

METODOLOGI PENGEMBANGAN SISTEM

Pertemuan 2 — Analisis Desain Sistem Informasi (ADSI)

Memahami System Development Life Cycle (SDLC) sebagai kerangka kerja utama dalam pengembangan sistem informasi yang terstruktur dan sistematis.



Mengapa Aplikasi Bisa Gagal?

Pertanyaan Pembuka — Pertemuan 2:
ADSI



Faktor kegagalan: kebutuhan tidak jelas, komunikasi buruk, perubahan scope, kurang pengujian, anggaran habis, dan tidak ada metodologi yang tepat.



Dari Masalah ke Sistem

Alur Pengembangan Sistem Informasi

Setiap sistem lahir dari sebuah masalah nyata. Proses pengembangan mengubah masalah menjadi solusi melalui tahapan sistematis yang terstruktur.

 Problem →  Requirement →  Analysis →  Design
Mengidentifikasi masalah, mengumpulkan kebutuhan, menganalisis proses bisnis, dan merancang solusi sistem yang tepat.

 Development →  Testing →  Deployment
Membangun sistem sesuai desain, menguji fungsionalitas dan performa, lalu meluncurkan sistem ke lingkungan produksi.

 Maintenance
Memastikan sistem berjalan optimal, memperbaiki bug, dan melakukan pembaruan sesuai kebutuhan pengguna yang berkembang.

