DefineX Bootcamp Homework Week 1 - Hakan Afat

1- Java dünyasında Spring Boot, JSF, Apache Struts, Vaadin, Micronaut, vb. gibi web uygulamaları geliştirmede kullandığımız frameworkler bulunmaktadır. En çok bilinen ve kullanılan framework Spring Boot dur. Kod örnekleri ayrıca verilecektir.

1.1- Spring Boot sahip olduğu MVC, Dependency Injection veya Inversion of Control (IoC) özellikleri ile birçok ek projeye sahip olması nedeniyle tercih edilmektedir. Data, Security, Session, AI, vb. gibi. Inversion of Control sistemi yazdığımız kodlar üzerindeki objelerin yönetimini Bean sistemi üzerinden devralır. Framework gerektiğinde methodlar aracılığı ile bu objelere erişimi verir ve sonrasında geriye devir alır. Bu sayede multithreading ve performansın artması mümkün olur. Database bağlantıları için (JDBC, JPA) gerekli integrasyon desteği de bulunmaktadır.

1.2- Java EE üzerinde Servlets, JSP, JSF gibi frameworkleri vardır. JSF en çok bilinen MVC frameworkleri arasındadır. Component tabanlı sistemi ile çok katmanlı mimariye sahip olup endüstrideki büyük ölçekli firmaların terchilerindendir.

1.3- Apache Struct bir MVC framework udur. Üzerinde bulunan form processing, form validation ve interceptor özellikleri ile tercih sebebi olabilir. JSP ve Database bağlantıları için (JDBC, JPA) gerekli integrasyon desteği de bulunmaktadır.

1.4- Vaadin, UI öncelikli ve SSR(Server Side Rendering) dahili olarak tasarım odaklı bir frameworktur. Bu sayede web uygulamasındaki UI elementlerini backend tarafında eventleri de dahil olmak üzere paketlenip frontend kısmına gönderilmesini sağlar. Bu sayede HTML ve JS kullanmadan frontend geliştirmemizi sağlamaktadır.

1.5-Micronaut Microservice ve Serverless uygulamaları hedefleyen bir frameworktur. Düşük bellek tüketimi ve hızlı şekilde geliştirme yapmamızı amaçlar. Dahili bir test sistemi mevcuttur.

2- SOA servis tabanlı mimari anlamına gelmektedir. Kurumsal ölçekte artan ihtiyaçların çözümü için oluşturulmuştur. Çeşitli servislerin birbiri ile ortak bir bus sistemi üzerinden iletişim kurarak sorunu çözmesi hedeflenir. Bu sayede bir sistem çökse de onu etkilemeyen diğer sistemler ayakta kalabilir. Tek bir sistemin güncellenmesi yerine servislerin kendi içinde güncellenmesinin maliyeti çok daha azdır. Servisler SOAP, RESTful, Mesajlaşma protokolleri AMQP, binary iletişim protokolleri gRPC vb. kullanılarak haberleşebilirler.

SOAP Servisler XML üzerinden GET ve POST methodları üzerinden çalışan servislerdir. Request headerları arasına “Content-Type: application/soap+xml” header ını eklememiz gerekir. Örnek bir stock price isteği için request body si görülebilir.

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope" xmlns:m="http://www.example.org">
<soap:Header>
</soap:Header>
<soap:Body>
    <m:GetStockPrice>
        <m:StockName>T</m:StockName>
    </m:GetStockPrice>
</soap:Body>
</soap:Envelope>
```

RESTful Servisler çeşitli HTTP methodları üzerinden çalışan, veriyi request ve response body üzerinde XML veya JSON formatlarında taşıyan bir mimaridir.

2.1- GET /users - Bu örnekte user listesini json array formatında almak için göndeririz. Request bodysi boş olmalıdır. Herhangi bir limit veya search yapmak istersek;

GET /users/:userId
GET /users?limit=10
GET /users?search=55 şeklinde kullanabiliriz.

2.2- POST /users - Bu örneği yeni bir user ekleme için kullanabiliriz. Request e bir body vermemiz gerekir.

POST /users
Body {“firstName”: “Hakan”, “lastName”: “Afat”}

ve mutlaka request header kısmına gönderdiğimiz veri tipine göre “Content-Type: application/json” header i eklememiz gerekir.

2.3- PUT /users/:userId - Bu örneği bir user bilgilerini güncellemek için kullanabiliriz.

PUT /users/1
Body {“firstName”: “test”, “lastName”: “test”}

ve mutlaka request header kısmına gönderdiğimiz veri tipine göre “Content-Type: application/json” header i eklememiz gerekir.

2.4- DELETE /users/:userId - Bu örneği bir user silmek için kullanabiliriz.

DELETE /users/1

2.5- PATCH /users/:userId - Bu örneği bir user bilgilerinde bir kısmı güncellemek için kullanabiliriz.

PATCH /users/1
Body {“lastName”: “test 2”}

ve mutlaka request header kısmına gönderdiğimiz veri tipine göre “Content-Type: application/json” header i eklememiz gerekir.

2.6- OPTIONS ve HEAD - Bu istek browserlar tarafından sunucuyu kontrol amaçlı atılır. Bu sayede istek atılan route hangi methodlara izin verdiğini görürüz.

3- Proje içerisinde kullanıldı.
4- Bizimkredi projesi repoda bulunmaktadır. İstenen özellikler eklenmiştir.
5- Online alışveriş sistemi projesi repoda hw1 klasörü altında bulunmaktadır.