

PinkBox

PinkBox is e-commerce movie streaming service to allow its customers to stream any movies at a discounted price. The challenge, at present time, is that existing movie streaming providers, Netflix, Hulu, AmazonPrime etc. require monthly subscription irrespective of how many movies one can watch. Each provider is contracted to host selected set of movies. MovieMart, virtual store, negotiated with all providers at a discounted price for each view and provides a unified UI to select any movie from any of the streaming providers.

PinkBox is built using MERN tech stack. It has three projects, each to support frontend, backend and Inventory.

Frontend: It is built on using react.js and makes calls to endpoints exposed by the backend.

Backend: It is hosted on Express server which is connected to MongoDB hosted on Google Cloud. Express server exposes various endpoints for frontend to all to get data in json format from MongoDB.

Inventory: This project is meant for loading movies to MongoDB database.

Backend Design:

MongoDB is chosen database to persist data. There are two schemas (model) to support this e-commerce application.

Schema 1: movies

To store all the movies

Schema 2: users

To store users and their shopping cart details

MongoDB hosting:


MongoDB Atlas is chosen as it's a fully managed cloud database service provided by MongoDB. Having MongoDB on the cloud enables everyone to access the database on their laptop and loading all the movies. Any changes require redo every time for everyone in the project team. Any changes made to the database or the data is a one time task and it is available to all the team members. It saves time and effort in delivering the project.

Deploy MongoDB database on the cloud.

MongoDB Atlas on the cloud is a free service with limited space. Google is the chosen cloud provider and therefore to deploy the database on Google Cloud, a Gmail account is required.

Step 1: Deploy MongoDB database

The instructions to deploy the MongoDB Atlas on the cloud is available at <https://www.mongodb.com/atlas>

 MongoDB

Products ▾

Resources ▾

Solutions ▾

Company ▾

Pricing

🔍

🌐 Eng ▾

Support

Sign In

Try Free

MONGODB ATLAS

MongoDB Atlas.

The multi-cloud developer data platform.

An integrated suite of cloud database and data services to accelerate and simplify how you build with data.

Try Free

Document model benefits →

Cluster

Read Write

Connections

Network In Network Out

Disk Usage

Serverless

Read Write

Connections

Network In Network Out

Disk Usage

Connect To Your Database →

Click on “Try Free”



Sign up

See what Atlas is capable of for free

 Sign up with Google

First Name*

Last Name*

Company

Email*

Password*



☐ I agree to the [Terms of Service](#) and [Privacy Policy](#).

Create your Atlas account


Click on "Sign up with Google"

Sign in with Google


Sign in to mongodb.com

By continuing, Google will share your name, email address, language preference, and profile picture with mongodb.com. See mongodb.com's [Privacy Policy](#) and [Terms of Service](#).

You can manage Sign in with Google in your [Google Account](#).

 aarithir@gmail.com

[Cancel](#) [Continue](#)

English (United States) 

[Help](#) [Privacy](#) [Terms](#)

Click on "Continue"



Accept Privacy Policy & Terms of Service

Please acknowledge the following terms and conditions to finish creating your account.

☒ I accept the [Privacy Policy](#) and the [Terms of Service](#)

[Cancel Signup](#)

[Submit](#)

Accept and submit



Welcome to Atlas. Let's build something great.

Help us tailor your experience by taking a minute to answer the questions below.

GETTING TO KNOW YOU

What is your primary goal?

Learn MongoDB

How long have you been developing software with MongoDB?

I've never developed software with MongoDB before

GETTING TO KNOW YOUR PROJECT

What programming language are you primarily building on MongoDB with?

JavaScript / Node.js

What type(s) of data will your project use?

You can choose as many as you want

Not sure... X

Will your application include any of the following architectural models?

You can choose as many as you want

Not sure... X

Finish

Answer the questions and “Finish”

Deploy your cluster

Use a template below or set up advanced configuration options. You can also edit these configuration options once the cluster is created.

☐ **M10** **\$0.08/hour**

Dedicated cluster for development environments and low-traffic applications.

STORAGE	RAM	vCPU
10 GB	2 GB	2 vCPUs

☐ **Flex** **From \$0.011/hour**
Up to \$30/month

For application development and testing, with on-demand burst capacity for unpredictable traffic.

STORAGE	RAM, vCPU	OPS/SEC
5 GB	Shared	0 - 500

☒ **Free**

For learning and exploring MongoDB in a cloud environment.

STORAGE	RAM, vCPU	OPS/SEC
512 MB	Shared	0 - 100

✓ **Free forever!** Your free cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

Configurations

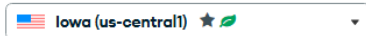
Name

You cannot change the name once the cluster is created.

Provider



Region



★ Recommended ⓘ 🌿 Low carbon emissions ⓘ

Quick setup

- ☒ Automate security setup ⓘ
- ☒ Preload sample dataset ⓘ

Select "Free" and "Google Cloud"
Select the Region and leave the rest to Default
click on "Create Deployment" and
Note that storage allowed for free is only 512MB.

Connect to Cluster0

1

2

3

Set up connection securityChoose a connection methodConnect

You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. [Read more](#)

1. Add a connection IP address

✓ Your current IP address (73.148.221.151) has been added to enable local connectivity. Only an IP address you add to your Access List will be able to connect to your project's clusters. Add more later in [Network Access](#).

2. Create a database user

This first user will have [atlasAdmin](#) permissions for this project.

We autogenerated a username and password. You can use this or create your own.

i You'll need your database user's credentials in the next step. Copy the database user password.

Username

Password
 HIDE

Copy

Create Database User

Close

Choose a connection method

Pick username (pinkbox) and password (pinkbox)
Click on "Create Database User"

Connect to Cluster0



1

Set up connection security

2

Choose a connection method

3

Connect

You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. [Read more](#)

1. Add a connection IP address

✓ Your current IP address (73.148.221.151) has been added to enable local connectivity. Only an IP address you add to your Access List will be able to connect to your project's clusters. Add more later in [Network Access](#).

2. Create a database user

✓ A database user has been added to this project. Create another user later in [Database Access](#).

You'll need your database user's credentials in the next step.

Close

Choose a connection method

Click on "Close"

Atlas aarithi's Org... Access Manager Billing

Project 0 Data Services Charts

Overview

DATABASE
Clusters

SERVICES
Atlas Search
Stream Processing
Triggers
Migration
Data Federation

SECURITY
Quickstart
Backup
Database Access
Network Access
Advanced

New On Atlas 4
Goto

Your organization does not have a designated security contact. Add an Atlas Security Contact in [Organization Settings](#) to receive security-related notifications.

AARITHI'S ORG - 2025-02-02 > PROJECT 0

Overview

Clusters Create cluster ...

Cluster0

✓ Sample Dataset successfully loaded! [Browse this collection.](#)

[Connect](#) [Edit configuration](#)

Looking to interact with Atlas using CLI/laC tools? [View Infrastructure as Code options.](#)

[Browse collections](#) → [View monitoring](#) →

[+ Add Tag](#)

Application Development Get connection string ...

JavaScript / Node.js

Connect your application to your database deployments with the official MongoDB libraries.
[Start with a Node.js Sample App](#)

Step 2: To get the connection string to connect to the database just created,
Click “Connect”

Connect to Cluster0



Connect to your application



Drivers

Access your Atlas data using MongoDB's native drivers (e.g. Node.js, Go, etc.)



