# NICSPACE Documentation

### 1. Overview 1.1 Purpose and Key Features

This section introduces the website, outlining its primary goals and highlighting key features such as image and video posting, pagination, tag-based search, post saving, responsive UI, and API integration.

### 1.2 Technologies Used

Offering insights into the technologies employed in the development, including MongoDB for database storage, Multer for frontend image handling, Node.js for backend operations, and Firebase for image and video storage.

## 2. Architecture

### 2.1 Frontend

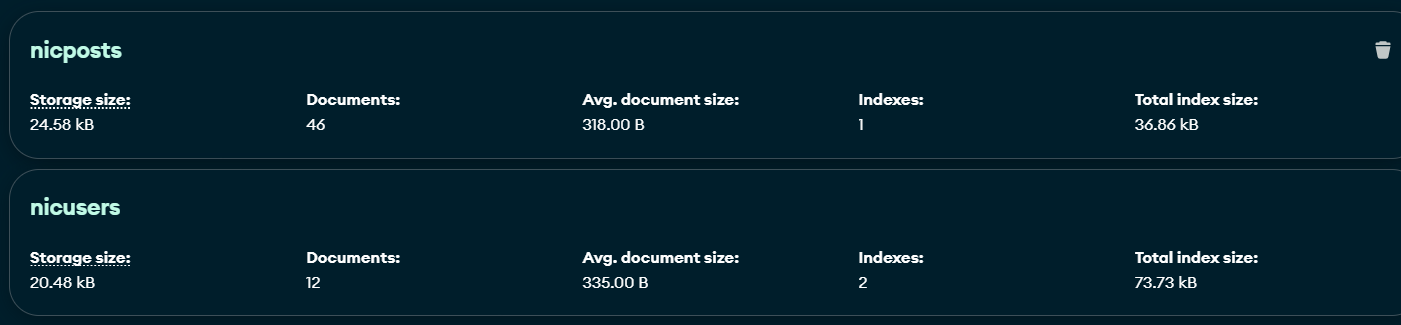
Used React Js as frontend framework with tailwind CSS for good User Interface experience.

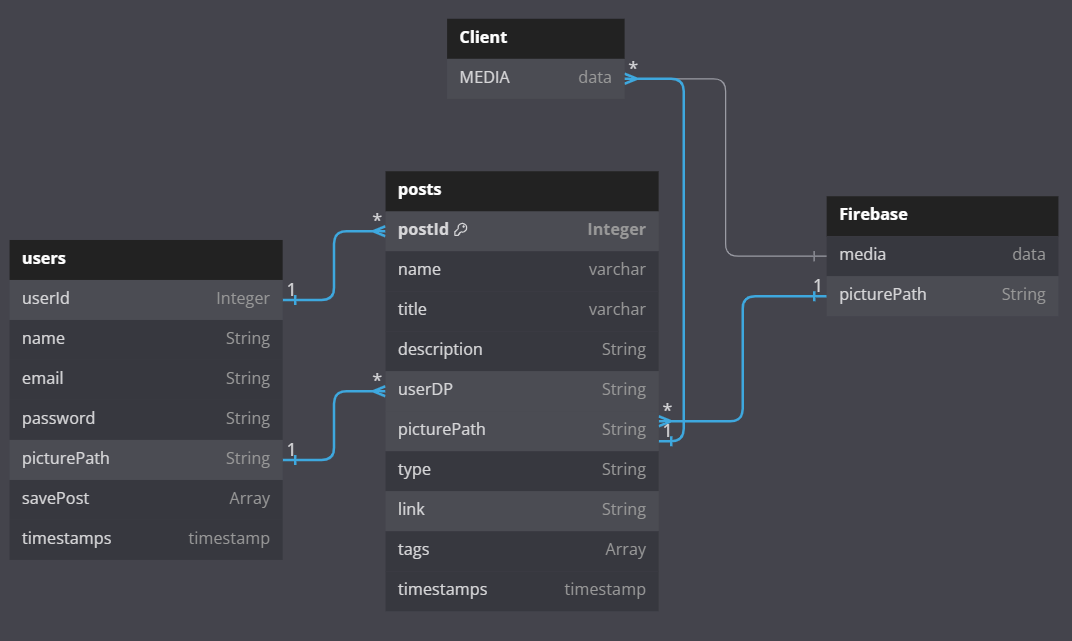
### 2.2 Backend

Create seamless backend using Nodejs, Express for RESTful API’s, multer for storing images from frontend to backend,

