MIRACLE
SOFTWARE SYSTEMS

Creating CRUD Application with MongoDB, IBM Cloud and Node JS

AP Cloud 2018 Workshop

AP Cloud Team
Miracle Software Systems, Inc.

www.miraclesoft.com

Creating CRUD Application with MongoDB, IBM Cloud and Node JS

Goal

In this Lab we will guide you how to create a CRUD Application with MongoDB using Node JS as Backend, Bootstrap and HTML as frontend. Then we will push the application into IBM Cloud.

Pre-Requisites

The following installations will need to be completed for this lab to be run successfully,

- mLabs Account for storing the data in MongoDB
- Account with IBM Cloud
- Node JS and NPM installed
- Text Editor such as sublime Text (or) notepad++

Technology Involved

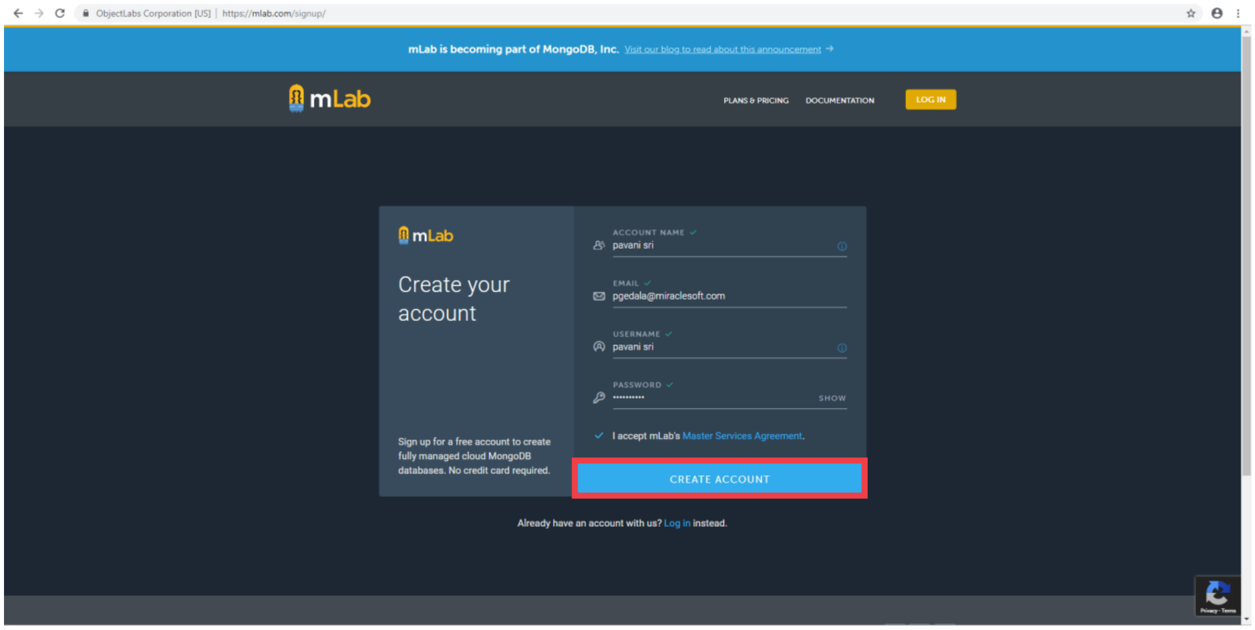
- Server Side - Node JS
- Client Side(HTML, CSS and Bootstrap)
- Cloud Technologies - IBM Cloud
- Database - MongoDB

Lab Steps

Let's get started with the lab!

Step #1 | Creating a mLabs account

For creating a new mLabs account visit the below link and click on **Sign Up** button,
https://mlab.com/signup/


ObjectLabs Corporation [US] | https://mlab.com/signup/
mLab is becoming part of MongoDB, Inc. Visit our blog to read about this announcement →
mLab
PLANS & PRICING
DOCUMENTATION
LOG IN
Create your account
ACCOUNT NAME
EMAIL
USERNAME
PASSWORD
SHOW
CREATE ACCOUNT

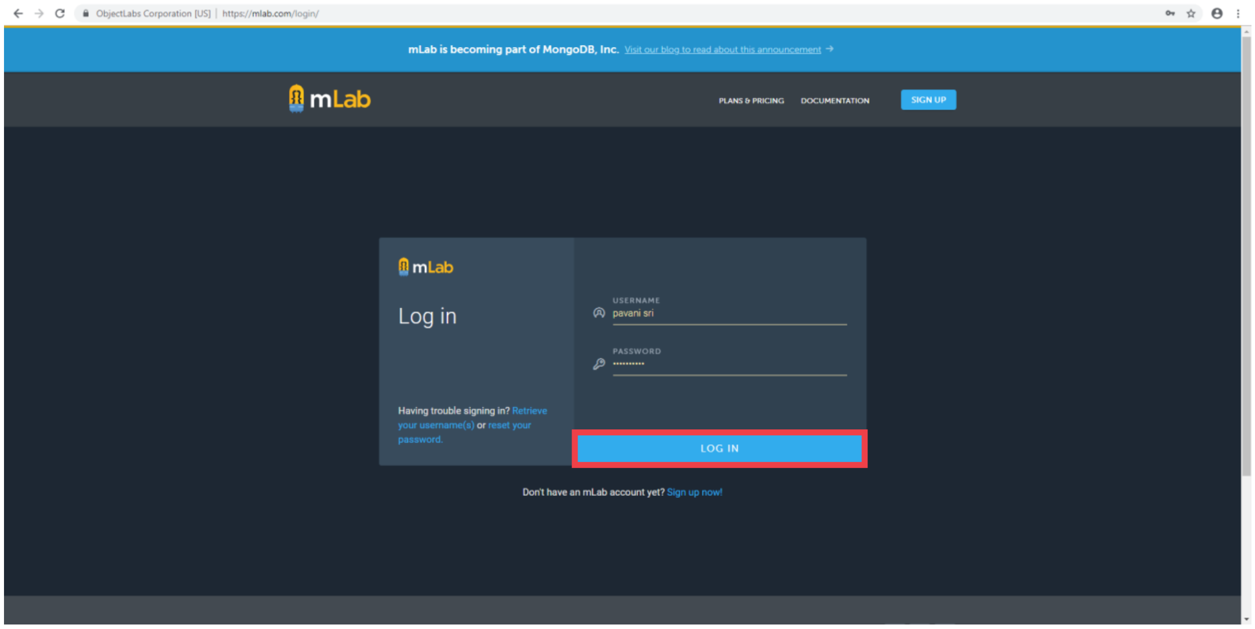
Account name: pavani sri
Email: pgedala@miraclesoft.com
Username: pavani sri
Password: ••••••••••

✓ I accept mLab's Master Services Agreement.

Sign up for a free account to create fully managed cloud MongoDB databases. No credit card required.

Already have an account with us? Log in instead.

Provide the necessary details and click on **Create Account** button for new registration to mLabs.

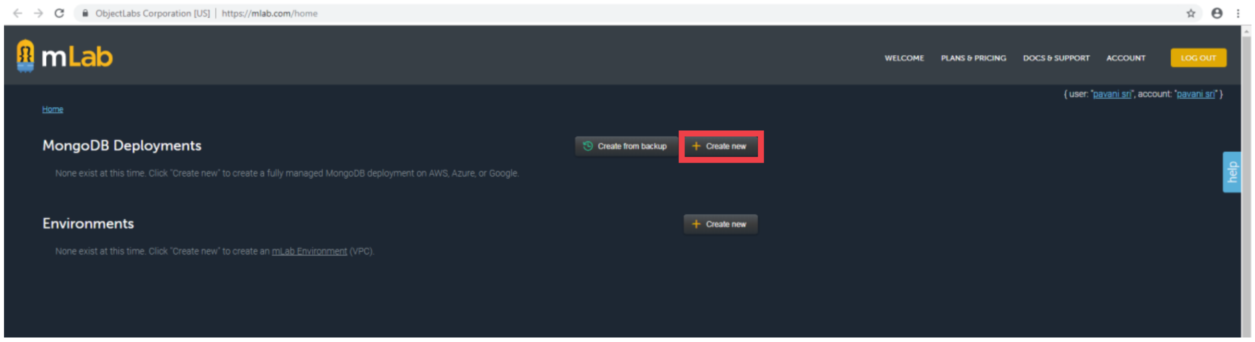

ObjectLabs Corporation [US] | https://mlab.com/login/
mLab is becoming part of MongoDB, Inc. Visit our blog to read about this announcement →
mLab
PLANS & PRICING
DOCUMENTATION
SIGN UP
Log in
USERNAME
PASSWORD
LOG IN

Username: pavani sri
Password: ••••••••••

Having trouble signing in? Retrieve your username(s) or reset your password.

Don't have an mLab account yet? Sign up now!

To login to your account, use mLabs credentials. After signing in you should be able to see the below dashboard.



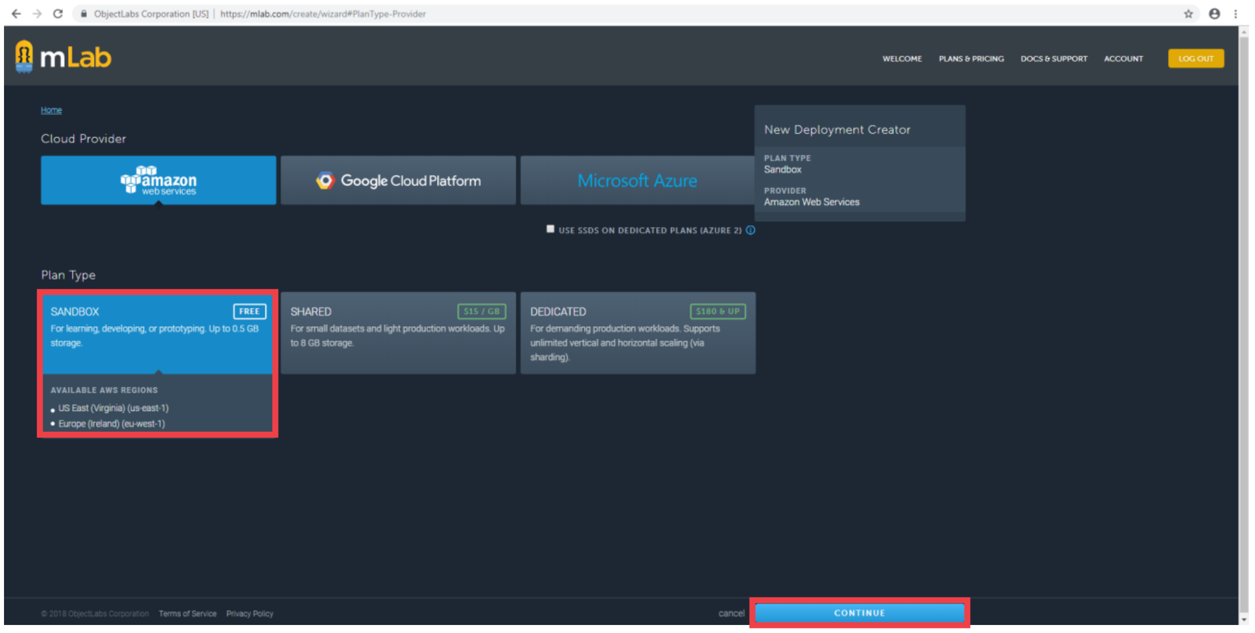
MongoDB Deployments

None exist at this time. Click "Create new" to create a fully managed MongoDB deployment on AWS, Azure, or Google.

Environments

None exist at this time. Click "Create new" to create an mLab Environment (VPC).

Click on **Create new** to create a new database.



