Task 2

Write a script to get a list of your last 200 emails(Gmail) and display the list of sender and subject of the mail. You are free to use the Gmail APIs.

Tanvi V. Chaudhari	LinkedIn
B.Tech	GitHub
CSE Branch	Email
IIT Gandhinagar	

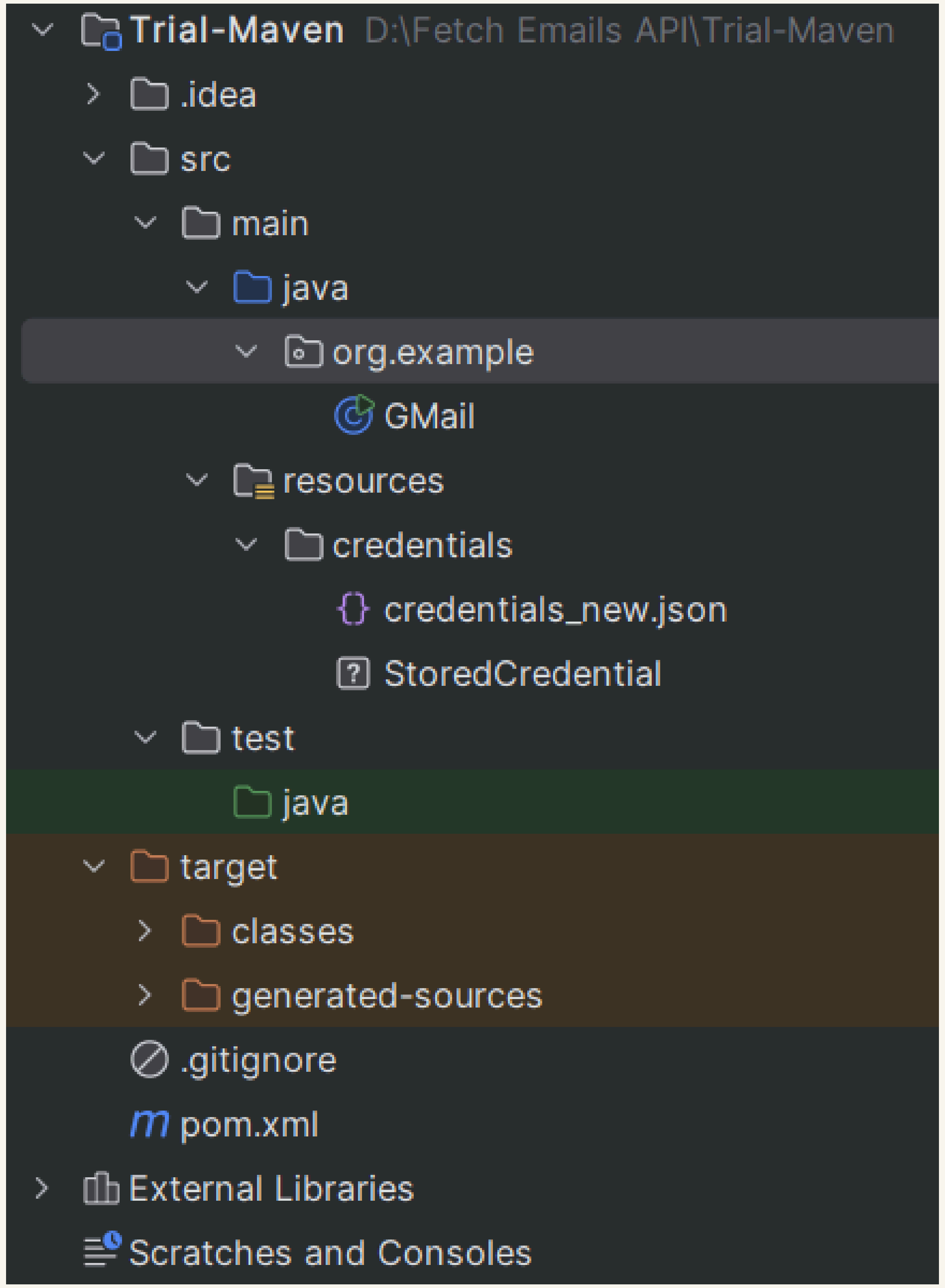
Requirements

1. Java SDK Version 17
2. Gradle Version 8
3. IntelliJ Community 2024
4. A Google Cloud Project & Enabled Gmail

4. A Google Cloud Project & Enabled Gmail

1. Create an account on Google Cloud [here](#).
2. Create a New Project.
3. Enable the API
4. Configure the OAuth consent screen
5. Authorize credentials for a desktop application
6. Save the downloaded JSON file as credentials.json, and move the file to your working directory.
7. If you go to Menu menu > APIs & Services > Credentials, you'll see your application name, in my case, it's "Fetching Emails".

