DBMS MINI PROJECT
ASSIGNMENT 4 REPORT
STORE MANAGEMENT SYSTEM

TEAM NUMBER : 10

PRAMATHA GAJANAN BHAT	**PES1UG19CS339**	F
PRATIKSHA D NAYAK	**PES1UG19CS349**	
PRERANA HADADI	**PES1UG19CS352**	

- **The language choice for the front end design is HTML ,CSS**

Reason for choosing HTML and CSS :

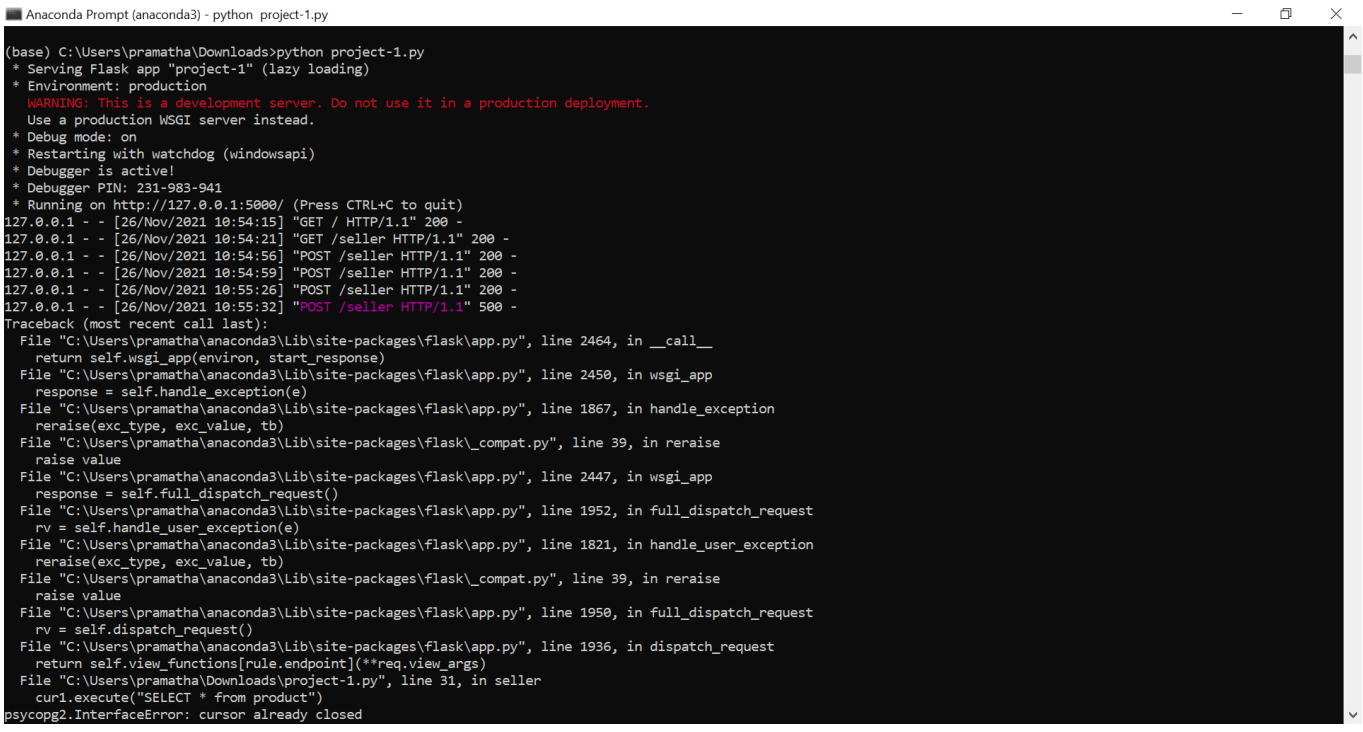
- Every browser supports HTML.
- It is easy to learn, use and modify.
- It is by default available in all of the browsers, so no need to purchase and instal
- CSS saves time- You can write CSS once and then reuse same sheet in multiple HTML pages.
- Easy maintenance − To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.