New Deployment Creator

PLAN TYPE
Sandbox

PROVIDER
Amazon Web Services

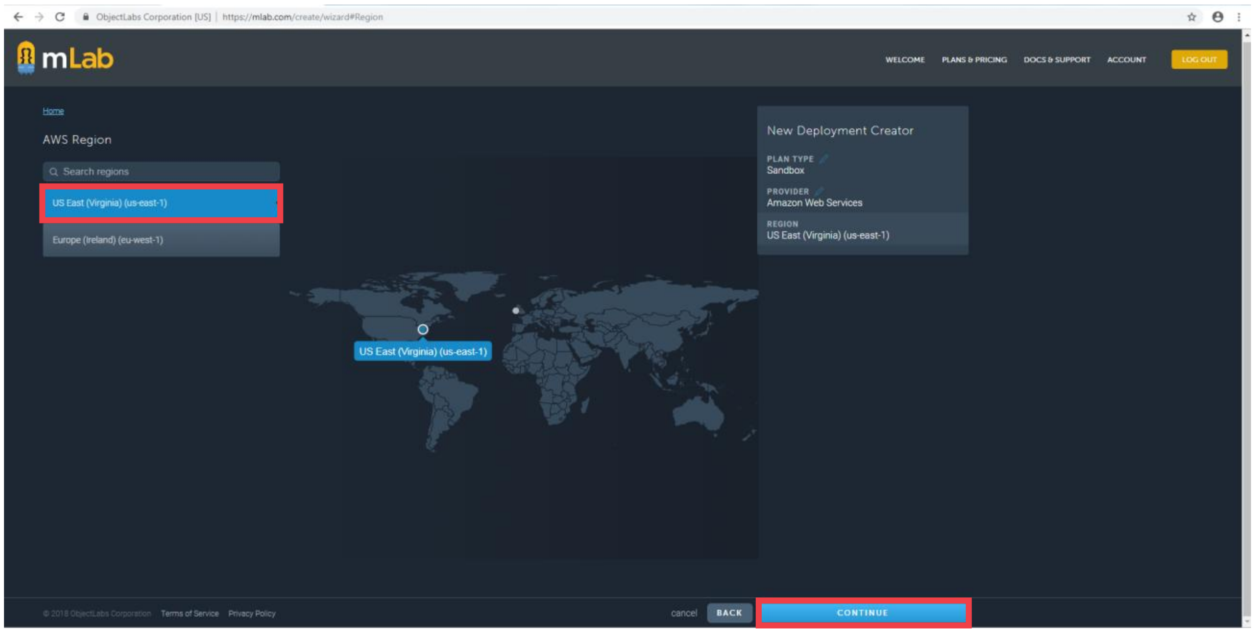
SANDBOX — FREE
For learning, developing, or prototyping. Up to 0.5 GB storage.

AVAILABLE AWS REGIONS
- US East (Virginia) (us-east-1)
- Europe (Ireland) (eu-west-1)

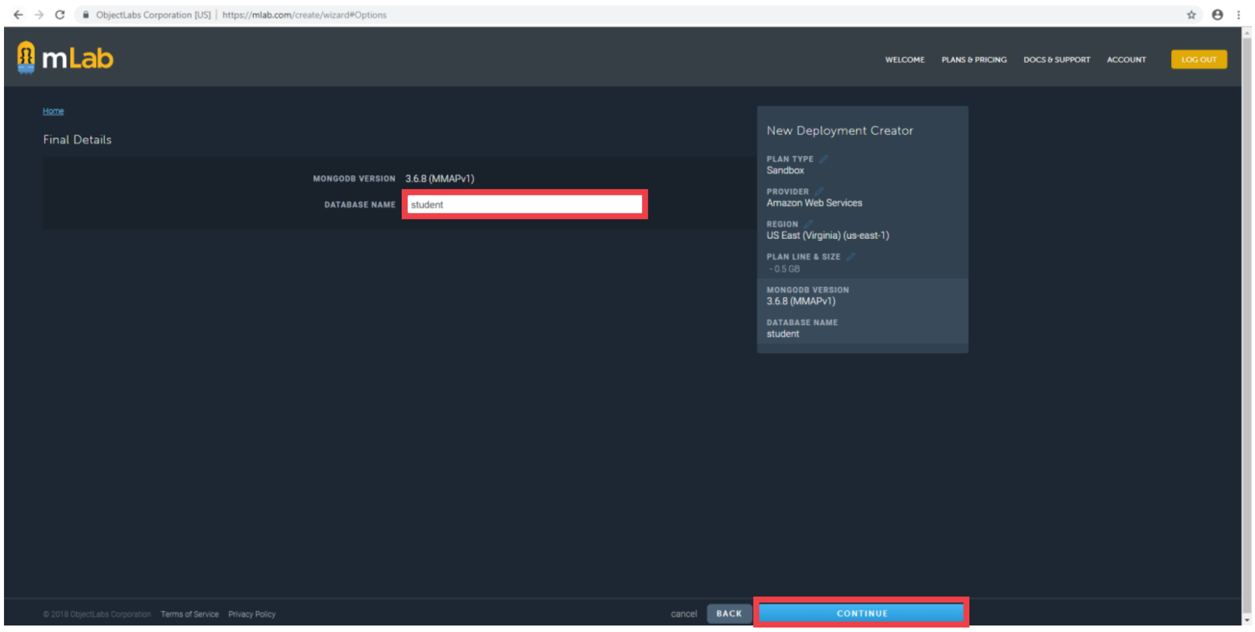
SHARED — $15 / GB
For small datasets and light production workloads. Up to 8 GB storage.

DEDICATED — $180 & UP
For demanding production workloads. Supports unlimited vertical and horizontal scaling (via sharding).

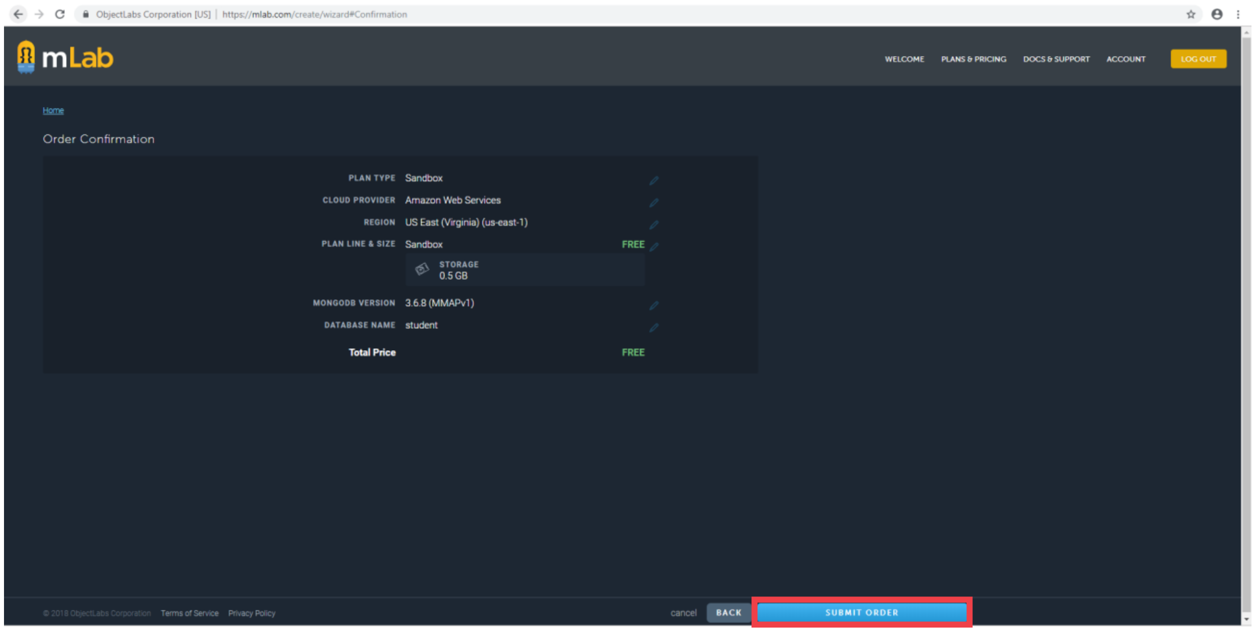
Select **SANDBOX** plan for free trail and click on **Continue** button



Select any one of the AWS regions and click on **Continue**



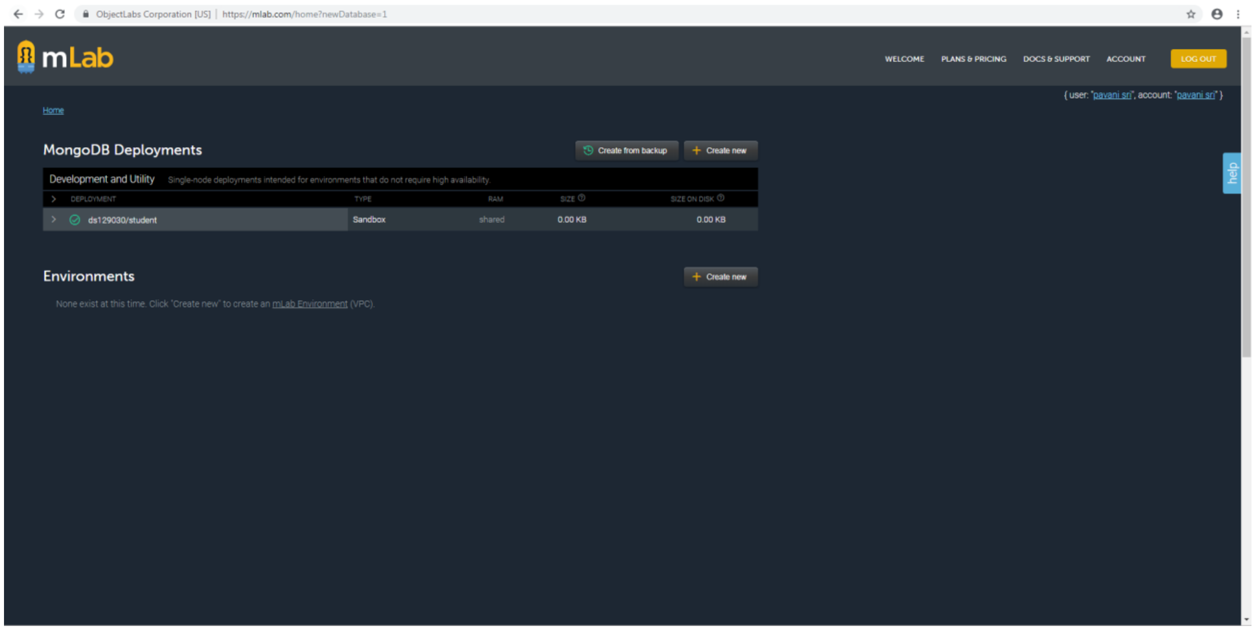
You will get a text box where you need to provide the database name and click on **Continue**. Now, you should be able to see the below dashboard.



Order Confirmation

PLAN TYPE	Sandbox	
CLOUD PROVIDER	Amazon Web Services	
REGION	US East (Virginia) (us-east-1)	
PLAN LINE & SIZE	Sandbox STORAGE 0.5 GB	FREE
MONGODB VERSION	3.6.8 (MMAPv1)	
DATABASE NAME	student	
Total Price		FREE

Click on **Submit Order** button to complete the **MongoDB** deployment.



{ user: "pavani sri", account: "pavani sri" }

MongoDB Deployments

Development and Utility Single-node deployments intended for environments that do not require high availability.

