

TESTING

Testing atau pengujian merupakan salah satu tahapan pengembangan perangkat lunak yang harus dilakukan untuk memastikan bahawa program yang dihasilkan sudah sesuai dengan kebutuhan. Berikut jenis-jenis testing:

A. Manual Testing

Dalam pengujian manual, pengujian perangkat lunak dilakukan secara manual tanpa menggunakan tools atau aplikasi otomatis yang tersedia di pasaran. Dalam bentuk pengujian ini, penguji perangkat lunak menguji atau memeriksa bug untuk pengguna akhir (End-User) dan memeriksa proyek untuk mengidentifikasi setiap perilaku abnormal atau bug di dalamnya. Berbagai fase beragam ada untuk menguji proyek secara manual. Ini adalah:

- Pengujian unit
- Tes integrasi
- Pengujian sistem dan
- Pengujian terhadap Request User

B. Automation Testing

Dalam pengujian otomasi (juga disebut Otomasi Pengujian Perangkat Lunak), penguji perangkat lunak harus menulis skrip yang berbeda. Pengujian otomasi melibatkan proses manual yang dilakukan secara otomatis. Ini diimplementasikan untuk menjalankan kembali situasi dan status pengujian yang dilakukan secara manual dan pada saat yang sama dengan langkah cepat dan berulang kali dilakukan untuk memeriksa apakah ada bug atau kesalahan yang tertinggal atau tidak dari pengujian sebelumnya. Jenis pengujian ini juga berkaitan dengan pemeriksaan beban, statistik kinerja, penggunaan CPU dan memori, serta aktivitas. Pengujian otomasi lebih menguntungkan daripada pengujian manual karena menghemat waktu penguji.

a) Autometed Test Tool

Pengujian otomatis membutuhkan dukungan berbagai bahasa seperti VBScript bersama dengan aplikasi otomatis. Berbagai tools tersedia dan sering digunakan yang dapat diimplementasikan untuk menulis skrip semacam itu, adalah sebagai berikut :

- Katalon Studio
- Selenium
- Postman
- Appium
- Test Complete
- Robot FrameWork

Seperti disebutkan di atas, semua tools ini dimaksudkan untuk masing-masing Bahasa pemrograman tetapi untuk tujuan yang sama, yaitu untuk melakukan pengujian dari suatu Aplikasi.

b) Processes and Situations

Dalam berbagai situasi, berbagai proses dapat digunakan untuk mengotomatisasi proses pengujian. Ini adalah:

- Mengidentifikasi suatu case yang berbeda dalam perangkat lunak untuk pengujian otomasi.
- Memilih tools yang tepat untuk pengujian otomasi juga penting karena setiap tools dimaksudkan untuk tujuan tertentu.
- Menulis skrip pengujian.
- Developt terhadap Environment Setting
- Mengeksekusi dan debugging skrip
- Hasil tes dituangkan ke dalam laporan.
- Mengidentifikasi kemungkinan bug atau masalah dalam kinerja perangkat lunak.

Software Testing Methods

Ada berbagai metode untuk menguji perangkat lunak. Metode ini dipilih oleh beberapa penguji yang berbeda berdasarkan kebutuhan dan metodologi mereka. Ada tiga metode pengujian perangkat lunak mendasar bersifat viral dan digunakan di hampir setiap pengembangan proyek. Adapun beberapa metode pengujian tersebut, adalah sebagai berikut :

A. Black Box Testing

Blackbox Testing merupakan suatu pengujian yang dilakukan melalui struktur internal, desain, dan implementasi serta UI dan UX dari produk yang sedang diuji, yang tidak diketahui oleh penguji. Blackbox Testing bersifat fungsional atau non-fungsional, tetapi sebagian besarnya biasanya bersifat fungsional. Teknik pengujian ini mencoba menemukan kesalahan dalam beberapa kategori di bawah ini:

- Fungsi perangkat lunak yang tidak akurat atau hilang,
- Kesalahan di antarmuka (User Interface),
- Kesalahan dalam konsep dan implementasi struktur data,
- Kesalahan terkait basis data,
- Kesalahan dalam memulai atau menghentikan suatu produk.

B. White Box Testing

White Box Testing juga disebut sebagai Pengujian Kotak Terbuka atau Transparan atau Pengujian Kotak Kaca. Terkadang dari sudut pandang pengembang, ini dikenal sebagai Pengujian Berorientasi Code atau pengujian struktural. Jenis teknik pengujian ini berkaitan dengan pengujian struktur internal, desain logika, dan implementasi modul yang berbeda. Di sini, penguji menggunakan jalur input atau latihan pilihannya melalui Code untuk menentukan output yang tepat atau tepat. Karena juga disebut sebagai pengujian berorientasi code, ini berisi pengujian teknis dan pengujian berbasis skrip sebagai bagian dari fase pengujiannya.

Levels of White Box Testing

Teknik pengujian ini berkaitan dengan metodologi pengujian perangkat lunak yang disebutkan di bawah ini:

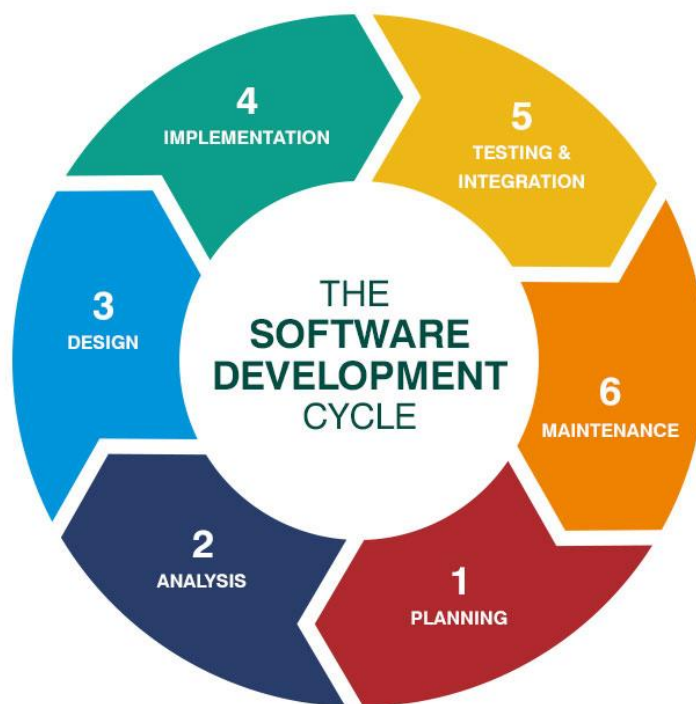
- Unit Testing : Pengujian jalur di dalam unit dan modul
- Integration Testing : Pengujian jalur di antara unit yang berbeda
- System Testing : Pengujian jalur di antara sub-sistem suatu produk

C. Gray Box Testing

Dalam teknik pengujian perangkat lunak ini menggabungkan konsep pengujian Black box dan white box. Dalam Gray Box Testing, bagian dalam produk ini sebagian diketahui oleh penguji. Ini memiliki akses parsial ke struktur data yang berada secara internal untuk merancang berbagai kasus pengujian dan pengujian dari perspektif pengguna atau seperti black box testing. Gray box testing memiliki nama abu-abu karena seperti kotak semi-transparan dari sudut pandang penguji, dan menggabungkan warna hitam dan putih memberikan nuansa abu-abu.

