DevOps Final Project

AWS Infrastructure Deployment with Terraform, Docker, and BI Integration

Ali Muhammad – 18585

GitHub:

https://github.com/AliMuhammadAslam/DevOpsProject-AWSInfrastructureDeployment

Loom Video – Part 1:

https://www.loom.com/share/3ad97a14eb81482e811e2d18e2468b6b?sid=199816a0-d06d-409a-8abd-99d6d2e8cdc1

Loom Video – Part 2:

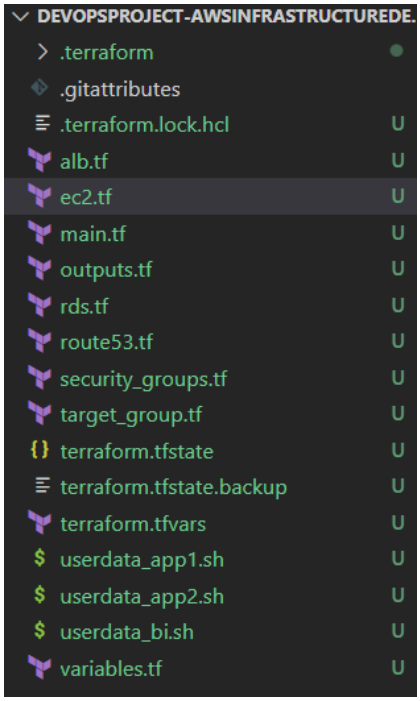
https://www.loom.com/share/3ca357988b1f46c8bafacd3147dee49e?sid=c7e93728-74c4-4e8f-9a00-6ce1f9ce2cf2

Introduction:

This project focuses on the integration of cloud-based infrastructure and data visualization tools to analyze real-time data. By utilizing AWS services, including EC2, RDS, and Application Load Balancer (ALB), we have designed and deployed a scalable architecture that securely handles data storage and retrieval. The primary objective of this project is to demonstrate the effective use of PostgreSQL for data management and Metabase for dynamic data visualization. Through the use of real-world data, this project emphasizes the value of interactive dashboards and efficient database management in driving business intelligence and informed decision-making. By implementing these systems in a cloud environment, the project highlights how modern infrastructure and visualization tools can enhance data accessibility and analysis.

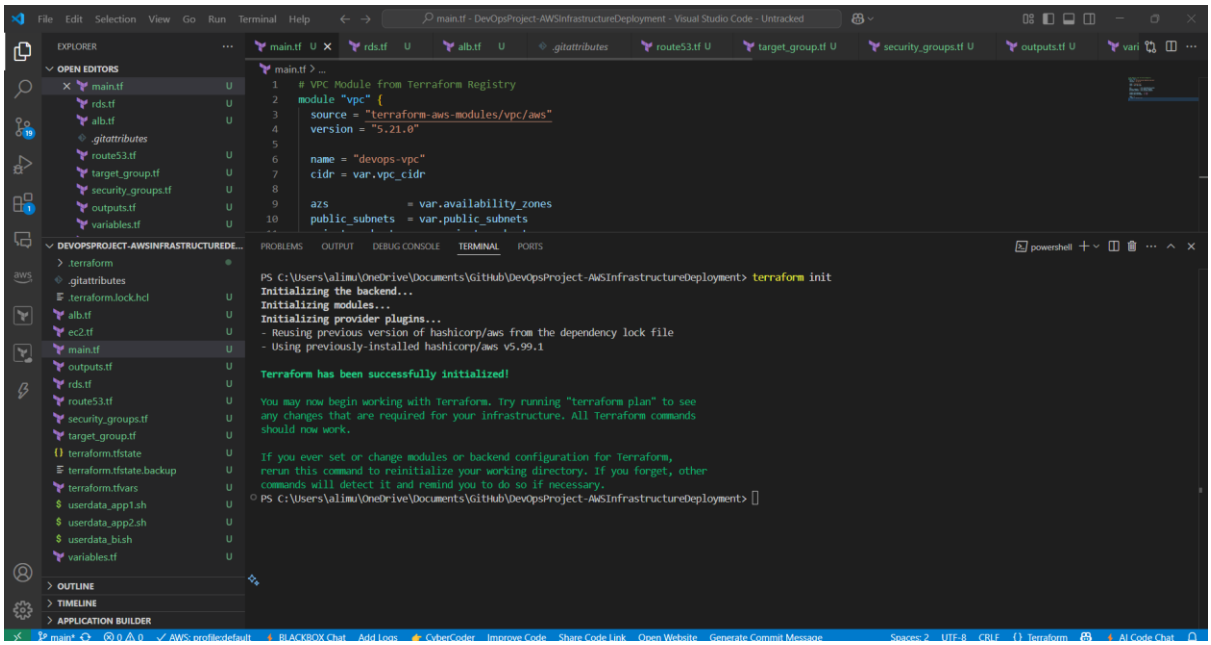
Terraform File Structure:

The Terraform project structure includes several custom files that I have created to configure the required AWS infrastructure. **alb.tf** defines the configuration for the Application Load Balancer (ALB), specifying the rules and routing for traffic. **ec2.tf** configures the EC2 instances, detailing their types, key pairs, and security groups. **outputs.tf** sets the output values to be displayed after deployment, such as IP addresses or URLs. **rds.tf** contains the configuration for the Amazon RDS database, including instance type and settings. **route53.tf** configures DNS records with Route 53 to map domain names to AWS resources. **security_groups.tf** defines the security rules for EC2 instances and other resources. **target_group.tf** defines the target groups for the load balancer, ensuring traffic is directed to healthy instances. **terraform.tfvars** holds variable values to be used across the configuration. **userdata_app1.sh** and **userdata_app2.sh** automate the setup of the respective EC2 instances, while **userdata_bi.sh** configures the BI tool instance. **variables.tf** defines the variables used throughout the configuration, allowing for easy adjustments.

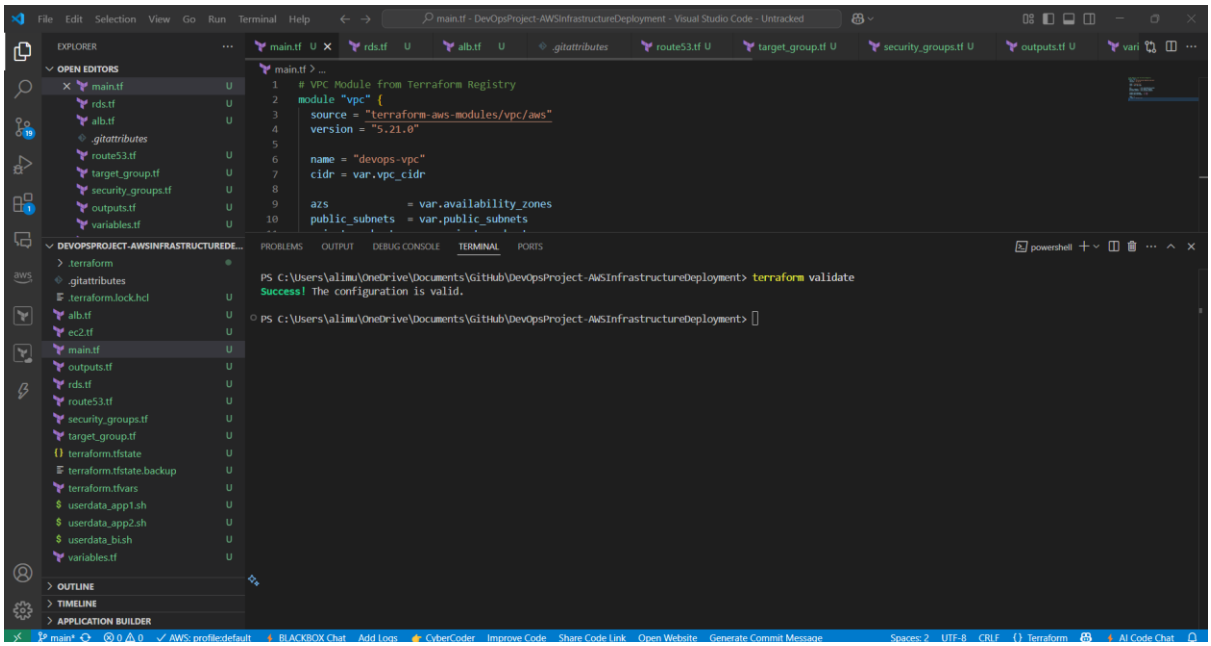


Execution of Terraform Commands:

