Summary of Introduction Software Engineering

A. What do you know about Fullstack development ?

Fullstack development atau yang dapat diartikan dalam bahasa yaitu pengembangan fullstack merupakan salah satu bagian dari jenis web development yang dimana memperhatikan dua bidang sekaligus frontend dan backend. Frontend sendiri berhubungan dengan sisi depan baik berupa tampilan atau responsivitas website atau dapat dikatakan mengatur client-side. Di sisi lain pada backend berkontribusi di belakang layar yang berkaitan dengan tersedia dan pengelolaan server atau dapat bisa disebut mengurus bagian server-side. Dalam menjalankan website yang lebih dinamis dan responsif dibutuhkan pengelolaan data atau bisa disebut dengan database management

Dapat dikatakan fullstack developer apabila seseorang tersebut mampu mengoperasikan dan memiliki kemampuan di bidang frontend, backend, dan juga database management. Fullstack developer sendiri memiliki tugas sebagai pengembang perangkat lunak atau software baik situs web dan sebuah aplikasi yang digunakan pada mobile device. Ketika menjadi seorang fullstack developer terdapat keuntungan yang didapatkan yaitu dalam sebuah tim pengembangan adalah mereka dapat bekerja pada berbagai aspek proyek dengan lebih fleksibel, memiliki pemahaman yang lebih holistik tentang seluruh sistem, dan seringkali dapat mengurangi overhead komunikasi antara tim front-end dan back-end.Namun, dengan scope bidang ini yang diwajibkan untuk memiliki pengetahuan yang luas tentang berbagai teknologi dan bahasa pemrograman, menjadi seorang fullstack developer bisa menjadi tugas yang menantang, dan mereka perlu terus memperbarui pengetahuan mereka untuk tetap relevan dalam dunia pengembangan perangkat lunak yang terus berubah.

B. How to make proper website ?

Tentu dalam membuat website diperlukan rangkaian proses yang terstruktur dan mendalam supaya pengembangan perangkat lunak dapat berjalan dengan baik dan sesuai dengan kemauan stakeholder. Terdapat istilah siklus SDLC (Software Development Lifecycle) yang berisikan tahapan ataupun siklus bagaimana mengembangkan software dengan baik dan benar. Siklus terdiri dari 6 fase diantaranya yaitu :

1. Perencanaan dan analisis, identifikasi masalah atau kebutuhan bisnis berkaitan software
2. Desain Produk, planning atau gambaran umum desain software dari insight yang didapat
3. Pengembangan produk, implementasi perancangan software dengan programming
4. Pengujian produk, pengujian kelayakan atau evaluasi software
5. Penerapan produk, implementasi rancangan software sehingga dapat digunakan user
6. Pemeliharaan produk, memelihara, debugging, peningkatan fitur, dan menjaga software

Adanya siklus SDLC ini memiliki manfaat dan sangat membantu seorang fullstack developer dari Perencanaan dan menganalisis hal-hal yang diperlukan sampai tahap terakhir pemeliharaan produk. Berikut beberapa manfaat yang diperoleh diantaranya :

- Prediktabilitas dan PengendalianProyek
- Peningkatan Kualitas PerangkatLunak
- Pengelolaan Risiko yang Lebih Baik
- Efisiensi Tim dan Kolaborasi

Dalam mengimplementasikan SDLC sendiri dibagi lagi menjadi beberapa model yang disesuaikan dengan kebutuhan sebuah project, diantaranya yaitu :

- Waterfall Model, linear dan berurutan dengan persyaratan yang jelas dan stabil
- V-Shaped Model, menekankan pada pengujian pada tiap tahapannya
- Prototype Model, menciptakan prototipe atau contoh awal sebelum launch software
- Spiral Model, pendekatan inkremental dengan menghasilkan pengembangan software yang lebih baik
- Iterative Incremental Model, pengulangan siklus pembangunan dan peningkatan software
- Big Bang Model, tingkat terstrukturnya yang kurang dan tanpa perencanaan yang detail
- Agile Model, pendekatan kolaboratif dan iteratif dengan membagi tahapannya menjadi bagian lebih kecil untuk menghasilkan software yang kompleks

Selain memahami bagaimana software dapat dibangun dengan baik melalui menggunakan beberapa model SDLC, kini kita harus dapat menangani kebutuhan user terhadap software yang akan kita buat. Terdapat tahapan atau proses yang digunakan yaitu design thinking, dimana proses ini sangat membantu kita dalam memahami user dan bagaimana solve problem yang dimiliki oleh user, diantaranya sebagai berikut :

