

Full Stack



Caltech

**Center for Technology &
Management Education**

Post Graduate Program in Full Stack Web Development

Full Stack



Caltech

**Center for Technology &
Management Education**

**Implement Frameworks the DevOps
Way**



Create a Java Application on a Spring Boot Framework

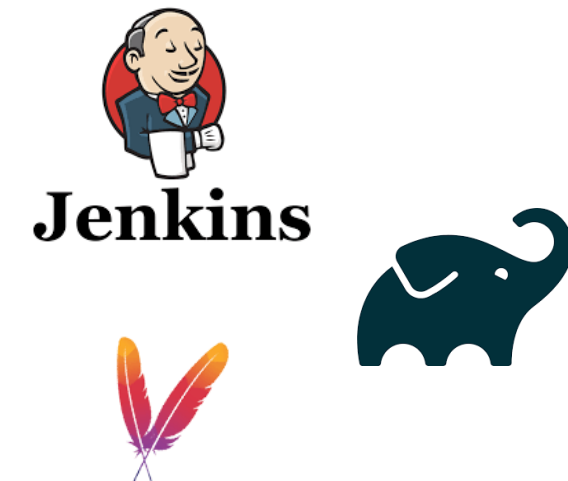
You Already Know

Course(s):

1. Master Hibernate and JPA with Spring Boot in 100 steps



2. DevOps: CI/CD with Jenkins pipelines, Maven, Gradle



- Explain Spring Boot
 - Spring Boot definition
 - Spring Boot setup
- Demonstrate Hibernate and JPA integration
 - Hibernate and JPA integration
 - Queries with entity using JPQL
 - Spring data and Spring REST
 - Caching with Hibernate and JPA



- Explain Jenkins
 - Explain DevOps
 - Explain role of Jenkins for a full stack developer
 - Explain CI/CD pipeline
- Demonstrate Jenkins and AWS integration
 - Integrate Jenkins with AWS
 - Work with CI/CD pipeline for automation and development to staging environment link



A Day in the Life of a Full Stack Developer

In the last sprint, Joe had done a remarkable job. He had created an MVC web application that will retrieve users based on their user ID and update them in the database.

He has now been given a new project to work on. He has to create a Spring Boot project that will capture user feedback using a REST endpoint. The feedback data will then be added to a database table.

In this lesson, we will learn how to solve this real-world scenario to help Joe complete his task effectively and quickly.

Learning Objectives

By the end of this lesson, you will be able to:

- 🕒 List the differences between Spring and Spring Boot
- 🕒 Define exception handling in Spring Boot
- 🕒 Implement the steps for building RESTful web services
- 🕒 Explain edge server and routing and list the steps to implement routing
- 🕒 State the advantages of using Swagger
- 🕒 Discuss the use of Apache Kafka
- 🕒 Identify the modes of database and their main features
- 🕒 Understand how HTTPS is implemented in Spring Boot



Spring Boot Fundamentals

Introduction to Spring Boot

- Is an open-source Java-based framework
- Can be used to create Microservices
- Makes it easy to create stand-alone, production-grade Spring-based applications that you can **just run**
- Provides the **RAD** (Rapid Application Development) feature to the Spring application



Why Spring Boot?

Provides a flexible way to configure Java Beans, XML configuration, and database transactions

Uses dependency injection approach

Includes embedded servlet container

Reduces the cost and development time of the application

Does not require manual configuration; everything is auto-configured

Spring Boot: Advantages

Advantages

1. Supports fast and easy development of stand-alone Spring applications that can be started using jar files
2. Helps to directly embed Tomcat, Jetty, or Undertow into an application
3. Does not need to deploy war files
4. Does not require XML configuration
5. Minimizes writing source code
6. Offers a number of plug-ins
7. Provides opinionated 'starter' POMs to simplify Maven configuration

Spring Boot: Disadvantages

Disadvantages

1. Has lack of control
2. Is not suitable for large-scale projects
3. Unused dependencies increase the size of the application

Goals

Spring Boot goals are to:

- Reduce development, unit test, and development time
- Provide a radically faster and widely accessible getting started experience
- Be opinionated out of the box, but get out of the way quickly as requirements start to diverge from the defaults
- Avoid code generation and requirement for XML configuration

Features

The various features of Spring Boot are:

Spring Application

Properties Files

Web Development

YAML Support

Application Events and Listeners

Type-Safe Configuration

Admin Features

Logging

Externalized Configuration

Security

Prerequisites

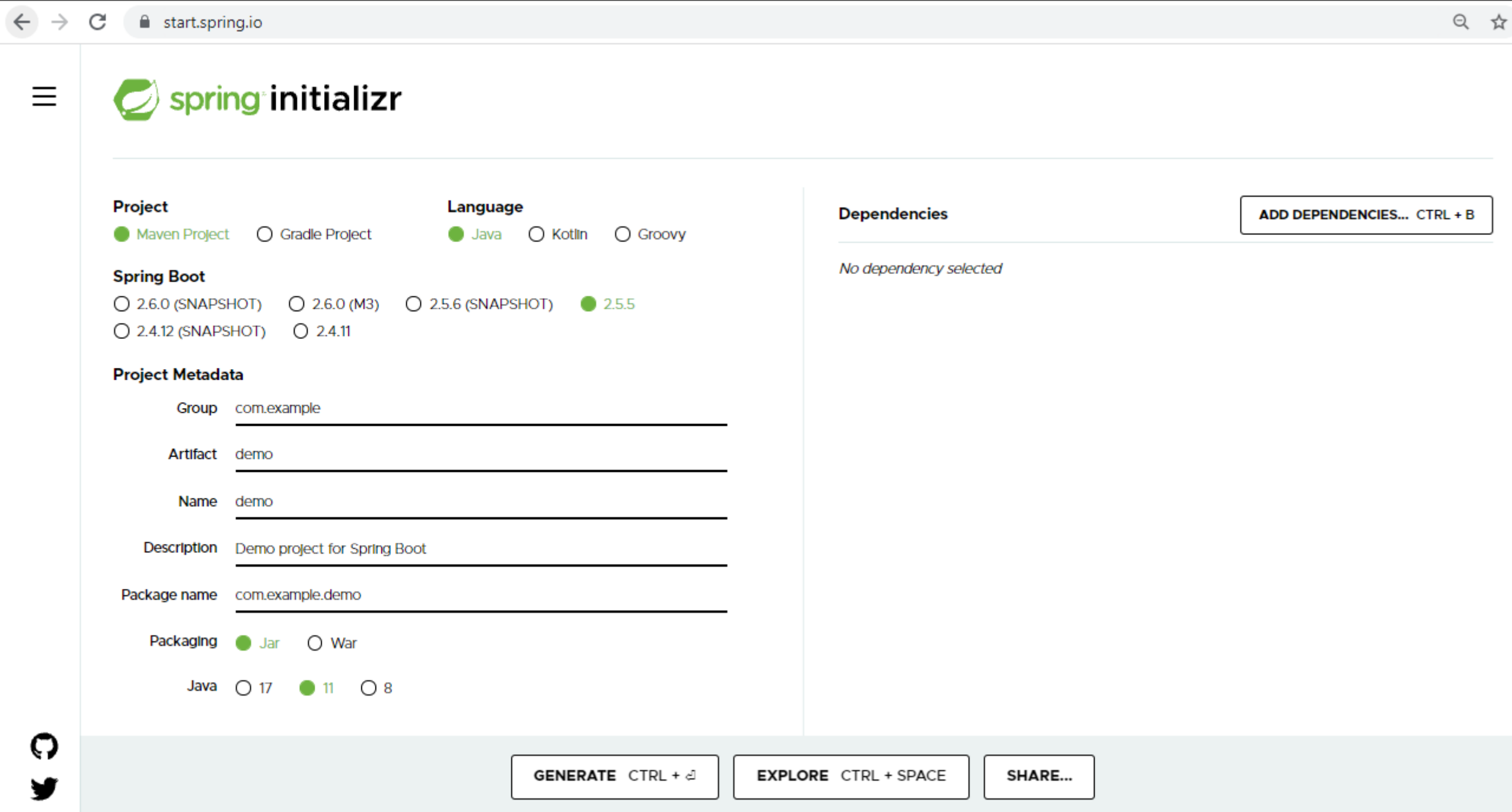
To create Spring Boot applications, following are the prerequisites:

- Java
- Spring
- Maven
- Gradle
- A suitable and recommended IDE
- A basic idea of REStful Web Service



Spring Boot Download

Go to <https://start.spring.io/> and download the required version of Spring Boot



The screenshot shows the Spring Initializr web application interface. The browser address bar displays `start.spring.io`. The page features the Spring logo and the text "spring initializr".

