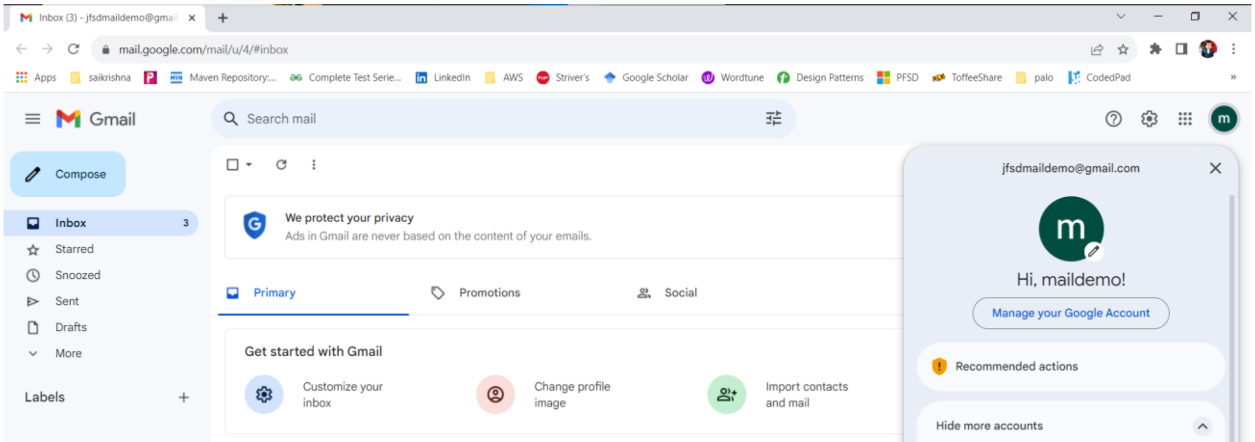
Sending Email Demo using Spring Boot MVC

For implementing the Java Mail Service, we need the following dependencies :

- Spring web
- Lombok
- Java mail
- Spring data JPA
- Mysql driver

For Password Generation :

Click on “Manage Your Google Account”

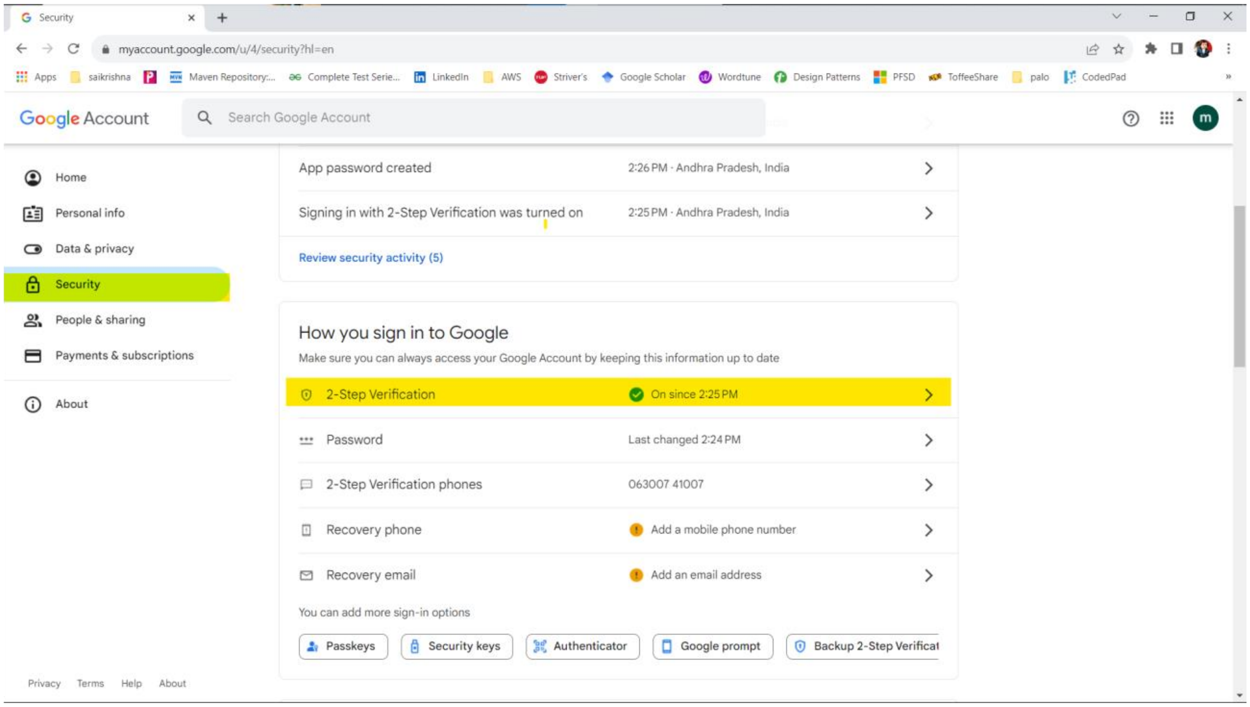


We protect your privacy
Ads in Gmail are never based on the content of your emails.

jfsdmaildemo@gmail.com

Hi, maildemo!

