Onakoya Korede
Posted on Jun 27, 2023

❤️ 8

Back To Basics: Tiers in Software Architecture

#systemdesign

The architecture upon which a business constructs its application has a substantial impact on its success or failure in software engineering. This seems obvious until you have to be the person making the choice for a million-dollar project from the beginning.

In this issue, we'll go back to the fundamentals to explain what software architecture is, what software architecture tiers are, how many there are, and when we utilize each of them.

A software architecture is a system that represents the collection of components that perform a certain function or set of operations. An architecture's foundation may have one or more components upon which software can be developed. A software architecture assists in defining and representing the component(s) and their connections.

Patent Owner, CleveMed - Ex. 2022, p. 1

We can understand software architecture with an example. Consider that in order to build a duplex on undeveloped property, we will need an architect to design a home plan. He collects various forms of information from us and creates a floor plan on paper. If all of the prerequisites are completed, the civil engineer can begin building. Similarly, a software architect creates a plan based on the requirements. He creates a high-level framework that fits all of the software's technical and operational needs. The developer team creates software in accordance with the software architecture and delivers it to the client.

There are three fundamental layers that are important steps to having a grasp of software architecture tiers. They are the presentation layer, the application layer, and the data layer.

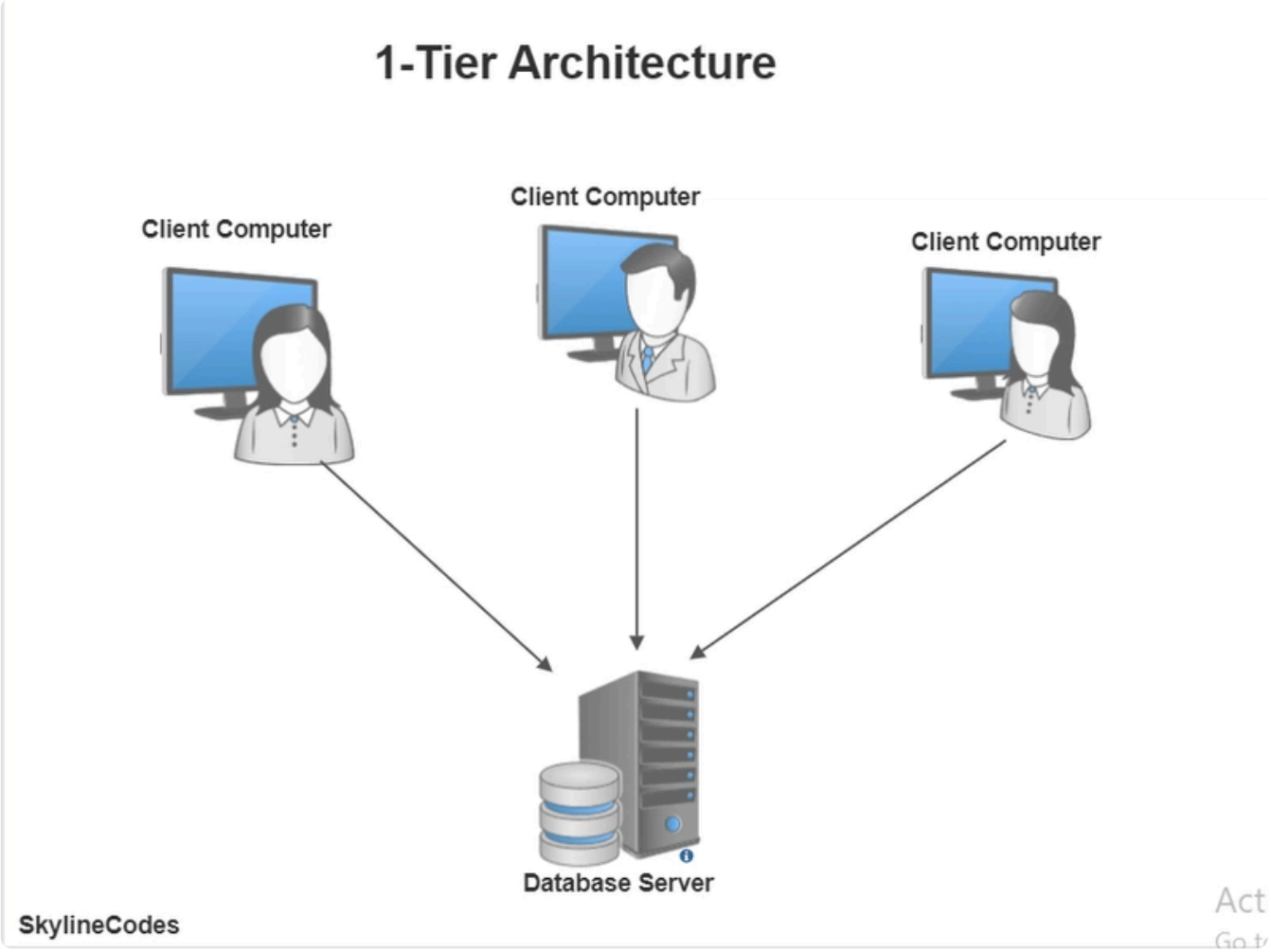
The Presentation Layer, also known as the User Interface, passes the information that is given by the user in terms of keyboard actions and mouse clicks to the Application Layer. The main function of this layer is basically to communicate with the next layer below—the Application Layer. For example, on the login page of an application, where the user may interact with the inputs and buttons, enter their credentials such as user ID, email, and password, and then click the sign-in button, The Presentation Layer manages and categorizes the whole process.

