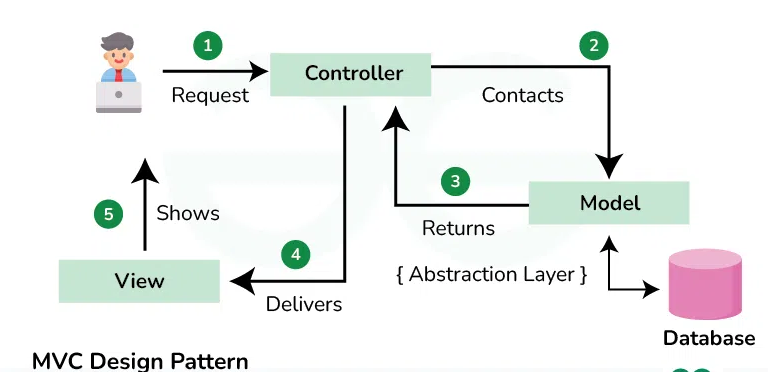
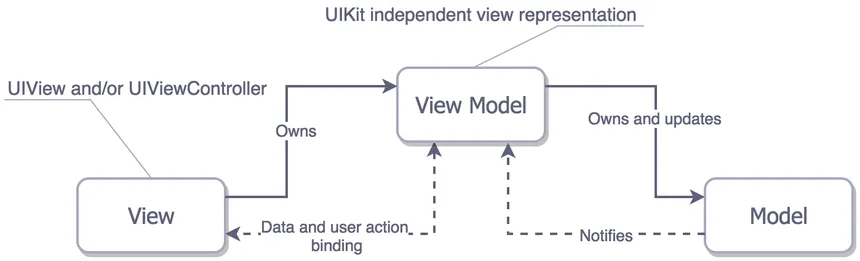
***Computer Architecture Day-2 Assignments Name: Ch.Vinay Kiran***

***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.***

* *Model-View-Controller: The Model-View-Controller is a popular software design pattern for creating user interfaces, which splits the associated computer functionality into three interconnected parts. These components include the internal information representations, the interface which displays and receive data sent by the user, and the controller software plays a vital role in the MVC (Model-View-Controller) as the controller is used to accept and read the users requests, and by the help of the controller the tasks or the functions are carried out by the Model and View components.*
* *The Model-View-Controller consists of three different components, each component as a unique functionality to perform. The three components are: Model, View and Controller.*

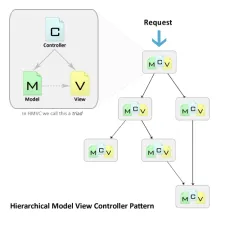
**

* *Controller: The Controller is the outer most component among the three components, as it is the only component were users can interact with, The controller acts as interface between model and view components to process all the user requests like access databases, manipulating the data, performing operations and delivery the raw data send by the model component into user understandable data by the help of view component.*
* *Model : All of the user's data-related logic is assigned to the Model component by the controller. This might be additional information relevant to firm logic, or it can represent the data being transmitted between the View and Controller components. All the logical tasks are performed by the model component. The model component is very important component in mvc in performing tasks.*
* *View: The component is used to convert the raw data from the model component. The controller transfers the raw data to the view component and the view component convert the raw data into user understandable data and delivery the data to the user.*
* *The variations of MVC: MVC is quite flexible and can be modified according to a variety of requirements. A common alternative is MVP (Model-View-Presenter), which regards the presenter as a go-between for the model and the view. The presenter manages user events and actions while updating the view with information from the model.*
* *The two variants of MVC are : MVVM (Model-View-model-View), HMVC (Hierarchical Model-View-Controller*
* *MVVM (Model-View-model-View) :*
* *The MVVM (Model-View-model-View) is a advance design pattern derived from the MVC design pattern, The MVVM (Model-View-model-View) is used to give our projects a efficient structure, usually means separation of consoles.*
* *In the MVVM (Model-View-model-View) it separates the user interface, data source and the business logic, so that we can easily understand, easily extend and easily test the projects. For the android applications development the MVVM (Model-View-model-View) is very popular and efficient.*
* *Some of the apps that uses MVVM (Model-View-model-View) are online shopping apps, Food recipe apps, video games apps and etc.*
* *In the MVVM (Model-View-model-View), the model is represented by a repository the stores all the data in one central place, data like data from databases or from remote APIs.*
* *The View-Model contains the business logic, as the business logic refers to our real program logic, that isn’t updating any eye elements or the data, like filtering the list or validating the user requests.*
* *The View-Model is in direct communication with the model component to access to the data requested by the user.*
* *The View in the MVVM (Model-View-model-View) contains all the eye elements and updates the elements and data by the help of view-model. The important thing in the view component is that it shouldn't contain any business logic but only the logic to manipulate the eye elements.*

**

*HMVC (Hierarchical Model-View-Controller):*

* *Hierarchical Model-View-Controller (HMVC) is a form of architecture design used in web development to arrange code in a hierarchical framework, improving code reuse and maintainability.*
* *The Hierarchical Model-View-Controller is derived from the MVC design pattern, In the MVC design pattern there are 3 components: Model, View, Controller. Whereas the application that is running in HMVC is set up as a hierarchy of parts. Usually, a module, page, or functional part of the program is represented by each component. These elements may have separate MVC triads*.

