

LAPORAN PRAKTIKUM ALGORITMA PEMOGRAMAN  
COMMIT KE GITHUB DAN KODE PROGRAM  
PEKAN 1



OLEH:  
LEXI MULIA YUNASPI

(2511531006)

DOSEN PENGAMPU:  
DR. WAHYUDI, S.T, M.T

FAKULTAS TEKNOLOGI INFORMASI  
DEPARTEMEN INFORMATIKA  
UNIVERSITAS ANDALAS

2025

## KATA PENGANTAR

Puji dan syukur saya panjatkan ke hadirat Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya sehingga laporan praktikum pekan pertama mata kuliah *Algoritma dan Pemrograman* ini dapat diselesaikan dengan baik dan tepat waktu. Laporan ini disusun sebagai bentuk pertanggungjawaban atas kegiatan praktikum yang telah dilakukan, sekaligus sebagai sarana mendokumentasikan proses, hasil, serta analisis dari praktikum pekan pertama membuat akun github dan kode program. Praktikum ini bertujuan untuk melatih pemahaman mahasiswa terhadap konsep algoritma serta mengimplementasikannya dalam bentuk program sederhana menggunakan bahasa pemrograman tertentu.

Penulis menyadari bahwa laporan ini masih jauh dari sempurna. Oleh karena itu, penulis sangat terbuka terhadap kritik dan saran yang membangun demi perbaikan di masa yang akan datang. Akhir kata, saya mengucapkan terima kasih kepada dosen pengampu, asisten praktikum, serta semua pihak yang membantu dalam pelaksanaan praktikum dan penyusunan laporan ini. Semoga laporan ini dapat bermanfaat bagi pembaca dan menjadi referensi yang berguna dalam memahami materi algoritma dan pemrograman.

Padang, 28 September

Lexi Mulia Yunaspi

## **DAFTAR ISI**

KATA PENGANTAR.....	i
DAFTAR ISI .....	ii
BAB I PENDAHULUAN .....	1
1.1 Pengertian .....	1
1.2 Tujuan .....	2
BAB II PEMBAHASAN .....	2
2.1 Langkah Kerja Praktikum.....	2
BAB III PENUTUP.....	23
3.1 Kesimpulan.....	23

# **BAB I**

## **PENDAHULUAN**

### **A. Pendahuluan**

#### **1. Pengertian GitHub**

GitHub merupakan tempat untuk menyiapkan web bersama dalam mengembangkan proyek perangkat lunak yang memanfaatkan sistem kendali versi Git dan layanan hosting secara daring. Hal ini memiliki banyak untung untuk komputer karena memberikan kontrol akses dan beberapa fitur kolaborasi.

#### **2. Pemrograman dalam Java**

Program adalah suatu cara dalam membuat satu satu atau menghubungkan lebih satu algoritma. Dalam program, diperlukan bahasa pemrograman agar kita sebagai manusia dapat membuat instruksi kepada komputer sehingga program tersebut tereksekusi dengan benar.

Bahasa pemrograman adalah seperangkat aturan sistematis yang digunakan untuk menggambarkan perhitungan dalam format yang dapat diedit manusia. Pada praktikum kali ini, penulis menggunakan bahasa pemrograman Java sebagai bahasa pemrograman utama hingga akhir perkuliahan.

### **B. Tujuan**

Tujuan dilakukannya praktikum ini adalah sebagai berikut:

1. Memahami cara membuat akun GitHub dan membuat repository GitHub
2. Memahami cara mengoneksikan Eclipse ke repository GitHub
3. Memahami cara membuat kode di Eclipse dan menjalankannya

