

# COMP 019 - Applications Development and Emerging Technologies

*Full-Stack Python Development*

## SESSION 2: DJANGO FUNDAMENTALS

### ACTIVITY 2.1: DJANGO PROJECT SETUP AND STRUCTURE

**Topic:** Creating and understanding Django project structure

**Description:** Create your first Django project and understand its file structure

#### INSTRUCTIONS:

Create Django project:

- `django-admin startproject myproject`
- `cd myproject`
- `python manage.py runserver`
- Access `http://127.0.0.1:8000` in browser

Understand project structure:

- `manage.py` - Command-line utility
- `settings.py` - Project configuration
- `urls.py` - URL routing
- `wsgi.py` / `asgi.py` - Server interfaces

Create a Django app:

- `python manage.py startapp blog`
- Register app in `INSTALLED_APPS`

Understand app structure:

- `models.py` - Database models
- `views.py` - Business logic
- `urls.py` - App-specific routes
- `templates/` - HTML templates

**Deliverables:** Submit project folder structure screenshot and running server screenshot

**25 Points**

### ACTIVITY 2.2: MODELS AND DATABASE

**Topic:** Creating Django models and database migrations

**Description:** Define data models and work with Django's ORM

#### INSTRUCTIONS:

Create models in `models.py`:

- Create 'Post' model with fields:
  - title (CharField, max\_length=200)
  - content (TextField)
  - author (CharField)

- created\_at (DateTimeField, auto\_now\_add=True)
- updated\_at (DateTimeField, auto\_now=True)
- Create 'Comment' model with fields:
  - post (ForeignKey to Post)
  - text (TextField)
  - commenter (CharField)
  - created\_at (DateTimeField)

Run migrations:

- python manage.py makemigrations
- python manage.py migrate

Test with Django shell:

- python manage.py shell
- Create, read, update, delete objects
- Query with filters

**Deliverables:** Submit models.py and shell operation screenshots

**30 Points**

## ACTIVITY 2.3: VIEWS AND TEMPLATES

**Topic:** Creating views and rendering templates

**Description:** Build views to handle requests and templates to display content

### INSTRUCTIONS:

Create views in views.py:

- post\_list - Display all posts
- post\_detail - Display single post with comments
- Use render() to return templates

Create templates:

- Create templates/blog/ folder
- base.html - Base template with common structure
- post\_list.html - List all posts (extends base)
- post\_detail.html - Single post view

Use Django template language:

- {% extends 'base.html' %}
- {% block content %}{% endblock %}
- {{ post.title }}, {{ post.content }}
- {% for post in posts %}...{% endfor %}
- {% if %}...{% endif %}

Configure URLs:

- Create blog/urls.py
- Include in main urls.py
- Test all routes in browser

**Deliverables:** Submit views.py, templates, and browser screenshots

**30 Points**

**TOTAL POINTS: 85**