- **The language choice for the back end design is psycopg2, flask.**

Reason for choosing Psycopg2 and flask :

Psycopg2 is a most stable module to connect to PostgreSQL, It is used in most of the Postgres framework .This is also thread safe and designed for multi-threaded Applications.

Flask reduces development time and allows programmers to build faster and smarter.

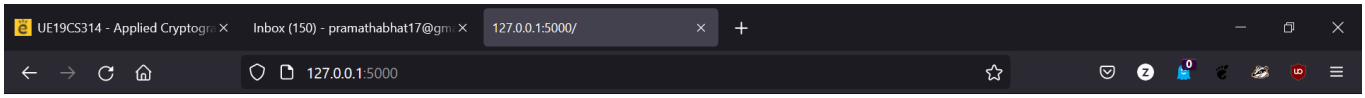


Anaconda Prompt (anaconda3) - python project-1.py

```
127.0.0.1 - - [26/Nov/2021 10:55:34] "GET /seller?__debugger__=yes&cmd=resource&f=style.css HTTP/1.1" 200 -
127.0.0.1 - - [26/Nov/2021 10:55:34] "GET /seller?__debugger__=yes&cmd=resource&f=debugger.js HTTP/1.1" 200 -
127.0.0.1 - - [26/Nov/2021 10:55:35] "GET /seller?__debugger__=yes&cmd=resource&f=ubuntu.ttf HTTP/1.1" 200 -
127.0.0.1 - - [26/Nov/2021 10:55:35] "GET /seller?__debugger__=yes&cmd=resource&f=console.png HTTP/1.1" 200 -
 * Detected change in 'C:\\Users\\pramatha\\Downloads\\project-1.py', reloading
 * Detected change in 'C:\\Users\\pramatha\\Downloads\\project-1.py', reloading
 * Detected change in 'C:\\Users\\pramatha\\Downloads\\project-1.py', reloading
 * Restarting with watchdog (windowsapi)
 * Debugger is active!
 * Debugger PIN: 231-983-941
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Detected change in 'C:\\Users\\pramatha\\Downloads\\project-1.py', reloading
 * Detected change in 'C:\\Users\\pramatha\\Downloads\\project-1.py', reloading
 * Detected change in 'C:\\Users\\pramatha\\Downloads\\project-1.py', reloading
 * Restarting with watchdog (windowsapi)
 * Debugger is active!
 * Debugger PIN: 231-983-941
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Detected change in 'C:\\Users\\pramatha\\Downloads\\project-1.py', reloading
 * Detected change in 'C:\\Users\\pramatha\\Downloads\\project-1.py', reloading
 * Detected change in 'C:\\Users\\pramatha\\Downloads\\project-1.py', reloading
 * Restarting with watchdog (windowsapi)
 * Debugger is active!
 * Debugger PIN: 231-983-941
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [26/Nov/2021 10:56:16] "POST /seller HTTP/1.1" 200 -
127.0.0.1 - - [26/Nov/2021 10:56:28] "POST /seller HTTP/1.1" 500 -
Traceback (most recent call last):
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 2464, in __call__
    return self.wsgi_app(environ, start_response)
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 2450, in wsgi_app
    response = self.handle_exception(e)
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 1867, in handle_exception
    reraise(exc_type, exc_value, tb)
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\_compat.py", line 39, in reraise
    raise value
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 2447, in wsgi_app
    response = self.full_dispatch_request()
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 1952, in full_dispatch_request
    rv = self.handle_user_exception(e)
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 1821, in handle_user_exception
```