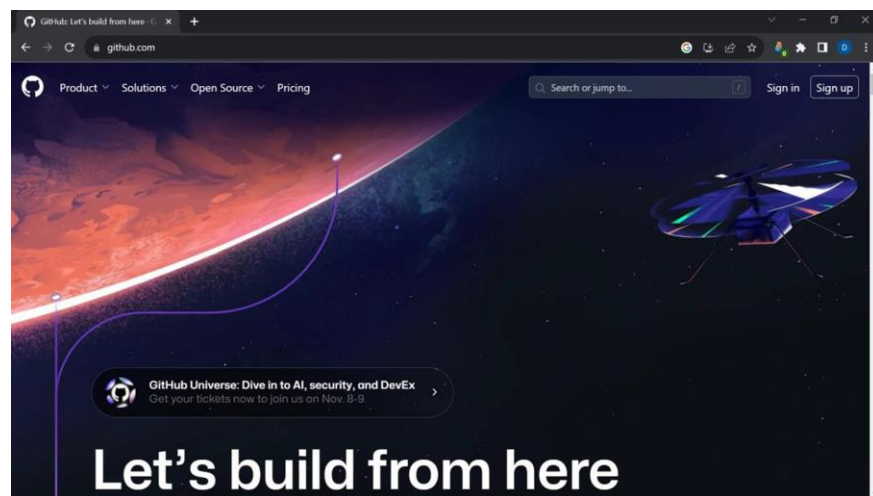
## BAB II

### PEMBAHASAN

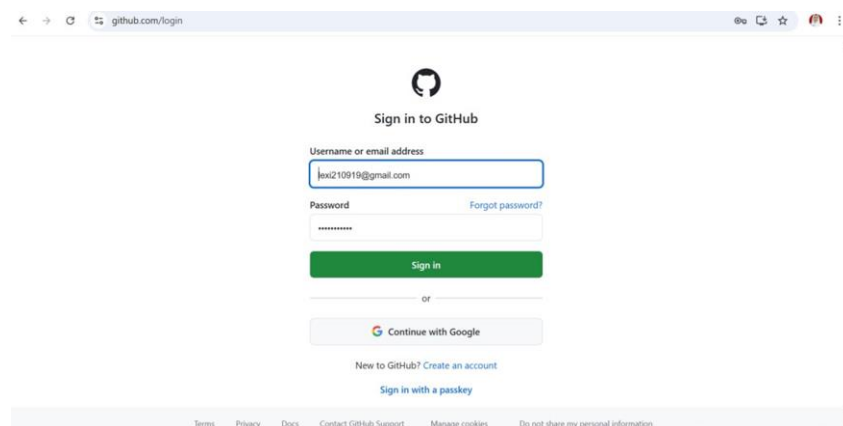
#### A.Langkah Kerja Praktikum

a. Membuat akun GitHub dan membuat Repository

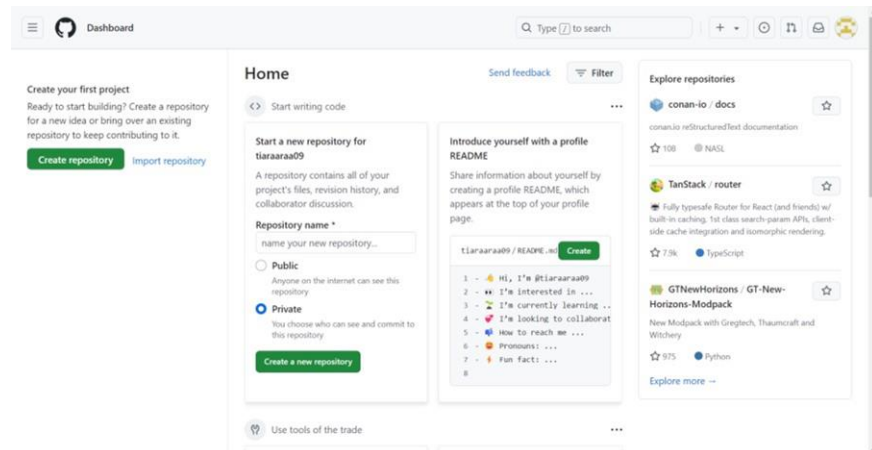
1. Buka laman github.com, klik tombol “Sign Up”. Jika sudah memiliki akun, klik tombol “Sign In”.



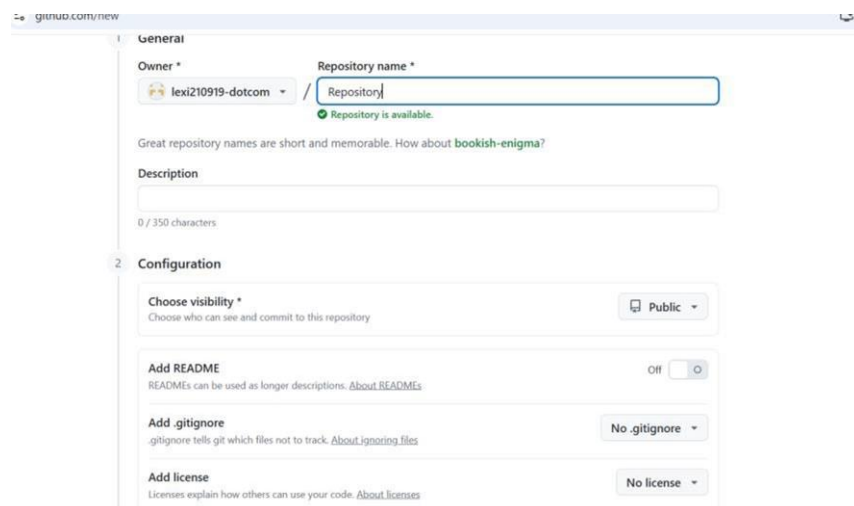
2. Kemudian masukkan email, password, dan nama profile.



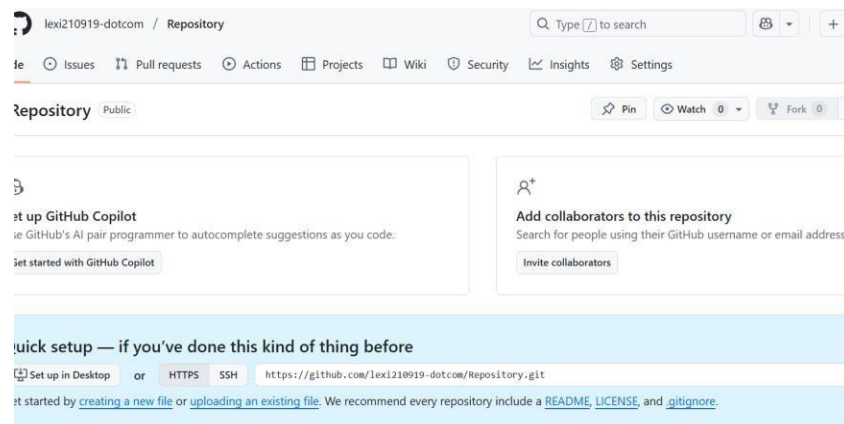
3. Setelah itu, ada beberapa pertanyaan agar akun dipersonalisasi. Setelah itu, tampilan akun akan menjadi seperti berikut. Lanjutkan dengan membuat repository dengan menekan tombol “Create Repository” atau “new”



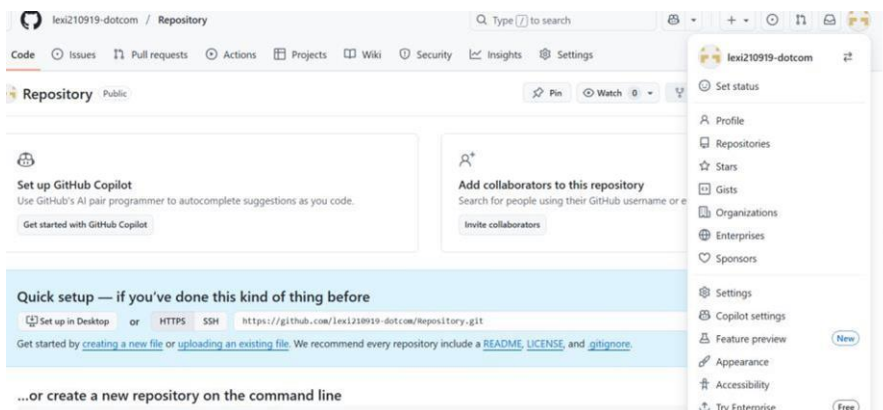
4. Setelah itu pengisian nama repository, deskripsi repository, dan pilih apakah projekmu dapat dilihat secara publik atau hanya undangan. Kemudian klik “Create Repository” di bawah



5. Setelah dibuat, tampilan akan seperti berikut. Simpan link pada “Quick setup” karna akan digunakan nanti.



6. Langkah selanjutnya yaitu membuat token untuk menghubungkan Eclipse ke GitHub. klik logo profil, lalu pilih “Setting”



7. Setelah itu, scroll hingga paling bawah sampai menemukan menu “Developer option”, Klik pada menu tersebut.

Security
Code security
Integrations
Applications
Scheduled reminders
Archives
Security log
Sponsorship log
Developer settings

Link to social profile

Link to social profile

Link to social profile

Company

You can @mention your company's GitHub organization to link it.

Location

☐ Display current local time  
Other users will see the time difference from their local time.

All of the fields on this page are optional and can be deleted at any time, and by filling them out, you're giving us consent to share this data wherever your user profile appears. Please see our [privacy statement](#) to learn more about how we use this information.

Update profile

Contributions & activity

☐ Make profile private and hide activity  
Enabling this will hide your contributions and activity from your GitHub profile and from social features like followers, stars, feeds, leaderboards and releases.

