Nama : Muhamad Ikhsan Rizqi Yanuar
Kelas : SIB 7 – IT Fullstack Developer
Kelompok Mentoring : Kelompok 1
Kelompok Final Project : Kelompok 3
Link GitHub Repository Kelompok:
https://github.com/sceptzrion/finpro-msib-7-kelompok-3
Link GitHub Branch Lokal:
https://github.com/sceptzrion/finpro-msib-7-kelompok-3/tree/ikhsan-homework

---

Proses Instalasi Tools

Berikut adalah langkah-langkah instalasi masing-masing tools:

Git

Git adalah salah satu sistem kontrol versi yang paling populer. Berikut langkah-langkah instalasinya:

1. **Download Git**: Kunjungi https://git-scm.com/downloads dan pilih versi yang sesuai dengan sistem operasi (Windows, macOS, Linux).
2. **Instalasi Git**:
    - Untuk **Windows**, jalankan file .exe yang diunduh dan ikuti wizard instalasi.
    - Untuk **macOS**, dapat menggunakan Homebrew dengan menjalankan perintah: brew install git.
    - Untuk **Linux**, instal melalui package manager seperti sudo apt install git di Ubuntu.
3. **Verifikasi Instalasi**: Buka terminal atau command prompt dan ketik perintah:

    ```
    git --version
    ```

    Jika terinstal dengan benar, akan muncul versi Git yang terpasang.

Visual Studio Code (VS Code)

Visual Studio Code adalah editor kode sumber terbuka yang ringan namun sangat kuat. Berikut langkah-langkah instalasinya:

1. **Download VS Code**: Kunjungi https://code.visualstudio.com/download dan pilih versi sesuai sistem operasi.
2. **Instalasi VS Code**:
    - Untuk **Windows**, jalankan file .exe yang diunduh dan ikuti wizard instalasi.
    - Untuk **macOS**, unduh dan buka file .dmg, kemudian pindahkan ke folder Applications.

    - Untuk **Linux**, unduh file .deb atau .rpm sesuai dengan distro yang digunakan, lalu instal menggunakan terminal.
3. **Verifikasi Instalasi**: Buka VS Code dari menu aplikasi dan pastikan tampilannya muncul dengan benar.

Browser

Instalasi browser biasanya sudah dilakukan sebelumnya, tetapi jika belum, berikut adalah langkah-langkahnya:

1. **Google Chrome**:
    - **Download** dari https://www.google.com/chrome/.
    - Ikuti langkah-langkah instalasi sesuai dengan OS.
2. **Mozilla Firefox**:
    - **Download** dari https://www.mozilla.org/firefox/.
    - Lakukan instalasi seperti yang diarahkan.

Browser digunakan untuk menjelajahi web, termasuk untuk mengakses dokumentasi dan panduan.

Ringkasan Materi Pengenalan Software Engineer

Berikut adalah ringkasan berdasarkan materi pengenalan software engineer yang telah dipelajari:

1. Fullstack Web Developer Career Path

Sebagai seorang Fullstack Web Developer, kita akan menguasai pengembangan aplikasi web baik pada sisi **front-end** maupun **back-end**. Berikut adalah beberapa aspek penting dari jalur karier ini:

- **Front-End Development**: Berkaitan dengan bagian tampilan atau user interface yang berinteraksi langsung dengan pengguna. Teknologi utama yang dipelajari meliputi:
    - **HTML**: Bahasa markup yang digunakan untuk struktur halaman web.
    - **CSS**: Bahasa styling yang digunakan untuk mengatur tampilan visual web.
    - **JavaScript**: Bahasa pemrograman yang membuat halaman web menjadi interaktif.
    - Framework seperti **React, Vue, atau Angular** yang mempermudah pembangunan UI interaktif.

- **Back-End Development**: Bagian dari pengembangan yang berhubungan dengan server, database, dan logika aplikasi. Teknologi yang digunakan meliputi:
    - **Node.js**, **Python**, atau **Ruby** sebagai bahasa back-end.
    - **Database**: Penggunaan database seperti **MongoDB** (NoSQL) atau **MySQL** (SQL) untuk penyimpanan data.
    - **API Development**: Membuat API untuk menghubungkan front-end dengan back-end.
- **Fullstack Developer**: Kombinasi kemampuan front-end dan back-end yang memungkinkan seorang developer menangani seluruh siklus pembangunan aplikasi web dari UI hingga pengelolaan data di server.

2. SDLC & Design Thinking Implementation

Software Development Life Cycle (SDLC) adalah kerangka kerja yang digunakan untuk mengelola proses pengembangan perangkat lunak. Dalam penerapannya, SDLC dapat digabungkan dengan **Design Thinking**, pendekatan kreatif untuk memahami kebutuhan pengguna dan menghasilkan solusi inovatif. Berikut tahapannya:

- **Planning**: Menentukan tujuan dan kebutuhan proyek berdasarkan analisis masalah.
- **Requirement Gathering & Analysis**: Mengumpulkan dan menganalisis kebutuhan sistem yang harus dipenuhi oleh perangkat lunak.
- **Design**: Merancang solusi perangkat lunak dengan mempertimbangkan arsitektur dan antarmuka pengguna.
- **Implementation**: Pengkodean atau pengembangan perangkat lunak berdasarkan desain yang telah dibuat.
- **Testing**: Menguji perangkat lunak untuk memastikan tidak ada bug dan bekerja sesuai spesifikasi.
- **Deployment**: Meluncurkan perangkat lunak ke lingkungan produksi.
- **Maintenance**: Memelihara perangkat lunak dengan memperbaiki bug dan menambah fitur baru sesuai kebutuhan pengguna.

