Deployment

Relevel
by Unacademy

Educator Introduction

Give a brief intro about yourself and talk about the features we have implemented so far in the MBA app.

Features we implemented so far:

- CRUD operation APIs on Theatre and Movie models and APIs relating movies and theatres.
- Authentication and Authorization, using JWT for token validation.
- Several middleware, from model data verification to user access level verification.
- Model for booking and transaction, like create booking, cancel booking and set timeout for a booking.

List of Topics Covered

- Hosting.
- Intro of GitHub.
- Repository Creation.
- Get Database for production.
- Intro to Heroku.
- Understand the deployment process of our movie booking application.

What is Hosting?

- Now that our project is working fine in the local system. It would be useful only when used by others, i.e., it is available to the world. Our purpose can be achieved when we make our application available to the users via the internet and that is what hosting means.
- Many cloud service providers help us rent their server where our application code runs 24/7. These services are provided based on a free trial and price-based model for individuals and organizations—for example, Amazon AWS, Heroku, Microsoft Azure, Google Cloud, etc.
- We will deploy our application using the services of Heroku.

Run a project locally

- Before deploying the project, ensure it’s running as expected by running the code and accessing the app locally.
- Execute below command in cmd terminals of root directory:
- **npm start**
- When we run npm start from terminal it will run above script which we need to add in package.json file which will be responsible to start up the application.
- As this is the standard script which many applications hosting service understand for a node application as a run command.

Intro to GitHub

- Now, you can serve your local setup online and make it live via the internet, but it's not easy to maintain and make your server available 24/7. That's why we need some remote server to deploy it and forget, so the cloud/server provider will maintain our application for us.
- We need to take care of one thing before deploying the app, i.e. maintenance and continuous changes to the application.
- We will likely add more features or make some critical changes to our code in the future or maintain different versions of our code.
- Also, things work differently in the industry. There will be many contributors for a single project; frontend developers, UI designers, backend developers, and software testers all working on the same project but locally on their respective machines.
- Finally, all the code from these machines will be merged as a whole, maintaining the flow and integrity of the code and won't break the code's functionality.
- So, here GitHub comes into the picture. As we can integrate Github with the remote server and the local setup to reflect the changes, we can easily change it without going to the server.

Here we will see how Github works and all the basic functionality/commands of GitHub:

1. **Creating Repositories:**
   a. **git init:** to initialise the repo.
2. **Creating branches**
   a. **git checkout -b <branch_name> :** to create new branch
   b. **git checkout <existing_branch_name> :** to get back to the original branch.
3. **Adding files:**
   a. **git add . :** to add all items
   b. **git add <file_name> :** to add specific elements
4. **Commits**
   a. **git commit -m <commit message> :** to commit the changes locally
5. **Pull**
   a. **git pull origin <branch_name> :** to get the latest code from the repo
6. **Push**
   a. **git push origin <branch_name>:** to push local code to the remote repo.

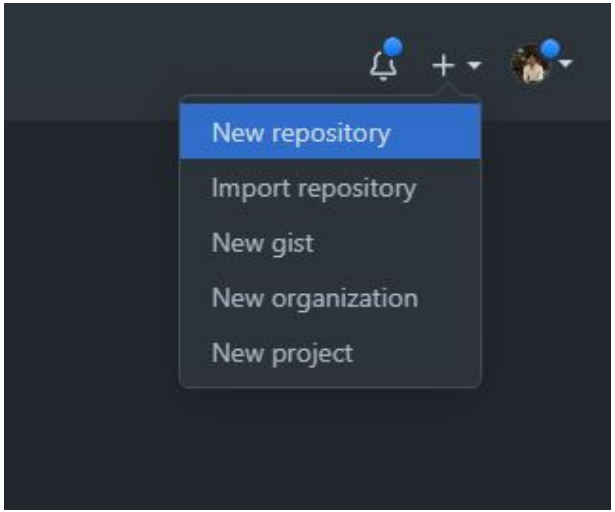
We have 3 ways to do all these steps:

1. Web Interface: Very easy to use but not for complicated steps like resets, rebase, etc.
2. Command Line: Not much user-friendly, but we can perform every operation using this one.
3. Github Desktop: Very much similar to web interface in terms of UI,

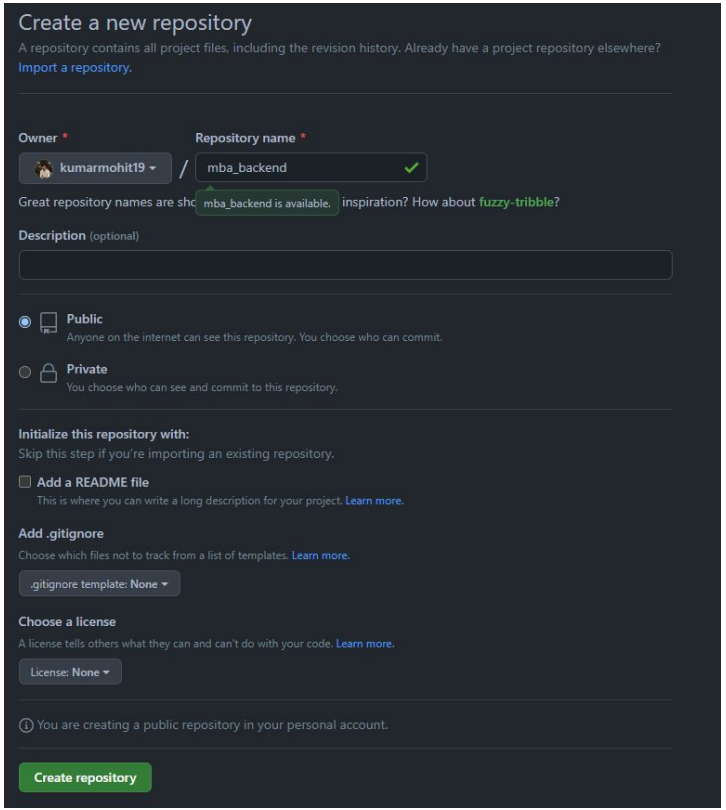
Creating Repositories

Using Web Interface:

1. Go to **Repositories** click on “**New**” or on the “+” icon in the top-right corner.
2. Click on **New repository**



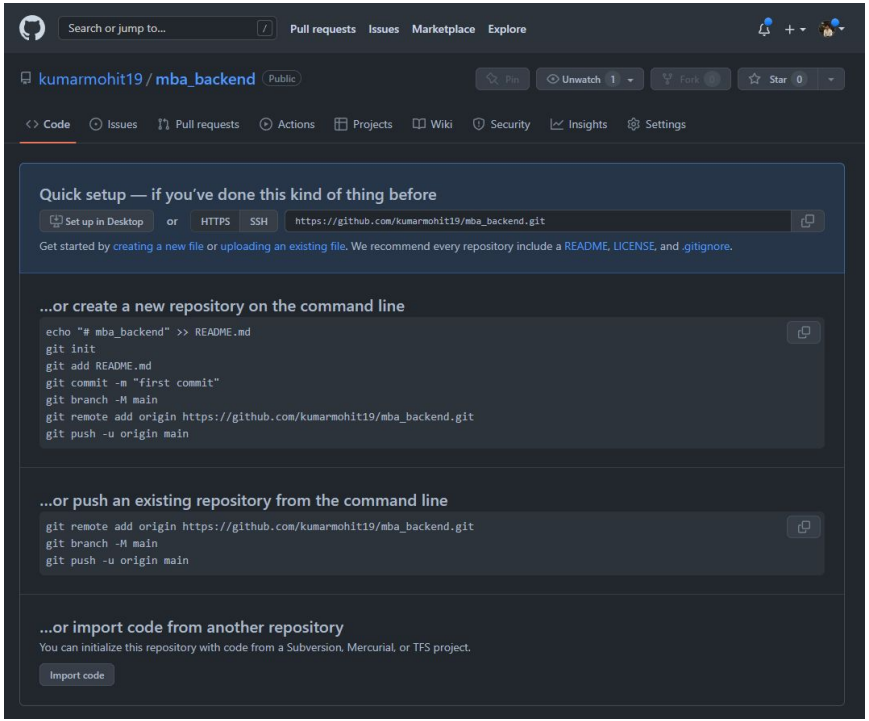
3. Name the repo, create it.



4. Clone it to local, open cmd in the desired directory and write command

git clone <urlname>

5. If you have existing code in the local machine that you want to push to Github it can be done by pasting the folders/files in this cloned directory and we can commit and push the files to the remote repo, using command line.



2. Using the Command line:

- **Step1: Create a directory in local or open cmd for the root directory of an existing project.**

- **Step 2: Type git init will make our current directory a local git repository.**
- Note: Explain to students the difference between a local repo and remote repo,
- Local repo is the app code present in the git repo in local machine
- Remote repo is where collaborators push/merge their code

```
PS D:\Mohit\Relevel\MBA\session8\mba_backend> git init
Initialized empty Git repository in D:/Mohit/Relevel/MBA/session8/mba_backend/.git/
PS D:\Mohit\Relevel\MBA\session8\mba_backend> git add .
```

Step 3: We can add and commit the existing project files to the remote repository.

Step 4: Type git add. (to add all the files) or just git add <file_name> you want to push.