**

* *A hierarchical structure is introduced by HMVC in contrast to typical MVC, in which controllers interact directly with models and views. To seek information or carry out operations, controllers in one module can call upon controllers in another module. Better code organization and code reuse are made possible by this.*
* *So, In the projects of android app development the MVVM is very efficient, as the MVVM is designed to build design pattern in development of the android applications, Which contains view-model, view and model, so for simple projects in android applications MVVM is used.*
* *The real-world examples of the application that uses the MVVM are in food recipes, online-shopping etc.*
* *Whereas, the HMVC is used in the projects like web development and service oriented projects, as by the help of HMVC we can reuse the components in different parts of the application.*
* *Both in MVVM and HMVC scalability is achieved as both design pattern can extend.*
* *In the HMVC each module can be developed and maintained independently, promoting team collaboration and code organization.*
* *So for large web application development the HMVC is the efficient design pattern, Whereas for the android application development the MVVM is the efficient design pattern.*

***Assignment 2: Principles in Practice - Draft a one-page scenario where you apply Micro services 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.***

* *Micro-Services Architecture: A software application that is developed as a group of loosely coupled, independently deploy-able services is known as a micro service architecture. Every service may be designed, implemented, and expanded independently and is centered around a particular business feature.*
* *Micro services architecture enables large teams and organizations to build scalable applications that are composed of many loosely coupled services.*
* *As micro services are loosely coupled each service handles a dedicated function inside a large-scale applications. For example, shopping cart, billing, user profile, push notification can all be individual micro-services. And these functional areas are called domains.*
* *Micro-Services communicate with each other via well-defined interfaces with small surface areas. And the small surface areas limit the blast radius of failures and defects. It makes each service easier, simple to reason about the context of the entire application.*
* *Micro services communicate to one and another over a combination of remote procedure calls (RPC), event streaming or message passing.*
* *Well architect-ed micro-services practice strong information hiding. This often means breaking up a monolithic database into its logical components and keeping each logical component well hidden inside its corresponding micro services.*
* *Event-Driven Architecture: A software architectural style called event-driven architecture (EDA) enables the creation, detection, consumption, and response to events that take place within or across systems.*
* *In the event -driven architecture services generate events, like user registration, product updates, or order placement, which are asynchronously consumed by other services. Asynchronous processing and decoupling are made possible by the use of events in service communication.*
* *Scalability and Flexibility: By enabling services to respond in real-time to events, event-driven architecture promotes scalability and flexibility.*
* *Events offer a full history of changes and act as a source of truth for the current state of the system.*
* *SOLID Principles:*
* *The Single Responsibility Principle (SRP) states that in order to make a micro-services cohesive and easy to maintain, it should have a single responsibility, such as order processing or user administration.*
* *The Open/Closed Principle (OCP): The Open/Closed Principle (OCP) states that services must be closed to modification but open to extension, enabling the addition of new features without changing the code that already exists.*
* *Liskov Substitution Principle (LSP): Services ought to be created so that they can be swapped out for one another without compromising the system's correctness.*
* *The Interface Segregation Principle (ISP) : The Interface Segregation Principle (ISP) states that in order to keep customers from becoming dependent on services they don't utilize, services should present particular interfaces catered to their needs.*
* *Dependency Inversion Principle (DIP): To encourage loose coupling, high-level modules should rely on abstractions rather than low-level modules.*
* *The DRY and KISS Principles:*
* *Don't Repeat Yourself (DRY): By removing common functionality into reusable components or libraries, you can prevent writing duplicate code.*
* *To save redundancy and preserve consistency, use common data models or schemes across services.*
* *KISS stands for "Keep It Simple, Stupid," which advocates for designing services with an emphasis on solving particular business problems without needless complexity.*
* *Select simple solutions and designs that satisfy the platform's needs without adding needless complexity to avoid over-engineering.*
* *The e-commerce platform can improve its scalability, maintainability, and robustness while maintaining a clean, consistent design by following the SOLID Principles, DRY and KISS Principles.*

***Assignment 3: Trends and Cloud Services Overview - Write a three-paragraph report covering: 1) the benefits of server-less 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?***

