



# Analisis Perancangan Sistem Informasi

Tahun 2025

## Modul 3

Component Diagram & Deployment Diagram

Arranged By:  
EAD Laboratory Team

# Asisten Praktikum



**ARFY**  
Rifah Arfiyah



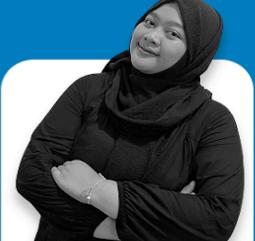
**JHON**  
I Gede Made Ari A.



**CIAA**  
Allicia Felicitas G.



**PALS**  
Naufal Akmal R.



**ICEA**  
Naisya Najmi



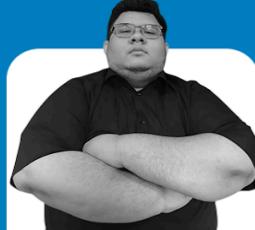
**RAYA**  
Muhammad Raya R.



**ASWW**  
Aswangga P.



**FRHN**  
Muhammad Ilyas



**ALEM**  
Matthew Alexander



**BGND**  
Stefanus Jesano P.



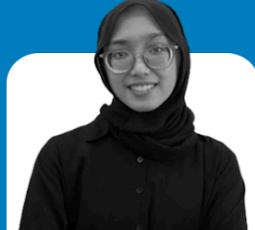
**YARE**  
Muhammad Ikhtiar



**NAVY**  
Neisy Nayyara A.



**AHRI**  
Balqis Eka N.



**MAUL**  
Maulida Afifi U.



**RARA**  
Annisa Agustina



**UCIK**  
Suci Larasati



**ZEAL**  
Azel Pandya M.



**LYAA**  
Yesitra Anugrah A.



**RRAS**  
Raras Nurhaliza H.



**NASA**  
Niken Ayu S.



**AARN**  
Aaron James E.



**ARCH**  
Ario Rafa M.

## PERATURAN PRAKTIKUM

1. **Setiap peserta praktikum** harus datang tepat waktu sesuai dengan jadwal.

**Toleransi keterlambatan hadir 15 menit.** Jika melebihi batas waktu tersebut diperkenankan mengikuti praktikum namun tidak mendapatkan tambahan waktu dan/atau nilai.

### 2. Perizinan Praktikum :

- A. Praktikan yang ingin melakukan perizinan diwajibkan untuk menghubungi asisten praktikum group plottingan, kemudian menghubungi komisi disiplin dan mengisi form perizinan.
- B. Form perizinan dapat diakses melalui LMS atau dikirimkan oleh komisi disiplin.
- C. Izin berkaitan dengan Sakit atau Kemalangan, maka praktikan dapat memberikan surat perizinan kepada pihak Komisi Disiplin maksimal 3 hari setelah jadwal (shift) praktikum.
- D. Izin berkaitan dengan Kematian dapat memberikan surat yang ditandatangani oleh keluarga terdekat.
- E. Izin lomba atau penugasan institusi tidak berlaku apabila tidak terdapat bukti dispensasi dari Igracias. NB: screenshot dispensasi dari igracias wajib dilampirkan dan dikirim melalui form perizinan yang ada di LMS atau didapat dari komisi disiplin.

### 3. Seragam Praktikum :

- A. Mahasiswa wajib menggunakan celana bahan hitam (bukan chino atau jeans) pada saat praktikum.
- B. Mahasiswi wajib menggunakan rok hitam/biru gelap panjang tidak ketat pada saat praktikum.
- C. Dresscode praktikum (Mengikuti Peraturan Telkom)
  - Senin: Mengenakan kemeja merah telkom atau kemeja putih polos.

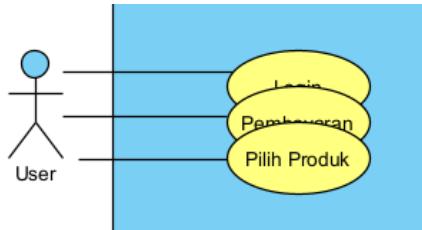
- Selasa s/d Rabu: Mengenakan kemeja putih Telkom atau kemeja putih polos.
  - Kamis s/d Sabtu: Mengenakan kemeja formal berkerah (bukan kerah sanghai dan bukan polo).
  - Jumat: Mengenakan kemeja/ baju batik bukan outer.
- D. Jika terdapat kendala dalam baju seragam maka praktikan diperbolehkan mengganti menggunakan kemeja putih Telkom atau kemeja putih polos.
- E. Praktikan dilarang untuk menggunakan aksesoris berlebih, seperti kalung, gelang, piercing, dan lain-lain. Kecuali aksesoris keagamaan.
- F. Membuka sepatu saat memasuki ruangan lab.
- G. Untuk pengecekan seragam bagi praktikan yang masih menggunakan topi atau jaket harap dilepas terlebih dahulu.

#### 4. Peraturan Pengerjaan :

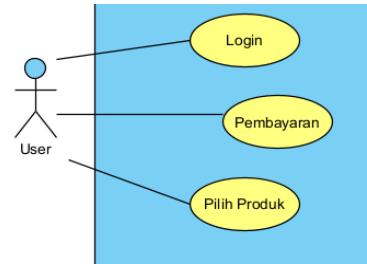
- A. Post test dilakukan secara individu dan dilarang membuka modul.
- B. Jika praktikan ketahuan membuka modul, searching di internet, menggunakan AI, dan alat komunikasi lainnya, nilai post test akan menjadi 0.
- C. Studi Kasus dikerjakan secara individu.
- D. Jawaban tidak boleh sama dengan setiap individu.
- E. Setiap diagram pada studi kasus wajib dibuat dengan memperhatikan kerapian agar mudah dipahami oleh asisten praktikum. Contoh diagram yang dianggap kurang rapi seperti notasi diagram yang bertumpuk-tumpuk, arah asosiasi yang tidak jelas, dan lain sebagainya.

## Penggambaran Diagram

Tidak Rapi



Rapi



- F. Penggambaran diagram yang kurang rapi akan mendapat pengurangan nilai maksimal 7 poin dan penggambaran diagram yang rapi akan mendapat penambahan nilai maksimal sebesar 5 poin.
- G. Pengerjaan jurnal dilarang searching di internet, menggunakan AI, bertanya kepada asisten praktikum dan berdiskusi dengan praktikan lainnya, dan alat komunikasi, melakukan hal tersebut akan dikenakan plagiarisme.
- H. Submit hasil pengerjaan berupa file format PDF
- I. Format Pengumpulan Hasil Pengerjaan:
- File penamaan Jurnal dalam bentuk PDF:  
**APSI\_MODULX\_KODEASISTEN\_NAMA LENGKAP\_NIM**  
Contoh:  
**APSI\_MODUL1\_JHON\_ARI ANANTA\_1021202200000**
  - File penamaan Tugas Pendahuluan dalam bentuk PDF:  
**APSI\_TP1\_KODEASISTEN\_NAMA LENGKAP\_NIM**  
Contoh:  
**APSI\_TP1\_JHON\_ARI ANANTA\_1021202200000**
- J. Screenshot hasil diagram tiap nomor dalam bentuk fullscreen dan menampilkan tanggal serta waktu. Jika tidak fullscreen maka akan dipotong 10% per nomor.

- K. Salah format nama pada file pengerajan nilai modul akan dipotong sebesar 10%
- L. Jika salah mengumpulkan file nilai akan dipotong sebesar 50%
- M. Terlambat mengumpulkan file pengerajan Jurnal dibawah 2 menit saat jurnal, maka nilai jurnal akan dipotong sebesar 0% . **(Jika terjadi gangguan bersama)**
- N. Terlambat mengumpulkan file pengerajan Jurnal, nilai jurnal akan dipotong sebesar 15% .
- O. Terlambat mengumpulkan file pengerajan Tugas Pendahuluan nilai Tugas Pendahuluan akan dipotong sebesar 35%.
- P. Tidak mengumpulkan Tugas Pendahuluan maka keseluruhan modul tersebut akan dipotong 70% .
- Q. Segala alat komunikasi harap dimatikan dan dimasukkan kedalam tas, dan tas disimpan dibawah meja.
- R. Jika ada perangkat praktikum yang bermasalah dapat menghubungi asprak yang bertugas.
- S. Segala bentuk kecurangan dan plagiarisme akan diproses ke komisi disiplin dan nilai akhir modul menjadi 0.

**DAFTAR ISI**

<b>PERATURAN PRAKTIKUM.....</b>	<b>4</b>
<b>LANDASAN TEORI.....</b>	<b>8</b>
A. Deployment Diagram.....	8
B. Component Diagram.....	12
<b>LANGKAH-LANGKAH PRAKTIKUM.....</b>	<b>21</b>
A. Cara Membuat Deployment Diagram.....	21
B. Cara Membuat Component Diagram.....	34
<b>CONTOH STUDI KASUS.....</b>	<b>44</b>
A. Studi Kasus 1 (Level Mudah).....	44
B. Studi Kasus 2 (Level Kompleks).....	50

## LANDASAN TEORI

### A. Deployment Diagram

#### 1) Definisi dan Fungsi

Deployment Diagram merupakan salah satu diagram dalam *Unified Modeling Language* (UML) yang digunakan untuk merepresentasikan hubungan antara komponen perangkat keras (hardware components) dalam infrastruktur fisik suatu sistem informasi. Diagram ini juga menunjukkan bagaimana perangkat lunak (software components) diinstal dan dijalankan dalam sistem fisik tersebut.

Deployment Diagram berguna untuk memvisualisasikan arsitektur fisik dari suatu sistem, termasuk bagaimana node dan komponen berinteraksi melalui jalur komunikasi tertentu. Diagram ini sering digunakan dalam tahap perancangan dan dokumentasi sistem untuk memahami struktur implementasi perangkat keras dan perangkat lunak. Deployment Diagram terdiri dari beberapa elemen utama dalam UML, diantaranya:

Nama	Deskripsi
Node	Node direpresentasikan sebagai kotak tiga dimensi dalam diagram dan dapat berupa <i>hardware</i> (server, komputer, router, dll.) maupun <i>software</i> (runtime environment, virtual machine, dll.).
<i>Artifact</i>	<i>Artifact</i> merupakan bagian dari sistem yang ditempatkan dalam node, biasanya mewakili komponen perangkat lunak, seperti file kode sumber, database, library, atau aplikasi yang dijalankan pada perangkat keras tertentu.
<i>Communication Path</i> (Jalur Komunikasi)	<i>Communication Path</i> menggambarkan hubungan antara

	node dalam arsitektur fisik. Hubungan ini dapat berupa koneksi LAN, Internet, serial, parallel, atau protokol lainnya yang digunakan dalam komunikasi antar-komponen.
--	---

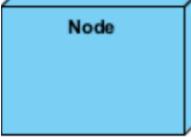
### Fungsi **Deployment Diagram**

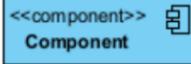
*Deployment Diagram* memiliki beberapa kegunaan utama, yaitu:

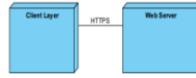
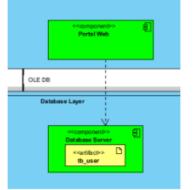
- Menunjukkan struktur sistem pada saat runtime
- Memberikan gambaran bagaimana hubungan antar-perangkat keras dalam sistem
- Menunjukkan lokasi instalasi perangkat keras dan perangkat lunak
- Membantu dalam analisis kinerja dan skala sistem

### 1) Notasi **Deployment Diagram**

Berikut merupakan notasi *Deployment Diagram*, antara lain :

Nama	Notasi	Deskripsi	Contoh	Keterangan
Node		Entitas fisik yang mengeksekusi satu atau lebih komponen, subsistem atau executable. Node dapat berupa perangkat keras, server, workstation		Pada contoh di samping, Node diberi nama Client Browser yaitu tempat pengguna bisa mengakses Website. Contoh lainnya

		dimana komponen perangkat lunak dapat digunakan atau dijalankan.		dapat berupa Application Server, Web Server, dll.
Artifact		Artifact adalah elemen model yang mewakili entitas fisik dalam sistem perangkat lunak. Artefak mewakili Unit implementasi fisik, seperti file yang dapat dieksekusi, Komponen perangkat lunak, dokumen, dan database.		Pada contoh di samping, Artifact berupa file.xml, ini berarti artifact mewakili bentuk file yang dapat Dieksekusi. Contoh lainnya dapat berupa file database, dokumen, dll
Component		Sebuah notasi yang mewakili bagian-bagian utama dari sistem perangkat lunak ataupun		Pada contoh di samping component berupa Database Server yang digunakan untuk

		<p>perangkat keras. Dapat berupa aplikasi, database, server, atau perangkat lainnya.</p>		<p>menyimpan, mengelola, dan mengakses data dalam basis data. Contoh lainnya dapat berupa Portal Web, Web Browser, dll.</p>
Association	_____	<p>Mengindikasikan bahwa instansi dari satu elemen model terhubung ke instansi unsur model lain.</p>		<p>Pada contoh disamping Node Client Layer berhubungan menggunakan protokol komunikasi HTTPS dengan Node Web Server.</p>
Dependency	----->	<p>Mengindikasikan bahwa sebuah komponen menggunakan / bergantung pada komponen lain untuk menjalankan</p>		<p>Pada contoh disamping Component Portal Web Ketergantungan kepada Component Database Server yang terletak di</p>

		fungsinya.		Node Database Layer, Ketergantungan ini menunjukkan bahwa portal web tidak memiliki kontrol langsung atas data. Portal Web berfungsi sebagai interface. Yang memungkinkan pengguna untuk berinteraksi dengan data, namun data itu sendiri disimpan dan dikelola oleh Database Server.
--	--	------------	--	---

## B. Component Diagram

### 1) Definisi dan Fungsi

Component Diagram adalah diagram yang digunakan untuk menunjukkan bagaimana komponen-komponen dalam sebuah sistem perangkat lunak disusun dan berinteraksi satu sama lain. Diagram ini memberikan gambaran visual tentang bagaimana berbagai bagian sistem saling terhubung dan bekerja bersama. Setiap komponen dalam diagram ini bisa berupa *class*, *package*, atau bahkan bagian yang lebih kecil, tergantung pada kompleksitas sistem yang dianalisis. Masing-masing komponen memiliki tugas atau fungsi tertentu dan akan berinteraksi dengan komponen lain sesuai kebutuhannya.

Component Diagram memberikan pemahaman bagaimana suatu sistem bekerja serta membantu dalam proses analisis dan perancangan sistem. Dalam UML versi 2, komponen biasanya digambarkan dalam bentuk persegi panjang dengan nama yang dapat diubah sesuai kebutuhan. Hubungan antar komponen ditunjukkan dengan garis yang merepresentasikan hubungan (*association*) atau ketergantungan (*dependency*). Selain itu, Component Diagram juga menggunakan *stereotype*, yaitu label tambahan yang memberikan informasi lebih lanjut tentang peran atau sifat suatu komponen. *Stereotype* ini memungkinkan kita untuk memahami konteks dan fungsi spesifik dari elemen dalam diagram dengan lebih jelas. Di bawah ini adalah beberapa stereotip yang sering ditemui:

Nama	Deskripsi
«interface»	Merupakan kontrak atau kesepakatan yang mendefinisikan bagaimana dua bagian dari program dapat berkomunikasi satu sama lain. Bayangkan seperti aturan main dalam sebuah permainan; semua pemain harus mengikuti aturan tersebut agar permainan berjalan lancar.
«controller»	Merupakan bagian dari program yang mengatur alur kerja. <i>Controller</i> menerima input dari pengguna, memprosesnya, dan kemudian mengarahkan ke bagian lain dari program untuk mendapatkan hasil. Seperti seorang sutradara film yang mengarahkan aktor dan menentukan bagaimana cerita berjalan.
«service»	Merupakan bagian yang melakukan tugas tertentu dalam program. <i>Service</i> biasanya berisi logika bisnis dan menjalankan fungsi-fungsi yang dibutuhkan. Bayangkan seperti seorang koki yang menyiapkan makanan sesuai pesanan.
«repository»	Merupakan tempat penyimpanan data. <i>Repository</i> bertanggung jawab untuk mengambil, menyimpan, dan mengelola data dari sumber tertentu, seperti database. Seperti lemari arsip yang menyimpan dokumen penting.
«boundary»	Merupakan batasan atau area yang memisahkan satu bagian dari program dengan bagian lainnya. <i>Boundary</i> membantu menjaga agar setiap bagian tetap terorganisir dan tidak saling mengganggu. Seperti pagar yang memisahkan dua kebun agar tanaman tidak bercampur.
«cache»	Merupakan tempat penyimpanan sementara untuk data yang sering diakses. <i>Cache</i> membantu mempercepat proses karena data yang sudah

	disimpan di sini bisa diambil lebih cepat daripada harus mencarinya dari sumber aslinya. Seperti menyimpan makanan di kulkas agar bisa diambil dengan mudah saat dibutuhkan.
<<provider>>	Merupakan komponen yang menyediakan layanan atau sumber daya tertentu untuk bagian lain dari program. <i>Provider</i> bisa dianggap sebagai penyedia barang atau jasa yang siap membantu ketika dibutuhkan. Seperti toko yang menyediakan bahan makanan untuk memasak.

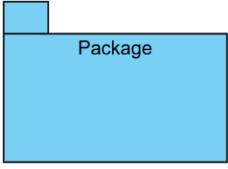
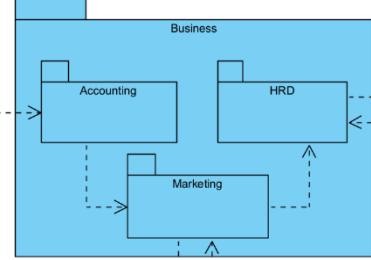
## Fungsi Component Diagram

Component Diagram memiliki fungsi sebagai berikut:

- Menunjukkan hubungan antar komponen
- Membantu memahami fungsi masing-masing komponen
- Mempermudah pengembangan dan perawatan
- Memvisualisasikan organisasi komponen dalam sistem
- Memodelkan struktur sistem sebelum dibangun
- Mengkomunikasikan fungsi-fungsi sistem yang sedang dibuat

### 2) Notasi Component Diagram

Berikut merupakan notasi Component Diagram, antara lain :

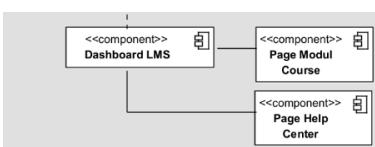
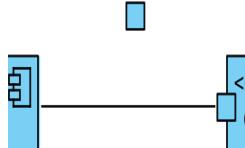
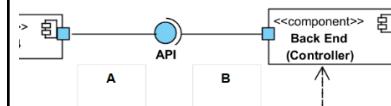
Nama	Notasi	Deskripsi	Contoh	Keterangan
Package		Bayangkan paket sebagai kotak yang berisi beberapa barang. Dalam konteks class		Gambar di samping adalah contoh paket dari sebuah website, yang di dalamnya

		<p>diagram, paket adalah cara untuk mengelompokkan kelas-kelas yang memiliki hubungan atau tema yang sama. Misalnya, jika kita memiliki beberapa kelas yang berkaitan dengan "Mobil", kita bisa mengelompokkannya dalam satu paket bernama "Kendaraan".</p>		<p>terdapat tiga paket, yaitu paket akuntansi, HR, dan marketing, yang saling terkait.</p>
Component		<p>Bagian dari sistem yang memiliki fungsi tertentu dan bisa berdiri sendiri, seperti modul login dalam aplikasi.</p>		<p>Gambar di samping adalah contoh komponen yang disebut Database, yang berfungsi untuk menyimpan Database EAD.</p>
Interface		<p>Interface digambarkan sebagai lingkaran kecil yang memiliki nama di</p>		<p>Gambar di samping adalah contoh antarmuka di mana komponen A (komponen di</p>

	<p>bawahnya. Fungsi dari interface ini adalah sebagai antarmuka yang memungkinkan komponen lain untuk berkomunikasi dalam sistem. Komunikasi ini ditandai dengan adanya operasi yang diperlukan (<i>required</i>) dan operasi yang disediakan (<i>provided</i>) oleh suatu komponen.</p> <p><b>Provide Interface</b></p> <p>Ini adalah interface yang menunjukkan layanan atau fungsi yang disediakan oleh suatu komponen. Misalnya, jika</p>	sebelah kanan) memerlukan komponen Database untuk menyimpan data yang tersedia. Oleh karena itu, Database akan menyediakan DBSystem untuk komponen A agar kedua komponen dapat berkomunikasi.
--	---	---

	<p>kita memiliki interface bernama '<b>PaymentService</b>', interface ini mungkin memiliki metode seperti '<b>processPayment()</b>' dan '<b>refund()</b>'. Kelas yang mengimplementasikan <b>'PaymentService'</b> harus menyediakan detail tentang bagaimana metode tersebut dijalankan.</p> <p><b>Required Interface</b></p> <p>Ini adalah interface yang menunjukkan layanan atau fungsi yang dibutuhkan oleh suatu komponen. Misalnya, jika kita memiliki</p>	
--	--	--

