

**Name** - Jay prakash kumar  
**RollNo** - 171210030 (CSE 3rdyear)  
**Subject** - Networkprogramming

### **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 mongodbasits 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 overthere.

B) how I have deployeddatabaseof **Blog writing web application** on mongodb cloud server (**ATLAS**).

C)how I have deployed web Application on **HEROKU** cloud webservice.

### **End results-**

**Publically accessible URL :**

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

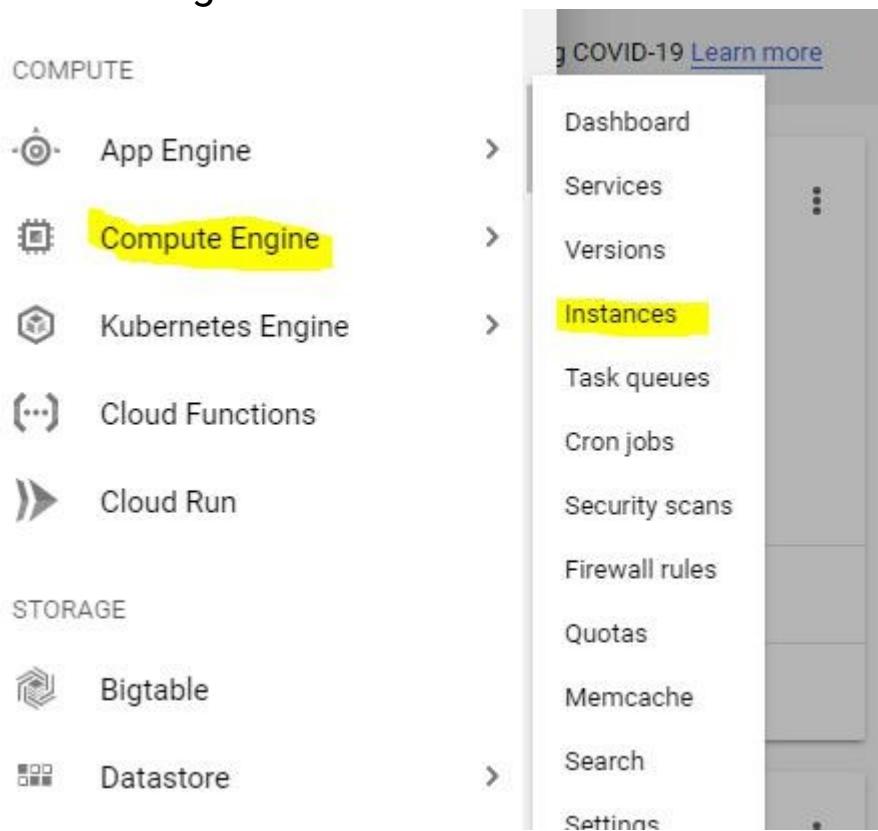
**GitHub link for code :**

<https://github.com/jayprakashkumar123/assignment-3>

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

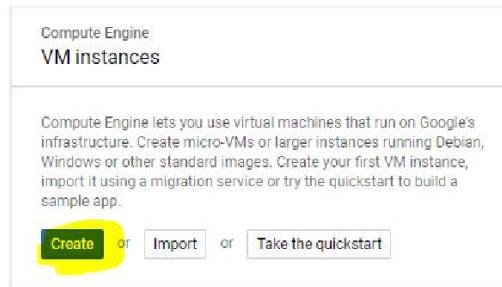
**Step1-> create google account.**

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 instances



< Create an instance

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

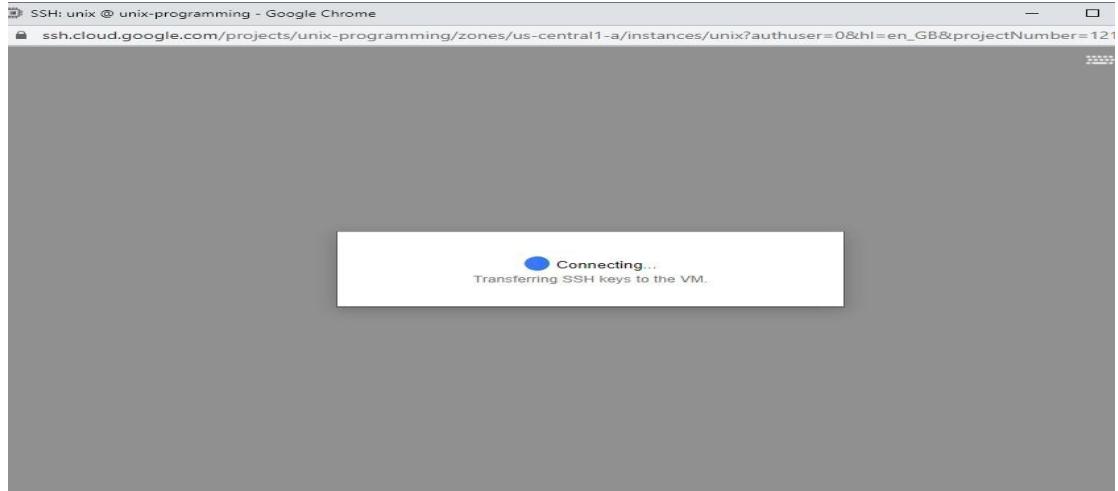
- New VM instance** Create a single VM instance from scratch
- New VM instance from template** Create a single VM instance from an existing template
- New VM instance from machine image** Create a single VM instance from an existing machine image
- Marketplace** Deploy a ready-to-go solution onto a VM instance

**Create** or **Import** or **Take the quickstart**

VM instance is ready

Filter VM instances		Columns ▾				
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



## Running VM

```
ssh.cloud.google.com/projects/unix-programming/zones/us-central1-a/instances/unix?authuser=0&hl=en_GB&projectNumber=121

Connected, host fingerprint: ssh-rsa 0 5E:8D:36:ED:31:EC:99:8A:B7:CE: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.

aman212yadav@unix:~$ ls
aman212yadav@unix:~$ pwd
/home/aman212yadav
aman212yadav@unix:~$ cd ..
aman212yadav@unix:/home$
```

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+0xfid): 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/jayprakashkumar123/assignment-3>

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

Step 1-> creating a mongoDB atlas account.

