Bulk Gmail Sender

OVERVIEW

A Spring Boot API that:

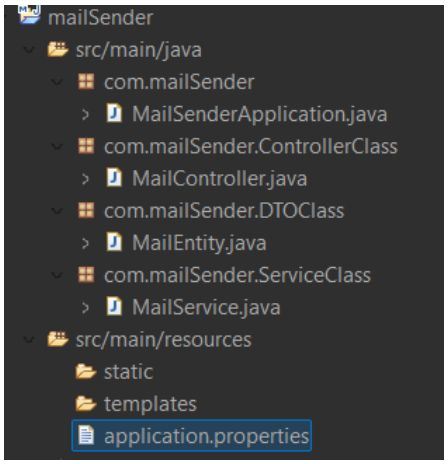
- Accepts POST requests with recipient emails, subject, content, and optional images.
- Sends personalized or bulk emails using Gmail's SMTP server.(Gmail's SMTP server address is smtp.gmail.com. It's used to send emails from your Gmail account using other email clients or applications. Gmail supports both SSL (port 465) and TLS (port 587) for secure connections)
- Optionally supports attachments or inline images.
- Handles exceptions and provides status responses.

📁 1. Create the Spring Boot Project

Use Spring Initializr with:

- Dependencies:
    - Spring Web
    - Spring Boot DevTools
    - Spring Mail
    - Lombok (optional)
    - Validation

2.Spring Boot Project Structure



2.Dto class

```java
@Getter
@Setter
public class MailEntity {

    private List<String> recipients;
    private String subject;
    private String content;
    private byte[] image;


}
```

3.Service Class

```java
@Service
public class MailService {

    @Autowired
    private JavaMailSender mailSender;

    public void sendMail(MailEntity mail) throws MessagingException
    {
        for(String recipient:mail.getRecipients())
        {
            MimeMessage message = mailSender.createMimeMessage();
            MimeMessageHelper helper = new MimeMessageHelper(message,true);
            helper.setTo(recipient);
            helper.setSubject(mail.getSubject());
            helper.setText(mail.getContent());

            if(mail.getImage()!=null & mail.getImage().length>0)
            {
                helper.addInline("inlineImage", new ByteArrayResource(mail.getImage()),"image/png");
            }

            mailSender.send(message);
        }
    }
}
```

4.controller

```java
@RestController
@RequestMapping("/mailsender")
public class MailController {

    @Autowired
    private MailService mailService;

    @PostMapping("/send")
    public ResponseEntity<String> sendMails(@RequestBody MailEntity emails)
    {
        try {
            mailService.sendMail(emails);
            return ResponseEntity.ok("sent successful");
        }catch (Exception e)
        {
            return ResponseEntity.status(500).body("internal error"+e.getMessage());
        }
    }
}
```

5.Application.property

```
spring.application.name=mailSender
spring.mail.host=smtp.gmail.com
spring.mail.port=587
spring.mail.username=your-email@gmail.com
spring.mail.password=your-app-password
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.starttls.enable=true
```

6.Testing Part