**2. Pemilihan Arsitektur Aplikasi**

**2.1. Tujuan Pemilihan Arsitektur**

* Memberikan kemudahan dalam menganalisis pembagian tugas kode setiap komponen yang akan digunakan.
* Memudahkan dalam reusabilitas kode dengan memisahkan logika bisnis (model), Antarmuka pengguna (view) dan control alur aplikasi (controller)
* Memastikan perlindungan terhadap keamanan serangan SQL Injection dan Cross-Site Request Forgery (CSRF)

**2.2. Pilihan Arsitektur**

**2.2.1. Model-View-Controller (MVC)**

* **Deskripsi**: Arsitektur MVC membagi aplikasi menjadi tiga komponen utama: Model (mengelola data dan logika bisnis), View (menyajikan data kepada pengguna), dan Controller (menangani input dan memperbarui model).
* **Keuntungan**: Pemisahan yang jelas antara logika bisnis, tampilan, dan kontrol. Memudahkan pemeliharaan dan pengembangan fitur baru.
* **Kekurangan**: Bisa menjadi kompleks jika tidak dikelola dengan baik, terutama dalam aplikasi yang sangat besar.

**2.2.2. Microservices**

* **Deskripsi**: Pendekatan arsitektur di mana aplikasi dibagi menjadi serangkaian layanan kecil dan independen yang berkomunikasi melalui API.
* **Keuntungan**: Skalabilitas tinggi, memungkinkan pengembangan dan penyebaran terpisah, fleksibel dalam pemilihan teknologi.
* **Kekurangan**: Kompleksitas tinggi dalam hal pengelolaan layanan dan komunikasi antar layanan, memerlukan manajemen distribusi yang cermat.

**2.2.3. Single Page Application (SPA)**

* **Deskripsi**: Aplikasi web yang memuat satu halaman HTML dan mengupdate konten secara dinamis dengan JavaScript.
* **Keuntungan**: Pengalaman pengguna yang lebih mulus dan interaktif, pengurangan waktu muat halaman.
* **Kekurangan**: SEO bisa menjadi tantangan, lebih kompleks dalam hal pengelolaan status dan rute.

**2.3. Rekomendasi Arsitektur**

* **Rekomendasi**: Berdasarkan analisis kebutuhan, proyek ini direkomendasikan untuk menggunakan Model-View-Controller (MVC). Hal ini dikarenakan di dalam arsitektur MVC memberikan pembagian aplikasi yang jelas mulai dari tanggung jawab model, view dan controller. Pemilihan MVC ini membuat pengembangan yang lebih cepat misalnya saja pada pengembangan di bagian tim front-end dapat mengembangkan view sedangkan tim back-end mengerjakan model dan controller sehingga mempercepat proses pengembangan secara keseluruhan. Selanjutnya, memberikan kemudahan dalam pemeliharaan yang lebih mudah karena Ketika terdapat perubahan yang diperlukan, seperti adanya penambahan fitur baru atau perbaikan bug maka pengembang dapat dengan cepat menemukan dan melakukan modifikasi pada kode yang relevan dan memudahkan dalam memisahkan logika bisnis dari presentasi, komponen model dapat digunakan Kembali di berbagai bagian aplikasi atau bahkan di proyek tersebut.

