



# **PES UNIVERSITY**

(Established under Karnataka Act No. 16 of 2013) 100 Feet Ring Road, BSK III Stage, Bengaluru - 560085

Department of Computer Science and Engineering Session: Aug-Dec 2021

## **SEMESTER-5**

DATE:21-11-2021

DBMS PROJECT ASSIGNMENT-3

### **PROJECT TITLE**

**PHARMACY MANAGEMENT  
SYSTEM**

### **TEAM MEMBERS (SECTION B)**

Basanagouda(PES2UG19CS082)

Bhagyashree Shankar (PES2UG19CS085)

Bhavana R(PES2UG19CS089)

## LANGUAGE CHOICE

We chose python for designing the frontend part and postgresql as our backend server database.

Python is considered one of the best programming languages for web development because it's relatively easy to understand and it has a huge array of tools and functionalities. It's also extremely scalable and can carry out a wide range of outcomes.

We chose Django framework for the front end. Django is a [Python-based free and open-source web framework](#)

Django's primary goal is to ease the creation of complex, database-driven websites. The framework emphasizes [reusability](#) and "pluggability" of components, less code, low coupling, rapid development, and the principle of [don't repeat yourself](#).<sup>[9]</sup> Python is used throughout, even for settings, files, and data models. Django also provides an optional administrative [create, read, update and delete](#) interface that is generated dynamically through [introspection](#) and configured via admin models.

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Hence we chose python and windows forms based front end as it can be built in hassle free. As mentioned above this would have not been possible without Visual Studio tool.

The visual studio app and python worked hand in hand to help us build a front end with lesser efforts.

The database located in SQL Server within Visual studio could be connected easily without any difficulties.

## ADDITIONAL QUERIES

### Administrator

**Query to list administrator with the entered Name and Password**

```
SELECT *FROM LOGIN  
SELECT *FROM LOGIN WHERE ID=2;
```

```
Administrator: Command Prompt - psql -U postgres

postgres=# SELECT *FROM LOGIN;
 name | type | date       | time       | id
-----+-----+-----+-----+---
 admin | Admin | 2017-02-17 | 10:30:24   | 1
 admin | Admin | 2017-02-17 | 10:32:48   | 2
 mark  | Employee | 2017-02-17 | 10:32:56   | 1
 admin | Admin | 2017-02-17 | 10:33:18   | 1
 bhavana | Employee | 2017-02-17 | 10:33:37   | 5
 admin | Admin | 2017-02-17 | 10:36:21   | 1
 basana | Admin | 2017-02-17 | 10:36:53   | 6
 admin | Admin | 2017-02-17 | 10:49:27   | 1
 bhagya | Admin | 2017-02-17 | 11:02:23   | 7
 admin | Admin | 2017-02-17 | 01:40:08   | 1
(10 rows)

postgres=# SELECT *FROM LOGIN WHERE ID=2;
 name | type | date       | time       | id
-----+-----+-----+-----+---
 admin | Admin | 2017-02-17 | 10:32:48   | 2
(1 row)

postgres=#
```

## Company Details

Query to list the details of the company where phone no starts from 6

**SELECT \*FROM COMPANY;**

**SELECT \*FROM COMPANY WHERE PHONE LIKE '6%';**

```
Administrator: Command Prompt - psql -U postgres

postgres=# SELECT *FROM COMPANY;
 name      | address | phone
-----+-----+-----
 Cipla     | Mumbai | 8523452596
 Sun Pharma | Mysore | 8971199805
 Med_City  | Nellore | 6669923589
 Sanofi    | Mumbai | 6894512397
 Johnson & Johnson | banglore | 8645321798
 Pfizer    | hyderabad | 9456312878
 Abbott    | kolkata | 8631597234
 Glenmark  | delhi | 7539518264
(8 rows)

postgres=# SELECT *FROM COMPANY WHERE PHONE LIKE '6%';
 name      | address | phone
-----+-----+-----
 Med_City  | Nellore | 6669923589
 Sanofi    | Mumbai | 6894512397
(2 rows)

postgres=#
postgres=#
```

