Database Layer in an application

Contents

Overview

In this module, we will introduce the idea of using a database for a web application.

Database layer

Most web applications require data to be persisted, such as user information. In order to persist data, we need to implement a database layer into the application structure.

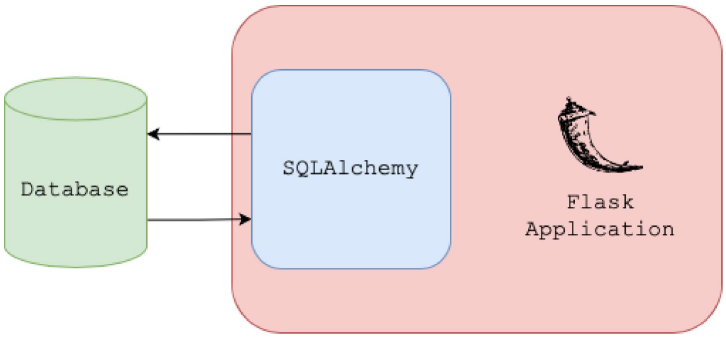
The data that the app requires will be stored in a database for it to access when it needs to.

SQLAlchemy

SQLAlchemy is a Python package that allows us to query and manipulate data stored in an SQL database. It can be easily integrated into the Flask web framework.

It is an **Object Relational Mapper (ORM)**, meaning that it uses classes and objects to represent the schema and entities in a database so that our Python code can interact with the data as Python objects.

This means we don't have to parse together SQL commands and keep our logic entirely Pythonic.



SQLAlchemy can communicate with various types of databases including sqlite, MySQL, Oracle and PostgreSQL.

Connecting the application to the database

To use SQLAlchemy, we must define the location of the database it will be interfacing with. We do so by defining an attribute in the application's Flask object like so:

```
from flask import Flask

app = Flask(__name__)

app.config['SQLALCHEMY_DATABASE_URI'] =
'mysql+pymysql://user:password@123.45.6.78:3306/mydatabase'
```

`.config` is a dictionary attribute that contains configurations for the Flask application. Above, we are setting the Uniform Resource Identifier (URI) by assigning a string value to the key `'SQLALCHEMY_DATABASE_URI'` that contains information about:

- The type of database it is connecting to. (`mysql+pymysql://`)
- The user and password it is connecting with. (`user:password`)
- The location of the database server. (`@123.45.6.78:3306`)
- The exact database hosted on the server. (`/mydatabase`)

Tutorial

There is no tutorial for this module.

Exercises

There are no exercises for this module.