# NATIONAL INSTITUTE OF TECHONOLOGY DELHI-40

# NETWORK PROGRAMMING

*AWS (cloud, any) assignment*

*Write a client-server program using Unix Domain Datagram sockets*

**SUBMITTED BY:**

**NAME: OMPRAKASH GUPTA**

**ROLL: 171210041**

**CSE 3RD YEAR**

**SUBMITTED TO:**

**Dr Ravi Arya**

**NIT, DELHI**

## **Summary:**

Cloud computing is the delivery of on-demand computing services -- from applications to storage and processing power -- typically over the internet and on a pay-as-you-go basis.

Cloud computing underpins a vast number of services. That includes consumer services like Gmail or the cloud back-up of the photos on your smartphone, though to the services which allow large enterprises to host all their data and run all of their applications in the cloud.

AWS, AZURE etc are some cloud service provider whereas Netflix etc are greatest setup over cloud computing.

As per given task, I have created a google cloud (not on AWS) and created a virtual machine instances. Apart from that I have made a simple web based blog writing application using basic of HTML, CSS, JAVASCRIPT, NODEJS and MONGODB as its database.

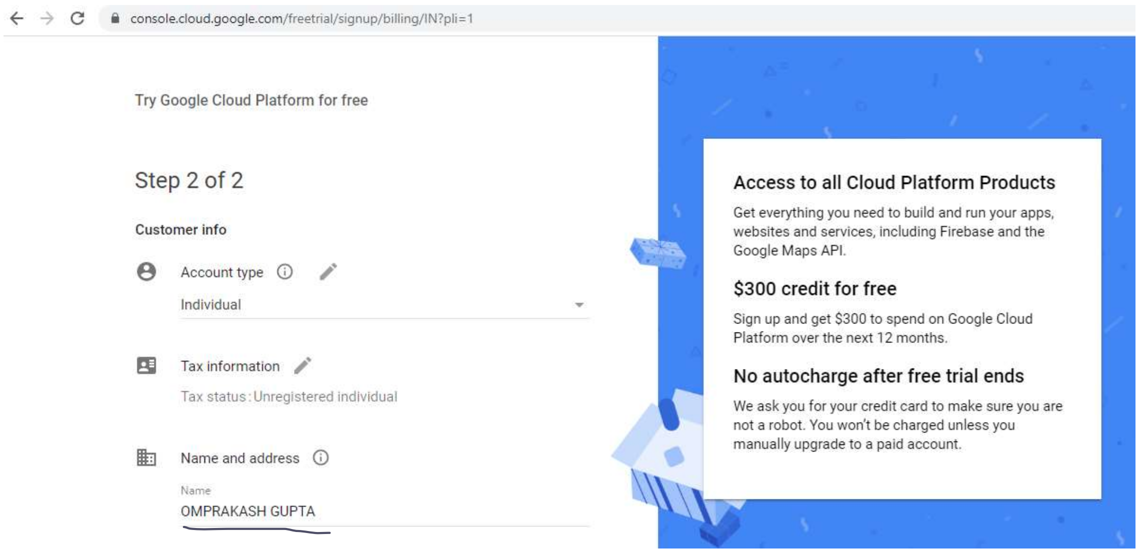
So, given task is divided into three:

1. Create virtual machine instances on google cloud and run server client program over there.
2. Deploy the database of blog writing application on MongoDB cloud server aka ATLAS.
3. Deploy web application on HEROKU cloud web service.

## **Create virtual machine instances on google cloud and run server client program over there.**

### **Step:1**

*Create google account.*



console.cloud.google.com/freetrial/signup/billing/IN?pli=1

Try Google Cloud Platform for free

Step 2 of 2

Customer info

Account type
Individual

Tax information
Tax status : Unregistered individual

Name and address
Name
OMPRAKASH GUPTA

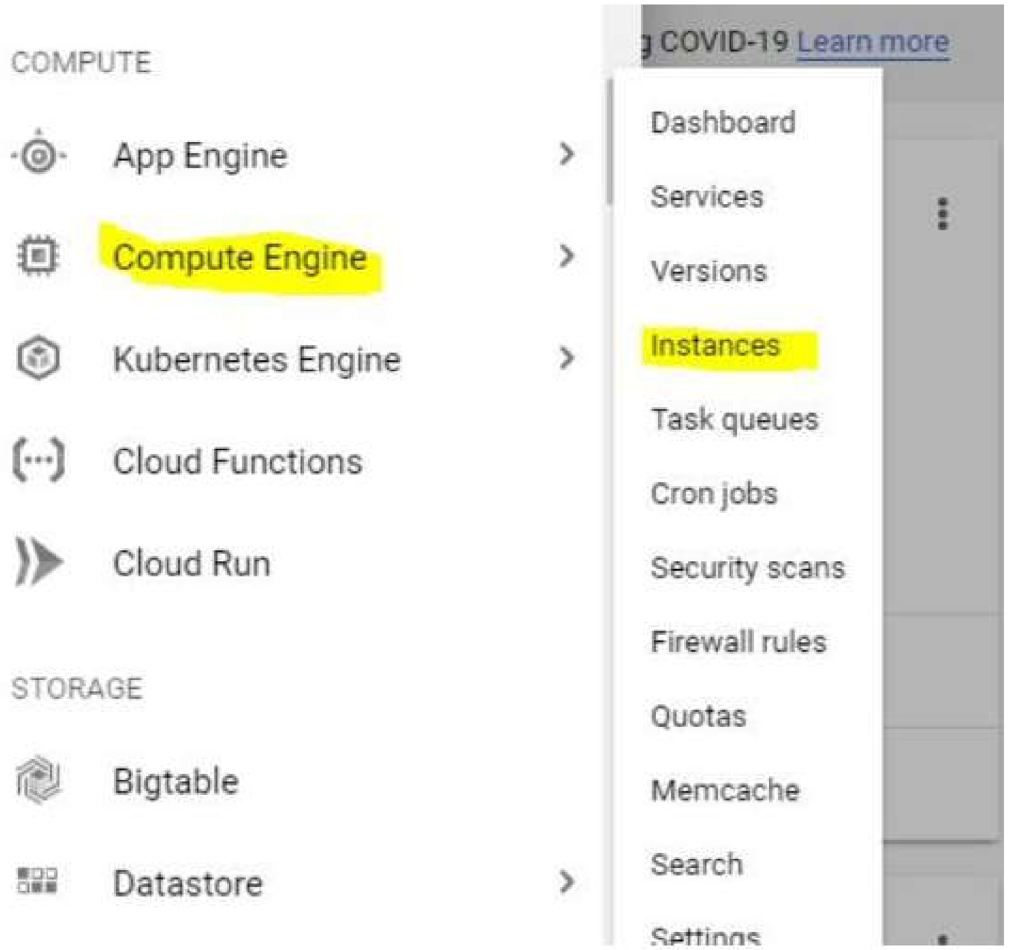
Access to all Cloud Platform Products
Get everything you need to build and run your apps, websites and services, including Firebase and the Google Maps API.

$300 credit for free
Sign up and get $300 to spend on Google Cloud Platform over the next 12 months.

No autocharge after free trial ends
We ask you for your credit card to make sure you are not a robot. You won't be charged unless you manually upgrade to a paid account.

**Step:2**

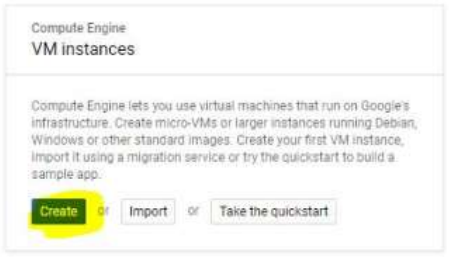
*Click on Instances option present in compute engine inside Navigation menu.*



**Step:3**

*Press create and fill the required details to create vm instance.*

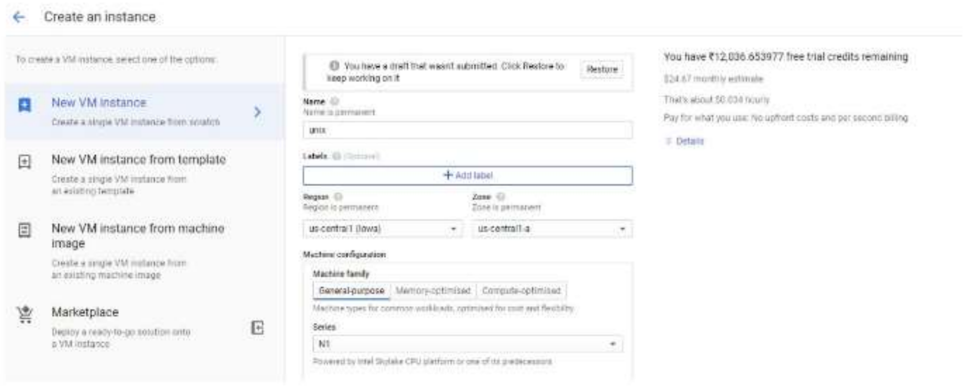
VM instances



Compute Engine

VM instances

Compute Engine lets you use virtual machines that run on Google's infrastructure. Create micro-VMs or larger instances running Debian, Windows or other standard images. Create your first VM instance, import it using a migration service or try the quickstart to build a sample app.