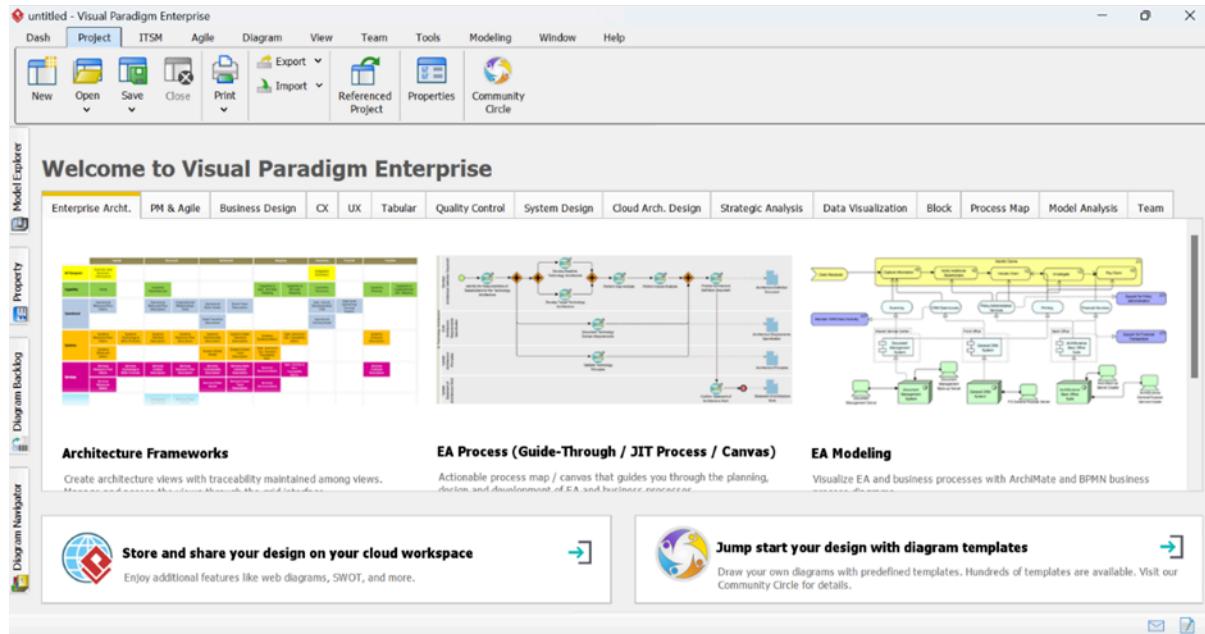
		<p>kelas 'OrderProcessor' yang memerlukan layanan pembayaran, kita bisa mendefinisikan interface 'PaymentService' sebagai required interface. Kelas ini akan memanggil metode dari interface tersebut untuk melakukan pembayaran.</p>		
Dependency	<-----	<p>Menunjukkan bahwa satu kelas bergantung pada kelas lain. Jika kelas yang menjadi sumber ketergantungan diubah, kelas yang bergantung juga mungkin perlu diubah.</p>	<pre>graph TD; subgraph Comp1 [ ]; direction TB; A[&lt;&lt;component&gt;&gt; Page Login]; end; subgraph Comp2 [ ]; direction TB; B[&lt;&lt;component&gt;&gt; Dashboard Pendaftar]; end; A --&gt; B;</pre>	<p>Gambar di samping merupakan contoh ketergantungan di mana Dashboard Pendaftar bergantung pada Halaman Log In, artinya pengguna harus mengakses</p>

				Halaman Log In terlebih dahulu sebelum dapat masuk ke Dashboard Pendaftar.
Association		Hubungan antara dua kelas yang menunjukkan interaksi, seperti "Mahasiswa" yang mendaftar untuk "Kursus".		Gambar di samping adalah contoh asosiasi di mana Dashboard LMS berhubungan dengan Halaman Modul Course dan Halaman Help Center.
Port		Titik masuk atau keluar untuk komunikasi antara komponen, seperti pintu yang menghubungkan bagian dalam dan luar rumah.		Gambar di samping adalah port yang menghubungkan komponen A ke komponen B dengan menggunakan antarmuka API agar kedua komponen tersebut dapat berkomunikasi.

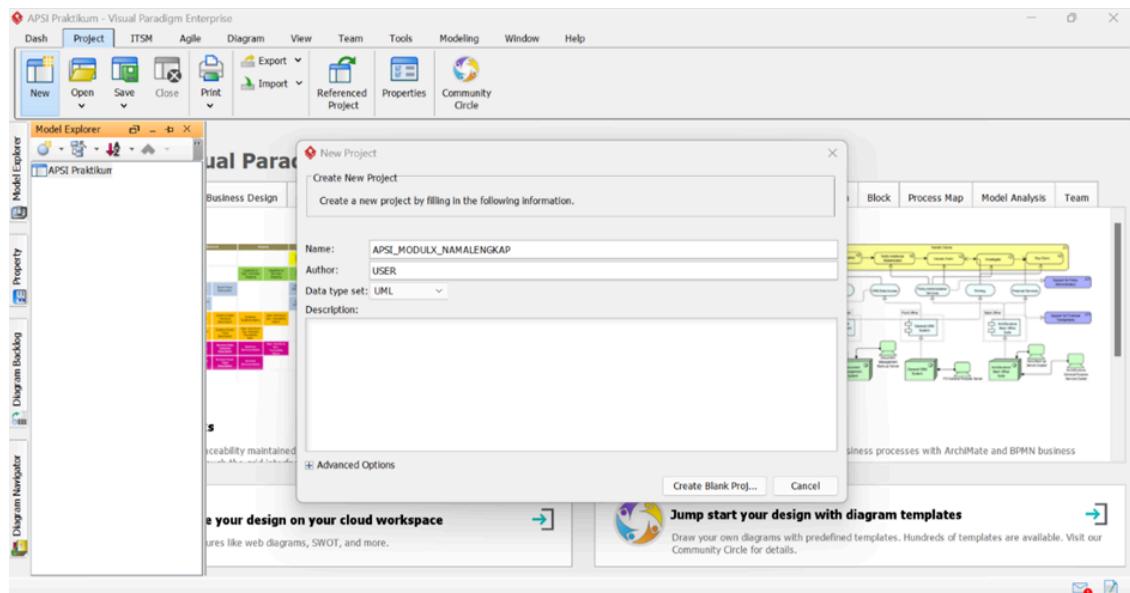
## LANGKAH-LANGKAH PRAKTIKUM

### A. Cara Membuat Deployment Diagram

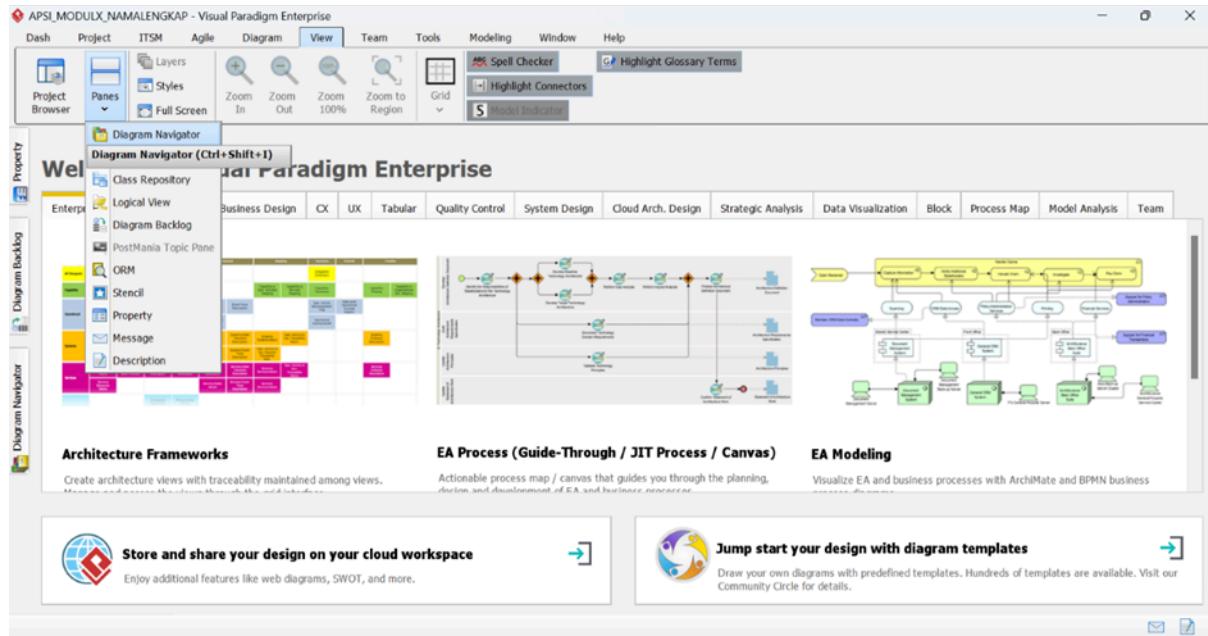
- I. Pada tampilan Dashboard Visual Paradigm, klik **Project** lalu klik **New**.



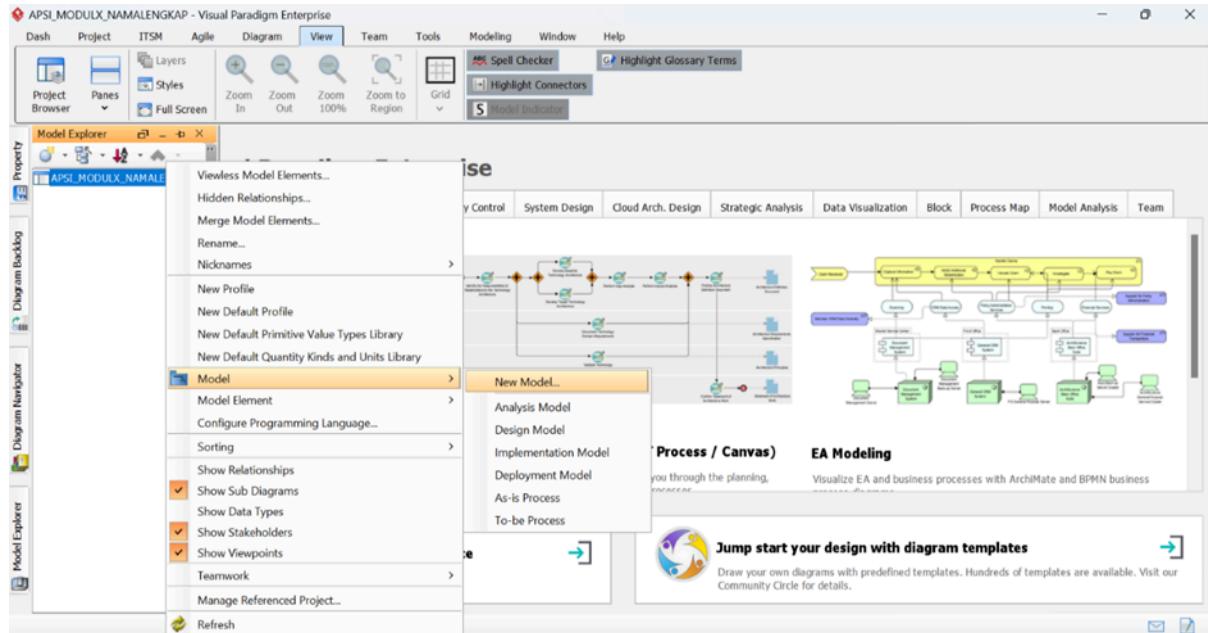
- II. Kemudian klik "**Create Blank Project**".



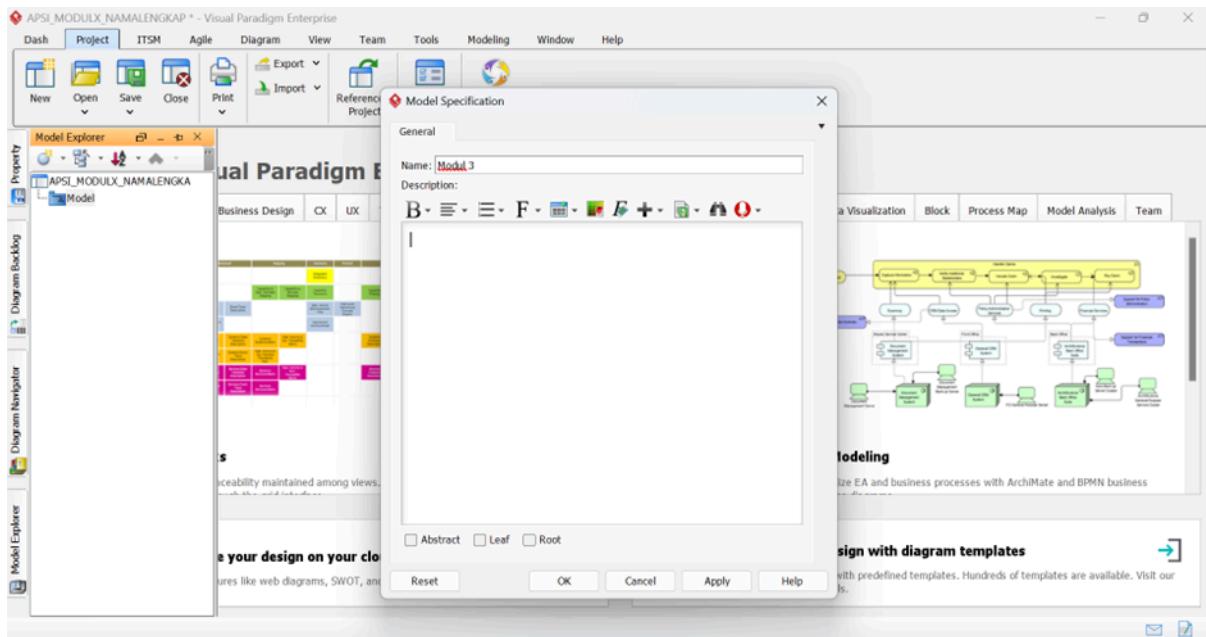
- III. Buka “**Model Explorer**” pada Tab “**View**” > “**Panes**”, atau bisa menggunakan shortcut CTRL + SHIFT + O .



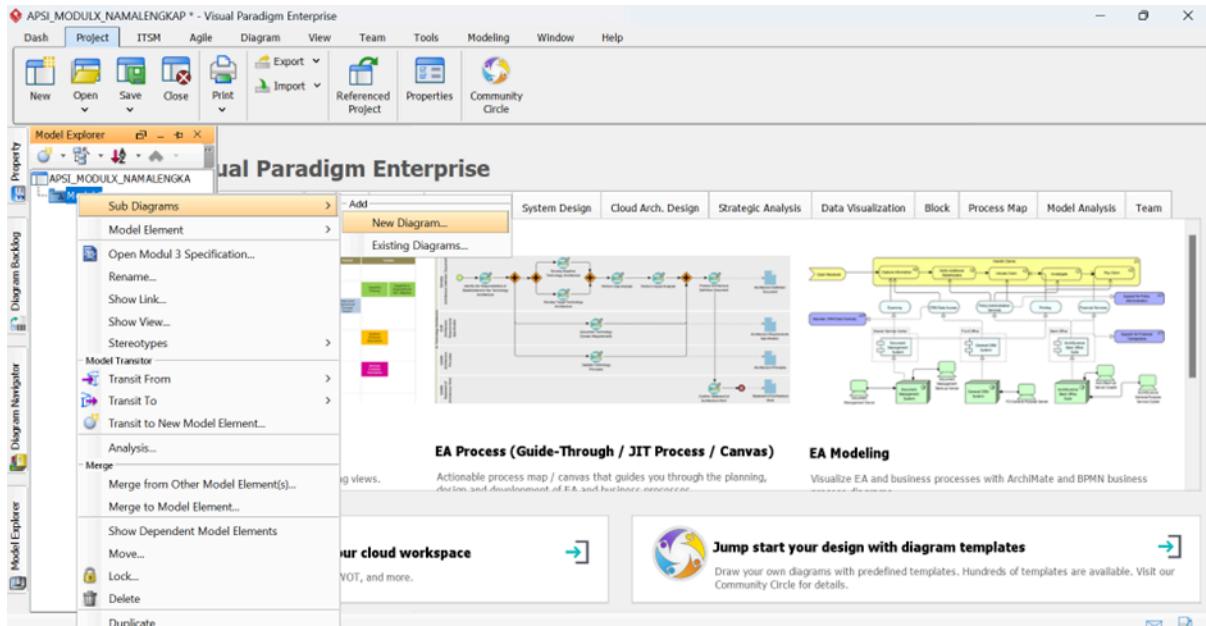
- IV. Klik kanan pada Project tersebut > klik **Model** > klik **New Model**.



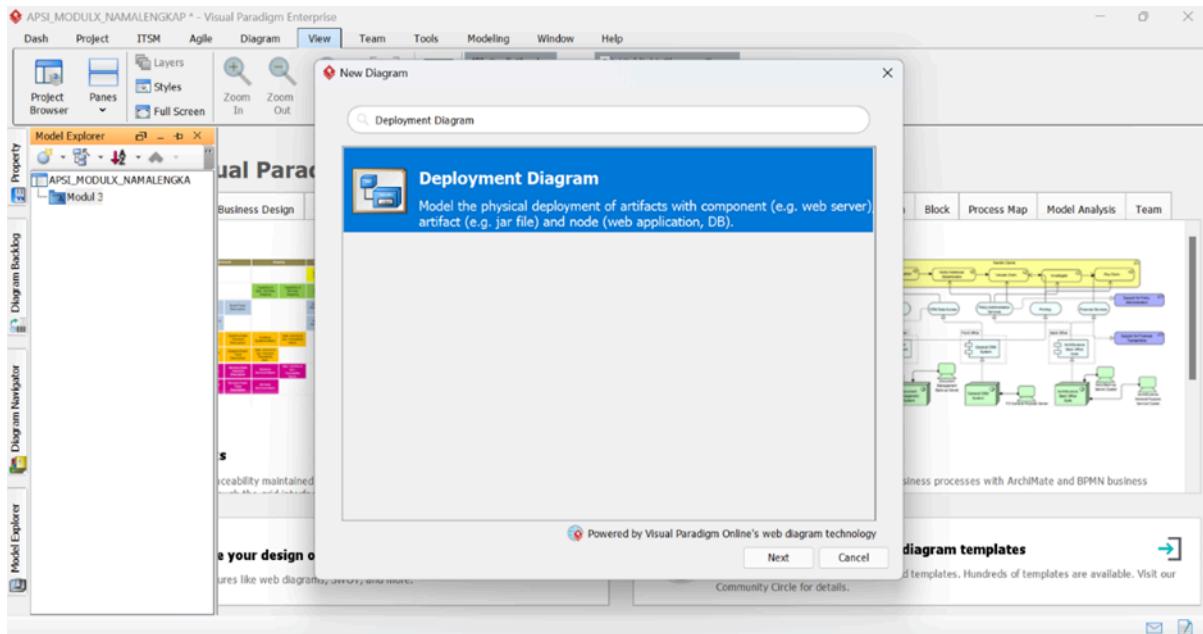
V. Beri nama model tersebut sesuai dengan modul yang dikerjakan.



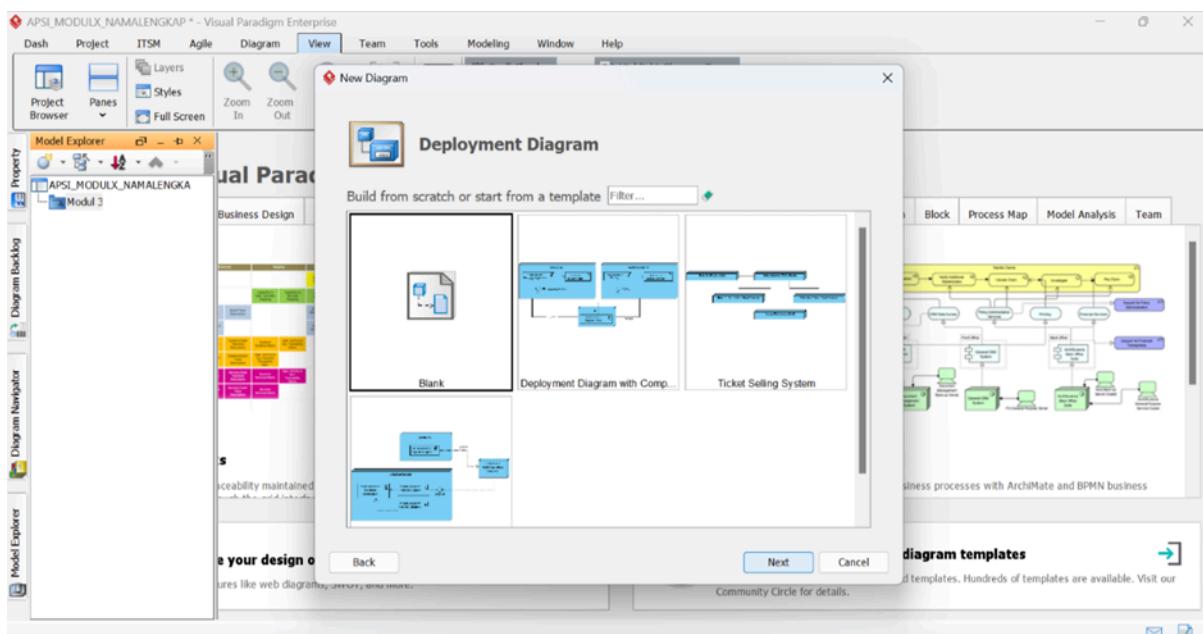
VI. Klik kanan pada model yang telah dibuat > klik **Sub Diagram** > **New Diagram**.



