

**Indian Institute of Information Technology,
Guwahati**

B Tech Computer Science

Software Engineering Project

Ambulance Tracker App

Group 5:

Atul Prakash (1801038)
Chayan Gurele (1801045)
Namit Goel (1801109)

Link to the project - <https://github.com/chayan101/AmbulanceTrackerApp>

Testing

The type of testing used in this project is **Black box Testing**.

It is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure, implementation details and internal paths. Black Box Testing mainly focuses on input and output of software applications and it is entirely based on software requirements and specifications.



Functional Requirements-

1. Users must have a valid User ID and password to login thus creating their individual profiles

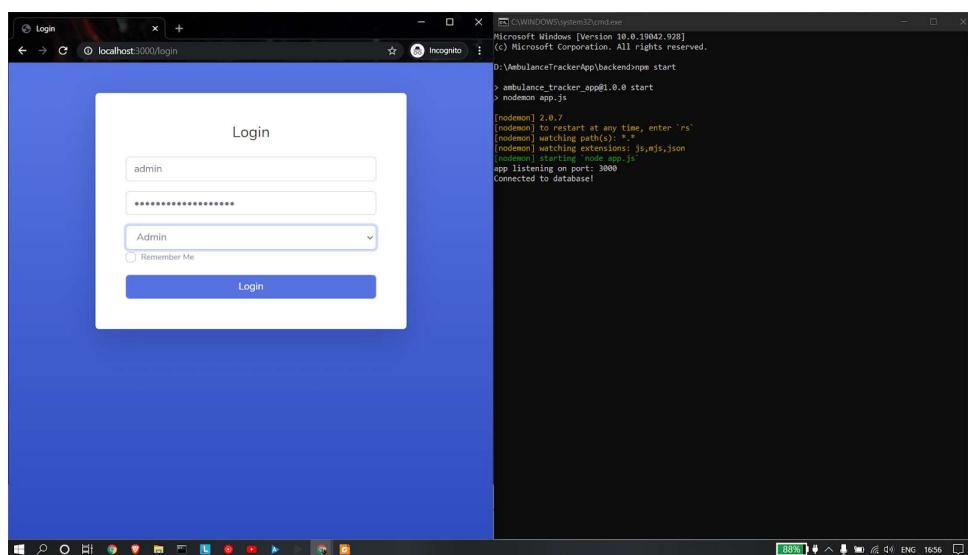
Login for Admin-

Case 1: Successful login of the admin

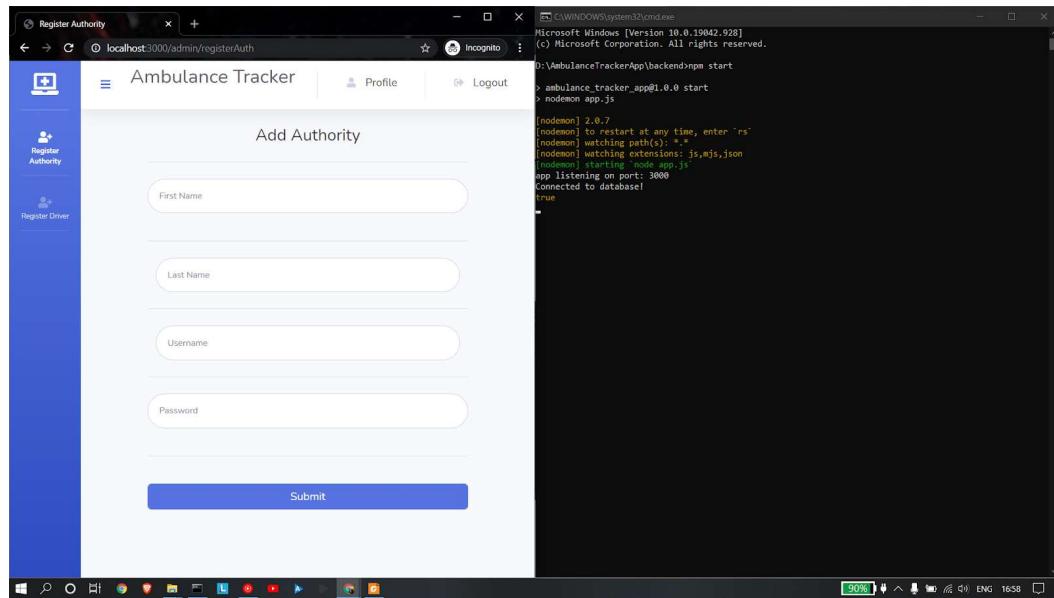
Username : admin

Expected o/p : Successful Login

Output :



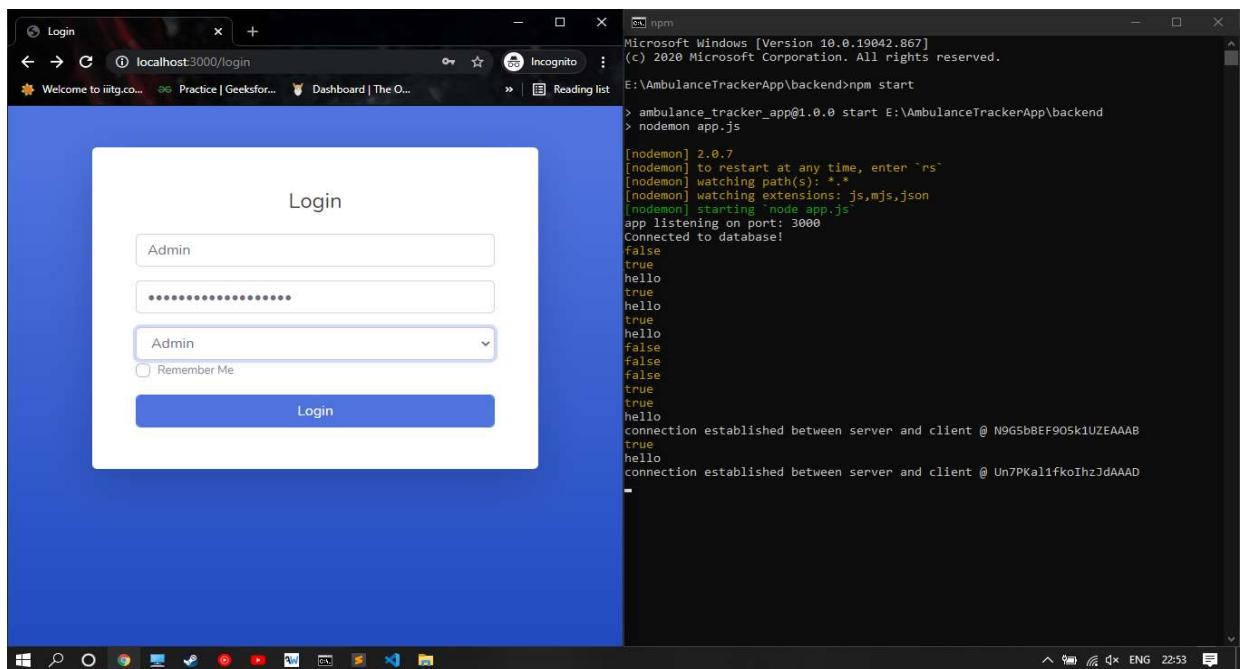
Successful Login :



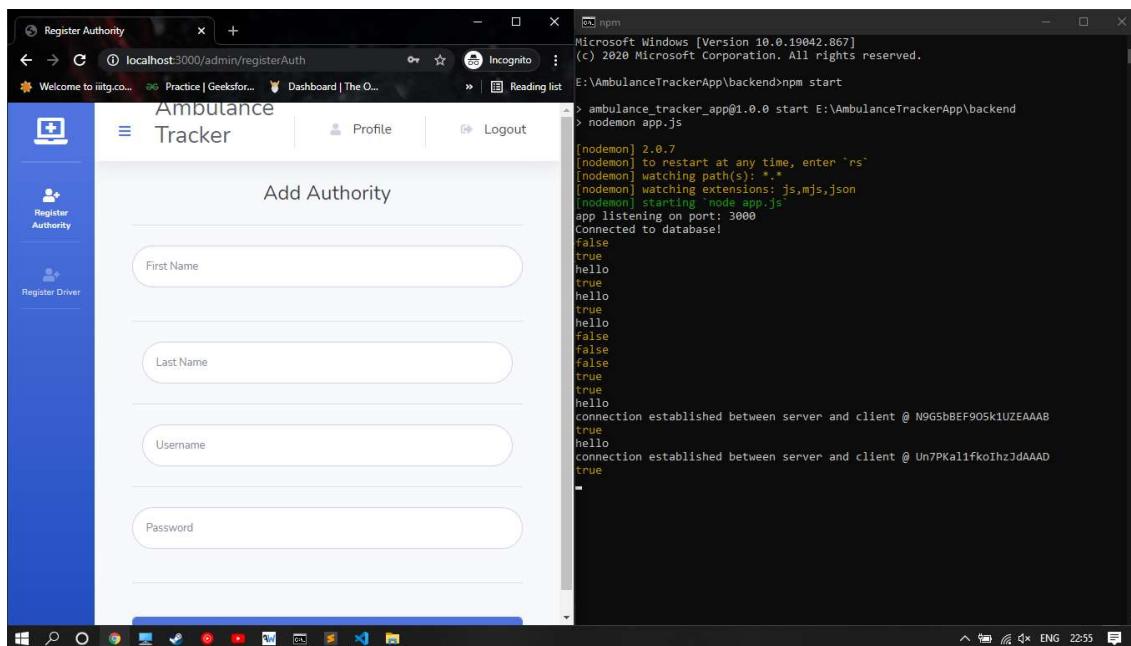
Username : Admin

Expected o/p : Successful Login

Output :



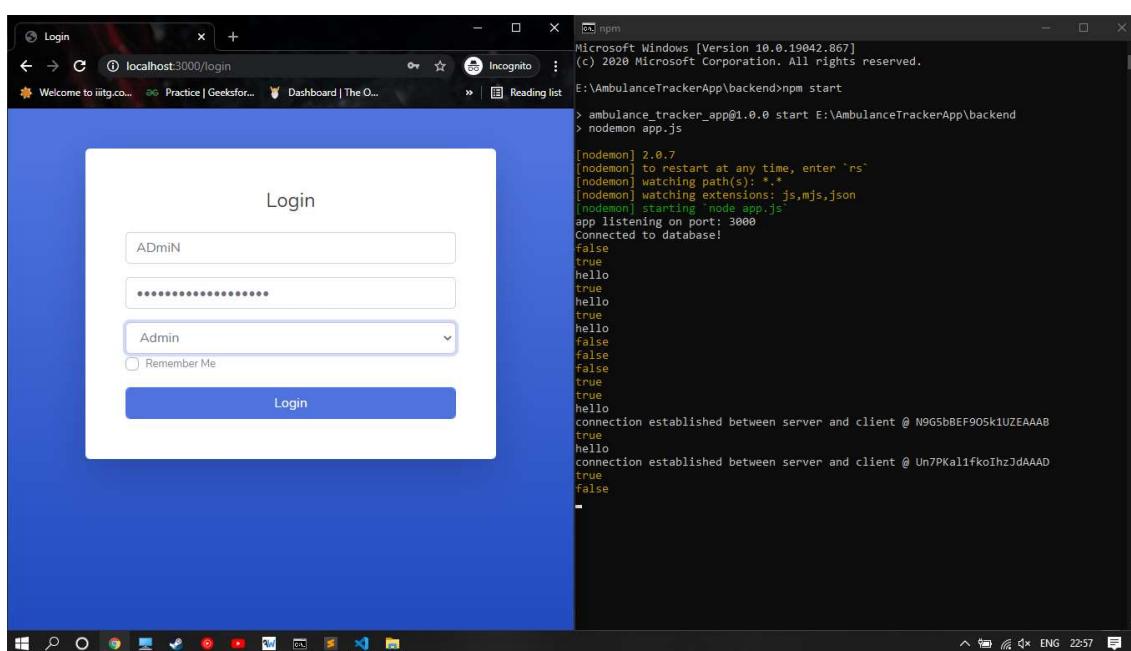
Successful Login :



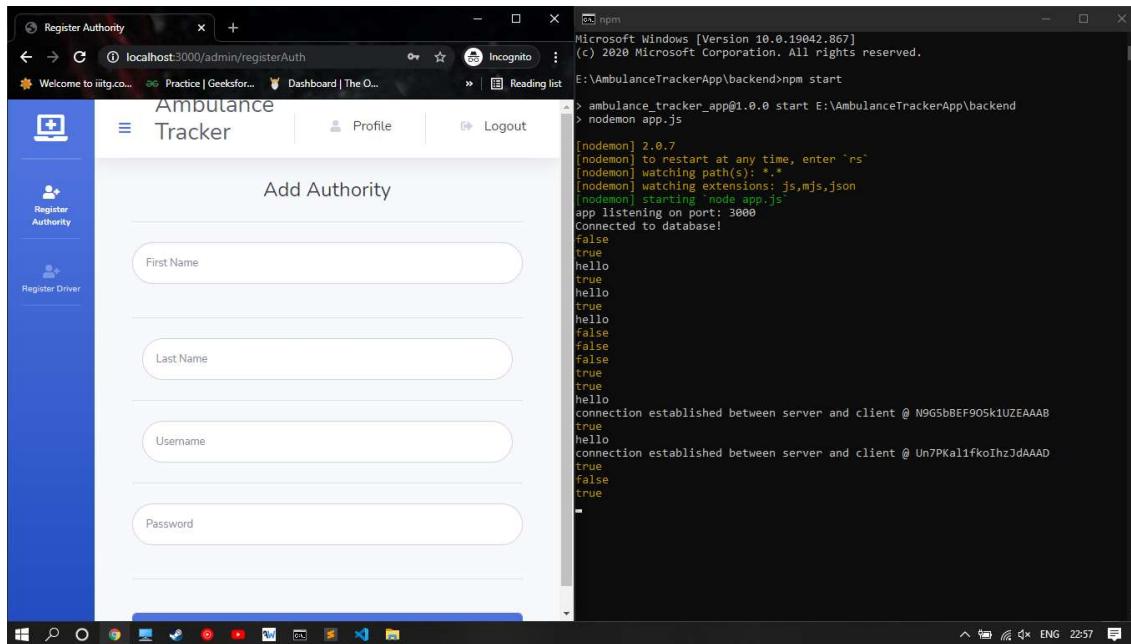
Username : ADmiN

Expected o/p : Successful Login

Output :



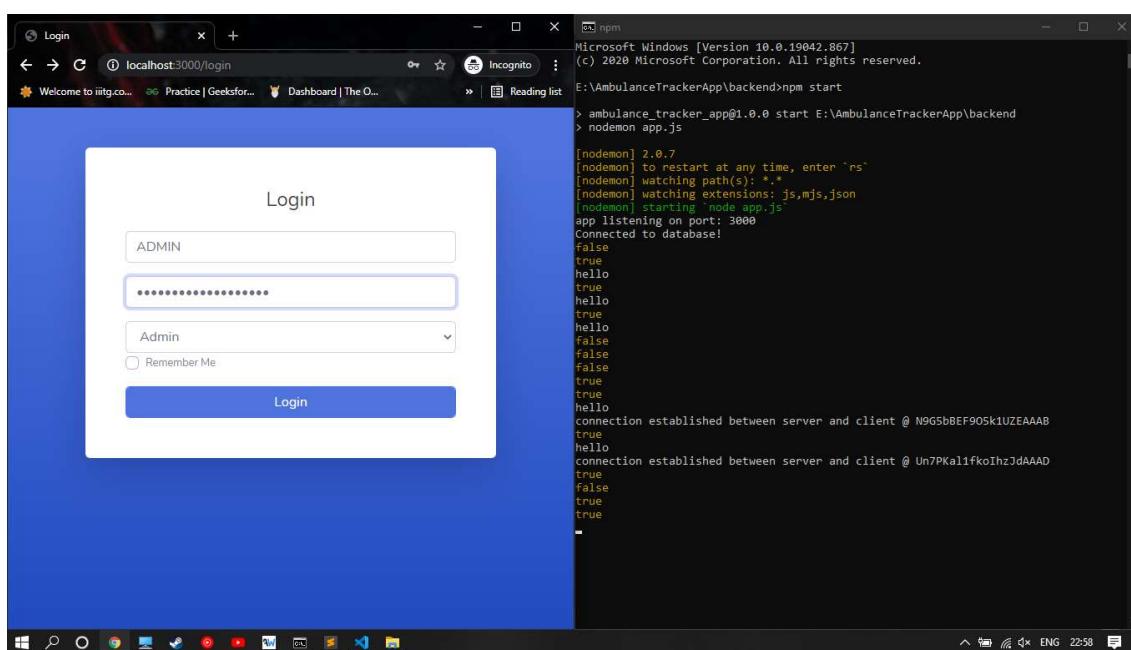
Successful Login :



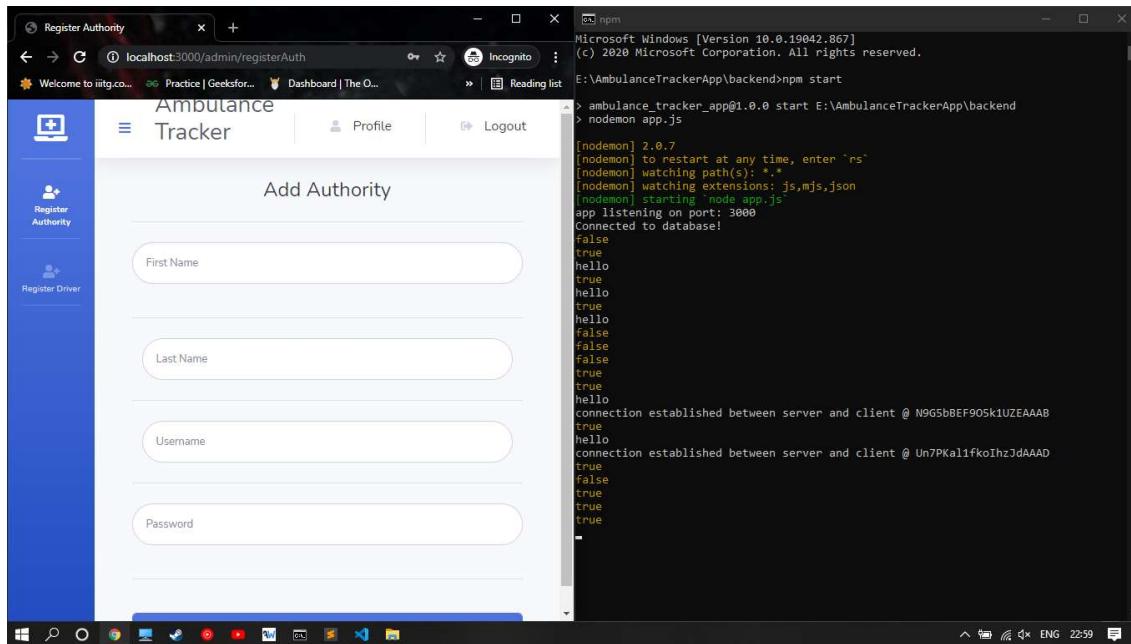
Username : ADMIN

Expected o/p : Successful Login

Output :



Successful Login :

