

## Summary Introduction to Software Engineering

### Definisi dan Scope Pengembangan Full Stack

Pengembangan Full Stack (Full Stack Development) merujuk pada pengembangan seluruh aplikasi secara end-to-end, dari sisi depan (front-end) hingga sisi belakang (back-end) dan, dalam beberapa kasus, hingga sisi klien (client-side)

#### Scope Penting Full Stack Development

- Front End Development  
Membangun antarmuka pengguna yang menarik dan interaktif menggunakan HTML, CSS, dan JavaScript.
- Back End Development  
Membangun server dan aplikasi yang berfungsi sebagai "otak" dari aplikasi, menerima permintaan dari sisi depan, memproses data, dan memberikan respons yang sesuai.
- Database Management  
Mendesain dan mengelola basis data untuk menyimpan, mengambil, dan memanipulasi data aplikasi.
- Integration of Front-End and Back-End  
Menghubungkan komponen front-end dengan layanan back-end melalui API (Application Programming Interface) untuk berkomunikasi dengan server dan database.
- Version Control and Collaboration  
Menggunakan sistem pengendalian versi, seperti Git, untuk mengelola perubahan kode dan kolaborasi dalam tim pengembang
- Mobile Development  
Beberapa Pengembang Full Stack juga memiliki kemampuan untuk mengembangkan aplikasi mobile menggunakan framework seperti React Native, Flutter.

#### Dasar-dasar Front End Web Development

- HTML
- CSS
- Javascript

#### Dasar-dasar Back End Web Development

- Bahasa Pemrograman Server-Side
- Server Framework
- Database Management

#### Dasar-dasar Database Management

- Database Management System
- Tipe Database
- Bahasa Query:

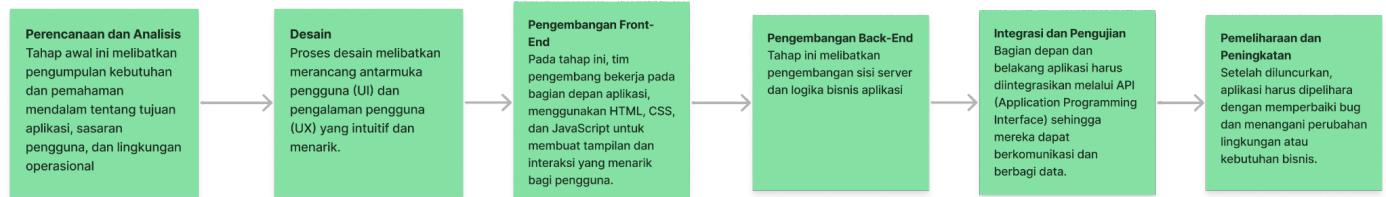
#### Dasar-dasar Mobile Development

- Platform Mobile
- IDE (Integrated Development Environment)

#### Pengembangan Aplikasi End to End

Merupakan pendekatan pengembangan perangkat lunak yang mencakup keseluruhan siklus pembuatan aplikasi, dari tahap perencanaan hingga tahap pengujian dan implementasi. Tujuannya adalah untuk menghasilkan aplikasi yang lengkap, fungsional, dan siap digunakan oleh pengguna akhir.

### Tahap tahap pengembangan aplikasi end-to-end



### Penggunaan Version Control untuk Berkolaborasi

- **Inisialisasi Proyek**
  - Tim memulai proyek dengan membuat repositori version control. Repositori ini akan menyimpan semua kode sumber, file, dan perubahan yang dilakukan selama pengembangan.
- **Pengembangan Paralel**
  - Setiap anggota tim akan memiliki salinan repositori pada komputernya sendiri. Mereka dapat bekerja secara paralel, membuat perubahan
- **Branching**
  - Version control memungkinkan pembuatan cabang (branch) yang terpisah dari kode utama. Ini memungkinkan tim untuk mengisolasi perubahan dan fitur yang sedang dikembangkan
- **Branching**
  - Version control memungkinkan pembuatan cabang (branch) yang terpisah dari kode utama. Ini memungkinkan tim untuk mengisolasi perubahan dan fitur yang sedang dikembangkan
- **Merge**
  - Setelah fitur atau perubahan selesai, cabang dapat digabungkan kembali ke cabang utama (biasanya disebut sebagai "merge").
- **Pull Request**
  - Di beberapa platform version control seperti GitHub, GitLab, dan Bitbucket, pull request adalah mekanisme yang memungkinkan pengembang untuk mengajukan perubahan mereka untuk ditinjau oleh anggota tim lain sebelum digabungkan ke cabang utama.

### Apa itu SDLC?

SDLC (Siklus Hidup Pengembangan Perangkat Lunak) adalah rangkaian proses yang terstruktur dan metodologi yang digunakan untuk mengembangkan perangkat lunak dari awal hingga selesai. SDLC terdiri dari serangkaian tahap yang saling terkait dan dilakukan secara berurutan untuk memastikan bahwa pengembangan perangkat lunak berjalan dengan baik dan sesuai dengan kebutuhan dan tujuan yang ditentukan.

