Python 101

Embedded Databases, ORM and Django - An Introduction

Contents

Embedded Databases

An Embedded Database is tightly integrated with the application that requires access to this database.

SQLite is an example of such databases. These are often very minimalistic.

These databases do not need a server and can reside with the application itself. Eg. In android SQLite resides with application itself.

We can install SQLite3 using the following command on linux-

```
sudo apt install sqlite3 libsqlite3-dev
```

See Examples/ex_sqlite3.py

Embedded Databases (sqlite3 commands)

After installing start sqlite3 on terminal as - >> sqlite3

.open DATABASE_NAME

.help ← ask for help

.databases ← list all databases

.tables ← list all tables in a database

We can perform CRUD using the SQL commands.

ORM (Object Relational Mapping)

We use relational databases using SQL Queries most often which in fact is a very cumbersome process. In order to get the SQL Relations as a programmable entity in our programs we represent the tables in form of classes. The objects of these classes represent a relation, This Objects to Relation Mapping is called ORM.

It makes accessing the SQL databases a very structured and well defined process.

In python we use **SQLAlchemy** to achieve this. `pip3 install sqlalchemy`

`See Examples/ex_sqlalchemy.py`

Web Server vs Application Server

Web Servers are the servers that support communication through HTTP only.

Application Servers are the servers that are not limited to HTTP they can work with RPC, FTP etc.

We can say Web Servers are subset of Application Servers.

Web Servers - Meant to serve static content eg. Webpages.

Application Servers - Mean to expose any business logic.

Mostly WebServers **directly** Serve the content to end user AppServers talk to WebServers and interface with the

What is Django?

Django is a Python Web Framework.

It provides a lot of functionalities bundled together into one, we do not need to separately setup our databases, or spin up an HTTP server separately, or even manage the static content like HTML/CSS/JS files separately. It's all bundled in one.

It is extremely scalable, robust and fast, Websites like instagram, pinterest, disqus and bitbucket use Django as their backend.

disqus has 400 million users whereas instagram has 700 million users. So we can safely say that it can scale high enough for normal project usage.

Django Installation

We can install django either using pip3 install django or

sudo apt install python3-django

We will use django2.0.6

It is the current version with a lot of improvements.

It comes with django-admin tools that helps us in setting up the project.

Django Project an overview

`django-admin startproject PROJECT_NAME` ← to start a project in django it creates a folder with basic settings etc required to run a django project.

`django-admin startapp APP_NAME` ← to create an app inside a project.

An app is a submodule inside a project.

eg. A whole website is a project whereas its components like authentication (handle signin, signout and signup) and dashboard are submodules ie. apps.

`See ex_django/COMMANDS`

Understanding the Project directory structure

Understanding the Settings

Making a Sample Application

Looking at Django Templates and Django's ORM