

## BAB III

### PELAKSANAAN KERJA MAGANG

#### 3.1 Kedudukan dan Koordinasi

Selama program magang berlangsung, peserta menggunakan beberapa aplikasi untuk mendukung proses kerja, di antaranya:

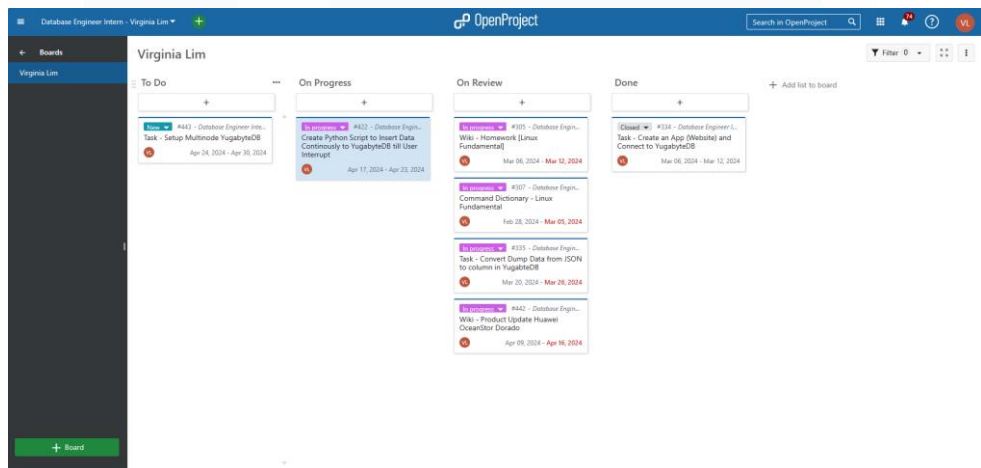
1. OpenProject → *platform* berbasis *web* yang digunakan untuk mencatat perkembangan setiap tugas yang diberikan oleh mentor, serta menyimpan catatan terkait tugas tersebut dalam bagian wiki.
2. Google Meet → alat komunikasi yang digunakan untuk melakukan pembaruan harian terkait tugas-tugas yang sedang dikerjakan.
3. Whatsapp → digunakan untuk bertukar informasi antara peserta magang dan mentor serta melakukan *update* jika tidak bisa mengadakan *meet*.

Alur kerja peserta magang telah ditetapkan oleh mentor, sehingga mereka mengikuti rutinitas harian yang terstruktur. Mentor menggunakan metode kanban dalam mendistribusikan tugas kepada peserta magang. Setiap tugas dikerjakan dalam periode *sprint*, yaitu waktu yang ditetapkan oleh mentor untuk menyelesaikan tugas. Satu *sprint* berlangsung selama lima hari, dimulai dari hari Rabu dan berakhir pada hari Selasa.

Untuk setiap tugas baru, peserta magang diwajibkan membuat task card di board OpenProject. Task card tersebut ditempatkan di bagian "To Do" dengan status "New". Di dalam task card, terdapat detail seperti deskripsi, penugasan, penanggung jawab, serta tanggal mulai dan berakhir *sprint*. Saat peserta magang mulai mengerjakan tugas, mereka harus memindahkan task card ke bagian "On Progress" dan mengubah status menjadi "In Progress". Setelah tugas selesai, task card dipindahkan ke bagian "On Review" tanpa mengubah statusnya. Mentor kemudian akan meninjau tugas tersebut. Jika tidak ada revisi yang diperlukan, mentor akan memindahkan task card ke bagian "Done" dan mengubah status menjadi "Closed". Namun, jika tugas memerlukan perbaikan,

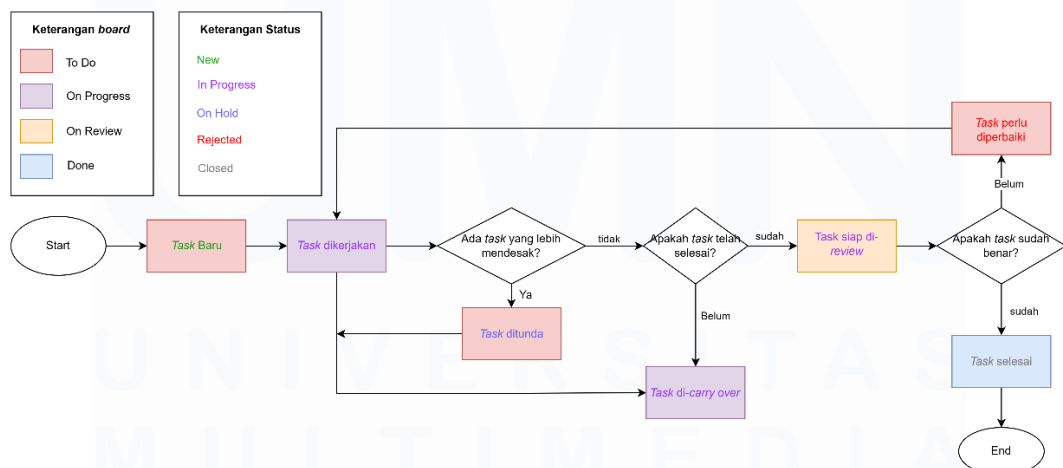
mentor akan mengubah status task card menjadi "Rejected" dan memindahkannya kembali ke bagian "To Do".

Jika tugas tidak dapat diselesaikan dalam satu *sprint*, maka tugas tersebut dapat diteuskan ke *sprint* berikutnya. Apabila tugas melebihi batas waktu yang ditentukan, tanggal pada task card akan ditandai dengan warna merah. Gambar 3.1 merupakan tampilan *board* pada OpenProject:



Gambar 3. 1 Tampilan Board OpenProject

Setiap hari Selasa dan Kamis, peserta magang mengikuti *daily stand-up* yang dilakukan setiap jam 09.00, dimana peserta magang melaporkan kerja yang telah dikerjakan dan rencana kerja kedepannya. Berikut alur magang yang dilaksanakan terdapat pada Gambar 3.2.



Gambar 3. 2 Alur Kerja Magang

### 3.2 Tugas dan Uraian Kerja Magang

Tugas dan tanggung jawab peserta magang adalah membantu mentor dalam menyelesaikan kebutuhan teknis klien. Namun, karena peserta magang masih baru dalam mengenal *tools* dan aplikasi yang digunakan, mereka di berikan waktu untuk belajar untuk menjalani pelatihan dan dasar-dasar yang diperlukan untuk membantu mentor dalam *project*. Tabel 1.1 adalah tabel penjelasan mengenai timeline pengerjaan setiap tugas.

Tabel 3. 1 Tabel Realisasi Kegiatan Magang

No	Task	Tanggal Mulai	Tanggal Selesai
1	<b>Research dan membuat PPT database untuk benchmark</b>	<b>10 Juni 2024</b>	<b>5 Juli 2024</b>
	Mencari 3 <i>database relational</i> DBMS	10 Juni 2024	14 Juni 2024
	Membuat PPT detail mengenai 3 <i>database relational</i>	10 Juni 2024	5 Juli 2024
	Mempresentasikan PPT ke mentor	5 Juli 2024	5 Juli 2024
2	<b>Research dan membuat PPT tools yang digunakan untuk benchmarking</b>	<b>8 Juli 2024</b>	<b>19 Juli 2024</b>
	Mencari 5 <i>tools benchmarking</i>	8 Juli 2024	10 Juli 2024
	Membuat PPT detail mengenai 5 <i>tools benchmarking</i>	10 Juli 2024	18 Juli 2024
	Mempresentasikan PPT ke mentor	19 Juli 2024	19 Juli 2024
3	<b>Belajar Mendix dari Mendix Academy</b>	<b>22 Juli 2024</b>	<b>30 Juli 2024</b>
	Pelatihan Rapid Mendix	22 Juli 2024	24 Juli 2024
	Pelatihan Intermediate Mendix	25 Juli 2024	29 Juli 2024
	Sertifikasi Mendix	30 Jul 2024	30 Juli 2024

No	Task	Tanggal Mulai	Tanggal Selesai
4	<b>Membuat aplikasi <i>shopping app</i> untuk demo</b>	<b>31 Juli 2024</b>	<b>18 Oktober 2024</b>
	Persiapan dan Perencanaan	31 Juli 2024	2 Agustus 2024
	<i>Sprint</i> User Autentication	5 Agustus 2024	9 Agustus 2024
	Memperbaiki <i>bug</i> User Autentication	12 Agustus 2024	13 Agustus 2024
	<i>Sprint</i> Product Catalog	14 Agustus 2024	16 Agustus 2024
	Memperbaiki <i>bug</i> Product Catalog	19 Agustus 2024	20 Agustus 2024
	<i>Sprint</i> Shopping Cart	21 Agustus 2024	23 Agustus 2024
	Memperbaiki <i>bug</i> Shopping Cart	26 Agustus 2024	27 Agustus 2024
	<i>Sprint</i> Checkout Process	28 Agustus 2024	30 Agustus 2024
	Memperbaiki <i>bug</i> Checkout Process	2 September 2024	3 September 2024
	<i>Sprint</i> Payment Method	4 September 2024	6 September 2024
	Memperbaiki <i>bug</i> Payment Method	9 September 2024	10 September 2024
	<i>Sprint</i> Dashboard Admin	11 September 2024	13 September 2024
	Memperbaiki <i>bug</i> dashboard admin	16 September 2024	17 September 2024
	<i>Sprint</i> Profile Dashboard	18 September 2024	19 September 2024
	Memperbaiki <i>bug</i> ya	20 September	23 September
	Menyiapkan aplikasi untuk demo	23 September	25 September
	Demo aplikasi kepada <i>stackholder</i>	26 September 2024	26 September 2024
	Maintenance	27 September 2024	18 Oktober 2024
5	<b>Membuat aplikasi <i>website</i> internal perusahaan “Reforms21”</b>	<b>1 Oktober 2024</b>	<b>18 Oktober 2024</b>

No	Task	Tanggal Mulai	Tanggal Selesai
	Menganalisis <i>design website</i> yang akan dibuat	1 Oktober 2024	1 Oktober 2024
	Menentukan halaman yang akan di duplikasi	2 Oktober 2024	2 Oktober 2024
	Mengumpulkan sumber daya yang dibutuhkan (gambar, logo, dll)	2 Oktober 2024	2 Oktober 2024
	Menyusun struktur navigasi	2 Oktober 2024	7 Oktober 2024
	Menyiapkan project baru dengan Tailwind CSS	8 Oktober 2024	8 Oktober 2024
	Membuat halaman utama dan layout	8 Oktober 2024	9 Oktober 2024
	Membuat halaman informasi lainnya	10 Oktober 2024	15 Oktober 2024
	Penyempurnaan UI/UX	16 Oktober 2024	17 Oktober 2024
	Review dan demo dengan <i>stackholder</i>	18 Oktober 2024	18 Oktober 2024
	Mengumpulkan <i>feedback</i> dan melakukan perbaikan dari saran	18 Oktober 2024	18 Oktober 2024
	Melakukan <i>Build App</i> untuk di <i>deploy</i>	18 Oktober 2024	18 Oktober 2024

Berikut uraian dari setiap task yang telah dilakukan berdasarkan detail timeline yang terdapat pada Tabel 2.1. Penjelasan ini mencakup langkah-langkah yang telah diambil dalam setiap tahapan, termasuk kegiatan persiapan, pelaksanaan, hingga evaluasi yang dilakukan untuk memastikan keberhasilan setiap task. Dengan mengikuti timeline secara sistematis, setiap task dapat diselesaikan sesuai target, mendukung pencapaian tujuan keseluruhan dari proyek atau penelitian ini.

### 3.2.1 Mencari *database* untuk benchmark

Pada *task* ini, peserta magang diberikan tanggung jawab untuk melakukan penelitian mengenai beberapa *database* yang cocok untuk digunakan dalam proses benchmarking bersama dengan YugabyteDB. Penelitian ini mencakup eksplorasi *database* yang populer, baik open source maupun komersial, serta menganalisis performa, fitur, kecepatan, skalabilitas, dan efisiensinya. Setelah melakukan riset, peserta magang diharuskan menyusun hasil penelitian tersebut dalam bentuk presentasi PowerPoint (PPT). PPT ini akan berisi perbandingan antara *database*, termasuk kelebihan dan kekurangan masing-masing, serta alasan mengapa *database* tersebut dipilih atau tidak dipilih untuk benchmark. Selain itu, peserta magang juga diharapkan untuk memberikan rekomendasi mengenai *database* mana yang paling sesuai untuk proses benchmarking berdasarkan analisis yang telah dilakukan.

#### 1. Mencari 3 *database relational* DBMS

Peserta magang melakukan riset mendalam mengenai tiga sistem manajemen basis data relasional (Relational *Database* Management Systems/RDBMS). Langkah ini melibatkan identifikasi dan pemilihan tiga RDBMS yang populer dan relevan untuk kebutuhan benchmarking. Beberapa aspek yang menjadi fokus penelitian meliputi arsitektur sistem, kemampuan distribusi data, performansi dalam menangani beban kerja yang berbeda (seperti OLTP atau OLAP), serta fitur tambahan seperti keamanan, skalabilitas, dan user-friendliness. Proses riset ini biasanya memanfaatkan berbagai sumber, termasuk artikel ilmiah, dokumentasi resmi, dan studi kasus dunia nyata untuk mendukung hasil analisis.

#### 2. Membuat PPT detail mengenai 3 *database relational*

Selanjutnya peserta magang membuat Power Point pada Gambar untuk menyajikan informasi secara rinci namun mudah dipahami. Setiap slide dirancang untuk mencakup poin-poin penting seperti perbandingan

fitur utama, performa teknis, dan kelebihan masing-masing RDBMS. Data hasil riset disajikan dalam bentuk tabel atau grafik perbandingan untuk memberikan visualisasi yang lebih jelas kepada audiens. Selain itu, presentasi juga dapat mencakup studi kasus penggunaannya dalam dunia nyata, memberikan konteks nyata mengenai implementasi masing-masing *database*. Gambar 3.3 merupakan salah satu slide yang menampilkan tabel perbandingan dari 3 *database* yang dibandingkan.