☐ Include private contributions on my profile  
Your contribution graph, achievements, and activity overview will show your private contributions without revealing any repository or organization information. [Read](#)

8. Setelah itu, pilih menu “Personal Access Token”, lalu ke “Token (Classic)”, lalu “Generate new token”

Settings / Developer Settings
Type to search
+

GitHub Apps
OAuth Apps
Personal access tokens

No GitHub Apps

Want to build something that integrates with and extends GitHub? Register a new GitHub App to get started developing on the GitHub API.

New GitHub App
View documentation

9. Setelah itu, isi note lalu tulis fungsi tokennya dibuat. Expiration date menyesuaikan. Lalu untuk pilihan yang di checklist, pilihan



wajib yaitu bagian “Repo”, selebihnya opsional.

GitHub Apps

OAuth Apps

Personal access tokens

Fine-grained tokens

Tokens (classic)

### New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

**Note**

Tugas Alpro

What's this token for?

**Expiration \***

30 days The token will expire on Wed, Nov 6 2024

**Select scopes**

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo:deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input checked="" type="checkbox"/> workflow	Update GitHub Action workflows
<input type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> read:packages	Download packages from GitHub Package Registry

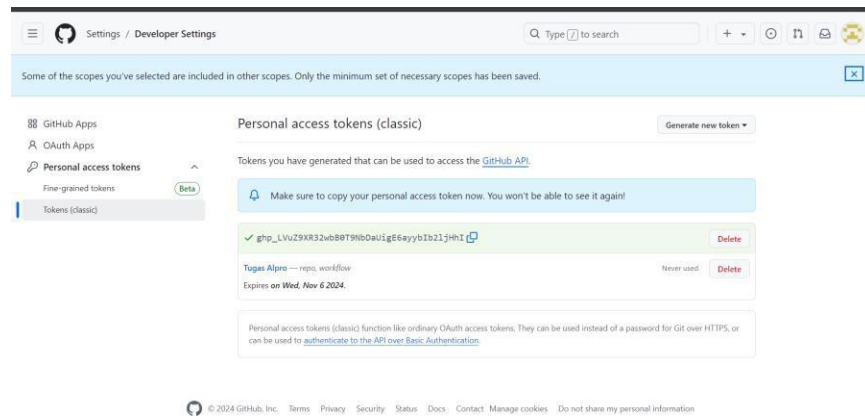
10. Scroll ke bawah untuk mengklik “genereta token”. Untuk menyimpan dan menyiapkan token

<input checked="" type="checkbox"/> scim:enterprise	Provisioning of users and groups via SCIM
<input type="checkbox"/> audit_log	Full control of audit log
<input type="checkbox"/> read:audit_log	Read access of audit log
<input type="checkbox"/> codespace	Full control of codespaces
<input type="checkbox"/> codespace:secrets	Ability to create, read, update, and delete codespace secrets
<input type="checkbox"/> copilot	Full control of GitHub Copilot settings and seat assignments
<input type="checkbox"/> manage_billing:copilot	View and edit Copilot Business seat assignments
<input type="checkbox"/> project	Full control of projects
<input type="checkbox"/> read:project	Read access of projects
<input type="checkbox"/> admin:gpg_key	Full control of public user GPG keys
<input type="checkbox"/> write:gpg_key	Write public user GPG keys
<input type="checkbox"/> read:gpg_key	Read public user GPG keys
<input type="checkbox"/> admin:ssh_signing_key	Full control of public user SSH signing keys
<input type="checkbox"/> write:ssh_signing_key	Write public user SSH signing keys
<input type="checkbox"/> read:ssh_signing_key	Read public user SSH signing keys

[Generate token](#) [Cancel](#)

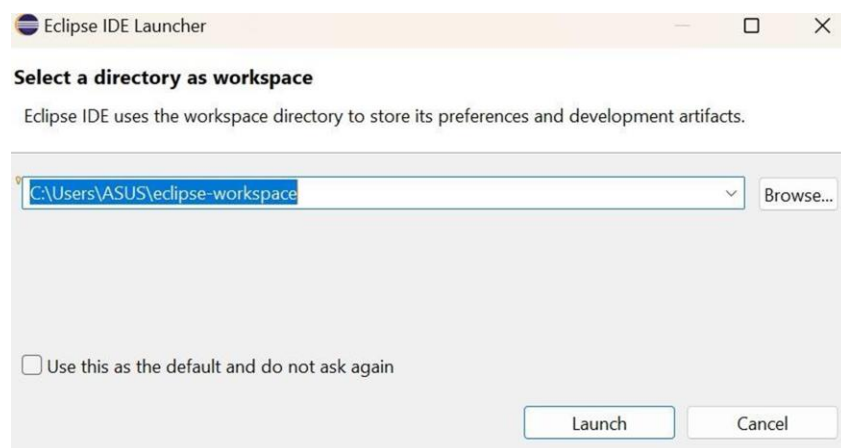
© 2024 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Docs](#) [Contact](#) [Manage cookies](#) [Do not share my personal information](#)

11. Kemudian tampilan akan seperti berikut. Simpan tokennya karena akan digunakan.

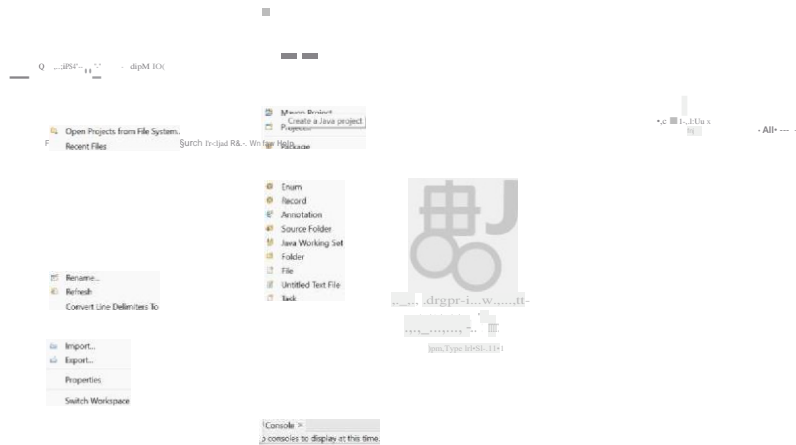


## b. Membuat Repository Lokal di Eclipse

1. Langkah pertama, buatlah workspace terlebih dahulu, lalu Launch.



2. Langkah selanjutnya yaitu buatlah Java project baru, caranya dengan klik menu “File” di pojok kiri atas, lalu Java project. Buat nama project dengan ketentuan awalan kata harus lowercase. Setelah buat, klik “Create”



## New Java Project

D X

### Create a Java Project

Discourage module name. By convention, module names usually start with a lowercase letter

Project name: Pratikum\_Alpro

Use default location

Location: C:\Users\ASUS\workspace\Pratikum\_Alpro

Browse...