The Application Layer, often known as the business Layer, comprises all of the models and logic that allow the application to run. It also serves as a bridge between the user interface and the database layer. Consider the sign-in instance from previously. When a user hits the sign-in button, the Application Layer communicates with the Data Layer and transmits the necessary information to the Presentation Layer.

The Data Layer interfaces with the application's database to retrieve data and deliver it back to the Application Layer. It's where the application's data is kept. It includes methods for connecting the database to the application and performing database operations like insert, UPDATE, DELETE, and so on. It saves all data passed to it after the Application Layer has manipulated it.

Tiers Architecture

Patent Owner, CleveMed - Ex. 2022, p. 2



One-Tier Architecture

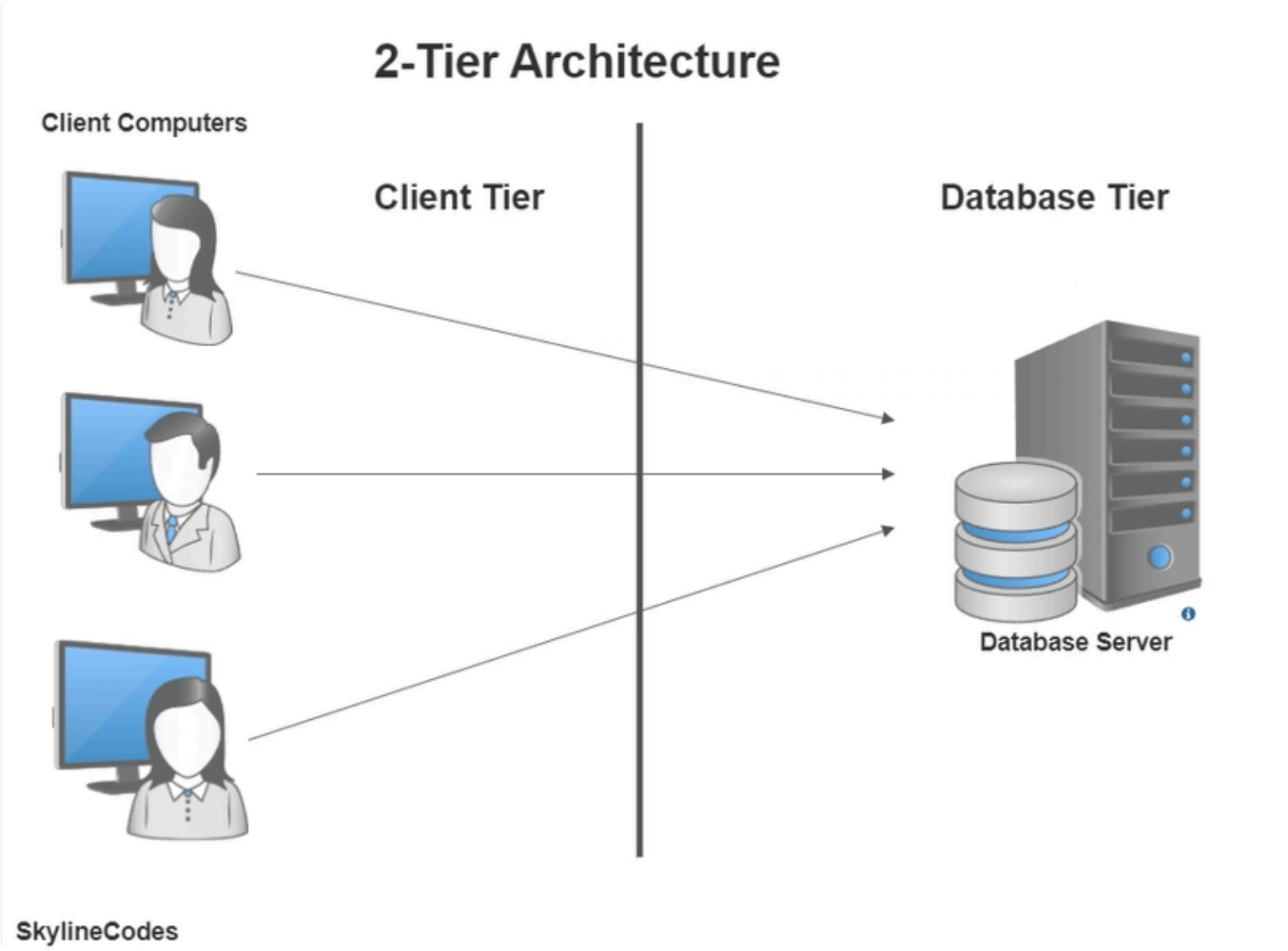
A single-tier application is one in which the backend logic, database, and user interface are all located on the same server. It entails hosting all of the essential components of a software application or technology on a single server or platform, such as the interface, middleware, and back-end data. Some also refer to it as any program that can be installed and executed on a single computer.

There are no network latency concerns because the application components all reside on the same single server. The data is readily available, and there is no need to request it from a database server. Furthermore, unlike a three-tier or multi-tier architecture, the attraction of a single architecture might be related to the expenses involved, where it may make more sense to have simpler applications confined to one basic platform.

However, unlike more advanced systems, it cannot transmit information from one client machine to another.

Patent Owner, CleveMed - Ex. 2022, p. 3

Two-Tier Architecture



The client is on the first tier of a two-tier architecture, while the database server is on the same server machine, which is on the second layer. This second layer provides the data and conducts the web application's business logic.

The user system interface is often hosted in the user's desktop environment in two-tier client/server systems, while the database management services are typically located on a server that is a more powerful computer that handles numerous clients.

To put things into perspective, in a two-tier design, the display layer and application layer run on a client, while the data layer is stored on a server. Organizations that use this design often desire to integrate their application and database server capabilities on a single layer. Languages such as C, C++, Java, Python, PHP, Rails, and others can be used to build the Client-Application layer.