Create an instance

To create a VM instance, select one of the options:

**New VM instance**
Create a single VM instance from scratch

**New VM instance from template**
Create a single VM instance from an existing template

**New VM instance from machine image**
Create a single VM instance from an existing machine image

**Marketplace**
Deploy a ready-to-go solution onto a VM instance

You have a draft that wasn't submitted. Click Restore to keep working on it

Name: unix (Name is permanent)

Labels (Optional)

Region: us-central1 (Iowa) (Region is permanent)

Zone: us-central1-a (Zone is permanent)

Machine configuration

Machine family: General-purpose | Memory-optimised | Compute-optimised

Machine types for common workloads, optimised for cost and flexibility

Series: N1

Powered by Intel Skylake CPU platform or one of its predecessors

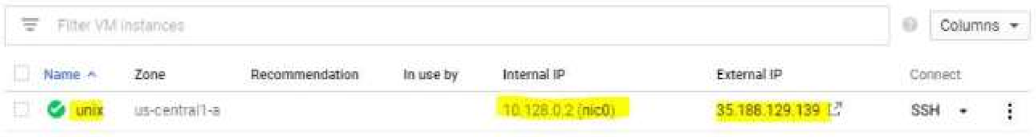
You have ₹12,036.653977 free trial credits remaining

$24.67 monthly estimate

That's about $0.034 hourly

Pay for what you use: No upfront costs and per second billing

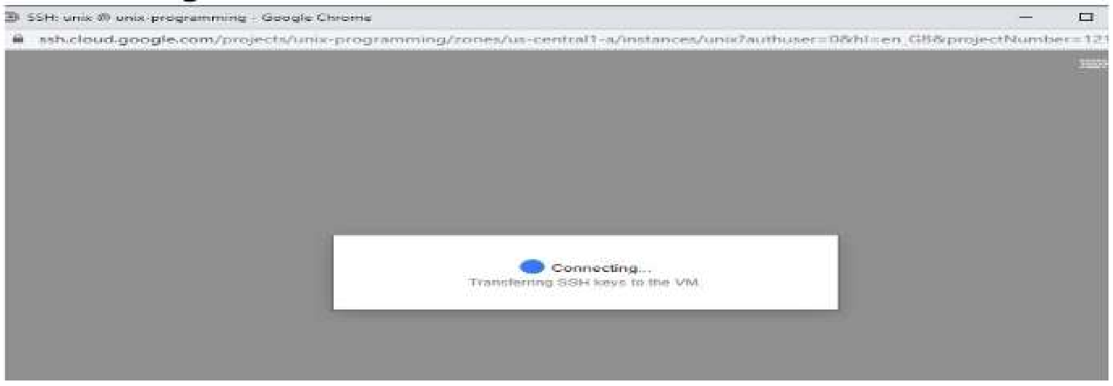
***VM instance is ready.***



Filter VM instances

| Name | Zone | Recommendation | In use by | Internal IP | External IP | Connect |
|---|---|---|---|---|---|---|
| unix | us-central1-a | | | 10.128.0.2 (nic0) | 35.188.129.139 | SSH |

## *Connecting to VM*



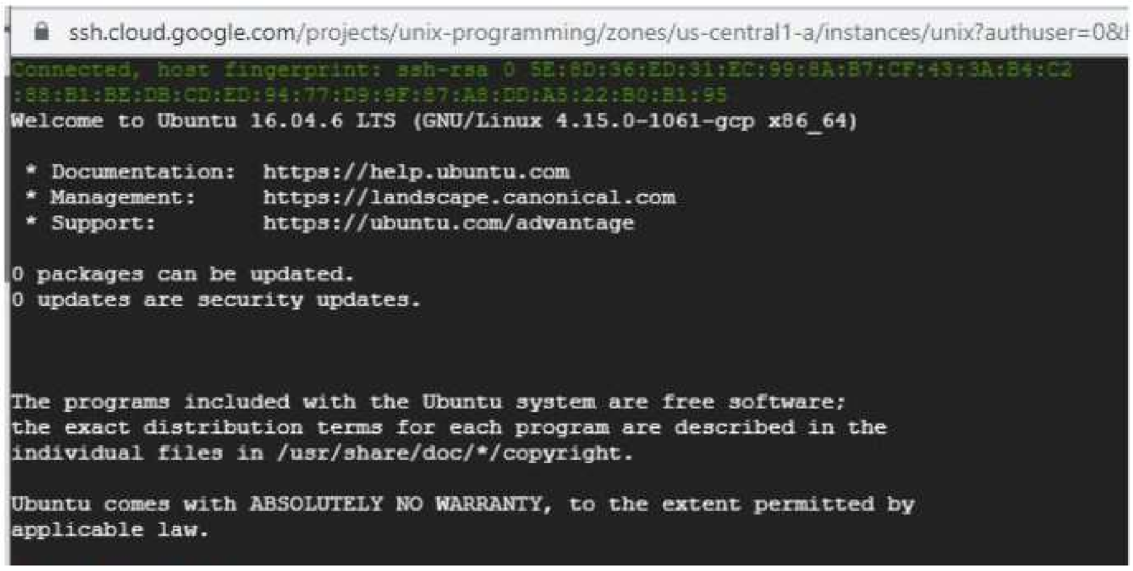
SSH: unix @ unix-programming - Google Chrome

ssh.cloud.google.com/projects/unix-programming/zones/us-central1-a/instances/unix?authuser=0&hl=en_GB&projectNumber=12

Connecting...

Transferring SSH keys to the VM.

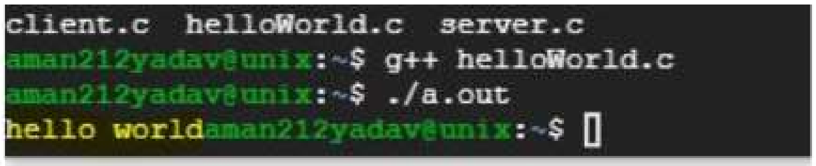
***Running VM***



```
ssh.cloud.google.com/projects/unix-programming/zones/us-central1-a/instances/unix?authuser=0&
Connected, host fingerprint: ssh-rsa 0 5E:8D:36:ED:31:EC:99:8A:B7:CF:43:3A:B4:C2
:88:B1:BE:DB:CD:ED:94:77:D9:9F:87:A8:DD:A5:22:B0:B1:95
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-1061-gcp x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.



The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

**<u>Step:4</u>**

*Writing and executing "Hello World" on google cloud.*



```
client.c  helloWorld.c  server.c
aman212yadav@unix:~$ g++ helloWorld.c
aman212yadav@unix:~$ ./a.out
hello worldaman212yadav@unix:~$ 
```

**<u>Step:5</u>**

*Writing and executing server.c program on google cloud.*

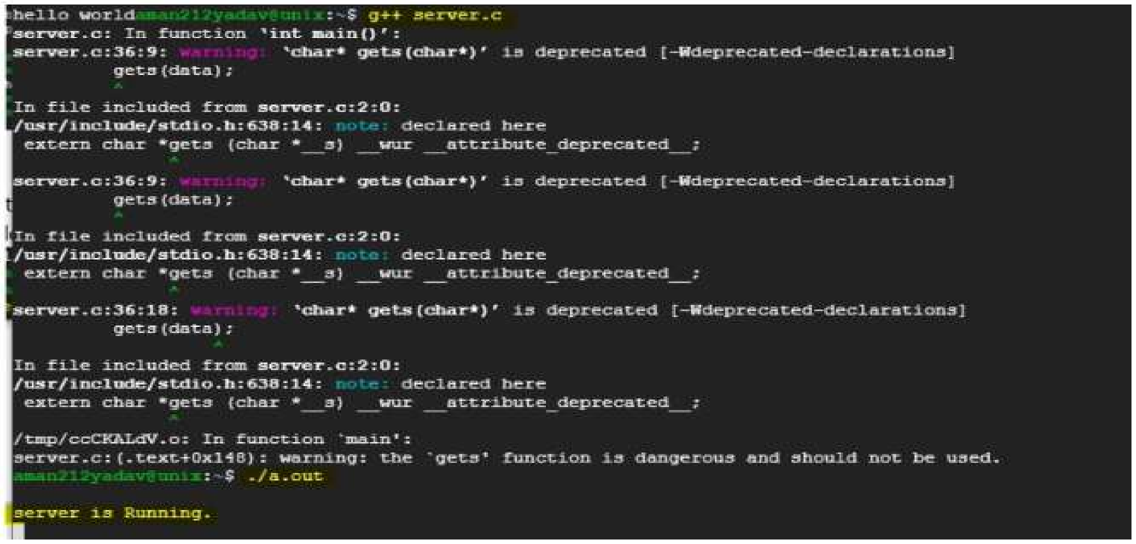
**Code:**

**<u>Server.c</u>**

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <time.h>
#include <unistd.h>
#include <sys/types.h>
#include <arpa/inet.h>
#include <sys/socket.h>
```

```
#include <netinet/in.h>


int main()
{


        char data[1025];
        int sock = 0, clintConnt = 0;
        struct sockaddr_in ipOfServer;
        sock = socket(AF_INET, SOCK_STREAM, 0); // creating
socket
        memset(&ipOfServer, '0', sizeof(ipOfServer));
        memset(data, '\0', sizeof(data));
        ipOfServer.sin_family = AF_INET;
        ipOfServer.sin_addr.s_addr = htonl(INADDR_ANY);
        ipOfServer.sin_port = htons(2020);
        bind(sock, (struct sockaddr*)&ipOfServer ,
sizeof(ipOfServer));
        listen(sock , 20);
    printf("\nserver is Running.\n");
        while(1)
        {


                clintConnt = accept(sock, (struct sockaddr*)NULL,
NULL);
      read(clintConnt, data, sizeof(data)-1);
      printf("server : one client sent me a message and the message is -
> %s \n",data);


      printf("server : enter response message for client -> ");
      gets(data);
                write(clintConnt, data, strlen(data));
                printf("server: response message sent to the client\n");


      close(clintConnt);
      sleep(1);
    }


    return 0;
}
```

