Politecnico
di Torino

Live Coding

Development with Streamlit

Objectives

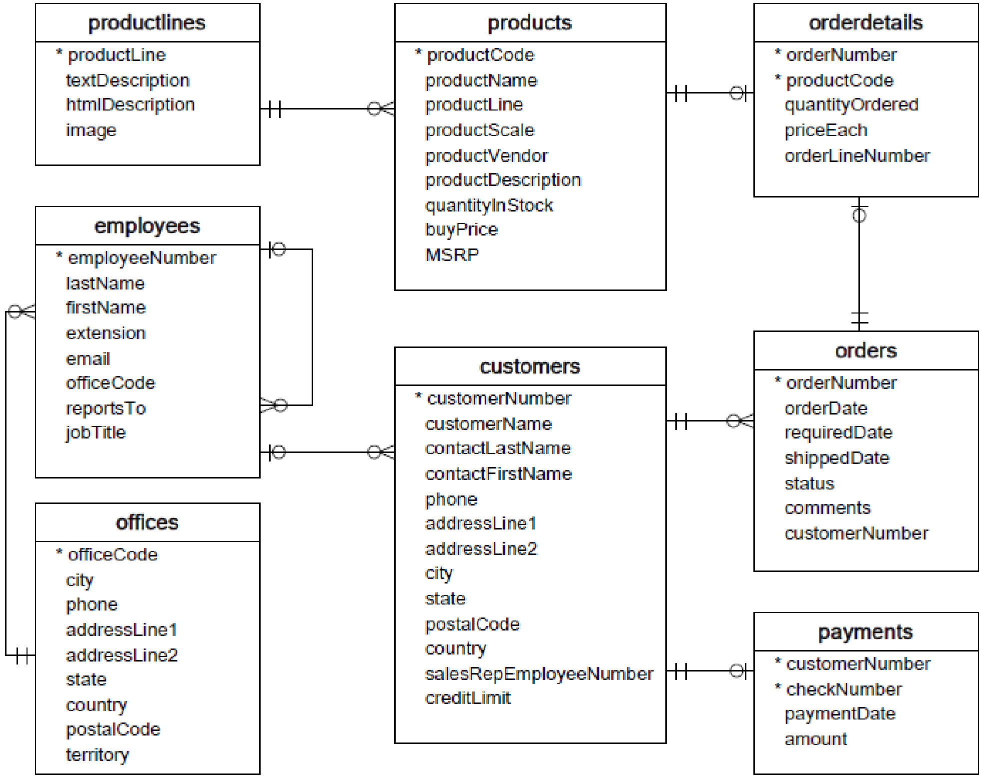
- Create a **multipage** application in Streamlit step-by-step
- Implement and learn about the **main** Streamlit **widgets**
- Integrate **SQL queries** into a web application to view and update data from a MySQL database
- Structure a project using the main development tools (i.e., GitHub, Docker)

Requirements:

- Install Git/GitHub
- Install Docker/Docker Compose
- Clone repositories

Database

- Database of a scale model car dealer
- This is a sample MySQL database called *classicmodels*
- It collects information about products, employees, customers, orders...



productlines
* productLine
textDescription
htmlDescription
image

products
* productCode
productName
productLine
productScale
productVendor
productDescription
quantityInStock
buyPrice
MSRP

orderdetails
* orderNumber
* productCode
quantityOrdered
priceEach
orderLineNumber

employees
* employeeNumber
lastName
firstName
extension
email
officeCode
reportsTo
jobTitle

customers
* customerNumber
customerName
contactLastName
contactFirstName
phone
addressLine1
addressLine2
city
state
postalCode
country
salesRepEmployeeNumber
creditLimit

orders
* orderNumber
orderDate
requiredDate
shippedDate
status
comments
customerNumber

offices
* officeCode
city
phone
addressLine1
addressLine2
state
country
postalCode
territory

payments
* customerNumber
* checkNumber
paymentDate
amount

https://www.mysqltutorial.org/mysql-sample-database.aspx

Git and GitHub

- Git is a distributed **version control system** and records changes to files over time
- Allows multiple people to work on the same project at the same time

- GitHub is a web platform that uses Git for **repository** management: directories that collect the code of software projects
- Allows to collaborate, to track changes, and to manage issues

https://github.com/

Git in a Nutshell

- **Commit**: represents a set of changes to the project files, creating a new version recorded in the history
- **Pull and Push**: commands to synchronize local repositories with remote repositories. *Pull* to download updates remotely and *Push* to apply local changes to the remote repository
- **Branch**: separate branch that allows you to work in parallel to multiple isolated changes of the main branch
- **Merge**: the process of combining the contents of one branch into another

