YAZILIM LABORATUVARI II

1.PROJE

KOCAELİ ÜNİVERSİTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

GONCAGÜL KOÇAK

200201108

200201108@kocaeli.edu.tr

Proje Tanımı

Bu dokümanda basit bir web arayüzünden girilen metinlerin belli kurallar doğrultusunda birleştirilerek bu birleştirilen metinleri JSON formatına dönüştürerek veritabanında depolamak amaçlanmaktadır.

Giriş

Projede basit bir web arayüzünde kullanıcıdan istediği kadar metin girişinin yapabildiği bir metin alanı oluşturulur.Arayüzde üç adet buton bulunmaktadır.Metin ekle, metin birleştir ve bu metinleri kaydedeceğimiz kaydet butonu vardır.Kullanıcı metin ekle butonuna tıkladıkça yeni metin alanları oluşacaktır ve bu alanlara metin girişleri yapılacaktır. Daha sonrasında kaydet butonuna basılarak kullanıcıdan girmiş olduğu metinler metinler.json dosyasında JSON formatında doküman olarak yazdırılacaktır. Metin birleştir butonuna basıldığında bu metinler birbirinin devamı olacak şekilde ortak kelimeler bir kere yazılacak şekilde birleştirme işlemi yapılacaktır.Metin birleştirme işlemi yapıldıktan sonra arayüzde birleştirilen metin cümlesi ve bu algoritmayı çözmek için geçen süreyi yazdıracaktır.

Araştırmalar ve yöntemler

Projede Java Spring, Html, JS,MongoDb kullanılmıştır.Bu proje VsCode da gerçekleşmiştir. İlk olarak Java Spring projesi açılı ve projeye entegre edilmesi gereken kütüphaneler, araçlar yüklenir.

```java
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class YazlabApplication {

    Run | Debug
    public static void main(String[] args) {
        SpringApplication.run(YazlabApplication.class, args);
    }
```

Bu kod Java Springle yapılan uygulamanın başlatılmasını sağlar.Burada YazlabApplication sınıfına bağlanarak SpringApplication.run metodu çalıştırılır.Uygulamanın çalışması için gerekli olan bileşenler otomatik olarak yapılandırılır.

```
import java.util.List;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapp
import org.springframework.web.bind.annotation.PostMap
import org.springframework.web.bind.annotation.Request
import com.fasterxml.jackson.core.JsonGenerationExcept
import com.fasterxml.jackson.databind.JsonMappingExcep
import com.fasterxml.jackson.databind.ObjectMapper;
import java.io.File;

import java.io.IOException;
```

KontrolSinifi.java sınıfında birleştirme işlemi yapılacaktır.Burada gerekli olan yapılar eklenmiştir.

```
@Controller
public class KontrolSinifi {
    private static final String FILE_NAME = "metinler.json";
    @GetMapping("/")
    public String index() {
        return "index";
    }

    @PostMapping("/submit-form")
    public String birlestir(@RequestParam List<String> metinler, Model model) {
```

@Controller, sınıfın bir Controller sınıfı olduğunu belirtir.Spring Frameworkün web isteklerini yönetmesini sağlar.Burada json formatında yazacağımız veriler için metinler.jsın dosyasını oluşturulur.

Return “index” sayfanın index sayfasına yönlendirilmesini sağlar.

Burada index.html sayfasından girilen metinler post edilerek birleştir fonksiyonuna gönderilir.HTTP POST isteği “submit-form” adresinde gerçekleşir ve bu istek, “metinler” adlı bir liste alır ve bu listeyi birleştirir.

```
long startTime = System.nanoTime();
long endTime = System.nanoTime();

long elapsedTime = endTime - startTime;


try {//metinler.json dosyasına kaydetme işlemi
    ObjectMapper mapper = new ObjectMapper();
    mapper.writeValue(new File(FILE_NAME), metinler);
} catch (JsonGenerationException e) {
    e.printStackTrace();
} catch (JsonMappingException e) {
    e.printStackTrace();
} catch (IOException e) {
    e.printStackTrace();
}
model.addAttribute("sure", elapsedTime);
```

