**Query Metadata Parser Documentation**

**Problem Statement**

**Solution**

**Architecture Diagram**

**Prerequisites**

Tools/Technologies used

1. Python
2. SQLite Database
3. Jinja Framework
4. Bootstrap Framework
5. HTML, CSS

Required Python packages

1. Flask
2. Sqlite3
3. Sqlparse

Project Structure

---------QueryMetadataParser  
 ├── \_\_init\_\_.py # setup your app  
 ├── query\_parser.py # Code for SQL Query Parsing  
 ├── create\_sqlite\_db.py # Database Creation   
 ├── app.py # Flask Code  
 ├── db.sqlite # your database  
 └── templates  
 ├── display\_all\_metadata.html # display all type of metadata  
 ├── display\_metadata.html # display selected metadata  
 ├── index.html # show the login form  
 ├── successful\_file\_upload.html # SQL script upload to directory   
 └── successful\_login.html # page after login

└── type\_of\_metadata.html # select type of metatdata

└── upload\_sql\_script.html # form to upload sql script

└── style.css # css

└── UploadedSQLScripts

└── SQL\_Script\_1.html # sql scripts uploaded in this directory

└── SQL\_Script\_2.html # sql scripts uploaded in this directory

**Technical Implementation**

1. Prepare your Environment
2. Create a virtual environment in python

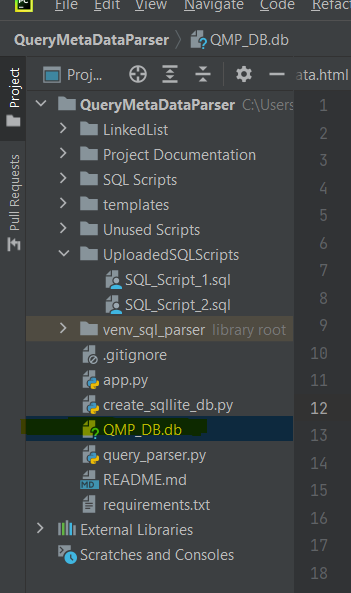
virtualenv venv sql\_parser

1. Install the python packages in the virtual environment described in the pre-requisite section.

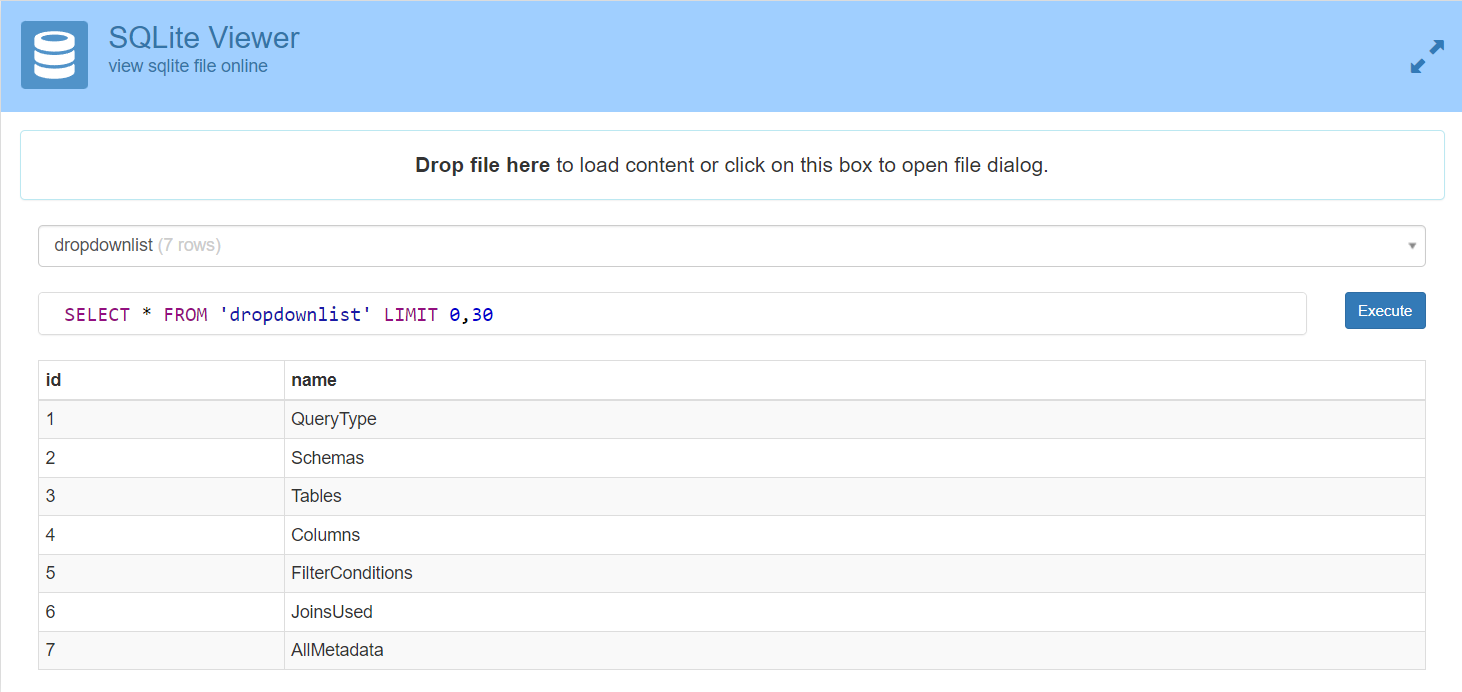
flask  
flask-sqlalchemy  
flask-login  
werkzeug  
pandas

