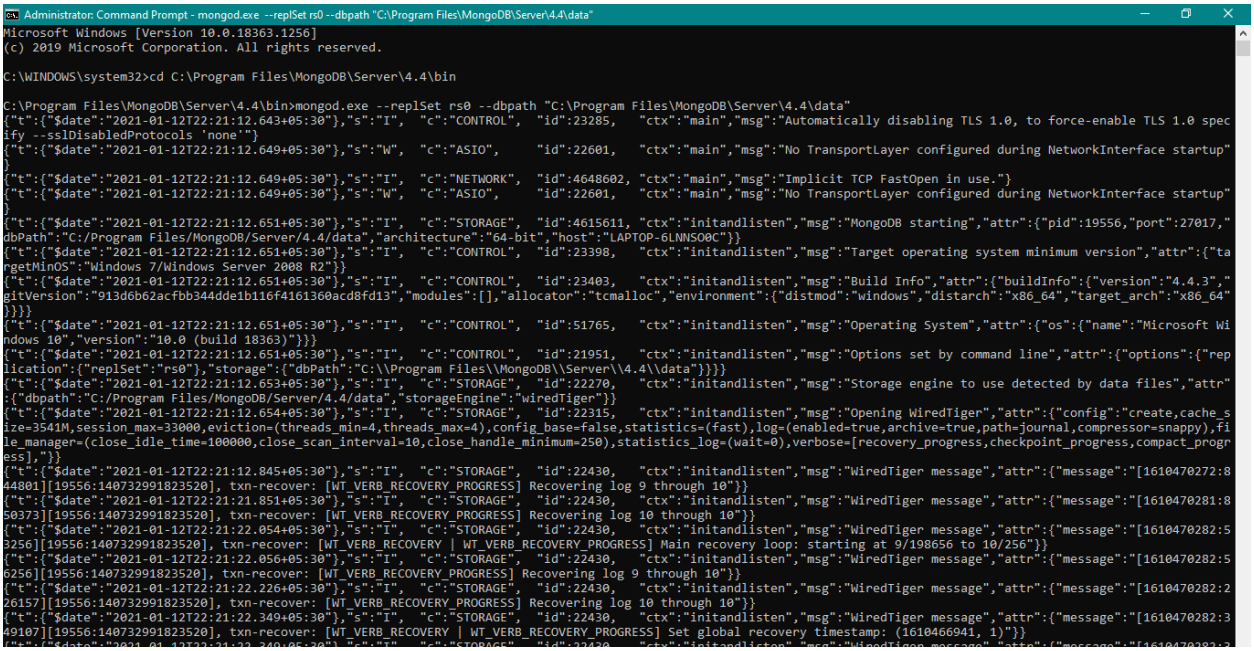
1. Flow of the program

Step 1 – Configuring mongod.ex file and creating rs replica dataset

```
cd C:\Program Files\MongoDB\Server\4.4\bin

mongod.exe --replSet rs0 --dbpath "C:\Program Files\MongoDB\Server\4.4\data"
```