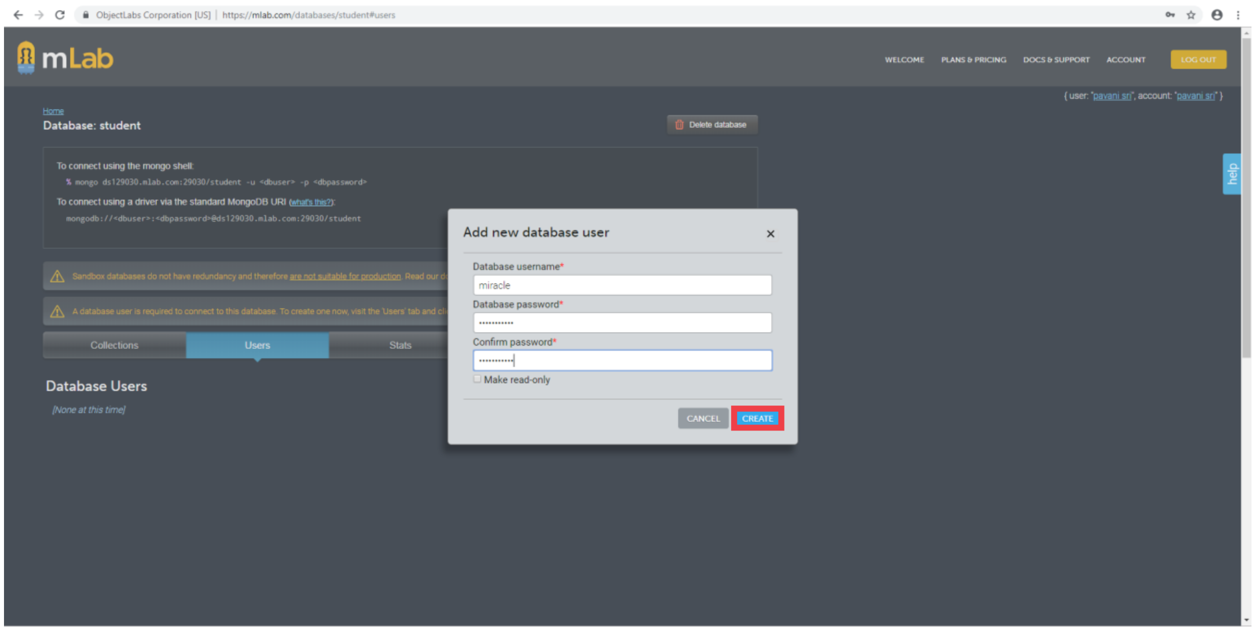
DEPLOYMENT	TYPE	RAM	SIZE	SIZE ON DISK
ds129030/student	Sandbox	shared	0.00 KB	0.00 KB

Environments

None exist at this time. Click "Create new" to create an mLab Environment (VPC).

In order to get the database details, click on the database that you had created.

Select **Users** to add credentials to your database



Database: student

To connect using the mongo shell:

```
% mongo ds129030.mlab.com:29030/student -u <dbuser> -p <dbpassword>
```

To connect using a driver via the standard MongoDB URI (what's this?):

```
mongodb://<dbuser>:<dbpassword>@ds129030.mlab.com:29030/student
```

Sandbox databases do not have redundancy and therefore are not suitable for production. Read our d

A database user is required to connect to this database. To create one now, visit the 'Users' tab and cl

Database Users

[None at this time]

Add new database user

Database username*: miracle

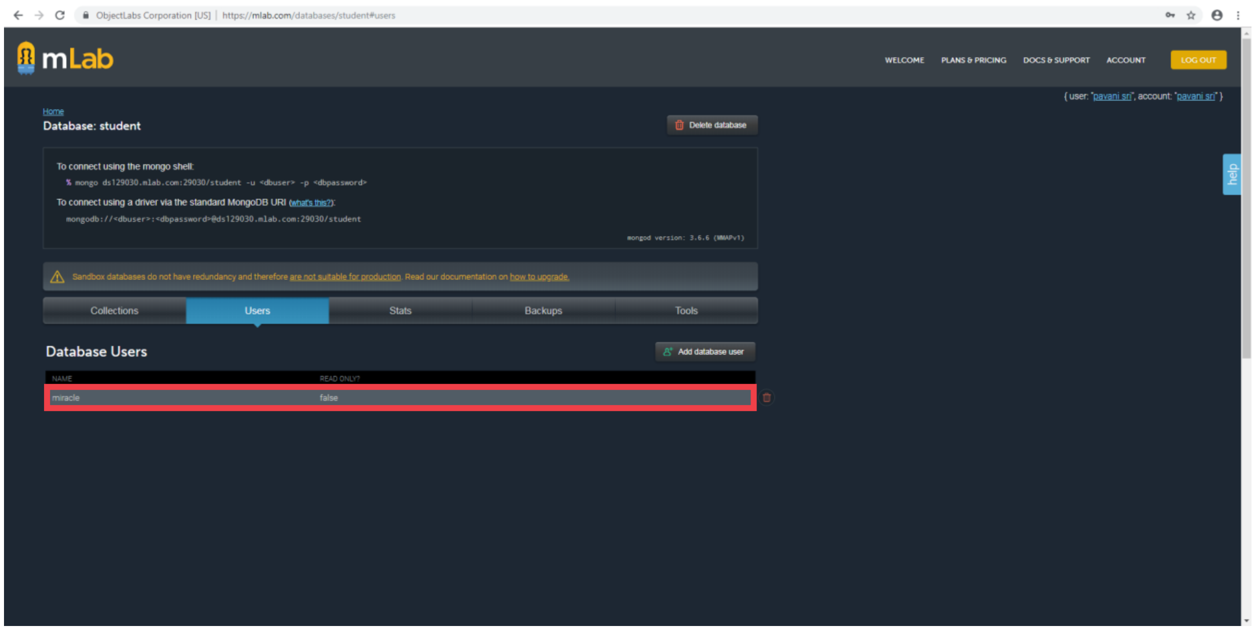
Database password*: ••••••••••

Confirm password*: ••••••••••

Make read-only

CANCEL CREATE

Provide your database **username** and **password** in the popup and click on **Create**



Database: student

To connect using the mongo shell:

```
% mongo ds129030.mlab.com:29030/student -u <dbuser> -p <dbpassword>
```

To connect using a driver via the standard MongoDB URI (what's this?):

```
mongodb://<dbuser>:<dbpassword>@ds129030.mlab.com:29030/student
mongod version: 3.6.6 (MMAPv1)
```

Sandbox databases do not have redundancy and therefore are not suitable for production. Read our documentation on how to upgrade.

Database Users

NAME	READ ONLY?
miracle	false

Now you can check if the user is added to your database users.

Below are the steps to perform **CRUD** operations with **MongoDB** using Node JS.

Step #2 | Initializing MongoDB in Node JS

Below are the steps to establish the connection between MongoDB and Node JS.

1. Install mongoose npm module in Node JS
2. Import the module in the code
3. Provide your mLabs MongoDB credentials(mLabs MongoDB URL) in Node JS

Installing mongoose Module

Open command prompt and execute the following command,

npm install mongoose --save

Import mongoose Module

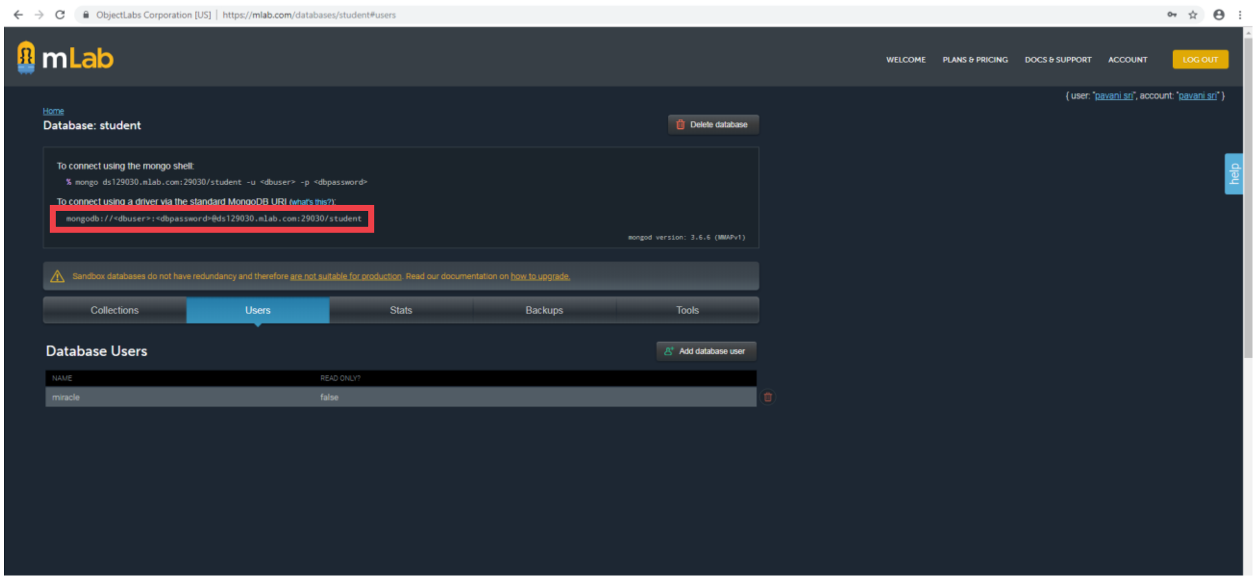
To import this module in Node JS, use the below statement in the script file

var mongoose = require('mongoose');

Database Connection

Below is the URL for connecting to MongoDB database,

mongoose.connect
('MongoDB ://< username>:<password>@ds131902.mlab.com:31902/<database name> ');



Database: student

To connect using the mongo shell:

```
% mongo ds129030.mlab.com:29030/student -u <dbuser> -p <dbpassword>
```

To connect using a driver via the standard MongoDB URI (what's this?):

```
mongodb://<dbuser>:<dbpassword>@ds129030.mlab.com:29030/student
```

```
mongod version: 3.6.6 (MMAPv1)
```

Sandbox databases do not have redundancy and therefore are not suitable for production. Read our documentation on how to upgrade.

Database Users

NAME	READ ONLY?
miracle	false

You will find this URL, username and password in the database page.

Step #3 | Creating MongoDB API's for CRUD Operations

To create and insert data into MongoDB database using Node JS follow the below steps,

Define Schema

Copy and paste the below code of **student.js** in GitHub repo **crud-app** for the schema,

```
var mongoose = require('mongoose');
var studentSchema = mongoose.Schema({
   _id: {
      type: String,
      required: true
   }
 sName: {
      type: String,
      required: true
   },
   sEmail: {
      type: String,
      required: true
   },
   sPhoneNumber: {
      type: String,
      required: true
   },
   sAddress: {
      type: String,
      required: true
   },
   sDepartment: {
      type: String,
      required: true
   },
   create_date: {
        type: Date,
        default: Date.now
    }
});
```

Inserting Data

Copy and paste the code of **Create.html** in GitHub repo **crud-app** folder for front end code and for API creation - below is the code snippet,

```
app.post('/api/insert', function(req, res) {
var student = ({
        _id: req.body.sEmail,
        sName: req.body.sName,
        sEmail: req.body.sEmail,
        sPhoneNumber: req.body.sPhoneNumber,
        sAddress: req.body.sAddress,
        sDepartment: req.body.sDepartment
    });
    Student.addStudent(student, function(err, student) {
        if (student) {
          response = {
              "result": "Data inserted succesfully"
           }
           res.json(response);
        } else {
          error = {
              "error": "Sorry insertion failed"
           }
           res.json(error);
        }
    });
});
```

Below is the definition of the function "**addStudent**" which you have called in above API. You can find this in Student.js file from the GitHub repo.

```
module.exports.addStudent = function(student, callback) {
   Student.create(student, callback);
}
```

Retrieving Data

Copy and paste the code of **Retrieve.html** in GitHub repo **crud-app** folder for front end code and for API creation - below is the code snippet,

```
app.get('/api/retrieve', function(req, res) {
   Student.getDetails(function (err, student) {
      if (student) {
        response = {
            "result": student
         }
         res.json(response);
      } else {
        error = {
            "error": "Sorry retrieve failed"
         }
         res.json(error);
      }
   });
});
```

Below is the definition of the function "**getDetails**" which you have called in the above API. You can find this in Student.js file from the repo,

```
module.exports.getDetails = function(callback, limit) {
   Student.find(callback).limit(limit);
}
```

Updating Data

Copy and paste the code of **Update.html** in GitHub repo **crud-app** folder for front end code and for API creation - below is the code snippet,

```
app.post('/api/update', function(req, res) {
   var id = req.body.sEmail;
   var student = ({
      sName: req.body.sName,
      sPhoneNumber: req.body.sPhoneNumber,
      sAddress: req.body.sAddress,
      sDepartment: req.body.sDepartment
   });
Student.updateStudent(id, student, {}, function(err, student) {
      if (student) {
        response = {
            "result": "Student Details have been updated!"
         }
         res.json(response);
      } else {
        error = {
            "error": "Sorry update failed"
         }
         res.json(error);
      }
   });
});
```

