1. **Spring Boot**

Spring Boot is an open source Java-based framework used to create a micro Service. It is developed by Pivotal Team and is used to build stand-alone and production ready spring applications. This chapter will give you an introduction to Spring Boot and familiarizes you with its basic concepts.

What is Micro Service?

Micro Service is an architecture that allows the developers to develop and deploy services independently. Each service running has its own process and this achieves the lightweight model to support business applications.

1. **Spring MVC** [**https://www.baeldung.com/spring-template-engines**](https://www.baeldung.com/spring-template-engines)

The Spring web framework is built around the MVC (Model-View-Controller) pattern, which makes it easier to separate concerns in an application. This allows for the possibility to use different view technologies, from the well established JSP technology to a variety of template engines.

JSP, Thymeleaf, freemarker,mustache

a Java library to generate text output (HTML web pages, e-mails, configuration files, source code, etc.) based on templates and changing data

**2. Hibernate ORM** (or simply **Hibernate**) is an [object-relational mapping](https://en.wikipedia.org/wiki/Object-relational_mapping) tool for the [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) programming language. It provides a [framework](https://en.wikipedia.org/wiki/Software_framework) for mapping an [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) domain model to a [relational database](https://en.wikipedia.org/wiki/Relational_database). Hibernate handles [object-relational impedance mismatch](https://en.wikipedia.org/wiki/Object-relational_impedance_mismatch) problems by replacing direct, [persistent](https://en.wikipedia.org/wiki/Persistence_(computer_science)) database accesses with high-level object handling functions.

**Hibernate Query Language** (HQL)Hibernate provides an [SQL](https://en.wikipedia.org/wiki/SQL" \o "SQL) inspired language called [Hibernate Query Language](https://en.wikipedia.org/wiki/Hibernate_Query_Language) (HQL) for writing SQL-like queries against Hibernate's data objects. Criteria Queries are provided as an [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) alternative to HQL. Criteria Query is used to modify the objects and provide the restriction for the objects. HQL (Hibernate Query Language) is the object-oriented version of SQL. It generates database independent queries so that there is no need to write database-specific queries. Without this capability, changing the database would require individual SQL queries to be changed as well, leading to maintenance issues.

1. **Spring Data**

https://habr.com/ru/post/482552/

<https://spring.io/projects/spring-data-jpa#overview>

Spring Data JPA, part of the larger Spring Data family, makes it easy to easily implement JPA based repositories. This module deals with enhanced support for JPA based data access layers. It makes it easier to build Spring-powered applications that use data access technologies.

Implementing a data access layer of an application has been cumbersome for quite a while. Too much boilerplate code has to be written to execute simple queries as well as perform pagination, and auditing. Spring Data JPA aims to significantly improve the implementation of data access layers by reducing the effort to the amount that’s actually needed. As a developer you write your repository interfaces, including custom finder methods, and Spring will provide the implementation automatically.

@Repository – Jpa, Crud

Validation of @Query annotated queries at bootstrap time

Summary:

* PagingAndSortingRepository extends CrudRepository
* JpaRepository extends PagingAndSortingRepository

The CrudRepository interface provides methods for CRUD operations, so it allows you to create, read, update and delete records without having to define your own methods.

The PagingAndSortingRepository provides additional methods to retrieve entities using pagination and sorting.

Finally the JpaRepository add some more functionality that is specific to JPA.

Spring data jdbc, mongodb, redis, rest, apache cassandra

1. **Spring annotations**

@Service, @Controller, @Repository = {@Component + some more special functionality}

The @Component annotation marks a java class as a bean so the component-scanning mechanism of spring can pick it up and pull it into the application context. The @Service annotation is also a specialization of the component annotation. It doesn’t currently provide any additional behavior over the @Component annotation, but it’s a good idea to use @Service over @Component in service-layer classes because it specifies intent better. Additionally, tool support and additional behavior might rely on it in the future.

Classic controllers can be annotated with the @Controller annotation. This is simply a specialization of the @Component class and allows implementation classes to be autodetected through the classpath scanning.

@Controller is typically used in combination with a @RequestMapping annotation used on request handling methods.

1. **Liquibase**

* Version-controlled database schema changes
* Automatically orders scripts for deployment
* Branching and merging for teams
* Embeds into your product or build tools, like Jenkins
* Easily rollback changes

Meet the changeset

Liquibase uses *changesets* to represent a single change to your database. Here’s an example of a changeset to create a table.

Each changeset has an “id” and “author” attribute which, along with the directory and file name of the changelog file, uniquely identify it.

1. **Keycloak**

<https://dashboard.heroku.com/apps/my-web-bank>

<https://my-web-bank.herokuapp.com/auth/admin/master/console/#/realms/WebBankKeycloak/users>

The following embedded containers are supported now and don't require any extra dependencies if using Spring Boot Keycloak Starter:

* Tomcat
* Undertow
* Jetty

Keycloak is an open source Identity and Access Management solution aimed at modern applications and services. It makes it easy to secure applications and services with little to no code.

Single Sign-in

Users authenticate with Keycloak rather than individual applications. This means that your applications don't have to deal with login forms, authenticating users, and storing users. Once logged-in to Keycloak, users don't have to login again to access a different application.

This also applied to logout. Keycloak provides single-sign out, which means users only have to logout once to be logged-out of all applications that use Keycloak.

User federation

Keycloak has built-in support to connect to existing LDAP or Active Directory servers. You can also implement your own provider if you have users in other stores, such as a relational database.