

RAE 553 Week 16

Mahendra Moorthy Kesavan

May 2020

1 flask-RESTful application:

- **Introduction**

Flask:

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more frequently than the core Flask program.

Flask-RESTful:

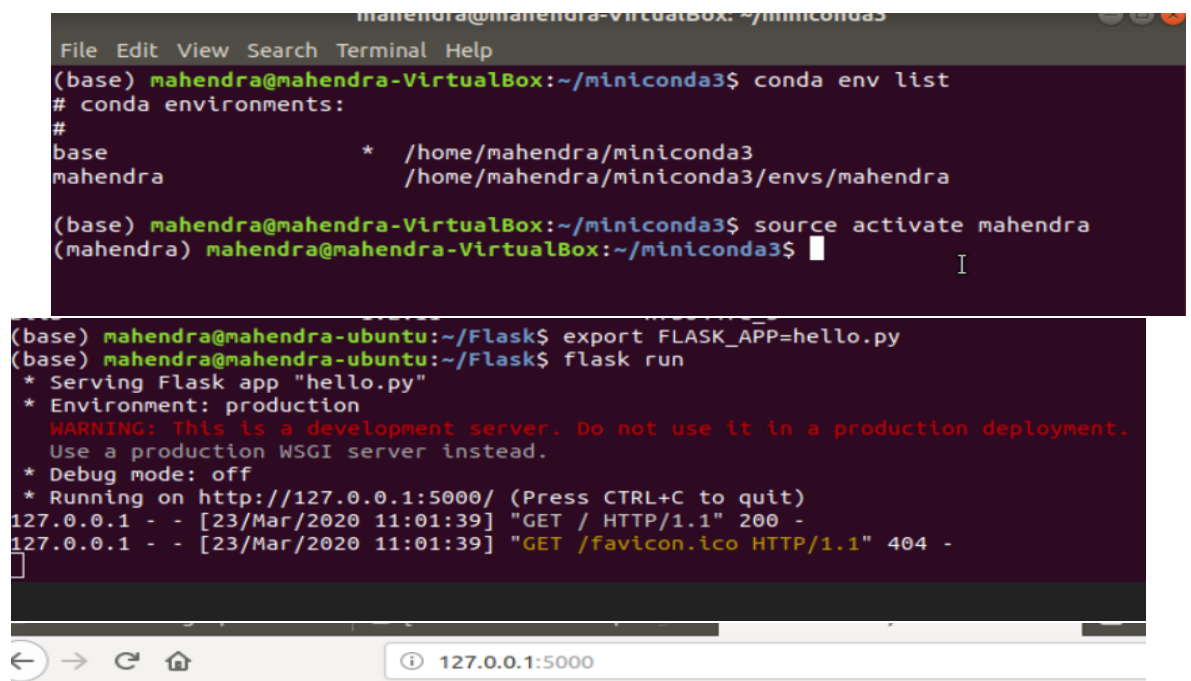
Flask-RESTful is an extension for Flask that adds support for quickly building REST APIs. It is a lightweight abstraction that works with your existing ORM/libraries. Flask-RESTful encourages best practices with minimal setup. know about the flask restful we need to install some of the softwares like linux operating system of mac os is need and Atom which is used to type the python program it can be install with the code of “sudo snap install atom –classic” and flask restful package with the code of “pip install flask-restful”. And finally Postman platform is used for API development, it is install with the help of this link <https://www.getpostman.com/>. By using these software we are going to create the REST API using flask based on flask application.

- **Methods:**

First step is to create the virtual environment using miniconda, before that we need to first download the miniconda to our operating system by using this link <https://docs.conda.io/en/latest/miniconda.html#linux-installers>. Then install it in the linux by the way of ubuntu terminal “ bash Miniconda3-latest-Linux-x86_64.sh” launch it in the folder where the miniconda3 in downloaded. Then create environment after the miniconda installation gets successful in the by using this code `\conda create – ntestenv(yourname)`. Check it whether you

are able to see the new environment by this code “conda env list”. Then we need to activate our environment by this code “source activate testenv”. Then we are directly getting inside to the Atom, where we are creating the new file called app.py. In this environment before we are typing the program and running it the flask and flask restful packages should be already installed in it. To check everything is correct by this command `export FLASK_APP = hello.py` run this and in next line “flask run” then enter, now you are able to see the “hello yourname” in browser. If you did this all the steps then you are able to see in your like the picture which displayed in the Result section.

