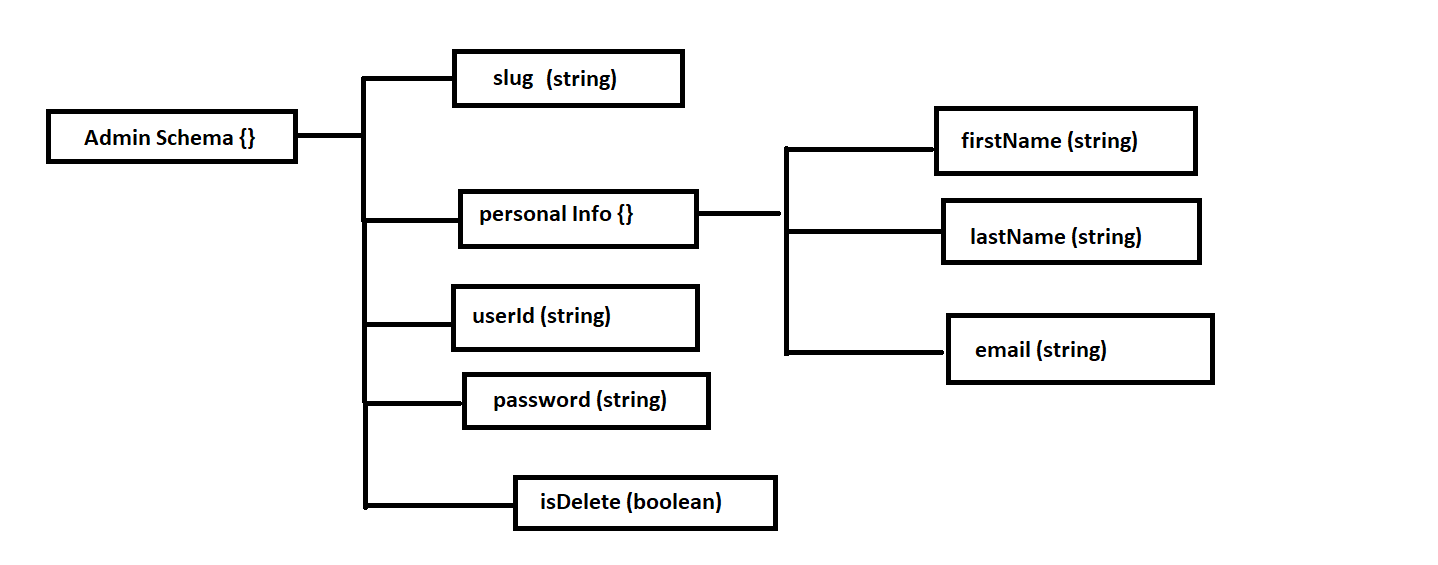
**API Documentation**

TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **SL** | **Schema Name** | **Page Number** |
| 01 | Admin | 3 |
| 02 | Client | 4 - 7 |
| 03 | User | 8 – 11 |
| 04 | Blog |  |
| 05 | Official | 21 - 26 |
| 06 | Comment | 27 – 28 |
| 06 | Contact | 29 - 31 |

**Admin Schema**



**1. Create a new Admin**

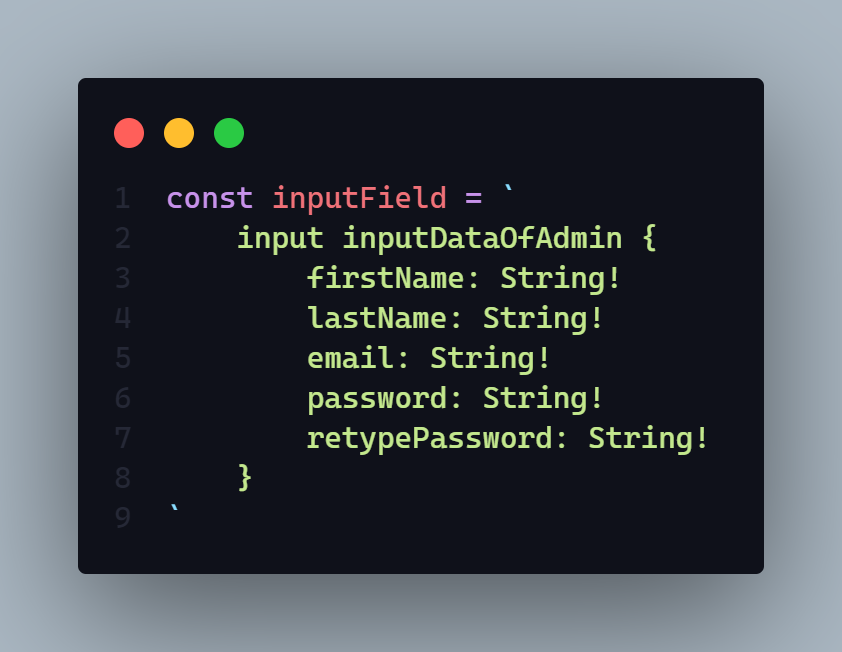
**AP**I: baseUrl/admin eg: http://localhost:3030/admin