Project

- ☒ Maven Project ☐ Gradle Project

Language

- ☒ Java ☐ Kotlin ☐ Groovy

Spring Boot

- ☐ 2.6.0 (SNAPSHOT) ☐ 2.6.0 (M3) ☐ 2.5.6 (SNAPSHOT) ☒ 2.5.5
- ☐ 2.4.12 (SNAPSHOT) ☐ 2.4.11

Project Metadata

Group

Artifact

Name

Description

Package name

Packaging ☒ Jar ☐ War

Java ☐ 17 ☒ 11 ☐ 8

Dependencies

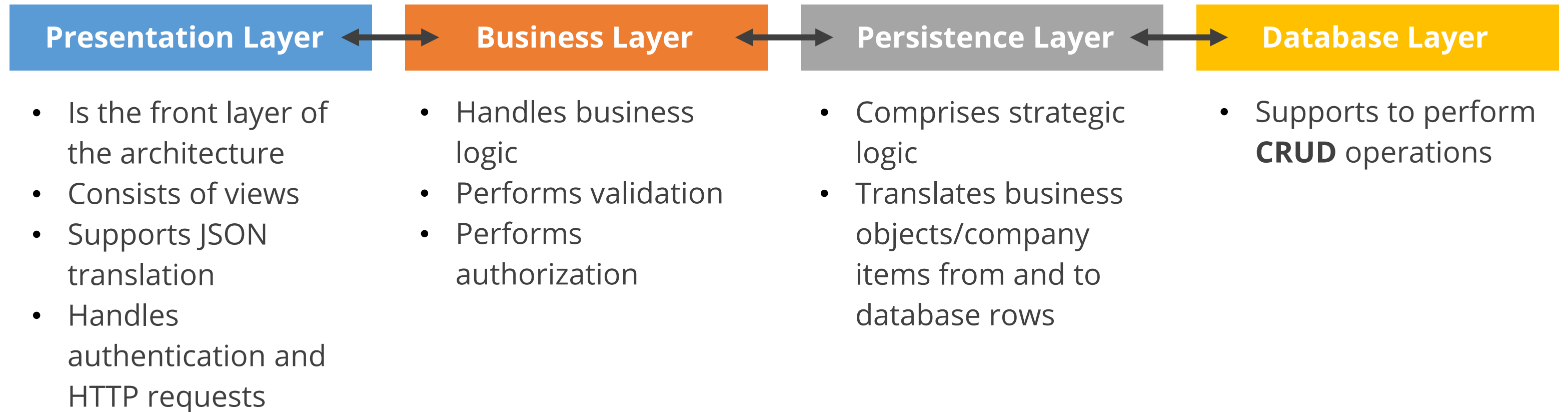
[ADD DEPENDENCIES... CTRL + B](#)

No dependency selected

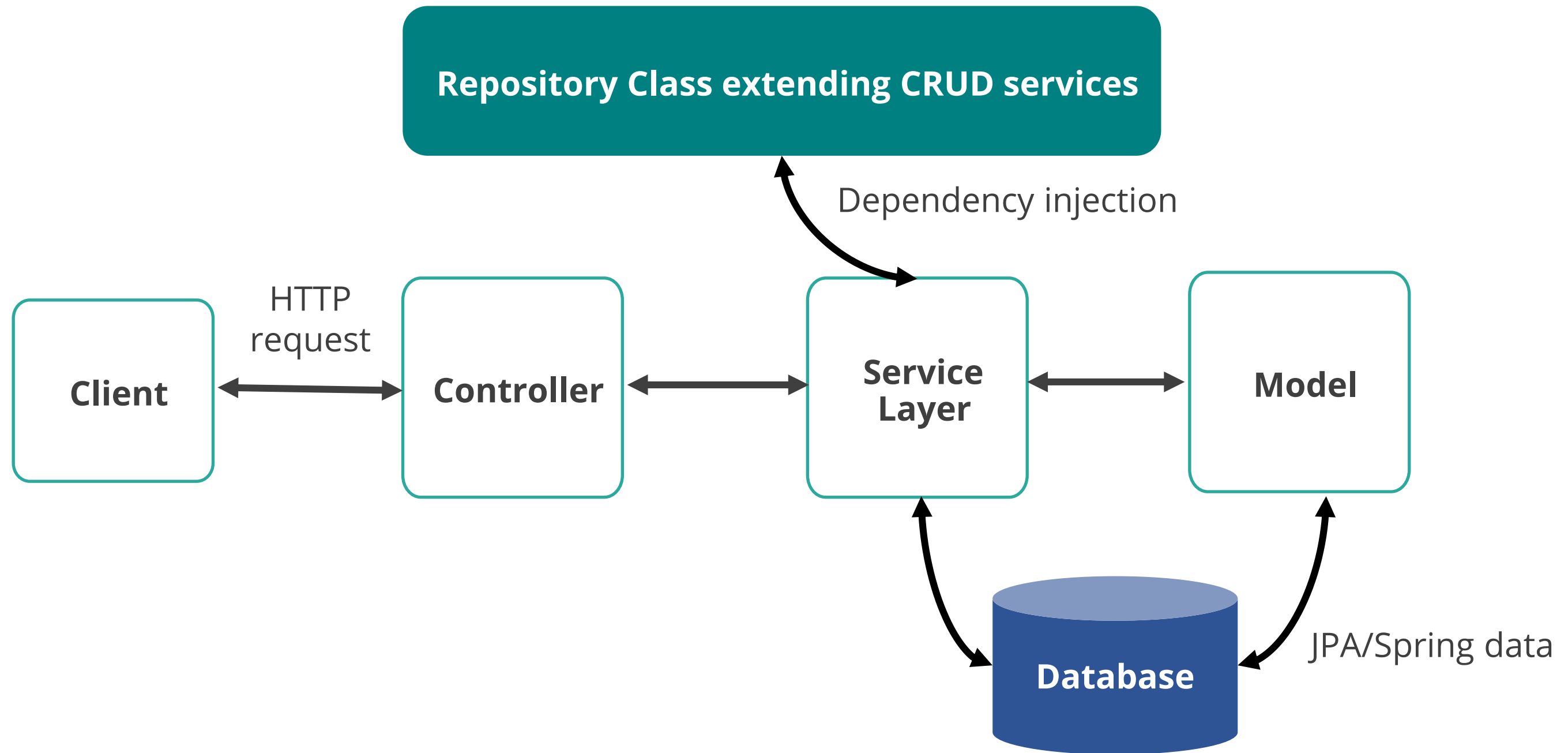
Buttons:

- [GENERATE CTRL + G](#)
- [EXPLORE CTRL + SPACE](#)
- [SHARE...](#)

Spring Boot Architecture



Spring Boot Flow Architecture



Spring Boot: Components

The four main components of Spring Boot are:

Spring Boot Starter

- Combines a group of common or related dependencies into a single dependency

Spring Boot AutoConfigurator

- Reduces the number of annotations
- Simplifies all XML-based configurations

Spring Boot: Components

The four main components of Spring Boot are:

Spring Boot CLI

- Runs and tests Spring Boot applications from the command prompt
- Uses Spring Boot Starter and Spring Boot AutoConfigurator internally, while executing the Spring Boot application

Spring Boot Actuator

- Adds several production grade services to your application with little effort on your part
- Provides EndPoint and Metrics

Spring Boot: CLI

The Spring Boot CLI is the command line interface used for a quick start with Spring.



