Essay

From Data to Application: A Hands-on Guide to CRUD Operations with Mongoose and Express.js

ডেটা থেকে অ্যাপ্লিকেশন পর্যন্ত: Mongoose এবং Express.js এর সাথে CRUD অপারেশনের জন্য একটি হাতে-কলমে নির্দেশিকা।

আজকের ডিজিটাল যুগে, ডেটা আমাদের জীবনের প্রতিটি ক্ষেত্রে একটি গুরুত্বপূর্ণ ভূমিকা পালন করে। ব্যক্তিগত তথ্য থেকে শুরু করে ব্যবসায়িক লেনদেন, দক্ষ ব্যবস্থাপনা এবং ডেটা ম্যানিপুলেশন যেকোনো অ্যাপ্লিকেশনের সাফল্যের জন্য অপরিহার্য। ওয়েব অ্যাপ্লিকেশন তৈরির জন্য ব্যবহৃত সবচেয়ে জনপ্রিয় ফ্রেমওয়ার্কগুলির মধ্যে একটি হল Express.js, যা এর সরলতা এবং নমনীয়তার জন্য পরিচিত। Mongoose এর সাথে একত্রিত হলে, MongoDB-এর জন্য একটি শক্তিশালী অবজেক্ট ডেটা মডেলিং (ODM) লাইব্রেরি, বিকাশকারীরা CRUD (Create, Read, Update, Delete) অপারেশনগুলির সম্পূর্ণ সম্ভাবনাকে কাজে লাগাতে পারেন। এই রচনাটির লক্ষ্য হল Mongoose এবং Express.js-এর সাথে CRUD ক্রিয়াকলাপের জন্য একটি হ্যান্ডস-অন গাইড প্রদান করা, অ্যাপ্লিকেশন বিকাশে ডেটা ম্যানিপুলেশনের গুরুত্ব তুলে ধরা।

১। CRUD অপারেশন:

Mongoose এবং Express.js এর স্পেসিফিকেশনে প্রবেশের আগে, আসুন CRUD অপারেশনের মৌলিক ধারণাটি দেখি। CRUD চারটি মৌলিক ফাংশন উপস্থাপন করে যা ডেটাতে সঞ্চালিত হতে পারে: তৈরি করুন, পড়ুন, আপডেট করুন এবং মুছুন (Create, Read, Update, and Delete)। এই ক্রিয়াকলাপগুলি Developer দেরকে ডেটাবেসের সাথে নির্বিঘ্নে যুক্ত করে, অ্যাপ্লিকেশনের প্রয়োজনীয়তাগুলি পূরণ করার জন্য প্রয়োজনীয় ডেটা ম্যানিপুলেট করে।

২। মঙ্গুজ:

Mongoose Node.js অ্যাপ্লিকেশনে MongoDB-এর জন্য একটি জনপ্রিয় অবজেক্ট মডেলিং লাইব্রেরি। এটি MongoDB-এর সাথে ইন্টারঅ্যাক্ট করার জন্য একটি সহজবোধ্য এবং স্বজ্ঞাত উপায় প্রদান করে এবং স্কিমাগুলি সংজ্ঞায়িত করার, বৈধতা সম্পাদন করা এবং Queries সম্পাদন করার প্রক্রিয়াকে সহজ করে তুলে। Mongoose-এর সাহায্যে, ডেভেলপাররা নির্বিঘ্নে জাভাস্ক্রিপ্ট অবজেক্টকে MongoDB নথিতে ম্যাপ করতে পারে, ডেটা ম্যানিপুলেশন সহজতর করে।

৩। Express.js:

অন্যদিকে, Express.js হল Node.js-এর জন্য একটি নমনীয় এবং সংক্ষিপ্ত ওয়েব অ্যাপ্লিকেশন ফ্রেমওয়ার্ক। এটি ওয়েব অ্যাপ্লিকেশন এবং API তৈরির জন্য বৈশিষ্ট্যগুলির (features) একটি শক্তিশালী সেট সরবরাহ করে। Express.js একটি অ্যাপ্লিকেশনের ব্যাকএন্ড গঠনের ভিত্তি হিসাবে কাজ করে, যা মঙ্গুজের মতো অন্যান্য লাইব্রেরির সাথে বিরামহীন একীকরণের অনুমতি দেয়।