## Sales History

Query to list the details of sales where quantity of the drug sales is greater than 5 and drug mrp is greater than 15

**1)SELECT \*FROM HISTORY\_SALES;**

**SELECT \*FROM HISTORY\_SALES WHERE QUANTITY>=5 AND AMOUNT>=15;**

```
Administrator: Command Prompt - psql -U postgres
postgres=#
postgres=#
postgres=# SELECT *FROM HISTORY_SALES;
 user_name | barcode | name | type | dose | quantity | price | amount | date | time
-----
 admin | fsdgjfihjorodsf | Novalo | Bills | Free used | 2 | 6 | 12 | 2017-02-12 | 05:02:06
 admin | ftrkl432432md | novafol | Bills | Free used | 2 | 6 | 12 | 2017-02-12 | 05:02:26
 admin | fbankjijcfnl56 | novafol | Bills | Free used | 4 | 6 | 24 | 2017-02-12 | 05:02:40
 admin | fnjldhfcfjaipsk | Amifostine | Injection | 1 (Day) | 2 | 14 | 28 | 2017-02-13 | 01:38:00
 admin | fois54hiusdh | Calcium | Injection | 1 (Day) | 2 | 14 | 28 | 2017-02-13 | 01:38:10
 admin | fljhsfc35bsyc | Clindamycin | Injection | 1 (Day) | 7 | 14 | 98 | 2017-02-13 | 01:38:28
 admin | fkjoifjp55jlhvu | Dopamine | Injection | 1 (Day) | 1 | 14 | 14 | 2017-02-13 | 01:38:46
 mark | fkjhsido54zsc | Morphine | Bills | Free used | 2 | 6 | 12 | 2017-02-13 | 01:59:34
(8 rows)

postgres=# SELECT *FROM HISTORY_SALES WHERE QUANTITY>=5 AND AMOUNT>=15;
 user_name | barcode | name | type | dose | quantity | price | amount | date | time
-----
 admin | fljhsfc35bsyc | Clindamycin | Injection | 1 (Day) | 7 | 14 | 98 | 2017-02-13 | 01:38:28
(1 row)

postgres=#
```

## Drug Details

Query to list the details of drugs where cost of the drug is greater than 20 and quantity is greater than 15

2)SELECT \*FROM DRUGS;