SDLC (System Development Life Cycle)

SDLC adalah singkatan dari System Development Life Cycle atau sering juga dikatakan sebagai siklus hidup pengembangan sistem. SDLC merupakan alur atau siklus yang digunakan dalam pembuatan atau pengembangan sistem informasi yang bertujuan untuk menyelesaikan masalah secara efektif. Dalam pengertian lain SDLC itu adalah tahapan kerja yang harus dilakukan untuk menghasilkan sistem berkualitas tinggi dan sesuai dengan kebutuhan pengguna (Requirement dari User).



1. Perencanaan (Planning)

Dalam tahap ini, tim *engineer* akan merencanakan berbagai persyaratan dalam pembuatan *software* baru atau *software* yang sudah ada. Hal ini biasanya dilakukan dengan cara berdiskusi antar Developer, Product Owner, Scrum Master, QA Tester serta Tim Bisnis maupun User untuk pemenuhan kebutuhan agar sesuai dengan Requirement yang akan di kerjakan.

2. Rancangan (Design)

Tahapan ini dimulai dari mengubah spesifikasi sebuah *software* ke dalam *design plan* yang disebut dengan Desain Dokumen Spesifikasi (DDS). Semua tim yang bersangkutan termasuk dengan klien akan membantu mereview dari rancangan ini dan memberikan *feedback*. Sangat penting juga untuk mengumpulkan semua *feedback* yang diberikan dalam dokumen ini. Jika ada kegagalan dalam tahap ini, akan berakibat biaya yang melonjak dan menjadi *over* atau kemungkinan terburuknya dapat membuat proyek menjadi gagal.

3. Membangun atau mengembangkan produk (Development)

Dalam tahapan ini, tugas *engineer* dan tim adalah untuk membangun atau mengembangkan sebuah produk dari barisan bahasa pemrograman. Biasanya proses ini akan dilakukan oleh para developer baik Developer Front End maupun Back End, dan untuk pengerjaan nya pasti memiliki estimasi waktu sesuai dengan jadwal yang telah di tentukan dari time line pengerjaan proyek tersebut.

4. Pengujian (Testing)

Dalam tahap pengujian, pertanyaan-pertanyaan seperti “sudahkah kita mendapat apa yang kita mau?” akan terus berulang.

Dalam tahapan ini, *Team* harus memastikan bahwa produk yang dibuat tidak memiliki cacat dan sesuai dengan kebutuhan User maupun Client. Biasanya pada proses ini, lebih dilakukan oleh seorang Tester yang memiliki tanggung jawab untuk memastikan produk yang dihasilkan berjalan dengan baik dan tepat. Walaupun proses ini lebih bertumpu kepada seorang Tester, namun tester juga harus bisa berkolaborasi dengan para developer dengan memberikan beberapa kemungkinan untuk case diluar dari scenario dan memiliki pikiran yang kritis untuk mencapai tujuan bersama.

5. Memasarkan Produk

Jika pengujian sudah selesai, proses kerja berikutnya yang bisa segera diluncurkan adalah pemasaran produk. Pada fase ini bukan tidak menunjukkan bahwa setiap tahap SDLC telah selesai

dilakukan. Namun, berbagai *feedback* yang disampaikan oleh *user* juga harus didengarkan lagi, dan bila perlu akan ada penyesuaian lagi terhadap produk yang telah di develop.

6. Melakukan pemeliharaan (*maintenance*)

Tahap kerja terakhir yang perlu dilakukan oleh semua perusahaan dalam sistem SDLC adalah *maintenance*. Dengan keadaan dan kondisi teknologi yang terus berubah, tahapan terakhir dari SDLC mengharuskan *engineer* dan tim tetap memelihara produk yang sudah selesai. Hal ini perlu dilakukan untuk mengurangi *error* dan penurunan kualitas pada produk.

Top Software Development Methodologies – Advantages and Disadvantages

Apa itu Software Development Methodology?

Software Development Methodology adalah metodologi atau proses terstruktur yang membentuk pengembangan solusi atau proyek perangkat lunak tertentu. Ini menggabungkan filosofi desain dan ide-ide praktis dan realistis yang bertujuan untuk menciptakan metode pengembangan perangkat lunak yang terorganisir.

Ada metodologi berbeda berdasarkan berbagai pendekatan, yang diperkenalkan selama bertahun-tahun.

Metodologi pengembangan perangkat lunak atau siklus hidup pengembangan perangkat lunak menyediakan struktur bagi pengembang dan perancang untuk berkolaborasi dan memfasilitasi kerja sama tim yang efektif. Mereka juga membantu menciptakan platform komunikasi formal dan menentukan bagaimana info dipertukarkan di antara anggota tim.

Sementara sebagian besar perusahaan pengembangan perangkat lunak mengakui perlunya metodologi, pendapat tetap terbagi atas metode mana yang terbaik; setelah semua, setiap metode memiliki kelebihan dan kekurangan.

Metodologi yang paling cocok juga bergantung pada tujuan dan kebutuhan tim dan proyek, serta struktur tim. Sebuah perusahaan juga dapat menggunakan metode yang berbeda untuk berbagai proyek.

Kebutuhan Metodologi Pengembangan Perangkat Lunak

Dapat dikatakan bahwa sangat penting untuk tidak hanya memilih metodologi pengembangan perangkat lunak yang tepat, tetapi juga menerapkannya dengan benar dan disiplin di seluruh siklus hidup pengembangan perangkat lunak proyek. Jika dilakukan sembarangan, itu bisa menjadi bencana.

