Assignment 1:

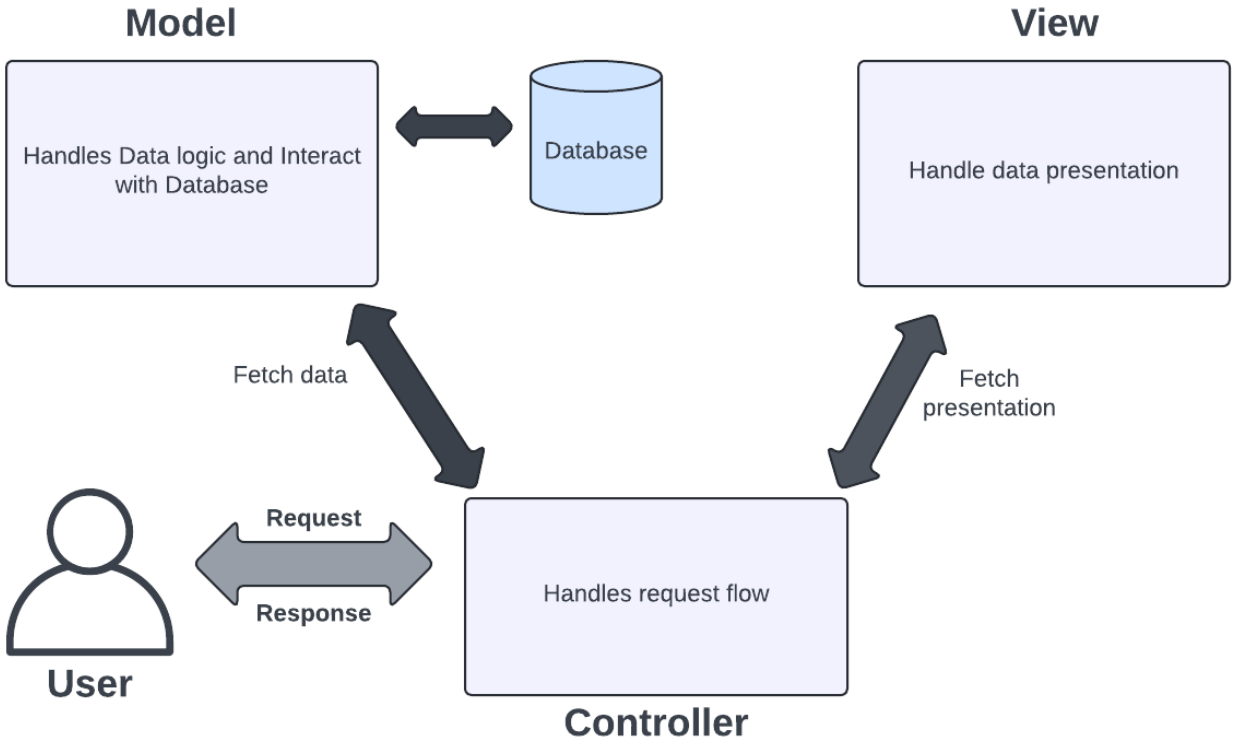
Design Pattern Explanation - Prepare a one-page summary explaining the MVC (Model-View-Controller) design pattern and its two variants. Use diagrams to illustrate their structures and briefly discuss when each variant might be more appropriate to use than the others.

MVC (Model-View-Controller):

MVC is an architectural design pattern that is separated into three logical components Model, View and Controller. It was traditionally used for desktop graphical user interfaces (GUI) and it is also used for mobile apps.

It supports large teams of web designers and developers and this will help for easy to maintain and can be extended easily.

MVC Architecture:



Controller:

- A controller is interconnected between the view and model.
- It tells the model what to do. It processes all the business logic and incoming requests.
- Manipulates data using model component and interacts with view to render the final output.

Model:

- Model represents the data and business logic of the application.
- It adds or retrieves data from the database.
- The model interacts with the database and gives the required data back to the controller.

View:

- The view component is used for all the UI logic of the application. It generates a user interface for the user.
- View which is created by the data which is collected by the model component but this data collected through a collector.

Variants:

Classic MVC

- When the model state changes, it notifies the view, which updates itself accordingly.
- The controller handles user input and updates the model based on the input received. It also updates the view if necessary.

Usage:

- Uses in small to medium-sized applications.
- Provides simplicity and implementation for simpler projects.

