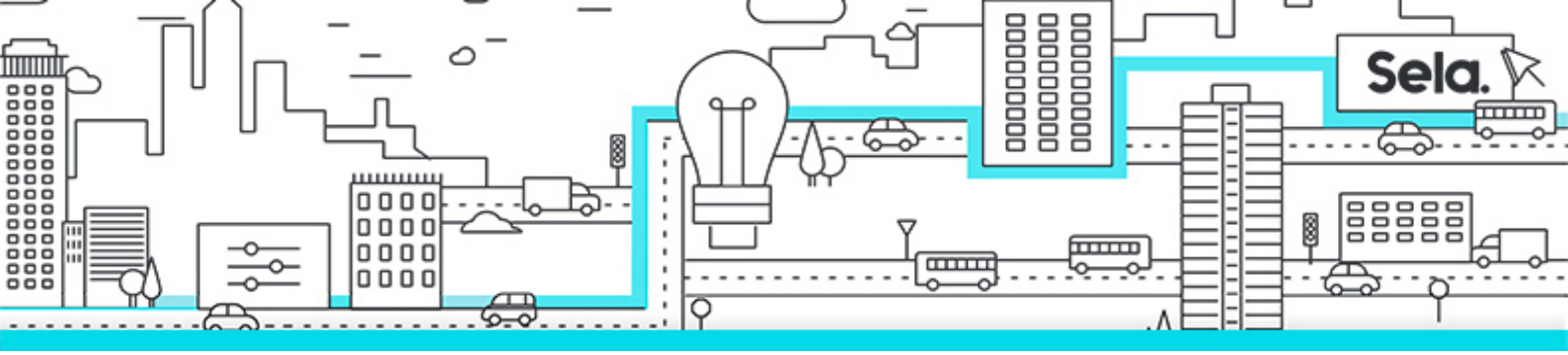
Sela.

MicroServ

Microservices

college@sela.co.il 03-6176666



Microservices

MicroServ - Version: 1

5 days course

Description:

A very important part of modern software development is the new discipline of Cloud-enabled microservices, and part of this is PAAS (Platform as a Service) which allows deploying such apps into production with the minimum effort. This course will introduce the participants into microservices and to its implementation with various frameworks

Intended Audience:

Developers, architects

Prerequisites:

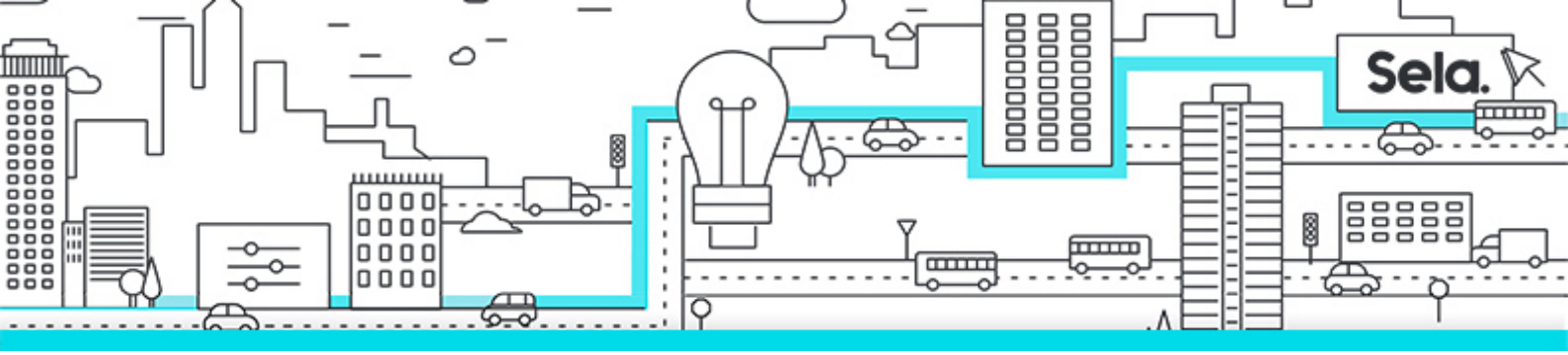
• Comfortable with command-line operations Familiar with software development

Objectives:

• Understand modern software development for stable fault-tolerant and scalable applications using microservices Understand the role of PAAS and PCF Being able to develop microservices and deploy it with PCF

Topics:

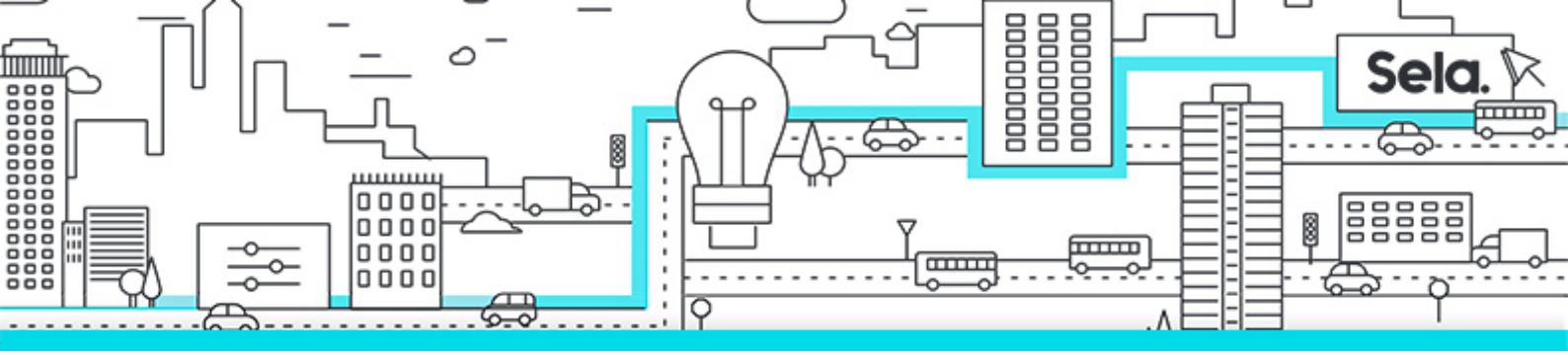
- **Course Intro**
  - PAAS - Platform as a Service
  - Monoliths versus Microservices
  - Microservices History
  - Pivotal Cloud Foundry



- **Microservice Fundamentals**
  - What is a Microservice Architecture
  - What is a Microservice
  - Benefits of Microservices
  - Downsides of Microservices
  - Use Cases for Microservices
  - Monolithic Architecture
  - Distributed Architecture
  - Service-oriented Architecture
  - Microservice and API Ecosystem
  - SOA vs. Microservices
  - Microservice & API

- **Containers and Docker**
  - Container Fundamentals
  - What is a Container
  - What is Docker
  - Hypervisor Virtual Machines
  - How containers work
  - Containers and Microservice Architectures
  - Getting Started with Docker
  - Labs

- **Getting Started With Docker**
  - Installing Docker – Native Linux
  - Installing Docker – Other Operating Systems
  - Docker Toolbox
  - Docker Machine Basics
  - Running your first Container
  - Developing a Microservice



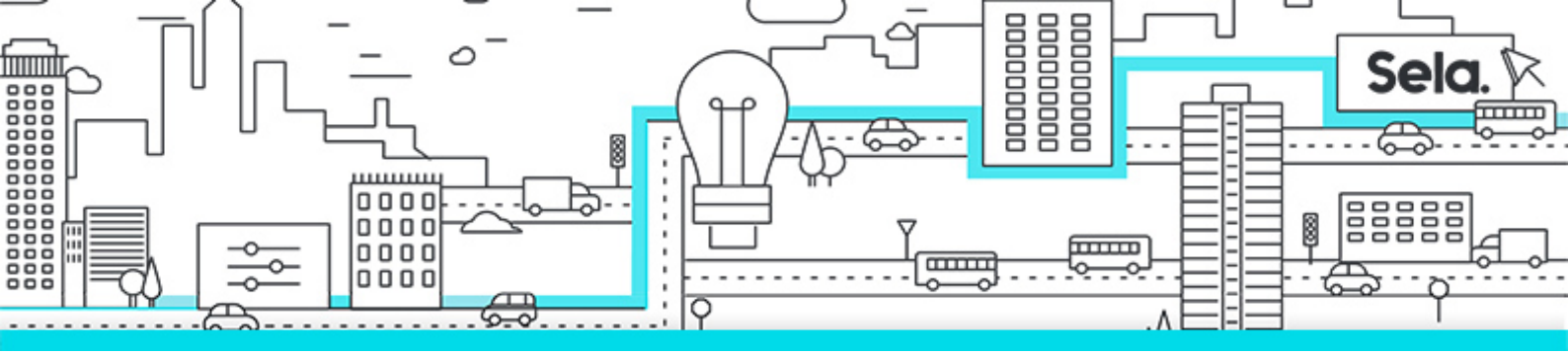
- **Dockerfile**
  - Instructions and images
  - FROM
  - RUN
  - Building Images
  - The Build Context
  - Adding files to an Image
  - Executing Commands
  - Specifying an Entrypoint

- **Docker Port Mapping**
  - Multi Container Hosting
  - Automatic Port Mapping
  - Specific Port Mapping