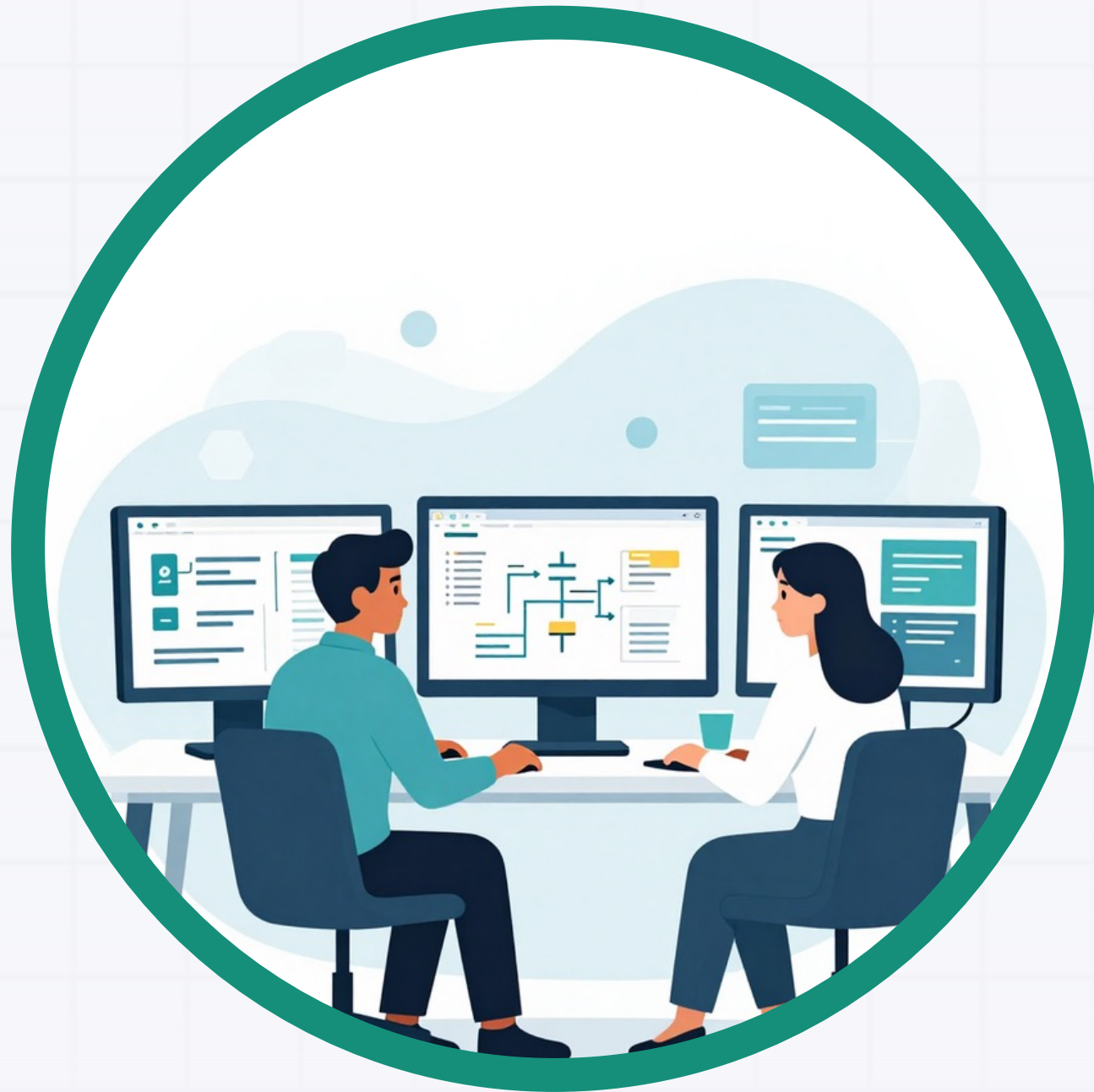


Apa itu Pengembangan Sistem?

Definisi & Ruang Lingkup



Pengembangan sistem adalah proses terstruktur untuk merancang, membangun, dan mengimplementasikan sistem informasi guna memenuhi kebutuhan organisasi secara efektif dan efisien.

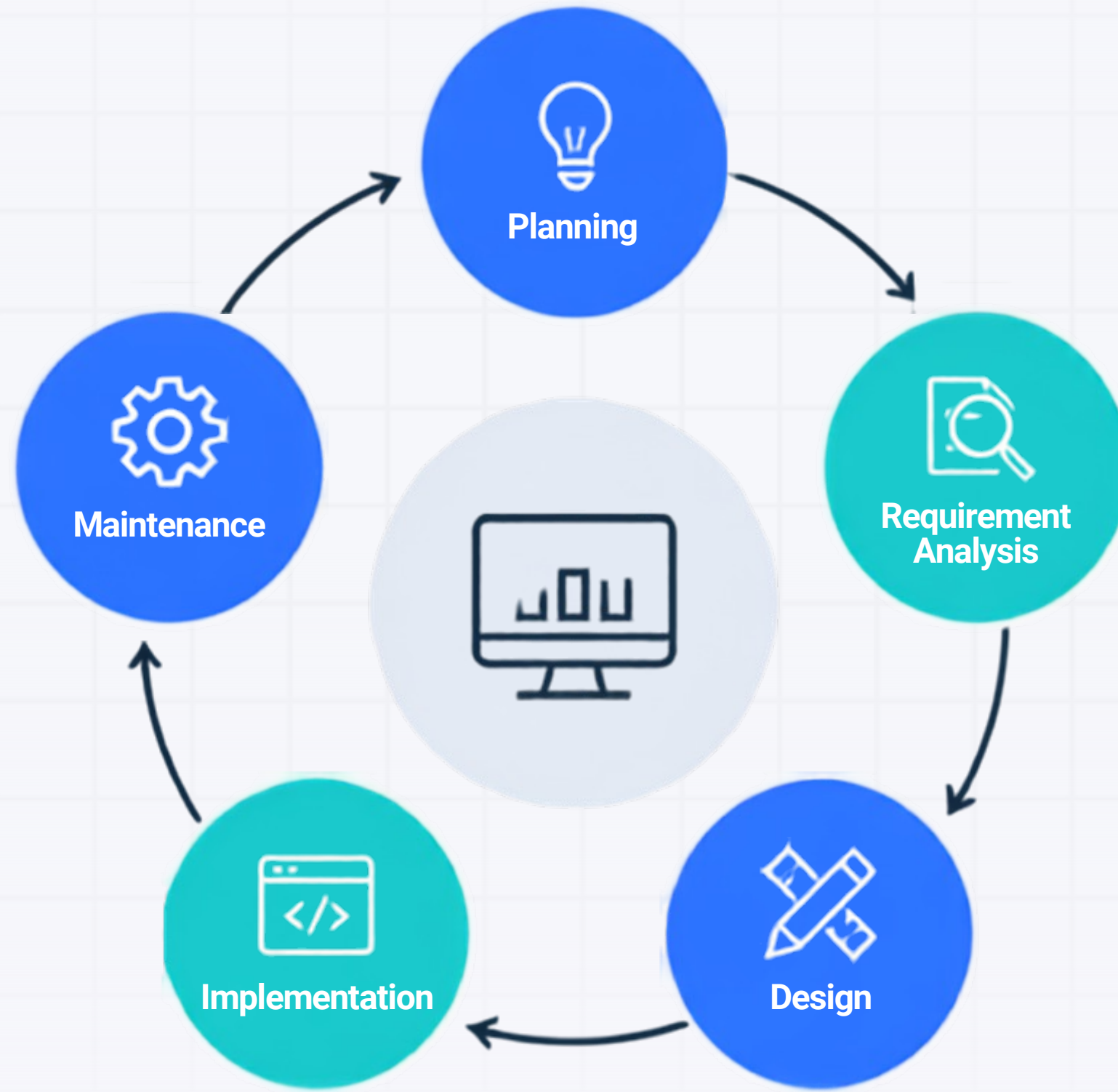


Apa itu SDLC?