Parameter	<u>YugabyteDB</u>	<u>CockroachDB</u>	<u>TiDB</u>
Database model	Relational DBMS, document store, wide column	Relational DBMS	Relational DBMS, Document store
Lisensi	Open Source	Open Source	Open Source
Pengembang	Yugabyte Inc	Cockroach Labs	PingCAP, Inc
Bahasa Implementasi	C dan C++	Go	Go dan Rust
Sistem Operasi Server	Linux, OS X	Linux, macOS Windows	Linux

YugabyteDB, CockroachDB, TiDB

Gambar 3. 3 Contoh PPT Benchmarking

### 3. Mempresentasikan PPT ke mentor

Peserta magang akan mempresentasikan PPT yang telah dibuat bertujuan untuk menjelaskan hasil analisis secara lisan dengan mendukungnya menggunakan data yang telah disusun. Dalam proses ini, disiapkan narasi yang jelas untuk menjelaskan setiap poin pada slide, serta memberikan wawasan tambahan dari riset yang dilakukan. Mentor akan memberikan masukan untuk perbaikan, yang kemudian dapat digunakan untuk meningkatkan kualitas hasil pekerjaan.

### 3.2.2 Research dan membuat PPT tools yang dapat digunakan untuk benchmark

Pada task ini, peserta magang diberikan tugas untuk melakukan riset tentang berbagai tools yang dapat digunakan untuk melakukan benchmarking terhadap *database*. Peserta magang perlu mencari dan

menganalisis berbagai alat benchmark yang populer dan andal, baik yang bersifat open-source. Alat-alat tersebut harus dievaluasi berdasarkan kemampuan mereka dalam mengukur performa *database*, termasuk kemampuan dalam mengukur throughput, latensi, dan scalability.

Setelah melakukan riset, peserta magang diminta untuk menyusun hasil penelitian ke dalam bentuk presentasi dalam bentuk PPT mengenai masing-masing tools termasuk kelebihan, kekurangan, fitur utama, serta cara kerja dari setiap alat. Selain itu, peserta juga perlu memberikan rekomendasi tools yang paling cocok untuk digunakan dalam proses benchmarking.

1. Mencari 5 *tools benchmarking*

Pada task ini, peserta magang melakukan riset tentang lima tools benchmarking yang dapat digunakan untuk mengukur performa sistem. Benchmarking tools ini harus relevan dengan kebutuhan, seperti pengukuran performa basis data, aplikasi, atau infrastruktur tertentu. Proses ini mencakup identifikasi tools yang populer dan diakui dalam industri, kemudian mempelajari fitur utama, kelebihan, dan kekurangannya. Informasi mengenai tools ini dapat ditemukan melalui berbagai sumber, seperti dokumentasi resmi, artikel teknis, ulasan pengguna, atau studi kasus.

2. Membuat PPT detail mengenai 5 *tools benchmarking*

Peserta magang kemudian menyusun presentasi dalam format PowerPoint (PPT) yang ditunjukkan pada Gambar 3.4. Presentasi ini dirancang untuk memberikan informasi rinci mengenai kelima tools benchmarking yang telah dipilih. Setiap tool akan dijelaskan berdasarkan:

- a. Deskripsi umum dan fungsionalitas utama.
- b. Jenis metrik yang dapat diukur (misalnya performa CPU, memori, throughput, latensi).
- c. Lingkup penggunaannya, seperti apakah untuk basis data, aplikasi, atau sistem operasi.
- d. Keunggulan dan kekurangannya.



- e. Contoh kasus penggunaan di dunia nyata. Data disusun secara informatif dengan tabel atau grafik perbandingan untuk membantu audiens memahami kekuatan dan kelemahan masing-masing tools secara visual.

**TABEL PERBANDINGAN**

Parameter	<u>TPC-C</u>	<u>sysbench</u>	<u>YCSB</u>	<u>HammerDB</u>	<u>pgbench</u>
Fokus Utama	OLTP	CPU, IO	YSQL performance	OLTP	OLTP PostgreSQL
Model	Relational	Relational/key-value	Relational	Relational	Relational
Usecase	Evaluasi sistem OLTP	Evaluasi peforma database dan sistem	Evaluasi peforma dalam berbagai skenario	OLTP	OLTP

Gambar 3. 4 Tabel Perbandingan Tools Benchmarking

### 3. Mempresentasikan PPT ke mentor

Presentasi ini dirancang untuk memberikan penjelasan terstruktur dan menyeluruh, dengan narasi yang mendukung setiap slide. Tujuan dari presentasi ini adalah memastikan bahwa mentor memahami pilihan tools yang diajukan dan mendapatkan masukan atau rekomendasi untuk memperbaiki atau melengkapi analisis. Diskusi dengan mentor juga berfungsi untuk memvalidasi bahwa tools yang diusulkan relevan dengan kebutuhan spesifik organisasi atau proyek.

#### 3.2.3 Belajar Mendix dari Mendix Academy

Pada task ini, peserta magang diberikan kesempatan untuk mempelajari mendix, sebuah *platform* low-code yang digunakan untuk pengembangan aplikasi secara cepat dan efisien. Peserta magang memulai proses belajar dengan mengikuti sertifikasi resmi dari Mendix Academy. Melalui sertifikasi ini, peserta diharapkan memahami dasar-dasar Mendix, termasuk bagaimana membangun aplikasi menggunakan

drag-and-drop interface, pengelolaan data, dan pembuatan microflow yang menjadi bagian penting dalam alur kerja aplikasi.

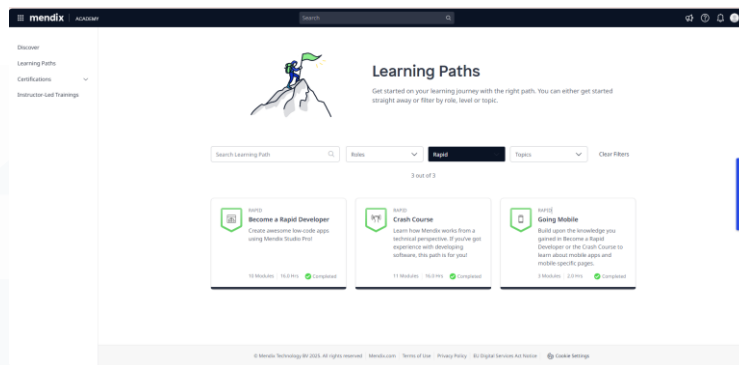
Selain itu, peserta magang juga belajar bagaimana memanfaatkan Mendix untuk berbagai kebutuhan bisnis, seperti pengembangan aplikasi berbasis *web* dan *mobile* dengan cepat, kolaborasi tim dalam pengembangan, serta integrasi dengan sistem lain. Dengan pemahaman yang baik mengenai Mendix, peserta magang akan mampu mendukung proyek pengembangan aplikasi selama masa magang dan mempersiapkan diri untuk proyek-proyek berbasis Mendix di masa depan.

Tugas ini melibatkan pembelajaran teknologi *low-code* menggunakan *platform* Mendix. Proses ini dilakukan melalui kursus yang tersedia di Mendix Academy, yang mencakup pelatihan terstruktur dari tingkat dasar hingga menengah. Berikut penjelasan dari setiap tahapannya:

#### 1. **Pelatihan Rapid Mendix**

Pelatihan ini adalah tahap awal yang dirancang untuk pemula. Tujuannya adalah memahami dasar-dasar pengembangan aplikasi menggunakan Mendix.

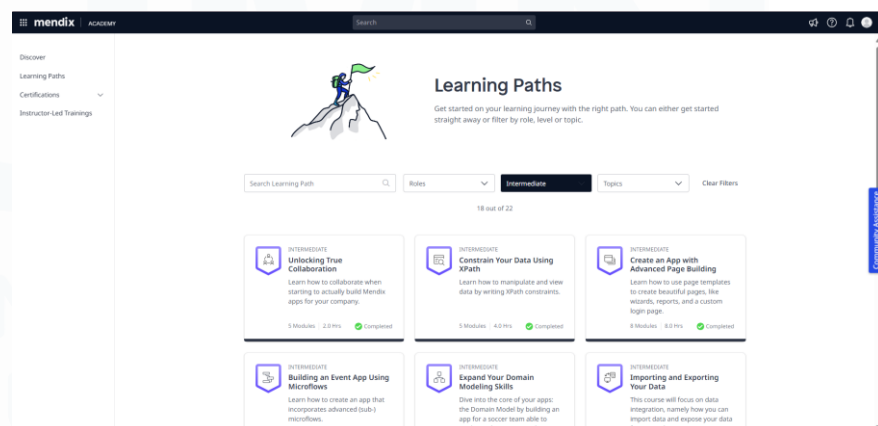
Materi meliputi pengenalan antarmuka Mendix Studio, pembuatan domain model, microflow, dan pengaturan halaman (*pages*). Peserta diajak untuk membangun aplikasi sederhana dengan praktik langsung sehingga dapat memahami alur kerja *platform* secara menyeluruh. Pada Gambar 3.5 merupakan tampilan dari *learning paths* Mendix Academy.



Gambar 3. 5 Tampilan dari Course Rapid

## 2. Pelatihan Intermediate Mendix

Setelah menyelesaikan pelatihan Rapid, pelatihan Intermediate difokuskan pada pengembangan keterampilan yang lebih kompleks. Tampilan pelatihan *intermediate* ditunjukkan pada Gambar 3.6. Gambar 3.6 menampilkan beberapa *course* yang harus dikerjakan pada tahap *intermediate*. Materi mencakup topik seperti integrasi dengan sistem eksternal melalui API, pengelolaan *database* yang lebih mendalam, logika bisnis yang kompleks menggunakan microflow, dan desain user interface yang lebih canggih. Latihan dalam pelatihan ini berbasis proyek sehingga peserta bisa menerapkan konsep-konsep lanjutan ke dalam aplikasi nyata.



Gambar 3. 6 Tampilan course Intermediate

### 3. Sertifikasi Rapid Mendix

- a. Setelah menyelesaikan pelatihan Rapid, peserta dapat mengikuti ujian sertifikasi resmi dari Mendix.
- b. Sertifikasi ini bertujuan untuk menguji pemahaman dan kemampuan dalam menggunakan Mendix, termasuk pembuatan aplikasi dasar, pemodelan data, dan pengelolaan alur kerja.
- c. endapatkan sertifikasi Rapid Mendix menunjukkan bahwa peserta telah menguasai kompetensi dasar yang diperlukan untuk menjadi seorang pengembang low-code.

Pada Gambar 3.7 menunjukkan sertifikat jika telah menyelesaikan course RAPID di Mendix Academy. Melalui pembelajaran ini, peserta tidak hanya menguasai kemampuan teknis, tetapi juga memahami bagaimana Mendix dapat digunakan untuk membangun aplikasi secara cepat dan efisien. Dengan menyelesaikan pelatihan dan sertifikasi, peserta memperoleh pengakuan profesional yang dapat meningkatkan kredibilitas mereka dalam pengembangan aplikasi low-code, baik di dunia kerja maupun dalam proyek personal.



Gambar 3. 7 Sertifikat Mendix Academy

### 3.2.4 Membuat aplikasi *shopping app* untuk demo

Pada task ini, peserta magang diberikan tanggung jawab untuk membuat aplikasi *shopping app* sebagai bagian dari demo untuk salah satu klien perusahaan menggunakan *platform* Mendix. Aplikasi ini dirancang untuk mendemonstrasikan bagaimana *platform* Mendix dapat digunakan dalam pengembangan aplikasi *e-commerce* yang fungsional. Aplikasi ini memiliki tiga *role* utama yaitu: user, admin, dan seller. Masing masing dengan fitur dan hak akses yang berbeda:

1. User

Pengguna dengan *role* ini dapat melakukan pencarian produk, melihat detail produk, menambahkan produk ke keranjang belanja, dan melakukan pembelian.

2. Admin

Admin memiliki akses untuk mengelola seluruh konten aplikasi, termasuk memonitor aktivitas user dan seller, serta manage *database* produk dan transaksi.

3. Seller

Penjual dengan *role* seller dapat menambahkan produk baru, mengelola stok, serta melihat dan memproses pesaran yang masuk dari user di tokonya.

4. *Anonymous*

*Anonymous* merupakan *role* yang digunakan untuk user yang belum melakukan sign in. Selain itu, *role* ini digunakan juga untuk menguji flow sistem.

Aplikasi ini dibuat dengan menggunakan alur kerja Mendix, dimana setiap *role* memiliki fitur-fitur yang disesuaikan dengan kebutuhan mereka. Pengembangan ini juga melibatkan pembuatan halaman login, pengelolaan produk, keranjang belanja, hingga sistem checkout dan lain-lain, semuanya dilakukan dengan Mendix secara low-code. Tugas utama dalam task ini:

1. Membuat aplikasi *shopping app* dengan peran user, admin, dan seller.
2. Mengembangkan fitur seperti pencarian produk, pengelolaan produk oleh seller, dan manajemen pengguna oleh admin.
3. Menerapkan alur kerja dan navigasi yang memudahkan pengguna dalam berbelanja dan mengelola produk.

Setiap task diharapkan memberikan kontribusi yang signifikan dalam mendukung keberhasilan proyek secara keseluruhan. Selain itu, implementasi yang efektif di setiap tahapan akan menghasilkan outcome yang sesuai dengan standar yang diinginkan. Hasil yang diharapkan dari pelaksanaan task ini sebagai berikut:

1. Aplikasi *shopping app* yang fungsional sebagai demo
2. Pembagian peran user, admin, dan seller yang berjalan dengan baik.
3. Peningkatan pemahaman peserta dalam mengembangkan aplikasi *e-commerce* di Mendix.

Setiap bagian aplikasi dirancang untuk memastikan pengalaman pengguna yang lancar dan efisien. Dengan tampilan antarmuka yang mudah digunakan dan responsif di berbagai perangkat untuk memberikan kemudahan dan kenyamanan bagi pengguna dalam melakukan belanja online.

### **1. Persiapan dan Perencanaan**

Persiapan dan perencanaan merupakan tahap pertama untuk membuat aplikasi *e-commerce*. Pada bagian akan dikembangkan metode pengembangan sistem yaitu menggunakan metode Agile. Mendix merekomendasikan penggunaan metode Agile dengan model scrum karena sangat cocok untuk pengembangan sistem yang memerlukan fleksibilitas dan perubahan yang sering terjadi selama proses pengembangan. Berikut merupakan langkah-langkah yang dilakukan dengan metode pengembangan Agile model scrum. Gambar 3.8 menunjukkan scrum board pada Mendix

Dashboard. Dimana terdapat 4 tahapan yaitu To Do, In Progress, Refinement, dan Done. Tahapan-tahapan tersebut mewakili proses-proses tersendiri. Berikut penjelasan tahap-tahapnya:

a. To Do

Tahap ini berisi daftar tugas yang telah diidentifikasi dan di prioritaskan untuk diselesaikan dalam *sprint*. Tugas-tugas ini masih menunggu untuk dikerjakan. Tim pengembang akan memulai pekerjaan mereka dengan memindahkan tugas dari daftar To Do ke In Progress sesuai dengan prioritas dan kapasitas yang tersedia.

b. In Progress

Pada tahap ini, tugas-tugas yang telah dipilih untuk dikerjakan dalam *sprint* mulai dikerjakan oleh tim. Setiap anggota tim yang terlibat akan fokus tugas yang ada di kolom ini. Tahap ini menunjukan aktivitas pengembangan atau perbaikan yang sedang dilakukan.