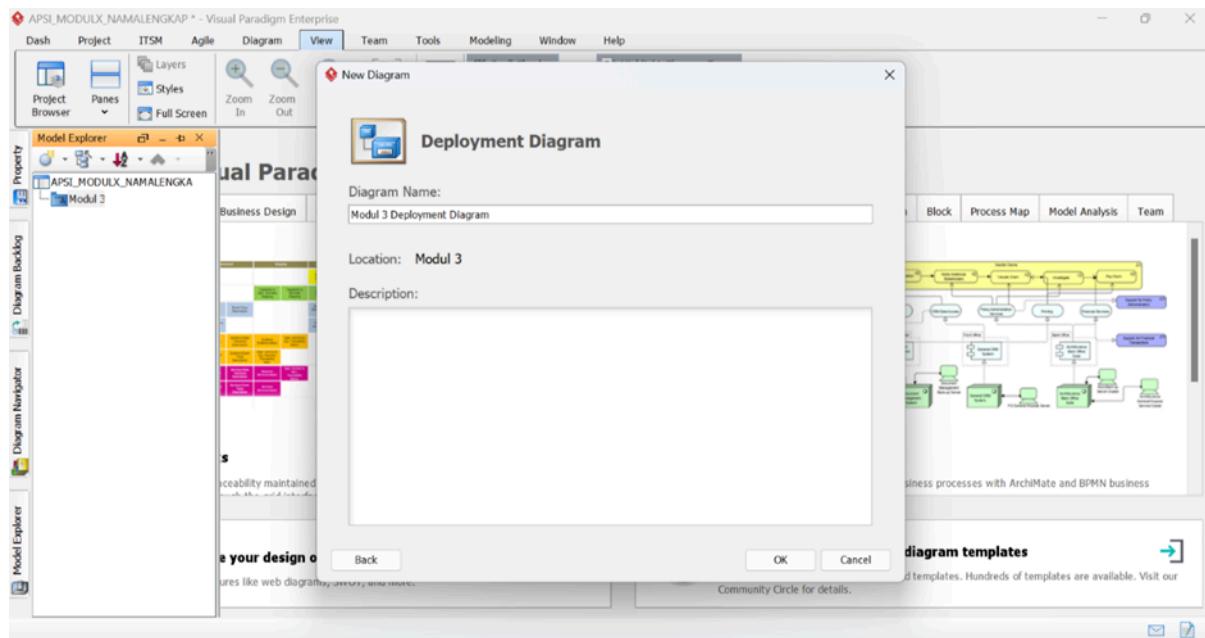
VII. Cari “**Deployment Diagram**” di pencarian, lalu klik **Next**.



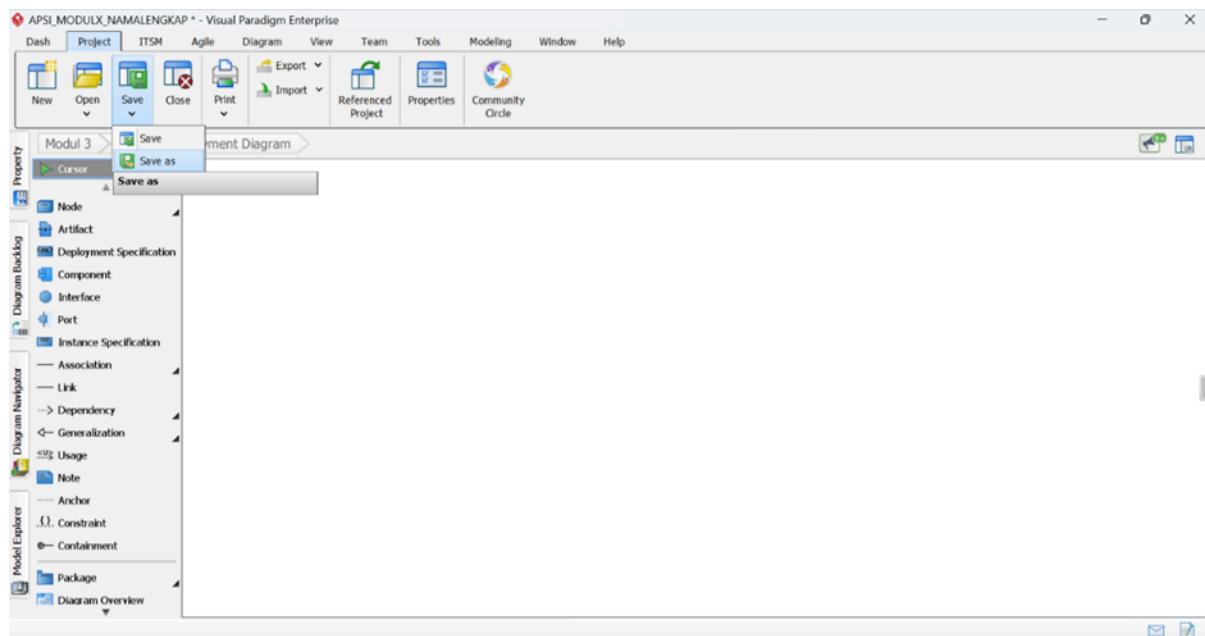
VIII. Klik “**Blank**” lalu klik **Next**.



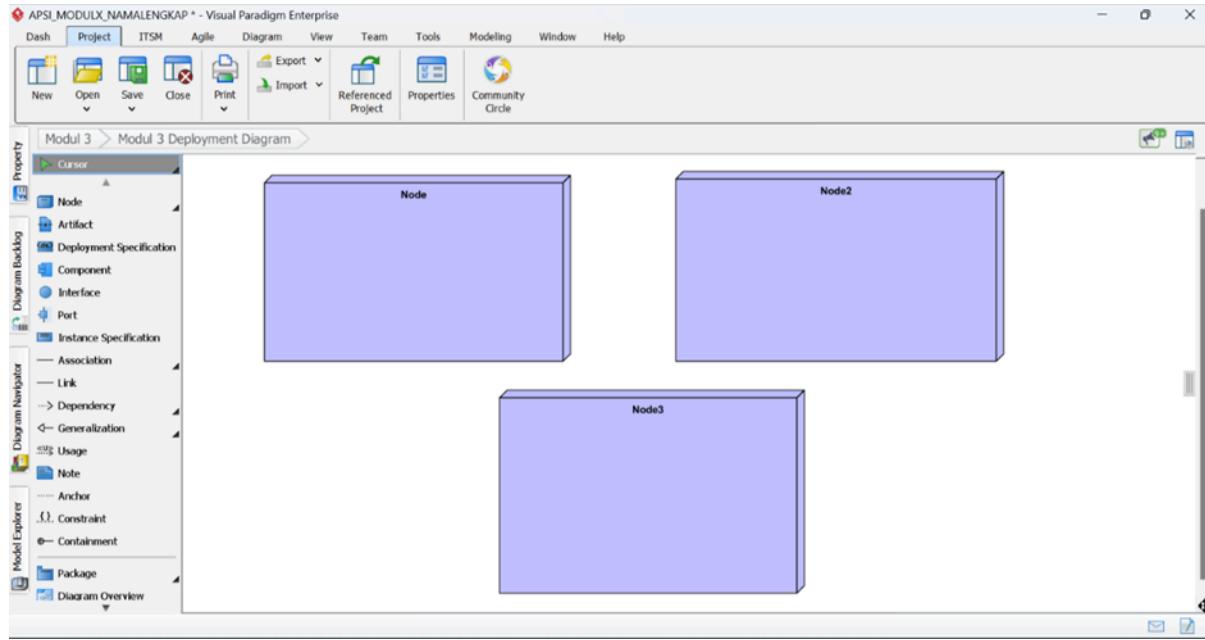
- IX. Berikan nama sesuai modul dan diagram yang dikerjakan.



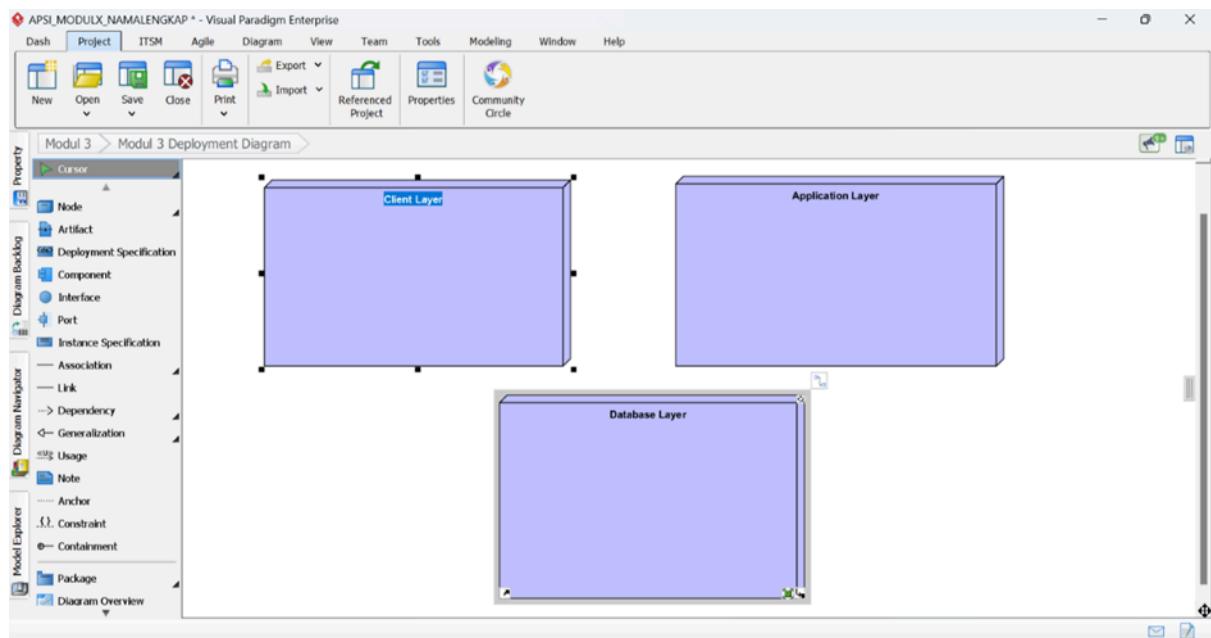
- X. Sebelum memulai pengerjaan, pastikan untuk menyimpan project pada device masing-masing praktikan dengan klik **Project** lalu **Save as**. Lokasi penyimpanan dibebaskan kepada praktikan.



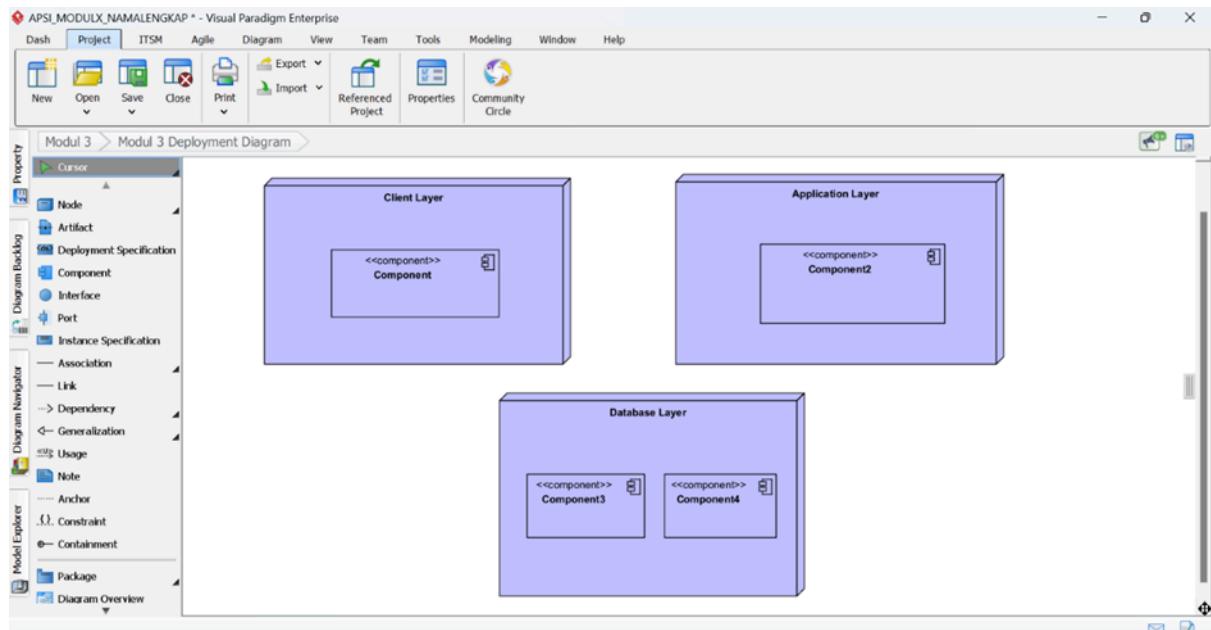
XI. Dari toolbar pilih **Node** untuk ditambahkan ke worksheet



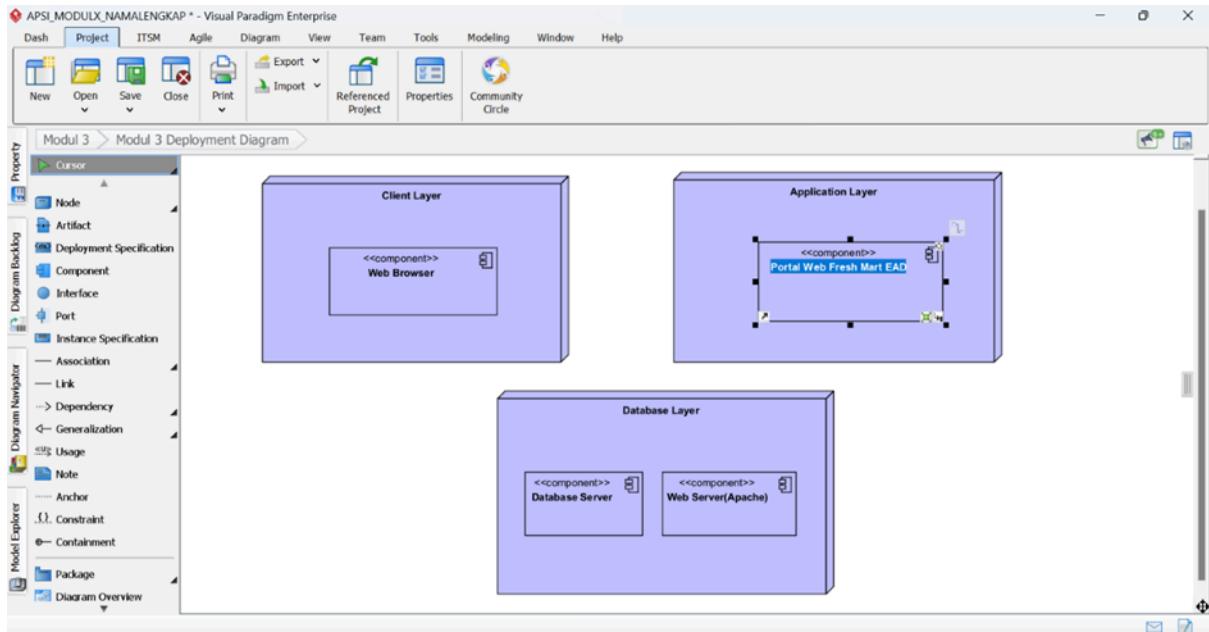
XII. Klik dua kali pada **Node** untuk mengganti nama **Node** sesuai dengan studi kasus.



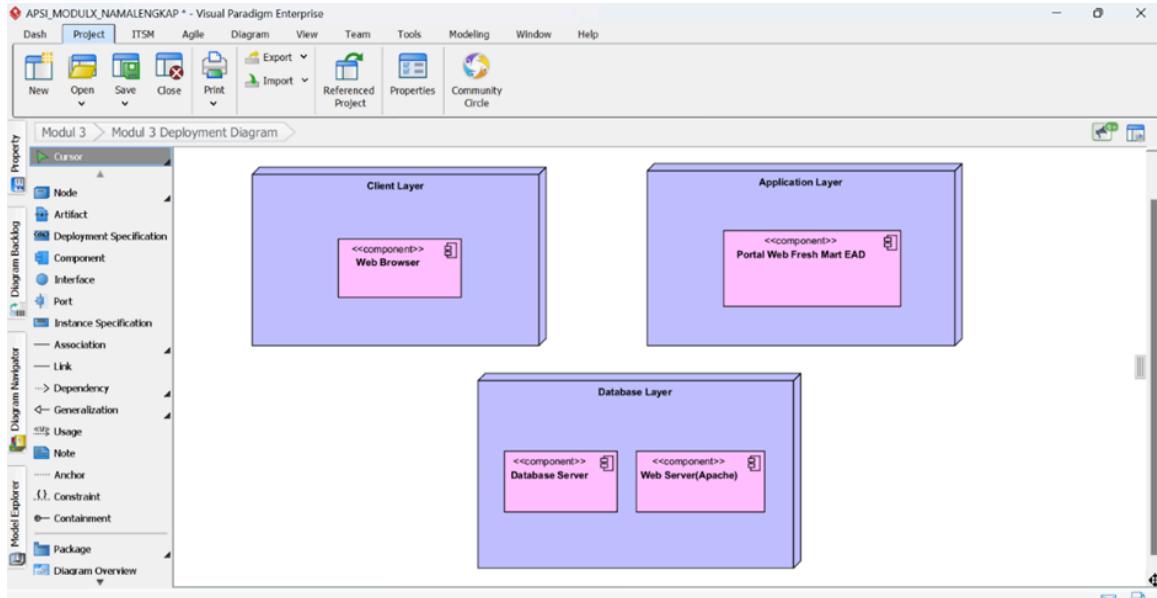
- XIII. Dari toolbar, pilih **Component** untuk ditambahkan ke worksheet, letakkan **Component** di dalam **Node**.



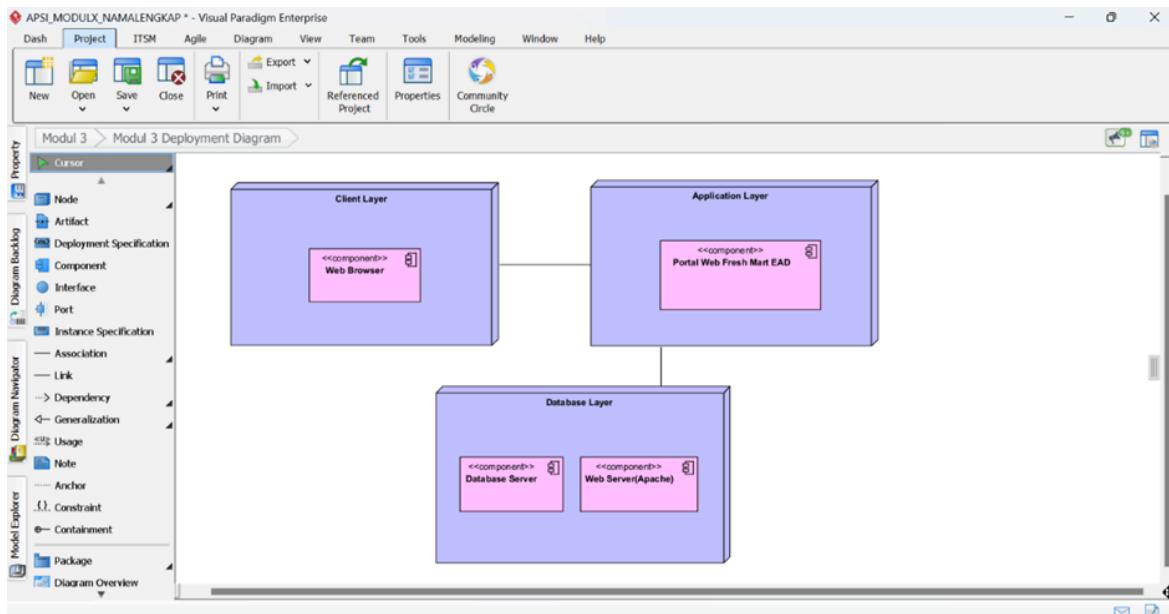
- XIV. Klik dua kali pada **Component** untuk mengganti nama **Component**.