**Type: GraphQl**

**Method :** POST / Mutation

**Access**: **:** All

**Body :**



**Params:** N/A

**Query:** N/A

**Response:**



Here

* **message** will sent a response message. Every time it will come mandatory
* **data** will sent create admin data
* **status** will return a http status code here it will sent **406** and **201**

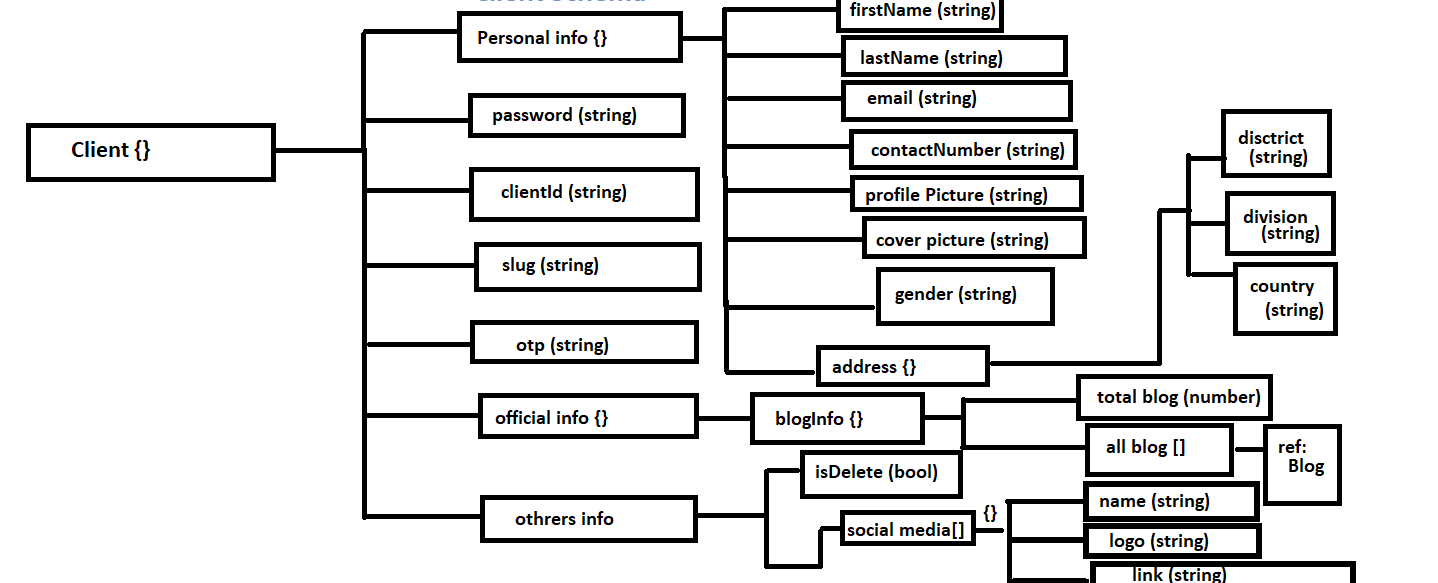
**Demo:**

****

**Description:** If status code is

* **201 =>**
  + Admin successfully created
* **406 =>** 
  + Any type of runtime error
  + Data Joi validation error
  + User Creation Failed
  + Admin Creation Failed
  + Admin Id is not unique
  + Password Hashing Problem
  + Email is exist please try with another email