### 2.3 Database

MongoDB for fast API responses and use of Firebase for storing images and videos, which are being fetched by client using MongoDB, provides images unique name, and use it to retrieve data from firebase





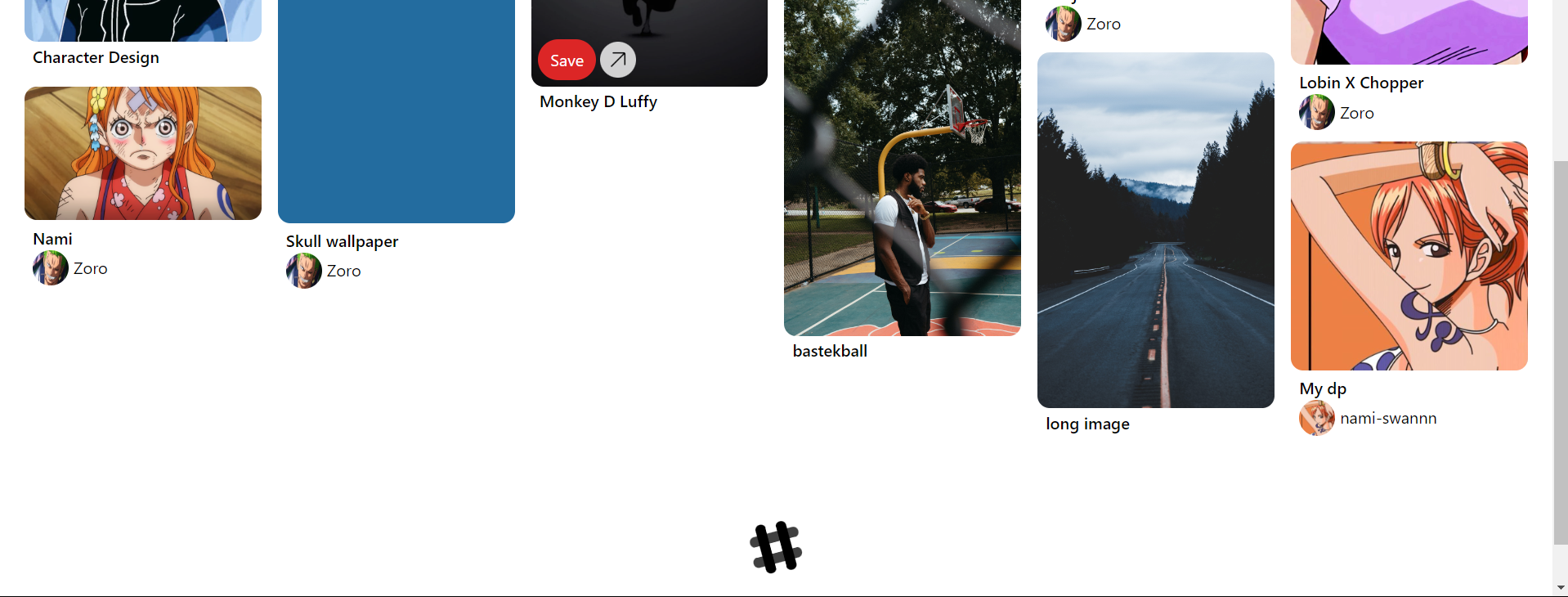
## 3. Features

#### 3.1 Image and Video Upload

Users can easily upload images and videos directly from the frontend interface. The process is streamlined with the implementation of Multer, a middleware for handling file uploads in Node.js. Uploaded media files are efficiently stored in Firebase storage, ensuring secure and reliable storage.