Below is the definition of the function "**updateStudent**" which you have called in above API. You can find this in Student.js file from the repo,

```
module.exports.updateStudent = function(id, student, options, callback) {
    var query = {
        _id: id
    };
    var update = {
        sName: student.sName,
        sPhoneNumber: student.sPhoneNumber,
        sAddress: student.sAddress,
        sDepartment: student.sDepartment
    }
    Student.findOneAndUpdate(query, update, options, callback);
}
```

Deleting Data

Copy and paste the code of **Delete.html** in GitHub repo **crud-app** folder for front end code and for API creation - below is the code snippet

```
app.post('/api/delete', function(req, res) {
  var id = req.body.sEmail;
  Student.removeStudent(id, function(err, student) {
    if (student.result.n != 0) {
      response = {
        "result": "Student Record has been deleted!"
      }
      res.json(response);
    } else {
      error = {
        "error": "Please check entered ID"
      }
      res.json(error);
    }
  });
});
```

Below is the definition of the function "**removeStudent**" which you have called in above API. You can find this in Student.js file from the repo

```
module.exports.removeStudent = function(id, callback) {
  var query = {
    _id: id
  };
  Student.remove(query, callback);
}
```

Step #4 | Rendering HTML Pages

For rendering HTML pages in Node JS, we need to add the following snippet in our server code.

```
app.use(express.static(__dirname + '/public'));
```

Step #5 | Consuming CRUD API's

In the above application, we have created 4 APIs named Create, Update, Retrieve and Delete. Here is the code snippet for consuming API in front-end

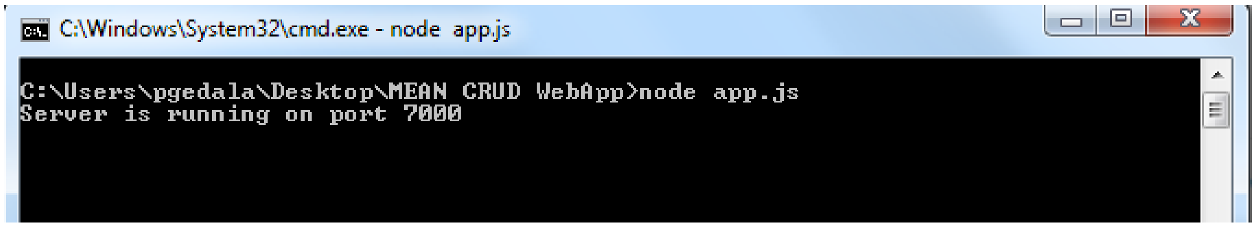
```
$.post("/api/insert", formData, function(response) {
        if (response.error == undefined)
        showSuccess(response.result)
        else {
        showError(response.error)
        }
});
```

You can find the complete code in GitHub repository

Step #6 | Run the Application

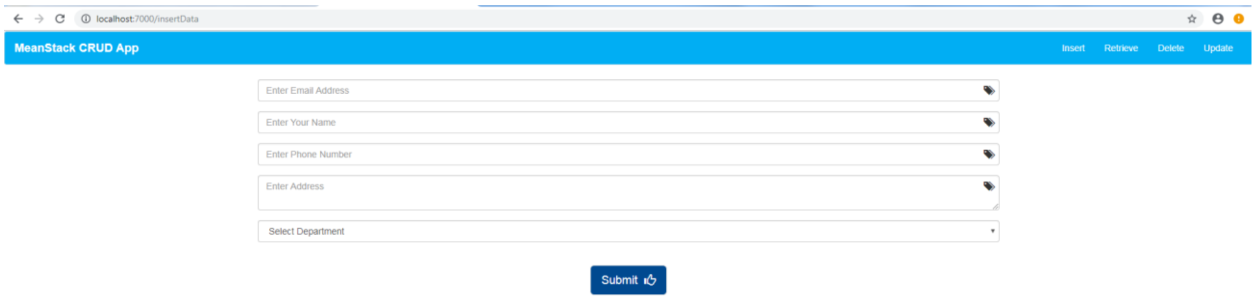
Navigate to the workspace folder where the code exists, and open command prompt

Run **node app.js**

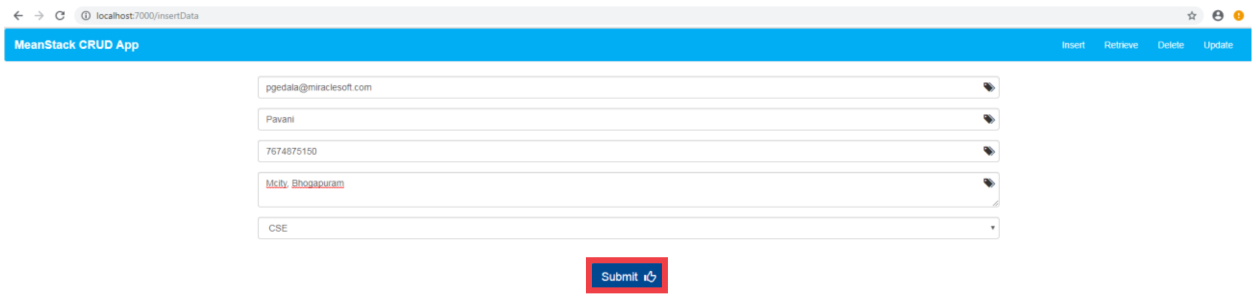


```
C:\Users\pgedala\Desktop\MEAN CRUD WebApp>node app.js
Server is running on port 7000
```

The application is running at http://localhost:7000.



Access the URL: http://localhost:7000/insertData in your browser, you can view the insert student details page.



localhost:7000/insertData

MeanStack CRUD App

pgedala@miraclesoft.com

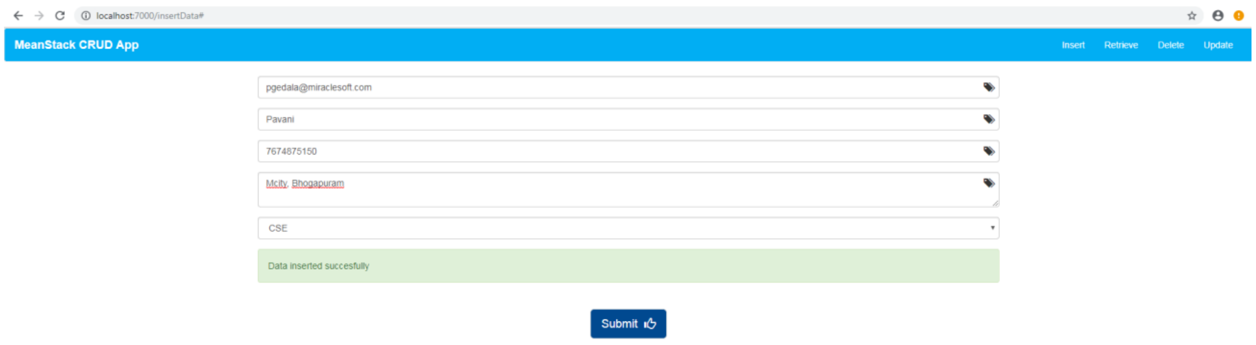
Pavani

7674875150

Mcity, Bhogapuram

CSE

Fill the details as above and click on **Submit** button.



localhost:7000/insertData#

MeanStack CRUD App

pgedala@miraclesoft.com

Pavani

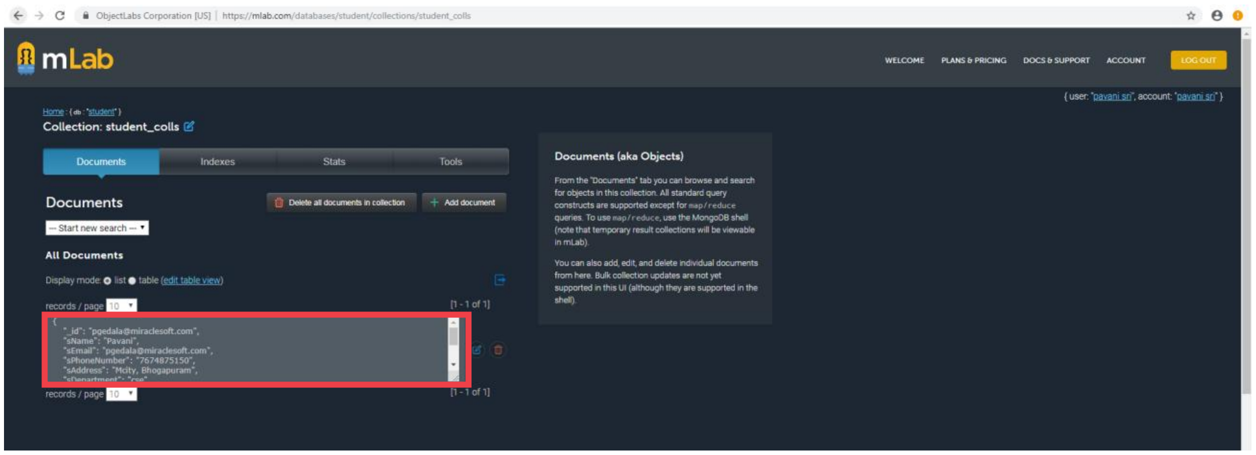
7674875150

Mcity, Bhogapuram

CSE

Data inserted succesfully

After successfully insertion, it shows a success alert as data inserted successfully. Navigate to **Collections** and click on **Documents**.



ObjectLabs Corporation [US] | https://mlab.com/databases/student/collections/student_colls

{ user: "pavani sri", account: "pavani sri" }

Home : { db : "student" }

Collection: student_colls

Documents (aka Objects)

From the "Documents" tab you can browse and search for objects in this collection. All standard query constructs are supported except for map/reduce queries. To use map/reduce, use the MongoDB shell (note that temporary result collections will be viewable in mLab).

You can also add, edit, and delete individual documents from here. Bulk collection updates are not yet supported in this UI (although they are supported in the shell).

Display mode: ● list ○ table (edit table view)

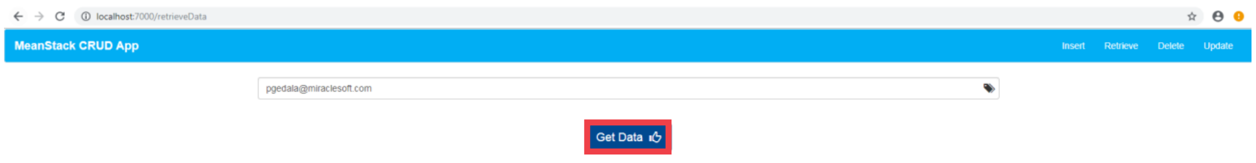
records / page 10 [1 - 1 of 1]

```
{
    "_id": "pgedala@miraclesoft.com",
    "sName": "Pavani",
    "sEmail": "pgedala@miraclesoft.com",
    "sPhoneNumber": "7674875150",
    "sAddress": "Mcity, Bhogapuram",
```

records / page 10 [1 - 1 of 1]

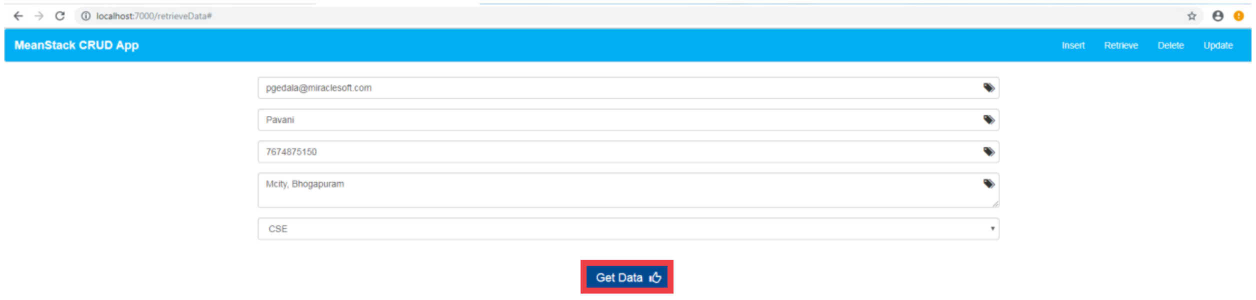
You can now find the newly inserted document in the collection of student database.

Let's check the retrieve functionality.