Tanpa struktur yang tepat, ada kemungkinan miskomunikasi yang lebih tinggi, dan harus sering menggabungkan perubahan dalam desain dan fungsionalitas karena permintaan pelanggan.

Memodifikasi perangkat lunak berkali-kali dapat berdampak negatif pada proyek – terutama dalam hal biaya dan waktu. Ini dapat menyebabkan pemborosan dan duplikasi usaha, serta pengeluaran dan waktu yang berlebihan, yang akhirnya menghasilkan aplikasi yang tidak berkualitas tinggi.

1. Model Waterfall

Waterfall adalah metode pengembangan software yang paling tua. Secara substansi ia mensimplifikasi proses software engineering ke dalam diagram proses yang linear dimana penyelesaian dari task sebelumnya sangat penting bagi pengembang untuk bisa mengerjakan pekerjaan yang lain dan tidak dapat diubah atau dilakukannya penambahan di pertengahan proses develop.

1. Keunggulan

1. Mudah dimengerti, sehingga baik digunakan oleh pemula
2. Mudah untuk di-manage karena setiap fase memiliki deliverablesnya masing-masing dan proses review
3. Cepat untuk diimplementasi untuk proyek dengan skala kecil dimana requirement dapat dimengerti dengan baik
4. Desain yang sederhana membuatnya mudah untuk ditesting dan dianalisis

2. Kelemahan

1. Metode ini hanya cocok untuk proyek dengan requirement yang sudah sangat jelas dengan detail requirement yang bisa di-deliver diawal
2. Metode ini tidak cocok untuk proyek maintenance atau proyek jangka panjang

3. Tidak fleksibel: ketika aplikasi di launch, tidak memungkinkan untuk memodifikasi atau merubah sistem yang dibuat
4. Tidak bisa membuat software yang lain sampai seluruh proses waterfall selesai

2. Metode Prototype

Model ini mendukung developer untuk membuat prototype sehingga mereka bisa mendemonstrasikan fungsionalitas softwarenya kepada klien dan membuat modifikasi berdasarkan feedback yang di berikan. Metode ini mirip dengan membuat MVP, kita menciptakan versi pre-develop dulu sebelum menginvestasikan waktu dan uang untuk menciptakan produk yang lebih lengkap.

1. Keunggulan

1. Dengan metode ini, kita bisa memberikan klien experience yang lebih awal untuk software yang akan digunakan dan memperbaiki serta melengkapinya dengan feedback yang diberikan klien
2. Karena kita telah mengidentifikasi risiko dan isu yang mungkin terjadi di awal, kita juga dapat mengurangi risiko kegagalan
3. Komunikasi antara klien dan tim pengembang yang intens akan memperkuat hubungan antara kedua belah pihak

2. Kelemahan

1. Metode ini bisa terbilang cukup mahal. Disisi lain, metode ini dapat mengurangi risiko, sehingga kita dapat meminimalisir potensi budget terbuang di-awal waktu
2. Pelibatan diawal dengan klien bisa saja menjadi hal yang buruk, mereka mungkin akan terlalu banyak ikut campur dan meminta banyak perubahan tanpa sepenuhnya memahami proyek secara keseluruhan
3. Terlalu banyak modifikasi akan mengganggu workflow dari tim development

3. Metode Agile Software Development

Metode pengembangan software berbasis agile fokus pada perencanaan yang adaptif, evolutionary development, dan improvement yang berkelanjutan melalui respon yang fleksibel terhadap

perubahan. Tujuan akhir dari metode ini adalah release yang lebih cepat dengan risiko bugs/issue yang lebih sedikit

1. Keunggulan

1. Pendekatan yang adaptif sehingga dapat merespon perubahan requirement dengan sangat cepat dan efisien
2. Feedback yang berkesinambungan akan meminimalisir risiko dengan signifikan
3. Komunikasi yang berkelanjutan meningkatkan transparansi antara klien dan tim development
4. Fokus pada pengerjaan software, sehingga tidak perlu terlalu khawatir pada dokumentasi

2. Kekurangan:

1. Scope pengerjaan yang bisa berubah kapanpun dapat menyebabkan kurangnya fokus dari tim development dan menyebabkan isu jika brief yang diberikan tidak jelas
2. Kurangnya dokumentasi dapat meningkatkan risiko miscommunication

4. Rapid Application Development (RAD)

Tujuan utama adalah menghasilkan pengerjaan yang lebih cepat dan kualitas yang lebih tinggi dibandingkan yang bisa didapatkan dengan opsi yang lain. Metode ini terkait dengan agile development, dimana menekankan pada pengerjaan software dan feedback user dalam perencanaannya. Bisa dibilang, “lebih sedikit bicara, lebih banyak testing”

1. Keunggulan

1. Mengurangi risiko karena identifikasi issue di awal dan feedback client
2. Feedback berkala meningkatkan transparansi antara tim development dan klien
3. Klien yang dapat melihat hasil dari prototype diawal akan menghasilkan kualitas produk yang lebih baik di akhir
4. Perencanaan dan dokumentasi yang lebih sedikit meningkatkan kecepatan development

2. Kelemahan

1. Mengurangi fitur karena “time boxing” dimana ketika fitur di dorong kembali ke versi yang lebih baru untuk menyelesaikan rilis dalam waktu singkat
2. Tidak cocok untuk proyek dengan budget yang rendah karena biaya modeling dan automated generation code sangat tinggi
3. Metode ini relatif baru sehingga cukup berisiko
4. Butuh kerjasama tim yang tinggi di kantor untuk proses yang sangat cepat bergerak agar bisa sukses.

5. Dynamic Systems Development Model

Metode ini prinsipnya mengacu pada RAD model yang telah dijelaskan sebelumnya. Pendekatan iterative yang menekankan pada improvement berkelanjutan dan keterlibatan user. Metode ini tidak hanya berfokus pada software development dan codingan, namun juga melibatkan project management dan project delivery – dimana alasannya adalah karena kebanyakan digunakan oleh proyek non-IT. Tujuan utamanya adalah menciptakan produk atau layanan yang tepat waktu dan sesuai budget.

