Name - Aman Yadav
Roll No - 171210007 (CSE 3rd year)
Subject - Network programming

Summary -

I have created a google cloud account and created a vm instance
Apart from that I have made a simple blog writing web application using basic web technologies
html,css,javascript ,nodejs and mongodb as its database .

so In this report I am going to write about three things
A) How I create vm instance on google cloud and run client sever program over there.
B) how I have deployed **database** of **Blog writing web application** on mongodb cloud server (**ATLAS**).
C) how I have deployed web Application on **HEROKU** cloud web service.

End results -

Publically accessible URL :
https://fathomless-basin-80101.herokuapp.com/

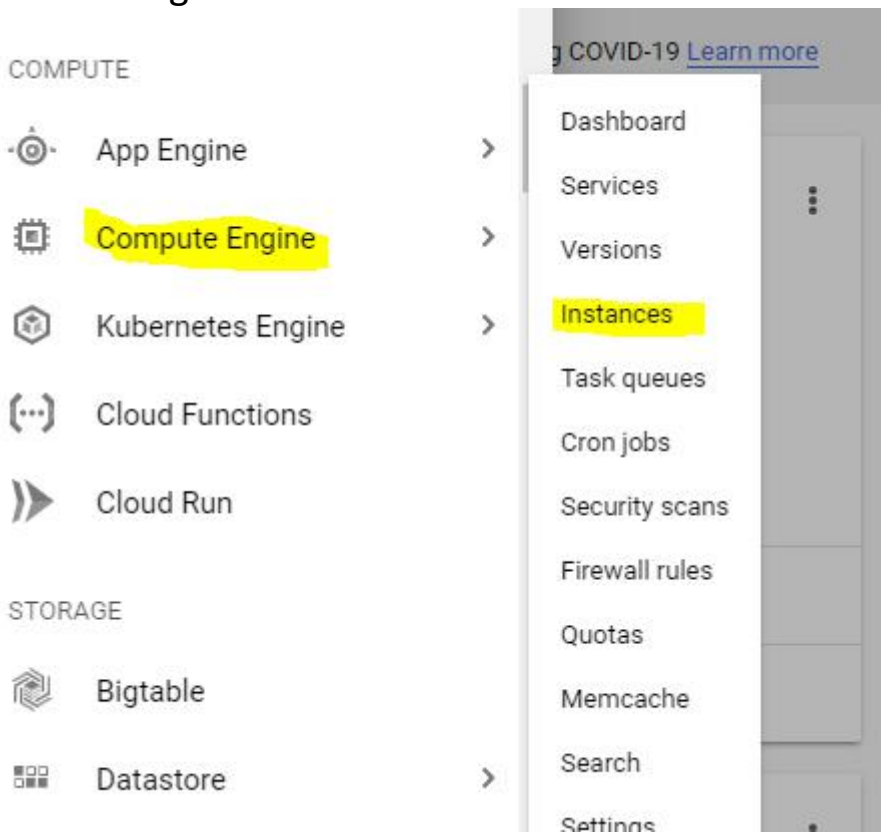
GitHub link for code :

https://github.com/aman212yadav/assignment/tree/master/assignment3

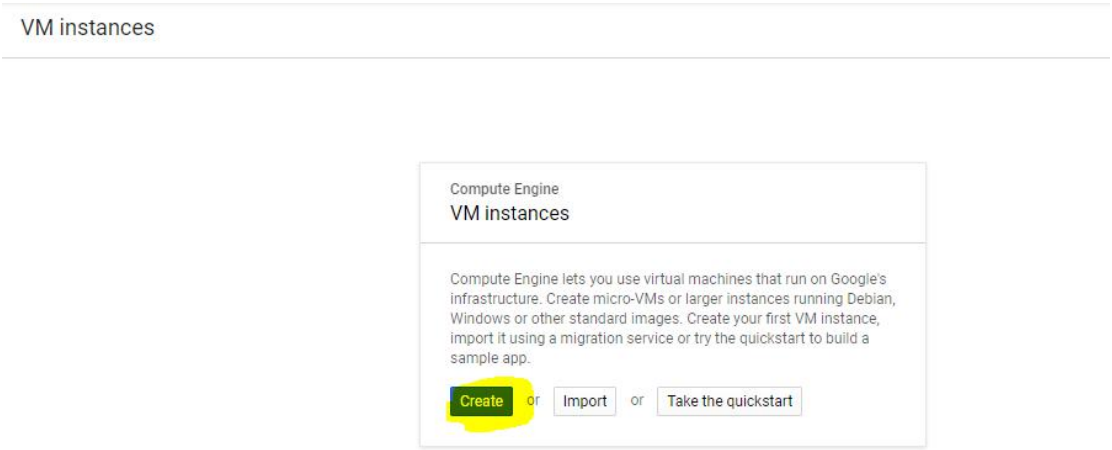
Creating VM instance on google cloud and running unix program on it.

