



## Course Overview



## View Discussion

### Chapter 0: Introduction and Setup

## README

---

### Download Course Materials

---

#### Handouts (1)

 [m220/mflix-java.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-java** project, which you will download shortly. This lesson serves as a guide for setting up these necessary tools. After following this README, you should be able to successfully run the Mflix application. First, you will need to download the **mflix-java** project, as described below.

## Download the mflix-java.zip file

You can download the **mflix-java.zip** file by clicking the link in the "Handouts" section of this page. Downloading this handout may take a few minutes. When the download is complete, unzip the file and **cd** into the project's root directory, **mflix-java**.

```
cd ~/Downloads
unzip mflix-java.zip
cd mflix-java
```

 COPY

## Table of Contents

1. Project Structure
2. Database Layer
3. Local Environment Dependencies
4. Java Project (Mflix) Installation
5. MongoDB Installation
6. MongoDB Atlas Cluster

6.1 Using an existing MongoDB Atlas Account 6.2 Creating a new MongoDB Atlas Account 6.3 Creating an M0 free tier cluster **mflix**

7. Importing Data
8. Running the Application
9. Running the Unit Tests

## Project Structure

Mflix is composed of two main components:

- *Frontend*: All the UI functionality is already implemented for you, which includes the built-in React application that you do not need to worry about.
- *Backend*: *Java Spring Boot* project that provides the necessary service to the application. The code is managed by a Maven project definition file that you will have to load into your Java IDE.

Most of what you will implement is located in the `src/main/java/mflix/api/daos` directory, which contains all database interfacing methods.

The unit tests in `src/test/java/mflix/api/daos` will test these database access methods directly, without going through the API.

The UI will run these methods as part of the integration tests, and therefore they are required for the full application to be running.

The API layer is fully implemented, as is the UI. By default the application will run on port 5000, but if you need it to run on a port other than 5000, you can edit the **index.html** file in the **build** directory to modify the value of **window.host**. You can find **index.html** in the `src/main/resources/build` directory.

We're using *Spring Boot* for the API. Maven will download this library for you. More on that below.

## Database Layer

We will be using *MongoDB Atlas*, MongoDB's official Database as a Service (DBaaS), so you will not need to manage the database component yourself. However, you will still need to install MongoDB locally to access the command line tools that interact with Atlas, to load data into MongoDB and potentially do some exploration of your database with the shell.

The following set of steps are here to get you setup for this course.

## Local Environment Dependencies

There are two main system dependencies in this course:

### 1. Java 1.8

- The Java version this course is built against is Java 1.8. You can download the appropriate version for your operating system by clicking [here](#).

### 2. Maven

- We use Maven to manage dependencies for the Mflix project. Click here to download [Maven](#). You can find detailed installation instructions on [the Apache website](#).

## Java Project (Mflix) Installation

The Mflix project is supported by a *Maven* POM file that deals with all the dependencies required, as well as providing the **test** and **run** commands to control our project. This means that you can run all the tests and deploy the Mflix backend from the command line with *Maven*.

However, we recommend you use a Java IDE to follow along with the lessons and to accomplish the **Tickets** assigned to you in the course.

You can use any IDE that you like, as you do not need to have a specific product to complete the course. It would be better if your IDE supports *Maven* POM files, so it can set the dependencies correctly, otherwise you will need to download and install manually the different libraries and drivers used by the project.

That said, all the lectures and examples of this course have been produced using IntelliJ IDEA CE edition. You will find a lesson dedicated to setting up and exploring this IDE for the course.

Once you downloaded and unzipped the **mflix-java.zip** file, you will find the project folder. The project folder contains the application code, the **pom.xml** file that you would import into your IDE, and the dataset required that you will have to import to Atlas.

---

```
$ ls
mflix README.rst
$ cd mflix
$ ls src pom.xml data
```

## 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 shell for exploring data in MongoDB.

To download these command line tools, please visit the [MongoDB download center](#) and choose the appropriate platform.

## MongoDB Atlas Cluster

*Mflix* uses *MongoDB* to persist all 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.