Access your data through tools



Compass

Explore, modify, and visualize your data with MongoDB's GUI



Shell

Quickly add & update data using MongoDB's Javascript command-line interface



MongoDB for VS Code

Work with your data in MongoDB directly from your VS Code environment



Atlas SQL

Easily connect SQL tools to Atlas for data analysis and visualization



Go Back

Close

Click on "Compass"

Connect to Cluster0

✓

✓

3

Set up connection securityChoose a connection methodConnect

Connecting with MongoDB Compass

I don't have MongoDB Compass installed

I have MongoDB Compass installed

1. Choose your version of Compass

1.38 or later

See your Compass version in "About Compass"

2. Copy the connection string, then open MongoDB Compass

Use this connection string in your application

mongodb+srv://pinkbox:<db_password>@cluster0.jf5f4.mongodb.net/

Replace **<db_password>** with the password for the **pinkbox** user. Ensure any options are [URL encoded](#). You can edit your database user password in [Database Access](#).

RESOURCES

[Connect with Compass](#)

[Access your Database Users](#)

[Import and Export Data](#)

[Troubleshoot Connections](#)

Go Back

Done

Copy the connection string:

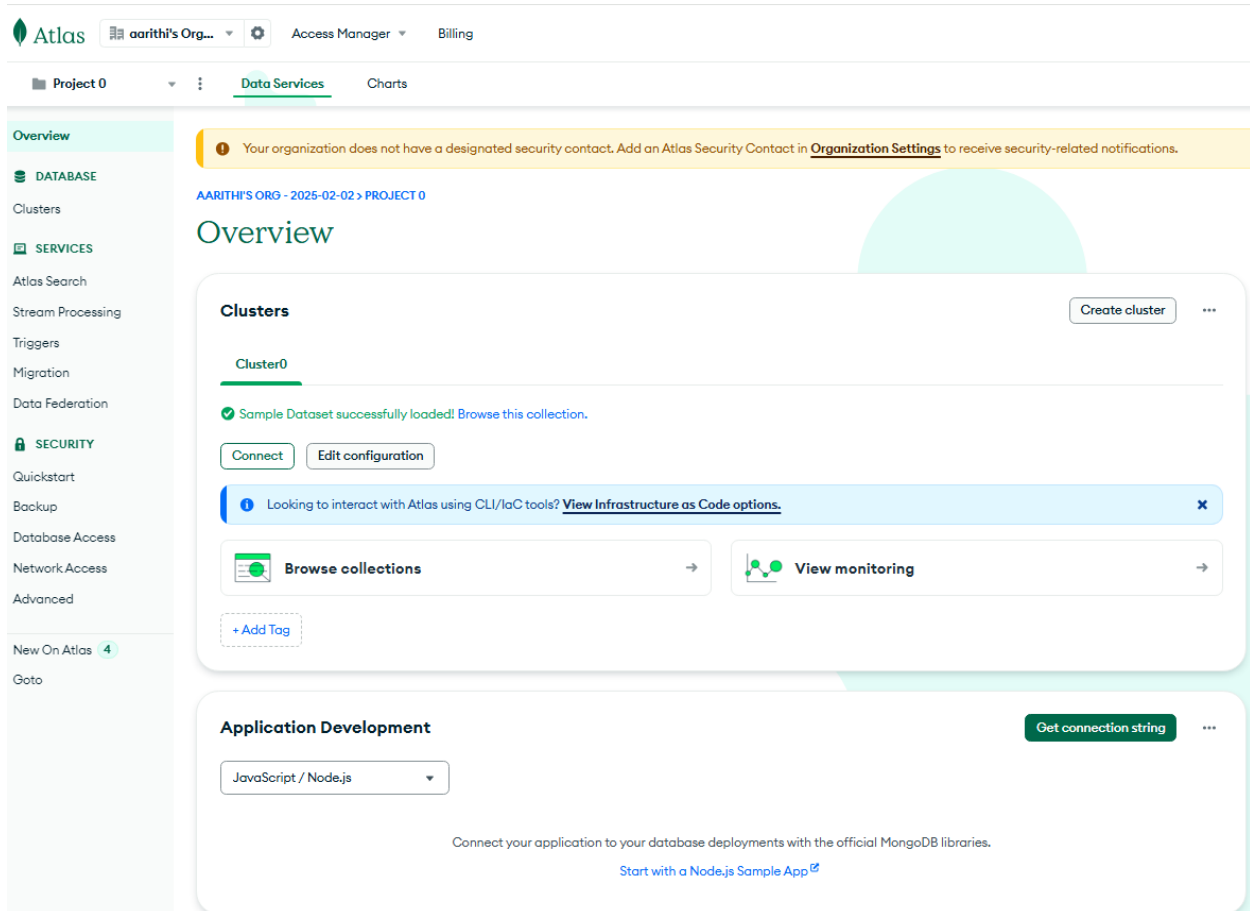
```
mongodb+srv://pinkbox:<db_password>@cluster0.jf5f4.mongodb.net/
```

This is the string to be used when connecting the MongoDB database in Express server.
Replace `<d_password>` with the actual password "pinkbox"

```
mongodb+srv://pinkbox:pinkbox@cluster0.jf5f4.mongodb.net/
```

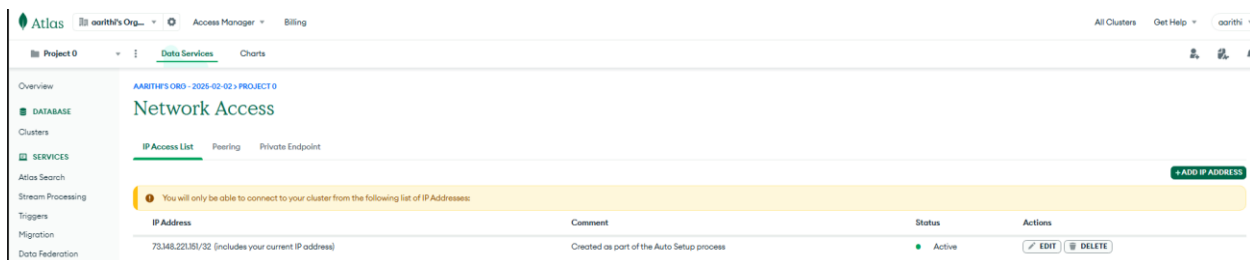
Click on “Done”

Step 3: To access the database from any host/IP address:



The screenshot shows the Atlas Overview page for 'AARITH'S ORG - 2025-02-02 > PROJECT 0'. The left sidebar contains navigation links for Overview, DATABASE (Clusters), SERVICES (Atlas Search, Stream Processing, Triggers, Migration, Data Federation), SECURITY (Quickstart, Backup, Database Access, Network Access, Advanced), and New On Atlas (4). The main content area has a yellow warning banner about a missing security contact. Below this, the 'Clusters' section is active, showing 'Cluster0' with a 'Connect' button and an 'Edit configuration' button. A blue banner提示 suggests using CLI/laC tools. Below are 'Browse collections' and 'View monitoring' buttons. The 'Application Development' section shows 'JavaScript / Node.js' selected and a 'Get connection string' button. A note at the bottom says 'Connect your application to your database deployments with the official MongoDB libraries. Start with a Node.js Sample App'.

Click on “Network Access”

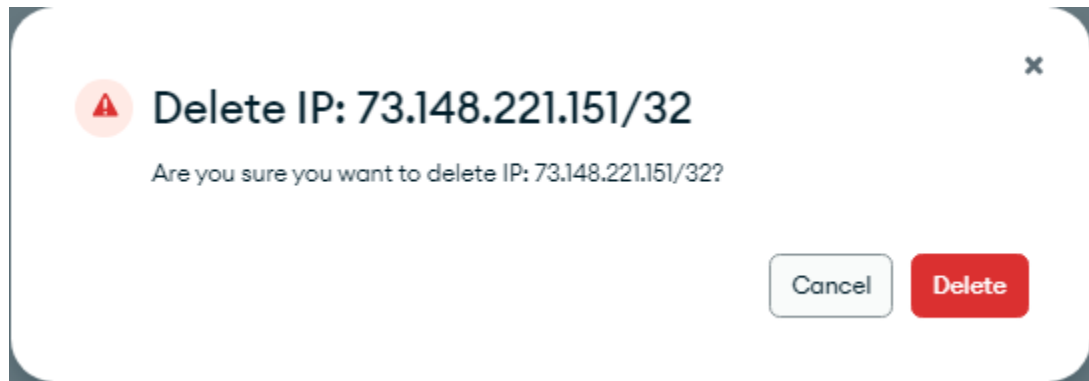


The screenshot shows the Atlas Network Access page for 'AARITH'S ORG - 2025-02-02 > PROJECT 0'. The left sidebar is the same as the previous screenshot. The main content area has a yellow warning banner stating 'You will only be able to connect to your cluster from the following list of IP Addresses:'. Below this is a table with columns: IP Address, Comment, Status, and Actions. The table contains one entry: IP Address '73.148.221.35/32 (includes your current IP address)', Comment 'Created as part of the Auto Setup process', Status 'Active', and Actions 'EDIT' and 'DELETE'. A '+ ADD IP ADDRESS' button is in the top right corner.