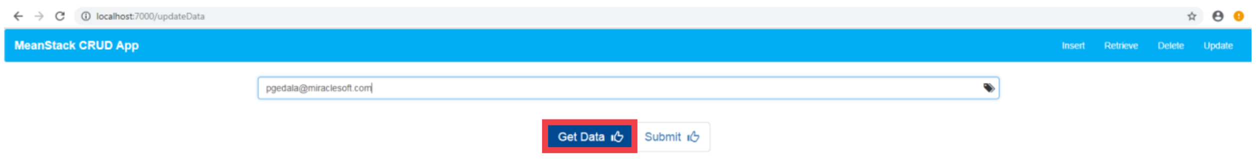
pgedala@miraclesoft.com

For retrieving the data, you will be asked to provide your email address as input and click on **Get Data** button.



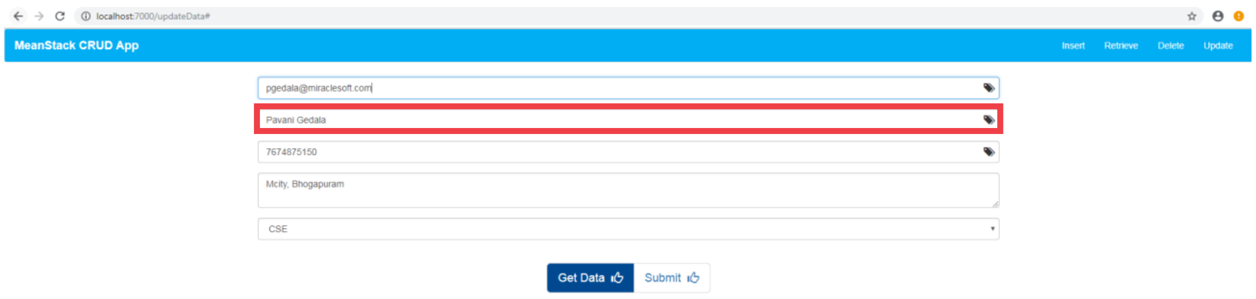
pgedala@miraclesoft.com
Pavani
7674875150
Mcity, Bhogapuram
CSE

After clicking the **Get Data** button the user details will gets displayed as above.



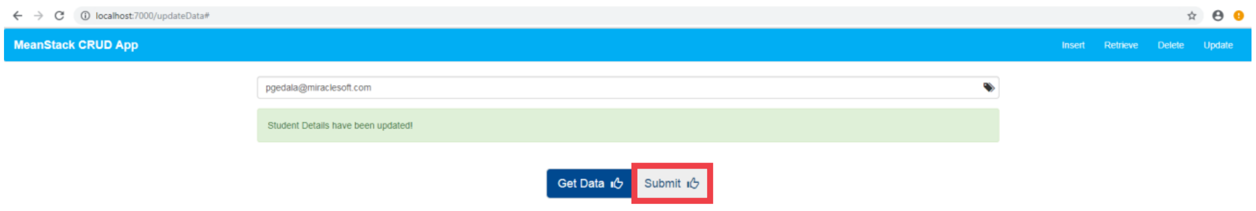
pgedala@miraclesoft.com

For updating the data, you will be asked to provide your email address as input and click on **Get Data** button.



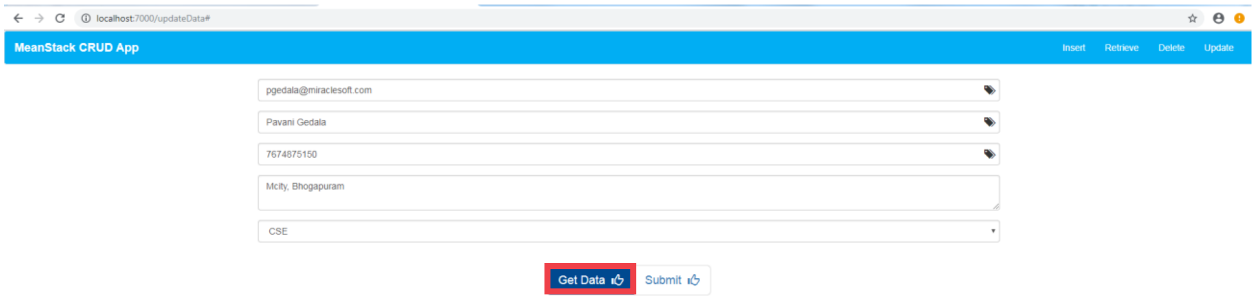
pgedala@miraclesoft.com
Pavani Gedala
7674875150
Mcity, Bhogapuram
CSE

Change any one of the above fields you want to update.



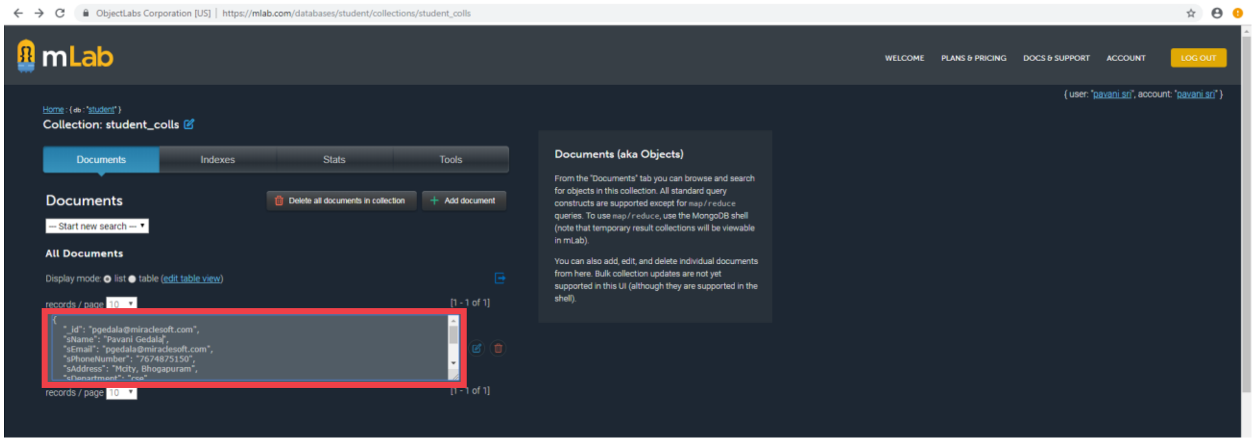
pgedala@miraclesoft.com
Student Details have been updated!

Click on the Submit button. You will get a Success alert as details are updated successfully.



pgedala@miraclesoft.com
Pavani Gedala
7674875150
Mcity, Bhogapuram
CSE

Check if the data is updated by clicking on **Get Data** button.


ObjectLabs Corporation [US] | https://mlab.com/databases/student/collections/student_colls
mLab
WELCOME
PLANS & PRICING
DOCS & SUPPORT
ACCOUNT
LOG OUT
Home : { db : "student" }
Documents
Indexes
Stats
Tools
Delete all documents in collection
Add document
--- Start new search ---
Display mode: list table (edit table view)
records / page 10
[1 - 1 of 1]

{ user: "pavani sri", account: "pavani sri" }

Collection: student_colls

Documents

All Documents

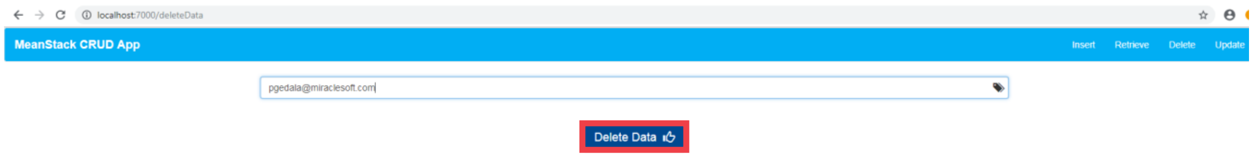
```
{
    "_id": "pgedala@miraclesoft.com",
    "sName": "Pavani Gedala",
    "sEmail": "pgedala@miraclesoft.com",
    "sPhoneNumber": "7674875150",
    "sAddress": "Mcity, Bhogapuram",
```

Documents (aka Objects)

From the "Documents" tab you can browse and search for objects in this collection. All standard query constructs are supported except for `map/reduce` queries. To use `map/reduce`, use the MongoDB shell (note that temporary result collections will be viewable in mLab).

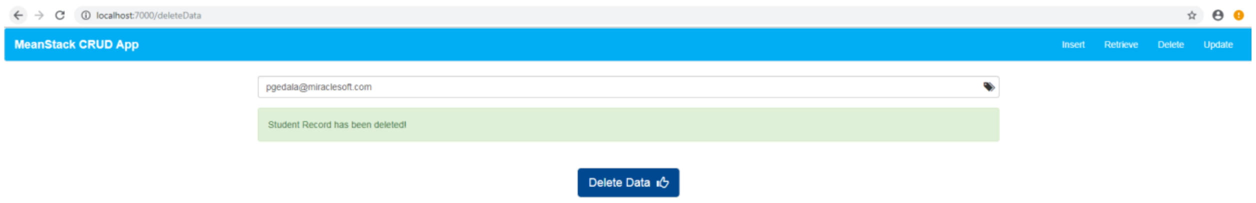
You can also add, edit, and delete individual documents from here. Bulk collection updates are not yet supported in this UI (although they are supported in the shell).

Go to the mLabs dashboard and refresh the page to check the updated details in the collection of student database.



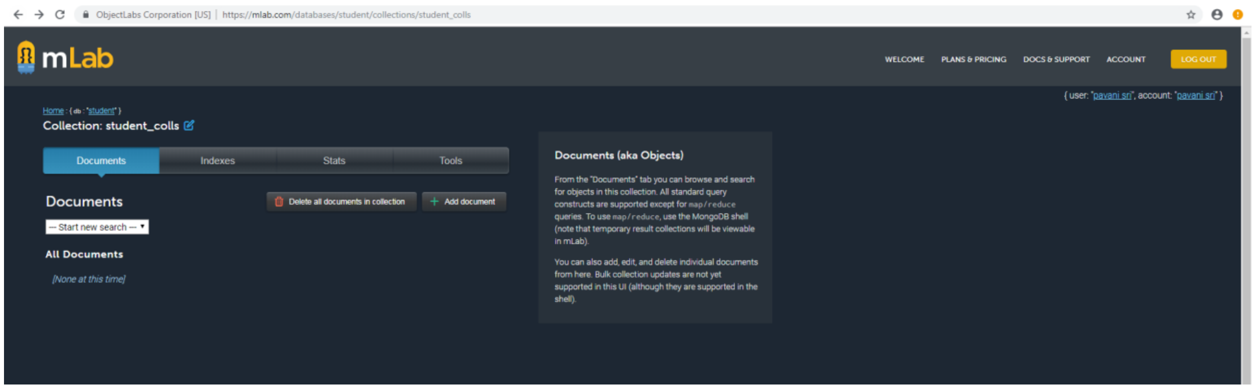
pgedala@miraclesoft.com

For deleting the data, you will be asked to provide your email address as input and click on **Delete Data** button.



Student Record has been deleted!

After clicking on the **Delete Data** button, you will get a success alert as student record had been deleted successfully.


