**LAPORAN PRAKTIKUM**

**MATA KULIAH DASAR PEMROGRAMAN**

Dosen Pengampu : Triana Fatmawati, S.T, M.T

**PERTEMUAN 2 : SISTEM VERSION CONTROL DAN KANBAR BOARD**



****

Nama : Yonanda Mayla Rusdiaty

NIM : 2341760184

Prodi : D-IV Sistem Informasi Bisnis

**JURUSAN TEKNOLOGI INFORMASI**

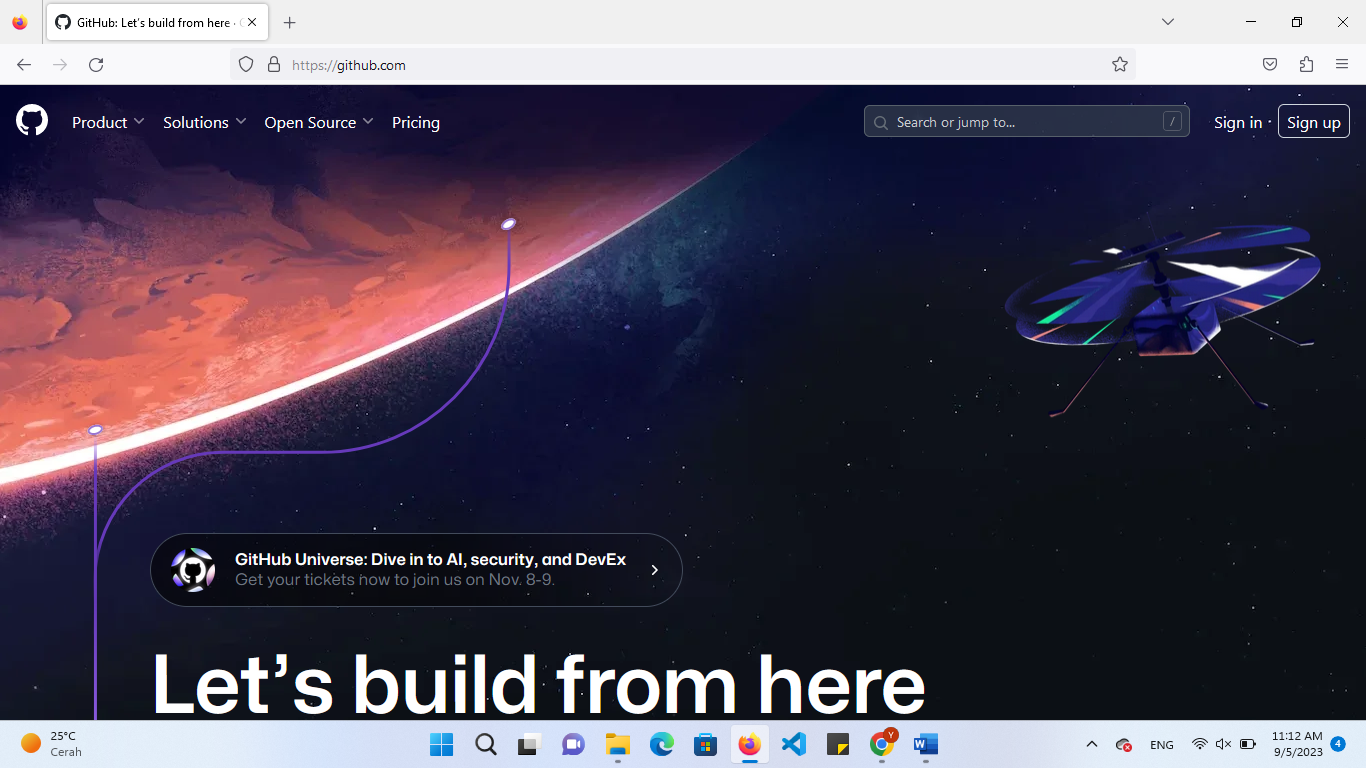
**POLITEKNIK NEGERI MALANG**

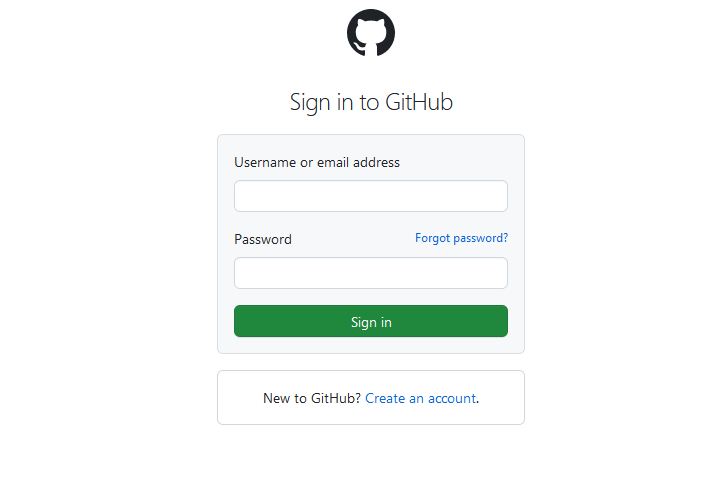
**2023**

**2.1 Percobaan 1 : Menggunakan Github**

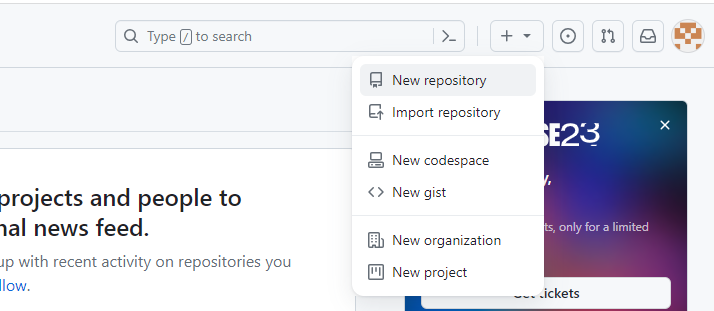