MongoDB.live, free & fully virtual | June 9th - 10th      [Register Now >](#)

mongoDB Cloud Software Learn Solutions Docs      [Contact](#) [Sign In](#) [Try Free](#)

## MongoDB Atlas

Move faster with a cloud MongoDB service. Built for agile teams who'd rather spend time building apps than managing databases. Available on AWS, Azure, and GCP.

[Start free](#)

Already have an account? [Log in here →](#)

Cloud Provider & Region  
Choose your preferred cloud provider and the region nearest to clients.

AWS, N. Virginia (us-east-1) >

Select a cloud provider to see its region availability:

AWS       Google Cloud Platform       Azure

Configure a free tier cluster by first selecting a region labeled with [FREE FOR AVAILABLE](#), then choose the M0 option in the Cluster Tier below.

● recommended region: ( )

Region	Availability
N. Virginia (us-east-1)	<span style="color: green;">FREE FOR AVAILABLE</span>
Otta (eu-west-1)	*
N. California (us-west-1)	*
Oregon (us-west-2)	*
Montreal (ca-central-1)	*
Ireland (eu-west-1)	*
London (eu-west-2)	
Frankfurt (eu-central-1)	<span style="color: blue;">FREE FOR AVAILABLE</span>
São Paulo (sa-east-1)	
Tokyo (ap-northeast-1)	
Seoul (ap-northeast-2)	
Singapore (ap-southeast-1)	
Mumbai (ap-south-1)	

## Step 2-> creating a new cluster

The screenshot shows the MongoDB Atlas interface. At the top, there are navigation links for 'Atlas' (which is highlighted with a green underline), 'Stitch', and 'Charts'. On the right side, there are icons for user profile, export, and notifications. Below the navigation, a header bar displays 'AMAN'S ORG - 2020-05-06 > PROJECT 0'. A prominent green button on the right says 'Build a New Cluster'. In the center, there's a search bar with the placeholder 'Find a cluster...'. Below the search bar, a modal window is open, showing a 'MongoDB Atlas Search' icon and the text: 'MongoDB Atlas Search makes it easy to build simple, integrated search capabilities on top of your data in the cloud.' A green button in the modal says 'Try the Beta Now'. At the bottom left of the main area, there's a blue button labeled 'SANDBOX'.

