Architecture for a SaaS platform specializing in modern REST API Design and Distribution using Javascript

Prof.Manisha Navale[1], Aryan Kathawale[2] ,Khushi Mahajan[3], Swapnil Kamble[4], Tanmay Pareek[5]

[1,2,3,4,5] *Computer Science Dept, NBN Sinhgad Technical Institute Campus.*

---***---

Abstract - PaaS allows API providers to have full control over their APIs by providing features such as API management, marketplace functionality, monetization options, and analytics. It simplifies the process of API creation, publication, versioning, and documentation generation. The marketplace functionality enables API providers to reach a broader audience and monetize their APIs effectively.

Moreover, a dedicated developer portal with comprehensive documentation, code samples, and interactive API consoles helps onboard developers seamlessly and accelerates API integration. The PaaS also supports integration with popular development tools, frameworks, and languages, making it flexible for developers to consume APIs.

Key Words: Typescript, RESTful APIs, SAAS , Javascript API architecture.

1.INTRODUCTION

Monet is actually an API as a service , where we custom make API endpoints for our customers based on the latest possible technologies , we also provide this service to others on our client side for other customers to read , with documentation on how to access.

An example of an API provided by Monet could be in the latest trends of AI , we handle all the logistics of how to provide the AI service in the API and let them handle the API endpoint , we bring to the table best in class service and they bring subscription costs .

the idea is similar to [Rapid API](https://rapidapi.com/hub) in a way , but we not only provide service of APIs but custom make our own . such an application is suited for small scale operation , for startups , and enterprise. Folks who want to try out the latest technology are also welcome to call us and make a deal.

2. Body of Paper

2.1 Overview :

Restful APIs, or Representational State Transfer APIs, play a crucial role in modern web development, serving as a standardized and efficient means of communication between different software applications. These APIs adhere to a set of architectural principles that prioritize simplicity, scalability, and reliability. The importance of Restful APIs lies in their ability to enable seamless integration and interoperability between systems, regardless of the programming languages or platforms they are built upon.

One key aspect of Restful APIs is their emphasis on using HTTP methods such as GET, POST, PUT, and DELETE to perform operations on resources. This standardized approach simplifies the development process and makes it easier to understand and interact with APIs. Additionally, Restful APIs utilize uniform resource identifiers (URIs) to identify and access resources, further enhancing their consistency and ease of use.

Standardization is another crucial aspect of Restful APIs. By following common conventions and best practices, Restful APIs ensure consistency and compatibility across different systems. This enables developers to build applications that can seamlessly interact with a wide range of services and platforms. Standardization also promotes reusability and reduces the learning curve for developers, as they can leverage their existing knowledge and experience when working with Restful APIs.

The importance of Restful APIs extends beyond their technical aspects. They are instrumental in enabling the integration of various applications and services, facilitating data exchange, and enabling the creation of complex and interconnected systems. Restful APIs are widely adopted by major companies and organizations, making them a de facto standard for building web services and applications.

In conclusion, Restful APIs are of paramount importance in modern web development due to their standardized nature, simplicity, and scalability. They provide a consistent and efficient means of communication between different systems, enabling seamless integration and interoperability. By adhering to common conventions and best practices, Restful APIs promote reusability and reduce the learning curve for developers. Their widespread adoption makes them a fundamental component of the web development ecosystem.

SaaS (Software as a Service) providers offering API as a service have become increasingly popular in the technology landscape. These providers serve as a bridge between different technologies, enabling seamless integration and communication between various applications and services. By offering APIs as a service, SaaS providers simplify the process of building and managing APIs, reducing development time and effort for businesses.

The need for API as a service arises from the ever-growing complexity of modern software systems. As organizations adopt diverse technologies and platforms, the ability to connect and exchange data between these systems becomes crucial. API as a service providers offer a unified and standardized approach to building, securing, and managing APIs, effectively acting as intermediaries that facilitate communication between different technologies.

JavaScript (JS) is an excellent choice for building applications that provide API as a service. JS has gained immense popularity as a programming language due to its versatility and wide adoption. It is supported by almost all major web browsers and can be used for both front-end and back-end development. This makes JS an ideal choice for building API as a service applications, as it allows developers to create end-to-end solutions using a single language.

Furthermore, JS frameworks and libraries, such as Node.js and Express.js, provide a robust foundation for building scalable and efficient APIs. These frameworks offer a wealth of tools and utilities that streamline the development process, making it easier to handle complex tasks such as routing, authentication, and data manipulation. Additionally, the vibrant JS community ensures a wealth of resources, tutorials, and support for developers working with API as a service applications.

Next.js is an excellent choice for building API as a service applications due to its SSR capabilities, API routes, seamless frontend integration, support for static site generation, incremental adoption, and developer-friendly features. It provides a robust and efficient development environment, enabling developers to create high-performance, scalable, and easily maintainable API as a service solutions.

2.2 Underlying Architecture :

Edgeless computing is an innovative concept that revolutionizes the way we think about data processing and storage. Unlike traditional computing architectures that rely on centralized servers and cloud infrastructures, edgeless computing brings computing power and storage capabilities directly to the edge of the network.