Turn the 2-Step Verification On



App password created 2:26 PM · Andhra Pradesh, India

Signing in with 2-Step Verification was turned on 2:25 PM · Andhra Pradesh, India

How you sign in to Google

Make sure you can always access your Google Account by keeping this information up to date

2-Step Verification On since 2:25 PM

Password Last changed 2:24 PM

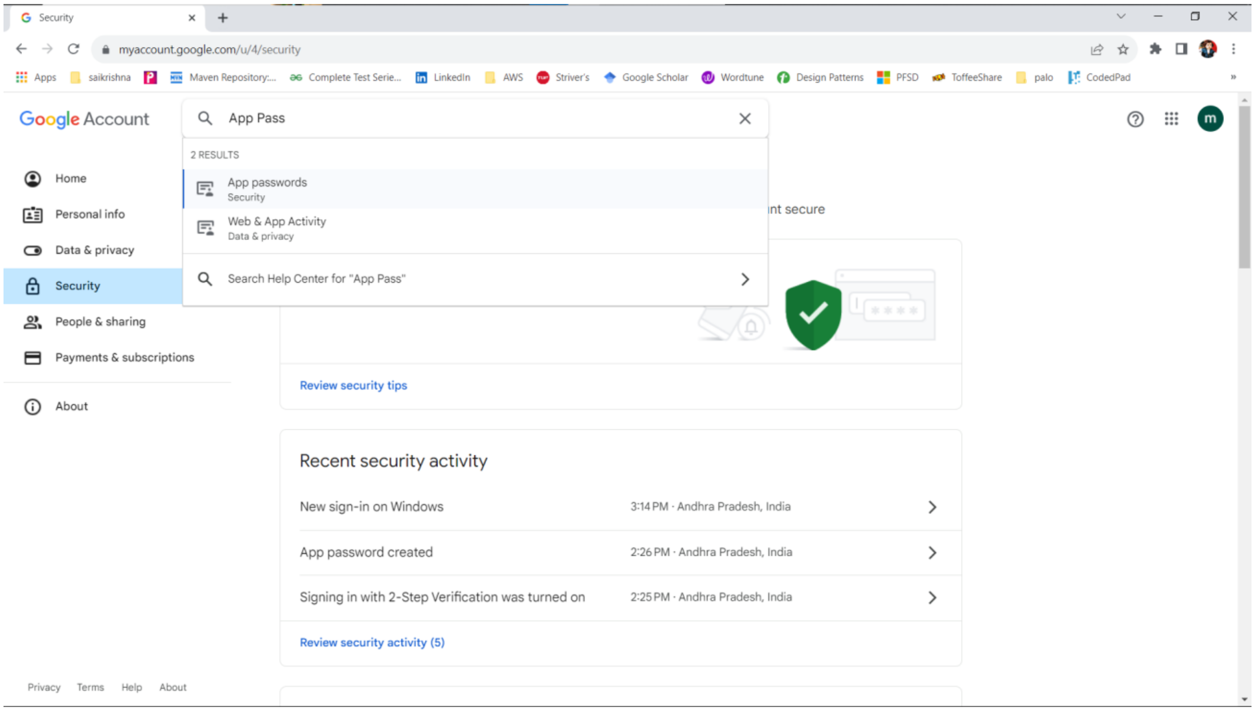
2-Step Verification phones 063007 41007

Recovery phone Add a mobile phone number

Recovery email Add an email address

You can add more sign-in options

Search for App Passwords, select the first one.



App Pass

2 RESULTS

App passwords
Security

Web & App Activity
Data & privacy

Search Help Center for "App Pass"

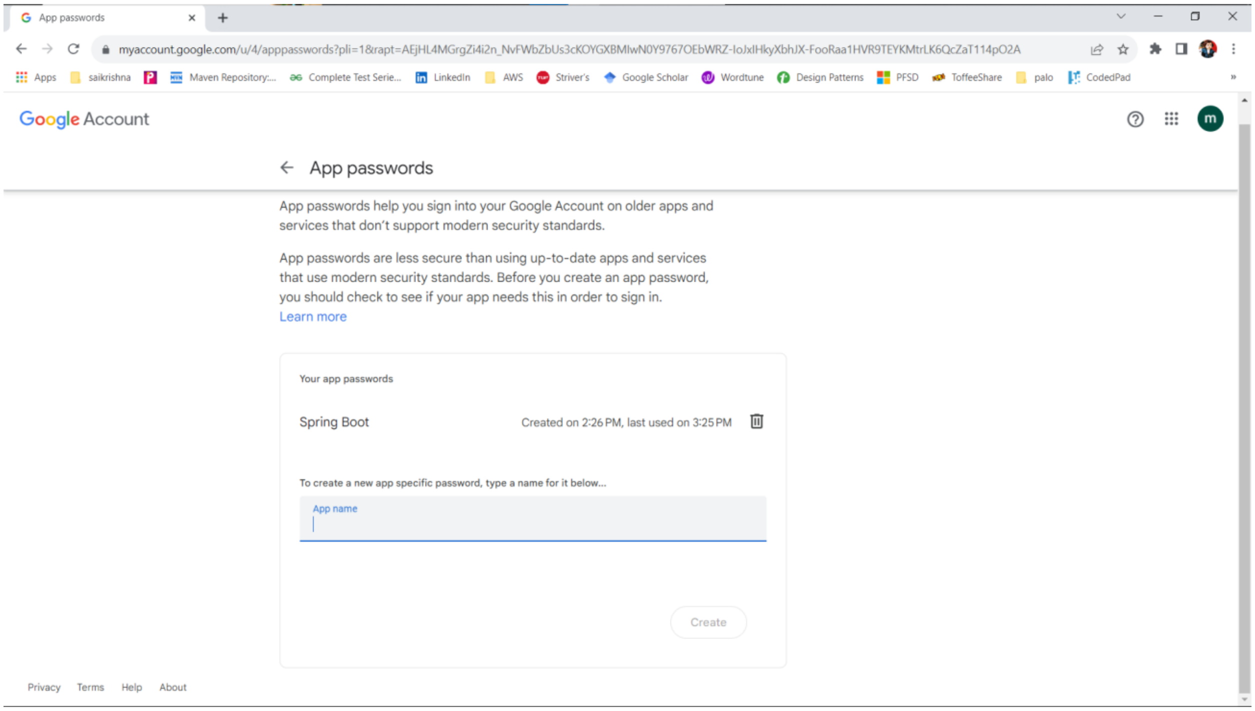
nt secure

Review security tips

Recent security activity

New sign-in on Windows	3:14 PM · Andhra Pradesh, India	›
App password created	2:26 PM · Andhra Pradesh, India	›
Signing in with 2-Step Verification was turned on	2:25 PM · Andhra Pradesh, India	›

Review security activity (5)



App passwords

App passwords help you sign into your Google Account on older apps and services that don't support modern security standards.