SELECT \*FROM DRUGS WHERE COST\_PRICE>=20 OR QUANTITY>=25;

```
Administrator: Command Prompt - psql -U postgres
postgres=# SELECT *FROM DRUGS;
 name | type | barcode | dose | code | cost_price | selling_price | expiry | company_name | production_date | expiration_date |
-----
 Novalo | Bills | fsdgjfihjorodsf | normal | 3d00 | 2 | 3 | Available for use | Cipla | 2021-03-03 | 2023-03-03 | N
-Right | 40
 novafol | Bills | ftrkl432432md | normal | 2xaa | 33 | 40 | Available for use | Sun Pharma | 2020-01-01 | 2022-01-01 | N
-Left | 27
 Abacavir | Infection | fbankjijcfnl56 | normal | bhab | 44 | 50 | Available for use | Med_City | 2019-06-09 | 2022-04-03 | 2
62 DannyThomasPlace | 100
 Amifostine | Anxiety | fnjldhfcfjaipsk | normal | 2cfs | 39 | 55 | Available for use | Sanofi | 2021-03-05 | 2023-05-07 | 2
62 DannyThomasPlace | 90
 Calcium | mineral | fois54hiusdh | normal | sd3d | 69 | 80 | Available for use | Johnson & Johnson | 2020-08-08 | 2025-11-18 | N
-Left | 150
 Clindamycin | skin | fljhsfc35bsyc | normal | sbcr | 50 | 60 | Available for use | Pfizer | 2021-12-12 | 2022-11-15 | N
-Left | 80
 Dopamine | Lbp | fkjoifjp55jlhvu | normal | 52kh | 100 | 150 | Available for use | Abbott | 2020-09-22 | 2025-09-30 | N
-right | 200
 Morphine | Painkiller | fkjhsido54zsc | normal | fs25 | 250 | 270 | Available for use | Glenmark | 2021-05-25 | 2024-12-09 | 2
62 DannyThomasPlace | 300
(8 rows)

postgres=# SELECT *FROM DRUGS WHERE COST_PRICE>=20 OR QUANTITY>=25;
 name | type | barcode | dose | code | cost_price | selling_price | expiry | company_name | production_date | expiration_date |
-----
 Novalo | Bills | fsdgjfihjorodsf | normal | 3d00 | 2 | 3 | Available for use | Cipla | 2021-03-03 | 2023-03-03 | N
-Right | 40
 novafol | Bills | ftrkl432432md | normal | 2xaa | 33 | 40 | Available for use | Sun Pharma | 2020-01-01 | 2022-01-01 | N
-Left | 27
 Abacavir | Infection | fbankjijcfnl56 | normal | bhab | 44 | 50 | Available for use | Med_City | 2019-06-09 | 2022-04-03 | 2
62 DannyThomasPlace | 100
 Amifostine | Anxiety | fnjldhfcfjaipsk | normal | 2cfs | 39 | 55 | Available for use | Sanofi | 2021-03-05 | 2023-05-07 | 2
62 DannyThomasPlace | 90
 Calcium | mineral | fois54hiusdh | normal | sd3d | 69 | 80 | Available for use | Johnson & Johnson | 2020-08-08 | 2025-11-18 | N
-Left | 150
 Clindamycin | skin | fljhsfc35bsyc | normal | sbcr | 50 | 60 | Available for use | Pfizer | 2021-12-12 | 2022-11-15 | N
-Left | 80
 Dopamine | Lbp | fkjoifjp55jlhvu | normal | 52kh | 100 | 150 | Available for use | Abbott | 2020-09-22 | 2025-09-30 | N
-right | 200
```

## Sales Details

Query to list the details of barcode from sales and count the barcode

3)SELECT \*FROM SALE;

SELECT BARCODE FROM SALE GROUP BY BARCODE HAVING COUNT(BARCODE)<12;

```
Administrator: Command Prompt - psql -U postgres

postgres=# SELECT *FROM SALE;
-----
barcode | name | type | dose | quantity | price | amount | date
-----
fkjoifp5s1hvu | dopamine | Lbp | normal | 2 | 150 | 300 | 2021-10-12
fljhsfc3sbsyc | Clindamycin | skin | normal | 4 | 60 | 200 | 2021-10-12
fnjldhlcfcjjaipsk | Amifostine | Anxiety | normal | 10 | 39 | 400 | 2021-10-13
fljhsfc3sbsyc | Clindamycin | skin | normal | 4 | 60 | 200 | 2021-10-13
fkjoifp5s1hvu | dopamine | Lbp | normal | 2 | 150 | 300 | 2021-10-14
fkjhsido54zsc | Morphine | Painkiller | normal | 10 | 270 | 2700 | 2021-10-14
fois54hiusdh | Calcium | mineral | normal | 50 | 80 | 4000 | 2021-10-14
fbankjijcfnl56 | Abacavir | infection | normal | 50 | 50 | 1000 | 2021-10-14
(8 rows)

postgres=# SELECT BARCODE FROM SALE GROUP BY BARCODE HAVING COUNT(BARCODE)<12;
barcode
-----
fkjoifp5s1hvu
fnjldhlcfcjjaipsk
fois54hiusdh
fljhsfc3sbsyc
fbankjijcfnl56
fkjhsido54zsc
(6 rows)

postgres=#
```