System Development Life Cycle



SDLC adalah kerangka kerja terstruktur yang mendefinisikan proses perencanaan, pembuatan, pengujian, dan pemeliharaan sistem informasi secara sistematis dan terorganisir.



Mengapa Kita Membutuhkan Metodologi?

Tanpa metode, proyek mudah keluar jalur. Metodologi memberi arah, kendali, dan hasil yang berkualitas.



Metodologi yang baik membantu tim bekerja terarah, mengurangi risiko, dan menghasilkan sistem yang berkualitas.

Tahapan Utama SDLC

Siklus berkelanjutan: hasil evaluasi menjadi masukan untuk perbaikan berikutnya.



Aktivitas dan Output Setiap Tahap

Tahap		Output	
	1 Perencanaan		tujuan, ruang lingkup, dan rencana proyek
	2 Analisis		kebutuhan pengguna dan spesifikasi
	3 Perancangan		arsitektur, model proses, dan antarmuka
	4 Implementasi		kode dan konfigurasi
	5 Pengujian		hasil uji dan daftar perbaikan
6 Pemeliharaan			versi perbaikan dan evaluasi

Planning

Fase Pertama dalam SDLC



Identifikasi masalah, studi kelayakan (teknis, ekonomi, operasional), dan perencanaan proyek mencakup ruang lingkup, jadwal, anggaran, serta sumber daya yang dibutuhkan.



Analysis

Pengumpulan Kebutuhan & Analisis Sistem



Fase analisis mencakup identifikasi kebutuhan pengguna, pemodelan proses bisnis, dan evaluasi sistem yang sedang berjalan untuk menemukan celah dan peluang perbaikan.



Design Phase

Perancangan Arsitektur, Database & Antarmuka



Tahap Design menerjemahkan kebutuhan menjadi diagram sistem: arsitektur teknis, skema database (ERD), desain antarmuka (UI/UX), dan alur proses (flowchart/DFD). Output utama: dokumen desain sistem yang siap diimplementasikan oleh developer.



Development / Implementasi

Pemrograman dan Pembangunan
Sistem



Kode ditulis berdasarkan desain sistem. Tim developer membangun modul, database, dan antarmuka sesuai spesifikasi yang telah disepakati pada tahap sebelumnya.



Testing

Pengujian Sistem Secara Menyeluruh



Unit Testing: menguji modul secara individual. Integration Testing: memastikan antar modul berjalan selaras. User Acceptance Testing (UAT): pengguna memvalidasi sistem sesuai kebutuhan.

Deployment & Maintenance

Implementasi Sistem & Pemeliharaan Berkelanjutan



Deployment: sistem diluncurkan ke lingkungan produksi nyata. Maintenance: pembaruan, perbaikan bug, dan peningkatan performa dilakukan secara berkelanjutan agar sistem tetap relevan dan andal.



SDLC vs Metodologi

Memahami Perbedaan Konsep

📌 SDLC (Siklus Hidup)

SDLC adalah kerangka kerja yang mendefinisikan tahapan-tahapan dalam pengembangan sistem: Planning → Analysis → Design → Development → Testing → Deployment → Maintenance.

SDLC bersifat universal — berlaku untuk semua proyek sistem informasi tanpa terkecuali.

📌 Metodologi

Metodologi adalah pendekatan atau cara spesifik dalam menjalankan tahapan SDLC. Contoh: Waterfall, Agile, Prototype, RAD.

Metodologi dipilih sesuai konteks proyek — bisa berbeda tiap proyek tergantung kebutuhan, tim, dan kompleksitas sistem.



Mengapa Proyek Berbeda?

Faktor Kompleksitas & Kebutuhan Proyek



Setiap proyek memiliki skala, risiko, tim, dan tujuan yang berbeda. Proyek kecil butuh kecepatan; proyek besar butuh struktur. Memilih metodologi yang tepat menentukan keberhasilan sistem.