https://github.com/git-guides/install-git

Docker

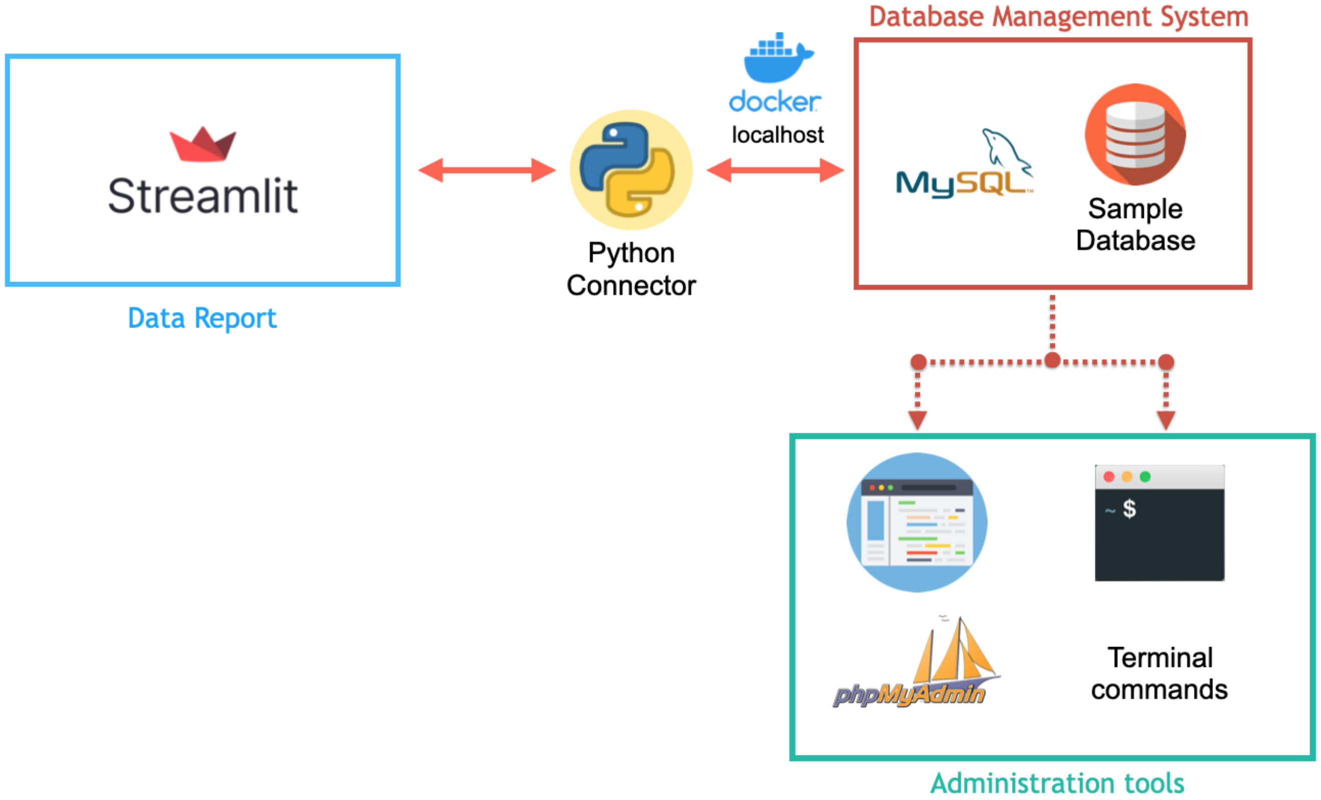
- **Docker**: open-source platform that simplifies the deployment and execution of applications through **containers**
- A container is a unit that contains everything you need to start your application (e.g., code, libraries)
- Containers are lightweight and portable, ensuring reliability in different development environments

- **Docker Compose**: used to define and manage multi-container applications
- Is configured with a YAML file

https://www.docker.com/

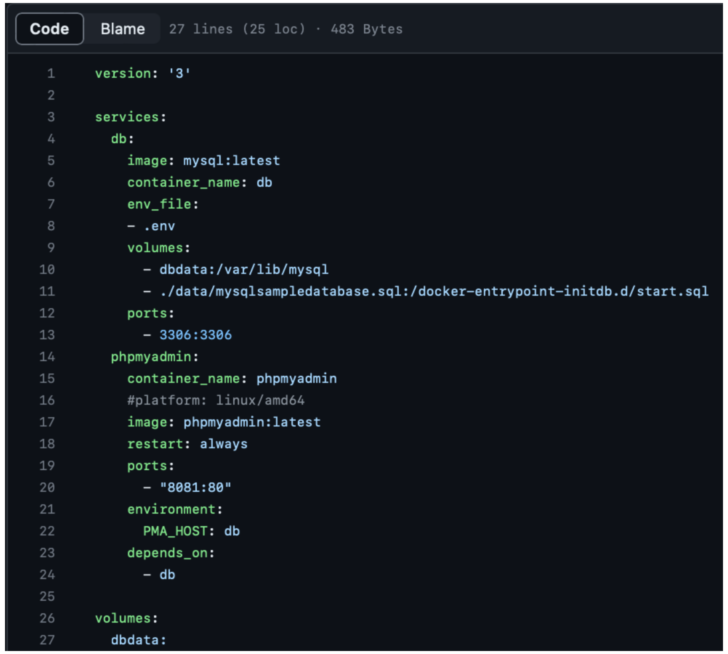
Project Overview

- The repository mysql-docker contains the MySQL database already configured (*localhost:3306*) and the phpMyAdmin web interface (*localhost:8081*)
- The repository streamlitTutorial contains the Streamlit project that will interact with the database



mysql-docker Repository

- The *docker-compose.yml* defines the MySQL and phpMyAdmin containers
- The *data/mysqlsampledatabase.sql* file contains SQL code
- The *.env* file contains the credentials (it is good practice to change them)

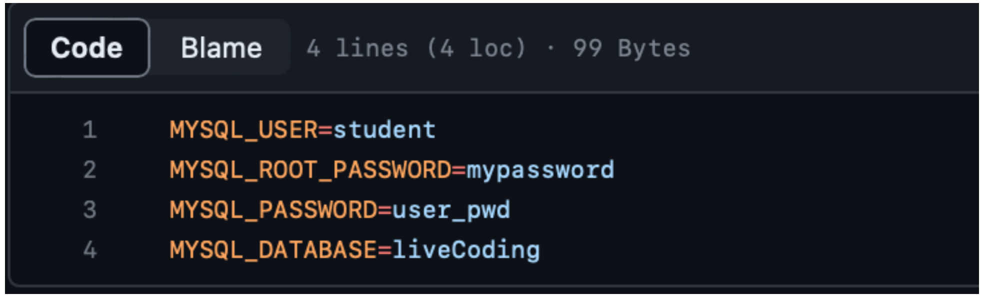


```yaml
version: '3'

services:
  db:
    image: mysql:latest
    container_name: db
    env_file:
    - .env
    volumes:
      - dbdata:/var/lib/mysql
      - ./data/mysqlsampledatabase.sql:/docker-entrypoint-initdb.d/start.sql
    ports:
      - 3306:3306
  phpmyadmin:
    container_name: phpmyadmin
    #platform: linux/amd64
    image: phpmyadmin:latest
    restart: always
    ports:
      - "8081:80"
    environment:
      PMA_HOST: db
    depends_on:
      - db

volumes:
  dbdata:
```

