

# Programming Introduction

---

Raih Mimpi #TanpaBatas





# Topik:

---

1. History of Apps & Web
2. Product Concept
3. SDLC
4. TechStack/Tools Introduction

1

# Sejarah Program dan Web

Untuk membahas Program dan Web ada baiknya kita membuat Mind Map

**Aplikasi**

**Web**

**Computer**

**Programming  
Language**

**Internet &  
World Wide Web**

**Desktop  
Applications**

**Web  
Applications**

**Mobile  
Applications**



## Tugas Kelompok

Silahkan riset untuk mencari hal - hal penting apa saja yang terkait dengan kata kunci di sebelah..



**Computer**

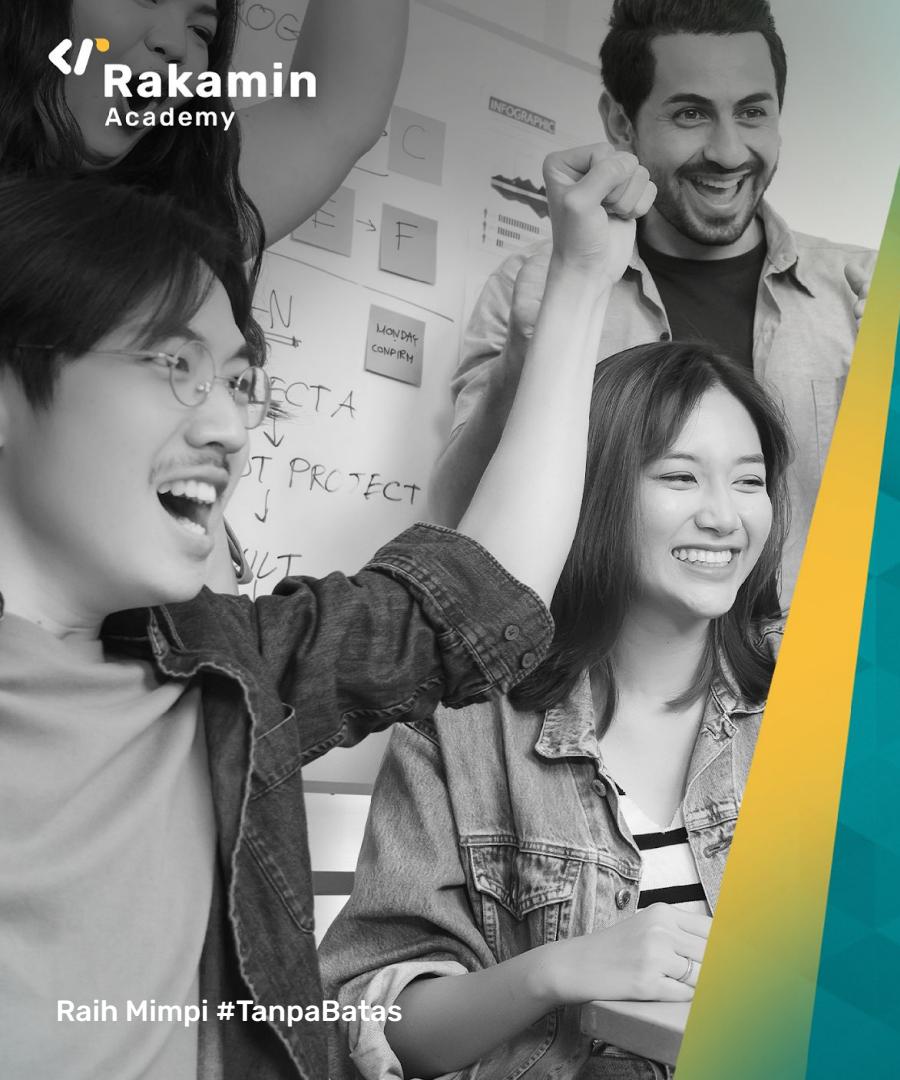
**Programming  
Language**

**Internet &  
World Wide Web**

**Desktop  
Applications**

**Web  
Applications**

**Mobile  
Applications**



Kemajuan teknologi komputer memungkinkan kita menggunakan untuk menyelesaikan masalah kita.

Contohnya, kita membuat program untuk mencatat dan mengatur pemasukan dan pengeluaran kita.

Ternyata banyak orang juga mengalami masalah yang sama dengan kita dan menginginkan program yang sama.