1. Keunggulan

1. User sangat dilibatkan dalam proses development sehingga mereka dapat melihat progress pengembangan software sejak dini
2. Meningkatkan transparansi dan pengembangan secara berkala mampu mengurangi risiko

2. Kelemahan

1. Costly untuk diimplementasikan karena membutuhkan lebih dari 10 peran dedicated, belum ditambah dengan frequent testing
2. Tidak cocok untuk organisasi yang kecil karena kebutuhan peran yang sangat tinggi dalam metode ini

6. Spiral Model

Model ini berfokus pada identifikasi risiko di awal. Developer mulai meneliti potensial isu ketika proyek masih dalam skala yang kecil. Kemudian mereka mengatasi risiko tersebut dengan

membuat perencanaan untuk tetap menjalankan project sebagaimana rencana awal atau dengan menyelesaikan isu yang mungkin terjadi dulu baru kemudian melanjutkan ke iterasi berikutnya.

1. Keunggulan

1. Tim butuh waktu yang lama dan energy yang besar untuk menganalisis risiko yang dapat meminimalisir risiko yang akan datang kemudian
2. Metode ini efektif untuk skala yang besar dan proyek yang sangat krusial
3. Cukup fleksibel dan mampu memberikan fungsionalitas tambahan di kemudian hari

2. Kelemahan

1. Mahal karena level analisis yang cukup *complicated*
2. Kesuksesan dari seluruh proyek tergantung dari kesuksesan dalam analisa risiko
3. Tidak ada definisi selesai yang pasti, dimana proyek bisa saja overtime dan budget jika tidak di manage dengan hati-hati

7. Metode Extreme Programming

Metode Extreme Programming (XP Methodology) digunakan oleh tim development untuk membuat software dalam environment yang tidak stabil seperti ketika requirement yang ada sangat cepat berubah. Metode ini memiliki release berkala dalam waktu development yang singkat.

1. Keunggulan

1. Pelibatan klien dalam proyek meningkatkan transparansi dan hubungan antara vendor dan klien
2. *Frequent checkpoints* dalam metode ini akan membantu developer membuat perencanaan dan jadwal
3. *Feedback* yang banyak akan membantu meminimalisir risiko dan meningkatkan improvement

2. Kelemahan

1. Model ini membutuhkan meeting berkala dimana bisa jadi mahal untuk kedua belah pihak

2. Perubahan yang terlalu sering bisa mengganggu developer dan cukup *tricky* untuk mengkalkulasi waktu dan estimasi harga
3. Biaya untuk merubah requirement di kemudian hari dalam proyek cukup mahal.

8. Feature-Driven Development (FDD)

FDD adalah proses development yang iterative yang banyak digunakan dalam industry praktis. Sebagaimana namanya, fokus metode ini adalah mengorganisir software development based on features yang ada. Tujuannya adalah untuk mendeliver pengerjaan software kepada klien secepat mungkin

1. Keunggulan

1. Lima proses yang sederhana (develop model, build feature list, plan, design and build by feature) memberikan struktur dan overview yang baik dari proyek
2. Model ini dibangun atas set standar yang digunakan dalam industri pengembangan software, dimana membuat developer lebih mudah untuk mengerti lebih cepat

2. Kelemahan

1. Kompleksitas dari metode ini membuatnya tidak cocok untuk proyek atau tim yang kecil
2. Banyak tanggungjawab yang diberikan kepada lead developer, sehingga mereka harus benar-benar terlatih dan siap untuk berperan menjadi koordinator, desainer dan mentor.

9. Metode Joint Application Development

Sistem development awalnya digunakan untuk mendesain sistem berbasis computer sebelum dimigrasikan kepada software development. Metode pengembangan software ini melibatkan interaksi antara user dan tim development untuk bekerja dengan sistem yang berbeda dan requirement yang berbeda ketika software sedang dikembangkan. Tujuan utama adalah melibatkan klien dalam proses development sebanyak mungkin via collaborative workshop bernama JAD sessions. Fokus dari pertemuan ini adalah lebih membahas tujuan perusahaan dibandingkan pembahasan teknis.

1. Keunggulan

1. Komunikasi yang intens meningkatkan transparansi dan kolaborasi
2. Pendekatan ini menghasilkan kualitas informasi yang sangat baik dalam waktu yang singkat
3. JAD mengurangi waktu dan biaya
4. Tim dapat menyelesaikan permasalahan dengan cepat

2. Kelemahan

1. Pertemuan berkala dan workshop dapat menghabiskan waktu, dan tentu saja mahal
2. Butuh perencanaan yang panjang dan detail; jika tidak, akan menghabiskan waktu lebih lama dari metode yang lain
3. Pendekatan ini membutuhkan pemimpin yang kuat dan developer yang berpengalaman untuk memastikan workshop berjalan dengan fokus dan produktif

10. Metode Lean Development

Metode ini menggunakan prinsip dari lean manufacturing (mengurangi cost, biaya, dan *waste*), dan mengaplikasikannya dalam pengembangan software untuk mengurangi effort programming dan budget. Metode ini memberikan conceptual framework yang sangat baik. Metode ini juga lebih fleksibel dari metode agile.

1. Keunggulan

1. Meningkatkan efisiensi dengan mempercepat development dan mengurangi biaya keseluruhan dari proyek
2. Development yang lebih cepat sehingga tim developer dapat men-deliver lebih banyak fungsionalitas dalam waktu yang singkat
3. Tim developer memiliki kewenangan lebih untuk mengambil keputusan dimana dapat memperkuat seseorang dan meningkatkan motivasi serta progress.

2. Kelemahan

1. Kesuksesan tergantung dari kedisiplinan tim, komitmen dan kemampuan teknis
2. Metode ini membutuhkan business analyst untuk memastikan dokumentasi sudah benar dan dimengerti oleh semua orang yang terlibat

3. Fleksibilitas developer yang sangat tinggi dapat menyebabkan kekurangan fokus dimana akan mempengaruhi workflow dari proyek keseluruhan.

11. Metode Rational Unified Process (RUP)

Metode ini adalah framework proses yang adaptable dengan organisasi. Metode RUP menentukan project life-cycle yang terdiri dari 4 fase (inception, elaboration, construction, dan transition), masing-masing dengan milestone dan objektifnya.

1. Keunggulan

1. Membantu anggota tim untuk mengidentifikasi dan menyelesaikan risiko proyek yang terkait dengan client melalui request management and review yang lebih berhati-hati.
2. Metode ini cukup scalable, sehingga cocok untuk berapapun jumlah tim atau proyeknya
3. Review berkala membantu untuk menjaga fokus dan meningkatkan transparansi bagi kedua belah pihak

2. Kelemahan

1. Proses development kompleks dan membutuhkan skill yang mendalam dari tim
2. Component testing yang berkelanjutan meningkatkan kompleksitas dan hasil dimana banyak issue yang terjadi ketika fase testing.