**Client Schema**



**1. Create a new Client**

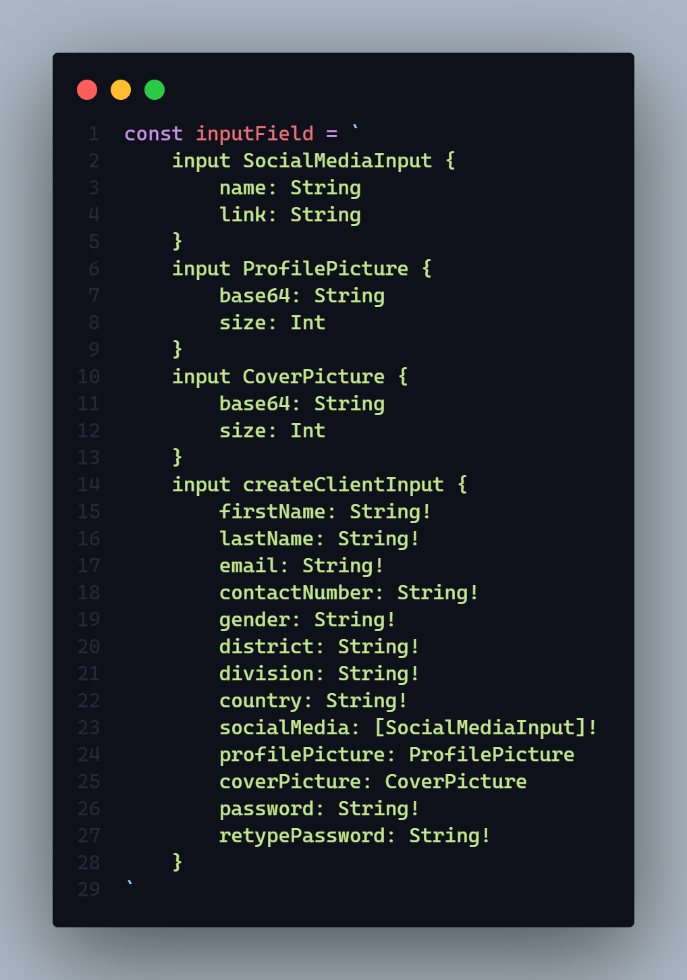
**AP**I: baseUrl/client eg: http://localhost:3030/client

**Type: GraphQl**

**Method :** POST / MUTATION

**Access**: **:** client

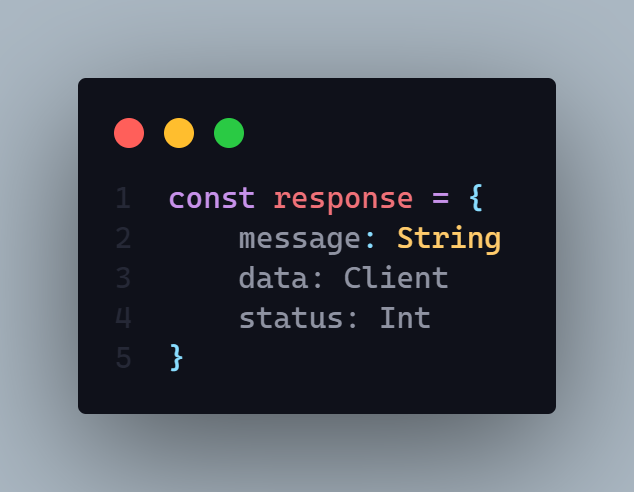
**Body :**



**Params:** N/A

**Query:** N/A

**Response:**

****

**Description:** If status code is

* **201 =>**
  + Client successfully created
* **406 =>** 
  + Any type of runtime error
  + Data Joi validation error
  + User Creation Failed
  + Client Creation Failed
  + Password Hashing Problem
  + Email is exist please try with another email
  + Only Jpg Jpeg Png are allowed
  + Cover picture upload failed
  + Client id is not unique
  + Profile Image upload failed