Use default JRE: JavaSE-22

Use project specific JRE: jre

Use default JRE and workspace compiler preferences

[Configure JREs...](#)

Project layout

Use project folder as root for sources and class files

Create separate folders for sources and class files

[Configure default...](#)

Working sets

Add project to working sets

None

Working sets

Select...

Module

Create module-info.java file

Module name:

Generate comments

module name will be "Pratikum\_Alpro" (if no module is specified, then project name will be used as module name)

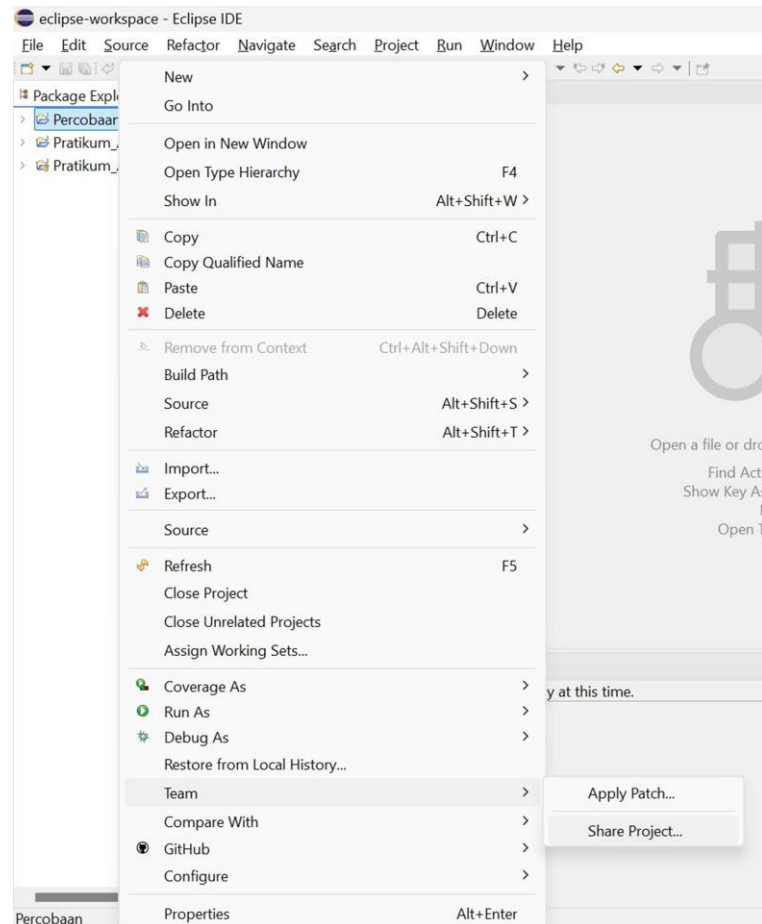
< Back

Next >

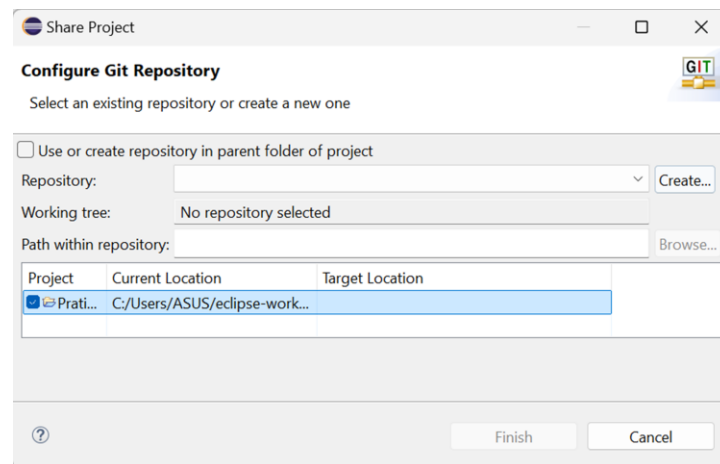
Finish

Cancel

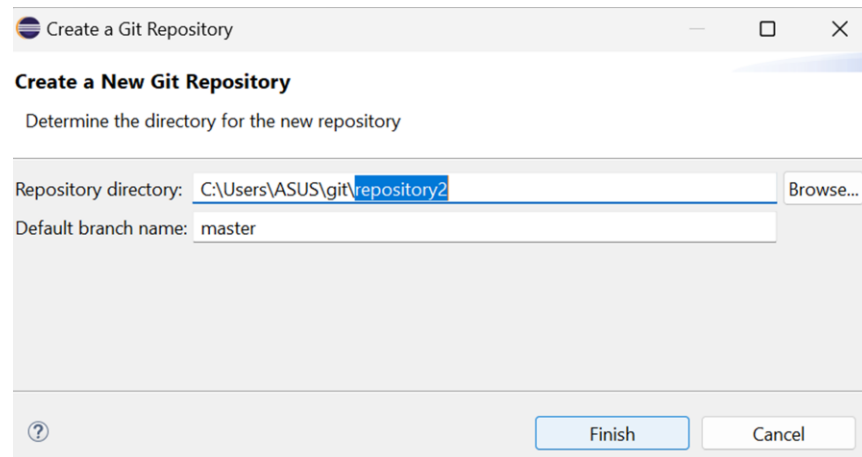
3. Lalu, klik kanan pada nama project yang telah dibuat, lalu ke bagian “Team”, lalu klik “Share Project”



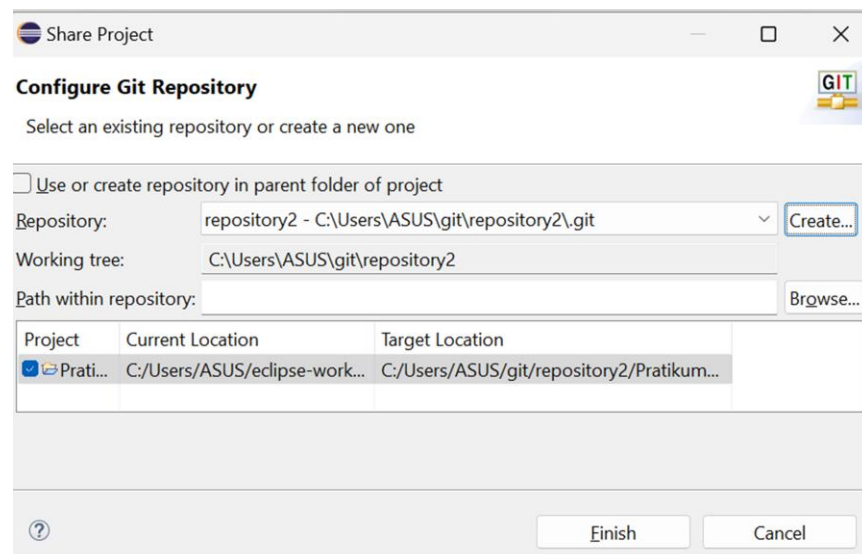
4. Kemudian klik pada menu “Create”