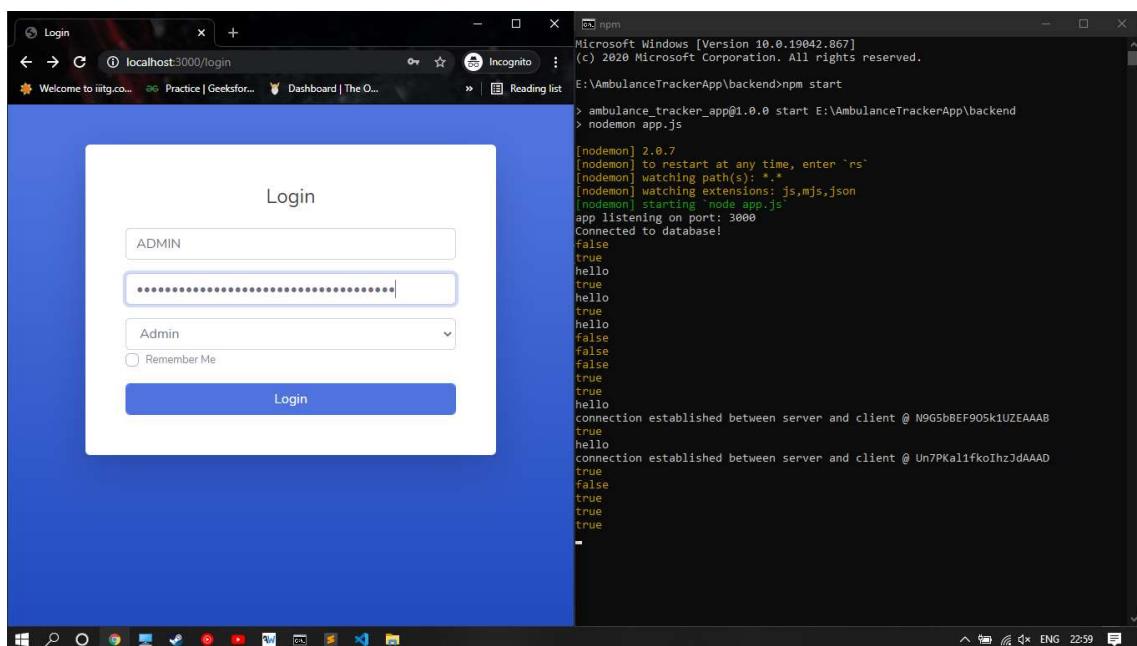


Case 2: Correct username, wrong password

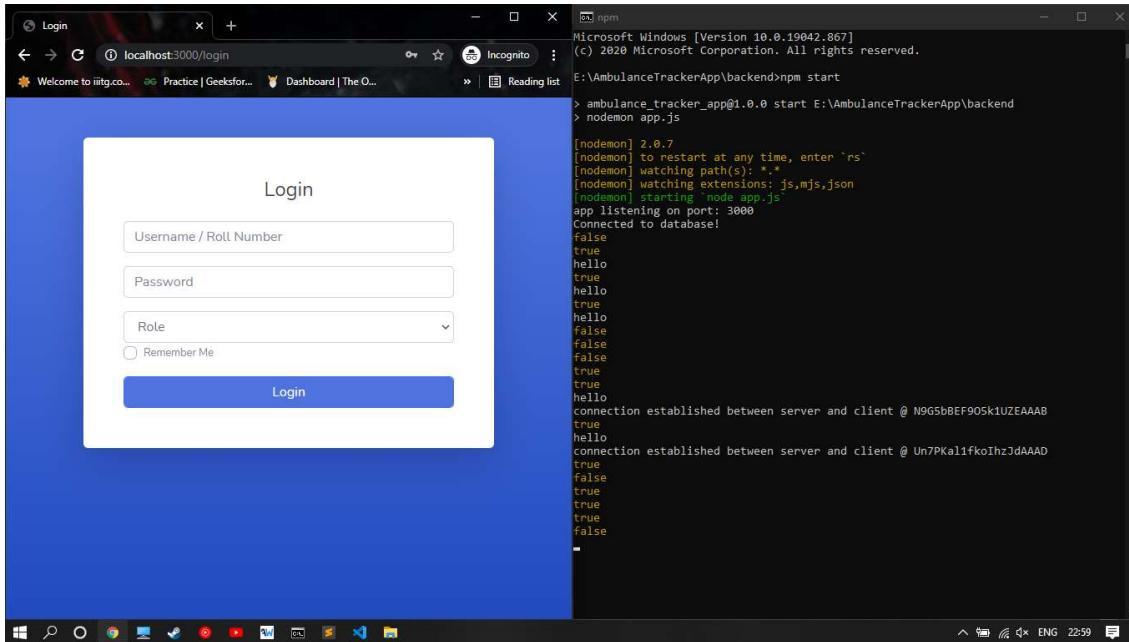
Username : ADMIN

Expected o/p : No Login

Output :



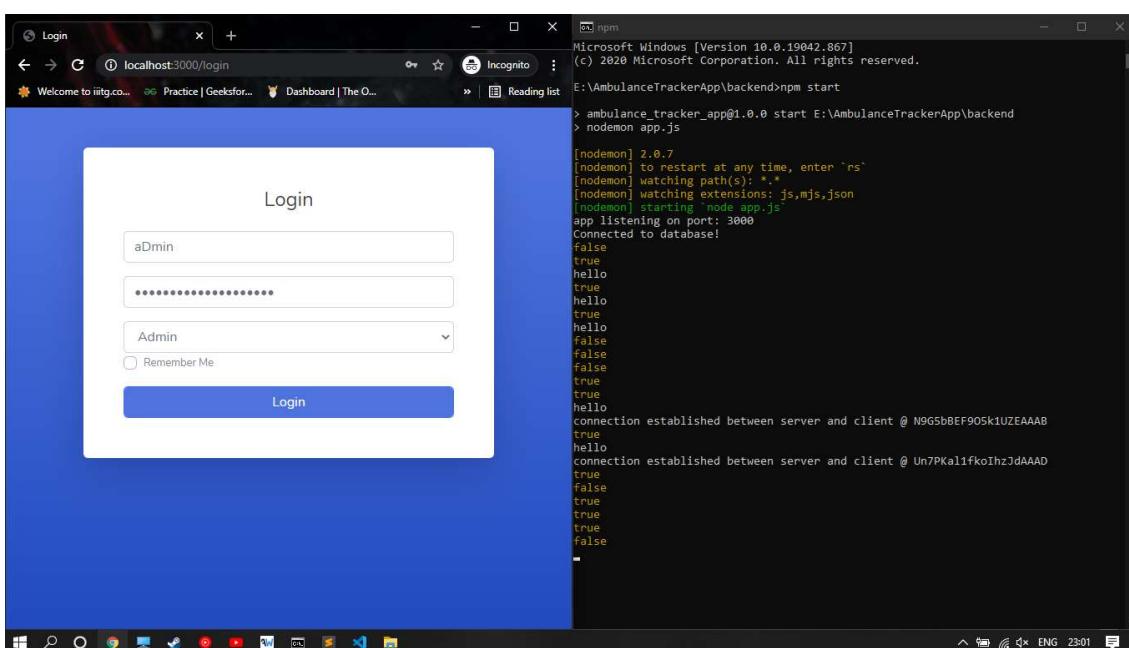
No Login :



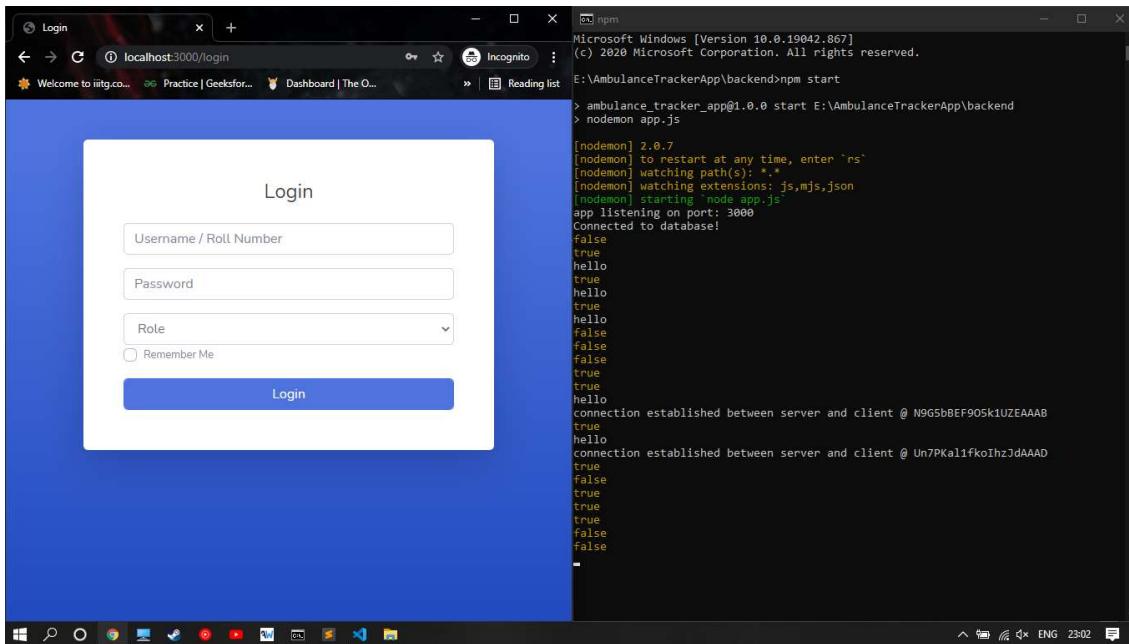
Username : aDmin

Expected o/p : No Login

Output :



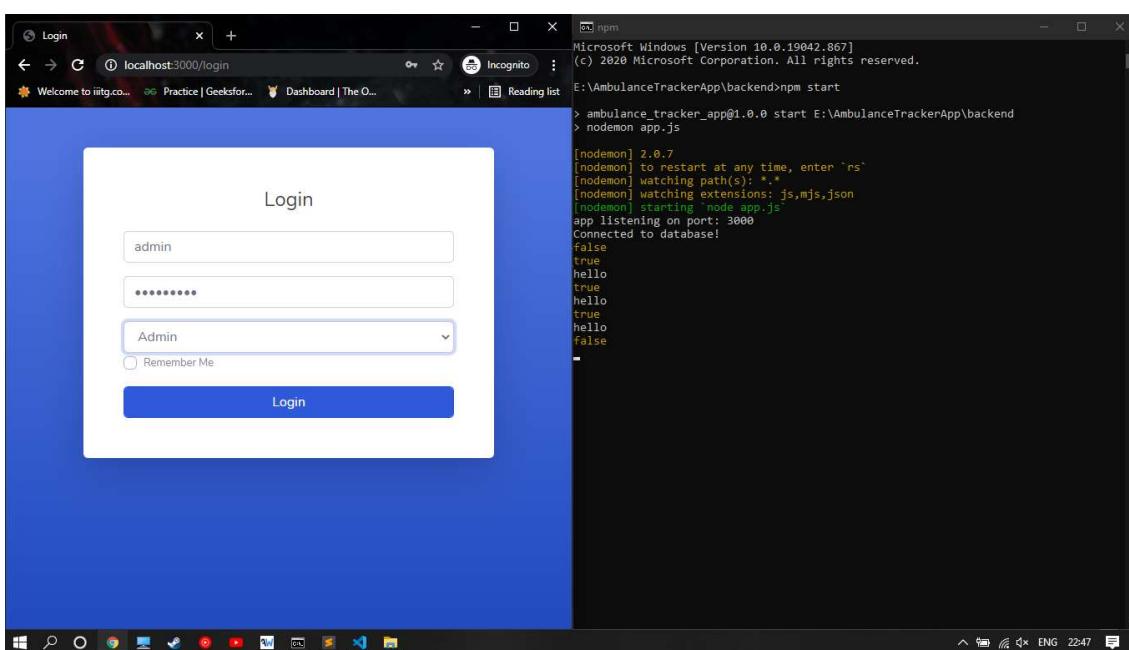
No Login :



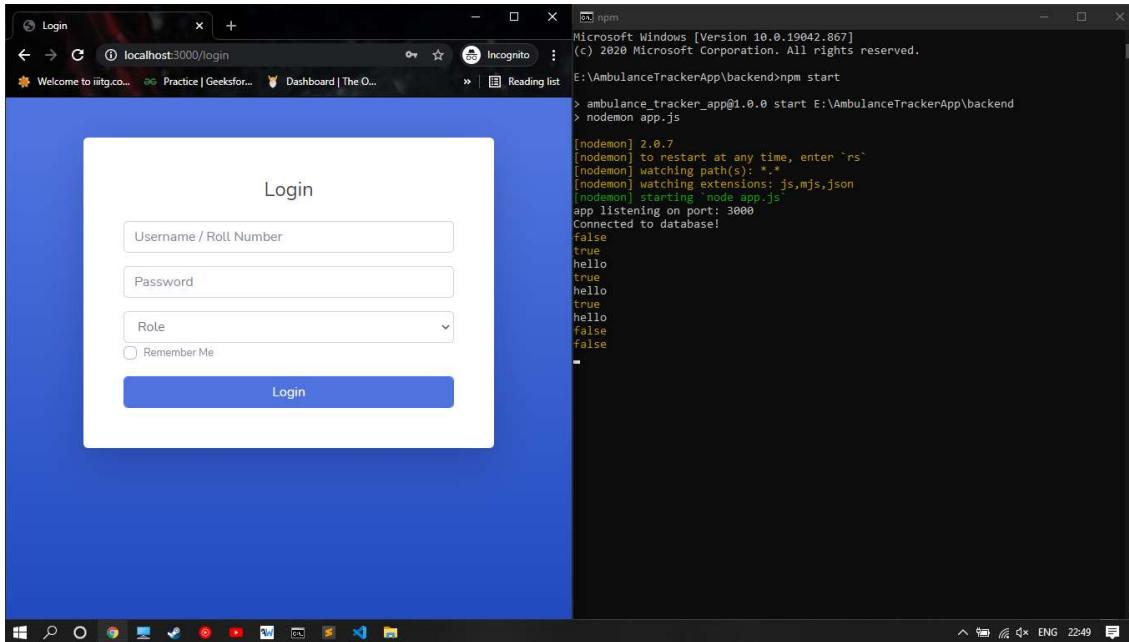
Username : admin

Expected o/p : No Login

Output :



No Login :



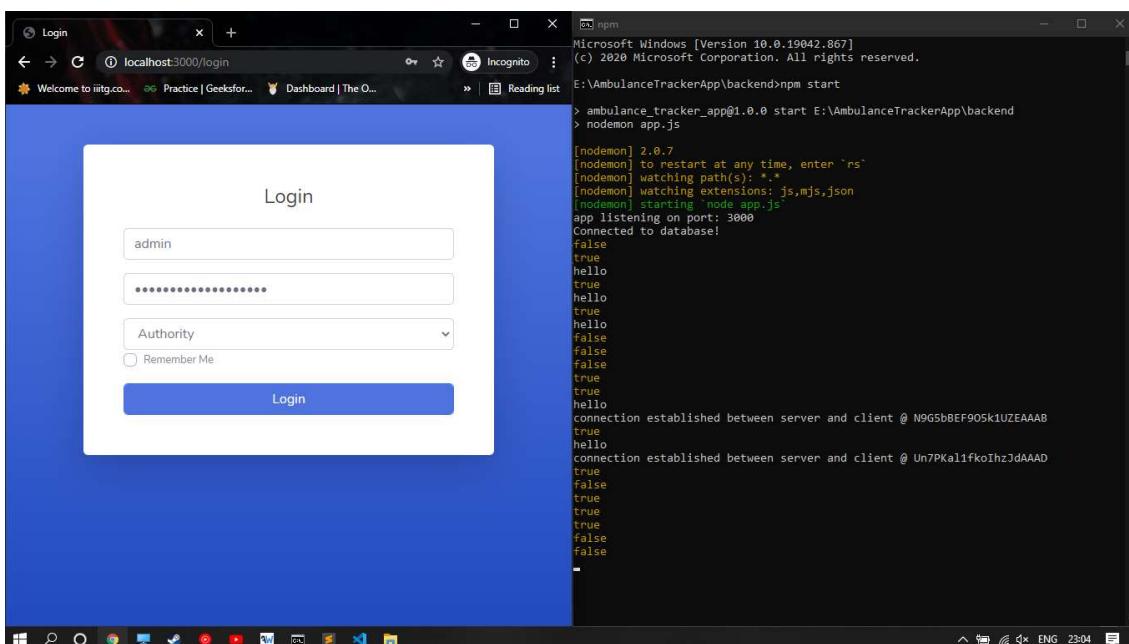
Case 3: Correct username, correct password, incorrect role

Username : admin

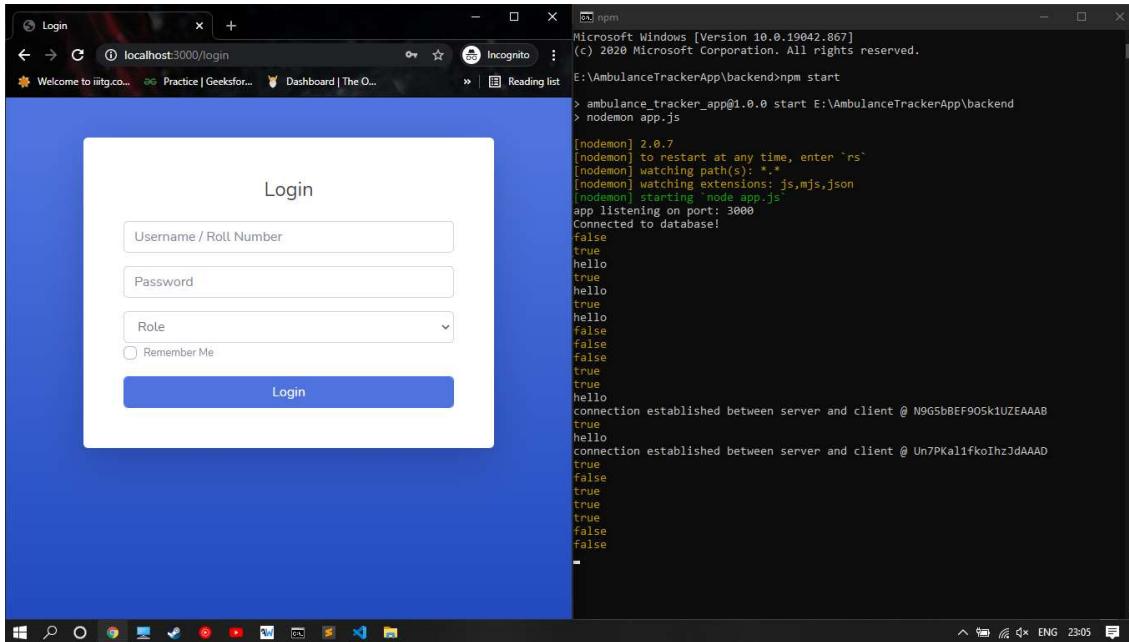
Role : Authority

Expected o/p : No Login

Output :



No Login :

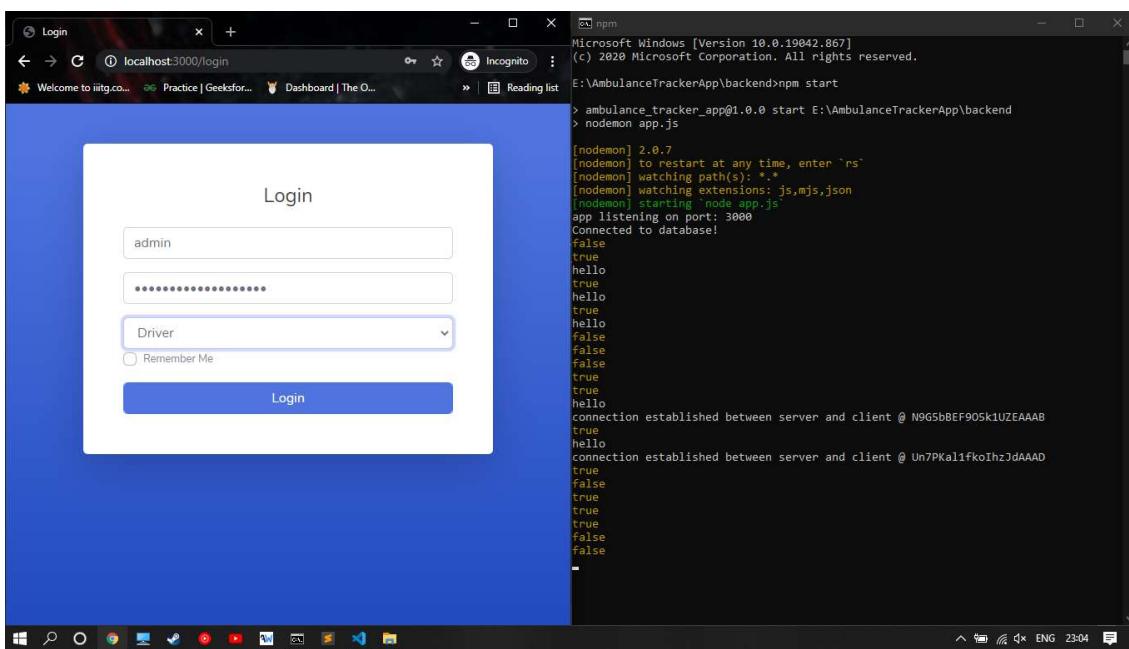


Username : admin

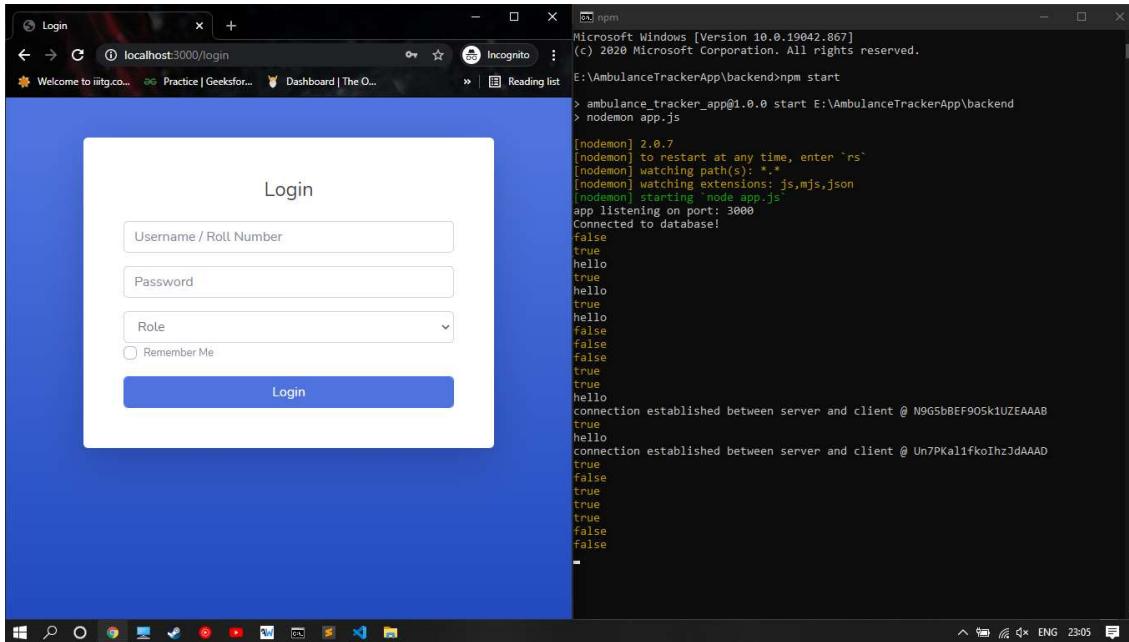
Role : Driver

Expected o/p : No Login

Output :