**Output:**



```
hello worldaman212yadav@unix:~$ g++ server.c
server.c: In function 'int main()':
server.c:36:9: warning: 'char* gets(char*)' is deprecated [-Wdeprecated-declarations]
          gets(data);
          ^
In file included from server.c:2:0:
/usr/include/stdio.h:638:14: note: declared here
 extern char *gets (char *__s) __wur __attribute_deprecated__;
              ^
server.c:36:9: warning: 'char* gets(char*)' is deprecated [-Wdeprecated-declarations]
          gets(data);
          ^
In file included from server.c:2:0:
/usr/include/stdio.h:638:14: note: declared here
 extern char *gets (char *__s) __wur __attribute_deprecated__;
              ^
server.c:36:18: warning: 'char* gets(char*)' is deprecated [-Wdeprecated-declarations]
          gets(data);
                   ^
In file included from server.c:2:0:
/usr/include/stdio.h:638:14: note: declared here
 extern char *gets (char *__s) __wur __attribute_deprecated__;
              ^
/tmp/ccCKALdV.o: In function `main':
server.c:(.text+0x148): warning: the `gets' function is dangerous and should not be used.
aman212yadav@unix:~$ ./a.out

server is Running.
```

## Step:6

*Write and Execute the client.c program on google cloud.*

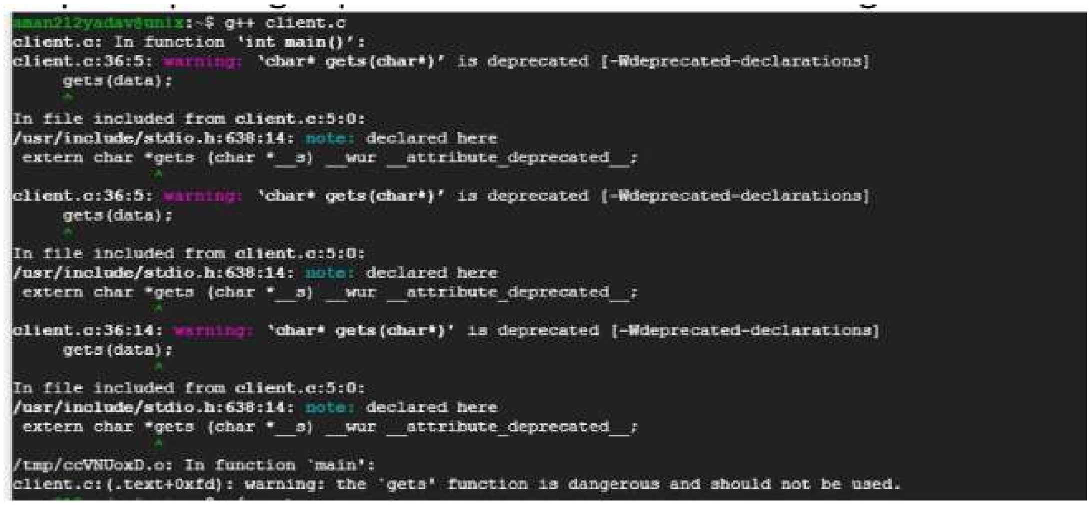
**Code:**

**Client.c**

```
#include <sys/socket.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <netdb.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>
#include <errno.h>
#include <arpa/inet.h>


int main()
{
   int CreateSocket = 0,n = 0;
   char data[1024];
   struct sockaddr_in ipOfServer;


   memset(data, '0' ,sizeof(data));
```

```
    if((CreateSocket = socket(AF_INET, SOCK_STREAM, 0))< 0)
    {
       printf("Socket not created \n");
       return 1;
    }


    ipOfServer.sin_family = AF_INET;
    ipOfServer.sin_port = htons(2020);
    ipOfServer.sin_addr.s_addr = inet_addr("127.0.0.1");


    if(connect(CreateSocket, (struct sockaddr *)&ipOfServer,
sizeof(ipOfServer))<0)
    {
       printf("Some Error occured\n");
       return 1;
    }
    printf("client is running : Enter ur message for server-> ");
    gets(data);
    write(CreateSocket, data, strlen(data));
    memset(data,'\0',sizeof(data));
     read(CreateSocket, data, sizeof(data)-1);
     printf("message from server : %s\n",data);


    return 0;
}
```

**Output:**

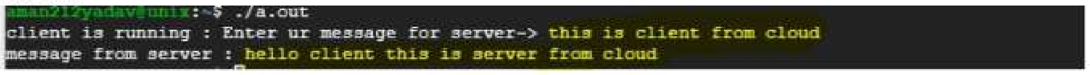


```
aman212yadav@unix:~$ g++ client.c
client.c: In function 'int main()':
client.c:36:5: warning: 'char* gets(char*)' is deprecated [-Wdeprecated-declarations]
     gets(data);
     ^
In file included from client.c:5:0:
/usr/include/stdio.h:638:14: note: declared here
 extern char *gets (char *__s) __wur __attribute_deprecated__;
              ^
client.c:36:5: warning: 'char* gets(char*)' is deprecated [-Wdeprecated-declarations]
     gets(data);
     ^
In file included from client.c:5:0:
/usr/include/stdio.h:638:14: note: declared here
 extern char *gets (char *__s) __wur __attribute_deprecated__;
              ^
client.c:36:14: warning: 'char* gets(char*)' is deprecated [-Wdeprecated-declarations]
     gets(data);
              ^
In file included from client.c:5:0:
/usr/include/stdio.h:638:14: note: declared here
 extern char *gets (char *__s) __wur __attribute_deprecated__;
              ^
/tmp/ccVNUoxD.o: In function `main':
client.c:(.text+0xfd): warning: the `gets' function is dangerous and should not be used.
```

**Result:**

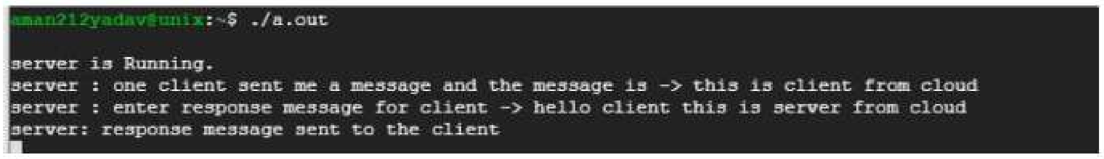
*Connection between server and client is established and Message is exchanged successfully.*

client



```
aman212yadav@unix:~$ ./a.out
client is running : Enter ur message for server-> this is client from cloud
message from server : hello client this is server from cloud
```

Server



```
aman212yadav@unix:~$ ./a.out

server is Running.
server : one client sent me a message and the message is -> this is client from cloud
server : enter response message for client -> hello client this is server from cloud
server: response message sent to the client
```

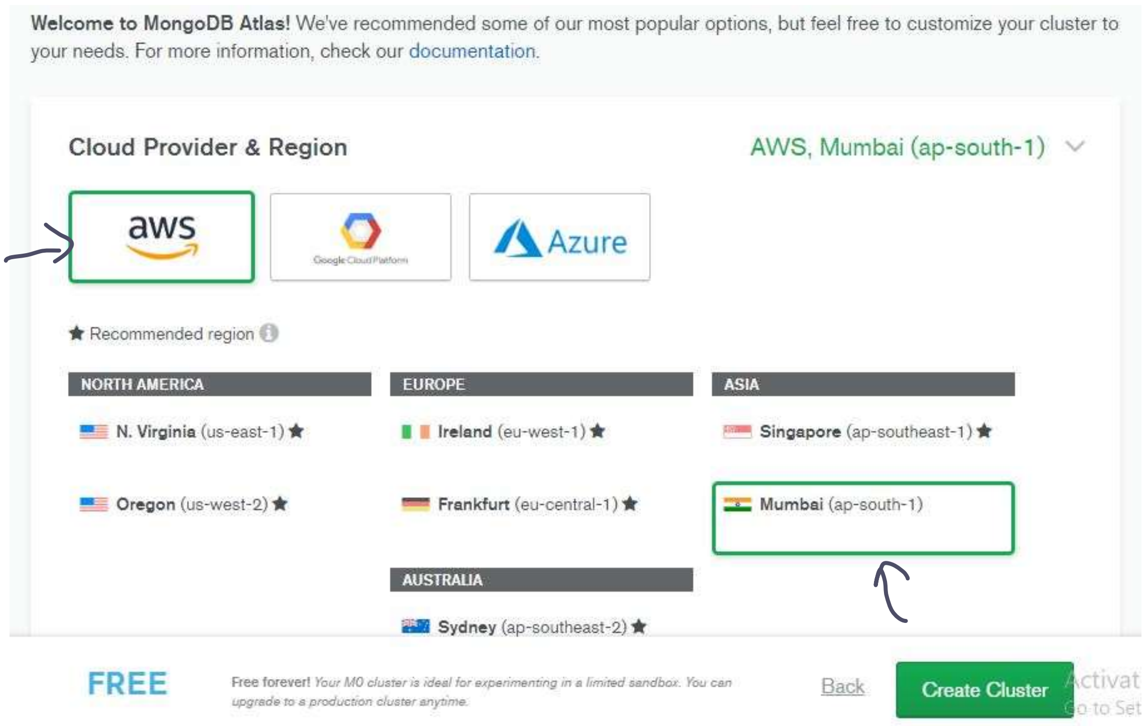
**--- I have attached all related codes to my GitHub account. Link is below:**

https://github.com/omchs10/Network_Programming/tree/master/Assignment_3/client_server

# Deploying Database of web application on MongoDB cloud server (ATLAS).

## Step:1

*Creating a account on MongoDB ATLAS.*



**Welcome to MongoDB Atlas!** We've recommended some of our most popular options, but feel free to customize your cluster to your needs. For more information, check our documentation.

