



# REKAYASA PERANGKAT LUNAK

Dosen : Mardainis



# Labor Rekayasa Perangkat Lunak Functional dan Non-Functional Testing

# Functional dan Non-Functional Testing



Dalam sebuah pengembangan aplikasi lunak, melakukan pengujian (*testing*) sangatlah penting. Pengujian seperti sebuah pintu persetujuan untuk sebuah aplikasi dapat digunakan oleh pengguna. Pengujian juga sebagai sebuah kesiapan jika aplikasi yang dibuat memang sudah sesuai dan tidak ada *issue* lagi.



Pengujian bukan hanya tanggung jawab QA (*Quality Assurance*)/*tester* saja, tetapi tanggung jawab semua tim. Namun, pengujian yang dilakukan antara tiap anggota tim berbeda. Misalnya, sebuah tim saat melakukan pengembangan sebuah aplikasi pasti memiliki *programmer* dan juga *tester*.





*Programmer* akan melakukan *testing* pada kode yang telah ia tulis, apakah sudah sesuai dengan kebutuhan atau tidak, kemudian jika sudah sesuai kebutuhan, maka *tester* akan melakukan pengujian lebih mendalam terutama melakukan pengujian kesalahan yang mungkin akan terjadi.

Secara garis besar, pengujian dalam *software development* dibagi menjadi dua tipe yaitu *Functional Testing* dan *Non-Functional Testing*.

## Functional Testing

*Functional testing* lebih berfokus pada hasil eksekusi dari proses setiap fitur di aplikasi sudah bekerja dengan baik berdasarkan kebutuhan klien. *Functional testing* tidak menitikberatkan pada *source code* tetapi lebih kepada bisnis proses yang akan berjalan. Sehingga, sangat mudah untuk melakukan pengujian ini secara manual atau secara otomatis menggunakan *automated testing*.

“

*Functional testing* dilakukan sebelum melakukan *non-functional testing*. *Functional testing* ini sering menggunakan teknik *black-box testing*.

”

Contoh: Berhasil melakukan *login* ketika inputan data valid.

# Tipe Functional Testing

Tipe Functional Testing terbagi sebagai berikut :

## 1. Unit Testing.

*Unit testing* adalah melakukan pengujian pada sebuah fitur/komponen. Tujuannya yaitu untuk melakukan validasi setiap fitur/komponen sudah sesuai.

## 2. Integration Testing.

*Integration testing* adalah melakukan pengujian pada fitur/komponen yang diintegrasikan. Testing ini bertujuan untuk memeriksa fungsional antara fitur/komponen tersebut saat berinteraksi.

## 3. System Testing.

Melakukan pengujian secara keseluruhan, yaitu semua fitur/komponen ketika diintegrasikan secara menyeluruh (*end-to-end testing*).

# Tipe Functional Testing

## 4. Smoke Testing.

*Smoke testing* dilakukan setelah pengembangan sebuah fitur/komponen selesai. *Smoke testing* bertujuan untuk memastikan fungsional fitur/komponen terpenting sudah berjalan dengan baik. Testing ini dilakukan oleh *developer* atau *tester*.

## 5. Sanity Testing.

Testing yang dilakukan setelah *smoke testing*. *Sanity testing* melakukan *testing* setelah ada perubahan/perbaikan pada *bug* yang telah ditemukan sebelumnya pada *smoke testing*.

## 6. Acceptance Testing.

*Testing* yang dilakukan oleh klien/pengguna. Hasil dari *testing* ini yaitu apakah aplikasi yang selesai dibuat sudah sesuai dengan proses bisnis dan kebutuhan atau tidak.



## Non Functional Testing

*Non-functional testing* merupakan pengujian yang melakukan verifikasi pada kebutuhan non-fungsional seperti performa dan kegunaan sebuah aplikasi. *Non-functional testing* lebih kepada meningkatkan kualitas sebuah aplikasi. Kualitas yang dimaksud yaitu bergantung pada Waktu, Akurasi, Stabilitas, Kebenaran dan *Durability* di berbagai macam keadaan.