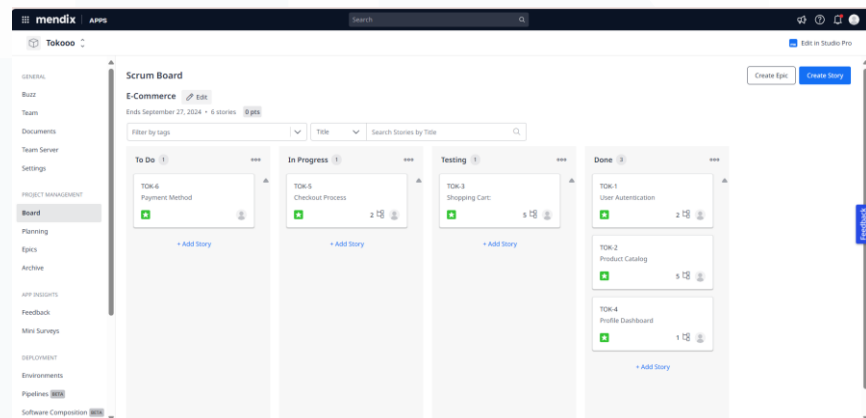
c. Refinement

Tahap refinement merupakan fase dimana tim melakukan pembaruan atau penyempurnaan terhadap tugas-tugas yang ada, mengklarifikasi persyaratan yang belum jelas, dan memecah tugas besar menjadi bagian yang lebih kecil atau lebih terperinci. Proses ini membantu agar backlog tetap relevan dan siap untuk dilaksanakan dalam *sprint* berikutnya. Tugas yang ada di sini bisa dianggap sebagai tugas yang masih membutuhkan beberapa penyesuaian sebelum dipindahkan ke In Progress.

d. Done

Ketika tugas telah selesai dikerjakan dan memenuhi definisi selesai (Definition of Done/DoD), tugas tersebut dipindahkan ke kolom Done. Tahap ini menandakan bahwa pekerjaan pada tugas tersebut telah selesai, telah diuji, dan siap untuk

digunakan atau dipresentasikan. Tugas yang sudah berada di kolom Done berarti telah menyelesaikan semua kriteria penerimaan dan siap untuk diterima oleh stakeholder atau dimasukkan ke dalam aplikasi.



Gambar 3. 8 Dashboard Scrum *E-commerce*

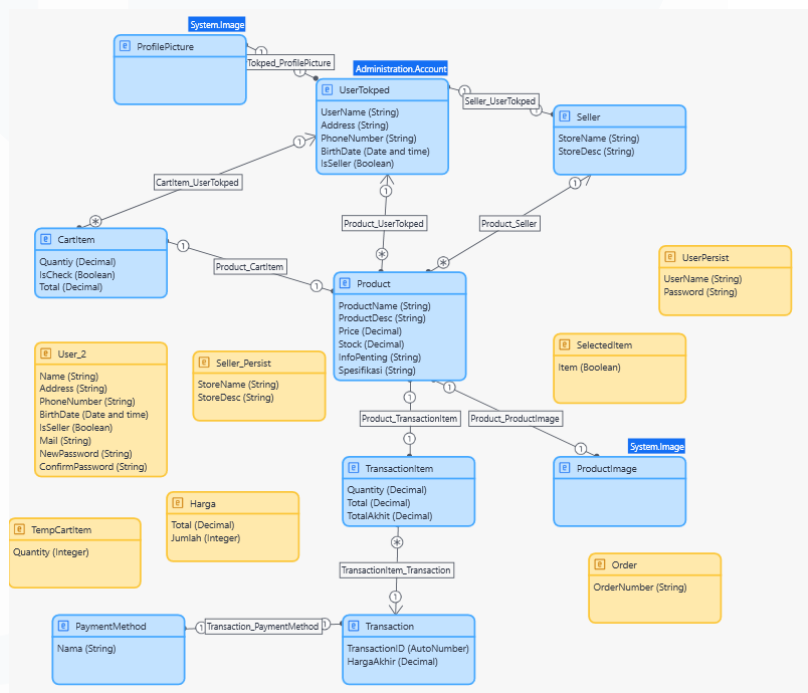
Pada tahap ini juga dilakukan pengembangan domain model atau entitas data menggunakan Mendix. Pada Gambar 3.9 menampilkan entitas data pada aplikasi *e-commerce*. Entitas data dapat dibagi menjadi dua jenis, yaitu persisted dan non-persisted:

- a. Persisted Entities adalah entitas yang datanya disimpan secara permanen dalam *database* aplikasi. Ini adalah entitas yang berisi informasi yang perlu disimpan dan dikelola sepanjang waktu. Contoh tabel yang termasuk persisted entities dalam aplikasi shopping app antara lain:
  - a) Product → Menyimpan data produk seperti nama, deskripsi, harga, stok, dan gambar.
  - b) Order → Mencatat transaksi pembelian, termasuk informasi pembeli, produk yang dibeli, jumlah, dan total harga.
  - c) User → Menyimpan data pengguna seperti nama, email, *role* (penjual atau pembeli), dan informasi login.
- b. PaymentMethod → Berisi informasi mengenai metode pembayaran yang



c. Non-persisted Entities adalah entitas yang tidak menyimpan data secara permanen dalam *database*. Mereka digunakan untuk menyimpan data sementara atau untuk memproses informasi sementara dalam aplikasi. Contoh tabel yang termasuk non-persisted entities adalah TempCartItem. TempCart Item digunakan untuk menyimpan item sementara yang ditambahkan ke keranjang belanja oleh pembeli, tetapi tidak disimpan secara permanen sampai pesanan dibuat

Secara umum, persisted entities mencakup data yang diperlukan untuk pengelolaan aplikasi yang berkelanjutan dan harus disimpan dalam *database*, sementara non-persisted entities digunakan untuk proses atau interaksi yang hanya diperlukan dalam jangka pendek.



Gambar 3. 9 Domain Model Shopping App

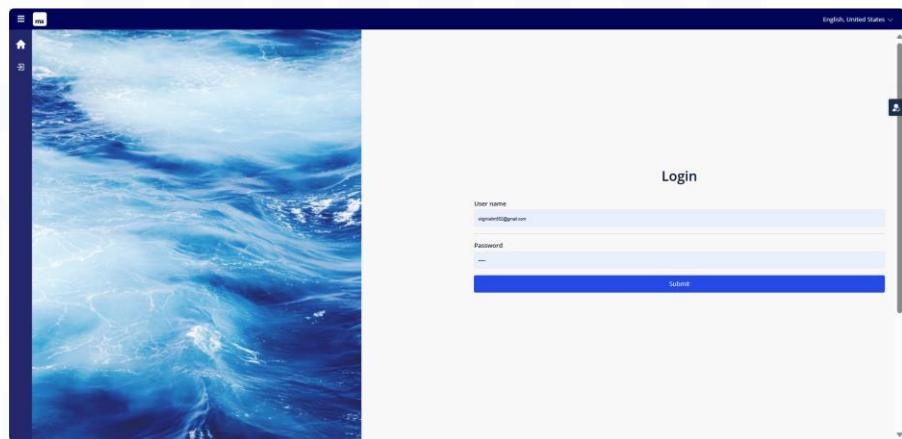
## 2. *Sprint* User Autentication

### a. Login

Login Halaman login dalam aplikasi shopping app yang dibuat menggunakan Mendix berfungsi sebagai pintu masuk utama bagi pengguna untuk mengakses shopping app. Setiap

pengguna dapat masuk terlebih dahulu tanpa melakukan login, namun setiap pengguna harus melakukan autentikasi dengan memasukkan username dan password yang telah terdaftar.

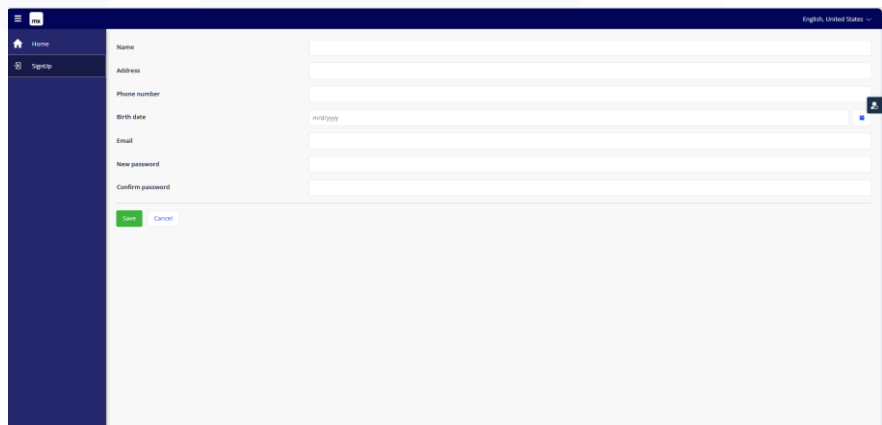
Halaman ini juga berfungsi untuk memverifikasi peran pengguna dan menyesuaikan pengalaman pengguna berdasarkan hak akses mereka. Fitur utama di halaman login adalah form Login dimana terdiri dari kolom untuk memasukkan username dan password. Jika kombinasi yang dimasukan benar, pengguna akan diarahkan ke halaman utama sesuai dengan perannya. Namun, jika username dan password salah, akan diberikan notification bahwa password salah atau username salah. Selain itu terdapat autentikasi *role-based* dimana sistem memastikan bahwa setelah login, pengguna hanya memiliki akses ke fitur yang sesuai dengan peran mereka di aplikasi. Misalnya: user akan diarahkan ke halaman beranda dengan opsi untuk berbelanja. Seller akan diarahkan ke dashboard untuk mengelola produk. Gambar 3.10 menunjukkan tampilan halaman login. Dengan halaman login, aplikasi memastikan bahwa hanya pengguna yang sah dengan akses yang tepat dapat masuk dan menggunakan fitur yang sesuai dengan peran mereka, menjaga keamanan dan efisiensi dalam penggunaan aplikasi.



Gambar 3. 10 Tampilan Halaman Login

## b. SignIn

Halaman sign in berfungsi sebagai tempat di mana pengguna baru bisa mendaftar dan membuat akun di aplikasi shopping app yang dibuat menggunakan Mendix yang ditampilkan pada Gambar 3.11. Halaman ini dirancang untuk mengumpulkan informasi dasar dari pengguna dan membuat profile baru yang akan menentukan peran mereka di dalam aplikasi. Pada halaman sign in, pengguna diminta untuk mengisi informasi seperti nama lengkap, email, nomor telepon, username, alamat, password, konfirmasi password. Sistem akan memvalidasi input seperti memastikan bahwa email atau username belum digunakan sebelumnya, dan bahwa password sesuai dengan persyaratan keamanan (misalnya panjang minimum dan kombinasi karakter).

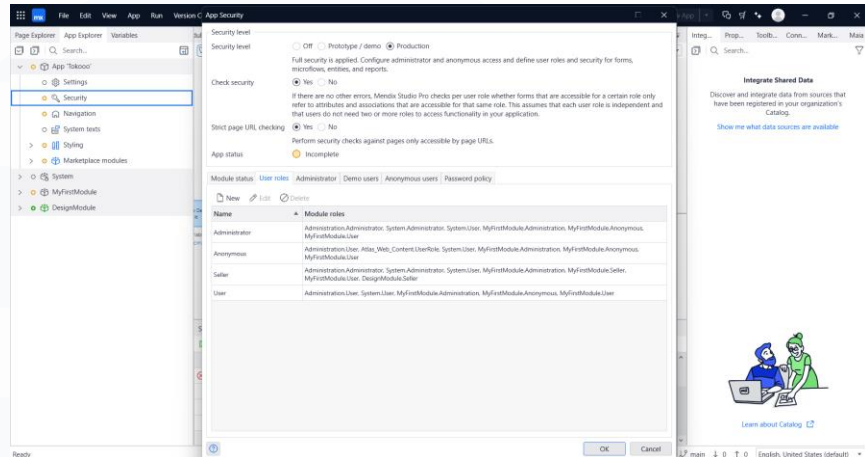


Gambar 3. 11 Tampilan Halaman Sign In

## c. *Role User*

*Role user* merupakan hak akses atau peran yang diberikan kepada pengguna dalam aplikasi. *Role user* dapat ditambahkan atau dihapus sesuai dengan kebutuhan. Pada Gambar 3.12 menunjukkan *role* yang telah dibuat yaitu administrator, anonymous, seller, dan user. Setiap *role* memiliki akses ke bagiannya masing masing. Contohnya *role user* tidak dapat

mengakses *role* yang di berikan kepada seller seperti menambah produk ataupun menghapus produk.



Gambar 3. 12 Halaman *Role User*

### 3. Memperbaiki bug User Autentication

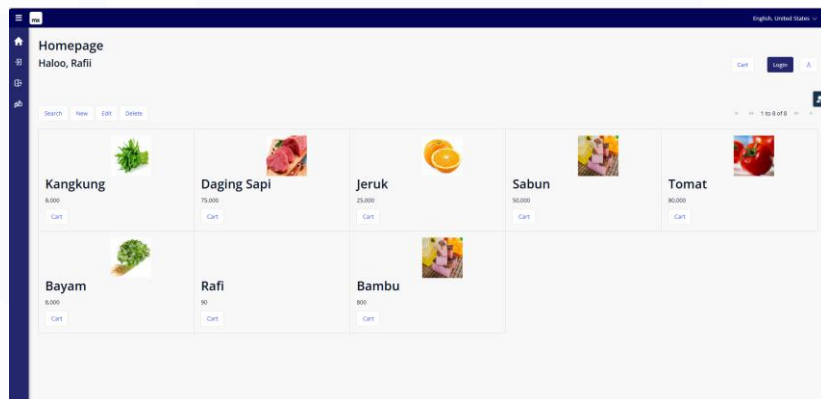
Error yang ditemukan adalah error pada saat registrasi, pengguna baru tidak menerima pesan kesalahan saat mendaftar. Hal ini menyebabkan pengguna tidak mengetahui apakah sudah terdaftar atau belum.