- Result



The image shows a terminal window and a web browser. The terminal window is titled "mahendra@mahendra-VirtualBox: ~/miniconda3" and shows the following commands and output:

```
File Edit View Search Terminal Help
(base) mahendra@mahendra-VirtualBox:~/miniconda3$ conda env list
# conda environments:
#
base                *  /home/mahendra/miniconda3
mahendra             /home/mahendra/miniconda3/envs/mahendra

(base) mahendra@mahendra-VirtualBox:~/miniconda3$ source activate mahendra
(mahendra) mahendra@mahendra-VirtualBox:~/miniconda3$
```

The terminal window is then titled "mahendra@mahendra-ubuntu: ~/Flask" and shows the following commands and output:

```
(base) mahendra@mahendra-ubuntu:~/Flask$ export FLASK_APP=hello.py
(base) mahendra@mahendra-ubuntu:~/Flask$ flask run
* Serving Flask app "hello.py"
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [23/Mar/2020 11:01:39] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [23/Mar/2020 11:01:39] "GET /favicon.ico HTTP/1.1" 404 -
```

The web browser shows the URL "127.0.0.1:5000" and displays the message "Hello, MahendraMoorthy!"

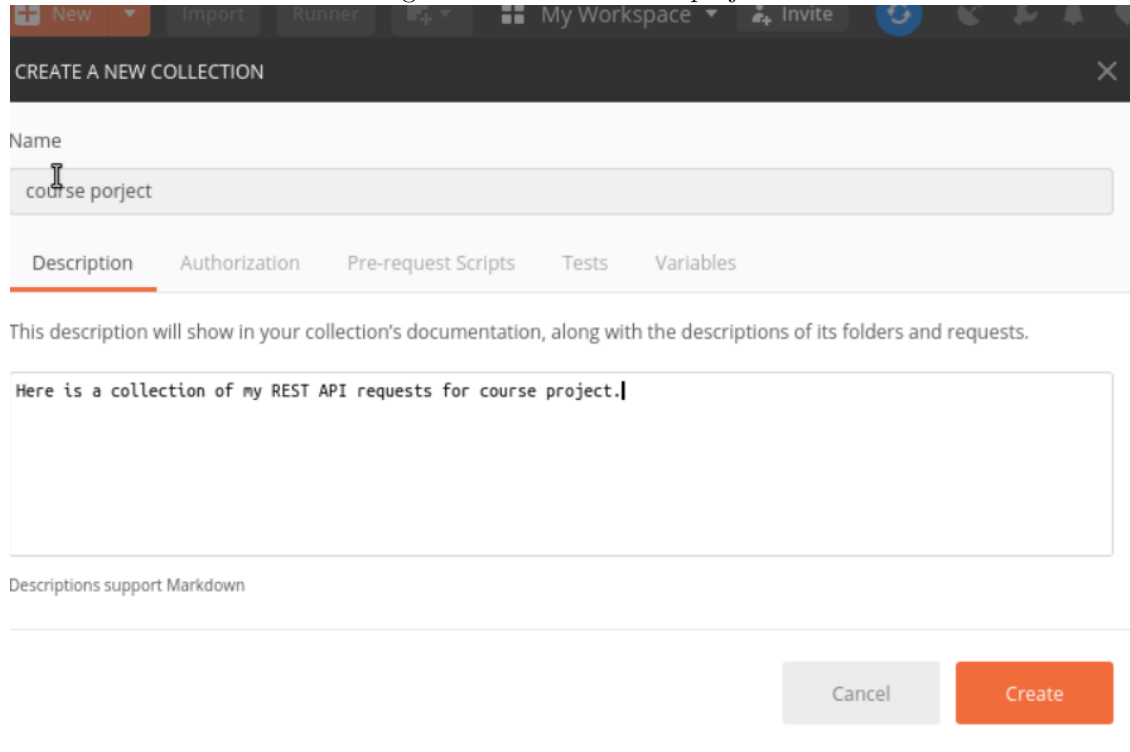
2 Training with API design

- **Introduction:**

In this part we are going to learn about the Postman API tool. How to create collections and requests in postman. Postman is an API(application programming interface) development tool which helps to build, test and modify APIs. ... It has the ability to make various types of HTTP requests(GET, POST, PUT, PATCH), saving environments for later use, converting the API to code for various languages(like JavaScript, Python).