App passwords are less secure than using up-to-date apps and services that use modern security standards. Before you create an app password, you should check to see if your app needs this in order to sign in.
Learn more

Your app passwords

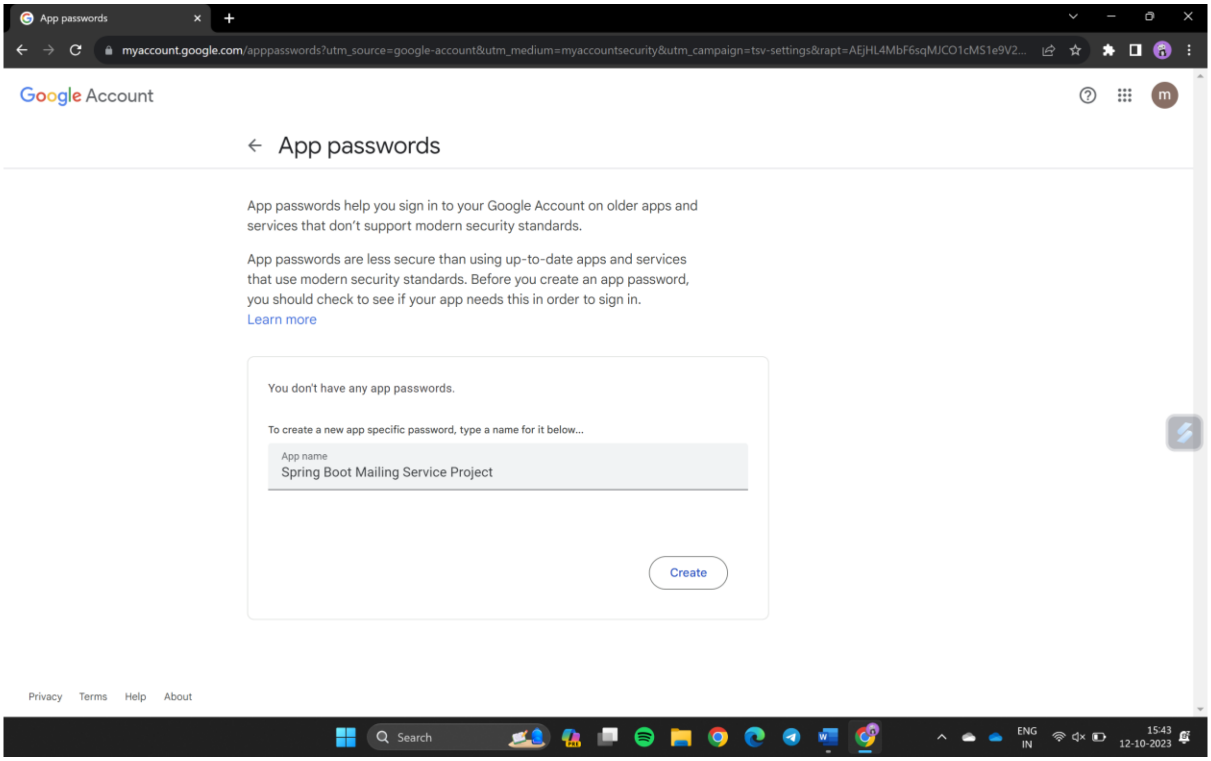
Spring Boot	Created on 2:26 PM, last used on 3:25 PM

To create a new app specific password, type a name for it below...

App name

Create

Give a name (Spring Boot)



App passwords
myaccount.google.com/apppasswords?utm_source=google-account&utm_medium=myaccountsecurity&utm_campaign=tsv-settings&rapt=AEjHL4MbF6sqMJCO1cMS1e9V2...
Google Account
Privacy
Terms
Help
About
Search
ENG
IN
15:43
12-10-2023

App passwords

App passwords help you sign in to your Google Account on older apps and services that don't support modern security standards.

App passwords are less secure than using up-to-date apps and services that use modern security standards. Before you create an app password, you should check to see if your app needs this in order to sign in.
Learn more

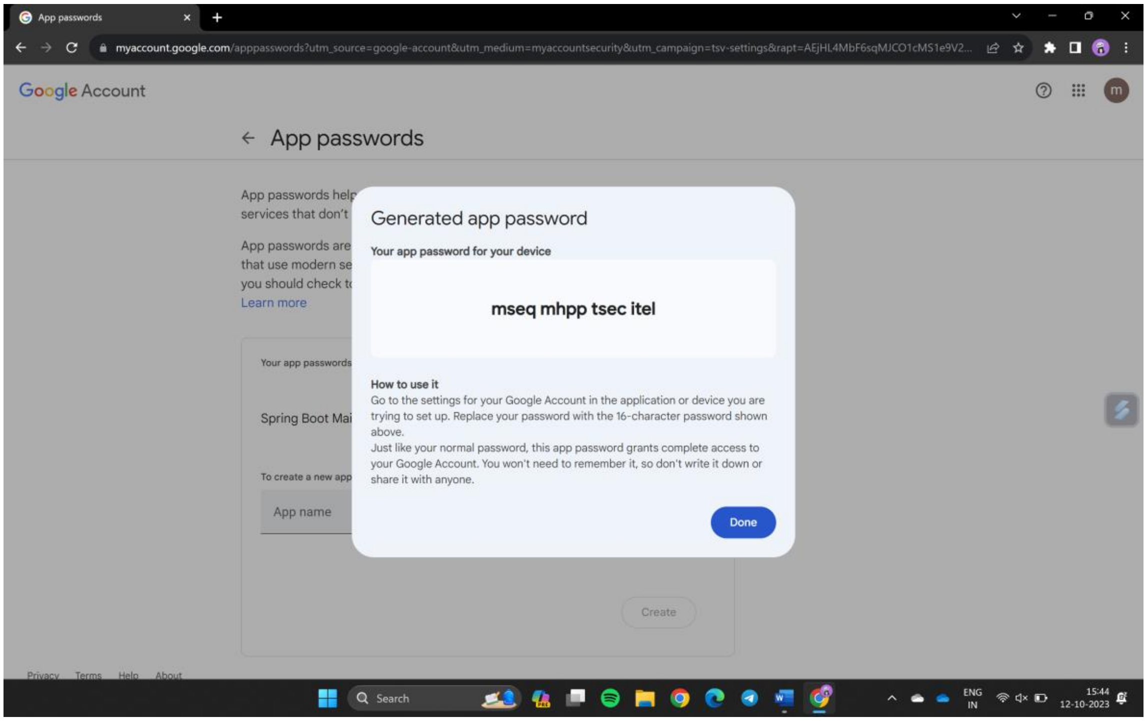
You don't have any app passwords.

