

# COMP 019 - Applications Development and Emerging Technologies

*Full-Stack Python Development*

## SESSION 7: REST API WITH DJANGO REST FRAMEWORK

### ACTIVITY 7.1: DRF SETUP AND SERIALIZERS

**Topic:** Introduction to Django REST Framework

**Description:** Set up DRF and create serializers for your models

#### INSTRUCTIONS:

Install and configure DRF:

- pip install djangorestframework
- Add 'rest\_framework' to INSTALLED\_APPS
- Configure REST\_FRAMEWORK settings

Understand serializers:

- Convert model instances to JSON
- Validate incoming data
- Similar to Django Forms but for API

Create serializers.py:

- PostSerializer using ModelSerializer
- CommentSerializer with nested post info
- Specify fields to include/exclude
- Add read\_only\_fields

Test serializers in shell:

- Serialize a Post object
- Deserialize JSON to Post
- Validate data
- Handle validation errors

**Deliverables:** Submit serializers.py and shell test screenshots

**25 Points**

### ACTIVITY 7.2: API VIEWS AND VIEWSETS

**Topic:** Creating API endpoints with DRF views

**Description:** Build RESTful API endpoints for your application

#### INSTRUCTIONS:

Create API views:

Option 1 - Function-based views:

- @api\_view(['GET', 'POST'])
- Handle different HTTP methods

Option 2 - Class-based views:

- APIView with get(), post() methods

- Generics: ListCreateAPIView, RetrieveUpdateDestroyAPIView

Option 3 - ViewSets (Recommended):

- ModelViewSet for full CRUD
- Automatic URL routing
- Less code, more functionality

Create API endpoints:

- GET /api/posts/ - List all posts
- POST /api/posts/ - Create new post
- GET /api/posts/{id}/ - Get single post
- PUT /api/posts/{id}/ - Update post
- DELETE /api/posts/{id}/ - Delete post

Configure URLs with DefaultRouter

**Deliverables:** Submit views.py, urls.py, and API endpoint tests

**35 Points**

## ACTIVITY 7.3: API AUTHENTICATION AND TESTING

**Topic:** Securing and testing your REST API

**Description:** Add authentication to API and test all endpoints

### INSTRUCTIONS:

Configure API authentication:

- Token Authentication setup
- Add to REST\_FRAMEWORK settings
- Create tokens for users
- Login endpoint returning token

Protect API endpoints:

- permission\_classes in views
- IsAuthenticated - logged in users
- IsAuthenticatedOrReadOnly - read public, write protected
- Custom permissions (IsOwnerOrReadOnly)

Test API with tools:

- DRF Browsable API (browser)
- Postman or Insomnia
- curl commands

Test all endpoints:

- Without authentication (should fail for protected)
- With token in header
- Document API with examples

**Deliverables:** Submit authentication code and Postman/API test screenshots

**30 Points**

**TOTAL POINTS: 90**