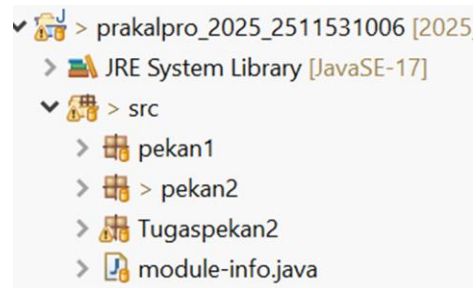
5. Buat nama repository lokal. Untuk default branch name nya dibiarkan.  
Lalu, klik “Finish”



6. Setelah itu, klik “Finish” Sekali lagi

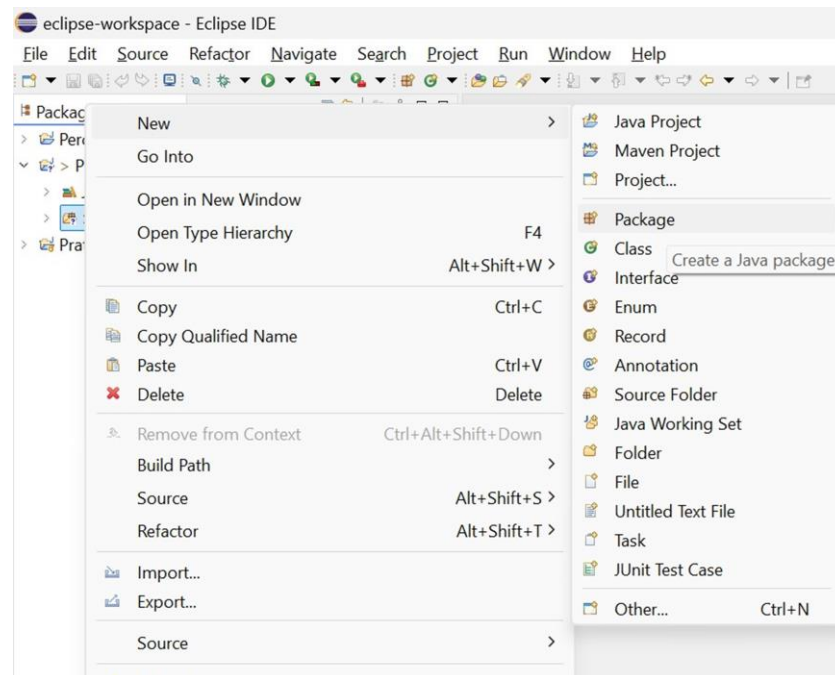


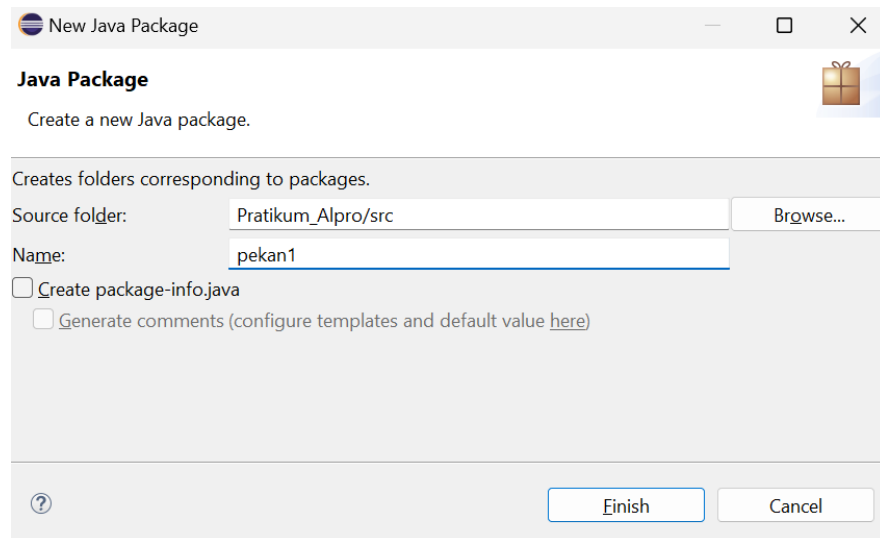
7. Maka muncul repository yang telah dibuat pada samping kiri project



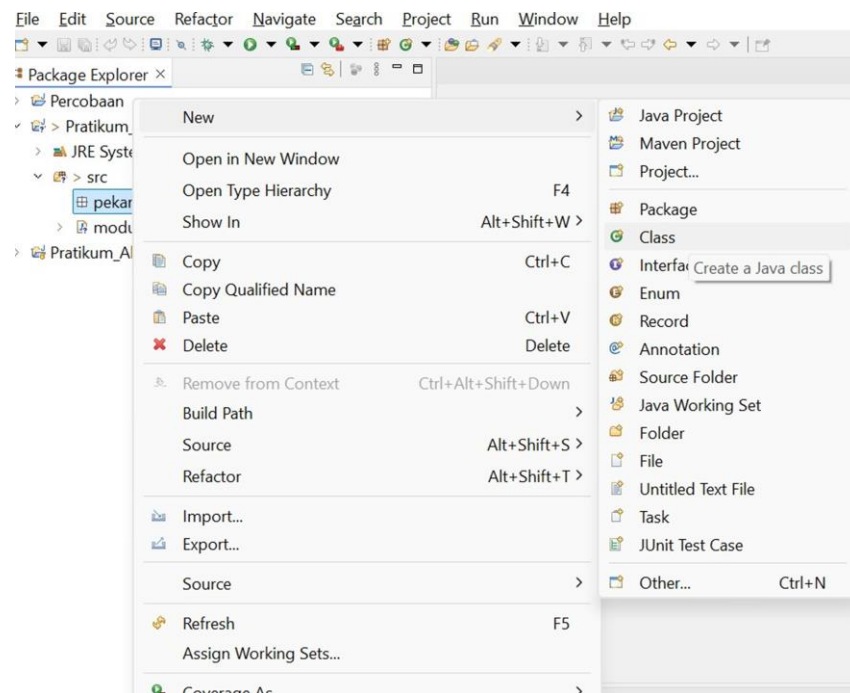
c. Program Pertama

1. Buatlah package terlebih dahulu dengan mengklik kanan di folder src. Setelah itu beri nama pada package tanpa huruf kapital, karakter khusus serta tanpa “space”.