Step 5: Type git commit -m <commit message> where commit message is a string that informs why this commit happens and its purpose.

```
PS D:\Mohit\Relevel\MBA\session8\mba_backend> git commit -m "mba initial commit"
[master (root-commit) b2392ef] mba initial commit
 34 files changed, 5793 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 README.md
 create mode 100644 configs/auth.config.js
 create mode 100644 configs/db.config.js
 create mode 100644 configs/server.config.js
 create mode 100644 controllers/auth.controller.js
 create mode 100644 controllers/booking.controller.js
 create mode 100644 controllers/movie.controller.js
 create mode 100644 controllers/payment.controller.js
 create mode 100644 controllers/theatre.controller.js
 create mode 100644 controllers/user.controller.js
 create mode 100644 middlewares/authjwt.js
 create mode 100644 middlewares/index.js
 create mode 100644 middlewares/verifyBookingReqBody.js
 create mode 100644 middlewares/verifyMovieReqBody.js
 create mode 100644 middlewares/verifyPaymentReqBody.js
 create mode 100644 middlewares/verifyTheatreReqBody.js
 create mode 100644 middlewares/verifyUserReqBody.js
 create mode 100644 models/booking.model.js
 create mode 100644 models/movie.model.js
 create mode 100644 models/payment.model.js
 create mode 100644 models/theatre.model.js
 create mode 100644 models/user.model.js
 create mode 100644 package-lock.json
 create mode 100644 package.json
 create mode 100644 routes/auth.routes.js
 create mode 100644 routes/booking.routes.js
 create mode 100644 routes/movie.routes.js
 create mode 100644 routes/payment.routes.js
 create mode 100644 routes/theatre.routes.js
 create mode 100644 routes/user.routes.js
 create mode 100644 server.js
 create mode 100644 utils/NotificationClient.js
 create mode 100644 utils/constants.js
PS D:\Mohit\Relevel\MBA\session8\mba_backend>
```

Step 6: Now we have to connect it to GitHub:

git remote add origin
git@github.com:username/<repo_name>

```
PS D:\Mohit\Relevel\MBA\session8\mba_backend> git remote add origin https://github.com/kumarmohit19/mba_backend.git
PS D:\Mohit\Relevel\MBA\session8\mba_backend>
```

Step 7: Finally, push using: git push origin master.

```
PS D:\Mohit\Relevel\MBA\session8\mba_backend> git push -u origin master
Enumerating objects: 42, done.
Counting objects: 100% (42/42), done.
Delta compression using up to 4 threads
Compressing objects: 100% (41/41), done.
Writing objects: 100% (42/42), 56.22 KiB | 1.76 MiB/s, done.
Total 42 (delta 6), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (6/6), done.
To https://github.com/kumarmohit19/mba_backend.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
PS D:\Mohit\Relevel\MBA\session8\mba_backend>
```

Code: https://github.com/kumarmohit19/mba_backend

Get Hosted MongoDB database:

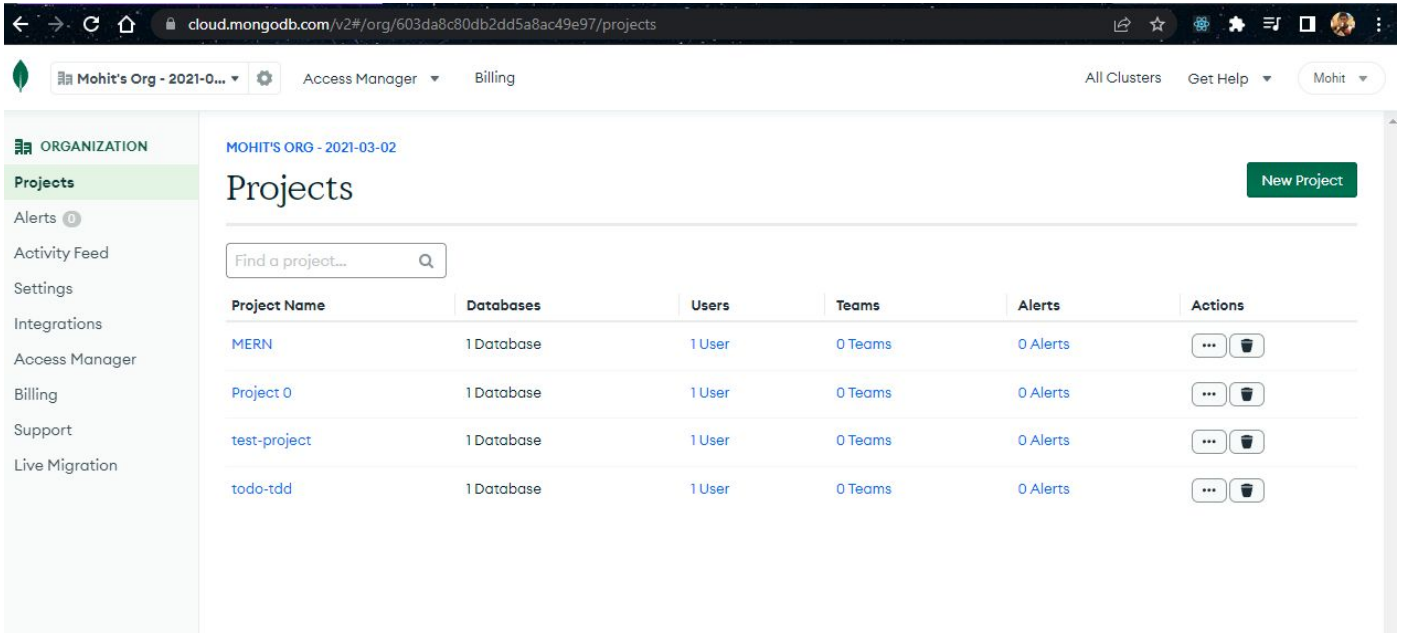
Why hosted DB

When we will be going to host our application on Heroku or any other platform it will be needing a database which will be live 24x7.
Now while developing the application, we used MongoDB in our local machine which was running in background and serving as database, which will not be working for application hosted on cloud service app, so we will be opting for a MongoDB cloud database hosting service here.

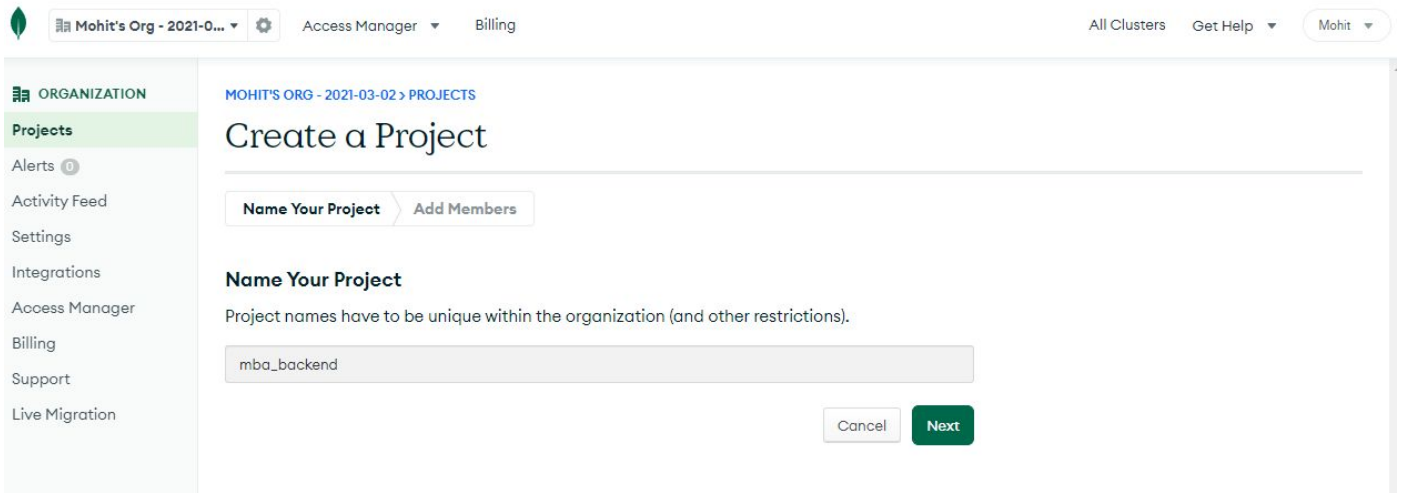
Steps to follow to get cloud MongoDB

First create an account on MongoDB atlas cloud database and then follow the steps.

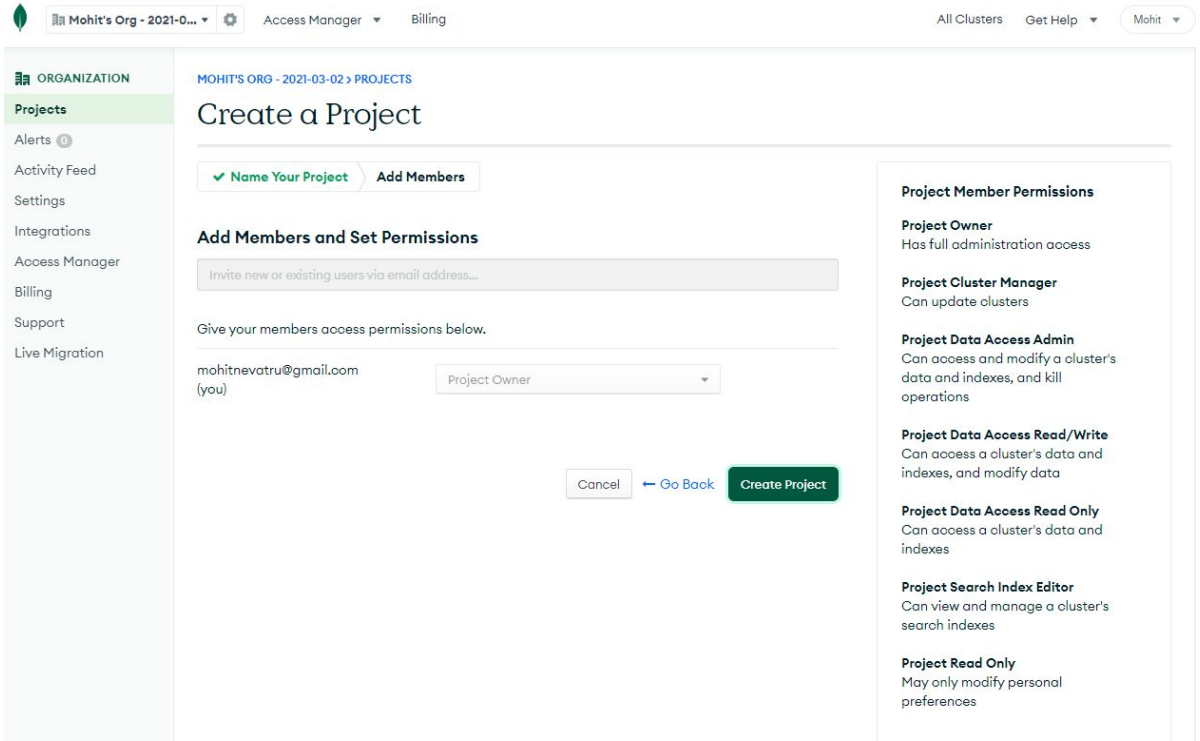
Step 1: Go to the home page and then navigate to Projects menu on left hand side bar. Here, you will see all your project which may be empty unlike the below image. Now, click on “New Project” on right hand top.