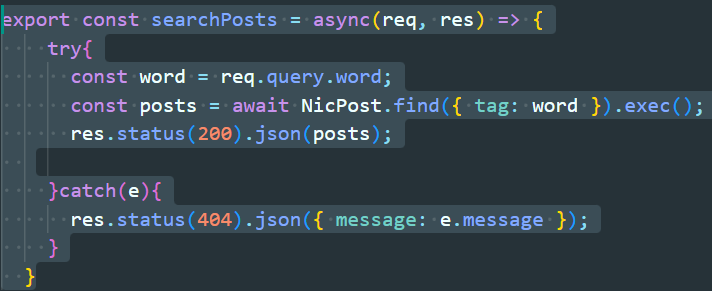
### 3.2 Pagination

### To enhance user experience, the website incorporates a robust pagination system. This feature allows users to navigate through many posts easily, ensuring a smooth and efficient browsing experience.



### 3.3 Tag-based Search

### The website includes a powerful tag-based search functionality, enabling users to find posts based on specific tags associated with each post. This feature enhances content discoverability and allows users to explore topics of interest efficiently.



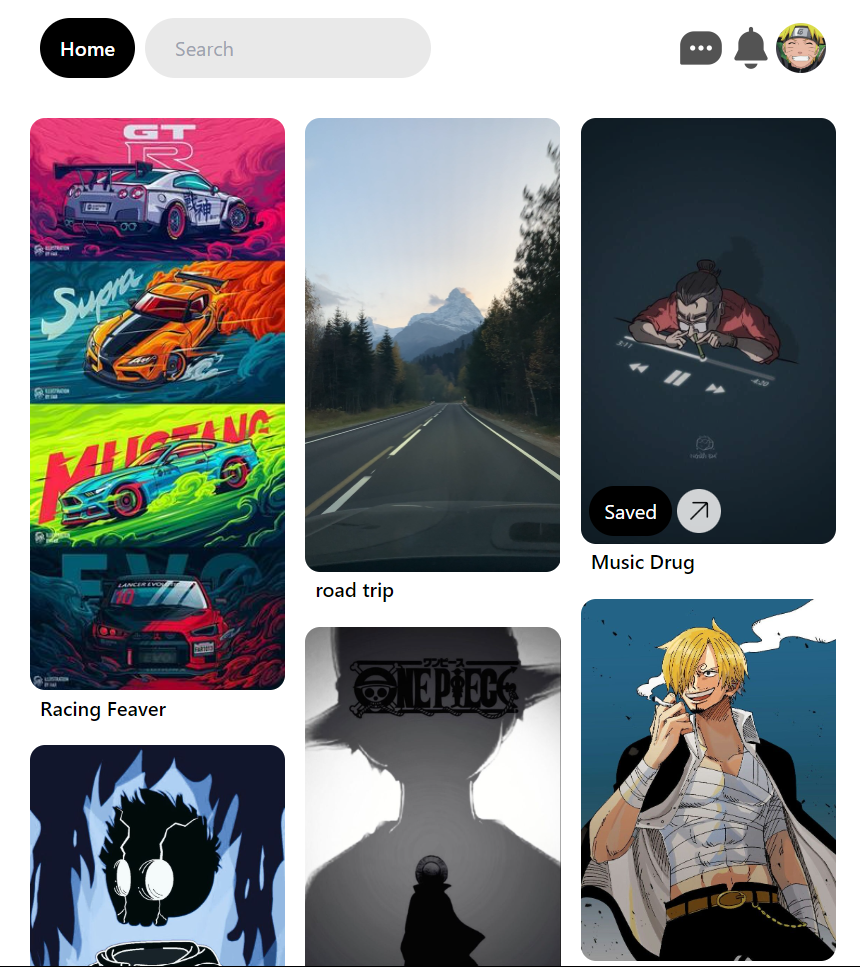
### 3.4 Post Saving

### Users can save their favourite posts for later viewing. This feature adds a personalized touch to the user experience, allowing individuals to curate their content and revisit it conveniently.



