My Courses › M220JS

Course Ends:
15d:22hr:39m
Aug 06, 17:00 UTC

Chapter Labs Due:
01d:22hr:39m
Jul 23, 17:00 UTC

Chapter 1: Driver Setup ▼

LESSONS | HANDOUTS

Lessons

- Lecture: Introduction to Chapter 1
- Lecture: README
- Lecture: MongoClient
- Lab (Ungraded): Import Dataset
- Ticket: Connection
- Lecture: Asynchronous Programming in Node.js
  - Quiz
- Lecture: Basic Reads
  - Quiz
- Ticket: Projection
- Ticket: Text and Subfield Search

Next Chapter

Report an issue

Course Overview

View Discussion

Chapter 1: Driver Setup

README

Download Course Materials

Handouts (1)

m220/mflix-js.zip

In order to run properly, the MFlix software project has some installation requirements and environmental dependencies.

These requirements and dependencies are defined in this lesson, and they can also be found in the **README.rst** file from the **mflix-js** handout. This lesson is to make sure you don't skip them!

After following this README, you should be able to successfully run the MFlix application.

Project Structure

You can find the **mflix-js.zip** file attached to this lecture as a handout. Downloading this handout may take a few minutes. Unzipping the file should create a new directory called **mflix-js**.

Most of your work will be implementing methods in the **dao** directory, which contains all database interfacing methods. The API will make calls to Data Access Objects (DAOs) that interact directly with MongoDB.

The unit tests in **test** will test these database access methods directly, without going through the API. The UI will run these methods in integration tests, and therefore requires the full application to be running.

The lesson handouts can be found in the **test/lessons** directory. These files will look like **<lesson-name>.spec.js**, and can be run with `npm test -t <lesson-name>`.

The API layer is fully implemented, as is the UI. The application is programmed to run on port **5000** by default - if you need to run on a port other than 5000, you can edit the **dotenv_win** (if on Windows) or the **dotenv_unix** file (if on Linux or Mac) in the root directory to modify the value of **PORT**.

Please do not modify the API layer in any way, under the **mflix-js/src/api** directory. This may result in the front-end application failing to validate some of the labs.

Node Library Dependencies

The dependencies for the MFlix application should be downloaded using the `npm` command-line tool. You can get this tool by downloading Node.js. Make sure to choose the correct option for your operating system.

Once the installation is complete, you may need to restart your computer before using the command line tools. You can test that it's installed by running the following command:

```
node -v
```

This should print out the version of `node` you currently have - we recommend using version 10 or later, so this command should print something like `v10.x`.

Once `npm` is installed, you can install the MFlix dependencies by running the following command from the **mflix-js** directory:

```
npm install
```

You must run this from the top level of the project, so `npm` has access to the **package.json** file where the dependencies are.

MongoDB Installation

It is recommended to connect MFlix with MongoDB Atlas, so you do not need to have a MongoDB server running on your host machine. The lectures and labs in this course will assume that you are using an Atlas cluster instead of a local instance.

That said, you are still required to have the MongoDB server installed, in order to be able to use two server tool dependencies:

- `mongorestore`
  - A utility for importing binary data into MongoDB.
- `mongo`
  - The MongoDB shell

To download these command line tools, please visit the MongoDB download center and choose the appropriate platform.

All of these tools are free to use. MongoDB Enterprise is also free to use for testing and evaluation purposes.

MongoDB Atlas Cluster

MFlix uses MongoDB to persist all of its data.

One of the easiest ways to get up and running with MongoDB is to use MongoDB Atlas, a hosted and fully-managed database solution.

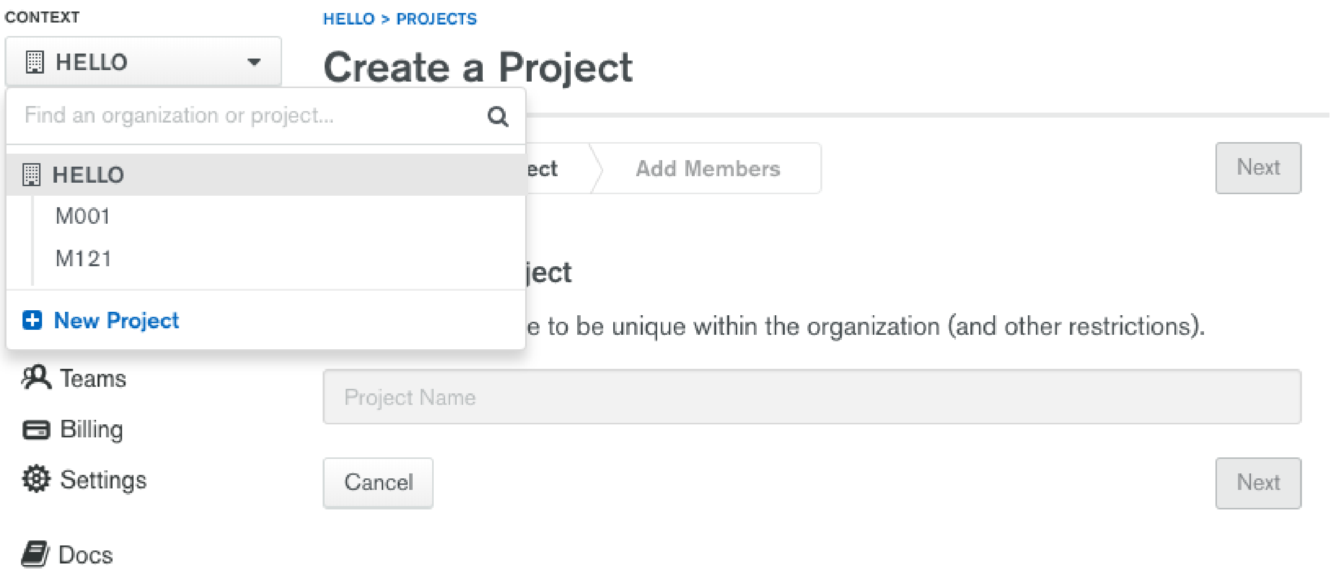
If you have taken other MongoDB University courses like M001 or M121, you may already have an account - feel free to reuse that cluster for this course.