Berikut adalah hasil dari praktikum percobaan 1 : Menggunakan Github

1. Langkah pertama yaitu membuka situs <https://github.com>. Jika sudah, klik tombol **“Sign up”** untuk yang belum mempunyai akun dan membuat akun Github terlebih dahulu. Namun jika sudah mempunyai akun, langsung saja masukkan email dan password yang digunakan.

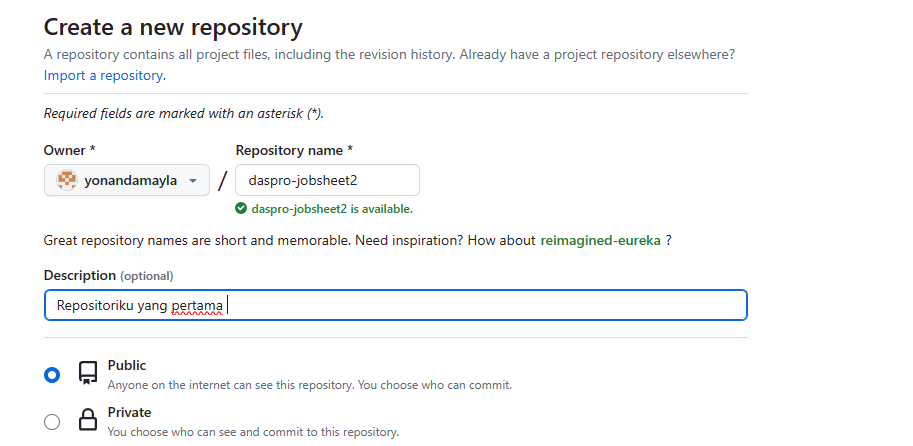




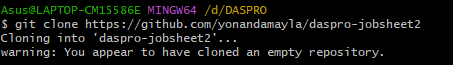
1. Setelah masuk ke akun Github masing-masing, klik tombol **“+”** yang berada di pojok kanan atas kemudian pilih menu **“New Repository”**.

