# **Introduction of Software Engineering**

**Definisi Fullstack Web/Mobile Developer :**

Fullstack Development merujuk pada pengembangan seluruuh aplikasi secara end-to-end, dari depan (Front-end) hingga sisi belakang (Back-end) terkadang hingga sisi klien (Client-side).

**Memahami Pengembangan Fullstack Web/Mobile :**

* Memahami Scope Penting Fullstack Development dimana ada Frontend Development, Backend Development, Database Management, Integration Fronted dan Backend yang disebut API (Aplication Programming Interface), Version Control and Collaboration, dan Mobile Development.
* Memahami Bahasa yang digunakan di masing-masing Scope Fullstack Development. Seperti Frontend menggunakan HTML, CSS, dan Javascript, Backend menggunakan Python/ Java/ Javascript, serta Database Management Menggunakan SQL dan No SQL.

**Memahami Framework dan Library Front-end untuk membangun UI yang Responsif :**

* Memahami Framework yang saat ini sedang populer untuk Fullstack Development bagian Front-end seperti React.js, Vue.js, dan Angular.js.
* Memahami Framewok untuk styling seperti Bootstrap dan Tailwind.

**Memahami Pengembangan Back-end, termasuk pemrograman sisi server dan database :**

* Memahami Framework yang saat ini sedang populer untuk Fullstack Development bagian Back-end seperti Laravel untuk PHP, Express JS untuk Javascript, Spring untuk Java, dan Ruby Rails untuk Ruby.
* Memahami Framework yang saat ini sedang populer untuk Fullstack Development bagian Database Management seperti SQL (Oracle,PostgreSQL, MySQL) dan NoSQL(MongoDB dan Redis).

**Memahami Konsep dan Framework pengembangan aplikasi mobile :**

* Memahami Framework yang saat ini sedang populer untuk Fullstack Development bagian Mobile Development seperti React Native, dan Flutter.

**Memahami Tools untuk Test dan Debugging serta Layanan Cloud :**

* Tools untuk test dan debugging yaitu seperti Jest untuk Javascript, mocha, chai, Junit5.
* Tools untuk Layanan Cloud seperti AWS, Google Cloud, dan Azure.

**Memahami Tahap-tahap Pengembangan Aplikasi End-to-End :**

* Perencanaan dan Analisis, Tahap ini mengumpulkan data analisis kebutuhan dan riset pasar untuk mengidentifikasi fitur utama yang harus dimasukkan dalam aplikasi.
* Desain, Tahap ini merancang antarmuka pengguna (UI) dan pengalaman pengguna (UX) yang intuitif dan interaktif.
* Pengembangan Front-end, Tahap ini membuat tampilan yang interaktif dan menarik bagi pengguna dari hasil Desain.
* Pengembangan Back-end, Tahap ini melibatkan pengembangan sisi server dan logika bisnis aplikasi.
* Integrasi dan Pengujian, Tahap ini dilakukan untuk memastikan semua fitur berdungsi dengan benar dan mengidentifikasi dan memperbaiki bug yang mungkin ada.
* Pemeliharaan dan Peningkatan, Tahap ini dilakukan untuk memperbaiki bug dan menangani perubahan lingkungan atau kebutuhan bisnis.

**Manfaat Version Control untuk Berkolaborasi :**

* Merekam Perubahan
* Pencatatan Riwayat
* Pemecahan Konflik
* Pemulihan Mudah

**Penggunaan Version Control untuk Berkolaborasi :**

* Inisialisasi Proyek, Membuat Repository dan menyimpan semua kode sumber, file, dan perubahan yang dilakukan selama pengembangan.
* Pengembangan paralel, Semua tim memiliki salinan dari Repository dan bekerja secara paralel.
* Branching, Version Control dapat membuat cabang terpisah dari branch utama untuk mengisolasi perubahan dan fitur yang sedang dikembangkan.
* Merge, menggabungkan perubahan yang telah selesai ke branch utama.
* Pull Request, Pengembang mengajukan perubahan mereka untuk ditinjau oleh anggota tim lain sebelum digabungkan ke cabang utama.

**Memahami apa itu Terminal dan Dasar Terminal Command :**

* Sejarah singkat Terminal yaitu Early Terminals pada 1960, lalu mengalami pengembangan DEC VT100, UNIX dan Shell pada 1970, lalu menjadi Terminal Emulator pada 1980, selanjutnya menjadi Personal Computers with Graphical User Interface, dan terkahir menjadi Modern Terminal Development.
* Memahami Command Line Dasar Terminal pada Windows, Linux dan MacOS.

**Memahami Version Control Git :**

* Version Control adalah Metode yang digunakan untuk melacak dan mengelola perubahan kode atau berkas proyek.
* Git Merupakan salah satu Version Control versi terdistribusi yang paling populer dan paling kuat.

**Memahami cara menginstall Git :**

* Tata cara penginstalan git : <https://git-scm.com/book/ms/v2/Getting-Started-First-Time-Git-Setup>

**Memahami Dasar-dasar Command Git :**

* **Git Init**, Menginisialisasi direktori sebagai repositori.
* **Git Clone**, mengduplikasi repositori Git yang sudah ada.
* **Git Status**, Menampilkan status perubahan yang belum dikomit di repositori.
* **Git Add**, Menambahkan perubahan ke area persiapan (Staging Area) untuk disiapkan menjadi commit.
* **Git Commit**, Membuat commit dari perubahan yang sudah di-staging dan menambahkan pesan commit.
* **Git Push**, Mengirimkan commit ke repository jarak jauh (remote repository).
* **GIt Pull**, Mengambil commit terbaru dari repository jarak jauh dan menggabungkannya ke repository lokal.
* **Git Branch**, Menampilkan daftar cabang (branch) yang ada di repository dan menunjukkan cabang aktif.
* **Git Checkout**, beralih ke cabang lain atau ke commit.
* **Git Merge**, Menggabungkan perubahan dari satu cabang ke cabang aktif.
* **Git Log**, Menampilkan daftar commit berserta riwayatnya dalam repository.
* **Git Remote**, menampilkan daftar repository jarak jauh yang terhubung dengan repository lokal.
* **GIt Fetch**, Mengambil informasi terbaru dari repository jarak jauh tanpa menggabungkan perubahan.
* **Git DIff**, Menampilkan perbedaan antara versi yang sudah di-staging dengan versi sebelumnya.
* **Git Reset**, mengembalikkan file yang sudah di-staging ke direktori kerja sebelumnya.