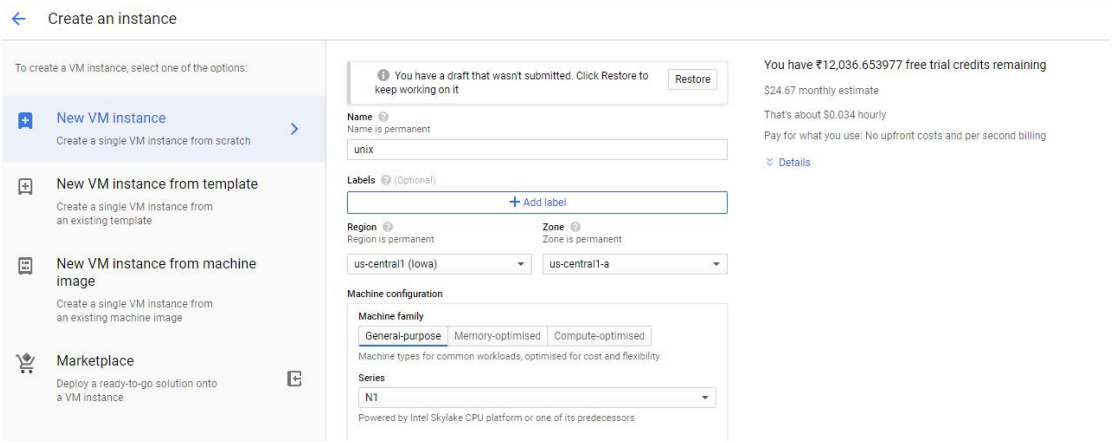
Step1-> create google account.

Step2-> click on **Instances** option present in **compute engine** inside Navigation menu.



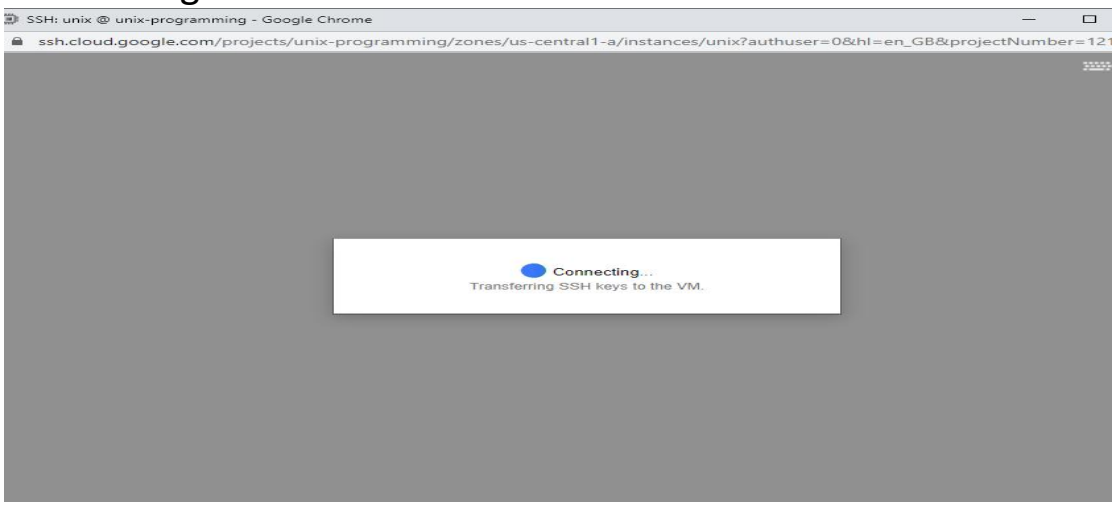
Step3-> Press create and fill the required details to create vm instance.



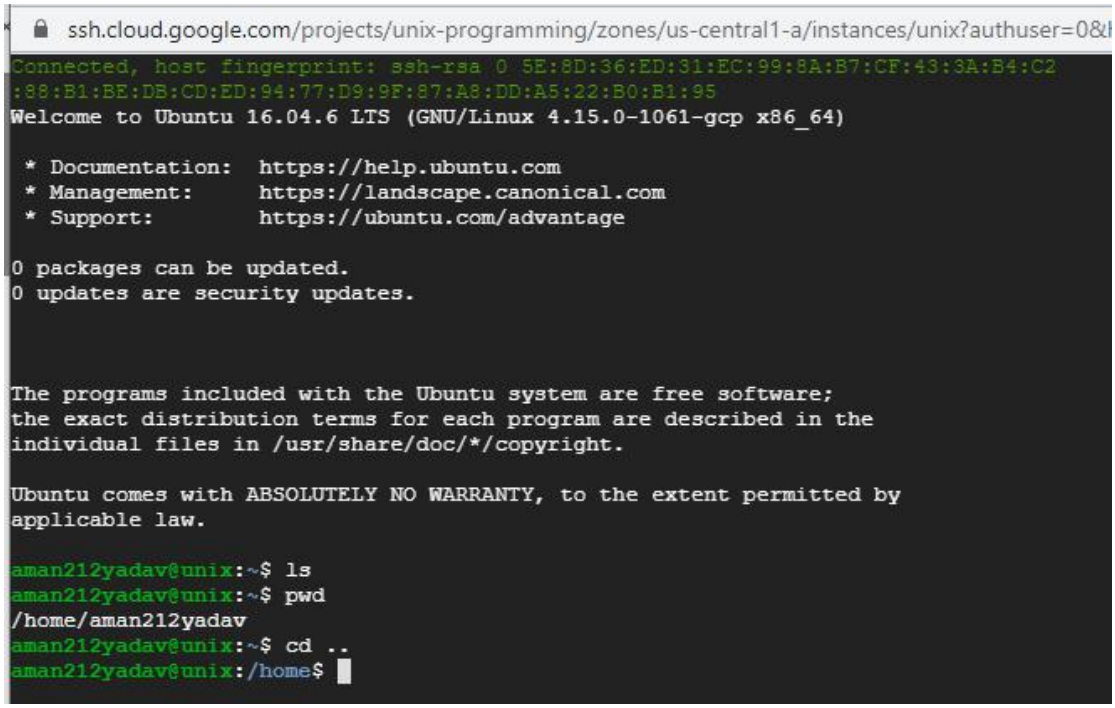


VM instance is ready

Connecting to VM



Running VM



Step4-> writing and executing hello world program on google cloud.

