Jawaban No. 1

Definisi Full Stack (Full Stack Development)

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 Full Stack Development

- **Front-End Development :** Membangun antarmuka pengguna yang menarik dan interaktif menggunakan HTML, CSS, dan JavaScript. Contoh : React, Angular, Vue.js, atau jQuery.
- **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. Contoh : Node.js, Python, Ruby, Java, PHP, atau C#.
- **Database Management :** Mendesain dan mengelola basis data untuk menyimpan, mengambil, dan memanipulasi data aplikasi. Contoh : MySQL, PostgreSQL, MongoDB.
- **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.
- **Version Control and Collaboration :** Menggunakan sistem pengendalian versi, seperti Git, untuk mengelola perubahan kode dan kolaborasi dalam tim pengembang.
- **Mobile Development** : Pengembangan aplikasi mobile mencakup beberapa aspek penting untuk memastikan aplikasi dapat berjalan dengan baik dan memberikan pengalaman pengguna yang optimal. Contoh : React Native, Flutter.

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

1. IDE - Code Editor : Visual Studio Code
2. Version Control
   - Repository : GitHub, GitLab, Bitbucket
   - Git Tools : Sourcetree, GitLens

3. DBMS : MySql, Oracle, MongoDB
4. API: Postman, Swagger
5. Tests & Debugging: Jest, Junit
6. Mobile Development: React Native, Flutter
7. Layanan Cloud: Google Cloud, AWS
8. CI/CD: Jenkins, Circleci
9. Desain UI/UX: Figma, Sketch

SDLC (Siklus Hidup Pengembangan Perangkat Lunak)

Rangkaian proses yang terstruktur dan metodologi yang digunakan untuk

mengembangkan perangkat lunak dari awal hingga selesai. 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.

Fase SDLC:

- Perencanaan dan Analisis
- Desain
- Pengembangan
- Pengujian
- Penerapan
- Pemeliharaan

Model SDLC:

- Waterfall Model
- V-Shaped Model
- Prototype Model
- Spiral Model
- Iterative Incremental Model
- Big Bang Model
- Agile Model

Design Thinking Implementation

Dengan menggabungkan pendekatan Design Thinking dalam Siklus Hidup Pengembangan Perangkat Lunak, tim pengembang dapat menciptakan produk yang lebih berfokus pada kebutuhan pengguna, mudah dipahami, dan berhasil dalam mencapai tujuan bisnis. Dengan menggunakan metode yang bersifat iteratif, Design Thinking memastikan bahwa perangkat lunak akan terus mengalami perkembangan dan penyesuaian untuk mengikuti perubahan dalam kebutuhan pengguna dan dinamika pasar.

Terminal

Dengan menggabungkan pendekatan Design Thinking dalam Siklus Hidup Pengembangan Perangkat Lunak (Software Development Life Cycle/SDLC), tim pengembang dapat menciptakan produk yang lebih berfokus pada kebutuhan pengguna, mudah dipahami, dan berhasil dalam mencapai tujuan bisnis. Dengan menggunakan metode yang bersifat iteratif, Design Thinking memastikan bahwa perangkat lunak akan terus mengalami perkembangan dan penyesuaian untuk mengikuti perubahan dalam kebutuhan pengguna dan dinamika pasar.

Version Control Git

Kontrol versi adalah metode yang digunakan untuk melacak dan mengelola perubahan dalam kode sumber atau berkas proyek. Git merupakan salah satu sistem kontrol versi terdistribusi yang paling populer dan kuat. 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.

Installing GIT

- Install Git pada Windows
- Install Git pada Linux
- Install Git pada MacOS

Collaborating Using Git/ Kolaborasi Menggunakan GIT

Menggunakan GitHub melibatkan pengaturan dasar repository, pembuatan cabang, membuat perubahan, mengelola konflik, dan menyelesaikannya dengan permintaan tarik (pull request) dan penggabungan (merge).