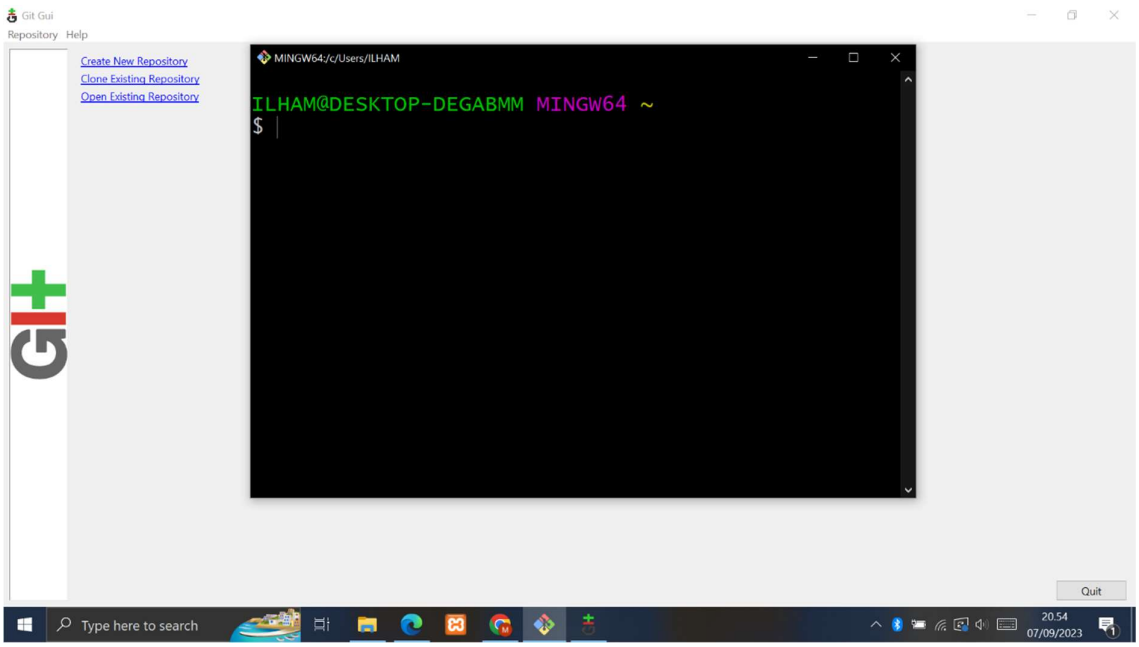
Muhammad Ilham Kurniawan – Kelompok 1

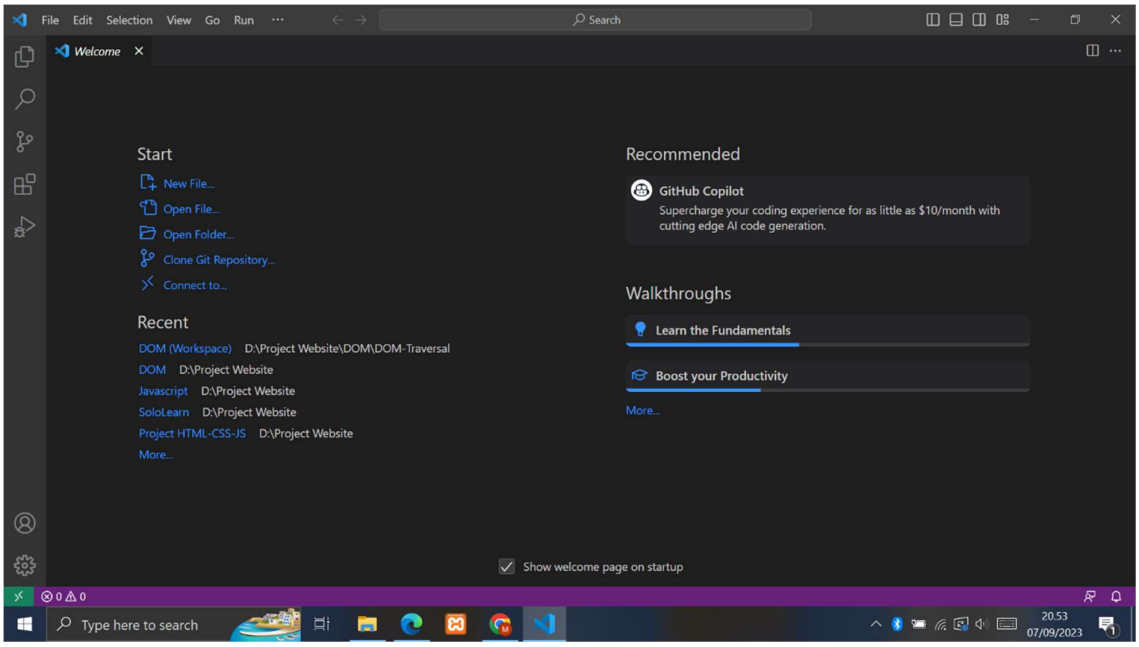
Homework 1

Instalasi Tools

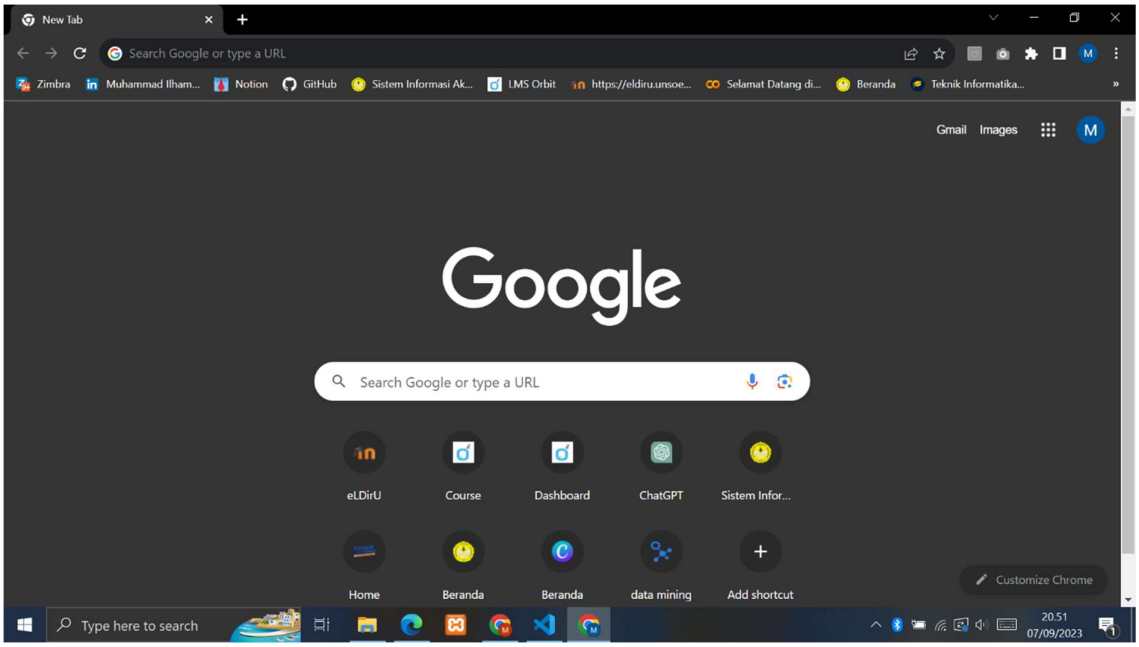
1. Git



2. Visual Studio Code



3. Browser



Summary Materi

Terdapat 3 materi pokok yang dipelajari pada minggu ke-3 ini yaitu Full Stack Developer, Career Path, SDLC & Design Thinking Implementation, serta Basic Git & Collaborating Using Git.

Materi pertama yaitu mengenai Pengembangan Full Stack (Full Stack Development) yang merujuk pada pengembangan seluruh aplikasi secara end-to-end, dari sisi depan (front-end) hingga sisi belakang (back-end) dan, dalam beberapa kasus, hingga sisi klien (client-side).

Scope penting yang ada pada Full Stack Development meliputi Front-End Development, Back-End Development, Database Management, Integration of Front-End and Back-End, Version Control and Collaboration, dan Mobile Development. Di tiap scope ini memiliki tujuan, lingkup, karateristik, serta framework yang berbeda-beda.

Pengembangan Aplikasi End to End merupakan pendekatan pengembangan perangkat lunak yang mencakup keseluruhan siklus pembuatan aplikasi, dari tahap perencanaan hingga tahap pengujian dan implementasi. Tujuannya adalah untuk menghasilkan aplikasi yang lengkap, fungsional, dan siap digunakan oleh pengguna akhir. Berikut adalah tahapan-tahapan dalam pengembangan aplikasi end-to-end.

Tahap-tahap pengembangan aplikasi end-to-end meliputi Perencanaan dan Analisis, Desain, Pengembangan Front-End, Pengembangan Back-End, Integrasi dan Pengujian, dan Pemeliharaan dan Peningkatan.