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

Step5-> writing and executing **server.c** file on google cloud.

```
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.
```

Step6->Opening separte vm terminal and running **client.c** file

```
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.
```

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

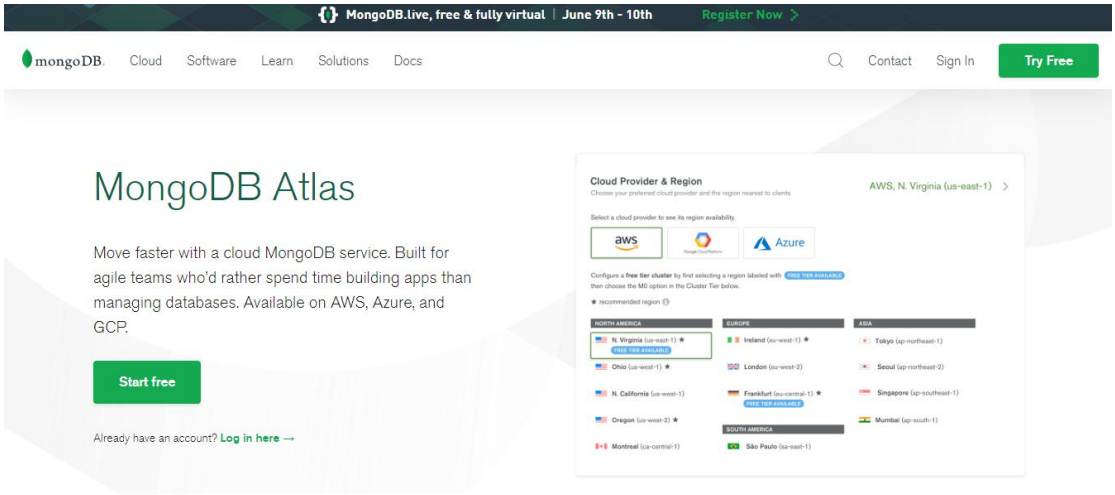
```
aman212yadav@unix:~$ ./a.out

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
```

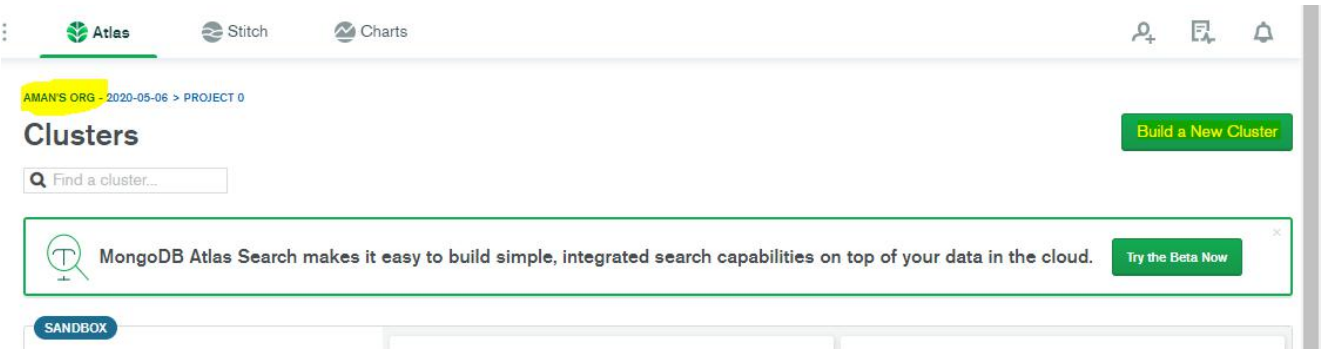
I have attached all the related code on my github account.
Link ->
https://github.com/aman212yadav/assignment/tree/master/assignment3

Deploying database of the web application on Mongo Db cloud server (ATLAS).