**Demo:**

****

**2. Delete a client by slug**

**AP**I: baseUrl/client eg: http://localhost:3030/client

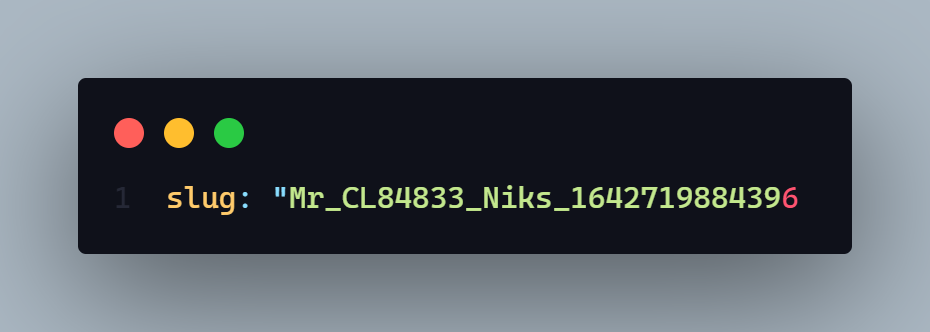
**Type: GraphQl**

**Method :**POST / MUTATION

**Access**: **:** admin, client

**Authorization:** True

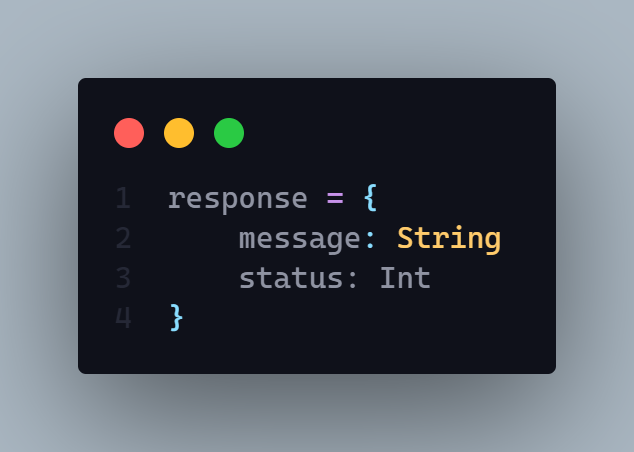
**Body :**



**Params:** N/A

**Query:** N/A

**Response:**

****

**Description:** If status code is

* **202 =>**
  + Client has been successfully deleted
* **406 =>** 
  + Any type of runtime error
* **304 =>**
  + Client delete failed
* **401** =>
  + Permission denied
  + Unauthorized user

As a input we need only the **slug** of respected user.

**Demo:**



**3. Client can see only his activity by year.**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Response:**

**Description:**

**Hints:** It will query how many blog logged in user published within year by month

**4. Update client data by id**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** Full name, email, password, contact number, bio, location

**Hints:** N/A

**Response:**

**Description:**

**5. Client can see only his published or draft blog**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints: Sort by –**

1. Latest
2. A – Z (by title name)
3. Z- A (by title name)
4. Viewers (high to low )

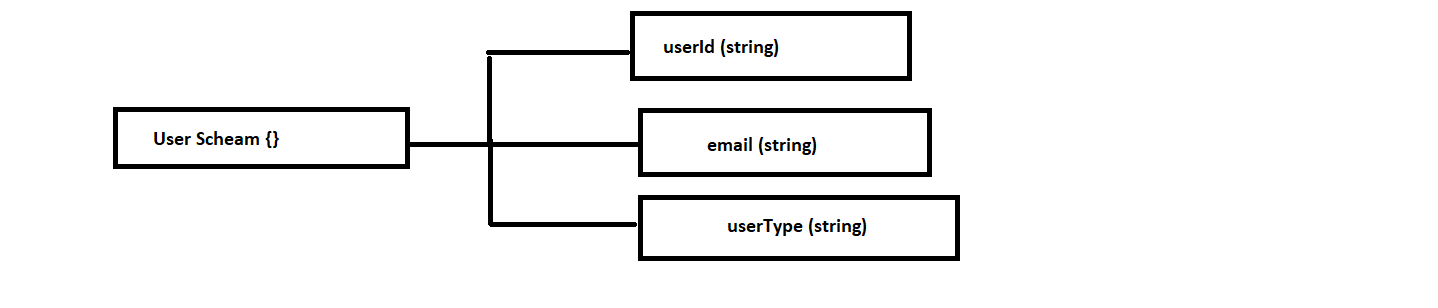
**Pagination Include**

**Data limit include (**Data will return by limit dynamicaly**)**

**Response:**

**Description:**

**User Schema**



**1. Upload or update logged in user profile picture and delete existing one from server.**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**2. Upload or update logged in user cover picture and delte existing one from server.**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**3. User can see his profile**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**4. Login user with email and password**

