Nama : Muhammad Arafie Setiawan

NIM : 2111500340

**Rangkuman Materi Introduction to Software Engineering**

Pengembangan Full Stack (Full Stack Development) merujuk pada pengembangan aplikasi dari sisi depan (front-end), sisi belakang (back-end), dan pada beberapa kasus, sisi klien (client-side). Pengembangan Full Stack mencakup pengembangan seluruh aplikasi secara end-to-end. Hal ini melibatkan pengembangan antarmuka pengguna (user interface), pemrograman server-side, pengolahan database, dan integrasi sistem. Tujuan dari pengembangan Full Stack adalah untuk memastikan bahwa aplikasi yang dibangun memenuhi semua aspek yang dibutuhkan oleh pengguna dan berfungsi dengan baik di setiap sisi aplikasi.

**Scope Pada Full Stack Development :**

1. **Front-End Development :**

Membangun antarmuka pengguna yang menarik dan interaktif menggunakan HTML, CSS, dan JavaScript. Selain itu, pengembang dapat menggunakan framework dan pustaka front-end seperti React, Angular, Vue.js, atau jQuery untuk mempercepat pengembangan dan meningkatkan efisiensi.

1. **Back-End Development :**

Mencakup pembangunan server dan aplikasi yang berfungsi sebagai "otak" dari aplikasi, menerima permintaan dari sisi depan, memproses data, dan memberikan respons yang sesuai. Pekerjaan ini melibatkan penggunaan bahasa pemrograman server-side seperti Node.js, Python, Ruby, Java, PHP, atau C#.

1. **Database Management :**

Proses mendesain dan mengelola basis data untuk menyimpan, mengambil, dan memanipulasi data aplikasi. Teknologi database yang sering digunakan termasuk MySQL, PostgreSQL, MongoDB, atau Firebase.

1. **Integration of Front-End and Back-End :**

Proses menghubungkan komponen front-end dengan layanan back-end melalui API (Application Programming Interface) untuk berkomunikasi dengan server dan database. Tujuannya adalah untuk menyelaraskan data dan tampilan antara sisi depan dan sisi belakang aplikasi.

1. **Version Control and Collaboration :**

Sebuah sistem pengendalian versi seperti Git yang digunakan untuk mengelola perubahan kode dan kolaborasi dalam tim pengembang. Dalam penggunaannya, sistem ini mampu memastikan bahwa kode akan terus berkembang dengan aman dan sesuai dengan tujuan proyek.

1. **Mobile Development :**

Beberapa pengembang Full Stack juga memiliki kemampuan untuk mengembangkan aplikasi mobile menggunakan framework seperti React Native dan Flutter.

**Tahap – tahap pengembangan aplikasi End-to-End**

1. **Perencanaan dan Analisis**

Tahap awal dari pengembangan aplikasi adalah perencanaan dan analisis. Tahap ini melibatkan pengumpulan kebutuhan dan pemahaman mendalam tentang tujuan aplikasi, sasaran pengguna, dan lingkungan operasional. Analisis kebutuhan dan riset pasar dilakukan untuk mengidentifikasi fitur utama yang harus dimasukkan dalam aplikasi.

1. **Desain**

Tahapan desain melibatkan perancangan antarmuka pengguna (UI) dan pengalaman pengguna (UX) yang menarik serta intuitif, sambil juga merencanakan arsitektur aplikasi termasuk pemilihan teknologi, database, dan framework yang sesuai.

1. **Pengembangan Front-End**

Tahapan pengembangan front-end melibatkan penggunaan HTML, CSS, dan JavaScript untuk menciptakan tampilan dan interaksi yang menarik bagi pengguna, dengan potensi penggunaan framework seperti React, Angular, atau Vue.js untuk mempercepat proses pengembangan.

1. **Pengembangan Back-End**

Tahapan pengembangan back-end adalah proses pengembangan sisi server dan logika bisnis aplikasi, menggunakan bahasa pemrograman server-side seperti Node.js, Python, Ruby, atau Java, dan memanfaatkan framework back-end seperti Express.js, Flask, atau Ruby on Rails.

1. **Integrasi dan Pengujian**

Tahapan Integrasi dan Pengujian melibatkan integrasi antara bagian depan dan belakang aplikasi melalui API (Application Programming Interface) untuk memungkinkan komunikasi dan pertukaran data, serta melakukan pengujian aplikasi untuk memastikan kinerja yang benar, mengidentifikasi, dan memperbaiki bug jika ada.

1. **Pemeliharaan dan Peningkatan**

Tahapan Pemeliharaan dan Peningkatan melibatkan pemeliharaan aplikasi dengan memperbaiki bug dan menyesuaikan dengan perubahan lingkungan atau kebutuhan bisnis. Selain itu, peningkatan terus menerus dilakukan untuk memperbarui fitur, meningkatkan kinerja, dan menjaga relevansi aplikasi dalam jangka waktu yang berkelanjutan.

**Tools yang Digunakan**

1. Code Editor : Visual Studio Code
2. Version Control :

Repository : GitHub, GitLab, Bitbucket

Git Tools : Sourcetree, GitLens

1. DBMS : MySQL, Oracle, MongoDB
2. API : Postman, Swagger
3. Tests & Debugging : Jest, Junit
4. Mobile Development : React Native, Flutter
5. Layanan Cloud : Google Cloud, AWS
6. CI/CD : Jenkins. Circleci
7. 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).