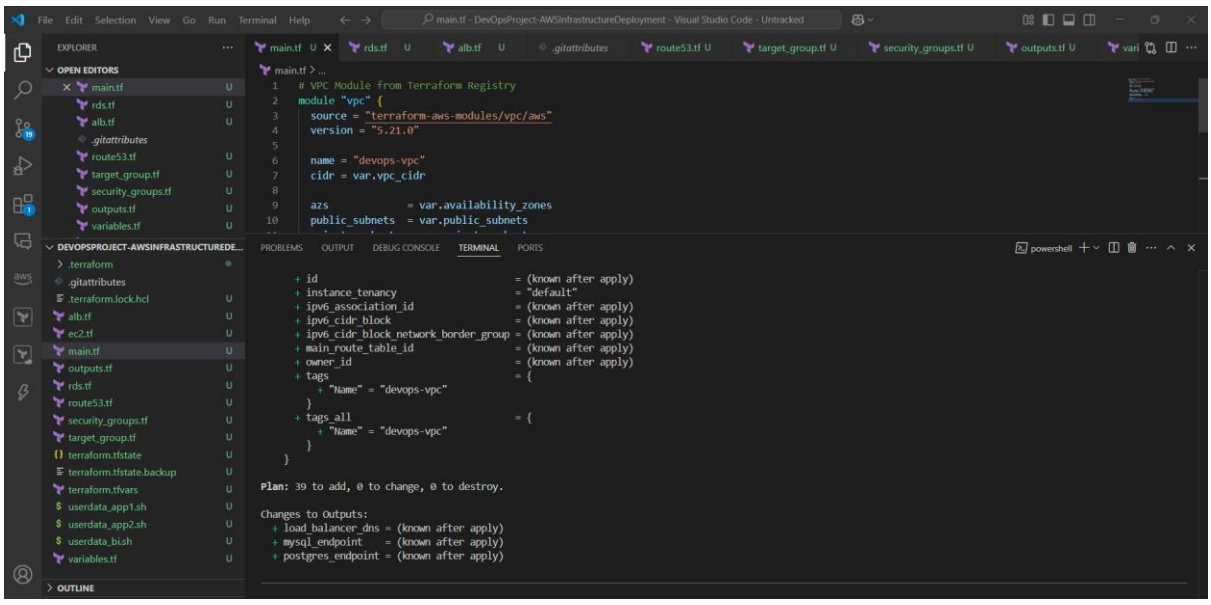
Terraform init (Initializes the working directory by downloading necessary provider plugins and setting up the backend for Terraform):



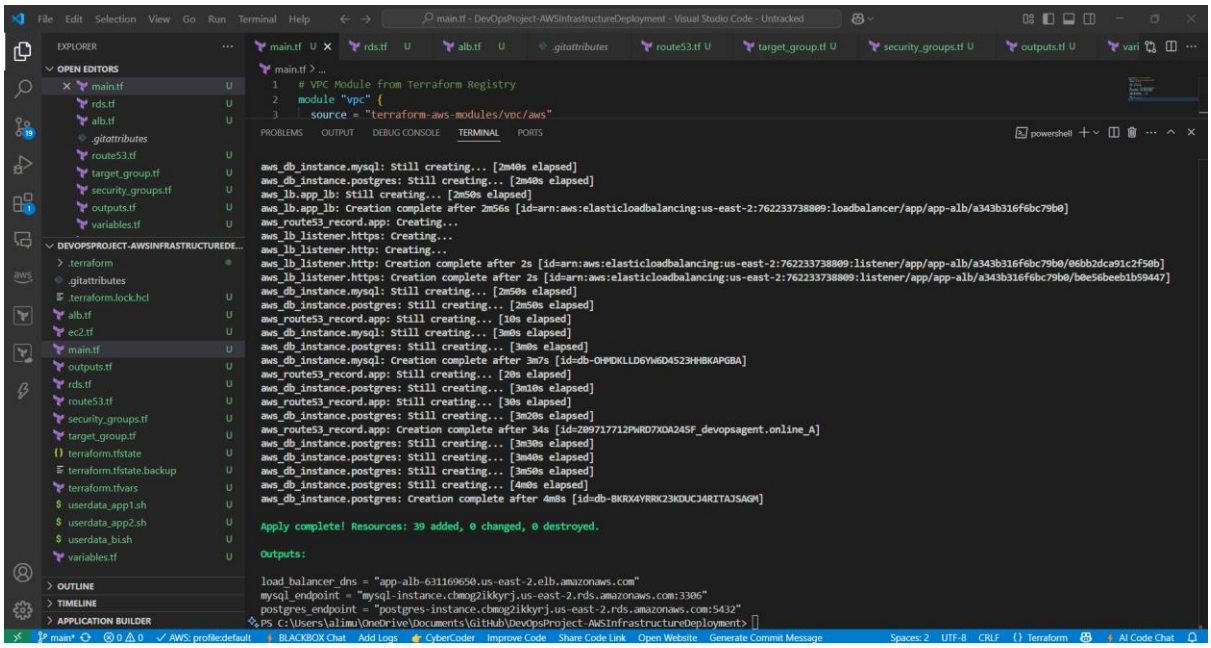
Terraform validate (Checks the syntax and configuration of the Terraform files for correctness, ensuring no errors before applying changes):



Terraform Plan (Creates an execution plan by comparing the current state of the infrastructure with the desired configuration, showing the changes that will be made):

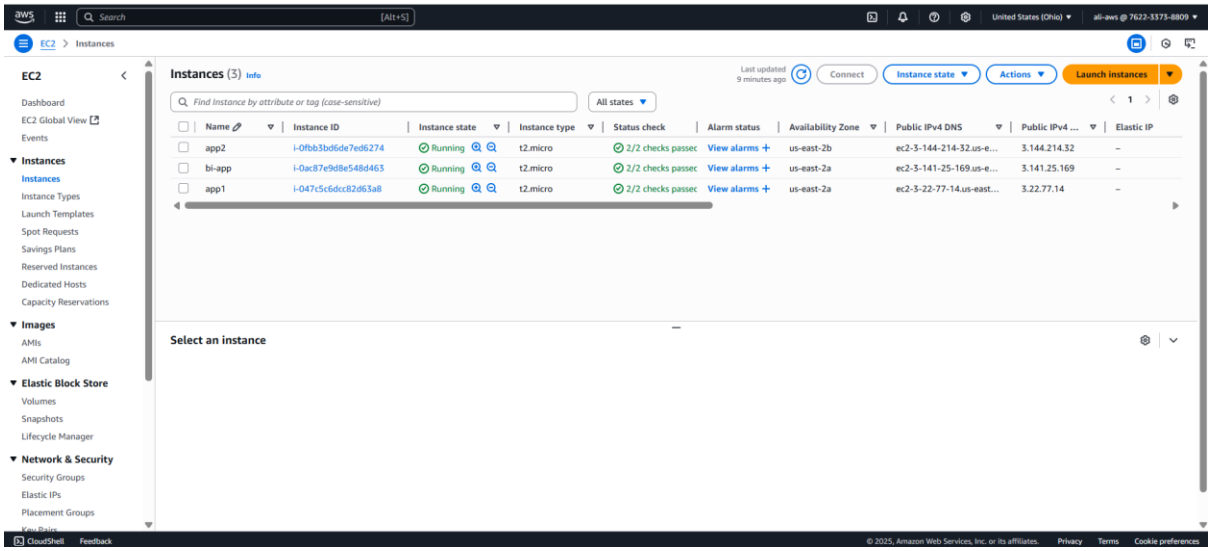


Terraform Apply (Applies the changes defined in the execution plan to the infrastructure, provisioning or modifying resources as necessary):

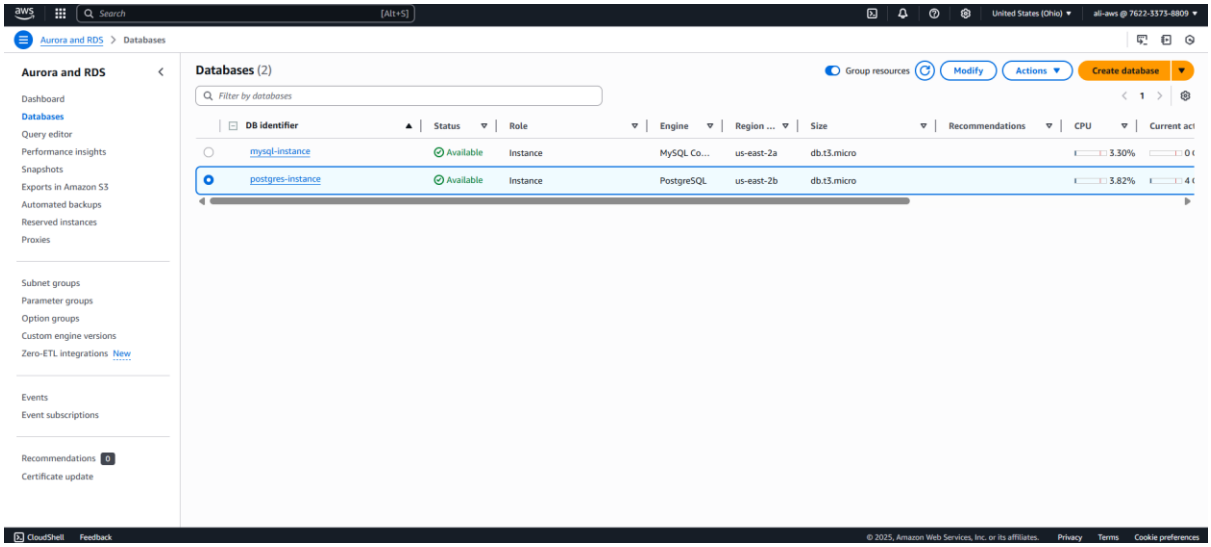


AWS Console – Infrastructure Provisioned:

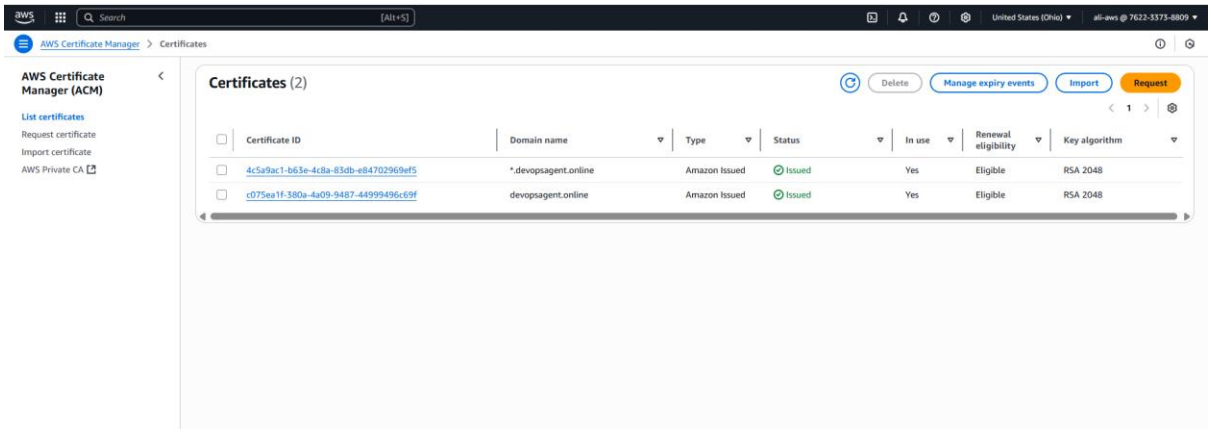
EC 2 Instances Created:



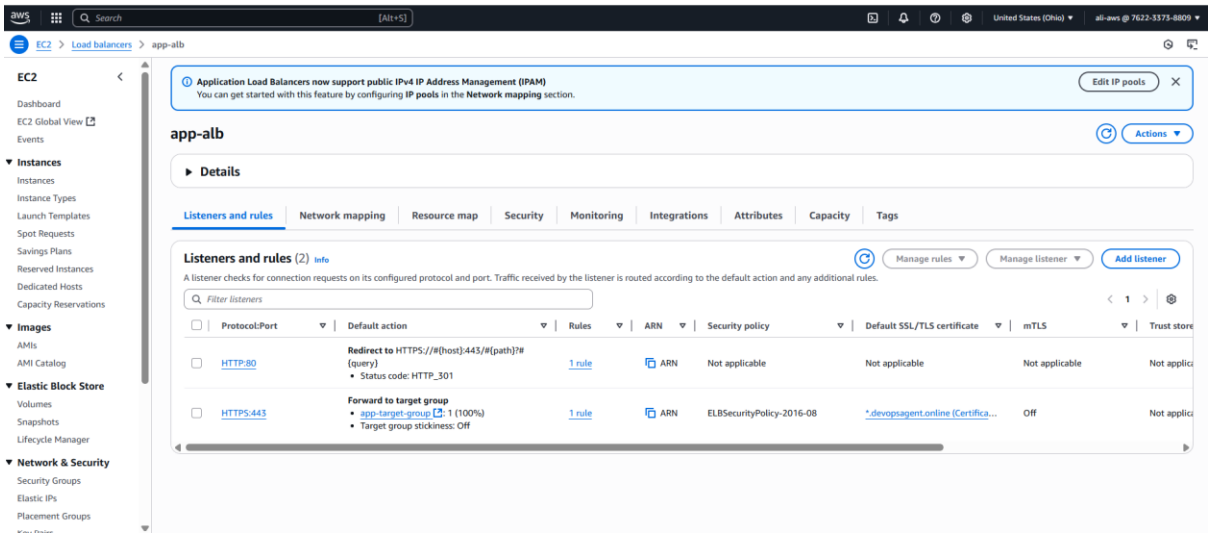
RDS Instances Created:



Wild card certificate issued - *.devopsagent.online:

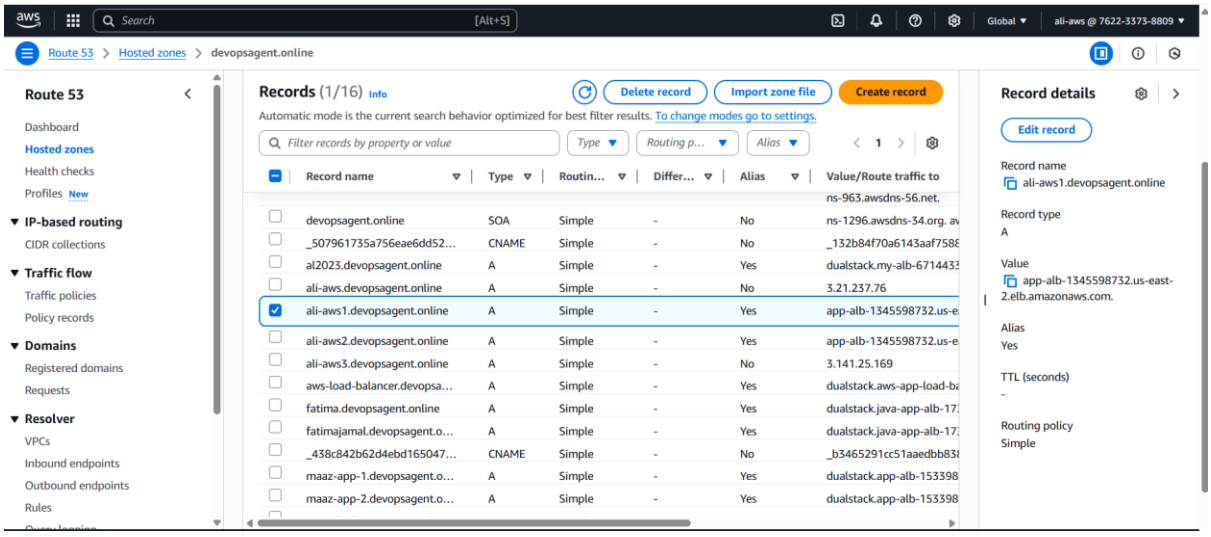


Load Balancer provisioned with required listeners and rules:

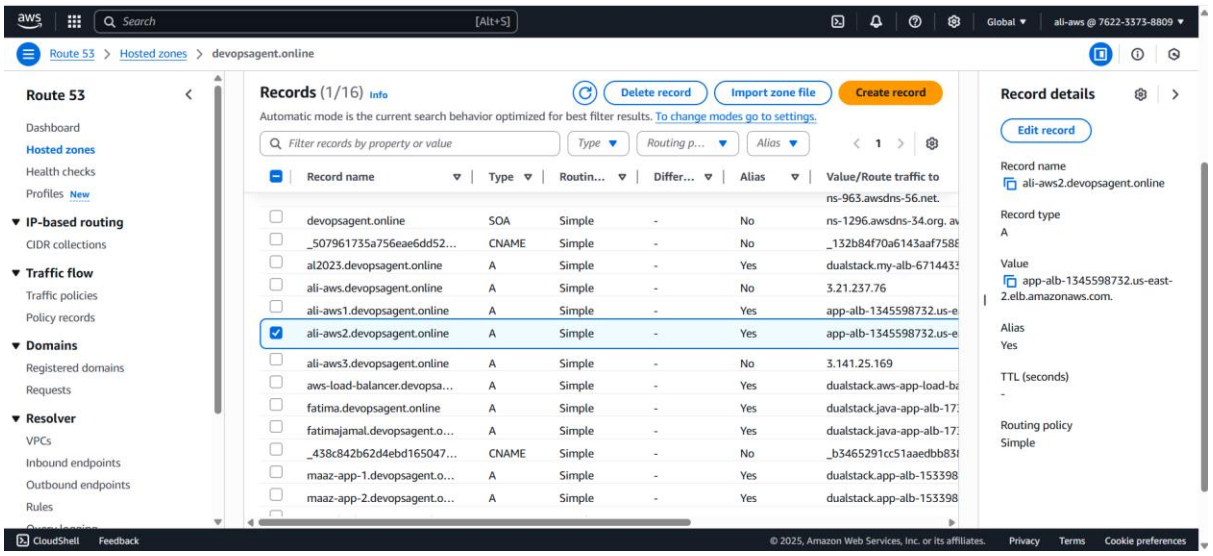


ALB and IP Values mapped with the domain names using Route 53:

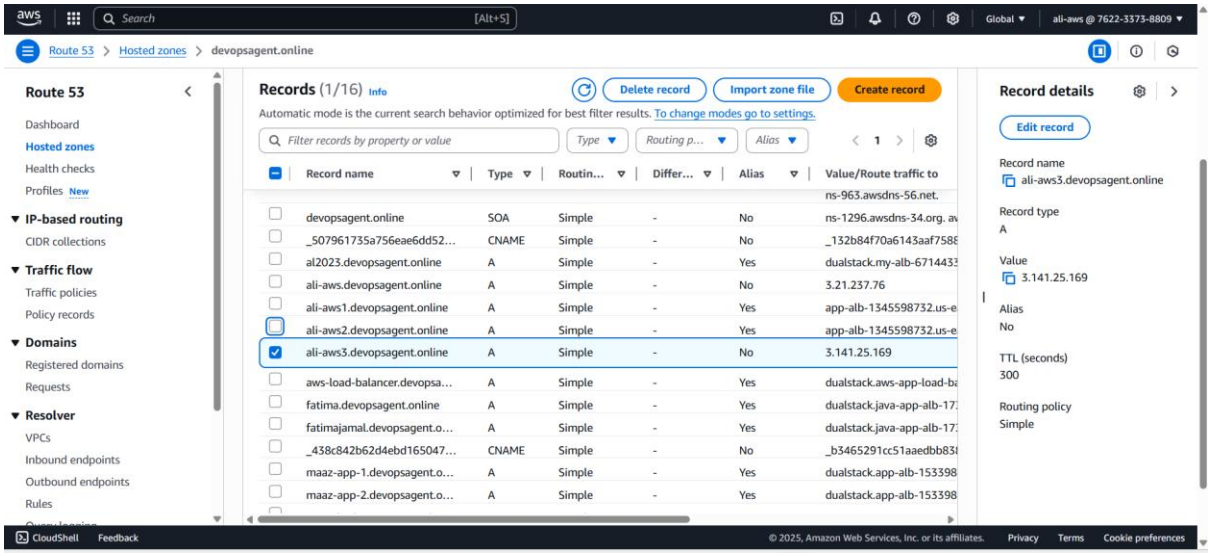
App 1:



App 2:

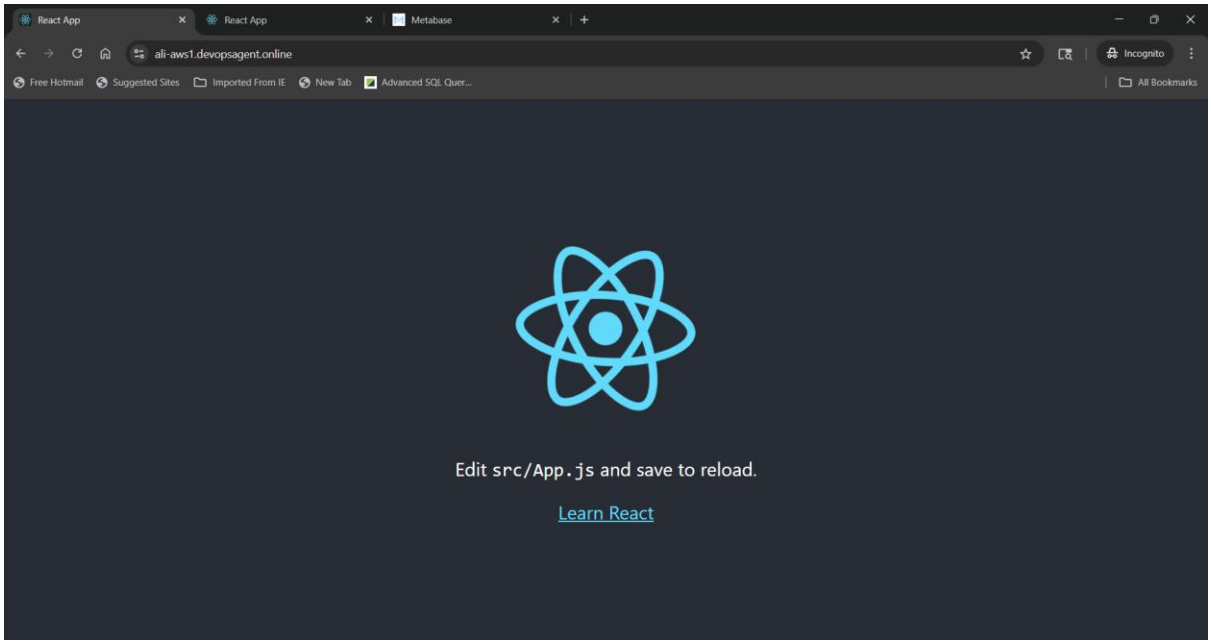


BI-APP:

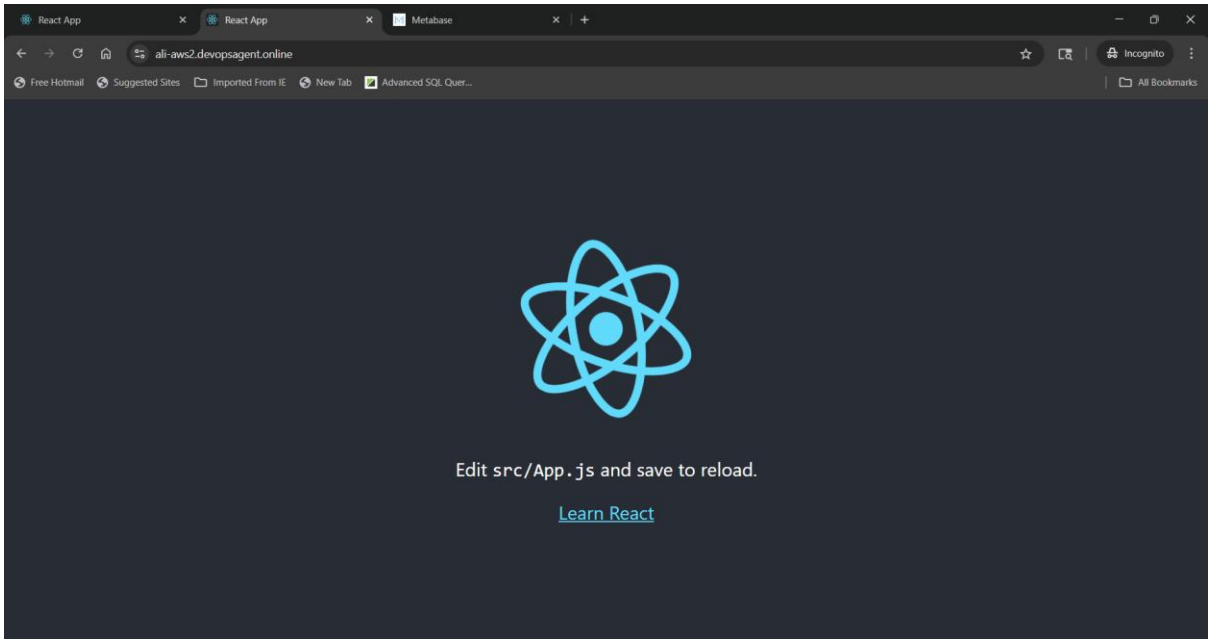


Accessing Apps (SSL Enabled):

