Nama : Reza Pahlevi Kurniawan

Kelompok : 2

Full Stack Development merupakan 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 Full Stack Development

Scope	Definisi	Dasar-Dasar	Popular Framework
Front End Devolepment	Membangun antarmuka pengguna yang menarik dan interaktif.	- HTML - CSS - Javascript	- React - Vue.js - AngularJS
Back End Development	Membangun server dan aplikasi yang berfungsi sebagai "otak" dari aplikasi, menerima permintaan dari sisi depan, memproses data, dan memberikan respons yang sesuai.	- Node.js -> Express.js - Python -> Flask - Ruby -> Rails - Java -> Spring - PHP -> Laravel	- ExpressJS - Laravel - Spring - Rails
Database Management	Mendesain dan mengelola basis data untuk menyimpan, mengambil, dan memanipulasi data aplikasi.	- SQL (Oracle, MySQL, PostgreSQL, SQL Server) - NoSQL (MongoDB, Firebase, Redis) - Query (SELECT, INSERT, UPDATE, DELETE)	
Integration of Front-End and Back-End	Menghubungkan komponen front-end dengan layanan back-end melalui API (Application Programming Interface) untuk berkomunikasi dengan server dan database.	- Postman - Swagger	
Version Control and Collaboration	Menggunakan sistem pengendalian versi, seperti Git, untuk mengelola perubahan kode dan kolaborasi dalam tim pengembang.	- GitHub - Git	
Mobile Development	Beberapa Pengembang Full Stack juga memiliki	- Android (Java, Kotlin)	- React Native - Flutter

	kemampuan untuk mengembangkan aplikasi mobile menggunakan framework seperti React Native, Flutter,	- Ios (Swift, Objective-C) - IDE	

- Tahap- tahap pengembangan aplikasi end-to-end
  1. Perencanaan dan Analisis
  2. Desain
  3. Pengembangan Front-End
  4. Pengembangan Back-End
  5. Integrasi dan Pengujian
  6. Pemeliharaan dan Peningkatan

- Tools sets sebagai Full Stack Developer

IDE – Code Editor	Visual Studio Code
Version Control – Repository	- GitHub - GitLab - Bitbucket
Version Control – Git Tools	- Sourcetree - GitLens
DBMS	- PostgreSQL - MySQL - Oracle - MongoDB - Redis
API	- Postman - Swagger
Tests dan Debugging	- Jest - Mocha and Chai - Junit 5
Mobile Development	- React Native - Flutter
Layanan Cloud	- AWS - Google Cloud - Azure
CI/CD	- Jenkins - Circleci
Desain UI/UX	- Figma - Sketch

SDLC (Siklus Hidup Pengembangan Perangkat Lunak) adalah rangkaian proses yang terstruktur dan metodologi yang digunakan untuk mengembangkan perangkat lunak dari awal hingga selesai.

- Fase-fase pada SDLC

No	Siklus	Definisi
1.	Perencanaan dan Analisis	Tahap pertama ini melibatkan identifikasi masalah atau kebutuhan bisnis yang perlu diselesaikan oleh perangkat lunak. Para pemangku kepentingan berinteraksi untuk mengumpulkan persyaratan dan menentukan ruang lingkup proyek.
2.	Desain Produk	Di tahap ini, perangkat lunak dirancang secara rinci berdasarkan persyaratan yang telah dikumpulkan. Desain mencakup arsitektur sistem, antarmuka pengguna, dan desain database.
3.	Pengembangan Produk	Tahap ini melibatkan implementasi rancangan perangkat lunak yang telah disetujui sebelumnya. Para pengembang menulis kode untuk menghasilkan produk perangkat lunak yang berfungsi.
4.	Pengujian Produk	Setelah perangkat lunak dikembangkan, tahap pengujian dilakukan untuk memastikan bahwa perangkat lunak berfungsi sesuai dengan persyaratan yang telah ditentukan.
5.	Penerapan Produk	Tahap ini melibatkan implementasi rancangan perangkat lunak yang telah disetujui sebelumnya. Para pengembang menulis kode untuk menghasilkan produk perangkat lunak yang berfungsi.
6.	Pemeliharaan Produk	Setelah perangkat lunak diimplementasikan, pemeliharaan dilakukan untuk memperbaiki bug, meningkatkan fitur, dan menjaga perangkat lunak agar tetap sesuai dengan perubahan kebutuhan bisnis.