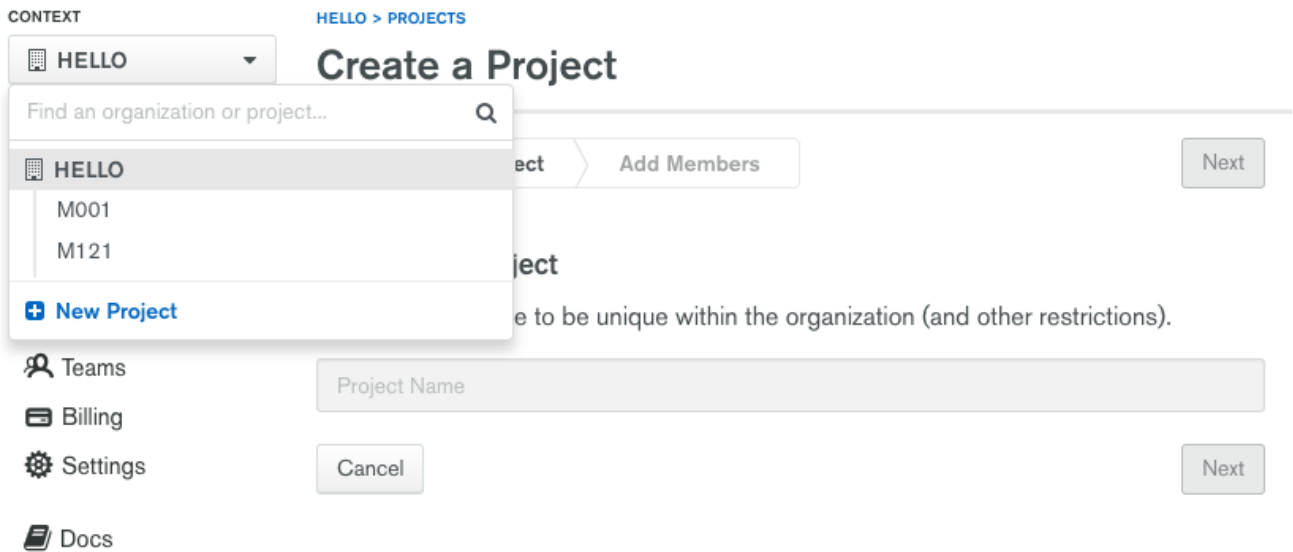
Make sure to use a **free tier cluster** for the application and course.

*Note: Be advised that some of the UI aspects of Atlas may have changed since the redaction 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 M220J.

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



After creating this new project, skip the next section and proceed to the *Creating an M0 free tier cluster Mflix* section.

## 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.



BuzzFeed



## 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](#)

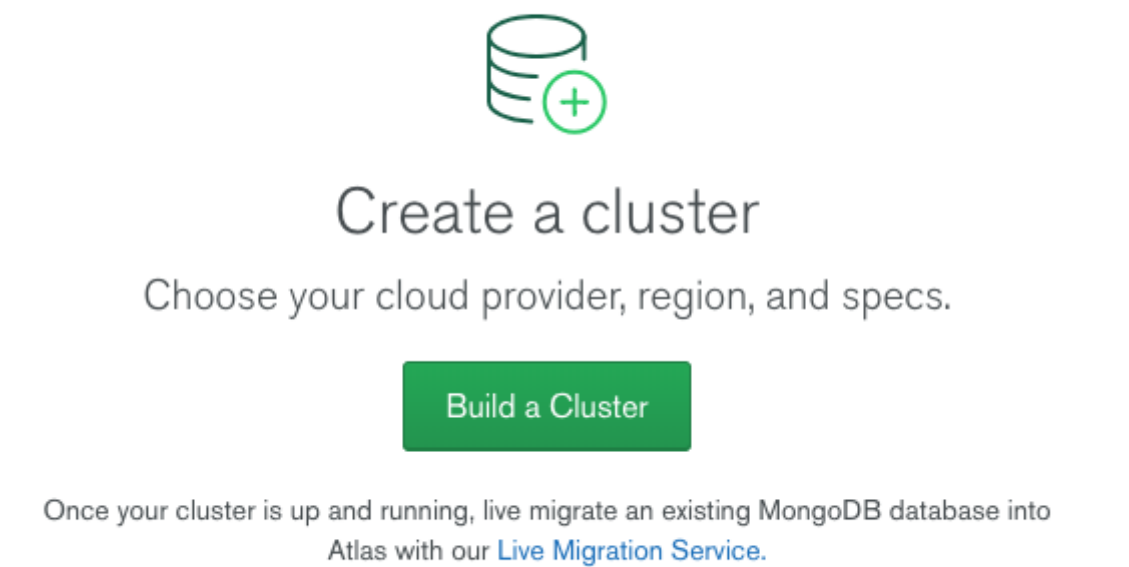
Already have an account? [Login](#)

