JavaScript with Python (Django)

1. Using JavaScript for Client-Side Interactivity

- **JavaScript (JS)** runs on the **client-side (browser)** and is used to make web pages interactive.
- In Django, **Python handles server-side logic** (like fetching data, processing forms), while JavaScript handles **dynamic behavior on the webpage**, such as:
  - Form validation before submission
  - Animations and visual effects
  - Showing/hiding elements
  - Sending asynchronous requests to the server (AJAX)
- JS enhances user experience without needing a full page reload.

---

2. Linking JavaScript Files in Django

Step 1: Place JS files in the static folder

```
my_app/
 ├– static/
|  └– my_app/
|     └– js/
|        └– script.js
 ├– templates/
|  └– my_app/
|     └– index.html
```

Step 2: Load static files in the template

```
{% load static %}
<!DOCTYPE html>
<html>
<head>
  <title>My Page</title>
```

```
</head>
<body>
   <h1>Hello, Django!</h1>

   <!-- Link external JS file -->
   <script src="{% static 'my_app/js/script.js' %}"></script>

   <!-- Internal JS code -->
   <script>
      console.log("Hello from inline JS!");
   </script>
</body>
</html>
```

Key Points:

- {% load static %} is required to use {% static %} for file paths.
- **External JS** files are preferred for organization and reusability.
- **Internal JS** (inline inside <script> tags) can be used for small scripts.

---

3. Best Practices

- Keep JS files in static/my_app/js/ folder.
- Minimize inline JS in templates; use external files for maintainability.
- Use JS for client-side interactivity; Python handles server-side operations.