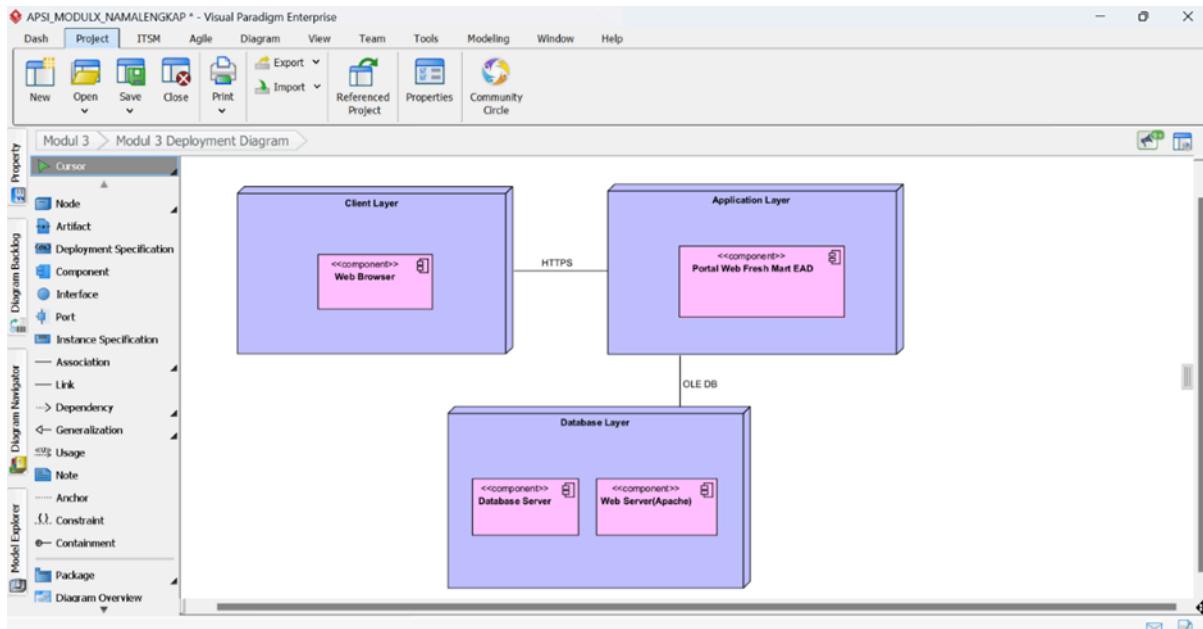
- XV. Klik kanan pada **Component**, lalu pilih “**Styles and Formatting**”, kemudian klik “**Formats**”, pilih warna yang diinginkan, lalu klik “**Set as Default for Component**”.

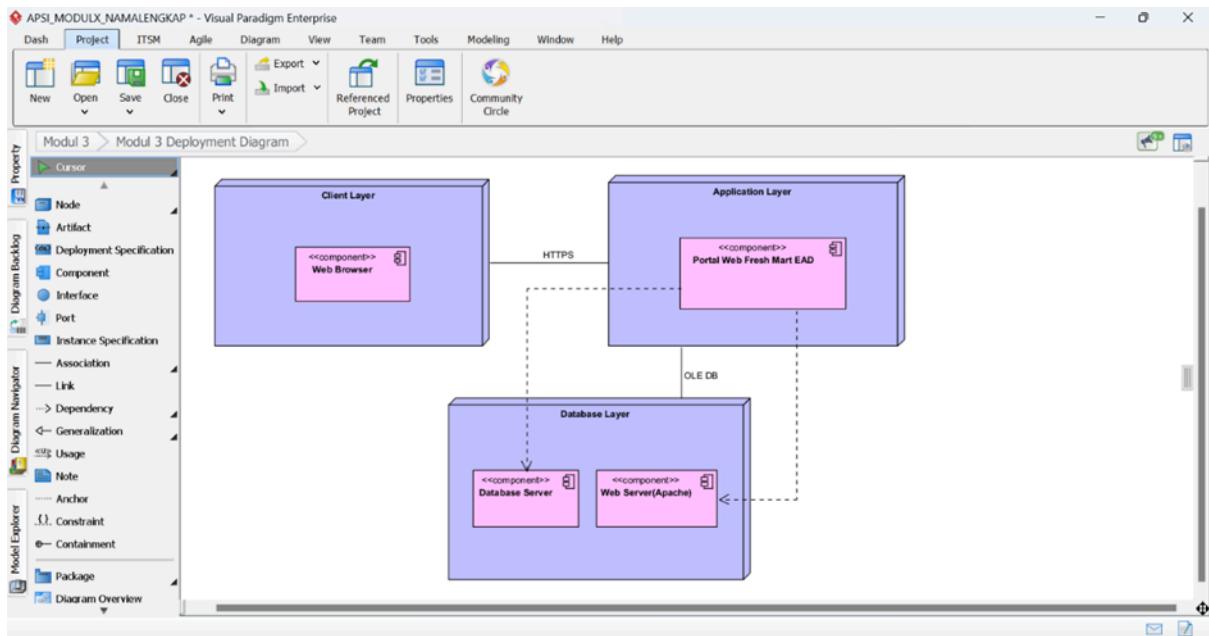
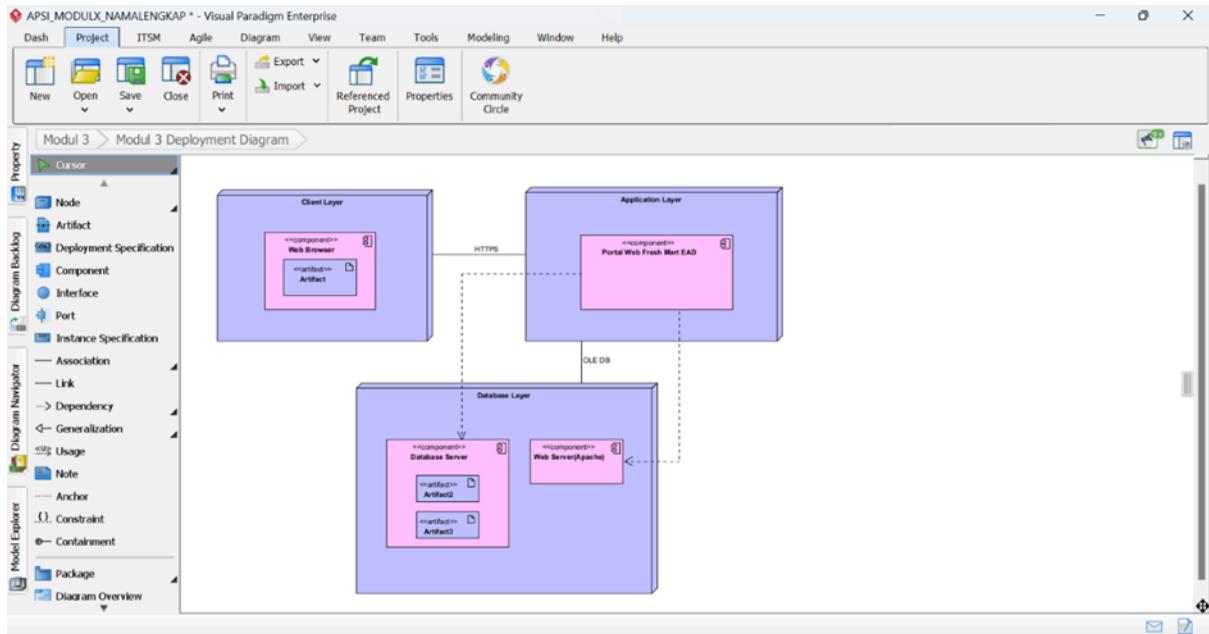


- XVI. Pada toolbar pilih notasi penghubung “**Association**” yang dapat menggambarkan hubungan antar **Component/Node**, lalu tarik ke **Component/Node** yang berhubungan.

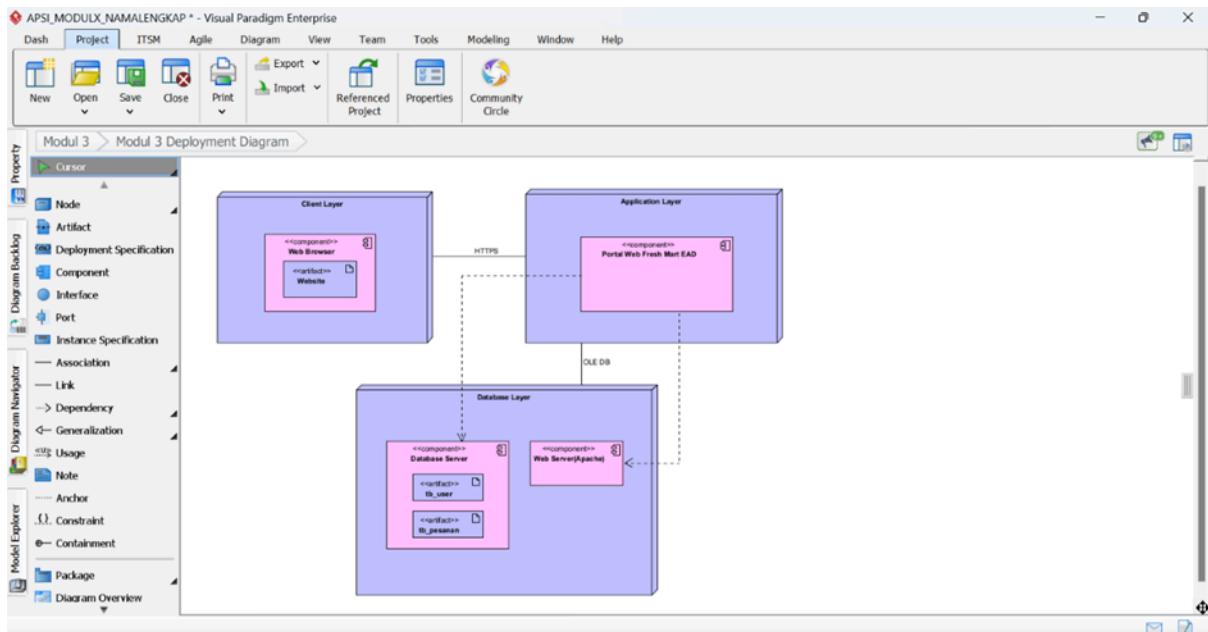


- XVII. Klik dua kali pada notasi penghubung untuk memberikan keterangan sesuai studi kasus.

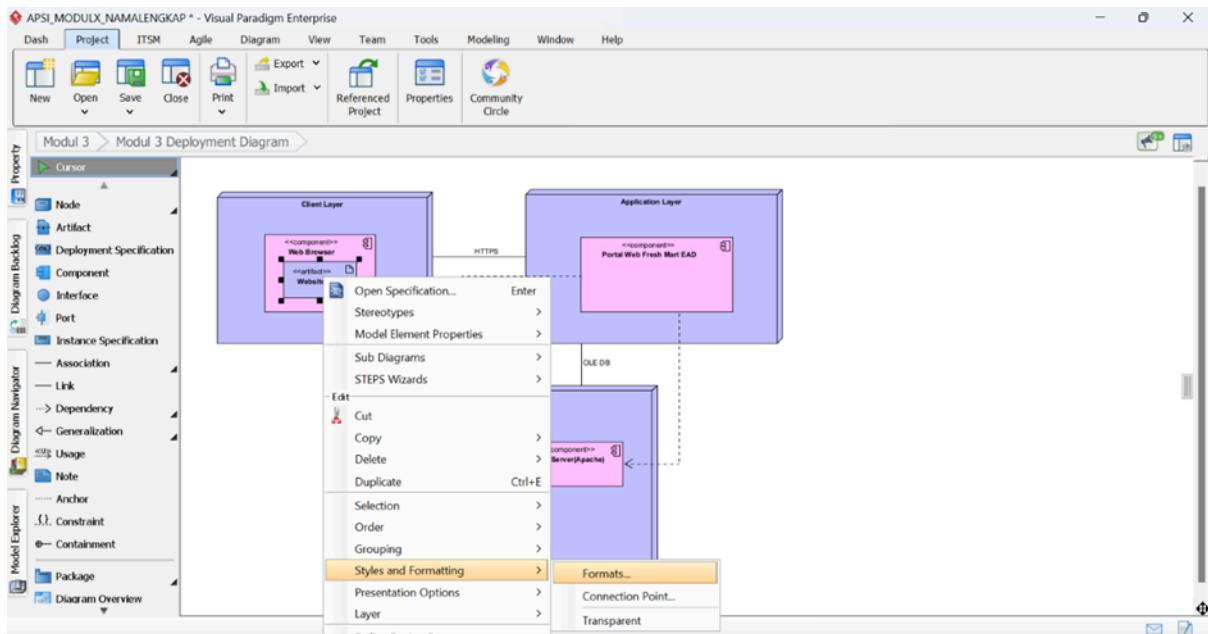


**XVIII.** Pada toolbar pilih notasi pengubung “**Depedency**”**XIX.** Pada toolbar pilih **Artifact** untuk ditambahkan ke dalam worksheet, pastikan **Artifact** berada di dalam **Component**.

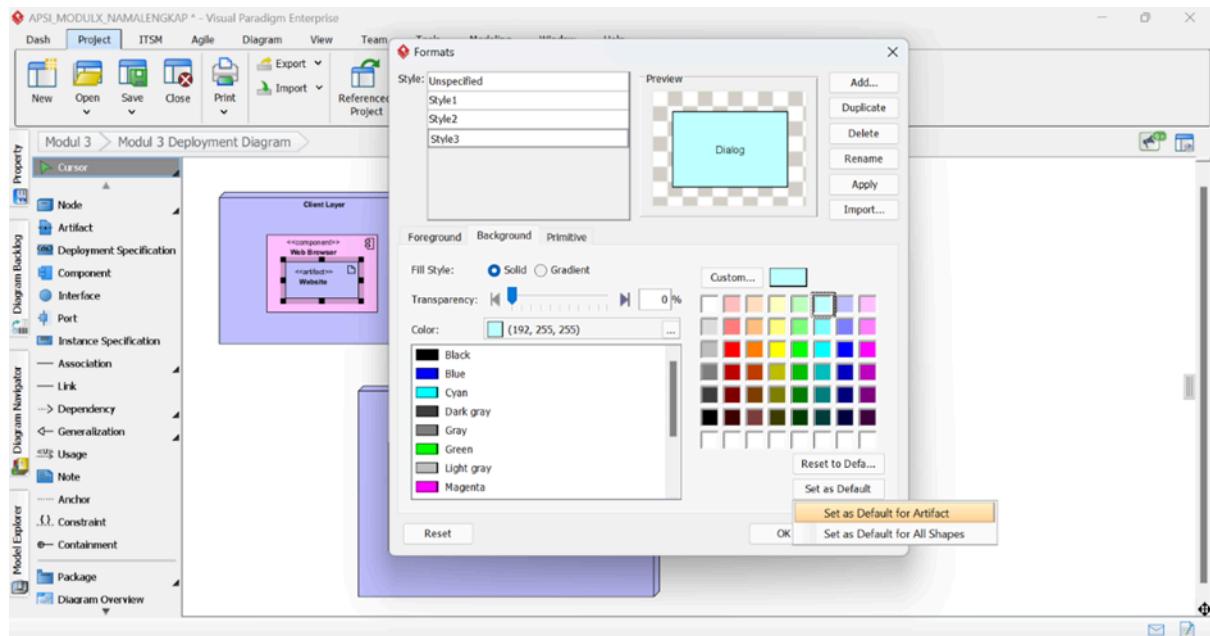
XX. Klik dua kali pada **Artifact** untuk mengganti nama sesuai dengan studi kasus.



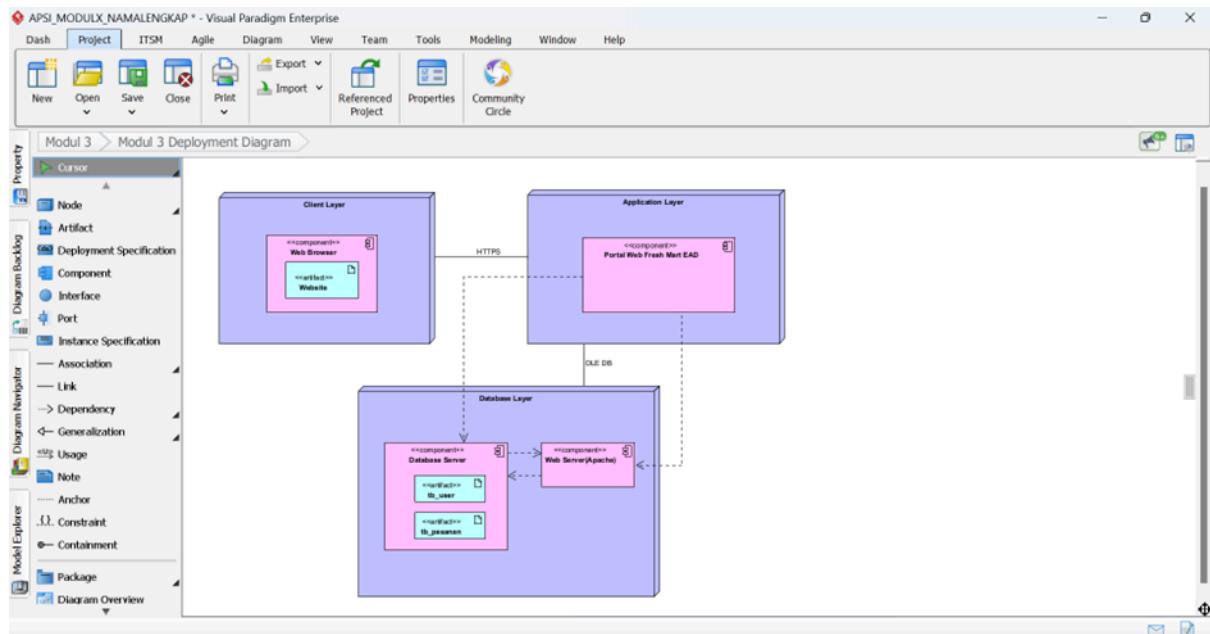
XXI. Klik kanan pada **Component**, lalu pilih “**Styles and Formatting**”, kemudian klik **“Formats”**



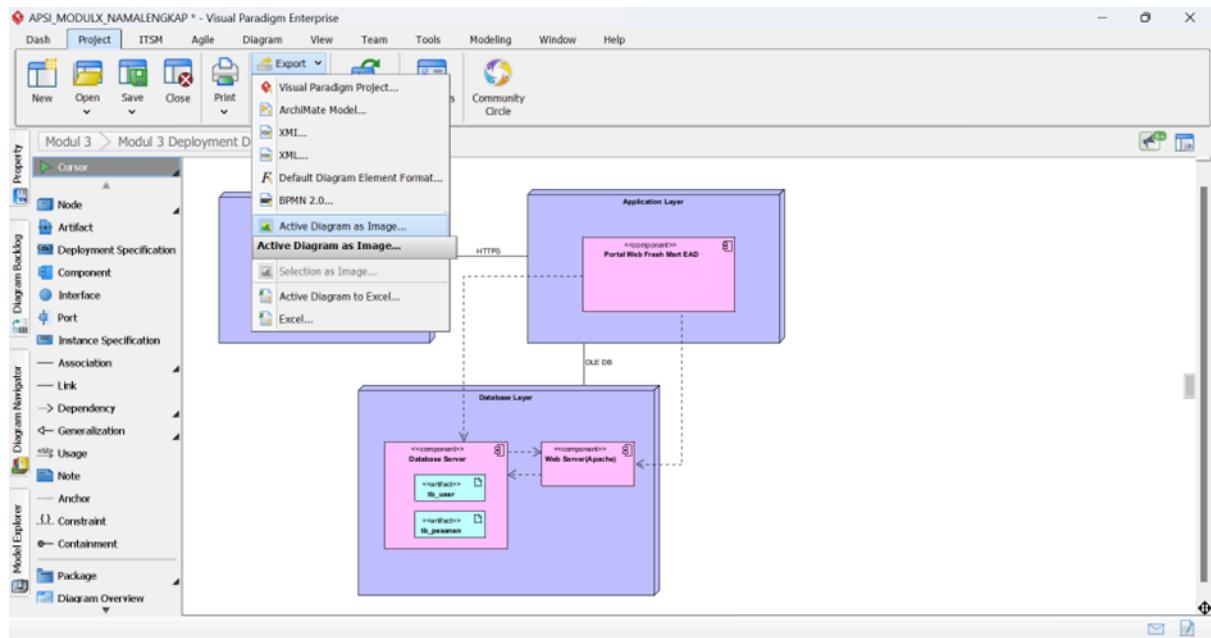
- XXII. Pilih warna yang diinginkan, lalu klik "**Set as Default for Component**", kemudian klik "**Apply to Current Diagram**".