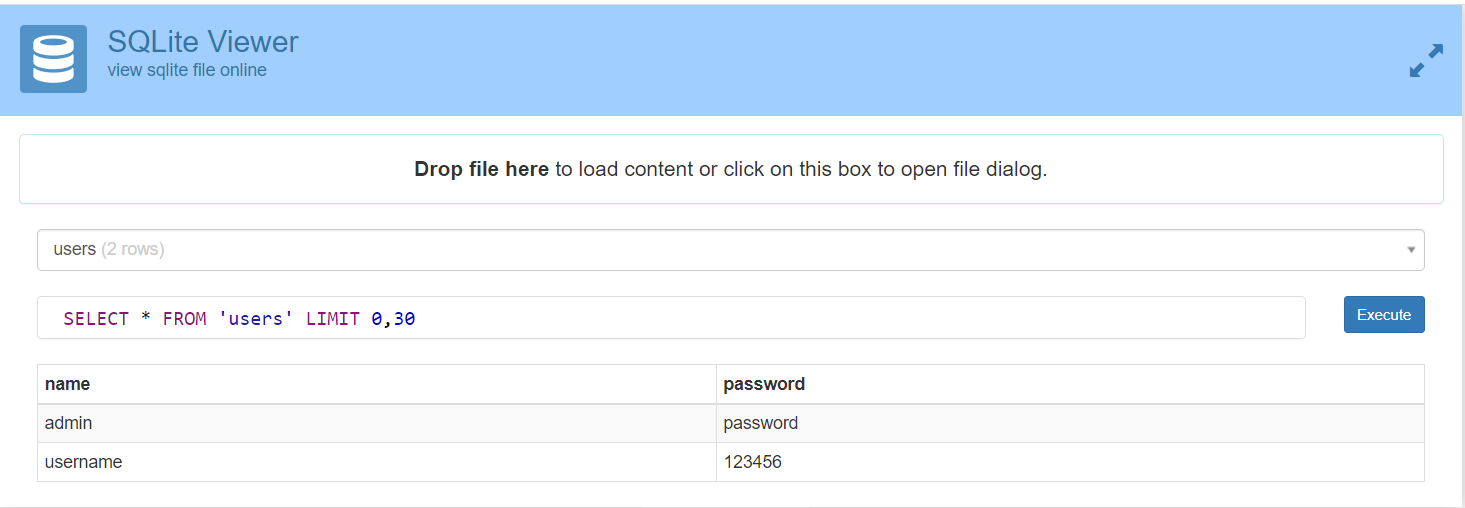
To install packages pip install package\_name or you can create a new file requirements.txt (this file will contains one package name each row) and install all packages once using : pip install -r requirements.txt

1. Create project structure as mentioned in the pre-requisite section.
2. Create Database
3. Create Sqlite Database – QMP\_DB.db