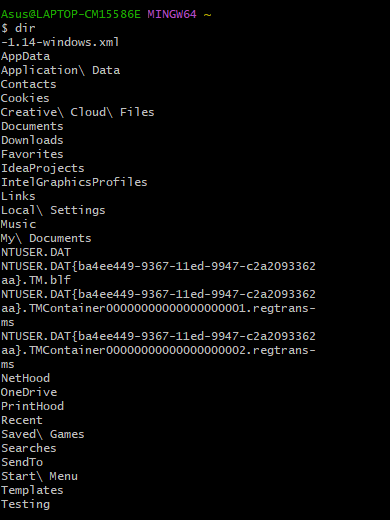


1. Isi nama repository, deskripsi yang bersifat opsional, dan konfigurasi lainnya. Kemudian, kita dapat memilih untuk repository publik atau pribadi sesuai kebutuhan, jika sudah klik tombol **“Create Repository”.**



1. Untuk mengelola repository secara lokal, dibutuhkan installasi Git client terlebih dahulu di <https://git-.scm.com/downloads> dalulu untuk mengklonanya ke komputer. Jika sudah selesai, silahkan lakukan proses installasi.
2. Gunakan perintah Git clone dari terminal CMD (Command Line) untuk mengklonkan repository. Perintah umumnya **git clone** [**https://github.com/userame/nama-repository.git**](https://github.com/userame/nama-repository.git).







1. Selanjutnya, buka visual studio code untuk membuat atau mengedit berkas-berkas yang ada di dalam repositori sesuai kebutuhan. Pilih menu open folder dan cari folder yang diinginkan.

A screenshot of a computer

Description automatically generated

1. Tambahkan file dengan **klik kanan – New File**, dan beri nama file tersebut dengan nama **“README.md”**. Isikan file **“README.md”** seperti berikut :

A screenshot of a computer

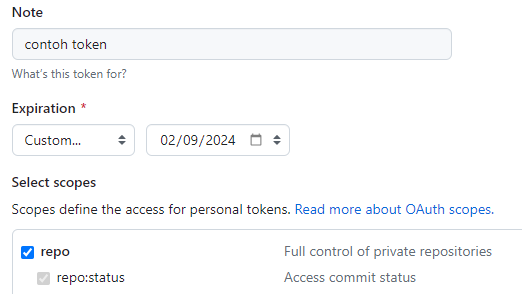
Description automatically generated

1. Jika sudah selesai, simpan perubahan pada visual studio code dan commit dengan perintah **git commit.** Setelah itu akan muncul commit yang menjelaskan perubahan yang telah dilakukan.

A computer screen with text

Description automatically generated

1. Untuk memperbarui repository di Github dengan perubahan yang dilakukan secara lokal, gunakan perintah **git push**, misalnya **git push origin nama-branch** akan mengirikmkan perubahan branch di Github.
2. Buka akun Github, klik **akun – Settings – Developer Settings – Tokens (classic) – Generate new token ( classic)**. Isikan bagian **Note, Expiration, dan Select scopes** sesuai gambar di bawah ini. jika sudah, klik tombol **Generate token**.



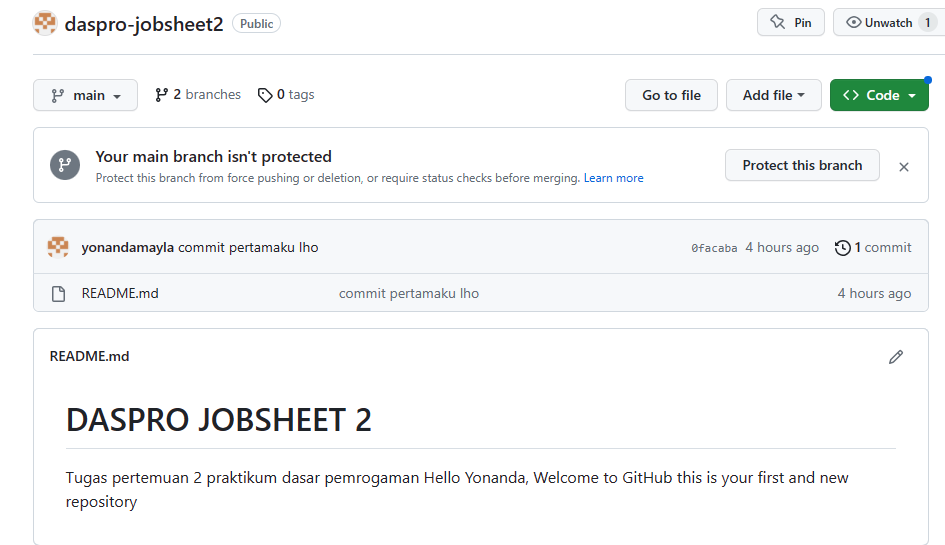
Simpan token tersebut karena tidak bisa dilihat Kembali untuk digunakan push yang selanjutnya.