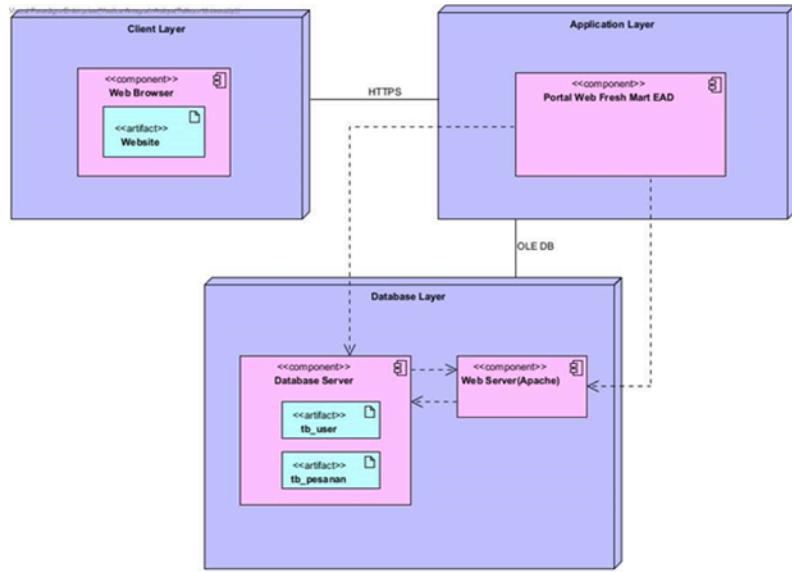
- XXIII. Pada toolbar pilih notasi penghubung "**Dependency**" dan letakkan pada **Component Database server** dan **Web Server**



- XXIV. Untuk menyimpan hasil penggerjaan dalam bentuk png/jpg, klik **Project** > klik **Export** > klik **Active Diagram as Image**.

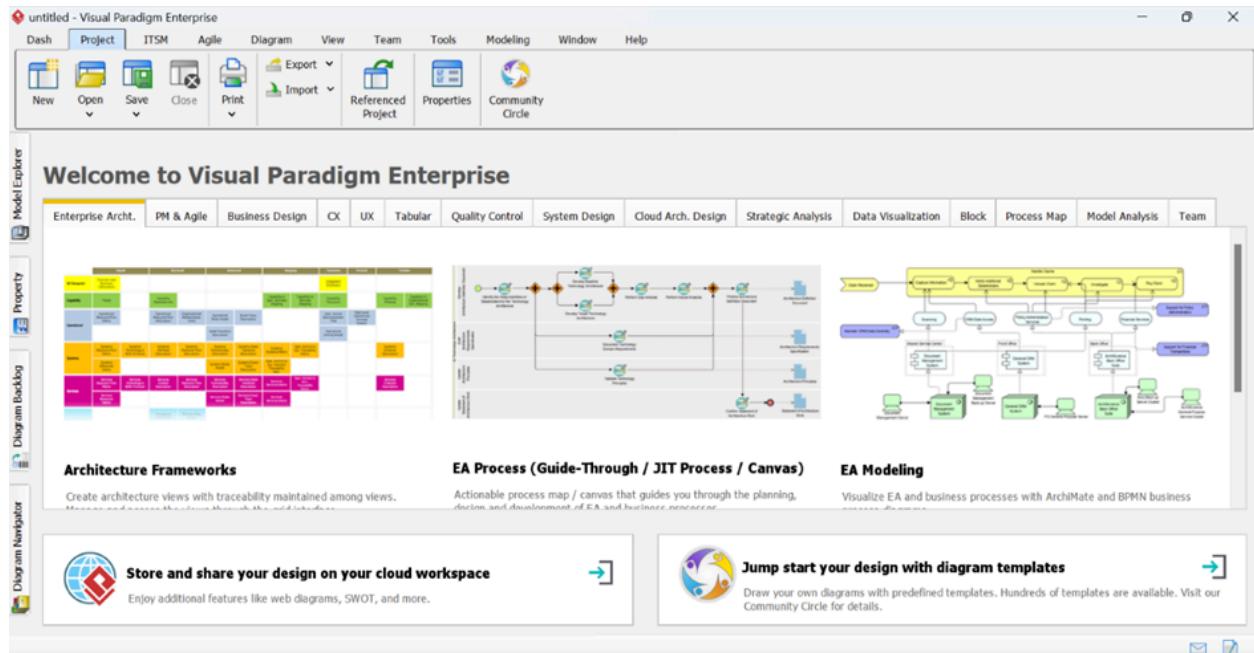


- XXV. Berikut hasil export nya

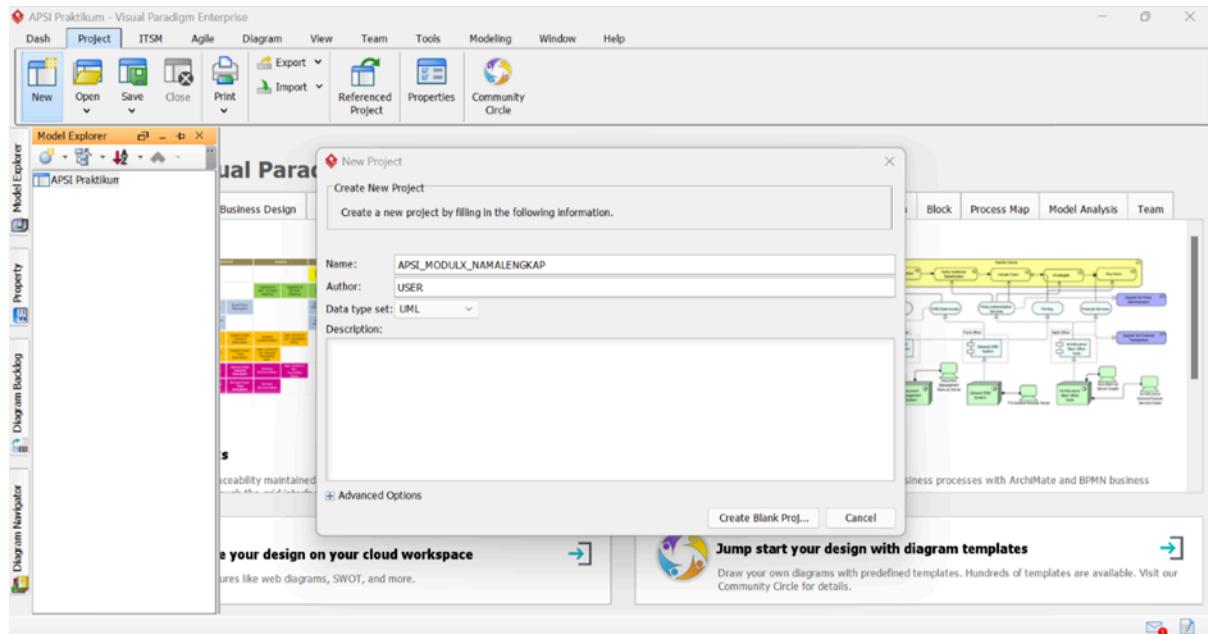


## B. Cara Membuat Component Diagram

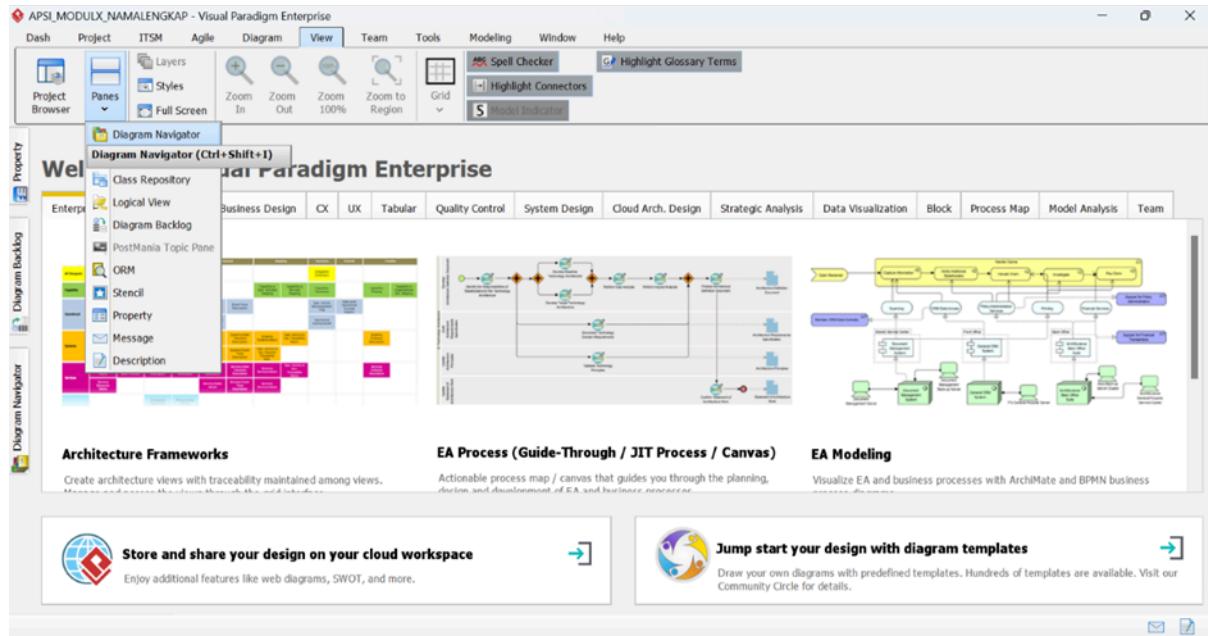
- I. Pada tampilan Dashboard Visual Paradigm, klik **Project** lalu klik **New**.



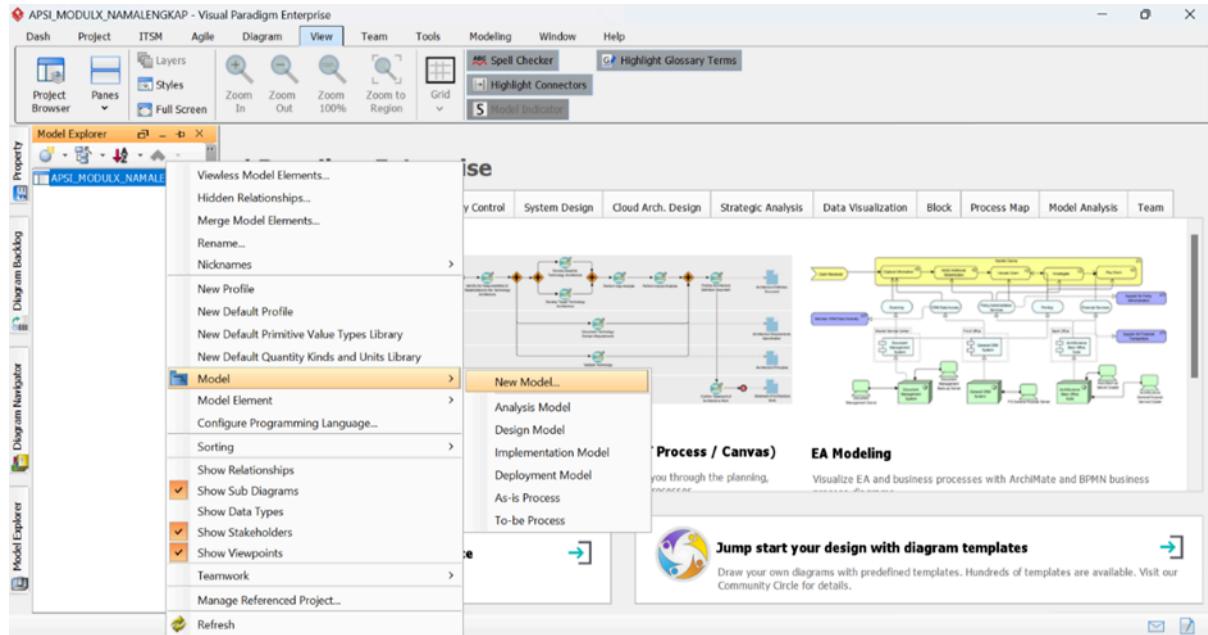
- II. Kemudian klik **"Create Blank Project"**.



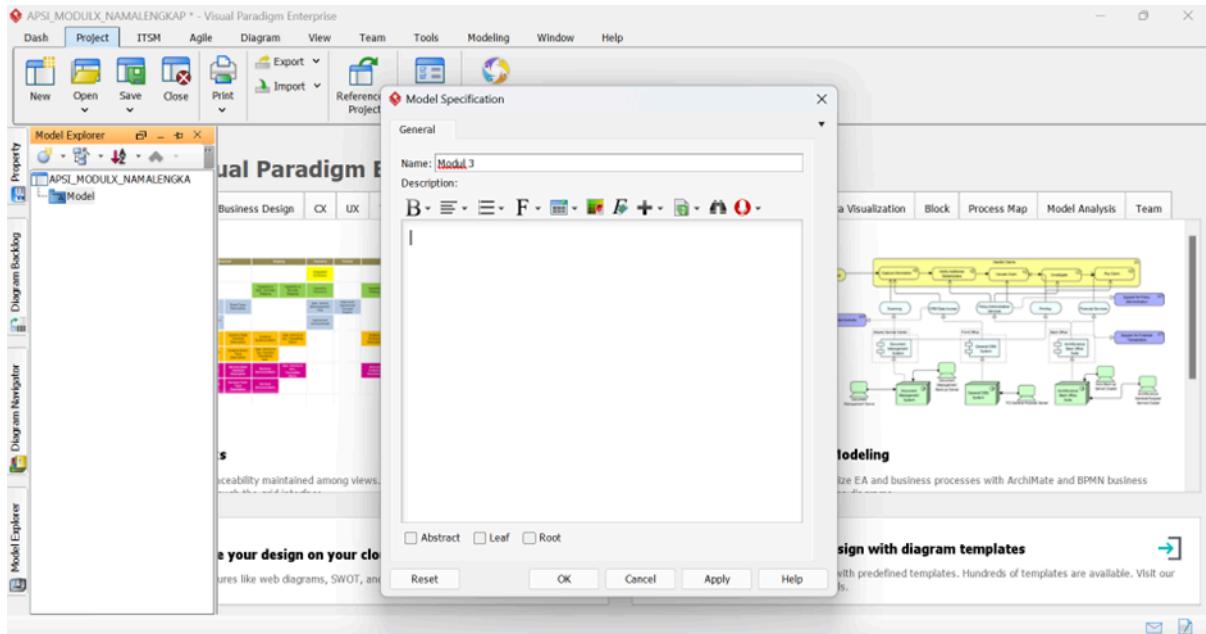
- III. Buka “**Model Explorer**” pada Tab “**View**” > “**Panes**”, atau bisa menggunakan shortcut CTRL + SHIFT + O .



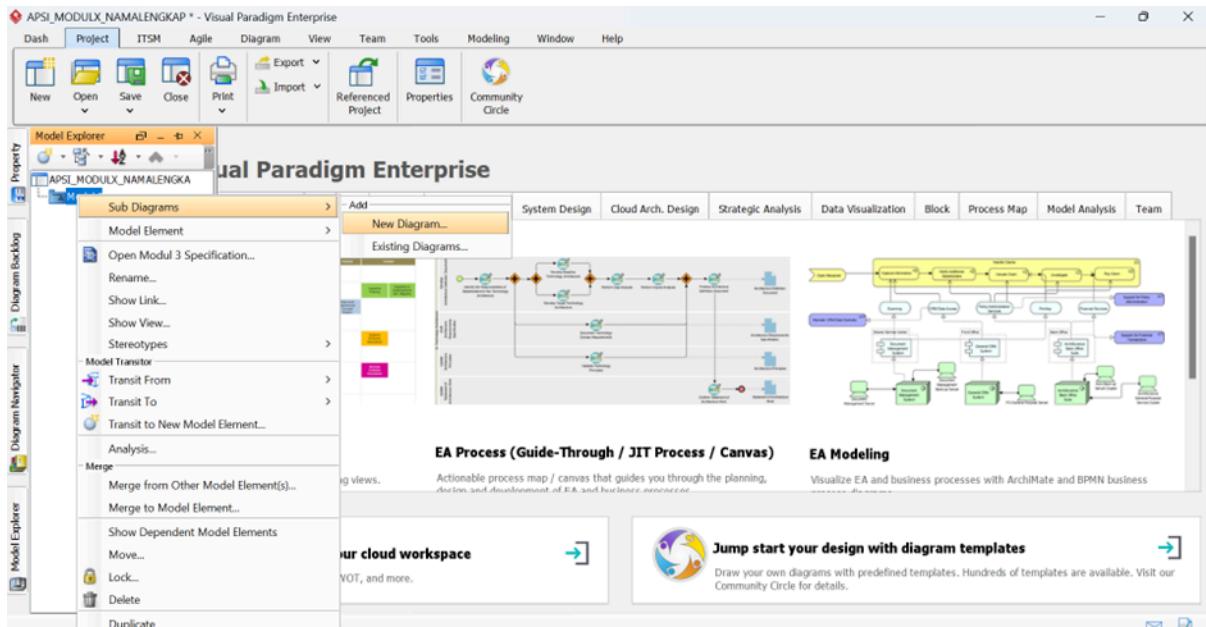
- IV. Klik kanan pada **Project** tersebut > klik **Model** > klik **New Model**.



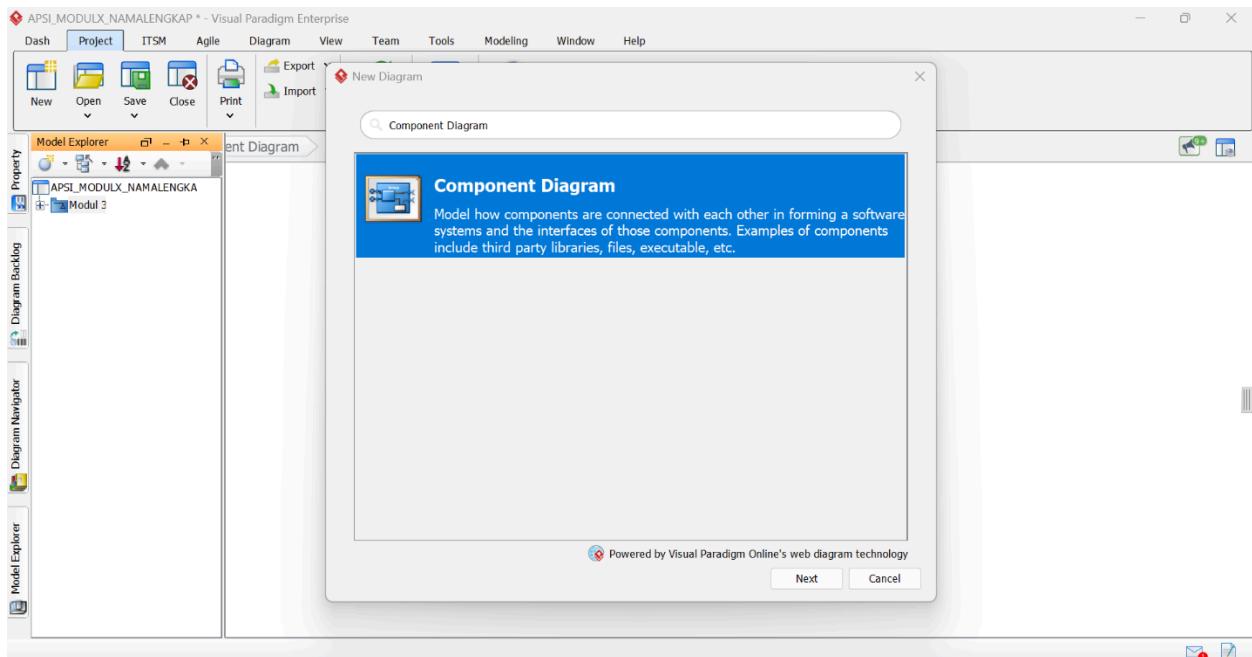
V. Beri nama model tersebut sesuai dengan modul yang dikerjakan.