Step 2 – opening mongo.exe(Run As administer) and initiate rs set

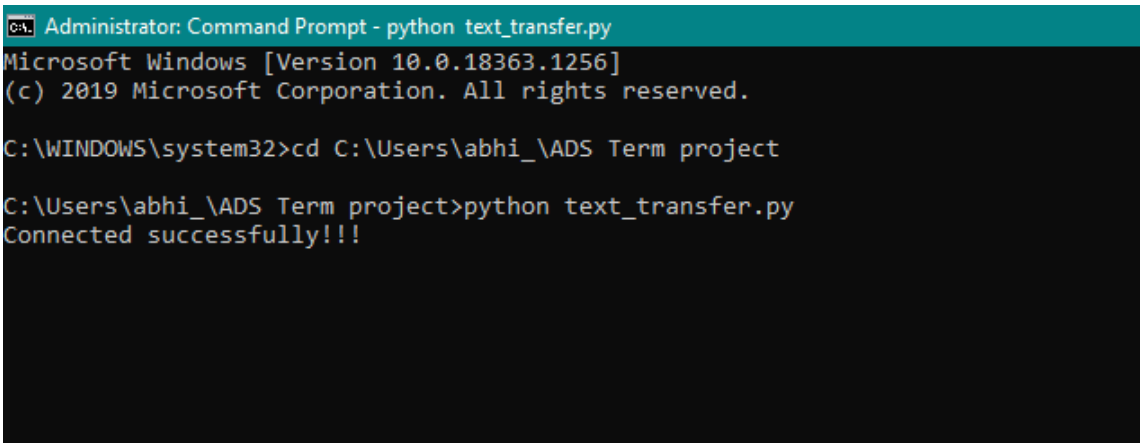
```
rs.initiate()
```

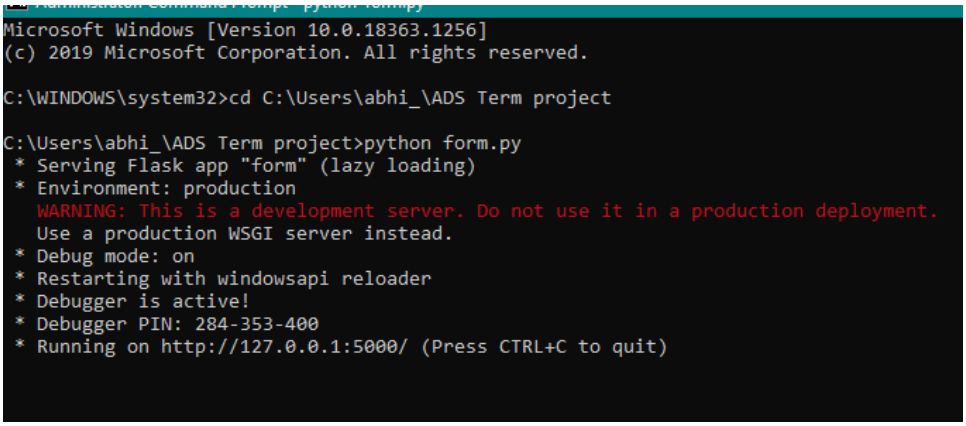
```
ess> to specify which IP addresses it should serve responses from, or with --bind_ip_all to bind to all interfaces. If this behavior is desired, start the server with -
-bind_ip 127.0.0.1 to disable this warning
---
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
rs0:PRIMARY> rs.initiate()
{
        "operationTime" : Timestamp(1610470585, 3),
        "ok" : 0,
        "errmsg" : "already initialized",
        "code" : 23,
        "codeName" : "AlreadyInitialized",
        "$clusterTime" : {
                "clusterTime" : Timestamp(1610470585, 3),
                "signature" : {
                        "hash" : BinData(0,"AAAAAAAAAAAAAAAAAAAAAAAAAAA="),
                        "keyId" : NumberLong(0)
                }
        }
}
rs0:PRIMARY>
```

Step 3 – Opening recieve.py file, text_transfer.py file, form.py file(in sequence)

```
python recive.py

Python text_transfer.py

python form.py
```

```
Administrator: Command Prompt - python  recieve.py
Microsoft Windows [Version 10.0.18363.1256]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd C:\Users\abhi_\ADS Term project

C:\Users\abhi_\ADS Term project>python recieve.py
 [*] Waiting for messages. To exit press CTRL+C
```





Step 3 – checking database status before inserting data

Mongo DB

```
Show databases;
Use database;
Show collections;
db.textsummary.find()
```

```
rs0:PRIMARY> show databases;
admin         0.000GB
config        0.000GB
database      0.000GB
local         0.001GB
pymongo_test  0.000GB
rs0:PRIMARY> use database
switched to db database
rs0:PRIMARY> show collections
textsummary
rs0:PRIMARY> db.textsummary.find()
rs0:PRIMARY>
```