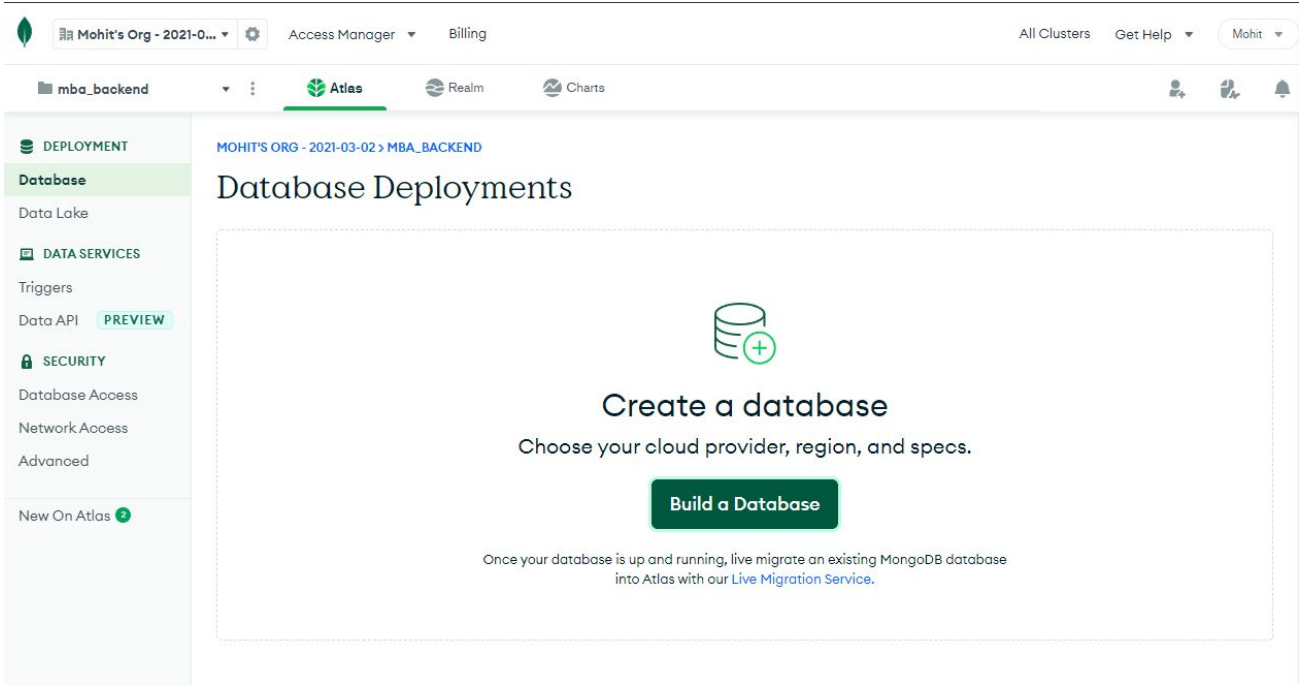
Step 2: Give a name to your project and click Next.



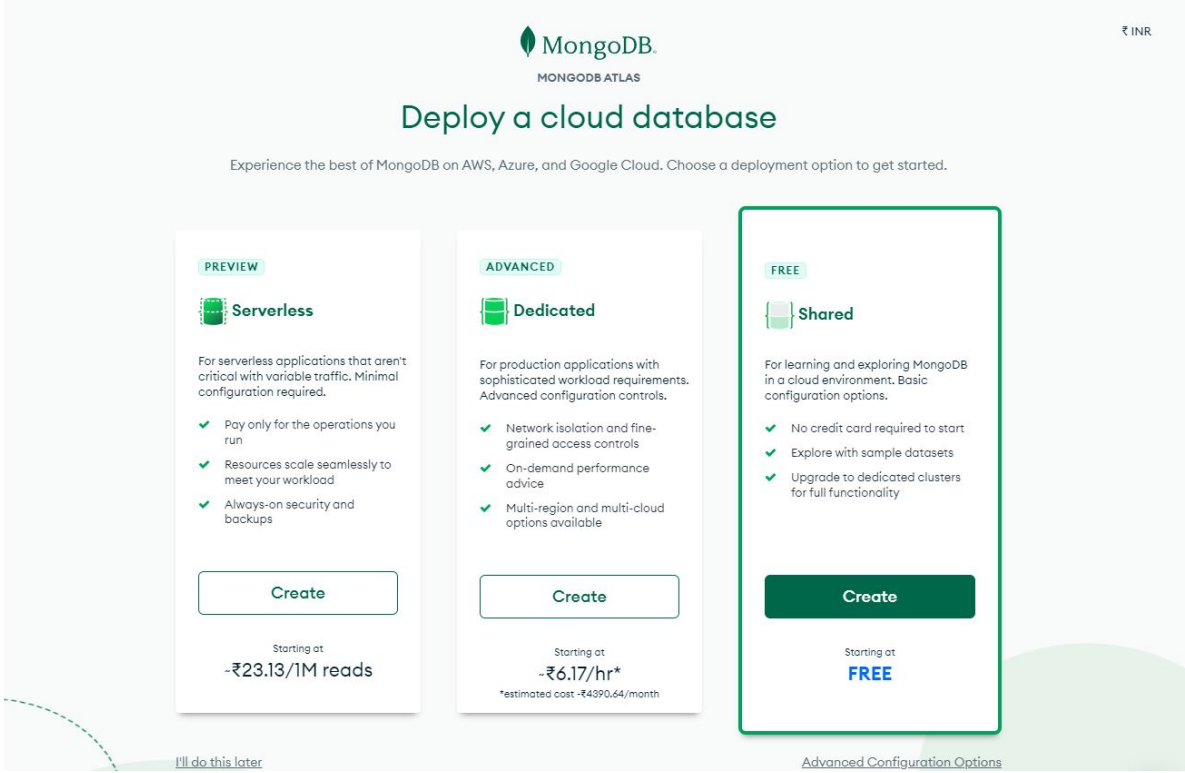
Step 3: Here, you do not need to changes anything unless you want to provide access to this project to others. Click on Create Project.



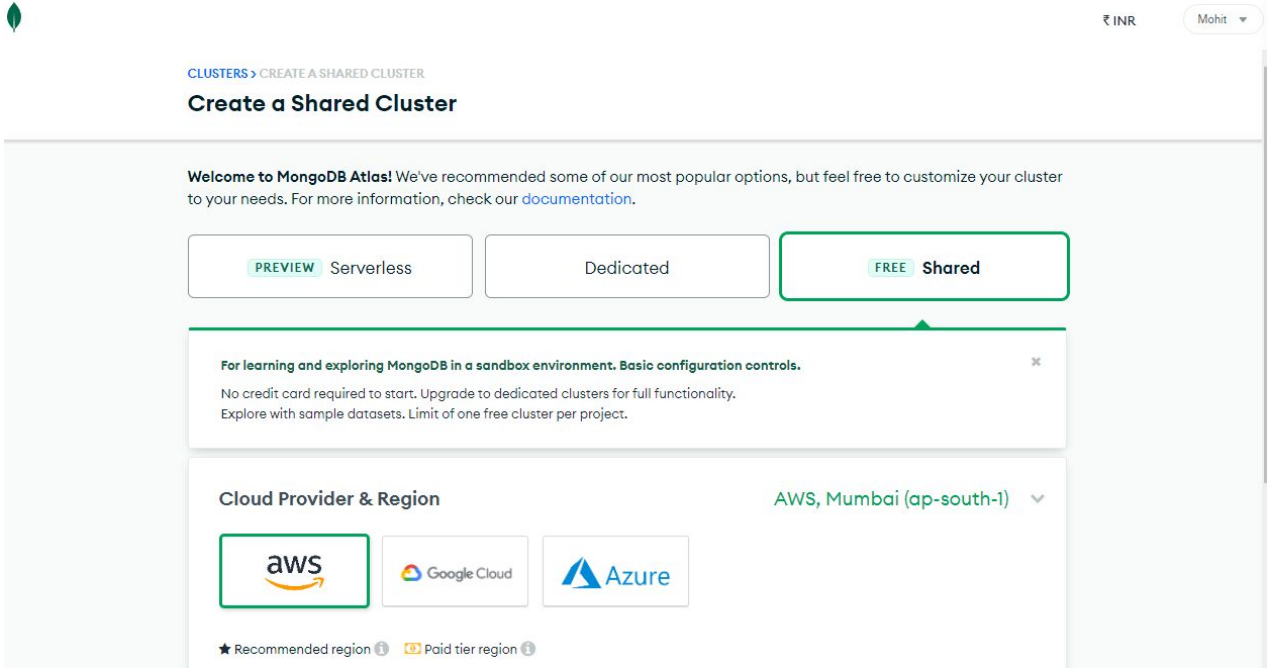
Step 4: Now, we will land on the database page where we need to create a new database, let's get started and click on Build a Database.