ObjectLabs Corporation [US] | https://mlab.com/databases/student/collections/student_colls
mLab
WELCOME
PLANS & PRICING
DOCS & SUPPORT
ACCOUNT
LOG OUT
{ user: "pavani sri", account: "pavani sri" }
Home : { db: "student" }
Collection: student_colls
Documents
Indexes
Stats
Tools
Documents
Delete all documents in collection
Add document
-- Start new search --
All Documents
[None at this time]

Documents (aka Objects)

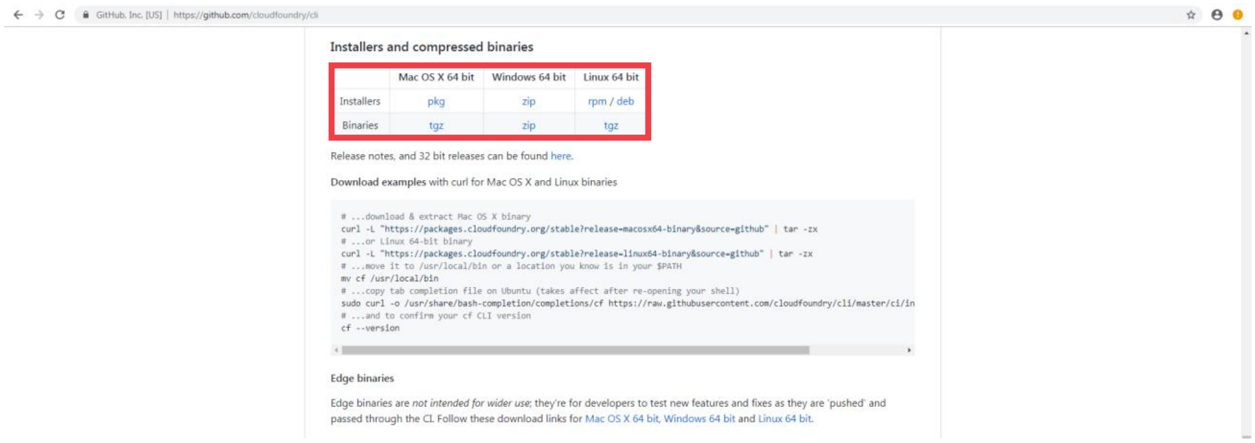
From the "Documents" tab you can browse and search for objects in this collection. All standard query constructs are supported except for `map/reduce` queries. To use `map/reduce`, use the MongoDB shell (note that temporary result collections will be viewable in mLab).

You can also add, edit, and delete individual documents from here. Bulk collection updates are not yet supported in this UI (although they are supported in the shell).

Go to the mLabs dashboard and refresh the page to check the deleted records. You can now see empty collection of student database.

Step #7 | Installing the CF CLI

For this lab, we will be using the **Cloud Foundry (CF) CLI** option. Open the below link for installing **CF-CLI**, https://github.com/cloudfoundry/cli



Installers and compressed binaries

	Mac OS X 64 bit	Windows 64 bit	Linux 64 bit
Installers	pkg	zip	rpm / deb
Binaries	tgz	zip	tgz

Release notes, and 32 bit releases can be found here.

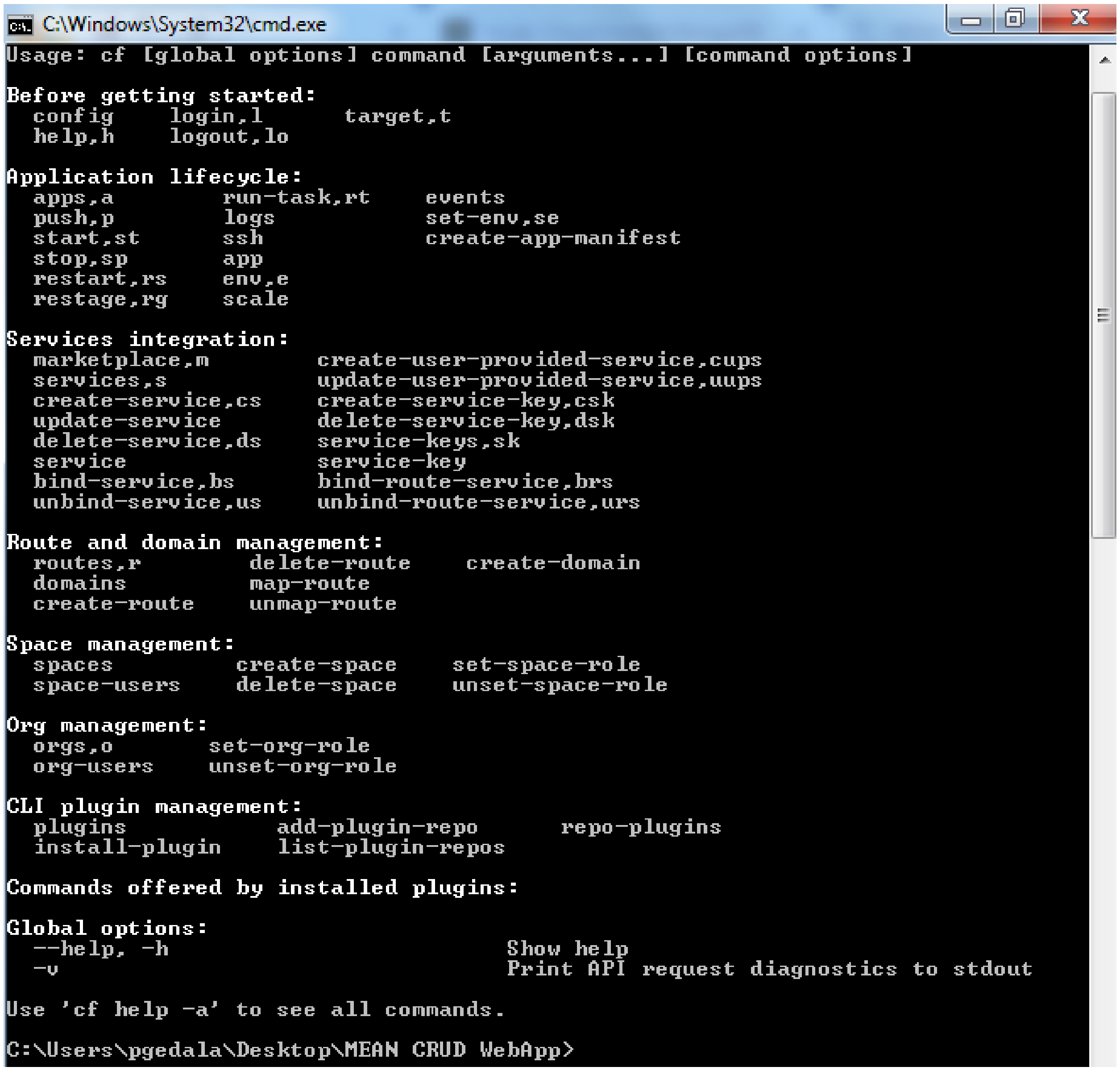
Download examples with curl for Mac OS X and Linux binaries

```
# ...download & extract Mac OS X binary
curl -L "https://packages.cloudfoundry.org/stable?release=macosx64-binary&source=github" | tar -zx
# ...or Linux 64-bit binary
curl -L "https://packages.cloudfoundry.org/stable?release=linux64-binary&source=github" | tar -zx
# ...move it to /usr/local/bin or a location you know is in your $PATH
mv cf /usr/local/bin
# ...copy tab completion file on Ubuntu (takes affect after re-opening your shell)
sudo curl -o /usr/share/bash-completion/completions/cf https://raw.githubusercontent.com/cloudfoundry/cli/master/ci/in
# ...and to confirm your cf CLI version
cf --version
```

Edge binaries

Edge binaries are *not intended for wider use*; they're for developers to test new features and fixes as they are 'pushed' and passed through the CI. Follow these download links for Mac OS X 64 bit, Windows 64 bit and Linux 64 bit.

Extract the downloaded zip file and run the **.exe** file for installation.


C:\Windows\System32\cmd.exe

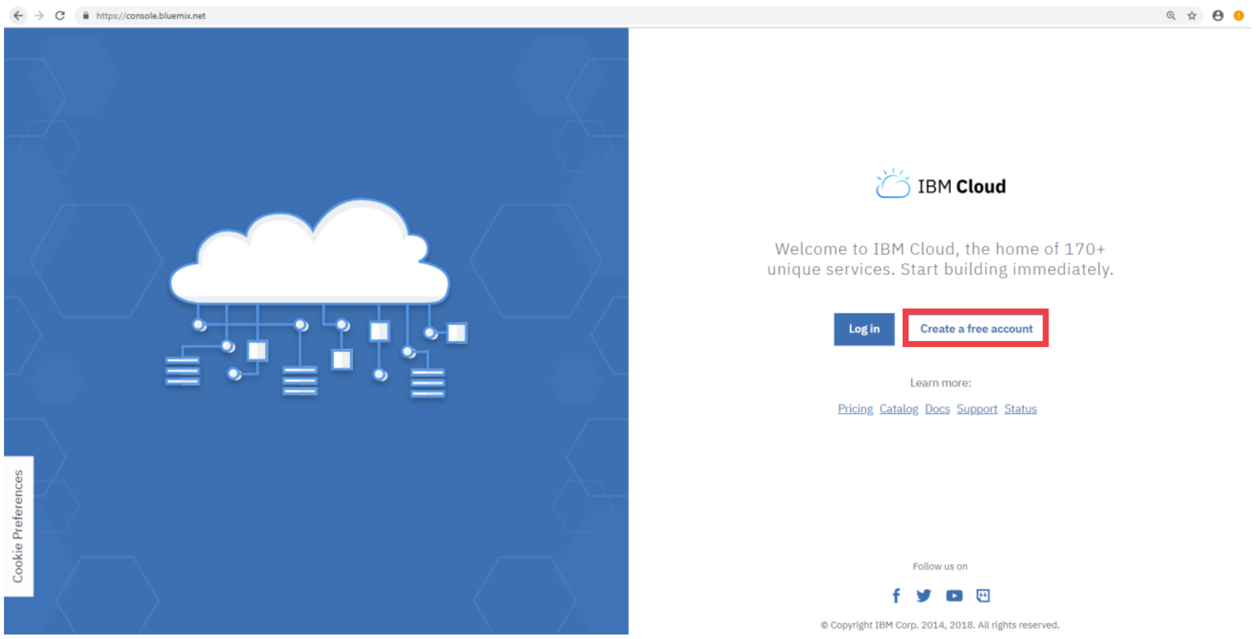
```
Usage: cf [global options] command [arguments...] [command options]

Before getting started:
  config      login,l       target,t
  help,h      logout,lo

Application lifecycle:
  apps,a          run-task,rt     events
  push,p          logs            set-env,se
  start,st        ssh             create-app-manifest
  stop,sp         app
  restart,rs      env,e
  restage,rg      scale

Services integration:
  marketplace,m          create-user-provided-service,cups
  services,s             update-user-provided-service,uups
  create-service,cs      create-service-key,csk
  update-service         delete-service-key,dsk
  delete-service,ds      service-keys,sk
  service                service-key
  bind-service,bs        bind-route-service,brs
  unbind-service,us      unbind-route-service,urs

Route and domain management:
  routes,r          delete-route     create-domain
  domains           map-route
  create-route      unmap-route

Space management:
  spaces          create-space     set-space-role
  space-users     delete-space     unset-space-role

Org management:
  orgs,o        set-org-role
  org-users     unset-org-role

CLI plugin management:
  plugins             add-plugin-repo       repo-plugins
  install-plugin      list-plugin-repos

Commands offered by installed plugins:

Global options:
  --help, -h                         Show help
  -v                                 Print API request diagnostics to stdout

Use 'cf help -a' to see all commands.

C:\Users\pgedala\Desktop\MEAN CRUD WebApp>
```

To check whether CF is installed properly or not, open command prompt and execute **CF** command. Then, it will show you a set of commands, which indicates that it is successfully installed on your machine.

Step #8 | Creating an IBM Cloud account

The next step will be to make sure that we have access to the IBM cloud console with either the free trial option (or) the paid subscription option.