To create a new app specific password, type a name for it below...

App name
Spring Boot Mailing Service Project

Create

Click Create.



Generated app password

Your app password for your device

mseq mhpp tsec itel

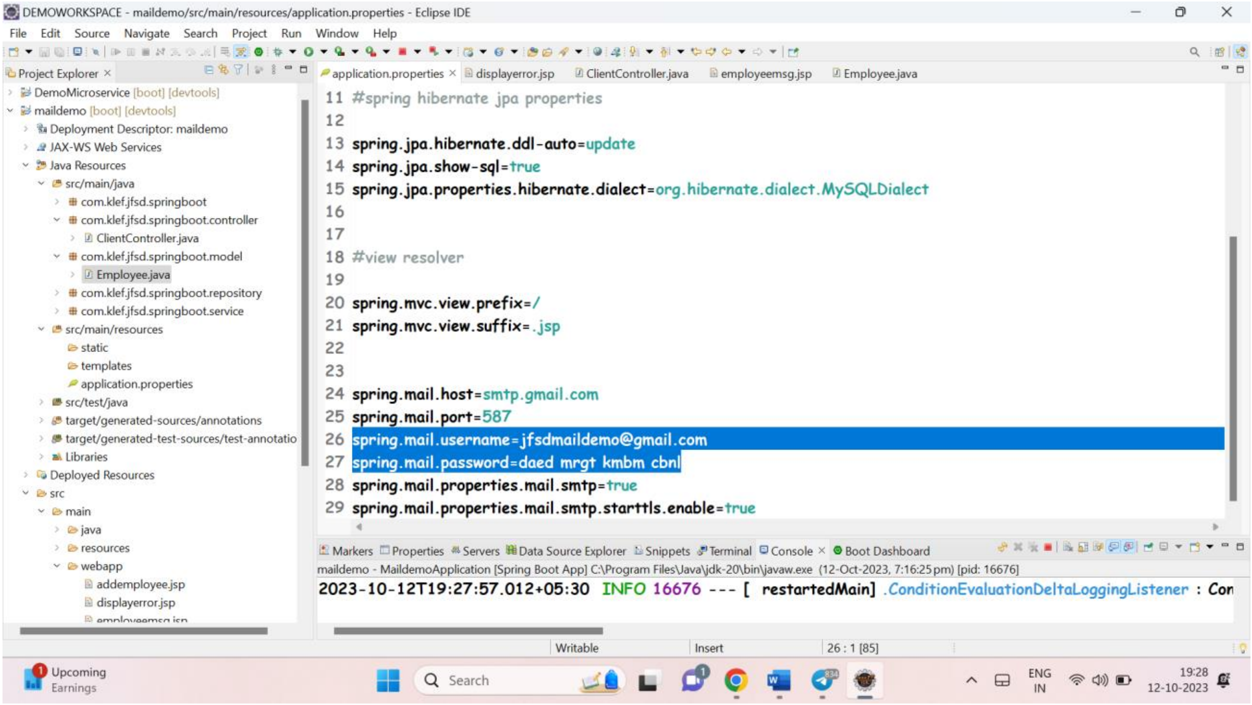
How to use it
Go to the settings for your Google Account in the application or device you are trying to set up. Replace your password with the 16-character password shown above.
Just like your normal password, this app password grants complete access to your Google Account. You won't need to remember it, so don't write it down or share it with anyone.

Done

A dialogue box will appear, copy it (Save it somewhere)