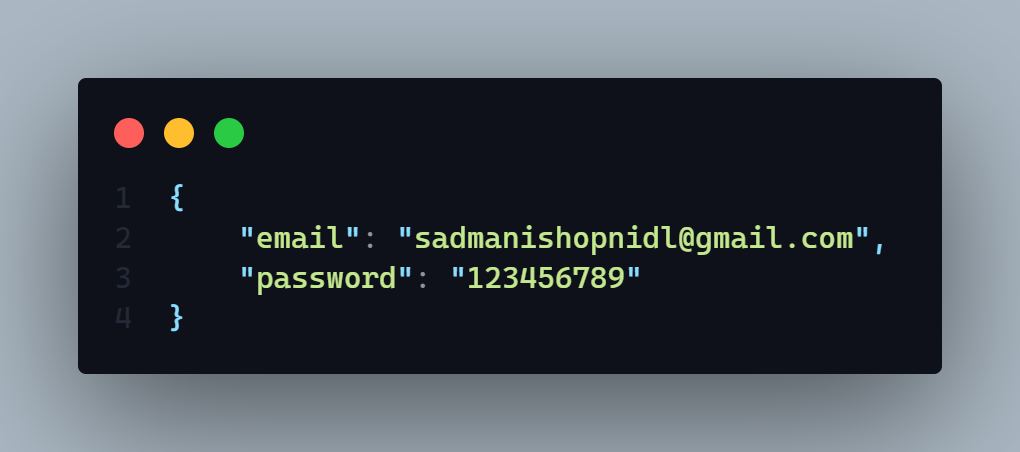
**AP**I: localhost:3030/user/login

**Type: REST**

**Method :** POST

**Access**: **:** ALL

**Body :**



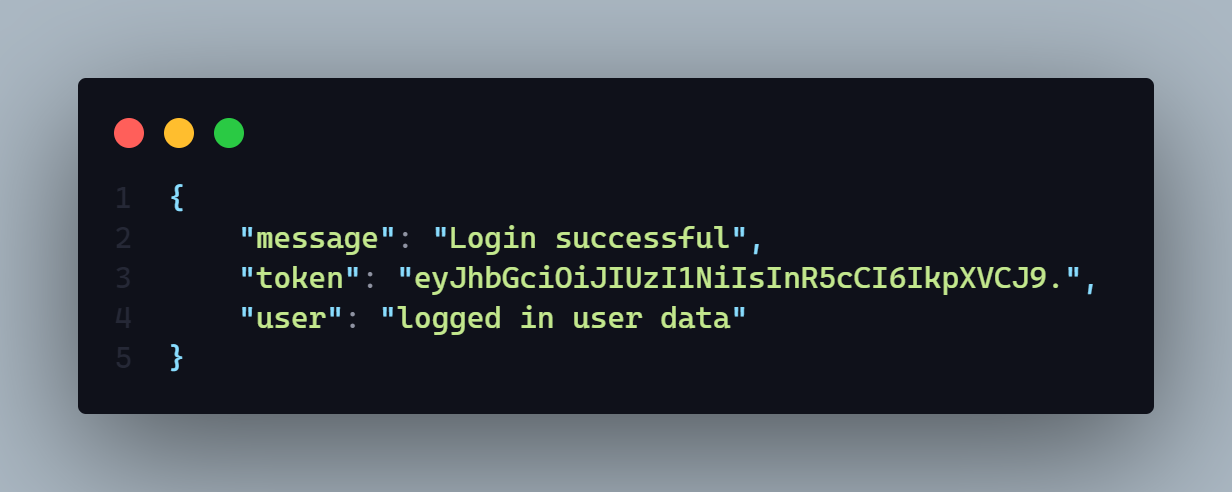
**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** Logged in user by email and password after match the data create a token and set is as auth in the browser cookies.

**Response:**

****

**Description:** If status code is

* **202 =>** Login Successful
* **200 =>** 
  + Token Creation error
  + Password does not match
  + User Not found

It will set the logged in jwt token into cookies by using cookie – parser.

Cookie name is => **auth**

**5. Update logged in user password after verify by rewrite password.**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Response:**

**Description:**

**Hints:** N/A

**6. Create forgot password and change password by verify the email by send a OTP to that email.**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

