

BHAI PARMANAND DSEU SHAKARPUR CAMPUS-II

(DELHI SKILLS AND ENTERPRENEURSHIP UNIVERSITY)

Project Report

On

**“TableEase”**

For the fulfilment of the degree Master in Computer Application

Master of Computer Application (MCA)

Semester – IV

**Submitted By**

Shubhangi Sharma

230313053024

**Abstract**

This project aims to develop a frontend web application using either **React with TypeScript** to display JSON data in a **well-structured, user-friendly tabular format**. The primary goal is to create a **configurable API interface** that allows users to dynamically set API endpoints and parameters from **Swagger-based API definitions**. By integrating with **OpenAPI Specification**, the application will provide a seamless way to fetch and display data without requiring hardcoded API configurations. The project follows an **Agile methodology**, ensuring iterative improvements and adaptability based on feedback. The application will be designed with a focus on **performance, usability, and maintainability**.

**Introduction**

Modern applications heavily rely on APIs for fetching and exchanging data. However, displaying JSON data in a structured manner is often a challenge, especially when dealing with **dynamic or unknown data structures**. This project addresses that challenge by providing a **configurable and interactive frontend** that allows users to:

* Select and configure API endpoints dynamically from a Swagger definition.
* Fetch and display JSON data in a **prettified tabular format**.
* Customize API request parameters directly within the UI.
* Enhance user experience through a clean and intuitive interface.

By leveraging **React or Angular with TypeScript**, the project will ensure **scalability, performance, and a modular architecture**. Using OpenAPI Specification will allow for easy integration with various APIs, such as the **Pet Store API (Swagger example)**, enabling real-time data retrieval and visualization.

This application can be useful for **developers, testers, and API consumers** who need a quick way to inspect API responses without manually parsing JSON. Additionally, its **flexibility** makes it adaptable for multiple use cases, including **internal API testing, dashboarding, and business intelligence applications**.

**Software Used**

For the successful implementation of this project, the following tools and technologies will be utilized:

**Frontend Development**

* **Framework:** React (with TypeScript) / Angular (with TypeScript)
* **State Management:** React Context API / Redux (for state management)
* **UI Styling:** Tailwind CSS (for React) / Bootstrap (for Angular)

**API Handling and Integration**

* **API Requests:** Axios / Fetch API
* **Swagger (OpenAPI) Integration:** OpenAPI Specification for dynamic API configurations
* **API Testing Tool:** Postman (for testing and debugging API responses)

**Development & Deployment Tools**

* **Code Editor:** Visual Studio Code
* **Version Control:** Git (with GitHub or GitLab for project tracking)
* **Package Manager:** npm / yarn

**Software Methodology**

The project follows the **Agile Software Development** methodology, which is best suited for dynamic and iterative development. Agile offers several advantages:

1. **Flexibility and Adaptability** – Since API structures and requirements can change, Agile allows quick modifications and iterative improvements.
2. **Frequent Updates and Testing** – Continuous development cycles ensure that bugs and performance issues are addressed quickly.
3. **User Feedback Integration** – As the project progresses, feedback can be incorporated easily, ensuring better usability.
4. **Incremental Development** – Features will be built in phases, allowing early testing and refinement before final deployment.

The Agile workflow will be implemented through **weekly sprints**, where specific development goals will be defined and achieved in a structured manner.

**Planned Agile Sprint Breakdown:**

* **Sprint 1:** Project setup, UI design, basic API integration
* **Sprint 2:** Implementing dynamic Swagger-based API configuration
* **Sprint 3:** JSON-to-table transformation with sorting and filtering
* **Sprint 4:** Enhancing user experience and final testing

By using Agile, the project will remain **organized, transparent, and adaptable**, leading to a more refined and functional product.

**Conclusion**

In this initial phase, the **project requirements, objectives, and technology stack** have been clearly defined. The decision to use **React or Angular with TypeScript** ensures **high performance, reusability, and maintainability**. The choice of **Swagger-based API integration** enables **dynamic API configuration**, making the application adaptable for various data sources.

Moving forward, the focus will be on:

1. **Setting up the project structure and UI framework.**
2. **Creating API integration mechanisms to fetch and display JSON data dynamically.**
3. **Building the JSON-to-table transformation logic.**

By following an **Agile development process**, the project will evolve **iteratively**, ensuring that it remains aligned with user needs and expectations.