

Last Edited : Aug 28, 2024

## Spring boot: Project Setup and Layered Architecture (Concept & Coding)

### "Concept & Coding" YT Video Notes

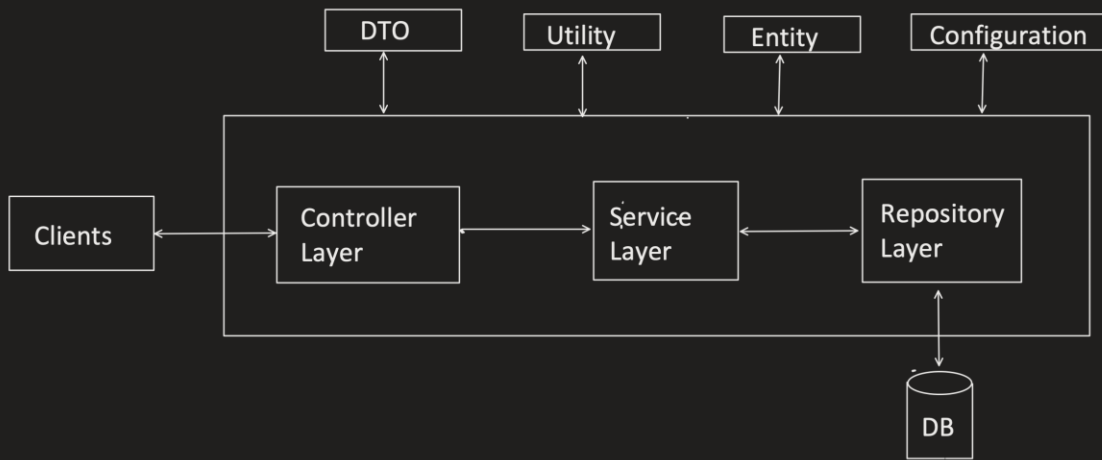
#### Setting up the Project

#### 1. Go to Spring Initializr i.e. "start.spring.io"

The screenshot shows the Spring Initializr web form for creating a new project. The form is divided into several sections: Project, Language, Spring Boot, Project Metadata, and Dependencies.

- Project:** Includes radio buttons for `Gradle - Groovy`, `Gradle - Kotlin`, and `Maven` (which is selected).
- Language:** Includes radio buttons for `Java` (selected), `Kotlin`, and `Groovy`.
- Spring Boot:** Includes radio buttons for versions `3.3.0 (SNAPSHOT)`, `3.3.0 (M1)`, `3.2.4 (SNAPSHOT)`, and `3.2.3` (which is selected).
- Project Metadata:** Includes text input fields for `Group` (filled with `com.conceptandcoding`), `Artifact` (filled with `learningspringboot`), `Name` (filled with `springboot application`), `Description` (filled with `project for learning spring boot`), and `Package name` (filled with `com.conceptandcoding.learningspringboot`).
- Packaging:** Includes radio buttons for `Jar` (selected) and `War`.
- Java:** Includes radio buttons for versions `21` and `17` (which is selected).
- Dependencies:** Includes a section for `Spring Web` (marked with a `WEB` tag) with a description: "Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container." There is also an `ADD DEPENDENCIES...` button with a plus icon.

## LAYERED ARCHITECTURE



```
@RestController
@RequestMapping("/payments")
public class PaymentController {

    @Autowired
    PaymentService paymentService;

    @GetMapping("/{id}")
    public ResponseEntity<PaymentResponse> getPaymentById(@PathVariable Long id) {
        //map incoming data to internal request DTO
        PaymentRequest internalRequestDb = new PaymentRequest();
        internalRequestDb.setPaymentId(id);

        //pass this internalRequestDb to further layer for processing
        PaymentResponse payment = paymentService.getPaymentDetailsById(internalRequestDb);

        //return the Response DTO
        return ResponseEntity.ok(payment);
    }
}
```

```
@Service
public class PaymentService {

    @Autowired
    PaymentRepository paymentRepository;

    public PaymentResponse getPaymentDetailsById(PaymentRequest internalRequestDb) {
        PaymentEntity paymentModel = paymentRepository.getPaymentById(internalRequestDb);

        //map it to response obj
        PaymentResponse paymentResponse = mapModelToResponseDto(paymentModel);
        return paymentResponse;
    }

    private PaymentResponse mapModelToResponseDto(PaymentEntity paymentEntity){
        PaymentResponse response = new PaymentResponse();
        response.setPaymentId(paymentEntity.getId());
        response.setAmount(paymentEntity.getPaymentAmount());
        response.setCurrency(paymentEntity.getPaymentCurrency());
        return response;
    }
}
```

```
@Repository
public class PaymentRepository {

    public PaymentEntity getPaymentById(PaymentRequest request){
        PaymentEntity paymentModel = executeQuery(request);
        return paymentModel;
    }

    private PaymentEntity executeQuery(PaymentRequest request){
        //connect with DB and fetch the data
        PaymentEntity payment = new PaymentEntity();
        payment.setId(request.getPaymentId());
        payment.setPaymentCurrency("INR");
        payment.setPaymentAmount(100.00);
        return payment;
    }
}
```