1. Setelah itu pilih “New”, lalu pilih class. Buat nama dengan ketentuan nama harus Uppercase pada awal kalimat dan tanpa “space”, lalu centang tanda “public static void main (string[] args)”



**New Java Class**

**Java Class**  
Create a new Java class.

Source folder:

Package:

☐ Enclosing type:

Name:

Modifiers: ☒ public ☐ package ☐ private ☐ protected  
☐ abstract ☐ final ☐ static  
☒ none ☐ sealed ☐ non-sealed ☐ final

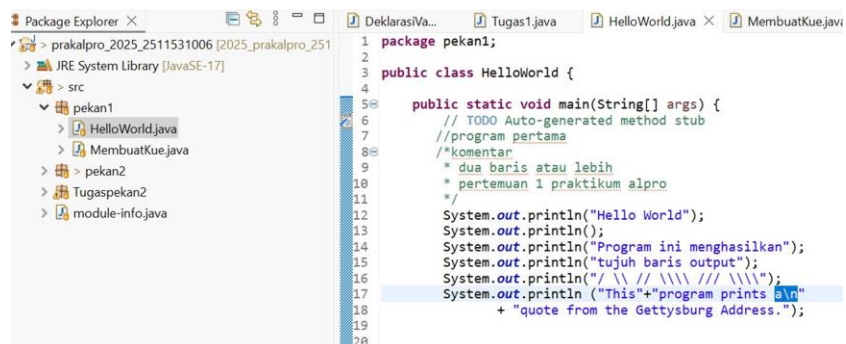
Superclass:

Interfaces:

Which method stubs would you like to create?  
☒ public static void main(String[] args)  
☐ Constructors from superclass  
☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))  
☐ Generate comments

## 2. Maka Tampilan akan seperti berikut



```

1 package pekan1;
2
3 public class HelloWorld {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         //program pertama
8         /*komentar
9          * dua baris atau lebih
10        * pertemuan 1 praktikum alpro
11        */
12        System.out.println("Hello World");
13        System.out.println();
14        System.out.println("Program ini menghasilkan");
15        System.out.println("tujuh baris output");
16        System.out.println("/ \\ // \\ \\ /// \\ \\");
17        System.out.println ("This"+ "program prints "+
18            + "quote from the Gettysburg Address.");
19
20    }
21 }
  
```

## 3. Lalu, masukkan syntax sebagai berikut:

System.out.println("Hello, World!");

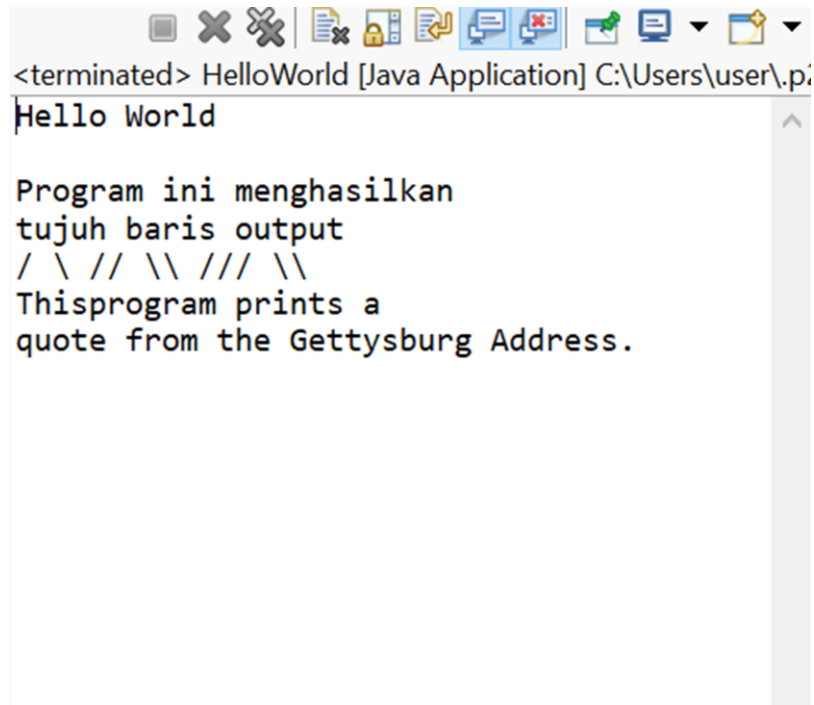


```

/
System.out.println("Hello World");
System.out.println();
System.out.println("Program ini menghasilkan");
System.out.println("tujuh baris output");
System.out.println("/ \ \ // \\\ \\\ \\\");
System.out.println ("This"+"program prints a\n"
+ "quote from the Gettysburg Address.");

```

4. Jalankan dengan mengklik bulatan hijau di bar menu (Run)



#### d.Program Dua

1. Klik kanan di package. Pilih "New" ambil class baru. Beri nama dengan ketentuan nama harus Uppercase pada awal kalimat dan tanpa "space", lalu centang tanda "public static void main (string[] args)"

**New Java Class**

Create a new Java class.

---

Source folder:  Pratikum Alpro/src Browse...

Package:  pekan1 Browse...

Enclosing type:  Browse...

---

Name:  Membuat Kue

Modifiers:

<input checked="" type="radio"/> public	<input type="radio"/> package	<input type="radio"/> private	<input type="radio"/> protected
<input type="radio"/> abstract	<input type="radio"/> final	<input type="radio"/> static	
<input checked="" type="radio"/> none	<input type="radio"/> sealed	<input type="radio"/> non-sealed	<input type="radio"/> final

Superclass:  java.lang.Object Browse...

Interfaces:  Add ... Remove

Which methods would you like to create?

- ☒ public static void main(String[] args)
- ☐ Constructors from superclass
- ☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value)

☐ Generate comments

2. Maka selanjutnya akan muncul tampilan seperti berikut ini

```
1 package pekan1;
2
3 public class MembuatKue {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7     }
8 }
9
10 }
```