docker-compose.yml

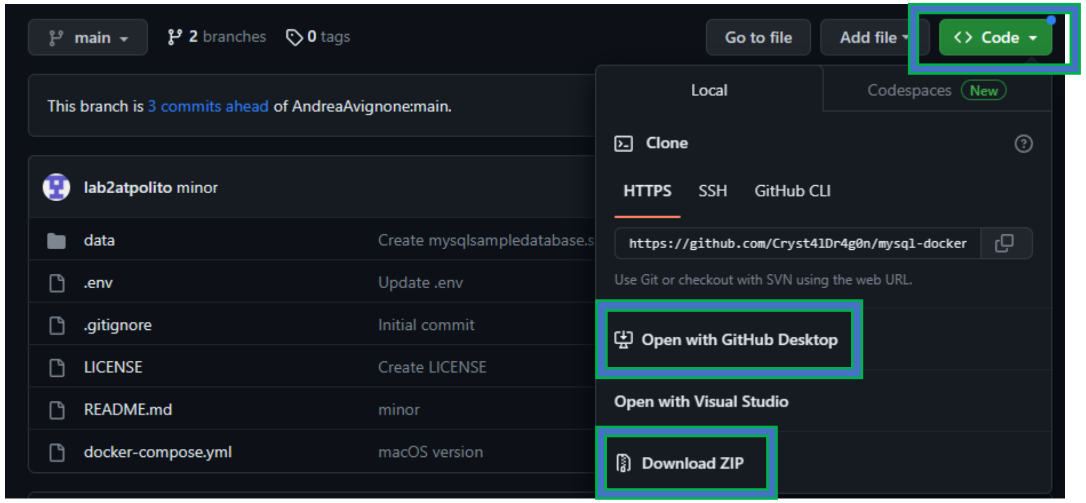


```
MYSQL_USER=student
MYSQL_ROOT_PASSWORD=mypassword
MYSQL_PASSWORD=user_pwd
MYSQL_DATABASE=liveCoding
```

.env

mysql-docker Repository

- Each repository has a ***README.md*** that provides essential information about the project to understand and use it
- Once git is installed, the repository can be cloned from the terminal:
  - ➢ git clone https://github.com/DavideCalandra/mysql-docker-eng.git
- Or by downloading the zip file or opening it in the GitHub Desktop application



This branch is 3 commits ahead of AndreaAvignone:main.

lab2atpolito minor

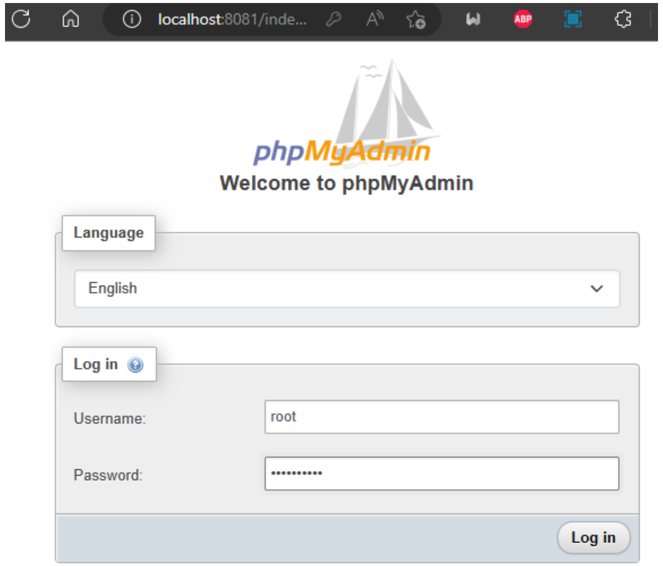
data	Create mysqlsampledatabase.s
.env	Update .env
.gitignore	Initial commit
LICENSE	Create LICENSE
README.md	minor
docker-compose.yml	macOS version

https://github.com/Cryst41Dr4g0n/mysql-docker

Use Git or checkout with SVN using the web URL.

mysql-docker Repository

- Once you start Docker, in the *mysql-docker* folder, type the command to start the containers (-d to start them in the background):
  - ➢ docker-compose up –d
- To stop containers:
  - ➢ docker-compose down
- (Optional) enter phpMyAdmin on *localhost:8081*

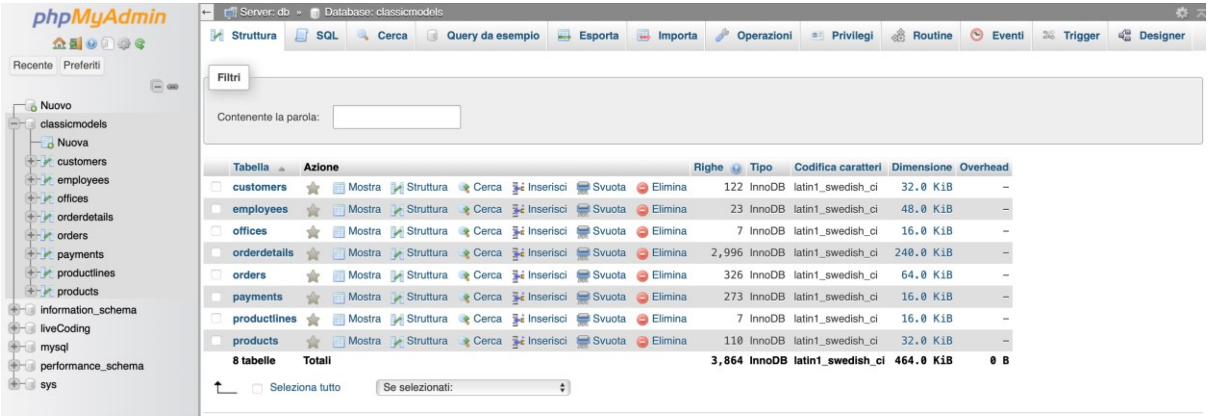


localhost:8081/inde...