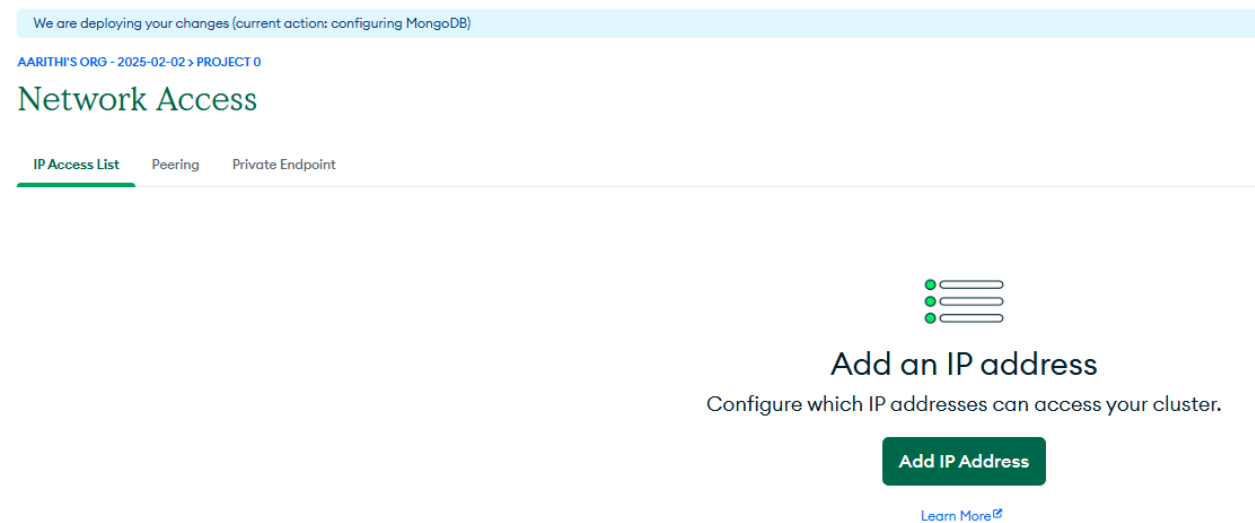
IP Address	Comment	Status	Actions
73.148.221.35/32 (includes your current IP address)	Created as part of the Auto Setup process	Active	EDIT DELETE

Database can be accessed only from host/ip address: 73.148.221.151

To allow any IP to access the database, deleted the existing one by clicking “DELETE”



Confirm Delete.



Click on "Add IP Address"

×

Add IP Access List Entry

Atlas only allows client connections to a cluster from entries in the project's IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. [Learn more](#)

ADD CURRENT IP ADDRESS

ALLOW ACCESS FROM ANYWHERE

Access List Entry:

0.0.0.0/0

Comment:

Optional comment describing this entry

☐

This entry is temporary and will be deleted in

6 hours

▼

Cancel

Confirm

Enter "0.0.0.0/0" to all allow IP addresses and click on "Confirm"

We are deploying your changes (current actions: configuring MongoDB)

AARITH'S ORG - 2025-02-02 > PROJECT 0

Network Access

IP Access List

Peering

Private Endpoint

+

ADD IP ADDRESS

ⓘ

You will only be able to connect to your cluster from the following list of IP Addresses:

IP Address	Comment	Status	Actions
0.0.0.0/0 (includes your current IP address)		Pending	<div>EDIT</div> <div>DELETE</div>

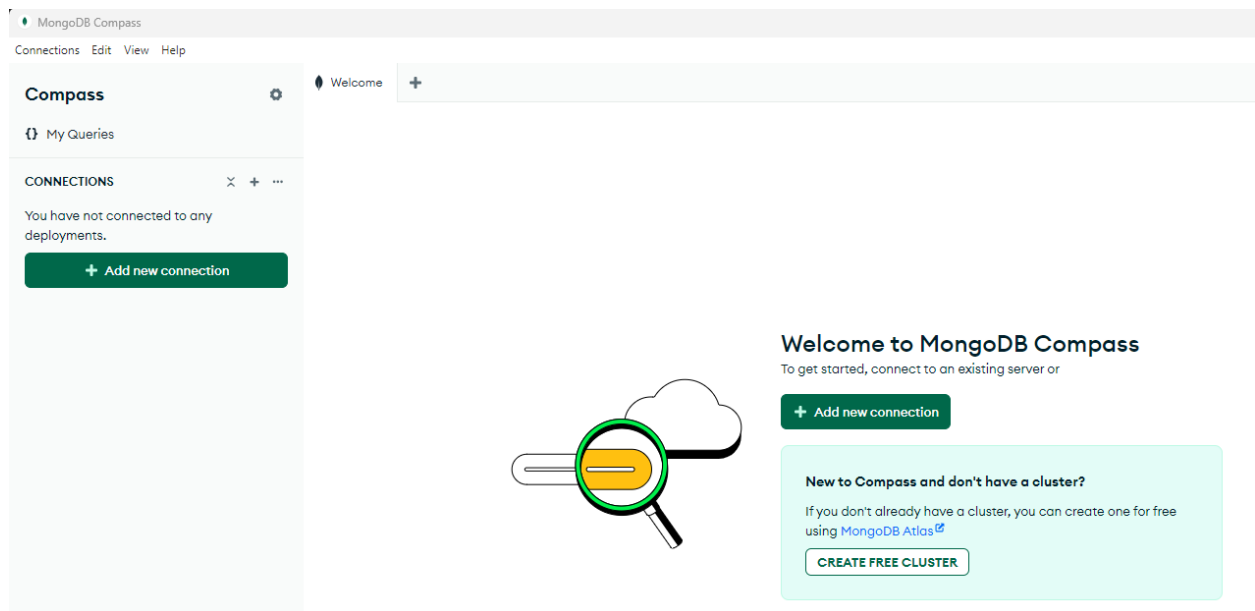
This completes deploying MongoDB database on Google Cloud and granting database access from any host/laptop/IP address.

MongoDB Compass:

MongoDB Compass is a free, open-source graphical user interface (GUI) for exploring, analyzing, and manipulating MongoDB data. MongoDB Atlas deployed on Google Cloud requires Google account credentials to access UI. Whereas Compass requires only connection string to connect to the database and it has more user-friendly UI to manage database.

Install MongoDB Compass:

Step 1: Download MongoDB Compass at <https://www.mongodb.com/try/download/compass>



Step 2. Add database

Click on “Add new conection”

`mongodb+srv://pinkbox:pinkbox@cluster0.jf5f4.mongodb.net/`

MongoDB Collections:

There are two collections are defined and they are:

1. Movie

// Schema for creating Movie|

```
const Movie = mongoose.model("Movie", {
  id: { type: Number, required: true },
  title: { type: String, required: true },
  description: { type: String, required: true },
  genre: { type: String, required: true },
  cost: { type: Number },
  image: { type: String, required: true },
  dor: { type: Date, required: true },
  streaming_url: { type: String, required: true },
  new_release: { type: String, required: true, default: false },
  available: { type: Boolean, default: true },
});
```

2. Users

```
// Schema for creating user model

const Users = mongoose.model("Users", {
  name: { type: String },
  email: { type: String, unique: true },
  password: { type: String },
  cartData: { type: Object },
  date: { type: Date, default: Date.now() },
});
```

PinkBox Setup Guide:

Step 1: Clone the PinkBox repository at <http://github.com/aarithi123/pinkbox>

The repository has four folders.