- **Deployment Patterns**
  - Service instance per host
  - Multiple service instances per host
  - Service instance per VM
  - Service instance per container

- **Communication Patterns**
  - API Gateway
  - Partial Failures
  - Circuit Breaker

- **Service Discovery Patterns**
  - Client Side Discovery
  - Server Side Discovery
  - Service Registry
  - Self Registration

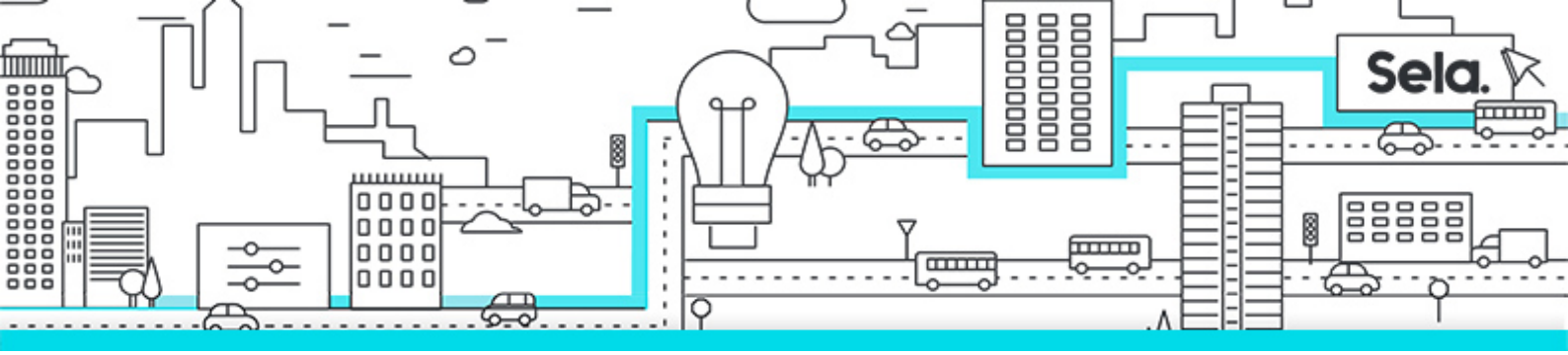


- 3rd Party Registration

• **Data Management Patterns**
  - Shared Database
  - Database per Service
  - Event Driven Architecture
  - Event Sourcing
  - Transaction Log Tailing
  - Database Triggers
  - Application Events
  - CQRS
  - Domain-Driven Design (DDD)

• **DockerHub**
  - What is Docker Hub
  - Creating an account
  - Creating a Repository
  - Markdown Format
  - Pushing an Image
  - Integrating Multiple Microservices
  - Labs

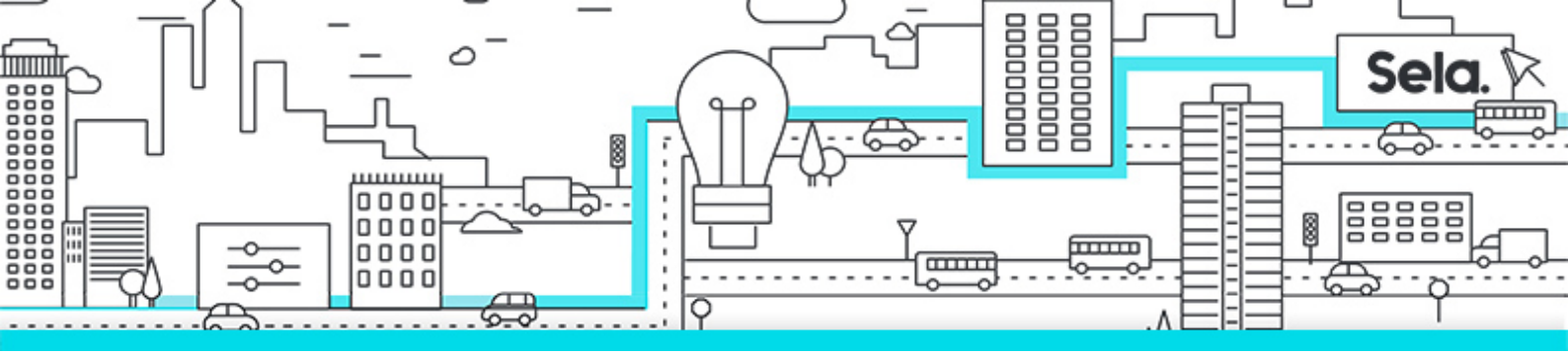
• **Microservices Detail**
  - The Monolith vs Microservices overview
  - Cloud-Native Apps
  - High Availability, Scalability, Efficiency
  - Code repositories (such as Git) (covered as needed)
  - Continuous Integration, Continuous Delivery (CI/CD) (such as Jenkins) (covered as needed)
  - How Pivotal Cloud Foundry Enables Continuous Delivery



