DBMS Mini Project Report

Space Database

Pranav Ambiga : PES1UG21CS434

Purab N : PES1UG21CS459

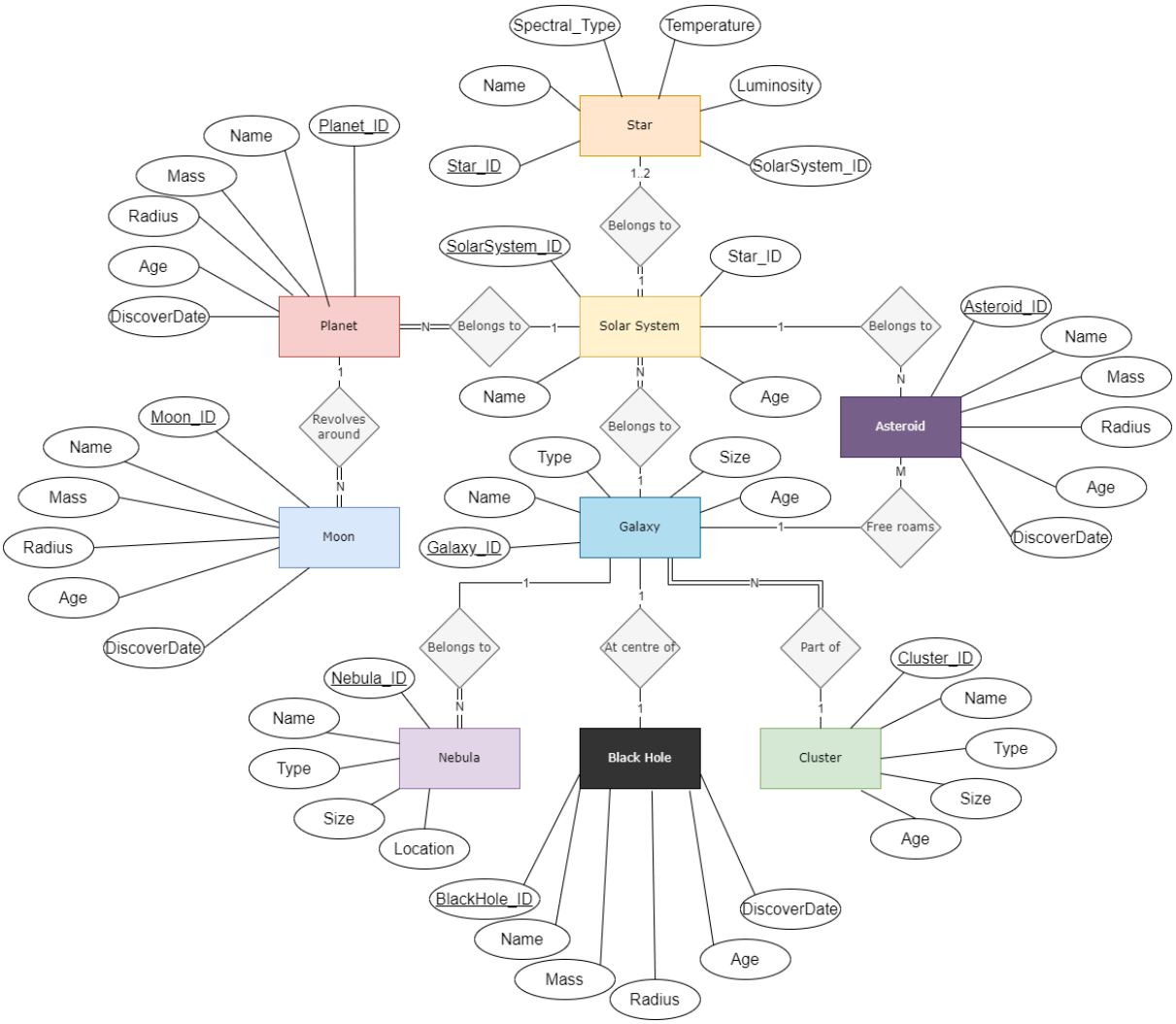
Abstract

Cosmo is an immersive and educational space exploration website designed to captivate users with the wonders of the cosmos. Offering a comprehensive journey through our galaxy and beyond, the website provides detailed information on various celestial bodies, including planets, moons, stars, asteroids, and galaxies. Users can navigate through visually stunning representations of the universe, access real-time data on astronomical phenomena, and engage in interactive learning experiences. With features like Dynamic queries, and a user-friendly interface, Cosmo transforms complex astrophysical concepts into an accessible and engaging learning adventure. Whether a casual stargazer or a budding astronomer, users of all ages can embark on a captivating journey to deepen their understanding of the vastness and beauty of the cosmos.