*Non-functional testing* sulit untuk dilakukan secara manual namun dapat dilakukan dengan bantuan aplikasi *automated testing*. *Non-functional testing* menggunakan teknik *white-box testing*.

Contoh: *Loading* halaman utama/*dashboard* hanya 5 detik setelah berhasil *login*.

# Tipe Non Functional Testing

1

## Performance Testing

Pada *performance testing* akan melakukan pengujian pada beberapa aspek performa aplikasi di bawah beban kerja yang diharapkan. Aspek-aspeknya yaitu:

- **Speed** : Kecepatan aplikasi cepat atau lambat
- **Scalability** : Jumlah banyaknya pengguna yang dapat aplikasi tangani
- **Stability** : Apakah aplikasi stabil pada berbagai jumlah pengguna
- **Reability** : Cara aplikasi menangani jika melebihi beban kerja (*overload*)

2

## Load Testing

*Load Testing* adalah pengujian yang merusak aplikasi. Maksudnya yaitu suatu aplikasi akan diberikan beban kerja dengan jangka waktu yang telah ditentukan dan diharapkan mampu menangani beban tersebut.

Pada load testing, beban kerja yang diberikan yaitu beban kerja dari batas kerja yang telah ditentukan.

## Tipe Non Functional Testing

3

### Stress Testing

Pengujian ini hampir sama dengan *load testing*. Hanya saja, *stress testing* memberikan beban kerja lebih dari beban kerja yang seharusnya dapat ditangani. Ini bertujuan untuk melihat bagaimana sebuah aplikasi berjalan ketika diberikan beban kerja yang lebih.

4

### Volume Testing

Melakukan pengujian dengan cara memberikan volume/ukuran data yang besar ke aplikasi. Pengujian ini bertujuan untuk mengetahui performa aplikasi ketika volume data meningkat.

5

### Security Testing

*Security Testing* ini sangat penting karena untuk menghindari kerugian yang diakibatkan data yang tercuri oleh kejahatan *hacking*. Fokus dari pengujian ini yaitu menemukan semua celah keamanan dan kelemahan aplikasi yang dapat mengakibatkan data hilang.

## Tipe Non Functional Testing

6

### Compatibility Testing

Pengujian yang memastikan jika sebuah aplikasi dapat berjalan dengan baik pada berbagai *hardware*, *operating system*, *software*, jaringan, *browser*, *devices*, *mobile*, dan versi.

7

### Usability Testing

Pengujian yang dilakukan dengan pengguna/klien untuk melihat seberapa mudah penggunaan aplikasi yang sudah dibuat ke berbagai kalangan pengguna. Pengujian ini biasanya menggunakan sebuah *prototype*. *Usability testing* bertujuan untuk dapat mengidentifikasi sejak awal masalah kegunaan aplikasi, sehingga dapat melakukan perbaikan sebelum desain diimplementasikan.

Namun, dalam melakukan pengujian ini sering kali terjadi kesalahan yaitu terlalu lama atau terlambat dalam mempelajari *design process* dan jika menunggu desain sampai benar akan memakan waktu dan uang.

8

### Configuration Testing

Pengujian yang memeriksa aplikasi dengan berbagai kombinasi *software* dan *hardware* yang berbeda. Tujuannya yaitu untuk mengetahui konfigurasi optimal aplikasi tanpa cacat atau masalah.

# Data Conversion & Loading

Konversi data adalah konversi data komputer dari satu format ke format lainnya. Di seluruh lingkungan komputer, data dikodekan dalam berbagai cara. Misalnya, perangkat keras komputer dibangun berdasarkan standar tertentu, yang mengharuskan data berisi, misalnya, pemeriksaan bit paritas. Demikian pula, sistem operasi didasarkan pada standar tertentu untuk penanganan data dan file. Selanjutnya, setiap program komputer menangani data dengan cara yang berbeda.