3. Tuliskan syntax seperti berikut ini

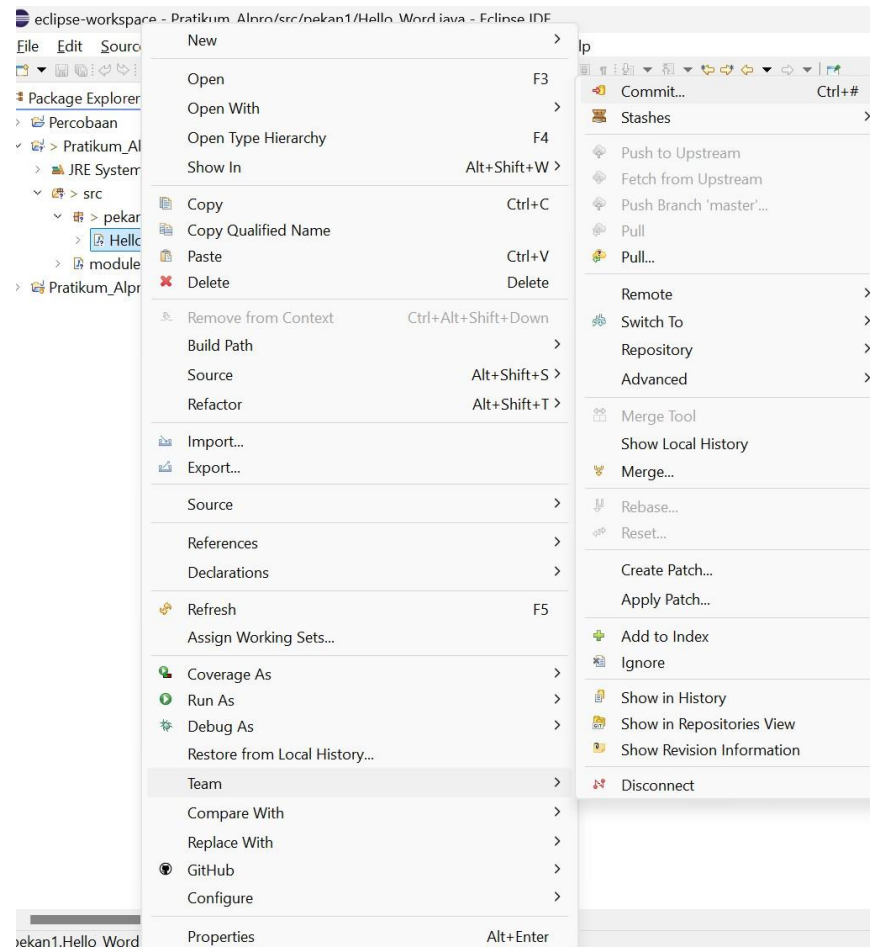
```
1 package pekan1;
2
3 public class MembuatKue {
4     public static void main (String[] args) {
5         makeBatter();
6         memanggang();
7         menghias();
8     }
9     // Langkah 1: Buat adonan kue.
10    public static void makeBatter() {
11        System.out.println("Campur bahan kering.");
12        System.out.println("Krim mentega dan gula.");
13        System.out.println("Kocok telurnya.");
14        System.out.println("Masukkan bahan kering.");
15    }
16    // Langkah 2: Panggang sekumpulan kue.
17    public static void memanggang() {
18        System.out.println("Setel suhu oven.");
19        System.out.println("Setel pengatur waktu.");
20        System.out.println("Masukkan kue ke dalam oven.");
21        System.out.println("Biarkan cookie untuk dipanggang.");
22    }
23    //Langkah 3: Hiasi cookie.
24    public static void menghias() {
25        System.out.append("Campur bahan untuk frosting.");
26        System.out.append("Taburkan frosting taburan.");
27    }
28 }
```

4. Jalankan program dengan menekan bulatan hijau di kiri atas, di bar menu

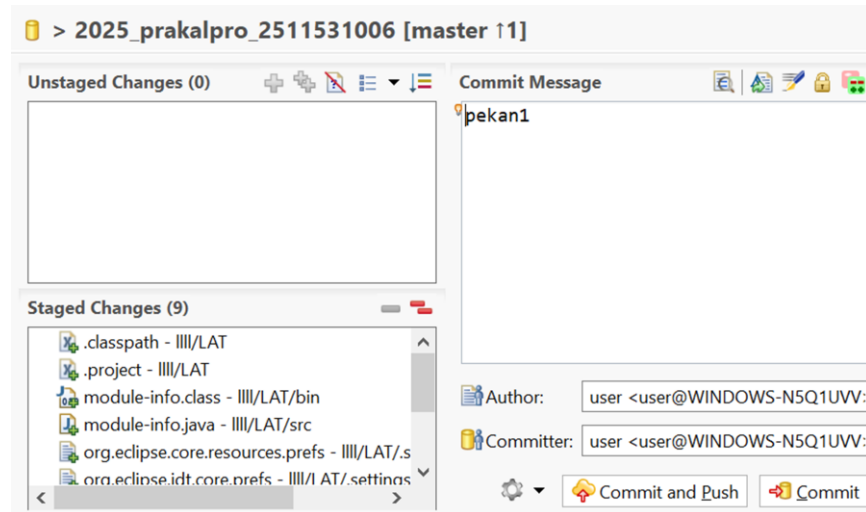
```
<terminated> MembuatKue [Java Application] C:\Users\user\
Campur bahan kering.
Krim mentega dan gula.
Kocok telurnya.
Masukkan bahan kering.
Setel suhu oven.
Setel pengatur waktu.
Masukkan kue ke dalam oven.
Biarkan cookie untuk dipanggang.
Campur bahan untuk frosting. Taburkan fros:
```

d. Menghubungkan Eclipse ke Github

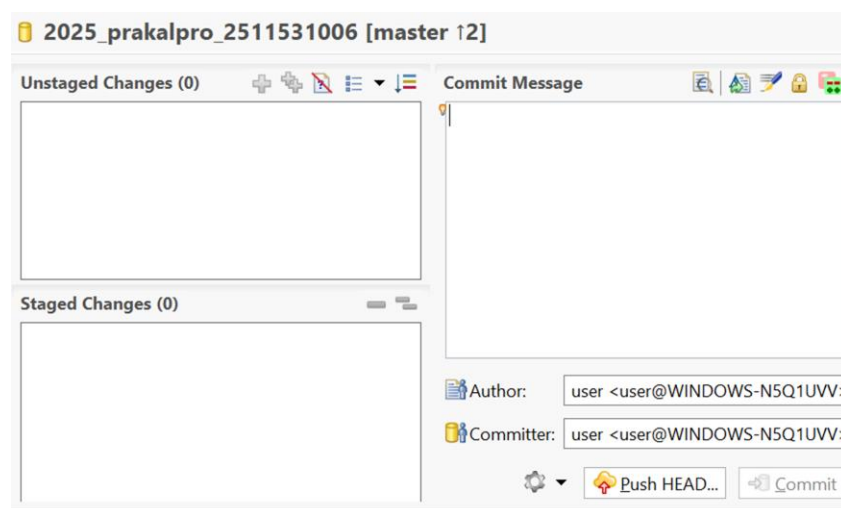
1. Klik kanan project yang telah dibuat, lalu pilih “Team”, lalu pilih “Commit”