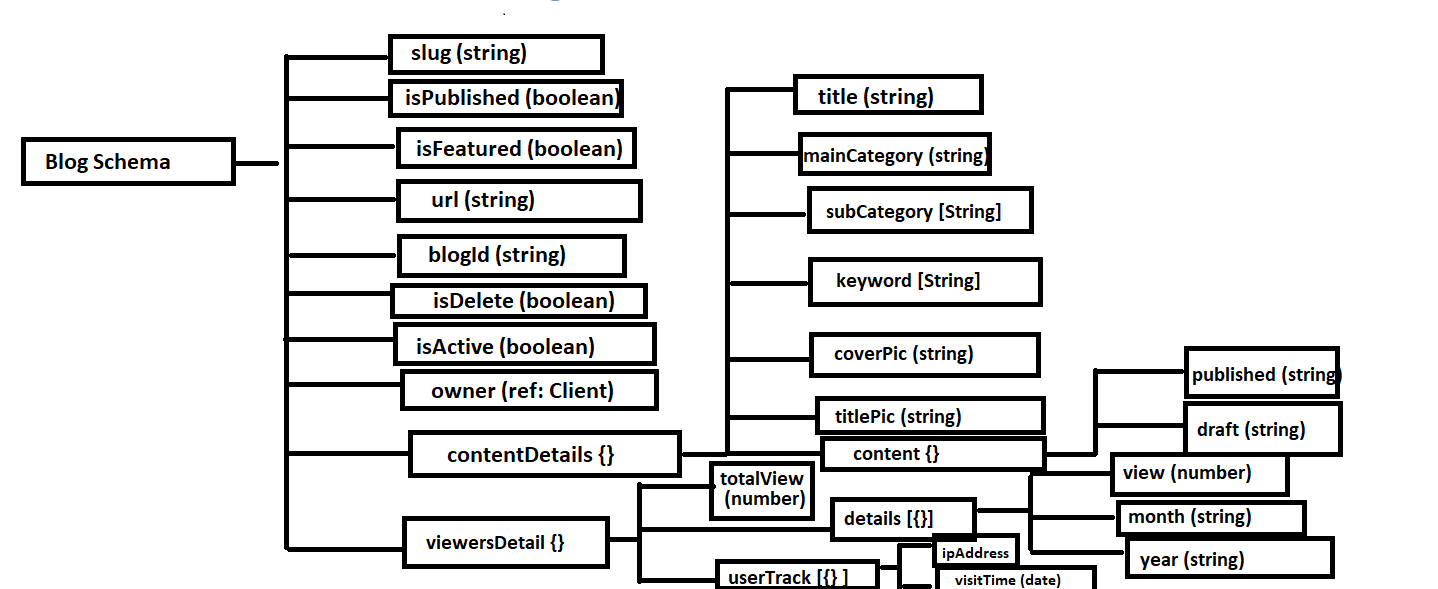
**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**Blog**  **Schema**

****

**1. Get blog by --**

**Query by –**

* **Recent Blog**
* **Top of this month** 
  + **Query by viewer count of current month**
* **Most read blog**
  + **By viewer count all time**
* **All featured blog**
* **By main category**
* **If no query then show all blog**

**Extra Fetcher –**

* **Search 🡺** 
  + **By main category**
  + **By Sub Category**
  + **By Title**
  + **By keyword**
    - **Search input will have to perfectly match**
* **Pagination Include**
* **Filter by =>** 
  + **Sub Category**
  + **Publish Year**
* **Data limit will provide dynamically If not provide then will give all found data**
* **Sorted by –**
  + **Latest**
  + **A – Z** 
    - **By title name in Ascending order**
  + **Z – A** 
    - **By title name in descending order**
  + **View** 
    - **In Descending Order**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**2. Get Two Top of the month blog category by viewers and then get single top blog from each category of the current month.**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**3. Mark Blog as featured**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**4. Count a new view if a user click on individual blog page and stay their at least 60 sec**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**5. Get the all amount of blog available of all categories in the website or official schema**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**6. Get all sub category name available from what blog categories blog we want to get .**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** It will first get all blog data by respected blog category and get all unique subcategory it has found from all blogs.

**Response:**

**Description:**

**7. Get Individual published blog by slug with related blogs also --**

* **Related blog will be query by** 
  + Select blog category
  + Select Blog sub category
  + Keyword

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**8. Save Blog**

* **If user creates new blog then create new one and store data in temporary space**
* **If User updates and save, then only store the new data into existing one that time we should have blog id to find that blog**
* **During update time it will return then updated blog also**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**9. Published a blog**

* **If user creates and direct publish then blog will be create and publish and store content both in temporary or draft and published section**
* **If user published later in the draft section then only update the published section.**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**10. Preview a blog**

* **It generally give individual blog by id** 
  + **It will query blog content from draft section**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**11. Delete an individual blog by blog id**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