Provides an interface to run Spring Boot Application along with testing from the command prompt



Supports Groovy Scripts without external Groovy installation



Uses Spring Boot Starter and AutoConfigure components to resolve the dependencies and executes the application

Spring Boot: CLI

The Spring Boot CLI is the command line interface used for a quick start with Spring.



Contains Groovy compiler and Grape Dependency Manager



Adds Spring Boot defaults and resolves all dependencies automatically

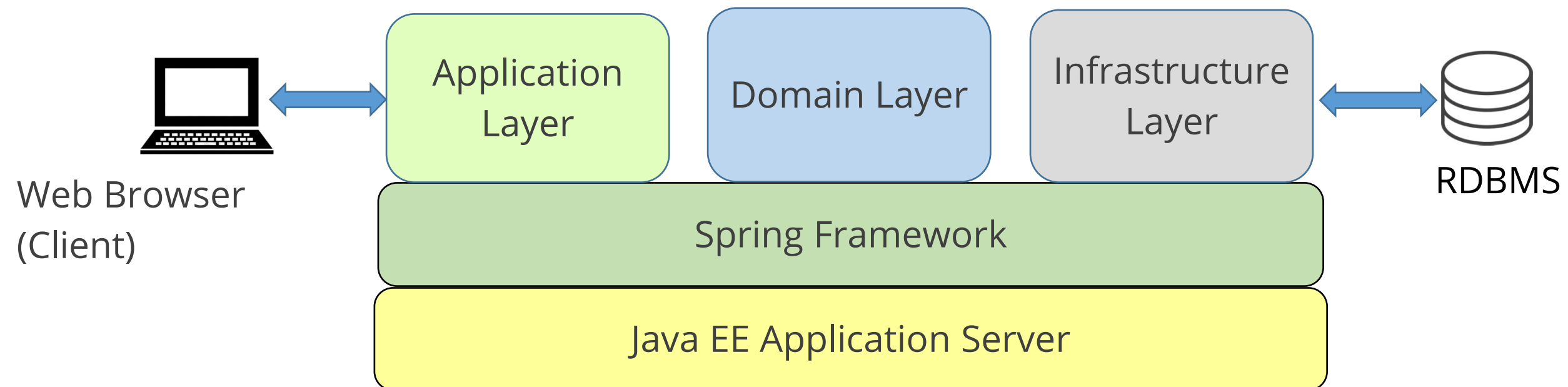


Helps provide new bootstrap project along with writing custom command for it

Difference between Spring Boot and Spring Framework

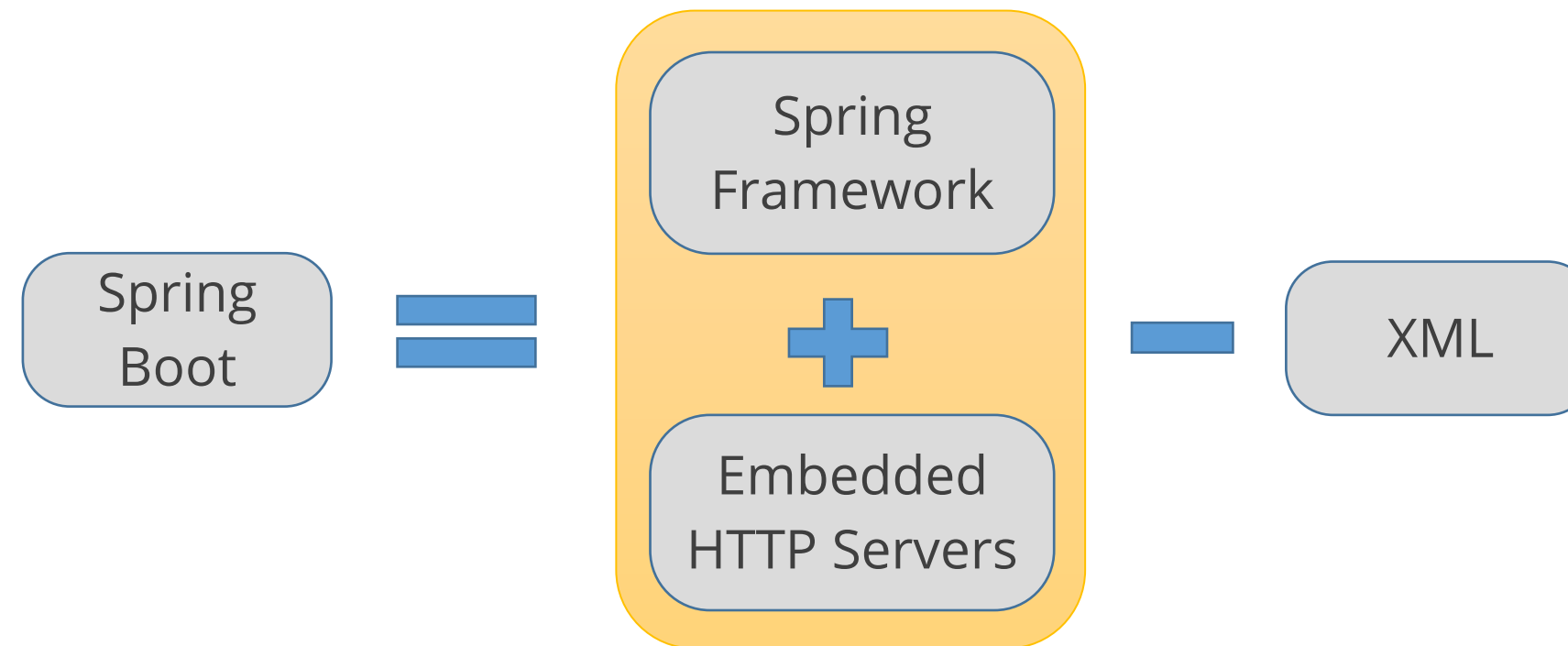
Spring Framework

Spring is a very popular open source development framework for Java. It can be used to create easy-to-use and easy-to-test high performing and reusable code. Web applications can be extended on top of the Java Enterprise Edition (EE) platform.



Spring Boot

Spring Boot is an open source Java-based project built on Spring framework. It allows creation of standalone applications in a simple and easy manner. The entire Spring configuration is not necessary to develop the services.



Difference between Spring and Spring Boot

Spring	Spring Boot
Different features for designing applications are not automatically handled	Default features are automatically handled
XML definitions are required to manage Spring applications	Complex XML configurations can be avoided
Application type and characteristics need to be properly defined to develop applications	Less number of configurations are required to develop applications

Spring Boot Exception Handling

Spring Boot Exception Handling

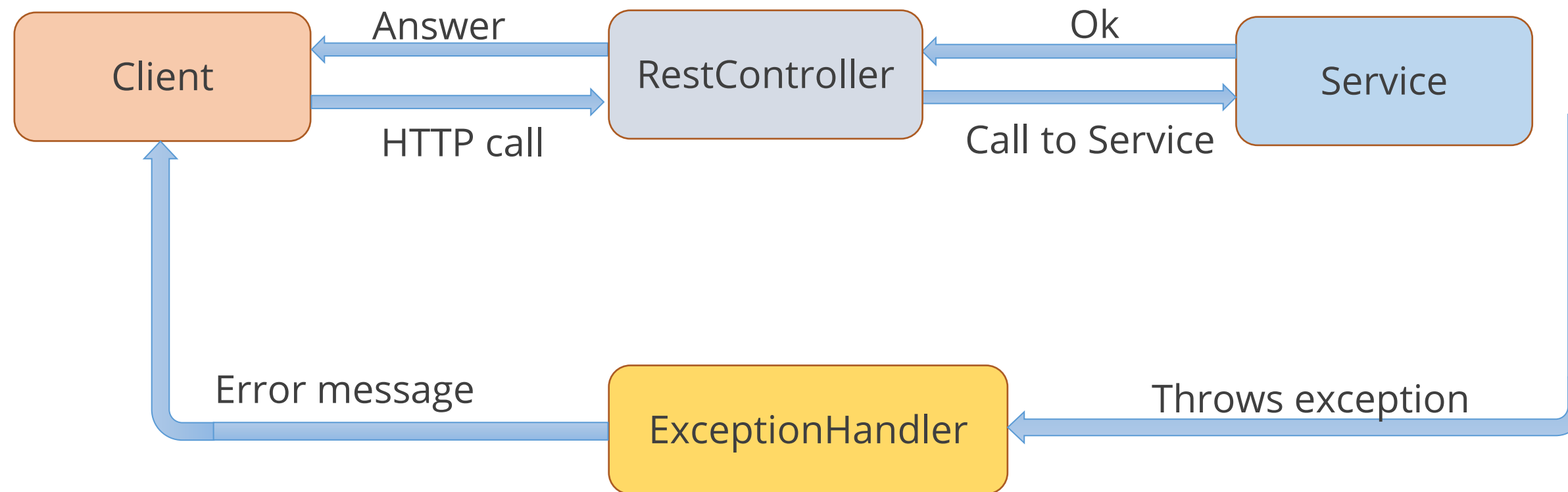
Exceptions and errors in APIs are required to be handled properly by the enterprise applications.

