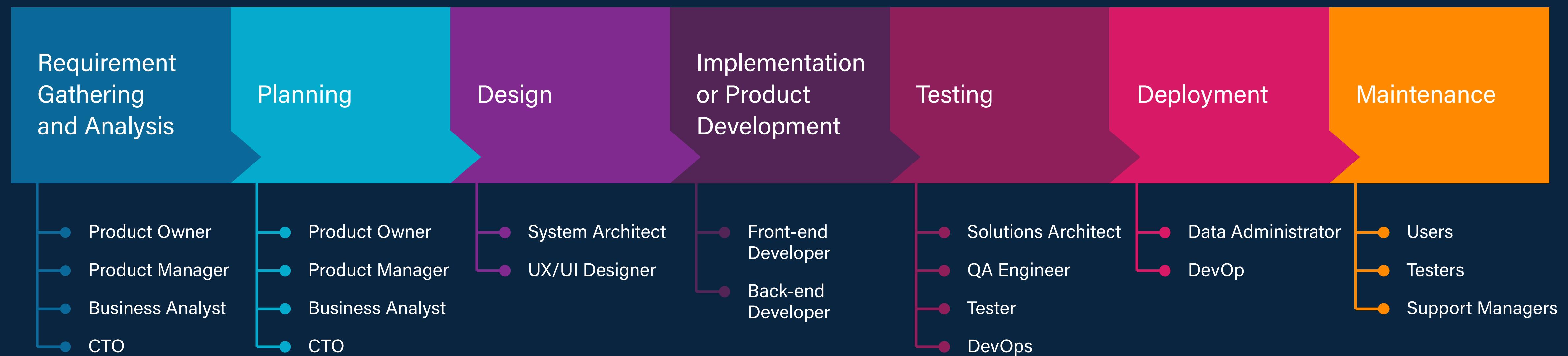


Software Development Life Cycle



apa itu **SDLC**

Pendekatan sistematis yang menghasilkan struktur bagi pengembang untuk merancang, membuat, dan menghasilkan perangkat lunak berkualitas tinggi berdasarkan kebutuhan pelanggan.

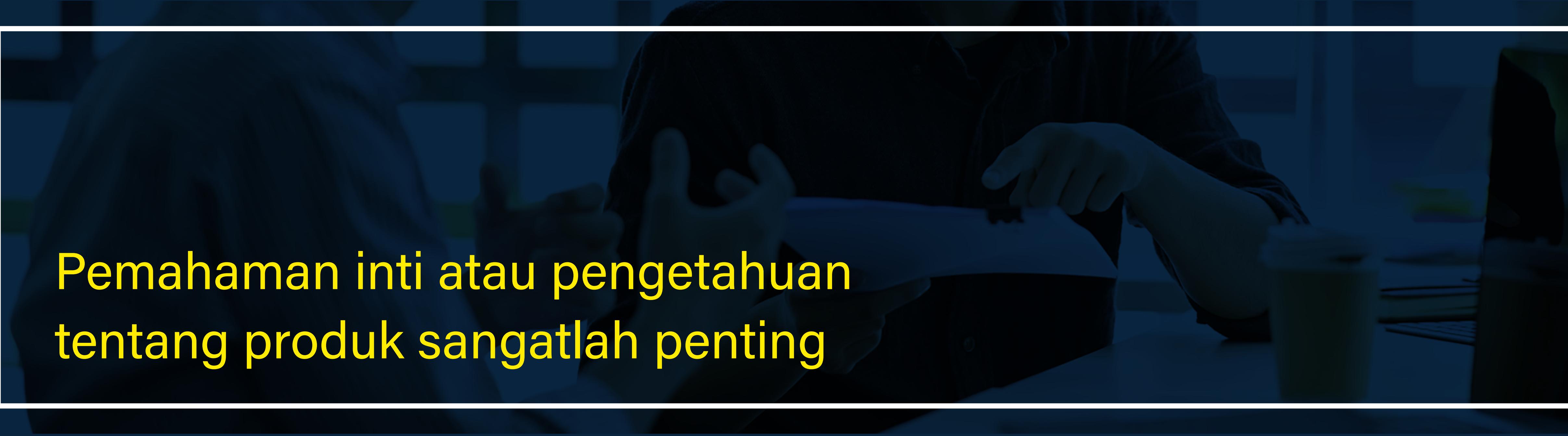


TUJUAN UTAMA

... dari proses SDLC adalah untuk menghasilkan produk yang hemat biaya dan berkualitas tinggi.

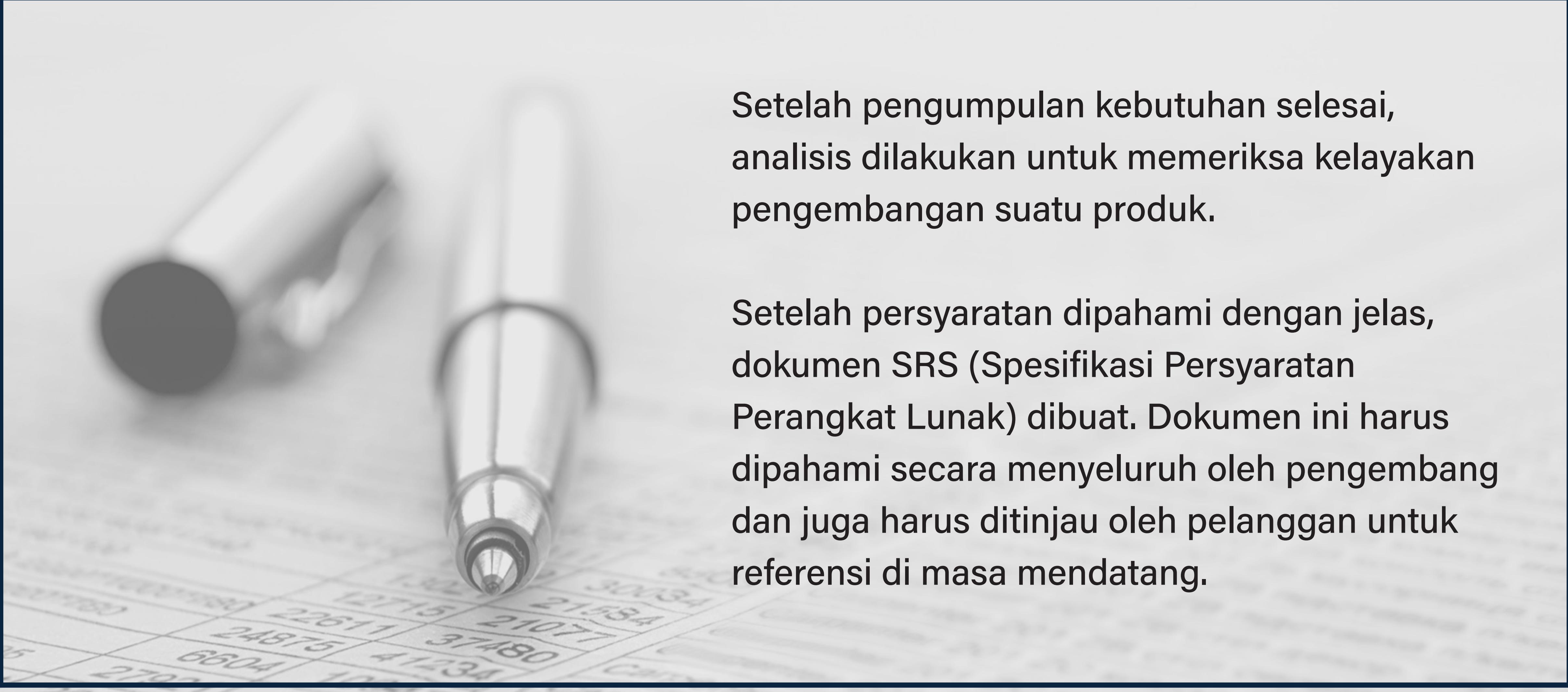
Requirement Gathering and Analysis

SDLC - fase pertama



Pemahaman inti atau pengetahuan tentang produk sangatlah penting

Fase ini biasa dilakukan oleh Analis Bisnis dan Manajer Proyek dengan mengadakan pertemuan bersama pelanggan, untuk mengumpulkan semua informasi yang relevan, dari apa yang ingin dibangun, siapa yang menjadi pengguna akhir, dan apa tujuan dari produk.



