INTRODUCTION
TO SOFTWARE ENGINEERING

MUH. ILHAM SAPUTRA J
KELOMPOK 7 - UNIVERSITAS HASANUDDIN

FULL STACK DEVELOPER CAREER PATH

1. Pengenalan *Full Stack Development*

   Pengembangan *Full Stack* terdiri dari pengembangan *Front-End, Back-End, Database Management,* hingga pada *client-side* pada sebagian kasus. Dalam pengembangan *full stack*, ada beberapa *scope* (ruang lingkup) penting sebagai berikut:

   a. *Front-end Development*
   Pada *Scope* ini *developer* membangun antar muka yang menarik menggunakan HTML, CSS, dan JavaScript. Untuk meningkatkan efisiensi dalam pengembangan, *developer* dapat memanfaatkan *framework* seperti React, Angular, Vue.Js dan JQuery

   b. *Back-End Developer*
   Scope terkait pembangunan *server* sebagai otak aplikasi, memproses data, dan memberi respon. Bahasa pemrograman yang digunakan adalah Node.Js, Phyton, Ruby, Java, PHP, dan C#. Pada *scope back-end developer* ini pengembang juga dapat memanfaatkan *framework* yaitu Express.Js untuk bahasa Node.Js, Flask untuk Phython, Ruby on Rails untuk Ruby, Spring untuk Java dan Laravel untuk PHP.

   c. *Database Management*
   Pada ruang ini pengembang dimungkinkan untuk melakukan pengelolaan basis data (*Select, insert, update, dan delate*). Bahasa pemrograman yang digunakan, yaitu MySQL, PostgreSQL, MongoDB, dan Firebase.

   d. *Integration of Front-End dan Back-End*
   Yaitu menyelaraskan data dan tampilan melalui API (*Application Programming Interface*)

   e. *Version control and collaboration*
   Yaitu memastikan kode terus berkembang sesuai dengan tujuan. Salah satu *tool* yang sering digunakan adalah GIT.

   f. *Mobile developmen*t
   Yaitu ruang pengembangan aplikasi melalui *framework* React Native, dan Flutter. Sebagai alat bantu dalam pengembangan aplikasi *developer* biasanya menggunakan IDE yang menyediakan alat bantu penyunting kode, pengelola proyek, *simulator* dan *debugging*. Pada sistem operasi Android, IDE yang digunakan adalah Android Studio. Pada sistem operasi IOS, menggunakan IDE Xcode. Selain itu, ada juga VSCode yang dapat digunakan secara universal.

2. *Skillset Full Stack Developer*

   a. Pengembangan Aplikasi *End-to-end*

   Merupakan pendekatan pengembangan perangkat lunak dari perancangan hingga pengujian dan implementasi.

   - Perencanaan : Analisis kebutuhan dan riset pasar untuk MVP.
   - Desain : merancang UI/UX, merencanakan arsitektur berupa pemilihan teknologi *database* dan *framework*.
   - *Front-end-development*

- *Back-end-development* : Pengembangan sisi *server* dan logika bisnis aplikasi.
- Integrasi dan pengujian : *Front-end* dan *back-en*d diintegrasikan melalui API sehingga dapat berkomunikasi dan berbagi data.
- Pemeliharaan dan peningkatan : memperbaiki bak menyesuaikan kebutuhan pasar dan menjaga relevansi produk.

b. Kolaborasi efektif dengan *version control*
*Version contro*l adalah **sistem untuk melacak perubahan pada kode sumber aplikasi** yang memungkinkan kolaborasi yang efisien dengan menggunakan *tools* GIT atau Mercurial. Ada beberapa manfaat dari *version control* ini yaitu dapat merekam perubahan, mencatat riwayat, pemecahan konflik, dan pemulihan yang mudah.

c. Penggunaan *version control* untuk berkolaborasi
- Inisialisasi produk yaitu membuat *repository Version contro*l untuk menyimpan kode.
- Pengembangan paralel Yaitu setiap Anggota tim memiliki salinan *repository* untuk bekerja secara paralel.
- *Branching* Yaitu pembuatan cabang untuk mengisolasi fitur lain yang masih dalam pengembangan.
- *Merge* Yaitu penggabungan Cabang-cabang.
- *Pull request* yaitu mekanisme untuk mengajukan perubahan untuk ditinjau anggota tim sebelum digabungkan ke cabang utama.