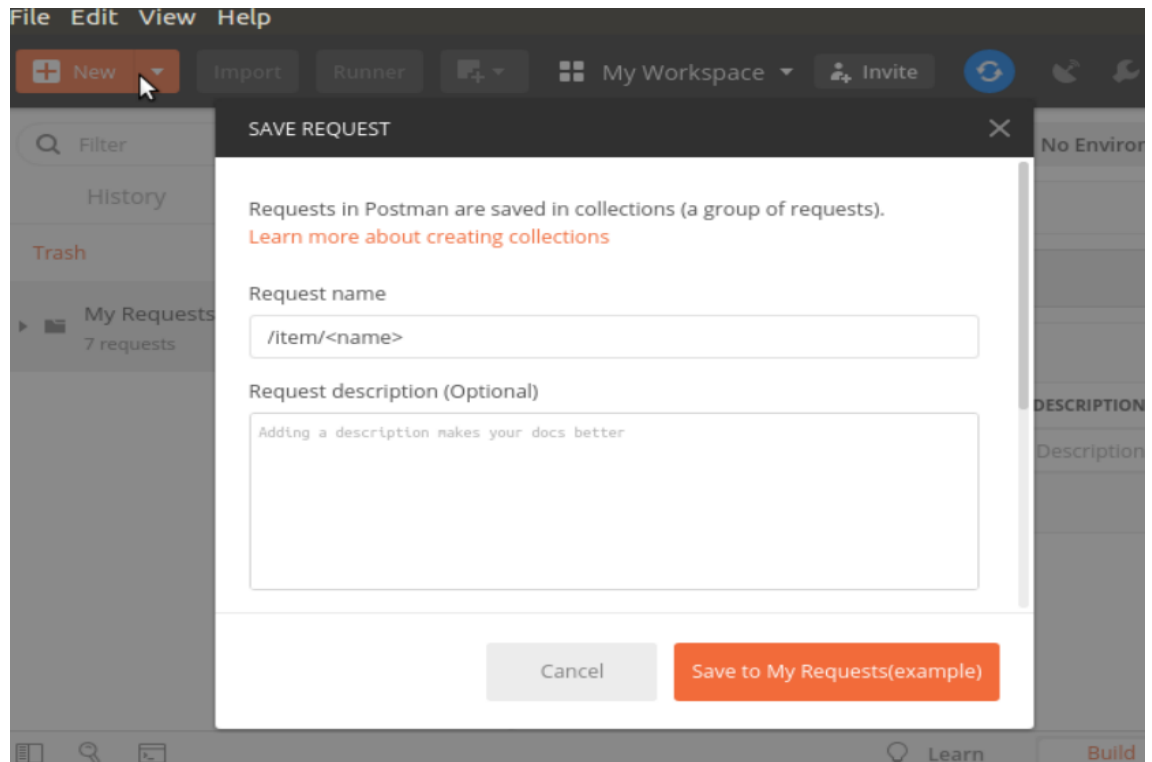
- **Methods:**

To create a new collection and give it a name as course project.

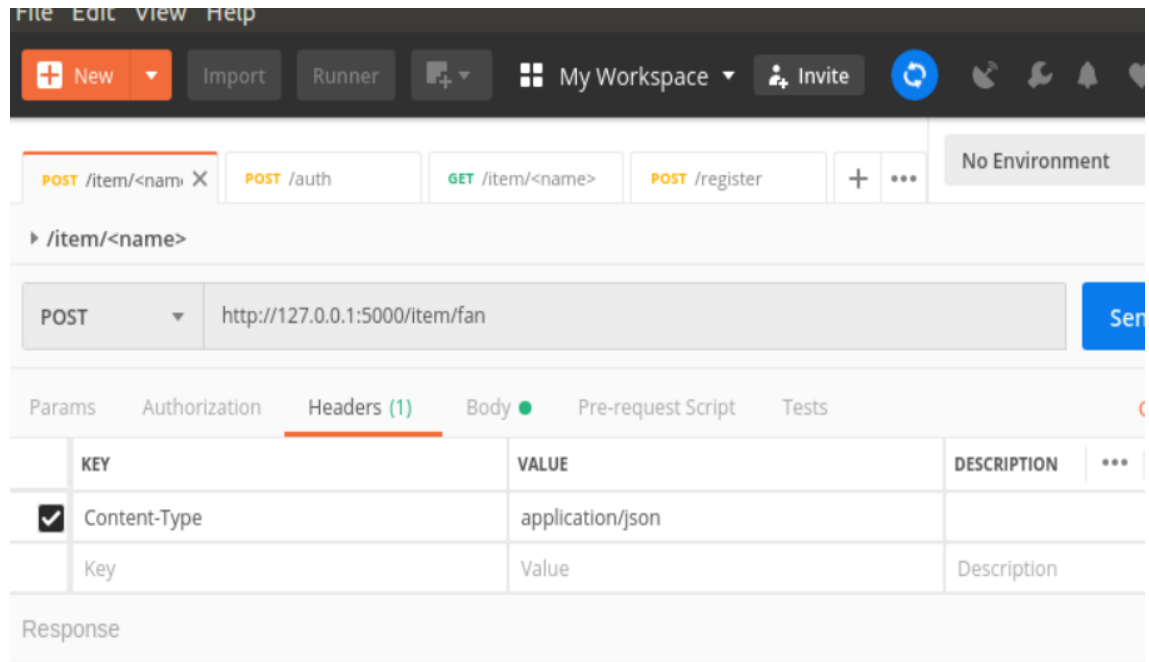


The screenshot shows the 'CREATE A NEW COLLECTION' dialog in Postman. The 'Name' field contains 'course project'. The 'Description' tab is active, displaying a text area with the text 'Here is a collection of my REST API requests for course project.' The 'Create' button is highlighted in orange, while the 'Cancel' button is grey.

