## **1. Introduction to npm json-server**

Json-server is a lightweight and simple yet powerful tool that allows developers to quickly create a REST API using a JSON file as the database. With its ease of use and flexibility, json-server has become a popular choice for prototyping and mocking up API responses. However, mastering json-server can be a daunting task for beginners. In this blog, we will provide essential CRUD examples to help you understand and leverage the full potential of json-server in your projects. Whether you are a seasoned developer or just starting out, this blog will guide you through the process of creating, reading, updating, and deleting data using json-server. So, let's dive in and master npm json-server together!



*https://unsplash.com/@campaign\_creators*

## **2. Understanding CRUD operations with json-server**

Understanding CRUD operations with json-server

Json-server allows you to perform all four main CRUD operations - Create, Read, Update, and Delete. These operations are essential for interacting with data in your json-server API. In this section, we will take a closer look at each operation and provide examples to help you grasp the concept.

1. Create (C) - This operation is used to add new data to your json-server API. You can create a new resource by making a POST request to the appropriate endpoint with the required data. Json-server will automatically generate a unique ID for the new resource and add it to the JSON database.

2. Read (R) - The read operation is used to retrieve data from your json-server API. You can fetch a single resource by making a GET request to the specific endpoint. If you want to retrieve multiple resources, you can make a GET request to the collection endpoint.

3. Update (U) - This operation allows you to modify existing data in your json-server API. To update a resource, you need to make a PUT or PATCH request to the specific endpoint, providing the updated data. Json-server will replace the existing resource with the new data.

4. Delete (D) - The delete operation is used to remove data from your json-server API. You can delete a resource by making a DELETE request to the specific endpoint. Json-server will remove the resource from the JSON database.

Understanding these CRUD operations is crucial for effectively working with json-server. In the next section, we will explore each operation in detail and provide practical examples to solidify your understanding. Stay tuned!

## **3. Creating data with json-server**

In this section, we will delve deeper into creating data with json-server. The ability to add new resources is a fundamental aspect of CRUD operations and a crucial skill for anyone working with json-server. We will walk you through the steps required to create new data in your json-server API, providing you with hands-on examples to reinforce your understanding.

To begin, you will need to make a POST request to the appropriate endpoint, along with the necessary data for the new resource. Json-server will automatically assign a unique ID to the newly created resource and add it to the JSON database. We will demonstrate this process using practical examples, ensuring you grasp the concept fully.

By mastering the skill of creating data with json-server, you will expand your capabilities and be well-equipped to handle any development task. Stay tuned for the upcoming examples that will empower you to become a proficient user of json-server.

## **4. Reading data with json-server**

In this section, we will explore the process of reading data with json-server. Being able to retrieve and access data from your json-server API is a fundamental aspect of working with json-server. We will guide you through the steps required to retrieve data, providing you with practical examples to solidify your understanding.

To read data from your json-server API, you will need to make a GET request to the appropriate endpoint. Json-server will return the requested data in the response, allowing you to manipulate and utilize it as needed. We will showcase various examples that demonstrate different scenarios and highlight useful techniques.

By becoming proficient in reading data with json-server, you will gain the ability to extract valuable information from your API and use it effectively in your applications. Stay tuned for the upcoming examples that will further enhance your mastery of json-server's read operations.

## **5. Updating data with json-server**

In addition to reading data, updating data with json-server is another essential skill to master. By being able to modify the data in your json-server API, you can ensure that your application remains up-to-date and relevant. In this section, we will guide you through the process of updating data using json-server's PUT and PATCH requests.

To update data with json-server, you will need to make a PUT request to the appropriate endpoint. This request will replace the existing data with the new values provided. Alternatively, you can use the PATCH request to update specific fields within the data, rather than replacing the entire object.

We will provide you with practical examples to demonstrate these concepts and help you gain a solid understanding of the updating process. By mastering the art of updating data, you will have full control over your json-server API and ensure that it meets the needs of your application. Stay tuned for the upcoming examples that will further enhance your mastery of json-server's CRUD operations.

## **6. Deleting data with json-server**

In addition to updating and reading data, deleting data is another fundamental skill to have when working with json-server. Being able to remove data from your json-server API allows you to maintain the cleanliness and relevance of your data store.

To delete data with json-server, you need to make a DELETE request to the appropriate endpoint. This request will remove the specified data from your API.

In this section, we will walk you through the process of deleting data using json-server's DELETE request. We'll provide you with practical examples and real-world scenarios to help you grasp the concept and become proficient in deleting data from your json-server API.

Stay tuned for the upcoming examples that will further enhance your understanding and mastery of json-server's CRUD operations.

## **7. Advanced features and tips for using json-server**

In addition to the basic CRUD operations, json-server offers several advanced features and tips that can further enhance your experience with the tool. These features will allow you to optimize your workflow and make the most out of your json-server API.

One of the key features of json-server is the ability to use query parameters to filter and sort data. By including query parameters in your API requests, you can easily retrieve specific subsets of data or sort the data based on specific criteria. This feature is incredibly powerful as it allows you to customize the data you receive from your json-server API, making it easier to work with and analyze.

Another advanced feature of json-server is the ability to use middleware to extend its functionality. You can create custom middleware functions to handle authentication, data validation, and other tasks. This allows you to customize json-server to suit your specific needs and requirements.

In addition to these features, there are also several tips and best practices that can improve your usage of json-server. For example, it is recommended to use version control to track changes to your json-server API and to use descriptive and meaningful field names in your JSON files.

By understanding and implementing these advanced features and tips, you can take your json-server skills to the next level and become even more proficient in working with JSON data. In the next section, we will delve into these features and tips in detail, providing you with practical examples and step-by-step instructions. So stay tuned!

## **8. Conclusion: Become a json-server pro!**

In this blog series, we have covered the basic CRUD operations of json-server and explored its advanced features along with useful tips and best practices. With the ability to use query parameters for data filtering and sorting, json-server offers excellent flexibility in customizing your API requests.

In addition, we discussed how you can extend json-server's functionality using middleware functions. By creating custom middleware, you can handle various tasks like authentication and data validation, tailoring json-server to fit your specific requirements.

We also emphasized the importance of version control for tracking changes to your json-server API and using descriptive field names in your JSON files to improve readability.

By implementing these advanced features and following the tips provided, you can truly master json-server and become a seasoned pro. In the next section, we will dive into practical examples and step-by-step instructions to solidify your understanding and help you apply these concepts effectively. So, get ready to take your json-server skills to the next level!