* *Server-less Architecture: Server-less Architecture is a cloud computing model where cloud providers manage the infrastructure and dynamically allocate computing resources as needed. Without having to worry about managing servers, operating systems, or supporting infrastructure, developers may concentrate on writing code, usually functions, in a server-less architecture.*
* *Server-less architecture is often associated with Function as a Service (FaaS) offerings, in which programmers create functions that carry out particular activities or react to events, are frequently linked to server-less architecture. These are transient, independent functions that run in reaction to events or triggers*.
* ***The benefits of server-less architecture:***
* *Reduced Operational Overhead: In server-less computing, cloud providers handle all aspects of the underlying infrastructure, such as scaling, maintenance, and server provisioning. Developers may now concentrate on building code and providing value to clients instead of worrying about server management duties.*
* *Cost-effectiveness: Server-less computing is billed on a per-use basis; you only pay for the resources used by your processes while they are running. Neither the initial nor recurring expenses for unused resources exist. As a result of not having to pay for resources when they're not being used, this can save money, particularly for applications with irregular or erratic usage patterns.*
* *Flexibility and Agility: Server-less systems allow for quicker cycles of development and implementation. Iterating fast, releasing updates without a pause, and writing and deploying functions independently are all possible for developers. Time-to-market is accelerated and continuous delivery processes are supported by this flexibility and agility.*
* *High Availability and Scalability: Server-less architectures are designed to be highly available and scalable by nature. In order to guarantee that your application can handle a range of workloads and maintain uptime even in the case of failures, cloud providers manage the scaling and distribution of resources across numerous data centers.*
* *Reduced Complexity: By removing the need for infrastructure administration, server-less architectures streamline the development and use of apps. Rather than worrying about server provisioning, scaling, or maintenance, developers can concentrate on building business logic, which results in shorter development cycles and less complexity.*
* ***The concept of Progressive Web Apps (PWAs)***
* *Progressive Web Apps (PWAs) are web applications that give consumers a native app-like experience through their web browsers by leveraging contemporary web capabilities.*
* *Because PWAs are built according to the principles of progressive enhancement, they are intended to function on any device and browser, irrespective of the user's network conditions. They gradually improve according to the user's device and browser's capabilities.*
* *The adaptable nature of PWAs allows them to adjust to different screen sizes and orientations, providing a consistent user experience on computers, smart phones, and tablets.*
* *PWAs provide features like push notifications, offline capability, home screen installation, and full-screen mode in an effort to give consumers a native app-like experience. As a result, users can engage with the online application in a manner that is akin to that of native mobile apps and feels comfortable and natural.*
* *PWAs use web app manifest files, which are JSON files including application meta-data like name, icons, colors, and desired display mode, are used by PWAs. Because of this, users may install the PWA to their home screen and open it just like a regular app.*
* ***The role of AI and Machine Learning in software architecture***
* *The design, development, and operation of software systems are all impacted by artificial intelligence (AI), which is a major component of contemporary software architecture.*
* *During the design process, software architects can benefit from the guidance provided by AI algorithms. AI-based technologies, for instance, may recognize design patterns, evaluate requirements, and recommend the best architectural solutions based on best practices and historical data.*
* *Artificial Intelligence (AI) methods like neural networks, reinforcement learning, and genetic algorithms can automate software architecture design and optimization. These algorithms are capable of searching through large design areas, finding the best configurations, and optimizing architectural choices in accordance with predetermined goals and limitations.*
* *Artificial intelligence (AI)-driven methods for performance optimization can examine how software systems behave, spot performance snags, and tweak system settings to maximize efficiency and minimize resource usage. To optimize speed and scalability, AI-based algorithms, for instance, might dynamically modify system parameters, distribute resources, or schedule jobs.*
* *AI-driven autonomous systems possess the ability to dynamically modify and progress their structure and conduct in response to shifting external circumstances, user inclinations, and operational objectives. Without human assistance, these systems are able to self-optimize, self-heal, and self-configure in order to accomplish desired goals.*
* *Software architects can use natural language interfaces (NLIs) to communicate with architectural documentation, design artifacts, and domain-specific knowledge. NLP-based solutions can produce architectural diagrams, draw conclusions from textual sources, and help stakeholders communicate and work together.*
* *AI methods that examine past data, user interactions, and system telemetry, such machine learning and data mining, can forecast future trends, spot possible problems, and proactively optimize software structures. Architects can benefit from predictive analytic by using it to forecast resource demands, anticipate scalability requirements, and reduce risks before they arise.*
* ***The cloud computing service models (SaaS, PaaS, IaaS)***
* *Software as a Service (SaaS): SaaS delivers software applications over the internet on a subscription basis, eliminating the need for users to install, maintain, and update software locally. The service provider hosts and manages the applications and takes care of all upkeep and support. Emails, document collaboration, project management, customer relationship management (CRM), and other SaaS apps are frequently used to improve teamwork and productivity.*
* ***Platform as a Service:*** *PaaS offers an all-inclusive cloud-based development and deployment environment that includes middleware, application run-time, development tools, and services. Without having to worry about the underlying infrastructure administration, users can concentrate on developing and implementing applications. Web and mobile application development, deployment, and management are ideally suited for PaaS systems. To speed up the creation and delivery of applications, developers can take advantage of built-in frameworks, services, and development tools.*
* ***Infrastructure as a Service:*** *IaaS enables users to rent infrastructure elements like virtual machines, storage, and networking on a pay-per-use basis by supplying visualized computing resources over the internet. Users have complete control over the middleware, applications, and operating system that are installed on the infrastructure.*
* *In order to rapidly provide and configure virtual environments for developing, testing, and deploying applications without the need for real hardware, development and testing teams frequently use infrastructure as a service (IaaS).*

.