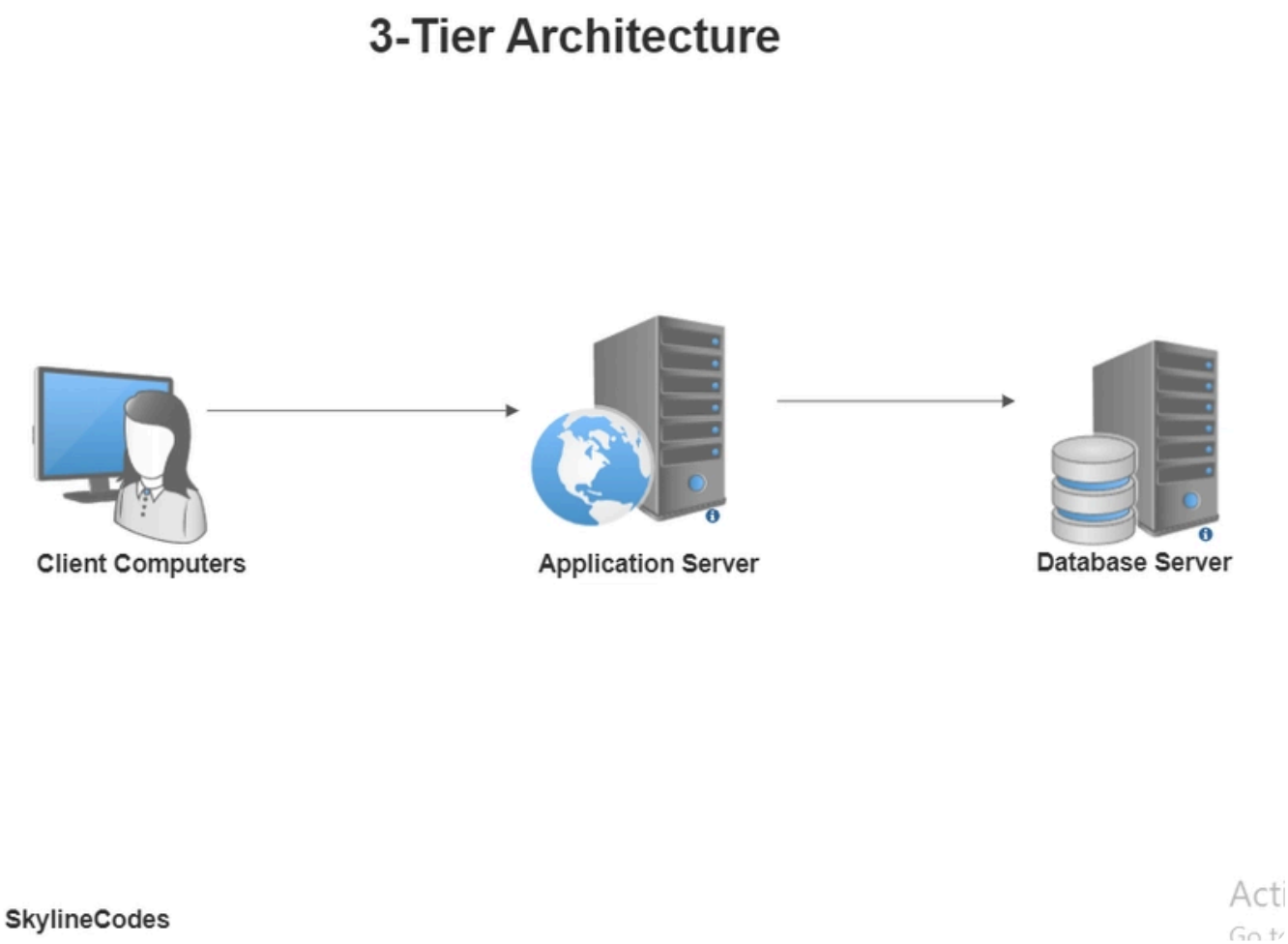
Patent Owner, CleveMed - Ex. 2022, p. 4

Data storage (a database or file system) and techniques for storing and retrieving data from storage comprise the data management layer. Databases that are often used include MySQL, MongoDB, PostgreSQL, and SQLITE. Desktop programs, games, and music players are examples of two-tiered apps.

The advantages of using two-tier apps include their speed and ease of implementation, as well as faster communication between the client and server. It's also best suited for environments with unchanging business rules or logic.

One of its drawbacks is that it is not easily scalable, so performance diminishes as the number of users increases. Furthermore, a data integrity issue might easily emerge as a result of the server responding to several requests at the same time.