Note: Be advised that some of the UI aspects of Atlas may have changed since the inception of this README, therefore some of the screenshots in this file may be different from the actual Atlas UI interface.

Using an existing MongoDB Atlas Account:

If you already have a previous Atlas account created, perhaps because you've taken one of our other MongoDB university courses, you can repurpose it for this course.

Log-in to your Atlas account and create a new project named **M220** by clicking on the **Context** dropdown menu:

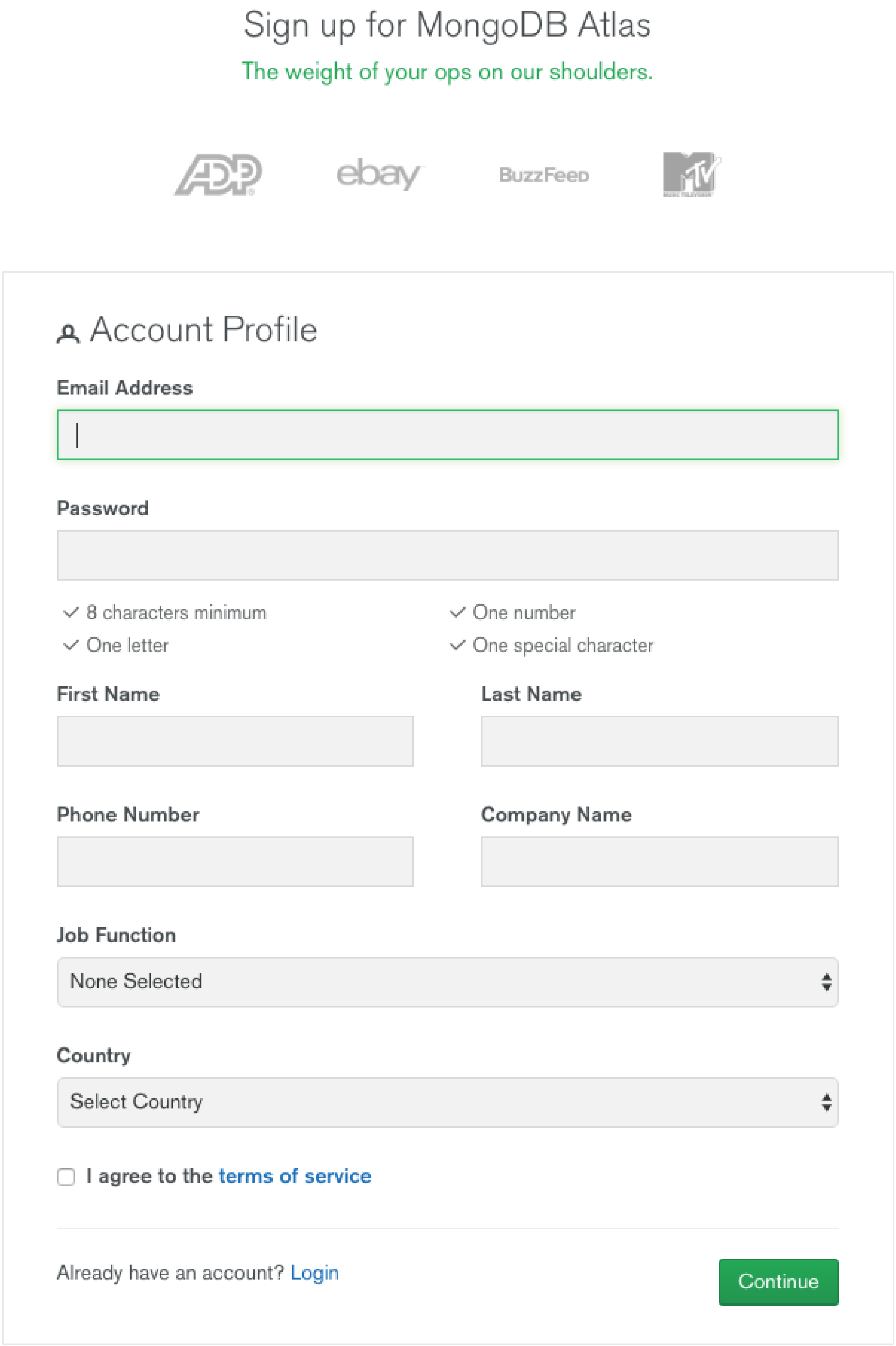


e to be unique within the organization (and other restrictions).