Model Pengembangan Sistem

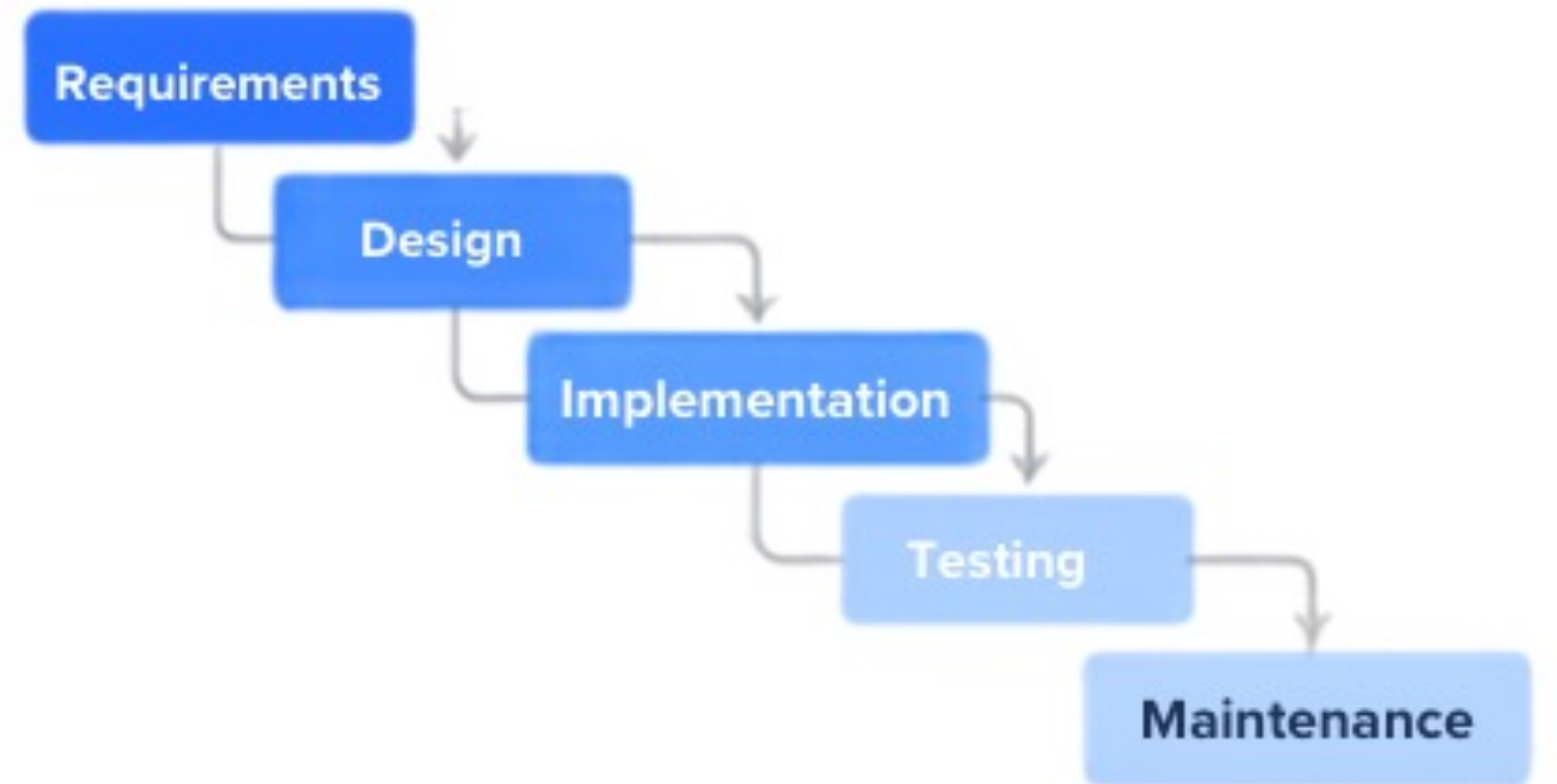


Model dipilih sesuai konteks, risiko, dan kebutuhan proyek.

Waterfall Model

Pendekatan Sekuensial & Terstruktur

Model Waterfall adalah metodologi SDLC klasik di mana setiap fase harus selesai sebelum fase berikutnya dimulai. Alur berjalan satu arah seperti air terjun.



Keunggulan: Mudah dipahami, terstruktur, dokumentasi lengkap, cocok untuk proyek dengan requirement yang sudah jelas dan tidak berubah sejak awal pengembangan.

Kelemahan: Tidak fleksibel terhadap perubahan requirement, feedback pengguna baru diperoleh di akhir, risiko kegagalan tinggi jika analisis awal kurang tepat.