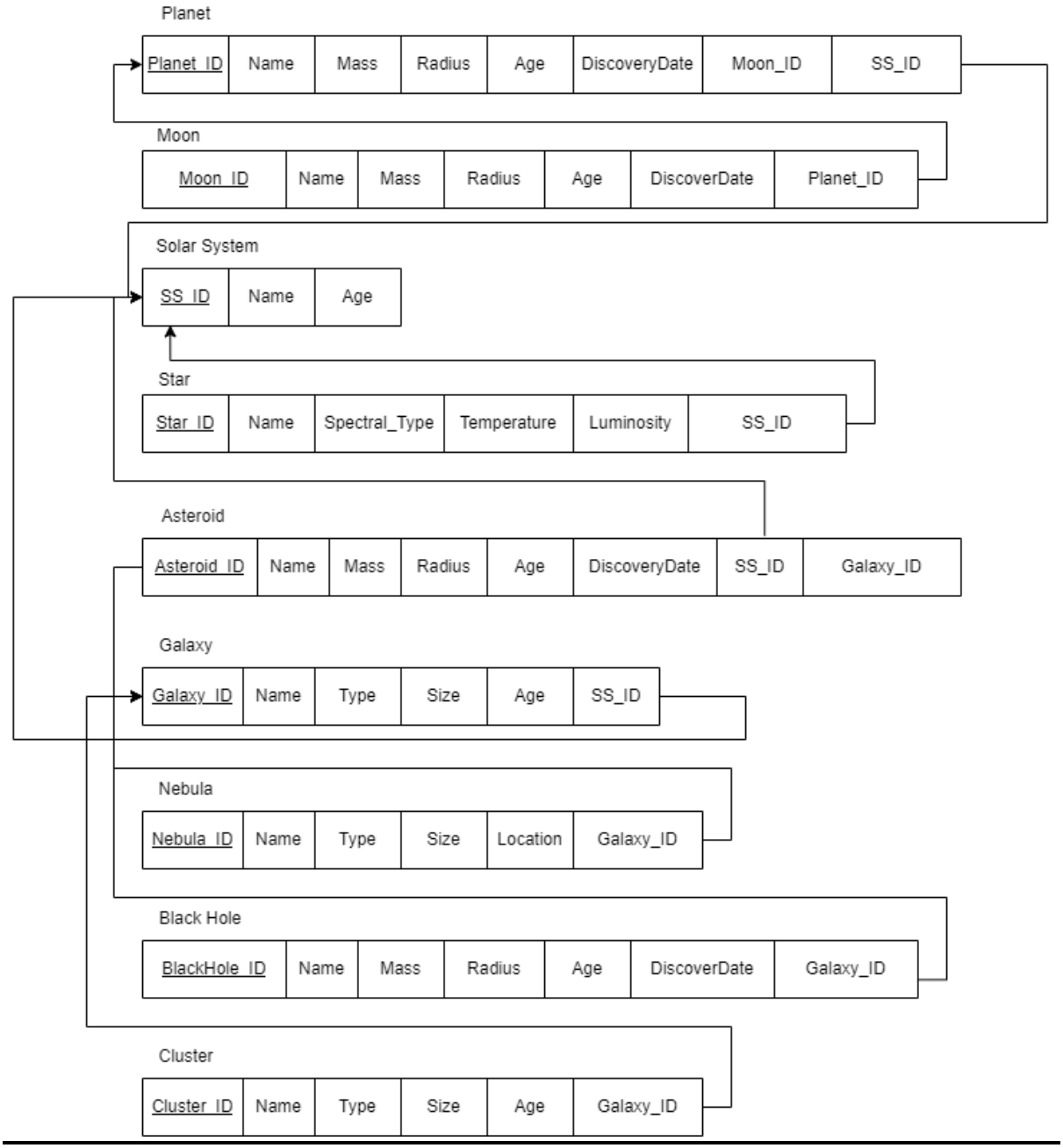
Technologies Used

1. MySQL
2. Flask
3. Python
4. React
5. HTML, CSS, JS

ER Diagram


Spectral_Type
Temperature
Name
Luminosity
Star
Star_ID
SolarSystem_ID
1..2
Belongs to
SolarSystem_ID
Star_ID
Name
Planet_ID
Mass
Radius
Age
DiscoverDate
Planet
N
Belongs to
1
Solar System
1
Belongs to
Asteroid_ID
Name
Mass
Radius
Age
DiscoverDate
Asteroid
N
M
Name
Age
1
N
Revolves around
Belongs to
Moon_ID
Name
Mass
Radius
Age
DiscoverDate
Moon
Type
Size
Name
Age
Galaxy_ID
Galaxy
Free roams
1
N
Belongs to
At centre of
Part of
Nebula_ID
Name
Type
Size
Location
Nebula
Black Hole
Cluster_ID
Name
Type
Size
Age
Cluster
BlackHole_ID
Name
Mass
Radius
Age
DiscoverDate

Relational Schema



DDL Queries

```sql
-- SolarSystems Table
CREATE TABLE SolarSystems (
    SolarSystemID INT PRIMARY KEY,
    Name VARCHAR(255),
    Age INT,
    StarID INT,
    FOREIGN KEY (StarID) REFERENCES Star(StarID)
);

-- Planets Table
CREATE TABLE Planet (
    PlanetID INT PRIMARY KEY,
    Name VARCHAR(255),
    Mass DECIMAL(18, 2),
    Radius DECIMAL(18, 2),
    Age INT,
    DiscoveryDate DATE,
    MoonID INT,
    SolarSystemID INT,
    FOREIGN KEY (SolarSystemID) REFERENCES SolarSystems(SolarSystemID)
);

-- Moons Table
CREATE TABLE Moon (
    MoonID INT PRIMARY KEY,
    Name VARCHAR(255),
    Mass DECIMAL(18, 2),
    Radius DECIMAL(18, 2),
    Age INT,
    DiscoveryDate DATE,
    PlanetID INT,
    FOREIGN KEY (PlanetID) REFERENCES Planet(PlanetID)
);

-- Stars Table
CREATE TABLE Star (
    StarID INT PRIMARY KEY,