After creating a new project, you need to create an **mflix** free tier cluster.

Creating a new MongoDB Atlas Account:

If you do not have an existing Atlas account, go ahead and create an Atlas Account by filling in the required fields:



Sign up for MongoDB Atlas

The weight of your ops on our shoulders.

Account Profile

Email Address

Password

✓ 8 characters minimum
✓ One letter
✓ One number
✓ One special character

First Name

Last Name

Phone Number

Company Name

Job Function

None Selected

Country

Select Country

□ I agree to the terms of service

Already have an account? Login

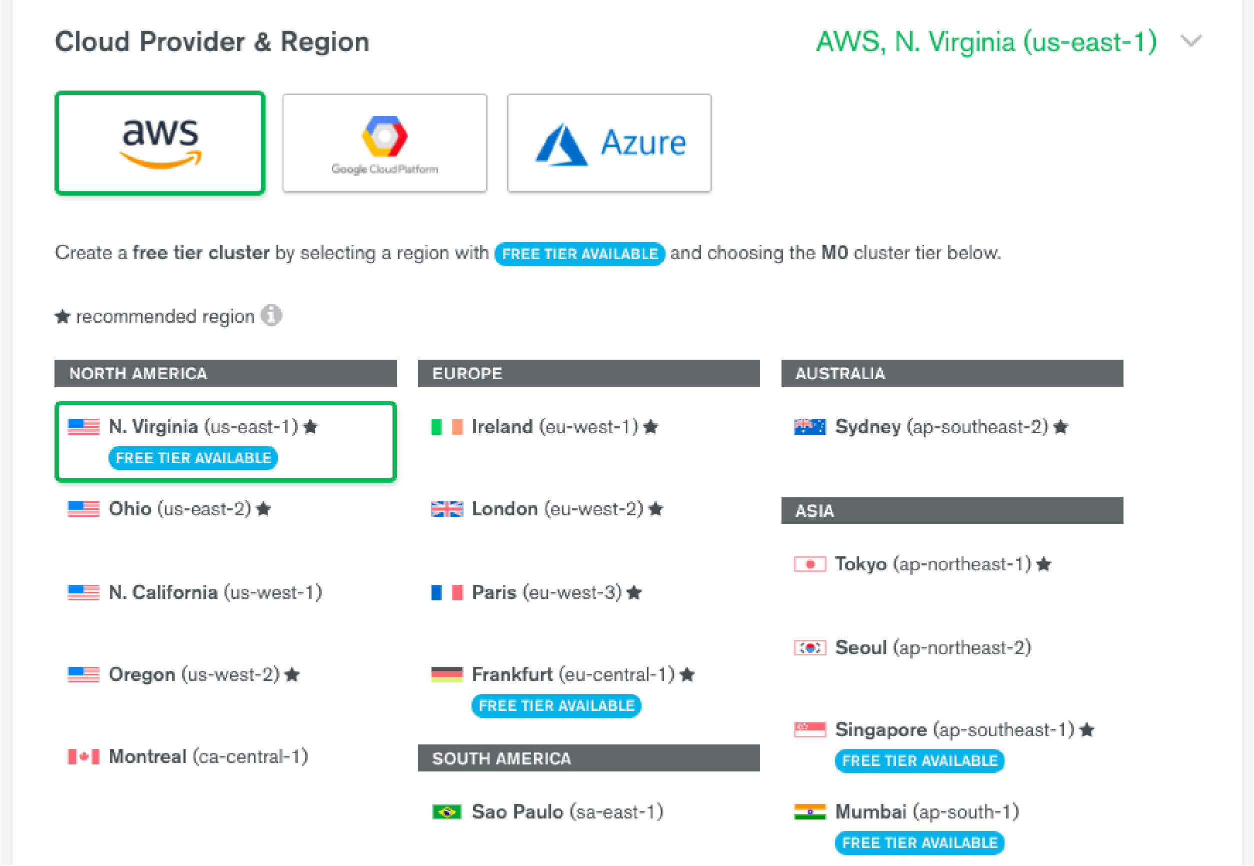
Continue

Creating a free tier cluster called "mflix":

Note: You will need to do this step even if you are reusing an Atlas account.

After creating a new project, click the "**Build a New Cluster**" button on the top right corner.

1. Select AWS as the cloud provider, in a Region that has the label **Free Tier Available**.



Cloud Provider & Region AWS, N. Virginia (us-east-1)

Create a **free tier cluster** by selecting a region with FREE TIER AVAILABLE and choosing the **M0** cluster tier below.

