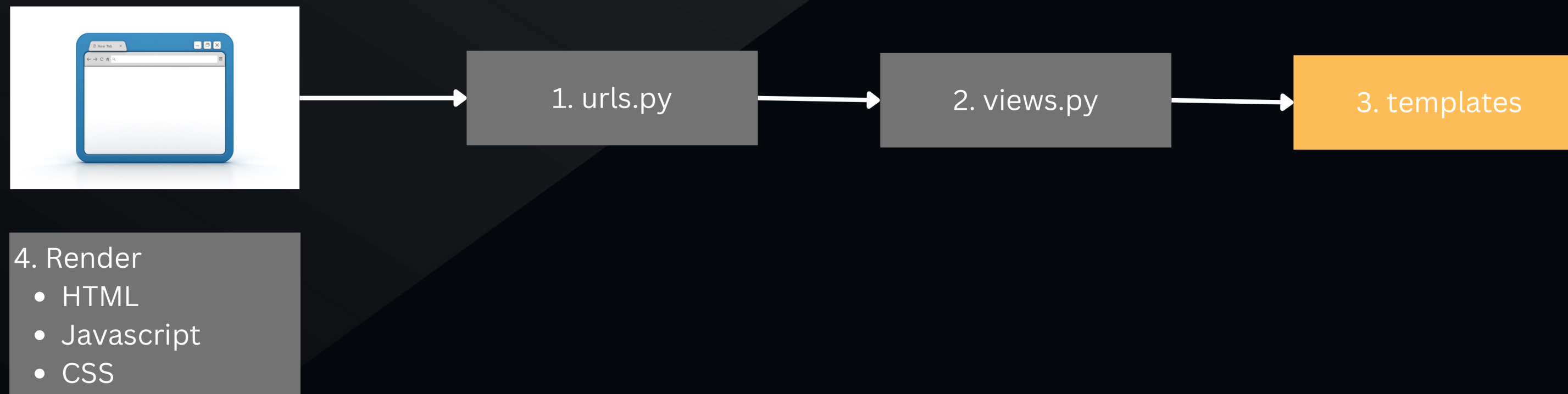


TEMPLATES

- A Django template is a text document or a Python string marked-up using the Django template language
- Syntax is similar to Jinja2
- Can load HTML, CSS and Javascript



TEMPLATES

- **render(request, template_name, context=None, content_type=None, status=None, using=None)**
 - Combines a given template with a given context dictionary and returns an HttpResponse object with that rendered text.
- **Steps**
 - Add the app into **INSTALL_APPS** in settings.py
 - Create **templates** directory under the app
 - import render from django.shortcuts, then pass the template file

TEMPLATES

- **Templates structure**

- templates in root

- DJANGO_ROOT/templates/<APP_NAME>/<TEMPLATES_FILE>

- easy to load entire statics to CDN(unless you use staticfiles)

- templates under the app

- <APP_NAME>/templates/<APP_NAME>/<TEMPLATES_FILE>

- good for packaging individual app

```
db.sqlite3
manage.py
mysite
├── __init__.py
├── __pycache__
│   ├── __init__.cpython-310.pyc
│   ├── settings.cpython-310.pyc
│   ├── urls.cpython-310.pyc
│   ├── views.cpython-310.pyc
│   └── wsgi.cpython-310.pyc
├── asgi.py
├── settings.py
├── urls.py
├── views.py
└── wsgi.py
polls
├── __init__.py
├── __pycache__
│   ├── __init__.cpython-310.pyc
│   ├── admin.cpython-310.pyc
│   ├── apps.cpython-310.pyc
│   ├── models.cpython-310.pyc
│   ├── urls.cpython-310.pyc
│   └── views.cpython-310.pyc
├── admin.py
├── apps.py
├── migrations
│   ├── __init__.py
│   ├── __pycache__
│   │   └── __init__.cpython-310.pyc
├── models.py
├── templates
│   └── polls
│       └── main.html
├── tests.py
├── urls.py
└── views.py
```