### 4. *Sprint Product Catalog*

a. Halaman home pada aplikasi ini merupakan tampilan utama yang menyambut pengguna setelah mereka berhasil login. Gambar 3.13 menampilkan halaman home yang dirancang untuk memberikan akses cepat dan mudah ke berbagai fitur dan informasi penting yang relevan dengan peran pengguna, apakah mereka seorang user, seller, atau admin. Terdapat beberapa fitur pada halaman home:

a) Navigasi utama → Halaman ini menampilkan navigasi yang mudah diakses untuk berbagai kategori dan product. pengguna dapat melihat dan memilih produk dari berbagai kategoriseperti elektronik, fashion, atau kebutuhan rumah tangga.

- b) Search bar → pengguna dapat melakukan pencarian cepat terhadap produk yang mereka inginkan melalui kolom pencarian yang tersedia di bagian atas halaman, Search bar ini terhubung dengan *database* untuk menampilkan hasil yang relevan.
- c) Tombol cepat → ada beberapa tombol cepat yang memudahkan pengguna untuk langsung menuju ke halaman-halaman penting seperti keranjang belanja, profile, dan membuat toko.



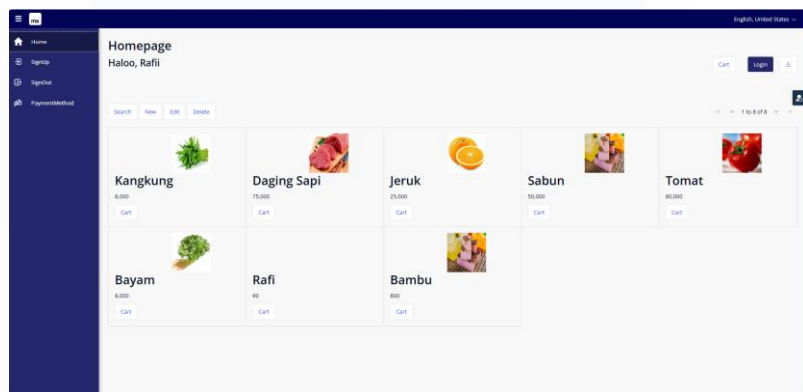
Gambar 3. 13 Tampilan Halaman Homepage

## b. Navigation Bar

*Navigation bar* pada aplikasi ini berfungsi sebagai elemen utama untuk memudahkan pengguna mengakses berbagai halaman dan fitur dalam aplikasi, Letaknya berada dibagian kanan halaman dan tetap terlihat meskipun pengguna berpindah antar halaman seperti pada Gambar 3.14. *Navigation bar* dirancang untuk *intuited* dan *responsive*, memberikan pengalaman yang nyaman bagi pengguna, baik itu user, seller, ataupun admin. Pada *navigation bar* diaplikasi ini terdapat :

- a) Home → yang akan mengarahkan pengguna kembali ke halaman utama untuk melihat produk yang dijual.

- b) Sign In → yang akan mengarahkan pengguna kembali ke halaman sign in. Sehingga, pengguna dapat membuat akun baru secara langsung.
- c) Sign Out → pengguna dapat langsung melakukan sign out tanpa masuk ke halaman manapun.
- d) PaymentMethod → payment method hanya dapat diakses oleh user yang telah mendapatkan *role* seller. Sehingga, user yang memiliki *role* seller akan langsung diarahkan ke halaman untuk menambahkan payment method. Jika pengguna hanya memiliki *role* user, maka tidak dapat mengakses halaman payment.



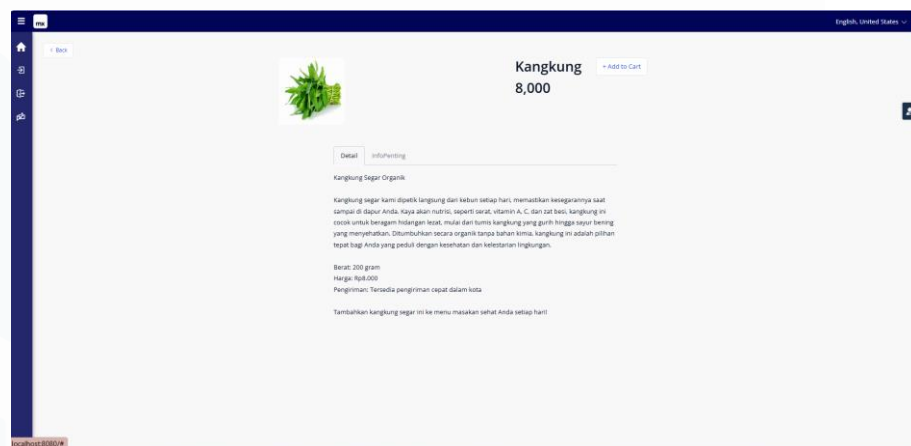
Gambar 3. 14 Tampilan Navigation Bar

### c. Detail Produk

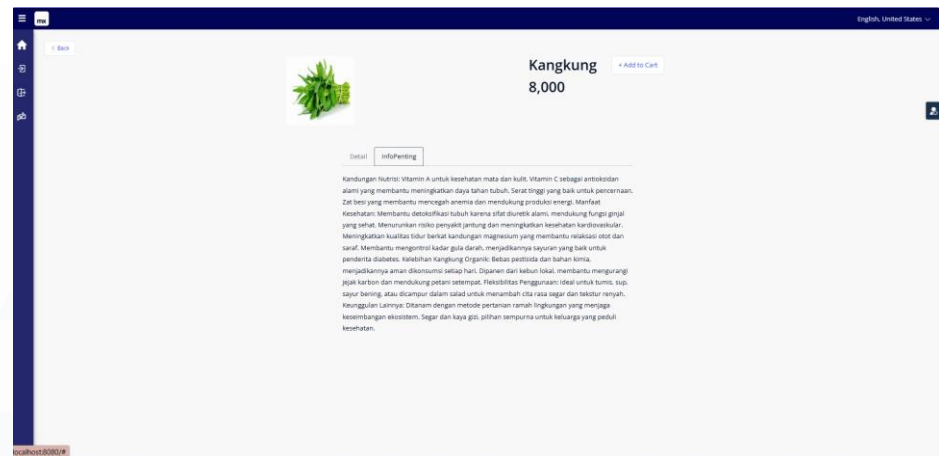
Pada halaman detail produk dalam aplikasi ini dirancang untuk memberikan informasi lengkap tentang produk yang dipilih oleh pengguna. Halaman ini sangat penting karena membantu pengguna mengambil keputusan pembelian dengan memberikan detail mendalam terkait produk, termasuk deskripsi, harga, spesifikasi, serta opsi pembelian. Beberapa isi elemen-elemen penting dari halaman detail produk:

- a) Gambar produk → menampilkan gambar produk dengan resolusi tinggi.

- b) Nama produk dan harga → nama produk ditampilkan dengan jelas di bagian atas halaman, diikuti oleh harga dengan jelas. Hal ini memberikan gambaran umum kepada pengguna.
- c) Deskripsi lengkap produk → pengguna dapat melihat deskripsi lengkap tentang produk yang mencakup detail lebih mendalam seperti spesifikasi teknis, bahan, ukuran, atau fitur-fitur khusus lainnya. Gambar 3.15 ini membantu pengguna untuk memahami dengan lebih baik apa yang akan mereka beli.
- d) Info penting → pada bagian ini, produk akan dijelaskan lebih kompleks yang dapat disajikan dalam bentuk tabel yang rapi dan mudah dibaca yang ditampilkan pada Gambar 3.16. Hal ini sangat membantu pengguna dalam membandingkan produk.
- e) Tombol “Add to Cart” dan “Buy Now” → tombol Add to Cart memungkinkan pengguna menambahkan produk ke keranjang belanja untuk dibeli nanti.



Gambar 3. 15 Tampilan halaman detail produk “Detail”



Gambar 3. 16 Tampilan halaman detail produk “info penting”

## 5. Memperbaiki bug Product Catalog

Error yang ditemukan pada *sprint* ini adalah gambar produk yang tidak muncul pada katalog atau daftar produk karena masalah pada logic backendnya. Selain itu, pengguna tidak dapat melakukan pencarian karena terdapat kesalahan dalam melakukan *retrieve* dari *database*.

## 6. *Sprint* Shopping Cart

### a. Add to cart

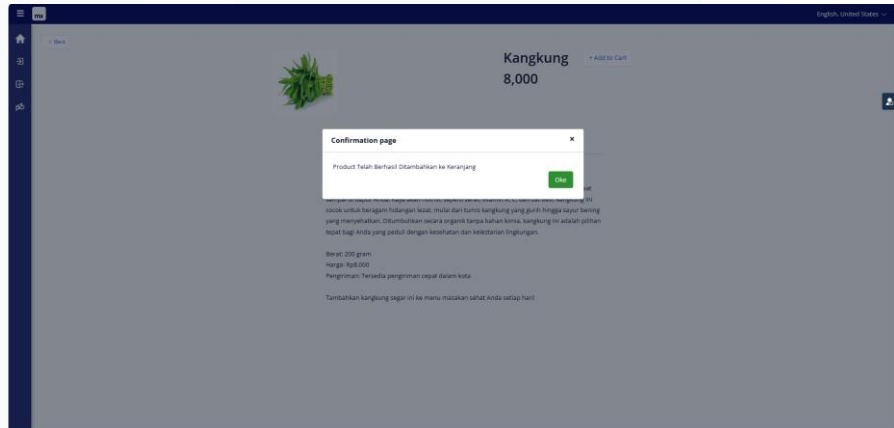
Tombol “add to cart” di halaman detail produk berfungsi sebagai elemen interaktif yang memungkinkan pengguna mengambahkan produk yang mereka pilih ke keranjang belanja sebelum melanjutkan ke proses checkout. Tombol ini memainkan peran penting dalam pengalaman berbelanja online dan dirancang untuk mudah diakses dan digunakan.

Tombol “add to cart” diletakan di dekat informasi harga dan nama produk, sehingga pengguna dapat dengan mudah menemukannya setelah membaca detail produk. Saat pengguna mengklik tombol, produk yang sedang dilihat langsung ditambahkan ke keranjang belanja pengguna. Pengguna dapat



menambahkan lebih dari satu produk, dan sistem akan otomatis memperbarui jumlah item yang ada di keranjang.

Setelah pengguna menekan tombol, aplikasi akan memberikan pop-up bahwa produk berhasil ditambahkan ke keranjang yang dapat dilihat pada Gambar 3.17.



Gambar 3. 17 Tampilan pop up confirmation page

## b. Cart

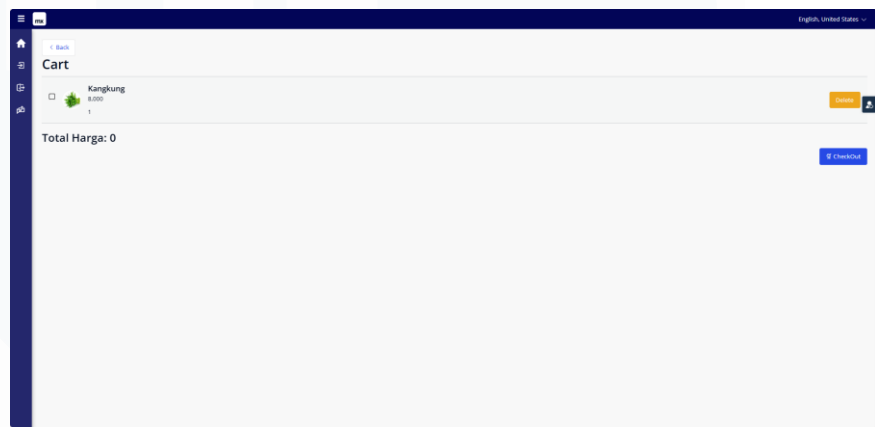
Halaman cart adalah tempat dimana pengguna dapat melihat produk yang telah mereka tambahkan sebelum melanjutkan ke proses pembayaran yang pada Gambar 3.18. Fungsi utama dari halaman ini adalah memberikan ringkasan barang-barang yang dipilih oleh pengguna, mengelola kuantitas, dan memastikan total biaya sebelum pembelian.

Setiap produk yang telah ditambahkan ke cart oleh pengguna ditampilkan dalam bentuk list. Untuk setiap item, ditampilkan informasi penting seperti nama produk, gambar produk, harga per unit, kuantitas, dan sub total (harga produk dikalikan dengan kuantitas). Pengguna juga dapat menghapus produk dari cart jika mereka memutuskan untuk tidak membeli item tersebut. Terdapat tombol “delete” di sebelah setiap produk untuk mempermudah pengguna menghapus item dari keranjang.

Di bagian bawah halaman cart, pengguna bisa melihat total harga keseluruhan dari semua produk yang ada di keranjang. Subtotal

dihitung dari total harga per produk berdasarkan kuantitas, sementara Total mencakup semua harga dari sub total yang ditampilkan pada Gambar 3.19.

Halaman cart juga menyertakan tombol “checkout” yang memungkinkan pengguna melanjutkan ke proses pembayaran. Pengguna biasanya diarahkan ke halaman pengisian detail pengiriman, metode pembayaran, dan konfirmasi pesanan setelah menekanan tombol ini.



Gambar 3. 18 Tampilan cart



Gambar 3. 19 Tampilan cart setelah di check

## 7. Memperbaiki bug Shopping Cart

Memperbaiki bug ini melibatkan pemecahan masalah pada bagian penyimpanan data keranjang (misalnya menggunakan session atau *database*), memastikan perhitungan harga berjalan dengan benar,

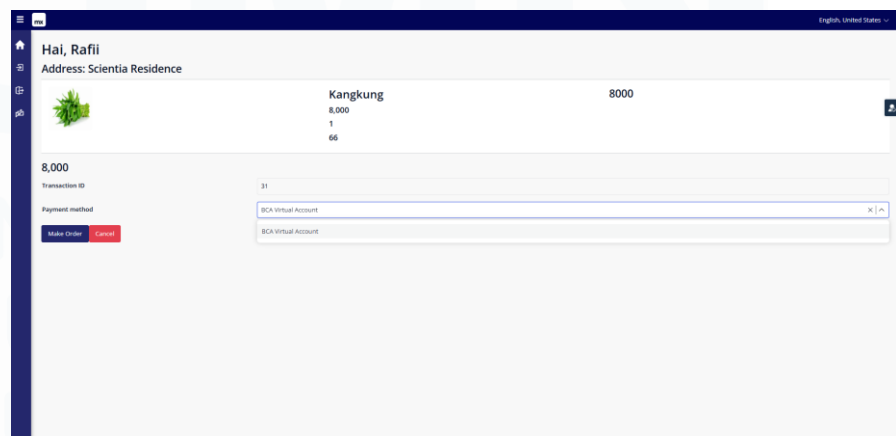
serta memperbaiki tampilan dan interaksi pengguna yang terkait dengan keranjang belanja.