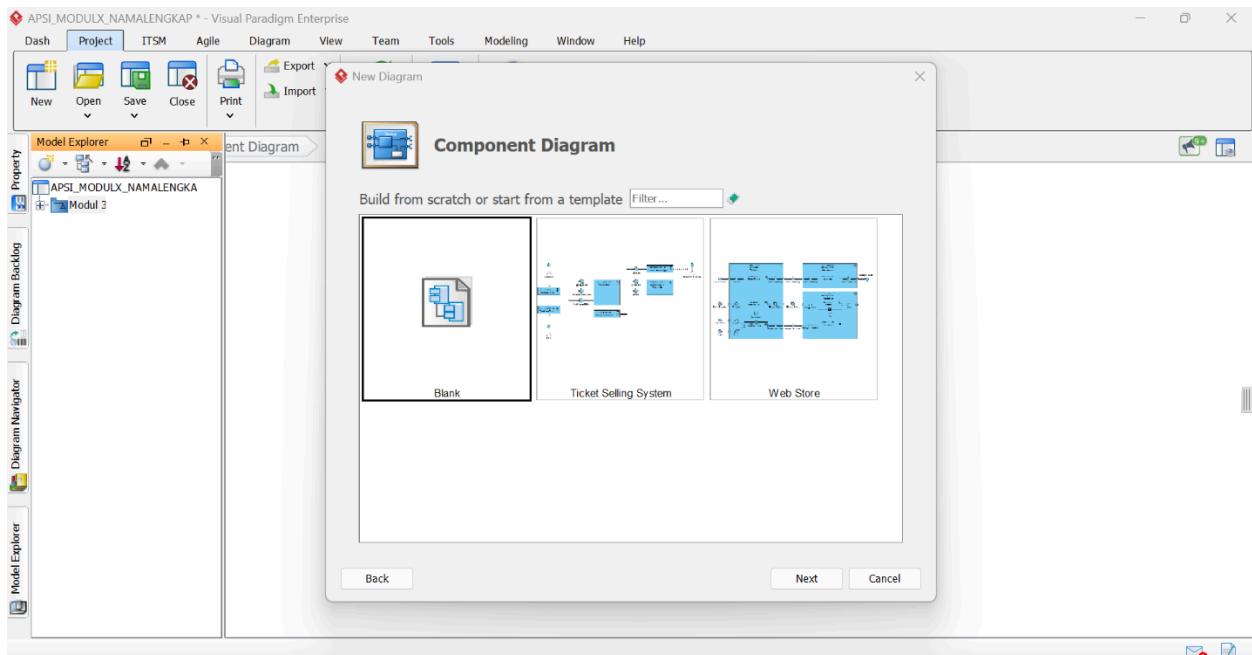
VI. Klik kanan pada model yang telah dibuat > klik **Sub Diagram** > **New Diagram**.



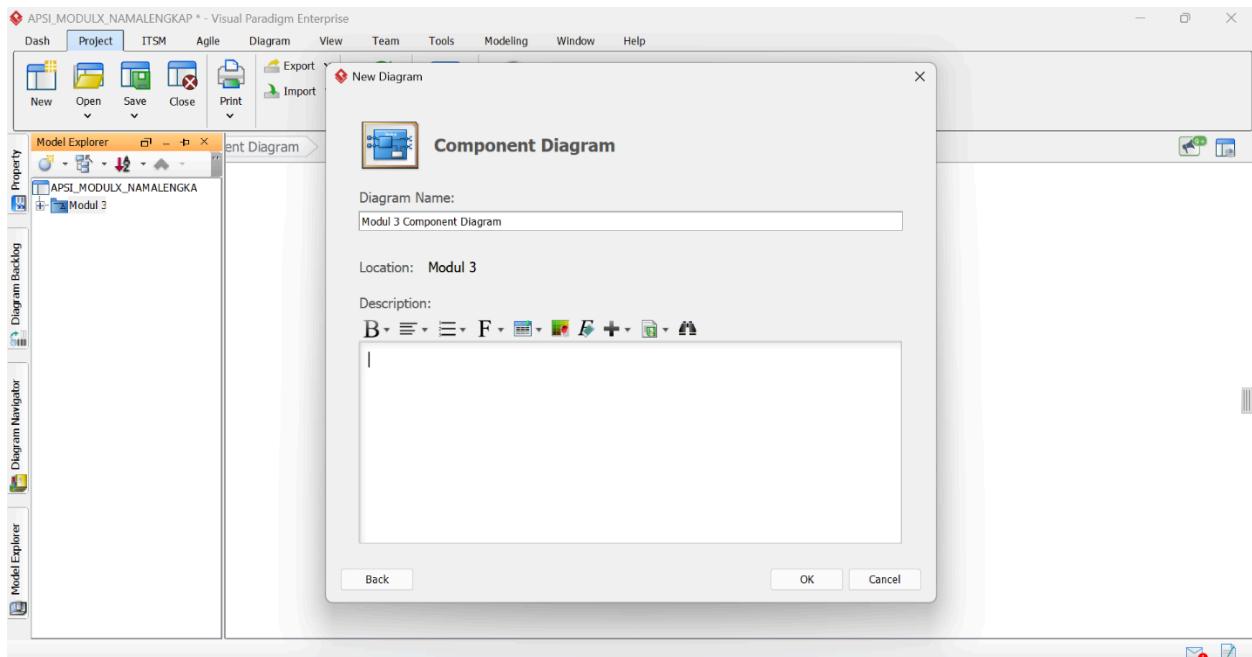
VII. Cari “**Component Diagram**” di pencarian, lalu klik **Next**.



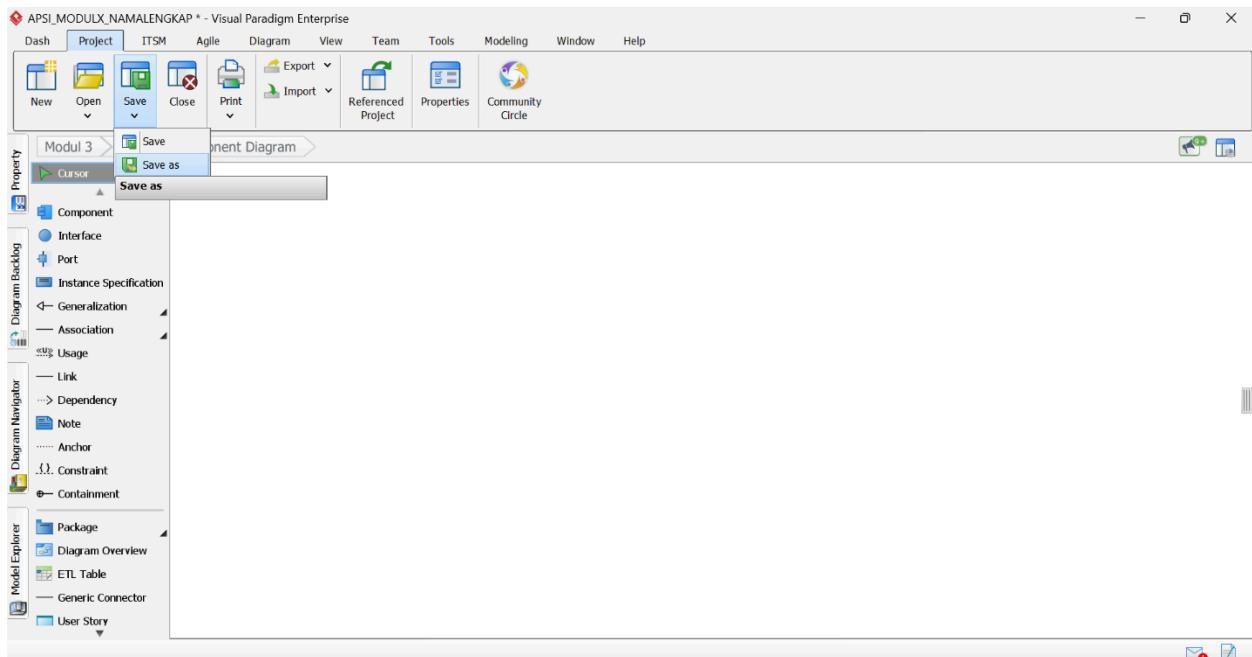
VIII. Klik “**Blank**” lalu klik **Next**.



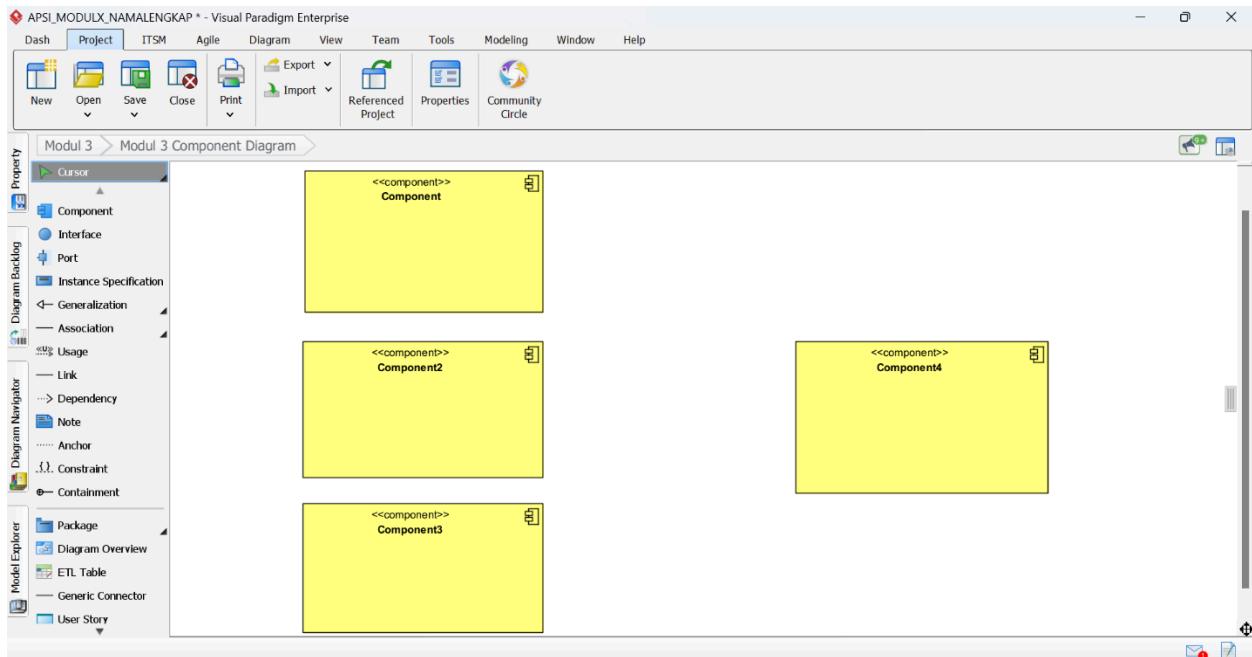
- IX. Berikan nama sesuai modul dan diagram yang dikerjakan.



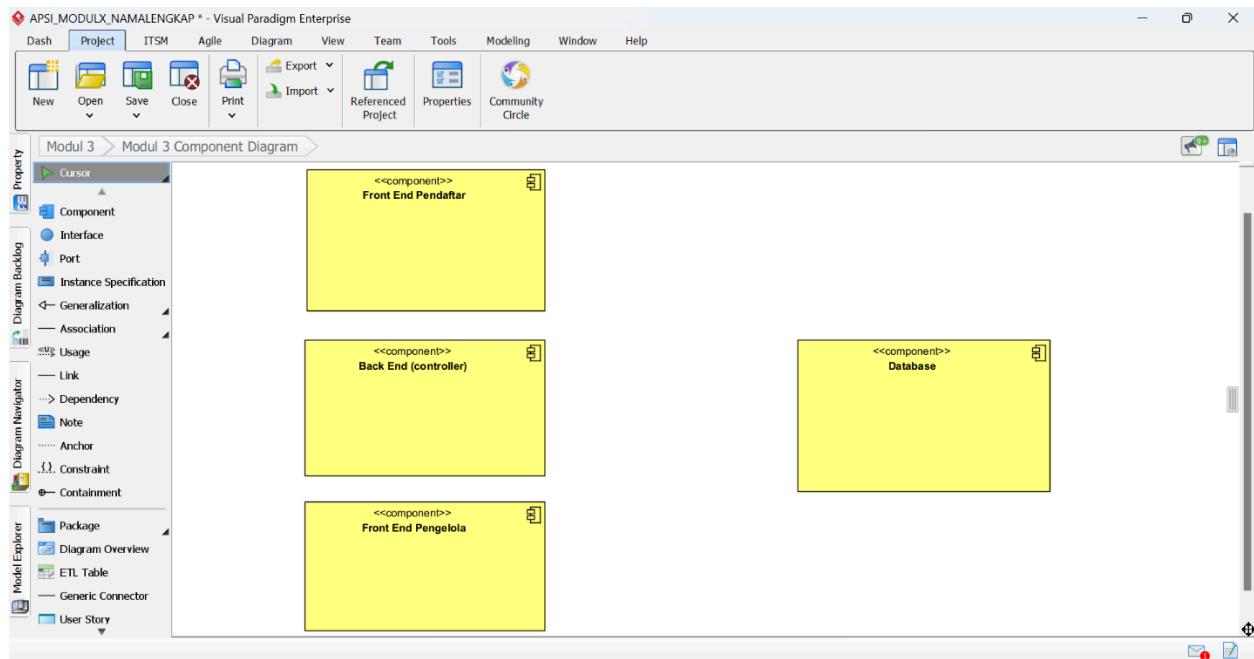
- X. Sebelum memulai penggeraan, pastikan untuk menyimpan project pada device masing-masing praktikan dengan klik **Project** lalu **Save as**. Lokasi penyimpanan dibebaskan kepada praktikan.



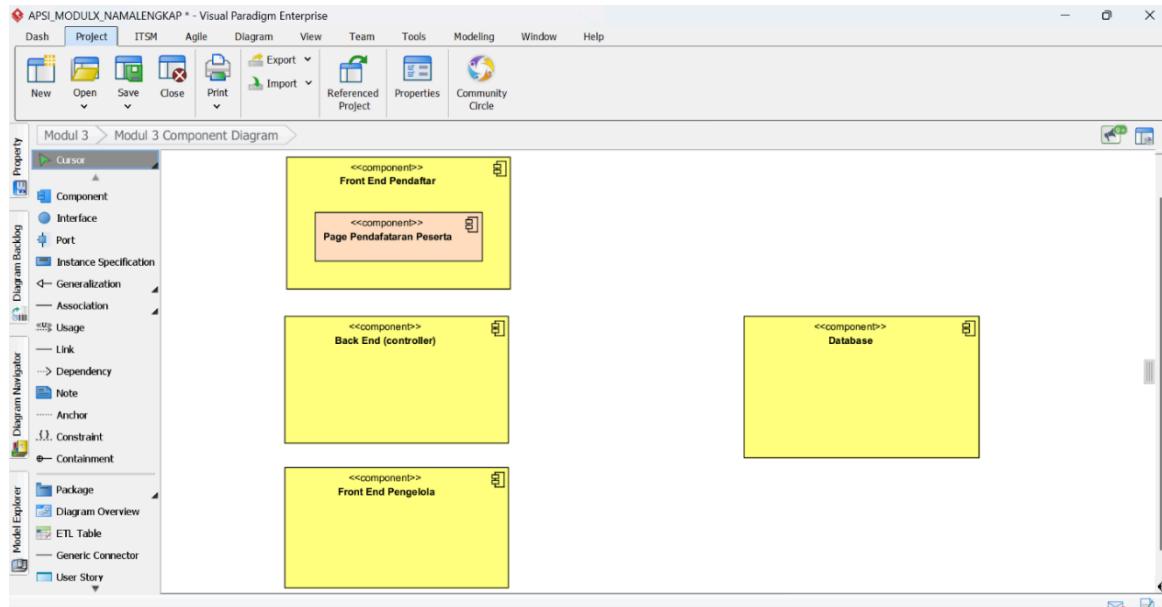
XI. Setelah project disimpan, pilih **Component** pada bagian **Toolbar**.



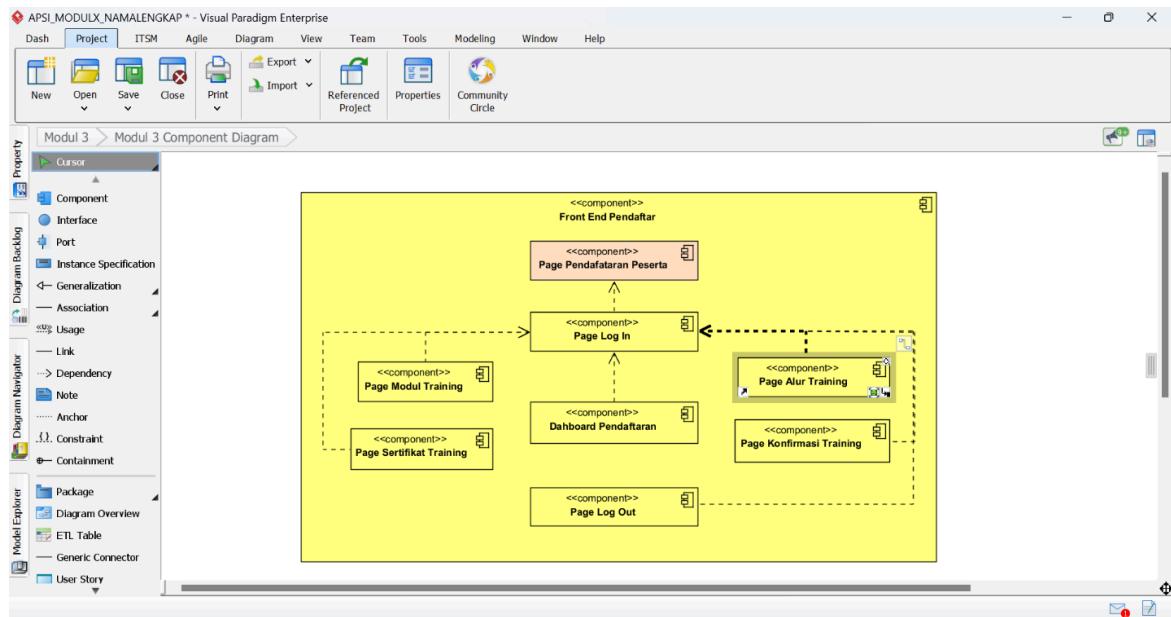
XII. Klik dua kali pada **Component** untuk mengganti nama **Component** sesuai dengan studi kasus.



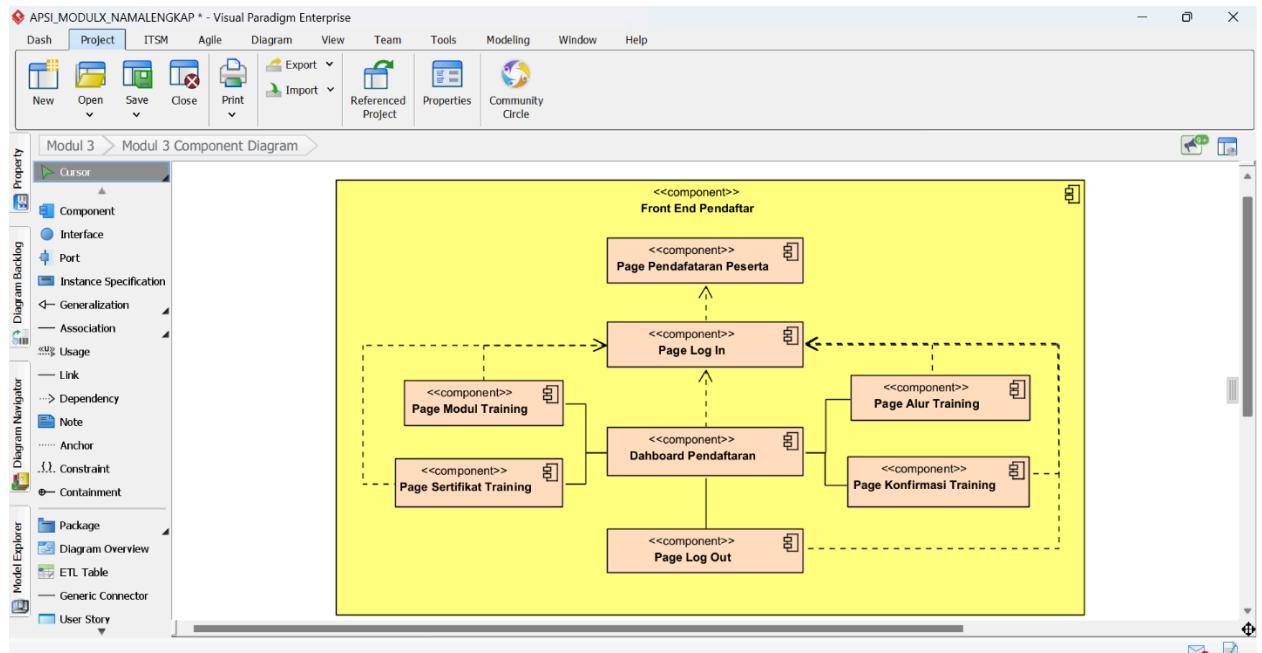
- XIII. Kemudian pilih **Component** pada **Toolbar** dan letakkan di dalam **Component Utama** sesuai dengan studi kasus. Jangan lupa untuk mengganti nama **Component**-nya.