1. Jalankan perintah git push.

A computer screen shot of a program

Description automatically generated

1. Cek halamannya Github kita dan tampilannya akan seperti ini



**Pertanyaan**

1. Jelaskan perbedaan perintah git commit dan git push?

2. Apakah bisa alurnya dibalik, membuat folder atau projek terlebih dahulu kemudian upload (push) ke Github? Jika bisa, buktikan!

**Jawaban :**

1. Perbedaan perintah antara git commit dengan git push yaitu :

**Git commit** adalah perintah yang digunakan untuk menyimpan perubahan yang telah dibuat dalam repositori. Hal ni membuat snapshot (commit) baru dari kode kita dengan pesan komit yang menjelaskan perubahan yang dilakukan.

Perintah git commit hanya berlaku di dalam repositori lokal dan tidak akan memengaruhi repositori jarak jauh (remote repository) seperti GitHub. Hal ini merupakan langkah penting dalam proses pengembangan karena memungkinkan kita untuk mengelola sejarah perubahan kode kita di repositori lokal.

**Git push** adalah perintah yang digunakan untuk mengirimkan (mendorong) perubahan yang telah dikomit di repositori lokal ke repositori jarak jauh, seperti GitHub. Hal ini adalah tindakan yang diperlukan jika kita ingin berbagi atau menggabungkan perubahan kode dengan tim atau mengamankan kode di repositori yang dapat diakses secara online.

Git push hanya berfungsi setelah kita melakukan git commit, sehingga kode yang telah dikomit dapat disinkronkan dengan repositori jarak jauh.

1. Membuat folder atau proyek terlebih dahulu kemudian mengunggahnya (push) ke GitHub adalah memungkinkan. Langkah-langkahnya adalah sebagai berikut :
2. Buat folder atau proyek lokal di komputer dan inisialisasi repository Git di dalamnya (jika belum ada). Ketikkan perintah “**mkdir nama-proyek”, “cd nama-proyek”, “git init”**.

A computer screen with text on it

Description automatically generated

1. Tambahkan file-file atau direktori-direktori yang ingin di unggah ke dalam proyek. Dengan terlebih dahulu memasukkan folder nama-proyek yang sudah dibuat ke dalam visual studio code dan membuat file baru yang diberi nama dengan **“README.md”**

A screenshot of a computer

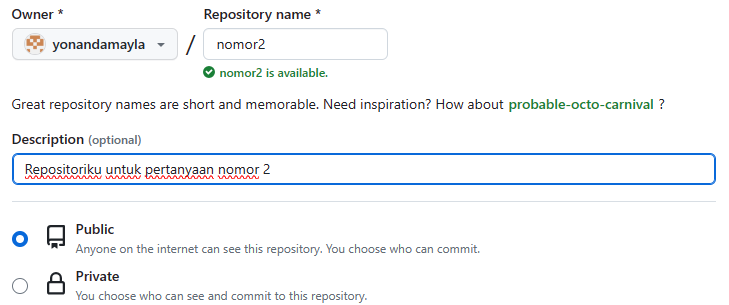
Description automatically generated

1. Buat commit untuk perubahan-perubahan. Dengan memberikan mengetikkan **git add . dan git commit -m “Pesan commit kita”**

A computer screen with text

Description automatically generated

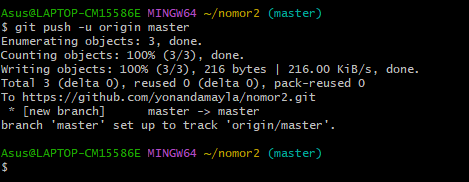
1. Buat repository kosong di Github melalui antarmuka web Github.