Kapan Waterfall Cocok?

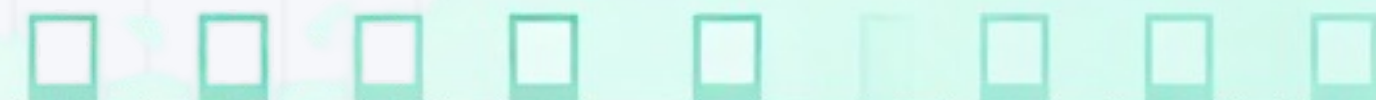
Karakteristik Proyek yang Sesuai

Waterfall bukan solusi universal. Model ini paling efektif ketika proyek memiliki ciri tertentu yang memungkinkan perencanaan dilakukan secara menyeluruh di awal sebelum pengembangan dimulai.

Requirement Jelas & Stabil: Kebutuhan sistem sudah didefinisikan secara lengkap sejak awal dan tidak akan berubah selama proses pengembangan berlangsung.

Teknologi & Scope Terdefinisi: Teknologi yang digunakan sudah dikuasai tim, ruang lingkup proyek terbatas, dan tidak ada ambiguitas dalam spesifikasi teknis.

Proyek Skala Besar & Terdokumentasi: Sistem pemerintahan, militer, atau infrastruktur kritis yang memerlukan dokumentasi lengkap, audit trail, dan kepatuhan regulasi ketat.



Prototype Model

Pendekatan Iteratif Berbasis Prototipe

Prototipe dibangun lebih awal agar pengguna dapat memberikan umpan balik nyata sebelum sistem final dikembangkan sepenuhnya.

Tahap 1 – Identifikasi Kebutuhan Awal: Analisis dan pengguna berdiskusi untuk menentukan kebutuhan dasar yang akan dijadikan dasar pembuatan prototipe pertama.

Tahap 2 – Bangun & Evaluasi Prototipe: Prototipe dibangun dengan cepat, lalu ditunjukkan kepada pengguna untuk dievaluasi dan diperbaiki berdasarkan masukan.

Tahap 3 – Refinement & Pengembangan Final: Setelah prototipe disetujui, sistem dikembangkan secara penuh berdasarkan spesifikasi yang telah disempurnakan bersama pengguna.



Contoh Prototype

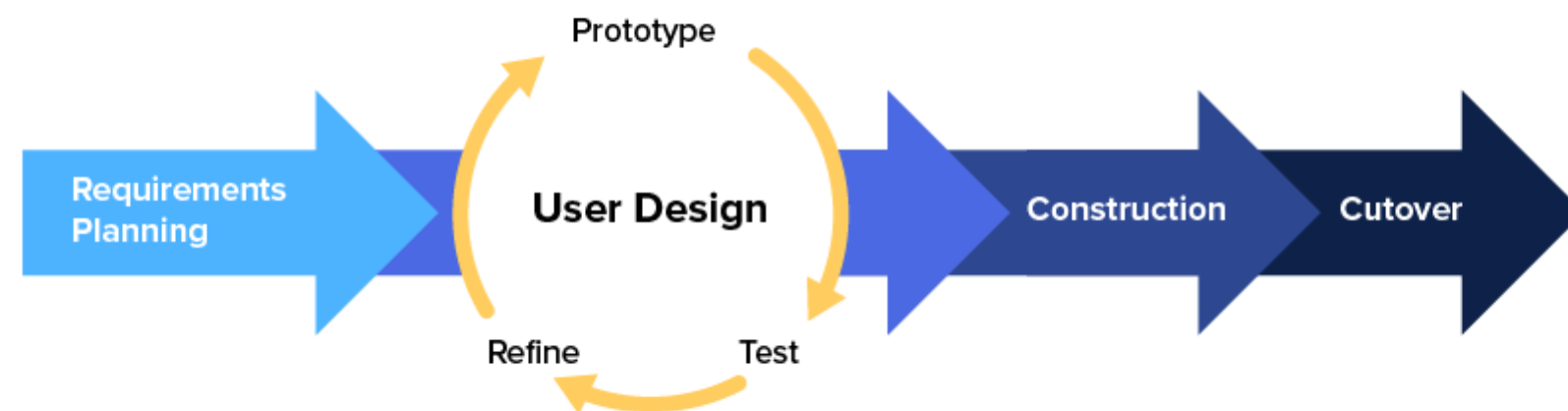
Mockup dan Wireframe Aplikasi Mobile



Prototype membantu tim dan stakeholder memvisualisasikan tampilan sistem sebelum dikembangkan. Wireframe menampilkan struktur, sedangkan mockup menampilkan desain visual yang lebih detail.