Copy the password and write it in the application.properties

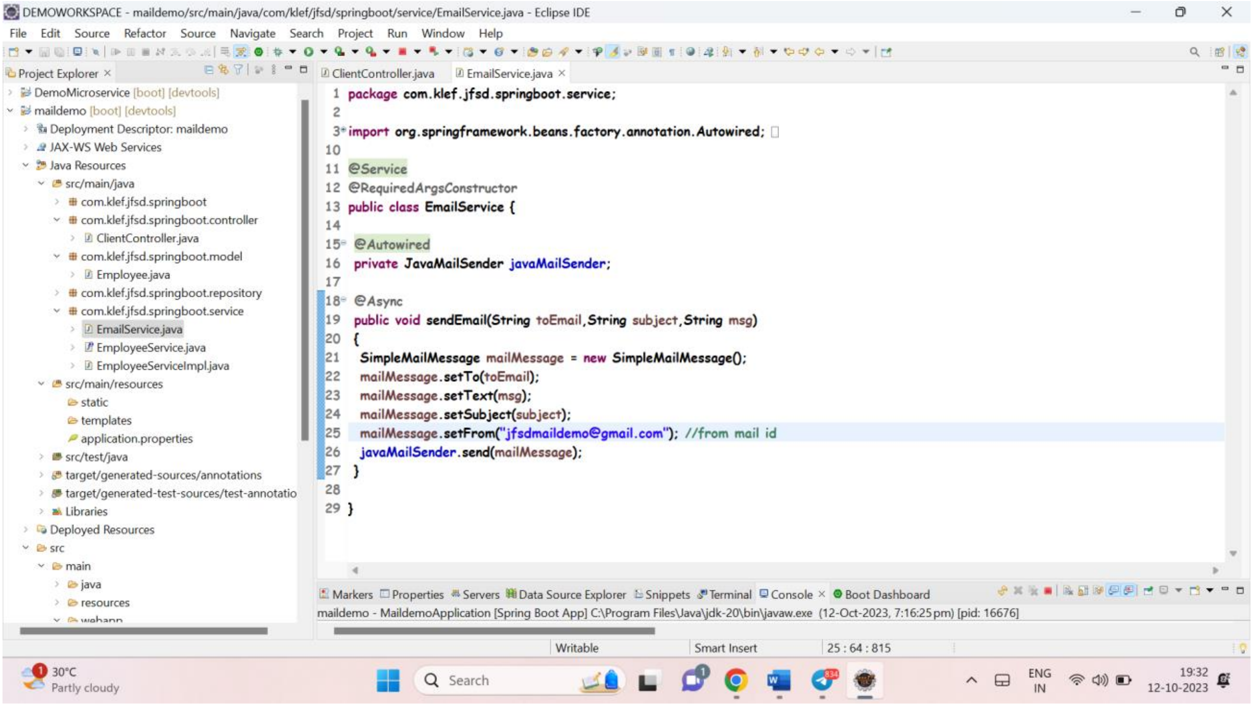
applications.properties



```
#spring hibernate jpa properties

spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect


#view resolver

spring.mvc.view.prefix=/
spring.mvc.view.suffix=.jsp


spring.mail.host=smtp.gmail.com
spring.mail.port=587
spring.mail.username=jfsdmaildemo@gmail.com
spring.mail.password=daed mrgt kmbm cbnl
spring.mail.properties.mail.smtp=true
spring.mail.properties.mail.smtp.starttls.enable=true
```

```
maildemo - MaildemoApplication [Spring Boot App] C:\Program Files\Java\jdk-20\bin\javaw.exe  (12-Oct-2023, 7:16:25 pm) [pid: 16676]
2023-10-12T19:27:57.012+05:30  INFO 16676 --- [  restartedMain] .ConditionEvaluationDeltaLoggingListener : Con
```

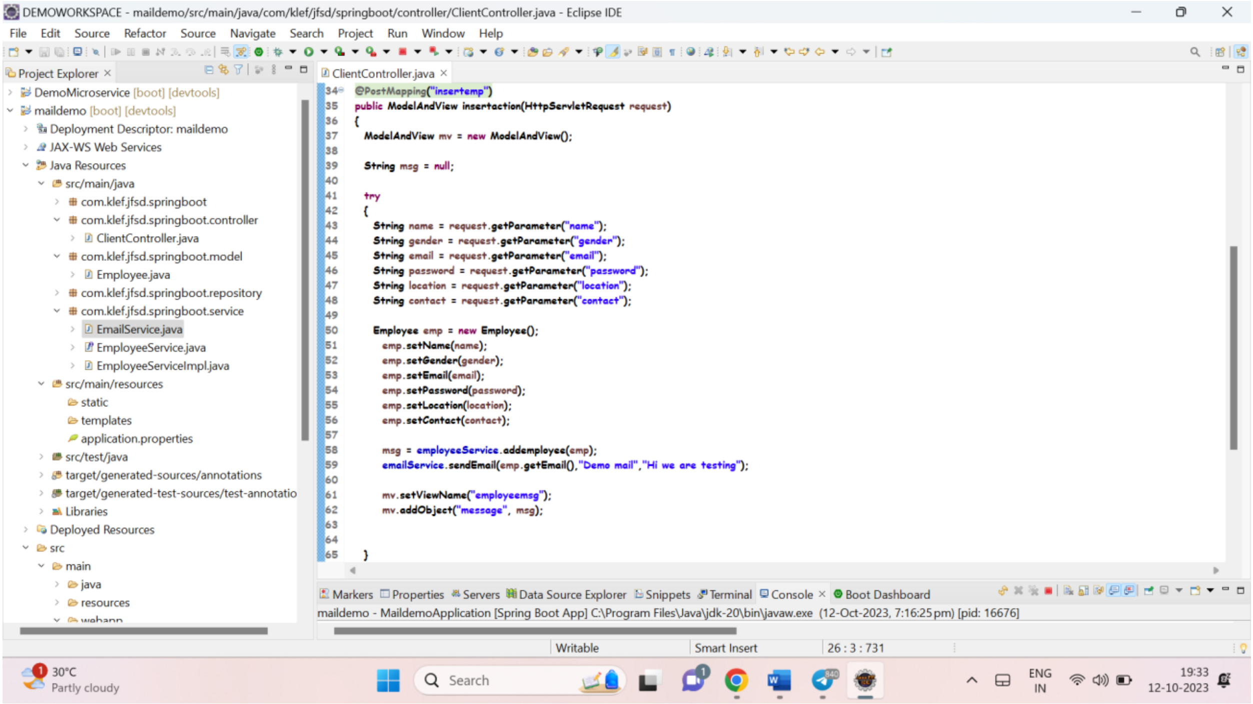
```
spring.mail.host=smtp.gmail.com
spring.mail.port=587
spring.mail.username=jfsdmaildemo@gmail.com
spring.mail.password=daed mrgt kmbm cbnl
spring.mail.properties.mail.smtp=true
spring.mail.properties.mail.smtp.starttls.enable=true
```



```java
package com.klef.jfsd.springboot.service;

import org.springframework.beans.factory.annotation.Autowired; □

@Service
@RequiredArgsConstructor
public class EmailService {

 @Autowired
 private JavaMailSender javaMailSender;

 @Async
 public void sendEmail(String toEmail,String subject,String msg)
 {
  SimpleMailMessage mailMessage = new SimpleMailMessage();
  mailMessage.setTo(toEmail);
  mailMessage.setText(msg);
  mailMessage.setSubject(subject);
  mailMessage.setFrom("jfsdmaildemo@gmail.com"); //from mail id
  javaMailSender.send(mailMessage);
 }

}
```

```
maildemo - MaildemoApplication [Spring Boot App] C:\Program Files\Java\jdk-20\bin\javaw.exe  (12-Oct-2023, 7:16:25 pm) [pid: 16676]
```