## Company User Details

Query to update the salary to 12000 whose id is 7

5)SELECT \*FROM USERS;  
UPDATE USERS SET SALARY=12000 WHERE ID=7;

```
Administrator: Command Prompt - psql -U postgres

postgres=#
postgres=# SELECT *FROM USERS;
-----
id | name | dob | address | phone | salary | password
-----
1 | admin | 23-12-1995 | Bangalore | 9800000000 | 50000 | admin
2 | mark | 3-2-1972 | Bangalore | 9632587418 | 2000 | mark
3 | Clark | 3-2-1971 | Richmond circle | 8521479633 | 4000 | rootaccess
4 | arya | 7-8-1977 | Bangalore | 6478932143 | 3000 | rootaccess
5 | bhavana | 3-4-2002 | Bangalore | 9988989711 | 10000 | rootaccess
6 | basana | 12-07-2001 | Belgium | 6547893218 | 7000 | rootaccess
7 | bhagya | 06-06-2001 | Bangalore | 8265479313 | 9000 | rootaccess
8 | Tom | 09-06-2001 | mumbai | 8971199085 | 10000 | rootaccess
(8 rows)

postgres=# UPDATE USERS SET SALARY=12000 WHERE ID=7;
UPDATE 1
postgres=#
```

# Screenshots of Visual Studio Code( Front End)

Pharmacy Management

127.0.0.1:8000

Maps Gmail YouTube BBMP Page PESU CSE 3 Sem ... Profile | Dashboard

Pharmacy Management

Company Details

Name	Address	Phonenumber
Cipla	Mumbai	8523452596
Sun Pharma	Mysore	8971199805
Med_City	Nellore	6669923589
Sanofi	mumbai	6894512397
Johnson & Johnson	banglore	8645321798
Pfizer	hyderabad	9456312878
Abbott	kolkata	8631597234
Glenmark	delhi	7539518264

Drugs details

name	type	barcode	dose	code	cost_price	selling_price	expiry	company_name	production_date	expiration_date	place	quantity
Novalo	Bills	fsdgi432432md	normal	3d00	2.0	3.0	Available for use	Cipla	March 3, 2021	March 3, 2023	N-Right	40
novafol	Bills	frkl432432md	normal	2xaa	33.0	40.0	Available for use	Sun Pharma	Jan. 1, 2020	Jan. 1, 2022	N-Left	27
Abacavir	infection	fbankijefnl56	normal	bhab	44.0	50.0	Available for use	Med_City	June 9, 2019	April 3, 2022	262 DannyThomasPlace	100
Amifostine	Anxiety	fujldhiefajpsk	normal	2cf5	39.0	55.0	Available for use	Sanofi	March 5, 2021	May 7, 2023	262 DannyThomasPlace	90
Calcium	mineral	fois54hiusdh	normal	sd3d	69.0	80.0	Available for use	Johnson & Johnson	Aug. 8, 2020	Nov. 18, 2025	N-Left	150
Clindamycin	skin	fljhsfc35bsyc	normal	sber	50.0	60.0	Available for use	Pfizer	Dec. 12, 2021	Nov. 15, 2022	N-Left	80
Dopamine	Lbp	fkjoifp5sjlhvu	normal	52kh	100.0	150.0	Available for use	Abbott	Sept. 22, 2020	Sept. 30, 2025	N-right	200
Morphine	Painkiller	fkjhsido54zsc	normal	fc25	250.0	270.0	Available for use	Glenmark	May 25, 2021	Dec. 9, 2024	262 DannyThomasPlace	300
paracetamol	Pain	fkjhsido54abi	normal	fc03	100.0	110.0	Available for use	Pfizer	Nov. 4, 2021	Dec. 9, 2023	west	100
ASPRIN	Pain	fkjhs03Ho54BAB	normal	fc33	100.0	110.0	Available for use	Pfizer	Nov. 4, 2021	Dec. 9, 2023	west	100