Anaconda Prompt (anaconda3) - python project-1.py

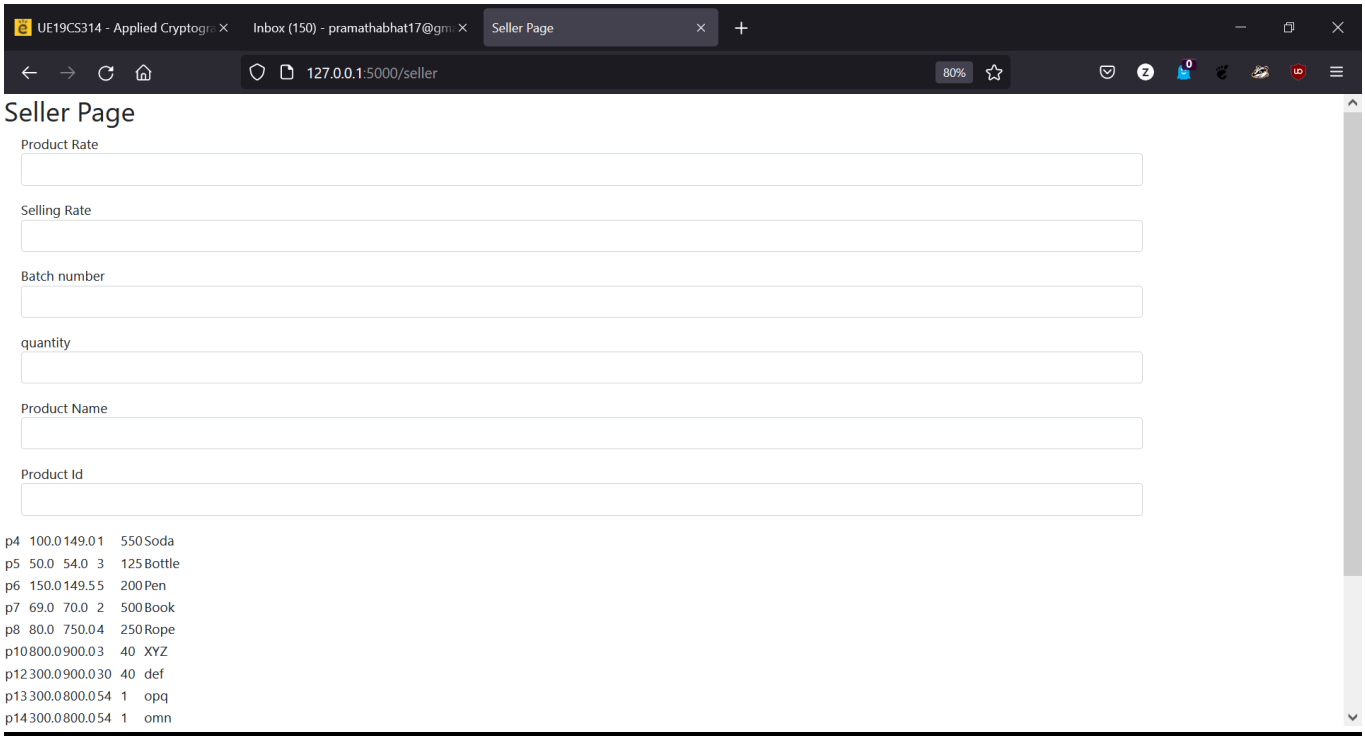
```
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 2464, in __call__
    return self.wsgi_app(environ, start_response)
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 2450, in wsgi_app
    response = self.handle_exception(e)
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 1867, in handle_exception
    reraise(exc_type, exc_value, tb)
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\_compat.py", line 39, in reraise
    raise value
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 2447, in wsgi_app
    response = self.full_dispatch_request()
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 1952, in full_dispatch_request
    rv = self.handle_user_exception(e)
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 1821, in handle_user_exception
    reraise(exc_type, exc_value, tb)
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\_compat.py", line 39, in reraise
    raise value
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 1950, in full_dispatch_request
    rv = self.dispatch_request()
  File "C:\Users\pramatha\anaconda3\Lib\site-packages\flask\app.py", line 1936, in dispatch_request
    return self.view_functions[rule.endpoint](**req.view_args)
  File "C:\Users\pramatha\Downloads\project-1.py", line 64, in seller
    cur1.execute("DELETE FROM PRODUCT WHERE product_id=%s",(ProductId,))
psycopg2.errors.ForeignKeyViolation: update or delete on table "product" violates foreign key constraint "inventory_product_name_fkey" on table "inventory"
DETAIL:  Key (prod_name)=(Soda) is still referenced from table "inventory".
127.0.0.1 - - [26/Nov/2021 10:56:30] "GET /seller?__debugger__=yes&cmd=resource&f=style.css HTTP/1.1" 200 -
127.0.0.1 - - [26/Nov/2021 10:56:30] "GET /seller?__debugger__=yes&cmd=resource&f=debugger.js HTTP/1.1" 200 -
127.0.0.1 - - [26/Nov/2021 10:56:30] "GET /seller?__debugger__=yes&cmd=resource&f=console.png HTTP/1.1" 200 -
127.0.0.1 - - [26/Nov/2021 10:56:40] "POST /seller HTTP/1.1" 200 -
127.0.0.1 - - [26/Nov/2021 10:56:47] "POST /seller HTTP/1.1" 200 -
127.0.0.1 - - [26/Nov/2021 10:56:50] "POST /seller HTTP/1.1" 200 -
127.0.0.1 - - [26/Nov/2021 10:57:02] "POST /seller HTTP/1.1" 200 -
127.0.0.1 - - [26/Nov/2021 10:57:06] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [26/Nov/2021 10:57:08] "GET /inventory HTTP/1.1" 200 -
127.0.0.1 - - [26/Nov/2021 10:57:11] "POST /seller HTTP/1.1" 200 -
```