Language: English

Username: root

Password: ••••••••••

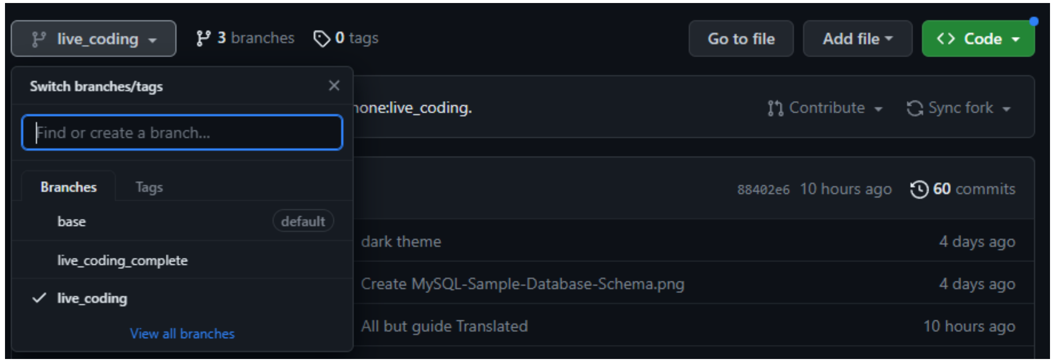


Server: db » Database: classicmodels

Tabella	Azione	Righe	Tipo	Codifica caratteri	Dimensione	Overhead
customers	Mostra Struttura Cerca Inserisci Svuota Elimina	122	InnoDB	latin1_swedish_ci	32.0 KiB	–
employees	Mostra Struttura Cerca Inserisci Svuota Elimina	23	InnoDB	latin1_swedish_ci	48.0 KiB	–
offices	Mostra Struttura Cerca Inserisci Svuota Elimina	7	InnoDB	latin1_swedish_ci	16.0 KiB	–
orderdetails	Mostra Struttura Cerca Inserisci Svuota Elimina	2,996	InnoDB	latin1_swedish_ci	240.0 KiB	–
orders	Mostra Struttura Cerca Inserisci Svuota Elimina	326	InnoDB	latin1_swedish_ci	64.0 KiB	–
payments	Mostra Struttura Cerca Inserisci Svuota Elimina	273	InnoDB	latin1_swedish_ci	16.0 KiB	–
productlines	Mostra Struttura Cerca Inserisci Svuota Elimina	7	InnoDB	latin1_swedish_ci	16.0 KiB	–
products	Mostra Struttura Cerca Inserisci Svuota Elimina	110	InnoDB	latin1_swedish_ci	32.0 KiB	–
8 tabelle	**Totali**	**3,864**	**InnoDB**	**latin1_swedish_ci**	**464.0 KiB**	**0 B**

streamlitTutorial Repository

- ➢ git clone https://github.com/DavideCalandra/streamlitTutorial-eng
- Consisting of 3 branches:
  - ***base**:* structure of a generic multi-page project (can be used as a starting point for laboratory, homework)
  - ***live_coding**:* starting point for live coding
  - ***live_coding_complete**:* complete final application
- To change branches from terminal:
  - ➢ git checkout *branch_name*
- Or:



one:live_coding.