Demikianlah penyelesaian masalah yang kita buat menjadi bernilai ekonomi dan kemudian disebut **produk**.

2

## Konsep Produk

# Konsep yang dipakai dalam membuat “produk terbaik”

**Konsep Produk**

**Konsep  
Produksi**

# Filosofi

## Konsep Produk

Konsumen menginginkan produk yang berkualitas tinggi dan mempunyai banyak fitur.

## Konsep Produksi

Konsumen menginginkan produk yang tersedia di mana - mana dan harganya serendah mungkin.

# Cara

## Konsep Produk

Secara terus - menerus meningkatkan kualitas produk.

## Konsep Produksi

Memperbesar efisiensi produksi dan kanal distribusi.

# Tujuan

## Konsep Produk

Konsumen mendapat produk berkualitas paling tinggi.

## Konsep Produksi

Mencapai economy of scale untuk harga terendah.

# Keunggulan

Konsep Produk

Menurut kamu apa?

Konsep Produksi

Menurut Kamu apa?

# Kelemahan

Konsep Produk

Menurut kamu apa?

Konsep Produksi

Menurut Kamu apa?

# Contoh

## Konsep Produk

Menurut kamu apa?

## Konsep Produksi

Menurut Kamu apa?

# Diskusi: untuk pengembangan perangkat lunak, konsep manakah yang lebih cocok dipakai?

**Konsep Produk**

**Konsep  
Produksi**

Buat Product  
Concept  
Statement

Evaluasi  
Product  
Concept  
Statement  
terhadap  
Business  
Goals

Lakukan  
Concept  
Testing

Kembangkan  
Produk yang  
sesuai dengan  
Product  
Concept yang  
disukai  
Pelanggan

Gunakan  
Product  
Concept ke  
Dalam  
Marketing  
Strategy

# Implementasi Konsep Produk

Buat Product  
Concept  
Statement

Evaluasi  
Product  
Concept  
Statement  
terhadap  
Business  
Goals

Lakukan  
Concept  
Testing

Kembangkan  
Produk yang  
sesuai dengan  
Product  
Concept yang  
disukai  
Pelanggan

Gunakan  
Product  
Concept ke  
Dalam  
Marketing  
Strategy

**Diskusi: di bagian manakah kemampuan programming kamu  
dapat digunakan?**



Konsep Produk yang baik akan sangat membantu kita dalam mengeluarkan ide - ide produk yang lebih mungkin menarik pelanggan.

Akan tetapi, ide produk yang menarik belum tentu akan menjamin sebuah produk laku. Kualitas produk tersebut sendiri juga banyak memberikan pengaruh.

Dalam hal pengembangan perangkat lunak, untuk menjamin kualitas perangkat lunak yang dibuat berkembanglah sebuah metodologi bernama SDLC.

3

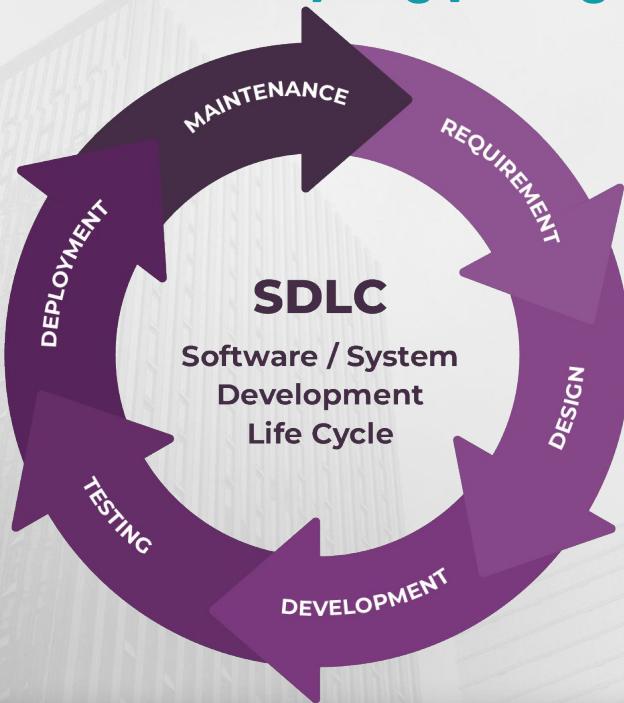
# Software Development Life Cycle

Software Development Life Cycle adalah metodologi yang menggunakan proses - proses tertentu dari awal hingga selesai untuk mengembangkan perangkat lunak.