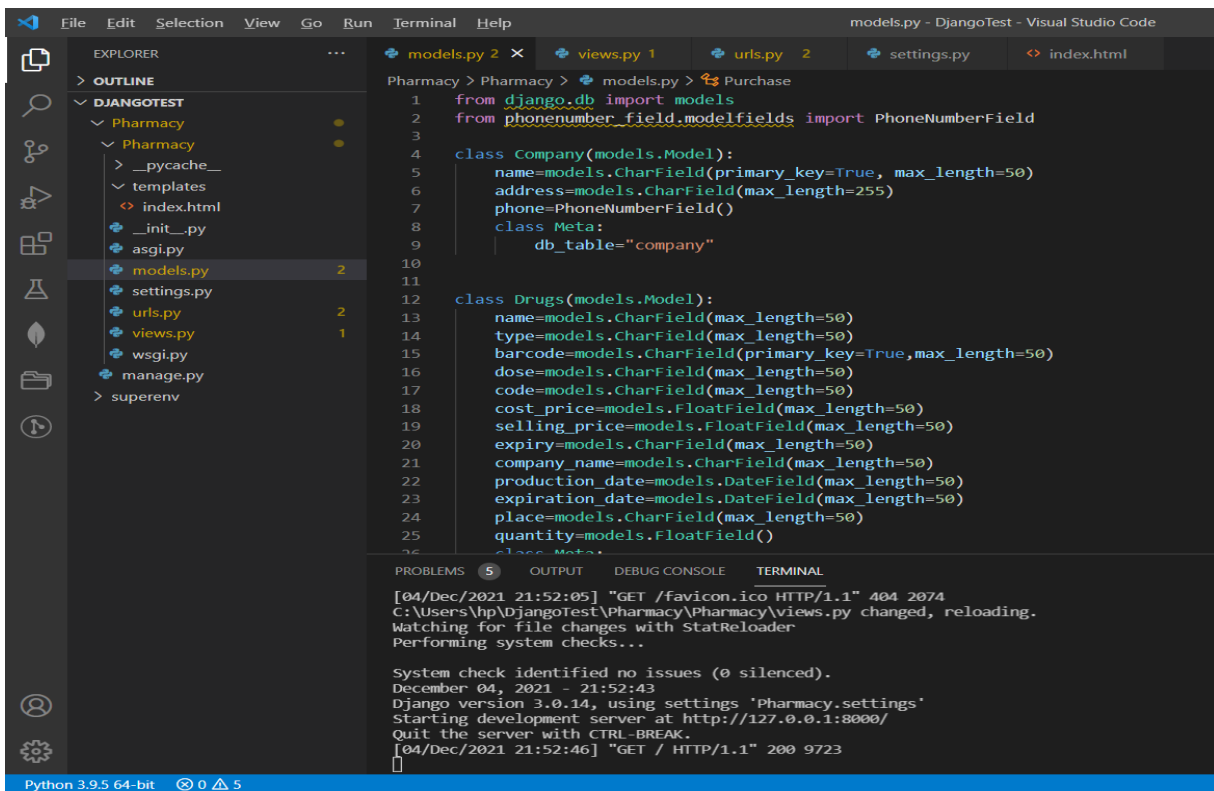
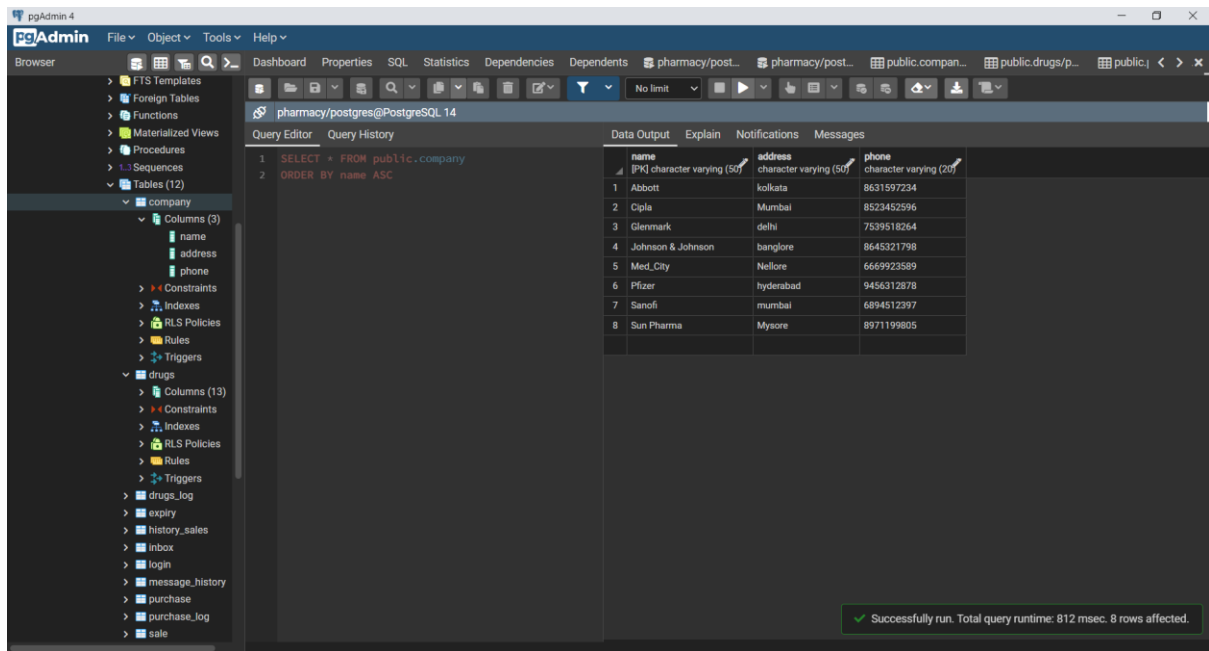
Drugs details

