CARANGUE, LIZAMIE G.
BSCPE -2A

Laboratory Activity No. 1:

Topic: **Introduction to Software Design, History, and Overview**

Title: *Setting Up the Development Environment for Django Project*

Introduction: This activity will guide you through the process of setting up your development environment to start building the Library Management System (LMS) in Django. The process involves installing necessary software, setting up Python and Django, and verifying the installation.

Objectives:

- Install Python and Django on your system.
- Create a virtual environment to manage dependencies.
- Verify the installation by running a simple Django project.

Theory and Detailed Discussion: To develop the Library Management System, we will use the Django framework. Django is a high-level Python web framework that allows developers to create robust web applications quickly and efficiently. Before we can start developing, we need to set up the development environment.

Materials, Software, and Libraries:

- **Python** (version 3.8 or above)
- **Django** (version 4.0 or above)
- **pip** (Python package manager)
- **Text Editor** (Visual Studio Code or PyCharm)
- **Database** (SQLite – comes with Django by default)

Time Frame: 1 Hour

Procedure:

1. **Install Python**:
    1. Go to python.org and download the latest version of Python.

    2. Install Python by following the installation instructions for your operating system.
2. **Install pip** (Python package installer):
    1. Open a terminal and type the following command:

```
python -m ensurepip --upgrade
```

3. **Install Virtual Environment**:
    1. Create a virtual environment for our project to avoid conflicts with global packages.

```
pip install virtualenv
```

    1. Create a new virtual environment:

```
python -m venv library_env
```

    1. Activate the virtual environment:
    1. On Windows:

```
.\library_env\Scripts\activate
```

- On Mac/Linux:

```
source library_env/bin/activate
```

1. **Install Django**:
    - After activating the virtual environment, install Django by running:

```
pip install django
```

2. **Verify the Django Installation**:

  - Run the following command to verify if Django is installed:

```
django-admin --version
```

3. **Create a New Django Project**:
   - Create a new Django project called "library_system":

```
django-admin startproject library_system
```

   - Navigate into the project directory:

```
cd library_system
```

4. **Run the Django Development Server**:

- Start the development server to verify everything is working:

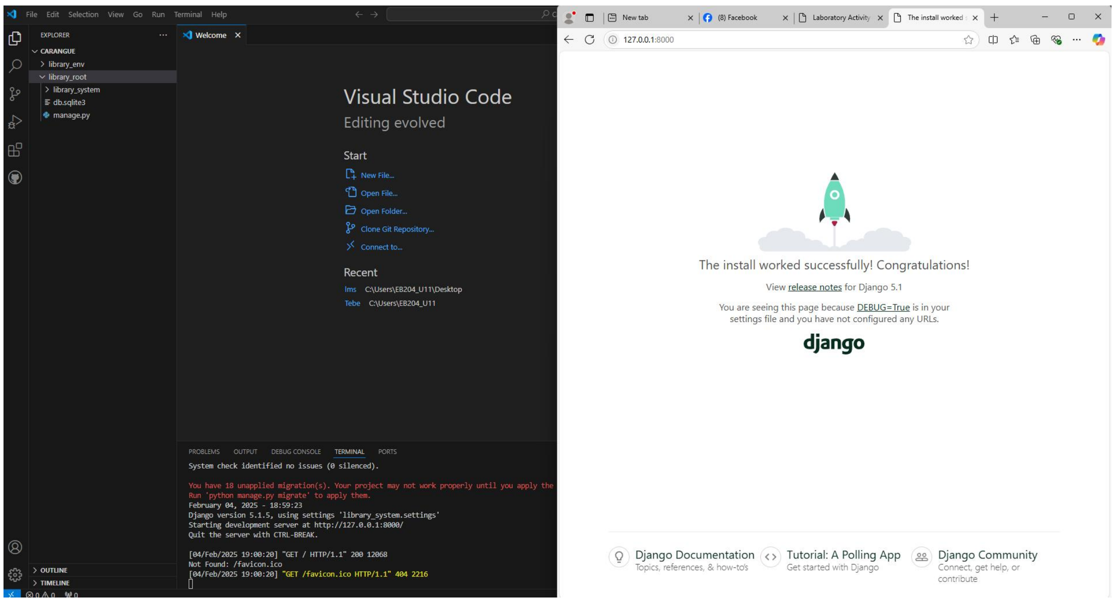
```
python manage.py runserver
```

- Open a browser and go to http://127.0.0.1:8000/. You should see the Django welcome page.

Program/Code: The code here is focused on setting up the environment. The following commands should be run in the terminal:

```
python -m venv library_env
source library_env/bin/activate  #
or .\library_env\Scripts\activate on Windows
pip install django
django-admin startproject library_system
cd library_system
python manage.py runserver
```

Results: (print screen the result and provide the github link of your work)



```
System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the
Run 'python manage.py migrate' to apply them.
February 04, 2025 - 18:59:23
Django version 5.1.5, using settings 'library_system.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

[04/Feb/2025 19:00:20] "GET / HTTP/1.1" 200 12068
Not Found: /favicon.ico
[04/Feb/2025 19:00:20] "GET /favicon.ico HTTP/1.1" 404 2216
```

The install worked successfully! Congratulations!

View release notes for Django 5.1

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

Follow-Up Questions:

1. What is the role of a virtual environment in Django development?

A **virtual environment** in Django development serves as an isolated workspace where project-specific dependencies (like Django versions, third-party libraries, etc.) are installed. This prevents conflicts between different projects that may require different versions of Python packages.

Benefits:

- Ensures dependency isolation, preventing conflicts.
- Allows different Django projects to use different package versions.
- Provides a clean and controlled development environment.
- Simplifies deployment by ensuring consistency between development and production environments.

2. What are the advantages of using Django for web development over other frameworks?

Django is a high-level Python web framework that emphasizes rapid development and clean, pragmatic design. Some advantages over other frameworks include:

✅ **Batteries-included:** Comes with built-in features like authentication, database ORM, admin panel, and security measures.
✅ **Security:** Protects against SQL injection, XSS, CSRF, and clickjacking by default.
✅ **Scalability:** Handles large-scale applications efficiently with caching, middleware, and load balancing support.
✅ **Rapid Development:** Built-in tools like Django Admin and generic views speed up development.

✅ **ORM (Object-Relational Mapping):** Simplifies database management by abstracting SQL queries.
✅ **Large Community & Documentation:** Extensive support, tutorials, and third-party packages available.

Findings:

* A **virtual environment** ensures isolated package management, preventing conflicts between dependencies across different projects.
* Without a virtual environment, projects may experience version mismatches, leading to compatibility issues.
* Django offers a **batteries-included** approach, providing built-in tools such as ORM, authentication, and an admin panel.
* Compared to other frameworks like Flask, Django prioritizes **scalability, security, and rapid development**.
* Django's extensive **documentation and community support** make it an ideal choice for beginners and large-scale applications alike.

Summary:

Django is a powerful web framework that simplifies development by including essential features out of the box. A virtual environment plays a crucial role in managing dependencies, ensuring that projects remain stable and maintainable. Compared to other frameworks, Django's security, scalability, and ease of use make it a top choice for developers.

Conclusion:

Using Django with a virtual environment enhances development efficiency and project organization. Its rich feature set and security-first approach provide a significant advantage over other web frameworks. For projects requiring rapid deployment, scalability, and robust security, Django stands out as a reliable choice.