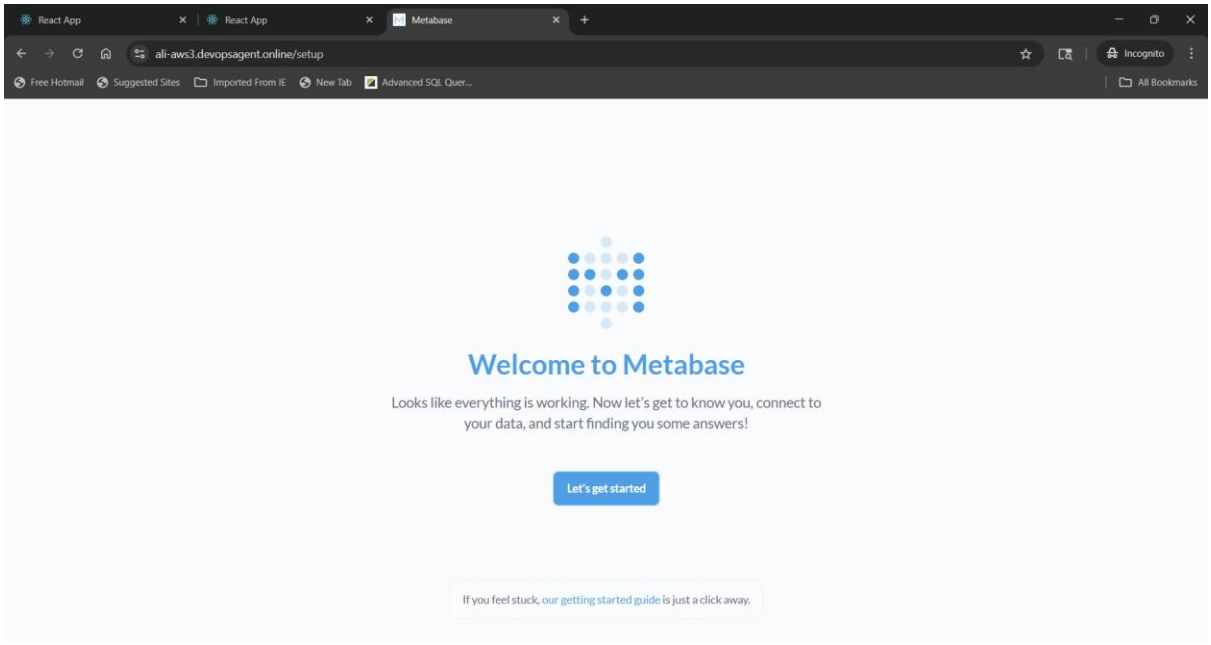
APP 1:


React App
React App
Metabase
ali-aws1.devopsagent.online
Incognito
Free Hotmail
Suggested Sites
Imported From IE
New Tab
Advanced SQL Quer...
All Bookmarks
Edit src/App.js and save to reload.
Learn React

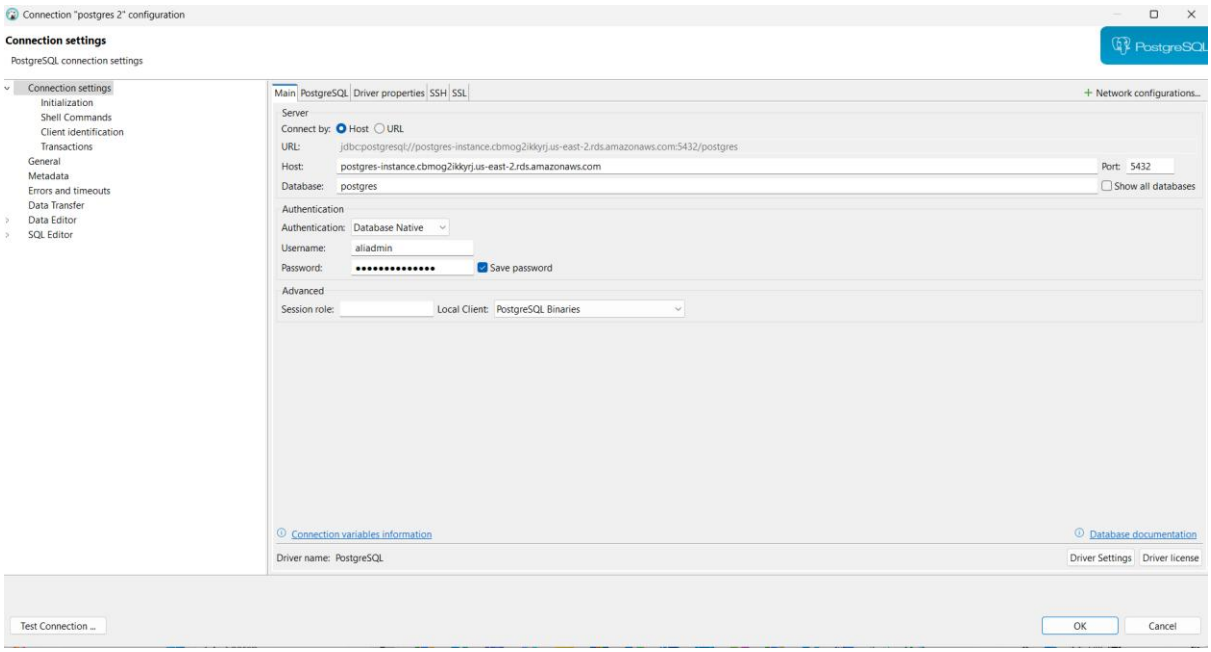
APP 2:

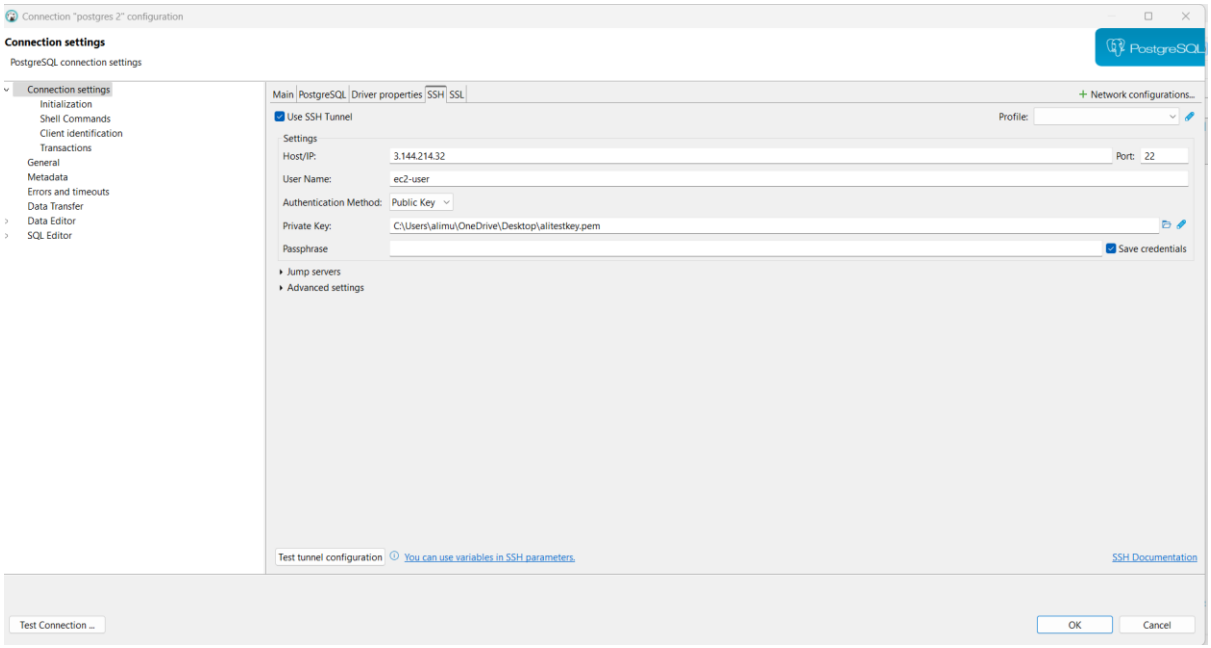

React App
React App
Metabase
ali-aws2.devopsagent.online
Incognito
Free Hotmail
Suggested Sites
Imported From IE
New Tab
Advanced SQL Quer...
All Bookmarks
Edit src/App.js and save to reload.
Learn React

BI-APP:



D Beaver Configurations to establish Connection:





Script to create table:

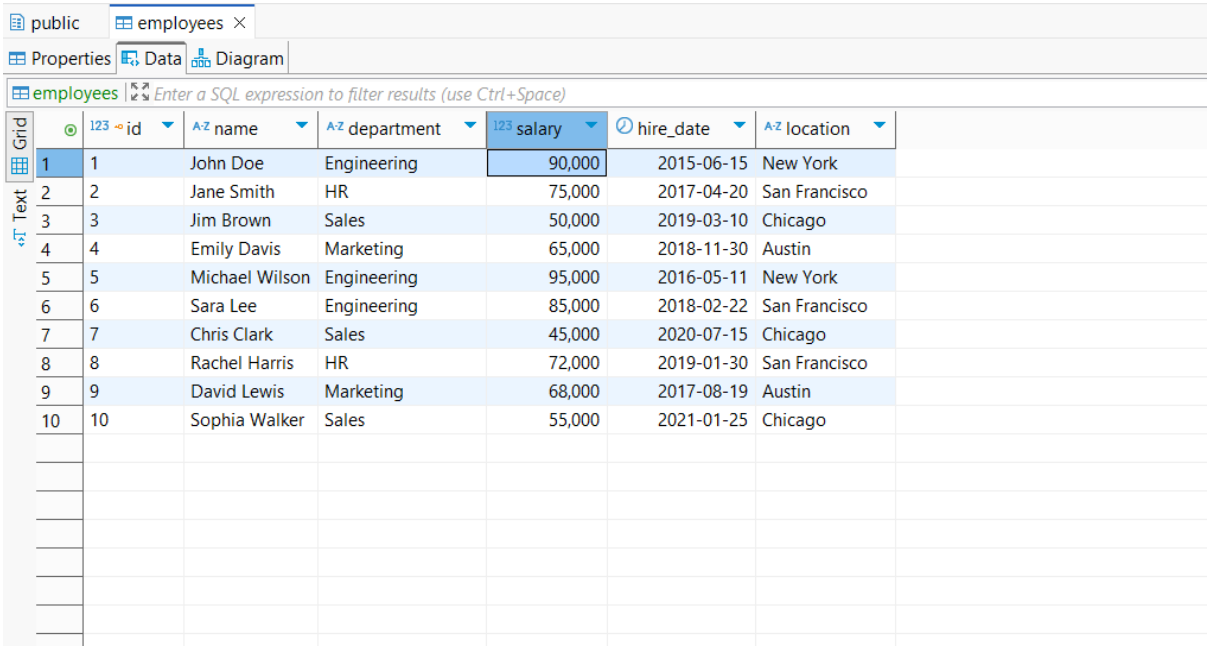
```sql
DROP TABLE IF EXISTS employees;

CREATE TABLE employees (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100),
    department VARCHAR(50),
    salary INT,
    hire_date DATE,
    location VARCHAR(100)
);
```