SDLC menyediakan alur fase yang terstruktur dengan baik yang membantu organisasi membuat perangkat lunak yang berkualitas tinggi (sudah di-test dan siap untuk dipakai di lingkungan produksi) dengan waktu yang sesingkat mungkin.

## Apa itu SDLC?

**Ada banyak versi tahapan SDLC, tapi beberapa tahapan ini adalah yang paling populer**





## Analisa Kebutuhan

Dalam fase ini tim pengembang akan mengumpulkan semua informasi yang bisa didapat sehubungan dengan produk yang akan dibuat.

Berdasarkan informasi tersebut, tim akan membuat perencanaan eksekusi pembuatan produk, baik dari ruang lingkup (scope), batasan waktu (time) dan sumber daya (resource).

**Requirement  
Analysis**

**Design**

**Development**

**Testing**

**Deployment**

**Maintenance**



## Desain

Dalam fase ini tim pengembang akan membuat perancangan perangkat lunak yang akan dibuat berdasarkan kebutuhan dan persyaratan yang sudah didapat dari tahap pertama. Perancangan ini juga akan diverifikasi ulang kepada narasumber yang berkepentingan agar mendapat kepastian bahwa perangkat lunak yang dibuat sudah sesuai dengan keperluan. Contoh desain yang perlu dibuat adalah arsitektur, database, interaksi / fitur, antarmuka.

**Requirement  
Analysis**

**Design**

**Development**

**Testing**

**Deployment**

**Maintenance**



## Pengembangan

Dalam fase ini, tim pengembang akan membuat perangkat lunak yang berfungsi dan mempunyai kualitas sesuai dengan desain yang sudah dibuat sebelumnya.

**Requirement  
Analysis**

**Design**

**Development**

**Testing**

**Deployment**

**Maintenance**



## Pengetesan

Dalam fase ini, pihak - pihak yang berkepentingan akan menguji perangkat lunak yang sudah dibuat untuk memastikan bahwa perangkat lunak sudah memenuhi persyaratan kebutuhan dan kualitas.

Ada berbagai jenis testing. Penentuan pemakaian jenis yang mana sangat tergantung kepada kebutuhan pelanggan.