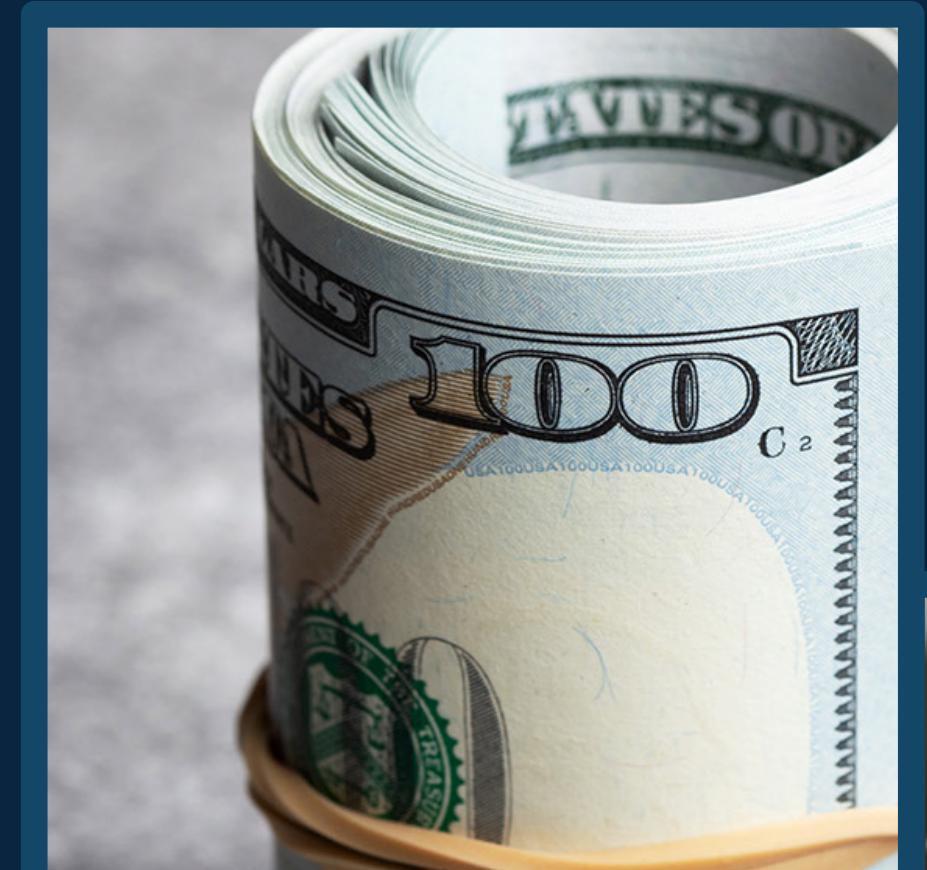
Setelah pengumpulan kebutuhan selesai, analisis dilakukan untuk memeriksa kelayakan pengembangan suatu produk.

Setelah persyaratan dipahami dengan jelas, dokumen SRS (Spesifikasi Persyaratan Perangkat Lunak) dibuat. Dokumen ini harus dipahami secara menyeluruh oleh pengembang dan juga harus ditinjau oleh pelanggan untuk referensi di masa mendatang.

SDLC - fase kedua Planning

Dalam tahap ini, tim development akan merencanakan pengembangan berdasarkan spesifikasi untuk menguraikan ruang lingkup masalah dan mengidentifikasi solusi.

**Sumber daya, biaya, waktu,
dan aspek prioritas, harus
dipertimbangkan di sini.**



SDLC - fase ketiga **Design**

Di fase inilah tim development akan membuat model cara kerja aplikasi berdasarkan lingkungan pengembangan, bahasa pemrograman, kerangka arsitektur, perangkat keras, dan juga menentukan strategi pengujian yang akan digunakan.



Selain itu, tim development juga harus memperhatikan beberapa aspek, antara lain ...

Communications

Metode atau cara aplikasi berkomunikasi dengan platform lainnya seperti server atau dengan aplikasi lain.

Programming

Tidak hanya menentukan bahasa pemrograman tapi juga termasuk metode pemecahan masalah dan tugas-tugas yang ada dalam aplikasi.

Architecture

Cetak biru/desain dan struktur proyek dari aplikasi.

User Interface

Bagaimana cara user berinteraksi dengan software dan bagaimana software tersebut dapat merespon input yang ada.

Platform

Platform di mana software akan dijalankan. Misalnya versi android, ios, linux atau game konsol.

Security

Langkah-langkah untuk mengamankan aplikasi. Misalnya membuat perlindungan kata sandi, enkripsi, dan membuat penyimpanan kredensial pengguna yang aman.

dan ...

Membuat **prototype** juga dapat menjadi bagian dari tahapan desain dalam SDLC. Prototype sendiri merupakan versi awal dari software yang dapat mendemonstrasikan ide dasar bagaimana aplikasi dapat terlihat dan bekerja.



Selain prototyping, kita juga dapat mengeksplorasi solusi dan menguji kelayakan hasil kerja kita dengan **Tracer code atau Tracer development.**



Image source:

http://www.defense.gov/dodcmsshare/newsphoto/2013-04/hires_130314-F-VU439-488.jpg

Tracer development adalah istilah yang berasal dari **tracer bullet**, yang merupakan peluru yang dibuat dengan muatan pyrotechnic kecil di ujung pangkalnya. Saat dinyalakan oleh bubuk mesiu, bahan pyrotechnic akan menyala terang dan membuat lintasan peluru terlihat. Jika pelacak mengenai target, maka peluru biasa juga mengenai target, inilah fungsi utama dari tracer bullet asli.

Tracer code dapat dianggap sebagai kode yang cukup bermanfaat untuk memberi developer arah dalam proyek tanpa jalur yang jelas. Sama seperti penembak yang mencoba untuk mencapai target dalam kegelapan, penggunaan tracer code memungkinkan developer untuk mendapatkan umpan balik langsung dan mengumpulkan spesifikasi yang dibutuhkan untuk diimplementasikan ke beberapa aspek sistem akhir dengan cepat dan berulang. Kode tidak harus berfungsi penuh, karena hanya digunakan untuk menentukan seberapa jauh kode tersebut dapat mencapai target dan membuat perubahan untuk menyesuaikan. Setelah mencapai target, menambahkan fungsionalitas akan menjadi mudah.



Keuntungan dari tracer bullet atau tracer development ...

User dapat melihat hasil progress lebih awal - pengguna dapat melihat kemajuan yang terlihat terhadap sistem mereka, terlepas dari apakah ada kekurangan fungsionalitas.

Developer dapat membangun struktur untuk pekerjaan mereka - tracer development menghasilkan garis besar spesifikasi yang terkandung dalam kode.

Membentuk satu platform integrasi - terbentuk satu lingkungan untuk menambahkan potongan kode baru setelah pengujian unit. Sistem yang terhubung end-to-end akan membuat integrasi lebih cepat dan lebih akurat.

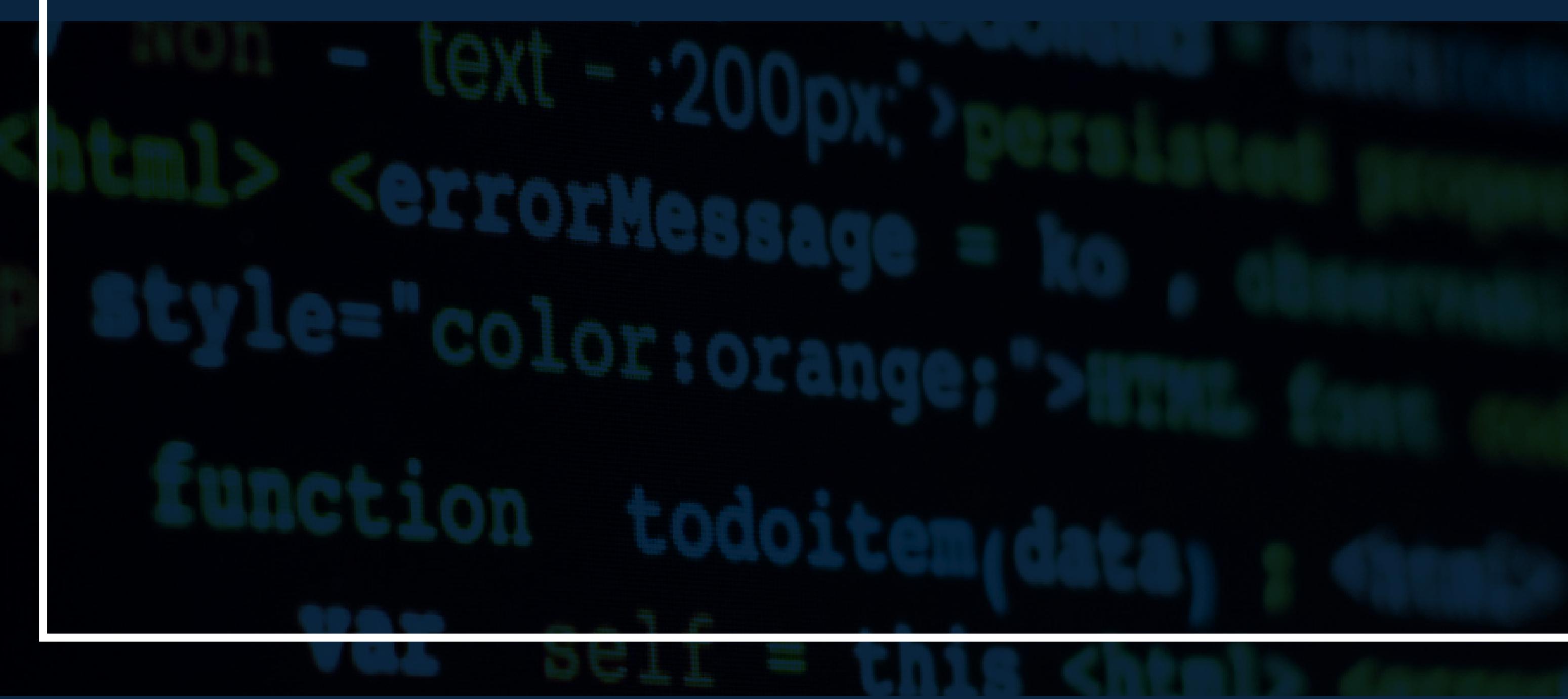
Ada sesuatu untuk didemonstrasikan - ketika bekerja dengan klien yang mengharapkan demo selama setiap pertemuan, cukup tunjukkan kode pelacak.