RAD Model



Fase RAD: Requirements Planning → User Design → Construction → Cutover. Setiap iterasi menghasilkan versi sistem yang lebih lengkap dan disempurnakan bersama user.

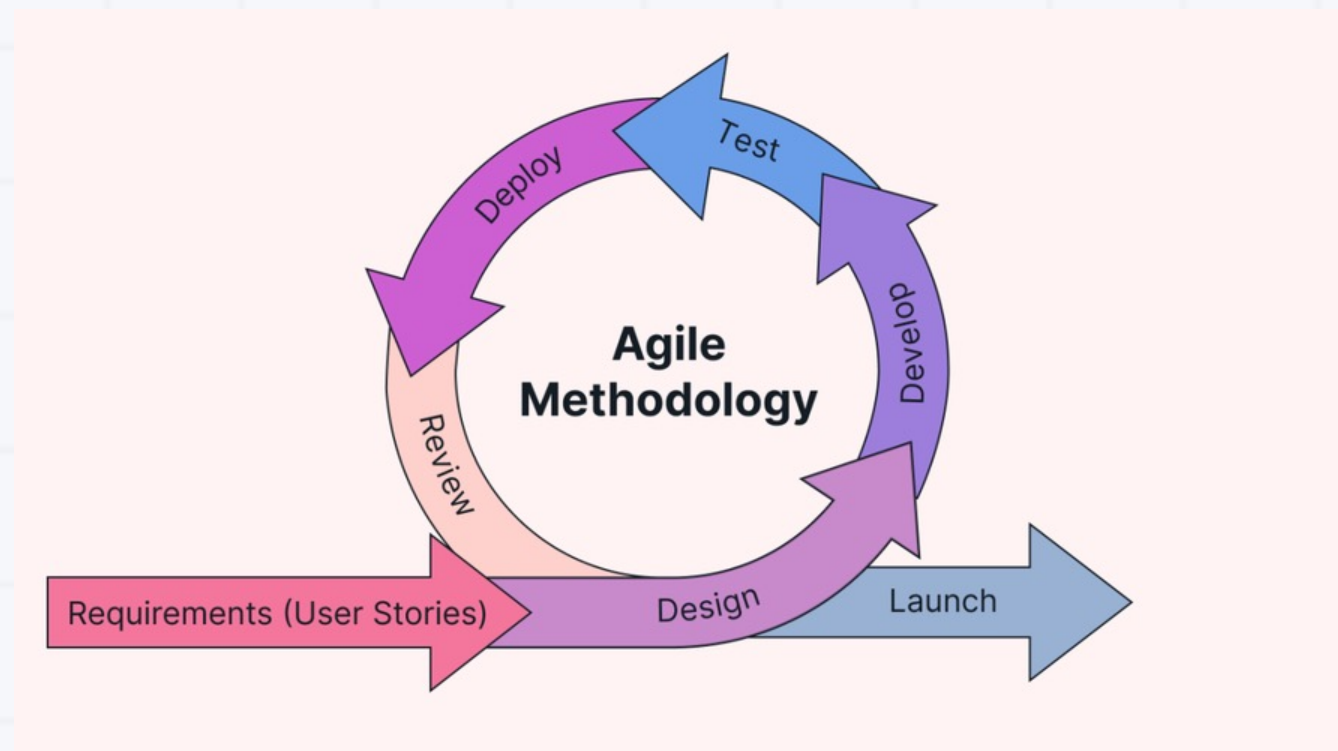
Keunggulan RAD: Waktu pengembangan lebih singkat (60–90 hari), umpan balik user langsung diterapkan, risiko kesalahan kebutuhan berkurang, dan kepuasan pengguna lebih tinggi.

RAD cocok untuk: sistem berukuran sedang, kebutuhan yang dapat dimodularisasi, tim kecil yang berpengalaman, dan proyek dengan tenggat waktu ketat serta user yang aktif terlibat.

Pendekatan pengembangan cepat yang menekankan keterlibatan aktif pengguna dan penggunaan prototipe berulang untuk menghasilkan sistem dalam waktu singkat.



Agile Methodology



Agile adalah metodologi pengembangan sistem yang bersifat fleksibel, kolaboratif, dan berbasis iterasi pendek (sprint). Cocok untuk proyek dengan kebutuhan yang terus berkembang.