## 8. *Sprint Checkout Process*

### a. CheckOut

Halaman konfirmasi adalah langkah akhir sebelum pengguna menyelesaikan pembelian dalam aplikasi *shopping app* yang ditunjukkan pada Gambar 3.20. Di halaman ini, pengguna akan meninjau semua detail pesanan mereka, termasuk informasi pengiriman, metode pembayaran, dan rincian produk. Pada halaman ini, pengguna akan melihat ringkasan pesanan, yang mencakup daftar produk yang mereka pilih beserta detail seperti nama produk, kuantitas, harga per item, dan subtotal. Terdapat juga tampilan Alamat pengiriman yang telah diinput oleh pengguna.

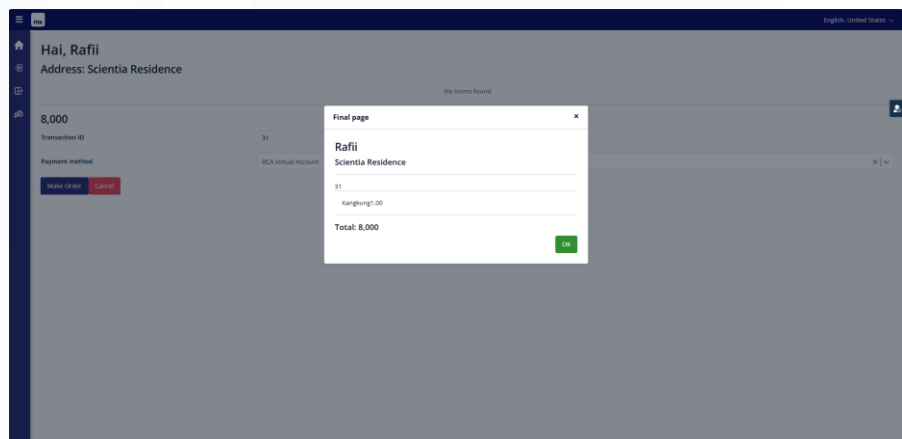
Informasi mengenai metode pembayaran yang dipilih pengguna juga juga ditampilkan di halaman ini. Pengguna dapat memilih metode pembayaran yang diinginkan. Terdapat opsi untuk mengubah metode pembayaran jika diperlukan. Setelah pengguna yakin atas pesanan yang diinginkan, terdapat tombol “confirm order” yang digunakan untuk menyelesaikan pesanan mereka. Setelah tombol ditekan, pesanan akan diproses, dan pembayaran akan dilakukan sesuai dengan metode yang dipilih.



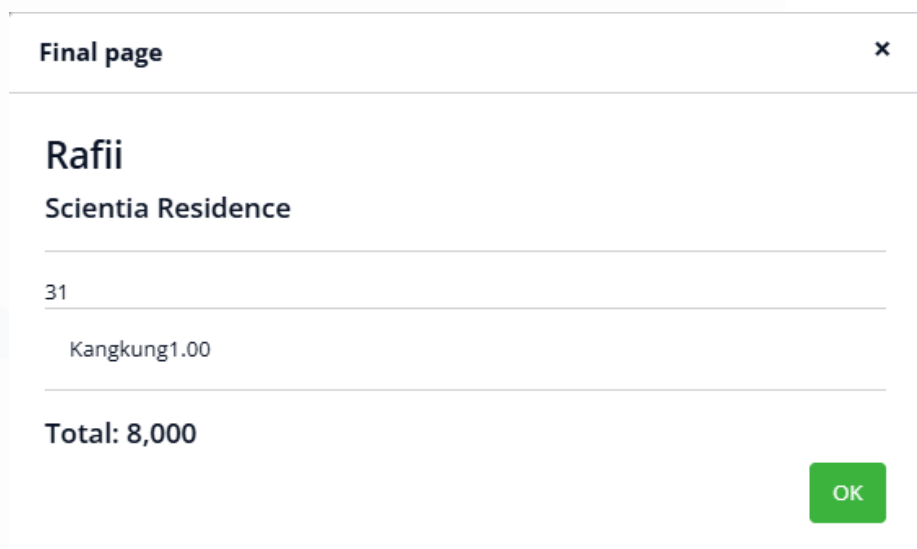
Gambar 3. 20 Tampilan halaman checkout

## b. Order Page

Halaman order dalam aplikasi ini merupakan tempat dimana pengguna dapat melihat dan mengelola pesanan mereka setelah melakukan konfirmasi. Halaman ini menyediakan ringkasan lengkap mengenai pesanan yang telah dilakukan oleh pengguna, status pengiriman, serta opsi untuk melacak pesanan. Pada Gambar 3.21 menampilkan pop-up dimana *user* telah melakukan konfirmasi checkout dan akan diarahkan ke halaman *order page* seperti pada gambar 3.22.



Gambar 3. 21 Tampilan pop-up order page



Gambar 3. 22 Tampilah halaman order page

## 9. Memperbaiki bug Checkout Process

Memperbaiki bug ini melibatkan pengecekan dan validasi setiap langkah dalam proses checkout, memastikan perhitungan harga dan biaya dilakukan dengan benar, serta memeriksa integrasi dengan sistem pembayaran dan pengiriman.

## 10. *Sprint* Admin Dashboard

### a. Add Product

Fitur "Add Product" pada aplikasi shopping app adalah salah satu komponen utama yang dirancang untuk mendukung penjual (seller) dalam menambahkan produk baru ke dalam marketplace atau katalog aplikasi. Fitur ini memastikan bahwa seller dapat dengan mudah mengelola inventaris produk mereka melalui antarmuka yang intuitif dan efisien.

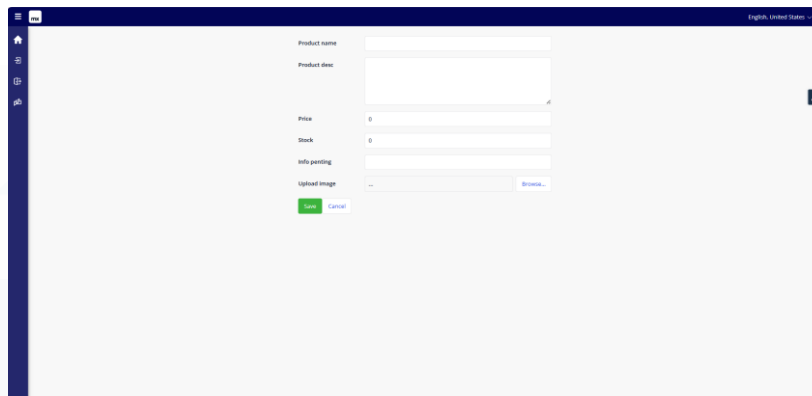
Pada halaman "Add Product" yang ditunjukkan pada Gambar 3.23 penjual disajikan dengan sebuah formulir yang berisi berbagai field input yang relevan untuk mendeskripsikan produk mereka. Field-field tersebut mencakup nama produk, deskripsi produk, kategori, harga, jumlah stok, dan opsi untuk mengunggah gambar produk. Nama produk membantu pembeli mengenali barang yang dijual, sedangkan deskripsi produk memberikan informasi detail tentang spesifikasi atau manfaat barang tersebut. Kategori biasanya disajikan dalam bentuk dropdown untuk mempermudah pengelompokan produk ke dalam kategori tertentu, seperti elektronik, pakaian, atau makanan. Harga dan jumlah stok memastikan informasi terkait biaya dan ketersediaan barang jelas bagi pembeli, sementara fitur unggah gambar memungkinkan seller menampilkan produk secara visual, yang penting untuk menarik minat pembeli.

Setelah data diinputkan, sistem melakukan validasi untuk memastikan bahwa informasi yang dimasukkan oleh penjual memenuhi kriteria tertentu. Misalnya, harga dan stok harus berupa angka positif, semua field wajib harus terisi, dan file gambar yang diunggah harus memiliki format yang didukung, seperti JPEG atau PNG. Validasi ini bertujuan untuk mencegah kesalahan input yang dapat menyebabkan masalah dalam pengelolaan data atau pengalaman pembeli.

Proses penyimpanan data dilakukan menggunakan microflow di Mendix. Microflow ini bertugas memeriksa data yang telah divalidasi, menyimpannya ke dalam *database* aplikasi, dan memberikan respons kepada penjual. Jika proses berhasil, sistem akan memberikan notifikasi, misalnya, "Produk berhasil ditambahkan." Sebaliknya, jika terjadi kesalahan, seperti data yang tidak lengkap atau format yang salah, sistem akan memberikan pesan error yang spesifik sehingga penjual dapat segera memperbaiki input mereka.

Produk yang berhasil ditambahkan oleh penjual akan langsung terintegrasi dengan halaman marketplace atau katalog aplikasi. Integrasi ini memungkinkan produk baru dapat dilihat oleh pembeli secara real-time tanpa perlu proses manual tambahan. Dengan demikian, penjual dapat memastikan produk mereka segera tersedia untuk transaksi setelah penambahan selesai.

Fitur ini tidak hanya meningkatkan efisiensi operasional penjual, tetapi juga memberikan fleksibilitas bagi mereka untuk mengelola produk secara mandiri kapan saja. Dengan desain yang user-friendly dan proses backend yang andal, fitur "Add Product" membantu menciptakan pengalaman pengguna yang optimal bagi penjual dan memastikan kualitas informasi yang diterima pembeli di marketplace tetap tinggi.



Gambar 3. 23 Tampilan Halaman Add Product

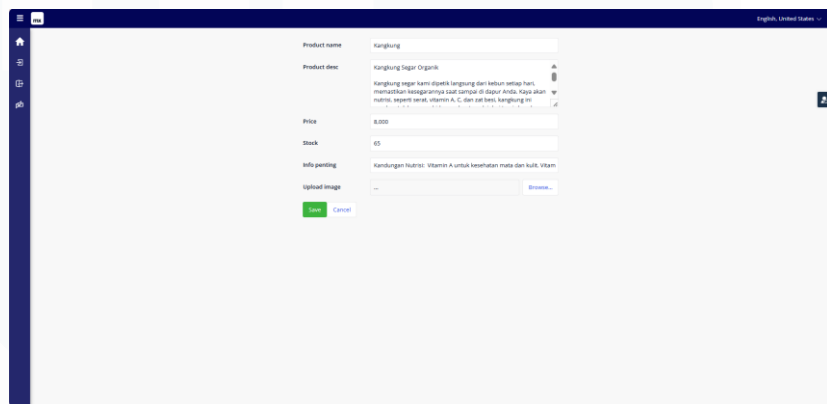
## b. Edit Product

Fitur "Edit Product" dalam aplikasi shopping app dirancang untuk memungkinkan penjual (seller) mengubah atau memperbarui informasi produk yang sudah ditambahkan ke katalog. Dengan fitur ini, penjual dapat dengan mudah memperbaiki detail produk jika terjadi kesalahan input atau menyesuaikan informasi yang relevan, seperti perubahan harga, stok, deskripsi, atau gambar produk. Pada halaman "Edit Product" yang ditunjukkan pada Gambar 3.24 sistem akan menampilkan formulir yang berisi informasi produk yang sudah ada, yang dapat langsung diedit oleh penjual sesuai kebutuhan dengan memilih produk yang ingin di ubah dan mengklik tombol edit.

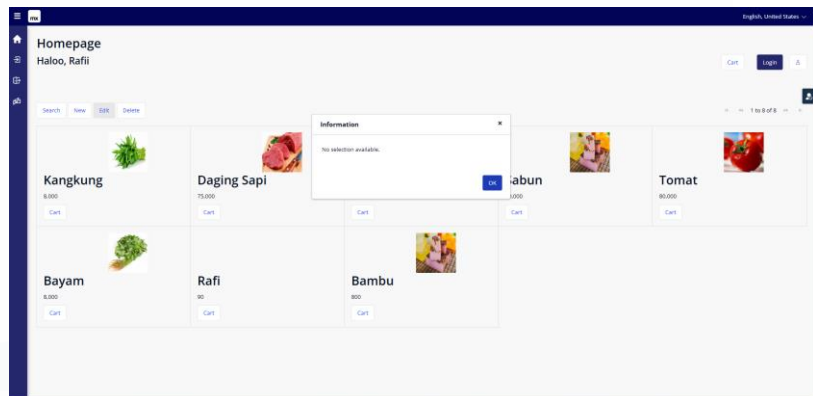
Prosesnya mencakup validasi data serupa dengan fitur "Add Product" untuk memastikan perubahan yang dilakukan sesuai dengan format yang diharapkan, seperti memastikan harga dan stok tetap berupa angka positif. Setelah perubahan disimpan, data produk di dalam *database* akan diperbarui menggunakan microflow di backend, yang menjamin bahwa informasi baru langsung tercermin di halaman marketplace.

Fitur ini juga memberikan notifikasi kepada penjual, baik ketika pembaruan berhasil dilakukan maupun jika terjadi kesalahan yang perlu diperbaiki, seperti field yang tidak diisi

dengan benar. Dengan integrasi real-time, pembeli akan segera melihat perubahan pada produk, seperti harga terbaru atau ketersediaan stok yang diperbarui. Fitur "Edit Product" membantu penjual menjaga informasi produk tetap akurat dan relevan, meningkatkan kepercayaan pembeli dan kualitas pengalaman pengguna dalam aplikasi. Jika user tidak memilih produk, system akan memunculkan *warning* bahwa belum ada produk yang dipilih. Hal ini ditampilkan pada Gambar 3.25.



Gambar 3. 24 Tampilan Halaman Edit Product



Gambar 3. 25 Pop Up Warning “No Selection Product”

## 11. Memperbaiki bug Admin Dashboard

Memperbaiki bug ini melibatkan pengecekan integrasi sistem backend dengan dashboard admin, memastikan data ditampilkan dengan benar dan sesuai, serta memeriksa tampilan dan



fungsionalitas antarmuka yang digunakan untuk mengelola aplikasi. Selain itu, sangat penting untuk memvalidasi peran dan izin akses admin untuk memastikan hanya pengguna yang berwenang yang dapat melakukan perubahan atau mengakses data sensitif.