System.nanoTime sitemin nanotime olarak geçen süreyi döndürür.Buradaki startTime başlangıç zamanı olarak verilmiştir, endTime değişkeni ise kod bloğunun bitişindeki süreyi verir.Daha sonra elapsedTime yani geçen, akan süre bulunur.elapsedTime ile algoritmanın ne kadar sürede çalıştığı bilgisine ulaşılır.Metinlerin birleştirilme işleminin süresi görünüme gönderilir.Web arayüzünde bu çıktı gösterilir.

```
StringBuilder birlesmisMetinBuilder = new StringBuilder();
String ortakKelime = "";

// ilk metni birleştir
String[] ilkKelimeler = metinler.get(0).split(" ");
for (int i = 0; i < ilkKelimeler.length; i++) {
    birlesmisMetinBuilder.append(ilkKelimeler[i]).append(
    if (i == ilkKelimeler.length - 1) {
        ortakKelime = ilkKelimeler[i];
    }
}
// diğer metinlerle karşılaştır ve birleştir
for (int i = 1; i < metinler.size(); i++) {
    String suankiMetin = metinler.get(i); // şu anki metin
    String[] suankiKelimeler = suankiMetin.split(" "); //

    int ortakKelimeIndex = -1;
    for (int j = 0; j < suankiKelimeler.length; j++) {
        if (suankiKelimeler[j].equals(ortakKelime)) {
            ortakKelimeIndex = j;
```

```
StringBuilder birlesmisMetinBuilder2 = new StringBuilder(birlesm
if (ortakKelimeIndex >= 0) {
    // şu anki metnin ortak kelimesinden sonra gelen kelimeleri
    for (int j = ortakKelimeIndex + 1; j < suankiKelimeler.lengt
        birlesmisMetinBuilder2.append(suankiKelimeler[j]).appen
    }
} else {
    // şu anki metni tamamen birleştir
    birlesmisMetinBuilder2.append(suankiMetin).append(" ");
}

ortakKelime = suankiKelimeler[suankiKelimeler.length - 1];
birlesmisMetinBuilder = birlesmisMetinBuilder2;
}

// birleşmiş metni model ile görünüme gönder
model.addAttribute("birlesmisMetin", birlesmisMetinBuilder.toString(
return "index";
}
```

Burada metin birleştirme işlemi gösterilmiştir.Geliştirmiş olduğum bu algoritma verilen metinlerin ortak kelimeye göre birleştirilmesini sağlar.İlk olarak ,ilk metin alınır ve kelimelerine ayrılır.Ardından bu kelime dizisi, bir StringBuilder nesnesi olan birlesmisMetinBuilder'da birleştirilir.Bu sırada son kelime de ortak kelime olarak olarak belirlenir.Daha sonra, diğer metinler bir döngü yardımıyla alınır ve her bir metin ortak kelimeye göre birleştirilir.İlk olarak şu anki metnin kelime dizisi, ortak kelimenin dizideki indeksini bulmak için taranır.Bu indeks bulunduktan sonra, şu anki metinde ortak kelimenin indesinden sonraki kelimeler birleştirilir.Ancak ortak kelime yoksa tüm şu anki metin direk olarak birleştirilir.Son olarak birleştirilen bu metin, model yardımıyla görünüme gönderilir ve index sayfasında gösterilir.

```
<p th:if="${birlesmisMetin != null}" >Birleştirilen Metin:</p>
<p th:if="${birlesmisMetin != null}" th:text="${birlesmisMetin}"></p>
<p th:if="${sure != null}">Algoritma Çözümleme Süresi:</p>
<p th:if="${sure != null}" th:text="${sure + ' milisaniye'}"></p>
```

index.html de birleştirilen metin ve geçen süre yazdırılır.Bu kod Thymleaf şablonu olarak web sayfasında görünüm için kullanılmıştır.

İlk iki satır birleşmiş metni diğer satırlar algoritma çözümleme sürecinde geçen süreyi milisaniye olarak yazdırmıştırç.

Bu başlıklar ve paragraflar sadece birleştirme işlmei yapıldığı vakit görüntülenmektedir.birlesmisMetin değişkeni null değilse, ilgili paragraf etiketi ve değişken değeri görüntülenir.

```
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Metin Birleştirme</title>
</head>
```

Bu kodların çalışması için bir web sunucusu gereklidir ve utgulama, bir web tarayıcısı üzerinden erişilebilir hale getirilir.Sayfa başlığı "Metin Birleştirme Uygulaması" olarak belirlenmiştir.Bu uygulama için bir

form eklenmiştir.Form, “submit-form” adlı bir eyleme gönderilecek şekilde belirlenmiştir.Bu form KontrolSinifi na gönderilir.Form kullanıların metinlerini girebilecekleri bir dizimetin kutusu içerir.Web arayüzünde metin ekle metinleri kaydet ve metinleri birleştir metodu bulunmaktadır.

Metin ekle düğmesine tıklandığında yeni bir metin kutusu eklenir.Bu kullanıcıların ihtiyaç duydukları kadar çok metin kutusunu ekleyebilmelerini sağlar.Metinleri kaydet düğmesine tıklayınca kullanıcının girmiş olduğu metinler JSON formatında kaydedilir.Kaydetme işlemi başarılı olursa alarm yayınlanır.Metinler bir dizi halinde yerel depolama alanında saklanır.

```
let metinSayisi = 0;
function metinEkle() {
    metinSayisi++;
    let metinKutulariDiv = document.getElementById("metinKutulari");
    let yeniMetinKutusu = document.createElement("inputsure");
    yeniMetinKutusu.type = "text";
    yeniMetinKutusu.name = "metinler";
    yeniMetinKutusu.classList.add("metinInput");
    let yeniMetinEtiketi = document.createElement("label");
    yeniMetinEtiketi.setAttribute("for", "metin" + metinSayisi);
    yeniMetinEtiketi.innerHTML = "Metin " + metinSayisi + ": ";
    metinKutulariDiv.appendChild(yeniMetinEtiketi);
    metinKutulariDiv.appendChild(yeniMetinKutusu);
    metinKutulariDiv.appendChild(document.createElement("br"));
}
```

metinEkle fonksiyonu çağrıldığında önce metinSayisi değişkenini 1 arttırır.Daha sonra metinKutularıDiv adlı bir değişkene HTML formunda yeni metin kutularının yer alacağı div elementini atar.Sonra “yeniMetinKutusu” adında bir input elementi oluşturulur.