Iterasi Pendek (Sprint): Pengembangan dibagi menjadi siklus 1–4 minggu. Setiap sprint menghasilkan produk yang dapat diuji dan dievaluasi oleh pengguna secara langsung.

Kolaborasi Tim & Pelanggan: Tim lintas fungsi bekerja bersama secara intensif. Pelanggan atau stakeholder terlibat aktif dalam setiap tahap untuk memastikan produk sesuai kebutuhan.

Adaptif terhadap Perubahan: Requirement dapat berubah kapan saja. Agile menyambut perubahan sebagai bagian dari proses, bukan hambatan, sehingga hasil akhir lebih relevan.



Waterfall vs Agile

Perbandingan Metodologi Pengembangan

◆ WATERFALL

- ✓ Terstruktur & terdokumentasi
- ✓ Cocok untuk proyek berskala besar dengan kebutuhan jelas
- ✓ Mudah dikelola secara manajerial
- ✗ Sulit mengakomodasi perubahan
- ✗ Feedback diterima di akhir
- ✗ Risiko tinggi jika requirement salah di awal



◆ AGILE

- ✓ Fleksibel & adaptif terhadap perubahan
- ✓ Feedback pengguna diterima setiap sprint
- ✓ Cocok untuk proyek dinamis & startup
- ✗ Dokumentasi kurang lengkap
- ✗ Sulit estimasi waktu & biaya
- ✗ Butuh keterlibatan klien yang tinggi

Faktor Memilih Metodologi

Tidak ada satu metodologi yang cocok untuk semua proyek. Pemilihan harus mempertimbangkan kondisi nyata proyek dan organisasi.

Kompleksitas & Timeline: Proyek besar dan kompleks butuh struktur ketat (Waterfall). Proyek cepat dengan tenggat pendek cocok RAD atau Agile.

Budget & Sumber Daya: Anggaran terbatas mendorong pendekatan iteratif agar risiko terkontrol. Tim kecil lebih fleksibel dengan Agile atau Prototype.

User Involvement: Jika pengguna aktif terlibat, gunakan Prototype atau Agile. Jika kebutuhan sudah jelas dan stabil, Waterfall lebih efisien.



Decision Exercise

Sistem Peminjaman Ruang dan Fasilitas Kampus

Sebuah kampus ingin membangun sistem peminjaman ruang kelas dan fasilitas (lab, aula, lapangan). Peminjaman dilakukan oleh mahasiswa dan dosen. Proses saat ini masih manual via form kertas. Tentukan: metodologi apa yang paling tepat untuk proyek ini? Berikan alasan berdasarkan faktor kompleksitas, keterlibatan user, dan deadline.



Case Discussion

Marketplace Produk UMKM

Sebuah startup ingin membangun marketplace digital untuk produk UMKM lokal. Identifikasi kebutuhan sistem, tentukan stakeholder, dan pilih metodologi pengembangan yang paling sesuai berdasarkan kondisi proyek.