**Requirement  
Analysis**

**Design**

**Development**

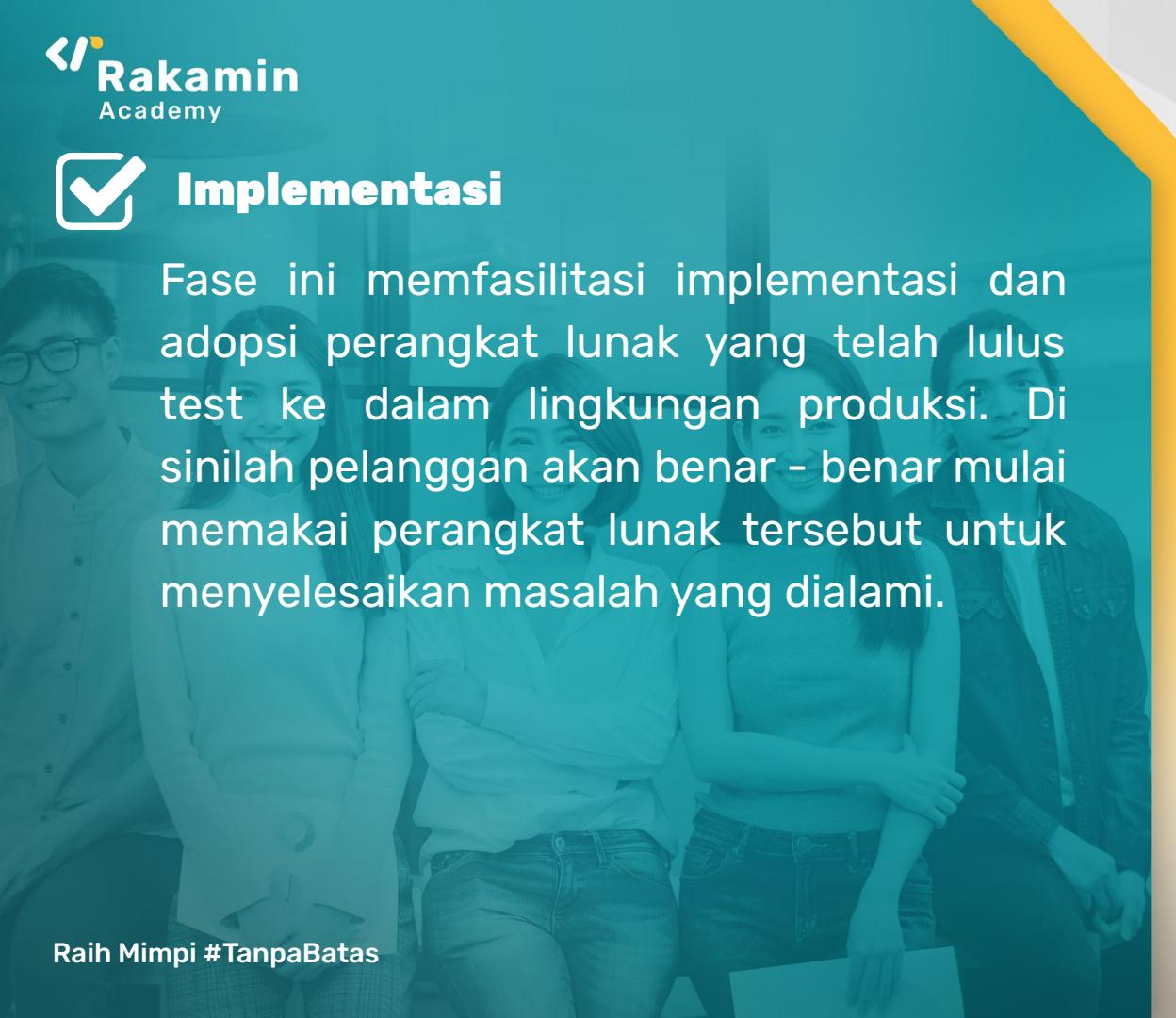
**Testing**

**Deployment**

**Maintenance**



## Implementasi



Fase ini memfasilitasi implementasi dan adopsi perangkat lunak yang telah lulus test ke dalam lingkungan produksi. Di sinilah pelanggan akan benar - benar mulai memakai perangkat lunak tersebut untuk menyelesaikan masalah yang dialami.