12. Metode Scrum Development

Metode ini merupakan metode yang terbaik untuk proyek yang cepat berubah dengan deadline yang juga mepet. Metode ini didesain untuk tim dengan tiga sampai 9 orang dimana pekerjaannya dibagi dengan istilah *sprints* untuk menyelesaikan suatu scope pekerjaan pada waktu yang disepakati (biasanya dua pekan). Setiap hari, tim akan mereview progress dalam sebuah meeting yang dinamakan daily scrums.

1. Keunggulan

1. Meningkatkan kecepatan dalam proses development dan dapat membawa proyek yang lambat kembali ke track

2. Pengambilan keputusan sebagian besar berada dalam tangan tim developer. Hal ini membantu mereka untuk fokus dan meningkatkan motivasi
 3. Fleksibel, dimana memudahkan update dan perubahan berkala
 4. Daily meeting membantu manajer untuk mengukur produktifitas individual. Metode ini juga meningkatkan kolaborasi dan produktifitas dalam tim
2. Kelemahan
1. Sangat cocok untuk skala kecil, dan proyek yang cepat berubah. Tidak cocok untuk skala besar
 2. Metode ini membutuhkan orang berpengalaman yang pernah bekerja di proyek yang mirip dengan yang ingin dikerjakan saat ini.
 3. Anggota tim harus memiliki skills yang banyak sehingga mampu membantu mereka dalam mengerjakan task diluar dari area spesialisasinya. Beberapa anggota tim, oleh karena itu, membutuhkan training tambahan
 4. Membagi development produk dalam sprint singkat membutuhkan perencanaan yang Amatang dan hati-hati.

API TESTING

API (Application program interface) adalah service yang menyediakan data dari server yang di request oleh client yang biasanya dalam format json, API ini adalah bagian yang paling penting dalam sebuah aplikasi Mobile / web frontend. Design API yang jelas akan memudahkan kita dalam mengembangkan suatu aplikasi.

Bayangkan ketika sebuah API didesign dengan kembalian data yang tidak konsisten, nama path yang tidak merepresentasikan dari fungsi API, data yang dikembalikan terlalu berbelit belit dalam artian seharusnya bisa 1x panggil tapi harus 2x panggil dengan path API yang berbeda itu akan menambah effort untuk developing.

Bagaimana API yang Baik?

A. Restful

Untuk penamaan path API yang baik kita bisa memperhatikan aturan Restful. Restful ini hanya menjadi acuan, mungkin ada beberapa kondisi tidak bisa menggunakan aturan Restful.

HTTP Verb	Path	Action	Used for
GET	/admin/posts	index	admin_posts_path
GET	/admin/posts/new	new	new_admin_post_path
POST	/admin/posts	create	admin_posts_path
GET	/admin/posts/:id	show	admin_post_path(:id)
GET	/admin/posts/:id/edit	edit	edit_admin_post_path(:id)
PATCH/PUT	/admin/posts/:id	update	admin_post_path(:id)
DELETE	/admin/posts/:id	destroy	admin_post_path(:id)

(Table Restful API)

Sebagai contoh ada Restful pada resource admin/post diatas, dengan path yang sama yaitu **/admin/posts/** kita bisa membuat beberapa action sesuai dengan HTTP Verb. Untuk nama path yang biasa digunakan adalah *plurar* (jamak).

B. HTTP Status Code

HTTP Status code ini berfungsi untuk membantu kita dalam mengenali response balik dari API yang telah dijalankan pada saat user sedang melakukan sebuah Activity di dalam platform ataupun Device yang sedang digunakan.

1. Kode Status HTTP Sukses

Kode status yang kedua menunjukkan bahwa permintaan berhasil diterima, dipahami, dan dimengerti.

1. **200 Ok (Successful)**

Permintaan sudah diterima dan dipahami kemudian sedang diproses.

2. **201 Permintaan Berhasil Dibuat (Created)**

Permintaan berhasil dan server membuat sumber/resource baru.

3. **203 Bukan informasi otoritas (Non-authoritative information)**

Server berhasil memproses permintaan, tetapi menampilkan informasi yang mungkin berasal dari sumber lain.

4. **204 Tanpa Konten (No content)**

Server berhasil memproses permintaan, akan tetapi tidak menampilkan konten apa pun.

5. **205 Konten di Reset (Reset content)**

Server berhasil memproses permintaan, tetapi tidak menampilkan konten apa pun. Berbeda dengan respon 204, respons ini mengharuskan pemohon mereset tampilan dokumen.

6. **206 Konten Parsial (Partial content)**

Kode status 206 adalah menanggapi permintaan bagian dari dokumen. Ini digunakan oleh alat caching canggih, ketika agen pengguna meminta hanya sebagian kecil dari halaman, dan hanya sebagian yang diberikan.

2. Kode Status HTTP Pengalihan (Redirection)

Kelas kode status yang ketiga ini menunjukkan bahwa tindakan lebih lanjut perlu dilakukan oleh agen pengguna untuk memenuhi permintaan tersebut. Kode status ini digunakan untuk pengalihan (redirect) URL.

1. **300 Banyak Pilihan (Multiple Choice)**

Permintaan tersebut (300) memiliki lebih dari satu respon yang dapat dipilih. Pengguna harus memilih salah satunya. Tidak ada cara standar untuk memilih salah satu tanggapan.

2. **301 Pengalihan Permanen (Moved Permanently)**

Kode tanggapan ini berarti bahwa URI sumber daya yang diminta telah diubah. Kemudian, URI baru akan diberikan sebagai tanggapan.

3. **302 Ditemukan (Found)**

Berbeda dengan 301 yang permanen, kode status 302 ini berarti bahwa URI sumber daya yang diminta telah diubah untuk sementara.

4. **303 Lihat Lainnya (See Other)**

Server mengembalikan kode ini ketika pemohon membuat permintaan GET terpisah ke lokasi yang berbeda untuk menerima respon. Untuk semua permintaan selain permintaan HEAD, server secara otomatis mengarahkan ke lokasi lain.

5. **304 Belum Dirubah (Not Modified)**

Halaman yang diminta belum dimodifikasi sejak permintaan terakhir. Ketika server menampilkan respons ini, tidak mengembalikan isi halaman.

