Basic	Intermediate	Advance
• OS • Install Ubuntu • Install Intellij Idea • Install Slack • Install sdkman • Install Docker Desktop • Use sdkman to install • Java • Maven • Java Basic Syntax • Object • Class • Interface • Package • Methods • Instance variables • Intellij Idea • Run Simple Program • Use this IDE for all the following concepts • Learn shortcuts • How to add module in IntelliJ Idea • Git & Github • Create Github Account • Use git inside Intellij • configure GitHub • create change list • commit change list • push change list • view git logs • create new branch • change branch • View all branches • Use git inside Intellij to commit all the following concepts • Use git in CLI • git init • git add • git commit • git push	• Maven • Dependency Management • Maven compiler plugin to compile Java project • JDBC • Data Source & its different implementations (Postgres & mySQL) • Common Design Patterns • Creational Design Patterns • Factory Method • Singleton • Prototype • Builder • Structural Design Pattern • Facade • Proxy • Decorator • Behavioral Design Pattern • Strategy • Observer • Postman • interact with APIs • Docker • run hello-world container • build an image • docker-file • docker-compose • Hibernate • Basic • Session Factory • Session • Transaction • hibernate.cfg.xml • @Entity, @Table, @Id, @Column, @GeneratedValue • HQL • Create/Read/Update/Delete an Entity • Mapping • @OneToOne, @OneToMany, @ManyToOne, @ManyToMany, @JoinColumn, @JoinTable • Cascade Type	• Postgres • multi-schema • CRUD operation using SQL • Triggers • Functions • Angular • JavaScript • TypeScript • Angular.js vs Angular • Architecture overview • Angular Basics • Interpolation • Property Binding • Event Binding • Components • Styling • #ref variables • ngIf, <ng-template> • ngFor • ngClass, ngStyles, • Pipes • Templates • How Angular Render HTML (virtual DOM) • Forms, Inputs, Validations • Services & HTTP • Routing & Gaurds • SSO (OIDC) with KC • Unit Tests • Junit-5 • Integration Tests • Mockito • Test Containers • Jhipster • Create a micro-service architecture • A Front-end with API gateway • 3 microservices • Consul discovery server & Client • OIDC with KeyCloak • Feign Client

    - git branch
    - git pull
    - git fetch
    - git add remote
  - Create a separate module for each following concept and commit
- Objects & Classes
- Constructor
- Data Type
  - Primitive Data Type
  - Reference/Object Data Types
- Variable Types
  - Instance variables
  - Class (static) variables
- Modifier Types
  - Access Control Modifier
  - Non-access control Modifier
- Operators
  - Arithmetic Operators
  - Relational Operators
  - Bitwise Operators
  - Logical Operators
  - Assignment Operators
  - Conditional Operator
  - Precedence of Operators
- Loops
  - for
  - for each
  - while
  - do-while
- Decision Making
  - if
  - if-else
  - else
  - nested if
  - switch statement
  - switch expression
- Java Methods
- Java Strings, Number & Character
- Java Arrays
- Java Data & Time
- Regular Expression
- Files/IO

    - Fetch Type
      - Lazy
      - Eager
      - HQL Join Fetch
    - One to One Unidirectional & bidirectional
    - One to Many Unidirectional & bidirectional
    - Many to Many
- Spring Framework
  - Spring Core
    - DI
    - IOC
      - Spring BeanFactory Container
        - XmlBeanFactory
        - DefaultListableBeanFactory
      - Spring ApplicationContext Container
        - ClassPathXmlApplicationContext
        - AnnotationConfigApplicationContext
        - WebXmlApplicationContext
      - Event Handling on Application Context
        - ContextRefreshedEvent
        - ContextStartedEvent
        - ContextStoppedEvent
        - ContextClosedEvent
        - RequestHandledEvent
    - Spring Beans
      - Beans Definition
        - Beans Properties
          - class
          - name
          - scope
          - constructor-arg
          - properties
          - auto wiring mode
          - lazy initialization mode
          - init method
          - destroy method
      - Bean Scope
        - singleton
        - prototype
        - request
        - session

  - each service (server) is containerized
  - docker-compose for the whole mesh
  - 95% test coverage
  - API test suit using Postman

- Exceptions Handling
- Inner Classes
- Java Stream API
  - Lambda Expression
  - map & filter operation
  - Optional
- Java Collection Framework
- Java Generics
- Java Serialization
- OOPS concept
  - Polymorphism
  - Inheritance
  - Encapsulation
  - Abstraction
- SOLID Principles
  - Single Responsibility Principle
  - Open/Closed Principle
  - Liskov Substitution Principle
  - Interface Segregation Principle
  - Dependency Inversion Principl

    - global-session
  - Bean Lifecycle
    - Java Interfaces
      - InitializingBean
      - DisposableBean
    - XML props
      - init-method
      - destroy-method
  - Bean Post Processor
  - Bean Definition Inheritance
  - Bean Definition Temlate
- XML based configuration
  - Bean Creation
  - Dependency Injection
    - Constructor based dependency Injection
    - Setter based dependency Injection
    - Inner Bean Injection
    - Collection Injection
  - Auto-wiring
    - byName
    - byType
    - constructor
    - auto-detect
- Annotation-based configuration
  - Bean Construction
    - @Configuration
      - @Bean
      - @Import
    - @ComponentScan
      - @Component
      - @Service
      - @Controller
      - @Respoistory
  - Auto wiring
    - @Autowired
      - constructor
      - setter
      - field
    - @Qualifier
    - @Required
    - JSR-250 annotations

- Spring Transactions
    - @Transaction annotation
        - propagation
- Spring AOP
- Spring JDBC template
- Spring Hibernate template
- Spring MVC
    - Dispatcher Servlet
    - Controller & Rest Controller
    - Thymeleaf (View)
    - XML based configuration
        - WEB.XML file and deployment in tomcat
    - Annotation-based configuration
        - @EnableWebMvc
        - AbstractAnnotationConfigDispatcherServletInitializer
        - @RestController
            - @GetMapping
            - @PostMapping
            - @PutMapping
            - @DeleteMapping
    - maven-war plugin for creating war
- Spring Security
    - Authentication & Authorization
        - Learn about OAuth2 & OIDC
        - Spin up KeyClaok server using Docker container
        - Authenticate User of Spring App using KC
        - Authorize User (role-based, scope based) of Spring App using KC
- Spring Boot
    - spring-boot-starter-web
    - spring-boot-starter-data-jpa
    - Spring Cloud
        - spring-cloud-starter-gateway
        - spring-cloud-starter-netflix-eureka-server
        - spring-cloud-starter-netflix-eureka-client
        - spring-cloud-starter-openfeign

	◦ create 2 micro-services talking by API gateway • Spring Security ◦ spring-boot-starter-security ◦ spring-boot-starter-oauth2-resource-server • create two micro-services along with API gateway, talking to each using API gateway, secure with KC server (OIDC). Containerized using Docker and one-click setup using docker-compose	