Group 3: Jahlil Owens, Trishelle Leal, and Landon Strappazon

Milestone 2: Systems architect

Trishelle Leal Uribe

Nutrition and Fitness Tracking

For our application, we are considering the use of an n-tier client/server architecture. This is due to the flexibility this architecture model provides, while giving the opportunity to easily maintain the system in the future. The n-tier architectural model allows for a flexible number of tiers, all of which work independently and are made suited for the application. For the nutrition tracking application, we have chosen to have a total of three tiers: Presentation, Application and Data and Persistence. This can provide us with a structure that can make it easy to maintain the application without any further reworking if it grows.

The main advantage of the three-tier architecture is the flexibility it gives the project. The structure allows for changes in the individual layers, without having to affect the other layers. Each layer's independence allows for a separation of concerns that ensures modularity and lowers the complexity of the layers. Additionally, the independence amongst the layers allows every layer to be scaled independently, to its needs. Lastly, this independence among the tiers allows for sensitive data to be isolated within the data tier, which can reduce the chances of unauthorized access. Of course, this architectural model is not always perfect, but it allows for easier creation and management in this case, which is why it has been chosen.

For the major components of the application, we first have the Presentation Tier. The presentation layer is what the end-user sees and can interact with. (“What is 4 tier architecture? - Architecture”) Some of the components that will be part of the presentation layer for this application is the user interface, in which the user will interact with the application and

communicate with the software. This will include the input validation necessary and the management of any data that will remain local. In this layer also reside the dashboards and any other input form that the user uses within the application usage. The purpose of this tier is to provide an interface where the user can interact with the application with ease, while providing a pathway for structured requests for the application layer to be created.

The Application layer contains the logic used to exchange data between each layer. When a user has any activity on the presentation layer, the application layer processes it and ensures that the data layer is provided with the information necessary to complete the action (Indeed). In this layer, we will create rules and operations, such as user validation and the processing of analytics. Abstraction is achieved through APIs, which function as intermediaries between tiers. This design reduces coupling, and promotes modularity, which enables independent updates to components without affecting other functionalities or the overall system.

In the application, we ensure low coupling by defining each tier and maintaining some sort of separation amongst one another. APIs mediate interactions between the tiers, minimizing cascading changes and isolating any dependencies that might occur. On another hand, high cohesion is achieved by assigning a single responsibility to each module. This way there are no created dependencies, and the components remain focused and maintainable.

Architectural Elements

Security

Security is a critical consideration for our application, as it involves sensitive information like personal health metrics and dietary preferences. For this same reason, it is crucial for the applicant to implement different practices that will ensure that our users are well protected. As developers, we understand the importance of keeping data safe and look to implement everything

possible to ensure that data is kept secure. Another feature that is important to security is data encryption.

Hardware

The hardware for a nutrition tracking app must support scalability and provide a reliable system that can manage data storage and user requests. The app will be hosted on a cloud platform, such as Azure, to leverage managed services or any redundancy the app may have. This host should be able to distribute traffic within the presentation tier, therefore ensuring smooth user experience during peak usage time. High performance SSDs will back the persistence tier, which will enable quick retrieval and storage of user logs and analytics.

User Interface

For this application, we look for a user interface that is user-friendly, accessible, and appealing to the user, to encourage regular usage of the app. Given that we look forward to having users be able to constantly be able to access, we look for the app to have a mobile first design, which will ensure compatibility across distinct types of devices. On this note, the app will be set to include accessibility tools that can comply with ARIA standards, to accommodate as many users as possible.

The personalized dashboard is the heart of the app's user interface, by providing the centralized view of the user's nutritional progress. Visual elements like graphs will allow users to see their short term or long-term trends. For example, a weekly bar chart might show the difference between calorie consumption across the whole week. In this dashboard, users can personalize the dashboard by selecting the widgets they find the most relevant to them.

Internal Interface

This interface ensures that the app processes, validates and retrieves data efficiently, ensuring that the functionality for the users is working seamlessly. To achieve this, the app will make use of APIs, which will manage queries, track user progress and analyze eating patterns. Middleware will also be the connecting point for requests onto the database and will process and validate any input given to ensure consistency and security.

External Interface

On the other hand, the external interface will allow for integration with third-party services, which will enhance the app's functionality. Integration with external food databases and connection to fitness trackers, such as Fitbit and Apple Watch, will make the data and analytics given by the app more accurate. In addition to this, it will allow for more data to be acquired by the app, and therefore more information given by the dashboards.

Persistence and data tier

The persistence and data tier manages the app's storage and retrieval of information. It is important to maintain a well-structured tier, to ensure reliability and scalability. For the database, it was important to have a relational database that could store structured data, making sure it stays integral and supports complex queries. In addition to this, a Data Access Layer will abstract the interactions between the application tier and the databases, giving us consistent data access. This function ensures that we have optimized queries.

In conclusion, it is optimal for the application to hold three characteristics: accessible, functional, and secure. By doing this, we ensure that the application will be of excellent quality and of use for the end users. With the integration of APIs, we make sure that the app is more dependable, making sure that the users can trust the application. Moreover, by having thoughtful consideration of coupling and cohesion, the application components remain modular and

reusable, simplifying future updates and making the app more maintainable. The architecture and components of this application can support the current components of the application, while also giving flexibility and room for growth, ensuring that the application is dependable and aligned with user needs for a long time.

Works Cited

1. *N-Tier Architecture Style: Definition and Advantages | Indeed.Com*, www.indeed.com/career-advice/career-development/n-tier-architecture. Accessed 23 Nov. 2024.
2. *What is 4 tier architecture? - Architecture*, https://www.architecturemaker.com/what-is-4-tier-architecture/.