    Name VARCHAR(255),
    SpectralType VARCHAR(10),
    Temperature INT,
    Luminosity DECIMAL(18, 2),
    SolarSystemID INT,
    FOREIGN KEY (SolarSystemID) REFERENCES SolarSystems(SolarSystemID)
);

-- Asteroids Table
CREATE TABLE Asteroid (
    AsteroidID INT PRIMARY KEY,
    Name VARCHAR(255),
    Mass DECIMAL(18, 2),
    Radius DECIMAL(18, 2),
```

```
    Age INT,
    DiscoveryDate DATE,
    SolarSystemID INT,
    GalaxyID INT,
    FOREIGN KEY (SolarSystemID) REFERENCES SolarSystems(SolarSystemID)
);

-- Nebulae Table
CREATE TABLE Nebula (
    NebulaID INT PRIMARY KEY,
    Name VARCHAR(255),
    Type VARCHAR(50),
    Size DECIMAL(18, 2),
    Location VARCHAR(255),
    GalaxyID INT,
    FOREIGN KEY (GalaxyID) REFERENCES Galaxy(GalaxyID)
);

-- Galaxies Table
CREATE TABLE Galaxy (
    GalaxyID INT PRIMARY KEY,
    Name VARCHAR(255),
    Type VARCHAR(50),
    Size DECIMAL(18, 2),
    Age INT,
    SolarSystemID INT,
    FOREIGN KEY (SolarSystemID) REFERENCES SolarSystems(SolarSystemID)
);

-- BlackHole Table
CREATE TABLE BlackHole (
    BlackHoleID INT PRIMARY KEY,
    Name VARCHAR(255),
    Mass DECIMAL(18, 2),
    Radius DECIMAL(18, 2),
    Age INT,
    DiscoveryDate DATE,
    GalaxyID INT,
    FOREIGN KEY (GalaxyID) REFERENCES Galaxy(GalaxyID)
);

-- LocalGroup Table
CREATE TABLE Cluster (
    ClusterID INT PRIMARY KEY,
    Name VARCHAR(255),
    Type VARCHAR(50),
    Size DECIMAL(18, 2),
    Age INT,
    GalaxyID INT,
    FOREIGN KEY (GalaxyID) REFERENCES Galaxy(GalaxyID)
);
```