Continue

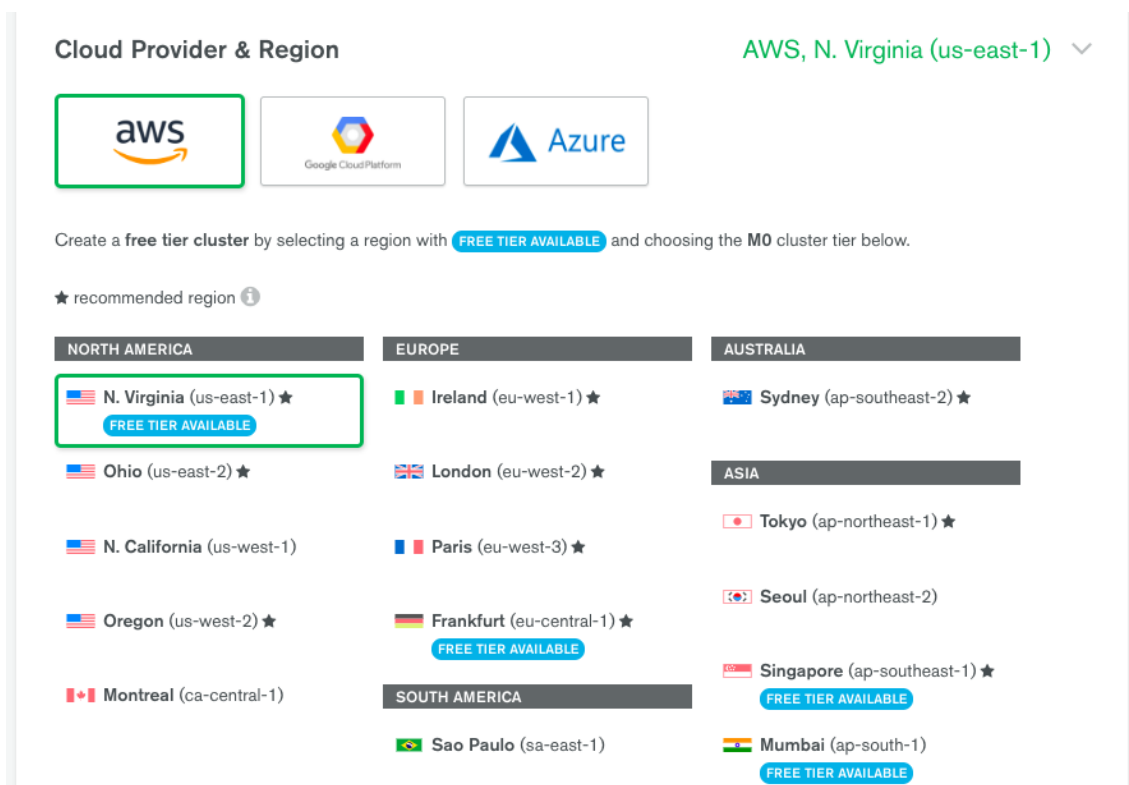
Creating an M0 free tier cluster mflix:

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

1. After creating a new project, you will be prompted to create the first cluster in that project:



2. Choose AWS as the cloud provider, in a Region that has the label **Free Tier Available**:



3. Select *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

<input checked="" type="checkbox"/> 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

- Set *Cluster Name* to **mflix** by scrolling to the appropriate section and clicking on the default name *Cluster0*, and click *Create 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.