No Login :

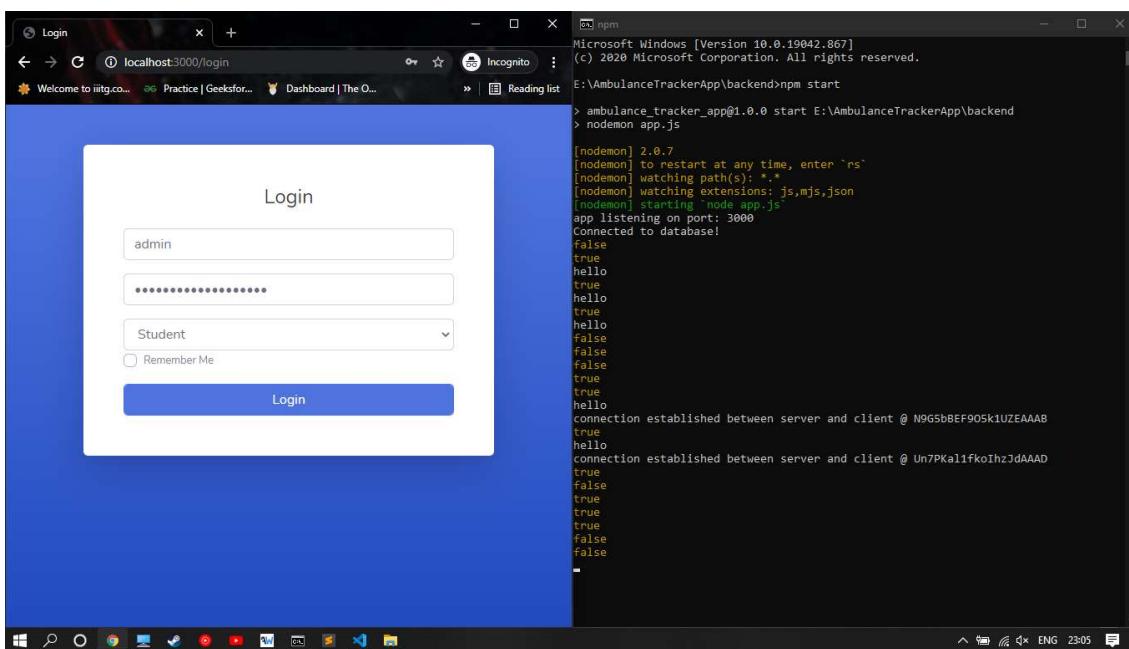


Username : admin

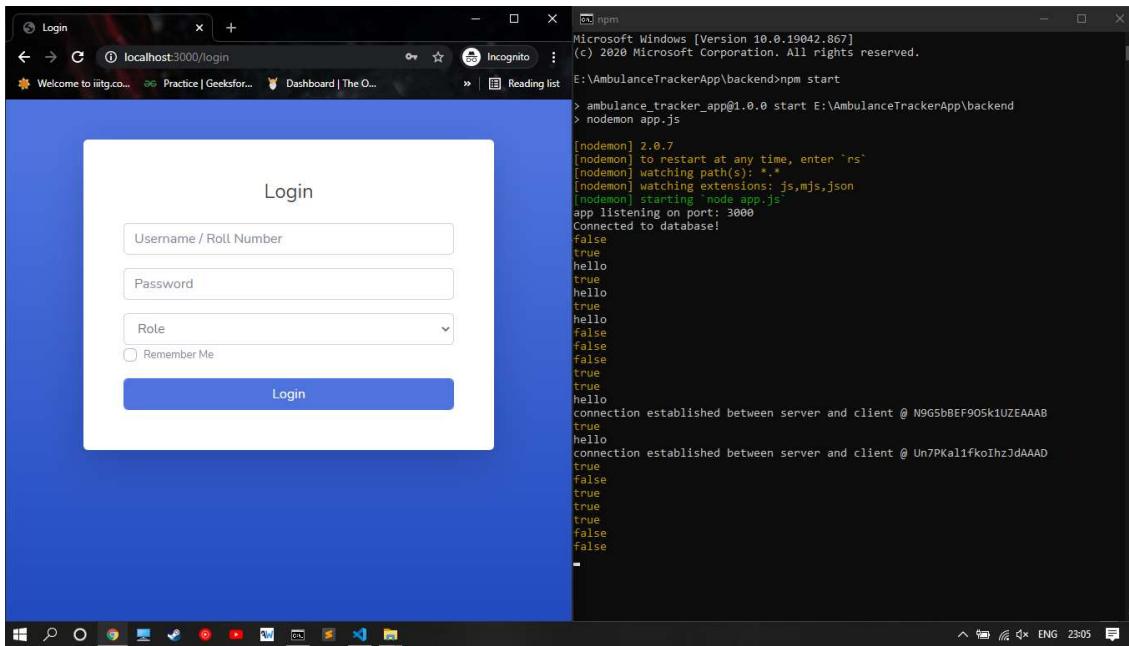
Role : Student

Expected o/p : No Login

Output :



No Login :



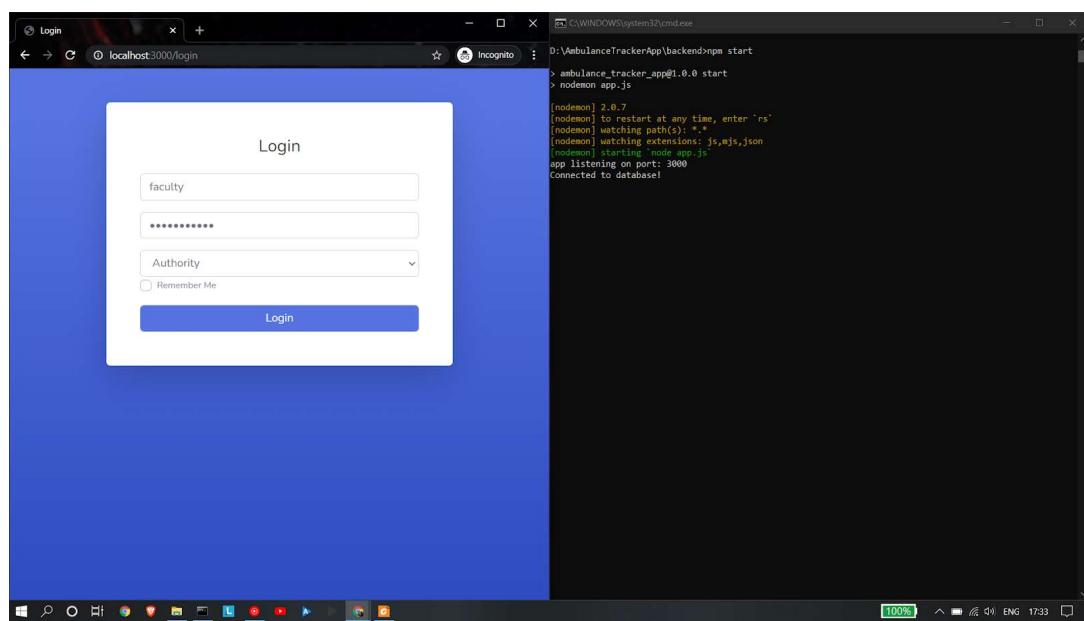
Login for Authority-

Case 1: Successful login of the authority

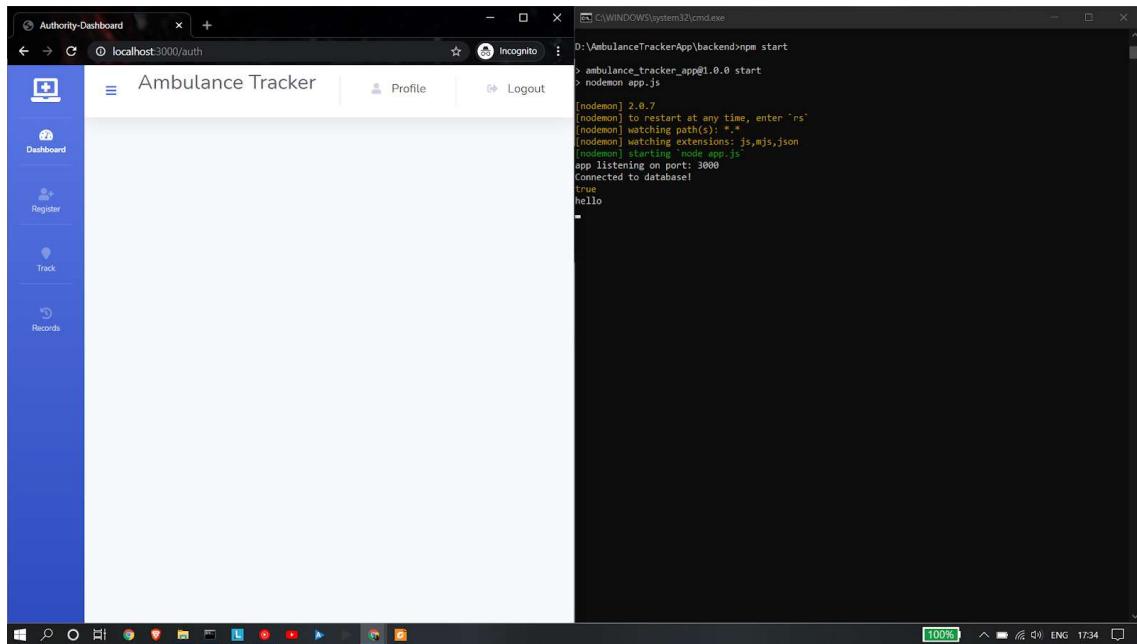
Username : faculty

Expected o/p : Successful Login

Output :



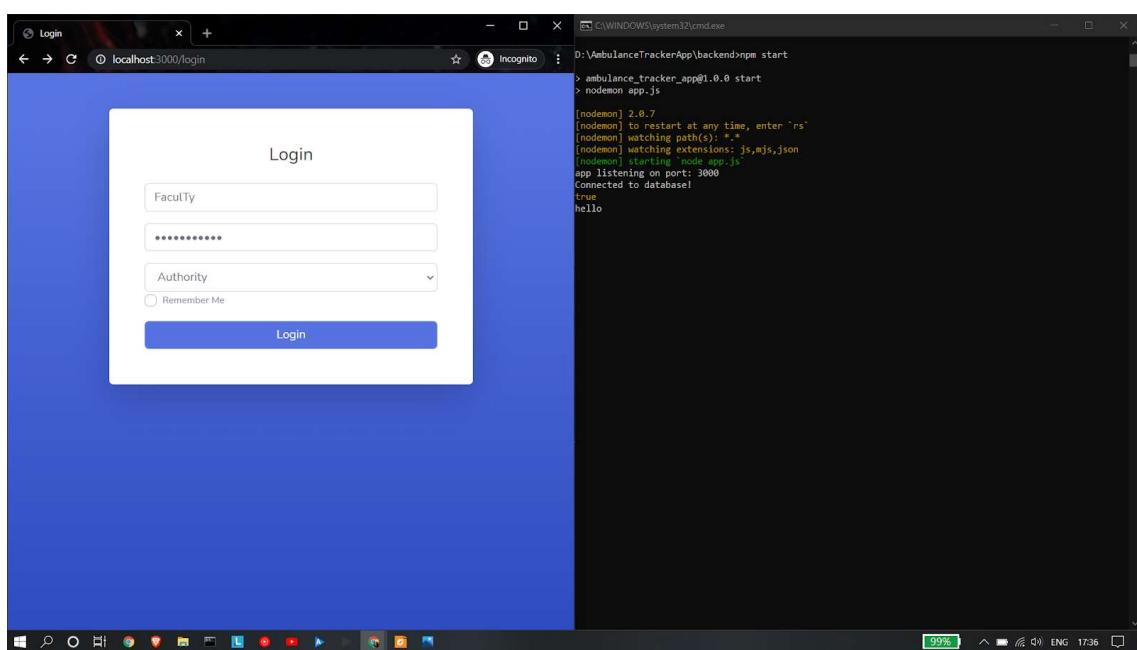
Successful Login :



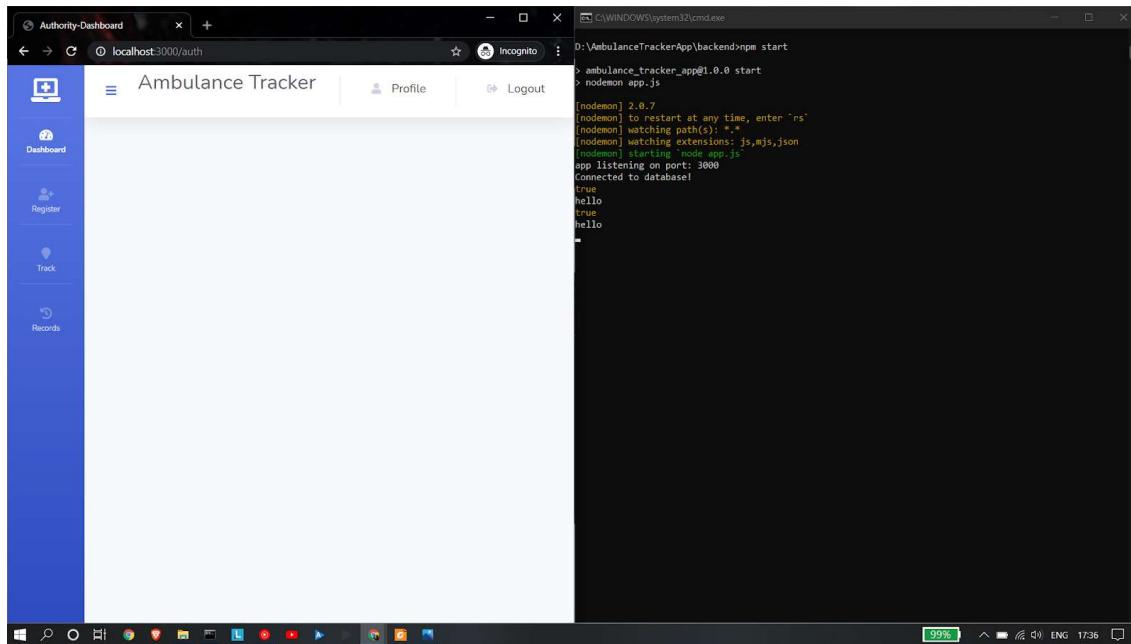
Username : FaculTy

Expected o/p : Successful Login

Output :



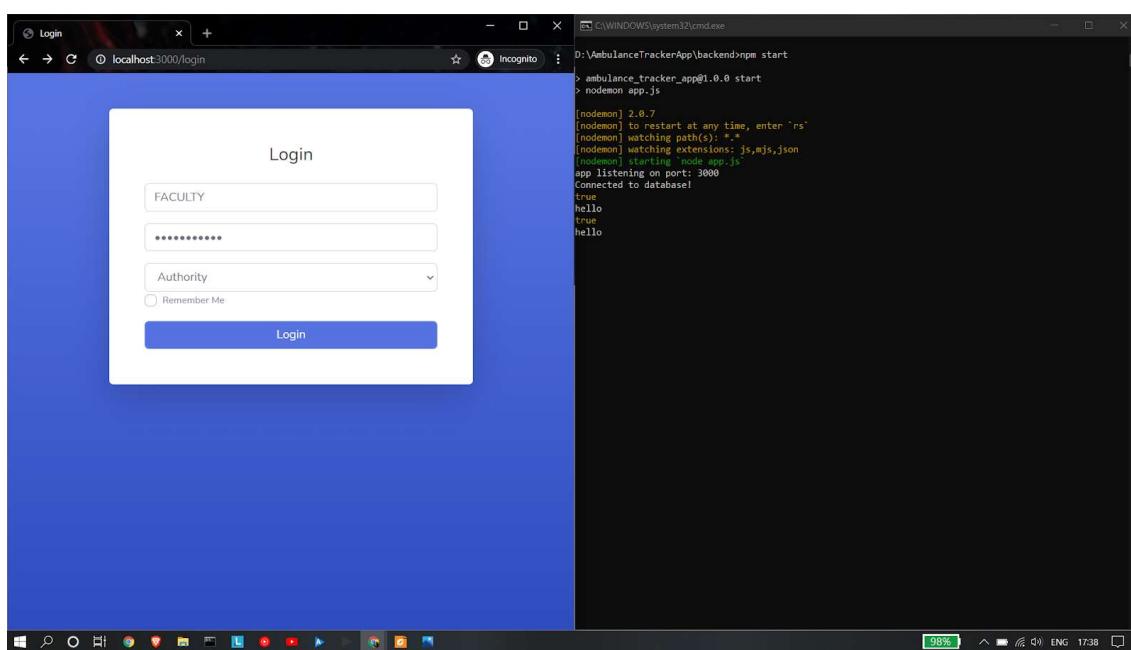
Successful Login :



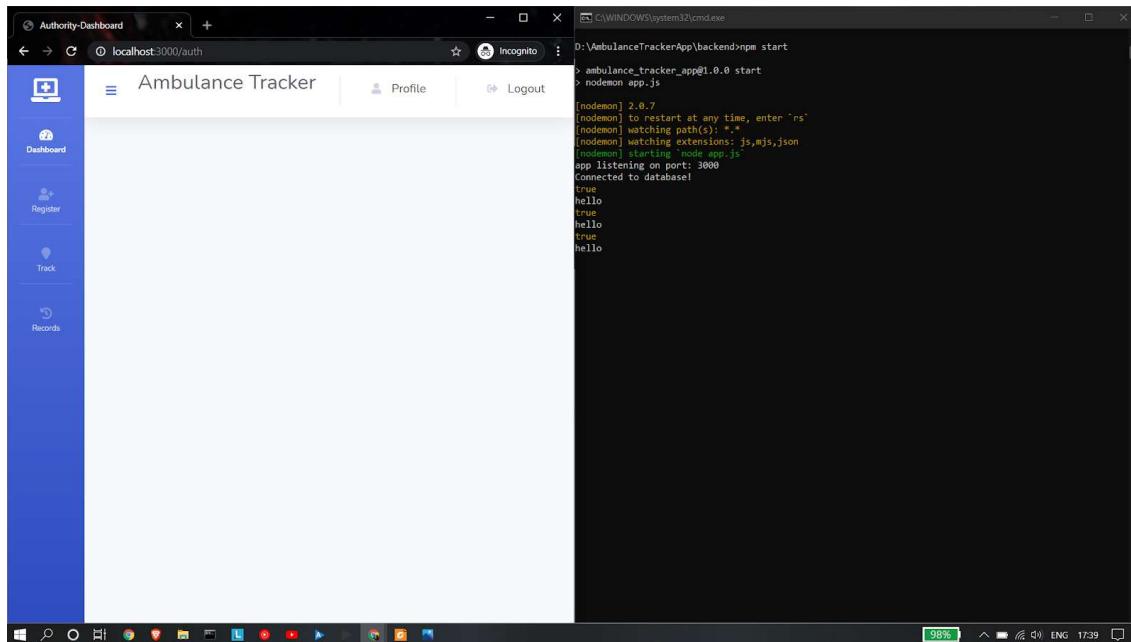
Username : FACULTY

Expected o/p : Successful Login

Output :



Successful Login :