Progress dapat dirasakan dan lebih baik - tracer development memungkinkan developer untuk menangani beberapa kasus kecil, satu per satu. Hal ini memungkinkan pengukuran kinerja yang lebih mudah, dan dapat menunjukkan progress kepada user.

SDLC - fase keempat

Implementation or Product Development

Fase ini adalah salah satu fase terpanjang dalam SDLC. Disini dilakukan pembagian tugas dengan membuat modul atau unit dan ditugaskan ke berbagai developer. Developer kemudian akan mulai membangun seluruh sistem menggunakan pedoman pengkodean yang telah ditentukan sebelumnya, dengan alat pemrograman seperti interpreters, compilers, debugger, dan lain-lain untuk mengimplementasikan kode.



Saatnya membuat
kode program!

SDLC - fase kelima
Testing



Tahapan testing sangat penting sebelum produk/aplikasi digunakan oleh user.

Pengujian harus dijalankan dengan cara yang benar untuk memastikan apakah sistem yang dikembangkan dapat bekerja dengan optimal atau tidak, sehingga dapat meminimalisir terjadinya bug/error agar tidak ada bolak-balik antara fase pengembangan dan pengujian, yang akan mempengaruhi biaya dan waktu.

Beberapa jenis testing yang biasa dilakukan adalah ...

System testing. Pengujian ini melibatkan sistem secara keseluruhan untuk memvalidasi bahwa semua hal telah memenuhi persyaratan yang ditentukan.

Integration testing. Modul individu digabungkan dan diuji sebagai sebuah kelompok. Tim penguji akan melakukan interaksi menggunakan software dengan mengklik tombol, melakukan scroll, swipe, dan lain-lain. Pengetahuan tentang cara kerja backend tidak diperlukan.

System testing. Pengetesan produk secara keseluruhan untuk memastikan sistem berjalan sesuai dengan spesifikasi. Sering juga disebut end-to-end testing.

User acceptance testing. Merupakan pengujian terakhir yang dilakukan sebelum peluncuran resmi software. Pengujian ini biasa dilakukan user atau klien potensial guna memvalidasi bahwa software dapat menangani skenario kehidupan nyata berdasarkan spesifikasi kebutuhan.

SDLC - fase keenam Deployment

Meskipun,

Harus menunggu perusahaan untuk menentukan apakah produk akan didistribusikan ke dalam segmen terbatas dan diuji dalam lingkungan perusahaan, distribusi secara bertahap, atau distribusi langsung ke seluruh pengguna.

Dan juga,

Ada beberapa proses yang harus dilakukan untuk memastikan agar proses peluncuran produk berhasil. Hal ini biasanya melibatkan pembuatan panduan dan penyebaran dokumen seperti panduan instalasi, panduan pengguna sistem, dan lain-lain

Tapi,

Bersoraklah, karena ...

**Pengujian sudah selesai,
produk siap digunakan,
dan inilah waktunya merilis
produk untuk digunakan
oleh pelanggan.**



SDLC - fase ketujuh

Maintenance

Ini adalah fase dimana kinerja dapat **diukur** dan **dingkatkan**.

Setiap peningkatan, koreksi, dan perubahan yang diperlukan dilakukan selama fase pemeliharaan untuk memastikan sistem terus bekerja dan tetap diperbarui untuk memenuhi tujuan bisnis. Hal ini diperlukan untuk memelihara dan meningkatkan sistem dari waktu ke waktu untuk beradaptasi dengan kebutuhan masa depan.

Tiga kegiatan utama yang terlibat dalam fase pemeliharaan adalah sebagai berikut:

Perbaikan Bug

Upgrade Sistem

Peningkatan Fitur

Terdapat beberapa bentuk dan model SDLC yang dapat digunakan dalam proses development. Setiap model memiliki tahapan penerapan yang berbeda sesuai dengan strategi dan pendekatan teknis yang digunakan perusahaan.



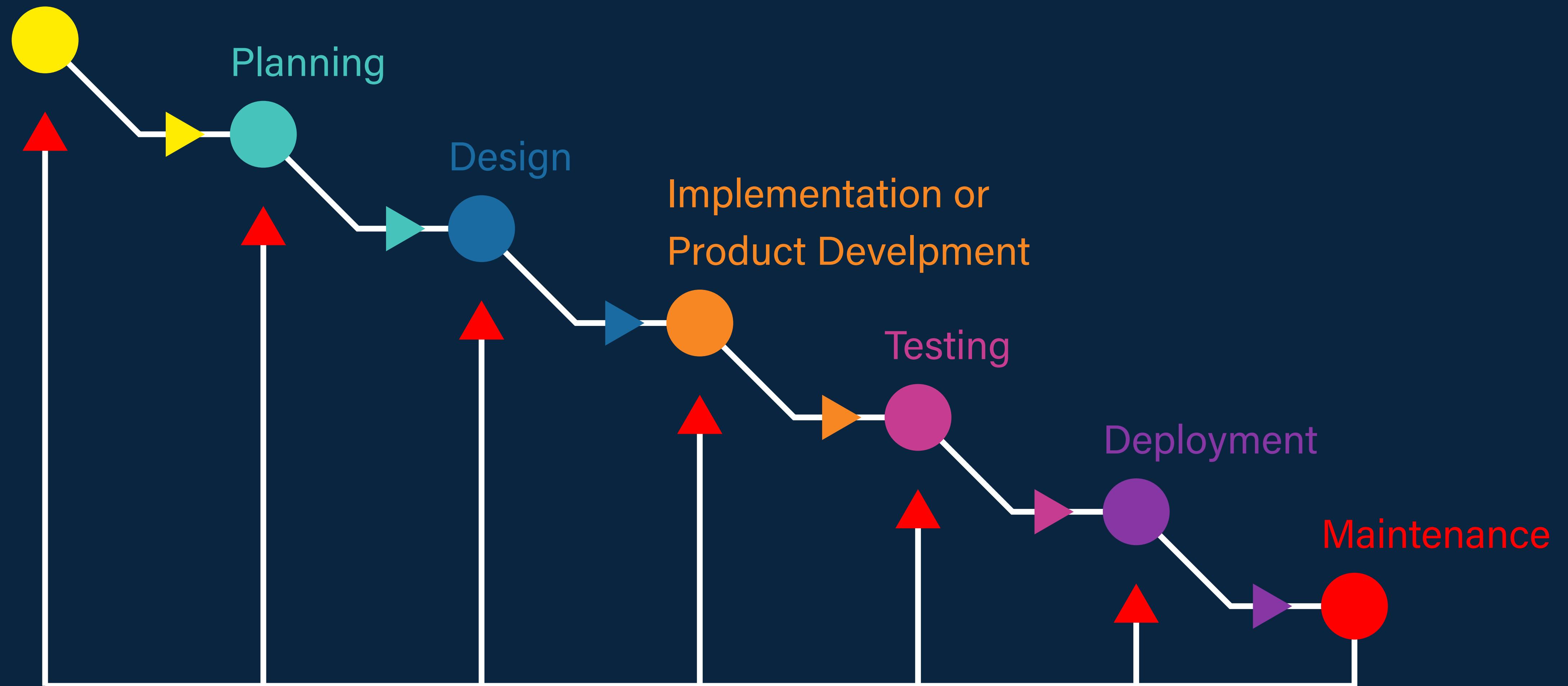
Beberapa model tersebut adalah

model pertama ...

Waterfall

adalah metodologi SDLC yang paling mudah tetapi juga yang paling kaku dari semuanya. Hal ini karena metodologi waterfall merupakan model sekuensial linier / garis lurus, sehingga tidak ada ruang gerak untuk perubahan atau penyesuaian.

Requirement Gathering and Analysis





Pendekatan waterfall paling baik digunakan ketika Requirement untuk mengembangkan suatu produk sudah terdefinisi dengan jelas dari awal, dan tidak ada requirement yang sifatnya ambigu, baik secara internal dari sisi tim produk dan bisnis, maupun secara eksternal dari sisi klien.