★ recommended region

NORTH AMERICA

- **N. Virginia** (us-east-1) ★ FREE TIER AVAILABLE
- **Ohio** (us-east-2) ★
- **N. California** (us-west-1)
- **Oregon** (us-west-2) ★
- **Montreal** (ca-central-1)

EUROPE

- **Ireland** (eu-west-1) ★
- **London** (eu-west-2) ★
- **Paris** (eu-west-3) ★
- **Frankfurt** (eu-central-1) ★ FREE TIER AVAILABLE

SOUTH AMERICA

- **Sao Paulo** (sa-east-1)

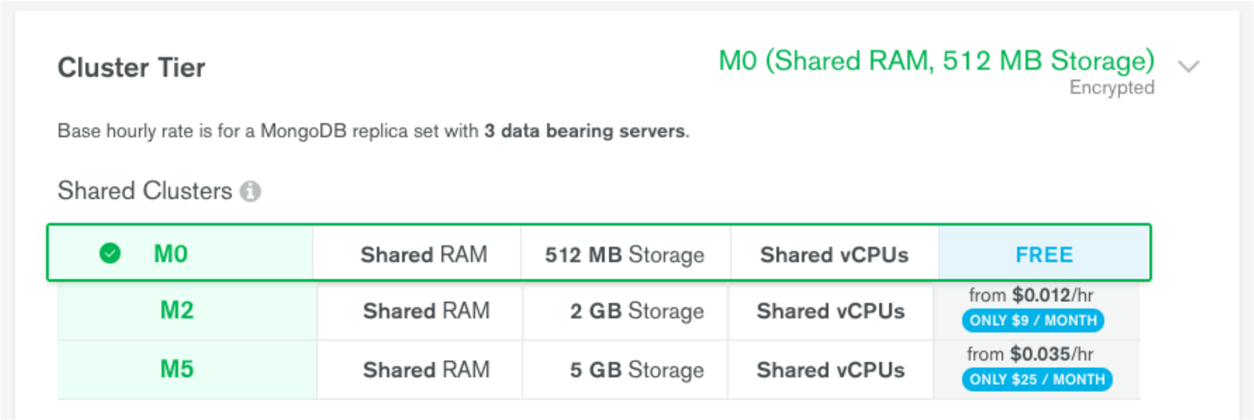
AUSTRALIA

- **Sydney** (ap-southeast-2) ★

ASIA

- **Tokyo** (ap-northeast-1) ★
- **Seoul** (ap-northeast-2)
- **Singapore** (ap-southeast-1) ★ FREE TIER AVAILABLE
- **Mumbai** (ap-south-1) FREE TIER AVAILABLE

2. Next, choose **Cluster Tier** M0:



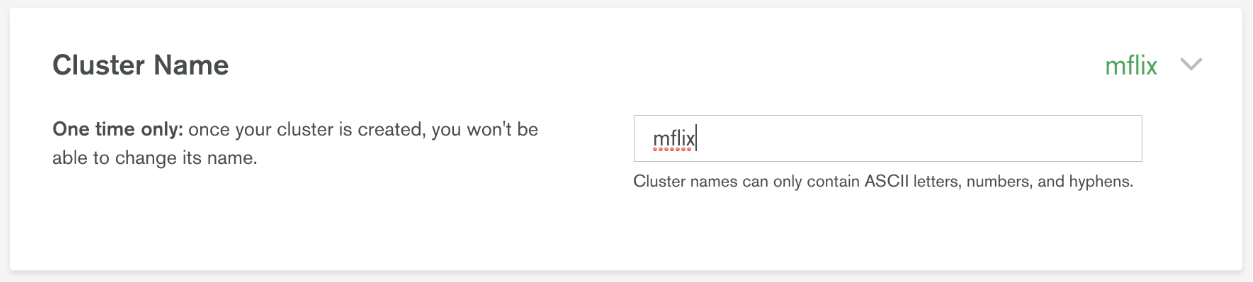
Cluster Tier M0 (Shared RAM, 512 MB Storage) Encrypted

Base hourly rate is for a MongoDB replica set with **3 data bearing servers**.

Shared Clusters

✓ M0	**Shared** RAM	**512 MB** Storage	**Shared** vCPUs	FREE
M2	**Shared** RAM	**2 GB** Storage	**Shared** vCPUs	from **$0.012**/hr ONLY $9 / MONTH
M5	**Shared** RAM	**5 GB** Storage	**Shared** vCPUs	from **$0.035**/hr ONLY $25 / MONTH

3. Set **Cluster Name** to **mflix** and click **Create Cluster**. It may take 7-10 minutes to successfully create your Atlas cluster:



Cluster Name

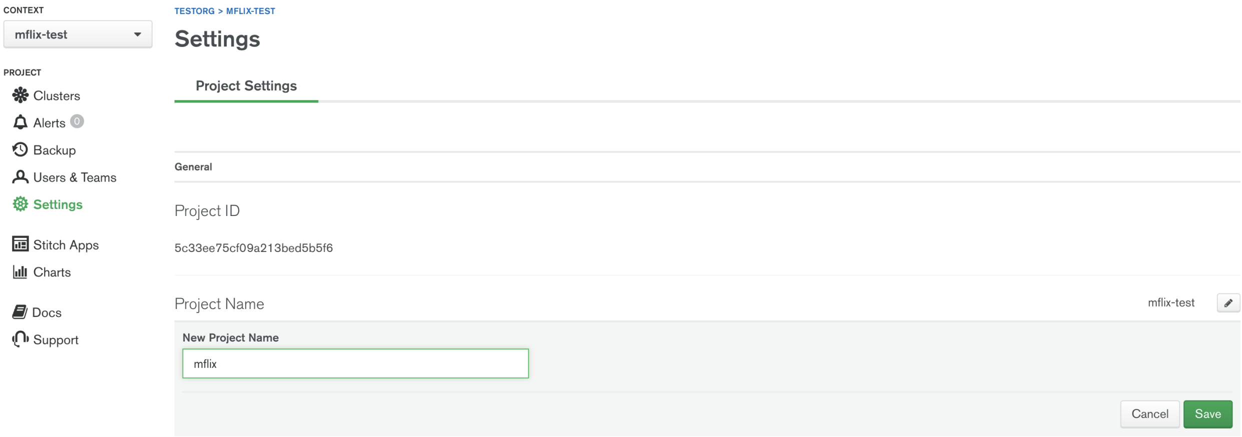
mflix

One time only: once your cluster is created, you won't be able to change its name.

mflix

Cluster names can only contain ASCII letters, numbers, and hyphens.

4. Once you press **Create Cluster**, you will be redirected to the account dashboard. In this dashboard, make sure you set your project name to **M220**. Go to **Settings** menu item and change the project name from the default **Project 0** to **M220**:



Settings

Project Settings

General

Project ID

5c33ee75cf09a213bed5b5f6

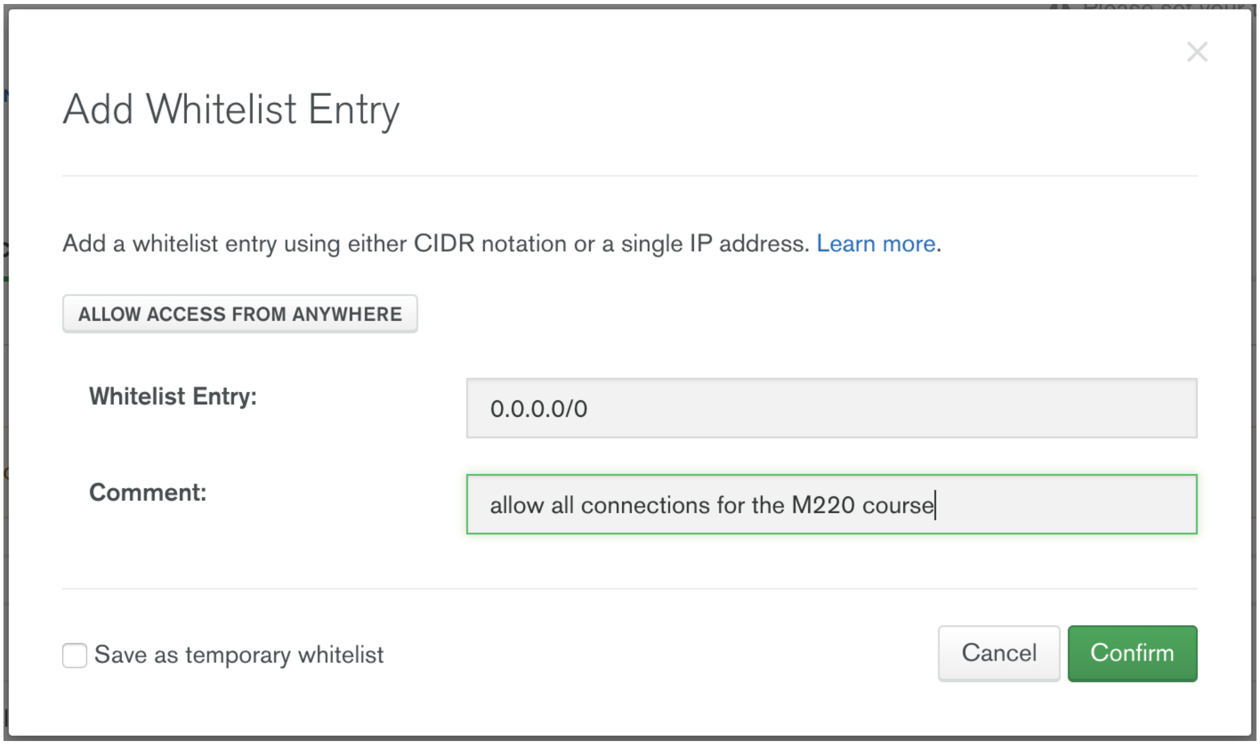
Project Name mflix-test

New Project Name

mflix

5. Next, configure the security settings of this cluster, by enabling the **IP Whitelist**.

   Update your IP Whitelist so that your app can talk to the cluster. Click the **Security** tab from the **Clusters** page. Then click **IP Whitelist** followed by **Add IP Address**. Finally, click **Allow Access from Anywhere** and click **Confirm**.