Cloud Provider & Region — AWS, Mumbai (ap-south-1)

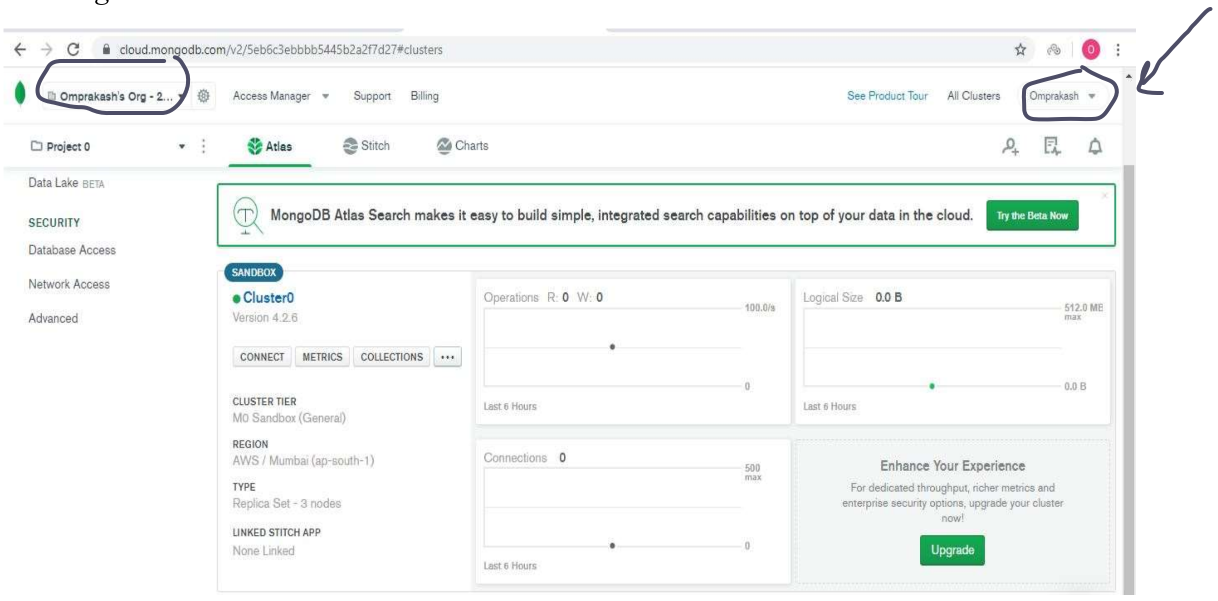
★ Recommended region

| NORTH AMERICA | EUROPE | ASIA |
|---|---|---|
| N. Virginia (us-east-1) ★ | Ireland (eu-west-1) ★ | Singapore (ap-southeast-1) ★ |
| Oregon (us-west-2) ★ | Frankfurt (eu-central-1) ★ | Mumbai (ap-south-1) |
| | AUSTRALIA | |
| | Sydney (ap-southeast-2) ★ | |

FREE — Free forever! *Your M0 cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.*

## Step:2

*Creating a new cluster.*



cloud.mongodb.com/v2/5eb6c3ebbbb5445b2a2f7d27#clusters

MongoDB Atlas Search makes it easy to build simple, integrated search capabilities on top of your data in the cloud. Try the Beta Now

SANDBOX

Cluster0
Version 4.2.6

CLUSTER TIER: M0 Sandbox (General)
REGION: AWS / Mumbai (ap-south-1)
TYPE: Replica Set - 3 nodes
LINKED STITCH APP: None Linked

Operations R: 0 W: 0 — 100.0/s — 0 — Last 6 Hours
Logical Size 0.0 B — 512.0 MB max — 0.0 B — Last 6 Hours
Connections 0 — 500 max — 0 — Last 6 Hours

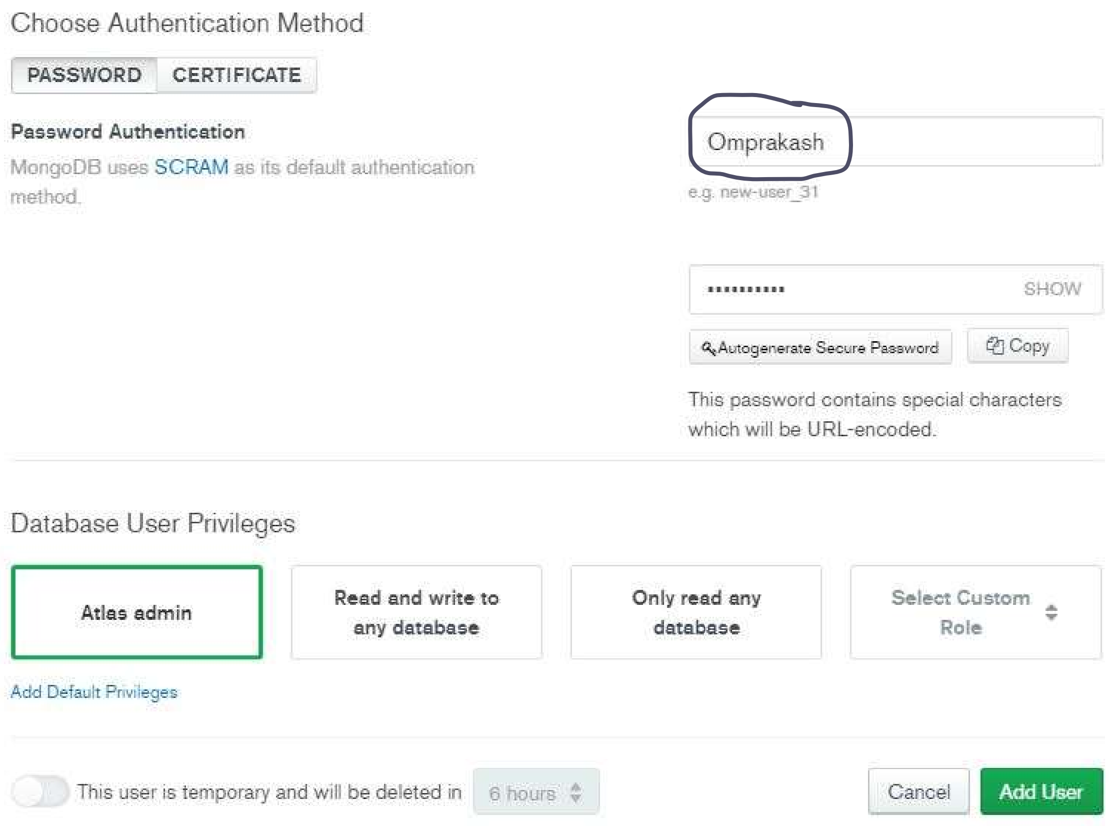
Enhance Your Experience
For dedicated throughput, richer metrics and enterprise security options, upgrade your cluster now!

**Step:3**

*Selecting Cloud provider.*

**Step:4**

*Getting access to cloud database by providing credential.*



Choose Authentication Method

PASSWORD | CERTIFICATE

**Password Authentication**

MongoDB uses SCRAM as its default authentication method.

Omprakash

e.g. new-user_31

•••••••••••• SHOW

Autogenerate Secure Password | Copy

This password contains special characters which will be URL-encoded.

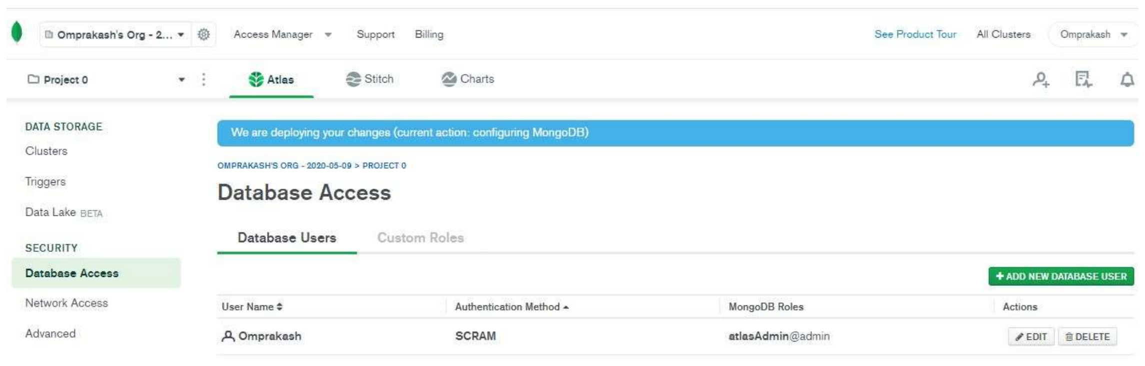
Database User Privileges

[✓ Atlas admin] | Read and write to any database | Only read any database | Select Custom Role