৪। Mongoose এবং Express.js ব্যবহার করে CRUD অপারেশন:

Mongoose এবং Express.js-এর শক্তি দেখতে, এই প্রযুক্তিগুলি ব্যবহার করে CRUD ক্রিয়াকলাপ বাস্তবায়ন দেখি।

১) তৈরি করুন (C):

ক্রিয়েট অপারেশন ডাটাবেসে নতুন ডেটা যোগ করে। Mongoose ব্যবহার করে, আমরা একটি স্কিমা সংজ্ঞায়িত করতে পারি যা আমাদের ডেটার গঠনকে উপস্থাপন করে। পাঠানো অনুরোধগুলি পরিচালনা করতে এবং সংজ্ঞায়িত স্কিমার উপর ভিত্তি করে একটি নতুন নথি তৈরি করে ডেটাবেসে ডেটা সংরক্ষণ করতে Express.js ব্যবহার করতে পারি।

২) পড়ুন (R):

রিড অপারেশন আমাদের ডাটাবেস থেকে ডেটা পুনরুদ্ধার করতে দেয়। Mongoose-এর সাহায্যে, আমরা চাহিদার ভিত্তিতে নির্দিষ্ট ডেটা ফিল্টার এবং পুনরুদ্ধার করার জন্য queries সংজ্ঞায়িত করতে পারি। Express.js আমাদের ডেটা পুনরুদ্ধারের জন্য পাঠানো অনুরোধগুলি পরিচালনা করতে এবং ডেটাবেস থেকে অনুরোধ করা তথ্যের সাথে প্রতিক্রিয়া জানাতে সক্ষম করে।

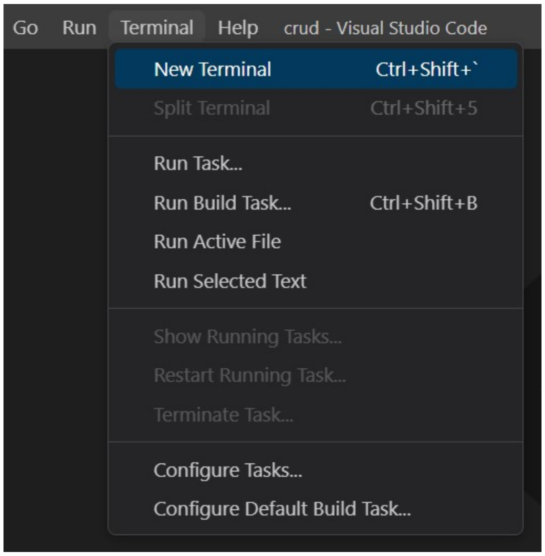
৩) আপডেট (U):
আপডেট অপারেশন ডাটাবেসে বিদ্যমান তথ্য পরিবর্তন করে। শক্তিশালী প্রক্রিয়ার মাধ্যমে মঙ্গুজ নির্দিষ্ট অবস্থার উপর ভিত্তি করে নথিগুলি আপডেট করে এবং আপডেটগুলি দক্ষ ভাবে সম্পাদন করে। Express.js তাৎক্ষণিক (incoming) আপডেটের অনুরোধগুলি পরিচালনার সুবিধা দেয়, নিশ্চিত করে যে ডাটাবেসের পছন্দসই নথিতে যথাযথ পরিবর্তন করা হয়েছে।

৪) মুছুন (D):
ডিলিট অপারেশন আমাদের ডাটাবেস থেকে ডেটা অপসারণে ব্যবহার হয়। মঙ্গুজ নির্দিষ্ট শর্তের উপর ভিত্তি করে সহজেই নথি মুছে ফেলতে পারে। Express.js এর মাধ্যমে, আমরা ডেটা মুছে ফেলার জন্য পাঠানো অনুরোধগুলি পরিচালনা করতে পারি এবং সংশ্লিষ্ট নথিগুলি ডাটাবেস থেকে সরানো হয়েছে তা নিশ্চিত করতে পারি।