name	type	barcode	dose	code	cost_price	selling_price	expiry	company_name	production_date	expiration_date	place	quantity
Novalo	Bills	fsdgi432432md	normal	3d00	2.0	3.0	Available for use	Cipla	March 3, 2021	March 3, 2023	N-Right	40
novafol	Bills	frkl432432md	normal	2xaa	33.0	40.0	Available for use	Sun Pharma	Jan. 1, 2020	Jan. 1, 2022	N-Left	27
Abacavir	infection	fbankijefnl56	normal	bhab	44.0	50.0	Available for use	Med_City	June 9, 2019	April 3, 2022	262 DannyThomasPlace	100
Amifostine	Anxiety	fujldhiefajpsk	normal	2cf5	39.0	55.0	Available for use	Sanofi	March 5, 2021	May 7, 2023	262 DannyThomasPlace	90
Calcium	mineral	fois54hiusdh	normal	sd3d	69.0	80.0	Available for use	Johnson & Johnson	Aug. 8, 2020	Nov. 18, 2025	N-Left	150
Clindamycin	skin	fljhsfc35bsyc	normal	sber	50.0	60.0	Available for use	Pfizer	Dec. 12, 2021	Nov. 15, 2022	N-Left	80
Dopamine	Lbp	fkjoifp5sjlhvu	normal	52kh	100.0	150.0	Available for use	Abbott	Sept. 22, 2020	Sept. 30, 2025	N-right	200
Morphine	Painkiller	fkjhsido54zsc	normal	fc25	250.0	270.0	Available for use	Glenmark	May 25, 2021	Dec. 9, 2024	262 DannyThomasPlace	300
paracetamol	Pain	fkjhsido54abi	normal	fc03	100.0	110.0	Available for use	Pfizer	Nov. 4, 2021	Dec. 9, 2023	west	100
ASPRIN	Pain	fkjhs03Ho54BAB	normal	fc33	100.0	110.0	Available for use	Pfizer	Nov. 4, 2021	Dec. 9, 2023	west	100