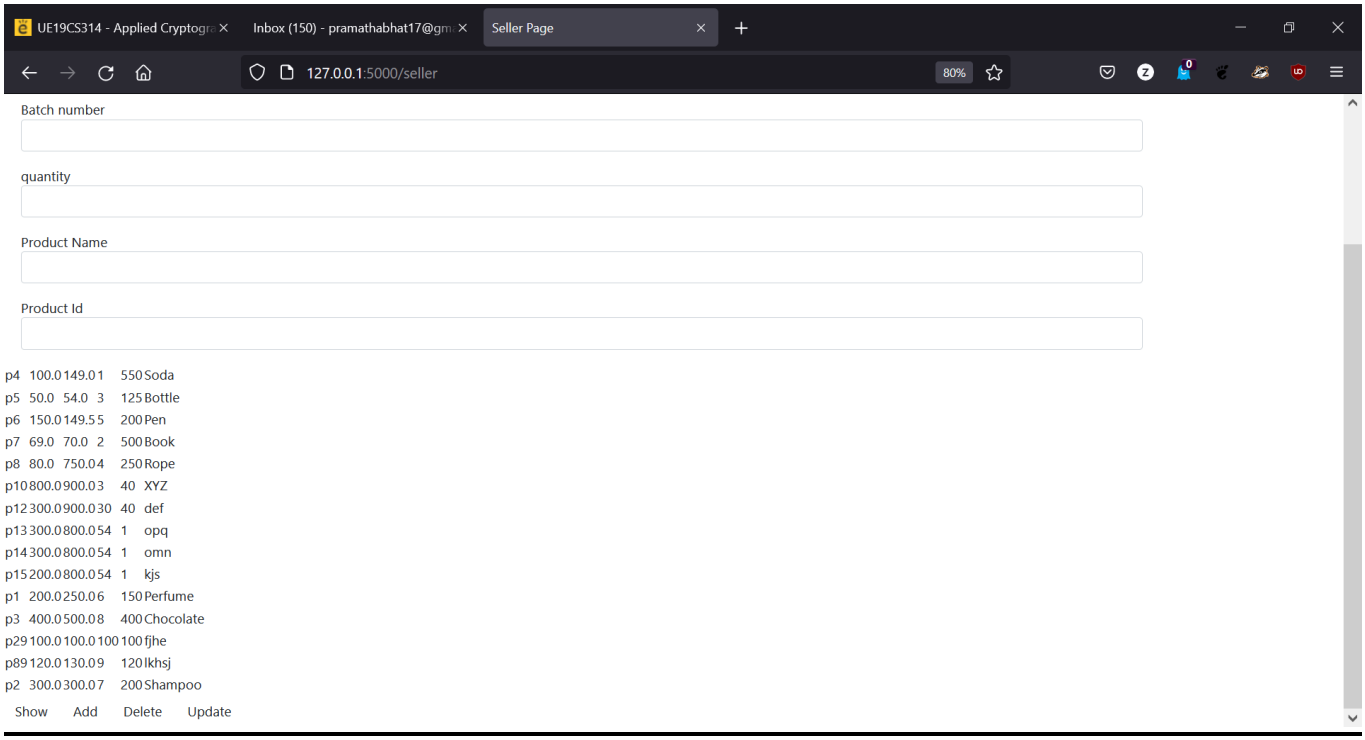


HomePage

Navigate from here

Home	Seller	Inventory





We made following changes to the schema and constraints:

Altered the customer and seller table by adding CHECK constaint to column gender.

```
store=# Alter table customer ADD CONSTRAINT gender_constraint CHECK(gender= 'F' or gender = 'M');
ALTER TABLE
store=# \d customer
                    Table "public.customer"
 Column  |         Type          | Collation | Nullable | Default
---------+-----------------------+-----------+----------+---------
 cust_id | character varying(15) |           | not null |
 name    | character varying(15) |           | not null |
 address | character varying(30) |           |          |
 phn_no  | character varying(13) |           | not null |
 age     | integer               |           |          |
 gender  | character(1)          |           | not null |
Indexes:
    "customer_pkey" PRIMARY KEY, btree (cust_id)
Check constraints:
    "gender_constraint" CHECK (gender = 'F'::bpchar OR gender = 'M'::bpchar)
Referenced by:
    TABLE "bill" CONSTRAINT "bill_cust_id_fkey" FOREIGN KEY (cust_id) REFERENCES customer(cust_id)
    TABLE "customer_buys" CONSTRAINT "customer_buys_c_id_fkey" FOREIGN KEY (c_id) REFERENCES customer(cust_id)


store=#
```

```
store=# Alter table seller ADD CONSTRAINT gender_constraint CHECK(gender= 'F' or gender = 'M');
ALTER TABLE
store=# \d seller
                     Table "public.seller"
  Column   |         Type          | Collation | Nullable | Default
-----------+-----------------------+-----------+----------+---------
 seller_id | character varying(15) |           | not null |
 name      | character varying(15) |           | not null |
 address   | character varying(30) |           |          |
 age       | integer               |           |          |
 gender    | character(1)          |           | not null |