# Data Conversion & Loading

Setiap kali salah satu dari variabel ini diubah, data harus dikonversi dalam beberapa cara sebelum dapat digunakan oleh komputer, sistem operasi, atau program yang berbeda. Bahkan versi yang berbeda dari elemen ini biasanya melibatkan struktur data yang berbeda. Misalnya, perubahan bit dari satu format ke format lain, biasanya untuk tujuan interoperabilitas aplikasi atau kemampuan menggunakan fitur baru, hanyalah konversi data. Konversi data mungkin sesederhana konversi file teks dari satu sistem pengkodean karakter ke yang lain; atau lebih kompleks, seperti konversi format file office, atau konversi format gambar dan format file audio.





# Data Conversion & Loading

## *Data Conversion & Loading*

Pada tahapan ini dilakukan pemindahan data dan konversi ke sistem database baru. Tahapan ini diperlukan jika sistem database baru mengganti sistem database lama. Saat ini, fitur untuk melakukan loading data dari satu sistem ke sistem baru telah sangat umum dimiliki oleh DBMS, sehingga dapat dipastikan perpindahan data dapat dilakukan dengan lancar.

## *Testing*

Sebelum dimasukkan ke dalam production, sistem yang baru dibangun akan melewati tahapan testing yang bertujuan untuk menemukan error yang mungkin terjadi. Kegiatan ini dilakukan dengan strategi dan data yang realistis. Situasi ideal dalam melakukan testing adalah tentunya kita harus memiliki backup data jika menggunakan data real, sehingga apabila terjadi error dapat dikembalikan ke posisi awal.

# Data Conversion & Loading

Beberapa kriteria yang dapat digunakan untuk melakukan evaluasi, diantaranya :

1. Learnability : Seberapa lama pengguna baru dapat menggunakan sistem baru secara produktif
2. Performance : Seberapa baik sistem merespon pekerjaan dari menggunakan
3. Robustness : Toleransi terhadap kesalahan pengguna
4. Recoverability : Seberapa baik sistem dapat melakukan recovery terhadap kesalahan pengguna
5. Adapatability : Seberapa mudah diintegrasikan dengan sistem lain.

# Data Conversion & Loading

*Operation Maintenance* Pada tahap sebelumnya, sistem database telah diimplementasi dan diuji, selanjutnya masuk dalam proses monitoring dan maintenance sistem database, yang terdiri dari beberapa kegiatan, diantaranya :

1. Monitoring terhadap performa sistem. Jika performa turun di bawah batas, maka akan dilakukan tuning dan reorganisasi database.
2. Maintaining dan upgrading sistem database. Kebutuhan baru yang tidak teridentifikasi pada tahap sebelumnya.

## METODE KONVERSI SISTEM

James A. O'Brien (2006) mengatakan bahwa operasi awal dari sistem bisnis yang baru, dapat menjadi tugas yang sulit. Hal ini biasanya memerlukan proses konversi (conversion) dari penggunaan sistem yang ada saat ini ke operasi aplikasi yang baru atau yang lebih baik. Pada saat menganalisis konversi sistem perlu dipertimbangkan pendekatan konversi yang paling bagus untuk dilakukan.

Konversi sistem adalah proses mengimplementasikan sistem baru agar dapat dioperasikan secara tepat dan benar. Terdapat beberapa teknik konversi sistem yang dapat digunakan untuk mengimplementasikan sistem baru, yaitu sebagai berikut:

## METODE KONVERSI SISTEM

- **Konversi Langsung (CUT OVER)**  
Sistem lama dihentikan, dan sistem baru mulai dioperasikan.

