National University of Science and Technology POLITEHNICA Bucharest

Faculty of Electronics, Telecommunications and Information Technology

Semester Project

Relational Database Interface

Student: Crăciun Alexandru Coordinator:

Group: 431F Prof. Valentin Pupezescu

2024

Introduction

The goal of this project is to create a relational database interface to manage data. The project implements a many-to-many (M:N) relational database model that involves plants, characteristics and their associations. The purpose is to allow users to perform CRUD (Create, Read, Update, Delete) operations on all entities in the database, ensuring seamless data management.

This documentation focuses on the first application, developed using **Flask**, **MySQL**, **HTML** and **CSS**.

Technologies Used

**Backend**:

* **Flask**: A lightweight web framework used to build the server-side of the application
* **Flask-MySQLdb**: A library for integrating Flask with a MySQL database
* **Python**: The primary programming language used for backend logic

**Database**:

* **MySQL**: A relational database management system used for data storage and retrieval

**Frontend**:

* **HTML/CSS**: Used for designing the user interface templates rendered by Flask

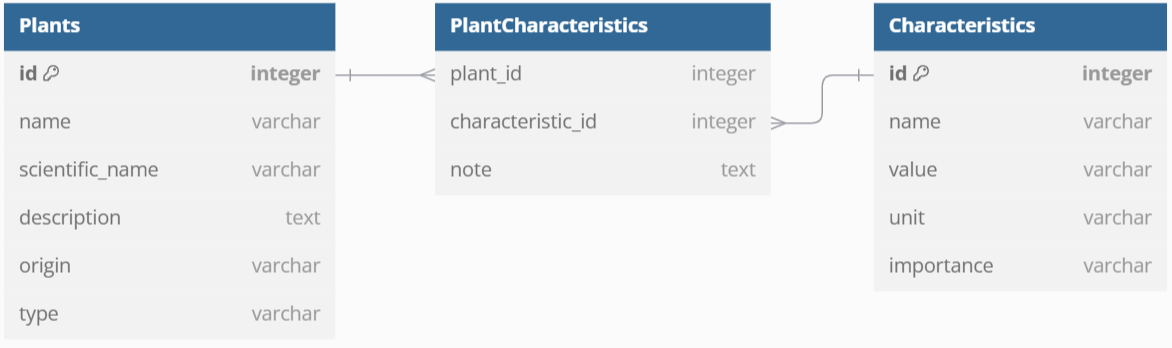
Application Overview

The purpose of this Flask-based application is to allow users to manage plants and their characteristics efficiently, while handling the many-to-many relationships between plants and characteristics through the link table “PlantCharacteristics”.

The web application implements the CRUD operations, enabling the user to apply all the necessary methods to add, visualize, modify and delete the data. This process can be achieved through the user-friendly interface, which contains a navigation bar for easy movement between different sections, forms for adding or editing data, and tables displaying all entities and their relationships.

