AAD - 68
Phase 2 - API develop with Spring framwork
Start: July. 28, 2024

Main scope	Sub scopes	What should able to manage and what have to know
Intro to Spring FW	What is Spring	Spring is a main Java framework
	Why Spring	Explain why it is Spring important when compare with Java EE
	Main Spring projects	Point out some Spring projects and those are in Spring FW umbrella (this stage knows only the names. selected projects to be implemented) - Spring Core - Spring Web - Spring Data - Spring Security
	Structure of Spring FW - Explain with a diagram	Focus 3 main areas as Core, Web, Data
	Program model highlight of Spring	Inspire declarative programming with Java IOC and use the concept of Context
	Basic concepts of Spring	Have an idea with the following - Spring Beans and POJO concept - Context and types of Spring context
		ApplicationContext vs BeanFactory (more explained on Feb 10)
Model structure for Spring Core app	Key classes of a project with the following the naming convention	**AppInitializer** - Include the main method and create context
		AppConfig - Config class - Source for bean definition - Agent to collect components with scan the component/s - Annotations of ***ComponentScan*** and ***Configuration*** and their duties - base package and base-package class params in ComponentScan
		Component classes - Annotated with Component annotation (Given control to Spring is after annotated as Component)
	Context-based functions in practice	Usage of the methods like, ***refresh(), getBean(), isSingleton()..*** Change scopes with **Scope** annotation - There are 4 main scopes; Singleton, Prototype, Session, and Request (Session, and Request found in Web App) - Diff between close() and registerShutdownHook() - See the IOC with create object none of use **new** k/w -Full mode and light mode
	Bean vs Component	Diff between these two annotations. But both used to give control to Spring Use rarely **new** k/w when using Bean annotated methods
Bean ambiguty and how it is resolve	What is bean ambiguty and practice the ways to resolve it	Usage of **@Primary,@Qualifier** and create custom annotation with use of retention policy
Explain about AOP (Aspect Orient Programming)	What is AOP What is Aspect J Demostrate the support of Spring to AOP	Get the Aspect J and usage to manage Aspects Usage of **@Aspect, @EnableAspectJAutoProxy,@Before and @After** annotations
Spring Web	What is Spring Web	There are 2 flavers of Spring Web as **Web MVC and Web Flux a**nd basic idea of thease 2s
	Initiate Spring MVC project	The process of create - Implement Spring web to existing Servlet project - Concept of MVC - Concept of Dispatcher servlet and front control design pattern - Two approches -- extend **WebApplicationInitializer class** -- extend **AbstractAnnotationConfigDispatcherServletInitializer class** (preferred approch of here)
	Structue of Spring MVC project	- **Context, Dispatcher servlet and point the Configuration classes** have been proceed after extend of AbstractAnnotationConfigDispatcherServletInitializer with few configs - Explain the 2 types of config classes as **WebAppConfig** and **WebRootConfig** -Reason for use **@EnableWebMvc** - Use of **@RestController** and annotations for pointed out HTTP methods (**@GetMapping, @PostMapping, @DeleteMapping,** and **@PutMapping or @PatchMapping**
	Mapping Spec with symbols	**? - match with a single character** **?? - match with character count (here it is 2)** *** - Match single path segment** **** - Match any path segment/s (same as wildcard mapping)**
	Manipulate the request with annotations	**@PathVariable - Mapping based on path variable** **@RequestParam - Get request parameters** **@RequestHeader - Get request headers** **JSON process with jackson-databind** **consume and produce MIME types**
	Classification of Web services	Characteristics of **SOAP** and **REST** web services and related matters
	Finish the service layer	Prac the CRUD operation with selected endpoint (tested Inmemory) Add **@Valid** with hibernate validation HTTPResponse handle with ResponseEntity
Project managemnt and build tools	Discuss about Ant, Maven and Gradle	Neediness of build tool and management tool with pointed out the dependency management