MySQL

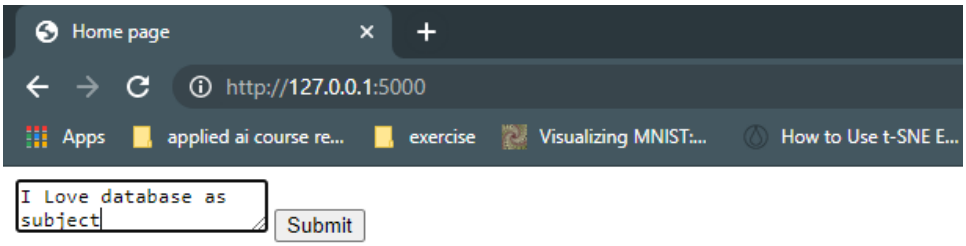
Show databases;

Use text_data

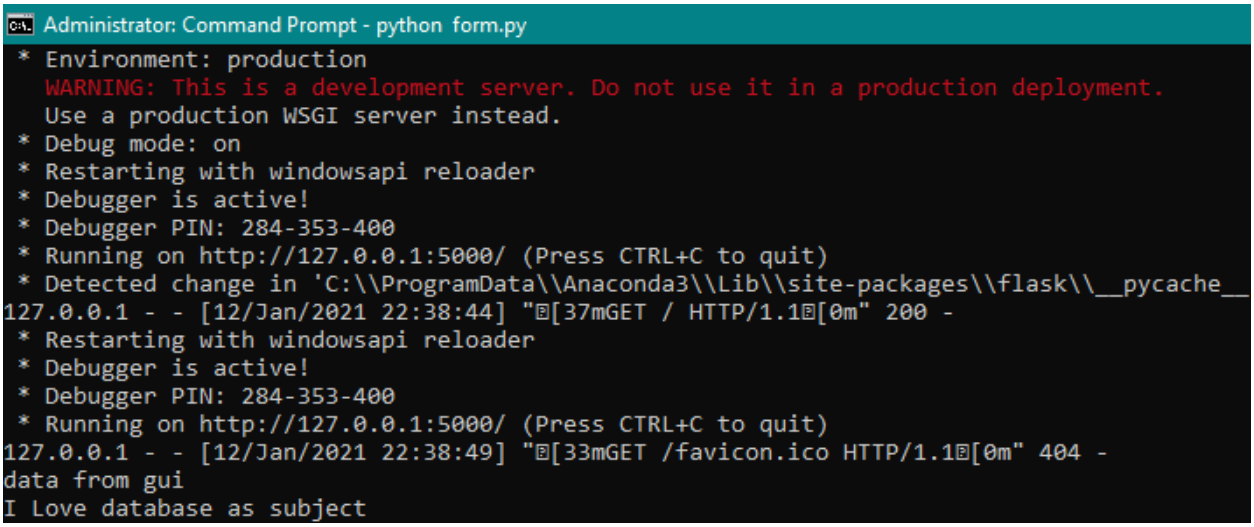
Select * from text;

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sakila             |
| sys                |
| text_data          |
| world              |
+--------------------+
7 rows in set (0.63 sec)

mysql> use text_data
Database changed
mysql> select *from text;
Empty set (0.08 sec)

mysql>
```

Step 4 – Inserting data into form



Step 5 – Data taken from GUI and inserted into mongodb



Step 6 – checking data in mongodb

```
db.textsummary.find()
```

Step 7 – Now checking message queue

```
C:\Users\abhi_\ADS Term project>python text_transfer.py
Connected successfully!!!
data not interested