Script to insert record into table:

```sql
INSERT INTO employees (name, department, salary, hire_date, location) VALUES
('John Doe', 'Engineering', 90000, '2015-06-15', 'New York'),
('Jane Smith', 'HR', 75000, '2017-04-20', 'San Francisco'),
('Jim Brown', 'Sales', 50000, '2019-03-10', 'Chicago'),
('Emily Davis', 'Marketing', 65000, '2018-11-30', 'Austin'),
('Michael Wilson', 'Engineering', 95000, '2016-05-11', 'New York'),
('Sara Lee', 'Engineering', 85000, '2018-02-22', 'San Francisco'),
('Chris Clark', 'Sales', 45000, '2020-07-15', 'Chicago'),
('Rachel Harris', 'HR', 72000, '2019-01-30', 'San Francisco'),
('David Lewis', 'Marketing', 68000, '2017-08-19', 'Austin'),
('Sophia Walker', 'Sales', 55000, '2021-01-25', 'Chicago');
```

Employees Table Created:



public | employees

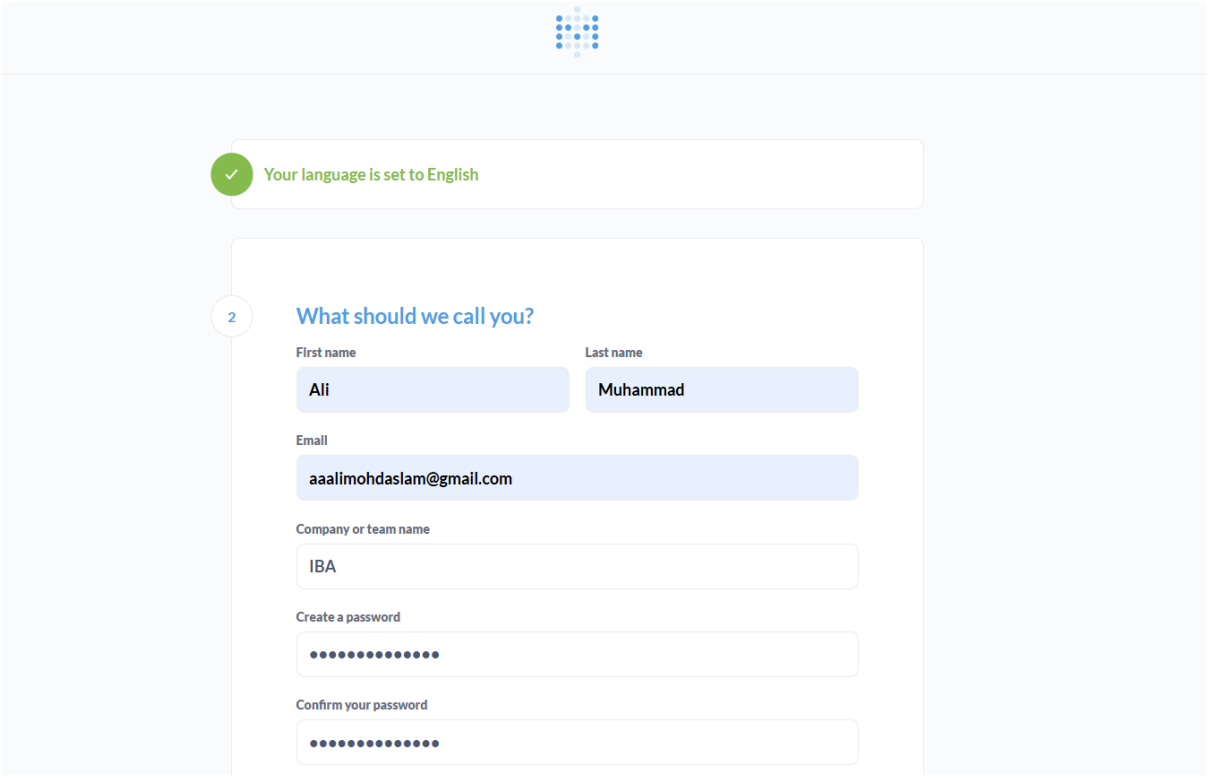
Properties | Data | Diagram

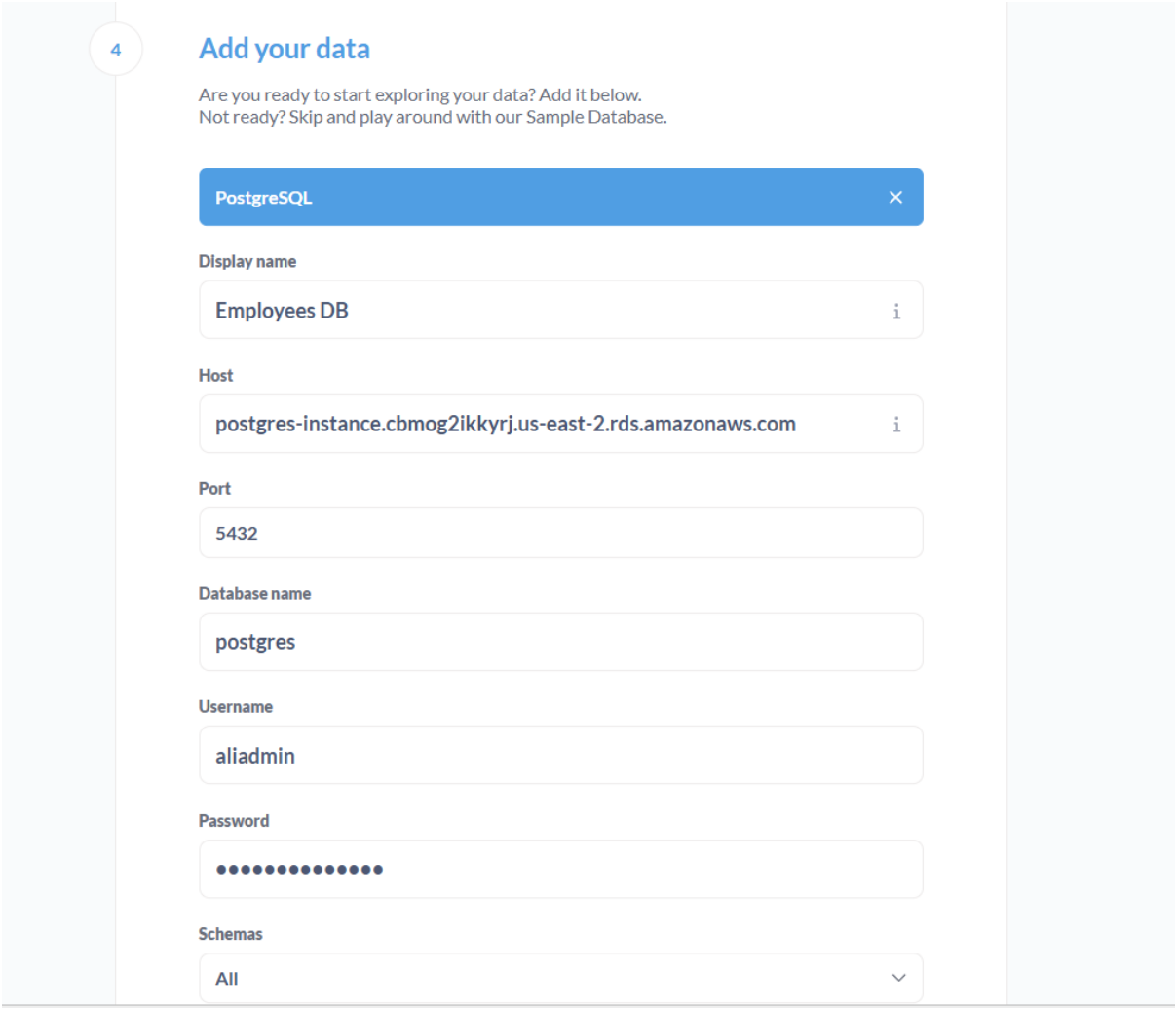
employees | Enter a SQL expression to filter results (use Ctrl+Space)