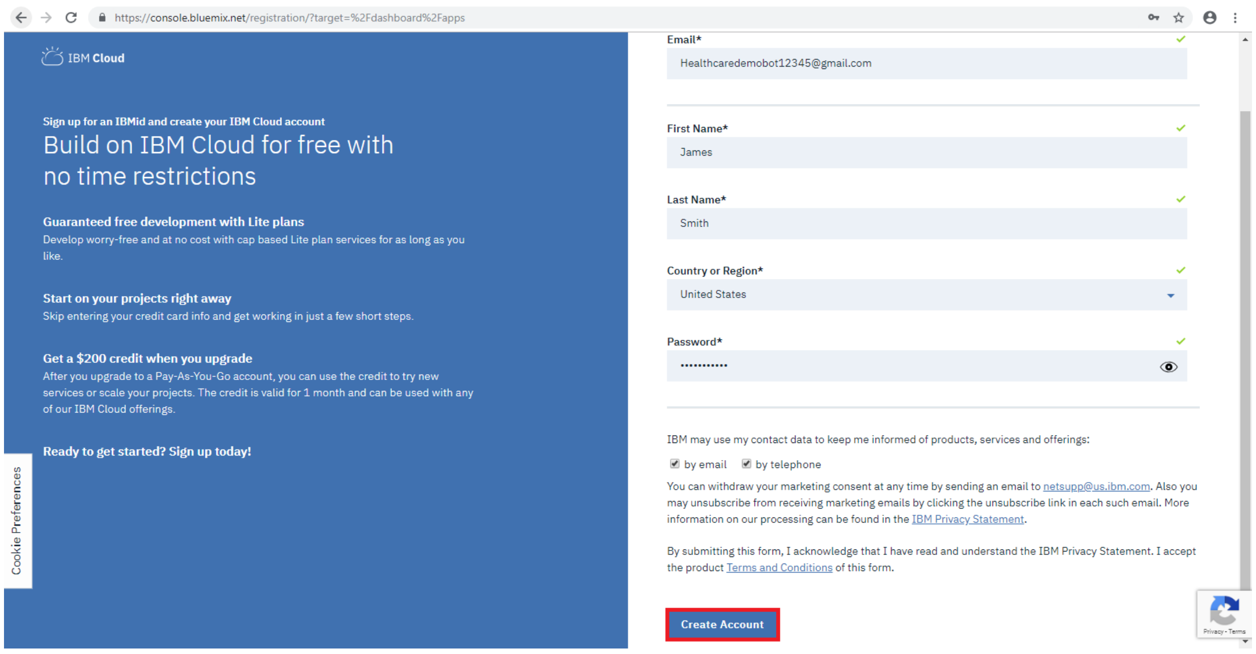
Login to IBM Cloud at http://bluemix.net (or) you can register at https://console.ng.bluemix.net/registration/



Welcome to IBM Cloud, the home of 170+ unique services. Start building immediately.

© Copyright IBM Corp. 2014, 2018. All rights reserved.

Click on **Create a free account**, and fill the fields as required.


https://console.bluemix.net/registration/?target=%2Fdashboard%2Fapps
IBM Cloud
Cookie Preferences
Create Account

Sign up for an IBMid and create your IBM Cloud account

Build on IBM Cloud for free with no time restrictions

Guaranteed free development with Lite plans

Develop worry-free and at no cost with cap based Lite plan services for as long as you like.

Start on your projects right away

Skip entering your credit card info and get working in just a few short steps.

Get a $200 credit when you upgrade

After you upgrade to a Pay-As-You-Go account, you can use the credit to try new services or scale your projects. The credit is valid for 1 month and can be used with any of our IBM Cloud offerings.

Ready to get started? Sign up today!

Email*: Healthcaredemobot12345@gmail.com

First Name*: James

Last Name*: Smith

Country or Region*: United States

Password*: ••••••••••••

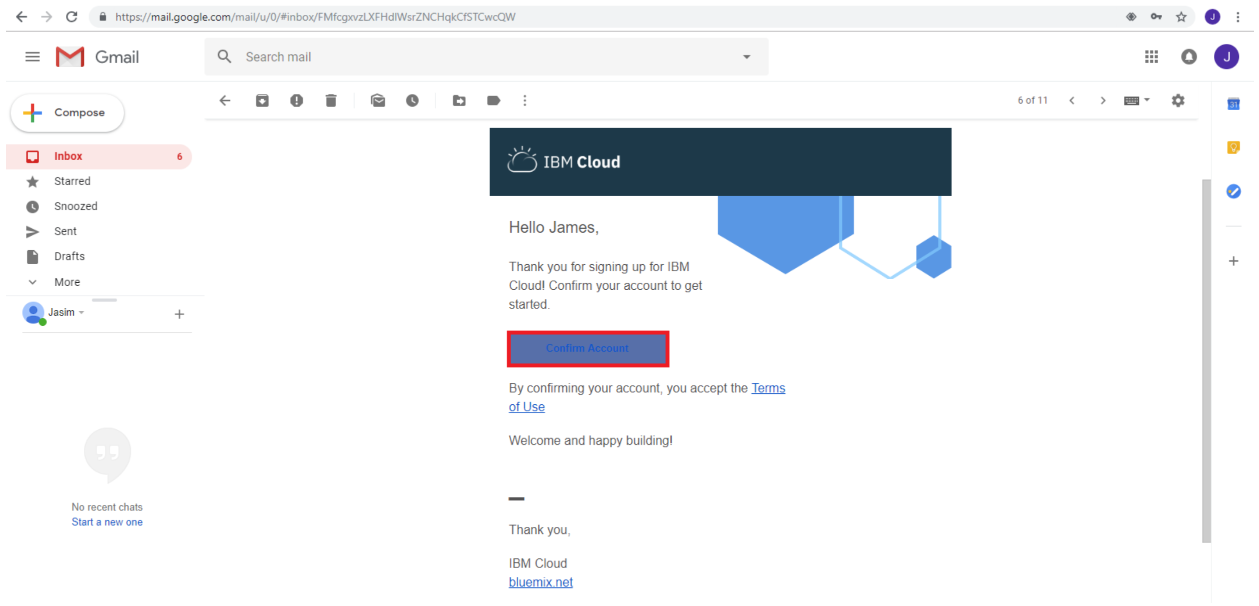
IBM may use my contact data to keep me informed of products, services and offerings:

[✓] by email [✓] by telephone

You can withdraw your marketing consent at any time by sending an email to netsupp@us.ibm.com. Also you may unsubscribe from receiving marketing emails by clicking the unsubscribe link in each such email. More information on our processing can be found in the IBM Privacy Statement.

By submitting this form, I acknowledge that I have read and understand the IBM Privacy Statement. I accept the product Terms and Conditions of this form.

After creating account, confirmation mail will be sent to the registered mail id.


Hello James,

Thank you for signing up for IBM Cloud! Confirm your account to get started.

Confirm Account

By confirming your account, you accept the Terms of Use

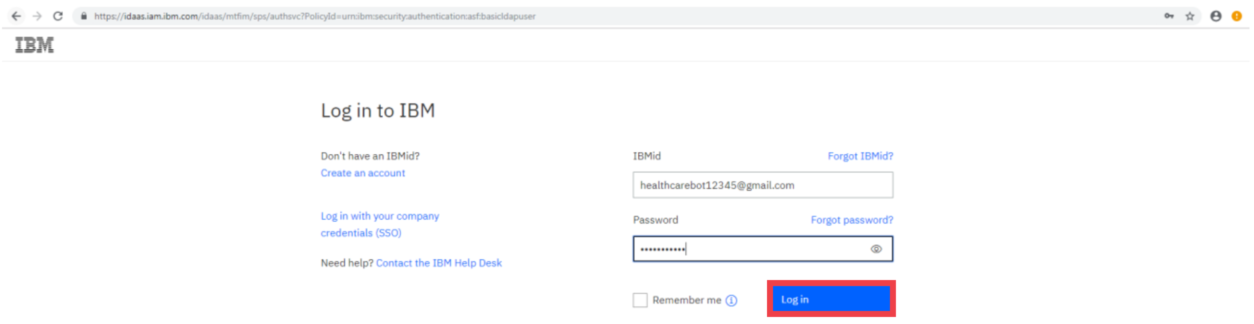
Welcome and happy building!

—

Thank you,

IBM Cloud
bluemix.net

Click on **Confirm Account** and then Login to your IBM Cloud account.


Log in to IBM

Don't have an IBMid?
Create an account

Log in with your company credentials (SSO)

Need help? Contact the IBM Help Desk

IBMid: healthcarebot12345@gmail.com

Password: •••••••••••

Remember me

Log in

Step #9 | Deploying the application onto IBM Cloud

The next step will be to take your application and deploy it back to IBM Cloud so that you can share it with your friends.

Add **manifest** file, for pushing the application to IBM Cloud

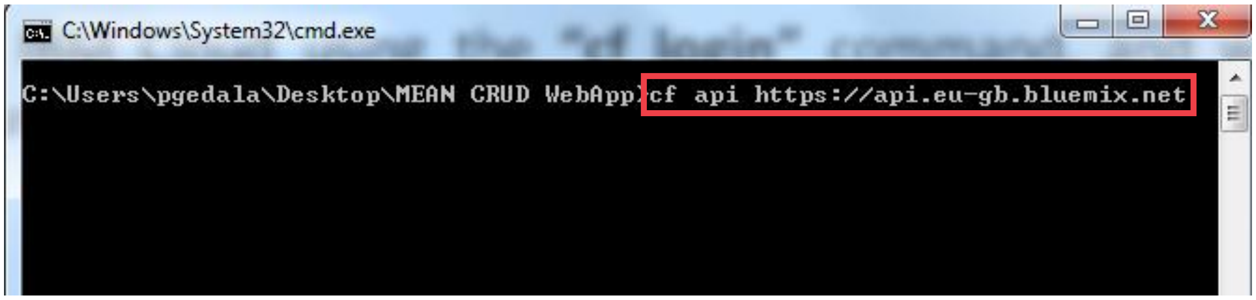
applications-
path: .
memory:256M
instances : 1
domain: eu-gb.mybluemix.net

name: crud-node-demoapp
host: crud-node-demoapp
disk-quota: 1024M

Name the file as **manifest.yml** and save in the same folder.

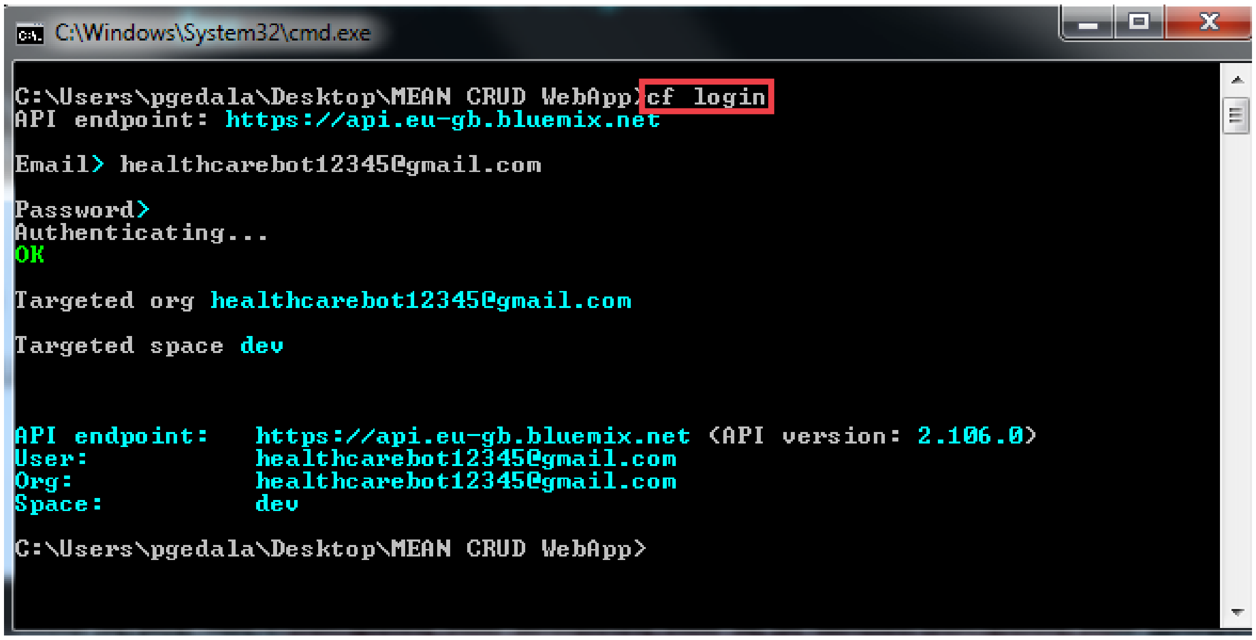
Open the **Command Prompt** and navigate to the location where you have your workspace. Then, connect to IBM Cloud using one of the following commands (Depends on which region you selected in your profile).