### **Keuntungan**

- Baik dilakukan untuk sistem yang tidak terlalu besar
- Biaya konversi sistem tidak terlalu mahal

### **Kerugian**



Resiko kegagalannya tinggi, jika sistem baru gagal dioperasikan pada waktunya.

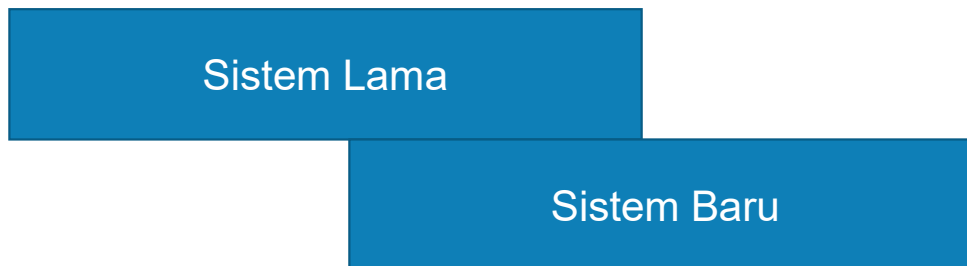
## METODE KONVERSI SISTEM

- **Konversi Paralel (PARALEL RUN)**

Sistem baru dan sistem lama dioperasikan secara bersamaan selama periode tertentu.

**Keuntungan :** Mengurangi resiko terhadap kegagalan dari sistem baru.

**Kerugian :**



Biaya konversi sistem cukup besar & mahal



## METODE KONVERSI SISTEM

- **Konversi Bertahap (PHASE IN CONVERSION)**

Penerapan masing-masing modul sistem yang berbeda secara urut

**Keuntungan :** Memperkecil kesalahan dalam sistem baru, karena dilakukan secara bertahap.

**Kerugian :**



Waktu untuk konversi sistem baru cukup lama, karena prosesnya dilakukan secara bertahap.

## METODE KONVERSI SISTEM

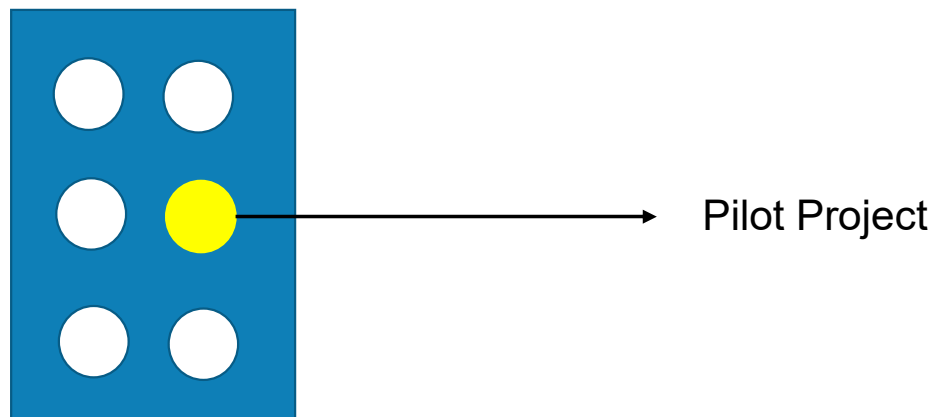
### Konversi Percontohan (PILOT APPROACH)

Dilakukan apabila beberapa sistem yang sejenis akan diterapkan di beberapa area yang terpisah.