## 12. *Sprint Payment Method*

### a. **Metode Pembayaran**

Fitur "Metode Pembayaran" dalam aplikasi shopping app memungkinkan seller untuk menambahkan dan mengatur metode pembayaran yang tersedia bagi pembeli. Fitur ini dirancang agar penjual dapat menawarkan berbagai opsi pembayaran yang sesuai dengan preferensi pembeli, seperti transfer bank, e-wallet, kartu kredit, atau metode lainnya, sehingga meningkatkan kenyamanan dan fleksibilitas dalam transaksi.

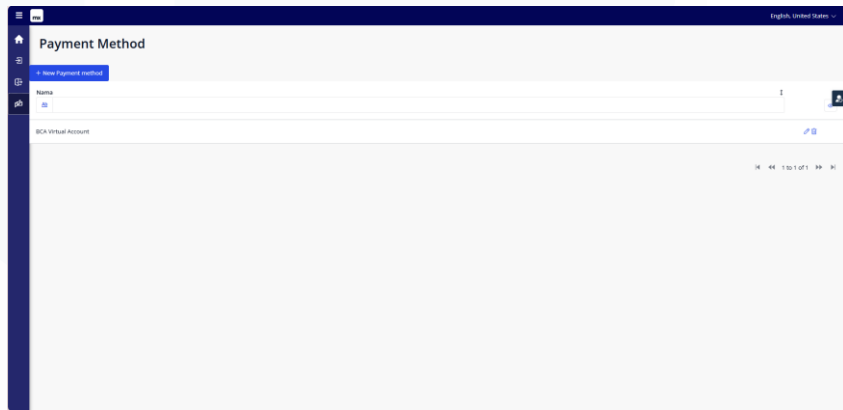
Pada halaman "Metode Pembayaran" dalam Gambar 3.26 penjual dapat mengakses formulir atau panel pengelolaan yang memungkinkan mereka menambahkan metode pembayaran baru. Setiap metode pembayaran yang ditambahkan akan mencakup informasi seperti nama metode (misalnya, "Transfer Bank BCA" atau "Gopay"), deskripsi singkat, dan, jika diperlukan, detail tambahan seperti nomor rekening atau ID e-wallet.

Data yang dimasukkan akan divalidasi untuk memastikan format yang benar, seperti memastikan nomor rekening hanya berisi angka atau ID e-wallet sesuai dengan standar *platform* yang digunakan. Setelah disimpan, metode pembayaran tersebut akan tersimpan dalam *database* dan langsung tersedia untuk pembeli saat mereka melakukan checkout.

Integrasi fitur ini memastikan bahwa setiap metode pembayaran yang ditambahkan oleh penjual akan muncul sebagai opsi yang dapat dipilih oleh pembeli pada halaman transaksi.

Selain itu, penjual juga dapat memperbarui atau menghapus metode pembayaran jika diperlukan, memberikan kontrol penuh atas pengelolaan opsi pembayaran.

Fitur ini memberikan keuntungan bagi penjual untuk memenuhi preferensi pembeli yang beragam, meningkatkan kemungkinan transaksi berhasil, dan menciptakan pengalaman belanja yang lebih user-friendly dan profesional. Dengan pengelolaan yang mudah dan fleksibel, fitur "Metode Pembayaran" membantu penjual dalam memberikan layanan yang optimal kepada pembeli.



Gambar 3. 26 Tampilan Halaman Add Payment Method

### 13. Memperbaiki bug Payment Method

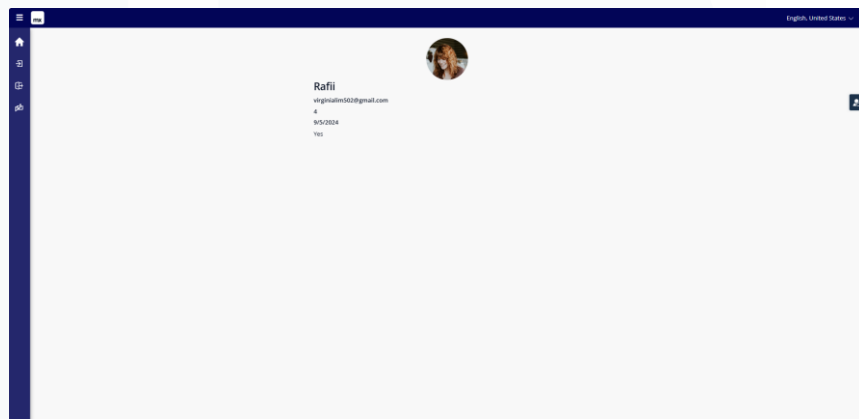
Langkah perbaikan ini harus diuji ulang melalui simulasi transaksi untuk memastikan metode pembayaran berfungsi dengan baik dan memberikan pengalaman pengguna yang optimal serta aman.

### 14. *Sprint Profile Dashboard*

#### a. Profile

Pada bagian *profile* dalam aplikasi *shopping app* yang dibuat menggunakan Mendix, setiap pengguna (user, admin, dan seller) dapat mengakses dan mengelola informasi pribadi mereka. Gambar 3.27 menunjukkan tampilan dari halaman *profile*. User dapat

melihat data pribadi seperti alamat, email. Pada aplikasi ini, user hanya dapat melihat tanpa ada opsi untuk melakukan perubahan. Namun, jika fitur pengeditan diperlukan, maka pengaturan di Mendix dapat disesuaikan dengan menambahkan alur untuk mengizinkan pengguna melakukan *update* pada informasi pribadi, tergantung pada kebutuhan aplikasi.



Gambar 3. 27 Tampilan Profile

### 15. Memperbaiki Profile Dashboard

Pada tahap ini terdapat beberapa yang di uji yaitu Uji fitur dasbor profil secara menyeluruh dengan berbagai skenario, pastikan semua data yang ditampilkan sesuai dengan informasi pengguna, periksa responsivitas desain di perangkat desktop dan mobile, simulasikan pengeditan data untuk memastikan perubahan tersimpan dengan benar, verifikasi keamanan data pengguna sesuai dengan standar. Dengan perbaikan ini, Profile Dashboard akan lebih stabil, fungsional, dan memberikan pengalaman pengguna yang lebih baik.

### 16. Menyiapkan aplikasi untuk demo

Tahap ini memastikan aplikasi siap untuk dipresentasikan kepada pemangku kepentingan. Langkah-langkahnya meliputi finalisasi fitur utama seperti autentikasi pengguna, katalog produk,

keranjang belanja, proses checkout, metode pembayaran, dan dasbor profil. Semua fitur diuji untuk memastikan berfungsi tanpa bug. Selain itu, aplikasi dioptimalkan untuk meningkatkan performa dan pengalaman pengguna. Data simulasi yang relevan juga disiapkan untuk mendukung skenario demo, seperti daftar produk, akun pengguna, dan transaksi uji coba.

### **17. Demo aplikasi kepada stakeholder**

Pada tahap ini, aplikasi yang telah selesai dikembangkan dan diuji dipresentasikan kepada para stakeholder. Demo dilakukan untuk menunjukkan bagaimana setiap fitur utama aplikasi berfungsi sesuai kebutuhan dan spesifikasi yang telah ditentukan. Selama sesi demo, dilakukan penjelasan mendetail mengenai alur kerja aplikasi, mulai dari proses autentikasi pengguna hingga transaksi akhir. Stakeholder juga diberikan kesempatan untuk mencoba aplikasi secara langsung dan memberikan masukan atau saran perbaikan jika diperlukan. Tahap ini bertujuan memastikan aplikasi memenuhi ekspektasi stakeholder sebelum dirilis ke lingkungan produksi.

### **18. Maintenance**

Tahap maintenance dilakukan setelah aplikasi dirilis untuk memastikan fungsionalitasnya tetap optimal dalam penggunaan jangka panjang. Kegiatan ini meliputi perbaikan bug yang teridentifikasi setelah peluncuran, peningkatan performa aplikasi, pembaruan fitur sesuai kebutuhan pengguna, serta penyesuaian terhadap perubahan teknologi atau regulasi. Selain itu, dilakukan monitoring secara rutin untuk mendeteksi dan mengatasi potensi masalah sebelum berdampak signifikan. Tujuan dari tahap ini adalah menjaga keandalan, keamanan, dan kepuasan pengguna aplikasi.

### 3.2.5 Membuat aplikasi *website* internal perusahaan “Reforms21”

"Reforms21" merupakan salah satu perusahaan yang bekerja sama dengan ICT. Sehingga dikembangkan aplikasi tiruan untuk internal yang akan digunakan untuk mempromosikan dan dijual. Aplikasi ini dibuat menggunakan Visual Studio Code sebagai editor kode dan Tailwind CSS untuk desain antarmuka pengguna (UI). Visual Studio Code digunakan untuk menulis kode HTML, CSS, dan JavaScript, serta untuk mengelola seluruh struktur proyek. Tailwind CSS, sebagai framework utility-first, memungkinkan pengembang untuk mendesain tampilan aplikasi dengan cepat dan efisien menggunakan kelas-kelas CSS siap pakai tanpa menulis banyak kode CSS custom.

Aplikasi ini dirancang untuk memenuhi kebutuhan fungsionalitas internal perusahaan, seperti menyediakan informasi, dashboard, atau fitur lainnya yang diperlukan oleh karyawan. Dengan menggunakan Tailwind, pengembang dapat fokus pada pembuatan antarmuka yang responsif, modern, dan mudah dipelihara, sementara Visual Studio Code mendukung proses pengembangan dengan berbagai ekstensi dan alat bantu. Proyek ini bertujuan untuk membuat *website* yang efektif, ramah pengguna, dan mudah disesuaikan dengan kebutuhan perusahaan.

#### 1. Menganalisis *design website* yang akan dibuat

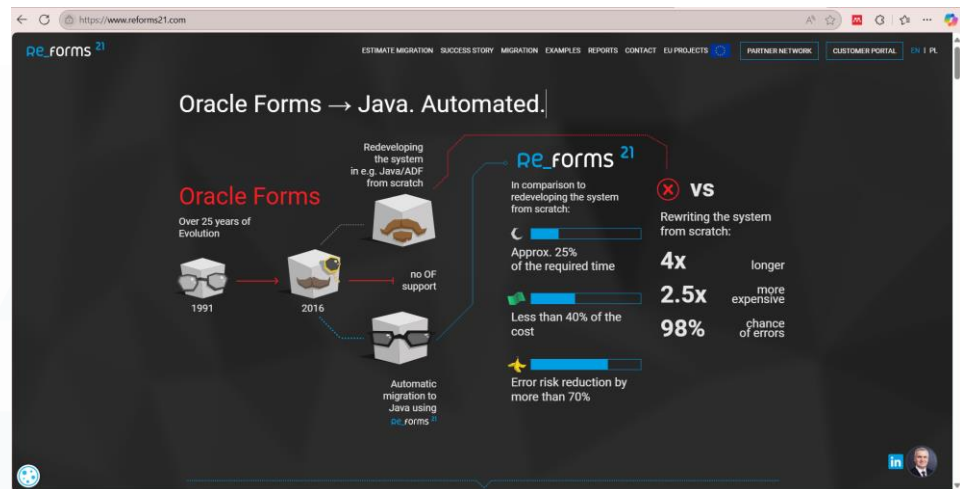
Peserta magang melakukan analisis terhadap elemen-elemen desain dan struktur fungsionalitas dari *website* yang ada, untuk memastikan bahwa *website* yang akan dibuat nantinya dapat meniru dan mempertahankan kualitas serta fitur-fitur yang ada pada *website* asli tersebut. Gambar 3.28 menunjukkan salah satu halaman pada *website* asli Reform21.

Langkah-langkah yang biasanya dilakukan pada tahap ini meliputi:

- a. Mempelajari Struktur Halaman: Mengidentifikasi bagaimana halaman-halaman *website* terorganisir, seperti bagaimana navigasi menu diatur, pengelompokan konten, dan pengaturan hierarki informasi.

- b. Memahami Desain Visual: Mengamati elemen desain seperti tata letak (layout), warna, font, ukuran teks, dan gaya visual lainnya yang digunakan pada *website* asli. Hal ini akan membantu dalam merencanakan penerapan desain serupa dalam pengembangan *website* yang akan dibuat.
- c. Mengidentifikasi Fitur Interaktif: Melihat fitur-fitur interaktif seperti formulir, tombol, slider, animasi, atau pop-up yang digunakan di *website* tersebut. Fitur-fitur ini harus ditiru agar aplikasi yang dikembangkan memiliki pengalaman pengguna yang serupa.
- d. Menilai Responsivitas: Memeriksa bagaimana *website* asli menyesuaikan tampilannya di berbagai perangkat (desktop, tablet, ponsel) dan memastikan bahwa tampilan serta fungsionalitasnya tetap optimal.
- e. Mengevaluasi Pengalaman Pengguna (UX): Mengamati bagaimana pengguna berinteraksi dengan *website* dan mengevaluasi elemen-elemen yang meningkatkan atau mengurangi kenyamanan penggunaan. Fokus di sini adalah pada alur navigasi dan kemudahan aksesibilitas informasi.
- f. Meninjau Konten: Memeriksa jenis dan format konten yang ditampilkan, apakah berupa teks, gambar, video, atau elemen lainnya, dan menentukan bagaimana konten tersebut dapat disajikan dalam *website* yang akan dibuat.

Dengan memahami secara mendalam struktur, desain, dan fitur *website* asli, pengembang dapat lebih mudah melakukan duplikasi dan penyesuaian desain pada *website* baru agar sesuai dengan model yang diinginkan, tanpa mengubah terlalu banyak elemen yang sudah ada.



Gambar 3. 28 Tampilan Asli website reform 21

## 2. Menentukan halaman yang akan di duplikasi

Pada tahap ini, pengembang perlu fokus pada halaman-halaman yang hanya memerlukan antarmuka pengguna (UI) tanpa melibatkan interaksi dengan server atau *database*. Halaman yang membutuhkan backend, seperti halaman login, halaman pemesanan, atau halaman yang memerlukan pengelolaan data dinamis, akan dikesampingkan untuk sementara karena membutuhkan implementasi backend yang tidak termasuk dalam lingkup proyek ini. Berikut merupakan Langkah Langkah yang digunakan untuk mencapai tahap ini :

- Identifikasi Halaman Statis:** Fokus pada halaman-halaman yang bersifat statis, yaitu halaman yang hanya menampilkan informasi tanpa memerlukan proses pengambilan data dari server atau *database*. Contohnya bisa mencakup halaman About Us, Contact Us, Product Information, Services, atau FAQ.
- Desain Visual dan Tata Letak:** Halaman yang hanya membutuhkan *frontend* akan berfokus pada desain visual dan tata letak. Pengembang akan menduplikasi elemen desain seperti header, footer, dan struktur grid halaman yang tidak melibatkan logika backend. Misalnya, halaman Home Page yang menampilkan

produk atau layanan dengan gambar dan teks tanpa interaksi dinamis.