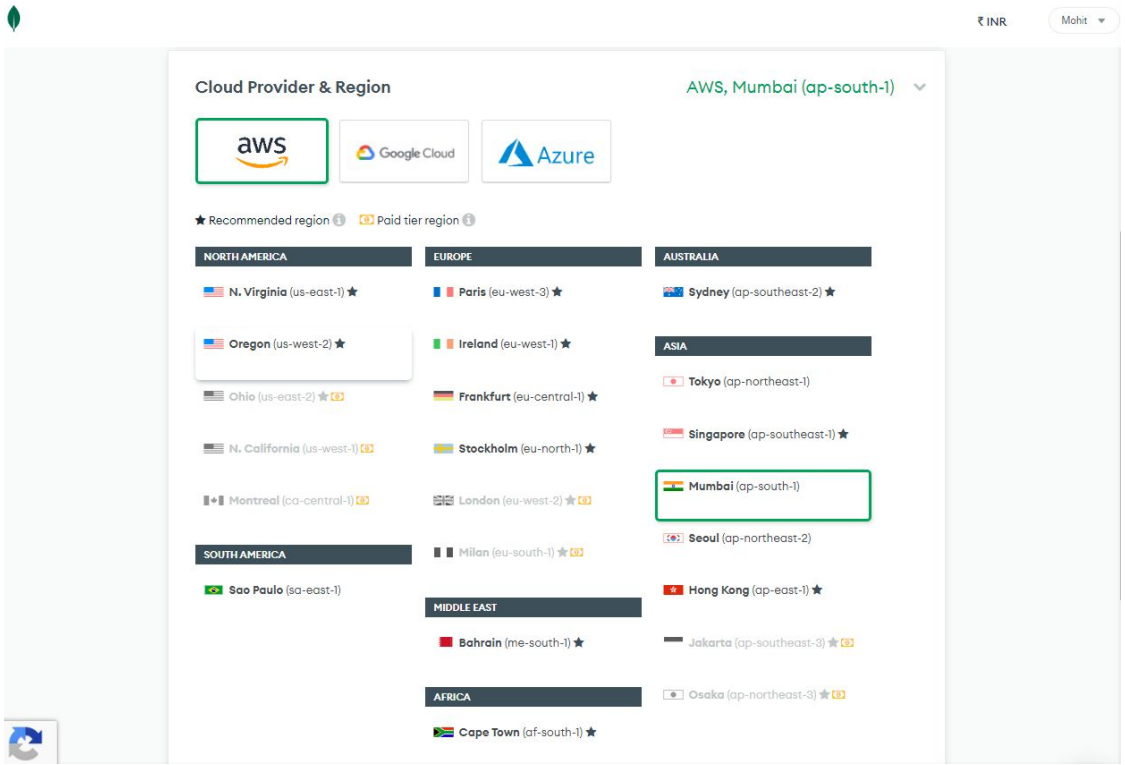
Step 5: Here, we will go with shared plan in which we will get the database with zero cost.



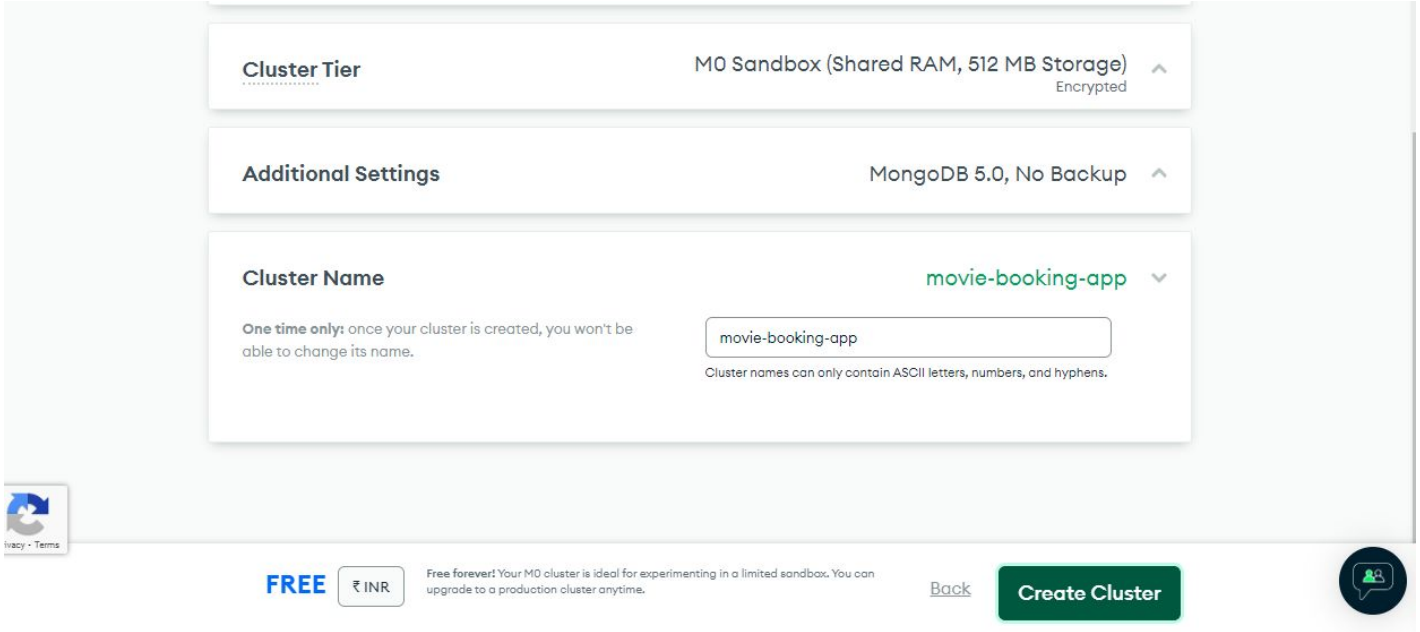
Step 6: We will be redirected to next page where most of the things are selected, we just need to verify and move forward.



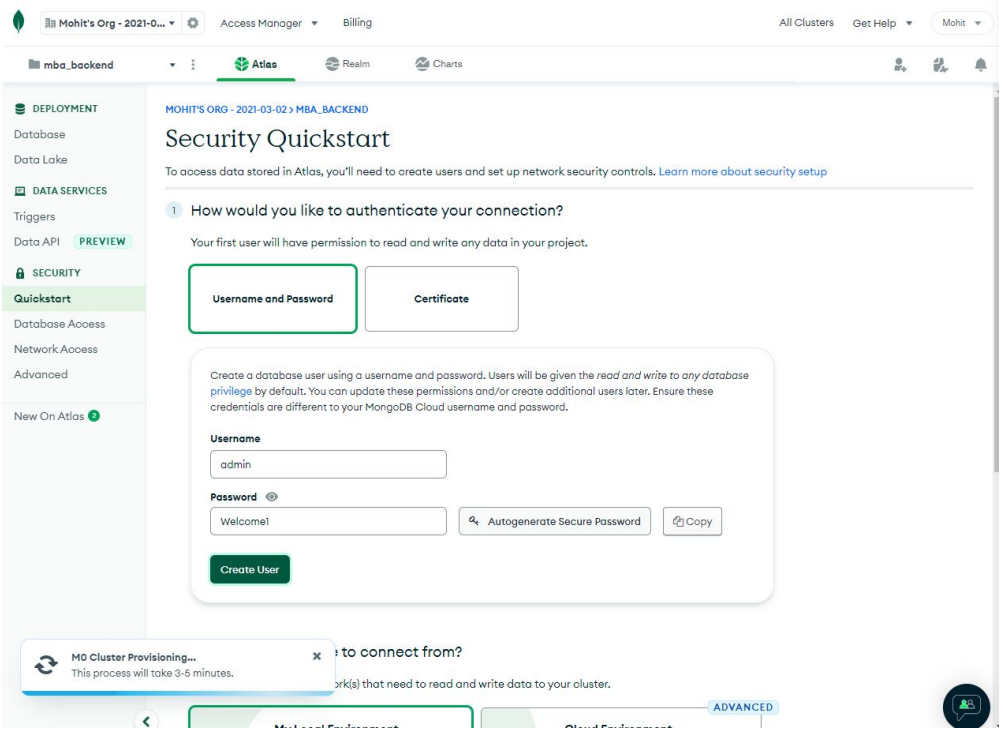
Step 7: Cloud provider region should be nearest available server. Which is Mumbai here.



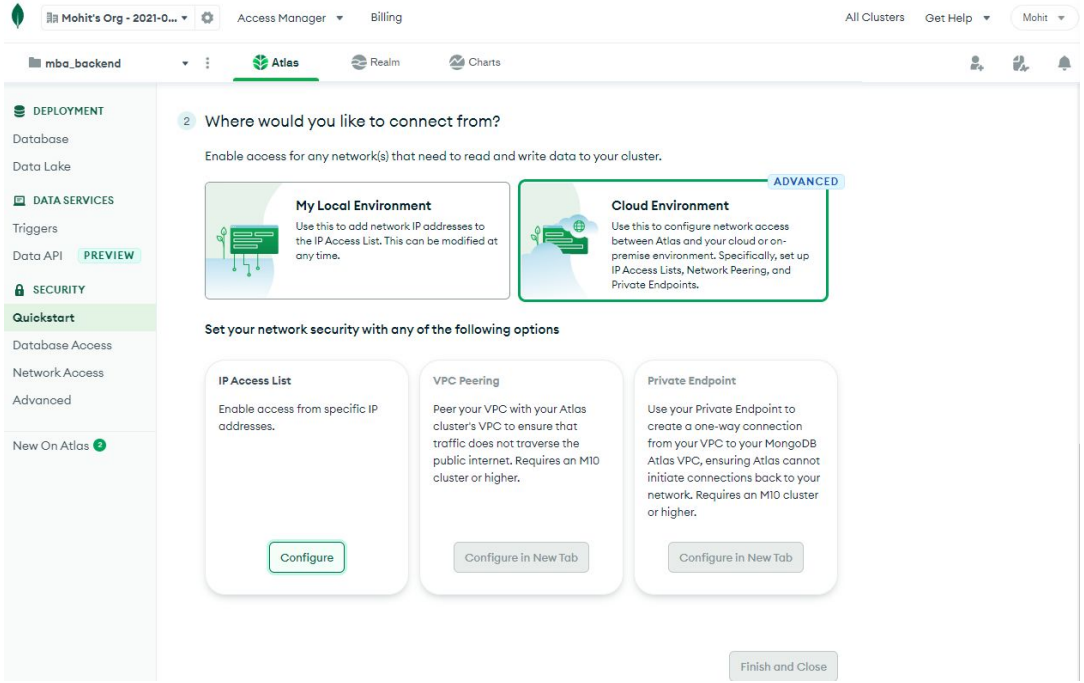
Step 8: Give a proper cluster name here, and then click on Create Cluster, which is basically a database.