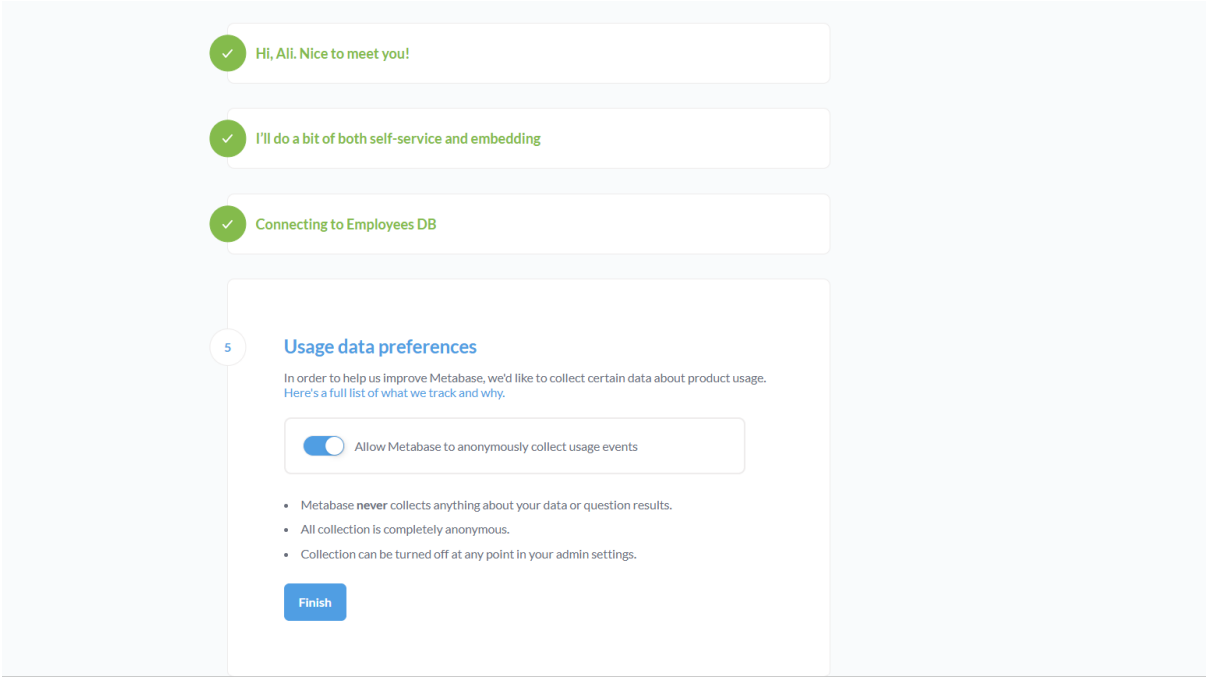
Grid | Text

	id	name	department	salary	hire_date	location
1	1	John Doe	Engineering	90,000	2015-06-15	New York
2	2	Jane Smith	HR	75,000	2017-04-20	San Francisco
3	3	Jim Brown	Sales	50,000	2019-03-10	Chicago
4	4	Emily Davis	Marketing	65,000	2018-11-30	Austin
5	5	Michael Wilson	Engineering	95,000	2016-05-11	New York
6	6	Sara Lee	Engineering	85,000	2018-02-22	San Francisco
7	7	Chris Clark	Sales	45,000	2020-07-15	Chicago
8	8	Rachel Harris	HR	72,000	2019-01-30	San Francisco
9	9	David Lewis	Marketing	68,000	2017-08-19	Austin
10	10	Sophia Walker	Sales	55,000	2021-01-25	Chicago

Metabase Setup:




4
Add your data
Are you ready to start exploring your data? Add it below.
Not ready? Skip and play around with our Sample Database.
PostgreSQL
Display name
Employees DB
Host
postgres-instance.cbmog2ikkyrj.us-east-2.rds.amazonaws.com
Port
5432
Database name
postgres
Username
aliadmin
Password
Schemas
All


Hi, Ali. Nice to meet you!
I'll do a bit of both self-service and embedding
Connecting to Employees DB
5
Usage data preferences
In order to help us improve Metabase, we'd like to collect certain data about product usage.
Here's a full list of what we track and why.
Allow Metabase to anonymously collect usage events
Metabase never collects anything about your data or question results.
All collection is completely anonymous.
Collection can be turned off at any point in your admin settings.
Finish

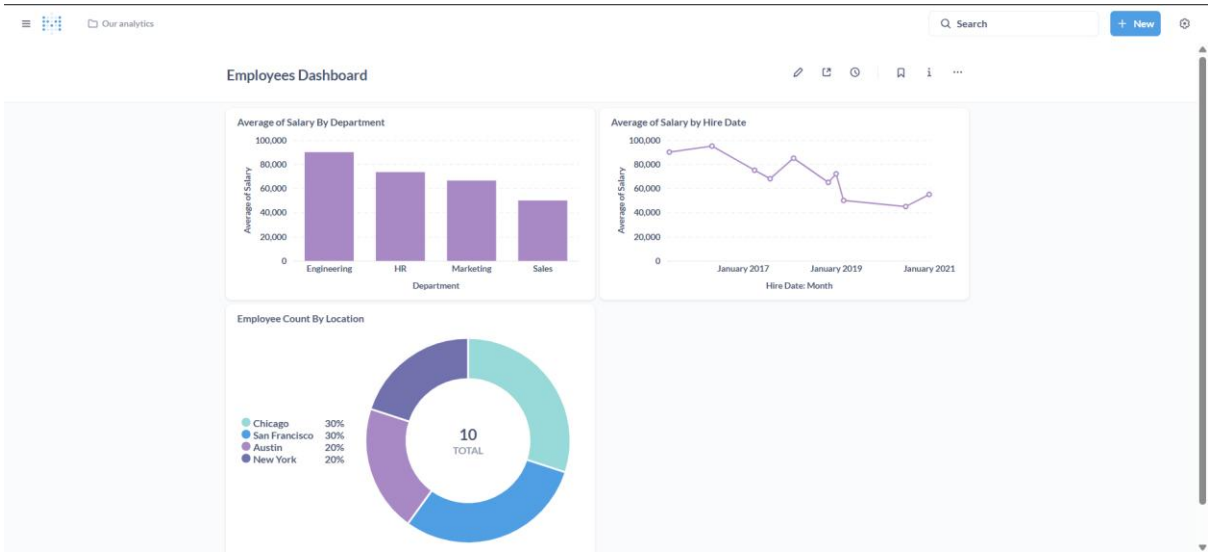
Employees Table Accessible Through Metabase:

Employees DB / **Employees**

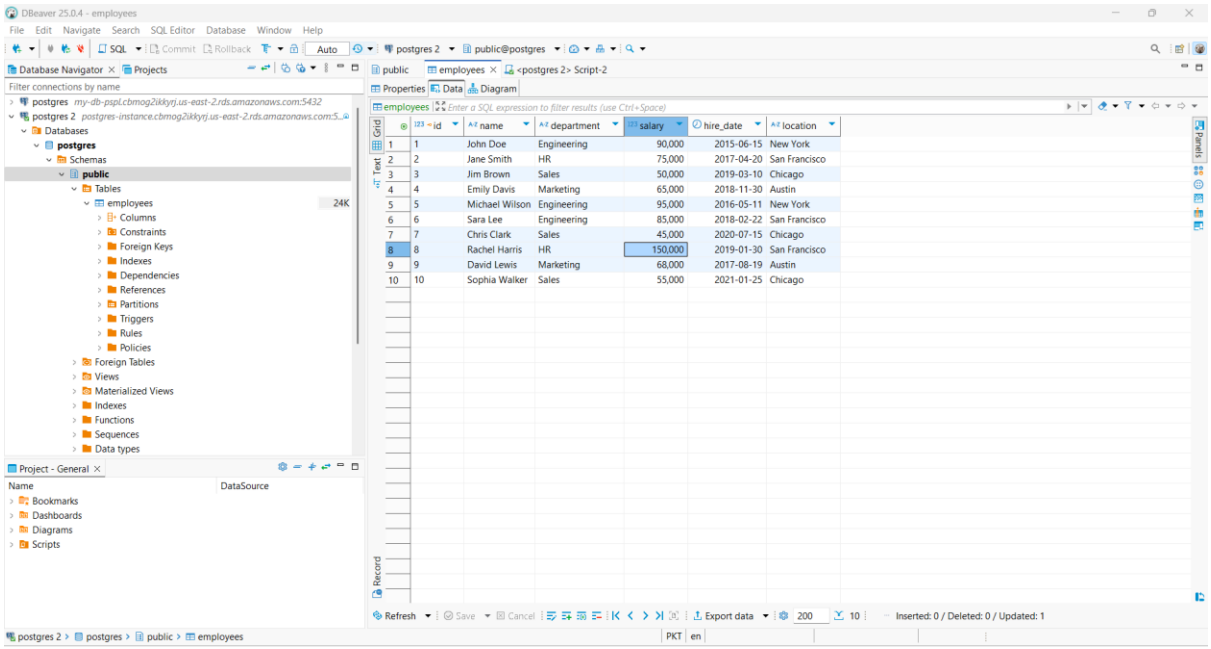
ID	Name	Department	Salary	Hire Date	Location
1	John Doe	Engineering	90,000	June 15, 2015	New York
2	Jane Smith	HR	75,000	April 20, 2017	San Francisco
3	Jim Brown	Sales	50,000	March 10, 2019	Chicago
4	Emily Davis	Marketing	65,000	November 30, 2018	Austin
5	Michael Wilson	Engineering	95,000	May 11, 2016	New York
6	Sara Lee	Engineering	85,000	February 22, 2018	San Francisco
7	Chris Clark	Sales	45,000	July 15, 2020	Chicago
8	Rachel Harris	HR	72,000	January 30, 2019	San Francisco
9	David Lewis	Marketing	68,000	August 19, 2017	Austin
10	Sophia Walker	Sales	55,000	January 25, 2021	Chicago