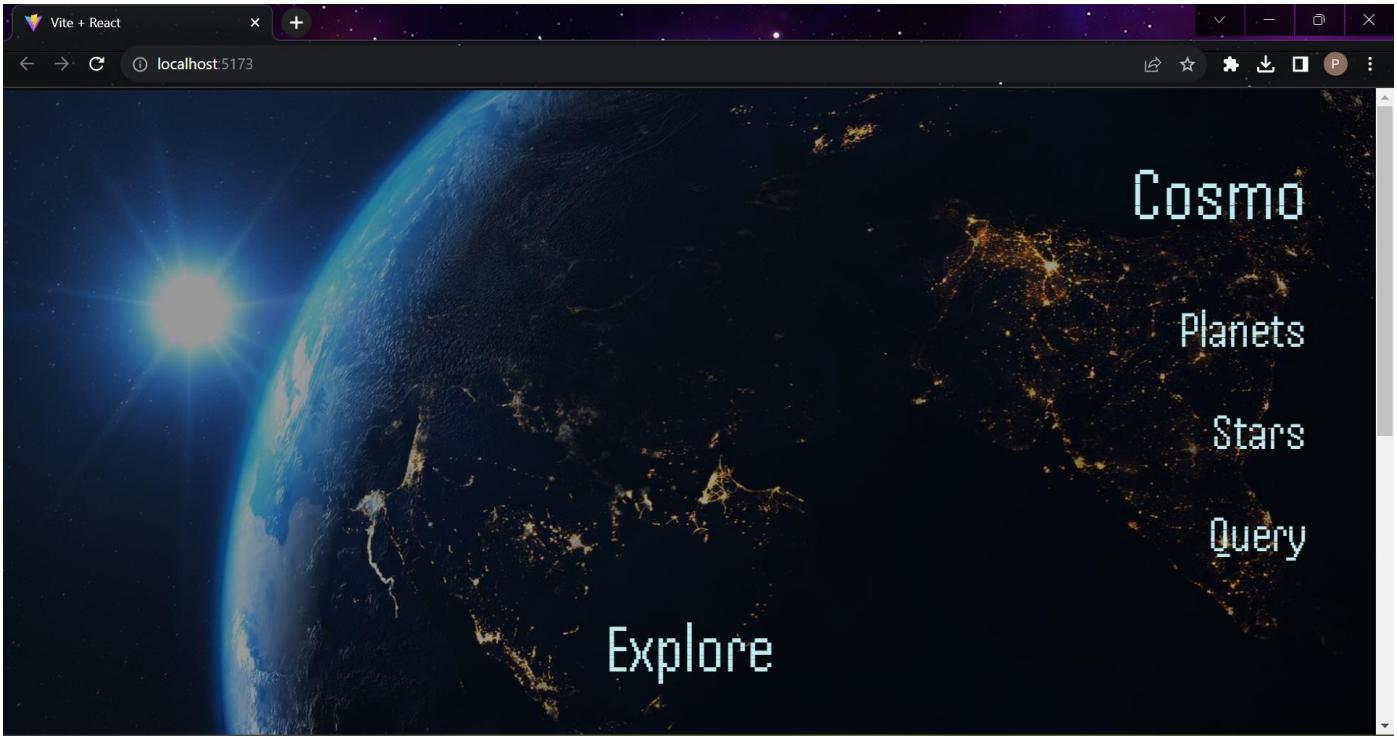
Crud Operations

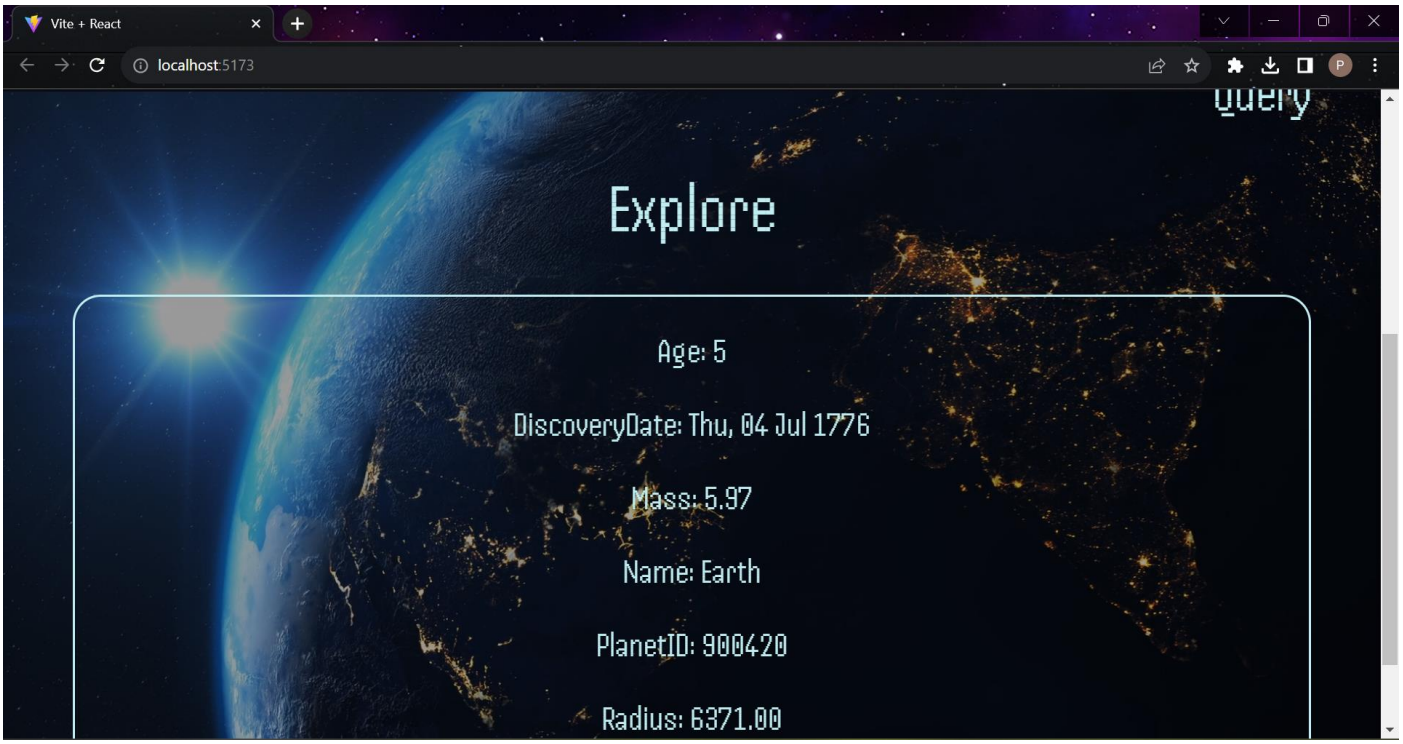
```sql
--Create
INSERT INTO Planet (Name, Mass, Radius, Age, DiscoveryDate, SolarSystemID)
VALUES ('NewPlanet', 10.5, 15.2, 500, '2023-11-24', 1);

--Read
SELECT Moon.Name, Moon.Mass, Moon.Radius
FROM Moon
JOIN Planet ON Moon.PlanetID = Planet.PlanetID
WHERE Planet.Name = 'Jupiter';

--Update
UPDATE Star
SET Temperature = 6000
WHERE StarID = 1;

--Delete
DELETE FROM Asteroid
WHERE AsteroidID = 1;
```