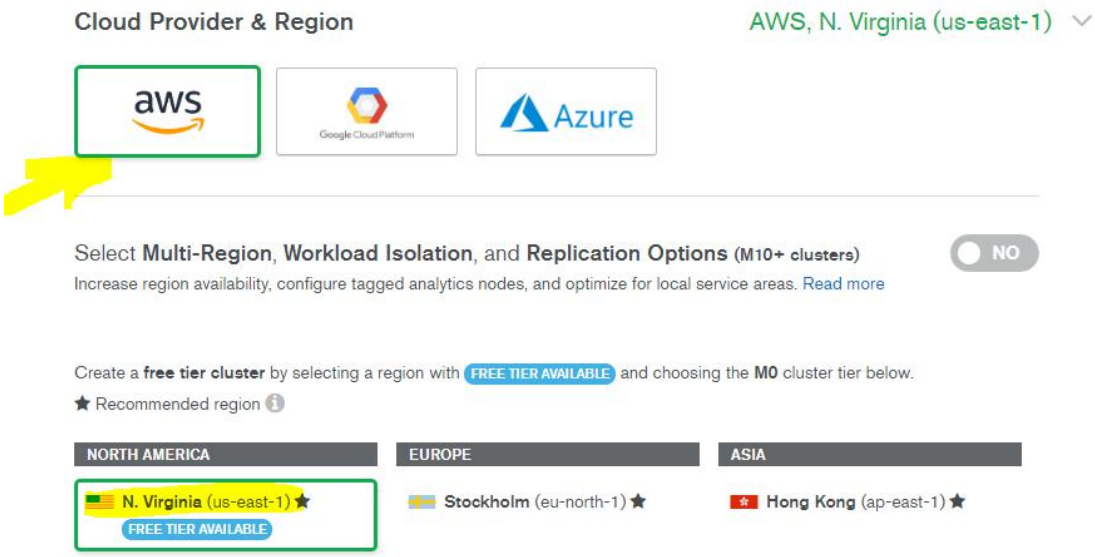
Step 1-> creating a mongoDB atlas account.



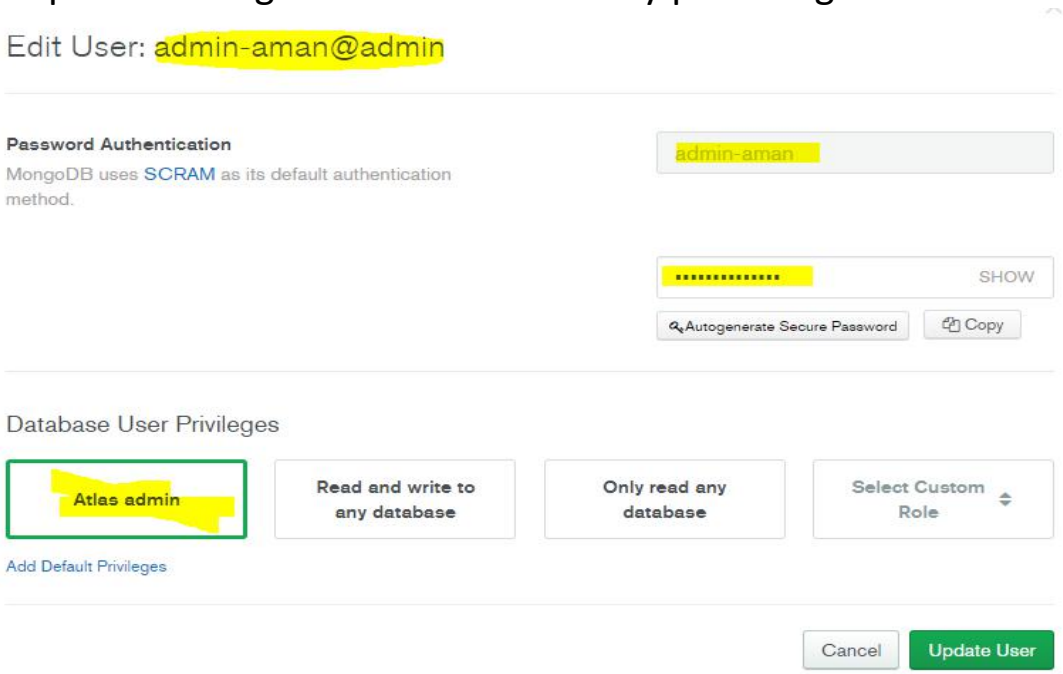
Step 2-> creating a new cluster



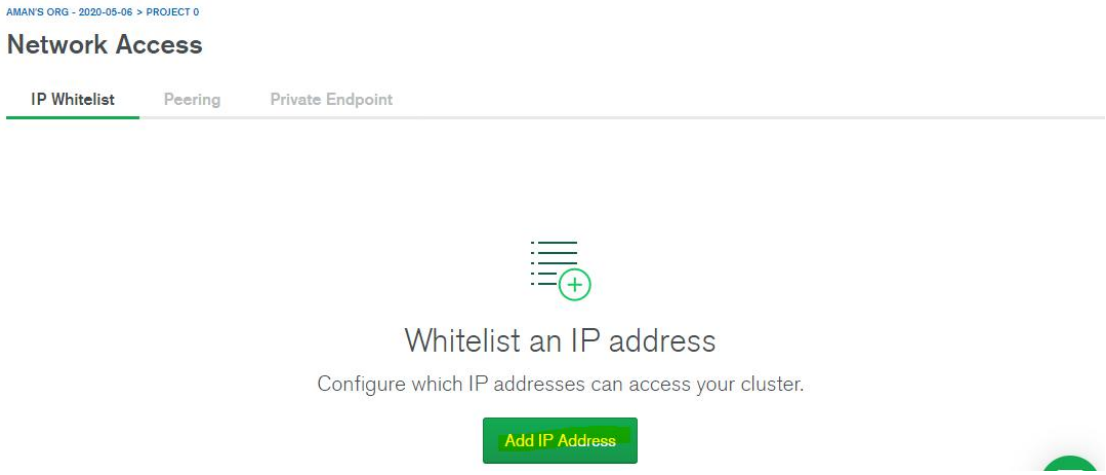
Step 3-> selecting cloud provider.