Database Diagram

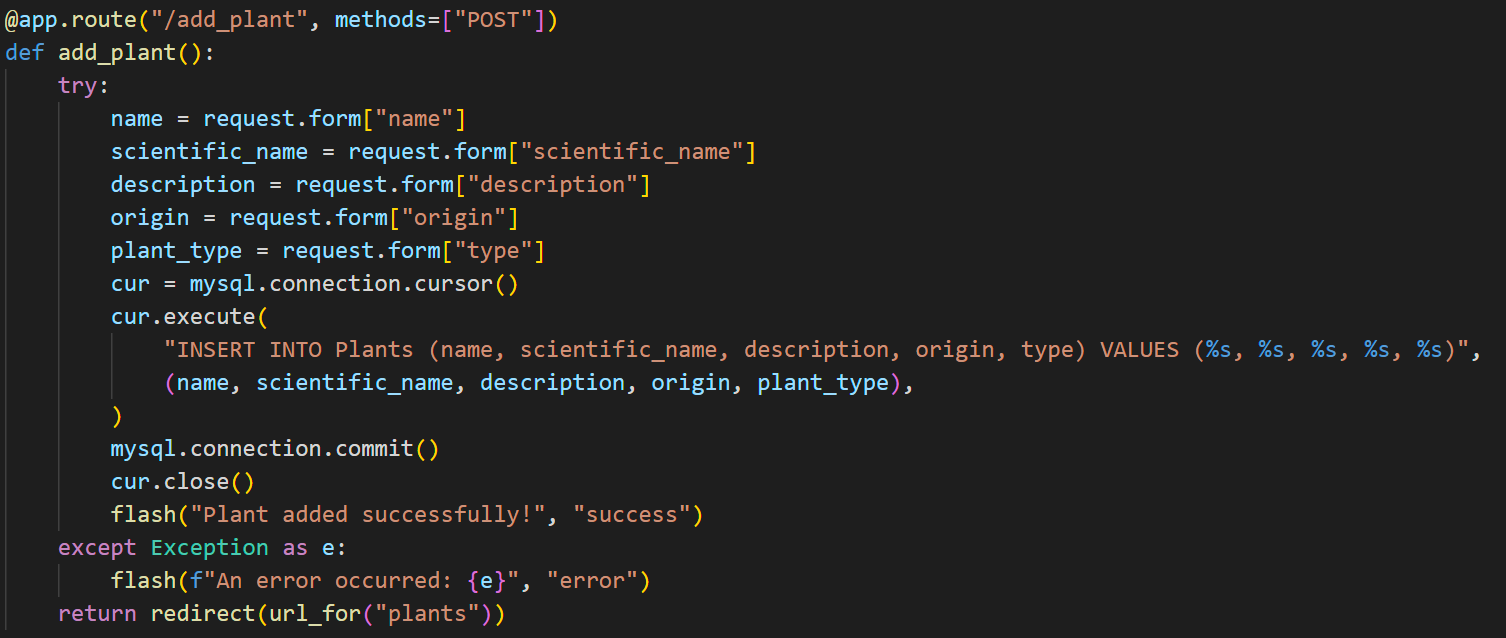
Below is the ER diagram representing the database structure. The **Plants** table stores details about plants, the **Characteristics** table contains attributes of the plant, while the **PlantCharacteristics** table is the link between the plants and characteristics.



Functional Code Snippets

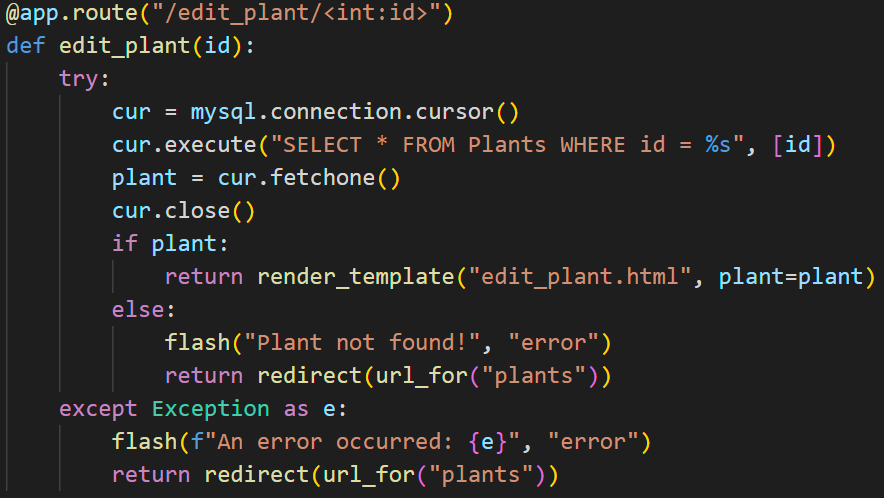
**Adding a Plant**

This route handles adding a new plant to the database



**Editing a Plant**

This route fetches the plant data to render the edit form



This route processes the updated data and saves it to the database



Bibliography

[1] dbdiagram.io, "Free Online Database Diagram Designer," Available: <https://dbdiagram.io>. [Accessed: Jan. 27, 2025].

[2] YouTube, "Video Tutorials and Resources," Available: <https://www.youtube.com>. [Accessed: Jan. 2, 2025].

[3] Flask Documentation, "Welcome to Flask," Available: <https://flask.palletsprojects.com>. [Accessed: Jan. 2, 2025].

4] MySQL Documentation, "MySQL 8.0 Reference Manual," Available: <https://dev.mysql.com/doc/>. [Accessed: Jan. 2, 2025].

[5] dotenv Documentation, "Managing Environment Variables with dotenv," Available: <https://pypi.org/project/python-dotenv/>. [Accessed: Jan. 2, 2025].

[6] W3Schools, "HTML and CSS Tutorials," Available: <https://www.w3schools.com>. [Accessed: Jan. 2, 2025].