By distributing computational resources across a vast network of interconnected devices, edgeless computing enables real-time processing, low latency, and enhanced privacy and security. This is also considerably better for us due to colocation and not codependence to the next js application , scaling is no longer an issue in such an application, with use of Vercel’s edgeless computing platform, architecturing a microservice sentric plan works best for such a product.

The framework in question , Next js also acts as a frontend website for the web , the main problem arises in sending and receiving the api information in real time , status of the .api , also the real time update of new and old apis etc .

The need of a database arises for such activities , prisma works best for such a job , it's already been used in production for a long time and is compatible with next js , also with its type safety and inbuilt constraints are good fit for this project with changing architecture.

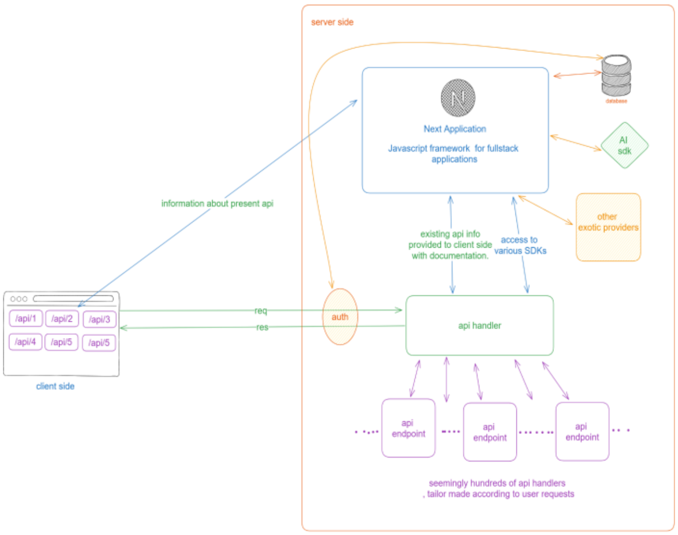
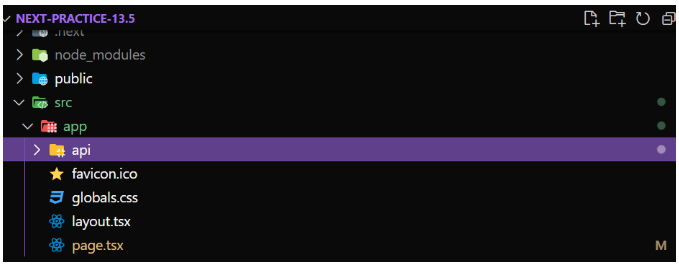

server side
database
Next Application
Javascript framework for fullstack applications
AI sdk
information about present api
other exotic providers
existing api info provided to client side with documentation.
access to various SDKs
/api/1
/api/2
/api/3
/api/4
/api/5
/api/5
req
res
auth
api handler
client side
api endpoint
api endpoint
api endpoint
seemingly hundreds of api handlers , tailor made according to user requests

Fig: Software Architecture Overview

2.3 Proof Of Concept

In this section we are going to give an example of how an api can be made using the next js router . To set up an next js application . the command to make a new next application is npx create-next-app@latest.

Underlying project structure is very robust. NextJs works on file based routing method meaning the route is matched with a file in the project. API directory is a special name given to this folder because it can be used to interact with the backend because of its capabilities to execute code on the server.



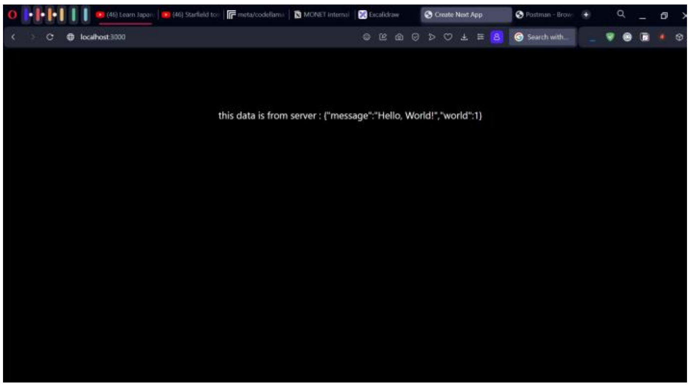
```
NEXT-PRACTICE-13.5
  > .next
  > node_modules
  > public
  v src
    v app
      > api
        favicon.ico
        globals.css
        layout.tsx
        page.tsx                                   M
```

fig 2 : Project structure.

api folder can theoretically hold hundreds of sub directories, one for each api .

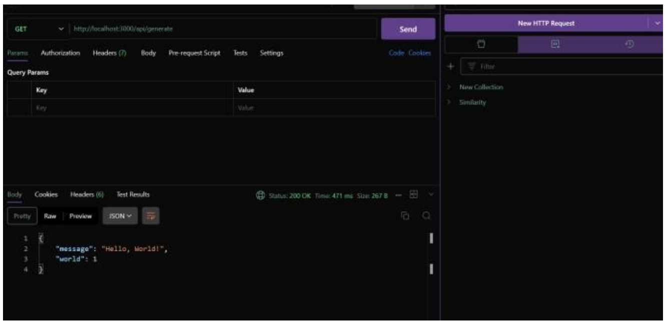
An example of an API provided by Monet could be in latest trends of AI , we handle all the logistics of how to provide the AI service in the API and let them handle the API endpoint , we bring to the table best in class service and they bring subscription costs . Custom GPTs and Image Generation as well as Video Generation , with standard apis as well like Weather apis etc .