Three-Tier Architecture



The three-tier architecture is a well-established software application architecture that organizes applications into three logical and physical computing tiers: the presentation,

Patent Owner, CleveMed - Ex. 2022, p. 5

or user interface; the application layer, where data is processed; and the data layer, where the data associated with the application is stored and managed.

The three-tier architecture offers a major advantage in that each tier can be developed and deployed independently on its own infrastructure. This means that separate teams can update or scale each tier as necessary without affecting the other tiers. Additionally, each tier can run on a dedicated operating system and server platform, whether it be a web server, an application server, or a database server, to best meet its functional needs.

And each tier runs on at least one dedicated server, whether hardware or virtual, so the services of each tier can be customized and optimized without impacting the other tiers.

Another benefit is faster development. By allowing different teams to develop each tier of the application simultaneously, an organization can bring it to market more quickly. Additionally, programmers can use the latest and best languages and tools for each tier, improving overall development efficiency.

Other benefits include improved reliability of the services. An outage in one tier is less likely to impact the availability or performance of the other tiers. Security is also important because the presentation tier and data tier can't communicate directly; a well-designed application would function as a sort of internal firewall against its services, preventing SQL injections and other malicious exploits.

How Does The Three-Tier Relate To Web Development?

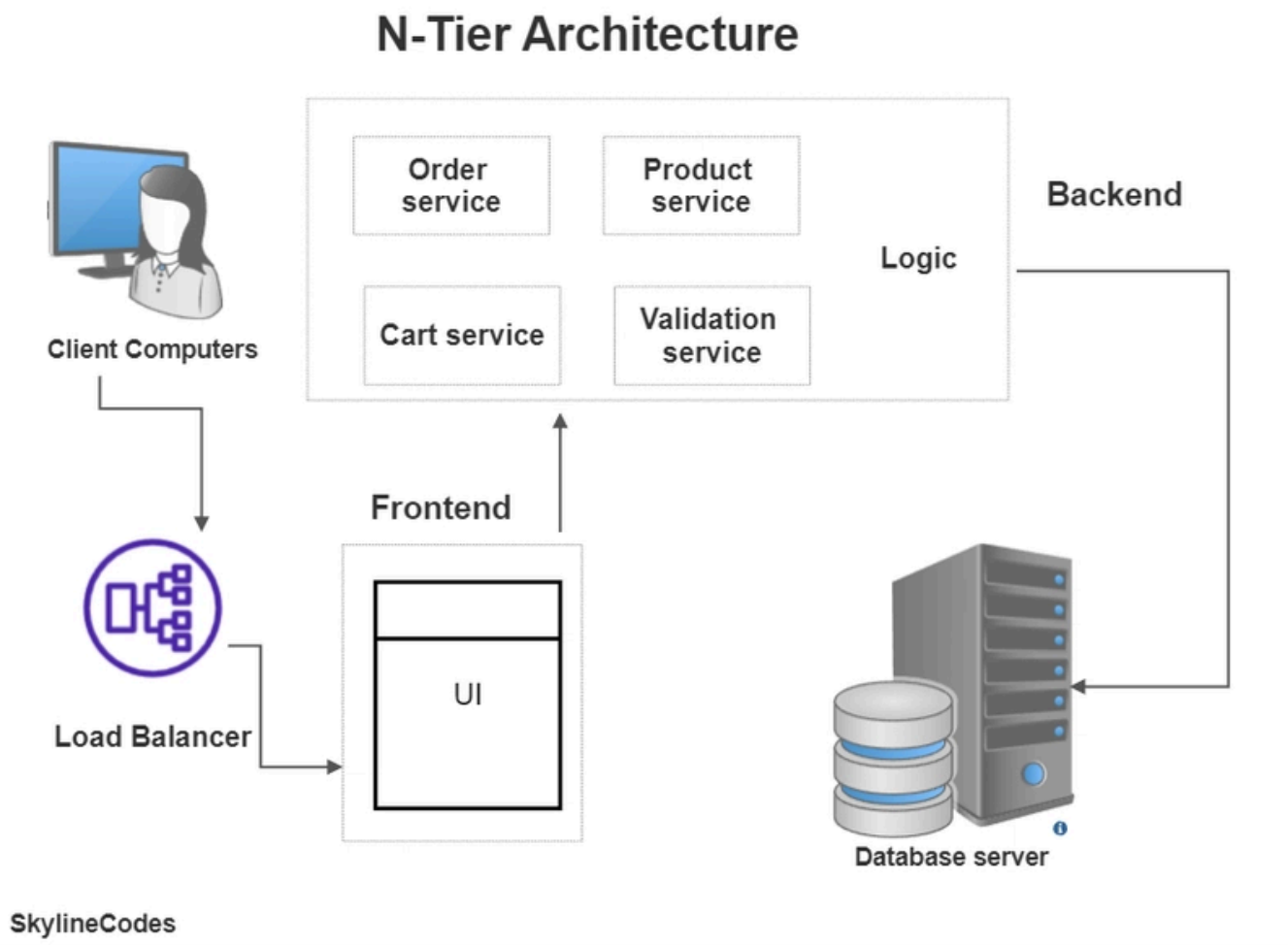
The web server is the presentation tier and provides the user interface. This is usually a web page or website, such as an e-commerce application, where the user adds products to the shopping cart, adds payment details, or creates an account. The content of this presentation tier could either be static or dynamic and is usually developed using HTML, CSS, and JavaScript.