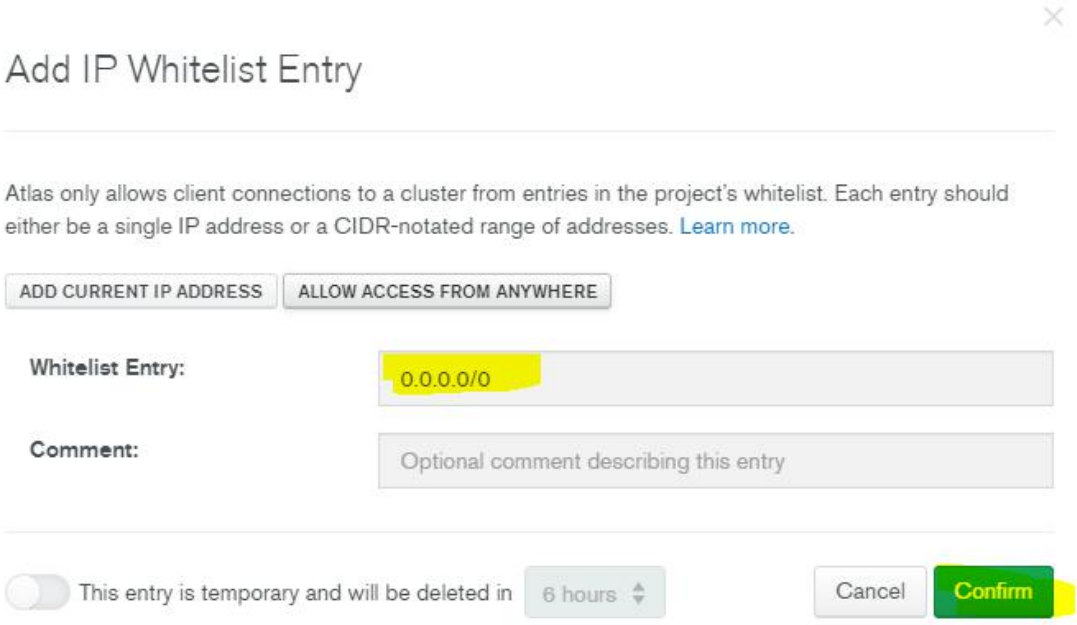


Step 4-> Getting access to database by providing credential.