3. Tools Full Stack Developer

Ada berbagai macam alat bantu yang dapat digunakan oleh pengembang dalam pengembangan produk di antaranya sebagai berikut:

IDE → VSCode, Android Studio, dan XCode
Version Control Repository → GitHub, GitLab, Bitbucket
Version Control Git Tools → Sourcetree, dan GitLens
DBMS → PostgreSQL, MySQL, Oracle dan MongoDB
API → Postman, dan Swagger
Test & debugging → Jest (JS), JUnit (Java)
Mobile Development → React Native, dan flutter
Cloud → AWS, google Cloud, dan Azure
CI/CD → Jenkins, CircleCI
UI/UX → Figma dan sketch

SDLC & DESIGN THINGKING IMPLEMENTATION

1. Pengenalan
   SDLC *(Software Developmet Live Circle)* adalah serangkaian proses untuk mengembangkan perangkat lunak. Fase dari SDLC ini adalah *planning, analysis, design, development, testing & integration, maintenance,* kembali lagi ke *planning*, dan seterusnya.
   Manfaat dari **siklus hidup pengembangan perangkat lunak** ini adalah agar organisasi dapat meningkatkan keberhasilan dan efisiensi pengiriman produk yang tepat waktu dan memberikan nilai yang besar bagi pelanggan.
2. Model-model SDLC
   a. *Waterfall* : Relevan dengan proyek dengan **persyaratan yang jelas dan stabil.**
   b. *V-shaped* : Relevan dengan proyek dengan **fokus pada kualitas tinggi.**
   c. *Prototype* : bertujuan untuk **menciptakan prototipe.**
   d. *Spiral* : Relevan dengan **proyek besar dan kompleks dengan banyak resiko.**
   e. *Iterative incremental* : Relevan dengan proyek dengan **waktu dan dana yang terbatas.**
   f. *Big bang model* : Relevan dengan proyek yang kecil, dan **sifatnya *prototyping*.**
   g. *Agile model* : Relevan dengan proyek dengan **lingkungan yang dinamis dan persyaratan yang berubah-ubah.**
3. *Design Thingking Implementation*
   Implementasi dari *design thinking* ini dapat dilihat dari 3 tahap *design thingking* yaitu ***understanding, explore dan materialize***. Tahap *understanding* terdiri dari *emphatize* dan *define*. Tahap *explore* terdiri dari *ideate* dan *prototype*. Dan tahap *materialize* terdiri dari *testing* dan *implement*.
   Manfaat dari *design thinking implementation* adalah agar *developer* dapat menciptakan produk yang lebih berorientasi pada pengguna dan tujuan bisnis.

BASIC GIT & COLLABORATING USING GIT

Pada sub materi ini saya belajar terkait command line dasar dan bagaimana caranya menginstall, menginisialisasi dan *commiting* git seperti berikut:

1. Melakukan pembuatan akun pada GitHub dan *download* GIT.
2. Membuat *folder*
   Melakukan proses upload ke GitHub dengan kode:
3. Git clone <link repository>
4. ls : menampilkan daftar file dan direktori
5. Cd <nama repository>
6. Membuat branch
7. Git branch : menampilkan branch
8. Git add . : menambahkan file
9. Git status : mengecek status file yang telah ditambahkan (fungsi git status fleksibel sesuai dengan inputan sebelumnya)
10. Git commit -m “pesan”
11. Git push origin <nama cabang>
12. Proses selesai