1. backend
2. frontend
3. inventory
4. documentation

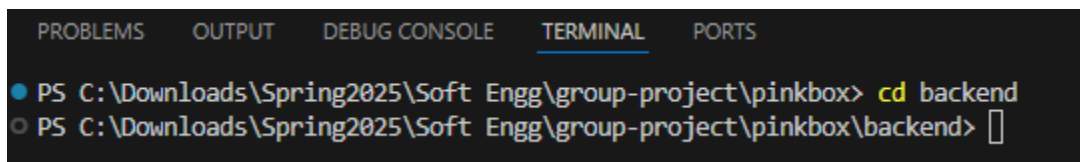
Before starting frontend, please make sure backend is started first as frontend requires endpoints exposed by backend to add/update/delete documents from MongoDB document database.

Backend:

How to restart the backend application?

Open the folder C:\Downloads\Spring2025\Soft Engg\group-project\pinkbox (pinkbox project) in vscode
OR navigate to <download location/pinkbox/backend

Open the Terminal and switch to backend project folder.

A screenshot of the Visual Studio Code interface, specifically the TERMINAL tab. The terminal shows a command prompt where the user has navigated to the backend folder. The command entered is 'cd backend' and the prompt has moved to the new directory.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
● PS C:\Downloads\Spring2025\Soft Engg\group-project\pinkbox> cd backend
○ PS C:\Downloads\Spring2025\Soft Engg\group-project\pinkbox\backend> █
```

Install backend application dependencies.

Pre-requisite: Node.js

To install Node.js, go to <https://nodejs.org/en/download>

nodejs.org/en/download

Download Node.js®

Get Node.js® v22.13.1 (LTS) for Windows using fnm with npm

```
1 # Download and install fnm:
2 winget install Schniz.fnm
3
4 # Download and install Node.js:
5 fnm install 22
6
7 # Verify the Node.js version:
8 node -v # Should print "v22.13.1".
9
10 # Verify npm version:
11 npm -v # Should print "10.9.2".
```

PowerShell Copy to clipboard

"fnm" is a cross-platform Node.js version manager. If you encounter any issues please visit [fnm's website](#)

Or get a prebuilt Node.js® for Windows running a x64 architecture.

Windows Installer (.msi) Standalone Binary (.zip)

Download windows installer and install “node.js”

To install dependencies, type the below command in the terminal

npm install

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

● PS C:\Downloads\Spring2025\Soft Engg\group-project\pinkbox> cd backend
● PS C:\Downloads\Spring2025\Soft Engg\group-project\pinkbox\backend> npm install

added 138 packages, and audited 139 packages in 15s

14 packages are looking for funding
  run `npm fund` for details

10 vulnerabilities (3 low, 1 moderate, 5 high, 1 critical)

To address all issues, run:
  npm audit fix

Run `npm audit` for details.
○ PS C:\Downloads\Spring2025\Soft Engg\group-project\pinkbox\backend>

To address all issues, run:
  npm audit fix

Run `npm audit` for details.
○ PS C:\Downloads\Spring2025\Soft Engg\group-project\pinkbox\backend>
```

Dependencies:

1. Install Express Server:

`npm install express`

2. Install jwt package for user auth:

`npm install jsonwebtoken`

3. Install mongodb package

`npm install mongoose`

4. Install multer for storing images

`npm install multer`

Note that the free version of MongoDB Atlas has only 512MB storage. The movie posters takes more space and therefore multer is used to store on folder on local storage.

5. Install cors to grant permission to the frontend application to access the backend application

`npm install cors`

(CORS stands for Cross-Origin Resource Sharing. It is a security feature implemented by web browsers to control how web pages can request resources from different origins (domains) than their own)

start the Express server:

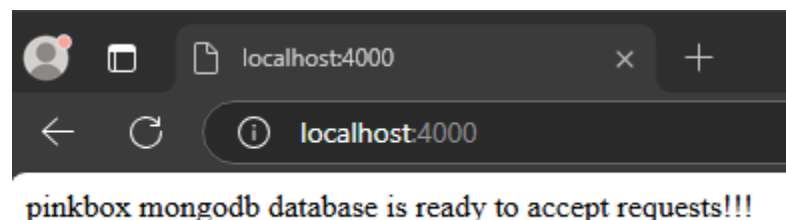
`node index.js`

It connects to mongodb database and exposes endpoints

```
PS C:\Downloads\Spring2025\Soft Engg\group-project\pinkbox\backend> node index.js
(node:15744) [DEP0040] DeprecationWarning: The `punycode` module is deprecated. Please use a userland alternative instead.
(Use `node --trace-deprecation ...` to show where the warning was created)
Server Running on port 4000
PS C:\Downloads\Spring2025\Soft Engg\group-project\pinkbox\backend> node index.js
(node:15744) [DEP0040] DeprecationWarning: The `punycode` module is deprecated. Please use a userland alternative instead.
(Use `node --trace-deprecation ...` to show where the warning was created)
Server Running on port 4000
```

Express server is now listening on port 4000

To check Express server is ready to accept requests, <http://localhost:4000>



Endpoints:

1. login endpoint for login the user and sending auth-token (jwt)

/login

Request:

Response:

Testing:

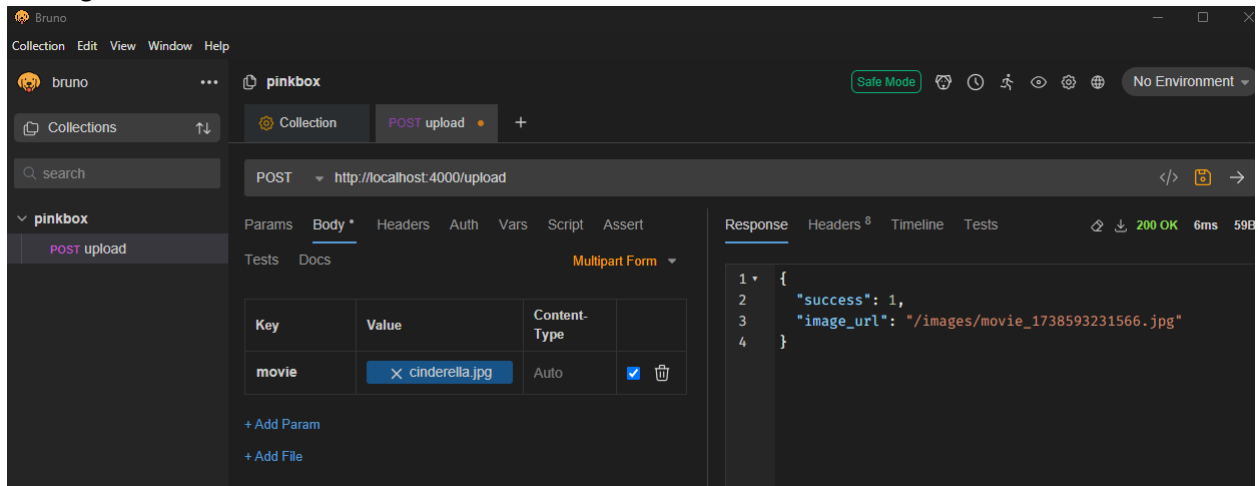
2. upload movie posters

/upload

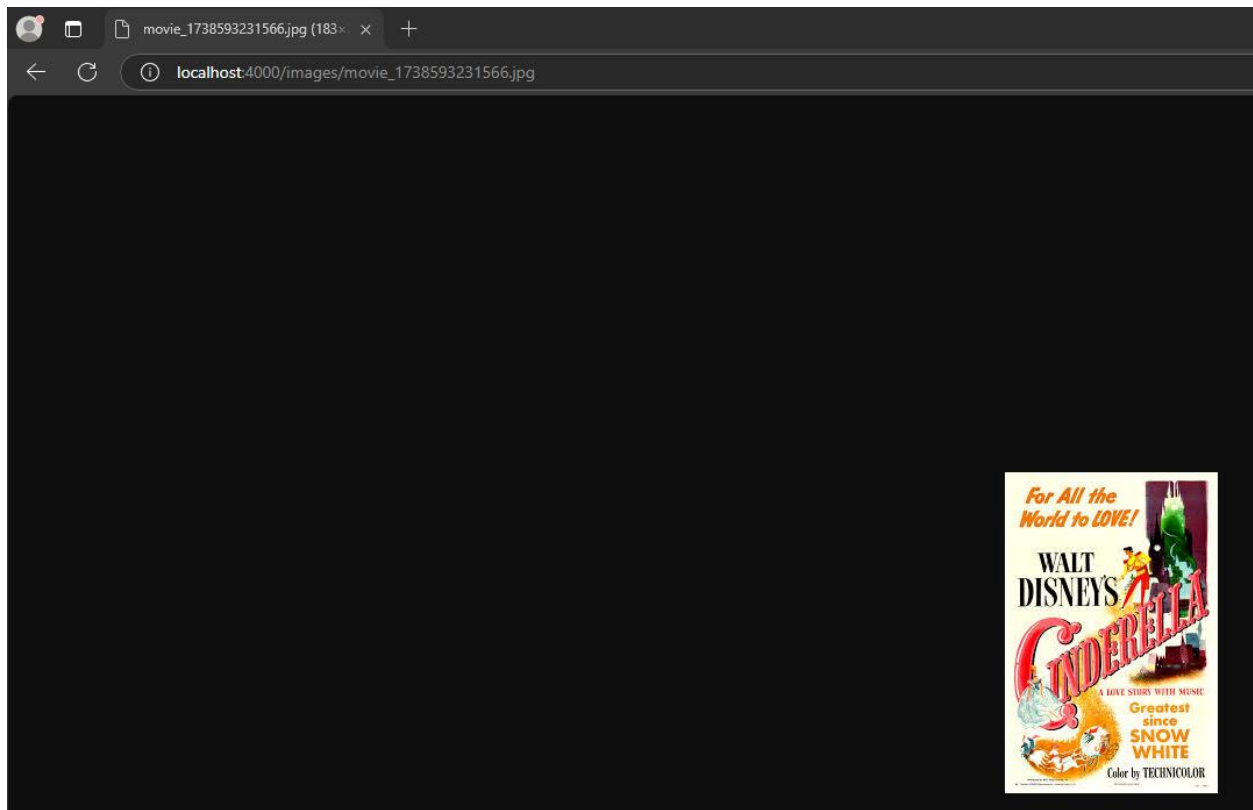
The free version of “Thunder Client” does not allow imageFile POST.

Please install open source Bruno from <https://www.usebruno.com/downloads>

Testing:



Once image upload is successful, it can be viewed at http://localhost:4000/<image_url>
http://localhost:4000/images/movie_1738593231566.jpg



/addmovie:

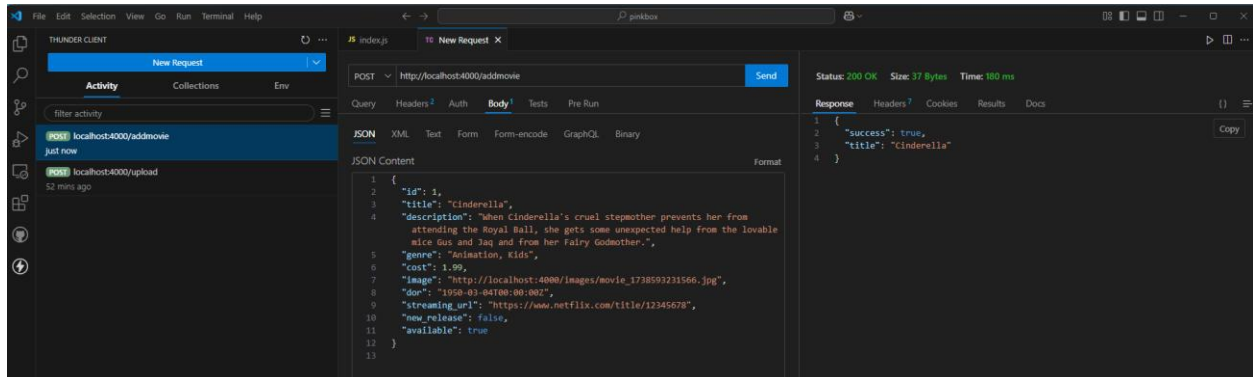
request:

```
{
  "id": 1,
  "title": "Cinderella",
  "description": "When Cinderella's cruel stepmother prevents her from attending the Royal Ball, she gets some unexpected help from the lovable mice Gus and Jaq and from her Fairy Godmother.",
  "genre": "Animation, Kids",
  "cost": 1.99,
  "image": "http://localhost:4000/images/movie_1738593231566.jpg",
  "dor": "1950-03-04T00:00:00Z",
  "streaming_url": "https://www.netflix.com/title/12345678",
  "new_release": false,
  "available": true
}
```

response:

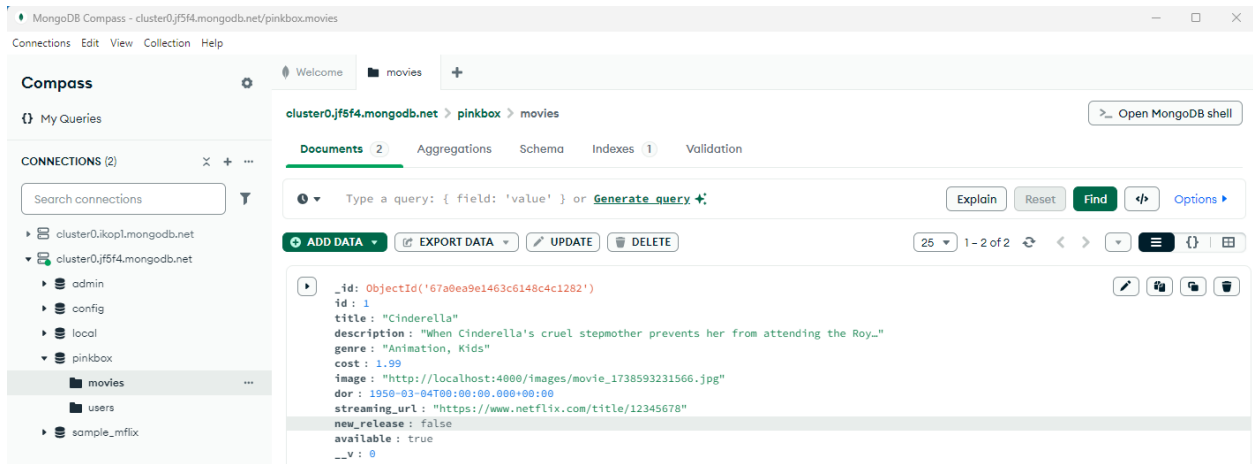
```
{
  "success": true,
```

```
"title": "Cinderella"
}
```



MongoDB Compass:

Success scenario



Failure scenario:

response:

```
{
  "success": false,
  "error": " Title must be unique. Duplicate title found."
}
```