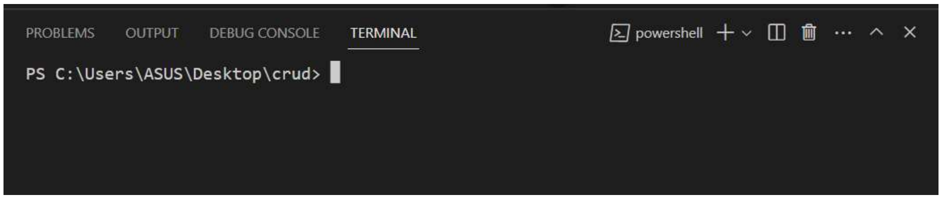
৫) সর্বোত্তম অনুশীলন এবং নিরাপত্তা:
CRUD ক্রিয়াকলাপগুলি বাস্তবায়ন করার সময়, সর্বোত্তম অনুশীলনগুলি অনুসরণ করা এবং নিরাপত্তার দিকগুলি বিবেচনা করা অপরিহার্য। অননুমোদিত অ্যাক্সেস বা দূষিত ক্রিয়াকলাপ রোধ করার জন্য কিছু অনুশীলনের মধ্যে ইনপুট বৈধতা, স্যানিটাইজেশন, প্রমাণীকরণ এবং অনুমোদনের প্রক্রিয়া অন্তর্ভুক্ত রয়েছে। এই অনুশীলনগুলি মেনে চলা নিশ্চিত করে যে অ্যাপ্লিকেশনটি শক্তিশালী এবং সুরক্ষিত।

ধাপ ১: ইনস্টলেশন এবং সেটআপ:

- Visual Studio Code ইনস্টল করুন।
- Node.js এবং MongoDB ইনস্টল করুন।
- আপনার project জন্য একটি নতুন ডিরেক্টরি বা folder তৈরি করুন এবং Folder টি Visual Studio Code এ open করুন।
- Terminal > New Terminal বা (Ctrl+Shift+`) Terminal Run করুন। নিচের চিত্র লক্ষ করুন।
- Terminal এ ”`npm init --y`” লিখে Enter press করুন।
- package.json ফাইল তৈরি হবে।

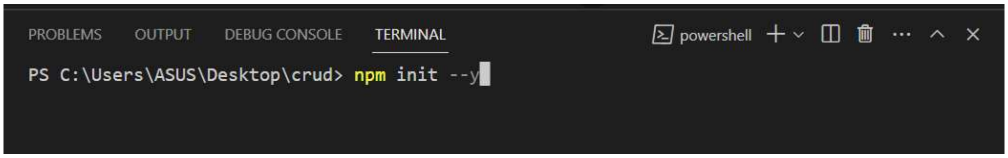


চিত্র-১: Menu



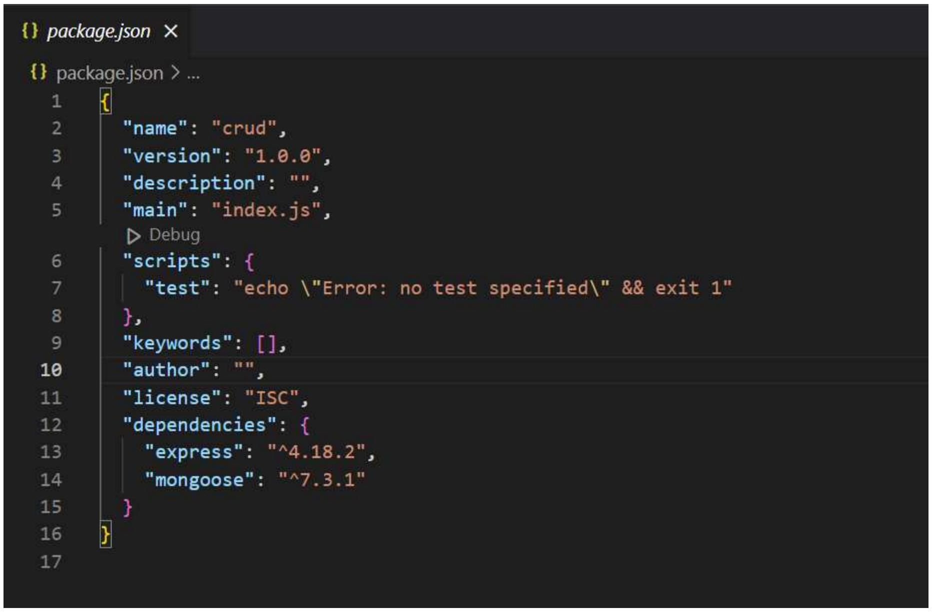
```
PS C:\Users\ASUS\Desktop\crud>
```

চিত্র-২: Terminal



```
PS C:\Users\ASUS\Desktop\crud> npm init --y
```

চিত্র-৩: `npm init --y`



```
{
  "name": "crud",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "express": "^4.18.2",
    "mongoose": "^7.3.1"
  }
}
```