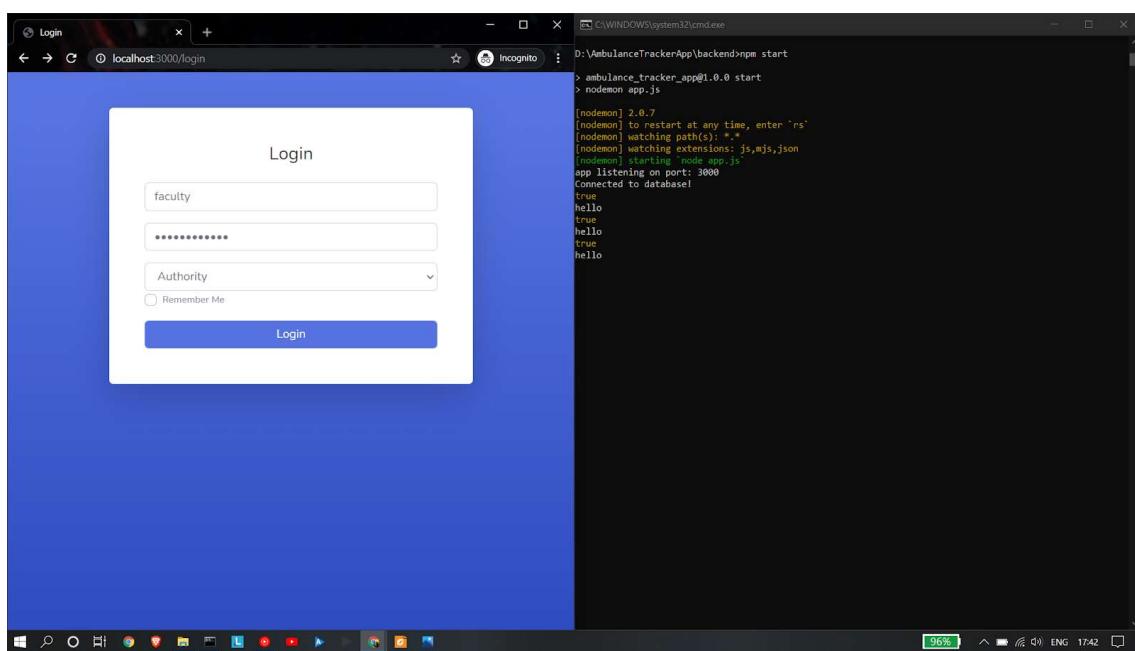


Case 2: Correct username, wrong password

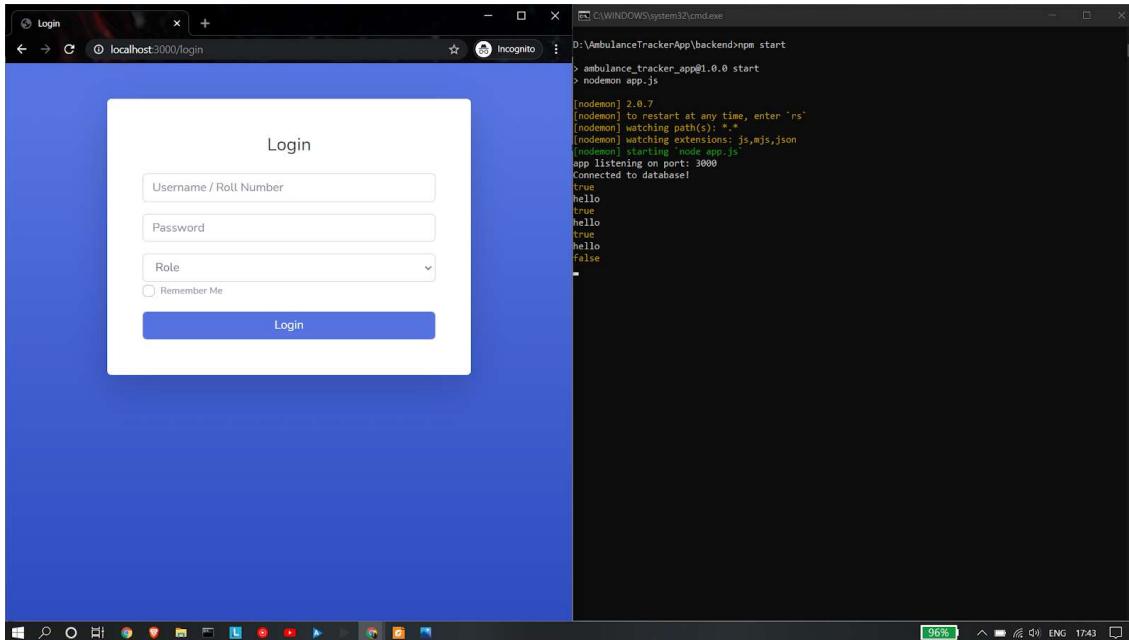
Username : faculty

Expected o/p : NoLogin

Output :



No Login :

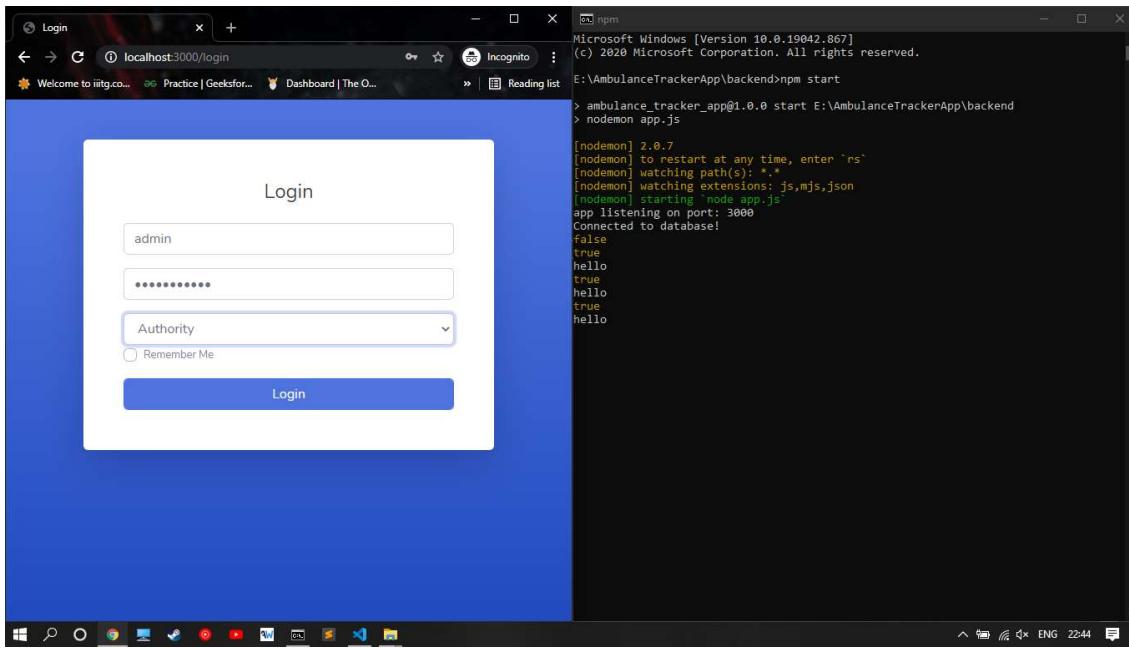


Case 3: Incorrect username, correct password and role

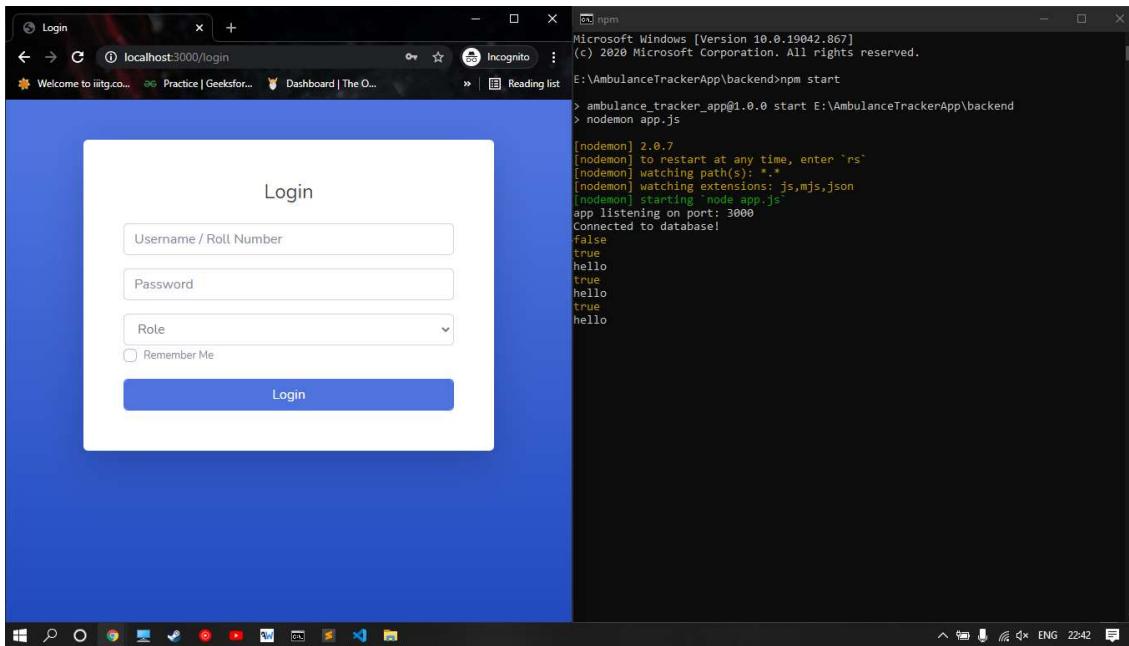
Username : admin

Expected o/p : No Login

Output :



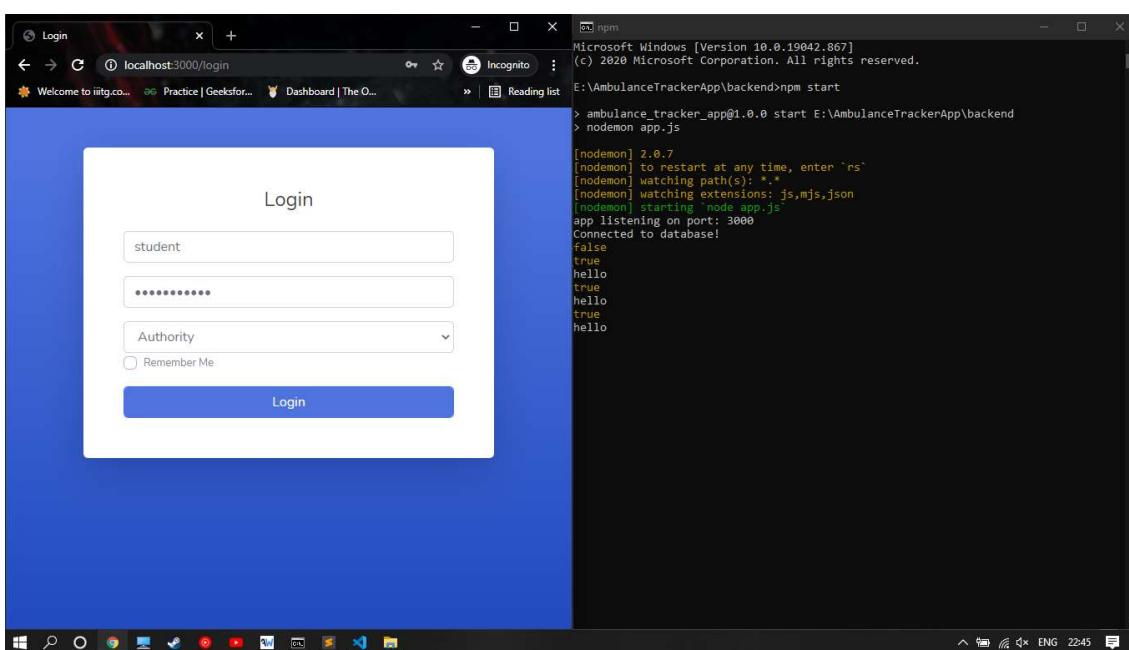
No Login :



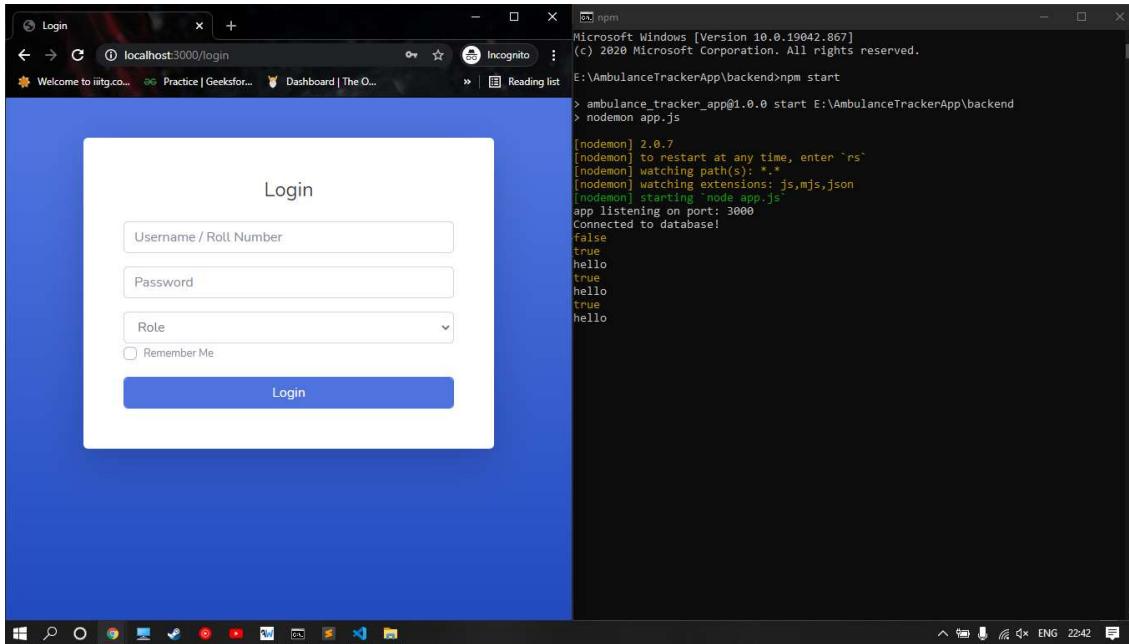
Username : student

Expected o/p : No Login

Output :



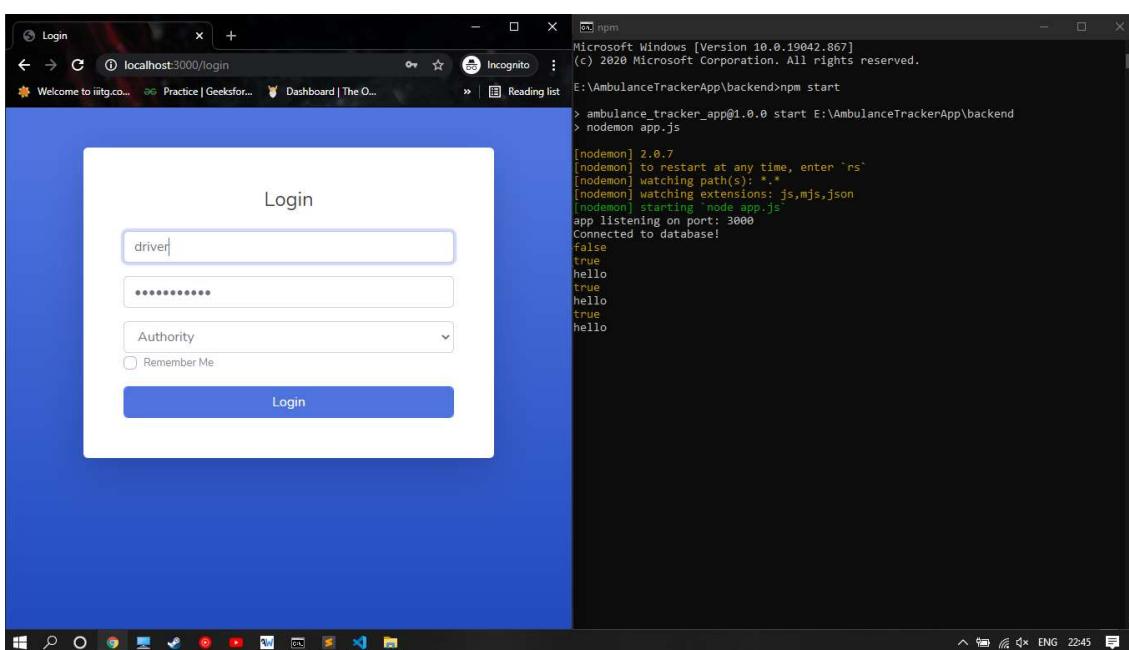
No Login :



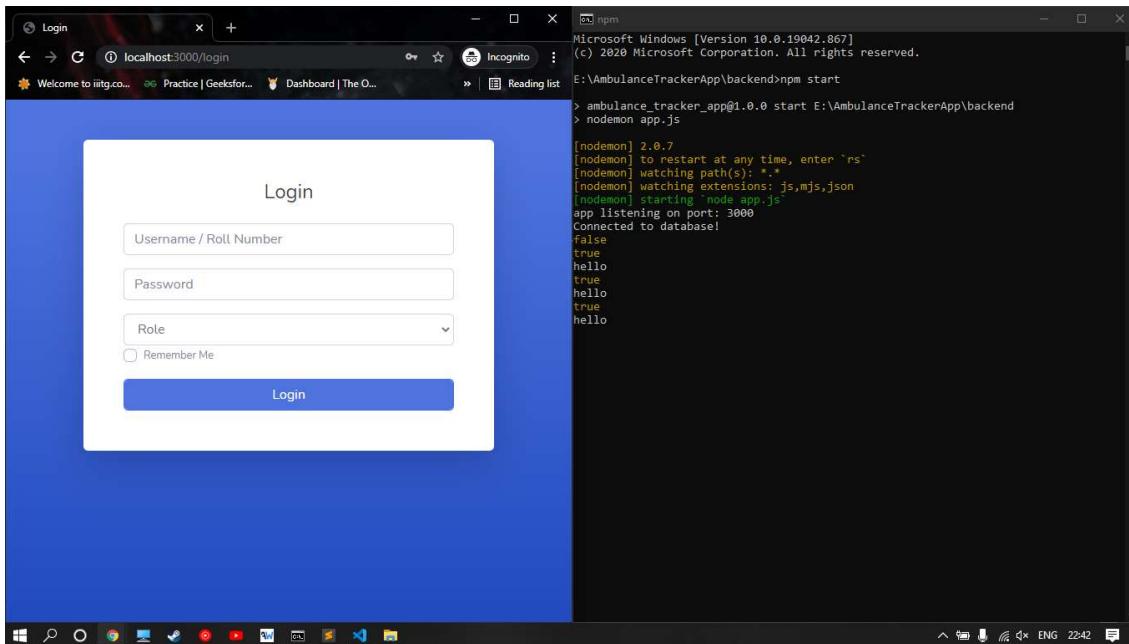
Username : driver

Expected o/p : No Login

Output :



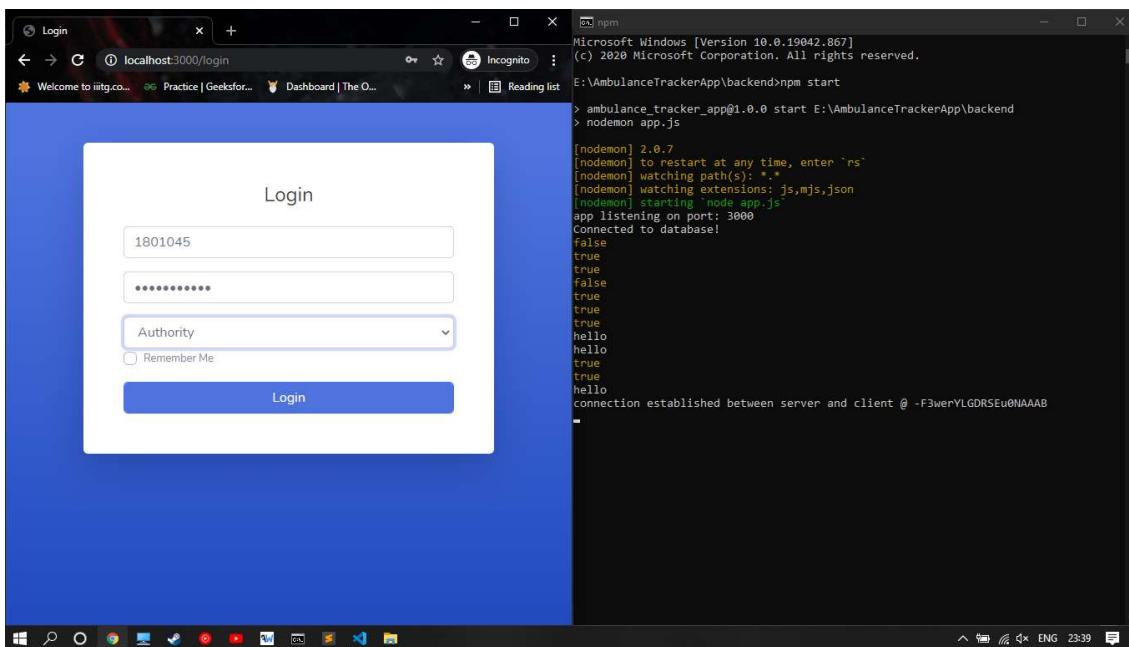
No Login :