Add Whitelist Entry

Add a whitelist entry using either CIDR notation or a single IP address. Learn more.

Whitelist Entry: 0.0.0.0/0

Comment: allow all connections for the M220 course

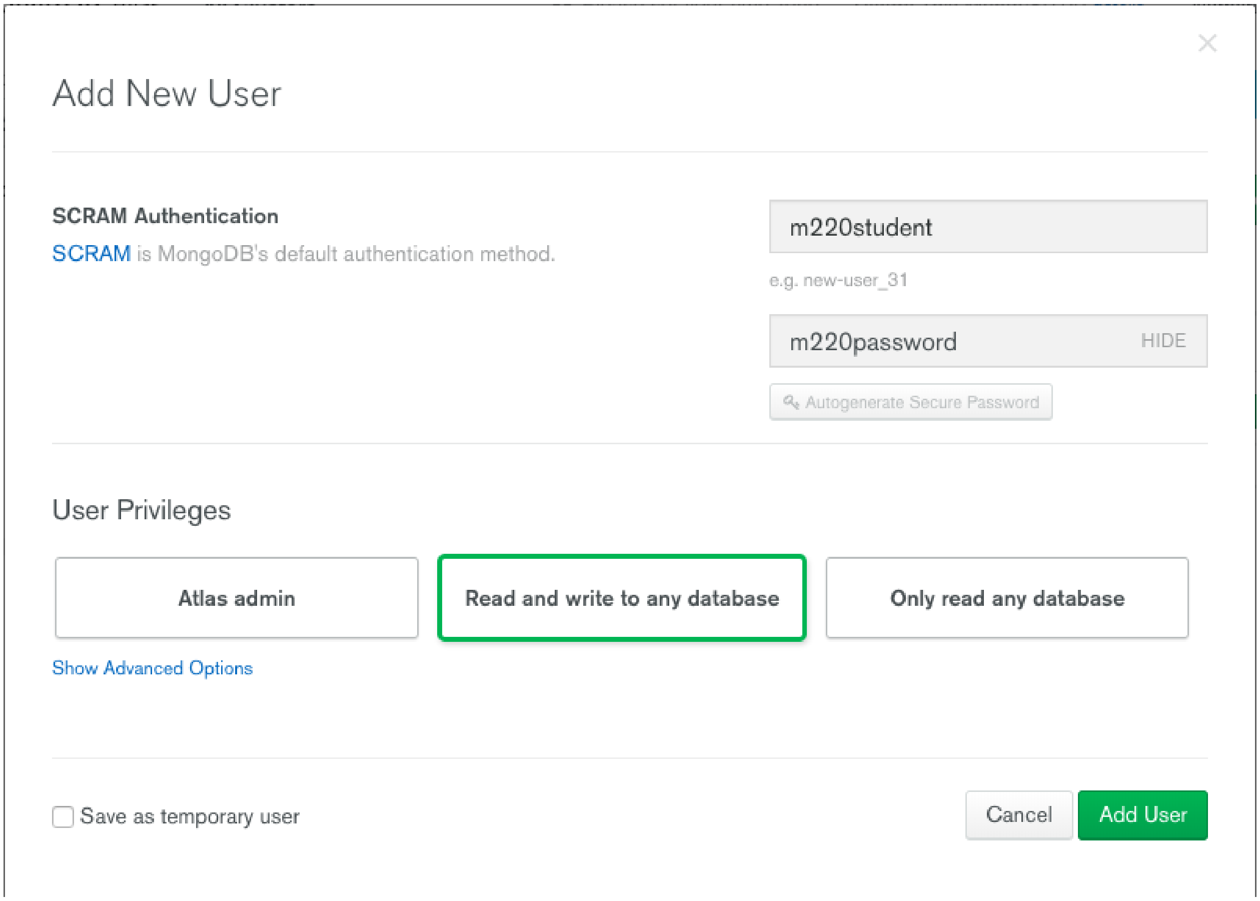
□ Save as temporary whitelist

6. Then create the application MongoDB database user required for this course:

    - username: **m220student**
    - password: **m220password**

    You can create new users through **Security** -> **Add New User**.