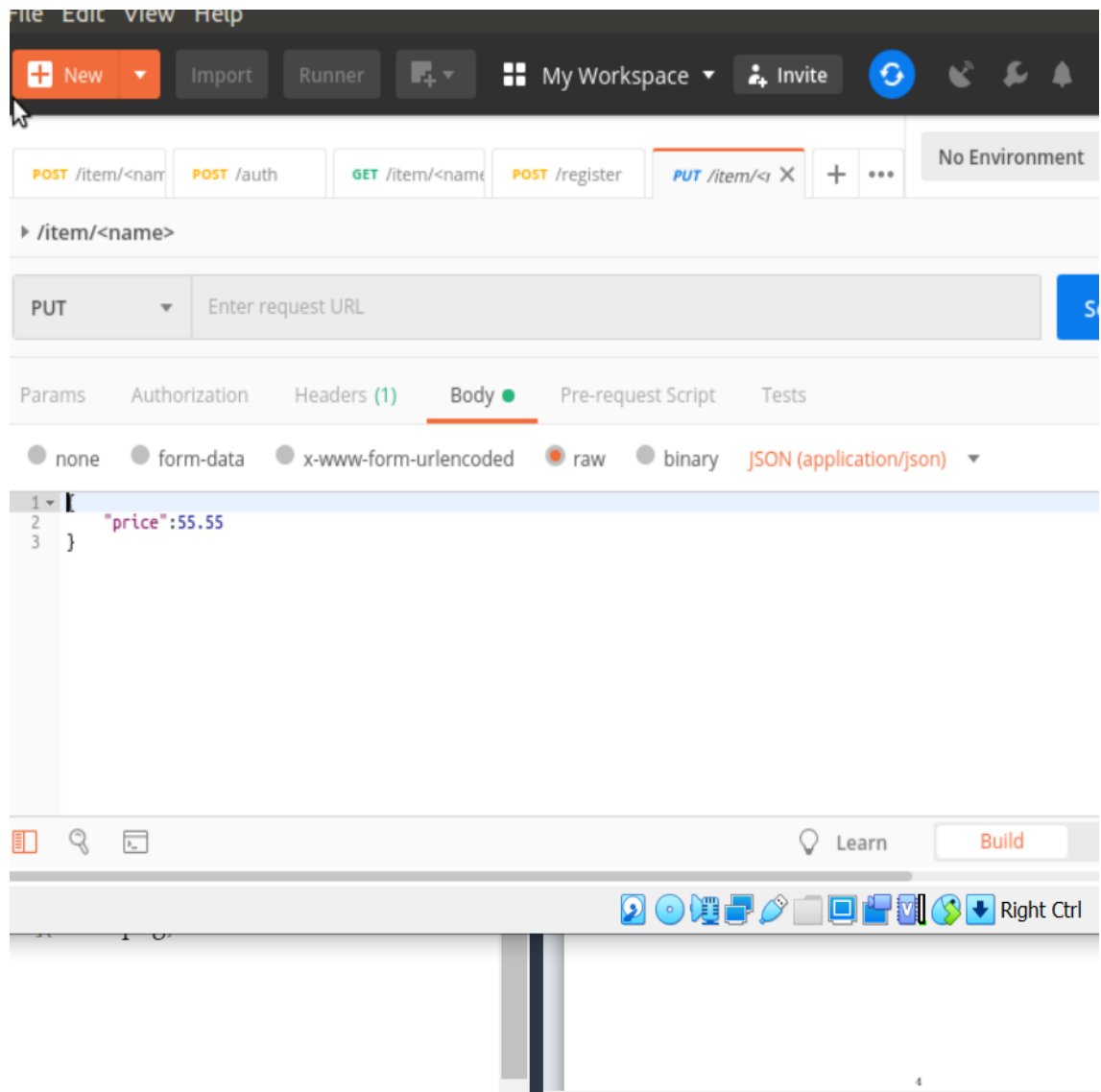
Next step is to create the new request in the collection for that you need to click the my request option(three dots) then click on the add request.



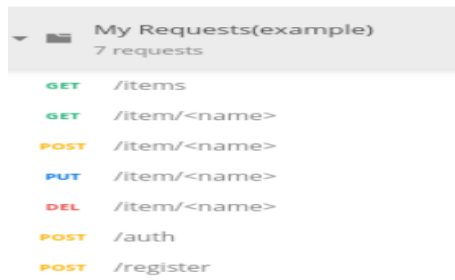
Create in this manner for four request which is **get**, **put**, **post**, **delete** after creating this, there are some changes in the headers section in the post and put section change the **KEY** to Content-Type and **VALUE** to applicaton/json.