- Model-model SDLC

No	Model	Pengertian
1.	Waterfall Model	Waterfall model adalah model SDLC yang linier dan berurutan. Setiap tahap dalam model ini harus selesai sebelum memulai tahap berikutnya. Tahapannya meliputi analisis, perencanaan, desain, pengembangan, pengujian, implementasi, dan pemeliharaan.
2.	V-Shaped Model	Model V-Shaped adalah model yang terkait erat dengan model waterfall, tetapi menekankan pada pengujian. Tahapan pengujian diwakili oleh garis miring "V", yang berarti bahwa setiap tahap pengembangan memiliki tahapan pengujian yang sesuai.
3.	Prototype Model	Model Prototype adalah model pengembangan perangkat lunak yang bertujuan untuk menciptakan prototipe atau contoh awal sebelum mengembangkan versi finalnya. Model

		ini fokus pada pemahaman kebutuhan pengguna dan mengumpulkan umpan balik untuk memastikan bahwa perangkat lunak akhir sesuai dengan ekspektasi dan persyaratan pengguna.
4.	Spiral Model	Model ini menggabungkan elemen model spiral dengan pendekatan inkremental. Setiap siklus spiral membangun pada inkrementasi sebelumnya, menghasilkan perangkat lunak yang semakin berkembang dengan fitur yang lebih banyak setiap siklusnya.
5.	Iterative Incremental Model	Model ini melibatkan pengulangan siklus pembangunan dan peningkatan perangkat lunak dalam tahapan-tahapan kecil. Setiap iterasi menambahkan lebih banyak fitur hingga produk akhir mencapai tingkat kesempurnaan yang diinginkan.
6.	Big Bang Model	Model Big Bang adalah model yang kurang terstruktur, di mana semua tahapan pengembangan dilakukan tanpa perencanaan yang detail. Pengembangan dimulai tanpa melakukan analisis dan perencanaan yang mendalam.
7.	Agile Model	Model Agile adalah pendekatan kolaboratif dan iteratif yang berfokus pada pengiriman perangkat lunak secara berkala dan inkremental. Tim bekerja dalam sprint (iterasi singkat) dan selalu terbuka untuk perubahan persyaratan pengguna.

- Steps Design Thinking
  1. Empathize: Understand User Needs
  2. Define: Define the Problem
  3. Ideate: Generate Ideas
  4. Prototype: Build Quick and Iterative Solutions
  5. Test: Gather User Feedback
  6. Implement: Develop the Software

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.

- Pengunaan Version Control untuk Berkolaborasi
  1. Inisialisasi Proyek
     Tim memulai proyek dengan membuat repositori version control. Repositori ini akan menyimpan semua kode sumber, file, dan perubahan yang dilakukan selama pengembangan.
  2. Pengembangan Paralel
     Setiap anggota tim akan memiliki salinan repositori pada komputernya sendiri. Mereka dapat bekerja secara paralel, membuat perubahan.
  3. Branching

Version control memungkinkan pembuatan cabang (branch) yang terpisah dari kode utama. Ini memungkinkan tim untuk mengisolasi perubahan dan fitur yang sedang dikembangkan.

4. Merge
   Setelah fitur atau perubahan selesai, cabang dapat digabungkan kembali ke cabang utama (biasanya disebut sebagai "merge").
5. Pull Request
   Di beberapa platform version control seperti GitHub, GitLab, dan Bitbucket, pull request adalah mekanisme yang memungkinkan pengembang untuk mengajukan perubahan mereka untuk ditinjau oleh anggota tim lain sebelum digabungkan ke cabang utama.