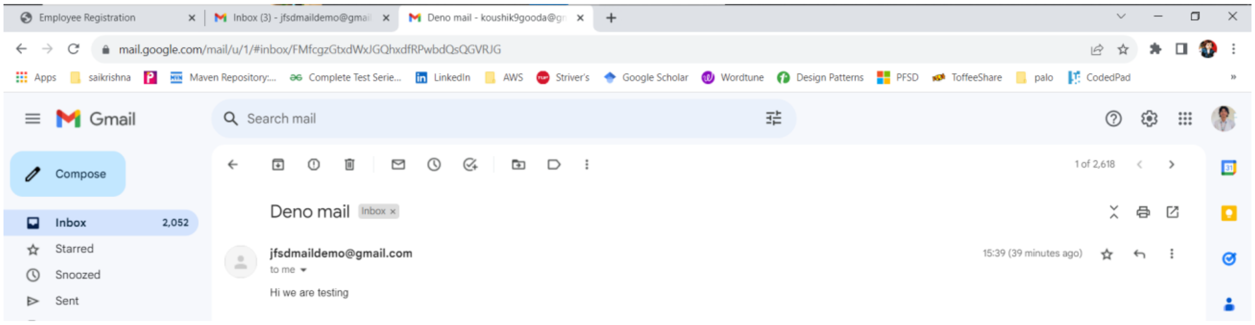
```
package com.klef.jfsd.springboot.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.mail.SimpleMailMessage;
import org.springframework.mail.javamail.JavaMailSender;
import org.springframework.scheduling.annotation.Async;
import org.springframework.stereotype.Service;

import lombok.RequiredArgsConstructor;

@Service
@RequiredArgsConstructor
public class EmailService {

 @Autowired
 private JavaMailSender javaMailSender;

 @Async
 public void sendEmail(String toEmail,String subject,String msg)
 {
  SimpleMailMessage mailMessage = new SimpleMailMessage();
  mailMessage.setTo(toEmail);
  mailMessage.setText(msg);
  mailMessage.setSubject(subject);
  mailMessage.setFrom("jfsdmaildemo@gmail.com"); //from mail id
  javaMailSender.send(mailMessage);
 }

}
```


DEMOWORKSPACE - maildemo/src/main/java/com/klef/jfsd/springboot/controller/ClientController.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Project Explorer
ClientController.java
Markers Properties Servers Data Source Explorer Snippets Terminal Console Boot Dashboard
Writable
Smart Insert
26 : 3 : 731

```java
@PostMapping("insertemp")
public ModelAndView insertaction(HttpServletRequest request)
{
  ModelAndView mv = new ModelAndView();

  String msg = null;

  try
  {
    String name = request.getParameter("name");
    String gender = request.getParameter("gender");
    String email = request.getParameter("email");
    String password = request.getParameter("password");
    String location = request.getParameter("location");
    String contact = request.getParameter("contact");

    Employee emp = new Employee();
      emp.setName(name);
      emp.setGender(gender);
      emp.setEmail(email);
      emp.setPassword(password);
      emp.setLocation(location);
      emp.setContact(contact);

      msg = employeeService.addemployee(emp);
      emailService.sendEmail(emp.getEmail(),"Demo mail","Hi we are testing");

      mv.setViewName("employeemsg");
      mv.addObject("message", msg);


  }
```

```
maildemo - MaildemoApplication [Spring Boot App] C:\Program Files\Java\jdk-20\bin\javaw.exe (12-Oct-2023, 7:16:25 pm) [pid: 16676]
```

call the "emailService.sendEmail" in the "insertemp" action in ClientController so that, after clicking on the register button, the user will get an email saying, "Demo Email" (Subject), "Hi we are testing" (message).


Employee Registration
Inbox (3) - jfsdmaildemo@gmail
Deno mail - koushik9gooda@g
mail.google.com/mail/u/1/#inbox/FMfcgzGtxdWxJGQhxdfRPwbdQsQGVRJG
Gmail
Search mail
Compose
Inbox 2,052
Starred
Snoozed
Sent
1 of 2,618

Deno mail Inbox

jfsdmaildemo@gmail.com
to me

15:39 (39 minutes ago)

Hi we are testing

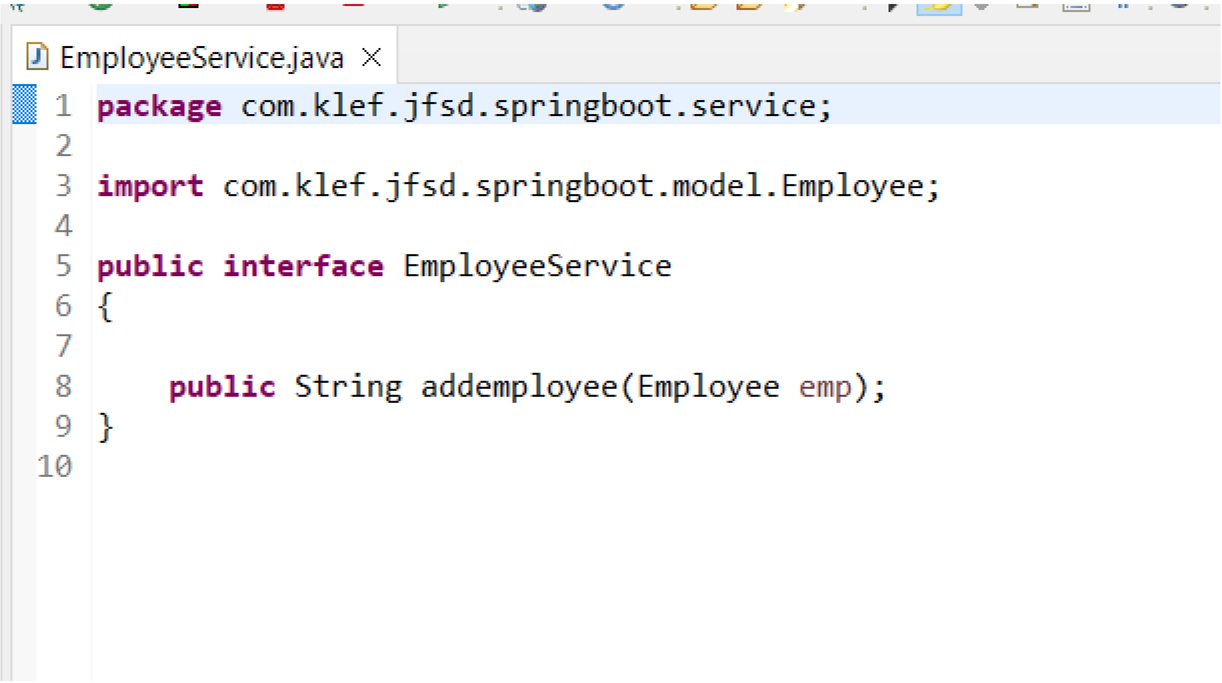
Change the subject and message accordingly.