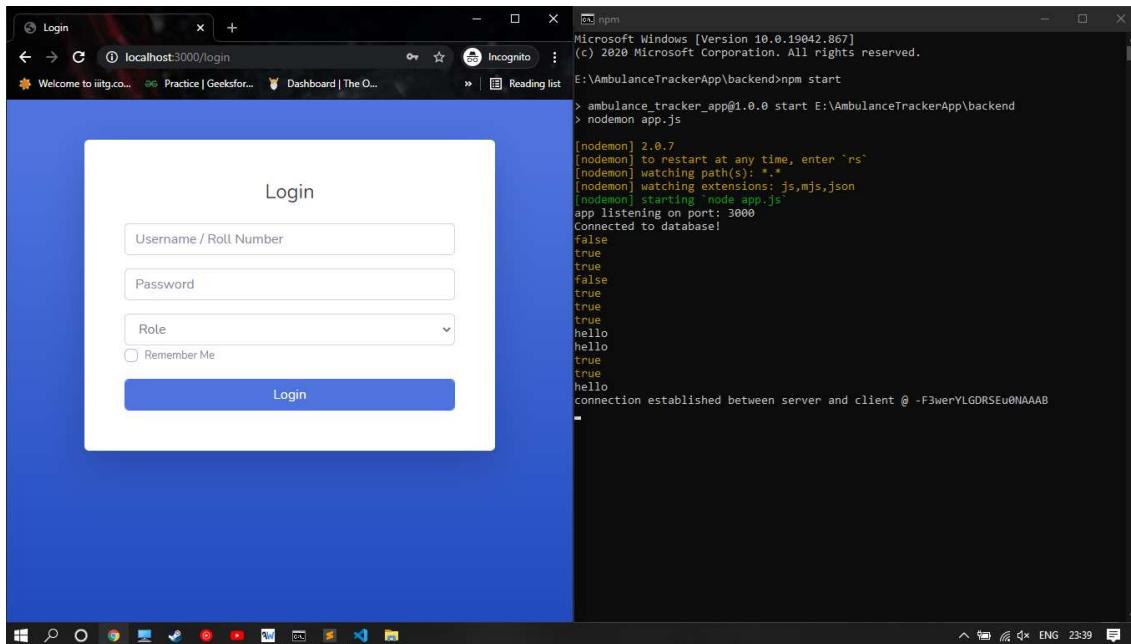
Username / Roll number : 1801045

Expected o/p : No Login

Output :



No Login :



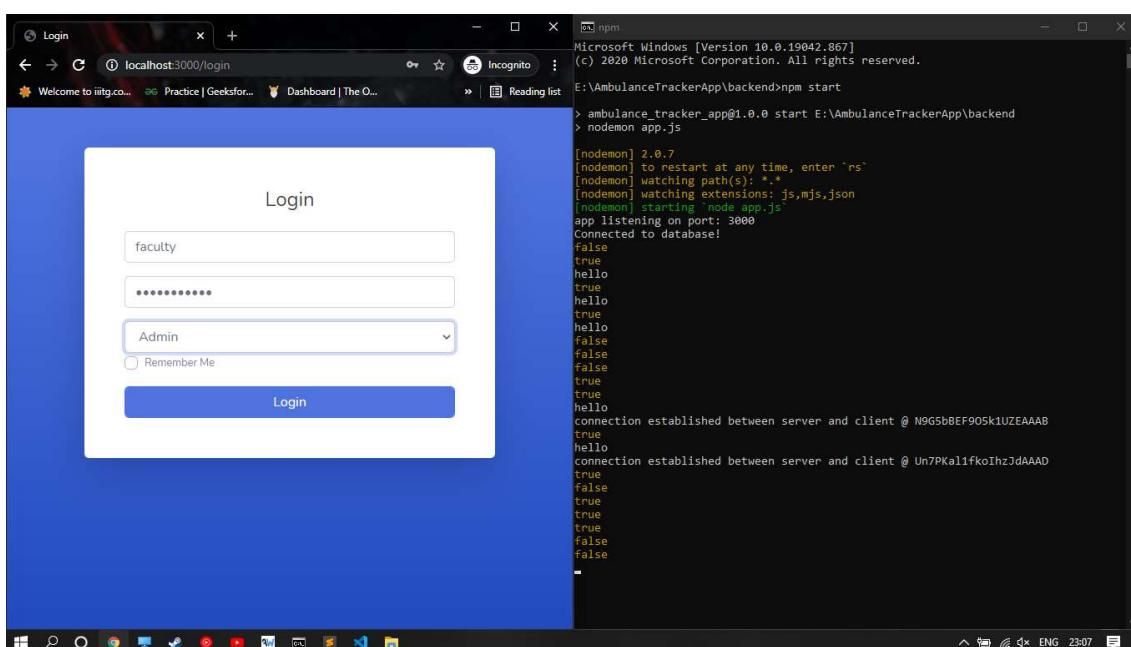
Case 4: Correct username, correct password but Incorrect role

Username : faculty

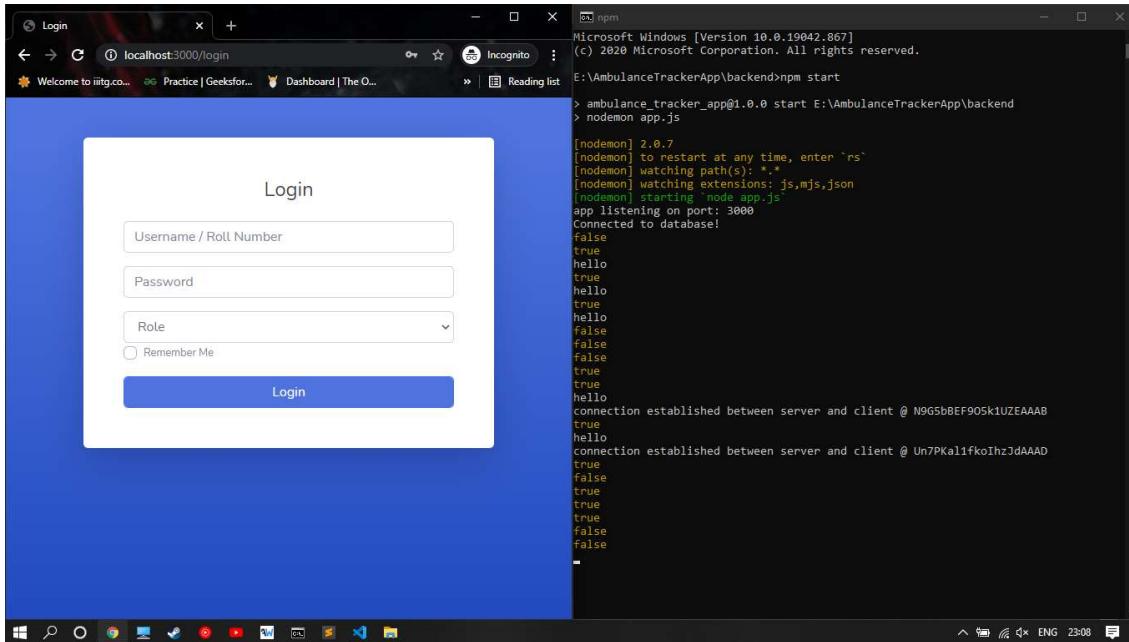
Role : Admin

Expected o/p : No Login

Output :



No Login :

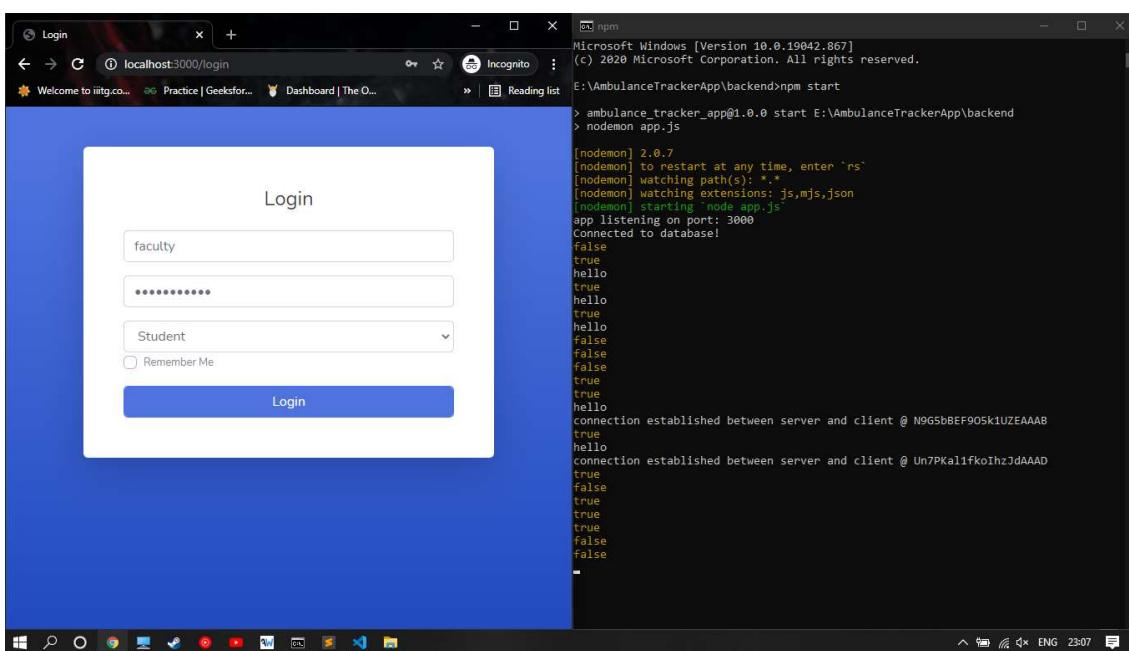


Username : faculty

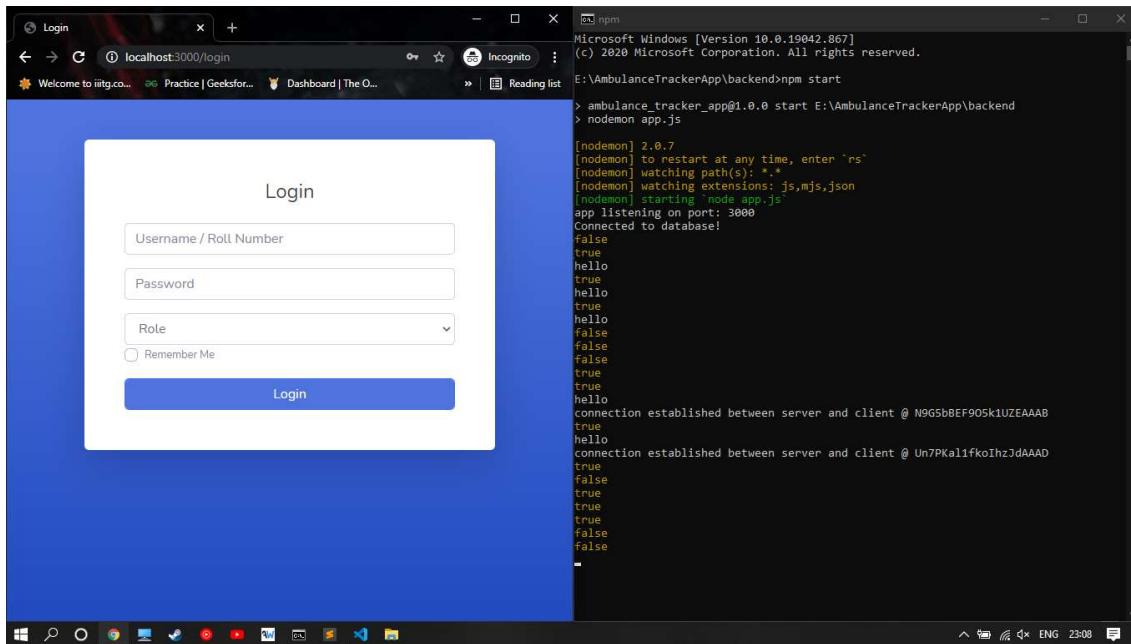
Role : Student

Expected o/p : No Login

Output :



No Login :

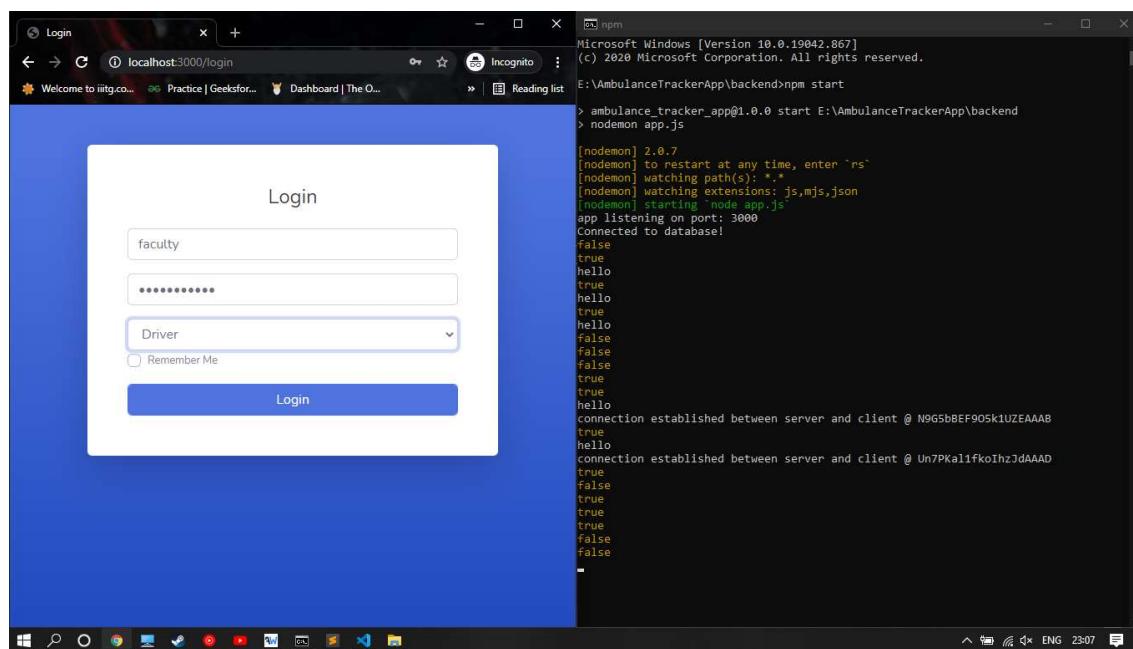


Username : faculty

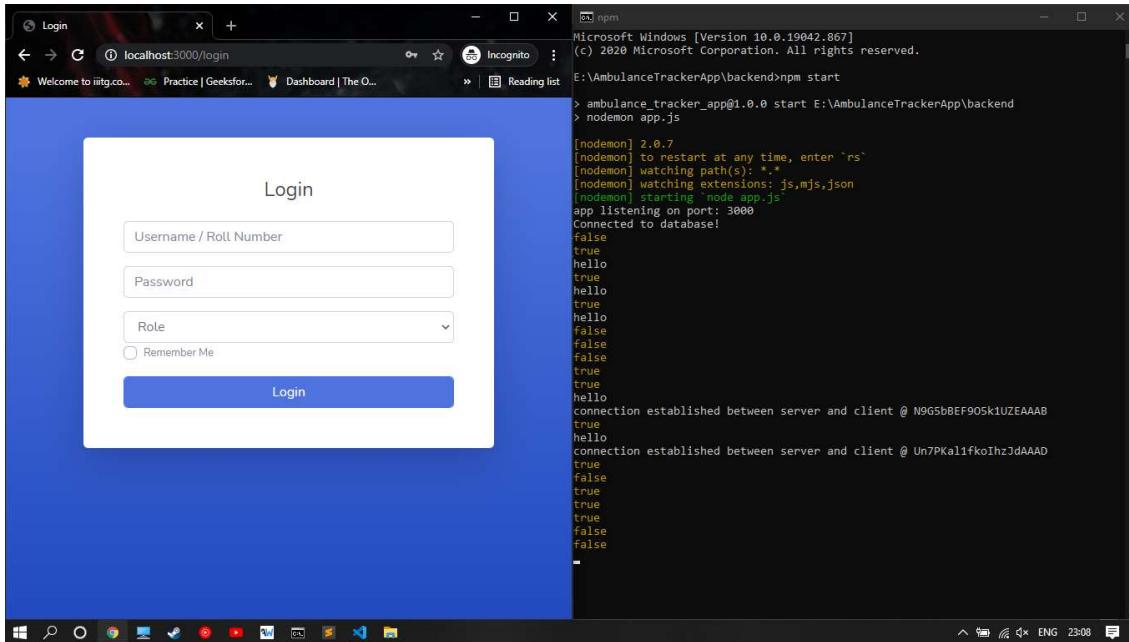
Role : Driver

Expected o/p : No Login

Output :



No Login :



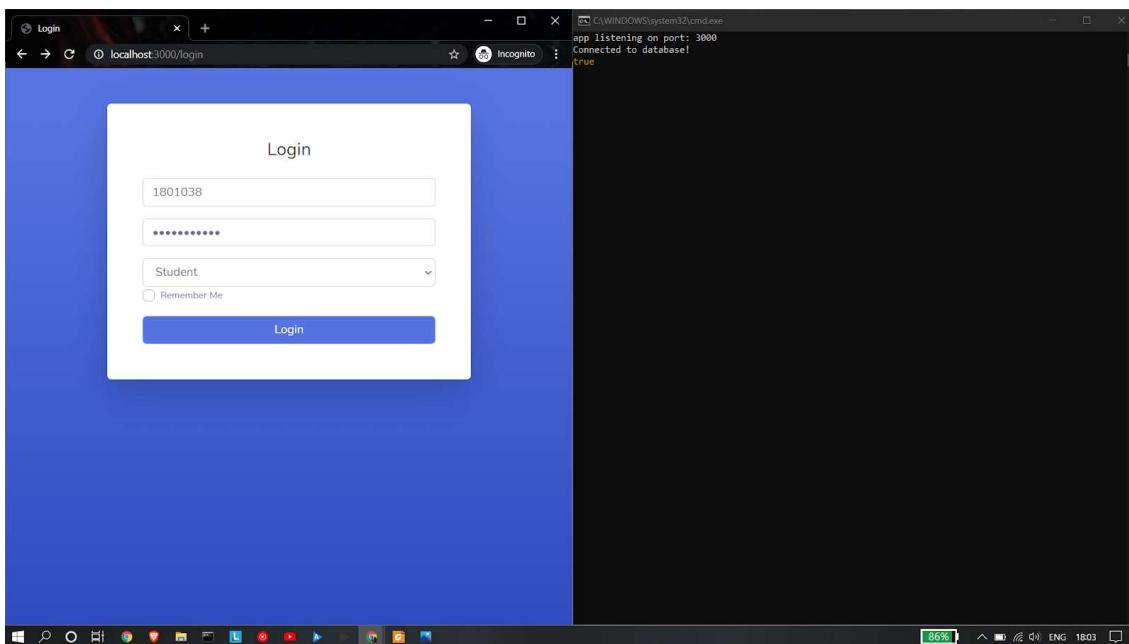
Login for Student-

Case 1: Successful login of the student

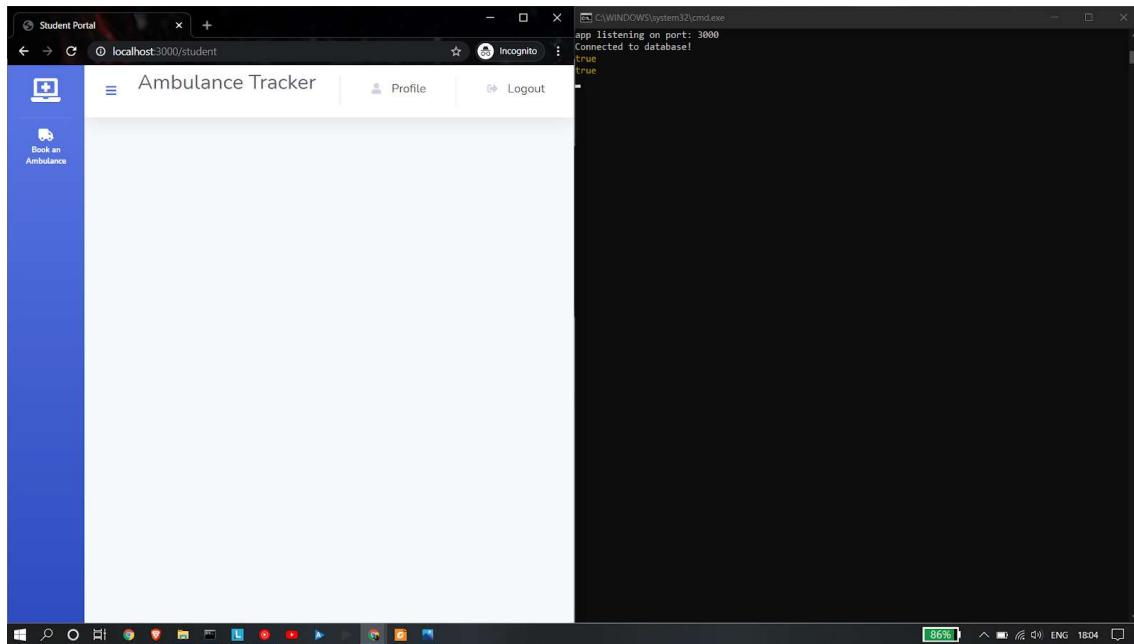
Username / Roll Number : 1801038

Expected o/p : Successful Login

Output :