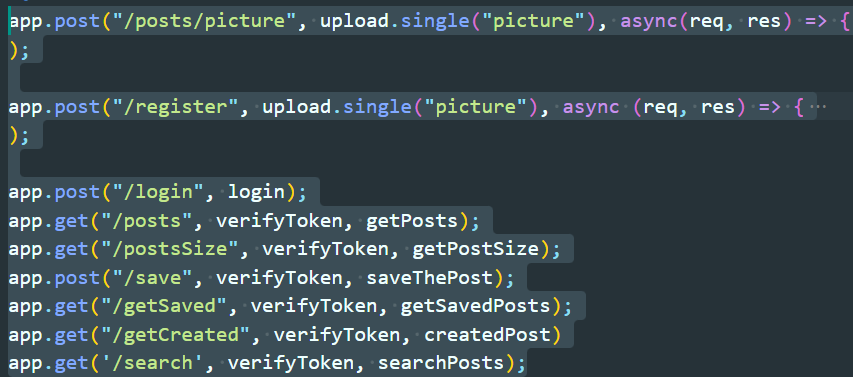
### 3.5 Responsive UI

### The website boasts a responsive user interface, ensuring a consistent and user-friendly experience across various devices and screen sizes. Whether accessed from a desktop, tablet, or smartphone, users can enjoy a visually appealing and fully functional interface.



#### 3.6 API Integration

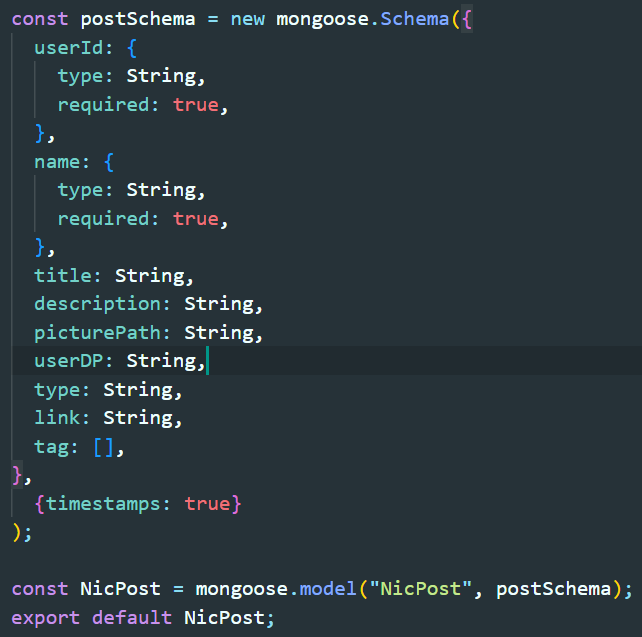
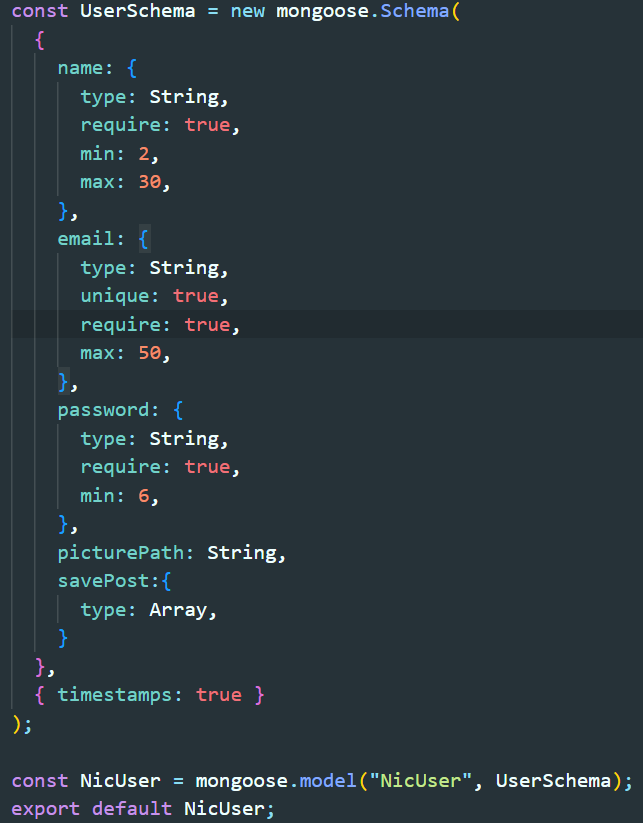
To extend functionality and provide users with additional features, the website integrates external APIs. This integration enhances the overall user experience by incorporating external data and services seamlessly into the platform.



