

**RESTful API Design and Development**

**Course Number:** PROG-102
**Duration:** 2 days

**Overview**

This RESTful API Design and Development course teaches attendees the core concepts and practices for RESTful API design and development.
The course focuses on the principles for designing a RESTful JSON API. Core concepts and best practices are covered for both development and consumption of a RESTful API.

**Prerequisites**

Other than object-oriented development or architecture experience, no prior expertise is presumed.

**Materials**

All RESTful API Design and Development students will receive comprehensive courseware including slides, design exercises, and sample solutions.

**Software Needed on Each Student PC**

* Google Chrome
* Other modern browsers as desired
* Other free software and lab files that Accelebrate would specify, including Node.js

**Objectives**

* Consider web architecture as the basis of API strategies
* Understand the REST architectural style and its motivations
* Utilize the Richardson Maturity Model as a way to discuss design choices and induced properties
* Implement security measures
* Work with client technologies
* Incorporate API management approaches
* Implement testing strategies
* Think about REST APIs vs. linked data, RPC models and GraphQL

**Outline**

* Introduction
	+ API Breakdown of Players
	+ How did we get here?
		- Past
		- REST and JSON
* Current state of API
	+ API Strategies
		- REST
		- GraphQL & GRPC
	+ OpenAPI
		- Tools
		- Definitions and Standards
		- Demos and Labs
* API Design
	+ General design concepts (separation of concern, error handling, API vs. RPC, etc.)
* REST API Key Concepts
	+ State
	+ Nouns
	+ Verbs Overview
* Paths
	+ GET
	+ Design in OpenAPI
	+ Verbs Details
	+ Query String Parameters
* Separation of Concerns
	+ Gathering
	+ Formatting
	+ Delivery
	+ Security and Scalability
	+ Operation Responses
	+ OpenAPI Schema
* Error Definitions
	+ Demo Server
		- Typicode
		- GitHub
* Richardson Maturity Model
	+ Levels 0-3
	+ Hypermedia/HATEOAS
* Components Section
	+ Reuse
	+ Response
* Example APIs
* API Development Standards
	+ OpenAPI
	+ Benefits
	+ Maturity
	+ Parameters
	+ POST, PUT and PATCH
	+ HEAD, OPTIONS and TRACE
* OpenAPI Tools
	+ Overview
	+ Insomnia
	+ Local Typicode Server
* CRUD: Create
	+ POST Create
	+ Request Body Schema
	+ Combining Schemas
	+ Responses in Components
	+ Array Schema Type
* Server Variables
* Path Parameters
	+ Path
* Parameters
	+ Path, header and cookie
	+ Required, default, min/max, enum
	+ Path Level Reuse
	+ Serialization
	+ Styles and Explode
* CRUD: Delete
* Design First vs. Develop First
* Automated Testing
	+ Tools, Insomnia
	+ Chai JS Assertion Library
* Legacy API Considerations
	+ Database
	+ Endpoints
	+ Functionality
* API Versioning
	+ URL vs. Header
* Security
	+ Documentation Options
	+ Authentication vs. Authorization
	+ JWT
	+ Security Definitions in OpenAPI
		- Types
		- Flow
		- Fields
* Pet Store API Review
* Automated Testing Principles
	+ Best Practices
	+ Integration Testing
	+ Insomnia CLI
	+ GitHub Actions
* Tutorials
	+ Jetty Web API Servlet
	+ C# .NET Server
	+ Java Server using Spark
	+ Python Server using Flask
* Conclusion