Model ini lebih cocok
untuk proyek dengan
skala kecil,
jangka waktu cepat,
dan resiko yang rendah.

waterfall -Kelebihan

Memiliki proses yang terurut, sehingga pengerajan dapat terjadwal dengan baik dan mudah;

Cocok digunakan untuk sistem dengan kompleksitas rendah (predictable);

Cocok digunakan untuk produk dengan spesifikasi jelas di awal, sehingga dapat meminimalisasi kesalahan;

Setiap proses yang dilakukan tidak dapat saling tumpah tindih.

waterfall -Kekurangan

Waktu pengerajan relatif lebih lama, karena harus menunggu tahap sebelumnya selesai;

Adanya kemungkinan waktu kosong bagi anggota tim karena harus menunggu anggota tim lainnya menyelesaikan tiap tahap;

Biaya yang dibutuhkan lebih mahal karena waktu pengembangan yang dibutuhkan lebih lama;

Kurang cocok untuk pengembangan proyek yang memiliki kompleksitas tinggi;

Kurang fleksibel, karena tidak mengakomodir perubahan-perubahan spesifikasi yang terjadi ketika proses telah berjalan;

Sebuah universitas ingin merancang sistem informasi untuk para alumni.

Kebutuhan universitas telah jelas, baik dari segi spesifikasi, anggota tim, dan platform yang digunakan, maka tim development memutuskan untuk menggunakan model waterfall.

Analisis kebutuhan dilakukan dengan cara melakukan interview terhadap koordinator alumni universitas. Dari interview didapatkan data-data seputar alumni, seperti total alumni yang lulus, alumni yang bekerja, dan alumni yang melanjutkan studi.

Dalam tahap desain, tim development memutuskan untuk menggunakan ERD agar lebih mudah dalam menggambarkan data yang memiliki hubungan / relasi.

Sistem informasi dibuat menggunakan bahasa pemrograman PHP dengan framework CodeIgniter.

Setelah sistem informasi selesai dibangun, pengujian dilakukan pada aspek fungsionalitas kepada petugas administrator dan alumni langsung.

Pemeliharaan akan dilakukan apabila terdapat pembaruan fitur atau memperbaiki error yang ditemukan pada saat sistem digunakan oleh pengguna.

model kedua ...

Spiral

adalah semacam kombinasi iteratif-incremental dengan penekanan pada analisis risiko yang dapat mendorong tim untuk mengadopsi elemen dari satu atau lebih model proses lainnya.

Determine
Objectives

Identify and
Resolve Risks

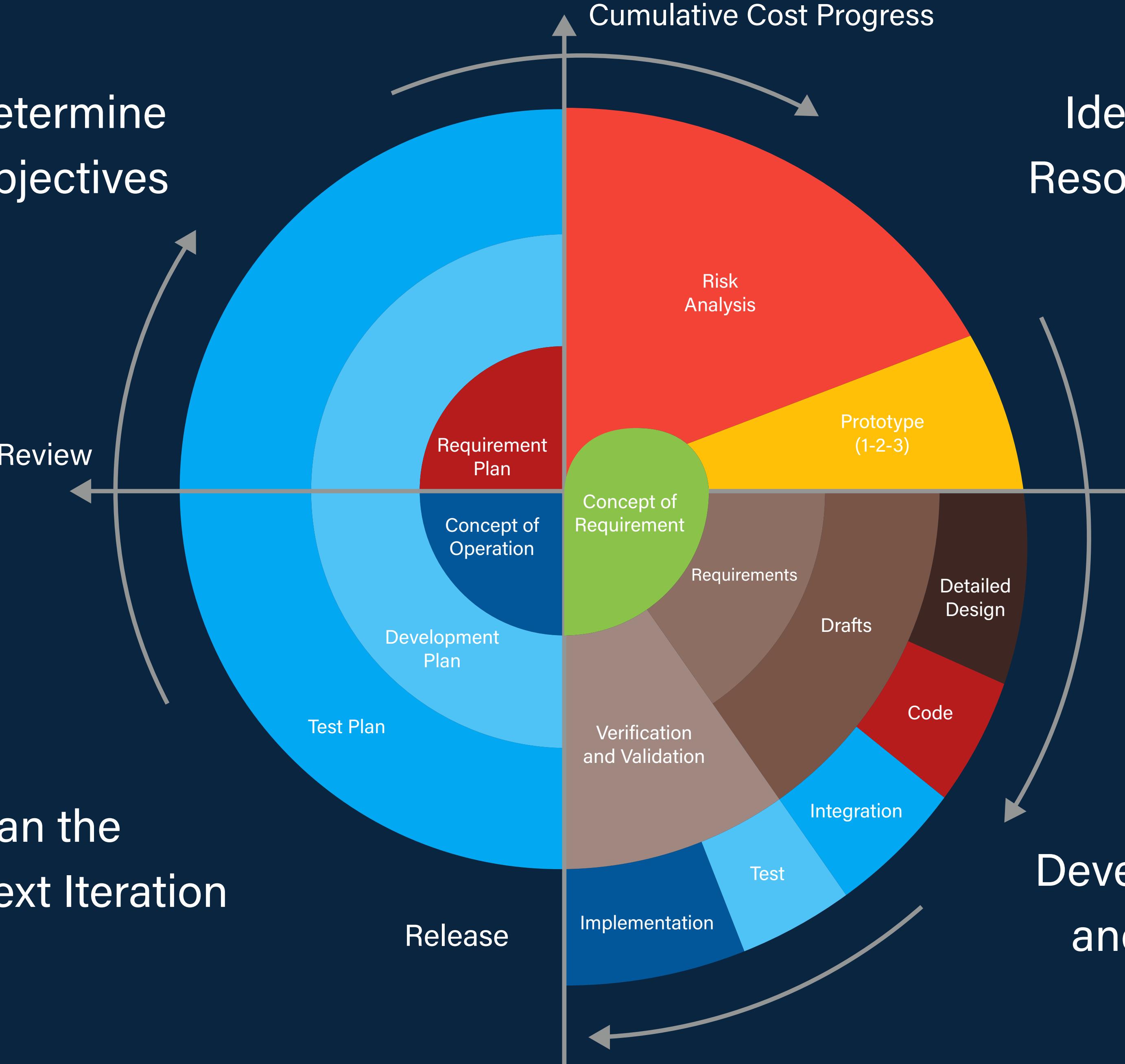
Review

Plan the
Next Iteration

Release

Development
and Testing

Cumulative Cost Progress



Tahapan Model Spiral

(secara sederhana)

Fase	Kegiatan	Hasil / Output
Determine Objectives	Persyaratan dikumpulkan dan dipelajari. Studi kelayakan. Ulasan dan panduan untuk merampingkan persyaratan.	Dokumen pemahaman requirements. Daftar akhir requirements.
	Penyampaian solusi alternatif.	