চিত্র-৪: package.json

- নিম্নলিখিত কমান্ডটি চালিয়ে প্রয়োজনীয় নির্ভরতাগুলি(dependencies) ইনস্টল করুন: `npm i express mongoose` চিত্র।

```
PS C:\Users\ASUS\Desktop\crud> npm i express mongoose
```

চিত্র-৫: ইনস্টল Dependencies

ধাপ ২: Express.js অ্যাপ্লিকেশন সেট আপ করা:

- create.js নামে একটি নতুন ফাইল তৈরি করুন এবং Visual Studio Code এ open করুন।
- create.js প্রয়োজনীয় মডিউল:

```
const express = require('express');
const app = express();
```

ধাপ ৩: Mongoose মডিউল এবং MongoDB ডাটাবেসের সাথে সংযোগ:

- Mongoose মডিউল
- MongoDB সংযোগ স্ট্রিং 'mongodb://127.0.0.1:27017/crud' (ডেটাবেসের নাম: crud)

```
const mongoose = require('mongoose');

mongoose.connect('mongodb://127.0.0.1:27017/crud', {
useNewUrlParser: true, useUnifiedTopology: true });
```

ধাপ ৪: Middleware functions

- এই মিডলওয়্যার গুলো Express.js-এ সাধারণত বডি থেকে পাঠানো রিকোয়েস্ট পার্স করতে ব্যবহৃত হয়।

  ```
  app.use(express.json());
  app.use(express.urlencoded({ extended: false }));
  ```

ধাপ ৫: স্কিমা এবং মডেল:

```
const Schema = mongoose.Schema;
const mySchema = new Schema({
     name: String,
     age: Number,
     email: String
});

const MyModel = mongoose.model('mymodels', mySchema);
```

ধাপ ৬: রেকর্ড তৈরি করার জন্য রুট:

- এই রুটটিতে JSON ব্যাবহার করে ডেটা পাঠানো হয়েছে।

```
app.post('/data', (req, res) => {
    const newData = req.body;
    MyModel.create(newData)
      .then((data) => {
        res.status(201).json(data);
      })
      .catch((error) => {
        res.status(500).json({ error: 'An error occurred' });
      });
  });
```

ধাপ ৭: সার্ভার Run করা:

সার্ভার Run করার জন্য create.js ফাইলের শেষে নিম্নলিখিত কোড গুলো যোগ করুন:

```
const port = 8000;
  app.listen(port, () => {
    console.log(`Server is running on port ${port}`);
  });
```

রেকর্ড দেখা, আপডেট করা, মুছে ফেলার জন্য কোড:

প্রতিটা ফাইলের জন্য ধাপ ২ থেকে ধাপ ৭ পযন্ত অনুসরণ করুন শুধু ধাপ ৬ পরিবরতন হবে।

ফাইলের নাম: read.js, singledata.js, update.js এবং delete.js

সমস্ত রেকর্ড দেখার জন্য রুট:

```
 app.get('/data', (req, res) => {
    MyModel.find()
      .then((data) => {
        res.json(data);
      })
      .catch((error) => {
        res.status(500).json({ error: 'An error occurred' });
      });
  });
```

আইডি দ্বারা একটি নির্দিষ্ট রেকর্ড দেখার জন্য রুট:

```
app.get('/data/:id', (req, res) => {
    const id = req.params.id;
    MyModel.findById(id)
      .then((data) => {
        res.json(data);
      })
      .catch((error) => {
        res.status(500).json({ error: 'An error occurred' });
      });
  });
```

একটি নির্দিষ্ট রেকর্ড আপডেট করার জন্য রুট:

```
app.put('/data/:id', (req, res) => {
    const id = req.params.id;
    const newData = req.body;
    MyModel.findByIdAndUpdate(id, newData, { new: true })
      .then((data) => {
        res.json(data);
      })
      .catch((error) => {
        res.status(500).json({ error: 'An error occurred' });
      });
  });
```