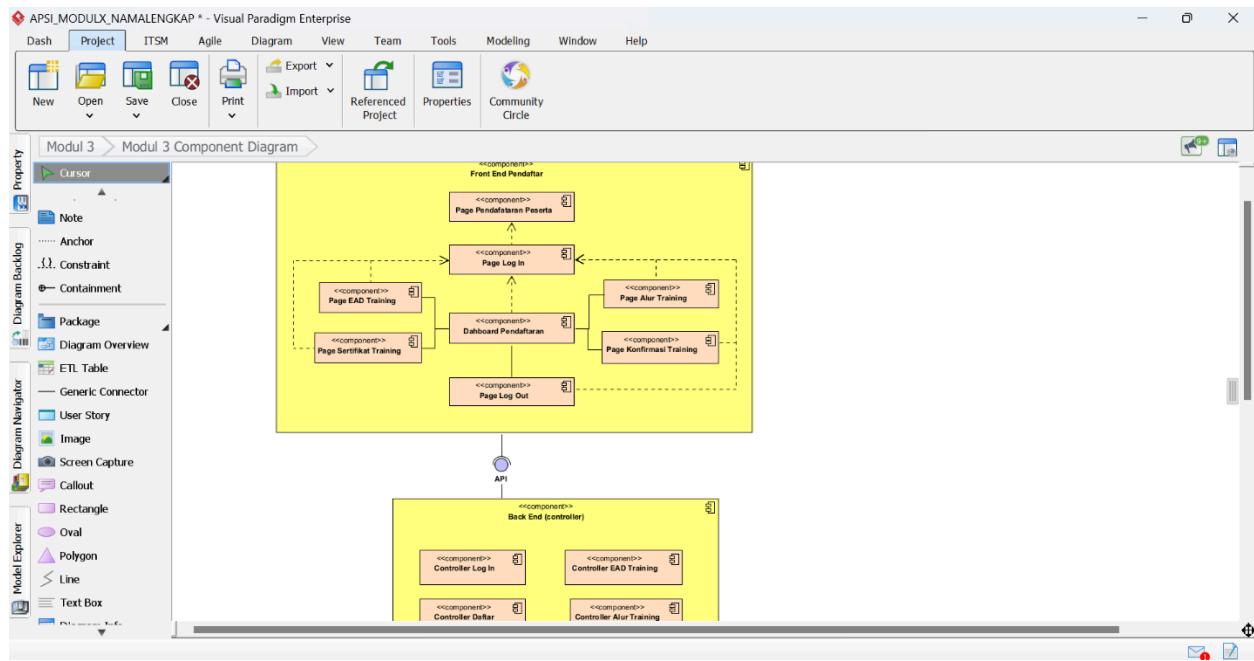
- XIV. Setelah itu, buat hubungan antar **Component** di dalam **Component** nya yaitu **Dependency**.



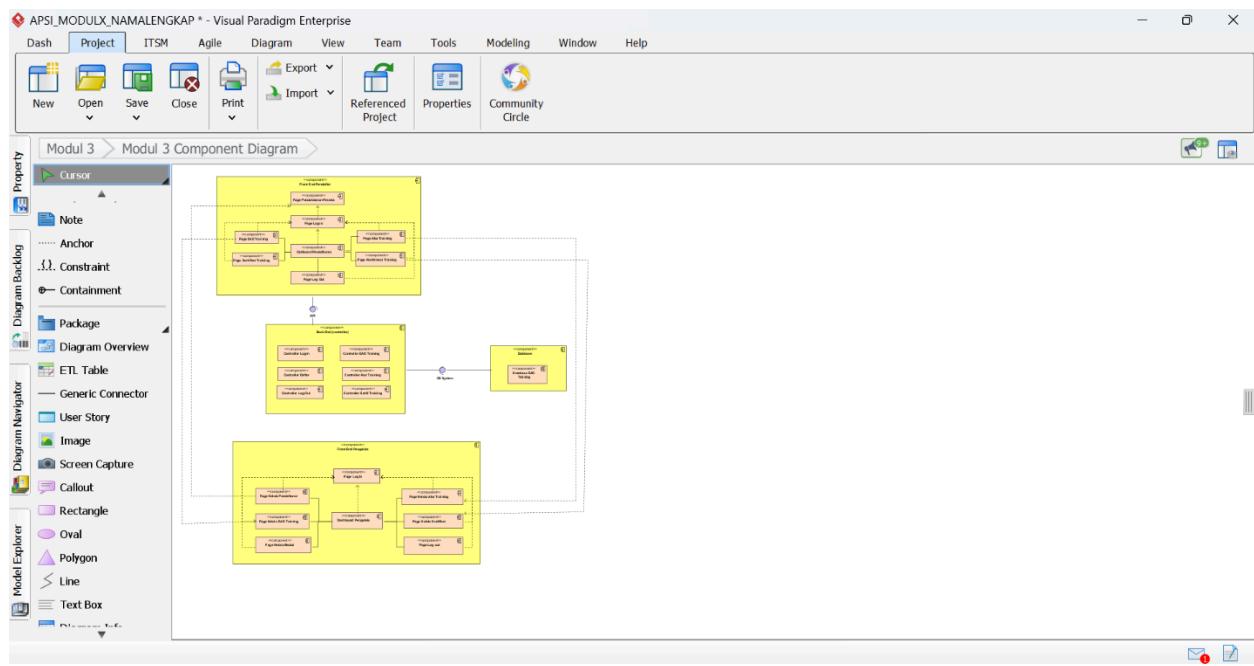
- XV. Setelah membuat **Dependency**, pilih **Association** untuk menghubungkan antar **Component** yang tidak bergantung.



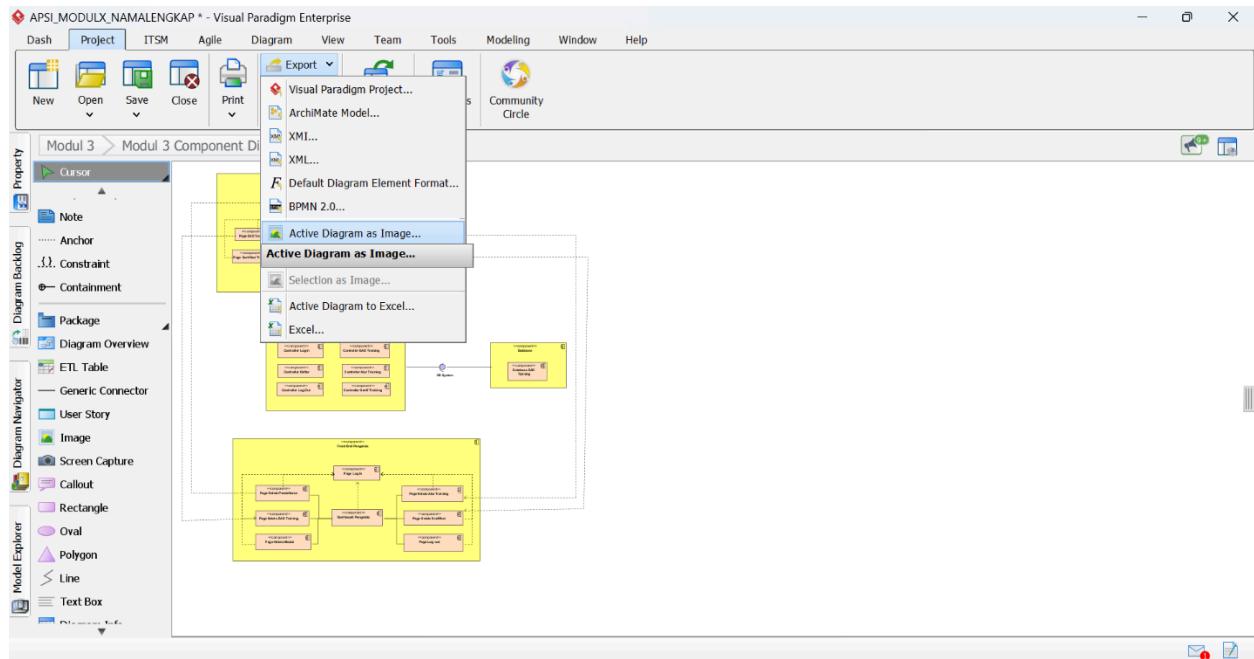
- XVI. Setelah **Component** Utama selesai dan dapat berkomunikasi, maka pilih **Port** dan sambungkan sesuai dengan **Provide** dan **Required Interface**.



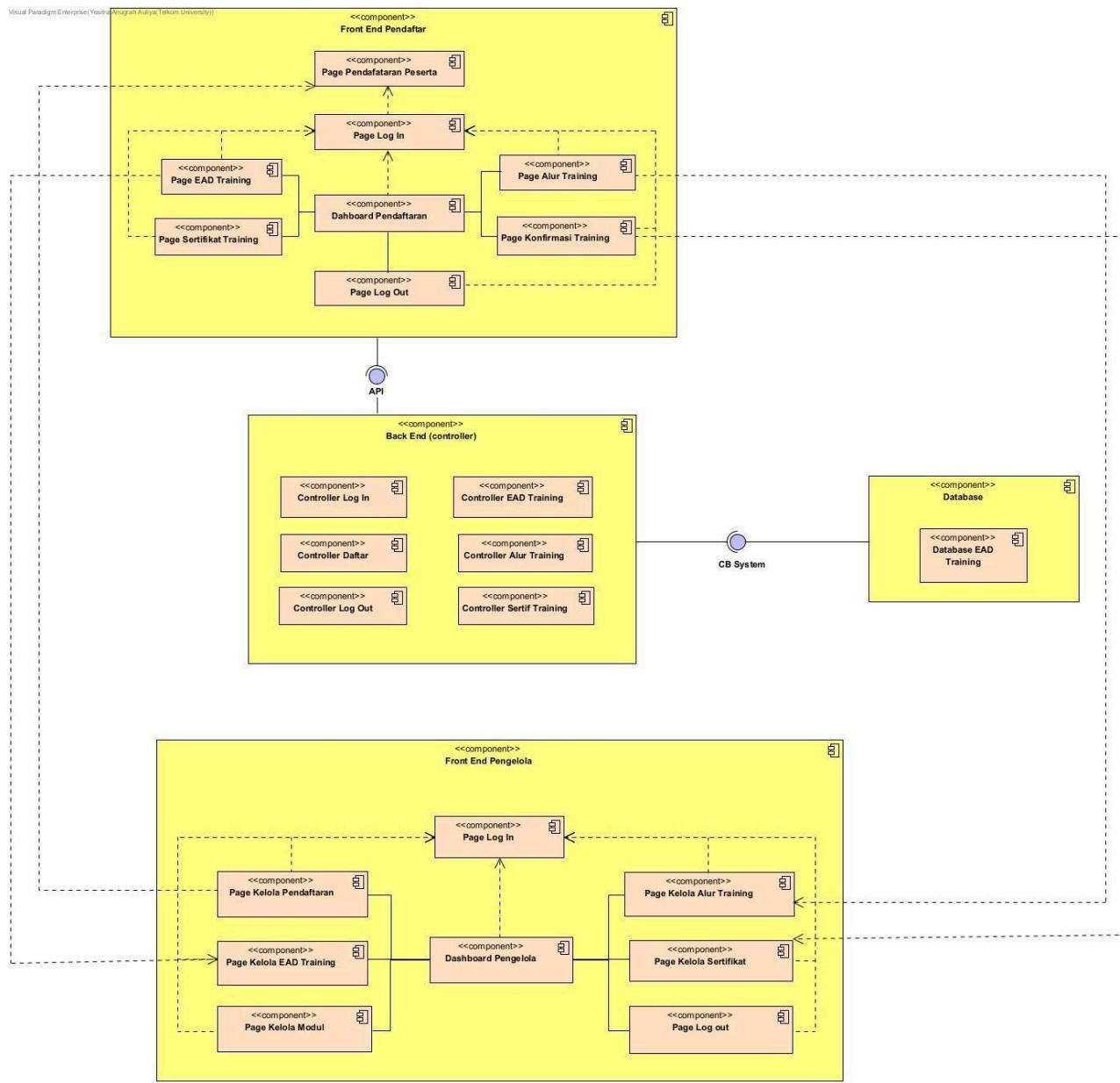
- XVII. Selesaikan studi kasus hingga seluruh narasi tergambarkan dalam **Component Diagram**.



XVIII. Eksport diagram dengan klik tab Project, Export, Active Diagram as Image.



XIX. Berikut hasil export nya.



## CONTOH STUDI KASUS

### A. Studi Kasus 1 (Level Mudah)

MyClean adalah sebuah layanan berbasis digital yang menyediakan jasa pembersihan rumah secara profesional. Layanan ini hadir untuk membantu masyarakat dalam menemukan tenaga kebersihan yang andal, efisien, dan sesuai dengan kebutuhan mereka. Dengan meningkatnya kesibukan masyarakat, banyak individu dan keluarga yang membutuhkan layanan kebersihan rumah secara fleksibel tanpa harus merekrut pekerja tetap.

Untuk mengoptimalkan layanan, MyClean mengembangkan Portal Web MyClean, sebuah platform yang memungkinkan pelanggan mencari, memesan, dan mengelola layanan pembersihan rumah secara mudah. Portal ini dapat diakses oleh dua jenis pengguna utama, yaitu Pelanggan dan Penyedia Jasa (Pembersih Rumah).

Sebelum layanan dapat digunakan, penyedia jasa harus mendaftarkan layanan pembersihan yang mereka tawarkan di portal web. Mereka juga memiliki opsi untuk memperbarui informasi atau menghapus layanan yang sudah tidak tersedia.

Sementara itu, pelanggan dapat melakukan pencarian layanan berdasarkan lokasi, harga, dan jenis layanan yang dibutuhkan. Setelah menemukan layanan yang sesuai, pelanggan dapat melihat detail kontak penyedia jasa untuk melakukan pemesanan atau konsultasi lebih lanjut.

#### 1. Deployment Diagram

Pada Sistem Website MyClean ini menggunakan four-tier architecture sehingga dapat meningkatkan performa dan juga meningkatkan integrasi data. Four-tier architecture terdiri dari:

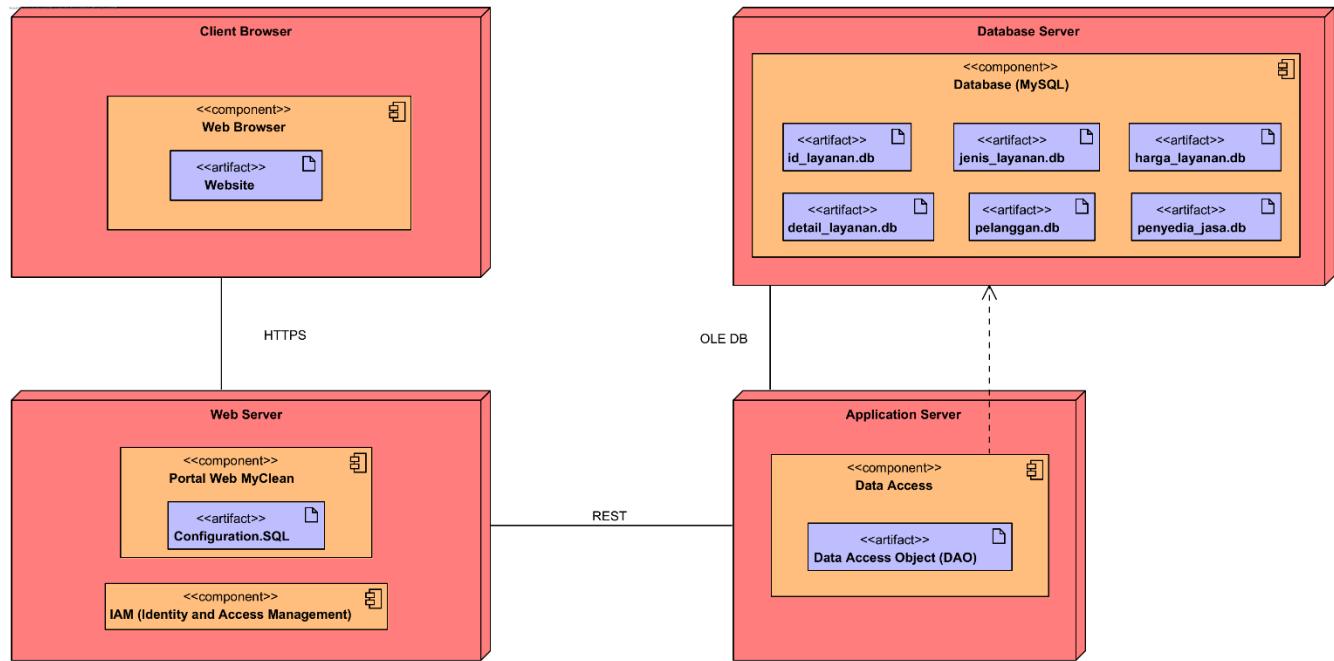
- 1) **Client Browser:** Representasi dari browser web yang digunakan oleh pelanggan dan penyedia jasa (pembersih rumah). Berfungsi sebagai antarmuka untuk menampilkan informasi dan hasil yang dibutuhkan oleh pengguna. Hal ini dapat dilakukan melalui website MyClean yang dapat diakses oleh pengguna melalui **Web Browser**. Di **Web Browser**

tersebut terdapat **Website** yang berfungsi sebagai antarmuka untuk menampilkan informasi layanan pembersihan serta memfasilitasi interaksi antara pelanggan dan penyedia jasa.

- 2) **Web Server:** Ini adalah tempat di mana logika website **Portal Web MyClean** dijalankan dan mengatur aliran data antara **Client Browser** dan **Web Server** menggunakan protokol **HTTPS**. Pada infrastruktur ini, MyClean membutuhkan **Portal Web MyClean** yang di dalamnya terdapat **Configuration.SQL** yang digunakan untuk mengatur berbagai parameter dan konfigurasi sistem. Terdapat juga **IAM** (Identity and Access Management) yang mengelola keamanan akses pengguna dengan membedakan hak akses antara pelanggan dan penyedia jasa. **Portal Web MyClean** bergantung pada **IAM** untuk memastikan bahwa setiap pengguna memiliki akses sesuai dengan perannya dalam sistem.
- 3) **Application Server:** Bertanggung jawab dalam eksekusi aplikasi, memproses permintaan pengguna, dan menyediakan berbagai layanan utama dari sistem. **Application Server** dihubungkan dengan **Web Server** menggunakan protokol komunikasi **REST**. Terdapat **Data Access** yang bertugas untuk mengelola interaksi antara aplikasi dan database. Dalam Data Access ini terdapat file **Data Access Object (DAO)** yang memungkinkan eksekusi operasi seperti pencarian, penambahan, pembaruan, dan penghapusan layanan pembersihan. **Data Access** bergantung pada **Database (MySQL)** untuk menyimpan dan mengambil data yang diperlukan oleh sistem.
- 4) **Database Server:** Database Server menyimpan semua data yang diperlukan oleh sistem, termasuk informasi terkait layanan pembersihan dan pengguna. Database Server terhubung dengan Application Server melalui protokol **OLE DB** untuk memastikan pengambilan dan penyimpanan data berjalan optimal. Pada layer ini, dibutuhkan **Database (MySQL)** yang mencakup berbagai tabel data,

seperti **id\_layanan.db**, **jenis\_layanan.db**, **harga\_layanan.db**, **detail\_layanan.db**, **penyedia\_jasa.db**, serta **pelanggan.db**.

## Hasil Study Case



## 2. Component Diagram

Portal Web MyClean menggunakan konsep **Front End, Back End, dan Database** dengan perspektif aliran fitur. Berikut merupakan peran dari masing-masing komponen:

### 1) Front End

Front End merupakan bagian dari sistem yang berhubungan langsung dengan antarmuka pengguna. Komponen ini terdiri dari

dua sub-komponen utama, yaitu **Penyedia Jasa (Pembersih Rumah)** dan **Pelanggan**.

- **Penyedia Jasa** yang bergabung dengan MyClean perlu **menambahkan layanan pembersihan** yang mereka tawarkan agar dapat ditemukan oleh pelanggan di portal web. Setelah layanan ditambahkan, penyedia jasa juga memiliki opsi untuk **memperbarui layanan pembersihan** jika ingin menyesuaikan detail layanan yang ditawarkan atau **menghapus layanan pembersihan** jika mereka tidak lagi tersedia. Selain itu, ketika pelanggan melakukan pemesanan, penyedia jasa dapat **melakukan konfirmasi pemesanan layanan pembersihan**, jika pesanan diterima maka jadwal pembersihan akan masuk ke sistem. Proses **menghapus layanan pembersihan** dan **memperbarui layanan pembersihan** ini memiliki relasi **dependency** terhadap proses **menambahkan layanan pembersihan**, karena layanan harus terlebih dahulu ada sebelum dapat dihapus atau diperbarui.
- **Pelanggan** yang mengakses portal web dapat **mencari layanan pembersihan rumah** yang tersedia sesuai dengan kebutuhan mereka. Setelah menemukan layanan yang sesuai, pelanggan dapat **melihat detail layanan pembersihan** dan pelanggan bisa langsung **memesan layanan pembersihan** melalui portal web. Setelah pemesanan dikonfirmasi, pelanggan akan **mendapat notifikasi konfirmasi pemesanan** yang sudah dipesan sebelumnya. Proses **memesan layanan pembersihan** memiliki relasi **dependency** terhadap **menambahkan layanan pembersihan** oleh penyedia jasa, karena pelanggan hanya dapat memesan layanan yang ditambahkan oleh penyedia jasa. Selain itu, proses **mendapat notifikasi konfirmasi pemesanan** memiliki relasi **dependency** terhadap **melakukan konfirmasi pemesanan layanan pembersihan** oleh penyedia jasa, karena

pelanggan hanya akan mendapatkan notifikasi konfirmasi pemesanan ketika penyedia jasa sudah melakukan konfirmasi pemesanannya.