6. **305 Menggunakan Proxy (Use proxy)**

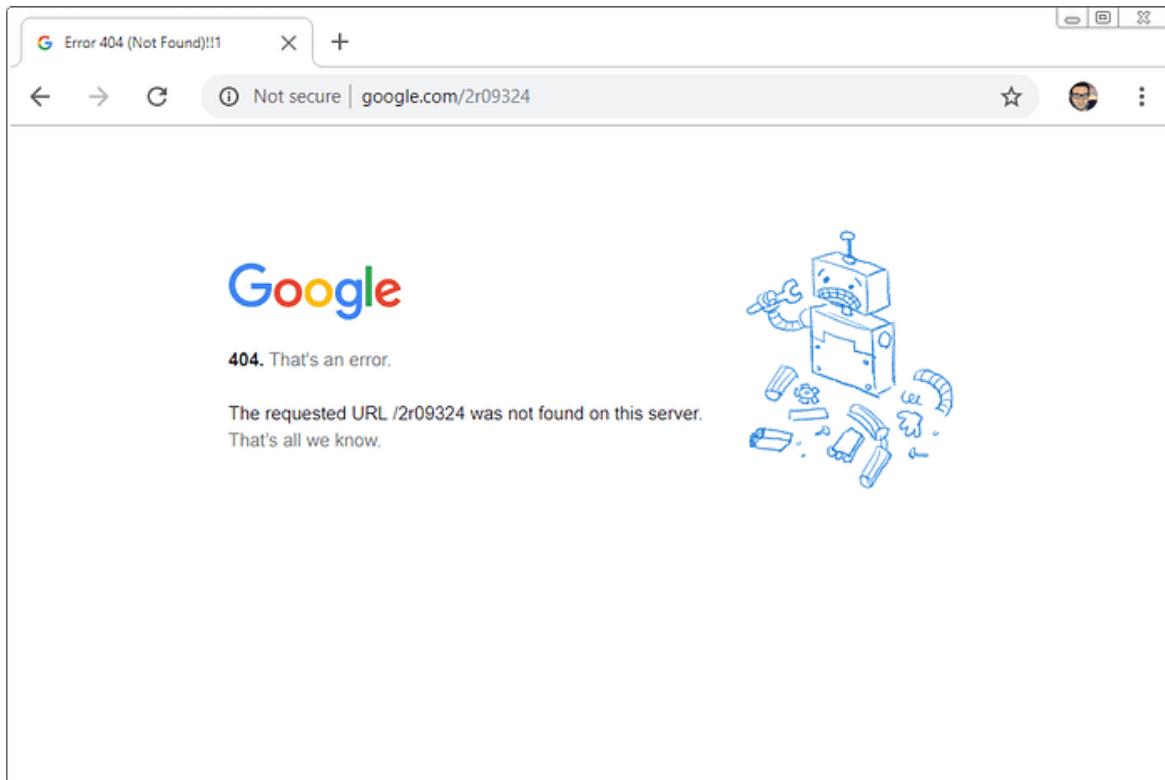
Pemohon hanya dapat mengakses halaman yang diminta dengan menggunakan proxy. Ketika server menampilkan respons ini, juga menunjukkan proxy yang harus digunakan pemohon.

7. **307 (Temporary redirect)**

Server sedang merespons permintaan dengan halaman dari lokasi yang berbeda, tetapi pemohon harus terus menggunakan lokasi asli untuk permintaan di masa depan. Kode ini mirip dengan 302, namun tidak sepenuhnya. Untuk pengalihan bersifat sementara sebaiknya gunakan 302.

3. Kode Status HTTP Kesalahan Klien (Client Error)

Kode status http ini menunjukkan bahwa ada kemungkinan kesalahan dalam permintaan pada klien / pengguna yang mencegah server untuk memprosesnya.



1. **400 Permintaan Tidak Layak (Bad request)**

Server tidak memahami sintaks/syntax permintaan dari klien.

2. **401 Tidak Ter-Autentikasi (Not authorized)**

Permintaan membutuhkan otentikasi. Server biasanya menampilkan respon ini untuk halaman setelah login (page behind a login).

3. **403 Terlarang (Forbidden)**

Server menolak permintaan tersebut. itu mungkin bahwa server atau host memblokir pengaksesan.

4. **404 Tidak Ditemukan (Not found)**

Server tidak dapat menemukan halaman yang diminta. Misalnya, server akan menampilkan kode ini jika permintaan untuk halaman tersebut tidak ada di server.

5. **405 Metode Tidak Diperbolehkan (Method not allowed)**

Metode yang ditentukan dalam permintaan tidak diperbolehkan.

6. **406 Tidak Dapat Diterima (Not acceptable)**

Halaman yang diminta tidak dapat merespons dengan karakteristik konten yang diminta.

7. **407 Meminta Otentikasi Proxy (Proxy authentication required)**

Mirip dengan 401 (not authorized), tetapi menetapkan agar pemohon harus mengotentikasi

menggunakan proxy. Ketika server mengembalikan respons ini, juga menunjukkan proxy yang harus digunakan pemohon.

8. **408 Melampaui Batas Waktu Permintaan (Request timeout)**

Server kehabisan waktu pada saat menunggu permintaan.

9. **409 Terjadi Konflik (Conflict)**

Server mengalami konflik saat memenuhi permintaan. Server menampilkan kode ini dalam menanggapi permintaan yang bertentangan dengan permintaan sebelumnya, bersama dengan daftar perbedaan antara permintaan.

10. **410 Permintaan Hilang (Gone)**

Server menampilkan respon ini ketika sumber yang diminta telah dihapus secara permanen. Hal ini mirip dengan kode 404 (Not found). Jika sumber telah dipindahkan secara permanen, Anda harus menggunakan kode 301 untuk menentukan lokasi baru sumber.

11. **411 Syarat Panjang (Length required)**

Server tidak akan menerima permintaan tanpa Content-Length header field yang sah.

12. **413 Permintaan Terlalu Besar (Request entity too large)**

Server tidak dapat memproses permintaan karena terlalu besar untuk server tangani.

13. **414 URL Terlalu Panjang (Requested URI is too long)**

URI yang diminta (biasanya, URL) terlalu panjang untuk diproses oleh server.

14. **415 Tidak Mendukung Format Media Tertentu (Unsupported media type)**

Permintaan tersebut dalam format yang tidak didukung oleh halaman yang diminta.

15. **416 Rentang Tidak Tersedia (Requested range not satisfiable)**

Server menampilkan kode status ini jika permintaan untuk rentang/range tidak tersedia untuk halaman tersebut.