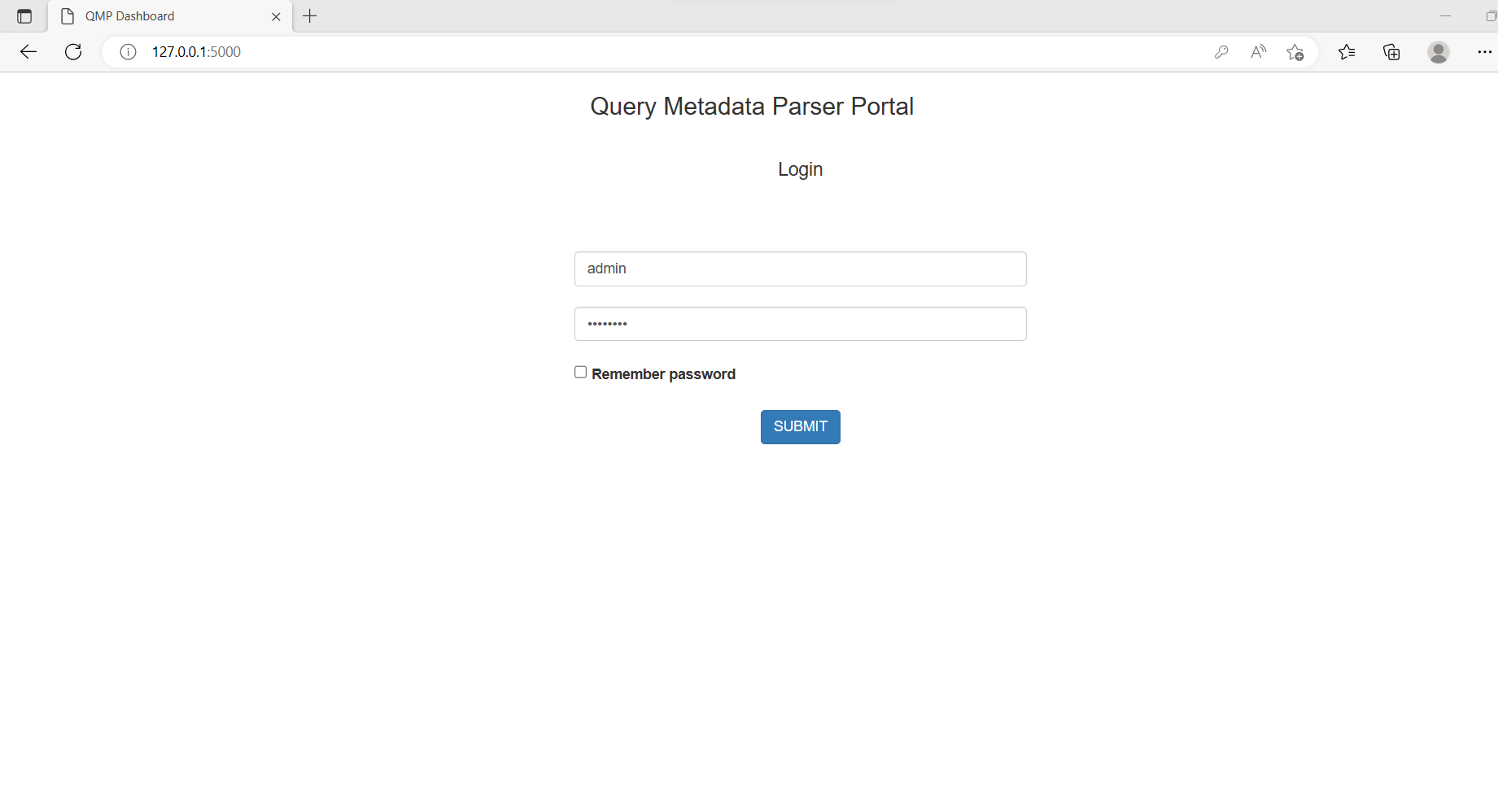
1. Create Tables dropdownlist and users in the database QMP\_DB.db