## Alur Proses SDLC



## Manfaat Penggunaan SDLC

- Prediktabilitas dan Pengendalian Proyek
- Peningkatan Kualitas Perangkat Lunak
- Efisiensi Tim dan Kolaborasi
- Peningkatan Dokumentasi
- Memenuhi Kebutuhan Pengguna
- Penghematan Biaya dan Waktu
- Pengelolaan Risiko yang Lebih Baik

## Model-model SDLC

- **Waterfall Mode**  
Waterfall model adalah model SDLC yang linier dan berurutan. Setiap tahap dalam model ini harus selesai sebelum memulai tahap berikutnya. Tahapannya meliputi analisis, perencanaan, desain, pengembangan, pengujian, implementasi, dan pemeliharaan. Cocok untuk proyek dengan persyaratan yang jelas dan stabil.
- **V-Shaped Model**  
Model V-Shaped adalah model yang terkait erat dengan model waterfall, tetapi menekankan pada pengujian. Tahapan pengujian diwakili oleh garis miring "V", yang berarti bahwa setiap tahap pengembangan memiliki tahapan pengujian yang sesuai.
- **Prototype Model**  
Model Prototype adalah model pengembangan perangkat lunak yang bertujuan untuk menciptakan prototipe atau contoh awal sebelum mengembangkan versi finalnya. Model ini fokus pada pemahaman kebutuhan pengguna dan mengumpulkan umpan balik untuk memastikan bahwa perangkat lunak akhir sesuai dengan ekspektasi dan persyaratan pengguna.
- **Spiral Model**

Model ini menggabungkan elemen model spiral dengan pendekatan inkremental. Setiap siklus spiral membangun pada inkrementasi sebelumnya, menghasilkan perangkat lunak yang semakin berkembang dengan fitur yang lebih banyak setiap siklusnya.

- **Iterative Incremental Model**

Model ini melibatkan pengulangan siklus pembangunan dan peningkatan perangkat lunak dalam tahapan-tahapan kecil. Setiap iterasi menambahkan lebih banyak fitur hingga produk akhir mencapai tingkat kesempurnaan yang diinginkan.

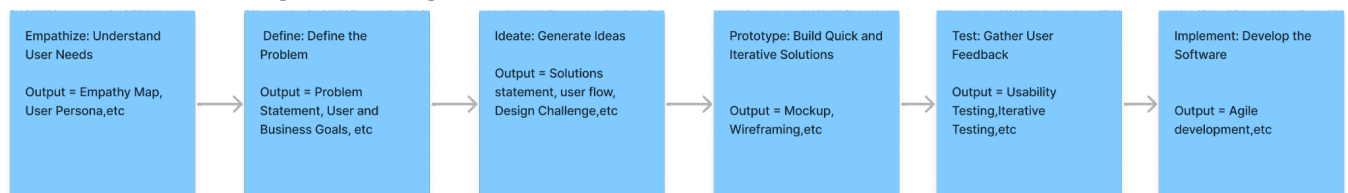
- **Big Bang Model**

Model Big Bang adalah model yang kurang terstruktur, di mana semua tahapan pengembangan dilakukan tanpa perencanaan yang detail. Pengembangan dimulai tanpa melakukan analisis dan perencanaan yang mendalam.

- **Agile Model**

Model Agile adalah pendekatan kolaboratif dan iteratif yang berfokus pada pengiriman perangkat lunak secara berkala dan inkremental. Tim bekerja dalam sprint (iterasi singkat) dan selalu terbuka untuk perubahan persyaratan pengguna.

### Tahap-tahapan Design Thinking



### Sejarah Singkat Terminal

Dengan perkembangan teknologi dan perangkat lunak, terminal tetap menjadi alat penting bagi para pengembang perangkat lunak, administrator sistem, dan pengguna teknis lainnya. Meskipun antarmuka grafis semakin canggih dan populer, terminal tetap memberikan fleksibilitas dan kekuatan untuk melakukan tugas-tugas khusus dan otomatisasi dalam lingkungan komputer modern.

### Pengertian GIT

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.