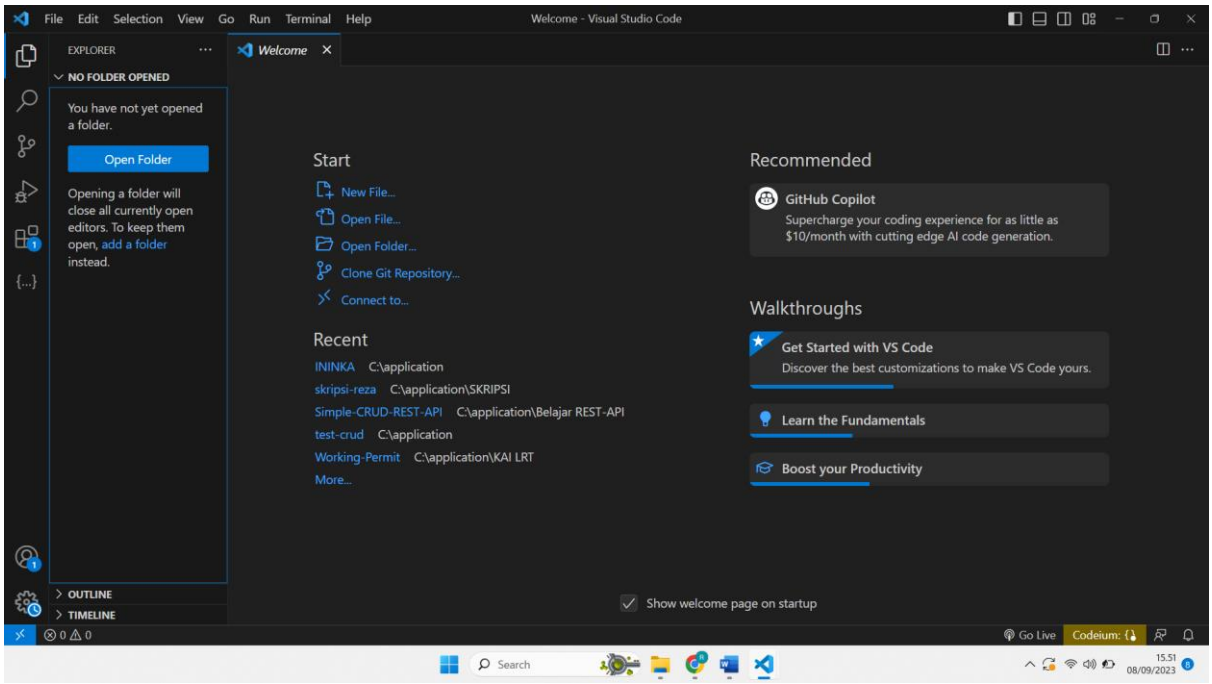
Git adalah sistem kontrol versi terdistribusi yang memungkinkan pengembang perangkat lunak untuk melacak perubahan dalam kode mereka, berkolaborasi dengan anggota tim, dan mengelola revisi kode secara efektif.