- Empathize, memahami kebutuhan user melalui user research, empathy mapping, dan user persona
- Define, menemukan masalah berasal dari insight yang didapatkan dengan membuat problem statement dan stakeholder alignment
- Ideate, menemukan ide sebagai solusi dari masalah yang muncul melalui brainstorming sessions dan idea consolidation
- Prototype, merepresentasikan ide-ide yang dipilih berupa low-fidelity high fidelity prototype
- Test, pengujian dengan mengumpulkan feedback melalui usability testing dan iterative testing
- Implement, mengimplementasikan dengan kode programming menggunakan agile development dan cross-functional collaboration

C. What are the preparations to become a fullstack developer

Sebagai seorang fullstack developer memiliki tugas yang cukup banyak diantaranya yaitu membangun website ataupun mobile development, yang dimana perlu untuk memiliki wawasan ataupun dapat mengoperasikan tools ataupun bahasa pemrograman yang digunakan pada bidang frontend, backend, ataupun database management, berikut ini apa saja yang harus dipersiapkan :

a. Frontend web development
   Structure & Styling : HTML & CSS
   Programming language : javascript
   Framework : React, Vue.js, AngularJS
b. Backend web development
   Programming language : Python, Ruby, Java, Javascript(node.js), PHP, C#, …
   Framework : Express.js(Node.js), Flask Python(Python), Ruby on Rails (Ruby), Spring (Java), Laravel (PHP)
c. Database Management
   SQL : Oracle, PostreSQL, MySQL
   No SQL : MongoDB, Redis

d. Dasar - dasar mobile development
   Programming language (android) : Java dan kotlin
   Programming language (iOS) : Swift dan Objective-C
   Framework : React Native dan Flutter

e. Other
   API : Postman dan Swagger
   Test & Debugging : Jest, mocha chai, JUnit5
   CI/CD : Jenkins, circle ci
   Cloud service : aws, google cloud, azure
   UI/UX design : Figma dan sketch

   Setelah mengetahui beberapa keperluan tersebut kita harus mengetahui platform yang dapat kita gunakan dalam mengimplementasikan bahasa pemrograman dan framework yang kita pakai atau yang biasa disebut dengan IDE (Integrated Development Environment). IDE merupakan aplikasi perangkat lunak yang membantu para pemrogram mengembangkan kode perangkat lunak secara efisien. Selain itu, dalam penggunaan IDE-code editor ketika programming kita memerlukan Version Control System yang dimana digunakan untuk merekam perubahan dari berkas selama kita melakukan programming terhadap code kita. Salah VCS yang paling terkenal adalah version control git atau biasa disebut GIT, sebagai sistem kontrol versi terdistribusi. Berikut beberapa IDE dan VCS yang umum dipakai :

a. Integrated Development Environment - code editor/text editor
   Visual Code Studio, Notepad++, Sublime Text, Vim, Atom, Adobe Dreamweaver

b. Version Control System
   Git, Mercurial, bitbucket, Gitlab dan bazaar

Terdapat beberapa penggunaan dasar dari command GIT yang harus diketahui diantaranya :
Git init, menginisialisasi direktori sebagai repo git
Git clone, menduplikasi repository Cit yang sudah ada direktori lokal.
Git status, Menampilkan status yang belum di commit di repositori lokal.
Git add, Menambahkan Færubahan ke area persiapan (staging area) untuk disiapkan menjadi commit.
Git commit, Membuat commit dari perubahan yang sudah di staging dan pesan commit
Git push, Mengirimkan commit ke repository
Git pull, Mengambil commit terbaru dari repositori jarak jauh dan menggabungkannya ke repositori lokal.
Git branch, Menampilkan daftar cabang (branch) yang ada di repositori dan menunjukkan cabang aktif.
Git checkout, Beralih ke cabang lain atau commit
Git merge, menggabungkan perubahan dari satu cabang ke cabang aktif
Git log, Menampilkan daftar commit beserta riwayatnya dalam repositori
Git remote, Menampilkan daftar repository jarak jauh yang terhubung dengan repositori lokal.
Git fetch, Mengambil informasi terbaru dari repositori jarak jauh tanpa menggabungkan perubahan.
Git diff, Menampilkan antara versi yang dan di-staging dengan versi Sebelumnya.
Git reset, Mengembalikan file yang sudah di-staging ke direktori kerja sebelumnya.
Dan masih banyak lainnya..