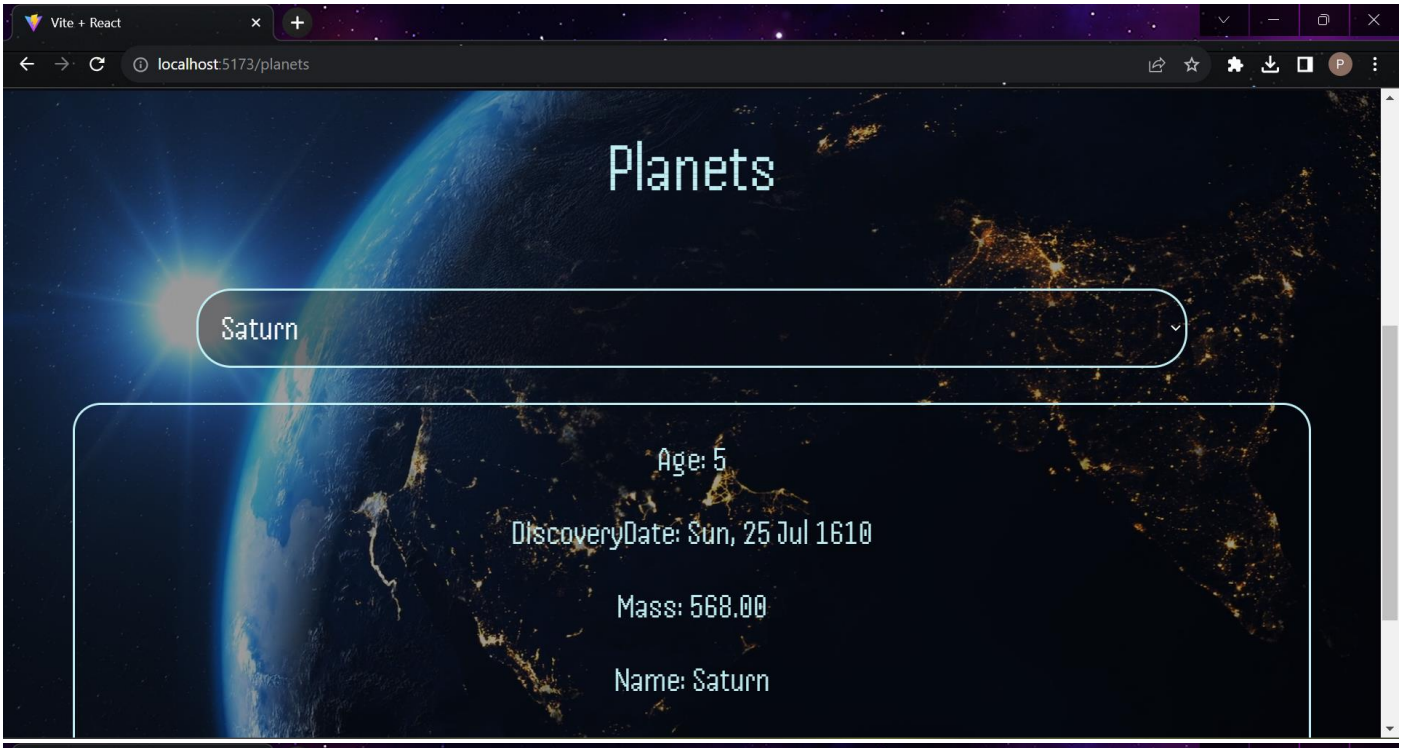
Functionalities

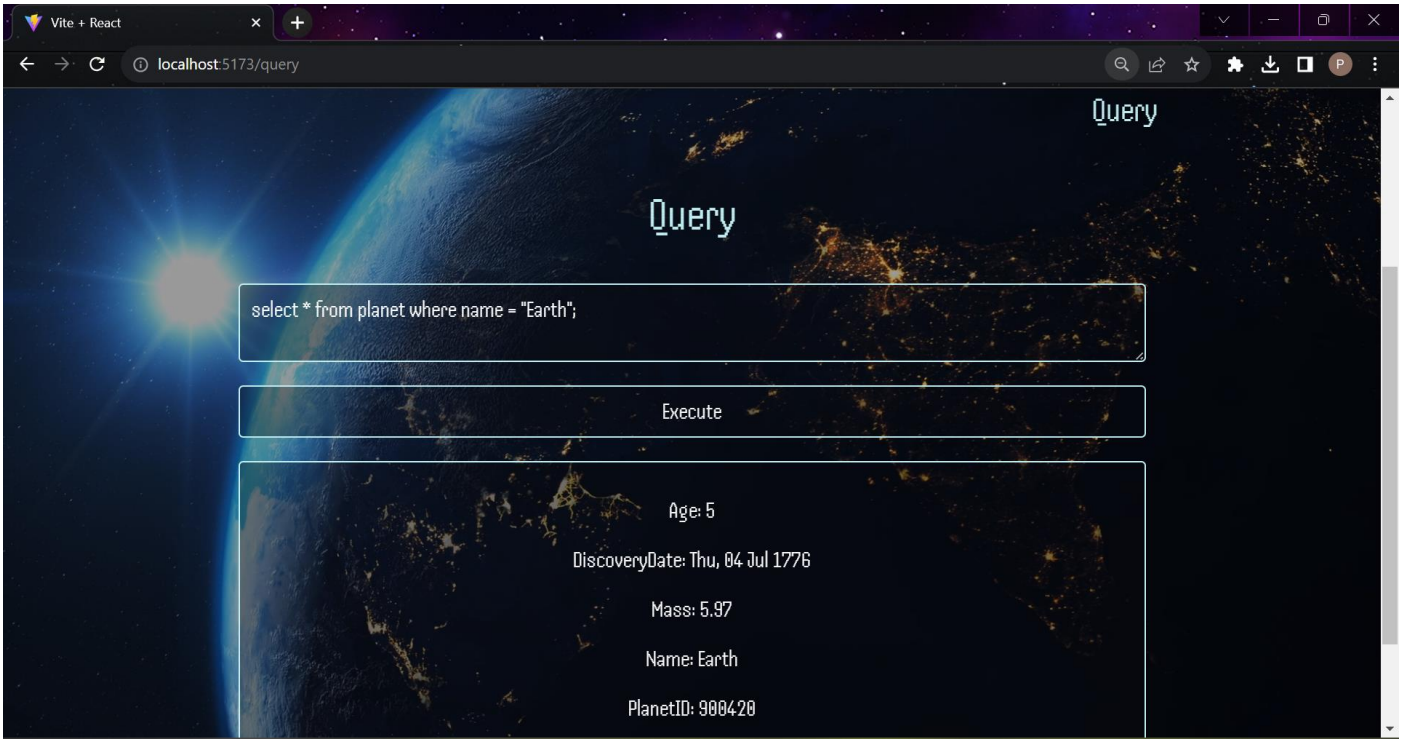
1. A comprehensive frontend built with React with a Flask backend.
2. Fully routed with 4 pages, Explore, Planets, Stars, Queries
3. Explore features clickable data attributes that start with Earth. Move from one planet to another or its star by clicking the ID associated with it.
4. Planets page features all the planets, same with Stars.
5. Query page features an input box that supports any SQL query. Execution results displayed appropriately.

Screenshots






Vite + React
localhost:5173/planets
Planets
Saturn
Age: 5
DiscoveryDate: Sun, 25 Jul 1610
Mass: 568.00
Name: Saturn


Vite + React
localhost:5173/query
Query
Query
select * from planet where name = "Earth";
Execute
Age: 5
DiscoveryDate: Thu, 04 Jul 1776
Mass: 5.97
Name: Earth
PlanetID: 900420