    Allow this user the privilege to **Read and write to any database**:



Add New User

SCRAM Authentication

SCRAM is MongoDB's default authentication method.

m220student

e.g. new-user_31

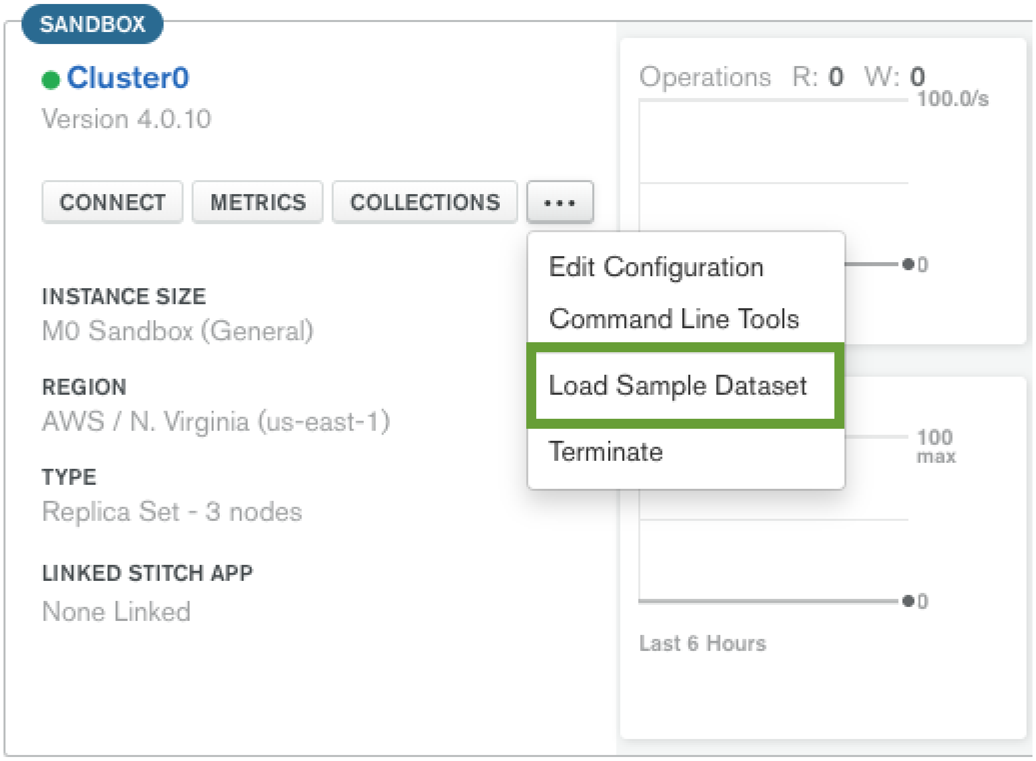
m220password

User Privileges

Read and write to any database

□ Save as temporary user

7. When the user is created, and the cluster deployed, you have the option to `Load Sample Dataset`. This will load the Atlas sample dataset, containing the MFlix database, into your cluster:



Cluster0

Version 4.0.10

INSTANCE SIZE

M0 Sandbox (General)

REGION

AWS / N. Virginia (us-east-1)