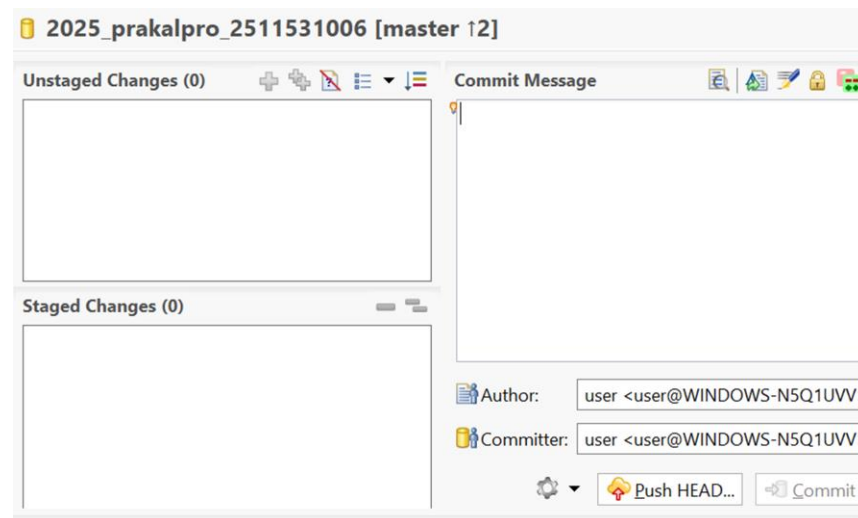
2. Block semua yang berada di atas dengan , lalu tekan logo tambah



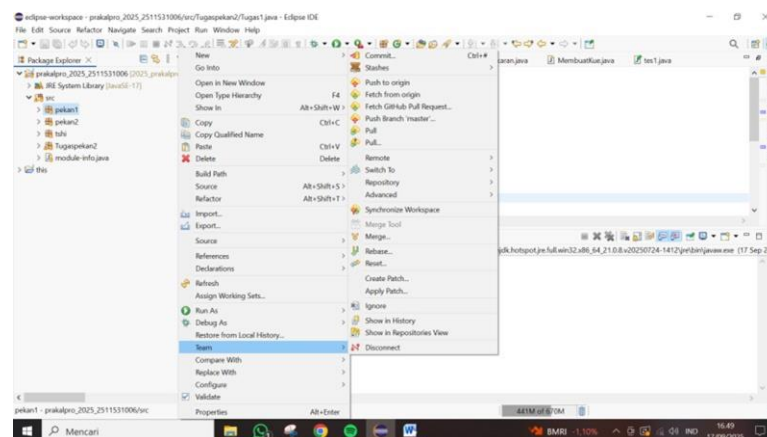
3. Setelah ditambah, buat pesan commitnya, lalu klik “Commit”



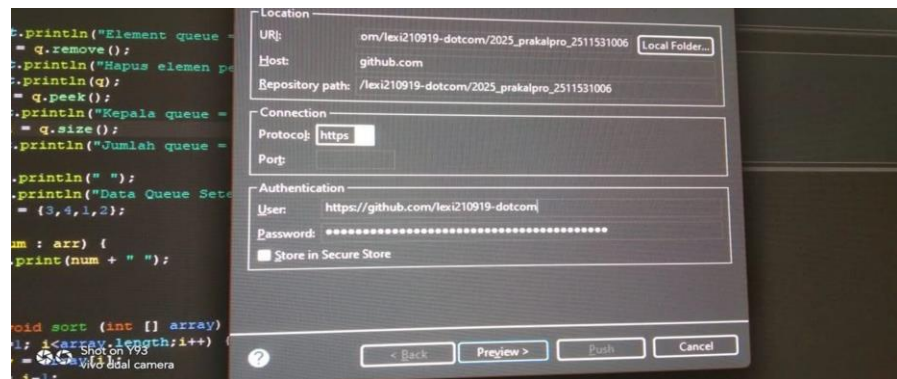
4. Jika tampilan sudah hilang, maka item berhasil di commit.



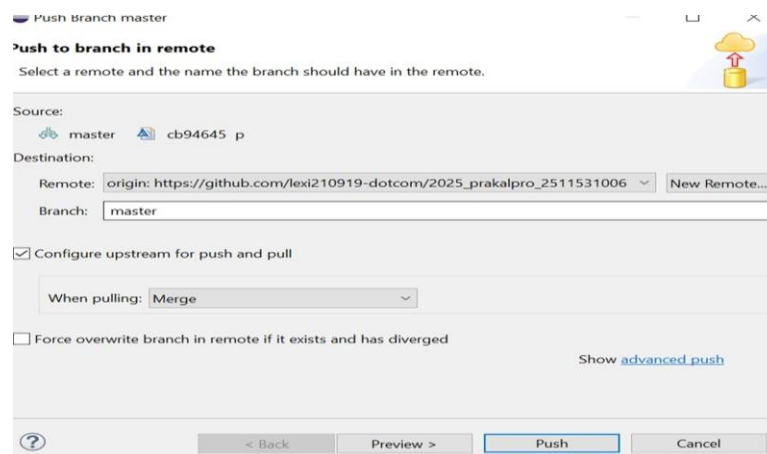
5. Kembali ke menu awal dengan cara mengklik proyek yang telah dibuat tadi, lalu klik kanan proyek, lalu pilih team, lalu pilih push branch master.



6. Masukkan link URL yang anda simpan tadi berdasarkan GitHub Repository yang telah dibuat di awal, untuk bagian user diisikan dengan nama usernam profile, dan untuk password diisikan dengan token yang telah dibuat di awal, lalu klik “Preview”



7. Setelah itu, tekan “Push”, maka akan tampil seperti berikut.



8. Maka file yang berada di Eclipse berhasil disalin di Repository GitHub. Anda bisa mengecek apakah sudah masuk di repository GitHub di akun anda

Q Go to 11N"

Addfi .. -

6b1234d - yesterday  3 Commits

ttrd ay

Repos,ton praktrkum Algotnma kelas 8

Ostars

V O forks

No,... po'id

NOI)t(9P<sup>100</sup> ,d

## Add a README

Help people interested in this repository understand your project by adding a README.





## **BAB III**

### **PENUTUP**

#### **3.1 Kesimpulan**

Berdasarkan praktikum ini, bisa disimpulkan bahwa dalam bahasa pemograman Java, ide diperlukan untuk membuat dan mengedit kode yang akan dibuat. Ide juga perlu untuk tempat mengcompile hasil codingan sehingga juga memudahkan untuk mengeksekusi code tanpa harus membuka lewat command prompt. Selain itu, repositor penting untuk mendokumentasikan code agar saat berkerja dengan tim tidak menjadi kebingungan. GitHub menyediakan repository berbasis online yang dapat memudahkan para programmer untuk berkerja dari jarak jauh dan berkerjasama dengan tim.