		Ant - Build tool only with legacy approcah		
		Maven - Build tool with depencency manage Architecture - Plggable arch: Explain about pluggins, goalls,phases and life cycle Directory structure and need to enhance best practices of programming Config details overview - pom.xml Explain about Maven is not a ... (like extended version of the Ant etc.)		
		Gradle - Build tool with depencency manage Same tsaks done as Maven Highlights of config data in build.gradle (Use Groovy prog.langugae, when apply configs it is inspire the programing ecosystem) Main project management tool in Android		
		MVN Repository - Repository for get both Maven and Gradle dependencies Versions of dependencies and thair supportice dependencies		
Spring Data - Simplify the DAO layer	Intro the Spring Data	Spring Data use to Data handle / DAO layer handle of a application		
		Several Flavers. Have sub projects to reach different levels. Ex: Spring Data JPA, Spring Data JDBC,Spring Data MongoDB.. Get more idea: https://spring.io/projects/spring-data		
	Intro to Spring Data JPA	Projects originate from the Spring Data umbrella		
		Comparative idea about JPA and JDBC(Can be refered the provided slide show)		
		Reducing underlaying configs and SQL commands for data manipulate		
	Integrate the Spring Data JPA to existing MVC project	Consider the main configurations : https://docs.spring.io/spring-data/jpa/reference/repositories/create-instances.html		
		Main 3 configs / beans : **DataSource, LocalContainerEntityManagerFactoryBean** and **PlatformTransactionManager**		
		DataSources - Use to manipulate the data with coperate the data points like DBs. Ex: **BasicDataSource, DriverManagerDataSource, ComboPooledDataSource, HikariDataSource** and **JNDI DataSource** (Refer the Slide show)		
		LocalContainerEntityManagerFactoryBean - Regards ORM and EntityManagerFactory, Need vender like Hibernare to perform (Consider the given overview about Hibernate and the respective slide show and your previous lecture notes of Hibernate)		
		PlatformTransactionManager - Related to Transaction handling (A transaction contains set of logical operations performs to manipulate the content of a DB)		
	Other core concepts	**JPARepository** - Abstract idea to effectualy perform the CRUD operations. There is alternative as CrudRepository for perform CRUD operations		
	Some main annotations	**@Repository** - Component with specialzice data operations **@EnableJpaRepositories** - Get true power of JPA repositories. Detect the JPA based data operations of the application **@EnableTransactionManagement** - Enable spring's transaction based annotations **@Transactional -** Declare the boundry to be executed as a transaction (use both class level and method level)		
Model mapper	Purpose of use	Use to mapping DTO and Entities and vise versa		

AAD - 68 / Phase 3 - API develop with Spring Boot
Start: Pending

Main scope	Sub scopes	What should able to manage and what have to know
Intro to Spring Boot	What is Spring Boot	Spring Boot is a another Spring project
	Why Spring Boot	For rapid application development with minimum configuration
	File struture and initialize a project	File struture is maintain the relevent project management tool as either Maven or Gradle. Config file is special and sugeastions avilable. Both .properties and .yml types are support. - YAML file type inspire to the configs with friendly with human readability with segmants. spring Project init with support of https://start.spring.io
	Program model	Few configs and powerd the project with main 3 annotations @SpringBootApplication - Enable 3 other annotations **@EnableAutoConfiguration - Enable Spring Boot's auto-configuration mechanism** **@ComponentScan - Enable component scan facility** **@Configuration - Import additional config classes and allow to register extra beans in the context**
Unit test with Junit and Mokito	Test with existing Database	Unit testing, which tests a piece of code, is important before moving to the next level and before developing the entire application
	Test with mock object	Not hit to the real DB and use mock object for the test purpose. Mockito Java framework applied @Mock - Create a mock object and belongs to Mockito FW @Test - Use in JUnit and mark as a test method
Secure an API- Key concepts	Authentication	Verifying the identity of the user (relevant HTTP status code if there is an issue: 401 - Unauthorized).
	Authorization	Determine what kind of actions to be allowed - (relevant status code if there is an issue: 403 - Forbidden).
	Refer the Slide Show - Spring Security	
Spring Security	What is Spring security	Spring FW project to handle security in a application Updated and dynamic implementation. Should read the documentation before implementation.
	Overview of the architecture	Refer the doc: https://docs.spring.io/spring-security/reference/servlet/architecture.html Special terms: ***DelegatingFilterProxy, FilterChainProxy, SecurityFilterChain,Security Filters***
	Security mechanisms	Here considered Basic Auth and OAuth 2.0
	Basic Auth	Implement Basic Auth with inmemory data
	Auth with Token based	Practice with JWT (JSON Web Token) - See all the details with https://jwt.io/
		Get an idea about story of JWT with read https://jwt.io/introduction
	Implement JWT mechaniam to the Spring Security	Refer the slide show and the project that developed in the class. The slide show included the description about the special interfaces, classes and methods that used when implmeint the mechanism
		Authorization implemant with consider the role and use the method security with **@PreAuthorize** R&D - what is @PostAuthorize and its usage
	Refer the Slide Show - Slides that regards the terms that belongs to the Spring Security	