## Step 3-> selecting cloud provider.

This screenshot shows the 'Cloud Provider & Region' selection step. At the top, it says 'Cloud Provider & Region' and 'AWS, N. Virginia (us-east-1)'. Below this, there are three options: 'aws' (selected and highlighted with a yellow arrow), 'Google CloudPlatform', and 'Azure'. A large yellow arrow points to the 'aws' button. Below the providers, a section titled 'Select Multi-Region, Workload Isolation, and Replication Options (M10+ clusters)' includes a note about increasing region availability and a 'Read more' link. To the right of this section is a radio button labeled 'NO'. Further down, there's a note about creating a 'free tier cluster' by selecting a region with 'FREE TIER AVAILABLE' and choosing the 'M0' cluster tier. It also mentions '★ Recommended region'. At the bottom, there are three tabs for 'NORTH AMERICA', 'EUROPE', and 'ASIA'. Under 'NORTH AMERICA', 'N. Virginia (us-east-1)' is selected and highlighted with a yellow box and a green 'FREE TIER AVAILABLE' button. Other options shown are 'Stockholm (eu-north-1)' and 'Hong Kong (ap-east-1)'.

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

Edit User: admin-aman@admin

### Password Authentication

MongoDB uses **SCRAM** as its default authentication method.

admin-aman

.....

SHOW

Autogenerate Secure Password

Copy

### Database User Privileges

Atlas admin

Read and write to any database

Only read any database

Select Custom Role

Add Default Privileges

Cancel

Update User

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

AMAN'S ORG - 2020-05-06 > PROJECT 0

### Network Access

IP Whitelist

Peering

Private Endpoint



Whitelist an IP address

Configure which IP addresses can access your cluster.

Add IP Address

## Add IP Whitelist Entry

Atlas only allows client connections to a cluster from entries in the project's whitelist. Each entry should either be a single IP address or a CIDR-notated range of addresses. [Learn more.](#)

[ADD CURRENT IP ADDRESS](#) [ALLOW ACCESS FROM ANYWHERE](#)

Whitelist Entry:

0.0.0.0/0

Comment:

Optional comment describing this entry



This entry is temporary and will be deleted in

6 hours ▾

[Cancel](#)

[Confirm](#)

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

### Connect to Cluster0

[✓ Setup connection security](#) [Choose a connection method](#) [Connect](#)

[Choose a connection method](#) [View documentation](#)

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



[Go Back](#)

[Close](#)

## Connect to Cluster0

The screenshot shows the MongoDB connection setup interface. At the top, there are three green checkmarks: 'Setup connection security', 'Choose a connection method', and a final 'Connect' button. Below this, step 1 is titled 'Select your driver and version', with dropdown menus for 'DRIVER' (set to 'Node.js') and 'VERSION' (set to '3.0 or later'). Step 2 is titled 'Add your connection string into your application code'. It provides two options: 'Connection String Only' (selected) and 'Full Driver Example'. A code editor window displays a connection string: 'mongodb+srv://admin-aman:<password>@cluster0-ydwhd.mongodb.net/test?r'. A 'Copy' button is available to the right of the code. Below the code editor, a note says: 'Replace <password> with the password for the user, admin-aman, and ensure all special characters are URL encoded.'

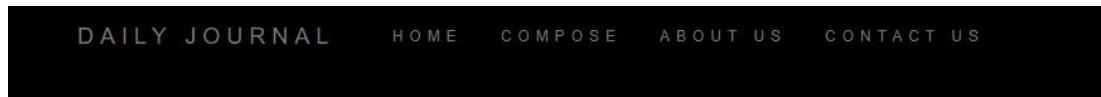
Having trouble connecting? [View our troubleshooting documentation](#)