/removemovie:

request:

```
{
  "title": "Cinderella"
}
```

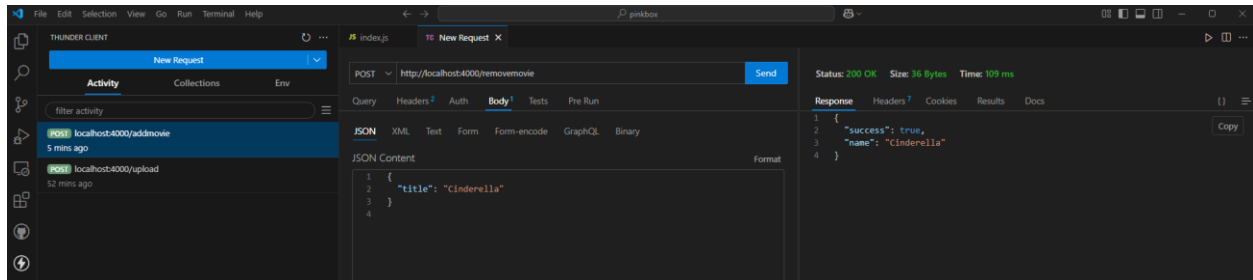
response:

```
{
```

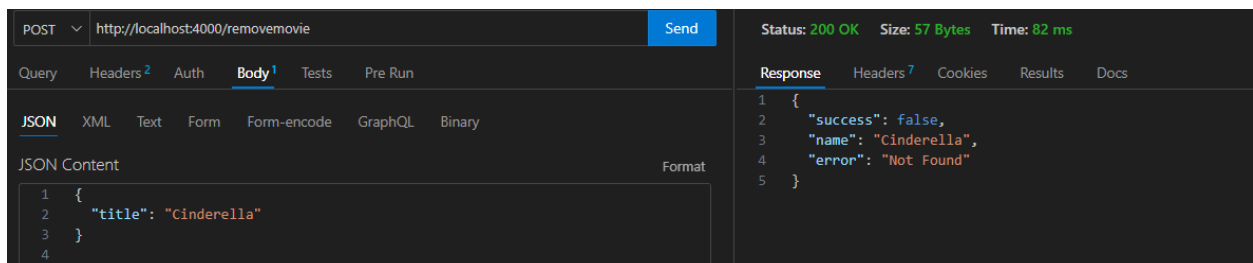
```

"success": true,
"title": "Cinderella"
}

```



If movie not found to delete:



/moviebytitle:
request:

```

{
  "title": "Cinderella"
}

```

response:

```

{
  "success": true,
  "movie": {
    "_id": "67a10990082e8f1aaae803ad",
    "id": 1,
    "title": "Cinderella",
    "description": "When Cinderella's cruel stepmother prevents her from attending the Royal Ball, she gets some unexpected help from the lovable mice Gus and Jaq and from her Fairy Godmother.",
    "genre": "Animation, Kids",
    "cost": 1.99,
    "image": "http://localhost:4000/images/movie_1738593231566.jpg",
    "dor": "1950-03-04T00:00:00.000Z",
    "streaming_url": "https://www.netflix.com/title/12345678",
    "new_release": false,
  }
}

```

```
"available": true,  
  "__v": 0  
}
```

GET

Status: 200 OK Size: 512 Bytes Time: 64 ms

Query Headers² Auth **Body¹** Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

JSON Content Format

```
1 {  
2   "title": "Cinderella"  
3 }  
4
```

Response Headers⁷ Cookies Results Docs {} ≡

```
1 {  
2   "success": true,  
3   "movie": {  
4     "_id": "67a10990082e8f1aaae803ad",  
5     "id": 1,  
6     "title": "Cinderella",  
7     "description": "When Cinderella's cruel stepmother prevents her from  
8       attending the Royal Ball, she gets some unexpected help from the lovable  
9       mice Gus and Jaq and from her Fairy Godmother.",  
10    "genre": "Animation, Kids",  
11    "cost": 1.99,  
12    "image": "http://localhost:4000/images/movie_1738593231566.jpg",  
13    "don": "1950-03-04T00:00:00.000Z",  
14    "streaming_url": "https://www.netflix.com/title/12345678",  
15    "new_release": false,  
16    "available": true,  
17    "__v": 0  
18  }  
19 }
```

Movie not found condition:

GET

Status: 404 Not Found Size: 63 Bytes Time: 67 ms

Query Headers² Auth **Body¹** Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

JSON Content Format

```
1 {  
2   "title": "Lion King"  
3 }  
4
```

Response Headers⁷ Cookies Results Docs {} ≡

```
1 {  
2   "success": false,  
3   "title": "Lion King",  
4   "error": "Movie not found"  
5 }
```

/moviesbygenre:
request:

```
{  
  "genre": "Animation"  
}
```

response:

```
{  
  "success": true,  
  "movies": [  
    {  
      "_id": "67a10990082e8f1aaae803ad",  
      "id": 1,  
      "title": "Cinderella",  
      "description": "When Cinderella's cruel stepmother prevents her from attending the Royal Ball, she  
gets some unexpected help from the lovable mice Gus and Jaq and from her Fairy Godmother.",  
      "genre": "Animation, Kids",  
      "cost": 1.99,  
      "image": "http://localhost:4000/images/movie_1738593231566.jpg",  
    }  
  ]  
}
```

```

    "dor": "1950-03-04T00:00:00.000Z",
    "streaming_url": "https://www.netflix.com/title/12345678",
    "new_release": false,
    "available": true,
    "__v": 0
  },
  {
    "_id": "67a10f19aac2c80a33ba5efe",
    "id": 2,
    "title": "Lion King",
    "description": "When Cinderella's cruel stepmother prevents her from attending the Royal Ball, she gets some unexpected help from the lovable mice Gus and Jaq and from her Fairy Godmother.",
    "genre": "Animation, Kids",
    "cost": 1.99,
    "image": "http://localhost:4000/images/movie_1738593231566.jpg",
    "dor": "1950-03-04T00:00:00.000Z",
    "streaming_url": "https://www.netflix.com/title/12345678",
    "new_release": false,
    "available": true,
    "__v": 0
  }
]
}

```

The screenshot shows a REST client interface with a GET request to `http://localhost:4000/moviesbygenre` and its corresponding JSON response. The response status is 200 OK, with a size of 1002 Bytes and a time of 58 ms.

Query: GET `http://localhost:4000/moviesbygenre` [Send](#)

Response: Status: 200 OK, Size: 1002 Bytes, Time: 58 ms

JSON Content:

```

1 {
2   "genre": "Animation"
3 }
4

```

Response Body:

```

9   attending the Royal Ball, she gets some unexpected help from the
10  lovable mice Gus and Jaq and from her Fairy Godmother.",
11  "genre": "Animation, Kids",
12  "cost": 1.99,
13  "image": "http://localhost:4000/images/movie_1738593231566.jpg",
14  "dor": "1950-03-04T00:00:00.000Z",
15  "streaming_url": "https://www.netflix.com/title/12345678",
16  "new_release": false,
17  "available": true,
18  "__v": 0
19 },
20 {
21   "_id": "67a10f19aac2c80a33ba5efe",
22   "id": 2,
23   "title": "Lion King",
24   "description": "When Cinderella's cruel stepmother prevents her from
25   attending the Royal Ball, she gets some unexpected help from the
26   lovable mice Gus and Jaq and from her Fairy Godmother.",
27   "genre": "Animation, Kids",
28   "cost": 1.99,
29   "image": "http://localhost:4000/images/movie_1738593231566.jpg",
30   "dor": "1950-03-04T00:00:00.000Z",
31   "streaming_url": "https://www.netflix.com/title/12345678",
32   "new_release": false,
33   "available": true,
34   "__v": 0
35 }
36 ]
37 }

```

Movies not found condition:

GET

http://localhost:4000/moviesbygenre

Send

Query

Headers²

Auth

Body¹

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

1

{

2

"genre": "Thriller"