- Dasar-dasar Command Git

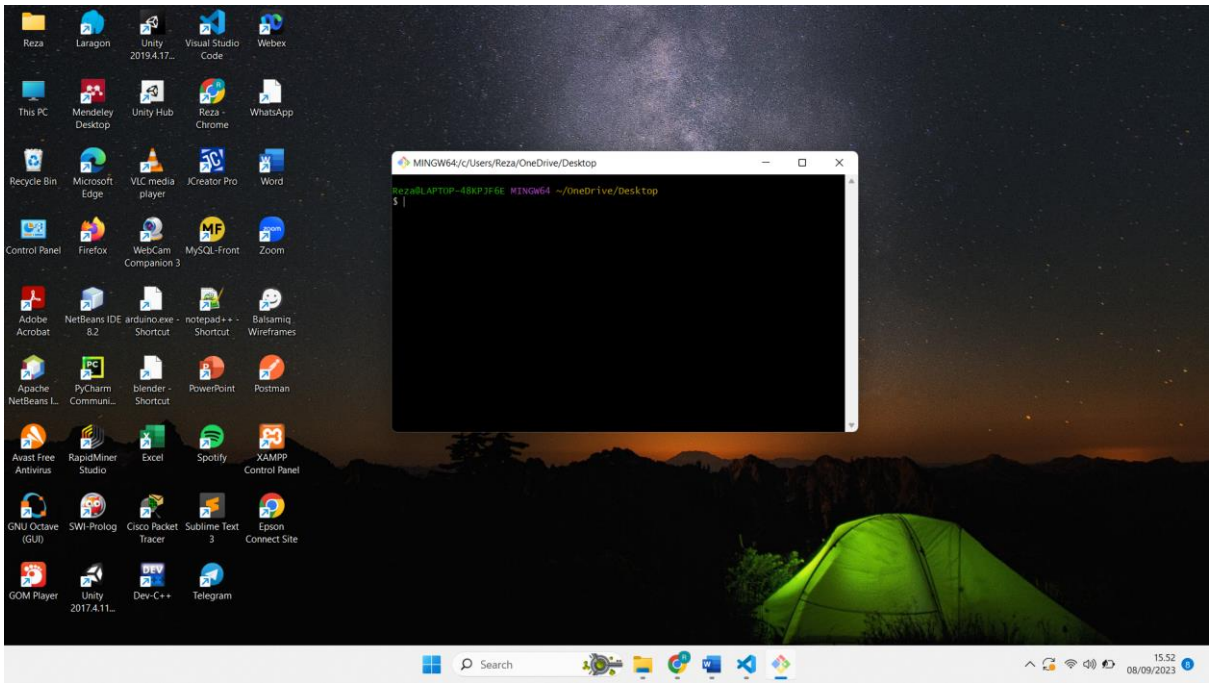
No	Nama	Command	Fungsi
1.	git init	`git init`	Menginisialisasi direktori sebagai repositori Git kosong.
2.	git clone	`git clone https://github.com/username/repo.git`	Menduplikasi repositori Git yang sudah ada ke direktori lokal.
3.	git status	`git status`	Menampilkan status perubahan yang belum dikomit di repositori local.
4.	git add	`git add file1.txt` `git add .`	Menambahkan perubahan ke area persiapan (staging area) untuk disiapkan menjadi commit.
5.	git commit	`git commit -m "Menambahkan fitur login"`	Membuat commit dari perubahan yang sudah di-staging dan menambahkan pesan commit.
6.	git push	`git push origin main`	Mengirimkan commit ke repositori jarak jauh (remote repository).
7.	git pull	`git pull origin main`	Mengambil commit terbaru dari repositori jarak jauh dan menggabungkannya ke repositori lokal.
8.	git branch	`git branch`	Menampilkan daftar cabang (branch) yang ada

			di repositori dan menunjukkan cabang aktif.
9.	git checkout	`git checkout feature-branch` `git checkout abc1234 (commit hash)`	Beralih ke cabang lain atau ke commit tertentu.
10.	git merge	`git merge feature-branch`	Menggabungkan perubahan dari satu cabang ke cabang aktif
11.	git log	`git log`	Menampilkan daftar commit beserta riwayatnya dalam repositori.
12.	git remote	`git remote -v`	Menampilkan daftar repositori jarak jauh yang terhubung dengan repositori local.
13.	git fetch	`git fetch origin`	Mengambil informasi terbaru dari repositori jarak jauh tanpa menggabungkan perubahan.
14.	git diff	`git diff`	Menampilkan perbedaan antara versi yang sudah di-staging dengan versi sebelumnya.
15.	git reset	`git reset file.txt`	Mengembalikan file yang sudah di-staging ke direktori kerja sebelumnya.

- **Capture Visual Studio Code**



➢ Capture Git


MINGW64:/c/Users/Reza/OneDrive/Desktop
Reza@LAPTOP-48KPJF6E MINGW64 ~/OneDrive/Desktop
$
Reza
Laragon
Unity 2019.4.17...
Visual Studio Code
Webex
This PC
Mendeley Desktop
Unity Hub
Reza - Chrome
WhatsApp
Recycle Bin
Microsoft Edge
VLC media player
JCreator Pro
Word
Control Panel
Firefox
WebCam Companion 3
MySQL-Front
Zoom
Adobe Acrobat
NetBeans IDE 8.2
arduino.exe - Shortcut
notepad++ - Shortcut
Balsamiq Wireframes
Apache NetBeans I...
PyCharm Communi...
blender - Shortcut
PowerPoint
Postman
Avast Free Antivirus
RapidMiner Studio
Excel
Spotify
XAMPP Control Panel
GNU Octave (GUI)
SWI-Prolog
Cisco Packet Tracer
Sublime Text 3
Epson Connect Site
GOM Player
Unity 2017.4.11...
Dev-C++
Telegram
Search
15.52
08/09/2023