**Requirement  
Analysis**

**Design**

**Development**

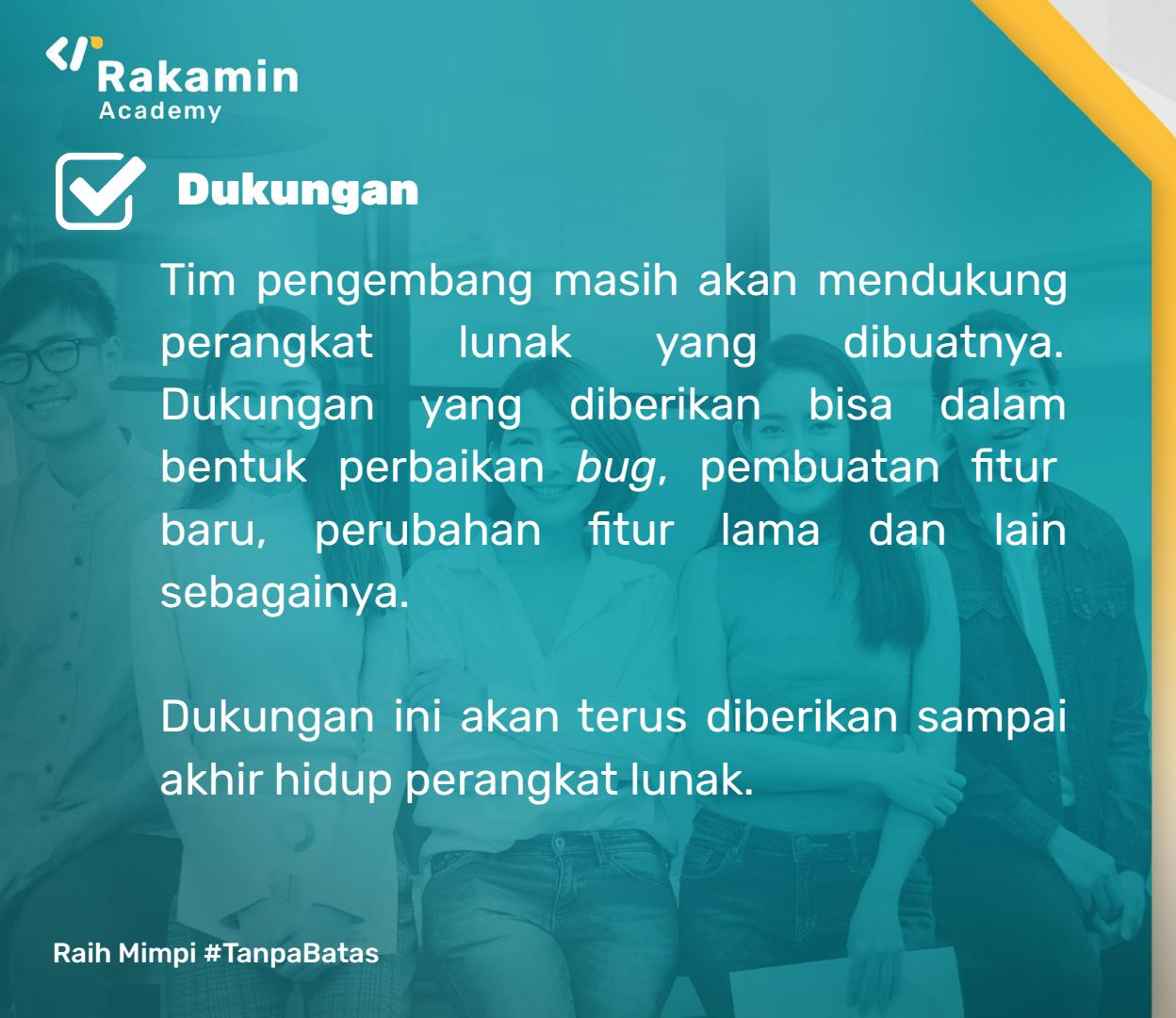
**Testing**

**Deployment**

**Maintenance**



## Dukungan



Tim pengembang masih akan mendukung perangkat lunak yang dibuatnya. Dukungan yang diberikan bisa dalam bentuk perbaikan *bug*, pembuatan fitur baru, perubahan fitur lama dan lain sebagainya.

Dukungan ini akan terus diberikan sampai akhir hidup perangkat lunak.

**Requirement  
Analysis**

**Design**

**Development**

**Testing**

**Deployment**

**Maintenance**

# Pengelompokan Model SDLC

## Adaptive

Membuat perangkat lunak dengan melakukan analisa dan perencanaan secukupnya (mis: cukup untuk satu iterasi) dan melakukan desain, pengembangan, testing dan implementasi (jika mungkin) atas kebutuhan di atas. Hasilnya akan langsung dicobakan oleh pengguna untuk kemudian menimbulkan umpan balik yang akan menjadi bahan analisa dan perencanaan iterasi selanjutnya.

## Predictive

Membuat perangkat lunak dengan melakukan analisa kebutuhan dan perencanaan secara menyeluruh terlebih dahulu, lalu melakukan desain, pengembangan, testing dan implementasi (bisa secara iteratif) sampai semua kebutuhan sudah dibuat.

# Keunggulan

## Adaptive

- Memfasilitasi kekurangan pengetahuan tentang kebutuhan serta perubahan atau pergantianya.
- Lebih mungkin meningkatkan kepuasan pelanggan karena perangkat lunak sudah bisa dipakai dalam waktu singkat dan bisa berubah sesuai kebutuhan.
- Sangat melibatkan pengguna sehingga akurasi dan prioritas perangkat lunak lebih terjamin.

Raih Mimpi #TanpaBatas

## Predictive

- Karena kebutuhan perencanaan sudah pasti, proses- proses setelahnya jadi lebih sederhana.
- Masih menawarkan fleksibilitas atas perubahan kebutuhan, terutama model iteratif.
- Sebagian besar identifikasi dan manajemen risiko dapat dilakukan di awal.

# Kelemahan

## Adaptive

- Perencanaan lebih minim karena hanya sedikit bagian di produk yang sudah jelas kebutuhannya.
- Membutuhkan anggota tim dengan T-shaped skill dan profesional.
- Sangat mungkin menghabiskan waktu membuat produk yang ternyata bukan yang dibutuhkan pelanggan jika pelanggan sendiri kurang jelas mengenai kebutuhannya.