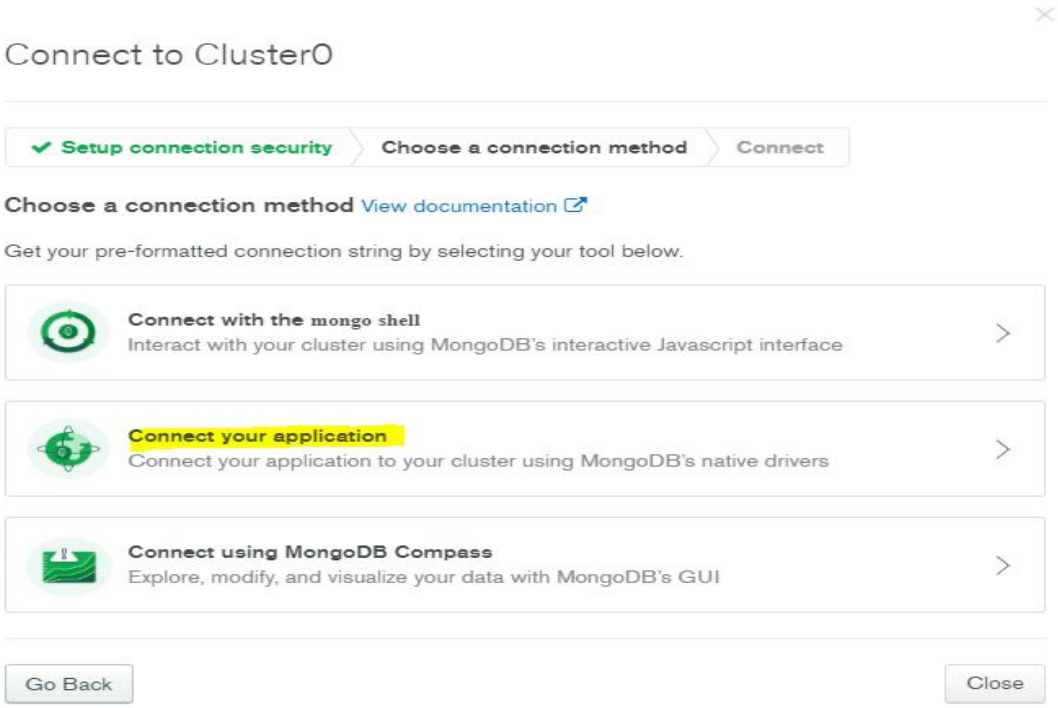


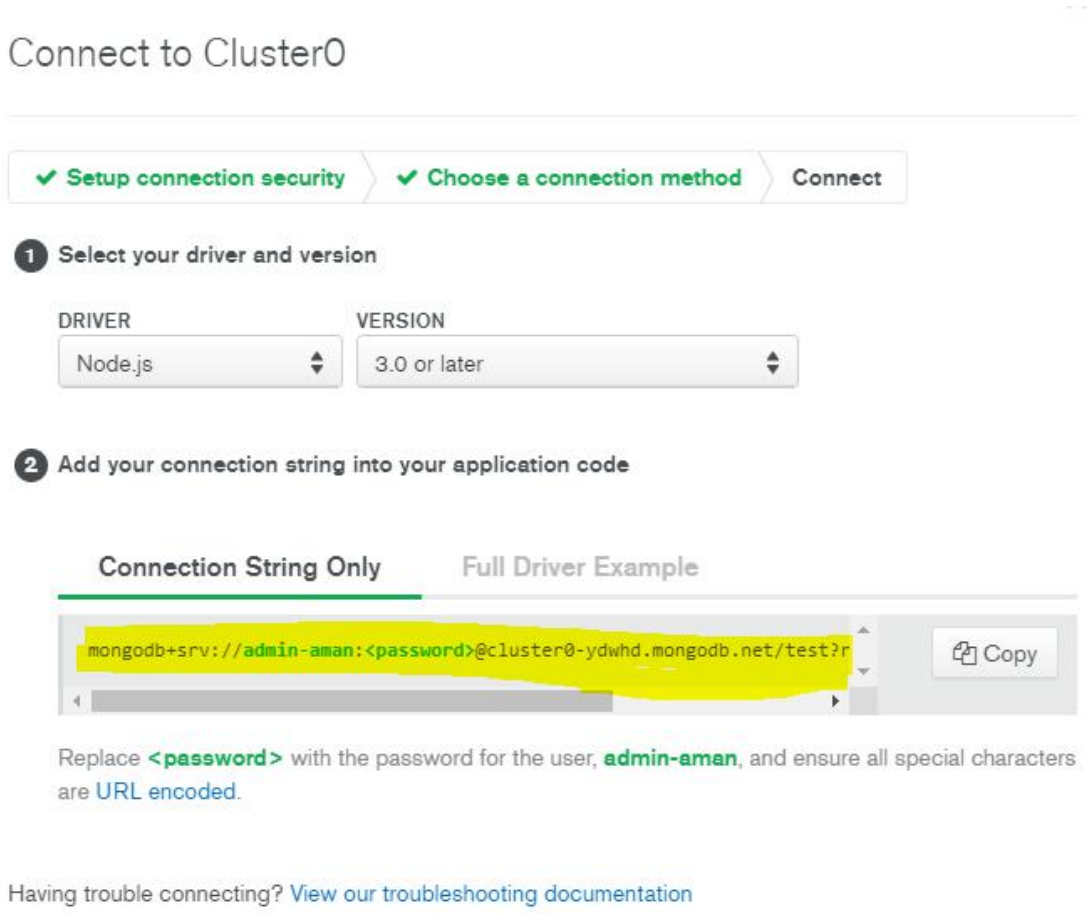
Step 5-> Making it accessible from Anywhere by setting a global IP address (0.0.0.0) .





Step 6-> Generating secure link to connect web application to the hosted database.