Exception Handling Classes:

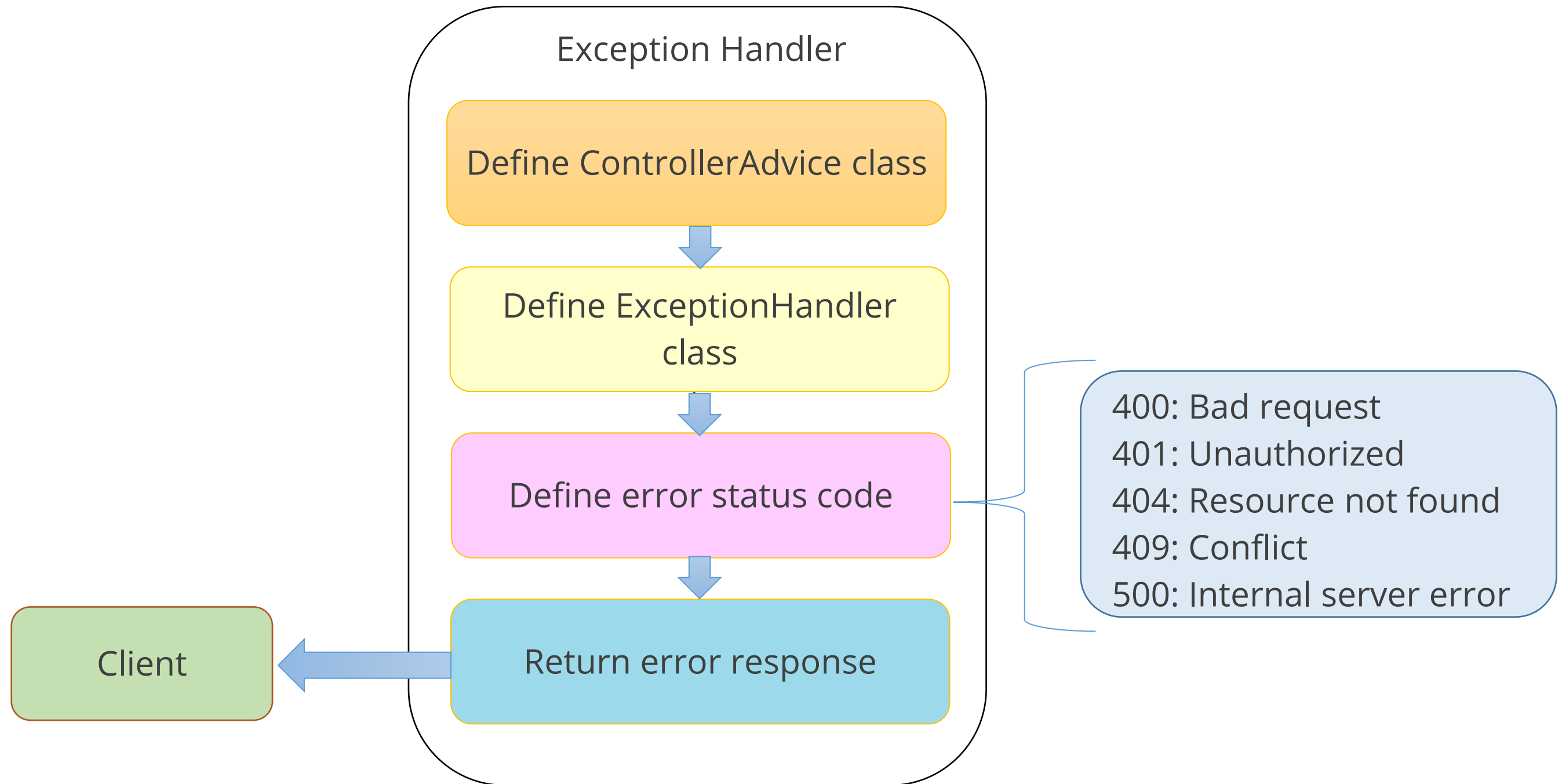
ControllerAdvice	Handles application-level exceptions
ExceptionHandler	Handles exceptions from methods in the same controller

Spring Boot Exception Handling

RestController is a base for classes handling REST operations. Exceptions are treated by the ExceptionHandler when thrown by the services.



Spring Boot Exception Handling: Implementation



Assisted Practice

Assisted Practice: Spring Boot Exception Handling

Duration: 45 min.

Problem Statement:

Create an example to handle exception for product not found.

Assisted Practice: Guidelines

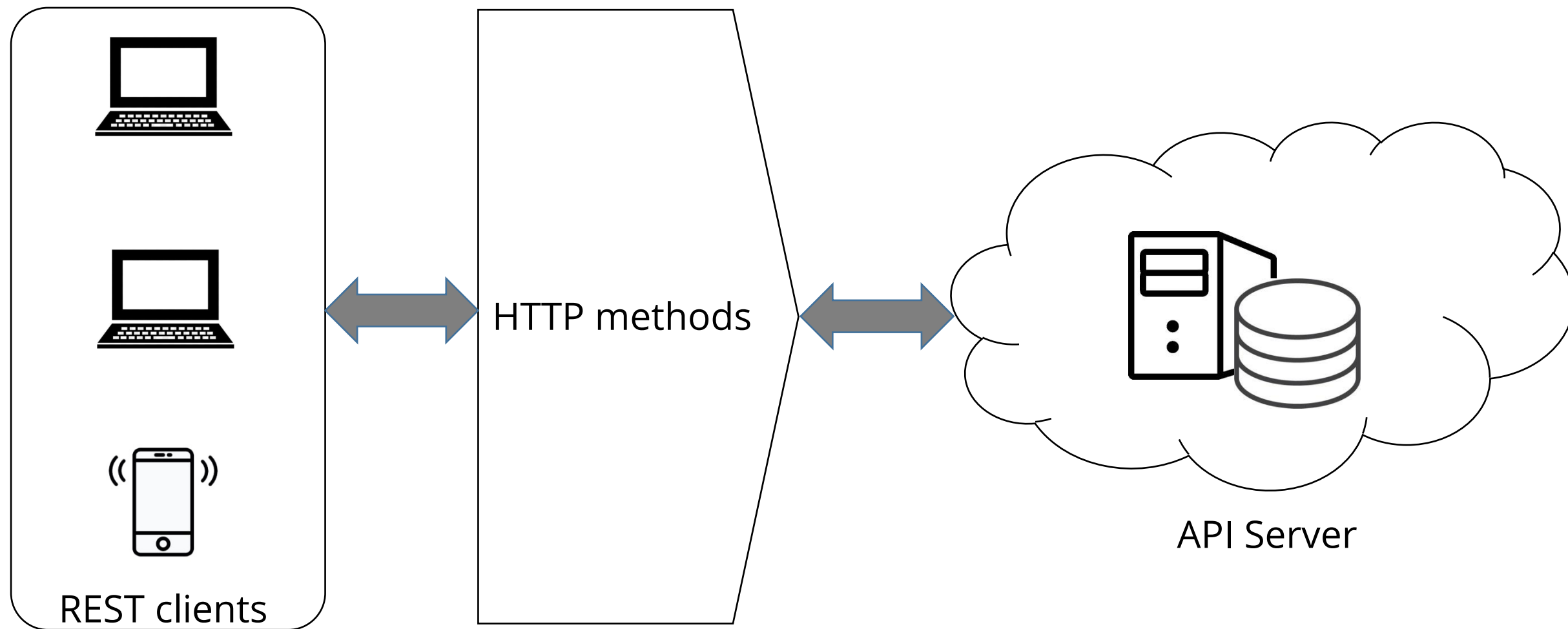
Steps to demonstrate spring boot exception handling:

1. Install the STS Plugin in Eclipse
2. Create a Spring Boot starter project that is web enabled
3. Create a EProduct entity class
4. Create a ProductNotFoundException class
5. Create a EProductExceptionController class
6. Create a MainController to throw ProductNotFoundException
7. Build the project
8. Publish and start the project
9. Run the project
10. Push the code to GitHub repositories

Building RESTful Web Services

Building RESTful Web Services

REST (Representational State Transfer) uses HTTP protocol for developing web services. Service built on REST architecture is a RESTful service and is scalable, easy to use, and lightweight.