Fase	Kegiatan	Hasil / Output
------	----------	----------------

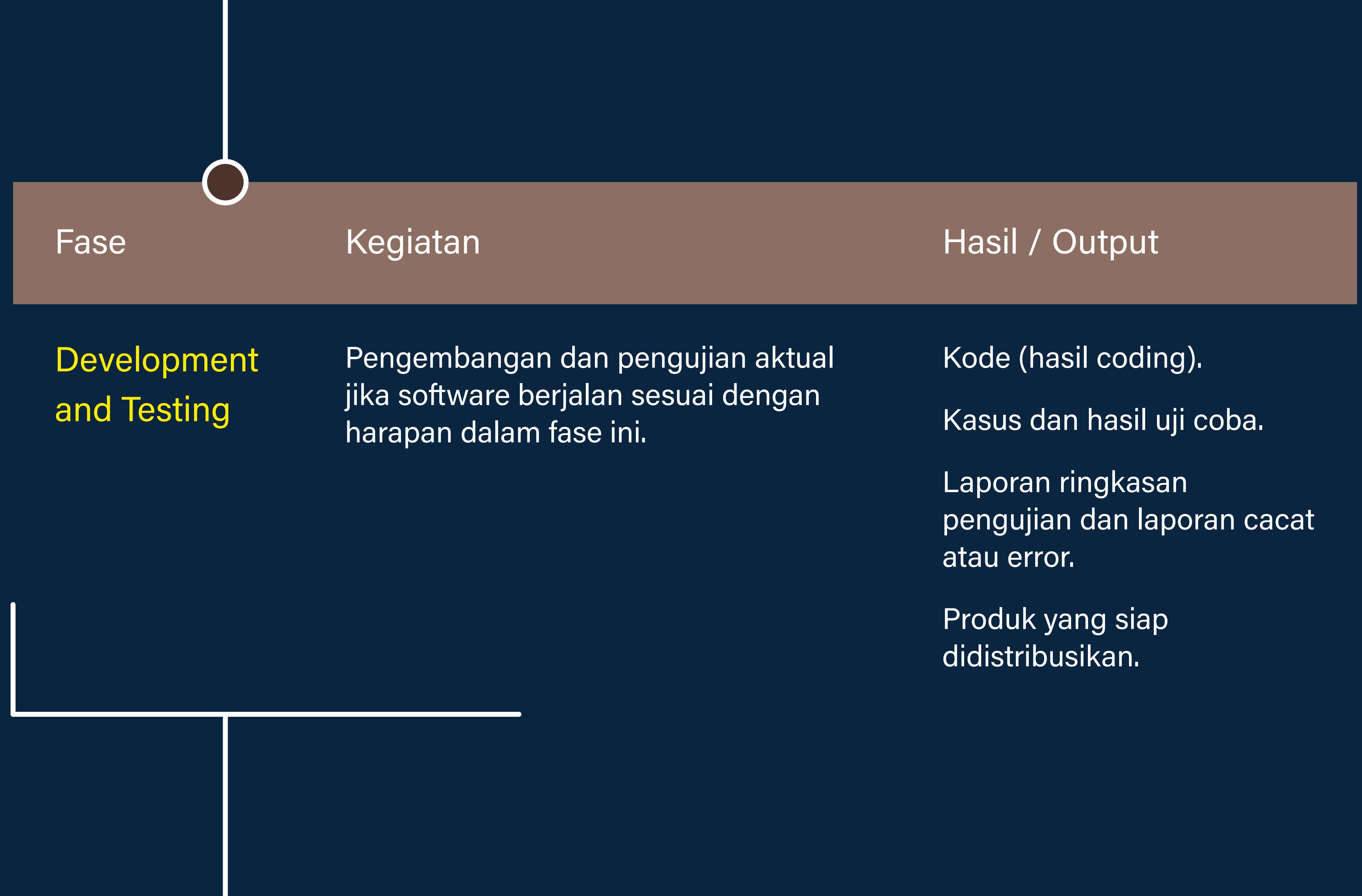
Identify and Resolve Risks

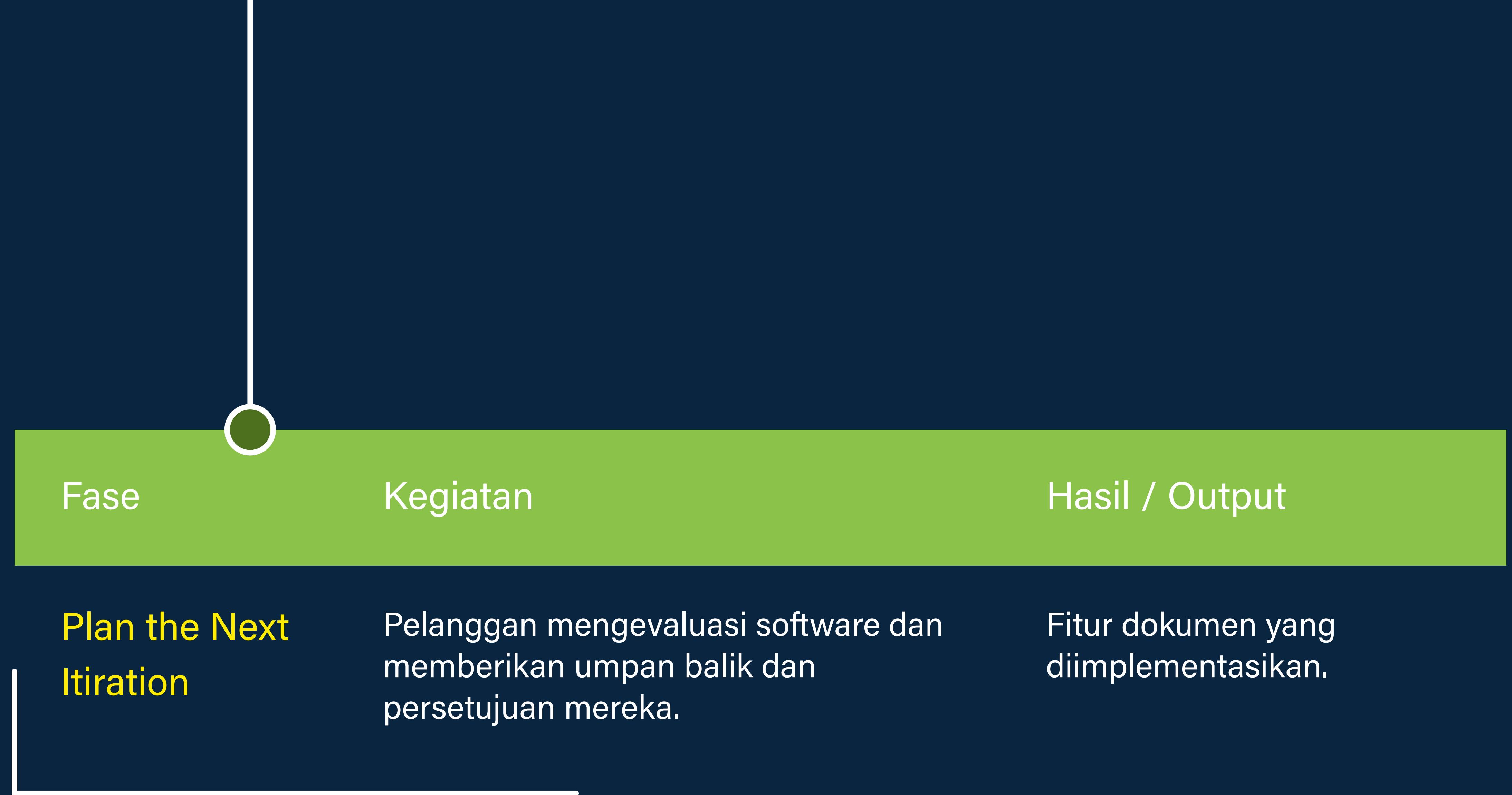
Persyaratan dipelajari dan sesi brain storming dilakukan untuk mengidentifikasi potensi risiko.

Setelah risiko diidentifikasi, strategi mitigasi risiko direncanakan dan diselesaikan.

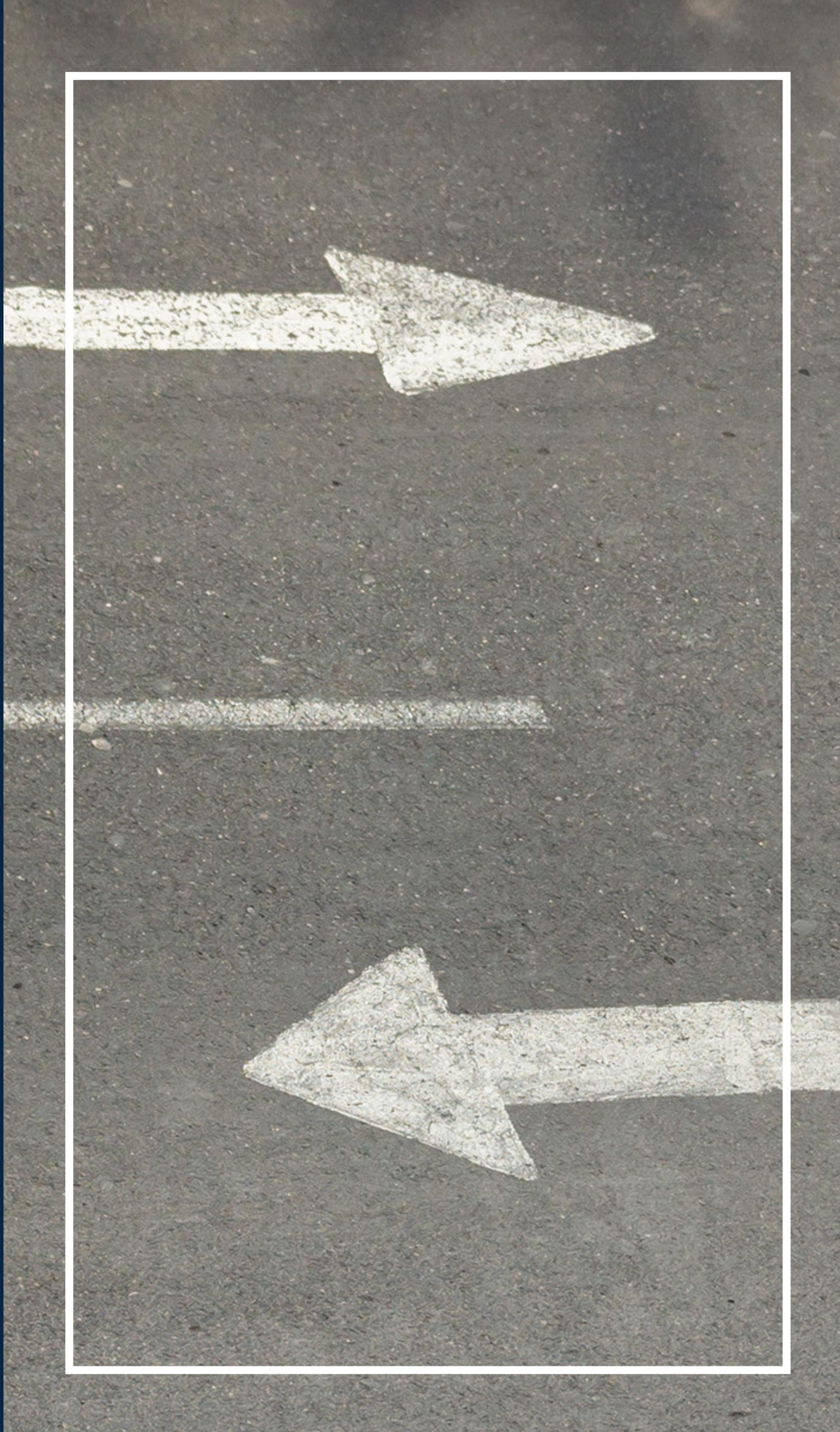
Pembuatan prototype.

Dokumen yang menyoroti semua risiko dan rencana mitigasinya.