**2. Desain Struktur Database Awal**

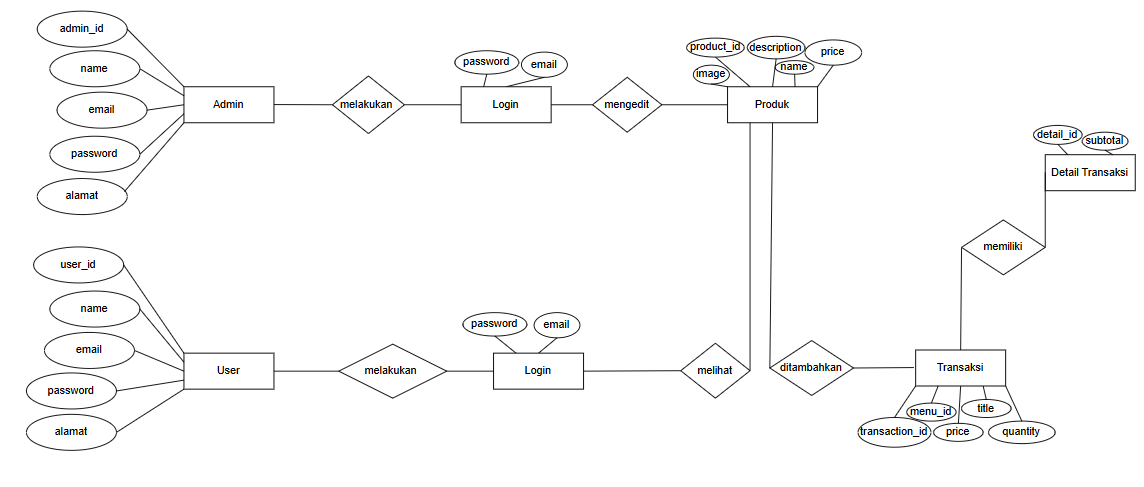
**2.1. Tujuan Desain Struktur Database**

* Memastikan penyimpanan data yang efisien dan terstruktur dengan baik.
* Mempermudah query data dan operasi pengelolaan data.

**2.2. Desain Struktur Database**

**2.2.1. Diagram Entity-Relationship (ER)**

* **Deskripsi**: Diagram ER menggambarkan entitas dalam sistem, atribut dan hubungan antar entitas.
* **Diagram**:



**2.2.2. Tabel-Tabel Utama**

* Tabel Admin
  + Kolom: admin\_id (PK), Username, Password, Email, Alamat,
* Tabel Pengguna
  + Kolom: user\_id (PK), Username, Password, Email, Alamat, created\_at, updated\_at
* Tabel Produk
  + Kolom: product\_id (PK), Name, Description, Price, Foto, created\_at, updated\_at
* Tabel Pesanan
  + Kolom: transaction\_id (PK), menu\_id, title, price, quantity
* Tabel Detail Pesanan
  + Kolom: product\_id (PK), Subtotal

**2.3. Rekomendasi Struktur Database**

* **Rekomendasi**: Struktur database yang diusulkan ini dirancang untuk memastikan integritas data dan efisiensi operasi. Tabel-tabel utama telah diidentifikasi dan hubungan antar tabel diatur untuk mendukung fungsionalitas aplikasi yang diinginkan.

**3. Pemilihan Teknologi Frontend dan Backend**

**3.1. Tujuan Pemilihan Teknologi**

* Memastikan teknologi yang dipilih dapat memenuhi kebutuhan fungsional dan teknis aplikasi.
* Mempertimbangkan aspek performa, skalabilitas, dan kemudahan pemeliharaan.

**3.2. Pemilihan Teknologi Frontend**

**3.2.1. Framework/Library**

* **Pilihan**: Laravel +blade
* **Deskripsi**: laravel adalah framework PHP sedangkan Blade adalah template yang disediakan laravel untuk mengatur bagian tampilan (frontend). Blade ini untuk menulis HTML dengan sintaksis yang bersih dan menyediakan control alur program
* **Kriteria Pemilihan**:
  1. Performa :blade memberikan performa yang baik dalam hal rendering tampilan. Blade akan mengkompilasi file template menjadi PHP yang dapat dieksekusi lebih cepat, mengurangi beban saat rendering tampilan
  2. Komunitas :laravel mempunyai komunitas yang sangat besar, sehingga Ketika menemui masalah kita dengan mudah dapat menemukan solusinya. Karena komunitas tersebut menghasilkan berbagai tutorial dan plugin yang dapat digunakan untuk mempercepat pegembangan aplikasi
  3. Dokumentasi :laravel menyediakan dokumentasi untuk blade secara rinci, seperti menjelaskan cara kerja sintaksis dan penggunaan control alur
  4. Kemudahan integrasi dengan Backend (laravel) :blade mudah digunakan dalam aplikasi berbasis MVC Dimana tampilan blade akan menghasilkan tampilan yang disediakan dari controller laravel.