1. Tambahkan repository Github sebagai remote untuk repository lokal: **git remote add origin** [**https://github.com/username/nama-repositori.git**](https://github.com/username/nama-repositori.git). Ganti URL dengan dengan URL repositori Github yang sesungguhnya.



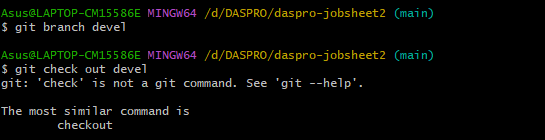
1. Push perubahan Anda ke repositori GitHub: git push -u origin master



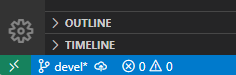
**2.2 Percobaan 2 : Dasar Kolaborasi di Github**

Berikut adalah hasil dari praktikum percobaan 2 : Dasar Kolaborasi di Github

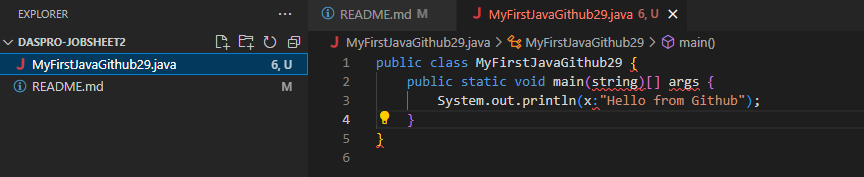
1. Gunakan perintah **git branch nama-branch** untuk membuat branch baru dan **git checkout nama-branch** untuk beralih ke branch tersebut.



Kemudian pada visual studio code seharusnya akan berganti menjadi branch **“devel”**, jika belum silahkan klik kemudian pilih branch “**devel”** .



1. Dalam folder daspro-jobsheet2, buat file baru yang diberi nama dengan **MyFirstJavaGithub29.java** (sesuai dengan nomor absen). Dan jalankan kode program menggunakan Langkah-langkah pada jobsheet 1.

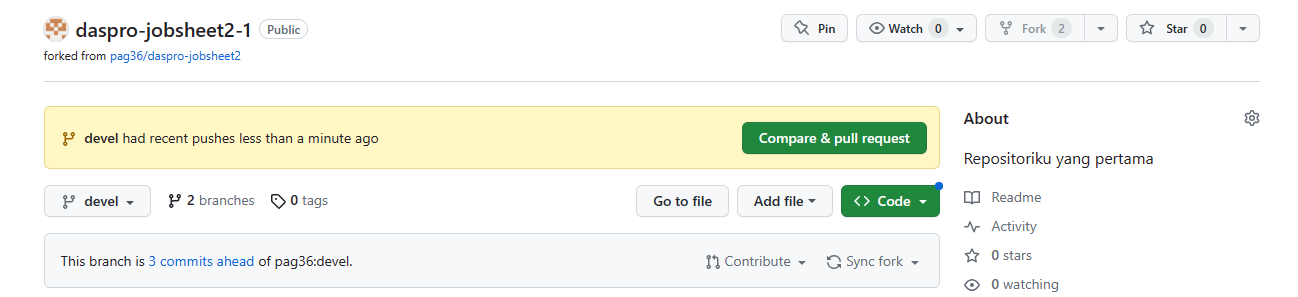


1. Simpan perubahan tersebut di local dengan cara **commit** kemudian **push** ke Github menggunakan visual studio code. Jangan lupa memberikan pesan Ketika akan melakukan **commit** . caranya adalah dengan meng **klik icon ranting – isikan pesan commit –** klik **tombol Commit – klik tombol Publish Branch.**

A screenshot of a computer

Description automatically generated

1. Buka halaman Github, seharusnya akan muncul **bracnch**  **devel**  yang bebrapa waktu telah di publish.



1. Selanjutnya, kitab isa membedakan antara **branch main** dan **branch devel.**

**A screenshot of a computer

Description automatically generated**

1. Klik tombol **Compare & pull request**, Anda dapat memilih branch mana yang akan digabungkan (devel ke master). Isikan pesan dan klik tombol **Create pull request,** tunggu beberapa saat kemudian klik tombol **Merge pull request**. Terakhir, klik tombol **Confirm merge**.