16. **417 Harapan Tidak Terpenuhi (Expectation failed)**

Server tidak dapat memenuhi persyaratan Expect request-header field.

4. Kode Status HTTP Permasalahan Server (Server Error)

Kode status yang terakhir menunjukkan bahwa server mengalami galat/error internal saat mencoba untuk memproses permintaan klien. Kesalahan ini cenderung dari server sendiri, tidak berkaitan dengan permintaan.

1. **500 Permasalahan Internal Server (Internal server error)**

Server mengalami galat/error dan tidak dapat memenuhi permintaan.

2. **501 Tidak Terimplementasi (Not implemented)**

Server tidak memiliki fungsi untuk memenuhi permintaan. Misalnya, server akan menampilkan kode ini ketika tidak mengenali metode permintaan.

3. **502 Gateway yang Buruk (Bad gateway)**

Server bertindak sebagai gateway atau proxy dan menerima respon tidak valid dari upstream server.

4. **503 Server Tidak Tersedia (Service unavailable)**

Server saat ini tidak tersedia (karena kelebihan beban atau dalam proses maintenance). Umumnya, ini bersifat sementara.

5. **504 Melebihi Batas Waktu Gateway (Gateway timeout)**

Server bertindak sebagai gateway atau proxy dan tidak menerima permintaan tepat waktu dari server upstream.

6. **505 Tidak Mendukung Versi HTTP (HTTP version not supported)**

Server tidak mendukung versi protokol HTTP yang digunakan dalam permintaan

Dengan mempelajari sedikit kode http sedikitnya anda menjadi mengetahui jika suatu saat error terjadi pada browser anda. Setidaknya mengetahui penyebabnya berdasarkan status dari jenis error yang terlihat. Hal ini juga seringkali menjadi acuan fokus perbaikan bagi teknisi website.

STUDY CASE:

1. Buatlah setiap kemungkinan terkait scenario ketika User akan melakukan Login di Aplikasi E-Commerce!
2. Gambarkan secara detail mengenai contoh issue/bug maupun result yang sukses terkait salah satu jenis aplikasi yang ada pada device anda!

JAWABAN

1. Sebelum masuk ke dalam contoh Skenario dari suatu e-commerce, berikut merupakan penjelasan mengenai Test Scenario :

- 1) Test Login Functionality yaitu Melakukan test terhadap fungsi login apakah berfungsi sesuai dengan memasukkan username dan password yang valid (terdaftar dalam system) dan username/password yang tidak valid.
- 2) Check New Order yaitu Melakukan Test terhadap fungsi New Order apakah berfungsi sehingga order baruyang dilakukan disimpan dalam sistem
- 3) Check Open Order yaitu Melakukan Test terhadap fungsi Check Open Order apakah melakukan test pada orderdengan memasukkan no_order dan tanggal order, melihat detail order.
- 4) Check Fax Order yaitu Melakukan test pada Fax Order dengan memverifikasi Fax Order dengan memasukkan no_fax
- 5) Check Help Section yaitu Melakukan test pada menu Help dalam aplikasi apakah berfungsi atau tidak.
- 6) Check About yaitu Melakukan test pada menu About apakah menu sesuai yakni berisi informasi aplikasi(versi, copyright, etc

Dari ke enam Test Scenario diatas harus dapat mentukan test terhadapat functionality dari system. Jadi dari 6 scenario tersebut dipilih test yang diprioritaskan untuk mempersingkat waktu. Adapun test scenario yang dipilih adalah scenario fungsionalitas dari system yakni :

- a. Login Functionality
- b. New Order
- c. Open Order
- d. Fax Order

Salah satu contoh aplikasi E-commerce yang saya gunakan sehari-hari adalah Grab. Berikut scenario ketika User akan melakukan login

- Pilih salah satu aplikasi e-commerce (misal Grab)
- Buka Aplikasi Grab untuk New User
- Apps akan menampilkan halaman Registrasi

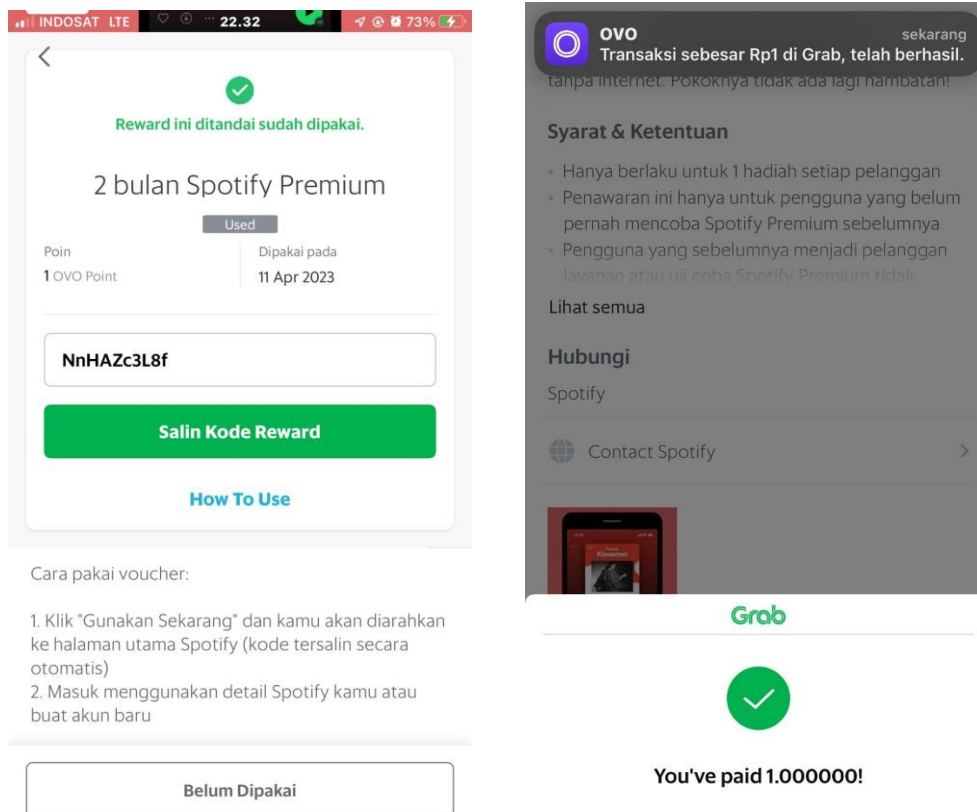
- Masukkan email yang aktif
- Untuk New User, OTP akan dikirimkan ke Gmail yang telah di daftarkan
- Sistem akan mengirim kode otp ke Gmail
- Untuk melakukan aktivasi masukkan kode OTP tersebut
- Apps menampilkan page success ketika OTP berhasil diinput
- Mengisi biodata => Sebelum aplikasi dijalankan, Maka user Baru harus terlebih dahulu mengisi data user (seperti nama, no.Telepon dan alamat)
- Aplikasi sudah dapat digunakan oleh User Baru



2.