For Sydney : cf api https://api.au-syd.bluemix.net
For US South : cf api https://api.ng.bluemix.net
For United Kingdom : cf api https://api.eu-gb.bluemix.net



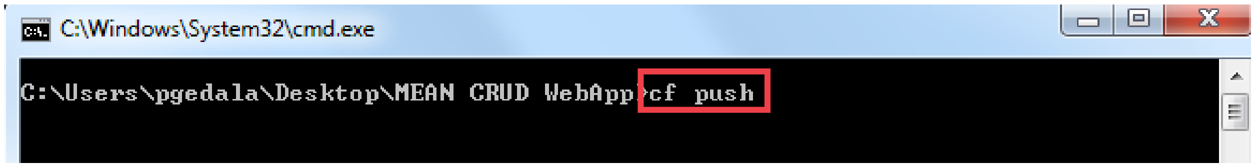
```
C:\Users\pgedala\Desktop\MEAN CRUD WebApp>cf api https://api.eu-gb.bluemix.net
```

Provide your Region API endpoint.



```
C:\Users\pgedala\Desktop\MEAN CRUD WebApp>cf login
API endpoint: https://api.eu-gb.bluemix.net

Email> healthcarebot12345@gmail.com

Password>
Authenticating...
OK

Targeted org healthcarebot12345@gmail.com

Targeted space dev



API endpoint:   https://api.eu-gb.bluemix.net (API version: 2.106.0)
User:           healthcarebot12345@gmail.com
Org:            healthcarebot12345@gmail.com
Space:          dev

C:\Users\pgedala\Desktop\MEAN CRUD WebApp>
```

Login to IBM Cloud using the **cf login** command, and when prompted enter your user ID and password to login.

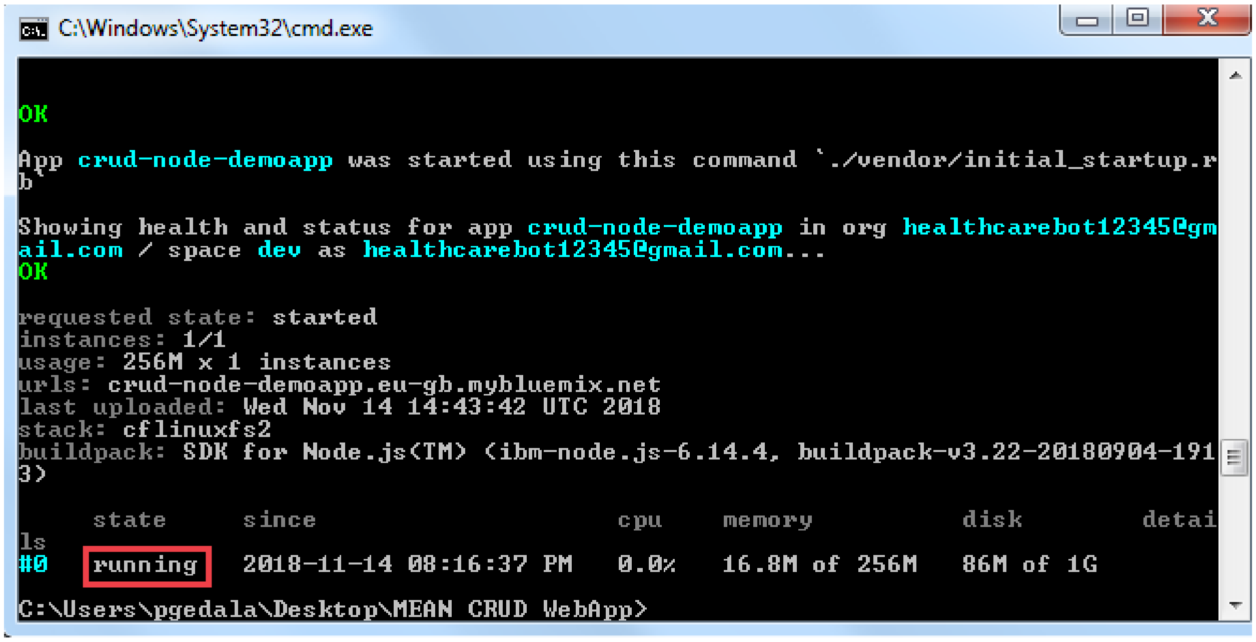


```
C:\Users\pgedala\Desktop\MEAN CRUD WebApp>cf push
```

Make sure that you are within your application's directory and use the **cf push** command to push your application to your IBM Cloud Organization.

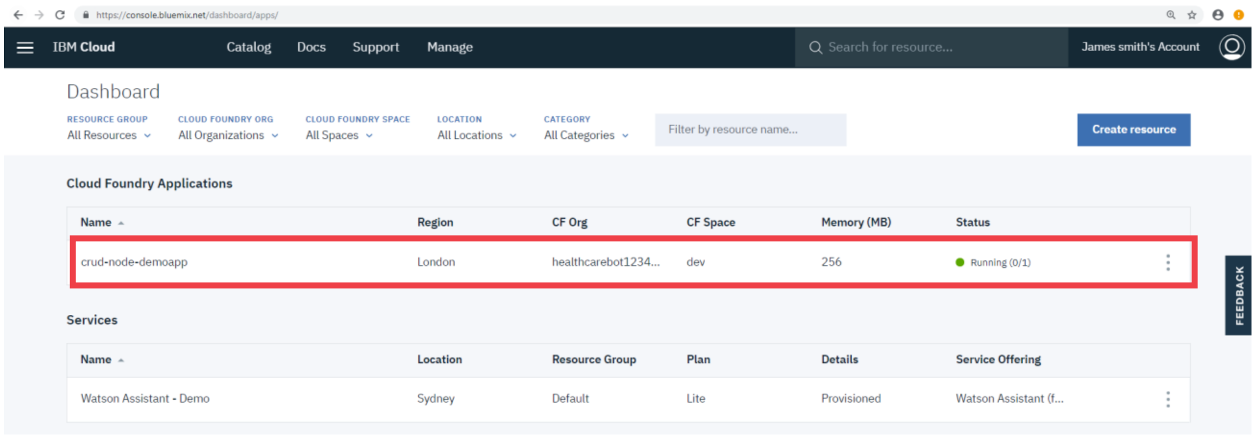
Note - This process might take around 3 to 5 minutes for completion.

Once your application is pushed, your command prompt should look as shown below,



```
OK

App crud-node-demoapp was started using this command `./vendor/initial_startup.r
b`

Showing health and status for app crud-node-demoapp in org healthcarebot12345@gm
ail.com / space dev as healthcarebot12345@gmail.com...
OK

requested state: started
instances: 1/1
usage: 256M x 1 instances
urls: crud-node-demoapp.eu-gb.mybluemix.net
last uploaded: Wed Nov 14 14:43:42 UTC 2018
stack: cflinuxfs2
buildpack: SDK for Node.js(TM) (ibm-node.js-6.14.4, buildpack-v3.22-20180904-191
3)

     state      since                    cpu    memory          disk          detai
ls
#0   running    2018-11-14 08:16:37 PM   0.0%   16.8M of 256M   86M of 1G

C:\Users\pgedala\Desktop\MEAN CRUD WebApp>
```

Now, you can go back to your IBM Cloud account in the browser and access your applications URL through **Dashboard**


https://console.bluemix.net/dashboard/apps/
IBM Cloud
Catalog
Docs
Support
Manage
Search for resource...
James smith's Account
Dashboard
RESOURCE GROUP All Resources
CLOUD FOUNDRY ORG All Organizations
CLOUD FOUNDRY SPACE All Spaces
LOCATION All Locations
CATEGORY All Categories
Filter by resource name...
Create resource
FEEDBACK

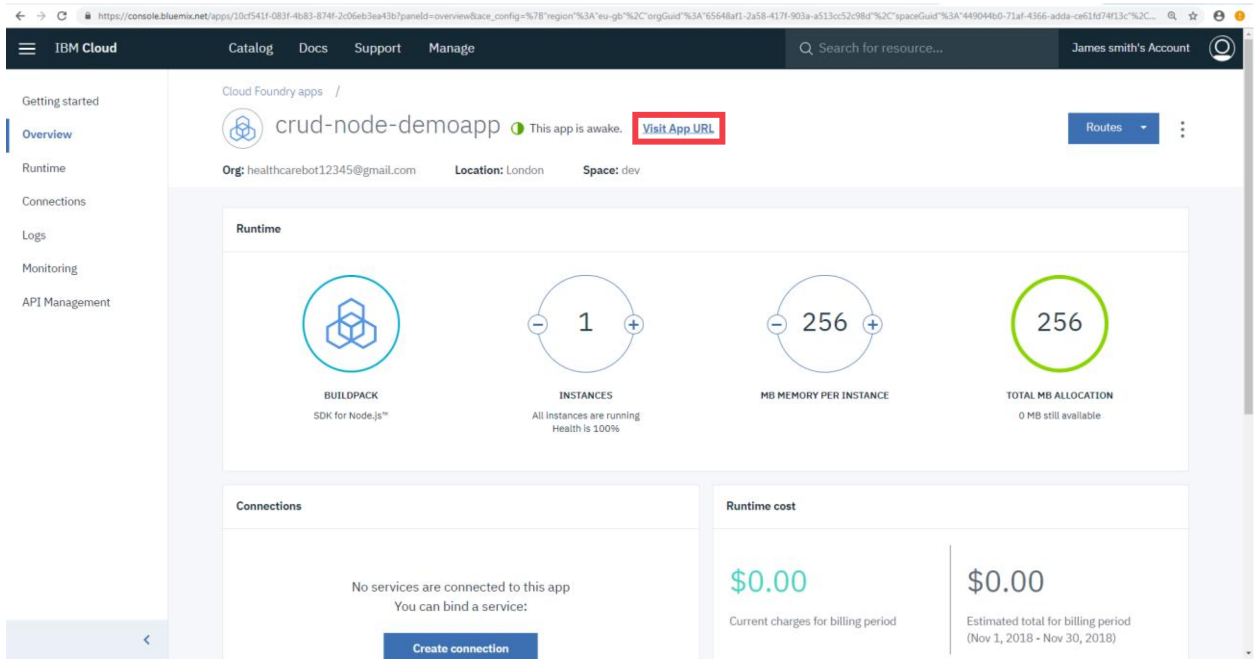
Cloud Foundry Applications

Name	Region	CF Org	CF Space	Memory (MB)	Status
crud-node-demoapp	London	healthcarebot1234...	dev	256	● Running (0/1)

Services

Name	Location	Resource Group	Plan	Details	Service Offering
Watson Assistant - Demo	Sydney	Default	Lite	Provisioned	Watson Assistant (f...

In the dashboard you can see all the cloud foundry applications. Find your application that you have deployed and click on it.



crud-node-demoapp ◑ This app is awake. Visit App URL

Org: healthcarebot12345@gmail.com Location: London Space: dev

Runtime

BUILDPACK: SDK for Node.js™
1 INSTANCES: All instances are running, Health is 100%
256 MB MEMORY PER INSTANCE
256 TOTAL MB ALLOCATION: 0 MB still available

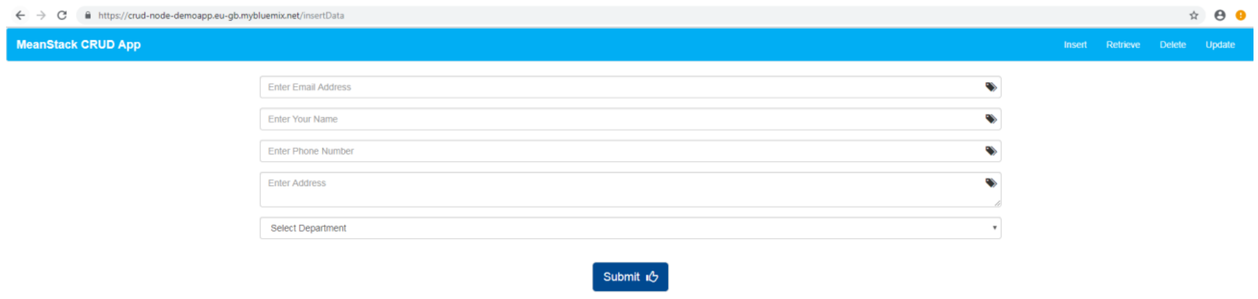
Connections

No services are connected to this app
You can bind a service:

Runtime cost

$0.00 Current charges for billing period
$0.00 Estimated total for billing period (Nov 1, 2018 - Nov 30, 2018)

Click on **Visit App URL** to access your application. Now you can see the below dashboard,



It means, the application is deployed and running successfully.