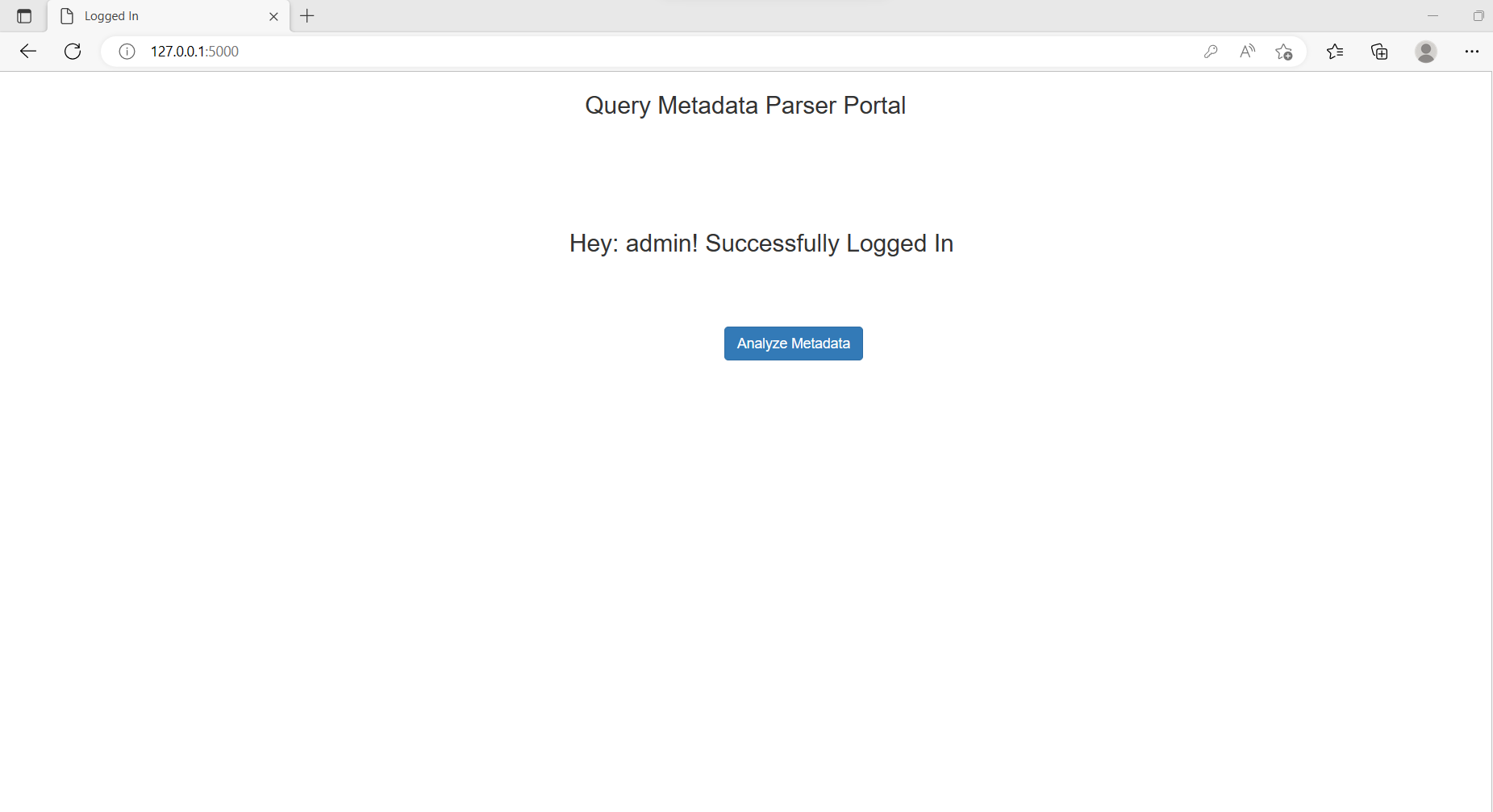


1. Create Web Pages

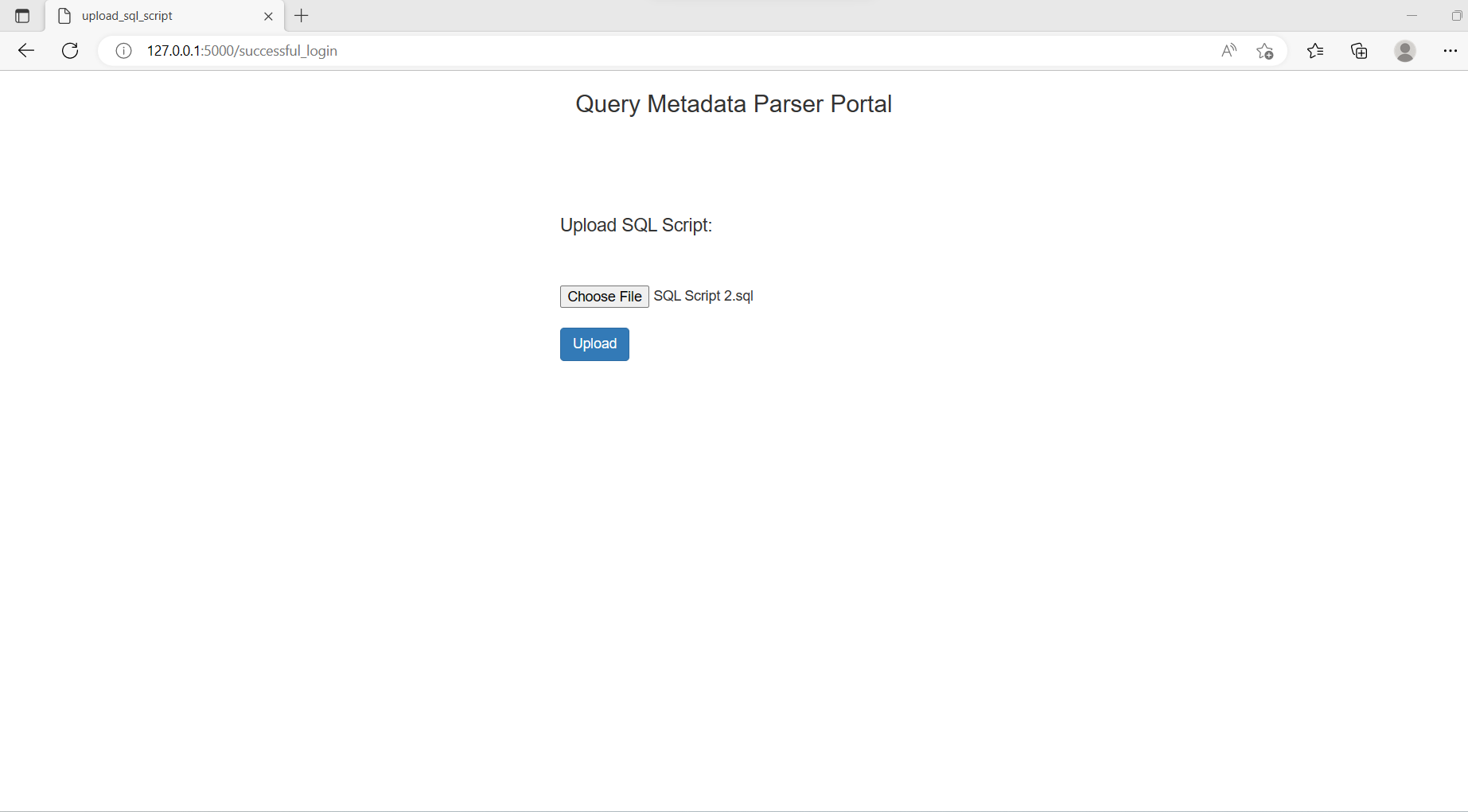
* Enter user details and login to the Query Metadata Parser portal.
* The application will validate the user details with the credentials stored in the Sqlite database and if it matches it will redirect to the next page.