Building RESTful Web Services

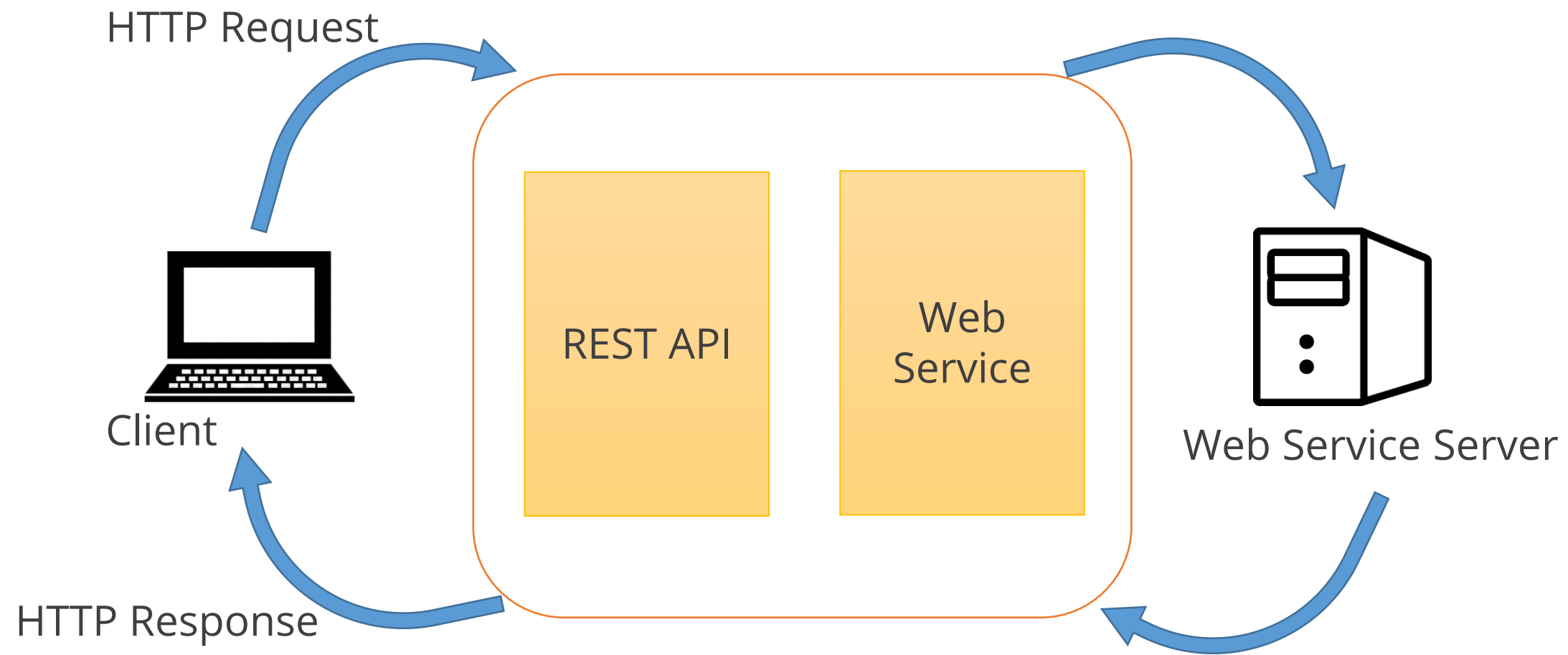
Steps to implement:

- 1 Create a blank web application in a new project
- 2 Choose a file of type WCF service (Ajax-enabled)
- 3 Add a service to project
- 4 Make changes in the Web.config configuration file
- 5 Add implementation code on the service file
- 6 Define a code for GET and POST methods
- 7 Set the project option to **Set as StartUp Project**

Consuming RESTful Web Services

Consuming RESTful Web Services

A client utilizing a web service also means consuming the web service.



Assisted Practice

Assisted Practice: Consuming RESTful Web Services

Duration: 45 min.

Problem Statement:

Create a project to consume a RESTful web service.

Assisted Practice: Guidelines

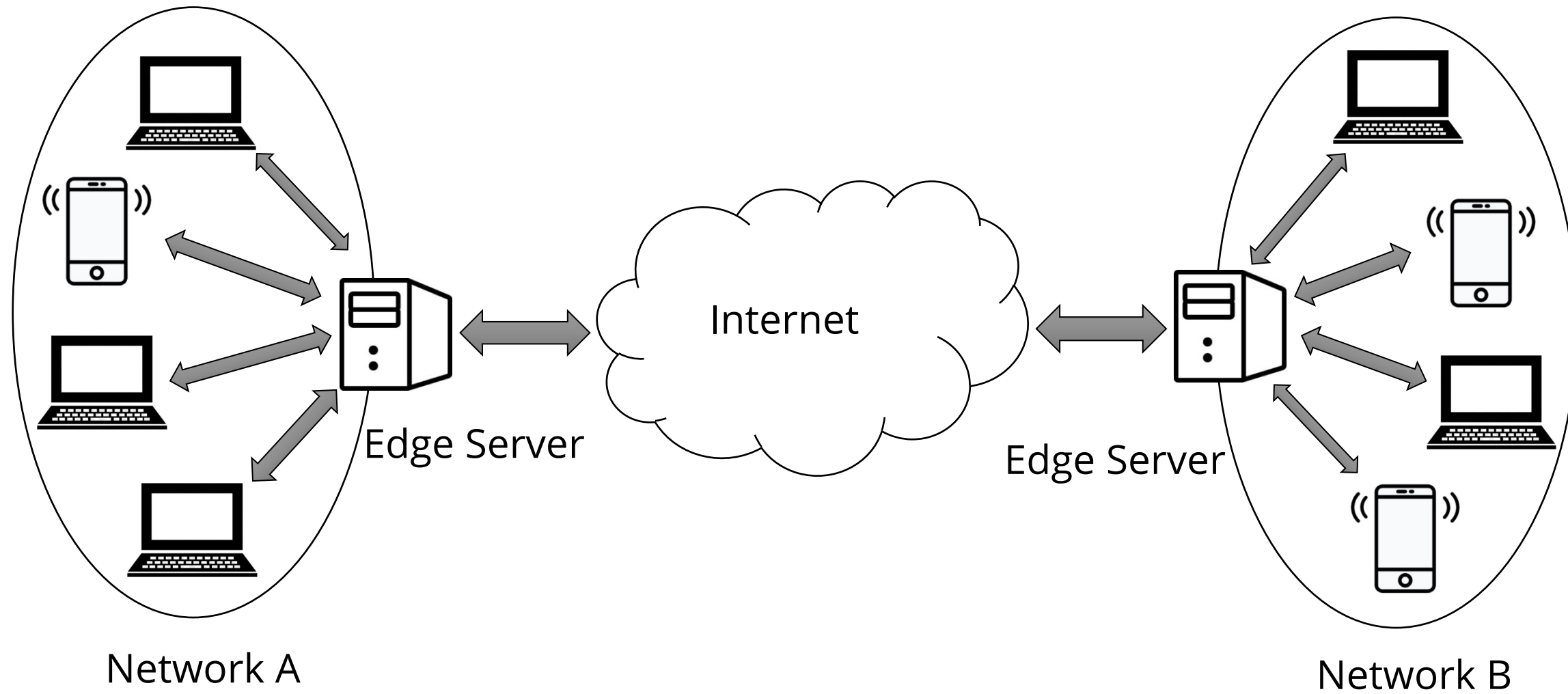
Steps to demonstrate consuming RESTful web services:

1. Install the STS Plugin in Eclipse
2. Create a Spring Boot starter project that is web enabled
3. Create a class Quote to work with the public REST service
4. Create a class Value to act as a wrapper for the REST data
5. Create a MainController to consume the REST service
6. Build the project
7. Publish and start the project
8. Run the project
9. Push the code to GitHub repositories

Edge Server and Routing

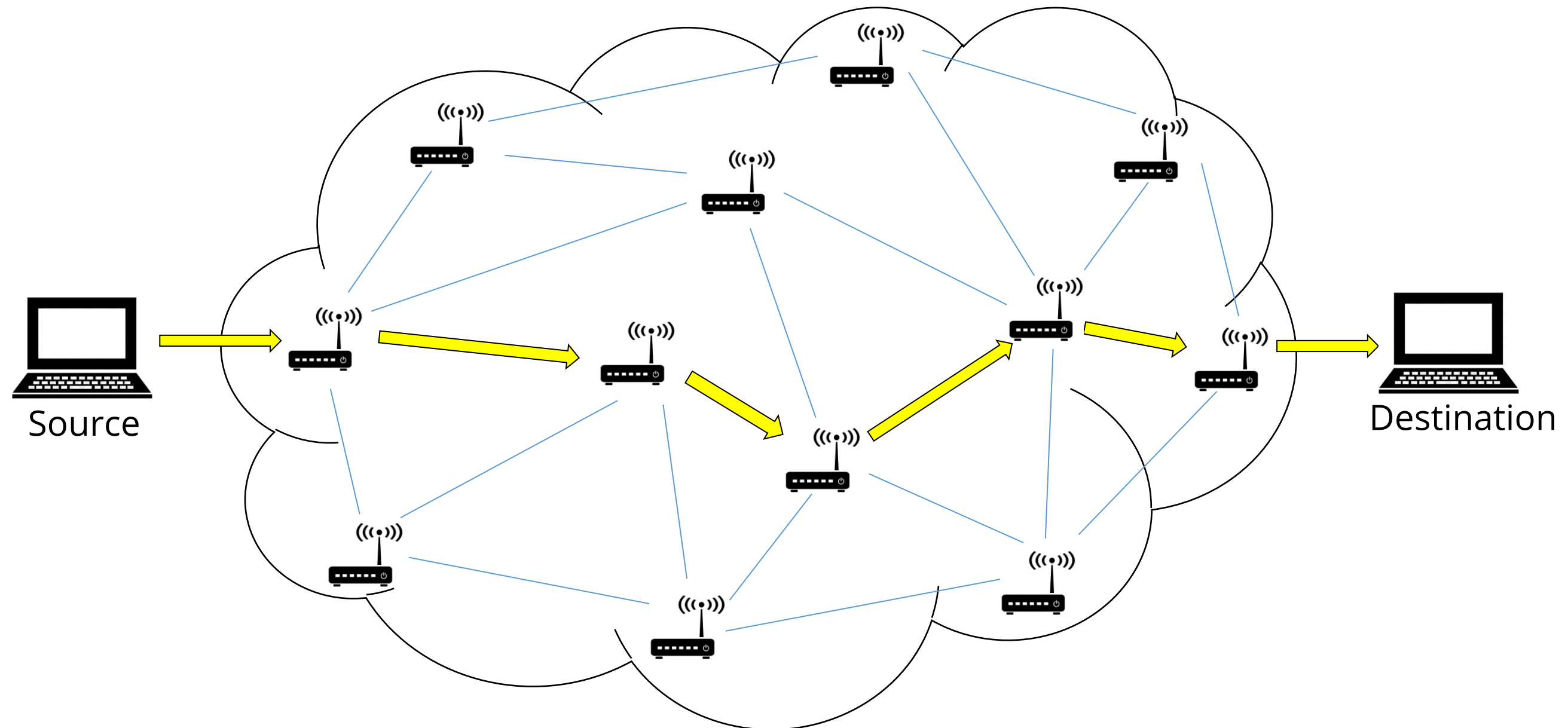
Edge Servers

- Edge server is a server placed at the edge or end of a network. This is used to connect to other networks.
- The network can be a private network or the Internet.



Routing

Routing means moving packets (fundamental units of information) across a network between different hosts. Devices called Routers use standard protocols to choose an optimal path between the networks to transfer the data packets.

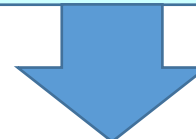


Routing: Steps to Implement

1. Remote network information is determined by the router from its neighbouring routers



2. Routing table is created by the table to select the best path for the data packets



3. Based on the connection between the networks, the router decides the type of routing

Types of Routing

Static Routing

The routes and network locations are added manually.
Since the path is between specific nodes, there is data security.

Default Routing

All packets are sent to a single router irrespective of the network.

Dynamic Routing

It uses protocols to determine the best path to transmit data over the network. The routers add network locations automatically.

Assisted Practice

Assisted Practice: Edge Server and Routing

Duration: 45 min.

Problem Statement:

Create a project to work as a dynamic router for Zuul.

Assisted Practice: Guidelines

Steps to demonstrate routing:

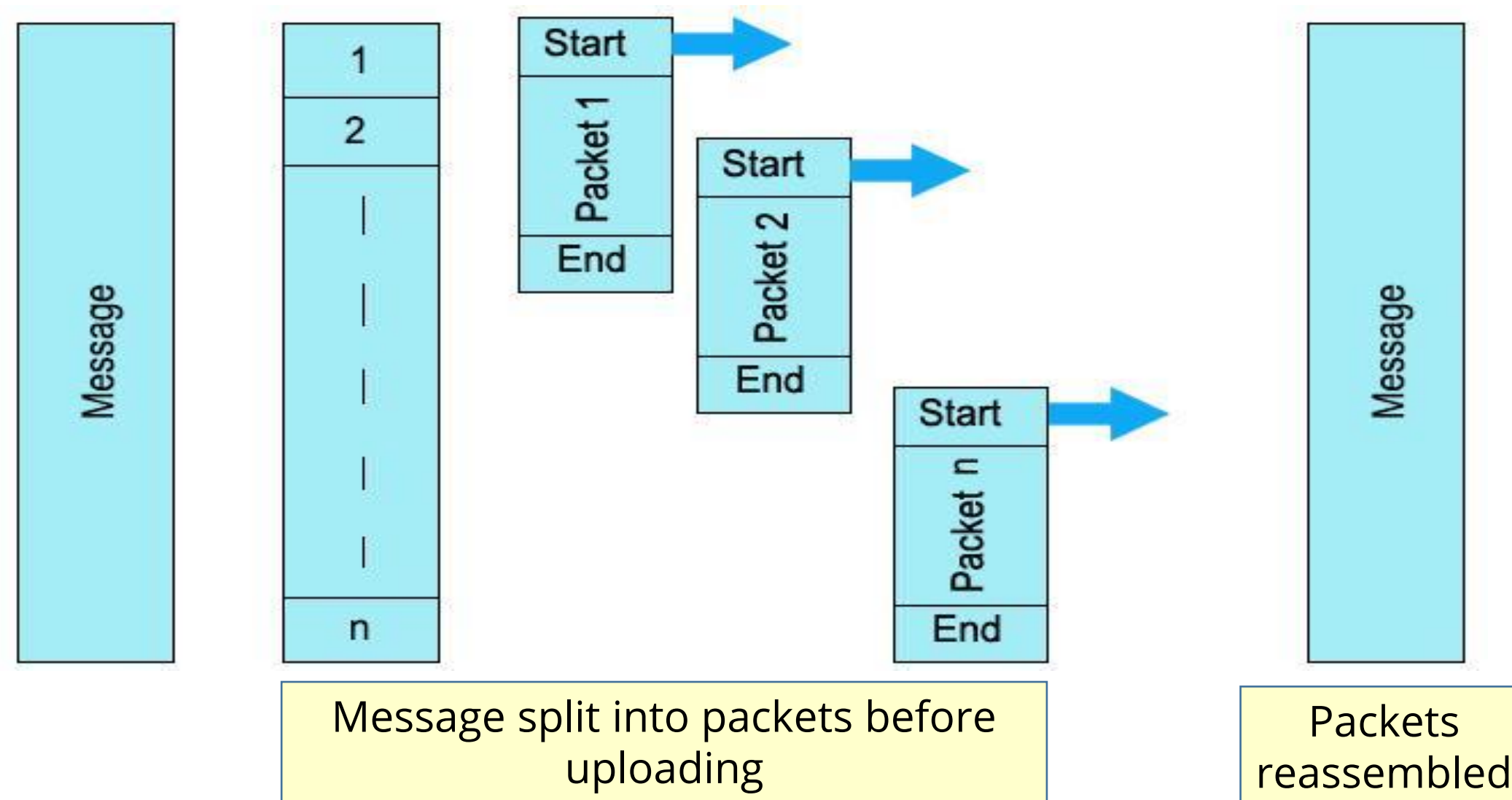
1. Install the STS Plugin in Eclipse
2. Create a Spring Boot starter project that is web enabled and configured to work with Zuul services
3. Configure a SpringEdgeApplication class to act as a dynamic router for Zuul
4. Configure application.properties for Zuul
5. Build the project
6. Publish and start the project
7. Run the project
8. Push the code to GitHub repositories