In the message queue
inserted data I Love database as subject
```

Step 8 – Checking on the receiving side of the messaging queue

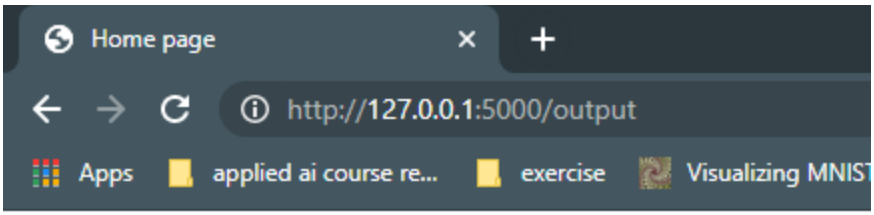
```
C:\Users\abhi_\ADS Term project>python recieve.py
 [*] Waiting for messages. To exit press CTRL+C
I Love database as subject



Data written to mysql database
```

Step 9 – checking on MySQL database

```
Select * from text;
```

```
mysql> select *from text;
+---------+---------------------------+
| text_id | text_message              |
+---------+---------------------------+
|      34 | I Love database as subject |
+---------+---------------------------+
1 row in set (0.00 sec)
```

Step 10 – Inserted data written back to output page



Data taken from MySQL

(34, 'I Love database as subject')

2. Important code snippets

Register.html

Step 1 – User form

```
<form method = 'POST' action = " ">
  {{form.messagetext}}
  <input type="submit" value = "Submit">
```

Form.py

Step 2 - Taking data from GUI (form.py)

```
@app.route('/',methods=['GET','POST'])
def register():
    form = message(request.form)
    if request.method == "POST" and form.validate():
        print("data from gui")
        print(form.messagetext.data)
```

Step 3– Connecting Mongo DB

```
    try:
        conn = MongoClient()
        print("Connected successfully!!!")
    except:
        print("Could not connect to MongoDB")
    #creating database connection instance
    db = conn.database
    #creating new database collection
    collection = db.textsummary
    Text1 = {
    "message":form.messagetext.data
    }
    rec_id1 = collection.insert_one(Text1)
```

Text_transfer.py

Step 4 – CDC checking for database for change

```
collection = db.textsummary
with collection.watch() as stream:
    while stream.alive:
        change = stream.try_next()
        if change is not None:
```

Step 5 - Capturing change and setting up message queue

```
channel = connection.channel()
message = change["fullDocument"]["message"]
print("\n\n\n")
print("In the message queue")
print("inserted data",message)
print("\n\n\n")
channel.queue_declare(queue='hello')
channel.basic_publish(exchange='',routing_key='hello',body=message)
```

Receiver.py

Step 6 – Queuing set up at receiver end

```
connection = pika.BlockingConnection(pika.ConnectionParameters(host='localhost'))
channel = connection.channel()

channel.queue_declare(queue='hello')
```

Step 7 – Connecting to MySQL and inserting data into it

```
def callback(ch, method, properties, body):
    mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="abhi1998",
    database="text_data"
    )

    # Cursor setup
    cursor_sq = mydb.cursor()
    #inserting into mysql DATABASE
    print(body.decode('ascii'))
    add_text = "INSERT INTO text(text_message) VALUES ('{}')".format(body.decode('ascii'))
    cursor_sq.execute(add_text)
```

Form.py

Step 8 – Connecting to MySQL and reterving data from it

```
def script_output():
    mydb = mysql.connector.connect(
    host="localhost",
    user="root",
    password="abhi1998",
    database="text_data"
    )
```

```
  cursor_sq = mydb.cursor()
  result = []
  query = "select * from text"
  cursor_sq.execute(query)
  print("printing on command prompt")
  for r in cursor_sq.fetchall():
      result.append(r)
```

Step 9 – sending data to output.html page

```
  for r in cursor_sq.fetchall():
      result.append(r)
  return render_template('output.html',response = result)
```

Output.html

Step 10 – Printing data on html page

```
  <form method = 'POST' action = " ">
  <p>Data taken from MySQL</p>
    {% for line in response %}
      <p>{{ line }}</p>
    {% endfor %}
```