- **Microservices Security**
  - Why it's so important
  - Microservice Security Principles
  - Access Tokens
  - Oauth 2.0
  - Kerckhoff's Principle
  - Shannon's Maxim
  - Security through obscurity
  - General Security Considerations
  - Middleware Security Considerations
  - Edge Services Security Considerations
  - Web and Other Client Security Considerations
  - People and Process Security Considerations
  - REST Interfaces to MicroServices
  - What is a RESTful Web service?
  - HTTP verbs
  - HTTP response codes
  - Versioning Strategy
  - Richardson Maturity Model
  - Example scenario
  - Key Principles of RESTful Web Services
  - Using JAX-RS

- **Spring Framework**
  - Review of Spring Essentials
  - Why Spring
  - Configuration using Spring
  - Bean creation
  - Data Management

- **Spring Boot Introduction**
  - Introduction to Spring Boot



- Value Proposition of Spring Boot
- High-level Spring Boot features
- Creating a simple Boot application using Spring Initializer web-site
- Spring Boot Dependencies, Auto-Configuration, and Runtime
- Dependency management using Spring Boot starters
- How auto-configuration works
- Configuration properties
- Overriding auto-configuration
- Using CommandLineRunner
- JPA With Spring and Spring Data

• Quick Intro to ORM with JPA

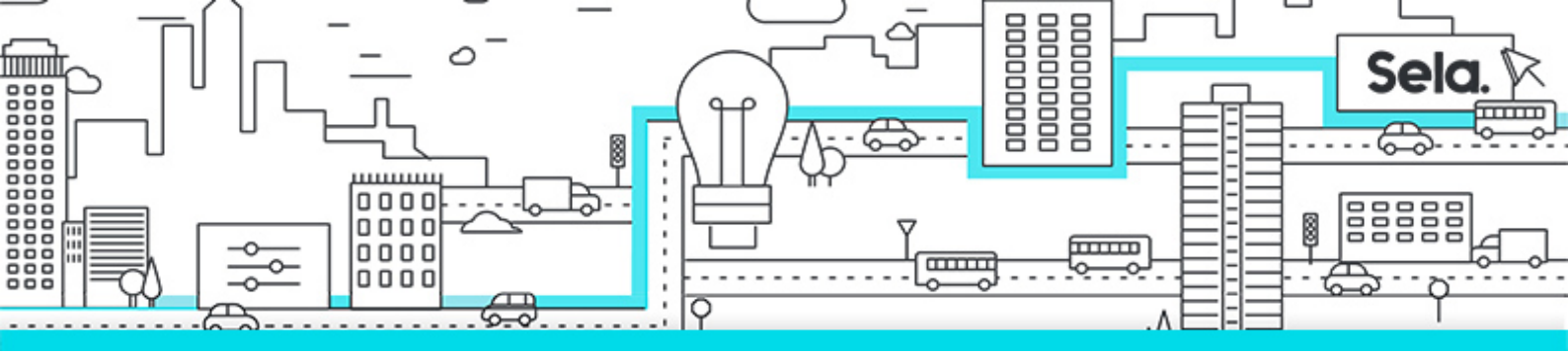
- Benefits of Using Spring with JPA
- Configuring Spring JPA with Spring Boot
- Spring Data JPA dynamic repositories.
- Spring MVC Architecture and Overview
- Introduction to Spring MVC and Request Processing
- Controller method signatures
- Using @Controller, @RestController, and @GetMapping annotations
- Configuring Spring MVC with Spring Boot
- Spring Boot packaging options: JAR or WAR
- REST With Spring MVC

• An Introduction to the REST Architectural Pattern

- Controlling HTTP Response codes with @ResponseStatus
- Implementing REST with Sprign MVC, @RequestMapping, @RequestBody and @ResponseBody
- Spring MVC's HttpMessageConverters and automatic content negotiation

• Container Orchestration with PKS

- Introducing PKS Container Orchestration
- Deploying the App in PCF and PKS



- PKS Overview

- **MongoDB**
  - Introducing MongoDB
  - NoSQL vs NewSQL vs SQL
  - MongoDB and Containers
  - MongoDB and Microservices

- **Project: Creating Containerized Spring Boot App**
  - Create Sample App With Spring Boot
  - Create Docker Image
  - Deploy Docker Image to PKS

- **Project: Conversion of Monolithic App to Microservices**
  - Create Monolithic App With Spring Boot
  - Rearchitect to Microservices
  - Containerize
  - Deploy to PCF