Model ini lebih cocok untuk proyek jangka panjang berskala resiko sedang-tinggi dengan potensi perubahan requirement yang kompleks dan membutuhkan evaluasi yang detail.



Model ini juga cocok untuk produk baru yang membutuhkan feedback customer dan penambahan fitur pada kebutuhan prototipe untuk memastikan produk sesuai dengan keinginan pengguna.

spiral -Kelebihan

Model yang fleksibel, karena perubahan spesifikasi kebutuhan dapat diakomodasi pada tahap selanjutnya meskipun tiba-tiba.

Memungkinkan penggunaan prototype secara ekstensif.

Pengguna dapat melihat sistem lebih awal.

Pengembangan dapat dibagi menjadi bagian-bagian yang lebih kecil, dan bagian-bagian yang berisiko dapat dikembangkan lebih awal yang membantu dalam manajemen risiko yang lebih baik.

Sangat cocok untuk produk dengan skala yang besar dan kompleksitas tinggi.

waterfall -Kekurangan

Manajemen lebih kompleks.

Akhir dari proyek mungkin tidak diketahui lebih awal.

Tidak cocok untuk proyek kecil atau berisiko rendah dan bisa menjadi lebih mahal untuk proyek kecil.

Prosesnya rumit.

Spiral dapat berlangsung tanpa batas.

Sejumlah besar tahap perantara membutuhkan dokumentasi yang berlebihan.

Karena jumlah iterasi yang tidak diketahui di awal, estimasi waktu menjadi cukup sulit dan biaya yang dibutuhkan juga besar.

Seorang klien ingin merancang sebuah sistem e-commerce.

Pada tahap awal, tim produk menentukan spesifikasi yang dibutuhkan, estimasi waktu dan biaya, juga fitur-fitur utama yang diinginkan oleh klien.

Setelah spesifikasi dikumpulkan, tim produk dan analisa mencoba mencari solusi apa saja yang dapat diimplementasikan, beserta dengan analisa terhadap resiko dari setiap solusi tersebut.

Solusi yang dipilih kemudian dibuat prototype-nya, dan dicoba oleh klien.

Klien merasa ada fitur tambahan yang perlu ditambahkan, yaitu fitur wishlist.

Karena ada penambahan fitur ini, tim produk kembali melakukan riset, merancang spesifikasi yang dibutuhkan, menganalisa resiko, dan membuat ulang prototype-nya.

Apabila prototipe disetujui klien, tim development akan melakukan coding sampai aplikasi versi awal selesai.

Tim pengujji melakukan pengetesan, jika sudah selesai, aplikasi siap didistribusikan ke user.

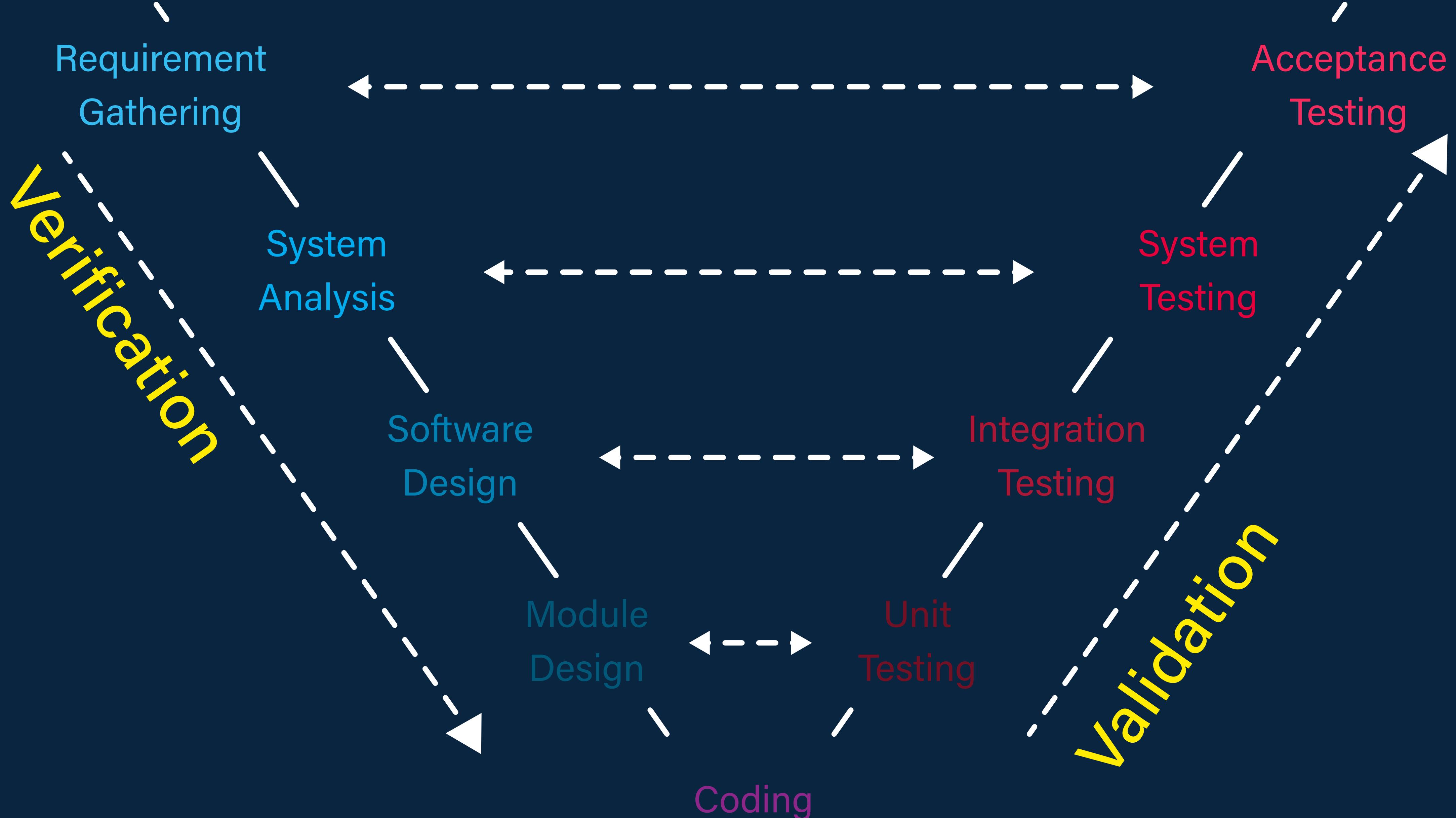
Setelah aplikasi didistribusikan, klien merasa perlu untuk menambahkan fitur pembayaran melalui dompet digital.

Tahap model spiral diulangi lagi dan versi terbaru dari aplikasi akan dikembangkan.

model ketiga ...

Model V

adalah pengembangan dari model waterfall dimana eksekusi proses terjadi secara berurutan dalam bentuk-V. Model ini dikenal juga sebagai model Verifikasi dan Validasi, dimana lebih menekankan pada tahap pengujian. Prosedur pengujian pada model ini bahkan dapat ditulis sebelum kode program dibuat.



Verification

Coding

Validation

Verification

Fase yang melibatkan teknik analisis statis (review) yang dilakukan tanpa mengeksekusi kode. Merupakan proses evaluasi fase pengembangan spesifikasi produk untuk menemukan apakah persyaratan yang ditentukan telah terpenuhi, sampai perancangan desain aplikasi oleh tim development.

Tahap verifikasi terdiri dari **analisa spesifikasi dan desain produk** yang terdiri dari high level design dan low level design, serta desain software.

Validation

Verification

Adalah proses untuk **menerjemahkan modul sistem** yang telah dirancang pada tahap desain ke dalam bentuk kode.

Coding

Baris kode yang ditulis akan melalui proses code review dan diperbaiki sehingga menghasilkan kualitas kode yang baik dan optimal sebelum dilanjutkan ke fase berikutnya.

Validation

Verification

Coding

Merupakan fase yang melibatkan **teknik analisis dinamis** (fungsiонаl, non-fungsional), dan pengujian dilakukan dengan mengeksekusi kode.

Validasi adalah proses untuk mengevaluasi perangkat lunak setelah selesainya tahap pengembangan untuk menentukan apakah software memenuhi harapan dan kebutuhan pelanggan.

Validation

model v -Kelebihan

Merupakan model yang sangat disiplin dan fase harus diselesaikan satu per satu.

V-Model digunakan untuk proyek kecil di mana spesifikasi proyek sudah jelas.

Sederhana serta mudah dipahami dan digunakan.

Model ini berfokus pada kegiatan verifikasi dan validasi di awal siklus sehingga meningkatkan kemungkinan membangun produk yang bebas kesalahan dan berkualitas baik.

Memungkinkan manajemen proyek untuk melacak kemajuan secara akurat karena selalu ada tahap pengujian pada setiap prosesnya.

model v -Kekurangan

Kurang fleksibel / kaku karena sulit mengakomodir perubahan ketika proyek sudah mulai.

Model ini tidak baik untuk proyek yang kompleks dan berorientasi objek.

Model ini tidak cocok untuk proyek di mana spesifikasinya tidak jelas dan mengandung risiko perubahan yang tinggi.