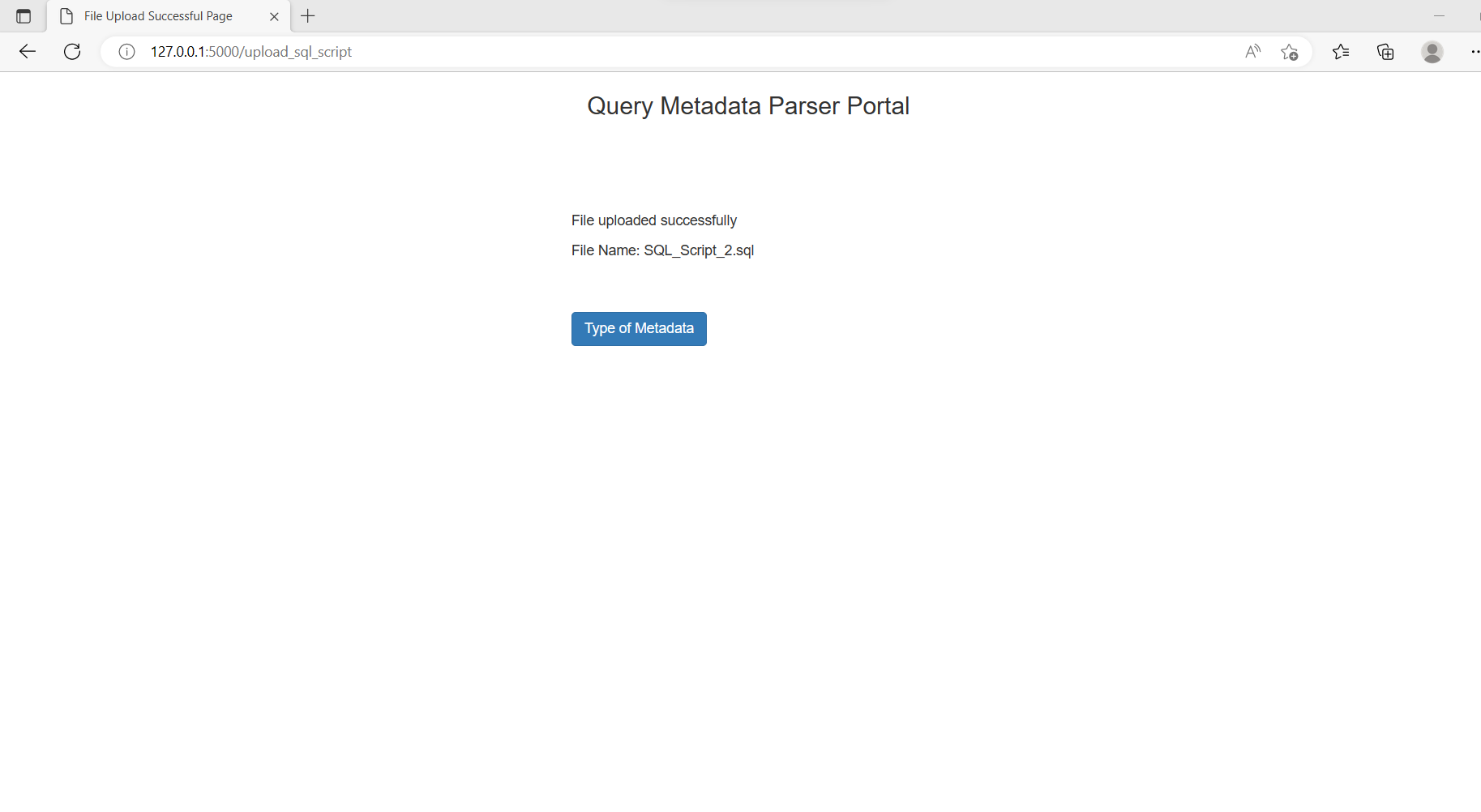


* Click on Analyze Metadata button post successful login.