A screenshot of a chat

Description automatically generated

1. Pindah ke **tab Code,** kemudian amati hasil antara **branch main** dan **branch devel.**



**Pertanyaan!**

1. Jelaskan fungsi dari Pull requests!

2. Mengapa kita perlu membuat sebuah branch, manfaatnya apa?

**Jawaban :**

1. Fungsi dari **Pull Requests (PR)** adalah untuk memfasilitasi kolaborasi dalam pengembangan perangkat lunak, terutama dalam lingkungan pengembangan perangkat lunak berbasis Git, seperti GitHub atau GitLab. **PR** memungkinkan seorang pengembang untuk mengusulkan perubahan kode kepada proyek yang dikelola oleh orang lain atau tim. Berikut beberapa fungsi utama dari PR:
2. **Memungkinkan Review Kode.** PR memungkinkan orang lain untuk melihat, mengevaluasi, dan memberikan umpan balik terhadap perubahan kode yang diajukan. Ini membantu untuk meningkatkan kualitas kode dan menghindari kesalahan atau bug yang tidak terdeteksi.
3. **Memisahkan Pengembangan.** PR memungkinkan pengembang membuat cabang (branch) khusus untuk pekerjaan mereka. Ini memungkinkan mereka untuk bekerja secara terisolasi tanpa mengganggu kode utama dan mencegah konflik yang tidak diinginkan.
4. **Pemantauan Proses**. PR dapat digunakan untuk melacak status dan progres perubahan kode. Ini mencakup diskusi, penambahan atau perubahan kode, serta tindak lanjut terkait perubahan tersebut.
5. **Penggabungan Kode.** Setelah perubahan kode dianggap selesai dan telah melewati proses review, PR dapat digabungkan (merged) ke dalam cabang utama (biasanya disebut sebagai "master" atau "main"). Ini memperbarui kode utama dengan perubahan yang diajukan.
6. Membuat branch adalah praktik umum dalam pengembangan perangkat lunak berbasis Git, dan ini memiliki beberapa manfaat utama, seperti :
7. **Solasi Pekerjaan.** Branch memungkinkan pengembang untuk bekerja pada fitur, perbaikan bug, atau perubahan lainnya dalam lingkungan yang terisolasi. Ini berarti perubahan yang dilakukan di branch tidak akan langsung memengaruhi kode di cabang utama (biasanya disebut sebagai "master" atau "main"). Ini membantu mencegah gangguan dan konflik dengan perubahan yang sedang dilakukan oleh orang lain.
8. **Pemantauan Perubahan.** Setiap branch memiliki riwayat perubahan yang terpisah. Ini memungkinkan pengembang untuk melacak dan memahami perubahan yang telah mereka buat, serta untuk memahami sejarah perubahan pada proyek.
9. **Kolaborasi yang Lebih Baik.** Dengan branch, beberapa pengembang dapat bekerja secara bersamaan pada berbagai fitur atau perbaikan. Mereka dapat membuat branch mereka sendiri, mengerjakan tugas mereka, dan kemudian menggabungkan perubahan mereka ke dalam cabang utama melalui PR, yang memudahkan kolaborasi dalam tim.
10. **Pengujian Terpisah.** Pengembang dapat menguji perubahan di branch mereka sendiri sebelum menggabungkannya ke dalam cabang utama. Ini memungkinkan untuk mengidentifikasi masalah atau bug sebelum perubahan tersebut memengaruhi kode utama.
11. **Rollback Mudah.** Jika ada masalah dengan perubahan yang diajukan di branch, pengembang dapat membatalkan atau menghapus branch tersebut tanpa memengaruhi kode utama.

Dengan demikian, membuat branch adalah praktik yang penting dalam pengembangan perangkat lunak yang membantu mengelola kode sumber secara efisien dan memfasilitasi kolaborasi tim.