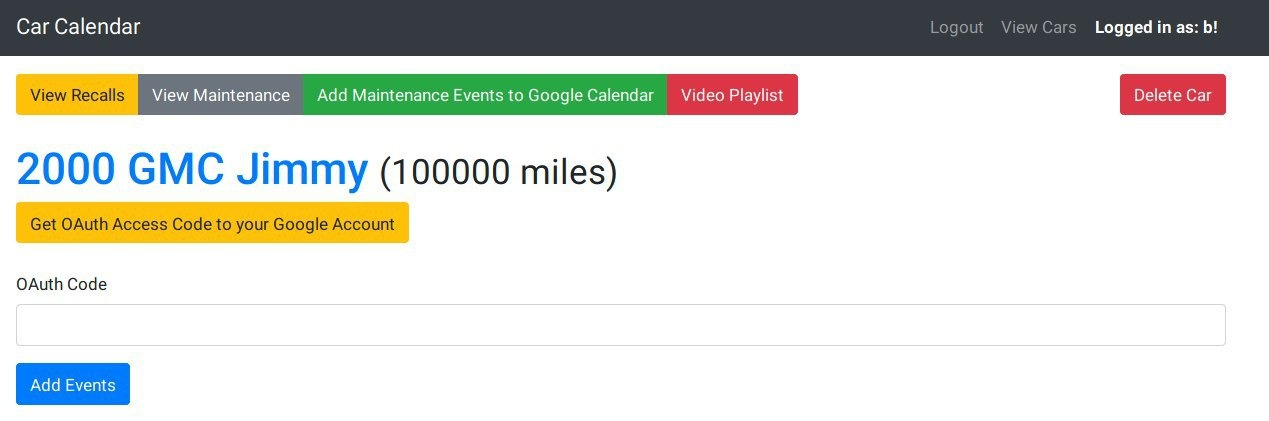
1. YouTube playlist related to the user b recall



1. YouTube video chosen from the playlist



1. Oauth access code to google account



1. User b car related videos



# **CODE DOCUMENTATION**

## ***Project Tree***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Directory | Main Folder | Sub Folder 1 | Sub Folder 2 | Sub Folder 3 | File |
| Text-color | ----------------- | **------------------** | **---------------** | **--------------** | **--------------** | **-------------** |

- **it490-car-calendar**

- **broker**

- **README.md** (Description of what the DMZ does)

- **provision-broker.sh** (Bash script to configure the Broker vm upon creation)

- **db** (The directory which contains all the files for the database vm)

- **data**

- **empty.txt** (the file was left empty intentionally)

- **logs** (The directory for all the locally stored log files)

- **empty.txt** (Test file test log file)

- **services** (The directory for the service bash script to start receiving/sending data)

- **auth-consumer.service** (The bash script which starts the consume\_auth.py process)

- **data-consumer.service** (The bash script which starts the consume\_data.py process)

- **log-consumer.service** (The bash script which starts the consume\_log.py process)

- **sql** (The directory for the mysql users and cars table)

- **0-users.sql** (mysql users table code where the value gets store)

- **1-cars.sql** (mysql cars table code where the value gets store)

- **src**

- **amqp** (The directory for the custom Python consumer classes)

- **\_\_pycache\_\_**

- **database**

- **\_\_pycache\_\_**

- **\_\_init\_\_.py** (Python script to initialize amqp package modules)

- **auth.py** (Python script to initialize authentication)

- **cars.py** (Python script to initialize users cars)

- **db.py** (Python script for database connections)

- **users.py** (Python script to initialize users)

- **example\_senders**

- **auth\_sender\_rpc.py** (Run the script and send authentication data)

- **dmz\_sender\_rpc.py** (Run the script and send dmz data)

- **log\_sender.py** (Run the script and send log data)

- **logger** (The directory for the custom Python logger classes)

- **\_\_pyache\_\_**

- **.env** (Bash script which sets the values for the environment variables)

- **\_\_init\_\_.py**

- **\_env** (Bash script which sets the values for the environment variables)

- **consume\_auth.py** (Run the script and consumes authentication data)

- **consume\_data.py** (Run the script and consumes data)

- **consume\_log.py** (Run the script and consumes log data)

- **requirements.txt** (Text file containing the dependencies to run the consum\_auth.py, consume\_data.py, and consume\_log.py python script)

- **motd** (run development server)

- **provision-db.sh** (Update package list and install python and amqp client)

- **README.md** (Information on what database do)

- **dmz** (The directory which contains all the files for the DMZ vm)

- **logs** (The directory for all the locally stored log files)

- **empty.txt** (Test file test log file)

- **services** (The directory for the service bash script to start receiving/sending data)

- **dmz-consumer.service** (The bash script which starts the consume\_dmz.py process)

- **src** (The directory for the source code used to create functionality)

- **.env** (Bash script which sets the values for the environment variables)

- **\_env** (Bash script which sets the values for the environment variables)

- **carmd.py** (Python script which retrieves the car’s maintenance/recall information from the CarMD API)

- **consume\_dmz.py** (The main python script which listens for actions from Rabbitmq, then runs functions based upon the received action)

- **email.py** (Python module script to send emails)

- **google\_calendar.py** (Python module script to gain access to Google Calendar, and schedule events)

- **requirements.txt** (Text file containing the dependencies to run the consum\_dmz.py python script)