Add Default Privileges

This user is temporary and will be deleted in 6 hours

Cancel | Add User



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

OMPRAKASH'S ORG - 2020-05-09 > PROJECT 0

Database Access

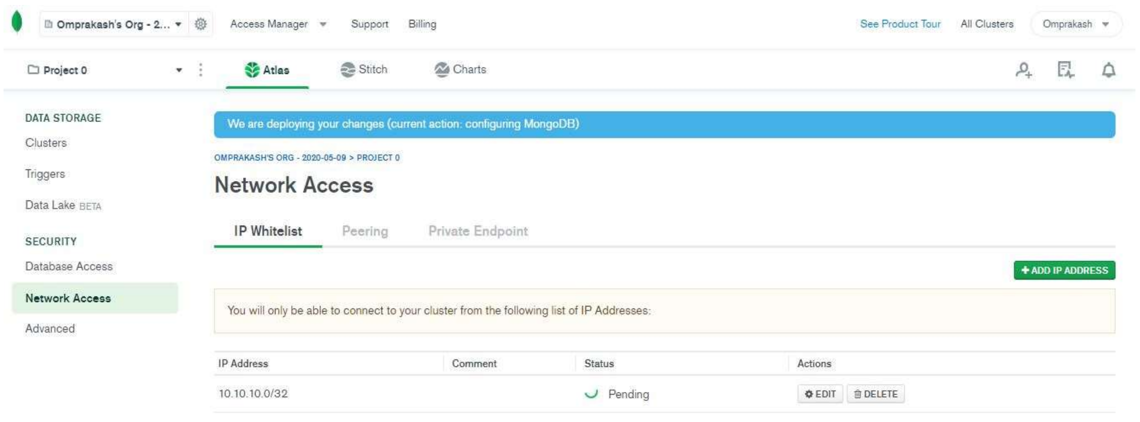
Database Users | Custom Roles

+ ADD NEW DATABASE USER

| User Name | Authentication Method | MongoDB Roles | Actions |
|---|---|---|---|
| Omprakash | SCRAM | atlasAdmin@admin | EDIT DELETE |

## Step:5

*Making it accessible from anywhere by setting IP address as 10.10.10.0*



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

OMPRAKASH'S ORG - 2020-05-09 > PROJECT 0

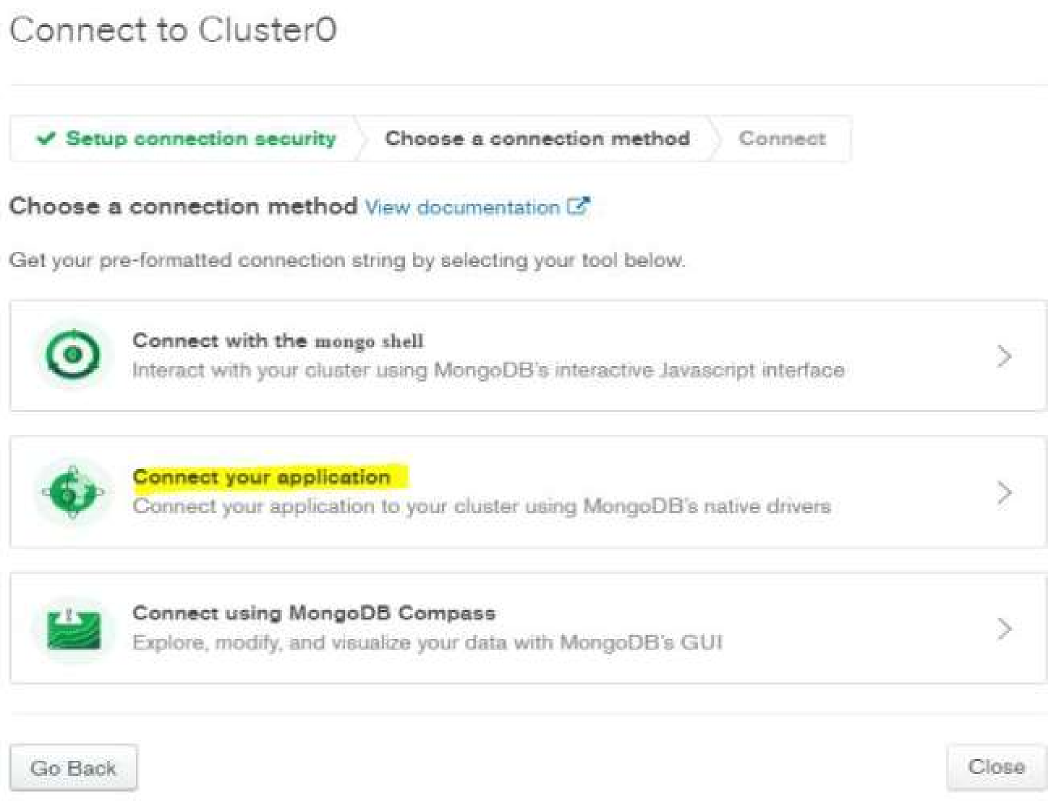
Network Access

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

| IP Address | Comment | Status | Actions |
|---|---|---|---|
| 10.10.10.0/32 |  | Pending | EDIT DELETE |

## Step:6

*Generating secure link to connect web application to database.*



Connect to Cluster0

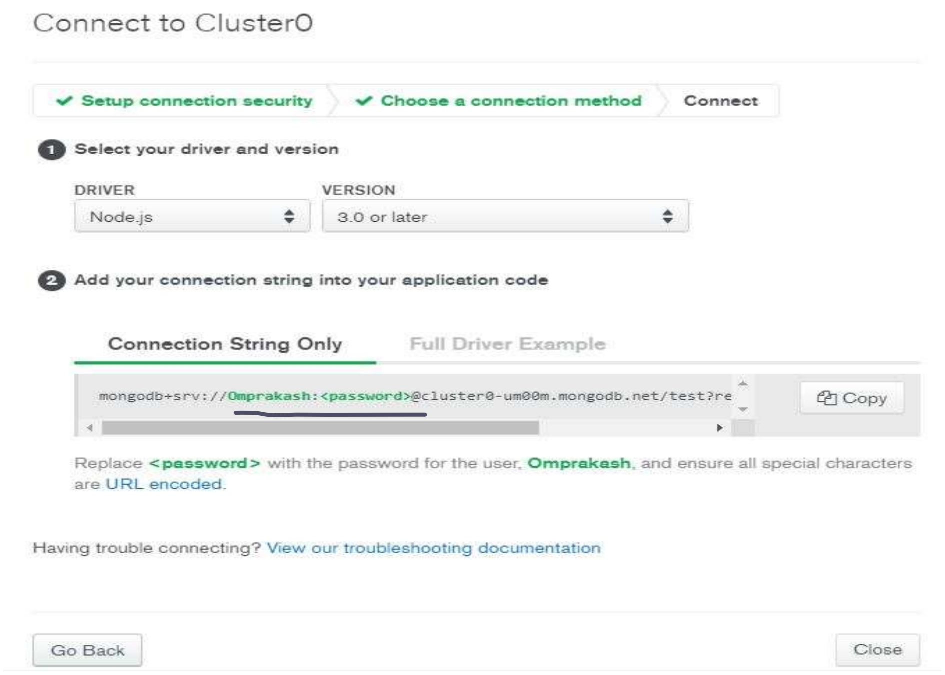
Choose a connection method

Get your pre-formatted connection string by selecting your tool below.

Connect with the mongo shell
Interact with your cluster using MongoDB's interactive Javascript interface

Connect your application
Connect your application to your cluster using MongoDB's native drivers

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



Connect to Cluster0

1 Select your driver and version

DRIVER: Node.js

VERSION: 3.0 or later

2 Add your connection string into your application code

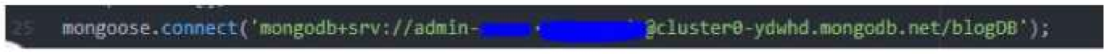
```
mongodb+srv://Omprakash:<password>@cluster0-um00m.mongodb.net/test?re
```

Replace **<password>** with the password for the user, **Omprakash**, and ensure all special characters are URL encoded.