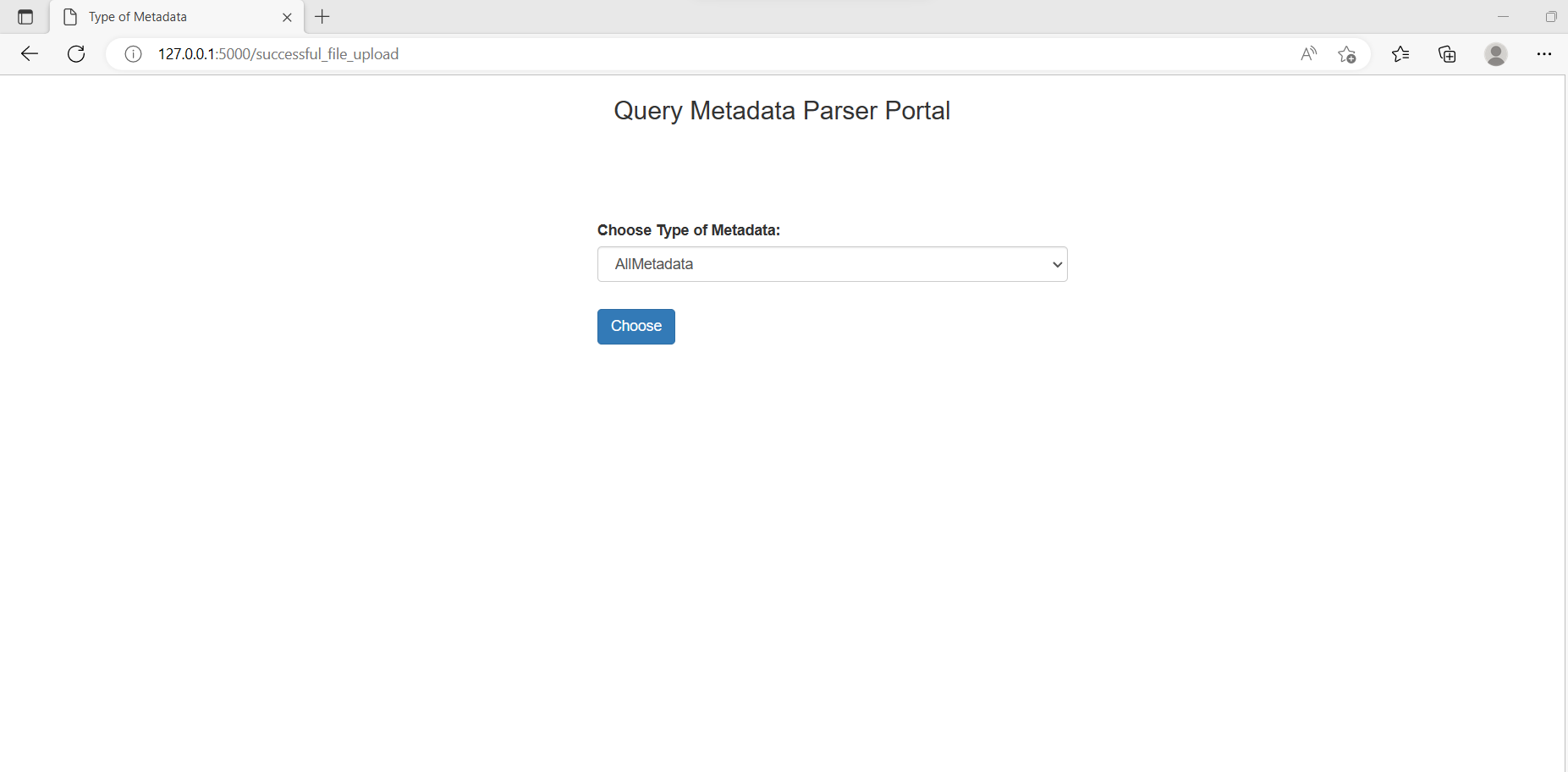


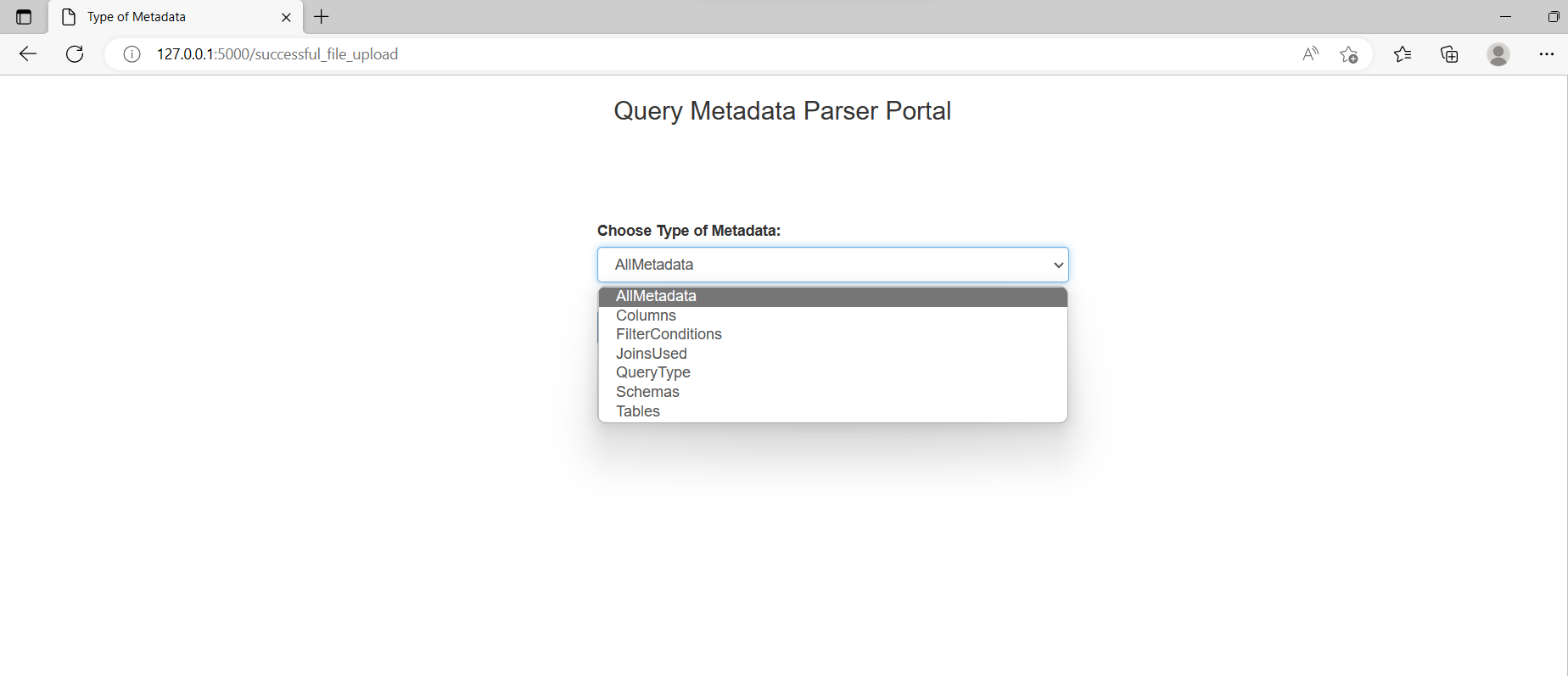
* Choose a SQL Script from your local device and upload it to the portal.