- **youtube.py** (Python module script to retrieve YouTube playlist)

- **README.md** (Description of what the DMZ does)

- **provision-dmz.sh** (Bash script to configure the DMZ vm upon creation)

- **packages** (The directory for the custom Python classes)

- **amqp** (The directory for the custom Python consumer classes)

- **\_\_init\_\_.py** (Python script to initialize amqp package modules)

- **consumer.py** (Python class to abstract the process of consuming Rabbitmq data)

- **ez.py** (Python module to consume/produce to Rabbitmq)

- **producer.py** (Python class to abstract the process of producing Rabbitmq data)

- **setup.py** (Python script to setup package modules)

- **logger** (The directory for the custom Python logger classes)

- **\_\_init\_\_.py** (Python script to initialize amqp package modules)

- **logger.py** (Python class to generate and send log files)

- **logger\_tests.py** (Tester Python class for generating/sending log files)

- **setup.py** (Python script to setup package modules)

- **web** ( the web Directory contains all the frontend directories)

- **services** (contains the gunicorn.service file)

- **gunicorn.service** (this where the calendar gunicorn server is located)

- **src** (contains all the files and directories for Frontend)

- **public** (contains the empty.txt file)

- **empyt.txt** (the file was left empty intentionally)

- **static** (the static directory contains the sunflower.jpg image)

- **sunflower.jpg** (image file for testing)

- **templates** (contains HTML auth and images files)

- **auth** (contains login and registration files)

- **login.html** (Front page of the app)

- **register.html** (user registration page)

- **cars** (all frontend html files for users)

- **create.html** (contains fields names)

- **display.html** (view cars)

- **display\_add\_events.html** (Oauth access code to google account)

- **display\_maintenance.html** (hold Descriptions, Due dates for next oil change)

- **display\_recalls.html** (all the recall events table)

- **display\_video.html** (link to YouTube video)

- **display\_videos.html** (link to YouTube videos)

- **list.html** (add car, make, model, year)

- **list\_item.html** (display model, year)

- **base.html** (Maintenance calendar)

- **index.html** (main page and link pages)

- **utils** (a folder containing dueshit.py)

- **dueshit.py** (import produce\_dmz)

- **views** (contains auth cars and general python files)

- **auth.py** (json file)

- **cars.py** (contains more json file)

- **general.py** (routing to connect)

- **.env** (rabbitmq files)

- **\_env**

- **main.py** (load environment variables, create and setup flash)

- **producers.py** (generic produce helper os import)

- **requirements.txt** (conatains pika, flask, gunicorn, python dotenv)

- **run\_dev\_server** (contains flask export variables)

- **run\_prod\_server** (route to gunicorn)

- **README.md** (Car calendar Frontend information)

- **motd** (run development server)

- **nginx.conf** (information about server)

- **provision-web.sh** (Update package list and install python and amqp client)

- **.gitignore** (Tells which files for Git to ignore)

- **README.md** (Text file containing the quick start instruction guide)

- **Vagrantfile** (Configuration file used by Vagrant to create VMs)

# **TEAM 2 INFORMATION**

## ***Rolls***

|  |  |
| --- | --- |
| **NAME** | **ROLLS** |
| Michael Mobilio | SYSTEM ADMIN |
| Luwai Wheagar | BACKEND |
| Mohamed Sangare | FRONTEND |
| Krushna Patel | DATABASE |

## ***Responsibilities***

|  |  |
| --- | --- |
| **NAME** | **RESPONSIBILITIES** |
| Michael Mobilio | Broker Server, RabbitMQ, Consumer, Producer, GitHub |
| Luwai Wheagar | DMZ Server, CarMD API, YouTubr API, Google Calendar API |
| Mohamed Sangare | Web Server, Python FLASK Server |
| Krushna Patel | Database Server, MySQL, Documentation, Trello |

## ***Content***

|  |  |
| --- | --- |
| **NAME** | **EMAIL** |
| Michael Mobilio | [Mcm53@njit.edu](mailto:Mcm53@njit.edu) |
| Luwai Wheagar | [Lgw3@njit.edu](mailto:Lgw3@njit.edu) |
| Mohamed Sangare | [Ms2247@njit.edu](mailto:Ms2247@njit.edu) |
| Krushna Patel | [Kp592@njit.edu](mailto:Kp592@njit.edu) |

# **CITATION**

1. <https://trello.com/?&aceid=&adposition=1t3&adgroup=77960762125&campaign=1407850971&creative=389415002626&device=c&keyword=trello&matchtype=e&network=g&placement=&ds_kids=p48959053108&ds_e=GOOGLE&ds_eid=700000001557344&ds_e1=GOOGLE&gclid=CjwKCAjw3c_tBRA4EiwAICs8CgOHhOKXAC5wYaJQ8W0o4hHm89ySLckfomWFGqrt-Eg_eJ6UT5eVZRoCtt0QAvD_BwE&gclsrc=aw.ds>
2. <https://discordapp.com/>
3. <https://github.com>
4. <https://www.virtualbox.org/wiki/Downloads>
5. <https://ubuntu.com/download/desktop>
6. <https://developers.google.com/calendar>
7. <https://developers.google.com/youtube>
8. <https://api.carmd.com/member/docs>
9. <https://www.youtube.com/watch?v=QbmRXJJKsvs>
10. <https://www.rabbitmq.com/tutorials/tutorial-one-python.html>