একটি নির্দিষ্ট রেকর্ড মুছে ফেলার জন্য রুট:

```
app.delete('/data/:id', (req, res) => {
    const id = req.params.id;
    MyModel.findByIdAndDelete(id)
      .then(() => {
        res.json({ message: 'Record deleted successfully' });
      })
      .catch((error) => {
        res.status(500).json({ error: 'An error occurred' });
      });
  });
```

ধাপ ৮: Endpoints পরীক্ষা করা:

- আপনার MongoDB সার্ভার Run করুন।
- আপনার Express.js অ্যাপ্লিকেশন Run করতে Terminal কমান্ড `node create.js`
- নমুনা ডেটা সহ CRUD Endpoints (`POST`, `GET`, `PUT`, `DELETE`) পরীক্ষা করতে পোস্টম্যান টুল ব্যবহার করুন।

***`crud` নামে MongoDB তে আগে ডেটাবেস তৈরি করে নিতে হবে।

সুবিধার জন্য Github repository link: https://github.com/mehedinewazsharif/Essaycrud

টুল্স (Tools)

Mongoose এবং Express.js-এর সাথে CRUD অপারেশনগুলির Development এবং পরীক্ষার (testing) জন্য আপনার প্রয়োজনীয় Tools এর তালিকা এখানে রয়েছে:

১। Node.js: একটি জাভাস্ক্রিপ্ট রানটাইম পরিবেশ (runtime environment) তৈরি করে যা আপনাকে সার্ভার-সাইডে জাভাস্ক্রিপ্ট কোড চালানোর অনুমতি দেয়। আপনি অফিসিয়াল ওয়েবসাইট থেকে Node.js ডাউনলোড করতে পারেন: https://nodejs.org/

২। MongoDB: একটি NoSQL ডাটাবেস, খুবই সহজ (flexible) ভাবে ডেটা সংরক্ষণ করে যেমন JSON format । আপনি অফিসিয়াল ওয়েবসাইট থেকে MongoDB কমিউনিটি সংস্করণ ডাউনলোড করতে পারেন: https://www.mongodb.com/try/download/community

৩। কোড এডিটর: আপনার কোড লেখার জন্য একটি কোড এডিটর বা ইন্টিগ্রেটেড ডেভেলপমেন্ট এনভায়রনমেন্ট (IDE)। কিছু জনপ্রিয় এডিটর বা IDE-র মধ্যে রয়েছে ভিজ্যুয়াল স্টুডিও কোড, সাব্লাইম টেক্সট, এটম বা ওয়েবস্টর্ম।

৪। টার্মিনাল: কমান্ড চালানো এবং আপনার প্রজেক্ট পরিচালনা করার জন্য একটি কমান্ড-লাইন ইন্টারফেস (CLI) টুল। এটি আপনার কোড এডিটরের অন্তর্নির্মিত টার্মিনাল বা একটি পৃথক টার্মিনাল অ্যাপ্লিকেশন হতে পারে।

৫। পোস্টম্যান: API পরীক্ষা এবং নথিভুক্ত করার জন্য একটি জনপ্রিয় টুল। এটি আপনাকে POST, GET, PUT, DELETE সহ HTTP অনুরোধ পাঠাতে এবং প্রতিক্রিয়া পেতে দেয়। আপনি অফিসিয়াল ওয়েবসাইট থেকে পোস্টম্যান ডাউনলোড করতে পারেন: https://www.postman.com/

৬। ব্রাউজার: আপনার অ্যাপ্লিকেশনের কার্যকারিতা পরীক্ষা করতে এবং আউটপুট দেখার জন্য একটি ওয়েব ব্রাউজার। জনপ্রিয় ব্রাউজার গুলির মধ্যে রয়েছে গুগল ক্রোম, মোজিলা ফায়ারফক্স, সাফারি, বা মাইক্রোসফ্ট এজ।

এই টুল গুলি আপনাকে Mongoose এবং Express.js এর সাথে আপনার CRUD operation গুলি বিকাশ (develop), পরীক্ষা (test) এবং ডিবাগ করার জন্য প্রয়োজনীয় পরিবেশ এবং উপযোগিতা (utilities) প্রদান করবে।