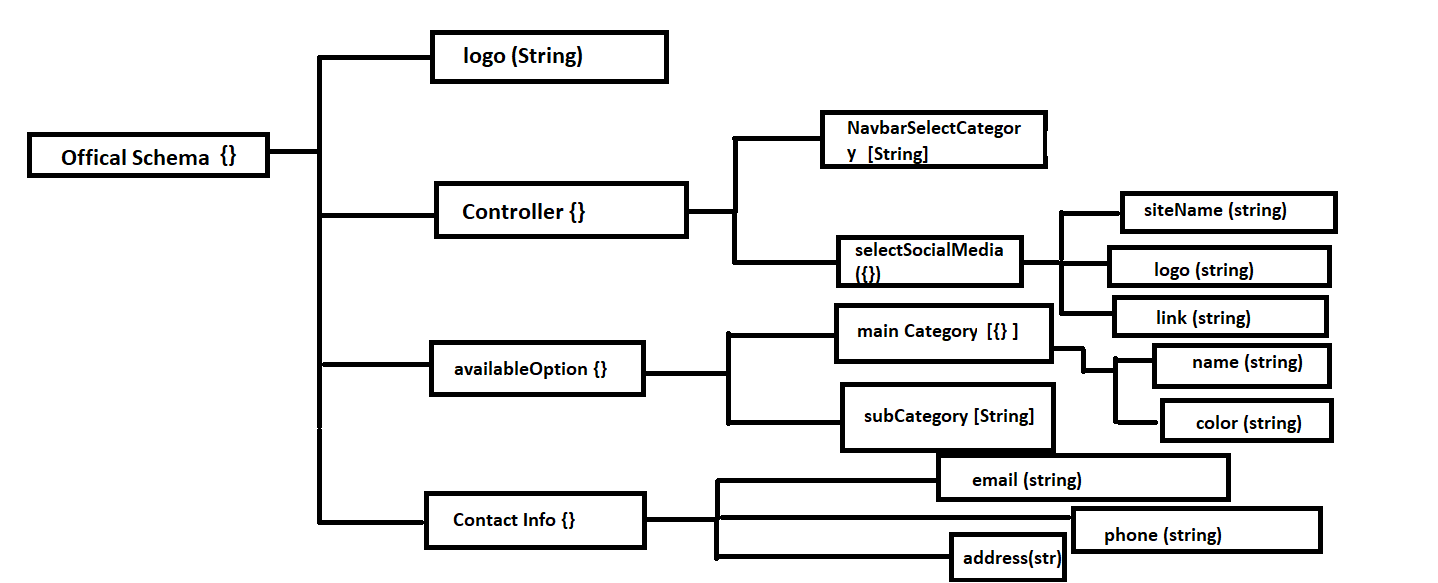
**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**Official**  **Schema**

****

**1. Create an official schema**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**2. Get all Official schema’s every info**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**3. Create available main category with color**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**4. Create available sub category**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**5. Get all main category in ascending order**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**6. Get all sub category in ascending order**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**7. Add contact social media to contact officially**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**8. Update contact social media to contact officially**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**9. Delete contact social media by site name**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

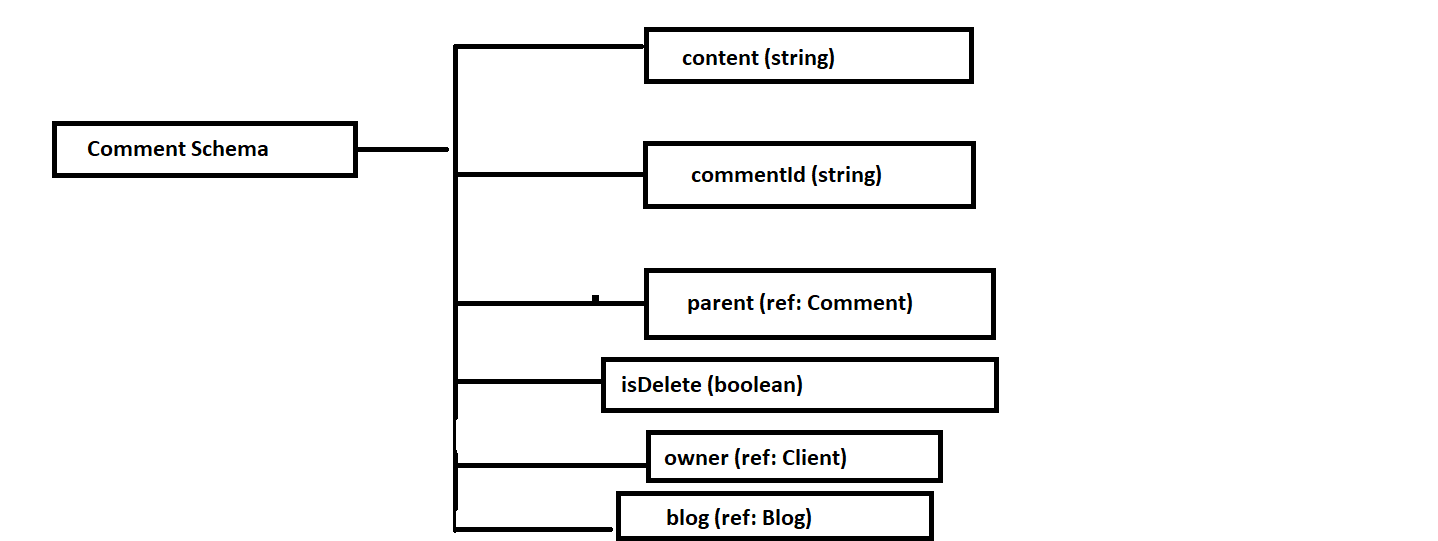
**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**Comment**  **Schema**