Having trouble connecting? View our troubleshooting documentation

**Step:7**

*Adding generated link to web application.*

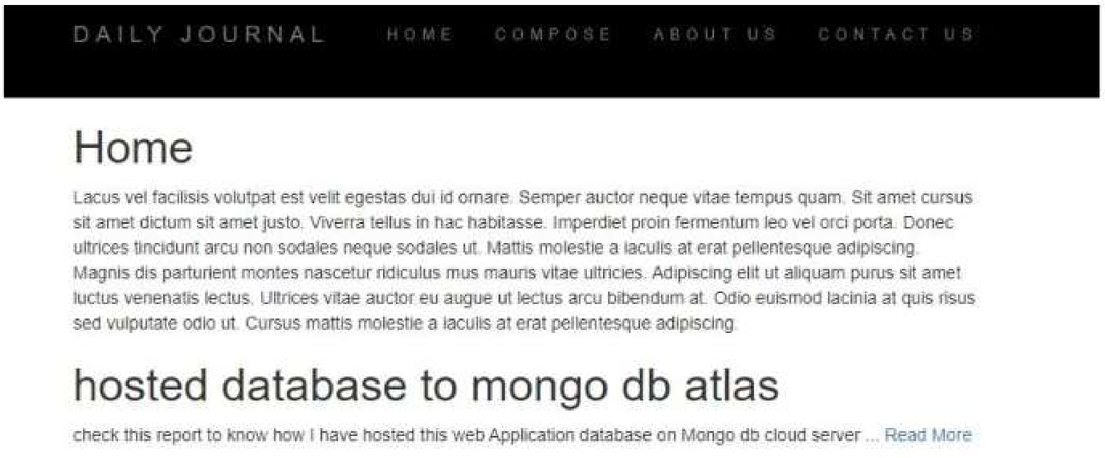


```
mongoose.connect('mongodb+srv://admin-[redacted]·[redacted]@cluster0-ydwhd.mongodb.net/blogDB');
```

(hiding some information for security reasons)

**Step:8**

*Application is ready to communicate with database hosted on cloud.*



Home

Lacus vel facilisis volutpat est velit egestas dui id ornare. Semper auctor neque vitae tempus quam. Sit amet cursus sit amet dictum sit amet justo. Viverra tellus in hac habitasse. Imperdiet proin fermentum leo vel orci porta. Donec ultrices tincidunt arcu non sodales neque sodales ut. Mattis molestie a iaculis at erat pellentesque adipiscing. Magnis dis parturient montes nascetur ridiculus mus mauris vitae ultricies. Adipiscing elit ut aliquam purus sit amet luctus venenatis lectus. Ultrices vitae auctor eu augue ut lectus arcu bibendum at. Odio euismod lacinia at quis risus sed vulputate odio ut. Cursus mattis molestie a iaculis at erat pellentesque adipiscing.

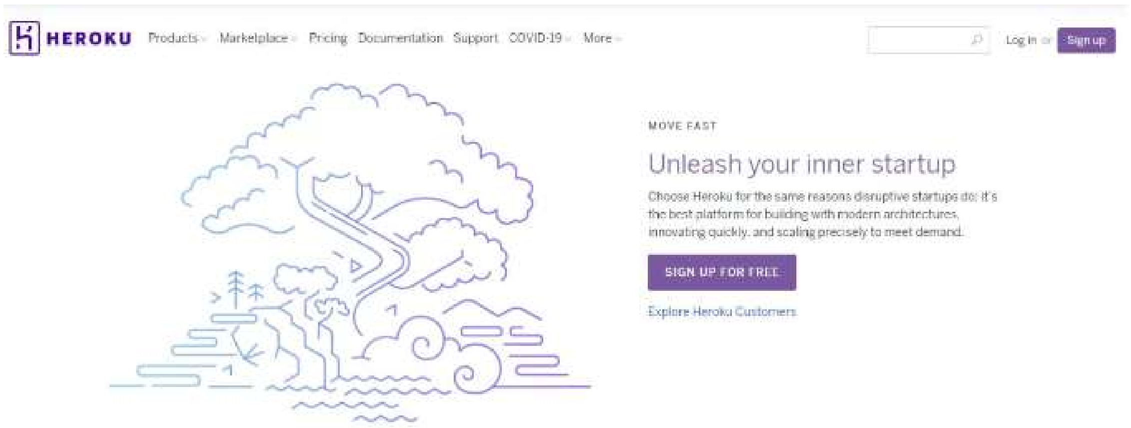
hosted database to mongo db atlas

check this report to know how I have hosted this web Application database on Mongo db cloud server ... Read More

## Deploying the blog writing application on HEROKU cloud service.

**Step:1**

*Creating a HEROKU account.*



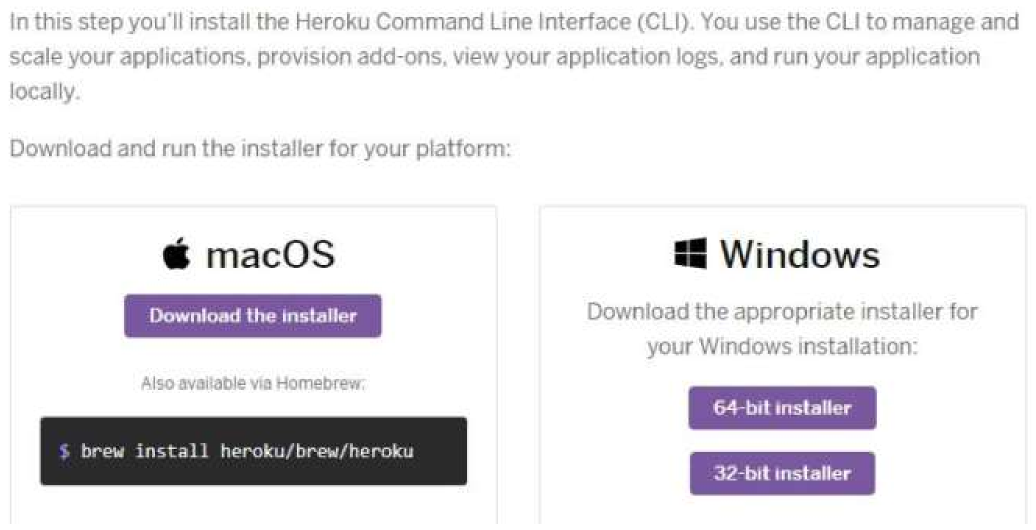
MOVE FAST

Unleash your inner startup

Choose Heroku for the same reasons disruptive startups do: it's the best platform for building with modern architectures, innovating quickly, and scaling precisely to meet demand.

**Step:2**

*Installing HEROKU CLI (Command Line Interface).*



In this step you'll install the Heroku Command Line Interface (CLI). You use the CLI to manage and scale your applications, provision add-ons, view your application logs, and run your application locally.

Download and run the installer for your platform:

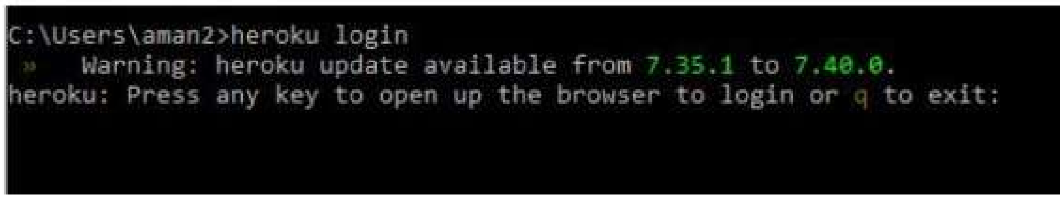
Also available via Homebrew:

```
$ brew install heroku/brew/heroku
```

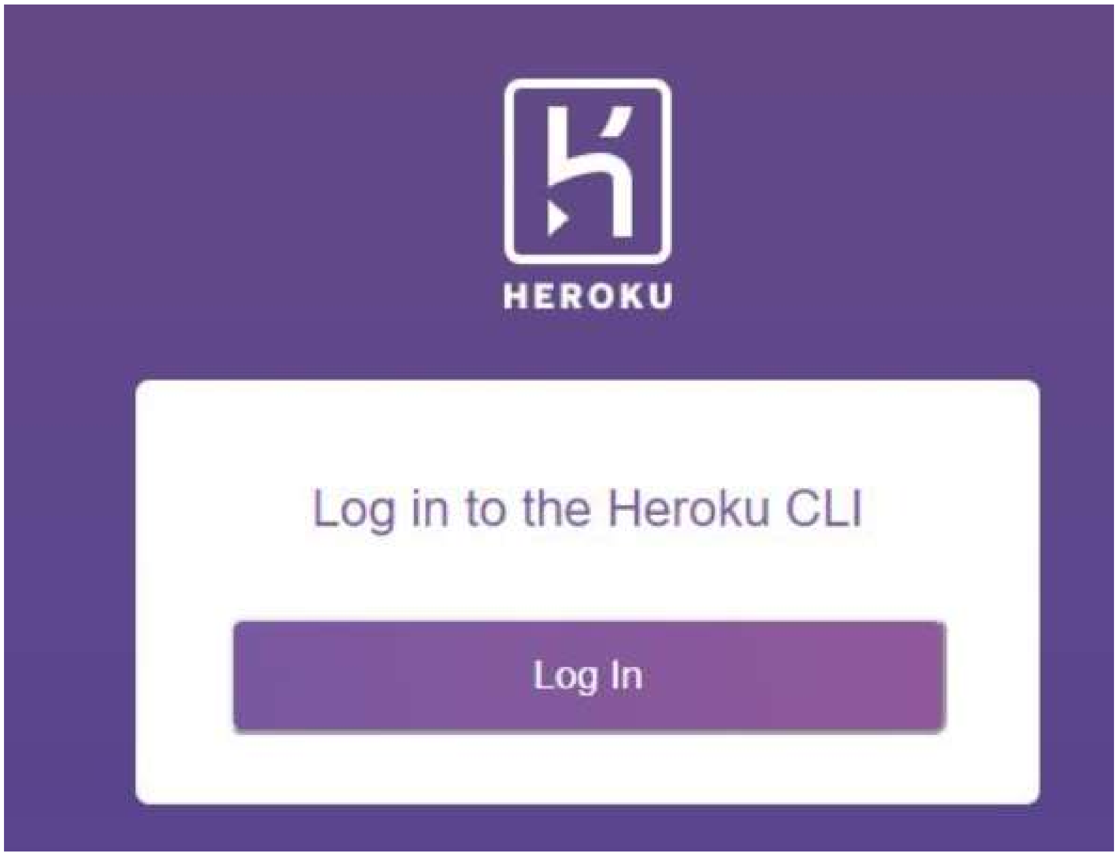
Download the appropriate installer for your Windows installation:

**Step:3**

*Login to HEROKU using HEROKU login command.*

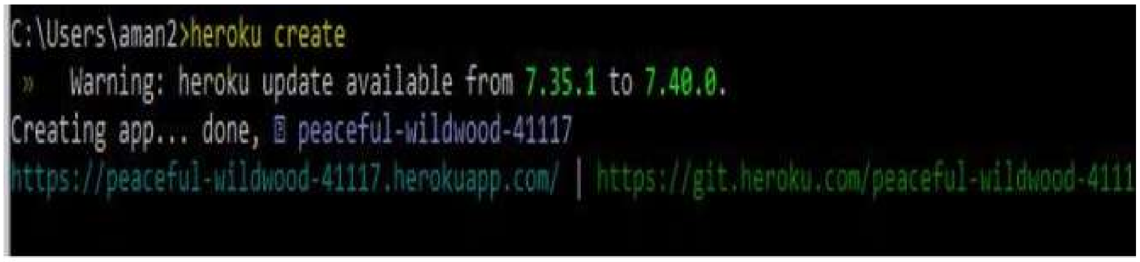


```
C:\Users\aman2>heroku login
 »   Warning: heroku update available from 7.35.1 to 7.40.0.
heroku: Press any key to open up the browser to login or q to exit:
```



**Step:4**

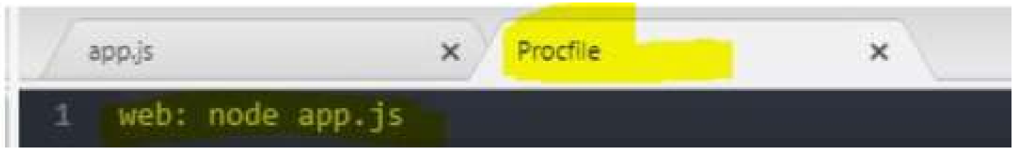
*Creating application on HEROKU which will prepare HEROKU to receive our source code.*



```
C:\Users\aman2>heroku create
 »   Warning: heroku update available from 7.35.1 to 7.40.0.
Creating app... done, ⬢ peaceful-wildwood-41117
https://peaceful-wildwood-41117.herokuapp.com/ | https://git.heroku.com/peaceful-wildwood-4111
```

**Step:5**

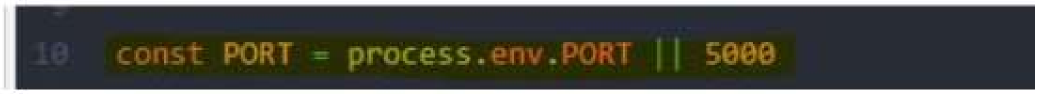
*Defining a procfile to declare explicitly what command HEROKU should use to start the application.*



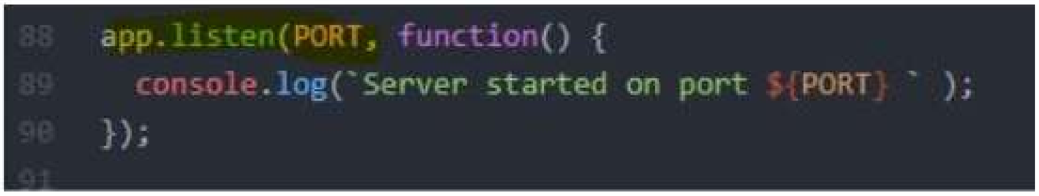
```
web: node app.js
```

**Step:6**

*Defining PORT on which application should listen/start.*



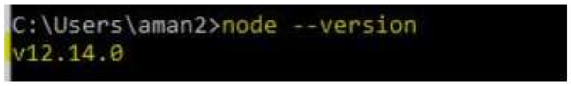
```
const PORT = process.env.PORT || 5000
```



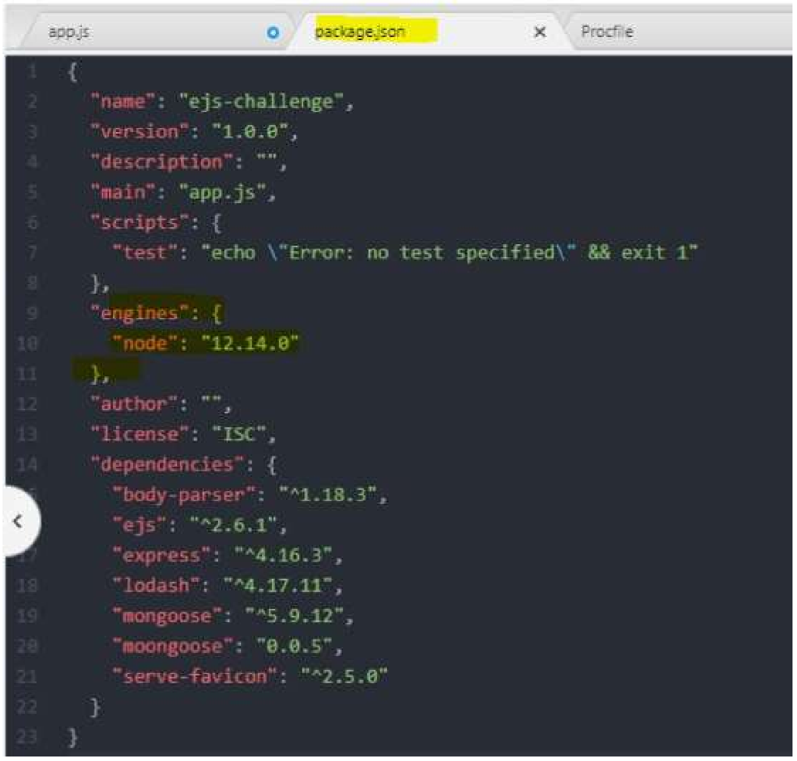
```
app.listen(PORT, function() {
  console.log(`Server started on port ${PORT} ` );
});
```

**Step:7**

*Finding Version of node and adding package.json file.*



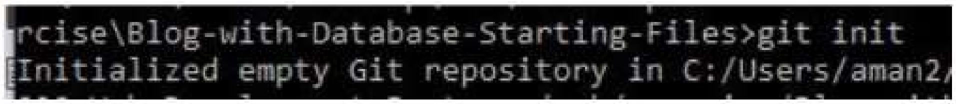
```
C:\Users\aman2>node --version
v12.14.0
```



```
{
  "name": "ejs-challenge",
  "version": "1.0.0",
  "description": "",
  "main": "app.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "engines": {
    "node": "12.14.0"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "body-parser": "^1.18.3",
    "ejs": "^2.6.1",
    "express": "^4.16.3",
    "lodash": "^4.17.11",
    "mongoose": "^5.9.12",
    "moongoose": "0.0.5",
    "serve-favicon": "^2.5.0"
  }
}
```

**<u>Step:8</u>**

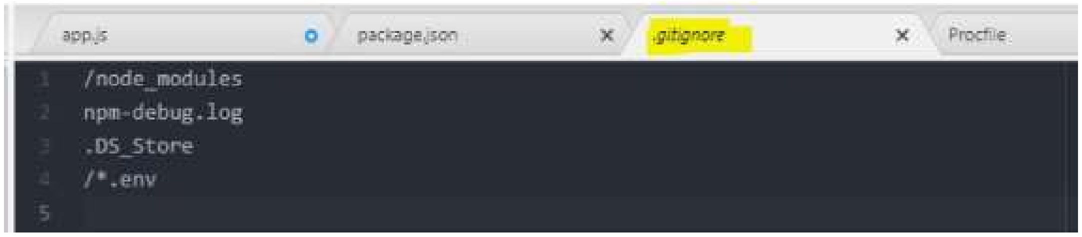
*Initialising git repository.*



```
rcise\Blog-with-Database-Starting-Files>git init
Initialized empty Git repository in C:/Users/aman2/
```

**<u>Step:9</u>**

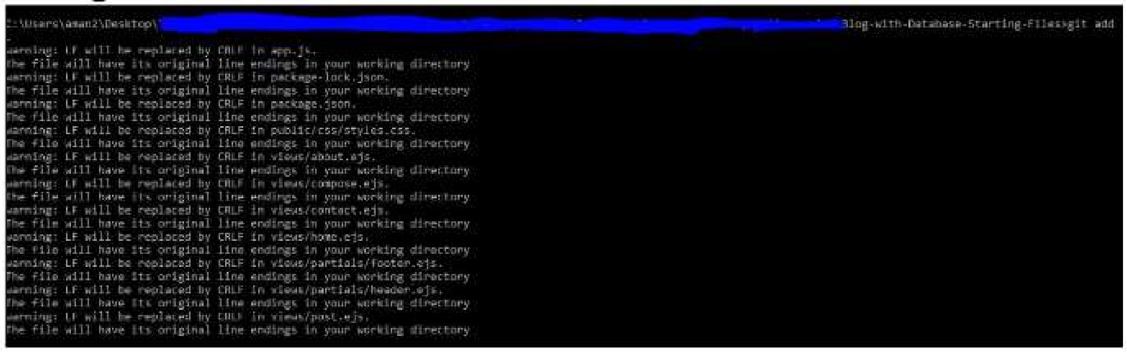
*Adding .gitignore file to avoid uploading unnecessary file/folder.*