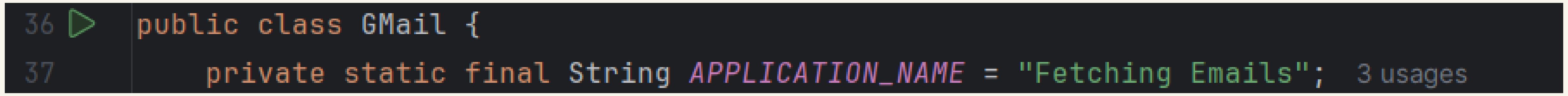
Steps to Follow



```
Trial-Maven  D:\Fetch Emails API\Trial-Maven
  .idea
  src
    main
      java
        org.example
          GMail
      resources
        credentials
          credentials_new.json
          StoredCredential
    test
      java
  target
    classes
    generated-sources
  .gitignore
  pom.xml
External Libraries
Scratches and Consoles
```

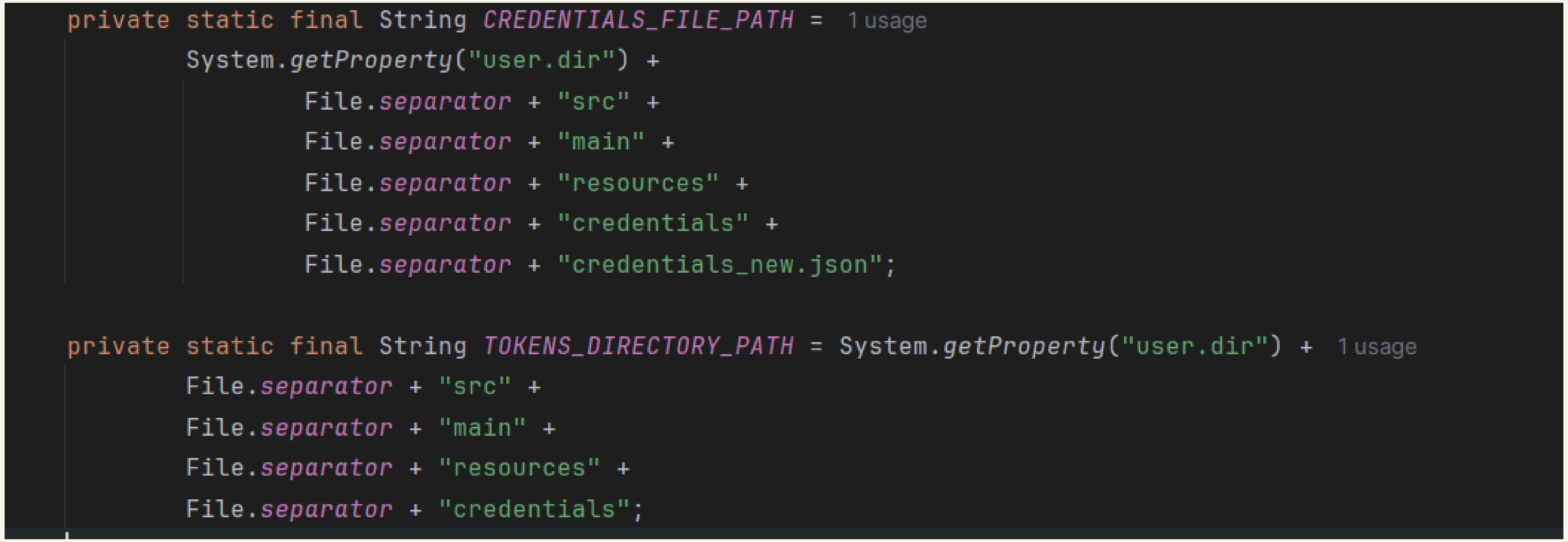
1. Create a new project on IntelliJ Community 2024 named Trial-Maven using Maven build.
2. Here, file and directory order for this project.
3. Starting by editing **GMail.java** file in **src/main/java/org.example/GMail.java**
4. In the **src/main/resources/** directory, copy the credentials.json
5. Installed various dependencies and imported numerous libraries.
6. Inspired from Java quickstart guide for writing the code.