Indexes:
    "seller_pkey" PRIMARY KEY, btree (seller_id)
Check constraints:
    "gender_constraint" CHECK (gender = 'F'::bpchar OR gender = 'M'::bpchar)
    "seller_age_check" CHECK (age > 25)
Referenced by:
    TABLE "seller_phnno" CONSTRAINT "seller_phnno_s_id_fkey" FOREIGN KEY (s_id) REFERENCES seller(seller_id)
    TABLE "seller_sells" CONSTRAINT "seller_sells_s_id_fkey" FOREIGN KEY (s_id) REFERENCES seller(seller_id)
```

Created seller_email table and added foreign key constraint to it .

Altered the table customer By adding column email if email column does not exists and added UNIQUE constraint to it.

```
store=# ALTER table customer ADD COLUMN if NOT EXISTS email varchar(50);
ALTER TABLE
store=# ALTER table customer ADD CONSTRAINT unique_email_address UNIQUE(email);
ALTER TABLE
store=# \d customer
                    Table "public.customer"
 Column  |         Type          | Collation | Nullable | Default
---------+-----------------------+-----------+----------+---------
 cust_id | character varying(15) |           | not null |
 name    | character varying(15) |           | not null |
 address | character varying(30) |           |          |
 phn_no  | character varying(13) |           | not null |
 age     | integer               |           |          |
 gender  | character(1)          |           | not null |
 email   | character varying(50) |           |          |
Indexes:
    "customer_pkey" PRIMARY KEY, btree (cust_id)
    "unique_email_address" UNIQUE CONSTRAINT, btree (email)
Check constraints:
    "gender_constraint" CHECK (gender = 'F'::bpchar OR gender = 'M'::bpchar)
Referenced by:
    TABLE "bill" CONSTRAINT "bill_cust_id_fkey" FOREIGN KEY (cust_id) REFERENCES customer(cust_id)
    TABLE "customer_buys" CONSTRAINT "customer_buys_c_id_fkey" FOREIGN KEY (c_id) REFERENCES customer(cust_id)


store=#
```