- Once you press *Create Cluster*, you will be redirected to the account dashboard. In this dashboard, make sure that the project is named **M220**. If not, go to the *Settings* menu item and change the project name from the default *Project 0* to **M220**:

CONTEXT

TESTORG > MFUX-TEST

mflix-test

PROJECT

Clusters

Alerts

Backup

Users & Teams

Settings

Stitch Apps

Charts

Docs

Support

Settings

Project Settings

General

Project ID

5c33ee75cf09a213bed5b5f6

Project Name

mflix-test

New Project Name

mflix

Cancel

Save

- Next, configure the security settings of this cluster, by enabling the *IP Whitelist* and *MongoDB Users*:



We are deploying your changes: 0 of 3 servers complete (current action: provisioning 3 servers)

TESTORG > MFLIX-TEST

## Clusters


Build a New Cluster

Overview **Security**

MongoDB Users MongoDB Roles IP Whitelist Peering Enterprise Security

+ ADD NEW USER

User Name	Authentication Method	MongoDB Roles	Actions
-----------	-----------------------	---------------	---------



### Create a database user

Set up database users, permissions, and authentication credentials in order to connect to your clusters.

[Learn more](#)

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".

*Note that in a production environment, you would control very tightly the list of IP addresses that can connect to your cluster.*

### Add Whitelist Entry

Add a whitelist entry using either CIDR notation or a single IP address. [Learn more.](#)

**ALLOW ACCESS FROM ANYWHERE**

**Whitelist Entry:**

**Comment:**

☐ Save as temporary whitelist

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

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

You can create new users through *Security -> MongoDB Users -> 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

HIDE

Autogenerate Secure Password

User Privileges

Atlas admin

Read and write to any database

Only read any database

Show Advanced Options

☐ Save as temporary user

Cancel

Add User

8. 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:

SANDBOX

Cluster0

Version 4.0.10

CONNECT

METRICS

COLLECTIONS

...

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: 0 100.0/s

Edit Configuration

Command Line Tools

Load Sample Dataset

Terminate

100 max

Last 6 Hours

**Note: The Mflix database in the Sample Dataset is called "sample\_mflix".**

9. Now you can test the setup by connecting via the **mongo** shell. You can find instructions to connect in the *Connect* section of the cluster dashboard:

## Connect to Cluster0

✓ Setup connection security

✓ Choose a connection method

Connect

- 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
```

 COPY

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.

Go to your cluster *Overview* -> *Connect* -> *Connect Your Application*. Select the option corresponding to MongoDB version 3.6+ and copy the **mongo** connection URI.

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
```

 COPY

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

## 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 the *Atlas* SRV string to import the data (including username and password credentials). If you have earlier versions of MongoDB installed on your machine, please make sure to use the 3.6 version for this project.

Replace the SRV string below with your own:

```
# navigate to mflix directory
cd mflix-java/data/mflix

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

 COPY

## Running the Application

In the `mflix/src/main/resources` directory you can find a file called `application.properties`.

Open this file 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:

```
spring.mongodb.uri=mongodb+srv://m220student:m220pass
```

 COPY

To run Mflix, run the following command:

```
cd mflix
mvn spring-boot:run
```

 COPY

And then point your browser to <http://localhost:5000/>.

It is recommended you use an IDE for this course. Ensure you choose an IDE that supports importing a Maven project. We recommend IntelliJ [Community](#) but you can use the product of your choice.

The first time running the application might take a little longer due to the initial setup process.

## Running the Unit Tests

To run the unit tests for this course, you will use **JUnit**. Each course lab contains a module of unit tests that you can call individually with a command like the following:

 COPY

```
cd mflix  
mvn -Dtest=<TestClass> test
```

For example to run the `ConnectionTest` test your shell command will be:

 COPY

```
cd mflix  
mvn -Dtest=ConnectionTest test
```

Alternatively, if using an IDE, you should be able to run the Unit Tests individually by clicking on a green play button next to them. You will see this demonstrated in the course as we will be using IntelliJ.

Each ticket will contain the command to run that ticket's specific unit tests. When running the Unit Tests or the Application from the shell, make sure that you are in the same directory as the `pom.xml` file.

Proceed to next section