Bu elementin tipi “text” olarak ayarlanır.Ardından “yeniMetinKutusu” elementine “metinInput” adında yeni bir sınıf ekler.Daha sonrasında “yeniMetinEtiketi” adında bir “label” elementi oluşturur ve bu elementin “for niteliğine “metin” ve yeni metinsayisi olarak ayarlanır.

En son olara yeni metin kutusu ve etiketi “metinKutulariDiv” elementine eklenir.Bu fonksiyon sayesinde kullanıcılar HTML formuna yeni metin kutuları ekleyebileceklerdir.

Sonuç:

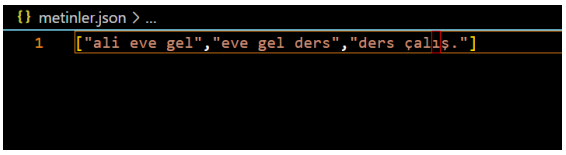
Sonuç

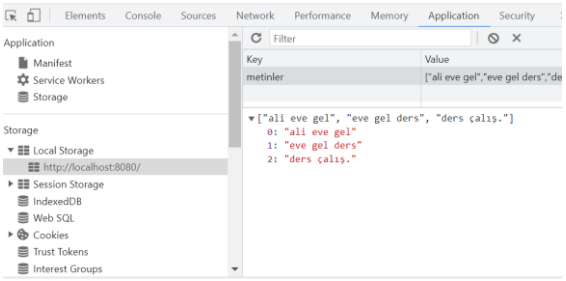
Birleştirilen Metin:

ali eve gel ders çalış.

Algoritma Çözümleme Süresi:

600 milisaniye





Deneysel Sonuçlar:

Projede Java Spring kullanılarak HTML CSS kullanılarak proje oluşturulmuştur.Genel çıktılar ekran görüntüsü olarak verilmiştir.

Geliştirme Ortamı:

Geliştirme ortamı olarak VsCode Java Spring, Css HTML kodlarından yararlanılmıştır.Veritabanı olarak MongoDb kullanılmıştır.

KABA KOD:

1:BAŞLA.

2:Tarayıcıdan localhost:8080 i aç.

3:Metin ekle butonuna tıkla.

4:Metin sayısını arttır.

5:Metin ekle butonuna tıkladıkça yeni metin alanı oluştur.

6:Metin girişi yap.

7:Kaydet butonuna tıkla.Kaydet fonksiyonunu çalıştır.

8:metinler.json dosyası oluştur.

9:Kaydedilen metinleri json formatında doküman olarak kaydet.

10:Metin birleştir butonuna tıkla.

11:Metin birleştirme fonksiyonunu çalıştır.

12:İlk iki cümlenin ortak kelimesini bul.İlk kelime ortak kelimeye kadar ikinci kelime ortak kelimeden itibaren yazdır.Birleşmiş metin olarak ata.

13.Yeni metin girişi yapıldığında birleşmiş metinle karşılaştır.Foksiyonu çalıştır.

14.Geçen süreyi hesapla.

15:Birleşmiş metni ve geçen süreyi arayüze yazdır.

16:BİTİR.

Kaynakça:

- https://medium.com/kodgemisi/spring-boot-ile-%C3%B6rnek-web-uygulamas%C4%B1-914c94c9099f
- https://www.youtube.com/watch?v=elDWNzvQtI0&list=PL5Y4hrqdSffum17oiTxpeG2-3jy777mTZ&index=2
- https://www.youtube.com/watch?v=GMNGPry979k
- https://medium.com/@deeksha.sharma25/spring-boot-web-app-serving-static-html-page-2cb5ab3a195b#:~:text=This%20is%20a%20very%20basic,World%E2%80%9D%20static%20HTML%20web%20page.&text=This%20will%20start%20the%20embedded,the%20box%20with%20Spring%20Boot
- https://medium.com/@deeksha.sharma25/spring-boot-web-app-serving-static-html-page-2cb5ab3a195b#:~:text=This%20is%20a%20very%20basic,World%E2%80%9D%20static%20HTML%20web%20page.&text=This%20will%20start%20the%20embedded,th

e%20box%20with%20Spring%20Boot.

- https://www.codejava.net/java-ee/servlet/handling-html-form-data-with-java-servlet
-