MVC with observer

- The controller remains responsible for handling user input and updating the model. It does not directly update the view.
- Update itself when notified of changes.

Usage:

- Recommended for large-scale applications with complex user interfaces.
- Beneficial when there’s a need for multiple views to observe the same model.

Assignment 2:

Principles in Practice - Draft a one-page scenario where you apply Microservices Architecture and Event-Driven Architecture to a hypothetical e-commerce platform. Outline how SOLID principles could enhance the design. Use bullet points to indicate how DRY and KISS principles can be observed in this context.

Microservice Architecture:

A microservice architecture is a type of application architecture where the application is developed as a collection of services.

Where do you apply Microservices Architecture to a hypothetical e-commerce platform

- Managing product information, including products, categories and attributes.
- Handles order processing, including order creation and status updates.
- Manages user authentication, authorisation and profile information.
- Provides personalized product recommendations based on user behaviour and preferences.

Event-Drive Architecture:

Event-Drive architecture is a software design pattern that enables an organization to detect “events” or important business moments and act on them in real-time or near real time.

Where do you apply Event-Drive Architecture to a hypothetical e-commerce platform

- Generated when a user successfully places an order. This event triggers downstream processes like inventory deduction, order confirmation mail, and payment processing.
- Indicates the outcome of a payment transaction. On successful payment, it triggers the fulfilment process on failure, it initiates a retry mechanism.
- Notifies the order management system and users about shipping status changes, such as shipped, out for delivery, or delivered.

Solid principles:

Single Responsibility Principle:

Each microservice has a single responsibility, ensuring it is focused and easy to maintain.

Open/Closed Principle:

Services are open for extension (adding new features) but closed for modification (existing functionalities remain intact).

Interface Segregation Principle:

Service interfaces are customized to the needs of their clients, avoiding unnecessary dependencies.

How DRY and KISS principles can be observed in this context

DRY (Don’t repeat yourself**):**

Shared components like authentication and database connections are centralized to avoid duplication and inconsistences across service.

KISS(Keep it simple stupid**):**

Service is designed to be simple and focused on there core functionalities, avoiding unnecessary complexities that could delay development and maintenance

Assignment 3:

Trends and Cloud Services Overview - Write a three-paragraph report covering: 1) the benefits of serverless architecture, 2) the concept of Progressive Web Apps (PWAs), and 3) the role of AI and Machine Learning in software architecture. Then, in one paragraph, describe the cloud computing service models (SaaS, PaaS, IaaS) and their use cases

Serverless architecture:

Serverless architecture is a type of cloud computing that enables development teams to get applications to market faster. Serverless means the server infrastructure is fully managed by a provider, so there's no need to manually track and manage application scale or cloud server configurations and provides benefits including reduced operating costs and always-on servers that support web and mobile applications. Servers are created on the fly only when required by the application. Instant availability creates additional scalability when compared to a cloud system.

Concept of Progressive Web Apps:

Progressive Web Apps are the advantages of web and mobile apps, utilizing web technologies like HTML, CSS, and JavaScript. They offer native app-like experiences across devices and platforms, including offline functionality and push notifications. These are responsive and installable directly from the browser, ensuring easy access and engagement for users. providing basic functionality on all browsers and enhanced features on modern ones, leading to faster, more reliable, and more engaging web experiences.

The role of AI and Machine Learning in software architecture

AI and machine learning are revolutionizing software architecture by enabling intelligent, data-driven decision-making and automation. These technologies are increasing integrated into various aspects of software development, including predictions, natural language processing, and recommendation systems. AI and machine learning algorithms can optimize performance, enhance security, and personalize user experiences. From automating tasks to identifying patterns in data, AI and machine learning play a crucial role in shaping the future of software architecture, empowering developers to create smarter and more efficient systems.

Cloud computing service models (SaaS, PaaS, IaaS) and their use cases

Software as a Service (SaaS) delivers applications over the internet on a subscription basis, ideal for end-user applications like email and collaboration tools.

Platform as a Service (PaaS) provides a platform for developers to build, deploy, and manage applications without worrying about the underlying infrastructure, suitable for application development and deployment.

Infrastructure as a Service (IaaS) offers virtualized computing resources over the internet, enabling users to provision and manage servers, storage, and networking infrastructure on-demand, suitable for businesses requiring more control over infrastructure and applications.