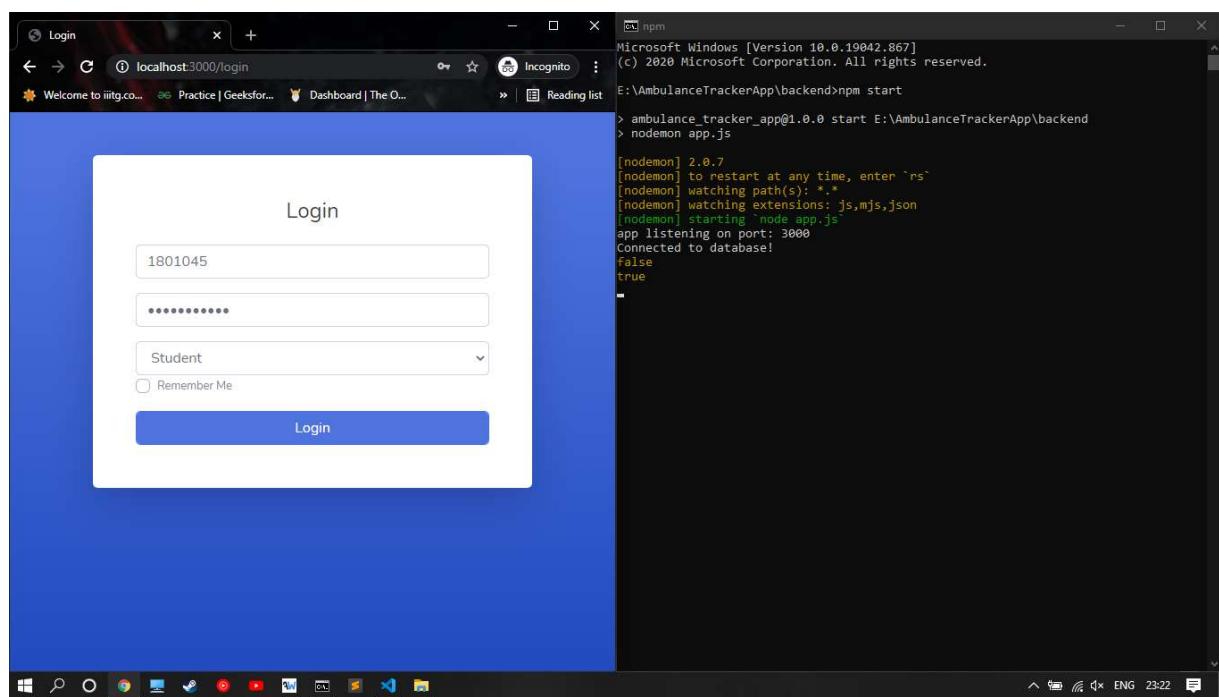
Successful Login :



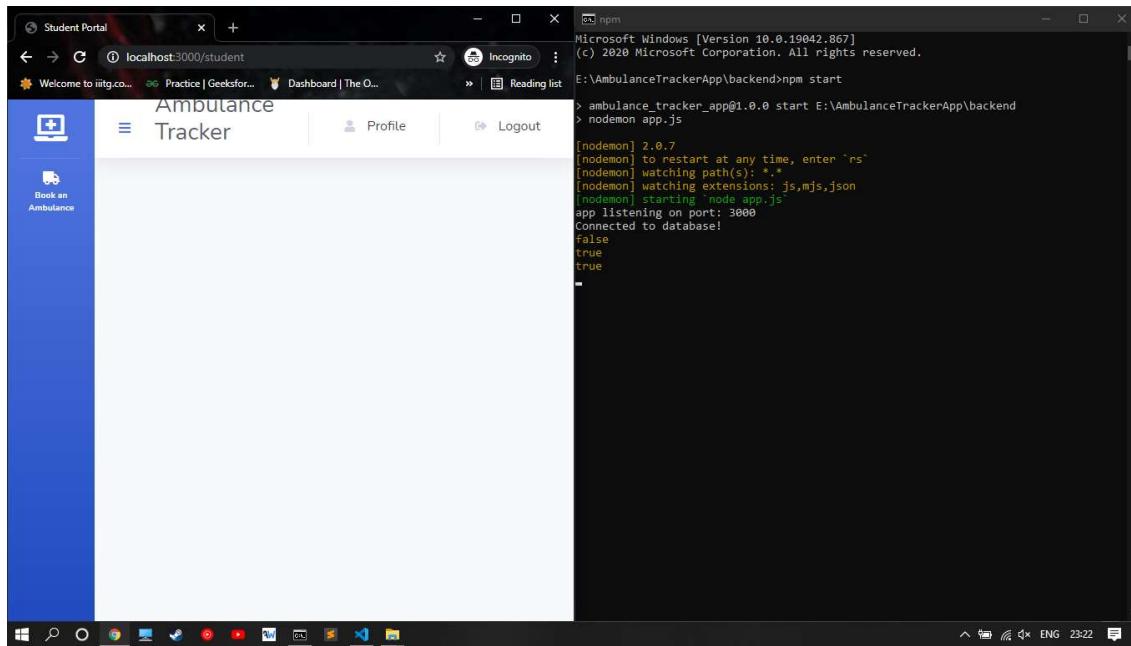
Username / Roll Number : 1801045

Expected o/p : Successful Login

Output :



Successful Login :

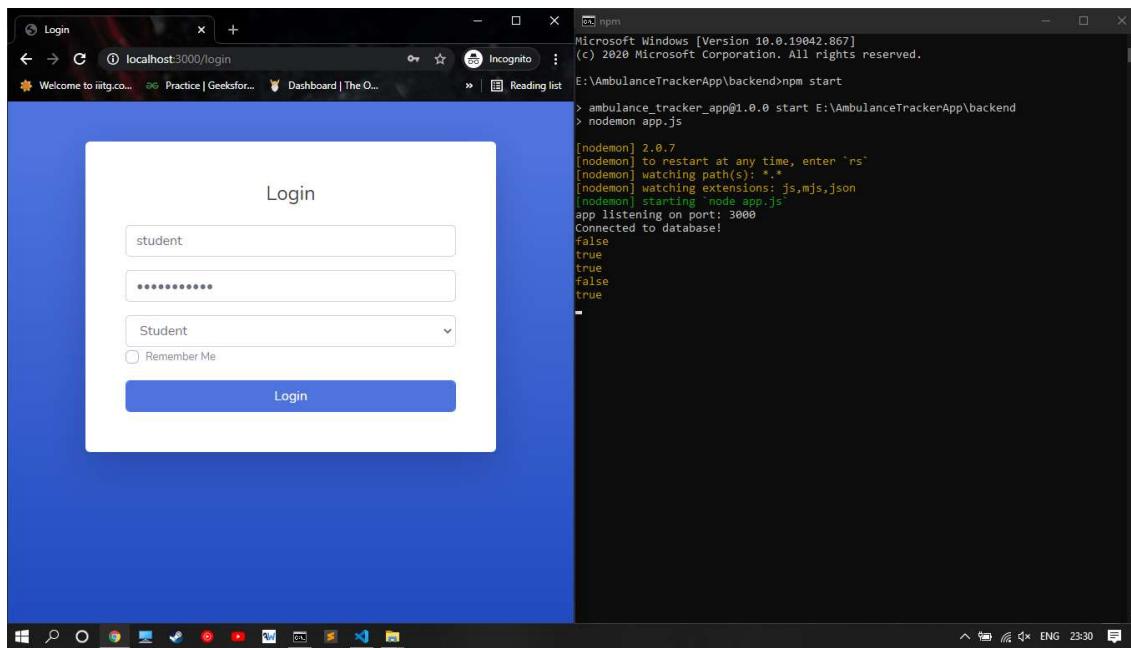


Case 2: Wrong Username / Roll Number, correct password and role

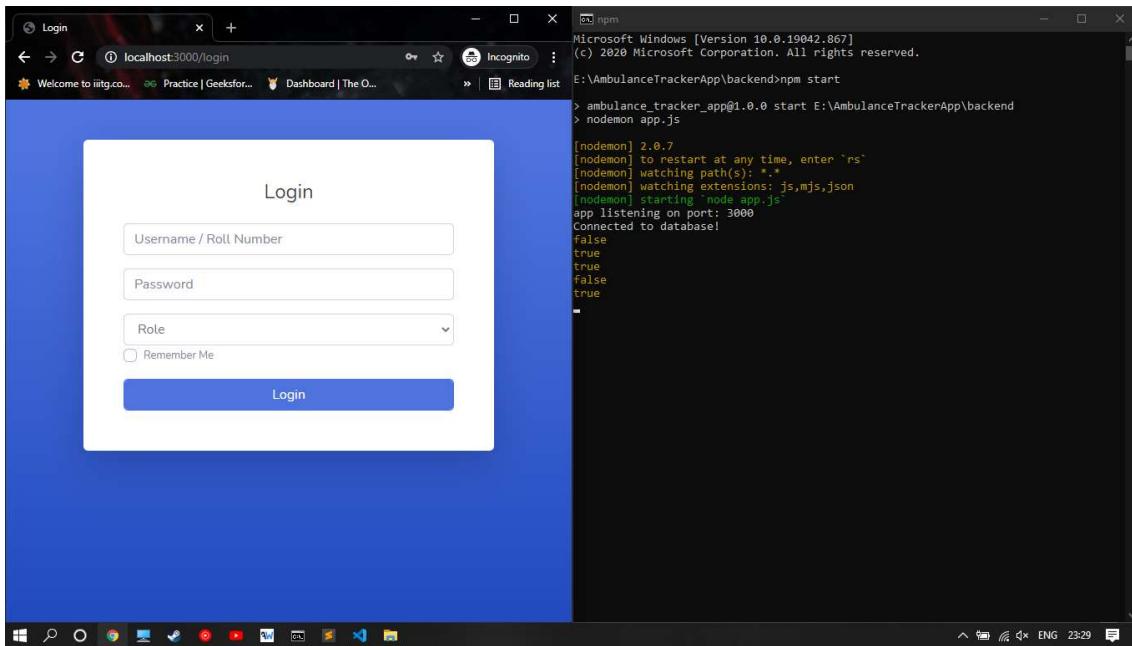
Username / Roll Number : student

Expected o/p : No Login

Output :



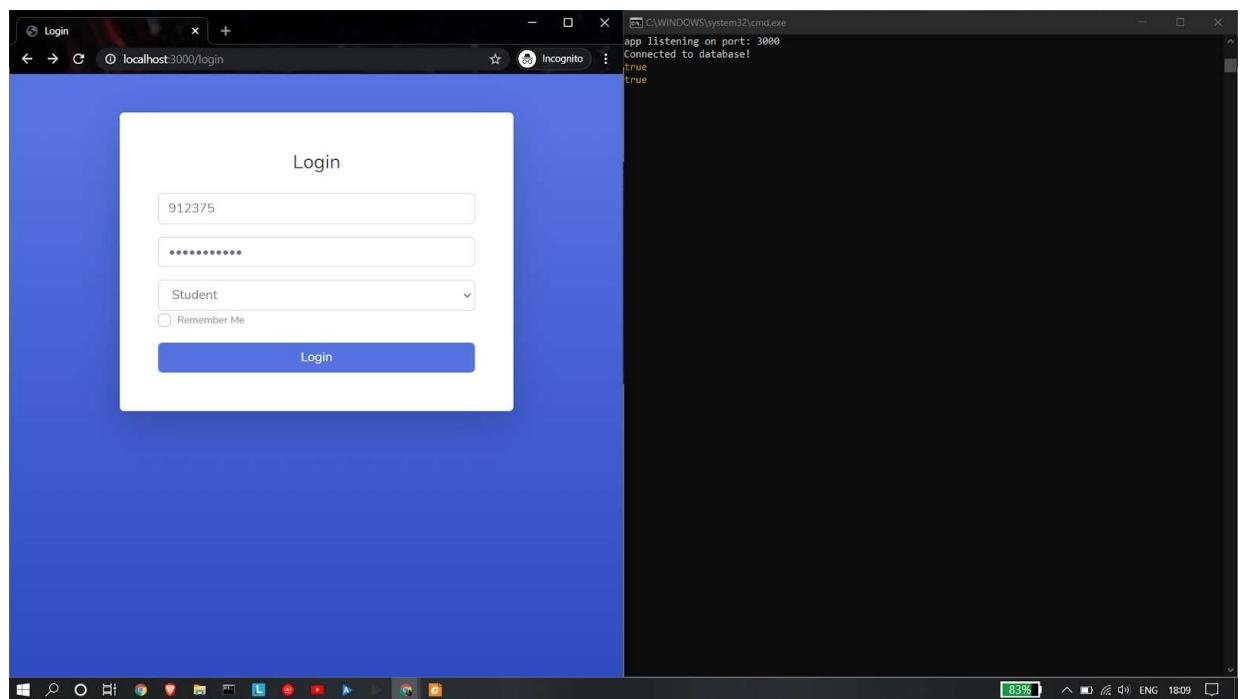
No Login :



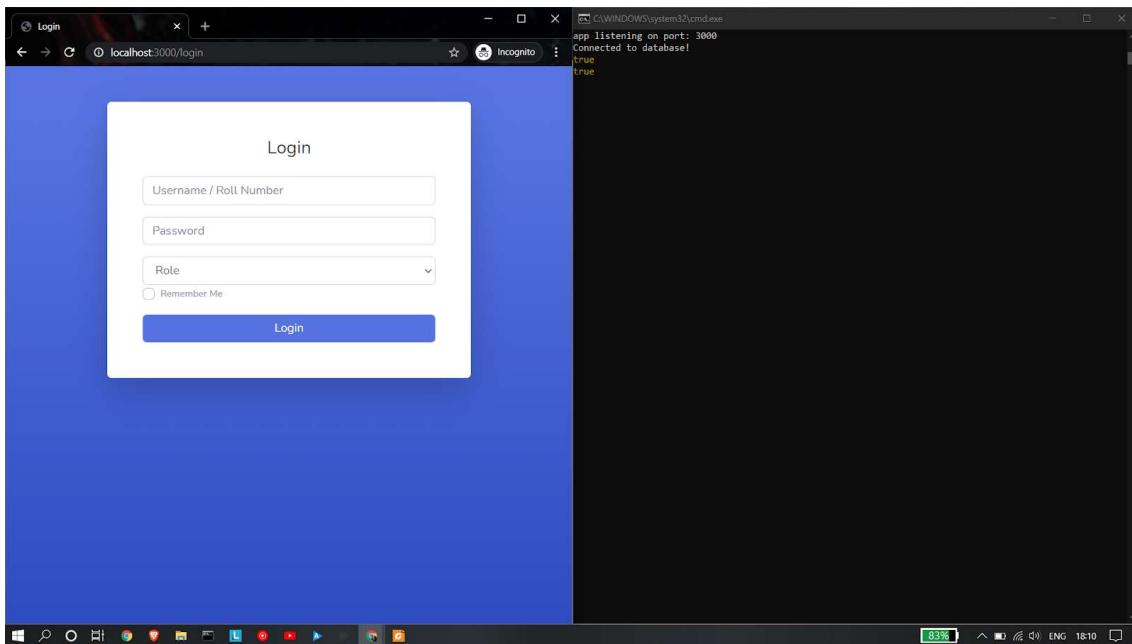
Username / Roll Number : 912375

Expected o/p : No Login

Output :



No Login :



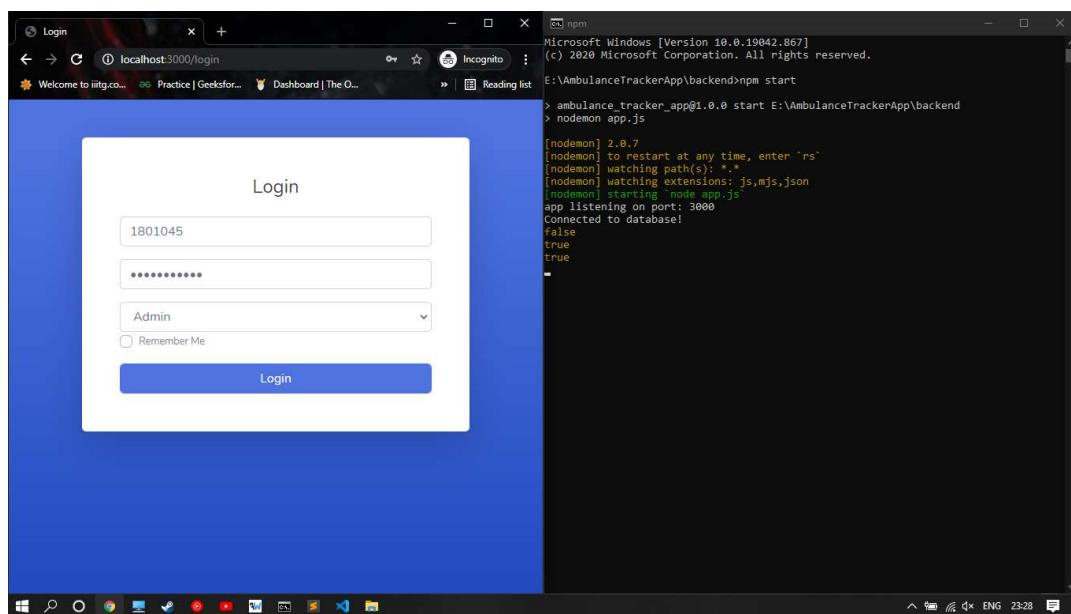
Case 3: Correct Username / Roll Number, Correct password but Incorrect role

Username / Roll Number : 1801045

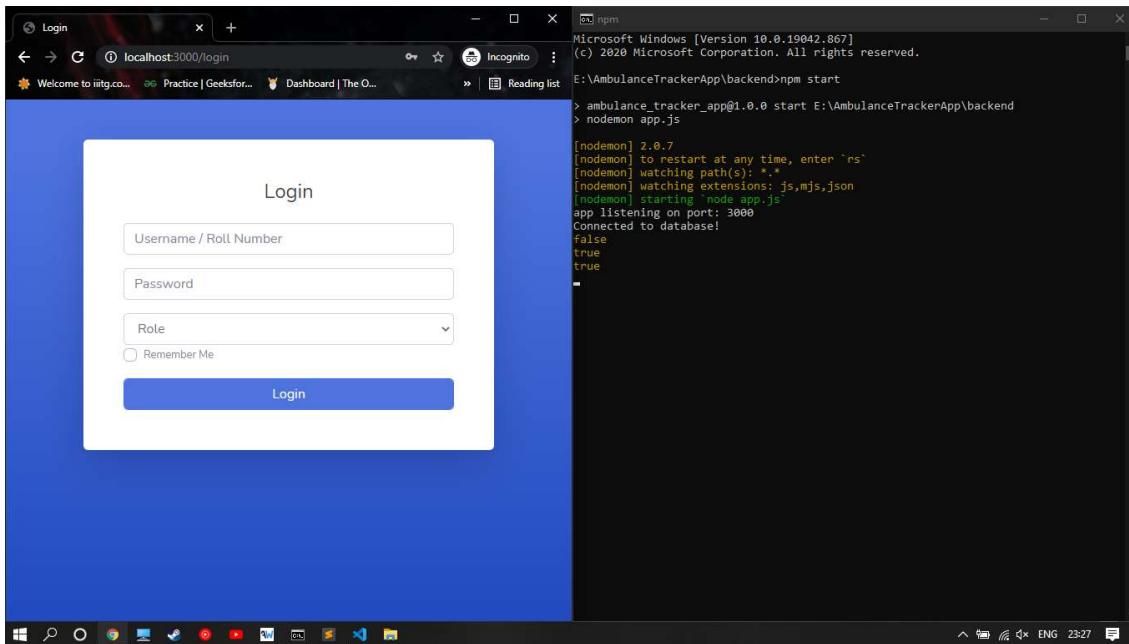
Role : Admin

Expected o/p : No Login

Output :



No Login :