Next, we move to the application server, which corresponds to the middle tier of the three-tier. It houses the business logic used to process user inputs. To continue the e-commerce example, this is the tier that queries the inventory database to return product availability or adds details to a customer's profile. This layer is often developed using Python, Ruby, or PHP and also runs on frameworks like Django, Rails, Laravel, or ASP.NET.

Patent Owner, CleveMed - Ex. 2022, p. 6

The database server is the data or backend tier of a web application. It runs on database management software such as MySQL, Oracle, and PostgreSQL.

N-Tier Architecture



An N-tier application is one that is distributed over three or more machines in a distributed network. The n-tier is a three-tier composition comprising display, application, and database that may be further broken into multiple sub-layers depending on the requirements.

Furthermore, N-tier architecture is a popular choice for developing enterprise-level applications that can handle large volumes of data and traffic. By breaking down the application into smaller, independent components, it becomes easier to manage and scale the system as needed.

This architecture paradigm enables software developers to construct reusable programs that are as flexible as possible. The n-tier architecture is used in applications such as the

Patent Owner, CleveMed - Ex. 2022, p. 7

Salesforce enterprise application, Amazon, Facebook, and Twitter.

N-tier architectures are also typically implemented as Infrastructure-as-a-service (IaaS) applications, with each tier running on a separate set of VMs.

N-tier is especially beneficial because it has a reduced learning curve for most developers. That is, developers may be assigned to certain microservices without having to learn about the entire application architecture.

On the other hand, it's common to end up with a middle tier that only performs basic CRUD operations on the database. While this may seem efficient at first glance, it can actually add extra latency without doing any significant work.

One solution to this problem is to introduce a caching layer between the application and the database. By caching frequently accessed data, the middle tier can quickly retrieve and serve this data to the user without needing to query the database each time.

Another option is to implement more advanced logic in the middle tier, such as data processing or analysis, which can provide additional value to the end user. By expanding the functionality of the middle tier, developers can ensure that it is doing meaningful work and contributing to the overall performance of the system.

N-tier architecture, also known as multi-tier architecture, separates application processing, data management, and presentation functions both physically and logically. This means that these functions are hosted on several machines or clusters, ensuring that services are provided without shared resources and are delivered at peak capacity.

Key Considerations For Using N-Tier Architecture

For n-tier architecture, it is imperative to have fast and responsive network bandwidth and hardware. Careful consideration of this factor is necessary to ensure that services are decoupled and communication between services does not suffer from noticeable lag. Failing to provide sufficient network bandwidth can significantly impact the performance of the architecture.

Furthermore, it is advisable to use as few tiers as possible in your software application. This is because adding more tiers can increase complexity, require additional server purchases, and result in higher maintenance and deployment expenses.