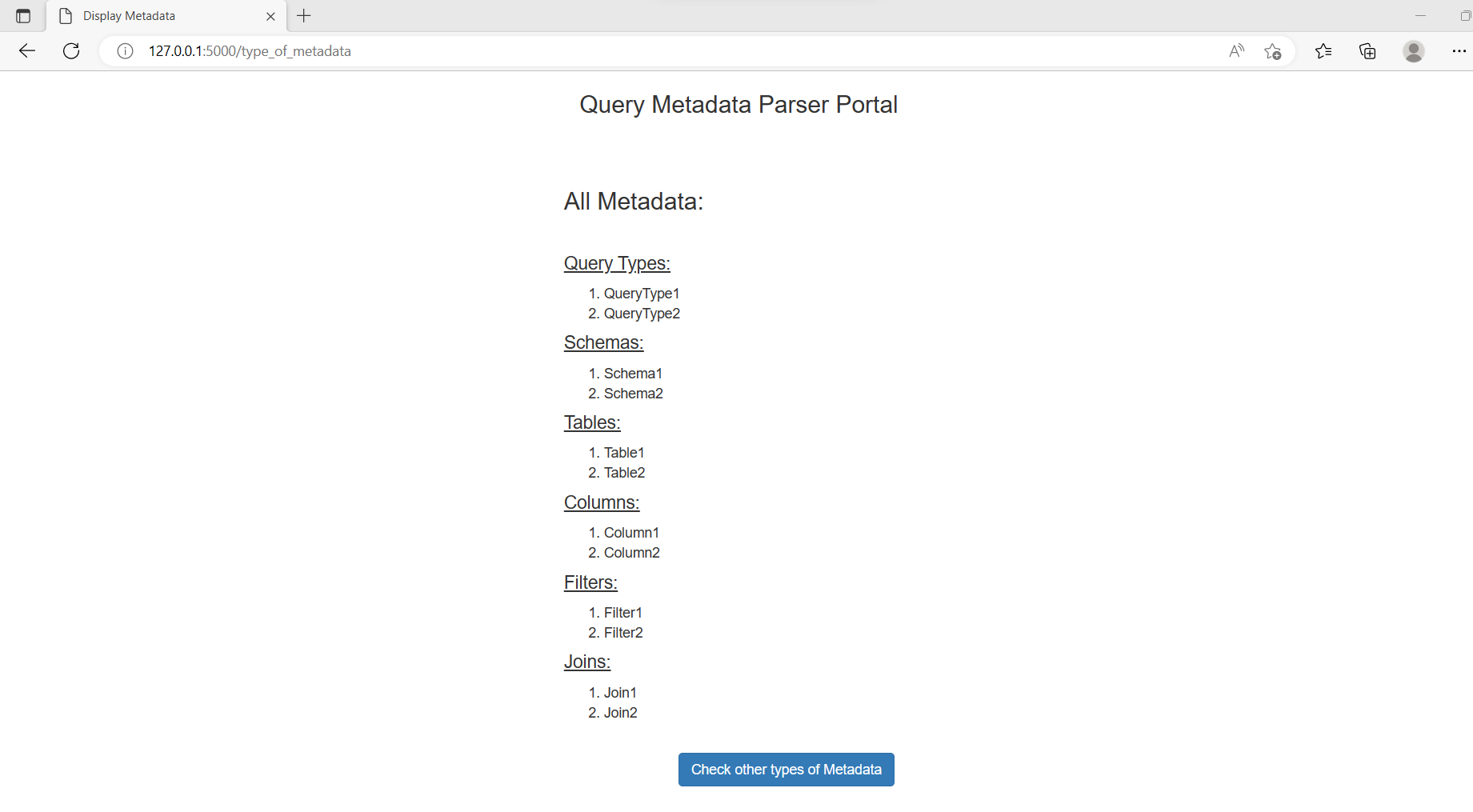


* Choose the type of Metadata that you want to display.