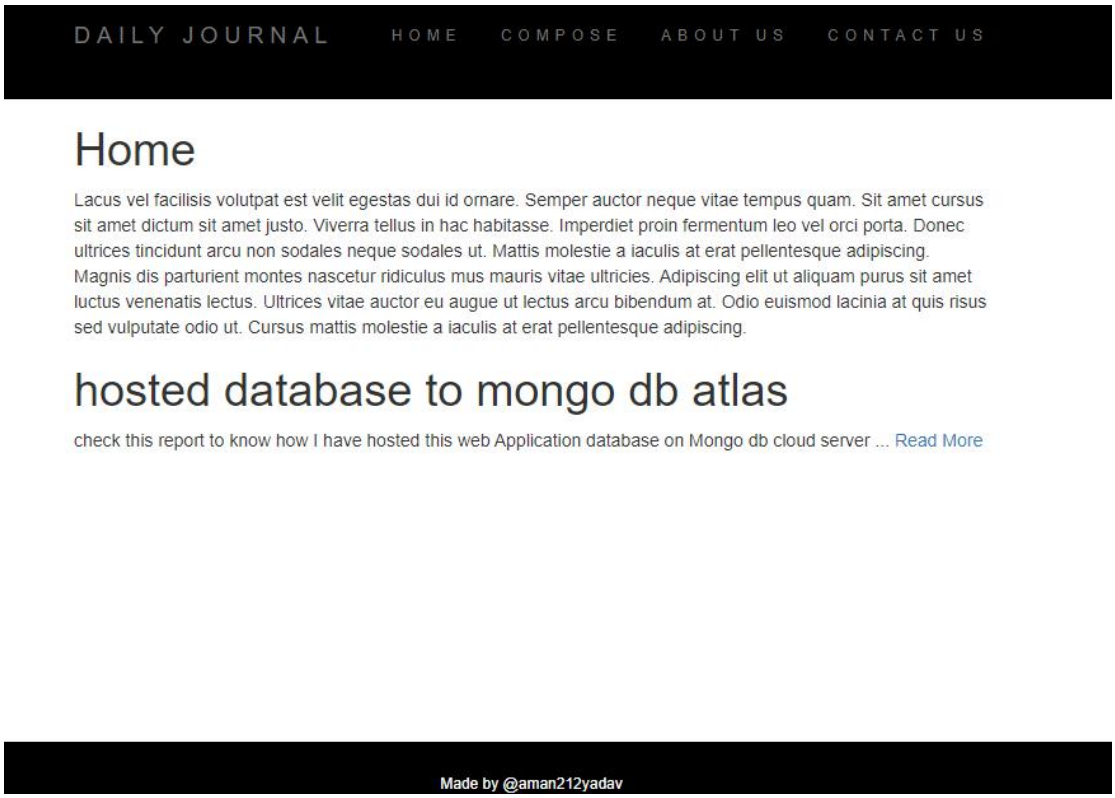


Step 7->Adding the generated link to the web application

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

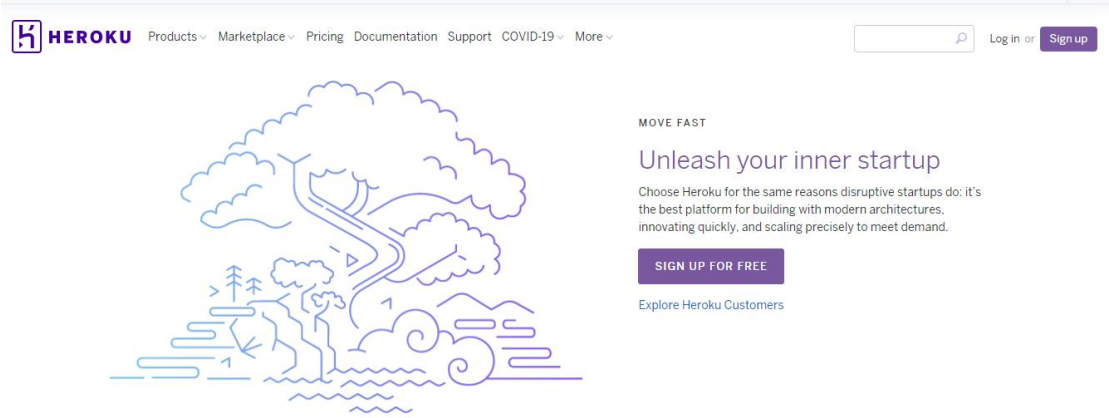
(hiding some information for security reasons)

Step 8->Web application is now able to communicate with database hosted on cloud (ATLAS).

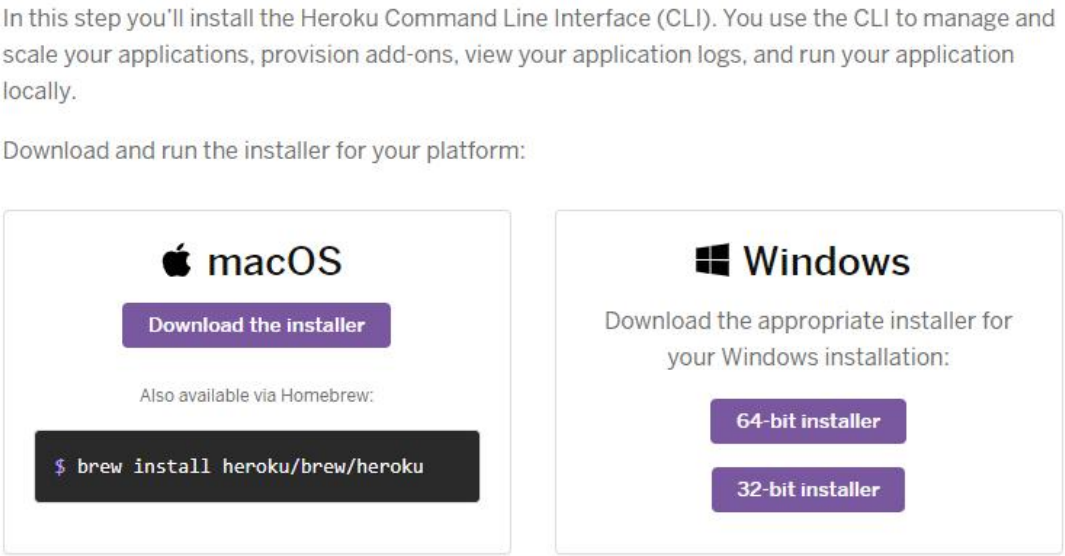


Deploying the Blog writing web application on HEROKU cloud service.

Step 1->Creating a Heroku account.