The idea is similar to Rapid API in a way , but we not only provide service of APIs but custom make our own . such an application is suited for small scale operation , for startups , and enterprise.



this data is from server : {"message":"Hello, World!","world":1}

fig 3 : Showing sent data in frontend.



GET http://localhost:3000/api/generate

Status: 200 OK Time: 471 ms Size: 267 B

```
{
    "message": "Hello, World!",
    "world": 1
}
```

fig 4 : Showing sent data in anonymous testing environment

2.4 Advantages :

1. First class support of Microservices to the companies with 99.99% uptime , guaranteed by vercel's cloud infrastructure .
2. Personalized GPTs designed specifically to work for the user

2.5 Disadvantages :

1. Scaling Up this service could be an issue for future and need to hire more people to keep the service running .
2. Scale of project could be hard to maintain , the codebase could be complex , team could be changing .

3. CONCLUSIONS

we know now that NEXT has a file based routing , that is true but , this does also mean we can theoretically have millions or routes and endpoints , and to scale have multiple apps running and handling different types of endpoints , meaning we can self host our endpoints also display endpoints from other users This combined with reach of JS could mean we can theoretically do anything , but our value comes with doing everything best which is where the learning to do things best comes in .Our main goal is like a tailor or a designer , everyone knows how to make clothes to some extent , it is only the best tailors which knows the problems, knows the features and design the best suit for the occasion.

ACKNOWLEDGEMENT

In result we can see the accessed data in both frontend and api endpoint active regardless . Typescript and server components make it easy to bridge between frontend and backend , and since everything is in a single codebase , we can simply have a lambda functions (experimental) endpoint in the future to make millions of edge functions for every endpoint .

REFERENCES

1. A Python application programming interface for accessing Philips iSyntax whole slide images for computational pathology byNita Mulliqi, Kimmo Kartasalo, Henrik Olsson, Xiaoyi Ji, Lars Egevad, Martin Eklund, Pekka Ruusuvuori .

2. Complementing JavaScript in High-Performance Node.js and Web Applications with Rust and WebAssembly Kyriakos-Ioannis D. Kyriakou and Nikolaos D. Tselikas * Communication Networks and Applications Laboratory, Department of Informatics and Telecommunications, University of Peloponnese, 221 00 Tripoli, Greece * Correspondence: ntsel@uop.gr; Tel.: +30-2710-372216

3. Comparative Evaluation of JavaScript Frameworks Andreas B. Gizas, Sotiris P. Christodoulou and Theodore S. Papatheodorou HPCLab, Computer Engineering & Informatics Dept., Univ. of Patras, 26500 Rion, Patras {gizas,spc,tsp}@hpclab.ceid.upatras.gr

4. Comparative Evaluation of JavaScript Frameworks Andreas B. Gizas, Sotiris P. Christodoulou and Theodore S. Papatheodorou HPCLab, Computer Engineering & Informatics Dept., Univ. of Patras, 26500 Rion, Patras {gizas,spc,tsp}@hpclab.ceid.upatras.gr

5. THE FUTURE OF JAVASCRIPT: EMERGING TRENDS AND TECHNOLOGIES Urinboev Abdushukur Abdurakhimovich, Assistant teacher of Ferghana branch of Tashkent University of Information Technologies.

6. MODERN WEBSITE DEVELOPMENT WITH STRAPI AND NEXT.JS by Niko Pinnis Thesis Study Programme in Information and Communication Technology Bachelor of Engineering 2022 .

7. React Apps with Server-Side Rendering: Next.js Harish A Jartarghar1 , Girish Rao Salanke1 , Ashok Kumar A.R 1 , Sharvani G.S1 and Shivakumar Dalali2 1Department of Computer Science and Engineering, R.V College of Engineering, Bengaluru, India. 2 Don Bosco Institute of Technology, Bengaluru, India. harishaj.cs18@rvce.edu.in.

8. Verma, Rajat & Dhanda, Namrata & Nagar, Vishal. (2023). Towards a Secured IoT Communication: A Blockchain

Implementation Through APIs. 10.1007/978-981-19-1142-2_53.

9. Do Machine Learning Models Produce TypeScript Types That Type Check? by Ming-Ho Yee{ Northeastern University, Boston, MA, USA } , Arjun Guha { Northeastern University, Boston, MA, USA Roblox Research, San Mateo, CA, USA }

10. Pasdar, A., Lee, Y. C., & Dong, Z. (2023). Connect API with blockchain: a survey on blockchain oracle implementation. ACM Computing Surveys, 55(10), 1-39. [208]. https://doi.org/10.1145/3567582 .