Purchase details

barcode	name	type	company_name	quantity	price	amount
fsdgi432432md	Novalo	Bills	Cipla	40	2.0	80.0
frkl432432md	novafol	Bills	Sun Pharma	20	33.0	660.0
fkjhsido54zsc	Morphine	Painkiller	Med_City	4	270.0	1088.0
fkjoifp5sjlhvu	Dopamine	Lbp	Sanofi	10	150.0	1500.0
fljhsfc35bsyc	Clindamycin	skin	Johnson & Johnson	5	60.0	300.0
fujldhiefajpsk	Amifostine	Anxiety	Pfizer	10	55.0	550.0
fois54hiusdh	Calcium	mineral	Abbott	30	80.0	1100.0
fbankijefnl56	Abacavir	infection	Glenmark	20	50.0	100.0





```
1 <!DOCTYPE html>
2 <head>
3   <title>Pharmacy Management</title>
4 </head>
5 <body>
6   <center>
7     <h1>Pharmacy Management</h1>
8   </center>
9   <hr>
10  <h1>Company Details</h1>
11  <table border="1">
12    <tr>
13      <th>Name</th>
14      <th>Address</th>
15      <th>Phonenumber</th>
16    </tr>
17    {% for result in Company %}
18    <tr>
19      <td>{{result.name}}</td>
20      <td>{{result.address}}</td>
21      <td>{{result.phone}}</td>
22    </tr>
23    {% endfor %}
24  </table>
25  <hr>
26  <h2>Drugs details</h2>
27  <table border=1>
28    <tr>
29      <th>name</th>
30      <th>type</th>
31      <th>barcode</th>
32      <th>dose</th>
33      <th>code</th>
34      <th>net_price</th>
```

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL

[04/Dec/2021 21:52:46] "GET / HTTP/1.1" 200 9723

```
1 from django.shortcuts import render
2 from Pharmacy.models import Company,Drugs,Purchase
3
4 def showdata(request):
5     commodeldisplay=Company.objects.all()
6     stumodeldisplay=Drugs.objects.all()
7     purmodeldisplay=Purchase.objects.all()
8     return render(request,'index.html',{'company':commodeldisplay,'Drugs': stumodeldisplay,'Purchase': purmodeldisplay})
9
10
11
12
```

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL

[04/Dec/2021 21:52:05] "GET /favicon.ico HTTP/1.1" 404 2074  
C:\Users\hp\ DjangoTest\Pharmacy\Pharmacy\views.py changed, reloading.  
Watching for file changes with StatReloader  
Performing system checks...  
System check identified no issues (0 silenced).  
December 04, 2021 - 21:52:43  
Django version 3.0.14, using settings 'Pharmacy.settings'  
Starting development server at http://127.0.0.1:8000/  
Quit the server with CTRL-BREAK.  
[04/Dec/2021 21:52:46] "GET / HTTP/1.1" 200 9723



## MIGRATION TO NO-SQL DATABASE

Big data and data analytics require migrating from relational databases (SQL) to NoSQL data structures to represent the data. Such transformation is challenging because of the lack of automatic transformation process and the requirement of guaranteeing both performance and accurate representation.

NoSQL datastores are designed for efficiently handling a lot more data than RDBMS. There are no relational constraints on the data, and it does not need to be even tabular. NoSQL offers performance at a higher scale by typically giving up strong consistency. Data access is mostly through REST APIs. The core principle of NoSQL is 'high availability'.

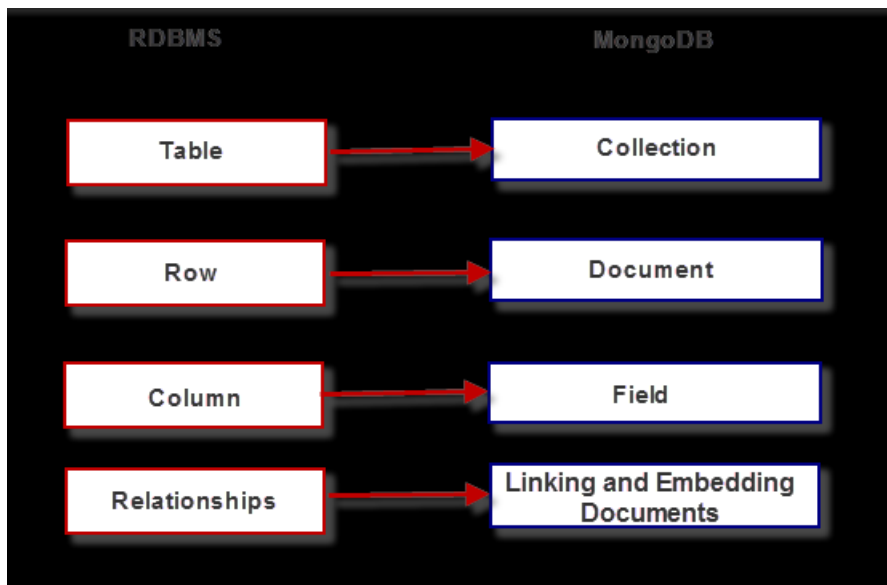
NoSQL is defined as a term which refers to a specific type of database model or DBMS.