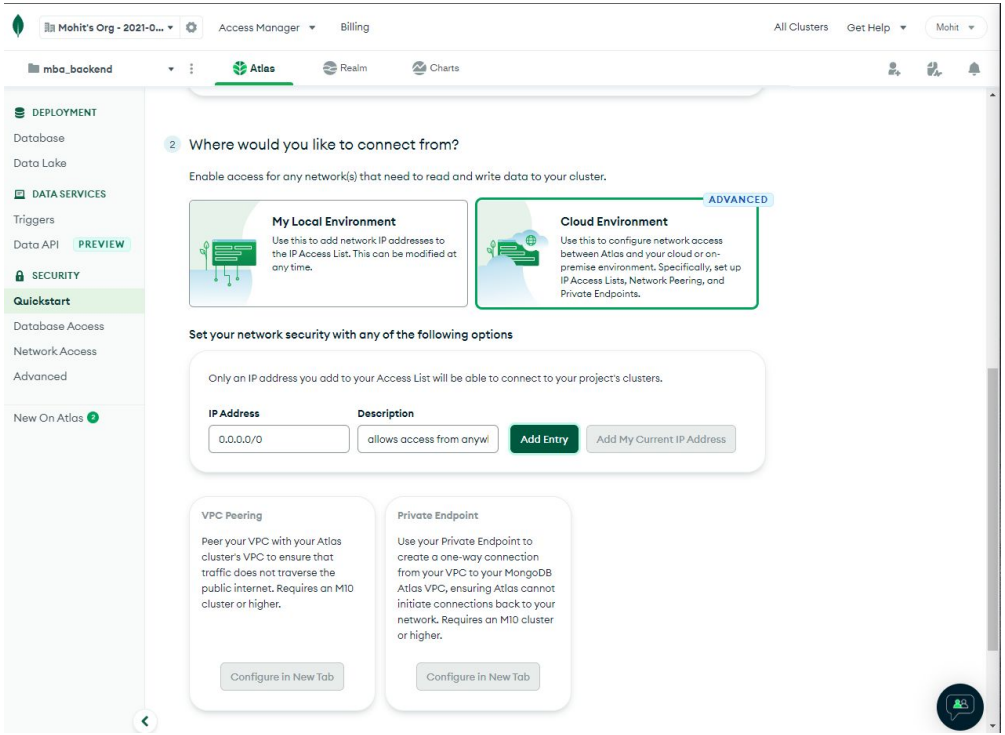
Step 9: Now, we will be redirected to Quickstart page, where we have to create user as show below.



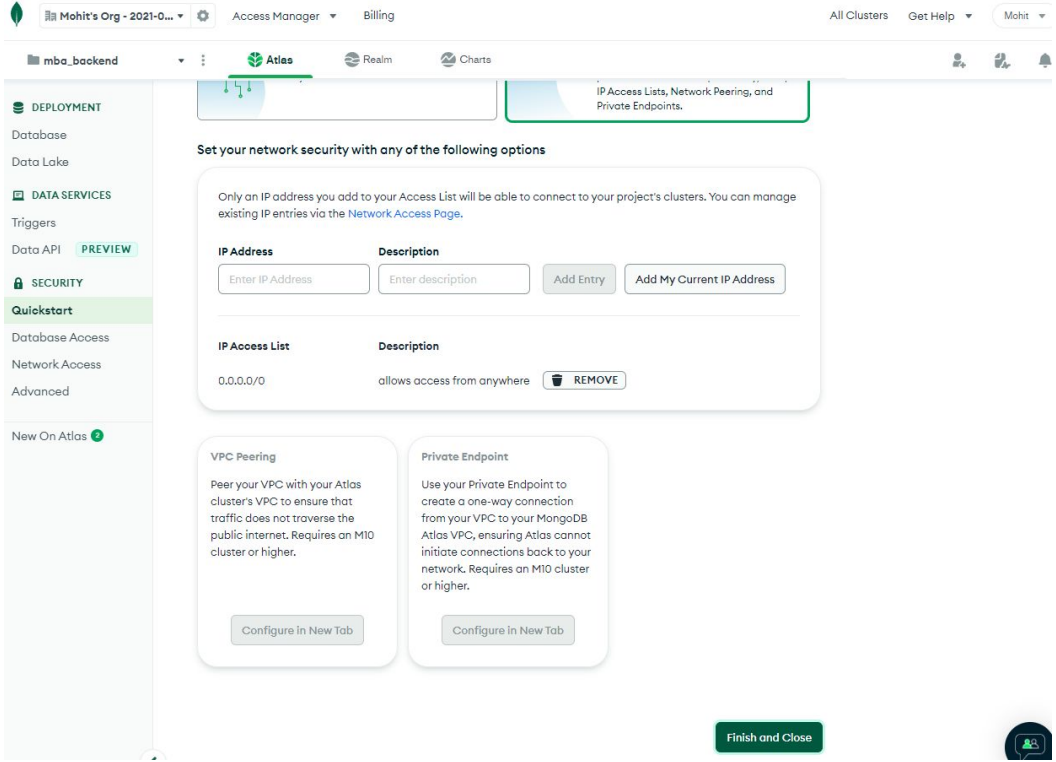
Step 10: Next, we need to setup the server access, where we we like to connect from, here since we are deploying our app on Heroku we have to select Cloud Environment and then the Configure button of IP Access List tab.



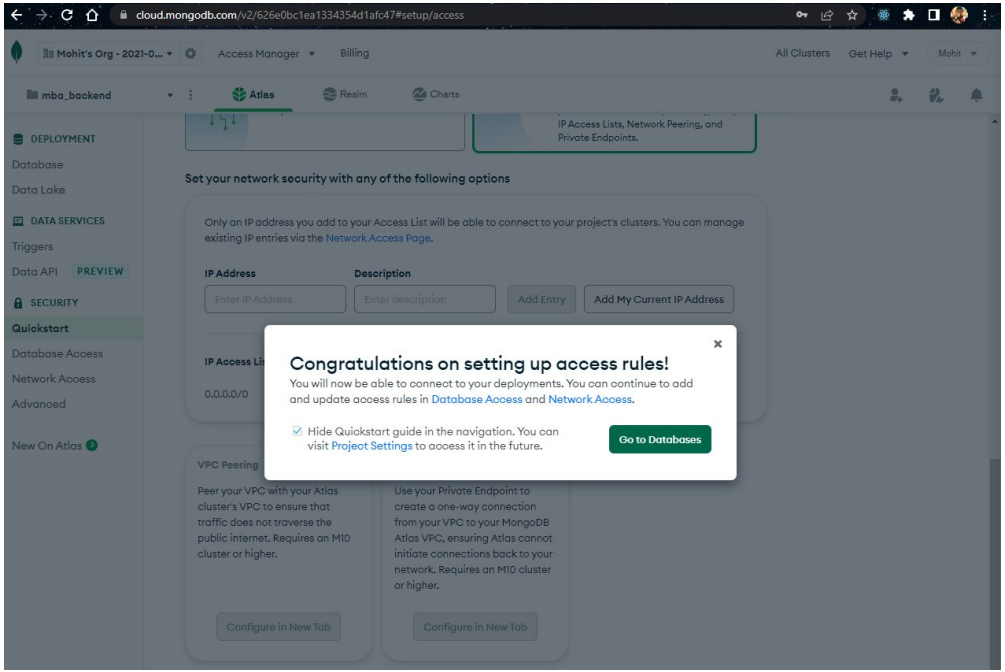
Step 11: We want to access it from anywhere for now, so we will be providing IP address 0.0.0.0/0 and the description as same.