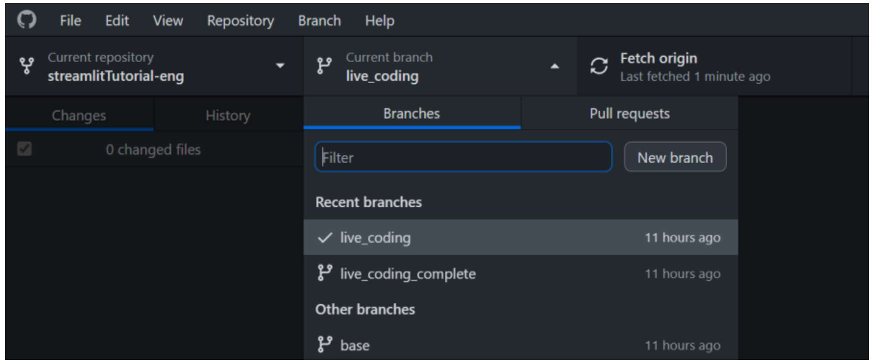
88402e6 10 hours ago 60 commits

dark theme 4 days ago

Create MySQL-Sample-Database-Schema.png 4 days ago

All but guide Translated 10 hours ago

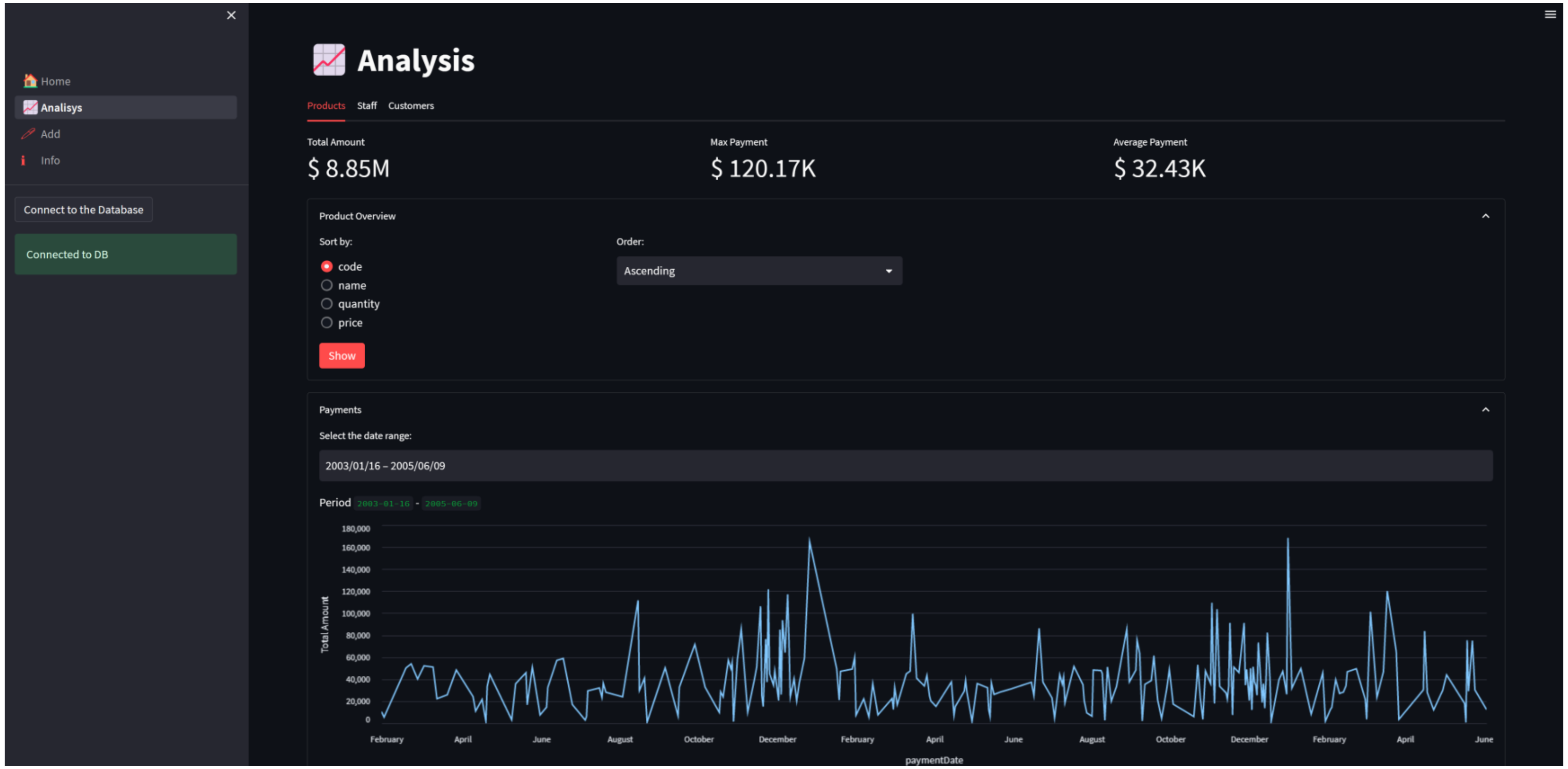
GitHub web



GitHub Desktop

Dashboard

- Create a final dashboard with visualization and data addition function
- 4 pages will be developed: Home, Analysis, Add, Info
- Includes user input widgets to customize queries and data display



Connected to DB

Analysis

Total Amount
$ 8.85M

Max Payment
$ 120.17K

Average Payment
$ 32.43K

Product Overview

Sort by:
● code
○ name
○ quantity
○ price

Order:
Ascending

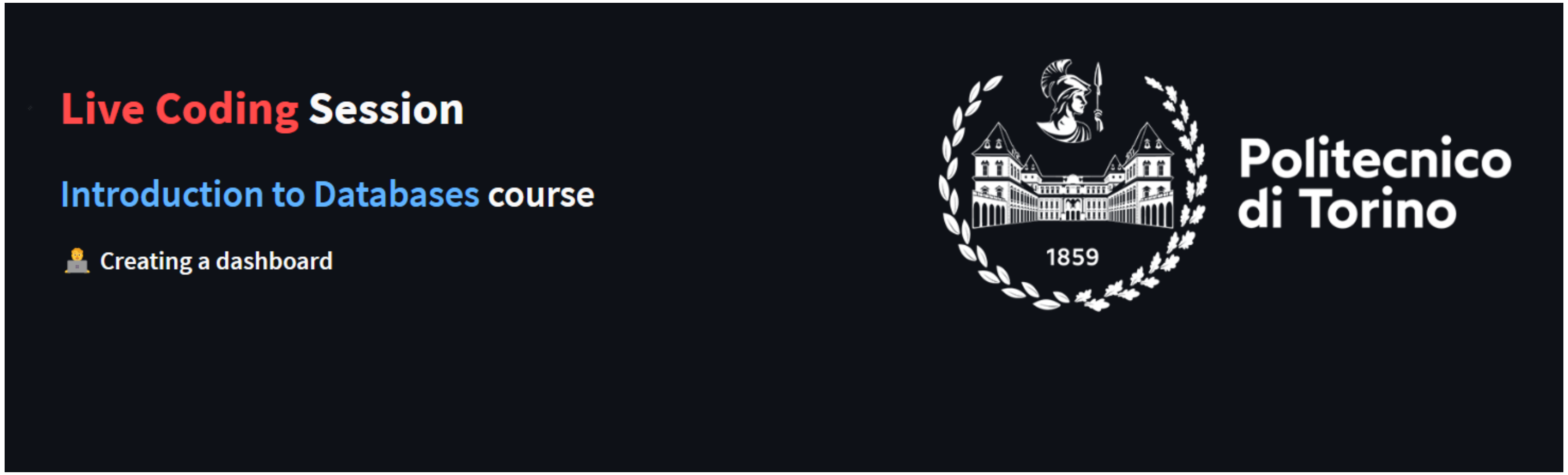
Payments

Select the date range:
2003/01/16 – 2005/06/09

Period `2003-01-16` - `2005-06-09`

Homepage Customization

- Insert text elements in custom markdowns on the homepage
- Add the Polito logo
- Initialize the *session* state