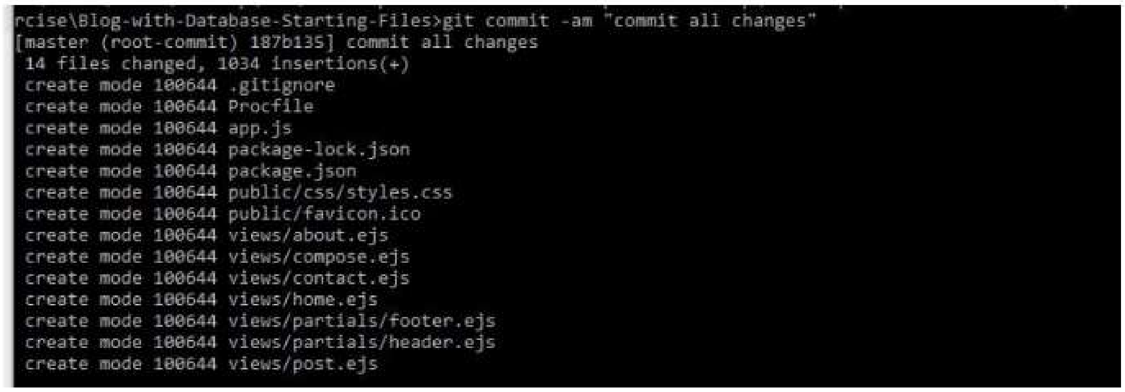
```
/node_modules
npm-debug.log
.DS_Store
/*.env
```

**<u>Step:10</u>**

*Adding all files to staging area and committing all changes.*



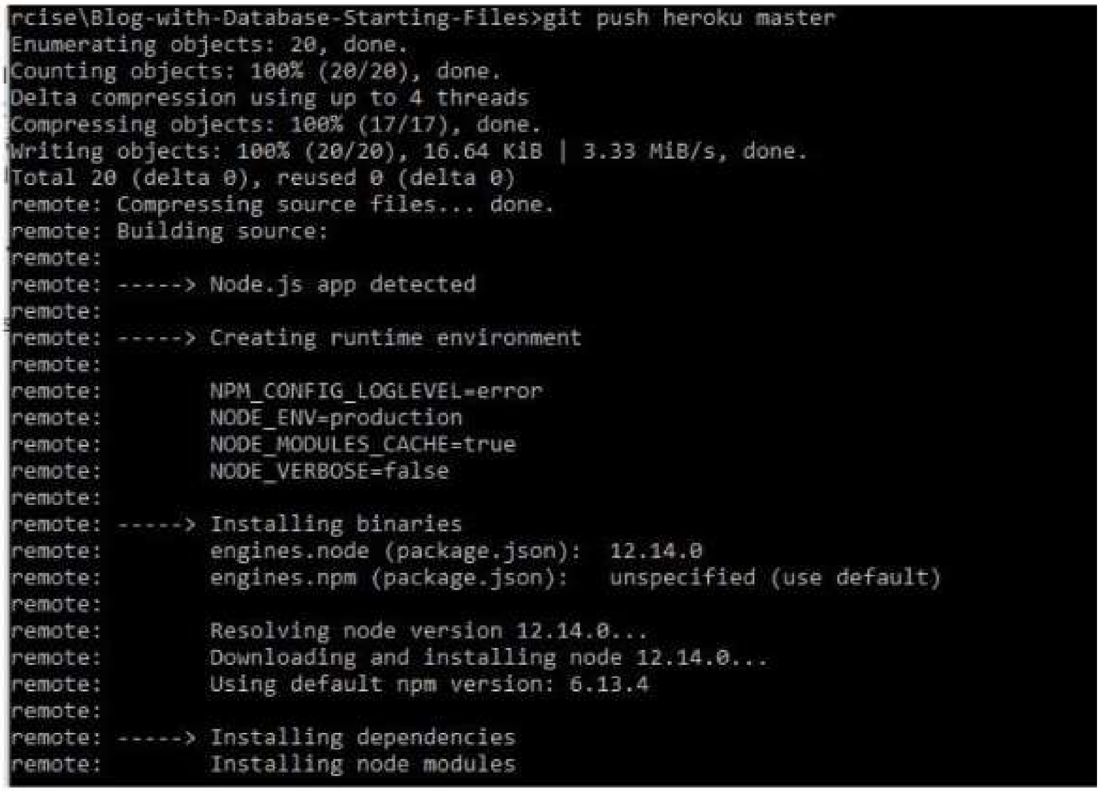
```
C:\Users\aman2\Desktop\[redacted]Blog-with-Database-Starting-Files>git add
.
warning: LF will be replaced by CRLF in app.js.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in package-lock.json.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in package.json.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in public/css/styles.css.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in views/about.ejs.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in views/compose.ejs.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in views/contact.ejs.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in views/home.ejs.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in views/partials/footer.ejs.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in views/partials/header.ejs.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in views/post.ejs.
The file will have its original line endings in your working directory
```



```
rcise\Blog-with-Database-Starting-Files>git commit -am "commit all changes"
[master (root-commit) 187b135] commit all changes
 14 files changed, 1034 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 Procfile
 create mode 100644 app.js
 create mode 100644 package-lock.json
 create mode 100644 package.json
 create mode 100644 public/css/styles.css
 create mode 100644 public/favicon.ico
 create mode 100644 views/about.ejs
 create mode 100644 views/compose.ejs
 create mode 100644 views/contact.ejs
 create mode 100644 views/home.ejs
 create mode 100644 views/partials/footer.ejs
 create mode 100644 views/partials/header.ejs
 create mode 100644 views/post.ejs
```

**<u>Step:11</u>**

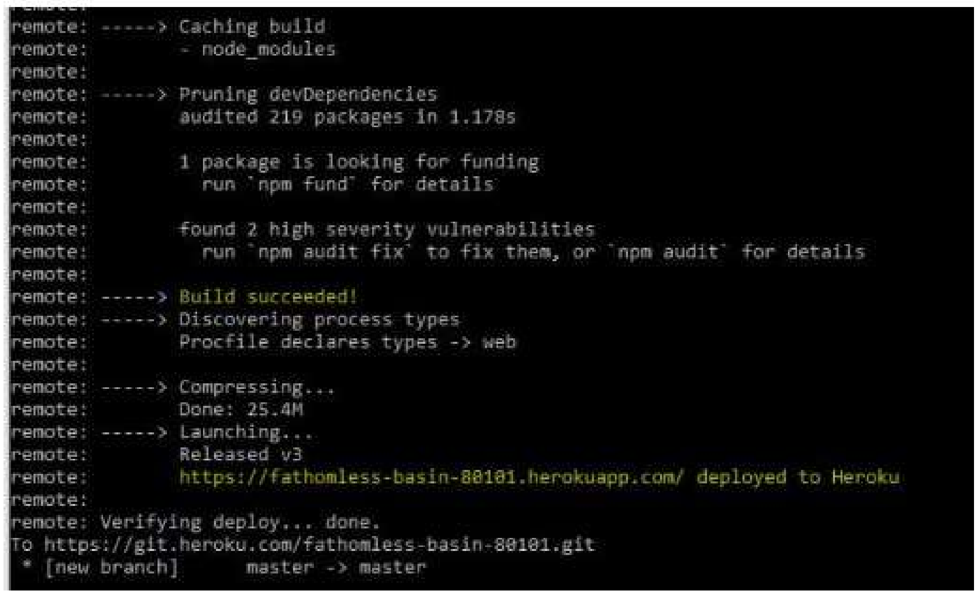
*Pushing all changes to remote repository of HEROKU master.*



```
rcise\Blog-with-Database-Starting-Files>git push heroku master
Enumerating objects: 20, done.
Counting objects: 100% (20/20), done.
Delta compression using up to 4 threads
Compressing objects: 100% (17/17), done.
Writing objects: 100% (20/20), 16.64 KiB | 3.33 MiB/s, done.
Total 20 (delta 0), reused 0 (delta 0)
remote: Compressing source files... done.
remote: Building source:
remote:
remote: -----> Node.js app detected
remote:
remote: -----> Creating runtime environment
remote:
remote:        NPM_CONFIG_LOGLEVEL=error
remote:        NODE_ENV=production
remote:        NODE_MODULES_CACHE=true
remote:        NODE_VERBOSE=false
remote:
remote: -----> Installing binaries
remote:        engines.node (package.json):  12.14.0
remote:        engines.npm (package.json):   unspecified (use default)
remote:
remote:        Resolving node version 12.14.0...
remote:        Downloading and installing node 12.14.0...
remote:        Using default npm version: 6.13.4
remote:
remote: -----> Installing dependencies
remote:        Installing node modules
```

**Step:12**

*Application deploy successfully on cloud.*



```
remote: -----> Caching build
remote:        - node_modules
remote:
remote: -----> Pruning devDependencies
remote:        audited 219 packages in 1.178s
remote:
remote:        1 package is looking for funding
remote:          run `npm fund` for details
remote:
remote:        found 2 high severity vulnerabilities
remote:          run `npm audit fix` to fix them, or `npm audit` for details
remote:
remote: -----> Build succeeded!
remote: -----> Discovering process types
remote:        Procfile declares types -> web
remote:
remote: -----> Compressing...
remote:        Done: 25.4M
remote: -----> Launching...
remote:        Released v3
remote:        https://fathomless-basin-80101.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/fathomless-basin-80101.git
 * [new branch]      master -> master
```

URL-> https://fathomless-basin-80101.herokuapp.com/