Visualization

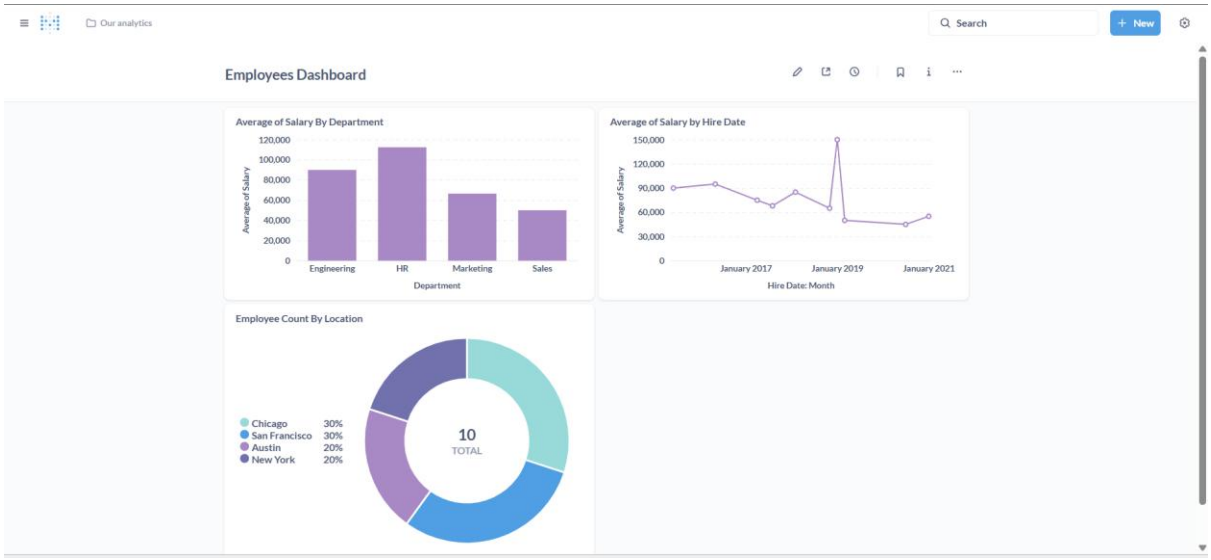
Employees Dashboard Created:



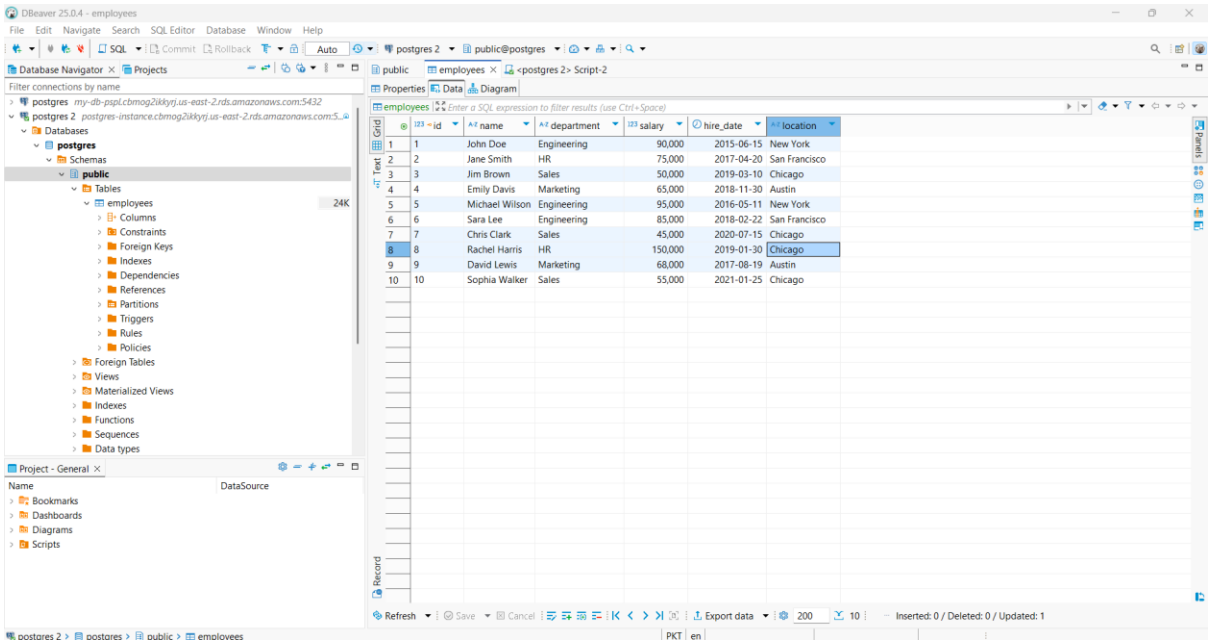
Editing a record (Rachel Harris Salary) in D Beaver:



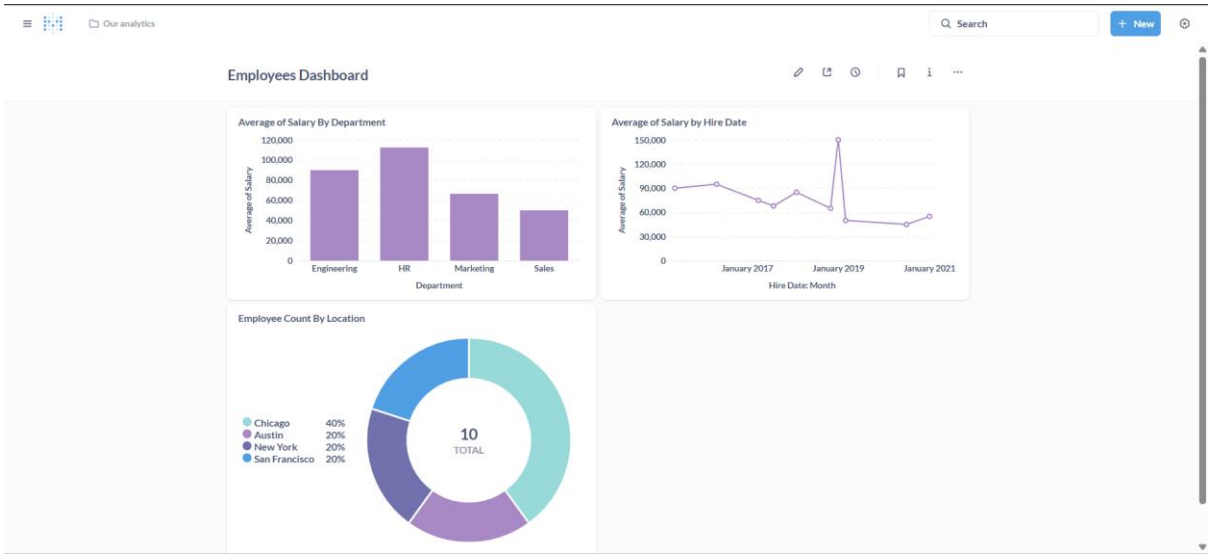
Dashboard in Metabase updated in Real Time (Average Salary of HR increased – Bar Chart):



Editing a record (Rache Harris location) in D Beaver:



Dashboard in Metabase updated in Real Time (Count of Chicago Increased and San Francisco Decreased):



Destroyed Resources Provisioned By Terraform (Ran terraform destroy command):