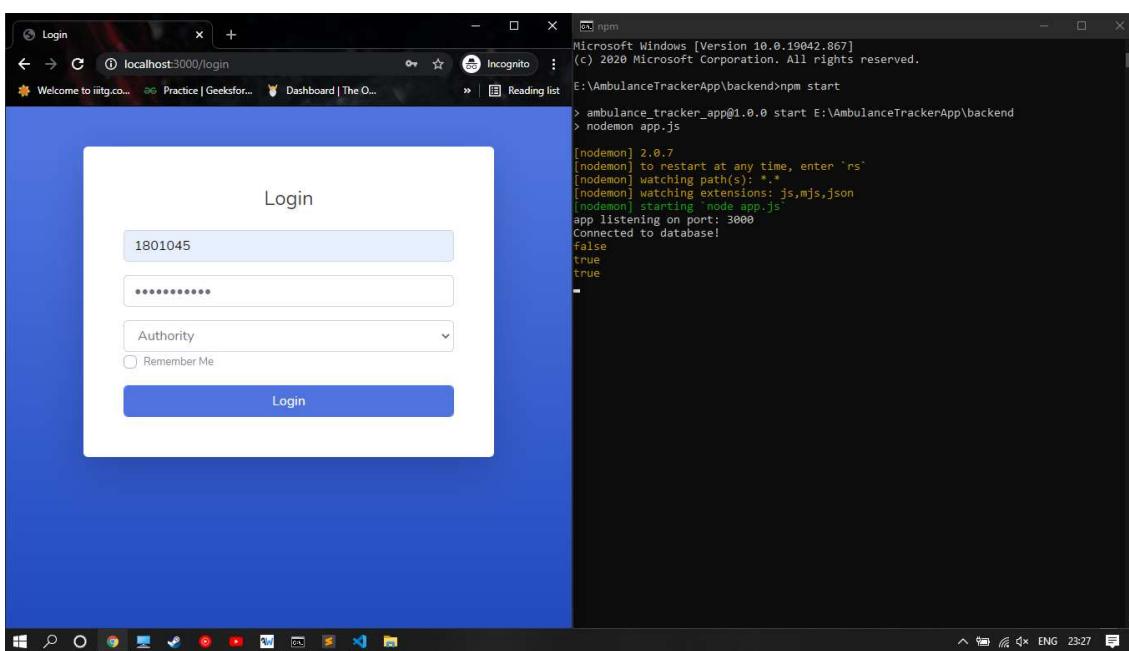


Username / Roll Number : 1801045

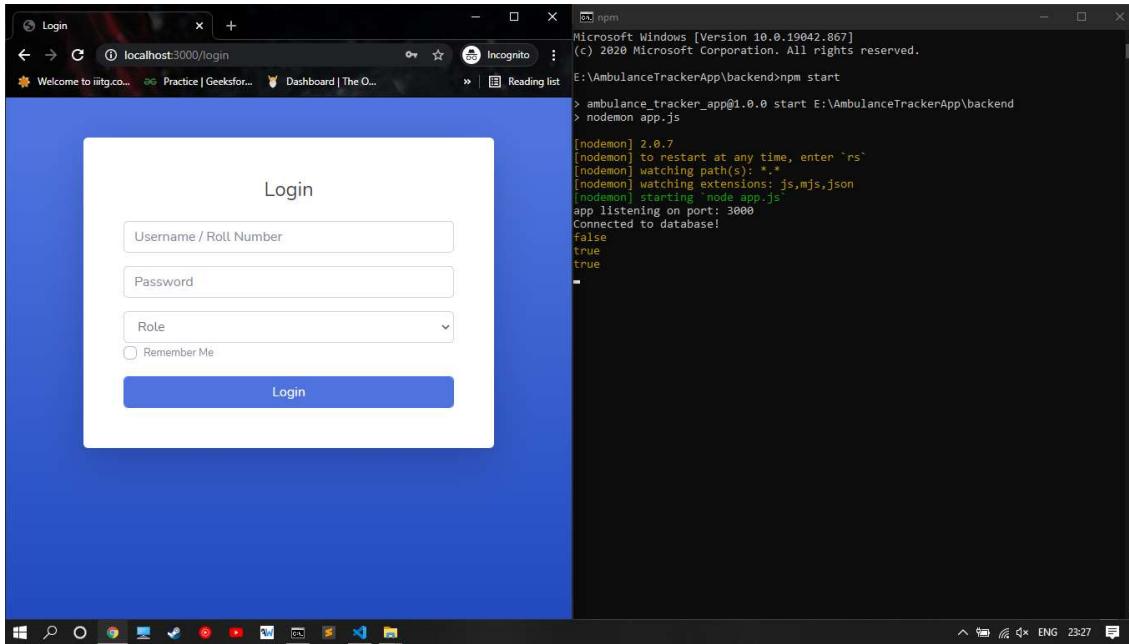
Role : Authority

Expected o/p : No Login

Output :



No Login :

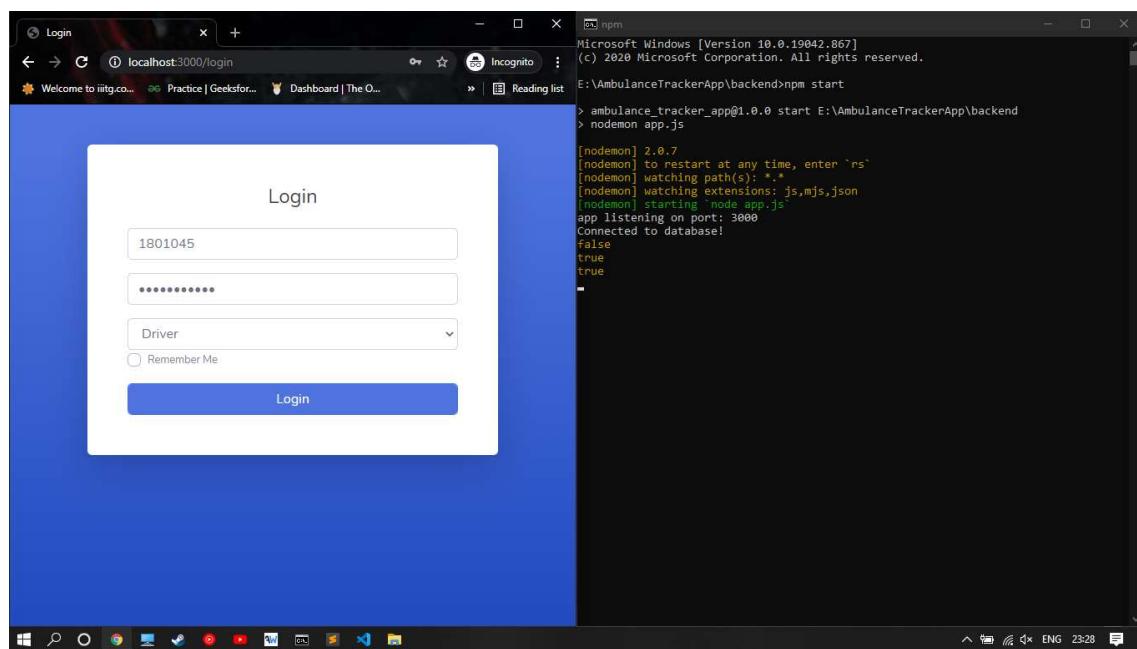


Username / Roll Number : 1801045

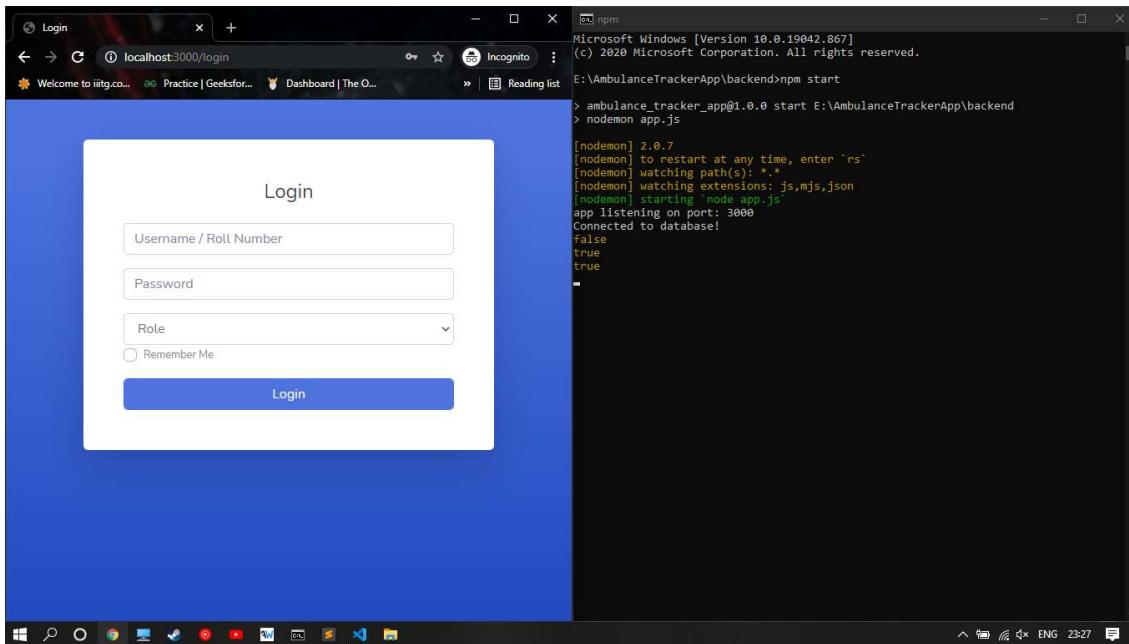
Role : Driver

Expected o/p : No Login

Output :



No Login :

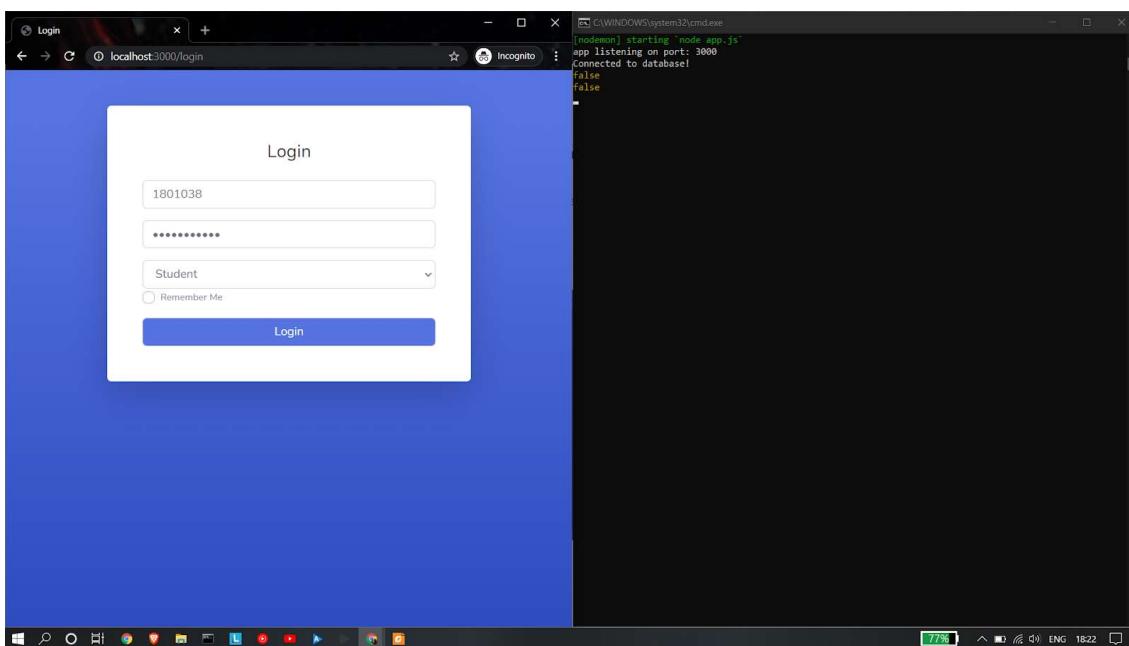


Case 4: Correct Username / Roll Number, Incorrect password

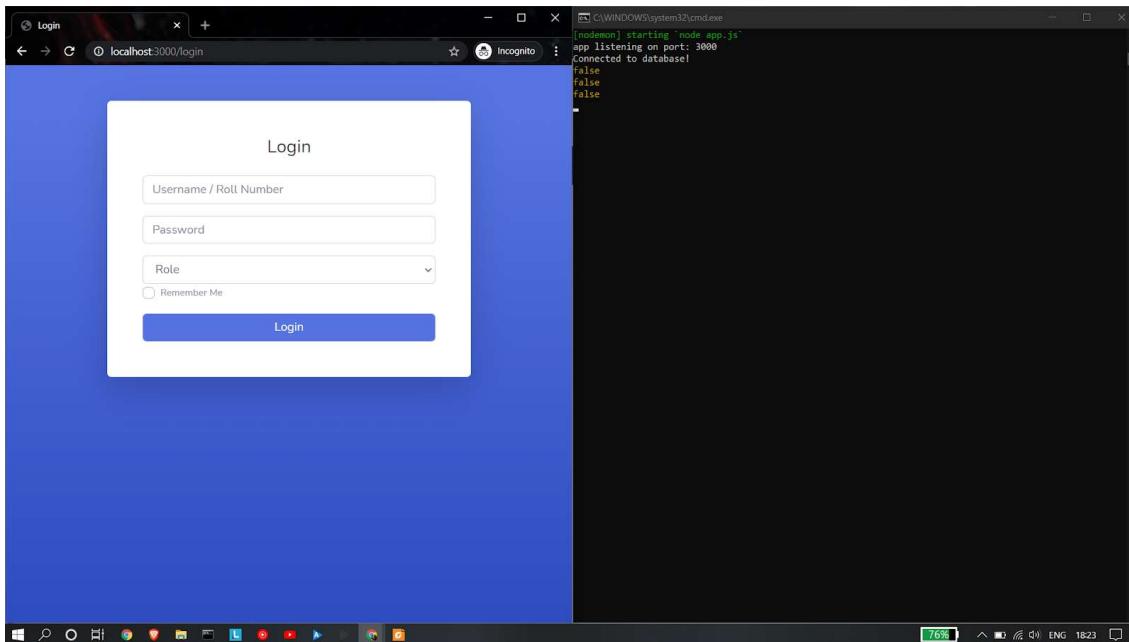
Username / Roll Number : 1801038

Expected o/p : No Login

Output :



No Login :



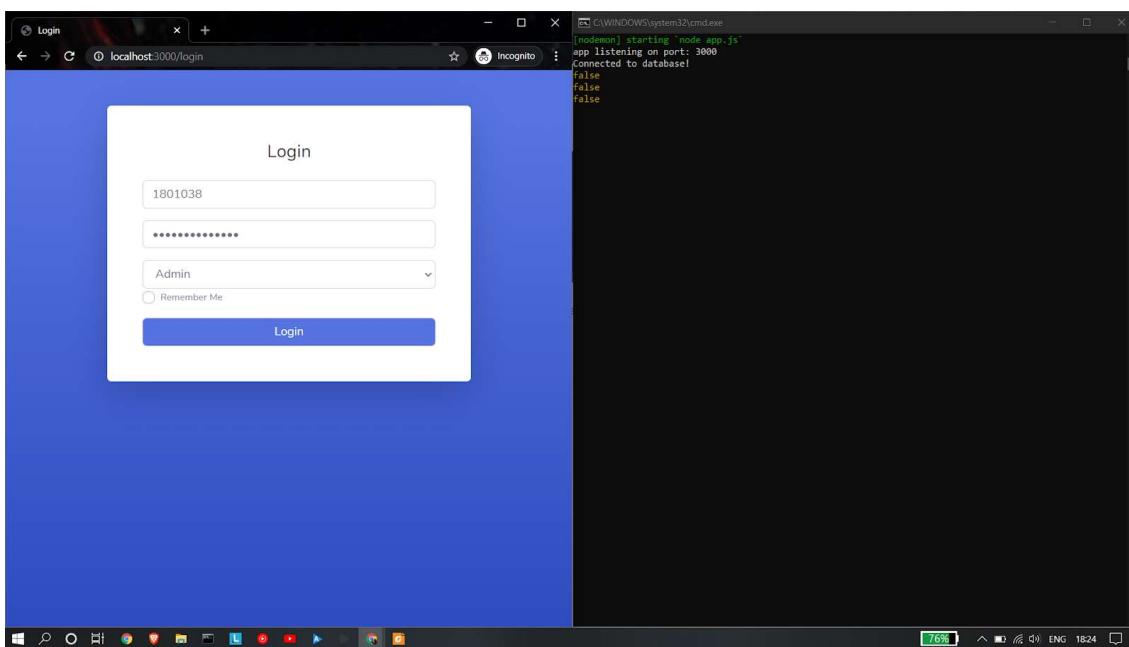
Case 4: Correct Username / Roll Number, Incorrect password and role

Username / Roll Number : 1801038

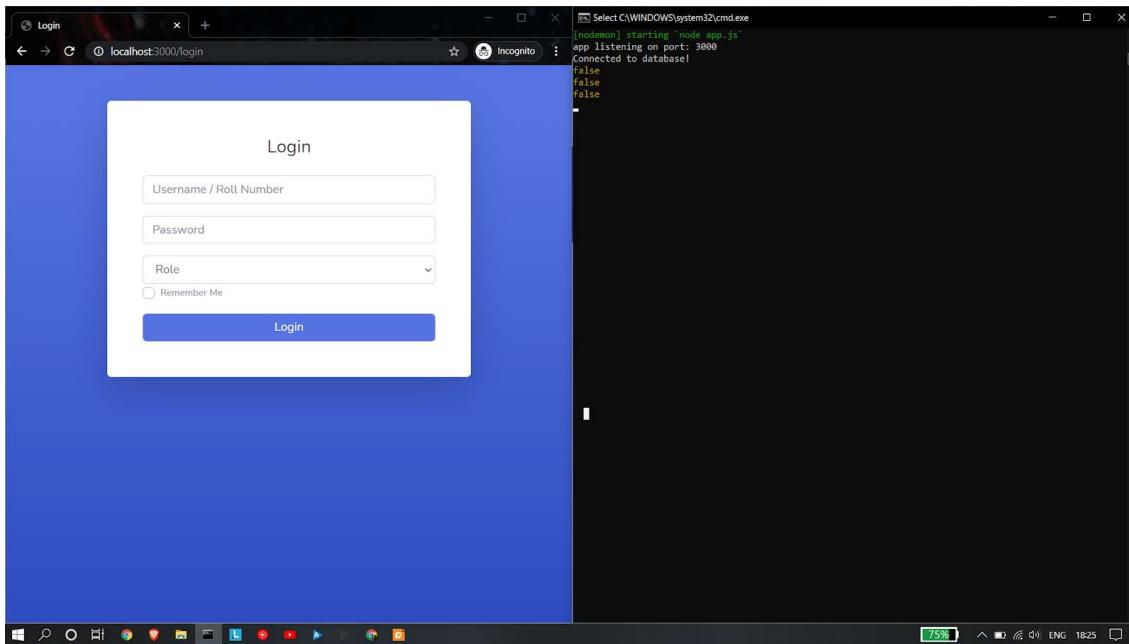
Role : Admin

Expected o/p : No Login

Output :



No Login :



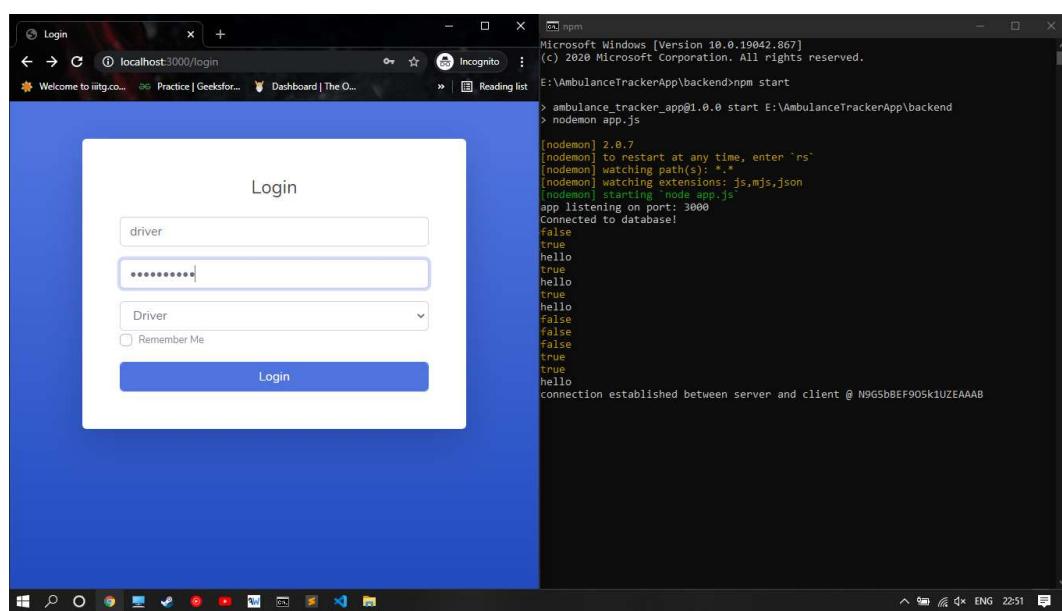
Login for Driver-

Case 1: Successful login of the driver

Username : driver

Expected o/p : Successful Login

Output :



Successful Login :

The screenshot shows a dual-monitor setup. On the left monitor, a Microsoft Edge browser window titled 'Driver-Dashboard' is open at `localhost:3000/driver`. The page displays the 'Ambulance Tracker' dashboard with sections for 'Dashboard' and 'Pending Rides'. On the right monitor, a Windows Command Prompt window titled 'npm' is running the command `npm start` in the directory `E:\AmbulanceTrackerApp\backend`. The terminal output shows the application starting and logging several 'hello' and 'true/false' messages, indicating successful database connection and socket communication.

```
Microsoft Windows [Version 10.0.19042.867]
(c) 2020 Microsoft Corporation. All rights reserved.

E:\AmbulanceTrackerApp\backend>npm start
> ambulance_tracker_app@1.0.0 start E:\AmbulanceTrackerApp\backend
> nodemon app.js

[nodemon] 2.0.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): ***!
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node app.js`
app listening on port: 3000
Connected to database!
false
true
hello
true
hello
true
hello
true
false
false
true
true
hello
connection established between server and client @ N9G5b8EF905k1UZEAAAB
true
hello
connection established between server and client @ Un7PKal1fkoIhzJdAAAD
-
```

