**API Documentation Template**

**Overview**

Welcome to the **VDINE Hub: A Web-based Seamless Restaurant Management System for VDINE Restaurant.** This allows users to add, retrieve, update, and delete menu items and take user orders.

**Base URL**

https://api.example.com/api/v1

**Features**

* Create, read, update, and delete users
* Secure authentication via API keys
* Rate limiting for fair usage
* Error handling and validation

**Authentication**

This API requires authentication via an API key.

**How to Authenticate**

1. Sign up and get your API key from the API Dashboard.
2. Include the API key in the request headers:

Authorization: Bearer YOUR\_API\_KEY

Alternatively, you can pass the API key as a query parameter:

<https://api.example.com/api/v1/users?api_key=YOUR_API_KEY>

**Note:** Always keep your API key secure and avoid exposing it in public repositories.

**Endpoints**

**1. Get All Menu**

Retrieves a list of menu items.

**Request:**

GET /menu

**Headers:**

Authorization: Bearer YOUR\_API\_KEY

**Response:**

{

"users": [

{ "id": 1, "name": "John Doe", "email": "john@example.com" },

{ "id": 2, "name": "Jane Doe", "email": "jane@example.com" }

]

}

**Status Codes:**

* 200 OK - Request successful
* 401 Unauthorized - Invalid API Key

**2. Get User by ID**

Fetches details of a single user.

**Request:**

GET /users/{id}

**Example Request:**

GET /users/1

**Response:**

{

"id": 1,

"name": "John Doe",

"email": "john@example.com"

}

**Status Codes:**

* 200 OK - Success
* 404 Not Found - User not found

**3. Create a User**

Adds a new user to the system.

**Request:**

POST /users

Content-Type: application/json

**Body:**

{

"name": "Alice Smith",

"email": "alice@example.com"

}

**Response:**

{

"id": 3,

"name": "Alice Smith",

"email": "alice@example.com"

}

**Status Codes:**

* 201 Created - User successfully created
* 400 Bad Request - Invalid input

**4. Update a User**

Modifies an existing user’s details.

**Request:**

PUT /users/{id}

**Body:**

{

"name": "Alice Johnson",

"email": "alice.johnson@example.com"

}

**Response:**

{

"id": 3,

"name": "Alice Johnson",

"email": "alice.johnson@example.com"

}

**Status Codes:**

* 200 OK - Updated successfully
* 400 Bad Request - Invalid data

**5. Delete a User**

Removes a user from the system.

**Request:**

DELETE /users/{id}

**Response:**

{

"message": "User deleted successfully"

}

**Status Codes:**

* 200 OK - User deleted
* 404 Not Found - User does not exist

**Error Handling**

**Common Error Responses**

All error messages follow this format:

{

"error": "Error message",

"code": 400

}

| **Status Code** | **Meaning** | **Description** |
| --- | --- | --- |
| 400 | Bad Request | Invalid input data |
| 401 | Unauthorized | Invalid API key |
| 403 | Forbidden | No access rights |
| 404 | Not Found | Resource not found |
| 500 | Server Error | Internal API failure |

**Rate Limits**

This API enforces rate limits to prevent misuse.

| **Plan** | **Requests per Minute** |
| --- | --- |
| Free Plan | 100 requests |
| Premium | 1000 requests |

Exceeding the limit returns:

{

"error": "Too many requests",

"code": 429

}

**Changelog & Versioning**

**Changelog**

* **v1.2.0** - Added rate limits
* **v1.1.0** - Added update user endpoint
* **v1.0.0** - Initial API release

**Versioning**

To ensure backward compatibility, include version numbers in API requests:

https://api.example.com/api/v1/users

**Contact & Support**

For API issues, contact our support team:

* Email: [support@example.com](mailto:support@example.com)
* API Dashboard: [https://dashboard.example.com](https://dashboard.example.com/)