ClientController.java

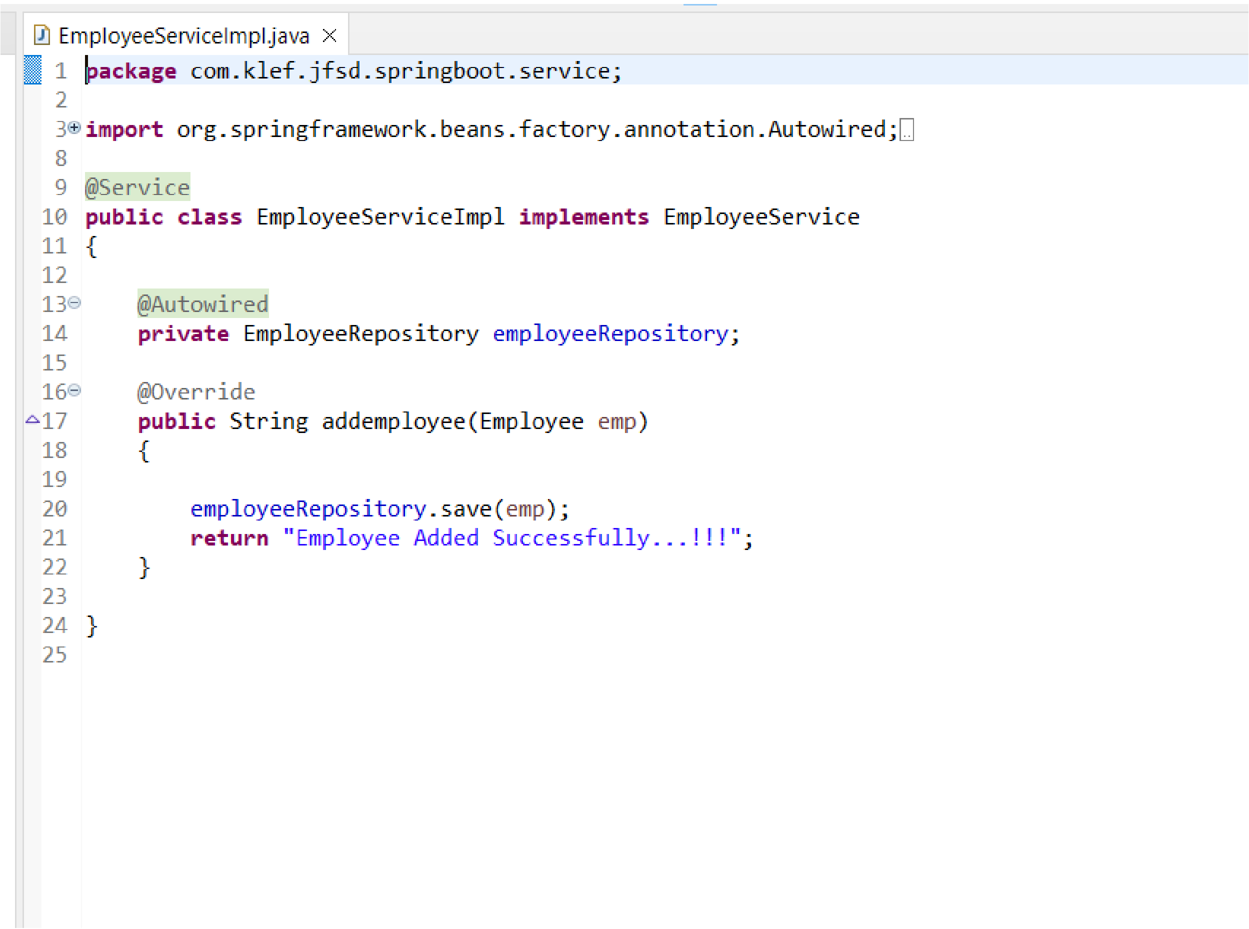
```
package com.klef.jfsd.springboot.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.servlet.ModelAndView;
import com.klef.jfsd.springboot.model.Employee;
import com.klef.jfsd.springboot.service.EmailService;
import com.klef.jfsd.springboot.service.EmployeeService;
import jakarta.servlet.http.HttpServletRequest;

@Controller
public class ClientController
{
  @Autowired
  private EmployeeService employeeService;
  @Autowired
  private EmailService emailService;
  @GetMapping("")
  public String empregistration()
  {
    return "addemployee";
  }
```

```
@PostMapping("insertemp")
public ModelAndView insertaction(HttpServletRequest request)
{
 ModelAndView mv = new ModelAndView();

 String msg = null;

 try
 {
  String name = request.getParameter("name");
  String gender = request.getParameter("gender");
  String email = request.getParameter("email");
  String password = request.getParameter("password");
  String location = request.getParameter("location");
  String contact = request.getParameter("contact");

  Employee emp = new Employee();
   emp.setName(name);
   emp.setGender(gender);
   emp.setEmail(email);
   emp.setPassword(password);
   emp.setLocation(location);
   emp.setContact(contact);
   msg = employeeService.addemployee(emp);
   emailService.sendEmail(emp.getEmail(),"Demo mail","Hi we are testing");
   mv.setViewName("employeemsg");
   mv.addObject("message", msg);
 }
```