Altered the table bill by changing datatype of column bill_date to Timestamp

```
store=# ALTER TABLE bill
store-# ALTER COLUMN bill_date TYPE timestamp;
ALTER TABLE
store=# \d bill
                          Table "public.bill"
     Column     |            Type             | Collation | Nullable | Default
----------------+-----------------------------+-----------+----------+---------
 bill_id        | character varying(15)       |           | not null |
 bill_date      | timestamp without time zone |           |          |
 bill_total     | double precision            |           |          |
 prod_purchased | character varying(30)       |           |          |
 cust_id        | character varying(15)       |           | not null |
Indexes:
    "bill_pkey" PRIMARY KEY, btree (bill_id)
Foreign-key constraints:
    "bill_cust_id_fkey" FOREIGN KEY (cust_id) REFERENCES customer(cust_id)
Referenced by:
    TABLE "inventory" CONSTRAINT "inventory_bill_id1_fkey" FOREIGN KEY (bill_id1) REFERENCES bill(bill_id)


store=#
```

Alter table customer by setting default to column name.

```
store=# ALTER TABLE customer
store-# ALTER COLUMN name set default('xyz');
ALTER TABLE
store=# \d customer
                          Table "public.customer"
 Column  |         Type          | Collation | Nullable |         Default
---------+-----------------------+-----------+----------+-------------------------
 cust_id | character varying(15) |           | not null |
 name    | character varying(15) |           | not null | 'xyz'::character varying
 address | character varying(30) |           |          |
 phn_no  | character varying(13) |           | not null |
 age     | integer               |           |          |
 gender  | character(1)          |           | not null |
 email   | character varying(50) |           |          |
Indexes:
    "customer_pkey" PRIMARY KEY, btree (cust_id)
    "unique_email_address" UNIQUE CONSTRAINT, btree (email)
Check constraints:
    "gender_constraint" CHECK (gender = 'F'::bpchar OR gender = 'M'::bpchar)
Referenced by:
    TABLE "bill" CONSTRAINT "bill_cust_id_fkey" FOREIGN KEY (cust_id) REFERENCES customer(cust_id)
    TABLE "customer_buys" CONSTRAINT "customer_buys_c_id_fkey" FOREIGN KEY (c_id) REFERENCES customer(cust_id)


store=#
```

If we were to migrate to another database framework, we would use NoSQL's mongodb

PostgreSQL is one of the most popular and well-regarded open-source relational databases in the world. PostgreSQL possesses an incredible number of features related to performance, security, programming extensions, and configuration among others.Although there is nothing wrong with rdbms system,it cannot handle big data very efficiently thus making it very hard to use in this generation . Therefore more efficient way for migration is **RDBMS to NoSQL' s MongoDB**

Advantage of migrating from RDBMS to NoSQL's MongoDB

- Schema less − MongoDB is a document database in which one collection holds different documents. Number of fields, content and size of the document can differ from one document to another.
- No complex joins.
- Conversion/mapping of application objects to database objects not needed.
- Enabling faster access of data.

The process for transferring data from PostgreSQL to MongoDB

First we must prepare the application to migrate to mongoDB .Next we must export dbms data in a readable format .To migrate data , you'll extract it from PostSQL and then import it to MongoDB using the mongoimport tool

There are two different ways to extract the data

1. Running queries as tab separated values
2. Returning queries as JOSON