1. Connecting with New Project on Google Cloud



```
public class GMail {
    private static final String APPLICATION_NAME = "Fetching Emails";  3 usages
```

2. Wrote credentials_file_path and tokens_directory_path

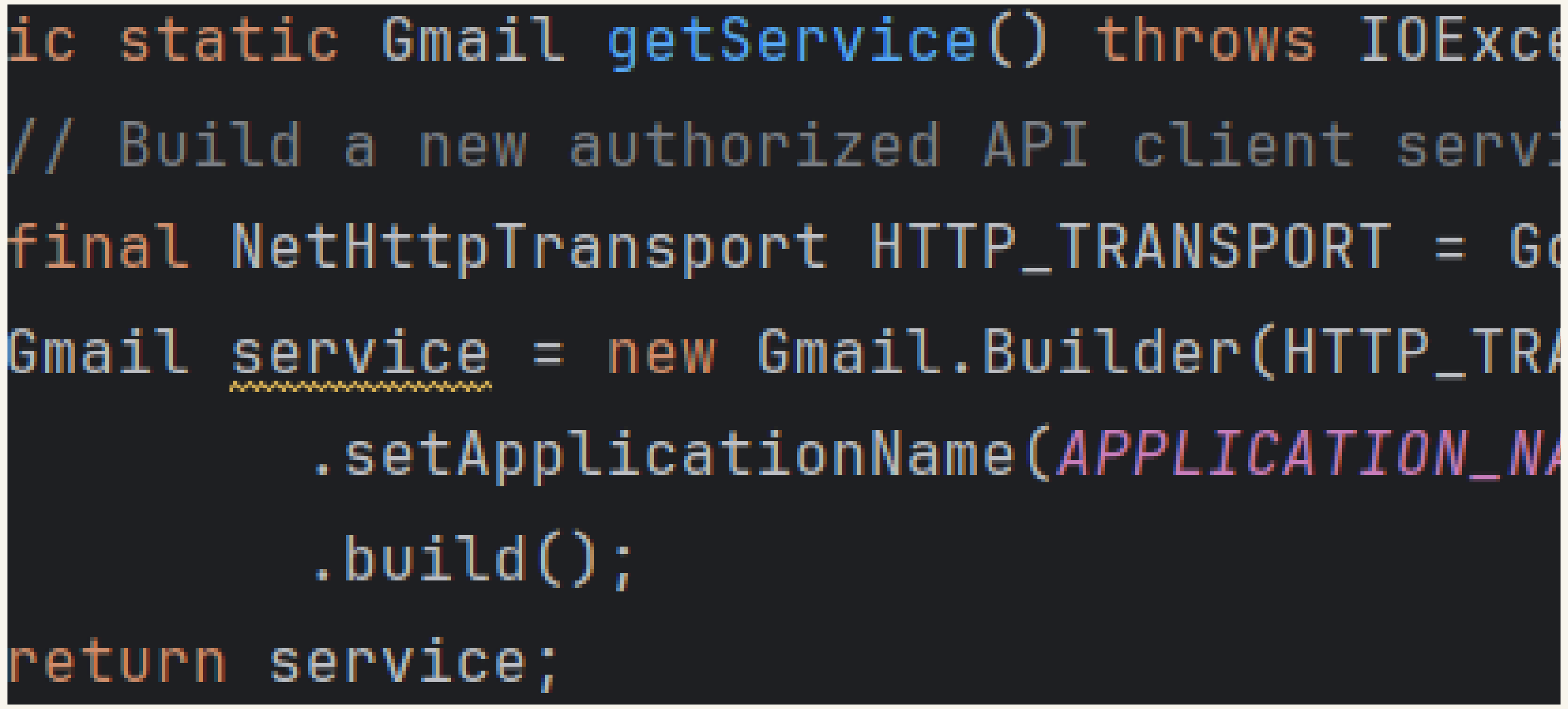


```
private static final String CREDENTIALS_FILE_PATH =  1 usage
        System.getProperty("user.dir") +
                File.separator + "src" +
                File.separator + "main" +
                File.separator + "resources" +
                File.separator + "credentials" +
                File.separator + "credentials_new.json";

private static final String TOKENS_DIRECTORY_PATH = System.getProperty("user.dir") +  1 usage
        File.separator + "src" +
        File.separator + "main" +
        File.separator + "resources" +
        File.separator + "credentials";
```