Model-Model SDLC

1) **Waterfall Model**:
    - Merupakan model SDLC yang paling tradisional dan linear. Setiap fase dilakukan secara berurutan, di mana fase berikutnya hanya dapat dimulai setelah fase sebelumnya selesai.
    - Tahapan utamanya adalah: **Requirement Gathering**, **Design**, **Implementation**, **Testing**, **Deployment**, dan **Maintenance**.
    - Kelebihan: Sederhana dan mudah dipahami.
    - Kekurangan: Kurang fleksibel karena tidak memungkinkan perubahan setelah fase sebelumnya selesai.

2) **V-Shaped Model**:
    - Pengembangan ini adalah varian dari waterfall model, namun setiap fase pengembangan memiliki fase pengujian yang terkait.
    - Model ini menekankan pada **verifikasi** dan **validasi** secara menyeluruh.
    - Setiap tahap desain diikuti oleh tahap pengujian, seperti **Unit Testing**, **Integration Testing**, **System Testing**, dan **Acceptance Testing**.
    - Kelebihan: Memastikan pengujian setiap tahap.
    - Kekurangan: Masih kurang fleksibel terhadap perubahan yang terjadi di tengah siklus pengembangan.

3) **Prototype Model**:
    - Dalam model ini, prototipe dibuat sebagai representasi awal dari sistem untuk mendapatkan umpan balik pengguna lebih awal.
    - Setelah prototipe divalidasi oleh pengguna, pengembangan penuh dilakukan.
    - Kelebihan: Memungkinkan pemahaman kebutuhan pengguna yang lebih baik melalui umpan balik.
    - Kekurangan: Mungkin memerlukan waktu dan biaya tambahan untuk membuat prototipe.

4) **Spiral Model**:
    - Menggabungkan elemen **iterasi** dan **prototyping**. Proyek melalui beberapa tahapan dalam bentuk spiral (pengulangan).
    - Setiap "loop" spiral terdiri dari perencanaan, analisis risiko, pengembangan, dan evaluasi.
    - Kelebihan: Fokus pada manajemen risiko dan fleksibel terhadap perubahan.
    - Kekurangan: Kompleks dan membutuhkan keahlian khusus dalam manajemen risiko.

5) **Iterative and Incremental Model**:
    - Dalam model ini, sistem dibangun secara bertahap melalui serangkaian iterasi. Setiap iterasi menghasilkan bagian dari produk yang dapat digunakan.
    - Setiap iterasi memperbaiki kesalahan dari iterasi sebelumnya dan menambah fungsionalitas baru.
    - Kelebihan: Mudah untuk mengakomodasi perubahan selama proses pengembangan.
    - Kekurangan: Memerlukan perencanaan yang matang dan proses berkelanjutan.

6) **Big Bang Model**:
    - Pendekatan ini tidak mengikuti perencanaan formal. Pengembangan dimulai dengan sumber daya dan waktu yang tersedia, tanpa spesifikasi yang jelas.
    - Pengembang mulai menulis kode dengan sedikit atau tanpa analisis yang memadai.
    - Kelebihan: Cocok untuk proyek kecil atau eksperimental.
    - Kekurangan: Risiko kegagalan sangat tinggi, terutama untuk proyek besar.

7) **Agile Model**:
    - Agile adalah pendekatan yang mengutamakan **iterasi cepat**, **kolaborasi tim**, dan **fleksibilitas** terhadap perubahan.
    - Pengembangan dilakukan dalam siklus singkat yang disebut **sprints**, biasanya 2–4 minggu.
    - Agile mengutamakan **interaksi antar manusia** daripada proses formal dan alat, serta memberikan perangkat lunak yang bekerja daripada dokumentasi yang komprehensif.
    - Kelebihan: Sangat fleksibel, memungkinkan perubahan berdasarkan umpan balik pengguna secara cepat.
    - Kekurangan: Memerlukan manajemen dan komunikasi yang kuat agar tetap efisien.

Design Thinking diterapkan pada tahap awal SDLC untuk lebih memahami pengguna melalui pendekatan **Empathy**, mengidentifikasi **Problems**, dan menciptakan solusi dengan ide-ide kreatif. Proses ini melibatkan beberapa tahap:

- **Empathize**: Memahami kebutuhan pengguna.
- **Define**: Menetapkan masalah yang perlu dipecahkan.
- **Ideate**: Menghasilkan berbagai ide solusi.
- **Prototype**: Membuat prototipe solusi untuk diuji.
- **Test**: Menguji prototipe pada pengguna dan mendapatkan umpan balik.

3. Basic Git & Collaborating Using Git

Git adalah sistem kontrol versi yang digunakan untuk melacak perubahan pada kode sumber dan memungkinkan kolaborasi antar-developer. Berikut adalah beberapa konsep dan perintah dasar yang perlu dipahami:

- **Git Init**: Perintah untuk menginisialisasi repository Git di dalam direktori proyek.
- **Git Add**: Menambahkan perubahan baru ke staging area sebelum commit.
- **Git Commit**: Menyimpan snapshot dari perubahan yang ada di staging area ke dalam repository lokal.
- **Git Push**: Mengirim commit dari repository lokal ke repository remote (misalnya GitHub, GitLab).
- **Git Pull**: Mengambil update terbaru dari repository remote ke repository lokal.
- **Branching**: Membuat cabang pengembangan untuk bekerja pada fitur atau perubahan baru tanpa memengaruhi kode utama.
- **Merging**: Menggabungkan perubahan dari cabang lain ke dalam cabang utama.