Endpoints Test

পোস্টম্যান ব্যবহার করে কীভাবে CRUD শেষ পয়েন্টগুলি পরীক্ষা করব তার একটি উদাহরণ এখানে রয়েছে:

১। একটি নতুন রেকর্ড তৈরি করা (POST):

- অনুরোধের পদ্ধতিটি POST-এ সেট করুন।

- অনুরোধের URLটি `http://localhost:8000/data`-এ সেট করুন।

- বডি বিভাগে, "Raw" নির্বাচন করুন এবং ডেটা বিন্যাসটিকে (format) JSON এ সেট করুন।

- JSON ফর্ম্যাটে নতুন রেকর্ডের জন্য ডেটা প্রদান করুন, উদাহরণস্বরূপ:

```
{
  "name": "Mehedi Newaz Sharif",
  "age": 33,
  "email": "mnsinfo98@gmail.com"
}
```

- অনুরোধ জমা দিতে " Send" বোতামে ক্লিক করুন। প্রতিক্রিয়াতে নতুন রেকর্ড তৈরি হবে।

২। সমস্ত রেকর্ড পুনরুদ্ধার করা (GET):

- অনুরোধের পদ্ধতিটি GET-এ সেট করুন।

- অনুরোধের URLটি ` http://localhost:8000/data`-এ সেট করুন।

- অনুরোধ জমা দিতে " Send" বোতামে ক্লিক করুন। প্রতিক্রিয়াতে সমস্ত রেকর্ডের একটি অ্যারে পাওয়া যাবে।

৩। ID দ্বারা একটি নির্দিষ্ট রেকর্ড পুনরুদ্ধার করা (GET):

- অনুরোধের পদ্ধতিটি GET-এ সেট করুন।

- অনুরোধের URLটিকে `http://localhost: 8000/data/:id`-এ সেট করুন, যেখানে `:id` হল আপনি যে রেকর্ডটি পুনরুদ্ধার করতে চান তার আইডি।

- অনুরোধ জমা দিতে " Send" বোতামে ক্লিক করুন। নির্দিষ্ট আইডি সহ রেকর্ড পাওয়া যাবে।

৪। একটি নির্দিষ্ট রেকর্ড আপডেট করা (PUT):

- অনুরোধের পদ্ধতিটি PUT -এ সেট করুন।

- অনুরোধ URLটিকে `http://localhost: 8000/data/{id}`-এ সেট করুন, যেখানে `{id}` হল সেই রেকর্ডের ID যা আপনি আপডেট করতে চান।

- বডি বিভাগে, "Raw" নির্বাচন করুন এবং ডেটা বিন্যাসটিকে (format) JSON এ সেট করুন।

- JSON ফর্ম্যাটে রেকর্ডের জন্য আপডেট করা ডেটা প্রদান করুন, উদাহরণস্বরূপ:

```
{
  "name": "Sharif",
  "age": 30,
  "email": "mnsinfo98@gmail.com"
}
```

- অনুরোধ জমা দিতে " Send" বোতামে ক্লিক করুন। প্রতিক্রিয়া রেকর্ড আপডেট ।

৫। একটি নির্দিষ্ট রেকর্ড মুছে ফেলা (DELETE):

- মুছে ফেলার জন্য অনুরোধের পদ্ধতি সেট করুন।

- অনুরোধের URLটিকে `http://localhost: 8000/data/:id`-এ সেট করুন, যেখানে `:id` হল আপনি যে রেকর্ডটি মুছতে চান তার আইডি।

- অনুরোধ জমা দিতে " Send" বোতামে ক্লিক করুন। প্রতিক্রিয়াটি নির্দেশ করবে যে রেকর্ডটি মুছে ফেলা হয়েছে।

এই পদক্ষেপগুলি অনুসরণ করে, আপনি পোস্টম্যান ব্যবহার করে CRUD এন্ডপয়েন্ট পরীক্ষা করতে পারেন এবং যাচাই করতে পারেন যে অপারেশনগুলি প্রত্যাশিত হিসাবে কাজ করছে। আপনি যে রেকর্ডটি পুনরুদ্ধার করতে, আপডেট করতে বা মুছতে চান তার আসল ID দিয়ে `:id` প্রতিস্থাপন করতে ভুলবেন না।