File Handling

File Handling

File Uploading:

To upload a file in a Spring Boot framework, MultipartFile is used as a request parameter. MultipartFile requests split a large file into smaller parts. Boundary markers are used to indicate the beginning and ending of a block.



File Handling

File Downloading

Methods to download files to the client:

InputStreamResource

Method to return ResponseEntity object is to be created.
This wraps InputStreamResource object for downloading the data to the client.

HttpServletResponse

File can be downloaded into HttpServletResponse using Reader or Stream.Write.
Reader is used for text data while Stream.Write is for binary data.

StreamingResponseBody

It is useful when downloading a large file.

Assisted Practice

Assisted Practice: File Handling

Duration: 45 min.

Problem Statement:

Create a project to upload and download a file in Spring Boot.

Assisted Practice: Guidelines

Steps to demonstrate file handling:

1. Install the STS Plugin in Eclipse
2. Create a Spring Boot starter project that is web enabled
3. Create dump.txt as a file for downloading
4. Create an HTML file that will show a form of file uploading
5. Create a MainController for handling file upload and download
6. Build the project
7. Publish and start the project
8. Run the project
9. Push the code to GitHub repositories

Swagger 2

Swagger 2

Swagger is an open source framework used to describe the structure of APIs using a common language for all machines to read.

Importance of Swagger 2:

Makes it easy to design and model APIs based on industry standards

Used to create stable and reusable code for the API in any language

Used to create interactive API documentation

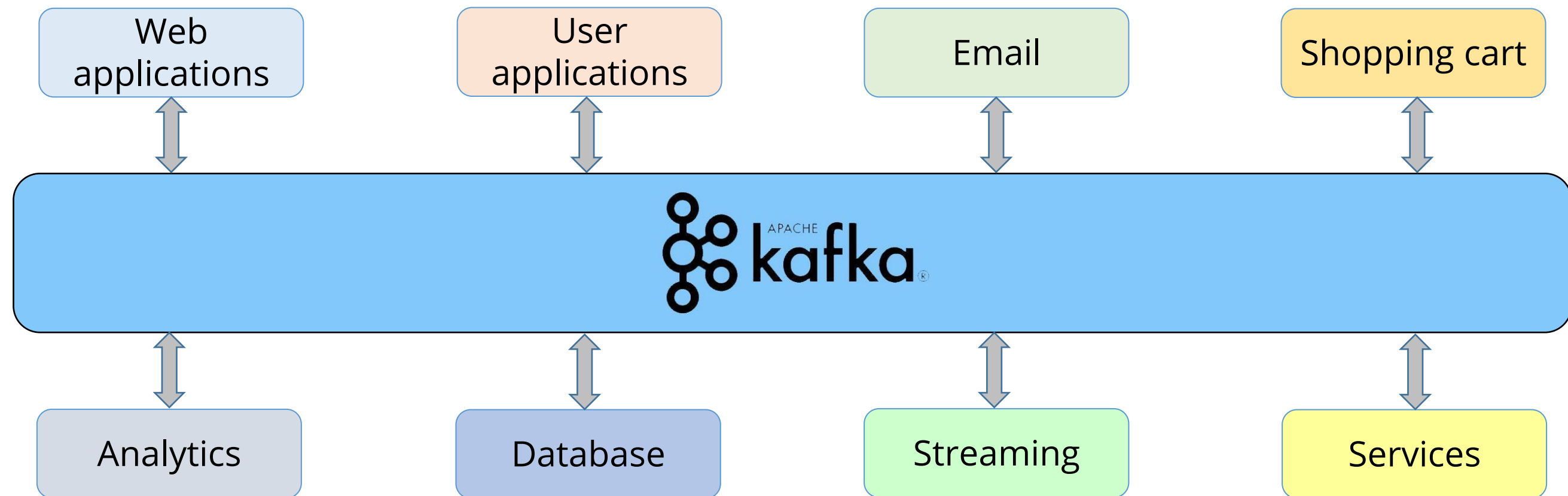
Used to perform simple functional tests on the API without overhead

Can set and enforce API style guidelines across the API architecture

Apache Kafka

Apache Kafka

Apache Kafka is a scalable and open-source data streaming and messaging system. It manages and maintains the real-time stream of data from different websites and applications.



Apache Kafka

Apache Kafka is a distributed publish-subscribe messaging system.

Some of its popular features:

- It is scalable in all directions
- It easily handles large volumes of data
- It easily transforms data
- It is fault tolerant toward data failure
- It is highly reliable, durable, and stable

Assisted Practice

Assisted Practice: Sending and Receiving Messages with Apache Kafka

Duration: 45 min.

Problem Statement:

Create a project to send and listen to Apache Kafka messages.

Assisted Practice: Guidelines

Steps to work with Apache kafka in spring boot:

1. Install the STS Plugin in Eclipse
2. Create a Spring Boot starter project that is web enabled and has Apache Kafka dependencies
3. Create a KafkaProducerConfig class
4. Create a KafkaConsumerConfig class
5. Create a MainController to send a Kafka message
6. Configure a SpringRestApplication to listen to Kafka messages
7. Build the project
8. Publish and start the project
9. Run the project
10. Push the code to GitHub repositories