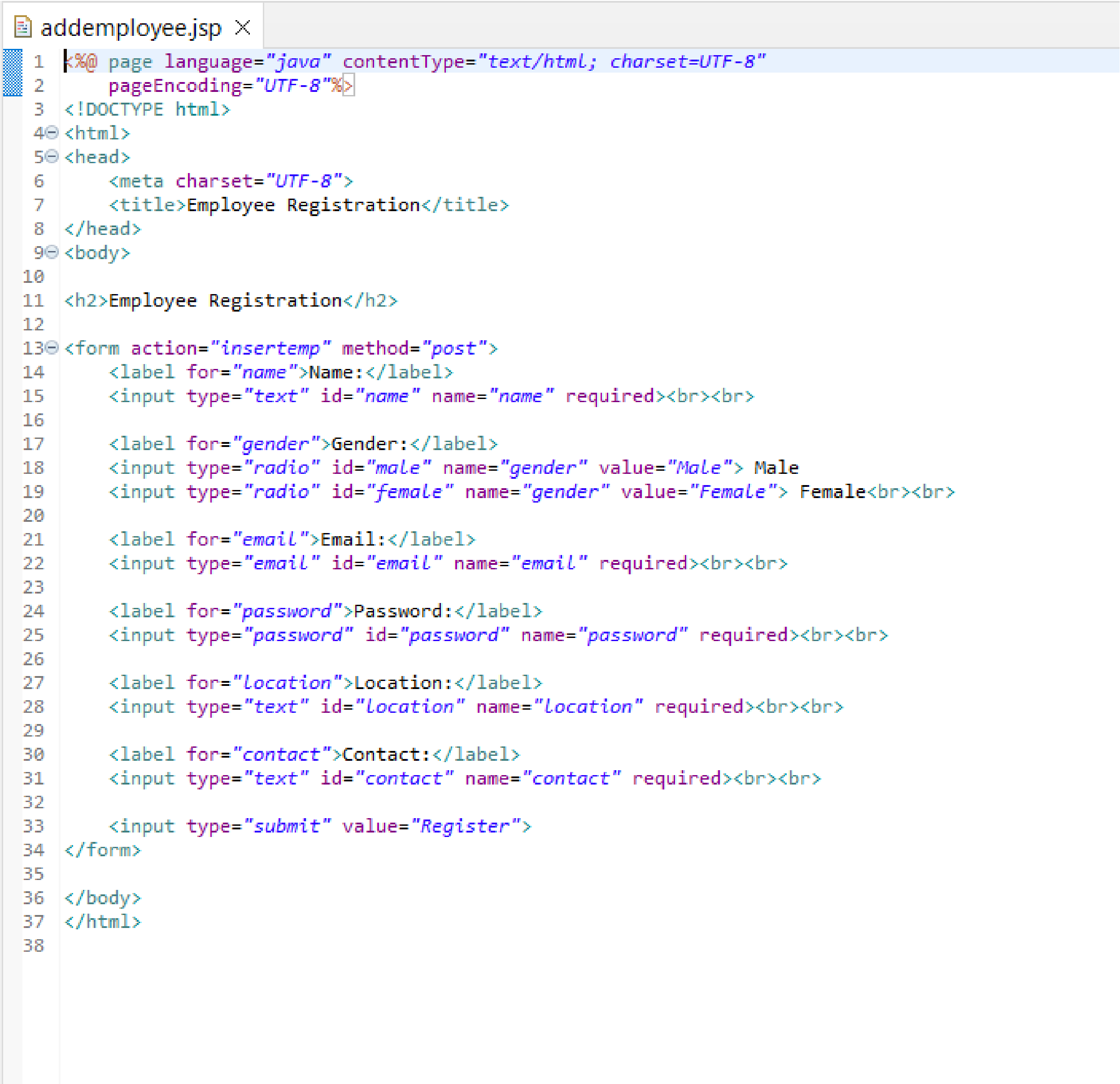
```
   catch(Exception e)
   {
    mv.setViewName("displayerror");
    msg = e.getMessage();
    mv.addObject("message", msg);
   }
   return mv;
  }
}
```



```
package com.klef.jfsd.springboot.service;

import com.klef.jfsd.springboot.model.Employee;

public interface EmployeeService
{

    public String addemployee(Employee emp);
}
```



```java
package com.klef.jfsd.springboot.service;

import org.springframework.beans.factory.annotation.Autowired;

@Service
public class EmployeeServiceImpl implements EmployeeService
{

    @Autowired
    private EmployeeRepository employeeRepository;

    @Override
    public String addemployee(Employee emp)
    {

        employeeRepository.save(emp);
        return "Employee Added Successfully...!!!";
    }

}

```



```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Employee Registration</title>
</head>
<body>

<h2>Employee Registration</h2>

<form action="insertemp" method="post">
    <label for="name">Name:</label>
    <input type="text" id="name" name="name" required><br><br>

    <label for="gender">Gender:</label>
    <input type="radio" id="male" name="gender" value="Male"> Male
    <input type="radio" id="female" name="gender" value="Female"> Female<br><br>

    <label for="email">Email:</label>
    <input type="email" id="email" name="email" required><br><br>

    <label for="password">Password:</label>
    <input type="password" id="password" name="password" required><br><br>

    <label for="location">Location:</label>
    <input type="text" id="location" name="location" required><br><br>

    <label for="contact">Contact:</label>
    <input type="text" id="contact" name="contact" required><br><br>

    <input type="submit" value="Register">
</form>

</body>
</html>

```

For reference visit

https://www.tothenew.com/blog/step-by-step-guide-sending-emails-in-spring-boot/

Surya

J SURYA KIRAN