Model ini tidak mendukung fase iterasi.

Apabila sudah dalam tahap testing, akan sulit untuk kembali ke fase sebelumnya dan merubah fungsionalitas, karena dapat mengubah beberapa tahap yang ada.

Kali ini kita akan menggunakan perancangan sistem investasi saham sebagai kasus.

Karena sifatnya yang sangat sensitif dan butuh akurasi yang tinggi, maka model V digunakan untuk menghindari risiko bug terjadi di akhir.

Tim produk dan tim desain akan menyediakan desain dari aplikasi yang ingin dibangun, kemudian divalidasi kepada tim development untuk memastikan desain tersebut dapat diimplementasikan.

Tim development merancang arsitektur, memilih bahasa pemrograman yang sesuai untuk membuat aplikasi, dan juga melakukan seluruh tahapan testing untuk memastikan aplikasi sudah berjalan dengan semestinya.

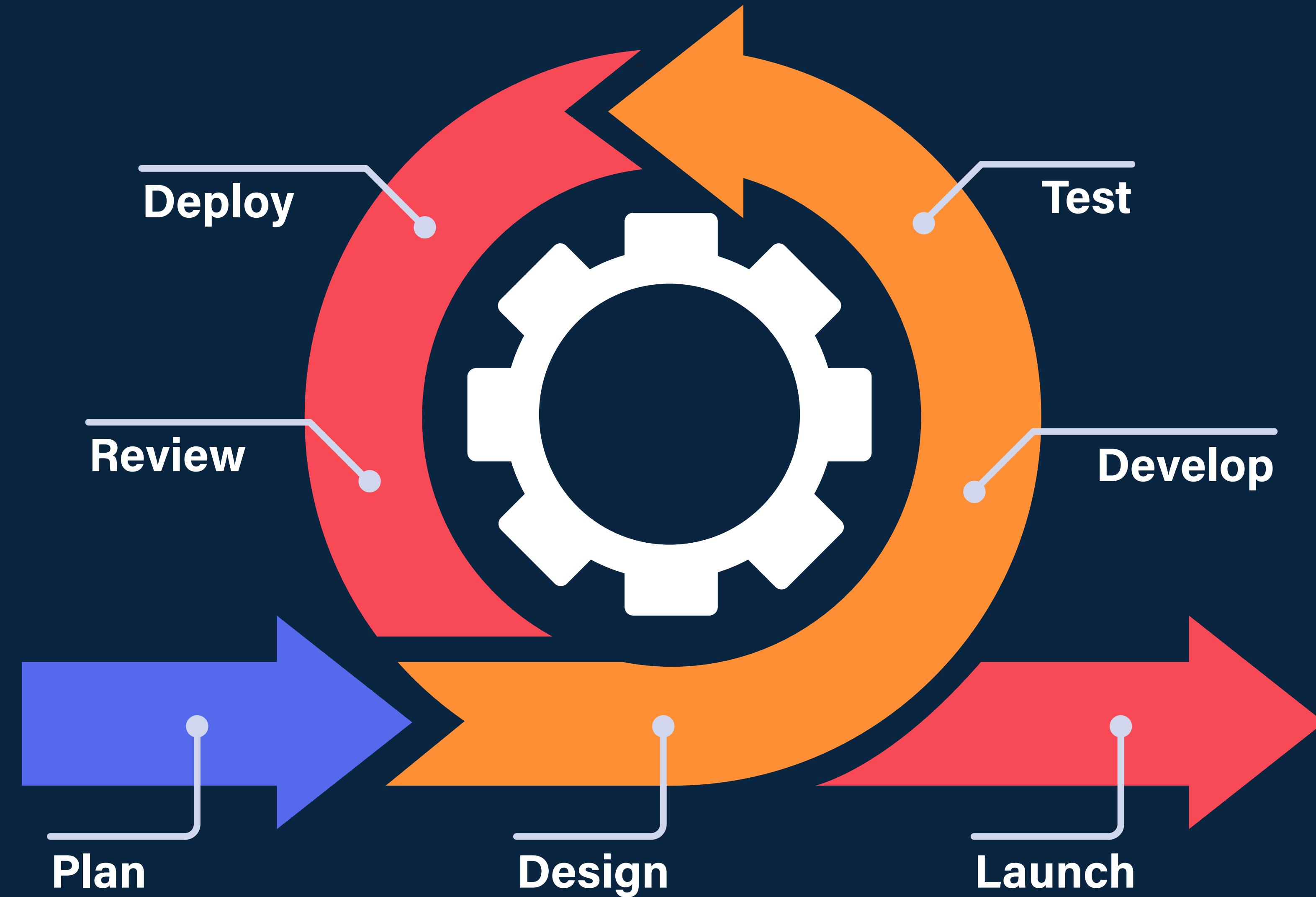
Setelahnya, tim penguji melakukan pengetesan secara menyeluruh. Dan apabila ditemukan bug, tim development akan memperbaikinya, mengulang proses tes, dan difinalisasi untuk divalidasi kembali oleh tim penguji.

Setelah seluruh proses pengetesan selesai dan sudah tidak ditemukan bug, maka aplikasi dapat didistribusikan kepada pengguna.

model keempat ...

Agile

Merupakan Metodologi yang memecah pengembangan produk menjadi iterasi (sprint) masing-masing sekitar dua minggu atau satu bulan, dengan masing-masing berfokus pada peningkatan produk. Metode ini memungkinkan perencanaan yang fleksibel, pengembangan progresif, penerapan awal, dan peningkatan konstan. Itulah sebabnya agile merupakan salah satu model yang populer digunakan oleh perusahaan-perusahaan besar.



Penggunaan Agile memiliki tujuan
menegakkan prinsip manifesto berikut ...

1. Kepuasan pelanggan melalui pengiriman software dari awal dan berkelanjutan.
2. Mengakomodasi perubahan spesifikasi selama proses pengembangan.
3. Pengiriman software yang berfungsi secara berkala.
4. Kolaborasi antara stakeholders dan developers sepanjang proyek.
5. Dukung, percaya, dan motivasi orang-orang yang terlibat.
6. Mengaktifkan interaksi tatap muka.

Part 1

Penggunaan Agile memiliki tujuan
menegakkan prinsip manifesto berikut ...

7. Software yang berfungsi adalah ukuran utama dari progress.
8. Proses agile mendukung kecepatan pengembangan yang konsisten.
9. Perhatian pada detail teknis dan desain untuk meningkatkan kecepatan.
10. Kesederhanaan.
11. Tim yang mengatur diri sendiri mendorong arsitektur, spesifikasi, dan desain yang hebat.
12. Refleksi teratur tentang bagaimana menjadi lebih efektif.

Part 2



Agile adalah metodologi yang umum digunakan untuk proyek yang kompleks karena sifat adaptifnya. Metode ini menekankan kolaborasi, fleksibilitas, peningkatan berkelanjutan dan hasil berkualitas tinggi.

Hasil kerja dari metodologi agile adalah ...

Product vision statement. Ringkasan yang mengartikulasikan tujuan produk.

Product roadmap. Pandangan tingkat tinggi tentang spesifikasi yang diperlukan untuk mencapai visi produk.

Product backlog. Diurutkan berdasarkan prioritas, ini adalah daftar lengkap dari apa yang dibutuhkan untuk proyek Anda.

Release plan. Jadwal rilis produk yang sudah selesai.

Sprint backlog. Spesifikasi, tujuan, dan tugas yang terkait dengan sprint yang sedang dijalankan.

Increment. Fungsionalitas produk yang disajikan kepada stakeholders di akhir sprint dan berpotensi diberikan kepada pelanggan.



beberapa **pendekatan**
metode agile yang sering
digunakan di berbagai organisasi dan perusahaan ...

Scrum



Scrum sangat bagus ketika Anda tidak tahu apa produk Anda nantinya dan tidak dapat memprediksi hasil yang tepat. Ini bisa bagus untuk startup yang matang dan ketika Anda perlu meningkatkan produk yang sudah ada.

Kerangka kerja sederhana untuk pengembangan produk yang dibagi-bagi menjadi beberapa proses yang disebut dengan sprint. Satu tim tidak dibebankan dengan beberapa tugas sekaligus, karena dalam scrum satu tim fokus dalam pengembangan aspek tertentu.

Scaled Agile Framework (SAFe)

Scaled Agile Framework (SAFe) adalah seperangkat pola organisasi dan alur kerja untuk menerapkan praktik agile pada skala perusahaan besar. SAFe mempromosikan keselarasan, kolaborasi, dan penyelesaian di sejumlah besar tim agile.

SAFe dibentuk dari dasar tiga badan pengetahuan utama ...