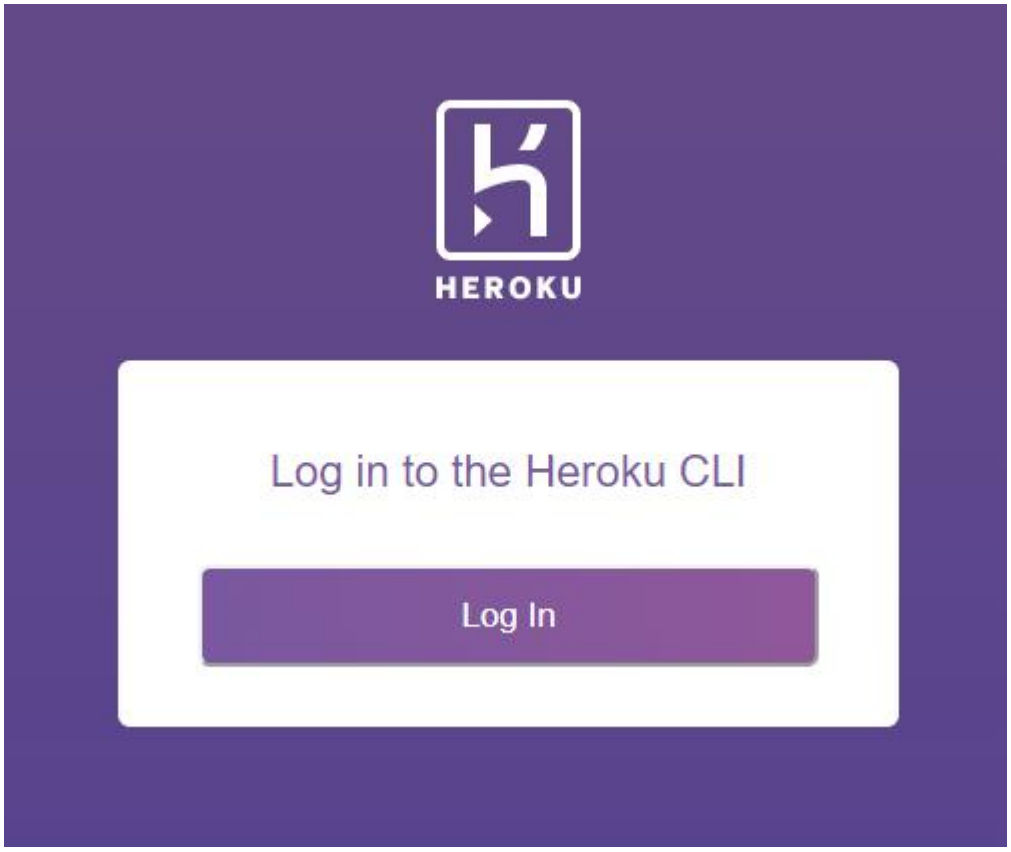


Step 2->Installing Heroku CLI (command line interface).



Step 3->Login to heroku by 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 an app 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-41117.git
```

Step 5->Defining a **procfile** to explicitly declare what command heroku should use to start the app.

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

Step 6 -> Defining **port** on which app 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 it **package.json** file

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

app.js | package.json | Procfile

```
{
  "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"
  }
}
```

Step 7 ->Initialising **git repository** .

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

Step 8 ->Adding **.gitignore** file to avoid uploading unnecessary files/folder.

app.js | package.json | .gitignore | Procfile

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

Step 9 ->Adding all files to **staging area** and **commiting** 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
```

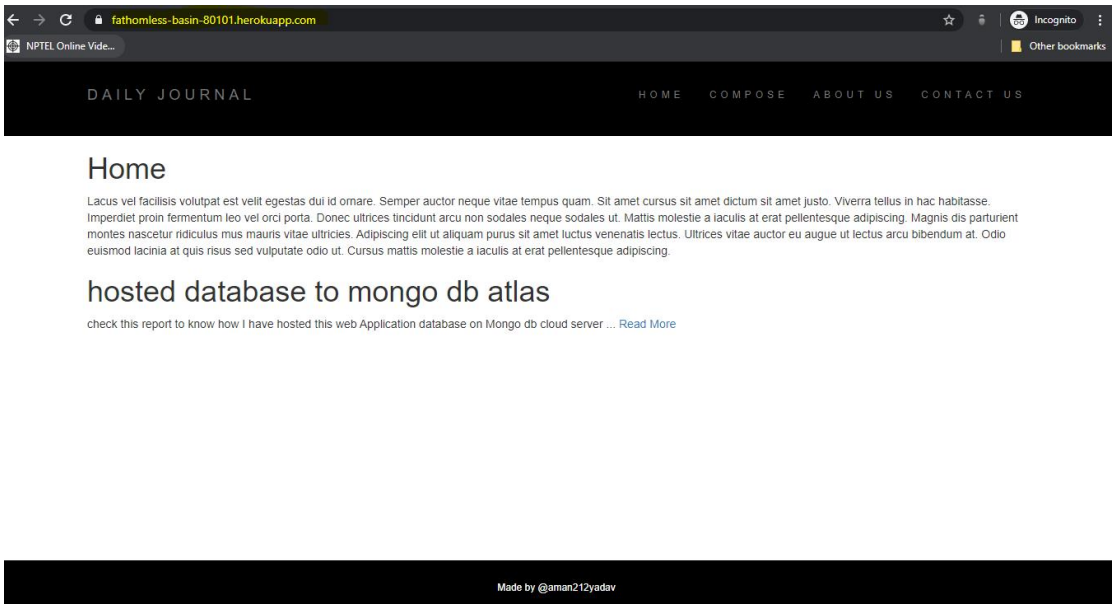
Step 10 ->Pushing all changes to remote repository heroku master.

```
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 11 ->Application deployed 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
```

Got a publically accessible URL.



Conclusion -> both database and web application hosted successfully on the cloud web services (ATLAS, Heroku).

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