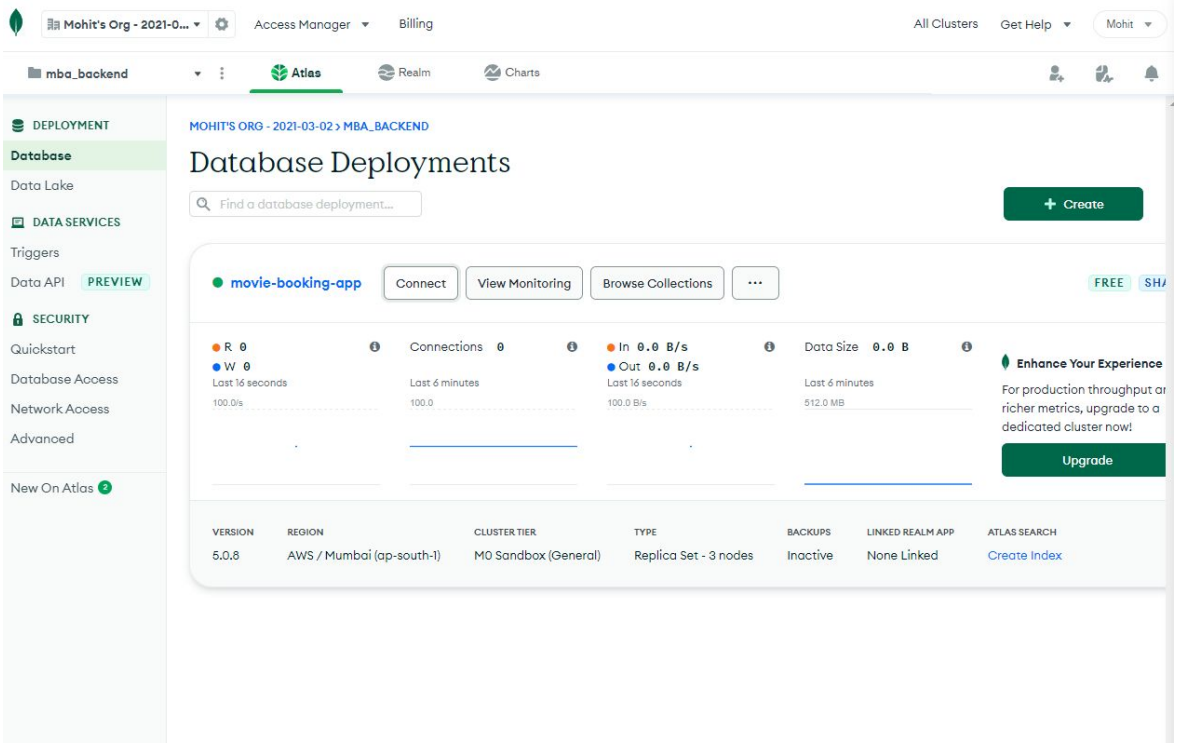
Step 12: Click on Finish and Close



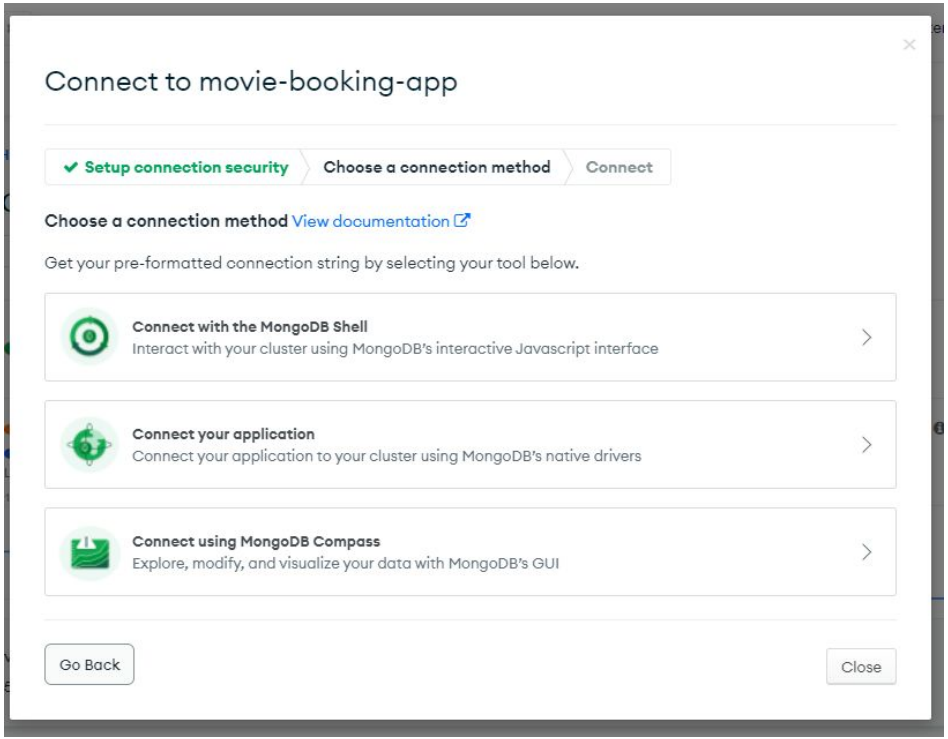
Step 13: Once step 12 is successful we will see the below prompt. Here, click on Go to Database button.



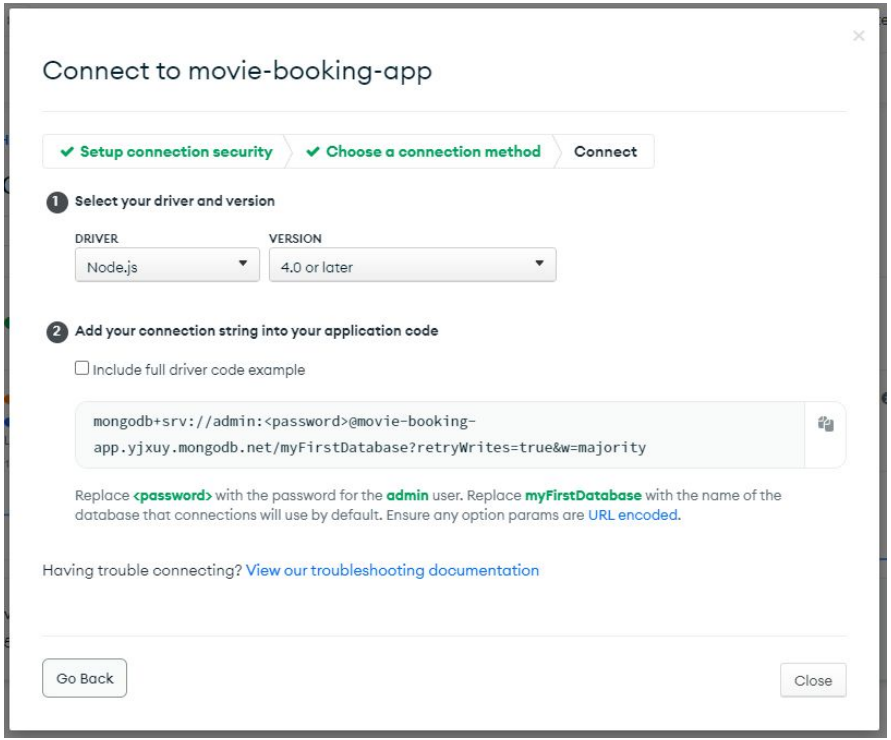
Step 14: This is how the database page will look, now let's use this database and connect to it with our express app. Click on Connect button.



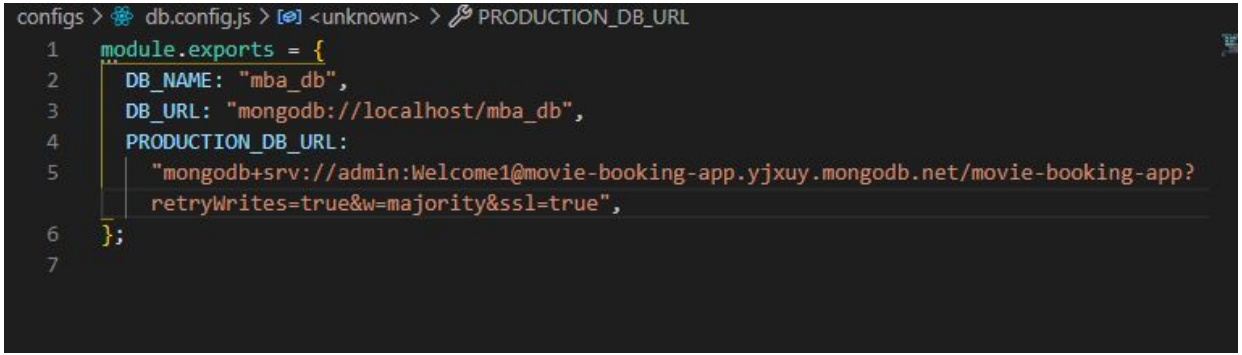
Step 15: Select Connect your application option as we want to connect to the DB using our node application.



Step 16: Here, you will see the connection string and proper notes to replace the <password> and myFirstDatabase string with actual ones.



Step 17: Adding production database URL into our app



Step 18: Modify server.js to use production database for production environment:

```
/**
 * DB Connection initialization
 */
if (process.env.NODE_ENV === "production") {
  try {
    console.log(process.env.NODE_ENV);
    // Connect to the MongoDB cluster
    mongoose.connect(
      dbConfig.PRODUCTION_DB_URL,
      { useNewUrlParser: true, useUnifiedTopology: true },
      () => {
        console.log("Connected to production Mongo DB ");
        init();
      }
    );
  } catch (err) {
    console.log("Could not connect to the datababse: " + err.message);
  }
} else {
  console.log(process.env.NODE_ENV);
  mongoose.connect(
    dbConfig.DB_URL,
    () => {
      console.log("Connected to Mongo DB ");
      init();
    },
    (err) => {
      console.log("Error :", err.message);
    }
  );
}

```

Intro to Heroku

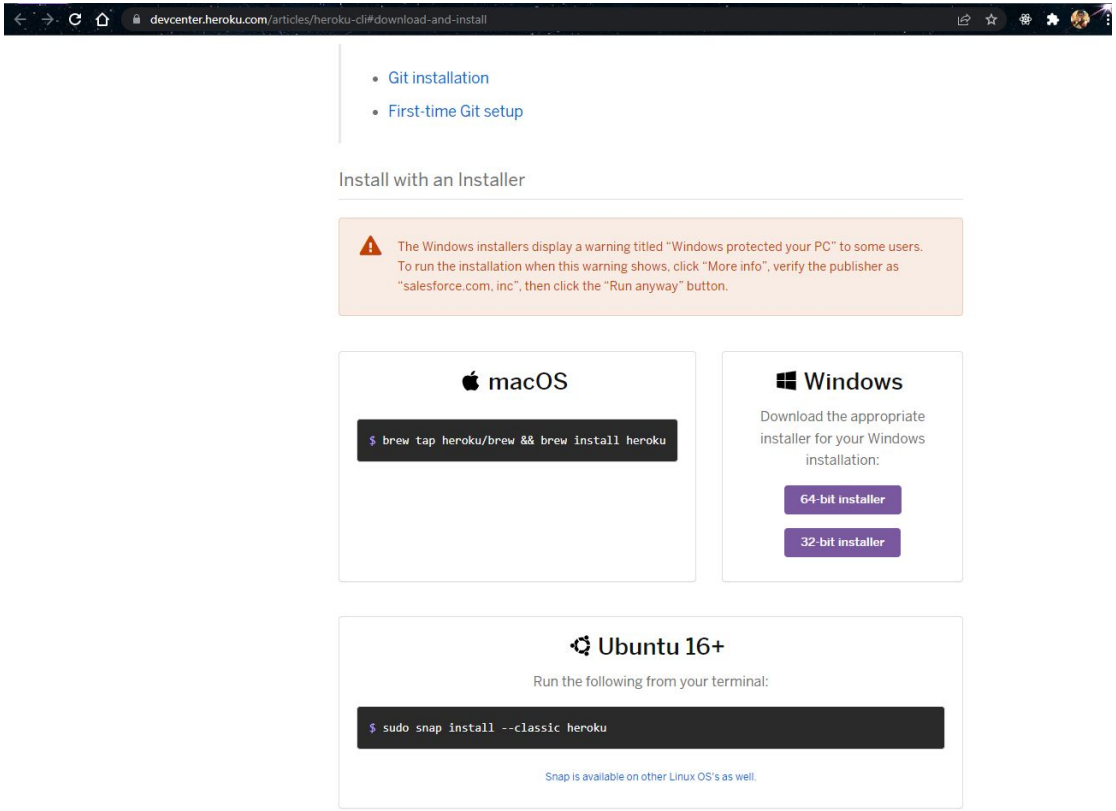
Heroku is a cloud platform that helps developers maintain, build, deliver, and scale apps.

