Product Creation & Update with Image Upload

1. Overview

To create or update a product with images:

- **Step 1:** Upload the image(s) to the backend using the /upload endpoint.
- **Step 2:** Use the returned image URL(s) in the images array when calling the GraphQL mutation to create or update the product.

---

2. Image Upload Endpoint

URL:
POST http://localhost:4000/upload

Form Data:

- image (file): The image file to upload.

Response:

```
{
  "url": "/uploads/<filename>"
}
```

- The url is the path to the uploaded image.
- To use it in the frontend, prepend your server address if needed (e.g., http://localhost:4000/uploads/<filename>).

---

3. Creating a Product

Step 1: Upload Image

```
const formData = new FormData();
formData.append('image', file); // file is a File object from an <input type="file">

const res = await fetch('http://localhost:4000/upload', {
  method: 'POST',
  body: formData,
});
const data = await res.json();
const imageUrl = data.url; // e.g., "/uploads/image-12345.jpg"
```

Step 2: Create Product via GraphQL

```
const mutation = `
  mutation CreateProduct($input: ProductInput!) {
    createProduct(input: $input) {
      id
      name
      images
    }
  }
`;

const productInput = {
  name: "Product Name",
  description: "Product Description",
  price: 99.99,
  quantity: 10,
  categoryId: "CATEGORY_ID_HERE",
  images: [imageUrl], // Use the URL from the upload step
};

const res2 = await fetch('http://localhost:4000/graphql', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json', Authorization: `Bearer ${token}` },
  body: JSON.stringify({ query: mutation, variables: { input: productInput } }),
});
const result = await res2.json();
```

---

4. Updating a Product

Step 1: (Optional) Upload New Image

- If you want to update the product image, repeat the upload step above to get a new image URL.

Step 2: Update Product via GraphQL

```
const mutation = `
  mutation UpdateProduct($id: ID!, $input: ProductUpdateInput!) {
    updateProduct(id: $id, input: $input) {
      id
      name
      images
    }
  }
`;

const updateInput = {
  name: "Updated Name",
  // ...other fields as needed
  images: [newImageUrl], // Use the new image URL, or keep the old one(s)
};

const res2 = await fetch('http://localhost:4000/graphql', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json', Authorization: `Bearer ${token}` },
  body: JSON.stringify({ query: mutation, variables: { id: productId, input: updateInput } }),
});
const result = await res2.json();
```

---

5. Notes & Best Practices

- You can upload multiple images by repeating the upload step and collecting all returned URLs in the images array.
- Always use the returned url from the upload endpoint in your product mutations.
- The server serves images at /uploads/<filename>. For display, use the full URL: http://localhost:4000/uploads/<filename>
- The images field in the product is an array of URLs (strings).

---

6. Example UI Flow

1. User selects image(s) in the product form.
2. For each image, upload to /upload and collect the URLs.
3. Submit the product form with all other details and the images array containing the uploaded image URLs.