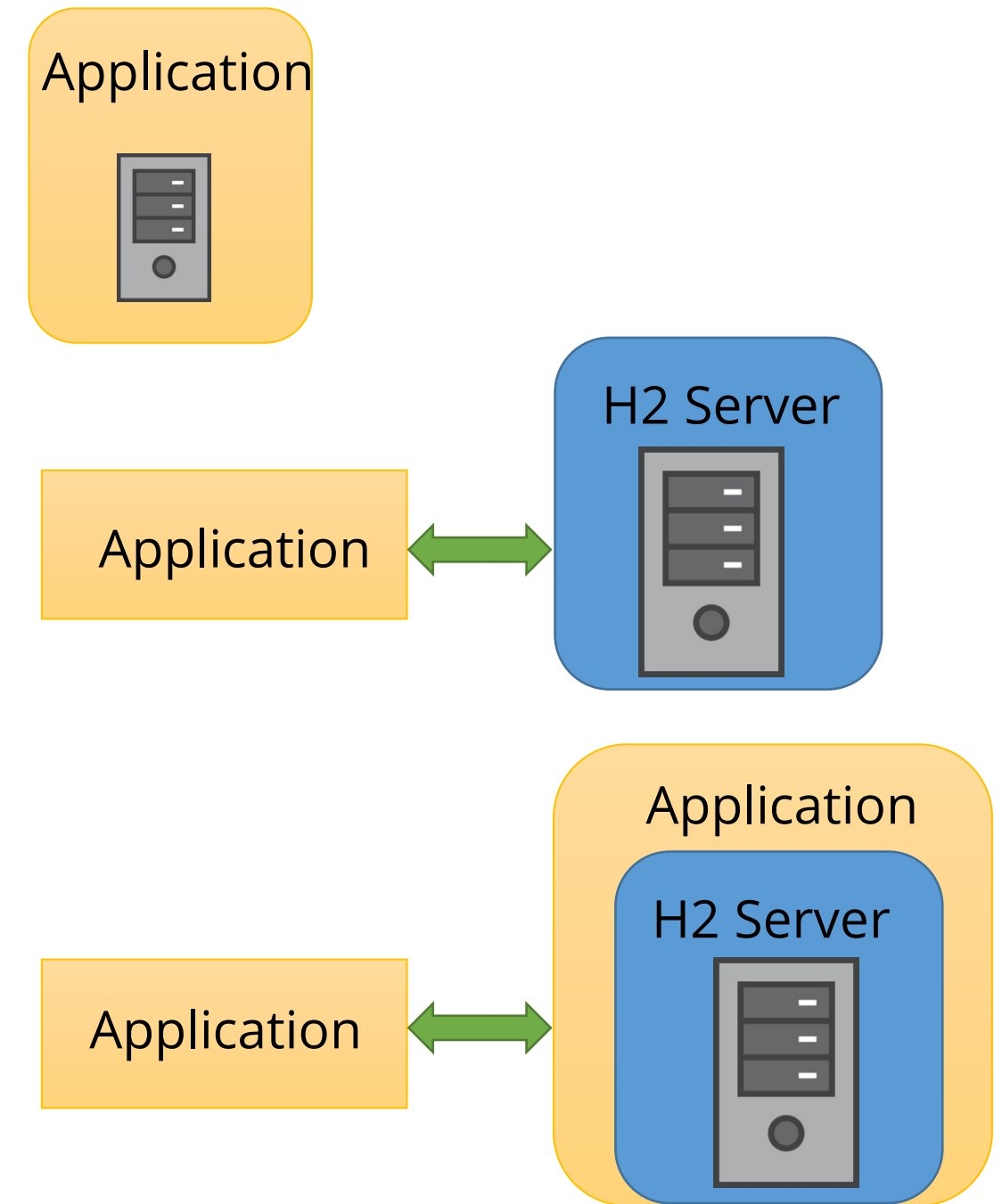
Database Handling

Database Handling

The database management system written in Java is H2. It is packaged and distributed with Java applications.

H2 can be run in three modes:

1. **Embedded mode:** Database is embedded inside the application
2. **Server mode:** Database and application are in separate servers
3. **Mixed mode:** Combination of embedded and server modes



Importance of H2 Database Management System

Its speed as well as its startup speed is fast

It is a browser-based console application

It has a low memory requirement of about 2 MB

It can generate SQL script files

It contains a recovery tool to dump the contents of the database

It is free from platform constraints

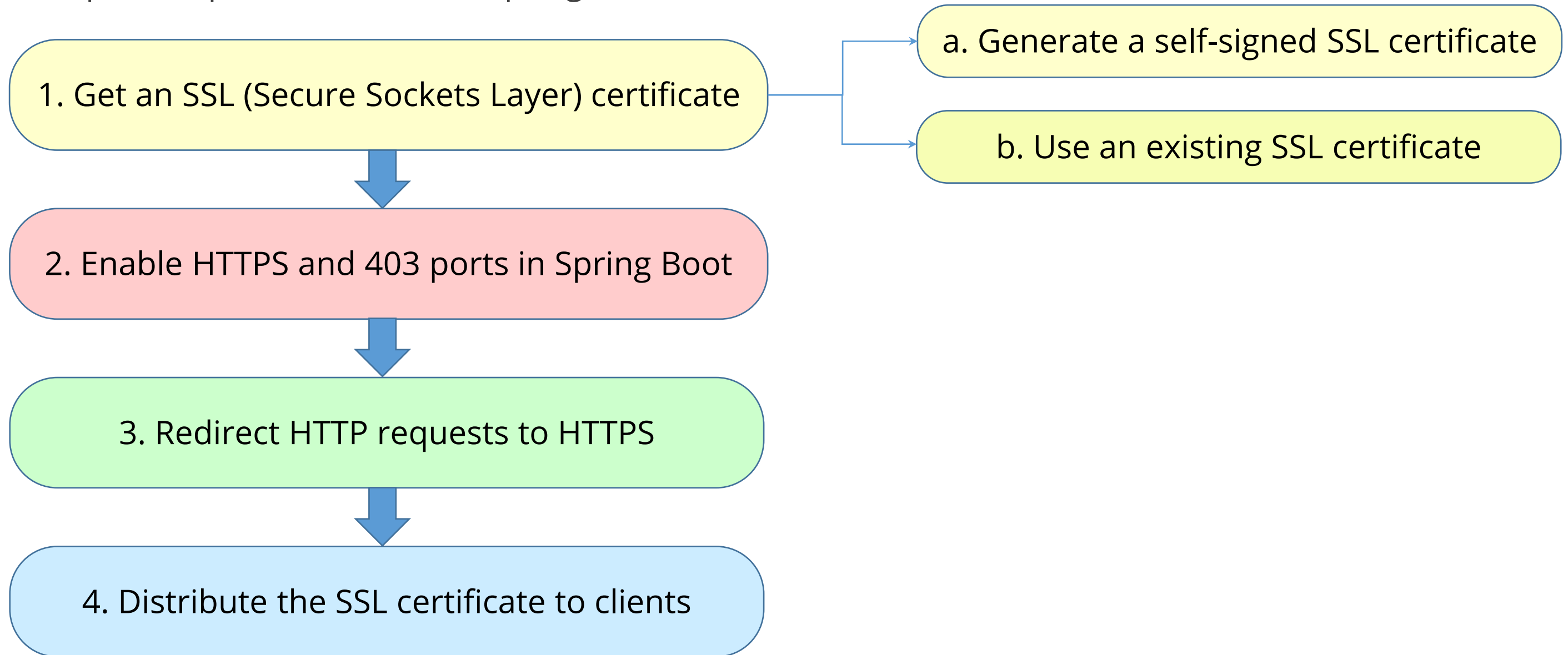
Spring Boot: HTTPS

Why Spring Boot HTTPS

- By default, Spring Boot application runs on HTTP 8080 port when the application starts
- So we need to configure HTTPS and enable 443 for Spring Boot application

Spring Boot: HTTPS

Steps to implement HTTPS in Spring Boot:



Assisted Practice

Assisted Practice: HTTPs for Spring Boot

Duration: 45 min.

Problem Statement:

Create a project to enable HTTPS and display in browser.

Assisted Practice: Guidelines

Steps to enable HTTPs in spring boot:

1. Install the STS Plugin in Eclipse
2. Create a Spring Boot Starter Project which is web-enabled
3. Use keytool (certificate manager) utility to create a self-signed SSL certificate key
4. Create MainController for showing a page in the browser under SSL
5. Configure application.properties to run the site in SSL
6. Build the Project
7. Publish and Start the Project
8. Run the Project
9. Push the code to GitHub repositories

Key Takeaways

- Spring Boot is an extension of Spring that makes development, testing, and deployment more convenient.
- There are two main approaches to handling exceptions in a Spring MVC application: `HandlerExceptionResolver` or the `@ExceptionHandler` annotation.
- Edge server is a server placed at the edge or end of a network and used to connect to other networks.
- Swagger avoids errors while writing code, ensures quality, and generates interactive documentation.
- Apache Kafka is used to build real-time pipelines and for streaming apps.



Displaying User Feedback

Problem Statement:

Create a Spring Boot project that will capture user feedback using a REST endpoint. The REST resource will take in parameters using HTTP POST. The feedback data will be then added to a database table.