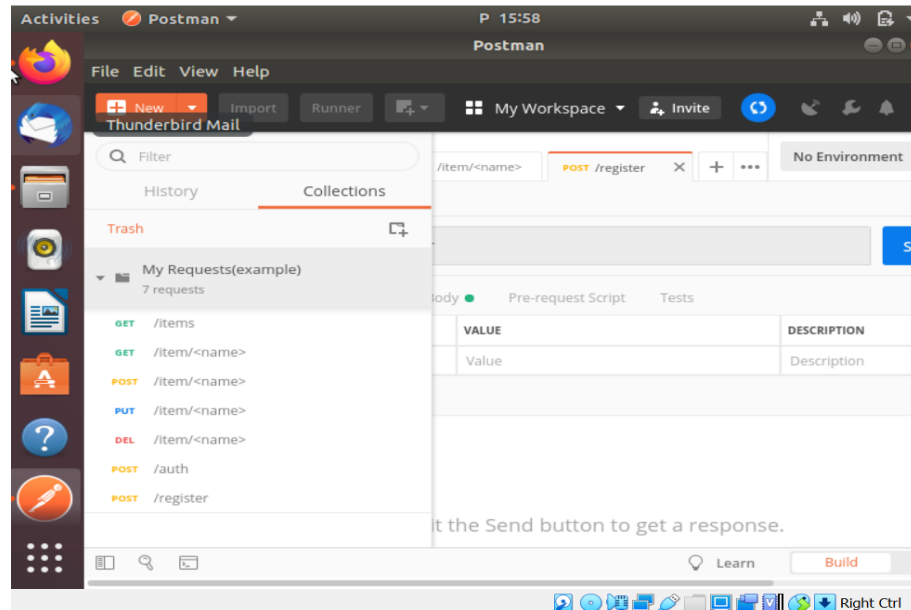
Now edit some price in the **Body**.



In final stage you will have total of 4 created items.



- Result:



3 Creating Resources

- Introduction:

Here we are going to learn about how we are going to retrieve items. We can making it accessible by name or id.

By the help of POST method. It is used to transfer or send data to the server, It will create a new item and if the item is already exists it will fail to create respectively.

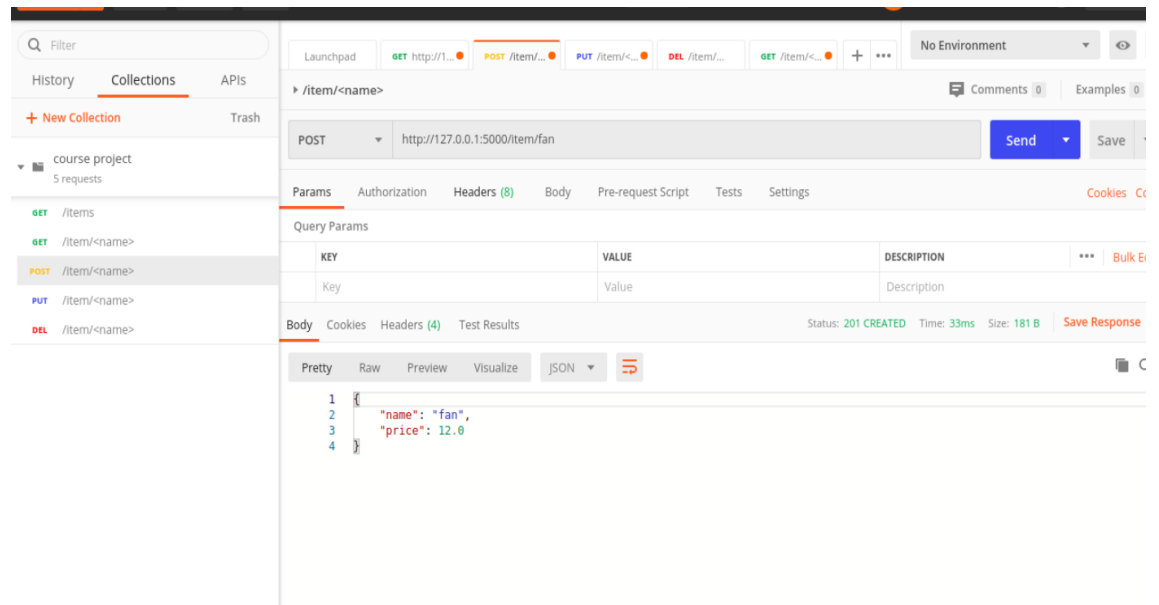
- Method:

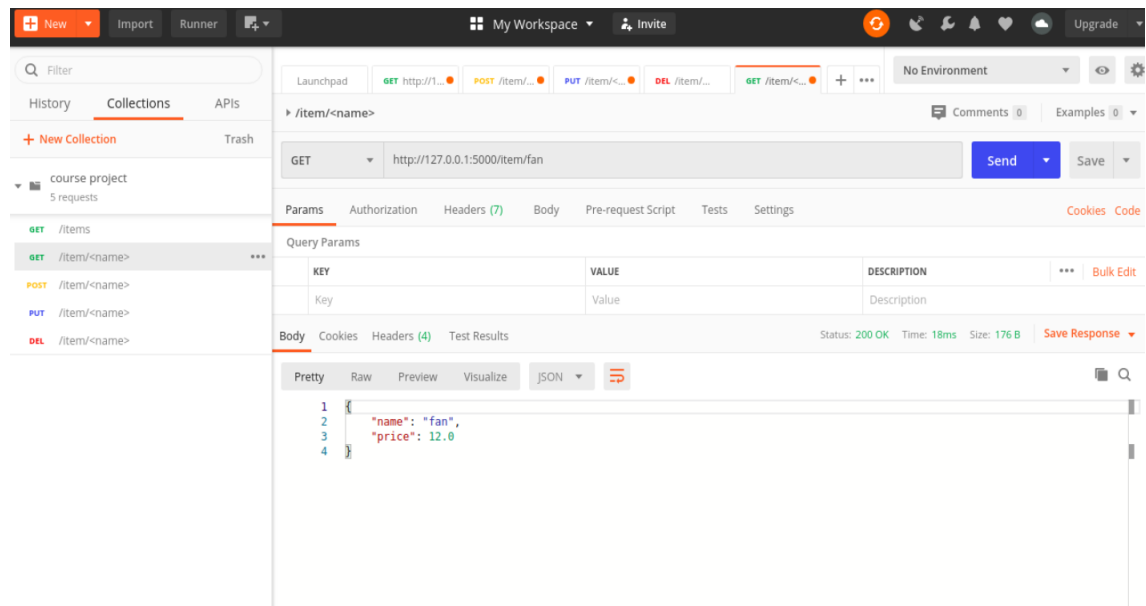
Now we need to create an item resource. Using the atom editor open the python app.py file. Here, we are going to just start only iwth GET and POST. There are some program code for items that has to enter the item in POST and retrieving the item in GET the coding are uploaded in the GIT repository, link is given at the end of this pdf.

After entering the coding in atom, run it in the terminal if it running then go to the Postman. Now its much more importance to the previously created items in the POST. What we entered in the previous section is going to display here after we are copy pasting the link which is displayed in the teminal and then enter SEND.

Now,it is final step to see it in the GET, what item we entered in the POST is that we a can see and we can recieve our item back with price in this place.

- Result:





4 ItemList

- Introduction:

A Postman Collection Item that holds your request definition, responses and other stuff. An Item essentially is a HTTP request definition along with the sample responses and test scripts clubbed together.

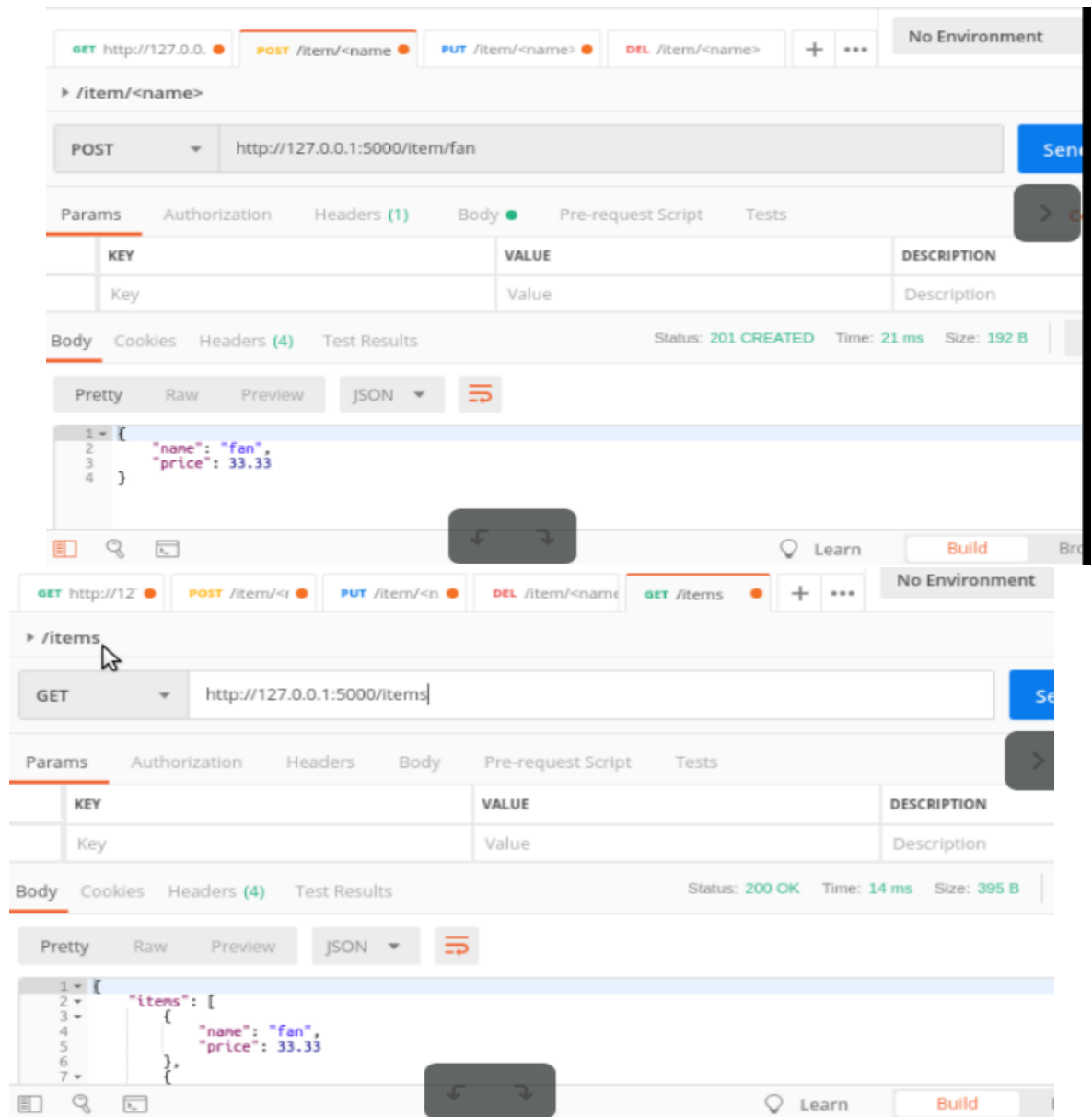
Here we are going to update our last section program. In previous program we added the price in body section but here we are going to add those segments in the python program.