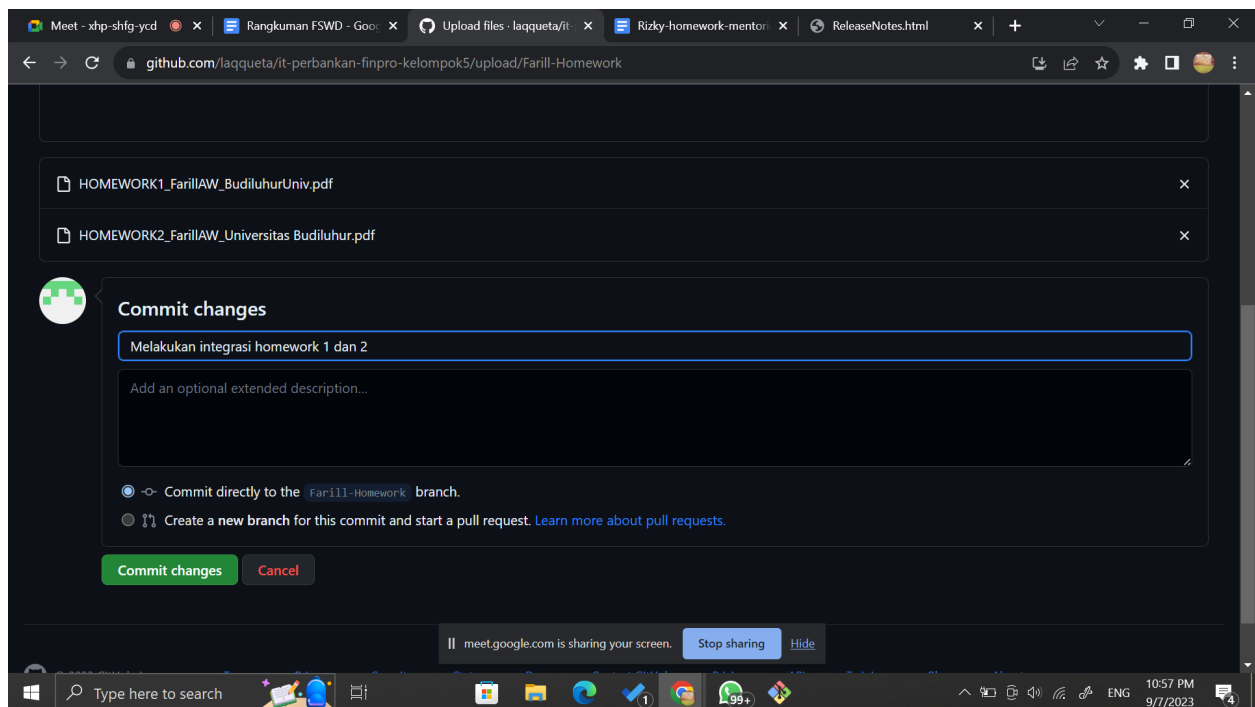
```
MINGW64/c/Users/lenovo/Downloads/IT Perbankan Cekkin
lenovo@DESKTOP-DQ3E11U MINGW64 ~/Downloads/IT Perbankan Cekkin
$ git clone https://github.com/laqueta/it-perbankan-finpro-kelompok5.git
Cloning into 'it-perbankan-finpro-kelompok5.git'...
remote: Repository not found.
fatal: repository 'https://github.com/laqueta/it-perbankan-finpro-kelompok5.git/' not found

lenovo@DESKTOP-DQ3E11U MINGW64 ~/Downloads/IT Perbankan Cekkin
$ git clone ^[[200-https://github.com/laqueta/it-perbankan-finpro-kelompok5.git
Cloning into 'it-perbankan-finpro-kelompok5'...
fatal: protocol '?[200-https' is not supported

lenovo@DESKTOP-DQ3E11U MINGW64 ~/Downloads/IT Perbankan Cekkin
$ git clone https://github.com/laqueta/it-perbankan-finpro-kelompok5.git
Cloning into 'it-perbankan-finpro-kelompok5'...
remote: Enumerating objects: 17, done.
remote: Counting objects: 100% (17/17), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 17 (delta 2), reused 10 (delta 1), pack-reused 0
Receiving objects: 100% (17/17), 993.26 KiB | 465.00 KiB/s, done.
Resolving deltas: 100% (2/2), done.

lenovo@DESKTOP-DQ3E11U MINGW64 ~/Downloads/IT Perbankan Cekkin
$
```

## Metode Clone di dalam GIT



The screenshot shows the GitHub web interface for the repository 'laqueta/it-perbankan-finpro-kelompok5'. The 'Commit changes' dialog box is open, displaying the commit message 'Melakukan integrasi homework 1 dan 2'. The dialog also shows the option to 'Commit directly to the Farill-Homework branch' selected. The 'Commit changes' button is highlighted in green.

## Penulisan Commit

Meet - xhp-shfg-yed x Rangkuman FSWD - Goo Melakukan integrasi hom Rizky-homework-mentor ReleaseNotes.html + -

github.com/laqueta/it-perbankan-finpro-kelompok5/pull/2

# Melakukan integrasi homework 1 dan 2 #2

Merged Farill344 merged 1 commit into main from Farill-Homework now

Conversation 0 Commits 1 Checks 0 Files changed 2 +0 -0

Farill344 commented now Collaborator ...

No description provided.

Melakukan integrasi homework 1 dan 2 Verified dc39d75

Farill344 merged commit 5d72ed4 into main now Revert

Pull request successfully merged and closed  
You're all set—the Farill-Homework branch can be safely deleted. Delete branch

Write Preview

meet.google.com is sharing your screen. Stop sharing Hide

Reviews  
No reviews

Assignees  
No one—assign yourself

Labels  
None yet

Projects  
None yet

Milestone  
No milestone

Development  
Successfully merging this pull request may close these

Type here to search

10:58 PM 9/7/2023

Pull and Merged