Task 2: Django Views, Templates, and URL Routing

Objective:
This task focuses on the core components that handle requests and return responses in Django — URLs, function-based views, and templates. You will learn how Django processes an HTTP request from the URL dispatcher to the view, and how the view returns either HTML or other response types.

Questions to Explore

Views

1. What is a view in Django?
2. What's the difference between a **function-based view** and a **class-based view (just try to understand)**?
3. What arguments does a function-based view typically take, and what does it return?
4. How can a view return:
    - HTML content?
    - JSON response?
5. What is HttpResponse vs render()? When do you use each?

Templates

6. What is the role of templates in Django?
7. How do you create and connect a template to a view?
8. Explain the concept of **template context** and how variables are passed from the view to the template.
9. What are template tags and filters? (How we can dynamically show data that we pass to templates)
10. How do you extend templates to avoid repeating HTML code (template inheritance)?

URLs

11. What is the role of urls.py in Django?
12. What is path()?

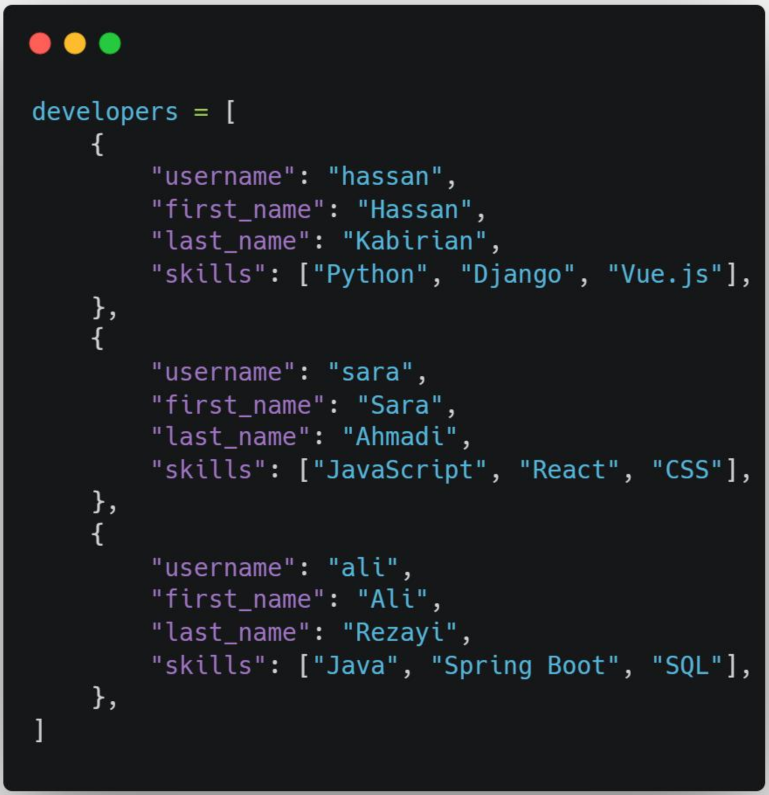
13. How do you name a URL pattern, and why is it useful?

Deliverables

You will implement a small Django app that lists developers and shows their CVs using function-based views, templates, and URL routing.

Data Source

Assume you have a list of developers as dictionaries (you can define this list inside your view file for now):



```python
developers = [
    {
        "username": "hassan",
        "first_name": "Hassan",
        "last_name": "Kabirian",
        "skills": ["Python", "Django", "Vue.js"],
    },
    {
        "username": "sara",
        "first_name": "Sara",
        "last_name": "Ahmadi",
        "skills": ["JavaScript", "React", "CSS"],
    },
    {
        "username": "ali",
        "first_name": "Ali",
        "last_name": "Rezayi",
        "skills": ["Java", "Spring Boot", "SQL"],
    },
]
```

Requirements

1. **Developers List Page (/developers/)**

- **Create a function-based view that passes the developers list to a template (developers_list.html).**
- **The template should display each developer's first name, last name, and username in a list.**
- **Each developer should have a "View CV" link that takes the user to their personal CV page.**

---

2. **Developer CV Page (/developers/<username>/)**
   - **Create a function-based view that takes username as a parameter.**
   - **The view should find the developer dictionary by username and render a template (developer_cv.html).**
   - **The CV page should display:**
     - **Full name**
     - **Username**
     - **Skills (as a bullet list)**
   - **Add a "Back to Developers List" link that takes you back to /developers/.**

If handling developer pages dynamically (e.g., /developers/<username>/) feels difficult, you may define the URLs statically instead.

---

4. **Templates with Inheritance (Bonus – Extra Point)**
   - **Create a base.html with a header and {% block content %} section.**
   - **Make developers_list.html and developer_cv.html extend base.html.**

GIT:

https://www.youtube.com/watch?v=vA5TTz6BXhY

or

https://www.youtube.com/watch?v=rH3zE7VlIMs

(watch section you want)

Django:

https://www.udemy.com/courses/search/?q=python+django&src=sac&kw=Django&sort=most-reviewed (select the one that is best for you and use downloadly)

https://www.w3schools.com/django/

https://www.djangoproject.com/start/ (using with notebook LM)