Deploy App:

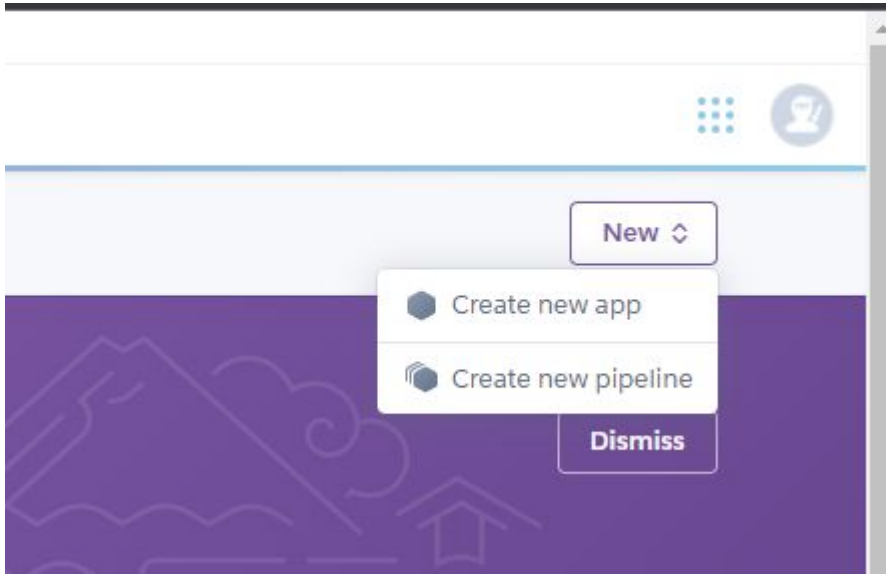
Let us use our application on Heroku, but before that, we have to do some steps:

Create an account on Heroku, (nothing much here just fill the prompted details and verify account for specified email id).

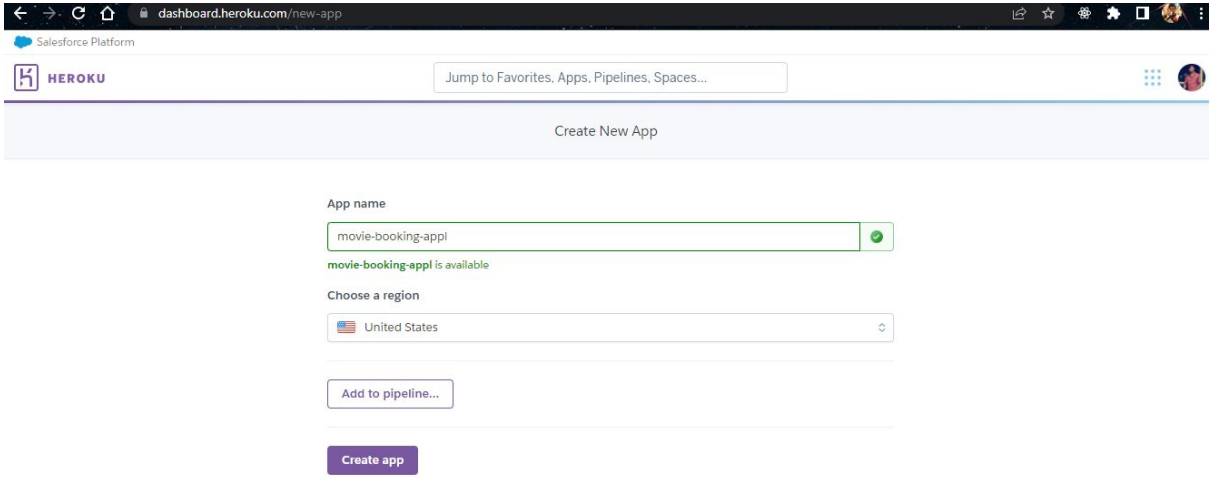
1. Download Heroku CLI:
https://devcenter.heroku.com/articles/heroku-cli#download-and-install


devcenter.heroku.com/articles/heroku-cli#download-and-install
Git installation
First-time Git setup
Install with an Installer
The Windows installers display a warning titled "Windows protected your PC" to some users. To run the installation when this warning shows, click "More info", verify the publisher as "salesforce.com, inc", then click the "Run anyway" button.
macOS
$ brew tap heroku/brew && brew install heroku
Windows
Download the appropriate installer for your Windows installation:
64-bit installer
32-bit installer
Ubuntu 16+
Run the following from your terminal:
$ sudo snap install --classic heroku
Snap is available on other Linux OS's as well.

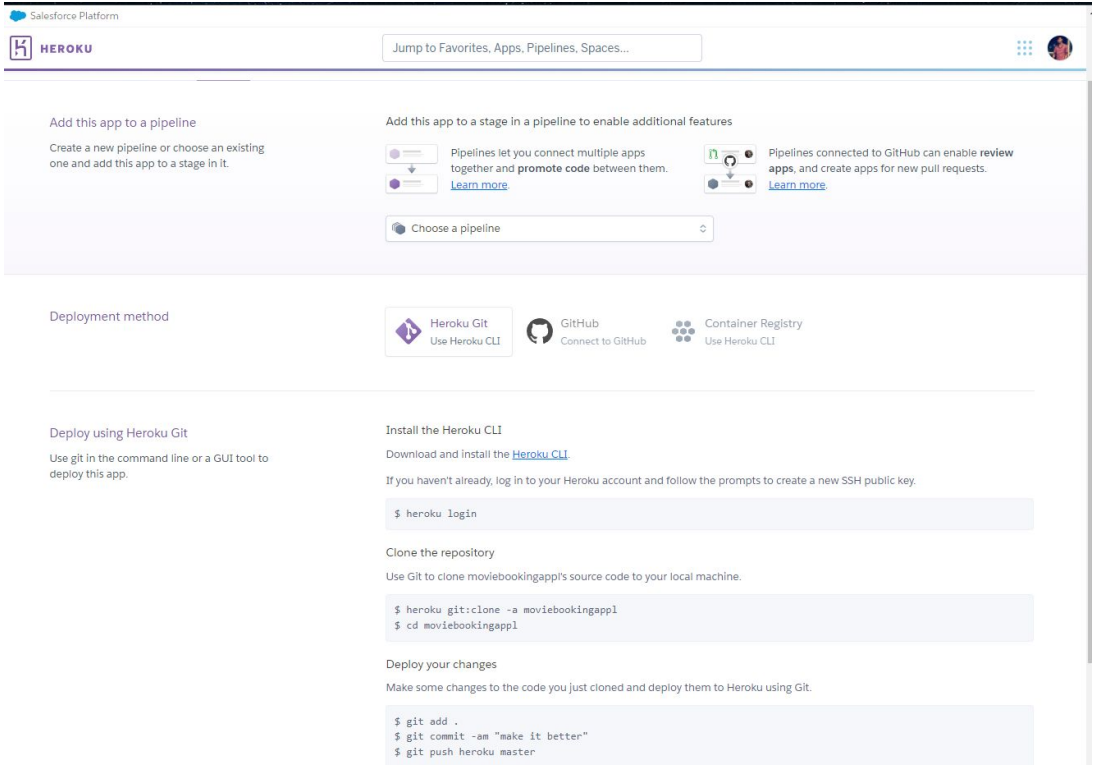
2. Go to the Heroku website https://dashboard.heroku.com/apps and log in. After login, click on “New.” and click “Create new app”.



3. The name must be unique, and click on Create app.



4. Now, we have to install Heroku CLI to proceed with the next steps:



5. After successful installation of Heroku, go to you project root directory and open command prompt and do Heroku login a shown below.

```
PS D:\Mohit\Relevel\MBA\session8\mba_backend> heroku login
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/cli/browser/691fe342-78b6-47d3-b6aa-9928a52f8723
?requestor=SFMyNTY.g2gDbQAAAA8xMjIuMTc2LjE5MC4xMTZuBgDY_7h-gAFiAAFRgA.pgsDctnQi-ySJg58upBgI5haAoQbMr
SGNPWrK0s_pUQ
Logging in... done
Logged in as mohitnevatru@gmail.com
```

6. After successful login we will assign another remote repo for Heroku using command:

`heroku git:remote -a <application-name-you-set-on-heroku>`

```
PS D:\Mohit\Relevel\MBA\session8\mba_backend> heroku git:remote -a moviebookingappl
set git remote heroku to https://git.heroku.com/moviebookingappl.git
PS D:\Mohit\Relevel\MBA\session8\mba_backend> git remote -v
heroku  https://git.heroku.com/moviebookingappl.git (fetch)
heroku  https://git.heroku.com/moviebookingappl.git (push)
origin  https://github.com/kumarmohit19/mba_backend.git (fetch)
origin  https://github.com/kumarmohit19/mba_backend.git (push)
```

7. Now, run the push command to trigger the deployment.
git push heroku master

```
PS D:\Mohit\Relevel\MBA\session8\mba_backend> git push heroku master
Enumerating objects: 42, done.
Counting objects: 100% (42/42), done.
Delta compression using up to 4 threads
Compressing objects: 100% (41/41), done.
Writing objects: 100% (42/42), 56.22 KiB | 1.94 MiB/s, done.
Total 42 (delta 6), reused 0 (delta 0), pack-reused 0
remote: Compressing source files... done.
remote: Building source:
remote:
remote: -----> Building on the Heroku-20 stack
remote: -----> Determining which buildpack to use for this app
remote: -----> Node.js app detected
remote:
remote: -----> Creating runtime environment
remote:
remote:        NPM_CONFIG_LOGLEVEL=error
remote:        NODE_VERBOSE=false
remote:        NODE_ENV=production
remote:        NODE_MODULES_CACHE=true
remote:
remote: -----> Installing binaries
remote:        engines.node (package.json):  unspecified
remote:        engines.npm (package.json):   unspecified (use default)
remote:
remote:        Resolving node version 16.x...
remote:        Downloading and installing node 16.15.0...
remote:        Using default npm version: 8.5.5
remote:
remote: -----> Installing dependencies
remote:        Installing node modules
remote:
remote:        added 216 packages, and audited 217 packages in 5s
remote:
remote:        23 packages are looking for funding
remote:          run `npm fund` for details
remote:
remote:        found 0 vulnerabilities
remote:
```