****

**1. Create a new comment or reply in the time of reply you need to pass parent comment id**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**2. Update comment by comment id**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**3. Get all comment by blog id name**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

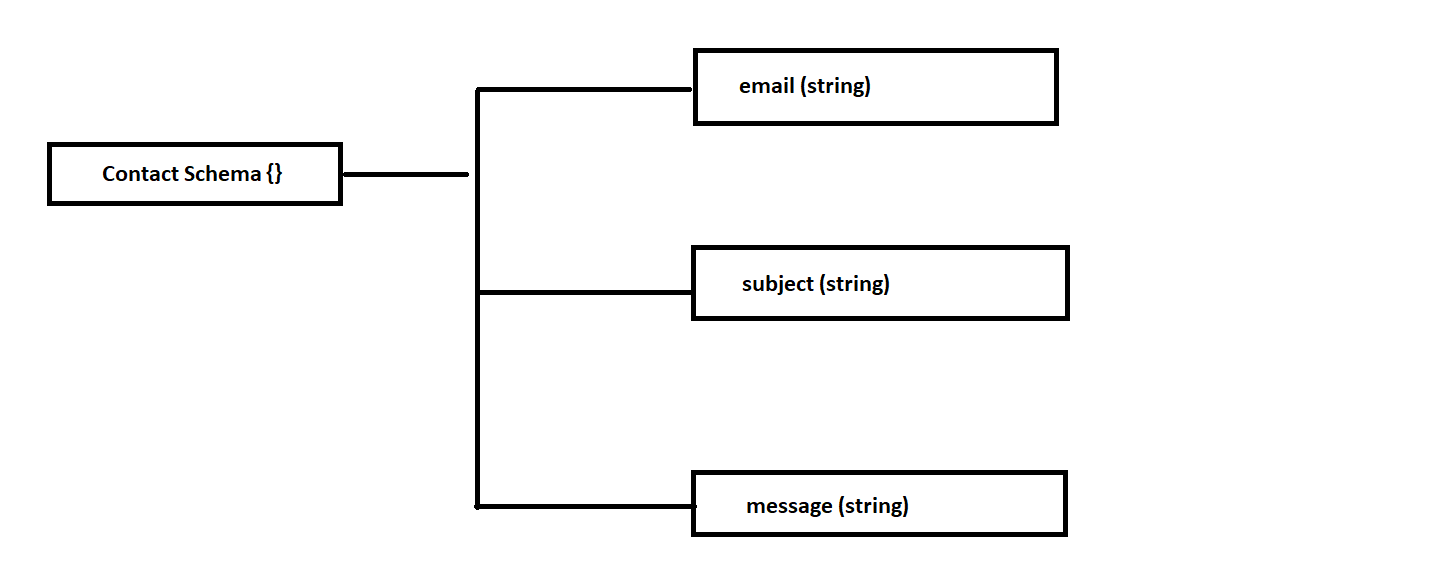
**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

****

**1. Create a new contact**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**2. Get all contact**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**3. Get individual contact by id**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**

**4. Reply to a contact message by provided email**

**AP**I:

**Type:**

**Method :** N/A

**Access**: **:** N/A

**Body :** N/A

**Params:** N/A

**Query:** N/A

**Data:** N/A

**Hints:** N/A

**Response:**

**Description:**