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

```
25 mongoose.connect('mongodb+srv://admin-aman:<password>@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).**



## 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.

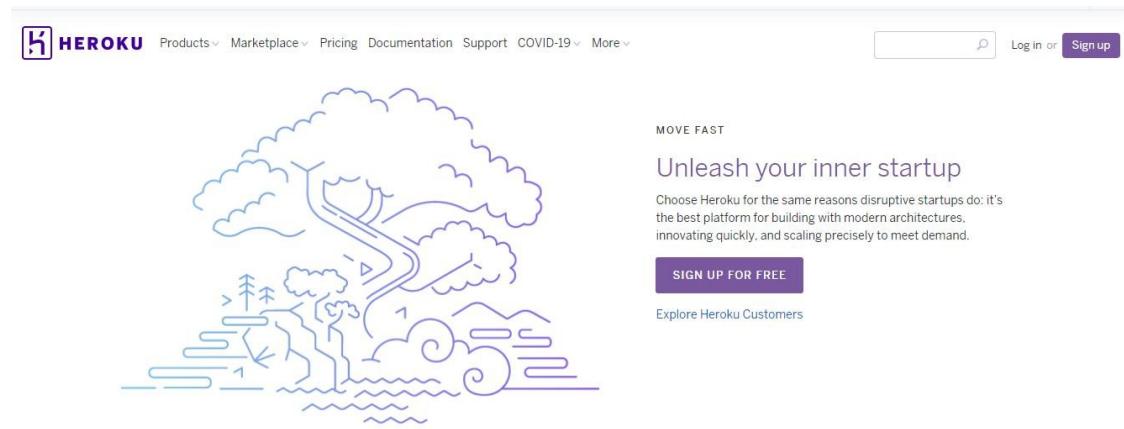
## hosted database to mongo db atlas

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

Made by [@aman212yadav](#)

# Deploying the Blog writing web application on HEROKU cloud service.

## Step 1->Creating a Heroku account.



## 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:

macOS

[Download the installer](#)

Also available via Homebrew:

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

Windows

Download the appropriate installer for your Windows installation:

[64-bit installer](#)

[32-bit installer](#)

Step 3->Login to heroku by using heroku login command.

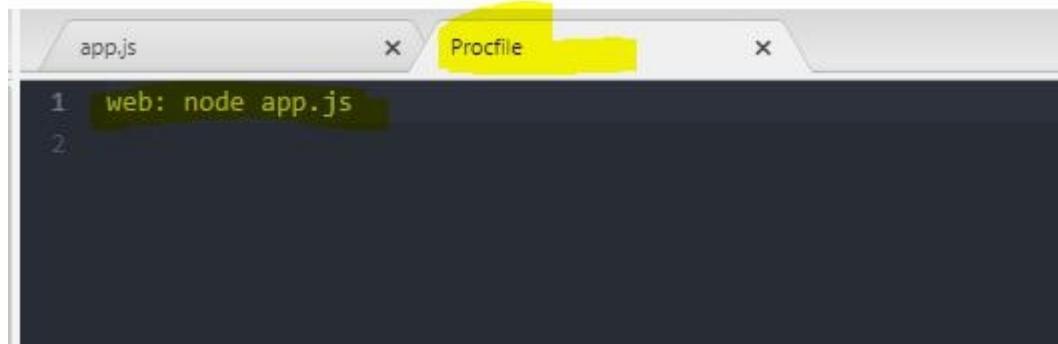
```
C:\Users\aman2>heroku login
» Warning: heroku update available from 7.35.1 to 7.40.0.
heroku: Press any key to open up the browser to login or q to exit:
```



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.



```
app.js x Procfile x
1 web: node app.js
2
```

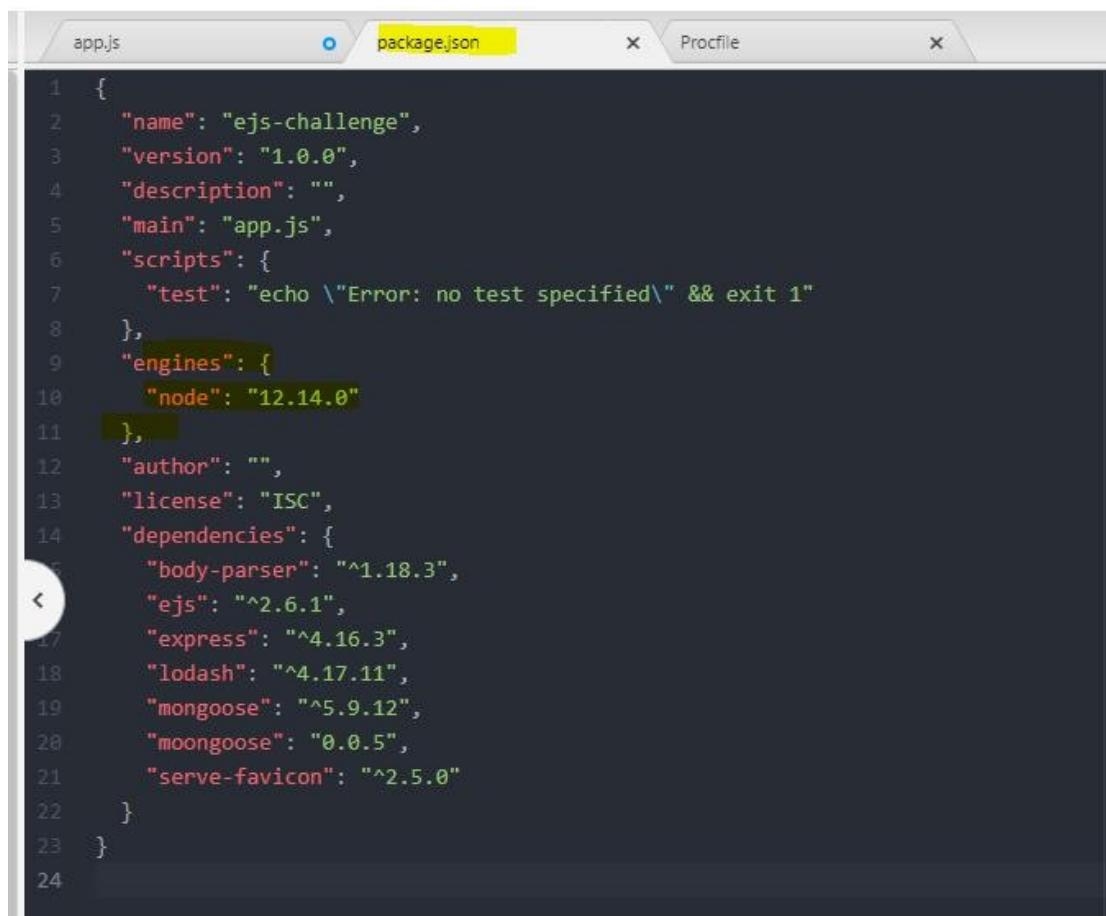
Step 6 -> Defining **port** on which app should listen/start

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