- a. Gambar diatas adalah salah satu contoh issue/bug yang ada di Aplikasi Grab. Dimana saat user ingin memesan makanan atau transportasi dari grab, muncul peringatan bahwa

cuaca kurang baik. Sedangkan cuaca yang ditampilkan di lokasi user pada saat itu cerah (dapat dilihat dari gambar 2).



- b. Berikut adalah salah satu contoh transaksi yang berhasil/sukses yang ada di device User. User berhasil menukar rewards yang di tawarkan oleh grab yaitu dengan 1 point akan mendapatkan 2 bulan gratis Spotify Premium. Tetapi user memiliki Notifikasi dibagian **“You’ve paid 1.000000!”** menurut saya notifikasi yang menunjukkan angka **1.000000** yang ditunjukkan pada Gambar diatas merupakan salah satu hal yang sangat ambigu bagi pengguna baru dikarenakan angka yang ditukarkan adalah 1 Poin dimana angka tersebut tidak memiliki sinkronisasi dengan jumlah poin, serta notifikasi yang ditampilkan tersebut tidak memiliki penggunaan Bahasa yang konsisten (dimana User menggunakan Bahasa dalam Apps, namun notifikasi tersebut ditampilkan dalam Bahasa Inggris).



Halaman Gagal Dimuat

Hal ini dapat terjadi karena koneksi internet tidak stabil, sesi berakhir atau hal lainnya. Coba dalam beberapa saat lagi, ya!

Muat Ulang

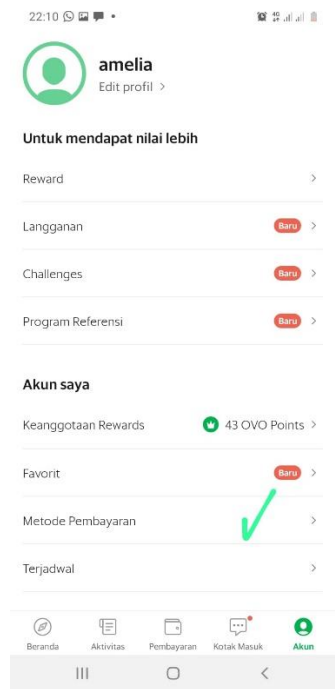


- c. Gambar diatas merupakan salah satu contoh Result yang expected (yang sesuai dengan harapan User) menurut Opini saya. Dikarenakan => Apps menampilkan halaman state Error pada saat koneksi tidak stabil/Disconnect data seluler. Jika User memiliki koneksi data yang kurang stabil, maka Aplikasi akan menampilkan halaman state error dan bukan menjadi logout atau semacamnya.

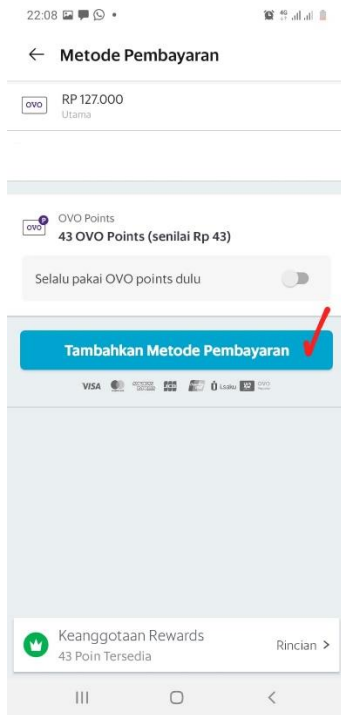
- d. Di bawah ini merupakan salah satu contoh Expected Result dimana User dapat menambahkan Metode Pembayaran (Payment Method) di dalam Aplikasi Grab beserta langkah-langkah atau steps nya untuk menggunakan platform Mobile :

Step untuk melakukan penambahan Metode Pembayaran di Aplikasi Grab:

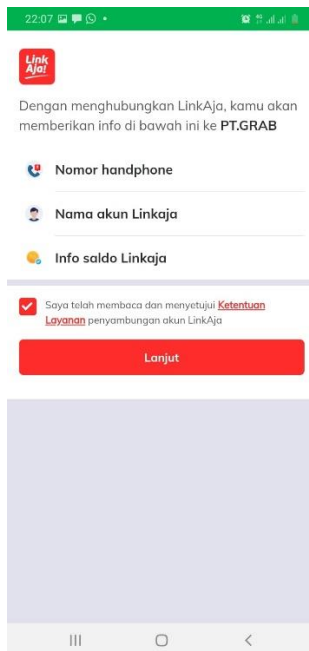
1. User masuk ke halaman akun dan pilih menu Metode Pembayaran



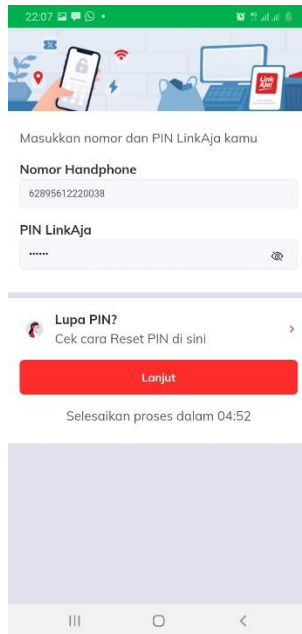
2. User klik button Tambahkan Metode pembayaran



3. Apps menampilkan beberapa metode pembayaran dan Usser memilih salah satunya (misal : LinkAja) dan klik metode pembayaran tersebut => kemudian akan tampil halaman ini dan klik checkbox yang disediakan > klik button lanjut



4. Apps akan masuk ke halaman Input PIN, kemudian masukkan PIN dan klik button Lanjut



5. Setelah berhasil memasukkan PIN dan kode verifikasi > maka Apps akan menampilkan Halaman sukses untuk menambahkan salah satu metode pembayaran.



6. Apps akan otomatis kembali ke halaman metode pembayaran dan menampilkan payment method yang baru saja sukses untuk ditambahkan.