3

}

4

Status: 404 Not Found

Size: 68 Bytes

Time: 54 ms

Response

Headers⁷

Cookies

Results

Docs

1

{

2

"success": false,

3

"error": "No movies found with the specified genre"

4

}

Copy

GET

http://localhost:4000/moviebytitle

Send

Query

Headers²

Auth

Body¹

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

1

{

2

"title": "Lion King"

3

}

4

Status: 404 Not Found

Size: 63 Bytes

Time: 67 ms

Response

Headers⁷

Cookies

Results

Docs

1

{

2

"success": false,

3

"title": "Lion King",

4

"error": "Movie not found"

5

}

Copy

/allmovies:

GET

http://localhost:4000/allmovies

Send

Query

Headers²

Auth

Body

Tests

Pre Run

JSON

XML

Text

Form

Form-encode

GraphQL

Binary

JSON Content

Format

1

Status: 200 OK

Size: 1002 Bytes

Time: 67 ms

Response

Headers⁷

Cookies

Results

Docs

1

{

2

"success": true,

3

"movies": [

4

{

5

"_id": "67a10990082e8f1aaae803ad",

6

"id": 1,

7

"title": "Cinderella",

8

"description": "When Cinderella's cruel stepmother prevents her from attending the Royal Ball, she gets some unexpected help from the lovable mice Gus and Jaq and from her Fairy Godmother.",

9

"genre": "Animation, Kids",

10

"cost": 1.99,

11

"image": "http://localhost:4000/images/movie_1738593231566.jpg",

12

"dor": "1950-03-04T00:00:00.000Z",

13

"streaming_url": "https://www.netflix.com/title/12345678",

14

"new_release": false,

15

"available": true,

16

"__v": 0

17

},

18

{

19

"_id": "67a10f19aac2c80a33ba5efe",

20

"id": 2,

21

"title": "Lion King",

22

"description": "When Cinderella's cruel stepmother prevents her from attending the Royal Ball, she gets some unexpected help from the lovable mice Gus and Jaq and from her Fairy Godmother.",

23

"genre": "Animation, Kids",

24

"cost": 1.99,

25

"image": "http://localhost:4000/images/movie_1738593231566.jpg",

Response

Chart

/newreleases

GET <http://localhost:4000/newreleases> Send

Status: 200 OK Size: 514 Bytes Time: 71 ms

Query Headers² Auth **Body¹** Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

JSON Content Format

```
1
```

Response Headers⁷ Cookies Results Docs {} Copy

```
1 {
2   "success": true,
3   "movies": [
4     {
5       "_id": "67a1099082e8f1aae803ad",
6       "id": 1,
7       "title": "Cinderella",
8       "description": "When Cinderella's cruel stepmother prevents her from
9         attending the Royal Ball, she gets some unexpected help from the
10        lovable mice Gus and Jaq and from her Fairy Godmother.",
11       "genre": "Animation, Kids",
12       "cost": 1.99,
13       "image": "http://localhost:4000/images/movie_1738593231566.jpg",
14       "dor": "1950-03-04T00:00:00.000Z",
15       "streaming_url": "https://www.netflix.com/title/12345678",
16       "new_release": true,
17       "available": true,
18       "_v": 0
19     }
20   ]
21 }
```

/signup

Request:

```
{
  "username": "aarithi",
  "password": "aarithi",
  "email": aarithir@gmail.com
}
```

Response:

```
{
  "success": true,
  "token":
  "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyIjp7ImklkjoInjdhMTIzMmIwNDA5OWNiZDliNzljYTQ4In0sImIhdCI6MTczODYxMzU0N30.Ej3gDomZLuQqCkLD_yF-8PAuvwQAUOn419R5NSZfAfc"
}
```

POST <http://localhost:4000/signup> Send

Status: 200 OK Size: 187 Bytes Time: 152 ms

Query Headers² Auth **Body¹** Tests Pre Run

JSON XML Text Form Form-encode GraphQL Binary

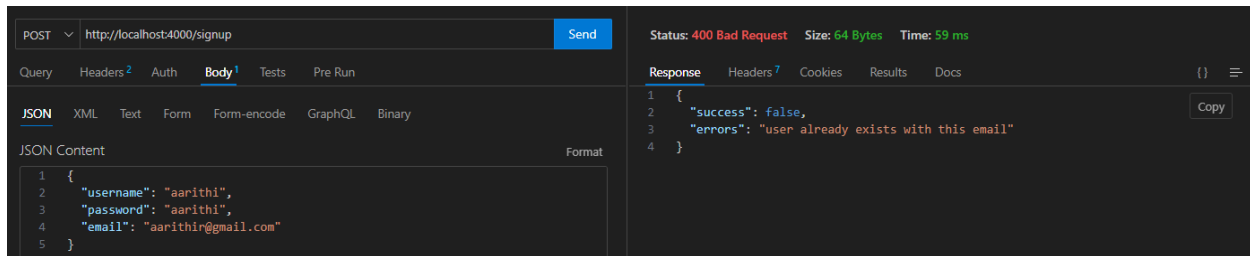
JSON Content Format

```
1 {
2   "username": "aarithi",
3   "password": "aarithi",
4   "email": "aarithir@gmail.com"
5 }
```

Response Headers⁷ Cookies Results Docs {} Copy

```
1 {
2   "success": true,
3   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyIjp7ImklkjoInjdhMTIzMmIwNDA5OWNiZDliNzljYTQ4In0sImIhdCI6MTczODYxMzU0N30.Ej3gDomZLuQqCkLD_yF-8PAuvwQAUOn419R5NSZfAfc"
4 }
```

Validation: email should be unique for each user



“token” in the above response is the jwt authentication token. The frontend application, when calls endpoints using http POST/GET methods, adds "auth-token" header with the value of “token”. During user login also, auth-token (jwt token) is generated and passed as response to the caller. Frontend application adds this token to http header by name "auth-token". This auth-token is valid until user logs off. Until then every call that frontend UI should validate the presence of this token. If not, UI should default to Login page. Backend application validates for below endpoints to make sure that user is authenticated.

/addtocart
 /removefromcart
 /getcart

/login

Request:

```

{
  "email": "aarithir@gmail.com",
  "password": "aarithi"
}

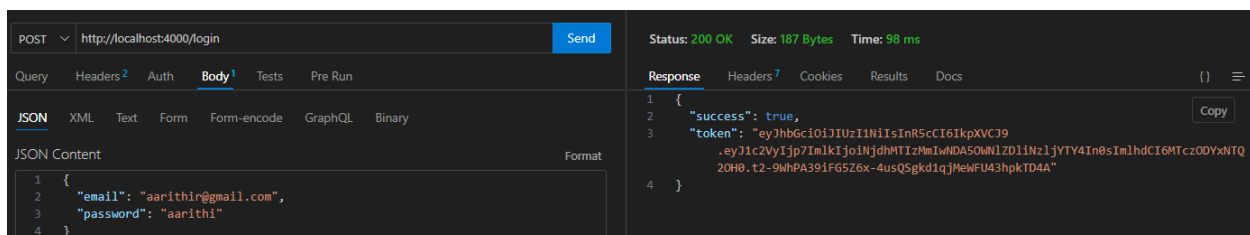
```

Response:

```

{
  "success": true,
  "token":
  "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyIjp7ImklkjoIjndhMTIzMmIwNDA5OWNlZDliNzljYTY4In0sImhhdCI6MTczODYxNTQ2OH0.t2-9WhPA39iFG5Z6x-4usQsgkd1qjMeWfU43hpkTD4A"
}

```



Validation: email should be unique for each user

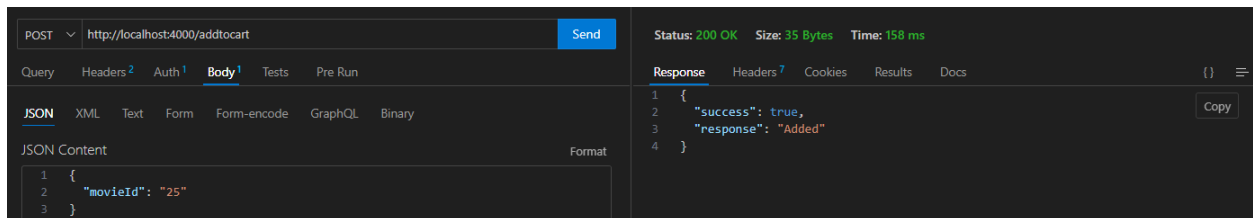
/addtocart

Request:

```
{
  "movieId": "25"
}
```

Response:

```
{
  "success": true,
  "response": "Added"
}
```



Note: Upon user signup, user document added to “Users” collection in mongodb. Each user has cartData object initialized with 100 items. The assumption is that there will 100 movies loaded to “Movie” collection with movieId starting from 1 to 100.