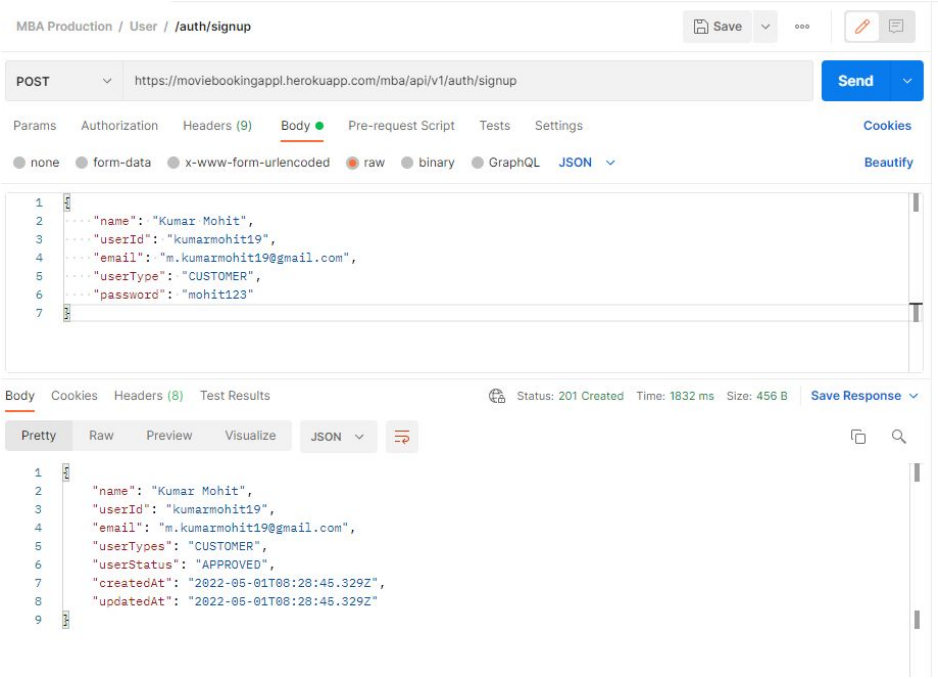
```
remote:        Resolving node version 16.x...
remote:        Downloading and installing node 16.15.0...
remote:        Using default npm version: 8.5.5
remote:
remote: -----> Installing dependencies
remote:        Installing node modules
remote:
remote:        added 216 packages, and audited 217 packages in 5s
remote:
remote:        23 packages are looking for funding
remote:          run `npm fund` for details
remote:
remote:        found 0 vulnerabilities
remote:
remote: -----> Build
remote:
remote: -----> Caching build
remote:        - node_modules
remote:
remote: -----> Pruning devDependencies
remote:
remote:        up to date, audited 102 packages in 623ms
remote:
remote:        7 packages are looking for funding
remote:          run `npm fund` for details
remote:
remote:        found 0 vulnerabilities
remote:
remote: -----> Build succeeded!
remote: -----> Discovering process types
remote:        Procfile declares types     -> (none)
remote:        Default types for buildpack -> web
remote:
remote: -----> Compressing...
remote:        Done: 34.9M
remote: -----> Launching...
remote:        Released v3
remote:        https://moviebookingappl.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/moviebookingappl.git
 * [new branch]      master -> master
```

Once you see the Build succeeded! Message your deployment will be successful and can access the application on below given URL.

Deployed code link: https://moviebookingappl.herokuapp.com/

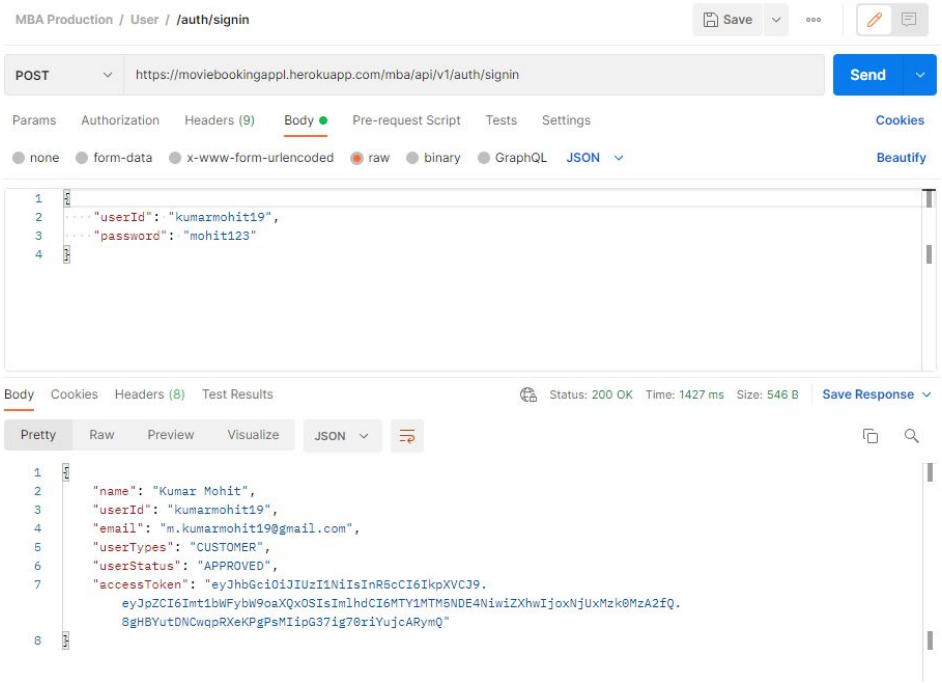
Testing our hosted App

SignUp: POST /auth/signup route



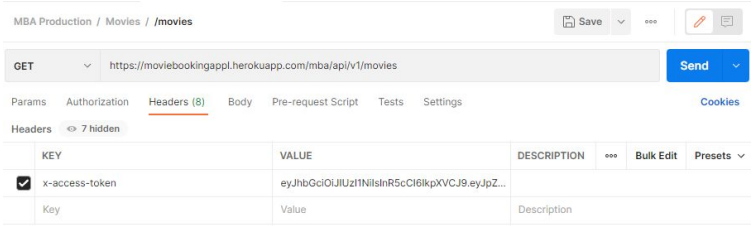
Testing our hosted App

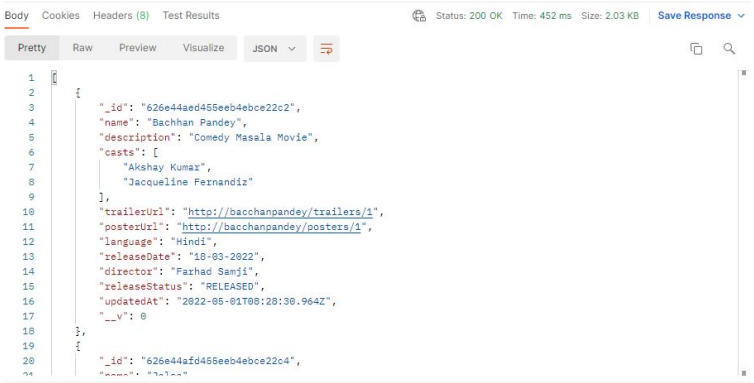
SignIn: POST /auth/signin route



Testing our hosted App

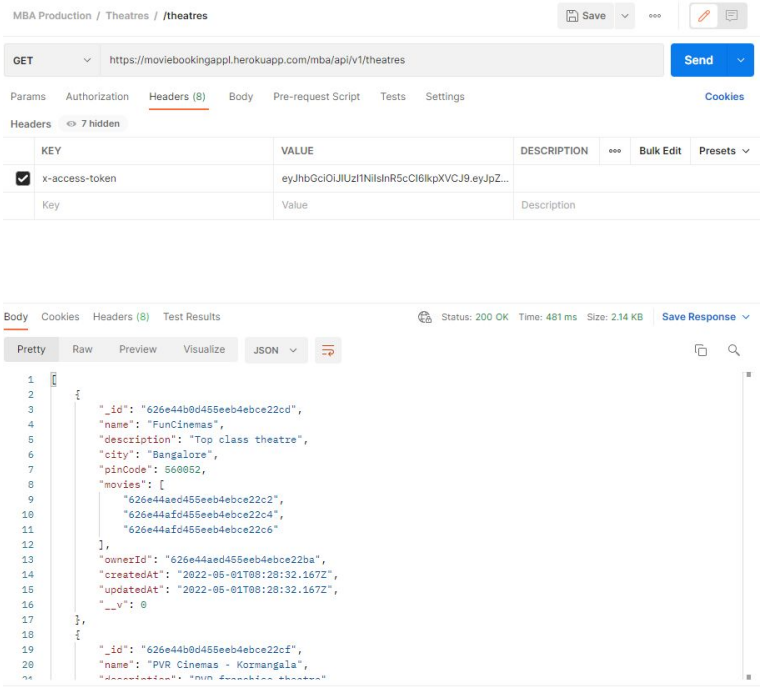
Using access token, GET /movies route





Testing our hosted App

And, GET /theatres route



MCQ's:

1. What is the meaning of origin in git command?

A. Origin is the name of the branch.
B. It's the name of the repo.
C. It signifies a remote name where the user wants to push or pull the changes.
D. None of the above.

2. What does a checkout argument do in git command?

A. Checkout means to go out of the repo.
B. It means the act of switching between different versions of a target entity or the branches of the repository.
C. It is used to commit the change and push in one command.
D. None of the above.

3. What Does a Heroku app URL look like with the app name "alpha"?

A. www.herokuapp.com/alpha
B. alpha.heroku.com
C. alpha.herokuapp.com
D. www.alpha.herokuapp.com

4. Is it possible to change the domain of the application deployed on Heroku.

A. No.

B. Yes

5. Which branch is used for deployment from GitHub to Heroku?

A. master

B. We can choose.

C. origin

D. None of the above.

Thank You!