TYPE

Replica Set - 3 nodes

LINKED STITCH APP

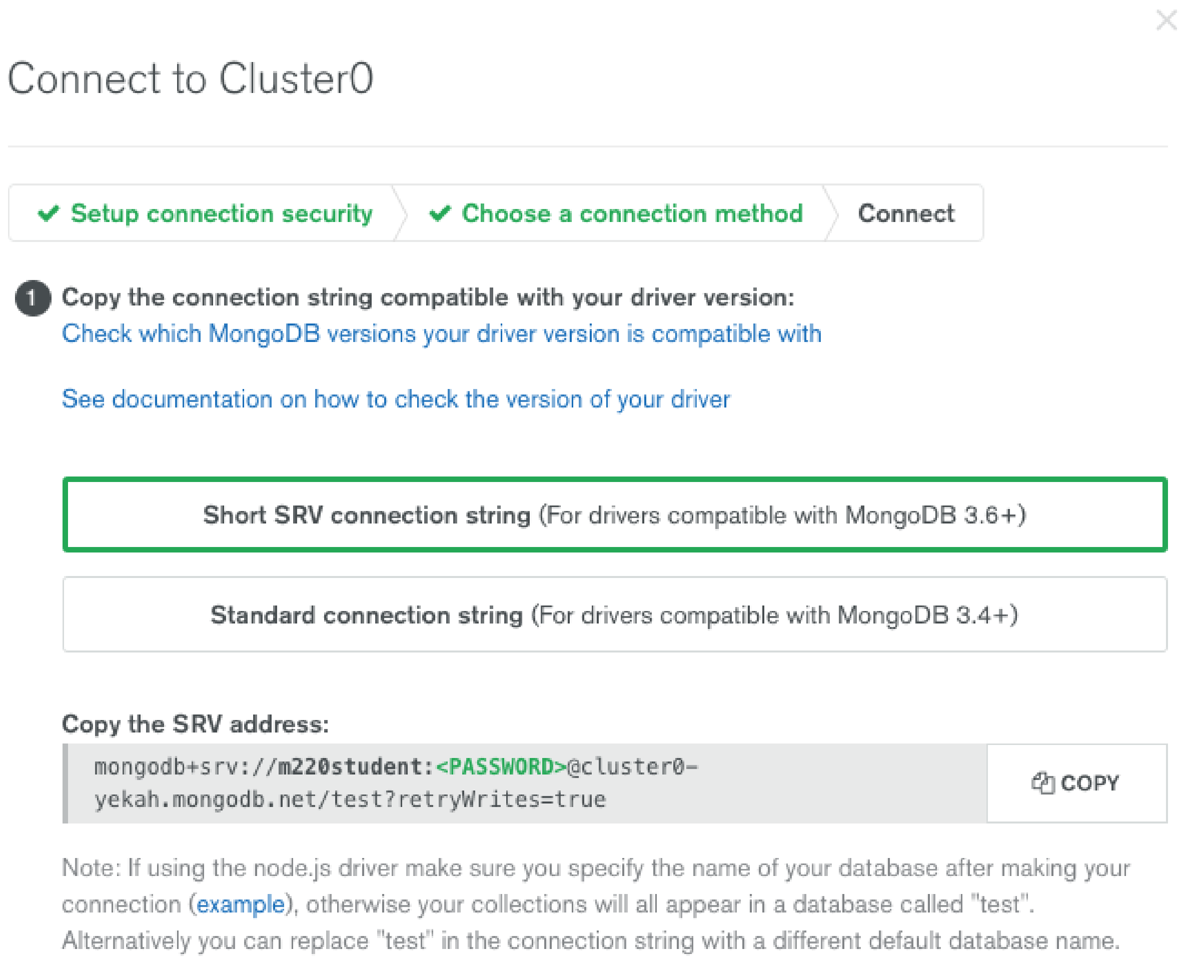
None Linked

Operations R: 0 W: 0

Note: The MFlix database in the Sample Dataset is called "sample_mflix".

8. Now you can test the setup by connecting via the Mongo shell. You can find instructions to connect in the **Connect Your Application** section of the cluster dashboard.

   Go to your cluster **Overview** -> **Connect** -> **Connect Your Application**. Select the option titled "**Short SRV connection string**", and copy the URI string:



Connect to Cluster0

1 **Copy the connection string compatible with your driver version:**
Check which MongoDB versions your driver version is compatible with

See documentation on how to check the version of your driver

Short SRV connection string (For drivers compatible with MongoDB 3.6+)

Standard connection string (For drivers compatible with MongoDB 3.4+)

Copy the SRV address:

```
mongodb+srv://m220student:<PASSWORD>@cluster0-
yekah.mongodb.net/test?retryWrites=true
```

Note: If using the node.js driver make sure you specify the name of your database after making your connection (example), otherwise your collections will all appear in a database called "test". Alternatively you can replace "test" in the connection string with a different default database name.

The below example connects to Atlas as the user you created before, with username **m220student** and password **m220password**. You can run this command from your command line:

```
mongo "mongodb+srv://m220student:m220password@clust
```

COPY

By connecting to the server from your host machine, you have validated that the cluster is configured and reachable from your local workstation.

You may see the following message when you connect:

```
Error while trying to show server startup warnings: use
```

This is a log message, **not** an error - feel free to ignore it.

Importing Data (Optional)

Note: if you used Load Sample Dataset, you can skip this step.

The `mongorestore` command necessary to import the data is located below. Copy the command and use an Atlas SRV string to import the data (including username and password credentials).

Replace the SRV string below with your own:

```
# navigate to mflix-js directory
cd mflix-js

# import data into Atlas
mongorestore --drop --gzip --uri \
  "mongodb+srv://m220student:m220password@<YOUR_CLUSTER_URI>"
data
```

The entire dataset contains almost 200,000 documents, so importing this data may take 5-10 minutes.

Running the Application

In order for the application to use Atlas, you will need a file called **.env** to contain the connection information. In the **mflix-js** directory you can find two files, **dotenv_unix** (for Unix users) and **dotenv_win** (for Windows users).

Open the file for your chosen operating system and enter your Atlas SRV connection string as directed in the comment. This is the information the driver will use to connect. Make sure **not** to wrap your Atlas SRV connection between quotes:

```
MFLIX_DB_URI = mongodb+srv://...
```

It's highly suggested you also change the **SECRET_KEY** to some very long, very random string. While this application is only meant for local use during this course, software has a strange habit of living a long time.

When you've edited the file, rename it to **.env** with the following command:

```
mv dotenv_unix .env  # on Unix
ren dotenv_win .env  # on Windows
```

COPY

Note: Once you rename this file to **.env**, it will no longer be visible in Finder or File Explorer. However, it will be visible from Command Prompt or Terminal, so if you need to edit it again, you can open it from there:

```
vi .env       # on Unix
notepad .env  # on Windows
```

COPY

In the **mflix-js** directory, run the following commands:

COPY

```
# install MFlix dependencies
npm install

# start the MFlix application
npm start
```

This will start the application. You can then access the MFlix application at http://localhost:5000/.

Running the Unit Tests

To run the unit tests for this course, you will use Jest. Jest has been included in this project's dependencies, so `npm install` should install everything you need.

Each course lab contains a module of unit tests that you can call individually with `npm test`. For example, to run the test **connection-pooling.test.js**, run the command:

```
npm test -t TICKET_TEST_NAME
```

COPY

Each ticket will contain the exact command to run that ticket's specific unit tests. You can run these commands from anywhere in the **mflix-js** project. Bear in mind that a tests will fail until the corresponding ticket is completed.

Proceed to next section