## Predictive

- Sangat membutuhkan analisa kebutuhan dan perencanaan yang baik.
- Keseluruhan definisi perangkat lunak harus sudah jadi dulu sebelum dikerjakan.
- Jika memakai model iteratif, perlu waktu tambahan untuk menyatukan hasil dengan versi sebelumnya.

# Contoh Model

## Adaptive

Semua model berbasis Agile:  
Scrum, Kanban, Extreme  
Programming, Feature-Driven  
Development.

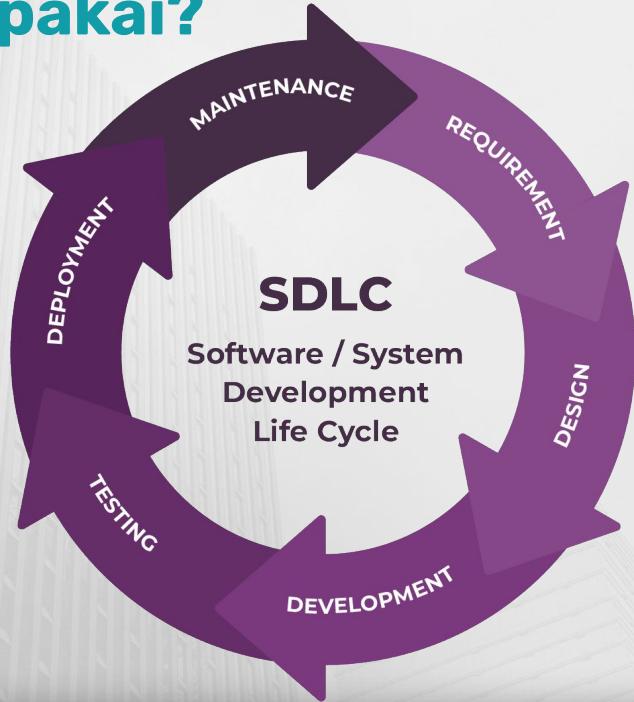
## Predictive

Waterfall, Iterative &  
Incremental, Spiral, V-Model.

## Waktunya berdiskusi!

- 1. Manakah kelompok model yang lebih baik, adaptive atau predictive?**
- 2. Berikan contoh bidang bisnis yang pengembangan perangkat lunaknya memakai masing - masing model adaptive dan predictive.**

### 3. Di bagian manakah keahlian programming kamu dapat dipakai?



4

## Tech Stack / Pengenalan Tools

# Apa itu Tech Stack?

**Tech Stack adalah gabungan beberapa framework, tools dan teknologi yang kita “tumpuk” dalam membuat sebuah produk.**

## Diskusi:

**Menurut kamu, kenapa ya kita perlu memakai tech stack tertentu?**

**Ternyata, tech stack menentukan tipe perangkat lunak yang dapat kita buat, level dari penyesuaian yang dapat dilakukan, dan resource yang diperlukan untuk membuat dan menjalankan perangkat lunak.**



## Manfaat Pemakaian Tech Stack

- Mempercepat proses pengembangan.
- Menghindarkan masalah scaling vertikal atau horizontal.
- Meningkatkan keamanan perangkat lunak.
- Membantu menentukan resource yang perlu disediakan.
- Dapat dipilih dan dikombinasikan hanya yang perlu saja.