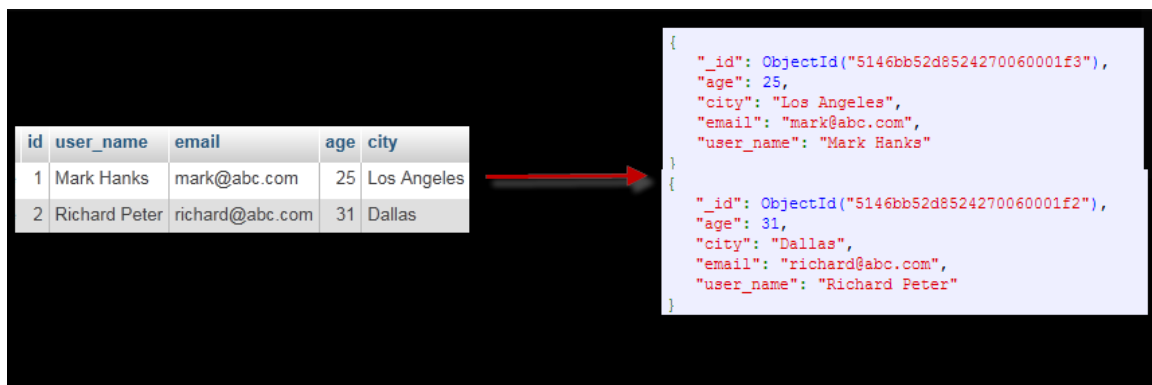
Benefits include cost benefits, performance, scalability, future proof for changes, reducing conversion jobs, and extensive supportability for analytics. Apache Cassandra and MongoDB are the most popularly used NoSQL databases.

Reasons to Use a NoSQL Database

- Semi-Structured and Unstructured Data. Data on the Web can be unstructured, semi-structured, or structured
- Agile Database Schema.
- Distributed
- Scalable
- Highly Available

## SQL v/s NoSQL database mapping





## Features Comparison

	Relational database	MongoDB
Rich Data Model	No	Yes
Dynamic Schema	No	Yes
Typed Data	Yes	Yes
Data Locality	No	Yes
Field Updates	Yes	Yes
Easy for Programmers	No	Yes
Complex Transactions	Yes	No
Auditing	Yes	Yes
Auto-Sharding	No	Yes

### Advantages of relational databases

- Good choice for applications that involve the management of several transactions.
- The structure of a relational database allows us to link information from different tables using foreign keys.
- Maintains ACID properties (the set of properties that guarantee database transactions are processed reliably) important for reliability.
- SQL is known by huge no of people.

### Advantages of non-relational databases

- Lightweight data interchange format.
- if you find yourself having to de-normalize your database schema, non-relational databases like Mongo may be the best way to go.
- Database is not at risk for SQL injection attacks.
- Sharding distributes the data across partitions to overcome hardware limitations.

### Disadvantages of non-relational databases

- Since there are no joins like there would be in relational databases, we need to perform multiple queries and join the data manually within our code – and that can get very ugly, very fast.
- Mongo doesn't automatically treat operations as transactions the way a relational database does, we must manually choose to create a transaction and then manually verify it, manually commit it or roll it back. To put it simply, some operations will succeed while others fail.

## **CONTRIBUTION**

### **Bhavana R:**

Front end, additional queries – 3 TO 4 HRS

### **Bhagyashree Shankar**

Migration to No SQL , Report – 2 TO 3 HRS

### **Basanagouda :**

Schema Changes and additional queries - 2 TO 3 HRS