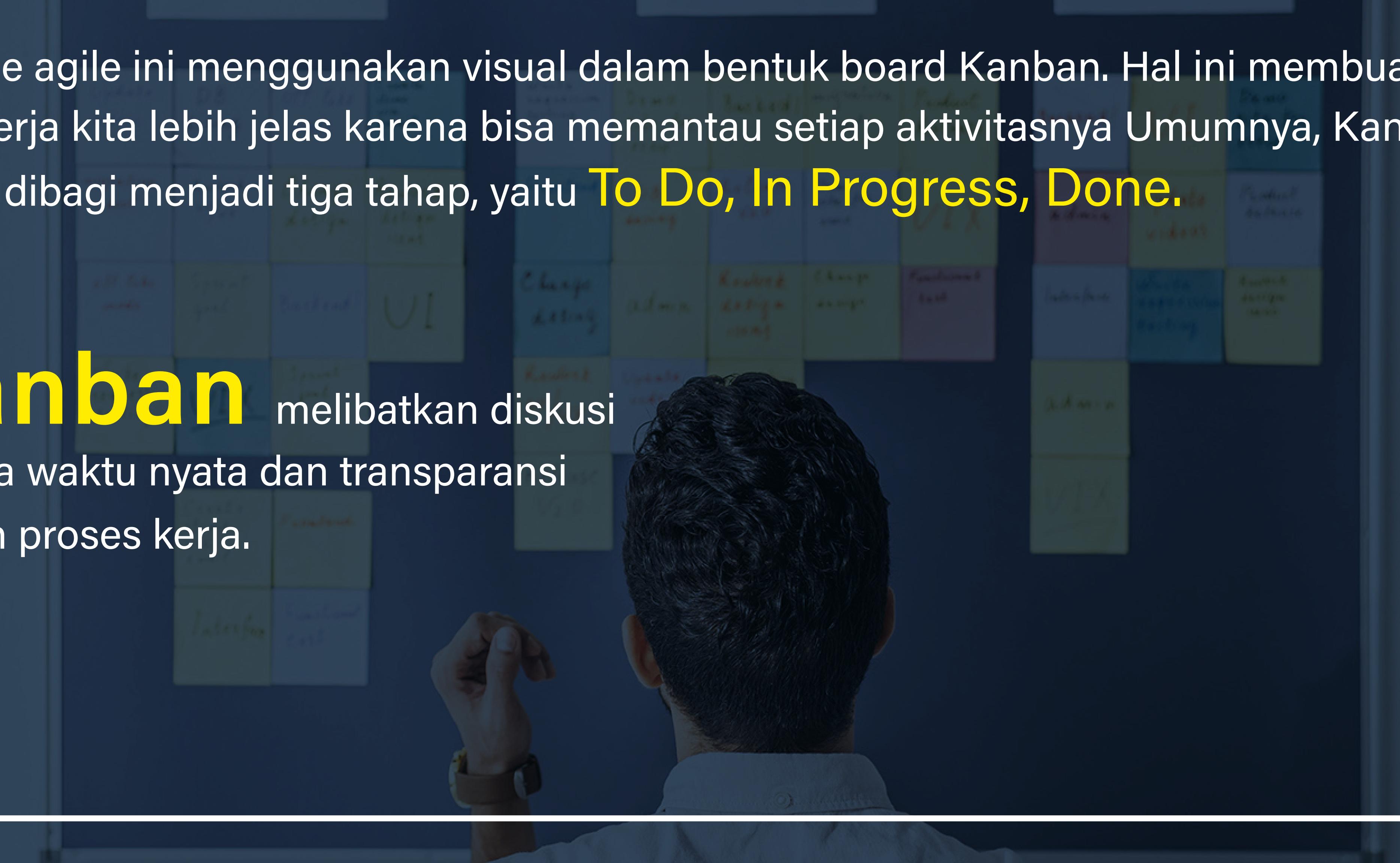
agile software development,
lean product development,
and systems thinking.

Lean Software Development (LSD) adalah kerangka kerja yang gesit berdasarkan pada pengoptimalan waktu dan sumber daya pengembangan, menghilangkan pemborosan, dan pada akhirnya hanya memberikan apa yang dibutuhkan produk.

Lean Software Development



Pendekatan Lean juga sering disebut sebagai strategi Minimum Viable Product (MVP), di mana sebuah tim merilis versi minimal produknya ke pasar, belajar dari pengguna apa yang mereka suka, tidak suka, dan ingin menjadi ditambahkan, dan kemudian mengulangi berdasarkan umpan balik ini.



Metode agile ini menggunakan visual dalam bentuk board Kanban. Hal ini membuat flow kerja kita lebih jelas karena bisa memantau setiap aktivitasnya. Umumnya, Kanban Board dibagi menjadi tiga tahap, yaitu **To Do, In Progress, Done**.

Kanban melibatkan diskusi kinerja waktu nyata dan transparansi penuh proses kerja.

Extreme Programming (XP)

Metodologi yang lebih berfokus ke aspek teknis pengembangan dengan mengedepankan nilai communication, simplicity, feedback, dan courage. Disebut extreme karena untuk mencapai tujuan, tim harus bekerja dengan extra dan keluar dari zona nyaman, dengan tujuan agar produk yang dihasilkan mempunyai kualitas tinggi, dan juga meningkatkan kemampuan tim development.

agile -Kelebihan

Memungkinkan lebih banyak fleksibilitas untuk beradaptasi dengan perubahan.

Kualitas produk lebih baik, karena setiap masukan dan bug dapat diperbaiki di iterasi selanjutnya.

Fitur baru dapat ditambahkan dengan mudah.

Kepuasan pelanggan sebagai umpan balik dan saran dapat dilakukan di setiap tahap.

Pengembangan produk membutuhkan waktu yang relatif cepat dan tidak membutuhkan sumber daya yang besar.

agile -Kekurangan

Kurangnya dokumentasi.

Membutuhkan sumber daya yang berpengalaman dan sangat terampil.

Jika pelanggan tidak jelas tentang bagaimana sebenarnya mereka menginginkan produk tersebut, maka proyek tersebut akan gagal.

Kurang sesuai untuk tim berjumlah kurang dari 10 orang.

Tim harus selalu siap stand-by karena perubahan dapat terjadi kapan saja.

Saat ini sedang dilakukan pengembangan aplikasi edukasi dengan menggunakan jenis scrum.

Setelah tim produk mempersiapkan spesifikasi yang dibutuhkan, seperti dokumen, desain fungsionalitas , dan lain-lain, perusahaan lalu menugaskan 3 tim development untuk menyelesaikan proyek tersebut, dengan setiap tim terdiri dari 4 mobile engineer, 2 backend, 1 QA, dan 1 team leader.

Masing-masing tim memiliki tugas yang berbeda. Tim A fokus membuat konten belajar, tim B membuat fitur registrasi, dan tim C membuat fitur langganan belajar.

Setelah beberapa kali melalui proses sprint, keseluruhan kode yang dibangun digabung lalu melalui tahap uji akhir. Dan setelah semuanya berjalan dengan baik, aplikasi ini dirilis.

Setelah aplikasi dirilis, terdapat masukan dari pengguna untuk mengadakan fitur latihan soal untuk setiap bab di mata pelajaran. Menjawab kebutuhan ini, tim development menjalani siklus sprint dengan menambahkan fitur tersebut.

Lalu terjadi pandemi yang memunculkan kebutuhan baru di mana siswa harus belajar dari rumah, sehingga dibutuhkan fitur live teaching. Maka tim development kembali menjalani siklus sprint untuk menambahkan fitur tersebut.

Dan iterasi akan terus berlangsung untuk mengoptimalkan produk sesuai dengan kebutuhan pengguna.

- Olomu, Maria. (2022). Software development lifecycle-The complete guide [2022]. Diakses pada 31 Agustus 2022, dari [Software Testing Help.](#)
- Software Testing Help. (2022). SDLC (Software Development Life Cycle) Phases, Process, Models. Diakses pada 31 Agustus 2022, dari [Software Testing Help.](#)
- Thampy, Renju. (2019). 7 Stages Of SDLC: How To Keep Development Teams Running. Diakses pada 31 Agustus 2022, dari [Thampy, Renju.](#)
- Xu, Tammy. (2020). What Are Tracer Bullets in Software Development?. Diakses pada 31 Agustus 2022, dari [Xu, Tammy.](#)
- Sung, Brian. (2019). Tracer Development and Prototypes. Diakses pada 31 Agustus 2022, dari [Sung, Brian.](#)
- Sitesbay. (2019). What is Spiral Model. Diakses pada 31 Agustus 2022, dari [Sitesbay.](#)
- Diakses pada 31 Agustus 2022, dari [Intelegain Technologies.](#)
- Intelegain Technologies. (Tanpa Tahun). Spiral Development Model | Software Development Methodology - Intelegain Technologies.
- Software Testing Help. (2022). Spiral Model – What Is SDLC Spiral Model?. Diakses pada 31 Agustus 2022, dari [Software Testing Help.](#)
- Tutorials Point. (Tanpa Tahun). SDLC - Spiral Model. Diakses pada 31 Agustus 2022, dari [Tutorials Point.](#)

Vorobiova, Anna. (2021). SOFTWARE DEVELOPMENT LIFE CYCLE: NIX APPROACH TO SDLC. Diakses pada 31 Agustus 2022, dari

Tutorials Point. (Tanpa Tahun). SDLC - V-Model. Diakses pada 31 Agustus 2022, dari

Asmo. (2018). Agile Methodology: An Overview. Diakses pada 31 Agustus 2022, dari

ruang
guru
cAMP

Terima Kasih