SIX PHRASE
Democratizing Skilling
a Veranda Enterprise

RESTful APIs

Unit -6

Lesson Plan

Subject/Course	Python Programming
Lesson Title	RESTful APIs

Lesson Objectives
About RESTful APIs
Implementing RESTful APIs
Building RESTful and Secure APIs

Objectives

- Understand RESTful APIs and their importance in web development.
- Explore REST architecture principles and constraints.
- Learn HTTP methods, status codes, and request–response flow.
- Design and implement RESTful APIs using Python (Flask & FastAPI).
- Apply CRUD operations with GET, POST, PUT, and DELETE.
- Structure URLs and resource representations using JSON/XML.
- Implement best practices: versioning, error handling, and scalability.
- Test APIs using tools like Postman, cURL, and Python requests.
- Build secure APIs with JWT, OAuth2, CORS, and HTTPS.
- Integrate REST APIs into full stack applications (frontend–backend).

Introduction to RESTful APIs and REST Architectural Style

- REST (Representational State Transfer) is an architectural style for designing networked applications.

  Key Principles:
  - Stateless Communication: Each request is independent.
  - Client-Server Separation: Frontend and backend are decoupled.
  - Cacheable Responses: Improves performance.
  - Layered System: Proxies, gateways can exist.
  - Uniform Interface: Uses standard HTTP verbs (GET, POST, PUT, DELETE).

  Example URLs:
  /api/v1/users – All users
  /api/v1/users/1 – Single user

Understanding the HTTP Protocol and Its Role in RESTful APIs

- HTTP (HyperText Transfer Protocol) enables client-server communication in REST.

  Key Elements:
  - Request: Method + URL + Headers + Body
  - Response: Status Code + Headers + Body

  Common Methods:
  GET: Retrieve
  POST: Create
  PUT: Update
  DELETE: Remove

  Example:
  GET /api/v1/users HTTP/1.1
  Host: example.com
  Accept: application/json

Implementing RESTful APIs

- Flask CRUD Example:

```
from flask import Flask, jsonify, request
app = Flask(__name__)
users = []

@app.route('/users', methods=['GET'])
def get_users():
    return jsonify(users)

@app.route('/users', methods=['POST'])
def add_user():
    data = request.get_json()
    users.append(data)
    return jsonify(data), 201

@app.route('/users/<int:id>', methods=['PUT'])
def update_user(id):
    users[id].update(request.get_json())
    return jsonify(users[id])
```

Using URLs and Resource Representations

- Resources are identified by URLs and represented in formats like JSON or XML.

  Example:
  GET /api/v1/books/1

  Response (JSON):
  ```
  {
    "id": 1,
    "title": "RESTful Design",
    "author": "Roy Fielding"
  }
  ```

  Content Negotiation:
  Accept header defines format (application/json, application/xml).

Implementing Best Practices (Status Codes, Versioning, Error Handling)

HTTP Status Codes:
200 OK: Success
201 Created: Resource created
400 Bad Request: Invalid input
404 Not Found: Missing resource
500 Internal Server Error

Versioning:
- URI: /api/v1/users
- Header: Accept-Version: v1
- Query: /users?version=1

Error Handling Example:

```
@app.errorhandler(404)
def not_found(e):
    return jsonify(error='Not Found'), 404
```

Consuming RESTful APIs (cURL, Postman, Requests Library)

- cURL Example:
  curl -X GET https://api.example.com/users

  Postman:
  GUI tool to test APIs interactively.

  Python requests:
  import requests
  r = requests.get('https://api.example.com/users')
  print(r.json())

  r = requests.post('https://api.example.com/users', json={'name': 'John'})
  print(r.status_code)

Building Scalable and Secure RESTful APIs (Flask & FastAPI)

Scalability:
- Stateless
- DB optimization
- Caching
- Load balancing

Security:
- HTTPS
- JWT/OAuth2
- Input validation
- CORS

FastAPI Example:

```
from fastapi import FastAPI
app = FastAPI()
```

Summary and Key Takeaways

- REST uses HTTP for data communication.
- Flask and FastAPI simplify API development.
- Follow best practices: correct methods, codes, versioning, and security.
- Use tools like Postman & cURL for testing.
- RESTful APIs are the backbone of modern full-stack apps.

Thank You