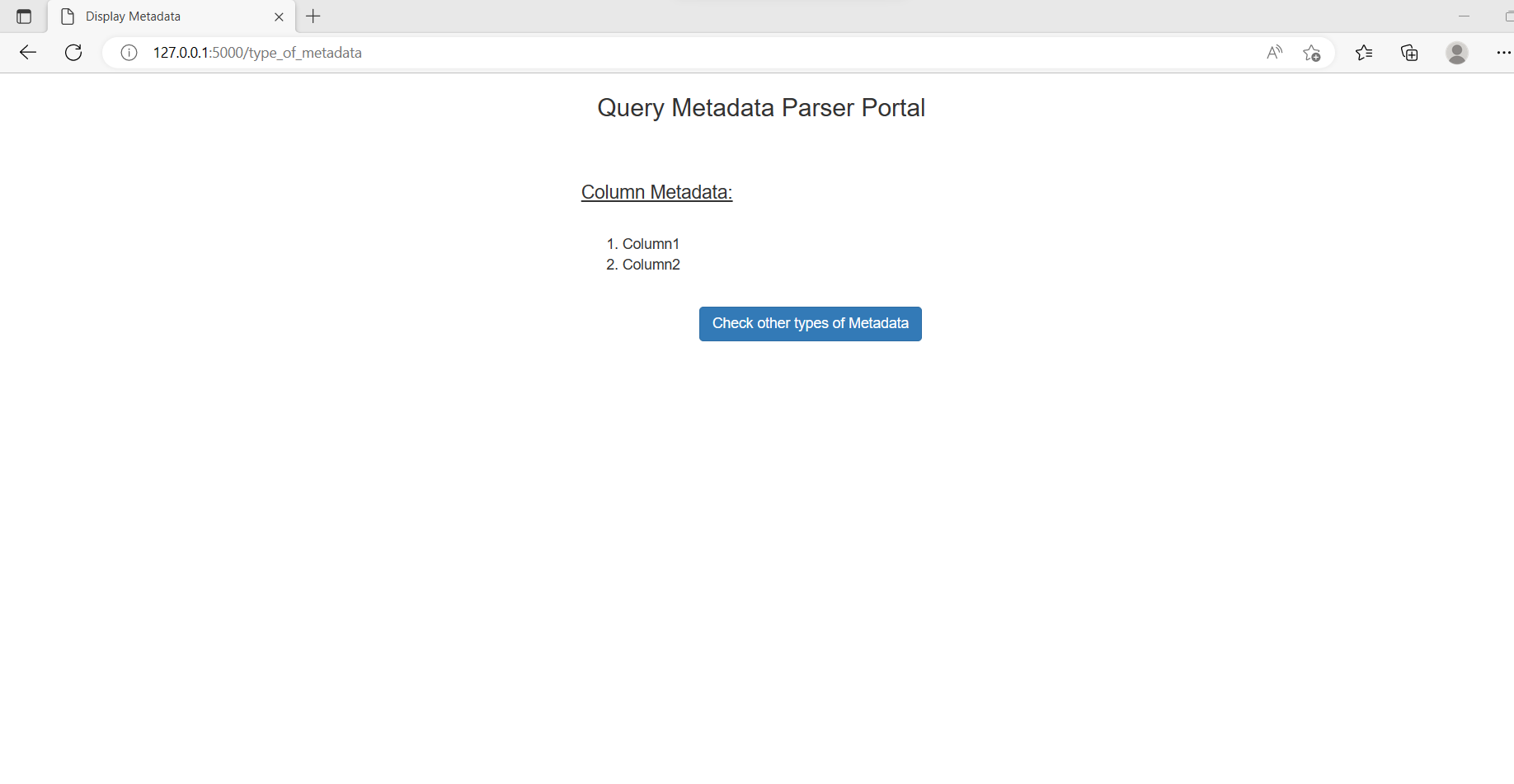




* Based on the Metadata type selection, Metadata information would be displayed.



* Check other types of metadata by clicking on the button “Check other types of Metadata”.



1. Connect Web Pages using the Flask Framework

* Connect to sqlite database.
* Create routing methods to route/redirect web pages based on the GET/POST.

**Conclusion**

**GitHub link**

* <https://github.com/tariniteam/QueryMetaDataParser>

**Contributors**

1. Harsha Navalkar ( <https://www.linkedin.com/in/harsha-navalkar-00085515b/> )
2. Vikram Mahapatra ()