# Contoh Tech Stack

LAMP

MEAN

**PHP/Perl/Python**  
Scripting Layer

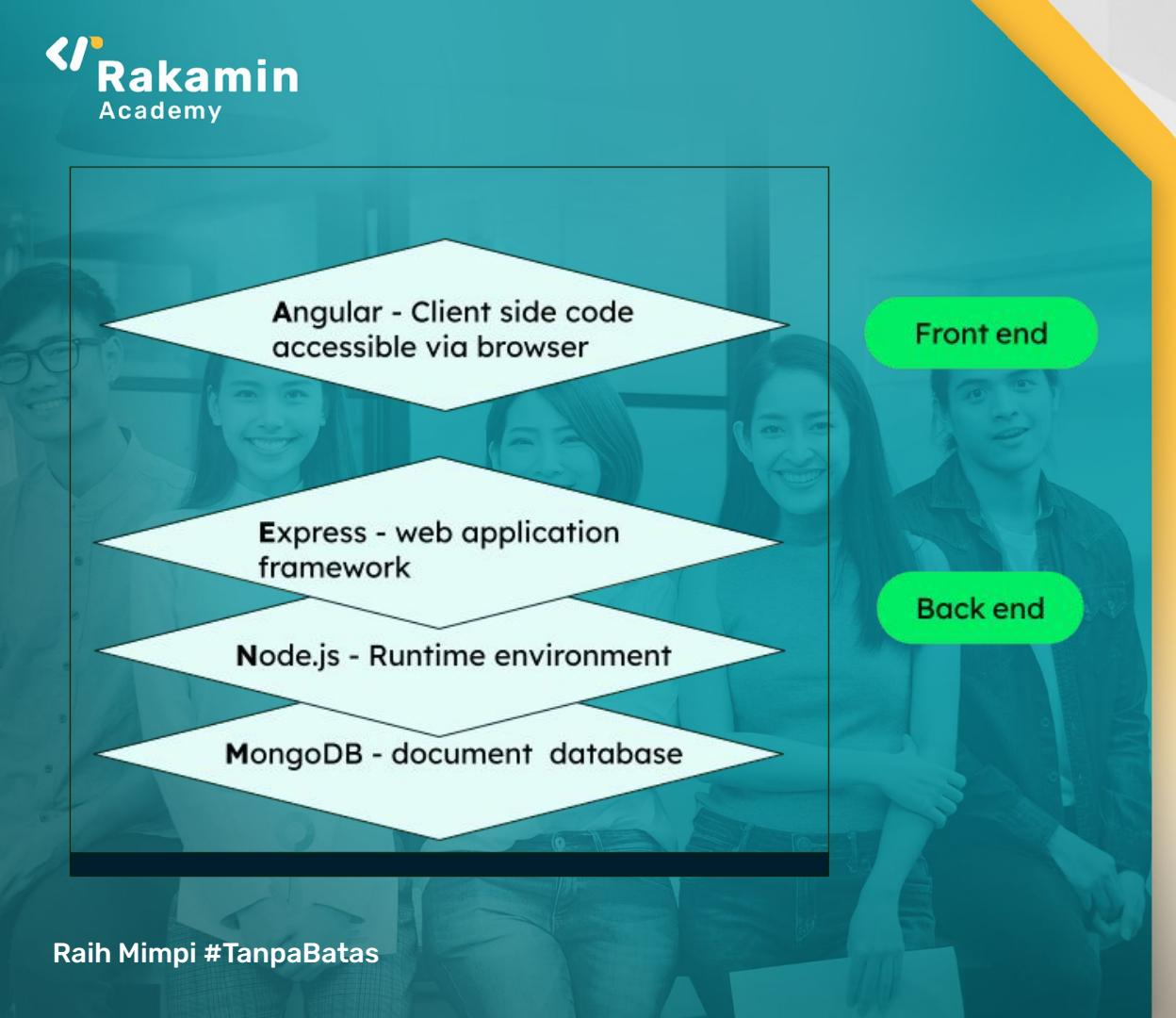
**Apache**  
Web Server Layer

**MySQL**  
Database Layer

**Linux**  
Operating System Layer

**LAMP**

**MEAN**



Angular - Client side code  
accessible via browser

Express - web application  
framework

Node.js - Runtime environment

MongoDB - document database

Front end

Back end

**MEAN**

**LAMP**

# Contoh Komponen Tech Stack

Tech Stack Component	Technologies
Frontend Framework	Vue, Angular, Bootstrap
Frontend Library	React
Programming Languages	Java, C, PHP, Python, Swift, Javascript
App & Web Framework	Spring, Django
Database	MySQL, MongoDB, PostgreSQL
Event & Messaging	Kafka
Cloud Infrastructure	AWS, Azure, Google Cloud
Virtualization	Kubernetes, Docker
Operating Systems	Windows, Linux, MacOS
Environment	Node.js, JRE

# Jadi saya bagusnya belajar tech stack apa, Pak?

**Tergantung:**

- 1. SOP Organisasi.**
- 2. Role dan kebutuhan kamu.**
- 3. Ada baiknya belajar tech stack yang ditangani rekan kerja kamu supaya kerjasamanya lebih lancar.**

# Diskusi: Menguasai Tech Stack otomatis membuat seseorang menjadi programmer handal.

# Apa Saja Hal yang Bisa Kita Simpulkan di Sesi Ini?



# Thank you