Patent Owner, CleveMed - Ex. 2022, p. 8

Finally, it is important to note that there is a minimum number of tiers required to ensure that your application is scalable, secure, and able to take full advantage of the benefits that this architecture provides.

In order to achieve the necessary scalability, for example, it may be necessary to add additional tiers to the architecture, such as an application server or a load balancer (read our last issue on load balancers here). Similarly, in order to ensure the security of each service, additional security measures may need to be implemented at each tier.

Finally, it is worth noting that using this architecture has many other benefits, such as improved performance, greater flexibility, and increased reliability. You can fully realize these benefits by carefully selecting the appropriate number and type of tiers for your application.

In the upcoming issue, we will provide an expansive overview of different types of software architectures. We will delve into Client-server architecture, which is used to separate clients and servers from each other and support efficient communication between them. We will also discuss peer-to-peer architecture, which is used to allow each node to act as both a client and a server. Additionally, we will cover event-driven architecture, which is used to respond to user events in real-time. Finally, we will explore micro-service architecture, which is used to develop software by breaking it down into modular components that can be independently developed and tested. By the end of the issue, you will have a comprehensive understanding of different software architectures and their applications.

I hope you found this week's issue useful! Our team puts a lot of effort into creating valuable content for our readers, and we're always looking for ways to improve.

If you enjoyed this issue, I invite you to subscribe to our newsletter here. We share this kind of content every week, covering a variety of topics related to software engineering, system design, and development. And if you know anyone who might find our newsletter helpful, feel free to invite them to subscribe too!

We appreciate your support and look forward to sharing more insights and knowledge with you in future issues.

Cheers.

Happy learning!

Patent Owner, CleveMed - Ex. 2022, p. 9

H Company PROMOTED

Check out this Runner H "AI Agent Prompting" Challenge Winner 👀

From culinary assistants to sports analysis tools to hackathon discovery agents, our submissions were full of diverse use cases!

Runner H Viral Content Factory Agent

Nikoloz Turazashvili (@axrisi) · Jun 6

#devchallenge #runnerhchallenge #ai #machinelearning

Top comments (1)

Charli • Apr 11 '24 • Edited

Diving into "Back To Basics: Tiers in Software Architecture" offers an invaluable journey into the core principles that shape and define robust, scalable software systems. This topic enlightens us on the significance of structuring software in tiers, ensuring each layer efficiently handles its designated functionality for enhanced performance and maintainability. For those looking to further enrich their understanding and practical skills in software architecture, especially in design tools like AutoCAD, visiting acad.prosoftstore.com/autocad.html can be a game-changer. This resource not only complements theoretical knowledge but also offers hands-on experience with industry-standard software, propelling your expertise to new heights.

Code of Conduct • Report abuse

LaunchDarkly PROMOTED

Called MCP tool create-feature-flag ✓

Parameters:

```
{
  "request": {
    "projectKey": "raccoon-api-upgrade",
    "FeatureFlagBody": {
      "name": "Awesome New Feature",
      "key": "awesome-new-feature",
      "description": "Gates access to my awesom
      "isFlagOn": false
    }
```

Create a feature flag in your IDE in 5 minutes with LaunchDarkly's MCP server 🏁

How to create, evaluate, and modify flags from within your IDE or AI client using natural language with LaunchDarkly's new MCP server. Follow along with this tutorial for step by step instructions.

Read full post

Onakoya Korede

LOCATION
Lagos, Nigeria

JOINED
Jul 16, 2020

Patent Owner, CleveMed - Ex. 2022, p. 11

More from Onakoya Korede

5 Software Patterns You Should Know As a Software Engineer - Part 1
#systemdesign

Load Balancers: Do We Actually Need Them?
#systemdesign

H Company PROMOTED

Check out what they built with Runner H "AI Agent Prompting" 👀

From culinary assistants to sports analysis tools to hackathon discovery agents, our submissions were full of diverse use cases!

Read more →

Patent Owner, CleveMed - Ex. 2022, p. 12