```
88 app.listen(PORT, function() {
89   console.log(`Server started on port ${PORT}`);
90 });
91
```

## Step 7 -> Finding version of node and adding it package.json file

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



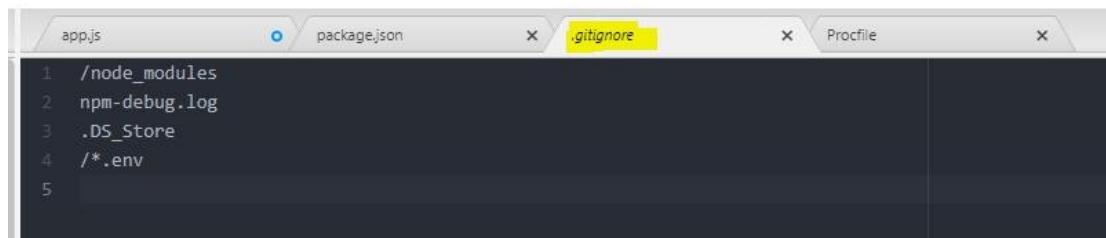
The screenshot shows a code editor window with three tabs: 'app.js' (disabled), 'package.json' (selected), and 'Procfile'. The 'package.json' tab contains the following JSON code:

```
1  {  
2    "name": "ejs-challenge",  
3    "version": "1.0.0",  
4    "description": "",  
5    "main": "app.js",  
6    "scripts": {  
7      "test": "echo \\"Error: no test specified\\" && exit 1"  
8    },  
9    "engines": {  
10      "node": "12.14.0"  
11    },  
12    "author": "",  
13    "license": "ISC",  
14    "dependencies": {  
15      "body-parser": "^1.18.3",  
16      "ejs": "^2.6.1",  
17      "express": "^4.16.3",  
18      "lodash": "^4.17.11",  
19      "mongoose": "^5.9.12",  
20      "mongoose": "0.0.5",  
21      "serve-favicon": "^2.5.0"  
22    }  
23  }  
24
```

## Step 7 ->Initialising git repository .

```
c:\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.



## Step 9 ->Adding all files to staging area and commiting all changes.

```
c:\Users\aman2\Desktop\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
```

```
c:\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
```

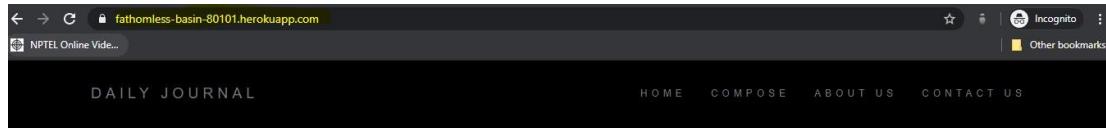
## Step 10 ->Pushing all changes to remote repository heroku master.

```
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/>