Dalam melakukan tahapan pengembangan aplikasi, kolaborasi akan menjadi efektif jika terdapat penggunaan Version Control. Version control (pengendalian versi) adalah sistem yang memungkinkan pengembang perangkat lunak untuk melacak perubahan pada kode sumber aplikasi selama pengembangan. Ini memungkinkan kolaborasi yang efisien di antara

anggota tim, terutama ketika banyak orang bekerja pada proyek yang sama. Version control populer yang sering digunakan adalah Git dan Mercurial. Manfaat dari Version Control sendiri yaitu untuk Rekam Perubahan, Pencatatan Riwayat, Pemecahan Konflik, Pemulihan Mudah

Materi yang kedua adalah SDLC (Siklus Hidup Pengembangan Perangkat Lunak) yang merupakan rangkaian proses yang terstruktur dan metodologi yang digunakan untuk mengembangkan perangkat lunak dari awal hingga selesai. SDLC terdiri dari serangkaian tahap yang saling terkait dan dilakukan secara berurutan untuk memastikan bahwa pengembangan perangkat lunak berjalan dengan baik dan sesuai dengan kebutuhan dan tujuan yang ditentukan.

Siklus pada SDLC ini secara umum meliputi Perencanaan & Analisis, Desain, Pengembangan, Pengujian, Penerapan, dan Pemeliharaan. Dengan menggunakan SDLC secara efektif, organisasi dapat meningkatkan keberhasilan dan efisiensi dalam mengembangkan aplikasi, memastikan pengiriman produk berkualitas tepat waktu, dan memberikan nilai yang lebih besar bagi pelanggan dan stakeholder

SDLC sendiri memiliki beberapa model diantaranya Waterfall Model, V-Shaped Model, Prototype Model, Spiral Model, Iterative Incremental Model, Big Bang Model, dan Agile Model. Setiap model SDLC memiliki kelebihan dan kelemahan tergantung pada jenis proyek dan kebutuhan organisasi. Pemilihan model SDLC yang tepat sangat penting untuk mencapai keberhasilan proyek pengembangan perangkat lunak.

Untuk mendukung SDLC terdapat pula yang namanya Design Thinking. Design Thinking ini memiliki beberapa tahapan diantaranya Empathize: Memahami Kebutuhan Pengguna, Define: Tentukan Masalahnya, Ideate: Hasilkan Ide, Prototype: Bangun Solusi Cepat dan Iteratif, Test: Kumpulkan Umpan Balik Pengguna, Implement: Kembangkan Perangkat Lunak.

Dengan mengintegrasikan Design Thinking dalam SDLC, tim pengembangan perangkat lunak dapat menciptakan produk yang lebih berorientasi pada pengguna, intuitif, dan sukses dalam memenuhi kebutuhan pengguna serta tujuan bisnis. Sifat iteratif dari Design Thinking memastikan perangkat lunak terus berkembang dan beradaptasi dengan perubahan kebutuhan pengguna dan dinamika pasar.

Materi yang terakhir adalah mengenai Sistem Kontrol serta GIT. Sistem Kontrol (Version Control) adalah metode yang digunakan untuk melacak dan mengelola perubahan dalam kode sumber atau berkas proyek. Sistem Kontrol ini dibedakan menjadi 2 macam yaitu Sistem Kontrol Versi Terpusat (Centralized Version Control System) dan Sistem Kontrol Versi Terdistribusi (Distributed Version Control System).

Dalam sistem kontrol versi terpusat, ada satu repositori sentral yang berfungsi sebagai "master" untuk menyimpan seluruh sejarah proyek. Setiap pengembang melakukan perubahan pada salinan lokal, kemudian mengirimkan perubahan tersebut ke repositori sentral. Contoh sistem kontrol versi terpusat adalah Subversion (SVN)

Dalam sistem kontrol versi terdistribusi, setiap anggota tim memiliki salinan lengkap dari seluruh repositori. Ini berarti setiap pengembang memiliki salinan lengkap sejarah

perubahan, tidak hanya salinan terbaru. Contoh sistem kontrol versi terdistribusi adalah Git, Mercurial, dan Bazaar.

Lalu bagaimana dengan GIT? Git adalah sistem kontrol versi terdistribusi yang memungkinkan pengembang perangkat lunak untuk melacak perubahan dalam kode mereka, berkolaborasi dengan anggota tim, dan mengelola revisi kode secara efektif. Dokumentasi ataupun cara instalasi dan penggunaan sudah banyak tersedia di website resmi dan sumber lain di internet.