Hubungkan dengan Project Kelompok

Terapkan Konsep SDLC pada Proyek
Anda



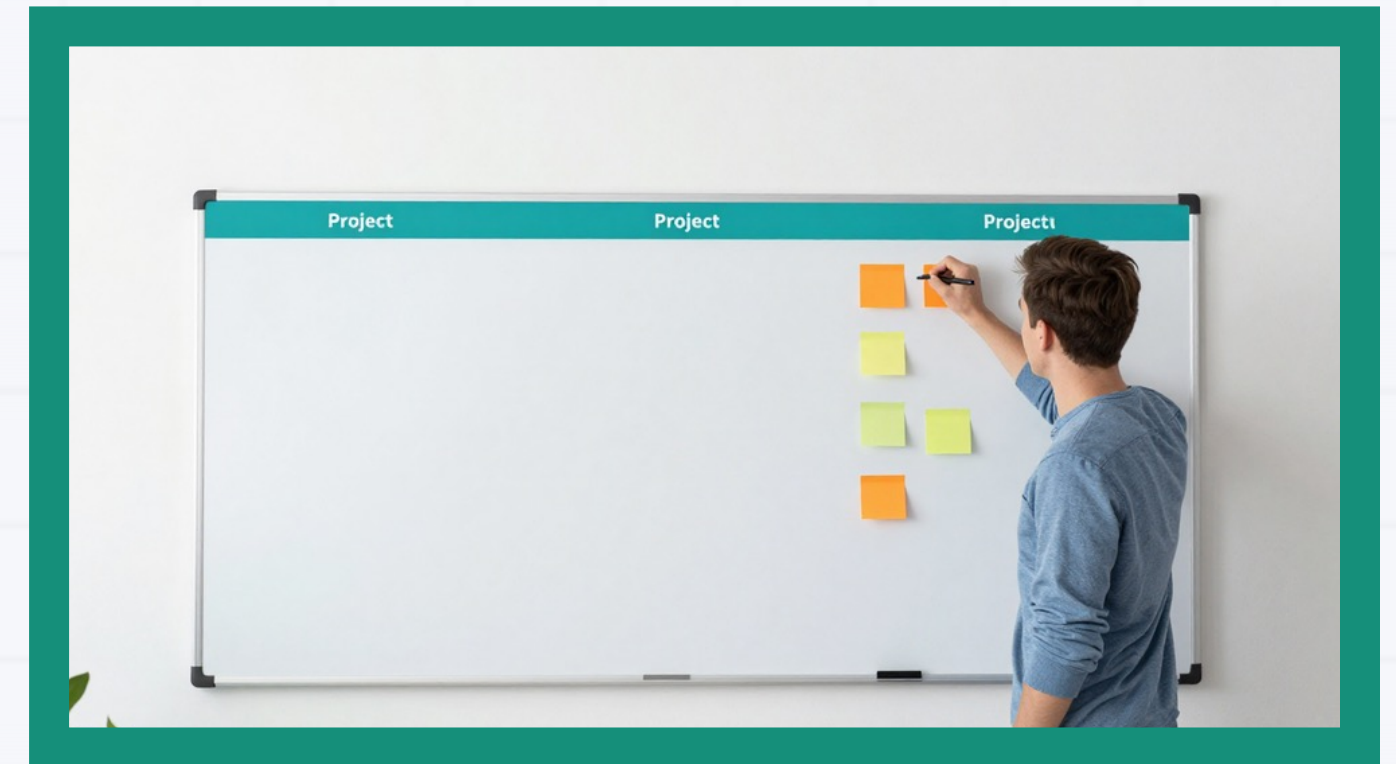
Diskusikan dalam kelompok: sistem apa yang akan dibangun?
Identifikasi masalah, tentukan requirement, pilih metodologi yang
tepat, dan rencanakan tahapan pengembangan bersama tim
Anda.

Mini Project Canvas

Template Perencanaan Proyek Kelompok

Isi canvas berikut untuk proyek kelompok Anda:

📌 Nama Sistem | ⚠️ Masalah | 👤 Target User | 📋 Requirement | 🔄 Perubahan | 🤝 User Involvement | ⌚ Deadline | ⚙️ Kompleksitas | 🛠️ Metodologi | 💡 Alasan Pemilihan



Kesimpulan

SDLC adalah kerangka kerja yang memandu pengembangan sistem melalui tahapan terstruktur: Planning → Analysis → Design → Development → Testing → Deployment → Maintenance.

Setiap metodologi memiliki kelebihan: Waterfall cocok untuk proyek stabil, Prototype untuk kebutuhan belum jelas, RAD untuk pengembangan cepat, dan Agile untuk lingkungan yang dinamis dan berubah.

Faktor pemilihan metodologi meliputi: kompleksitas sistem, keterlibatan pengguna, batas waktu, fleksibilitas kebutuhan, dan pengalaman tim pengembang.

Pemilihan metodologi yang tepat adalah kunci keberhasilan proyek sistem informasi. Tidak ada satu pendekatan yang cocok untuk semua situasi — sesuaikan dengan kebutuhan, risiko, dan konteks proyek.



Exit Ticket

Refleksi & Pemahaman Pertemuan 2



1. Apa itu SDLC dan mengapa penting?
2. Sebutkan 3 metodologi pengembangan sistem!
3. Metodologi apa yang kamu pilih untuk proyek kelompokmu?
Mengapa?



NEXT..

Pertemuan Berikutnya: Peran Profesi System Analyst

- 📌 Topik Pertemuan 3:
- Siapa itu System Analyst?
 - Peran & tanggung jawab dalam proyek SI
 - Skill yang dibutuhkan: teknis & non-teknis
 - Posisi System Analyst dalam tim pengembangan

- 📝 Persiapan untuk Pertemuan 3:
- Baca materi System Analyst dari modul
 - Cari 1 contoh lowongan kerja System Analyst
 - Siapkan pertanyaan tentang karir di bidang SI
 - Selesaikan Mini Project Canvas yang diberikan hari ini