Quick recap

The meeting covered the fundamentals of APIs, focusing on REST APIs and their design principles. It explained the importance of standardized approaches in API development, HTTP protocols, and different methods for sending data in API requests. The session concluded with recommendations for best practices in REST API design and an introduction to writing APIs using the Spring framework.

Next steps

- All attendees: Read the Microsoft REST API guidelines article shared by the instructor for better understanding of REST API conventions and recommendations
- All attendees: Review the concepts of HTTP request methods and their intended uses
- All attendees: Study the differences between request parameters, query parameters, and request body for API data transmission
- All attendees: Prepare for the next class which will cover practical implementation of APIs using Spring Framework and Spring Boot

Summary

APIs and System Procedures Overview

The session began with a discussion on the importance of following system rules and procedures, using a bank deposit scenario to illustrate this point. The instructor then introduced the concept of APIs (Application Programming Interfaces) as a way to expose functionalities to external systems, comparing it to how banks provide guidelines for customer interactions. They explained that APIs serve as contracts between the server and client, defining how requests and responses should be structured, without the client needing to understand the internal logic of the server. The session concluded with a brief mention of different types of APIs and their relevance in software development.

Understanding APIs and Their Types

AlgoCamp explained the concept of APIs, distinguishing between network APIs and browser APIs. They discussed how APIs expose functionalities, whether locally or across machines, and emphasized that APIs can be implemented without network interaction, such as in browser APIs or Java's stream API. AlgoCamp also clarified the differences between libraries, frameworks, and APIs, explaining that libraries and frameworks are code implementations that can be directly used, while APIs are a way to expose functionalities that might require network interaction. They concluded by discussing the open-source nature of many software tools and the importance of not reinventing the wheel.

APIs in Engineering: Standards and Flexibility

The discussion focused on understanding APIs, particularly network APIs, and their importance in engineering. AlgoCamp explained that while theory is less relevant for engineers, coding APIs is crucial. They emphasized that without a standardized approach, managing APIs would be chaotic, similar to how banking systems require common rules for interaction. The conversation also touched on how Bluetooth and other technologies have specific use cases and limitations, using analogies like ATMs to illustrate optimization in technology. The session concluded with a reminder that while recommendations for API development exist, they are not mandatory, allowing flexibility while encouraging best practices.

Understanding REST APIs and Alternatives

AlgoCamp explained that REST APIs are a set of recommendations for creating scalable and manageable APIs, emphasizing that they are widely used in the tech industry. He highlighted that while REST is the primary focus, they will also cover GRPCs and GraphQL. AlgoCamp clarified that REST stands for Representational State Transfer and explained that while the full form doesn't provide much insight, REST APIs are designed to ensure good API practices. He noted that major tech companies like Google, Facebook, and Amazon have started using GRPCs, but REST remains a popular choice.

Json for REST Data Transfer

Json was introduced as a data transfer format that facilitates key-value pair storage, similar to Javascript objects but independent of the language. It was explained that Json is lighter weight than Xml and more human-readable, making it the recommended format for data transfer over REST APIs. The discussion concluded with an announcement that more recommendations for REST would be covered in the next segment, and a link to a video presentation about GRPC was shared for those interested in learning more about that topic.

REST Endpoint Design Best Practices

The instructor explained the concepts of URIs, URLs, and endpoints, emphasizing that endpoints should be resource-oriented according to REST recommendations. They discussed the difference between resource-oriented (e.g., /users/22/orders) and action-oriented (e.g., /create-user) endpoint designs, noting that while REST prefers resource orientation, other frameworks like gRPC recommend action orientation. The instructor highlighted that resource-oriented endpoints can be confusing as they don't clearly indicate the action being performed, which led to the introduction of another REST recommendation for including HTTP verbs in the URI to better describe the action.

HTTP Fundamentals and REST APIs

The instructor explained the fundamentals of HTTP, the protocol used for network interactions in REST APIs. They defined HTTP as a protocol for transferring hypertext, breaking it down into its components: the request (containing the method, URL, headers, and body), and the response (containing a status code and body). The instructor emphasized that HTTP methods like GET, POST, PUT, and DELETE are indicators of the request's purpose, not strict instructions. They demonstrated these concepts using a live example from the Swiggy website, showing how HTTP requests are made and how to inspect them using the developer console.

HTTP and REST API Fundamentals

AlgoCamp explained the HTTP response status codes and their meanings, distinguishing between client errors (400-499) and server errors (500-599). They discussed REST API design principles, emphasizing the importance of combining endpoints with HTTP request methods to define functionality. AlgoCamp clarified the differences between POST and PUT requests, explaining that POST is generally used for creation while PUT is used for updating, and introduced PATCH as a method for partial updates. They also touched on the relationship between HTTP and other protocols like gRPC, noting that while both can use HTTP, gRPC optimizes for HTTP/2 and enforces stricter requirements.

API Data Transmission Methods Explained

The instructor explained the different ways to send data in API requests, including request parameters, query parameters, and request body. They discussed when to use each method, noting that request parameters are used to identify specific resources, query parameters are for filtering criteria, and the request body is for sensitive or complex data. The instructor also covered URL encoding and shared a Microsoft article on REST API best practices, recommending students read it before the next class where they will begin learning about writing APIs and the Spring framework.