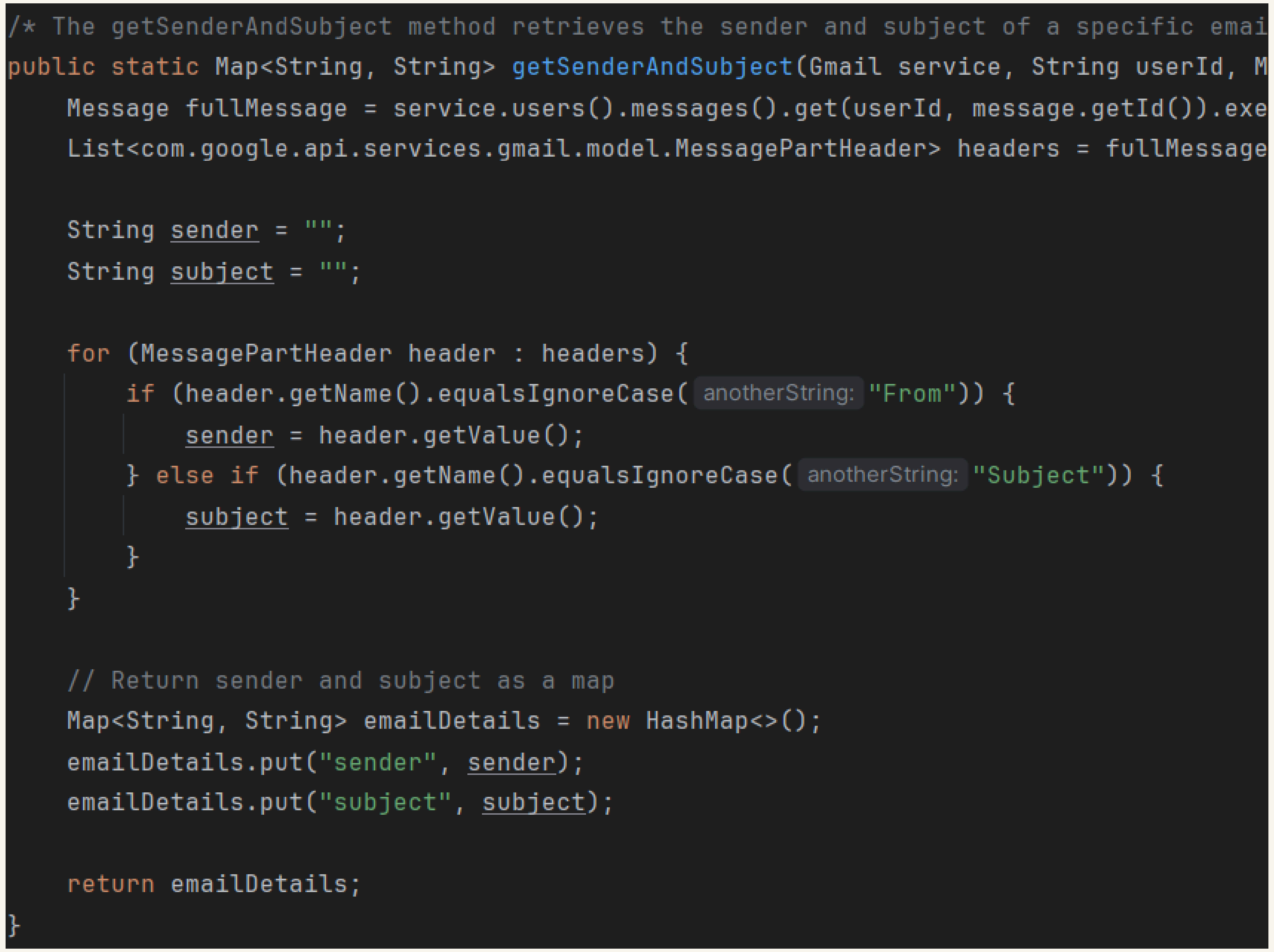
My Changes in the Source Code...

3. getService(): Returns an authorized Gmail API client that you can use to perform actions like reading, sending, or modifying emails in a user's Gmail account.



```
ic static Gmail getService() throws IOExce
// Build a new authorized API client servi
final NetHttpTransport HTTP_TRANSPORT = Go
Gmail service = new Gmail.Builder(HTTP_TRA
        .setApplicationName(APPLICATION_NA
        .build();
return service;
```

4. getSenderAndSubject(): retrieves the sender and subject of a specific email message from Gmail upto 200 latest emails.



```
/* The getSenderAndSubject method retrieves the sender and subject of a specific emai
public static Map<String, String> getSenderAndSubject(Gmail service, String userId, M
    Message fullMessage = service.users().messages().get(userId, message.getId()).exe
    List<com.google.api.services.gmail.model.MessagePartHeader> headers = fullMessage

    String sender = "";
    String subject = "";

    for (MessagePartHeader header : headers) {
        if (header.getName().equalsIgnoreCase( anotherString: "From")) {
            sender = header.getValue();
        } else if (header.getName().equalsIgnoreCase( anotherString: "Subject")) {
            subject = header.getValue();
        }
    }

    // Return sender and subject as a map
    Map<String, String> emailDetails = new HashMap<>();
    emailDetails.put("sender", sender);
    emailDetails.put("subject", subject);

    return emailDetails;
}
```

My Changes in the Source Code...

Github

1. Please find the github repository, here. (Link -> https://github.com/Tanvi-Chaudhari/Fetching-Emails)

2. I have provided requirements, steps to follow, github link, and resources in this presentation.

Resources

1. Java Quickstart Quide: https://developers.google.com/gmail/api/quickstart/java
2. Gmail OAuth 2.0 APIs: https://www.youtube.com/watch?v=PVCEPmcFA38
3. How to Configure Gmail API: https://www.youtube.com/watch?v=GtGLlay45cE