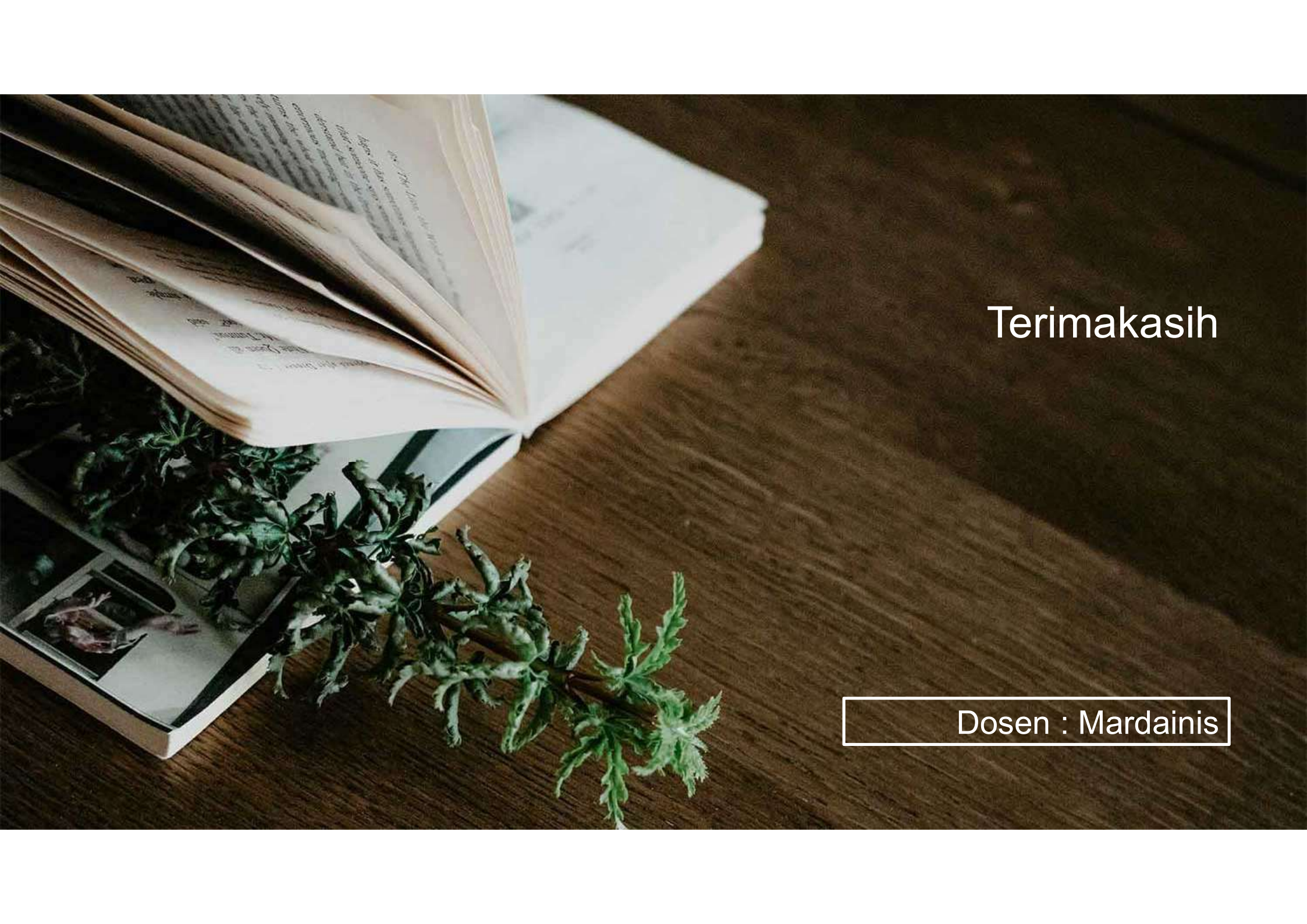
#### Keuntungan :

- Resiko kegagalan sistem hanya terletak pada area tertentu saja
- Kesalahan yang terjadi pada sistem yang baru dapat diperbaiki terlebih dahulu, sehingga kesalahan tidak terjadi pada area yang lain
- Personil di area lain dapat dilatih di area percontohan di dalam situasi yang nyata

#### Kerugian :



Proses konversi sistem menjadi sangat lama, karena harus melakukan proses ujicoba sistem dalam suatu area tertentu.



Terimakasih

Dosen : Mardainis