When “Add to Cart” is clicked, the frontend UI will make a call to /addtocart endpoint (<http://localhost:4000/addtocart>) with:

```
{ "movieId": "<movieId>" }
```

The frontend UI will know the movieId that user selecting either making the call to /moviebytitle endpoint OR string the movieId of somewhere

The movieId corresponds to the item number in User.cartData

Any operation on cart, /addtocart, /removefromcart and /getcart requires auth-token. Frontend will store the auth-token (jwt) created during sign-up or login and pass the same to cart operations by setting this http header.

Example:

```
fetch('http://localhost:4000/addtocart', {
  method: 'POST',
  headers: {
    Accept: 'application/form-data',
    'auth-token': `${localStorage.getItem('auth-token')}`, //assuming jwt token stored in localStorage
    'Content-Type': 'application-json',
  },
  body: JSON.stringify({ "movieId": movieId}),
})
.then
```

```

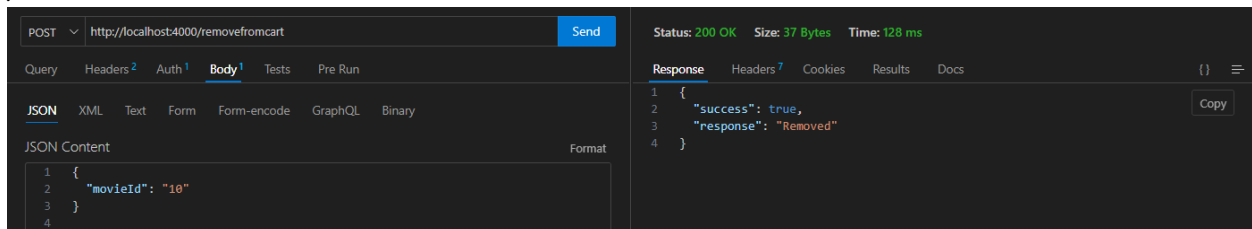
----
----
}

  _id: ObjectId('67a1232b04099ced9b79ca68')
  name: "aarithi"
  email: "aarithir@gmail.com"
  password: "aarithi"
  ▼ cartData: Object
    0: 0
    1: 0
    2: 0
    3: 0
    4: 0
    5: 0
    6: 0
    7: 0
    8: 0
    9: 0
    10: 0
    11: 0

```

The corresponding itemId (which is same as movieId) in cartData will be incremented by 1.
If there are five movies selected by the user, frontend UI will make call to /addtocart five times, each time changing the movieId.

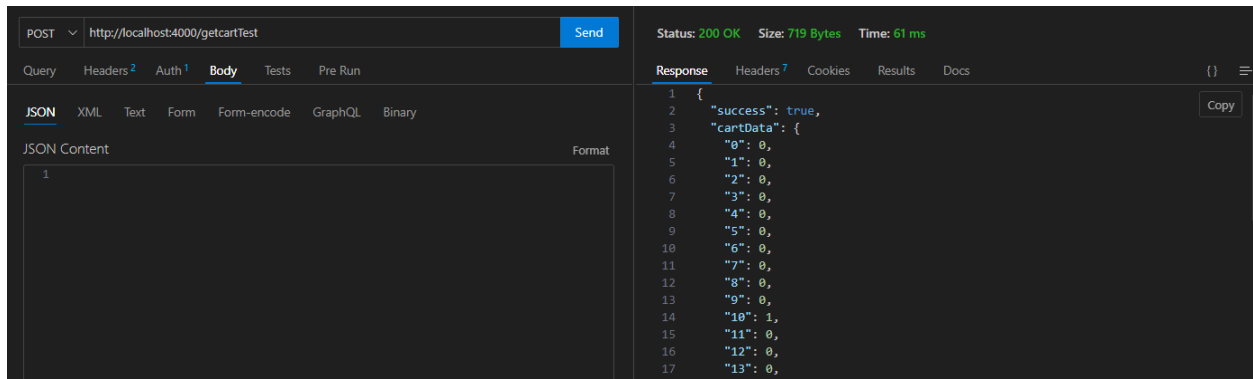
/removefromcart



The screenshot shows a REST client interface with a POST request to `http://localhost:4000/removefromcart`. The request body is a JSON object: `{ "movieId": "10" }`. The response status is `200 OK` with a size of `37 Bytes` and a time of `128 ms`. The response body is a JSON object: `{ "success": true, "response": "Removed" }`.

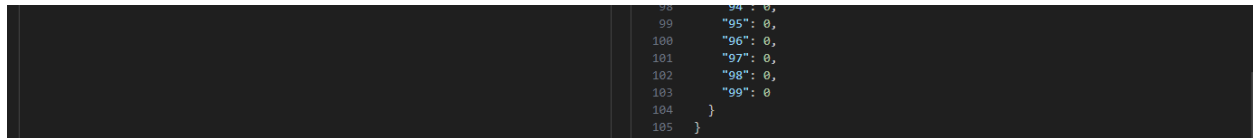
Tab	Content
Query	POST http://localhost:4000/removefromcart
Headers	Auth 1
Body	JSON Content: <pre>{ "movieId": "10" }</pre>
Response	Status: 200 OK, Size: 37 Bytes, Time: 128 ms. Response: <pre>{ "success": true, "response": "Removed" }</pre>

/getcart



....

....



npx create-react-app .

npm install react-router-dom

npm start

backend:

mongodb:

<https://www.mongodb.com/atlas>

techstack/techstack

mongodb+srv://techstack:<db_password>@cluster0.ikop1.mongodb.net/

login: <https://account.mongodb.com/account/login?signedOut=true>

nodejs:

<https://nodejs.org/en/download>

vscode terminal:

npm install

node index.js (start Express server)

`http://localhost:4000`

frontend:

vscode terminal

`npm install`

`npm start` (start frontend)

`http://localhost:3000` (app url)

backend:

Express.js (Express) to create API and json web token (jwt) for user authentication.

Express is a web application framework that provides a robust set of features for web and mobile applications. It simplifies the process of building server-side applications.

Mongodb Atlas database

to store the image -> Multer is a Node.js middleware for handling multipart/form-data, which is primarily used for uploading files. It is an essential part of any web application that needs to handle file uploads, especially when you're working with forms that include file inputs.

vscode terminal

`cd backend`

`npm init`

package name: backend

Entry point: index.js

Install Express Server:

`npm install express`

install jwt package for user auth:

`npm install jsonwebtoken`

install mongodb package

npm install mongoose

install multer for storing images

npm install multer

install cors to grant permission to the application to access the backend

npm install cors

(CORS stands for Cross-Origin Resource Sharing. It is a security feature implemented by web browsers to control how web pages can request resources from different origins (domains) than their own)

install ThunderClient in vscode

admin:

npm vite@latest .

vite is next gen frontend development tool and it requires Node.js v18 and above

select "React" framework

select "javascript"

npm install

npm install react-router-dom

to start => npm run dev