### 4. Implementation

#### 4.1 MongoDB Schema

The MongoDB schema serves as the backbone for storing post and user information efficiently. This section provides an in-depth look into the structure and organization of data within the MongoDB database.



Key Components:

**Users Collection:**

User information, including username, email, and hashed passwords.

User preferences and settings.

Unique identifiers and authentication tokens.

**Posts Collection:**

Post content, such as images, videos, and text.

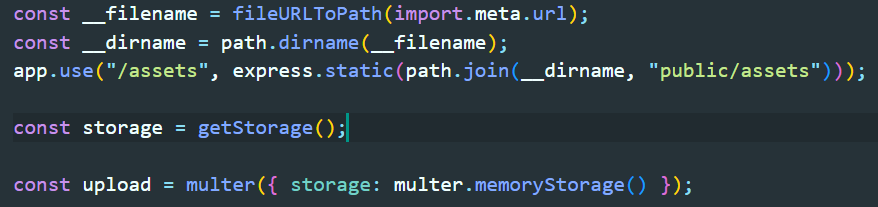
Associated user information for tracking post creators.

Timestamps for post creation and updates.

Tags and categories for efficient categorization and search.

#### 4.2 Multer for Image Storage

Multer is employed as a middleware for handling image uploads from the frontend, ensuring smooth and secure file handling.



Implementation Steps*:*

**Frontend Form:**

Users interact with a form to upload images or videos.

Multer processes form data and extracts files.

**Middleware Configuration:**

Multer middleware is configured with specified parameters.

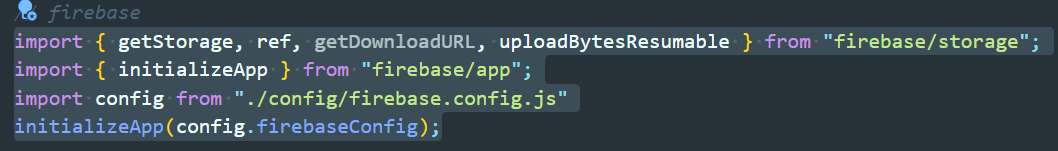
File type restrictions and size limits are enforced.

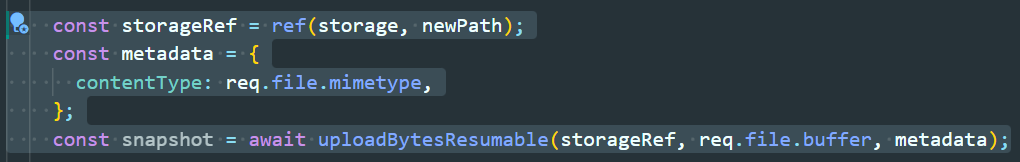
**File Storage:**

Multer directs files to appropriate storage destinations.

Firebase integration ensures secure storage of media assets.

**FIREBASE INTEGRATION**

****



#### 4.3 Node.js Backend

Node.js serves as the backend framework, orchestrating the overall functionality of the website, including routing, request handling, and database interactions