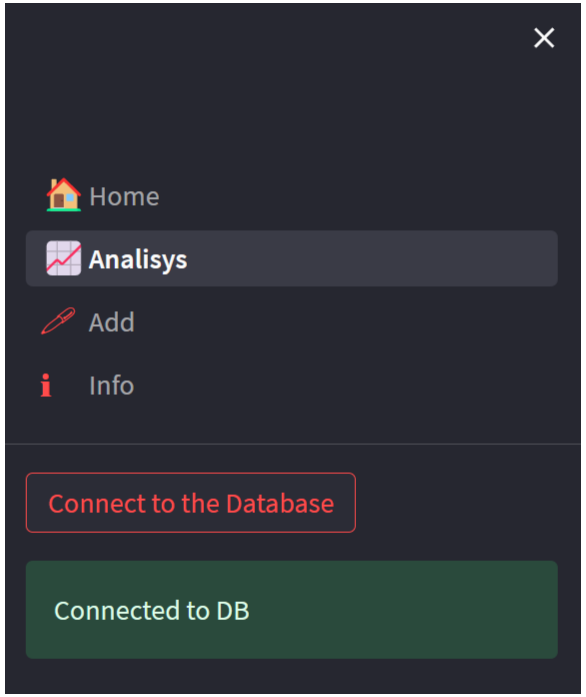
Live Coding Session

Introduction to Databases course

🧑‍💻 Creating a dashboard

Connecting to the DB

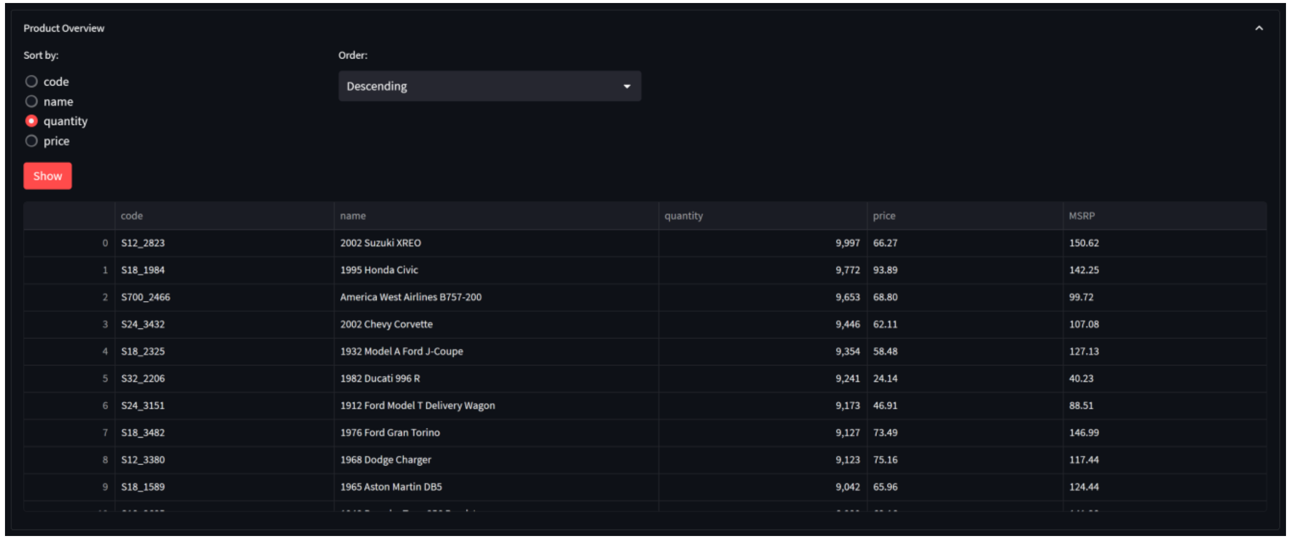
- Add functions to connect to the DB and execute queries
- Add a function that checks the status of the connection
- Add a button on the sidebar to connect and print the result (success or error) on the screen



Connected to DB

Product analysis

- Add to the *Products* tab of the *Analysis* page:
  - **3 Metrics:** total amount, maximum payment, average payment;
  - **2 Expanders**: product overview, payments;
  - **3 User Inputs**: radio button (product attribute to sort), select box (ascending or decreasing) and date range (interval in which to view the progress of payments)



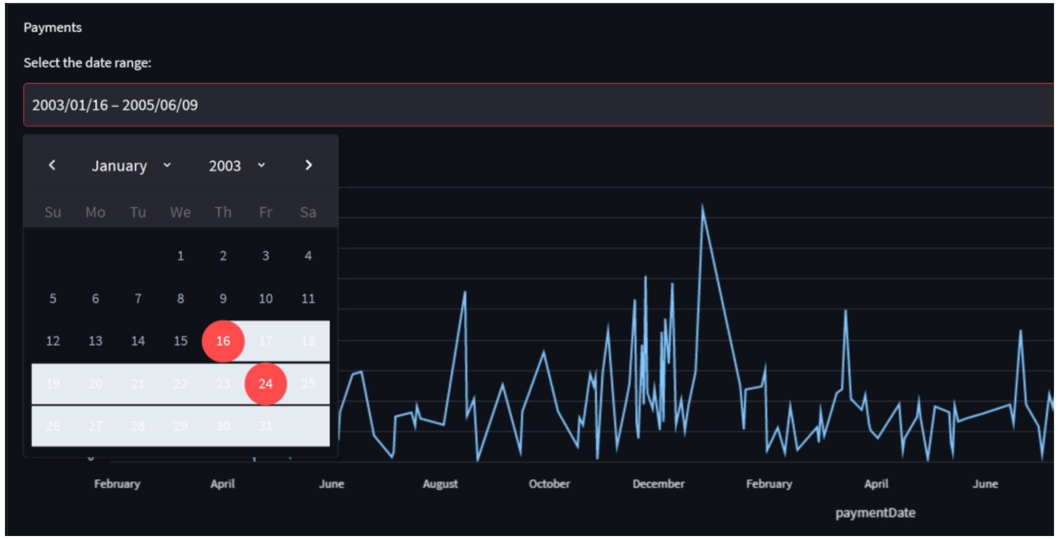
Product Overview

Sort by:

○ code
○ name
● quantity
○ price

Order: Descending

	code	name	quantity	price	MSRP
0	S12_2823	2002 Suzuki XREO	9,997	66.27	150.62
1	S18_1984	1995 Honda Civic	9,772	93.89	142.25
2	S700_2466	America West Airlines B757-200	9,653	68.80	99.72
3	S24_3432	2002 Chevy Corvette	9,446	62.11	107.08
4	S18_2325	1932 Model A Ford J-Coupe	9,354	58.48	127.13
5	S32_2206	1982 Ducati 996 R	9,241	24.14	40.23
6	S24_3151	1912 Ford Model T Delivery Wagon	9,173	46.91	88.51
7	S18_3482	1976 Ford Gran Torino	9,127	73.49	146.99
8	S12_3380	1968 Dodge Charger	9,123	75.16	117.44
9	S18_1589	1965 Aston Martin DB5	9,042	65.96	124.44

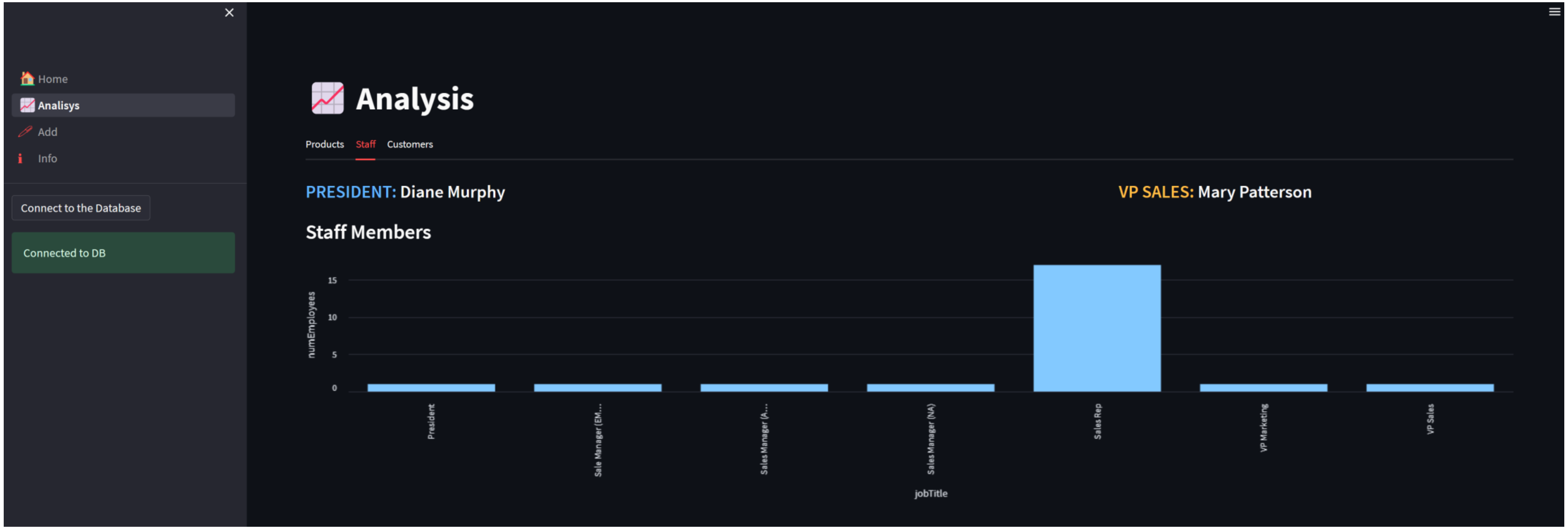


Payments

Select the date range: 2003/01/16 – 2005/06/09

Staff analysis

- View the first and last names of ***President*** and ***VP Sales***
- Represent with a **bar chart** the distribution of employees in the various roles