**3.2.2. Teknologi Tambahan**

* **CSS Framework**: -
* **Tooling**: -

**3.3. Pemilihan Teknologi Backend**

**3.3.1. Bahasa Pemrograman dan Framework**

* **Pilihan**: PHP (laravel)
* **Deskripsi**: PHP adalah Bahasa pemrograman yang digunakan untuk membangun aplikasi web dinamis. Laravel adalah framework PHP yang digunakan untuk meningkatkan efisiensi pengembangan aplikasi web dengan menyediakan struktur yang jelas untuk bisa menangani tugas-tugas umum dalam pengembangan aplikasi backend, seperti routing, otentifikasi dan manajemen database. Laravel menggunakan arsitektur Model-View-Controller (MVC) yang memisahkan logika aplikasi dari tampilan dan data sehingga pada saat melakukkan pengembangan ini lebih terorganisir dan terstruktur.
* **Kriteria Pemilihan**:
  1. Performa :laravel memiliki performa yang baik dengan berbagai fitur optimasi seperti query builder dan eloquent ORM yang efisien dalam mengakses database.
  2. Komunitas :laravel memiliki banyak sumber daya untuk membantu pengembang menyelesaikan masalah termasuk video tutorial, forum, slack dan stack overflow
  3. Skalabilitas :laravel dirancang untuk skalabilitas dan dapat menangani aplikasi dengan jumlah pengguna yang besar karena sistem queue, job processing dan caching yang terintegrasi
  4. Kemudahan integrasi dengan database :laravel memiliki integrasi database yang fleksibel dengan berbagai opsi ORM, query builder dan migrasi database. Eloquent ORM memungkinkan pengembang untuk berinteraksi dengan database menggunakan objek PHP daripada menulis SQL secara langsung sehingga mengurangi kesalahan

**3.3.2. Basis Data**

* **Pilihan**: MySQL
* **Deskripsi**: MySQL adalah sistem manajemen basis data relasional (RDBMS) yang banyak digunakan untuk menyimpan dan mengelola data dalam aplikasi web dan perangkat lunak lainnya. MySQL menyediakan struktur table untuk menyimpan data yang terorganisir.
* **Kriteria Pemilihan**:
  1. Model data :MySQL menggunakan model data relasional, yang menyimpan data dalam bentuk tabel yang terstruktur dan menjelaskan hubungan antar tabel melalui foreign key dan join sehingga memberikan pengorganisasian data yang kompleks dengan banyak relasi antar entitas.
  2. Fitur :MySQL mendukung penyimpanan data dalam berbagai format, dengan tipe data seperti integer, varchar, date, dan boolean yang memungkinkan penyimpanan berbagai jenis data yang terorganisir.
  3. Dukungan transaksi : Jika terjadi kegagalan selama transaksi, MySQL dapat rollback perubahan untuk menjaga konsistensi data dan memastikan data tidak berada dalam kondisi tidak valid atau setengah selesai.

**3.4. Rekomendasi Teknologi**

* **Frontend**: laravel/blade karena
  1. Blade memungkinkan pengembang untuk menulis HTML dengan sintaks yang bersih dan mudah dibaca
  2. Laravel + blade Ketika terdapat masalah, terdapat banyak tutorial dan dokumentasi yang membahas cara terbaik menggunakan blade
  3. Dengan blade, pengelolaan layout dan tampilan menjadi lebih terstruktur dan mudah dipelihara. Penggunaan yield dan section memungkinkan pengembang untuk membuat struktur layout yang fleksibel dan dapat digunakan Kembali di seluruh aplikasi
  4. Dengan menggabungkan blade dengan fitur-fitur laravel lainnya seperti database migration, seeding dan task scheduling pengembang dapat membangun aplikasi yang dapat ditingkatkan dan dikelola dengan mudah
* **Backend**: **PHP**  karena
  1. PHP adalah bahasa yang sudah sangat dikenal dan banyak digunakan untuk pengembangan web. Framework Laravel, yang dibangun di atas PHP, menyediakan struktur yang sangat baik untuk membangun aplikasi yang efisien dan terstruktur.
  2. Laravel menawarkan berbagai fitur bawaan yang membuat pengembangan aplikasi menjadi lebih efisien, seperti task scheduling, queue management, event broadcasting, dan notifications.