Queries

```sql
-- Explore moons of a particular planet:
SELECT Moon.Name, Moon.Mass, Moon.Radius
FROM Moon
JOIN Planet ON Moon.PlanetID = Planet.PlanetID
WHERE Planet.Name = 'Jupiter';

-- Discover stars in a solar system:
SELECT Star.Name, Star.SpectralType, Star.Temperature, Star.Luminosity
FROM Star
JOIN SolarSystem ON Star.SolarSystemID = SolarSystem.SolarSystemID
WHERE SolarSystem.Name = 'Our Solar System';

-- Find stars in each solar system with their spectral type:
SELECT SolarSystem.Name AS SolarSystemName, Star.Name AS StarName,
Star.SpectralType
FROM SolarSystem
JOIN Star ON SolarSystem.SolarSystemID = Star.SolarSystemID;

-- Find galaxies with their black holes and associated solar systems:
SELECT Galaxy.Name AS GalaxyName, BlackHole.Name AS BlackHoleName,
SolarSystem.Name AS SolarSystemName
FROM Galaxy
LEFT JOIN BlackHole ON Galaxy.GalaxyID = BlackHole.GalaxyID
LEFT JOIN SolarSystem ON Galaxy.SolarSystemID = SolarSystem.SolarSystemID;

-- Find the oldest planet in each solar system--
SELECT SolarSystem.Name AS SolarSystemName, Planet.Name AS OldestPlanet
FROM SolarSystem
JOIN Planet ON SolarSystem.SolarSystemID = Planet.SolarSystemID
WHERE Planet.Age = (SELECT MAX(Age) FROM Planet WHERE SolarSystemID =
SolarSystem.SolarSystemID);

-- List moons that are larger than the average moon radius:
SELECT Name, Radius
FROM Moon
WHERE Radius > (SELECT AVG(Radius) FROM Moon);

-- Find galaxies with the largest average black hole mass:
SELECT Galaxy.Name AS GalaxyName, AVG(BlackHole.Mass) AS AvgBlackHoleMass
FROM Galaxy
LEFT JOIN BlackHole ON Galaxy.GalaxyID = BlackHole.GalaxyID
GROUP BY Galaxy.GalaxyID
HAVING AVG(BlackHole.Mass) = (SELECT MAX(AvgMass) FROM (SELECT AVG(Mass) AS
AvgMass FROM BlackHole GROUP BY GalaxyID) AS AvgMassPerGalaxy);


-- Create Trigger to Update Total Number of Planets in a Solar System
DELIMITER //
-- Create Trigger to Update Total Number of Planets in a Solar System
CREATE TRIGGER UpdateSolarSystemTotalPlanets
AFTER INSERT ON Planet
```

```
FOR EACH ROW
BEGIN
    -- Update the total number of planets in the solar system
    UPDATE SolarSystem
    SET TotalPlanets = (
        SELECT COUNT(*)
        FROM Planet
        WHERE SolarSystemID = NEW.SolarSystemID
    )
    WHERE SolarSystemID = NEW.SolarSystemID;
END//
DELIMITER ;

-- Create Function to Calculate Average Mass of Planets in a Solar System
DELIMITER //
CREATE FUNCTION CalculateAverageMass(solarSystemID INT)
RETURNS DECIMAL(18, 2)
BEGIN
    DECLARE avgMass DECIMAL(18, 2);

    SELECT AVG(Mass) INTO avgMass
    FROM Planet
    WHERE SolarSystemID = solarSystemID;

    RETURN avgMass;
END //
DELIMITER ;


--Nested query that retrieves information about planets
--in solar systems that belong to a specific galaxy
SELECT *
FROM Planet
WHERE SolarSystemID IN (
    SELECT SolarSystemID
    FROM SolarSystem
    WHERE Solarsystem.GalaxyID = (
        SELECT GalaxyID
        FROM Galaxy
        WHERE Name = 'Milky Way'
    )
);

-- Procedure to get all planets of a solarsystem
DELIMITER //

CREATE PROCEDURE GetPlanetsInSolarSystem(IN solarSystemID INT)
BEGIN
    SELECT Name, Mass, Radius
    FROM Planet
    WHERE SolarSystemID = solarSystemID;
END //

DELIMITER ;
```

```sql
--query to find cluster names of all planets

SELECT
    Planet.Name AS PlanetName,
    Cluster.Name AS ClusterName
FROM
    Cluster
JOIN
    Galaxy ON Cluster.GalaxyID = Galaxy.GalaxyID
JOIN
    SolarSystems ON Galaxy.GalaxyID = SolarSystems.GalaxyID
JOIN
    Planet ON SolarSystems.SolarSystemID = Planet.SolarSystemID;
```