- c. Halaman Interaktif Tanpa Backend: Pengembang juga dapat memilih halaman yang memiliki elemen interaktif, seperti sliders, carousel gambar, tombol-tombol interaktif, dan formulir statis (misalnya, formulir kontak yang tidak memerlukan pengiriman data ke server). Halaman-halaman ini tidak memerlukan backend tetapi tetap memberikan pengalaman pengguna yang dinamis.
- d. Penghindaran Halaman Dinamis: Halaman yang memerlukan interaksi dengan *database* atau backend, seperti halaman Login, Cart, Checkout, atau Dashboard Pengguna, akan dikesampingkan pada tahap ini, karena membutuhkan proses autentikasi atau pengolahan data yang tidak dapat dilakukan hanya dengan *frontend*.

Dengan menentukan halaman yang hanya melibatkan *frontend*, pengembang dapat fokus pada desain antarmuka dan elemen visual yang akan mempercantik tampilan *website* tanpa perlu khawatir tentang pengolahan data atau koneksi dengan backend.

### **3. Mengumpulkan sumber daya yang dibutuhkan (gambar, logo, dll)**

Tahap ini merupakan langkah penting sebelum memulai pengembangan *frontend* sebuah *website*. Pada tahap ini, pengembang akan mengidentifikasi dan mengumpulkan semua bahan atau aset yang diperlukan untuk mendukung desain dan fungsionalitas *website*, terutama untuk elemen-elemen visual yang akan digunakan dalam antarmuka pengguna. Berikut adalah langkah-langkah yang digunakan untuk mencapai tahap ini:

- a. Mengumpulkan Gambar dan Ikon  
Pengumpulan gambar adalah salah satu langkah utama. Ini termasuk gambar-gambar seperti foto produk, gambar latar belakang (background), ilustrasi, atau ikon untuk tombol dan



navigasi. Pastikan gambar memiliki kualitas tinggi dan sesuai dengan ukuran yang dibutuhkan untuk tampilan yang responsif dan optimal di berbagai perangkat. Pengumpulan gambar dapat mencakup:

- a) Logo perusahaan atau brand
- b) Gambar produk jika ada, untuk halaman produk atau layanan
- c) Ilustrasi atau ikon untuk meningkatkan antarmuka pengguna
- d) Gambar latar belakang atau banner yang digunakan pada halaman utama atau halaman lainnya

b. Mendapatkan Font dan Warna:

Pengumpulan font dan skema warna adalah aspek penting dari desain *website*. Font yang digunakan harus konsisten dengan identitas merek dan mudah dibaca. Skema warna harus disesuaikan dengan tema atau citra merek yang ingin disampaikan, sehingga memberikan pengalaman visual yang harmonis.

- a) Font: Pastikan memiliki lisensi atau izin untuk menggunakan font yang dipilih. Font bisa didapatkan dari sumber seperti Google Fonts.
- b) Skema warna: Tentukan palet warna utama dan sekunder untuk digunakan di seluruh *website*.

c. Mendapatkan Logo:

Logo perusahaan atau aplikasi merupakan elemen penting dalam identitas visual. Logo harus tersedia dalam berbagai format (seperti .png, .svg, atau .jpg) dengan resolusi tinggi untuk memastikan tampilannya tetap tajam di berbagai ukuran layar. Pastikan logo tersebut sesuai dengan brand guidelines (jika ada) dan konsisten dalam penggunaannya.

d. Mengumpulkan Video atau Animasi (Jika Ada):

Jika *website* memerlukan media dinamis seperti video atau animasi, pastikan untuk mengumpulkan file video yang sudah dipersiapkan dan dioptimalkan untuk *web*. Video ini bisa

digunakan di halaman beranda atau sebagai bagian dari interaksi pengguna.

e. Mengumpulkan Aset UI (User Interface):

Ini termasuk elemen-elemen seperti tombol, slider, formulir input, dan widget lain yang mendukung interaksi pengguna. Sumber daya ini bisa berupa gambar atau SVG, dan juga termasuk template UI dari framework seperti Tailwind CSS yang digunakan dalam proyek.

f. Memastikan Izin dan Lisensi Penggunaan:

Sebelum menggunakan gambar, font, logo, atau elemen lainnya, pastikan bahwa semua aset memiliki izin penggunaan yang tepat dan tidak melanggar hak cipta. Jika menggunakan gambar dari pihak ketiga, pastikan untuk memperoleh lisensi yang sah atau menggunakan sumber gambar bebas royalti (misalnya, Unsplash atau Pexels).

g. Organisasi dan Pengelompokan Sumber Daya:

Setelah mengumpulkan semua sumber daya, penting untuk mengorganisasikannya dengan baik. Pengelompokan sumber daya dalam folder sesuai jenisnya, seperti gambar, font, video, dan ikon, akan mempermudah pengelolaan dan aksesibilitas selama pengembangan *website*.

Mengumpulkan sumber daya ini dengan cermat akan memastikan bahwa pengembang memiliki semua elemen visual yang dibutuhkan untuk merancang dan membangun tampilan *website* sesuai dengan desain yang telah direncanakan, serta memberikan pengalaman pengguna yang konsisten dan menarik.

#### **4. Menyusun struktur navigasi**

Tahap ini merupakan proses merancang kerangka navigasi *website* yang akan memandu pengguna untuk menemukan informasi dan fitur dengan mudah. Struktur navigasi yang baik adalah fondasi penting

untuk memastikan pengalaman pengguna yang intuitif, efisien, dan menyenangkan. Pada tahap ini, pengembang menentukan cara halaman-halaman dalam *website* akan saling terhubung, serta bagaimana pengguna akan berinteraksi dengan elemen navigasi.

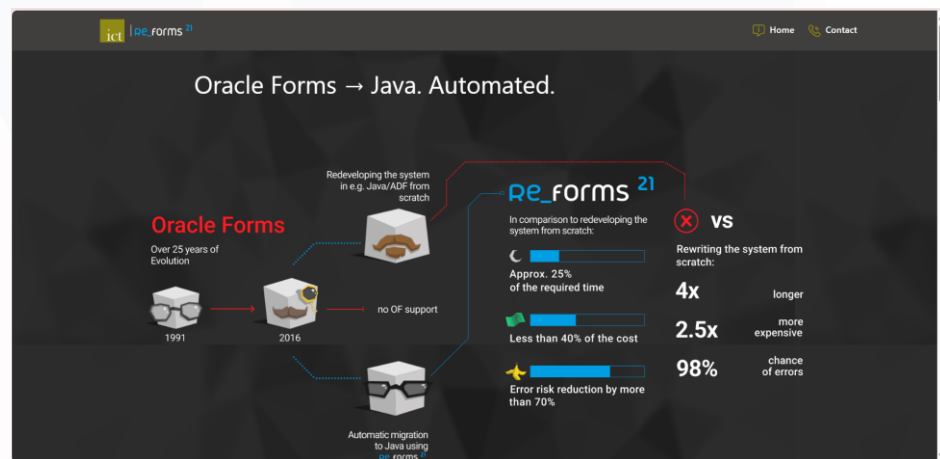
## **5. Menyiapkan project baru dengan Tailwind CSS**

Tahap ini Tailwind CSS menggunakan Vite adalah langkah awal dalam membangun aplikasi *frontend* yang efisien dan modern. Vite digunakan sebagai alat untuk membangun proyek karena kecepatannya dalam pengembangan, sementara Tailwind CSS memungkinkan pengembang untuk membuat desain antarmuka yang responsif dan konsisten dengan cepat menggunakan pendekatan *utility-first*. Vite dipilih karena kemampuannya untuk menyediakan lingkungan pengembangan yang cepat dan ringan. Vite dapat langsung di install pada terminal dan dapat langsung mengintegrasikan Tailwind CSS ke dalam project vite yang telah dibuat sebelumnya.

## **6. Membuat halaman utama dan layout**

Tahap ini merupakan langkah fundamental dalam pengembangan aplikasi *website* karena halaman utama adalah titik pertama interaksi pengguna dan mencerminkan inti dari tujuan *website*. Pada tahap ini, halaman utama dirancang dengan mempertimbangkan elemen-elemen seperti header, hero section, konten utama, dan footer. Header biasanya berisi navigasi utama, logo, dan elemen seperti search bar atau tombol login/register, sementara hero section digunakan untuk menarik perhatian pengguna dengan elemen visual seperti banner, teks besar, atau tombol *call-to-action*. Bagian konten utama menampilkan informasi atau fitur inti yang dapat dilihat pada Gambar 2.29. Footer melengkapi halaman dengan tautan tambahan, informasi kontak, dan akses ke media sosial.

Selanjutnya, layout dirancang untuk memberikan struktur yang konsisten di seluruh halaman *website*. Komponen seperti header dan footer menjadi elemen tetap yang memudahkan navigasi, sedangkan area konten utama dirancang fleksibel agar sesuai dengan kebutuhan halaman tertentu, seperti daftar produk, artikel, atau informasi pengguna. Tailwind CSS digunakan untuk mempercepat proses styling dengan menyediakan utility class yang memudahkan pengaturan grid, warna, margin, dan elemen visual lainnya. Dengan pendekatan ini, halaman utama dan layout tidak hanya dirancang untuk estetika tetapi juga untuk kenyamanan pengguna dalam menjelajahi *website*.

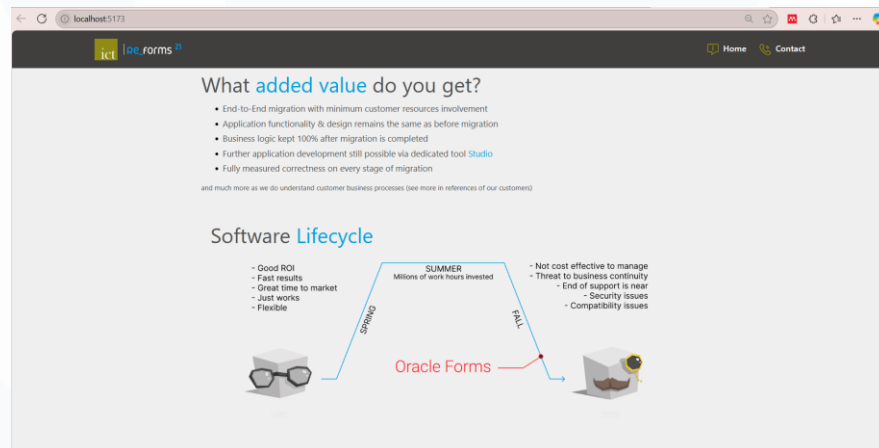


Gambar 3. 29 Tampilan utama *website*

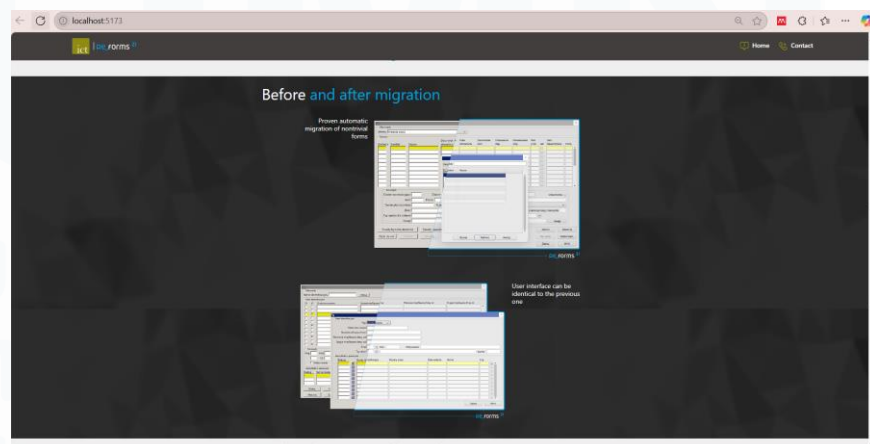
## 7. Membuat halaman informasi lainnya

Tahap ini merupakan proses melengkapi *website* dengan halaman-halaman tambahan yang berisi informasi penting bagi pengguna. Pada Gambar 3.30 dan Gambar 3.31 menunjukkan contoh halaman tambahan mengenai informasi mengenai *product*. Halaman ini biasanya mencakup informasi seperti *About Us*, *Contact Us*, *FAQ*, atau informasi lain yang relevan dengan tujuan *website*. Meskipun tidak menjadi fokus utama seperti halaman depan, halaman informasi ini berperan penting dalam memberikan transparansi, membangun kepercayaan pengguna, dan mendukung kebutuhan navigasi.

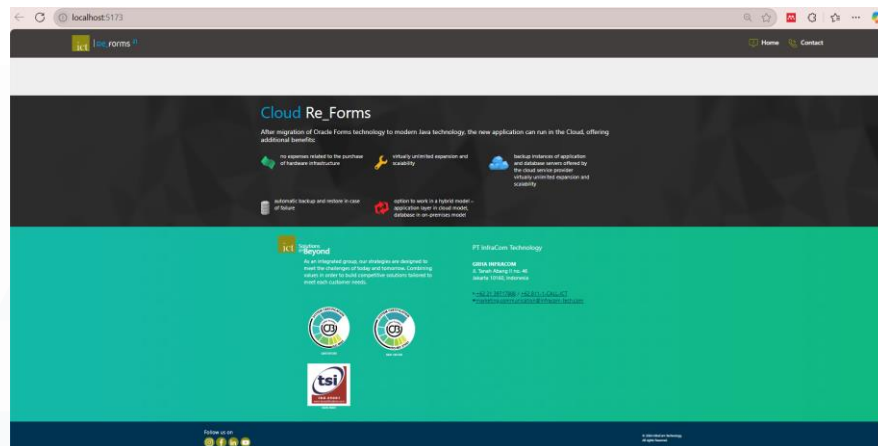
Pada tahap ini, setiap halaman dirancang dengan struktur yang sederhana dan informatif. Misalnya halaman *Contact Us* pada Gambar 3.32 menyediakan formulir kontak, alamat email, atau peta lokasi. Tailwind CSS digunakan untuk memastikan desain halaman tetap konsisten dengan gaya visual *website* secara keseluruhan. Dengan menggunakan kelas-kelas utility, elemen-elemen seperti heading, paragraf, tombol, atau form dapat diatur dengan cepat dan responsif. Selain itu, navigasi ke halaman informasi ini diintegrasikan ke dalam menu utama atau footer, sehingga pengguna dapat dengan mudah menemukannya. Halaman informasi ini dirancang agar ringan, mudah dipahami, dan berfungsi baik di berbagai perangkat, memastikan pengalaman pengguna yang optimal di seluruh bagian *website*.



