**Sports News Web Application Documentation**

**Overview**

The Sports News Web Application is a web-based application built with Laravel framework that allows users to read sports news and articles posted by the administrator. The application also provides the ability for users to sign up and log in to their account, where they can customize their profile and interact with other users.

**Features**

**Administrator Features**

* Create, edit, and delete news and articles
* Manage users, including adding and removing user accounts
* View user activity and manage user roles and permissions
* Manage application settings and configurations

**User Features**

* Sign up and create an account
* Log in to their account
* View and read sports news and articles
* Search and filter news by category and date
* Like and comment on news and articles
* Customize their profile and manage account settings

**Search functionality**

* Allow users to search for news articles based on keywords or categories.

**Related articles**

* Display related news articles at the end of each article to encourage users to read more content.

**Trending topics**

* Display a list of trending topics or popular articles to help users discover new content.

**The sports web application could have the following pages and functionality:**

1. **Home page:**

* Display latest news articles and game schedules
* Links to sign up and log in pages

1. **Sign up page:**

* Allow users to create a new account
* Validate user input to ensure it meets requirements (e.g. valid email address, strong password)
* Save user information to the **users** table in the database
* Redirect user to the log in page after successful sign up

1. **Log in page:**

* Allow users to log in to their account
* Validate user input and authenticate user against the **users** table in the database
* Redirect user to the user dashboard page after successful login

1. **User dashboard page:**

* Display user's account information
* Allow user to edit their account information
* Display user's saved articles and favorites
* Allow user to add articles to their favorites
* Allow user to delete articles from their favorites

1. **Admin dashboard page:**

* Display admin's account information
* Allow admin to edit their account information
* Allow admin to add new articles and games
* Allow admin to edit or delete existing articles and games
* Display user list and allow admin to manage users (e.g., view, edit, delete)

1. **Article page:**

* Display individual article with heading, content, photo, and publication date
* Allow user to add article to their favorites
* Allow user to share article on social media
* Allow admin to edit or delete article

1. **Contact page:**

* Allow users to send feedback or inquiries to the website administrator
* Validate user input and send email notification to website administrator

Note that this is just a basic outline of the functionality that the sports web application could have. Depending on the specific requirements and goals of the project, there may be additional pages, features, and functionality that need to be implemented.

**System Architecture**

The Sports News Web Application follows a Model-View-Controller (MVC) architecture pattern. The models represent the database tables and the relationships between them, the views are responsible for rendering the HTML templates using Blade templating engine, and the controllers handle the business logic and application flow.

The system consists of the following components:

* **Model:** The data layer of the system, responsible for data storage and retrieval using Laravel's built-in Eloquent ORM. The system includes nine tables: ***users, articles,categories, comments\_table, tag\_news\_table, Activity\_logs\_table, Permissions\_table and User\_roles table***
* **View:** The presentation layer of the system, responsible for rendering the user interface using Blade templating engine and Bootstrap 5.
* **Controller:** The logic layer of the system, responsible for handling user requests and updating the Model and View accordingly. The system includes three controllers: AuthController, ArticleController, and UserController.

**For the font-end development** I recommend Bootstrap because is a popular choice for building responsive and mobile-first web applications, and it comes with a wide range of pre-designed UI components and JavaScript plugins. It is also well-documented and has a large community of users and developers, which makes it easy to find support and resources.

**For the back-end development** in Laravel, you will be using the Laravel PHP framework itself, which provides a wide range of features and tools for building robust and scalable web applications.

Laravel follows the Model-View-Controller (MVC) architectural pattern, which separates the application logic into three main components: the Model (for data storage and retrieval), the View (for rendering the user interface), and the Controller (for handling user requests and updating the Model and View).

In addition to the basic MVC architecture, Laravel also provides many built-in features and tools for common web application tasks, such as authentication, routing, database migrations, and testing. It also supports popular databases such as MySQL, PostgreSQL, and SQLite.

The system uses a MySQL database to store user data, news and article content, and other application data.

**Here are some common tables that you might needed for the system**

1. Users table: This table stores information about website users, such as their username, email address, password, and other personal details.
2. Articles table: This table stores information about articles published on the website, such as their title, content, author, publication date, and other metadata.
3. Categories table: This table stores information about article categories, such as their name and description.
4. Tags table: This table stores information about article tags, such as their name and description.
5. Comments table: This table stores information about user comments on articles, such as their content, author, publication date, and article ID.
6. *User\_roles table*: This table stores information about user roles and permissions, such as their name, description, and associated permissions.
7. *Permissions table*: This table stores information about individual permissions, such as their name, description, and associated roles.
8. *Activity\_logs table*: This table stores information about user activity on the website, such as their login and logout times, page views, and other events.

**Technologies Used**

* Laravel Framework
* MySQL Database
* Bootstrap Framework
* Blade Templating Engine
* jQuery Library

**Installation and Setup**

1. Clone the repository from GitHub
2. Install the required dependencies using Composer
3. Configure the database settings in the .env file
4. Run the database migrations to create the necessary tables
5. Start the application by running the PHP built-in web server.

**Conclusion**

The Sports News Web Application is a fully functional web-based system built using the Laravel PHP framework. It provides users with access to the latest sports news and events, and allows them to create and manage their own accounts. With its clean and responsive design, and easy-to-use interface, the Sports News Web Application is an ideal solution for sports enthusiasts and fans who want to stay up-to-date with the latest happenings in the world of sports.

**Here's an example of the table schemes**

**1. Users table:**

```

CREATE TABLE users (

id INT(11) UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL,

email VARCHAR(255) NOT NULL,

password VARCHAR(255) NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

```

**2. Articles table:**

```

CREATE TABLE articles (

id INT(11) UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

title VARCHAR(255) NOT NULL,

content TEXT NOT NULL,

author\_id INT(11) UNSIGNED,

category\_id INT(11) UNSIGNED,

image\_url VARCHAR(255),

published\_at TIMESTAMP DEFAULT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (author\_id) REFERENCES users(id) ON DELETE SET NULL,

FOREIGN KEY (category\_id) REFERENCES categories(id) ON DELETE SET NULL

);

```

**3. Categories table:**

```

CREATE TABLE categories (

id INT(11) UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL,

description TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

);

```

**4. Tags table:**

```

CREATE TABLE tags (

id INT(11) UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL,

description TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

```

**5. Comments table:**

```

CREATE TABLE comments (

id INT(11) UNSIGNED AUTO\_INCREMENT PRIMARY KEY,

content TEXT NOT NULL,

author\_id INT(11) UNSIGNED NOT NULL,

article\_id INT(11) UNSIGNED NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

FOREIGN KEY (author\_id) REFERENCES users(id) ON DELETE CASCADE,

FOREIGN KEY (article\_id) REFERENCES articles(id) ON DELETE CASCADE

);

```

In the context of web development, a slug is a user-friendly and URL-valid text string used to identify a resource, such as a web page, blog post, or article. A slug is typically derived from the title of the resource, but it is modified to be more concise, meaningful, and easier to read.

For example, suppose you have an article with the title "10 Simple Tips for Running Faster." A typical slug for this article could be "10-simple-tips-for-running-faster." Slugs are important because they make it easier for users to remember and share the URL of the resource, and they also help with search engine optimization (SEO) by making the URL more readable and relevant to the content.

In Laravel, the **slug** field is commonly used to store the URL-friendly version of a resource's title, and it is often generated automatically from the title using a library such as **str\_slug()** or **slugify()**. This field is commonly used in models, controllers, and views to generate links to the resource, and it is often included in the route definition for the resource.