Connected to DB

Analysis

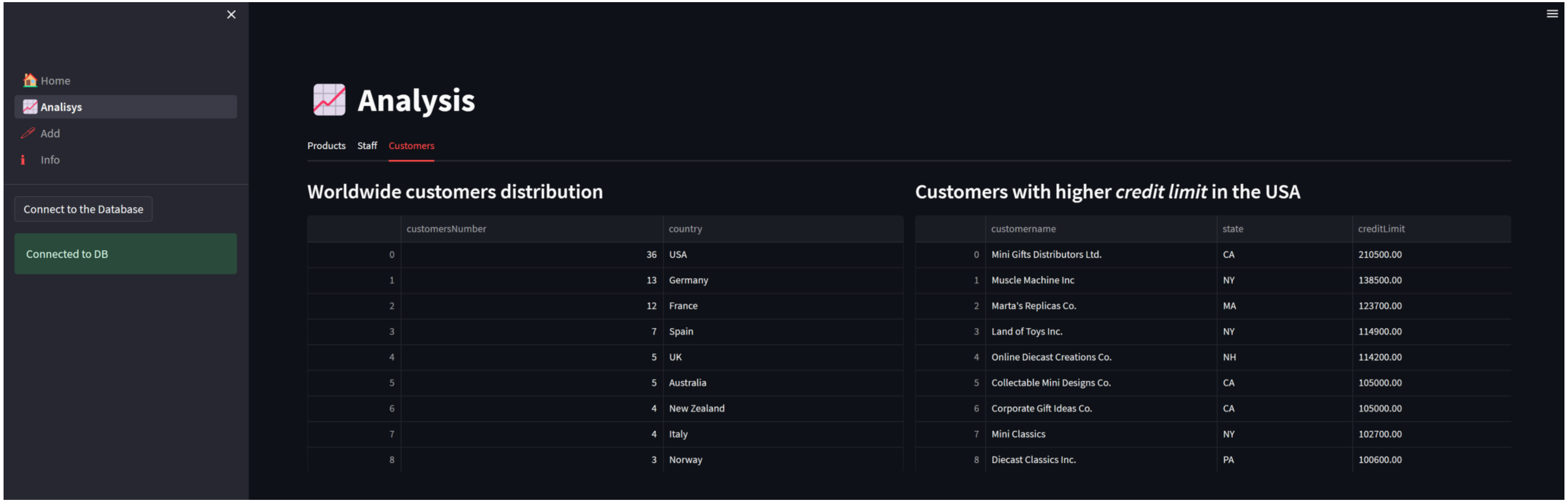
PRESIDENT: Diane Murphy

VP SALES: Mary Patterson

Staff Members

Customer analysis

- View customer information
- Insert a table that describes the origin of customers, grouping them by country and sorting them by number
- Insert a table showing U.S. customers with *creditLimit>100000*



Analysis

Worldwide customers distribution

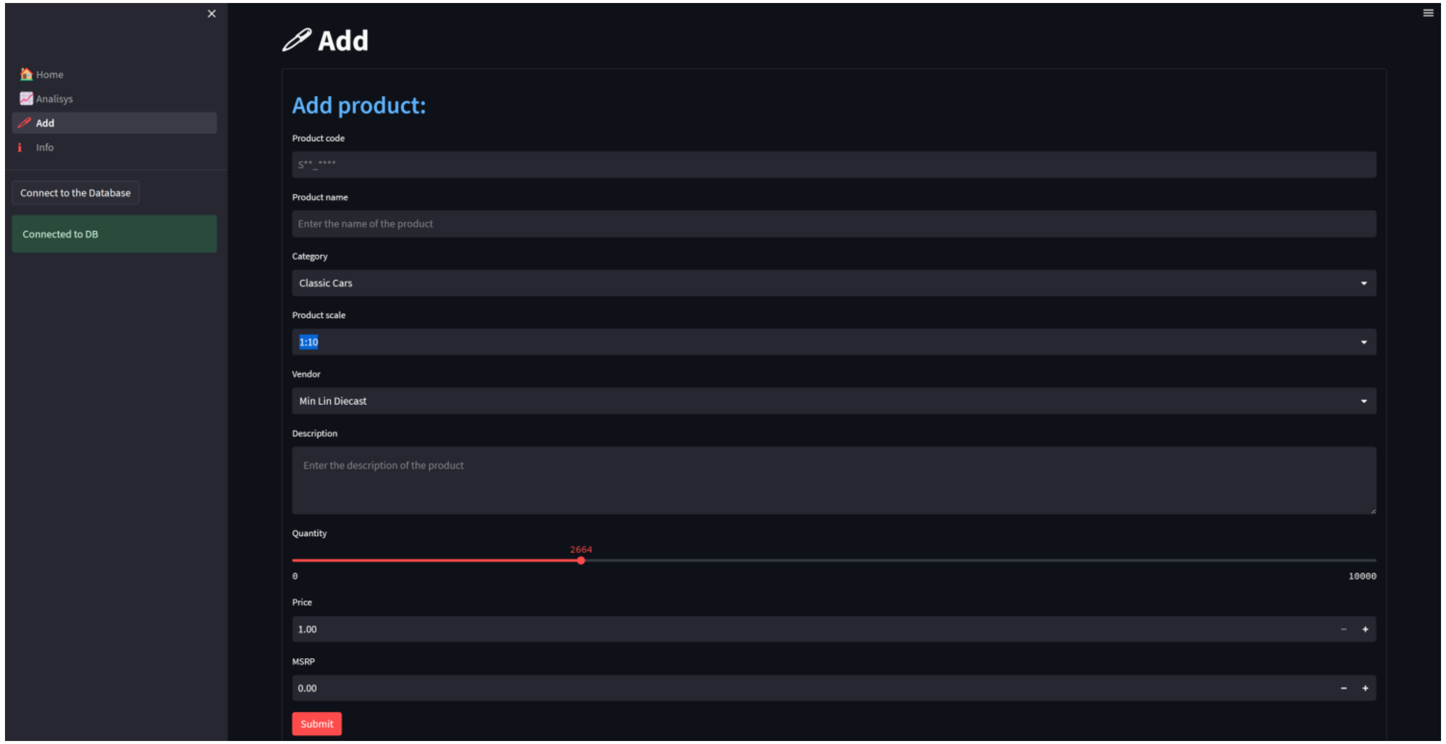
	customersNumber	country
0	36	USA
1	13	Germany
2	12	France
3	7	Spain
4	5	UK
5	5	Australia
6	4	New Zealand
7	4	Italy
8	3	Norway

Customers with higher *credit limit* in the USA

	customername	state	creditLimit
0	Mini Gifts Distributors Ltd.	CA	210500.00
1	Muscle Machine Inc	NY	138500.00
2	Marta's Replicas Co.	MA	123700.00
3	Land of Toys Inc.	NY	114900.00
4	Online Diecast Creations Co.	NH	114200.00
5	Collectable Mini Designs Co.	CA	105000.00
6	Corporate Gift Ideas Co.	CA	105000.00
7	Mini Classics	NY	102700.00
8	Diecast Classics Inc.	PA	100600.00

Adding a product

- Create a form to add a new product to the database
- Use text insertion, selectbox, slider, and numeric input widgets
- Add a submit button, run the query, and verify that it was successfully executed
- Capture the following parameters: *productCode, productName, productLine, productScale, ProductVendor, ProductDescription, quantityInStock, buyPrice, MSRP*



Add

Add product:

Product code: S**_****

Product name: Enter the name of the product

Category: Classic Cars

Product scale: 1:10

Vendor: Min Lin Diecast

Description: Enter the description of the product

Quantity: 2664 (0 – 10000)

Price: 1.00

MSRP: 0.00