- New chat
- Office Timing Calculator
- Vault API version interface
- Apache Spring from Rogger
- Search Features
- Pure Fair
- ERR: Microservice with Rust
- ERR: Template Microservice
- ECR: Code implementation
- application.properties not ap
- Using MinIO with Spring Boo

To remove the DOCTYPE text from Rise dropdown, you can simply remove the `<p>Select` element from your HTML file. The `<p>Select` element inserts the DOCTYPE text, so removing this element will remove the DOCTYPE text.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width">
    <title>Office Timing Calculator</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <p>Select</p>
    <div id="root">
        <div>
            <h1>Office Timing Calculator</h1>
            <form>
                <label for="start_time">Start time:</label>
                <input type="time" id="start_time" name="start_time">
                <br>
                <label for="end_time">End time:</label>
                <input type="time" id="end_time" name="end_time">
                <br>
                <button type="submit">Calculate</button>
            </form>
        </div>
    </div>
</body>
</html>
```

Note that I also added `<div id="root">` to the `<div>` elements to hide them from view.

1 ty. esp. mjh kafka parte sabin ha ty.

Haan, main esp.mjh Apache Kafka ka bare mein sun(hear mein musbat bar sabin ha. Kafka ek open-source distributed streaming platform hai jo high volume data streams ko handle karne mein madad karta hai. Kafka ka use case mein real time data streaming, data pipeline building, log aggregation, event sourcing, messaging systems jab mein samaaiye, etc. shamil hai.

Kafka mein, data ko topic ke andar publish kiya jata hai, jahan par multiple consumers bhi distribute kiya jata hai. Ye consumer groups ke saath messaging system jab mein samaaiye kiya sakte hai, jahan par consumer groups ke saath consumers same data ko consume karte hai. Kafka ek distributed system hai, jisaane brokers, ek cluster ke andar topic partition mein divide kiya jata hai. Yeh partition multiple brokers par replicated hota hai, jisse data redundancy aur high availability ke liye mein response hota hai.

Jaga esp.mjh Kafka ka bare mein aur bhi adhik jankari chahiye toh mein esp.mjh ke kafka ka bare

1 Iske spring boot project mein ty.i use ke example ka diaaaaa kaaay

Kafka ka Spring Boot project ek aya sana ka Ek Ejab esp.mjh kaa libraries aur dependencies add karne hoogi. Niche diye gaye steps follow karein:

1. Apna Spring Boot project ke `spring.kafka` dependency add karein. Iske liye aap apna `pom.xml` file me yeh dependency add kar sakte hain:

```
pom
<dependency>
    <groupId>org.springframework.kafka</groupId>
    <artifactId>spring-kafka</artifactId>
    <version>2.5.7.RELEASE</version>
</dependency>
```

2. Spring Boot ka configuration file me Kafka broker ka configuration add karein. Example ke liye, aap yh configuration use kar sakte hain:

```
config
spring.kafka.bootstrap-servers=localhost:9092
```

3. Kafka producer ka configure karne ka liye, aap `KafkaProducer` ka use kar sakte hain. Iske liye, aap `KafkaTemplate` ka inject karne aur uska `@Autowired` annotation ka use karke aur kar sakte hain. Example:

```
controller
@RestController
public class KafkaProducerController {

    @Autowired
    private KafkaTemplate<String, String> kafkaTemplate;

    @Value("${kafka.topic.name}")
    private String topicName;

    @Value("${kafka.topic.partitions}")
    private int[] partitions;

    @Value("${kafka.topic.replication.factor}")
    private int replicationFactor;

    @Bean
    public KafkaAdmin.NewTopic[] kafkaTopics() {
        return new KafkaAdmin.NewTopic[]{
            new KafkaAdmin.NewTopic(topicName, partitions, (short) replicationFactor)
        };
    }

    @Bean
    public KafkaAdmin.NewTopic[] kafkaTopics(String[] topicNames, int[] partitions, int[] replicationFactors) {
        int[] defaultPartitions = new int[topicNames.length];
        int[] defaultReplicationFactors = new int[topicNames.length];
        for (int i = 0; i < topicNames.length; i++) {
            defaultPartitions[i] = partitions[i];
            defaultReplicationFactors[i] = replicationFactors[i];
        }
        return new KafkaAdmin.NewTopic[]{
            new KafkaAdmin.NewTopic(topicNames[0], defaultPartitions[0], (short) defaultReplicationFactors[0])
            new KafkaAdmin.NewTopic(topicNames[1], defaultPartitions[1], (short) defaultReplicationFactors[1])
            new KafkaAdmin.NewTopic(topicNames[2], defaultPartitions[2], (short) defaultReplicationFactors[2])
            new KafkaAdmin.NewTopic(topicNames[3], defaultPartitions[3], (short) defaultReplicationFactors[3])
            new KafkaAdmin.NewTopic(topicNames[4], defaultPartitions[4], (short) defaultReplicationFactors[4])
            new KafkaAdmin.NewTopic(topicNames[5], defaultPartitions[5], (short) defaultReplicationFactors[5])
            new KafkaAdmin.NewTopic(topicNames[6], defaultPartitions[6], (short) defaultReplicationFactors[6])
            new KafkaAdmin.NewTopic(topicNames[7], defaultPartitions[7], (short) defaultReplicationFactors[7])
            new
```