Komponen **Front End** akan terhubung dengan **Back End** melalui **RESTful API** yang telah disediakan oleh **Back End**. **RESTful API** memungkinkan komunikasi antara kedua komponen secara terstruktur dan efisien. **Front End** akan menunjukkan ketergantungan required terhadap **Back End** karena membutuhkan layanan dari **Back End** untuk mendapatkan data dari Database serta menjalankan operasi bisnis.

## 2) Back End

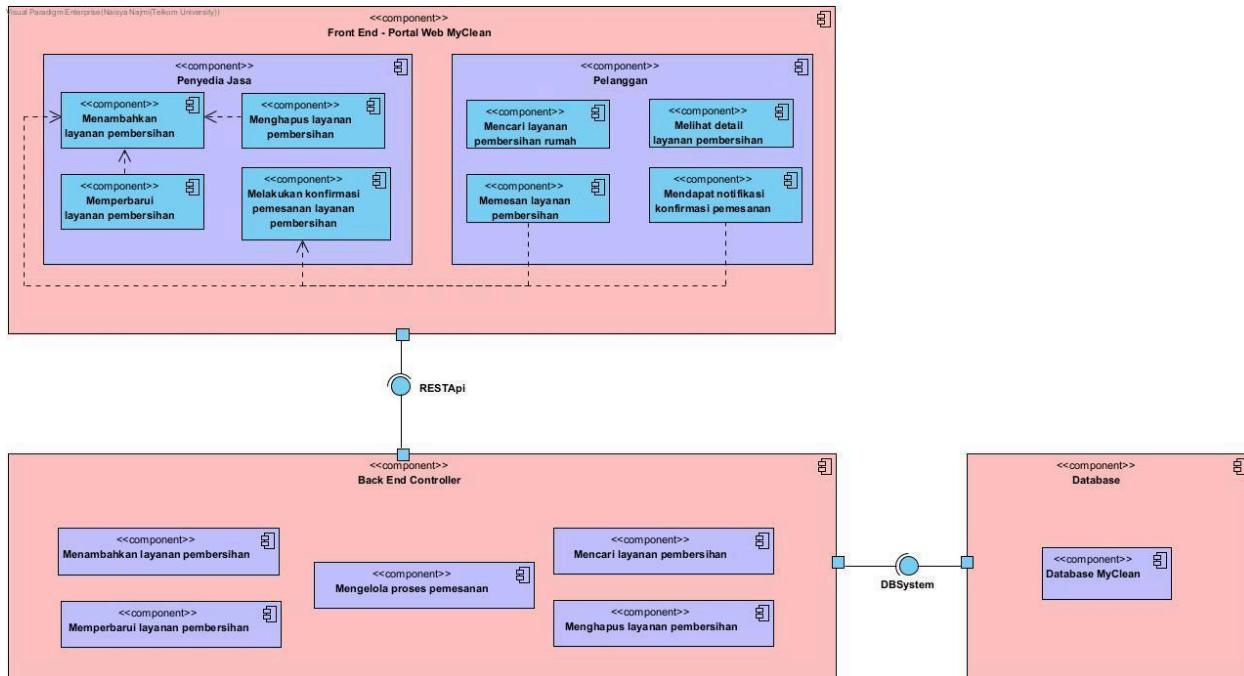
**Back End** merupakan bagian dari sistem yang menangani logika bisnis serta mengelola interaksi dengan database. Dalam **Portal Web MyClean**, Back End bertanggung jawab atas berbagai fitur utama yang memungkinkan operasional layanan berjalan dengan baik. Komponen ini terdiri dari beberapa controller, seperti **menambahkan layanan pembersihan**, **memperbarui layanan pembersihan**, **menghapus layanan pembersihan**, **mencari layanan pembersihan**, serta **mengelola proses pemesanan**. Komponen controller memiliki peran dalam mengelola aliran data serta menjalankan logika bisnis yang terkait dengan setiap fungsinya. **Back End** berinteraksi dengan **Database** melalui interface **DBSystem**, yang telah disediakan untuk mendukung pengelolaan data. Oleh karena itu, **Back End** memiliki ketergantungan terhadap **Database** karena memerlukan akses untuk melakukan operasi CRUD (Create, Read, Update, Delete) guna menyimpan, mengambil, memperbarui, dan menghapus data dalam sistem.

## 3) Database

Database merupakan tempat penyimpanan seluruh data yang digunakan dalam sistem. Dalam studi kasus ini, **Portal Web MyClean**

menggunakan komponen bernama **Database MyClean**, yang berfungsi sebagai tempat penyimpanan informasi mengenai layanan pembersihan, daftar pelanggan, daftar penyedia jasa, serta transaksi pemesanan.

### Hasil Study Case



### B. Studi Kasus 2 (Level Kompleks)

**Mithology Group** merupakan salah satu induk perusahaan konglomerasi di Indonesia yang tengah membuka rekrutmen besar-besaran untuk menjadi pegawai tetap di berbagai perusahaan yang berada dibawah Mithology Group, salah satu perusahaan yang membutuhkan banyak pegawai tetap adalah **PT. MAKHLUK ABADI** merupakan salah satu perusahaan yang bergerak di bidang Bioteknologi dan sedang memerlukan banyak pegawai tetap untuk berbagai posisi kosong yang ada di perusahaan. Namun, hingga saat ini, PT. MAKHLUK ABADI belum memiliki situs website resmi yang berfungsi

sebagai platform terpusat untuk mempromosikan dan mengelola lowongan pekerjaan. Oleh karena itu, Corporate Relations PT. MAKHLUK ABADI berinisiatif mencari penyedia layanan pengembangan aplikasi website yang berguna untuk membangun **Portal Karir Terintegrasi Mithology Group**. Selama proses pengembangannya, PT. MAKHLUK ABADI menyadari bahwa pembuatan website tersebut harus dilakukan secara terstruktur dengan kebutuhan yang jelas karena melibatkan berbagai perusahaan lainnya dibawah Mithology Group. Oleh karena itu, PT. MAKHLUK ABADI telah menetapkan beberapa kebutuhan utama yang harus dipenuhi dalam sistem portal karir ini. Portal karir yang akan dikembangkan harus dapat diakses oleh tiga kategori pengguna utama, yaitu End User (Pencari Kerja), HR Perusahaan (Seluruh HR dibawah naungan Mithology Group), dan Website Admin.

### 1. End User (Pencari Kerja)

- End User dapat mengakses Halaman Beranda tanpa perlu melakukan registrasi atau login.
- Halaman Beranda akan menyediakan berbagai fitur, antara lain:
  - o Cari Lowongan → Fasilitas pencarian dan filter lowongan pekerjaan.
  - o Pekerjaan Tersedia → Daftar lowongan aktif yang dapat dilihat oleh publik.
  - o Agenda Perusahaan → Informasi mengenai acara atau kegiatan rekrutmen.
  - o Tentang Kami → Profil PT. MAKHLUK ABADI dan perusahaan yang tergabung dalam Mithology Group.
  - o Registrasi & Login → Diperlukan bagi End User yang ingin mengakses fitur lanjutan.
- End User hanya dapat mengajukan lamaran pekerjaan setelah melakukan registrasi dan login.
- Setelah login, end user diwajibkan untuk melengkapi profil sebelum dapat melamar.

- Saat melamar pekerjaan, end user harus mengunggah CV dan Portofolio sebagai dokumen pendukung.
- Setelah pengajuan lamaran, sistem akan memberikan notifikasi konfirmasi bahwa dokumen telah berhasil dikirimkan dan sedang dalam proses peninjauan oleh HR perusahaan yang bersangkutan.

## 2. HR Perusahaan (Seluruh HR dibawah naungan Mithology Group)

- HR perusahaan memiliki akses untuk membuat, mengelola, dan memperbarui detail lowongan pekerjaan.
- Bertanggung jawab dalam meninjau dan menyaring dokumen pelamar untuk tahap seleksi berikutnya.
- Dapat memperbarui status kandidat (Diterima, Ditolak, atau Diproses Lebih Lanjut).
- Wajib melaporkan kepada Website Admin apabila suatu lowongan telah terisi, sehingga dapat diarsipkan atau dihapus sebagai bagian dari dokumentasi perusahaan.

## 3. Website Admin (Tim Pengelola Portal Karir Mithology Group)

- Bertanggung jawab atas pengelolaan, keamanan, dan pemeliharaan website.
- Berperan sebagai penghubung antara PT. MAKHLUK ABADI dan perusahaan-perusahaan didalam Mithology Group.
- Menindaklanjuti laporan dari HR perusahaan mengenai penutupan atau penghapusan lowongan yang telah terisi.

## 1. Deployment Diagram

1. **End User Device** : Representasi dari Perangkat yang digunakan oleh **Calon Pelamar**, **Pelamar** maupun **Guest Account**. Berfungsi sebagai antarmuka untuk menampilkan informasi dan hasil yang dibutuhkan oleh End User sekaligus untuk mengakses Portal Karir

Terintegrasi Mithology Group. End User dapat mengakses melalui **Web Browser**. Di Web Browser tersebut terdapat **URL** yang berfungsi mengakses halaman antarmuka website.

2. **HR Device** : Representasi dari Perangkat yang digunakan oleh seluruh Human Resource yang ada dibawah naungan Mythology Group. Berfungsi sebagai antarmuka untuk mengelola lowongan dan membuat laporan dari hasil berkas pelamar sekaligus untuk mengakses Portal Karir Terintegrasi Mithology Group. Human Resource dapat mengakses melalui **HR Browser**. Di HR Browser tersebut terdapat **URL** yang berfungsi mengakses halaman antarmuka website.
3. **Website Admin Device** : Merupakan perangkat yang digunakan oleh Website Admin untuk mengakses fitur administratif mengelola laporan pada Portal Karir Terintegrasi Mithology Group. Admin dapat mengakses melalui browser khusus Website Admin yaitu **Admin Broswer** yang didalamnya terdapat **URL** untuk menampilkan antarmuka web, lalu terdapat juga fitur khusus admin yaitu **Kelola Laporan** didalam perangkat yang digunakan oleh Website Admin ,dalam fitur Kelola Laporan terdapat data-data Laporan Lamaran yang terjadi didalam website Mythology Group berupa file **data\_laporan.log**.
4. **Web Server Mythology Group** : Merupakan server tempat Portal Karir Terintegrasi Mithology Group di host. Berfungsi untuk menyediakan layanan web kepada pengguna, **Web Server Mythology Group** terhubung kepada **End User Device**, **Website**

**Admin Device**, dan **HR Device** melalui protokol **HTTPS**. Didalam Web Server Mythology Group terdapat **Mythology Group Website** yang di host dan menggunakan teknologi **React.js / Vue.js** yang berfungsi untuk memastikan website dapat diakses dengan cepat dan responsif.

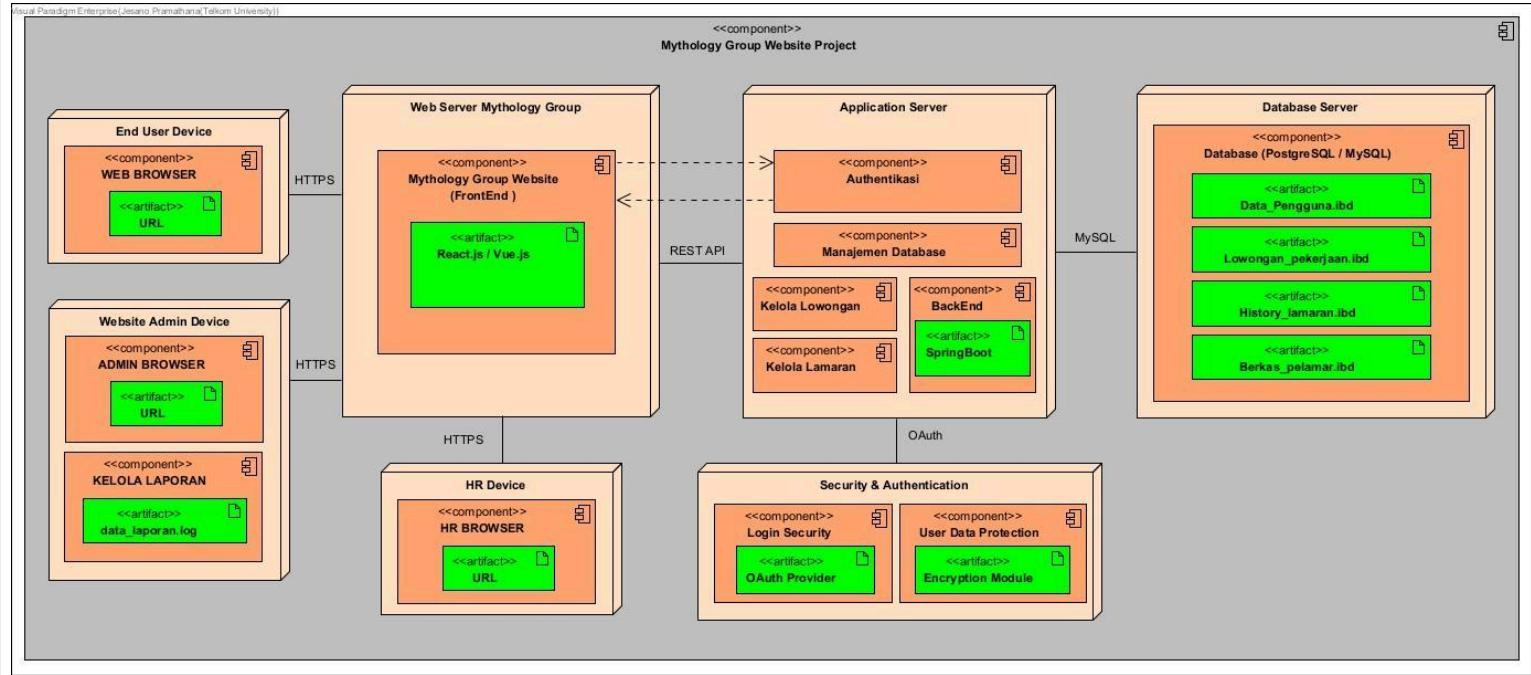
**5. Application Server** : Sebagai tempat untuk eksekusi aplikasi, memproses permintaan, dan menyediakan berbagai layanan yang diperlukan oleh aplikasi. Application Server dihubungkan dengan Web Server Mythology Group menggunakan protokol komunikasi **REST API**. Terdapat beberapa komponen yaitu **Authentikasi** yang bertanggung jawab untuk mengelola proses otentikasi pengguna, lalu selanjutnya ada komponen **Manajemen Database** yang mengelola seluruh data yang disimpan didalam database, lalu ada komponen **Kelola Iowongan** yang mengatur, menambahkan dan menghapus iowongan, setelah itu ada komponen **Kelola Lamaran** yang mengatur seluruh dokumen dari pelamar. Lalu terakhir ada **BackEnd** yang menggunakan teknologi **SpringBoot** sebagai database dan komunikasi antara frontend dan database. Komponen Authentikasi saling berketergantungan satu sama lain pada Mythology Group Website karena memiliki peran dalam proses otentikasi pengguna dan manajemen sesi pengguna.

**6. Database Server** : Database server menyimpan dan mengelola data dengan aman dan tersedia bagi aplikasi web untuk mengakses dan memperbarui data sesuai kebutuhan. Pada database server web mythology group membutuhkan Database

berbasis **MySQL** yaitu **Database(PostgreSQL/MySQL)**, yang didalamnya terdapat file data ; **Data\_Pengguna.ibd**, **Lowongan\_Pekerjaan.ibd**, **History\_Lamaran.ibd**, **Berkas\_Pelamar.ibd**.

**7. Security & Authentication** : Security & Authentication bertanggung jawab untuk memverifikasi identitas pengguna dan melindungi data dan akses sistem untuk ketiga pengguna. Pada bagian Security terdapat **Login Security** yang menggunakan **Oauth Provider** sebagai pengelola autentikasi berbasis token dan juga terdapat **User Data Protection** yang menggunakan **Encryption Module** berguna untuk mengenkripsi data sensitif sebelum disimpan kedalam database.

## Berikut Hasil Studi Kasus Deployment Diagram (Level Kompleks) :



## 2. Component Diagram

Pada contoh studi kasus ini **Mythology Group Website Project** dapat diakses oleh **Pelamar, Human Resource, Website Admin**. Pada contoh studi kasus ini, Mythology Group Website Project menggunakan konsep **Front End, Back End, Database, dan Security & Authentication** dengan menggunakan perspektif flow fitur. Berikut merupakan penjelasan dan peran dari masing masing komponen yang digunakan:

### 1. Front End

- Teknologi : React.js / Vue.js
- Menyediakan tampilan portal bagi end user, HR perusahaan, dan Website Admin.

### 2. Back End

- Teknologi : Node.js (Express.js) / Spring Boot
- Bertanggung jawab atas autentikasi pengguna, manajemen lowongan, dan pengolahan lamaran pekerjaan.
- Mengelola database dan komunikasi antara frontend dan database, serta komponen untuk keamanan system

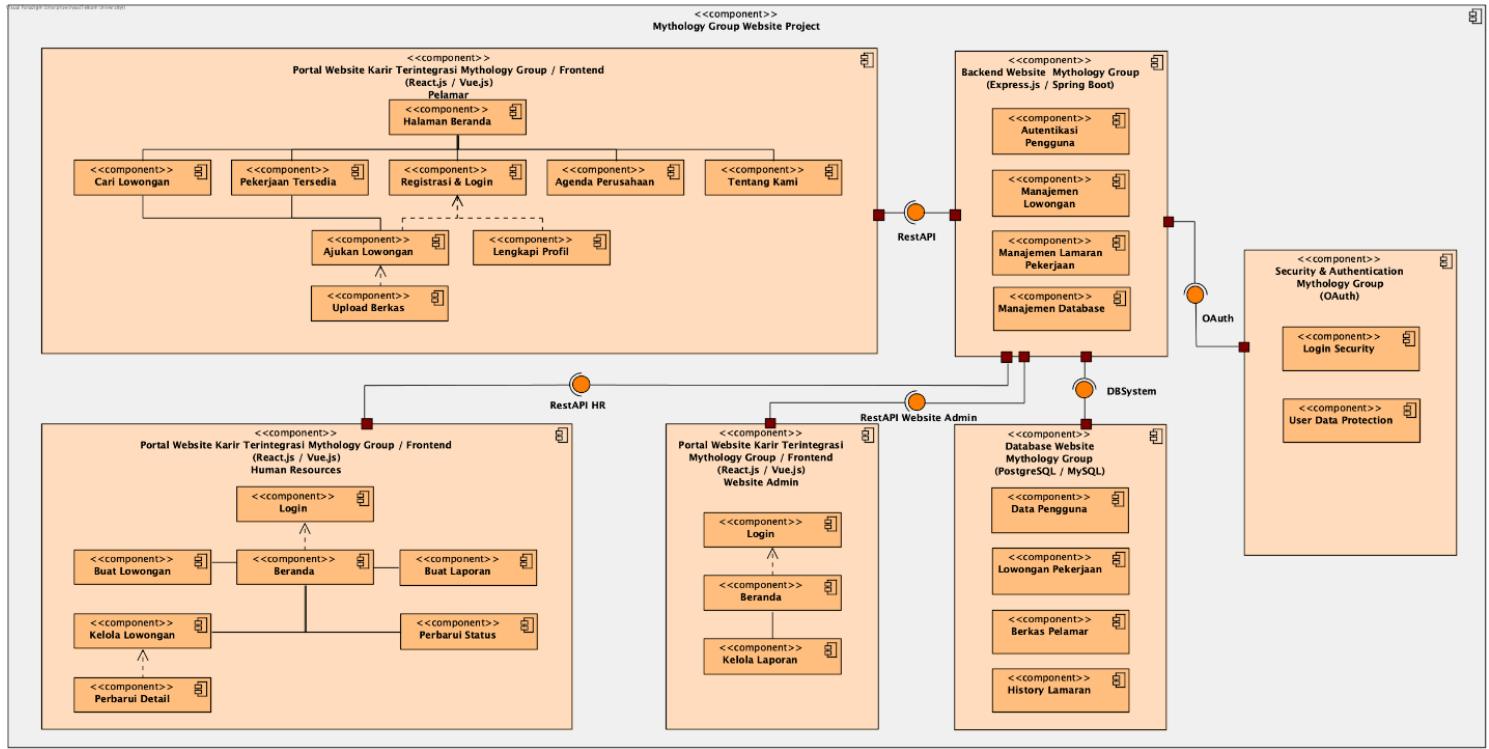
### 3. Database

- Teknologi: PostgreSQL / MySQL
- Menyimpan data pengguna, lowongan pekerjaan, CV dan dokumen pelamar, serta riwayat lamaran

### 4. Security & Authentication

- Teknologi: OAuth
- Digunakan untuk keamanan login dan proteksi data pengguna.

## Berikut Hasil Studi Kasus Component Diagram (Level Kompleks) :





“

## Sebagai Shinobi Konoha

Kita Harus Punya Sifat

**Pantang  
Menyerah!**

”