Gambar 3. 30 Halaman informasi mengenai produk



Gambar 3. 31 Halaman informasi mengenai migrasi *database*



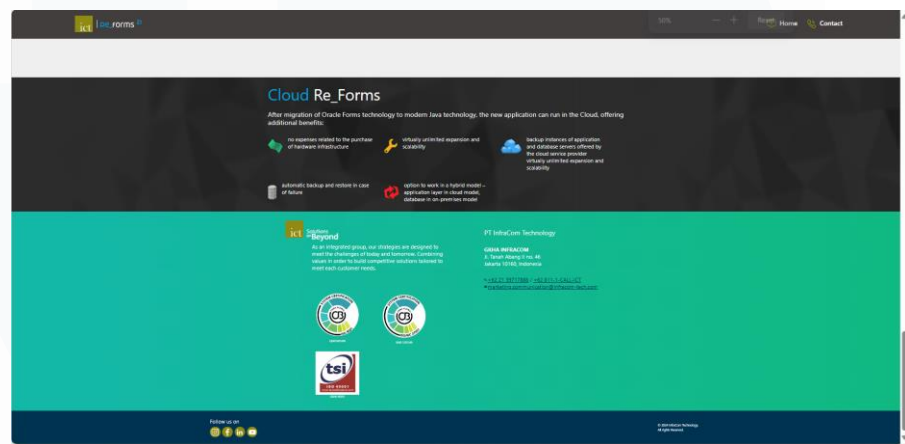
Gambar 3. 32 Halaman Get Contact

## 8. Penyempurnaan UI/UX

Tahap ini merupakan langkah akhir dalam pengembangan *website* yang bertujuan untuk meningkatkan kenyamanan visual dan interaksi pengguna. Pada tahap ini, desain antarmuka pengguna (*User Interface/UI*) diperhalus untuk memastikan elemen visual terlihat profesional, konsisten, dan menarik. Selain itu, pengalaman pengguna (*User Experience/UX*) dioptimalkan dengan memastikan navigasi yang mudah, responsivitas di berbagai perangkat, serta waktu pemuatan yang cepat. Proses penyempurnaan ini dimulai dengan melakukan evaluasi terhadap tampilan *website*, termasuk penggunaan warna, tipografi, ikon, dan elemen-elemen interaktif seperti tombol atau link. Perubahan kecil, seperti memperbaiki jarak antar elemen atau menambahkan animasi halus pada transisi, dapat memberikan dampak besar pada kesan visual. Tailwind CSS digunakan secara maksimal untuk menyesuaikan detail desain, seperti memastikan responsivitas melalui media query atau mengatur skala font agar sesuai dengan berbagai ukuran layar.

Dari sisi UX, pengujian dilakukan untuk memahami bagaimana pengguna berinteraksi dengan *website*. Feedback dari pengguna atau pengujian internal menjadi dasar untuk memperbaiki bagian yang

kurang intuitif.. Tahap ini juga mencakup pengoptimalan performa, seperti meminimalkan ukuran file gambar, memperbaiki struktur kode, atau menggunakan teknik *lazy loading* untuk konten berat. Dengan penyempurnaan UI/UX ini, *website* tidak hanya terlihat menarik secara visual tetapi juga memberikan pengalaman yang menyenangkan dan efisien bagi pengguna yang dapat dilihat pada Gambar 3.33.



Gambar 3. 33 Tampilan akhir *website*

## 9. Review dan demo dengan *stackholder*

Tahap ini merupakan langkah penting untuk memastikan bahwa *website* yang dikembangkan memenuhi harapan dan kebutuhan pengguna atau pemangku kepentingan utama. Pada tahap ini, hasil pekerjaan yang telah selesai ditinjau kembali bersama stakeholder untuk mendapatkan masukan, evaluasi, dan persetujuan sebelum *website* diluncurkan secara resmi. Proses ini dimulai dengan menyiapkan presentasi atau demo yang terstruktur. Tim pengembang biasanya menjelaskan fitur-fitur utama *website*, desain antarmuka, dan alur navigasi, serta bagaimana *website* tersebut memenuhi tujuan proyek. Stakeholder diajak untuk mencoba langsung *website* atau melihat simulasi dari berbagai halaman dan interaksi yang telah dirancang.

Selama sesi review, feedback dari stakeholder sangat penting. Mereka dapat memberikan saran perbaikan terkait desain, fitur, atau

pengalaman pengguna berdasarkan sudut pandang mereka. Tim pengembang harus siap mencatat setiap masukan dan menanggapi pertanyaan dengan jelas, termasuk menjelaskan keterbatasan teknis atau memberikan solusi alternatif jika ada permintaan yang belum dapat terpenuhi. Tahap ini juga menjadi kesempatan untuk menunjukkan komitmen terhadap kualitas. Sebelum demo, *website* telah melalui proses pengujian fungsionalitas dan responsivitas untuk memastikan bahwa tidak ada bug atau masalah besar yang mengganggu. Dengan melibatkan stakeholder secara aktif, tahap ini membantu menciptakan rasa kepemilikan bersama atas hasil proyek sekaligus memastikan bahwa semua pihak puas dengan *website* yang akan diluncurkan.

#### **10. Mengumpulkan *feedback* dan melakukan perbaikan dari saran**

Tahap ini merupakan langkah penting untuk menyempurnakan *website* berdasarkan masukan yang diperoleh dari stakeholder. Proses ini memastikan bahwa *website* tidak hanya memenuhi kebutuhan teknis tetapi juga ekspektasi pengguna atau pemangku kepentingan secara keseluruhan. Setelah sesi demo dan review, tim pengembang mengumpulkan semua feedback yang diberikan, baik terkait desain, fungsionalitas, maupun pengalaman pengguna. Feedback ini biasanya meliputi saran perbaikan, penambahan fitur, atau revisi pada elemen tertentu yang dianggap kurang optimal. Setiap masukan kemudian dianalisis dan dikategorikan berdasarkan prioritas, mulai dari masalah kritis yang harus segera diperbaiki hingga saran tambahan yang dapat diimplementasikan di kemudian hari.

Tim pengembang selanjutnya melakukan iterasi untuk mengakomodasi saran tersebut. Perbaikan dilakukan secara sistematis, dimulai dari bagian yang paling mendesak, seperti memperbaiki bug, menyempurnakan alur navigasi, atau mengubah elemen visual agar lebih sesuai dengan permintaan stakeholder. Proses ini juga mencakup



pengujian ulang setelah setiap revisi untuk memastikan bahwa perubahan yang dilakukan tidak menimbulkan masalah baru.

Setelah semua saran diimplementasikan, tim kembali mempresentasikan hasil revisi kepada stakeholder untuk validasi akhir. Tahap ini tidak hanya meningkatkan kualitas *website* secara keseluruhan tetapi juga menunjukkan komitmen terhadap kolaborasi dan kepuasan stakeholder. Dengan mengintegrasikan feedback secara efektif, *website* yang dihasilkan menjadi lebih matang, fungsional, dan sesuai dengan kebutuhan yang diharapkan.

#### **11. Melakukan *Build App* untuk di *deploy***

Tahap ini merupakan langkah terakhir dalam proses pengembangan *website* sebelum *website* tersebut dapat diakses oleh pengguna secara luas. Pada tahap ini, aplikasi yang sebelumnya dikembangkan dalam lingkungan pengembangan lokal disiapkan untuk diunggah ke server atau *platform* hosting yang akan digunakan.

Proses ini dimulai dengan menjalankan perintah *build* yang mengonversi kode menjadi versi produksi yang telah dioptimalkan. Tahap ini melibatkan penggabungan (*bundling*) semua file dan penghapusan elemen yang tidak diperlukan, seperti kelas CSS yang tidak digunakan, untuk memperkecil ukuran file dan mempercepat waktu pemuatan *website*. Proses *build* juga memastikan bahwa semua aset, seperti gambar atau font, dikompresi dan disiapkan dengan jalur akses yang benar untuk server produksi.

Setelah *build* selesai, tim melakukan uji coba terhadap hasil *build* tersebut untuk memastikan bahwa aplikasi berjalan dengan baik di lingkungan produksi. Pengujian ini mencakup pengecekan fungsionalitas, kompatibilitas pada berbagai browser, dan responsivitas di perangkat yang berbeda. Jika semua berjalan lancar, hasil *build* siap untuk diunggah ke server internal perusahaan.

Tahap ini memastikan bahwa aplikasi siap digunakan oleh pengguna dengan performa optimal dan keamanan yang memadai. Selain itu, proses *build* juga menjadi titik evaluasi terakhir untuk memastikan kualitas aplikasi sebelum benar-benar diluncurkan.

### 3.3 Kendala yang Ditemukan

Selama menjalankan proses magang, tentunya memiliki beberapa kendala yang dialami. Beberapa kendala-kendala yang dialami yaitu:

1. Saat mengerjakan task untuk membangun sebuah aplikasi mengalami kesulitan akibat fitur-fitur dari Mendix. Hal ini disebabkan oleh perbedaan versi antara yang digunakan saat ini dan yang digunakan saat belajar Mendix, sehingga menyebabkan sedikit hambatan dalam pengerjaan aplikasi yang dibuat.
2. Saat membuat microflow ataupun nanoflow untuk proses berjalannya aplikasi di Mendix terdapat kesulitan dalam pembuatan function ataupun rumus yang digunakan. Hal ini disebabkan oleh semua logic yang dibuat menggunakan microflow maupun nanoflow, sehingga menyebabkan kebingungan dalam pembuatannya.
3. Saat ingin membuat layout, terdapat kesulitan dalam menyusun layout dikarenakan fitur yang berbeda. Hal ini dikarenakan versi Mendix yang selalu diperbaharui sehingga harus melakukan research sendiri. Selain itu, terdapat kesulitan dalam mencari dokumentasi mengenai Mendix dengan versi yang paling baru.
4. Saat mengerjakan task yang bersifat teknis seperti coding, terdapat kesulitan saat mengalami banyak error yang terjadi ketika mengerjakan task tersebut. Hal ini dikarenakan peserta harus menggunakan logic berdasarkan task, library atau package yang telah update dari dokumentasi yang serupa, sehingga hal ini menyebabkan kebingungan dalam menyelesaikan error yang didapatkan.

5. Task yang diberikan terkadang tidak dikerjakan dengan teliti. Hal ini disebabkan oleh kurangnya pemahaman terhadap task yang diberikan, adanya satu poin yang terlewat, atau penjelasan yang kurang jelas.

### **3.4 Solusi atas Kendala yang Ditemukan**

Dari kendala yang ditemukan, tentunya membutuhkan solusi dalam menyelesaikannya. Mentor dan teman satu tim menjadi opsi pertama untuk melakukan diskusi ketika mengalami kesulitan. Terdapat solusi atas beberapa kendala yang telah dijelaskan sebelumnya, berikut solusi atas kendala yang telah ditemukan

1. Perbedaan versi Mendix yang digunakan dalam pembelajaran dan yang digunakan saat magang memang bisa menjadi tantangan tersendiri. Untuk mengatasi hal ini, penting untuk selalu memastikan bahwa dokumentasi terbaru dari Mendix dipahami dengan baik. Selain itu, berdiskusi dengan mentor atau rekan kerja mengenai fitur atau update baru pada versi yang digunakan dapat membantu mempercepat pemahaman dan meminimalisir kesalahan dalam implementasi. Membiasakan diri dengan *platform* dan melakukan eksperimen dengan versi yang sedang digunakan juga dapat mengurangi kebingungannya.
2. Memahami logika aplikasi yang perlu dibangun dalam bentuk microflow atau nanoflow memang membutuhkan waktu dan ketelitian. Hal ini sering kali terjadi karena keduanya mengharuskan logika aplikasi disusun dalam alur yang jelas dan terstruktur. Untuk mengatasi kendala ini, penting untuk terus berlatih dan memahami alur bisnis yang ingin dicapai. Selain itu, menggunakan tutorial dan contoh implementasi yang relevan atau meminta bantuan dari mentor atau tim teknis bisa mempercepat proses pembelajaran dan pemahaman.
3. Perubahan versi dalam Mendix yang terus berkembang sering kali membawa fitur baru atau perubahan pada cara menyusun layout. Untuk menghadapi kendala ini, sangat penting untuk selalu memperbarui pengetahuan tentang perubahan tersebut dengan merujuk pada dokumentasi terbaru yang diberikan oleh Mendix. Jika dokumentasi tidak mudah ditemukan, bergabung dengan

komunitas atau forum Mendix bisa menjadi alternatif untuk mencari solusi atau mendapatkan informasi yang diperlukan.

4. Menghadapi error dalam pengembangan aplikasi adalah hal yang wajar, apalagi ketika bekerja dengan pembaruan library atau package. Untuk mengatasi hal ini, penting untuk lebih sabar dan teliti dalam membaca pesan error yang muncul. Selain itu, dokumentasi yang lengkap dan pemahaman terhadap logika aplikasi yang sedang dikerjakan dapat membantu untuk mengidentifikasi masalah dan mencari solusi dengan lebih cepat. Jika error masih sulit dipecahkan, berdiskusi dengan tim atau mencari referensi dari sumber lain bisa memberikan solusi yang tepat.
5. Kurang teliti dalam mengerjakan task bisa terjadi karena kurangnya pemahaman terhadap detail task atau karena terlalu terburu-buru untuk menyelesaikannya. Untuk mengatasi masalah ini, penting untuk lebih berhati-hati dan mendalami setiap poin dalam task yang diberikan sebelum memulai pengerjaannya. Membaca dengan seksama dan memeriksa ulang pekerjaan setelah selesai juga dapat mengurangi kesalahan yang terlewat. Berkomunikasi dengan mentor atau rekan tim juga dapat membantu memastikan bahwa semua poin yang dibutuhkan telah tercakup dengan benar.

Dengan menghadapi dan mengatasi kendala-kendala ini, pengalaman berharga dapat diperoleh, dan kemampuan dalam pengembangan aplikasi menggunakan Mendix dapat ditingkatkan. Setiap tantangan yang dihadapi dianggap sebagai kesempatan untuk belajar dan berkembang dalam kemampuan teknis dan pemecahan masalah.