truecaller

Web Developer

Technical assignment

Dear Candidate,

This technical assignment is designed specifically to assess your knowledge about code quality, best practices, design patterns, and written communication skills in the context of web development. Approach this task as if you are already an employee at Truecaller and this is a part of a real project. The output of this assignment will be the base for evaluation followed by a technical interview, if qualified.

This assignment requires you to create a simple front-end for our blog with two different views; one **(1)** for listing multiple posts and one **(2)** for displaying a single post.

The **Design** and **Assets** folders contain the design and image assets. Ensure that the user interface closely matches the provided design.

For the APIs you have to run the server project under the folder `tech-assignment-blog-server`. Install the dependencies in the server project and use the server link with the API's given below to fetch the necessary details. The server runs on port **3000**.

(1) List View

Make a **GET** request to:

`http://localhost:3000/posts/page/1` (or your server's URL)

To get the first page of posts.

Make a **GET** request to:

`http://localhost:3000/categories` (or your server's URL)

To get a list of all available categories.

Posts should be filterable by category.

List all available categories in a select box.

When a category is selected, fetch the posts in that category by making a **GET** request to:

`http://localhost:3000/posts/category/{category}/page/{page}` (or your server's URL)

Each post should be rendered as a card that contains the post's category(ies), thumbnail, title, and date. Format the date so that it shows how long ago the post was published (such as 5 minutes ago, 3 days ago, etc.).

When a card is clicked, it should open that post's detail view (2).

Display at most 20 posts per page and create a paginator that allows older posts to be loaded (use the "page" parameter to get older entries).

(2) Detail View

Make a GET request to (replace `{id}` with the post's ID):

`http://localhost:3000/posts/{id}` (or your server's URL)

To get a single post.

Should display the full post including:

- Header with the post's featured image
- Title
- Author
- Date
- Post content

API Documentation

Get list of categories:

`GET /categories`

- Retrieves all unique categories from the posts.
- **Returns:**
  - `200` - An array of unique categories.
  - `500` - An error message if the categories cannot be loaded.

JavaScript

```
[
  "Technology",
  "Scam Alert",
  "Lifestyle",
  "Health"
]
```

Get paginated posts:

`GET /posts/page/{page}`

- Retrieves paginated blog posts.
- **Parameters:**
  - `page` - The page number to retrieve.
- **Returns:**
  - `200` - An object containing paginated posts, total posts, and total pages.
  - `500` - An error message if the paginated posts cannot be loaded.

Unset

```
{
  "page": 1,
  "limit": 20,
  "totalPosts": 40,
  "totalPages": 2,
  "posts": [
    {
      "id": "1",
      "uid": "post-1",
      "title": "First Blog Post",
      "description": "This is the first blog post.",
      "category": "Technology",
      "author": "John Doe",
      "author_image": "http://localhost:3000/images/author1.jpg",
      "featured_image":
"http://localhost:3000/images/featured1.jpg",
      "first_publication_date": "2024-03-08T13:07:03+0000",
      "last_publication_date": "2024-03-08T13:07:03+0000"
    },
    // ... more posts
  ]
}
```

Get paginated posts by category:

`GET /posts/category/{category}/page/{page}`

- Retrieves paginated blog posts filtered by category.
- **Parameters:**
  - `category` - The category to filter posts by.
  - `page` - The page number to retrieve.
- **Returns:**
  - `200` - An object containing paginated posts, total posts, and total pages for the specified category.
  - `500` - An error message if the paginated category posts cannot be loaded.

Unset


```
{
  "category": "technology",
  "page": 1,
  "limit": 20,
  "totalPosts": 40,
  "totalPages": 2,
  "posts": [
    {
      "id": "1",
      "uid": "post-1",
      "title": "First Blog Post",
      "description": "This is the first blog post.",
      "category": "Technology",
      "author": "John Doe",
      "author_image": "http://localhost:3000/images/author1.jpg",
      "featured_image":
"http://localhost:3000/images/featured2.jpg",
      "first_publication_date": "2024-03-08T13:07:03+0000",
      "last_publication_date": "2024-03-08T13:07:03+0000"
    },
```

```
    // ... more posts
  ]
}
```

Get single post:

`GET /posts/{id}`

- Retrieves a single blog post by ID.
- **Parameters:**
  - `id` - The ID of the post to retrieve.
- **Returns:**
  - `200` - An object containing the post metadata and its body content.
  - `404` - An error message if the post is not found.
  - `500` - An error message if the post cannot be loaded.

Unset

```
{
  "id": "1",
  "uid": "post-1",
  "title": "First Blog Post",
  "description": "This is the first blog post.",
  "category": "Technology",
  "author": "John Doe",
  "author_image": "http://localhost:3000/images/author1.jpg",
  "featured_image": "http://localhost:3000/images/featured1.jpg",
  "first_publication_date": "2024-03-08T13:07:03+0000",
  "last_publication_date": "2024-03-08T13:07:03+0000",
  "html": "<p>This is the content of the first blog post.</p>"
}
```

truecaller

Judging Criteria

Focus on clean, understandable and robust code and low complexity. Use your best judgment and previous experience to utilize the right amount of design patterns and best practices in your code. Pay attention to reusability, extensibility, correctness, state and error handling. The solution should be well structured, but also not over-engineered.

truecaller

Submission

Before submitting, please ensure you have correctly covered every requirement defined in the project specification.

- Submit your solution as a single <firstName>_<lastName>.zip archive file
- The archive should contain only one root folder named <firstName>_<lastName>
- The folder should contain the necessary files to successfully run the project
- The folder should contain a README.md file with a VERY BRIEF explanation about the solution. Here you can explain any assumptions or shortcuts that you have chosen to make.
- Exclude any files and folders you would usually put in the .gitignore file

Please be pragmatic and thank you in advance for your interest in Truecaller and for the time you have spent for us. We sincerely hope to see you as part of our team.

Frequently Asked Questions

Can I use third party libraries?

- Yes, use your best judgement for what is important. Avoid missing out on the opportunity to showcase your CSS and JS skills by relying too heavily on libraries.

Should I use Vue / React / ...?

- You are free to choose whatever framework you see fit to solve the assignment. We work mostly with the Vue.js ecosystem.

Do I have to build a complex UI?

- This is not a requirement, however you are free to do so if you think it will better showcase your skill-set.

Should I write tests?

- It is not a requirement, but you are encouraged to do so