- Methods:

Updating the coding from item to ItemList, and adding some lines like `api.addresource`.

After updating the coding in the atom we directly go terminal and run the file. Then we need to test this update in the postman. Here, if we click the SEND button, we are able to see it returns a name of our item and its price now it is like public source so we are moving to GET/items request. we obtain the list of our available items with the name and price.

- Result



5 Authentication and Logging:

- Introduction:

Here we are going to update our project like creating our ItemList using

JSON payload and designing security part like authentication and logging. APIs use authorization to ensure that client requests access data securely. This can involve authenticating the sender of a request and verifying that they have permission to access or manipulate the relevant data. If you're building an API, you can choose from a variety of auth models. If you're integrating a third-party API, the required authorization will be specified by the API provider.

You can pass auth details along with any request you send in Postman. Auth data can be included in the header, body, or as parameters to a request. If you enter your auth details in the Authorization tab, Postman will automatically populate the relevant parts of the request for your chosen auth type. You can use variables and collections to define authorization details more safely and efficiently, letting you reuse the same information in multiple places.

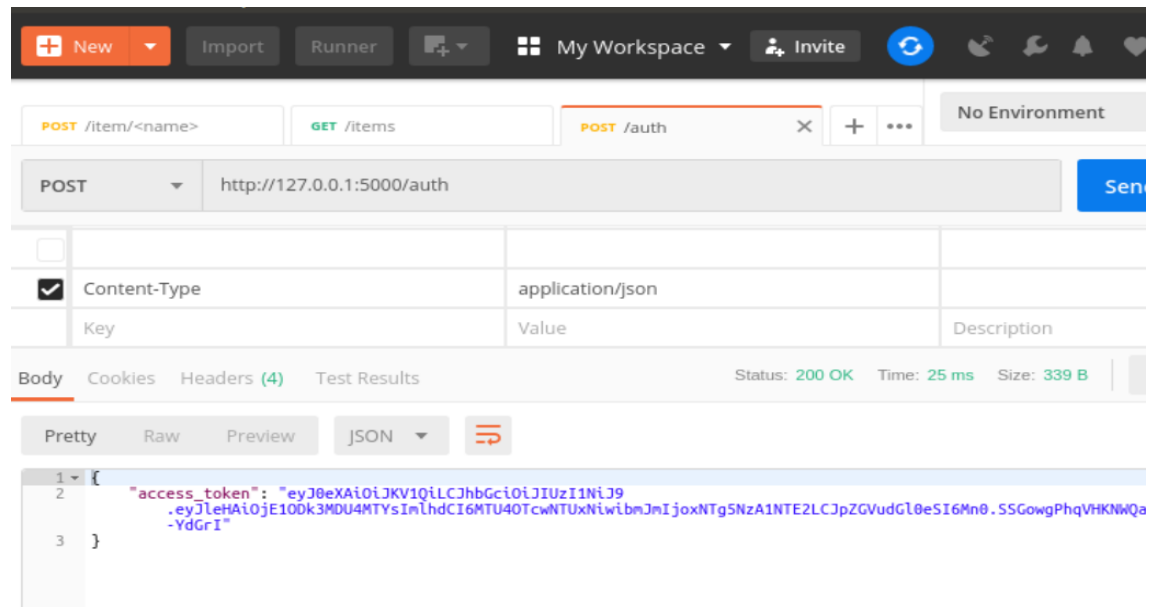
- Method:

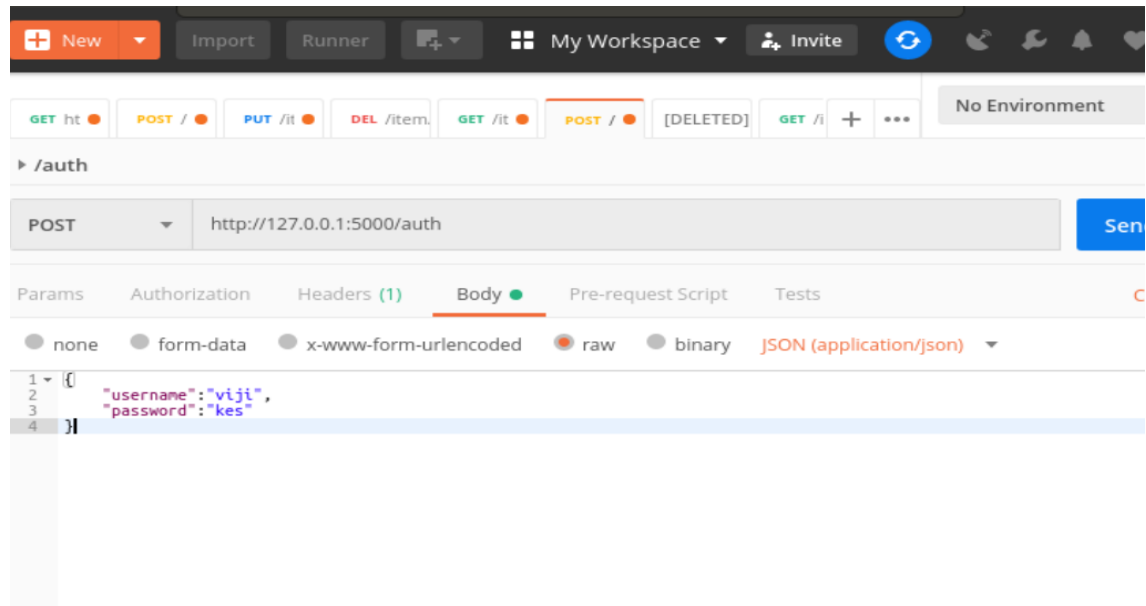
Before starting this section there are some library we need to install in the linux. Flask JWT is must to install with the code **"pip install Flask-JWT"** and also importat thing is we have to be in the virtual environment lwhile installing the library.

Now open the app.py and we need to add some line for the secret key which is should be complicate to guess for others. Make a new file with the name security.py and import werkzeug.security.safe_str_cmp this function is used to compare strings.

And create the new file user.py which help to create the new user. It help them to create id and username and if we want

- Result:





6 Resource storage in SQL database

- Introduction:

SQL Server cannot back up the Resource database. You can perform your own file-based or a disk-based backup by treating the mssqlsystemresource.mdf file as if it were a binary (.EXE) file, rather than a database file, but you cannot use SQL Server to restore your backups. Restoring a backup copy of mssqlsystemresource.mdf can only be done manually, and you must be careful not to overwrite the current Resource database with an out-of-date or potentially insecure version.

The Resource database should only be modified by or at the direction of a Microsoft Customer Support Services (CSS) specialist. The ID of the Resource database is always 32767. Other important values associated with the Resource database are the version number and the last time that the database was updated.

- Method:

As usual we are start creating environment for our work. But here we are going to use python environment manager (virtualenv). now we need to get inside to our environment and install new environment in it **venv**

If virtualenv is not yet installed on your system, try it with this code, **sudo apt install virtualenv**. To activate it with the command use this **source**

venc/bi /activate". Then install the flask packages

pip install Flask

pip install Flask-RESTful

pip install Flask-JWT

Now run this program in the terminal if it is running then it is ok we installed correctly.

Running SQLite

Create a new file named test.py in atom which has codes to create a database. Before that we need to initialize a connection and call to our database data.db. The cursor command will take care of the queries and the table contains id, username, password. At the end always commit and close command should be performed in order to save the data **connection.commit()** **connection.close()**. After completing the program and run it in terminal. first test.py has to run by the command **python test.py** it will generate one data.db file in the required folder so delete it before we are going to run the **python app.py** command.

- Result

The screenshot shows a database application interface. At the top, there are menu items: File, Edit, View, Help. Below the menu is a toolbar with buttons: New Database, Open Database, Write Changes, and Revert Changes. The main area is divided into two panels. The left panel, titled 'Database Structure', shows a table named 'users' with columns 'id', 'username', and 'password'. The table contains 5 records. The right panel, titled 'Edit Database Cell', shows the 'Mode' set to 'Text' and a text input field containing 'mahendra'. Below the text input field, it says 'Type of data currently in cell: 8 char(s)'. At the bottom right, there is a 'Remote' section with an 'Identity' dropdown and a 'Commit' button.

	id	username	password
1	1	mahendra	09876
2	2	iyer	asdf
3	3	kobhar	lkjh
4	4	umu	poiu
5	5	vind	qwert