Username : Driver

Expected o/p : Successful Login

Output :

The screenshot shows a dual-monitor setup. On the left monitor, a Microsoft Edge browser window titled 'Login' is open at `localhost:3000/login`. It displays a 'Login' form with fields for 'Driver' (text input), password (password input), role dropdown (set to 'Driver'), and a 'Remember Me' checkbox. On the right monitor, a Windows Command Prompt window titled 'npm' is running the command `npm start` in the directory `E:\AmbulanceTrackerApp\backend`. The terminal output shows the application starting and logging several 'hello' and 'true/false' messages, indicating successful database connection and socket communication.

```
Microsoft Windows [Version 10.0.19042.867]
(c) 2020 Microsoft Corporation. All rights reserved.

E:\AmbulanceTrackerApp\backend>npm start
> ambulance_tracker_app@1.0.0 start E:\AmbulanceTrackerApp\backend
> nodemon app.js

[nodemon] 2.0.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): ***!
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node app.js`
app listening on port: 3000
Connected to database!
false
true
hello
true
hello
true
hello
true
false
false
true
true
hello
connection established between server and client @ N9G5b8EF905k1UZEAAAB
true
hello
connection established between server and client @ Un7PKal1fkoIhzJdAAAD
true
false
true
true
true
false
false
-
```

Successful Login :

The screenshot shows a dual-pane interface. On the left is a web browser window titled 'Driver-Dashboard' displaying the 'Ambulance Tracker' dashboard. The dashboard has a sidebar with 'Dashboard' and 'Pending Rides'. The main area shows a message: 'Welcome to iitg.co... Practice | Geeksfor... Dashboard | The O...'. On the right is a terminal window titled 'npm' showing Node.js application logs. The logs indicate the application is running on port 3000, connected to a database, and receiving multiple 'hello' and 'true/false' messages from clients, with connection details like N9G5bBEF905k1UZEAAAB and Um7PKal1fkoIhzJdAAAD.

```
Microsoft Windows [Version 10.0.19042.867]
(c) 2020 Microsoft Corporation. All rights reserved.

E:\AmbulanceTrackerApp\backend>npm start
> ambulance_tracker_app@1.0.0 start E:\AmbulanceTrackerApp\backend
> nodemon app.js

[nodemon] 2.0.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node app.js`
app listening on port: 3000
Connected to database!
false
true
hello
true
hello
true
true
hello
false
false
false
true
true
true
hello
connection established between server and client @ N9G5bBEF905k1UZEAAAB
true
hello
connection established between server and client @ Um7PKal1fkoIhzJdAAAD
true
true
true
true
false
false
true
hello
connection established between server and client @ wWA0sJLhcWxoE_U1AAAF
-
```

Username : DriVER

Expected o/p : Successful Login

Output :

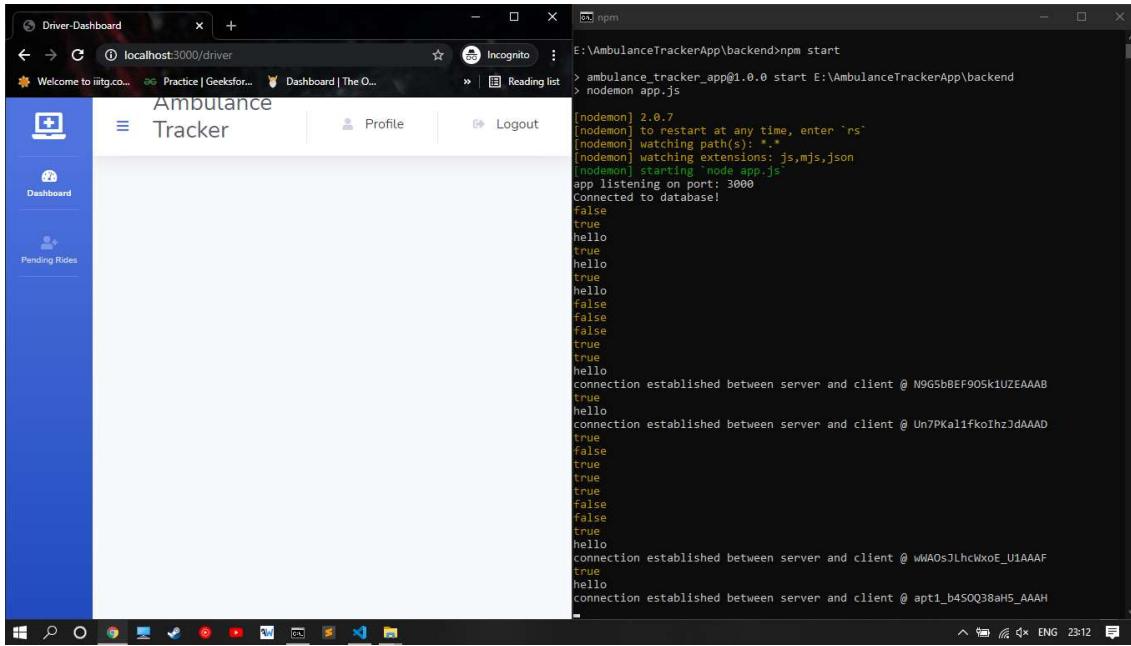
The screenshot shows a dual-pane interface. On the left is a web browser window titled 'Login' displaying the 'Ambulance Tracker' login page. The form fields are filled with 'DriVER' for username, '*****' for password, 'Driver' for role, and the 'Remember Me' checkbox is checked. On the right is a terminal window titled 'npm' showing Node.js application logs. The logs indicate the application is running on port 3000, connected to a database, and receiving multiple 'hello' and 'true/false' messages from clients, with connection details like N9G5bBEF905k1UZEAAAB and Um7PKal1fkoIhzJdAAAD.

```
Microsoft Windows [Version 10.0.19042.867]
(c) 2020 Microsoft Corporation. All rights reserved.

E:\AmbulanceTrackerApp\backend>npm start
> ambulance_tracker_app@1.0.0 start E:\AmbulanceTrackerApp\backend
> nodemon app.js

[nodemon] 2.0.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node app.js`
app listening on port: 3000
Connected to database!
false
true
hello
true
hello
true
true
hello
false
false
false
true
true
true
hello
connection established between server and client @ N9G5bBEF905k1UZEAAAB
true
hello
connection established between server and client @ Um7PKal1fkoIhzJdAAAD
true
false
true
true
true
false
false
true
true
hello
connection established between server and client @ wWA0sJLhcWxoE_U1AAAF
-
```

Successful Login :



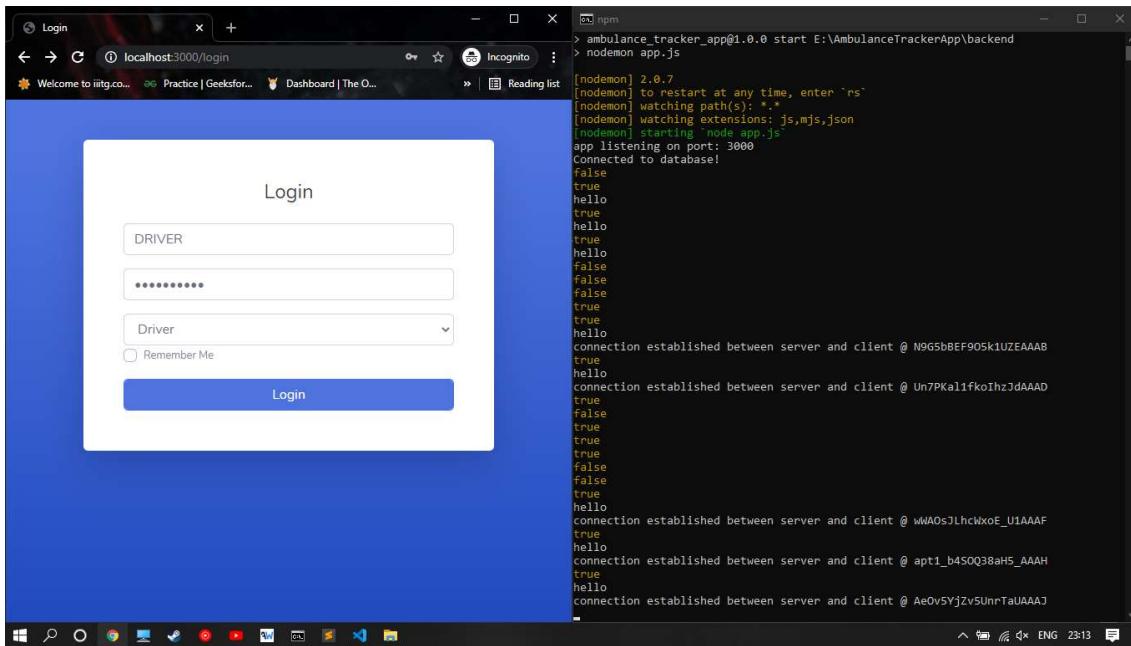
The screenshot shows a Windows desktop environment. On the left is a web browser window titled "Driver-Dashboard" displaying the URL "localhost:3000/driver". The page content is the "Ambulance Tracker" dashboard, showing a sidebar with "Dashboard" and "Pending Rides" options, and a main area with a blue background. On the right is a terminal window titled "npm" showing command-line output. The commands run are "E:\AmbulanceTrackerApp\backend>npm start" and "node app.js". The terminal output shows several "hello" messages, database connection logs, and client connection logs for three different clients.

```
E:\AmbulanceTrackerApp\backend>npm start
> ambulance_tracker_app@1.0.0 start E:\AmbulanceTrackerApp\backend
> nodemon app.js
[nodemon] 2.0.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): ***!
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node app.js`
app listening on port: 3000
Connected to database!
false
true
hello
true
hello
true
hello
false
false
true
true
true
hello
connection established between server and client @ N9G5bBEF905k1UZEAAAB
true
hello
connection established between server and client @ Un7PKalifkoIhzJdAAAD
true
false
true
true
true
false
false
true
true
hello
connection established between server and client @ wWAOsJLhcNxoE_U1AAAF
true
hello
connection established between server and client @ apt1_b4SOQ38ah5_AAAH
```

Username : DRIVER

Expected o/p : Successful Login

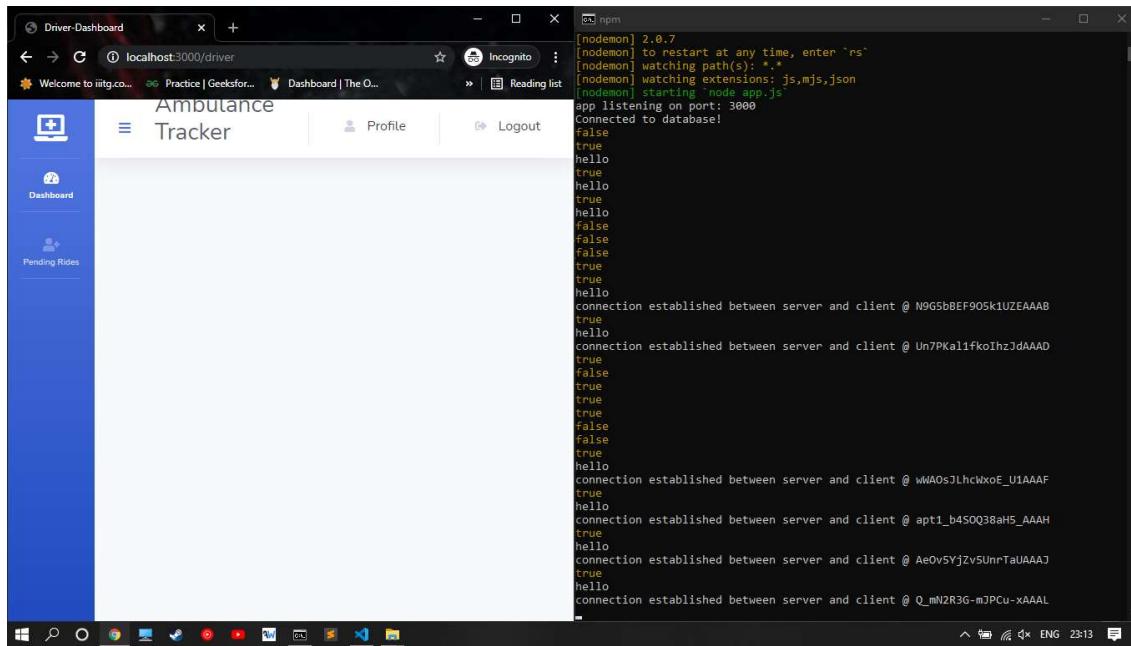
Output :



The screenshot shows a Windows desktop environment. On the left is a web browser window titled "Login" displaying the URL "localhost:3000/login". The page content is a "Login" form with fields for "Email" (containing "DRIVER"), "Password" (containing "*****"), "Role" (set to "Driver"), and a "Remember Me" checkbox. On the right is a terminal window titled "npm" showing command-line output. The commands run are "E:\AmbulanceTrackerApp\backend>npm start" and "node app.js". The terminal output shows several "hello" messages, database connection logs, and client connection logs for three different clients, identical to the previous screenshot.

```
> ambulance_tracker_app@1.0.0 start E:\AmbulanceTrackerApp\backend
> nodemon app.js
[nodemon] 2.0.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): ***!
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node app.js`
app listening on port: 3000
Connected to database!
false
true
hello
true
hello
true
hello
false
false
true
true
true
hello
connection established between server and client @ N9G5bBEF905k1UZEAAAB
true
hello
connection established between server and client @ Un7PKalifkoIhzJdAAAD
true
false
true
true
true
false
false
true
true
hello
connection established between server and client @ wWAOsJLhcNxoE_U1AAAF
true
hello
connection established between server and client @ apt1_b4SOQ38ah5_AAAH
true
hello
connection established between server and client @ Ae0v5YjZv5UnrTaUAAAJ
```

Successful Login :

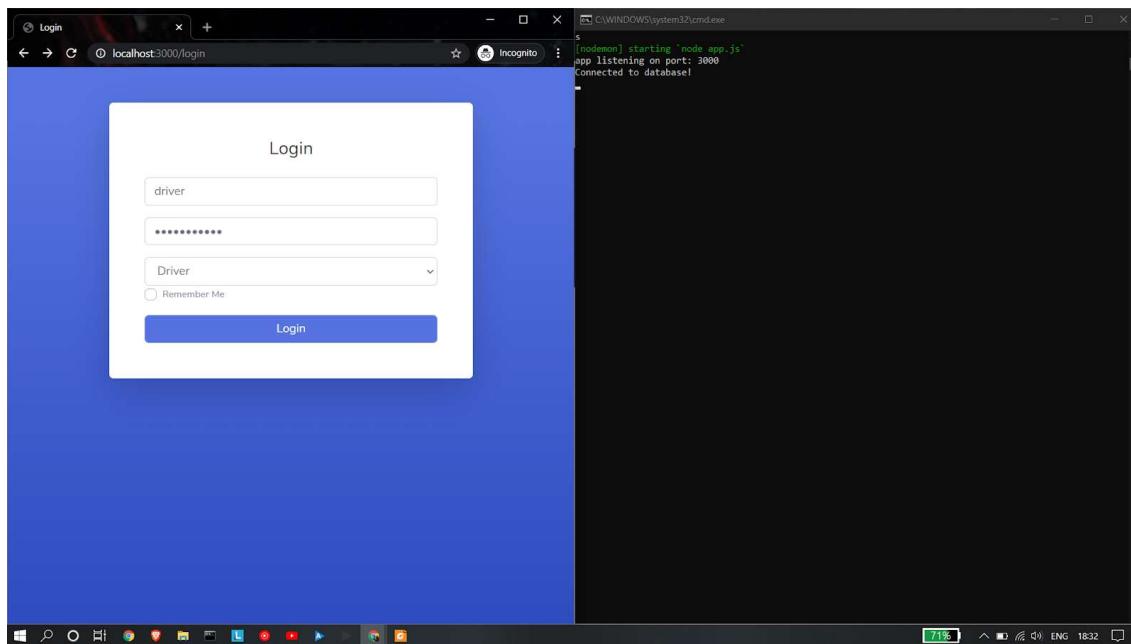


Case 2: Correct Username, wrong password

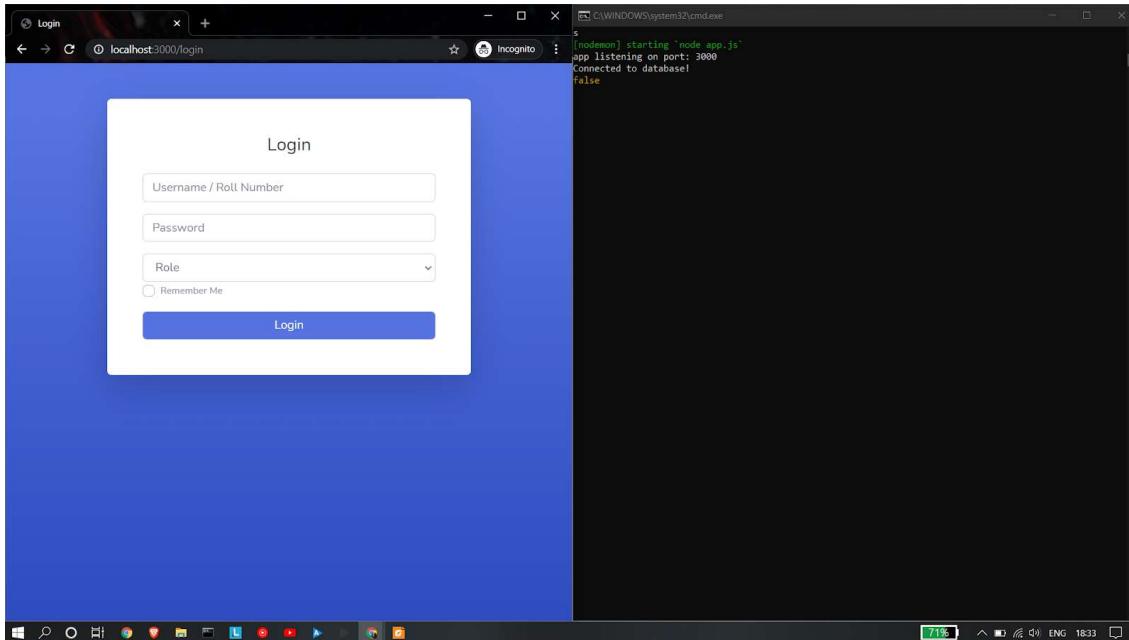
Username : driver

Expected o/p : No Login

Output :



No Login :



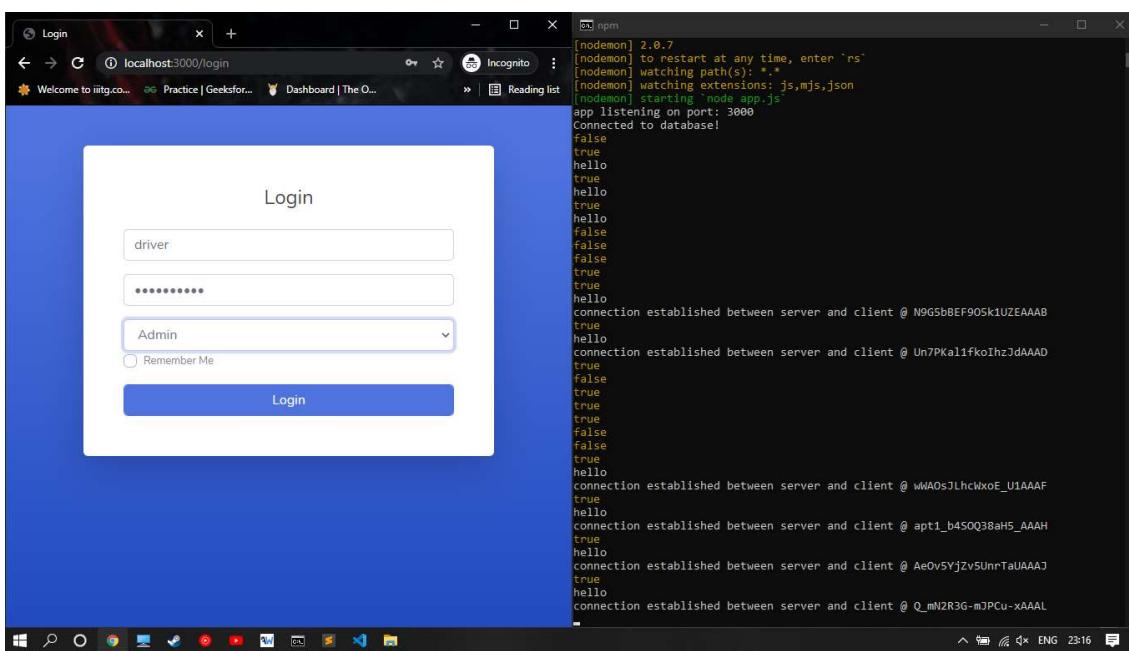
Case 3: Correct Username and password but Incorrect role

Username : driver

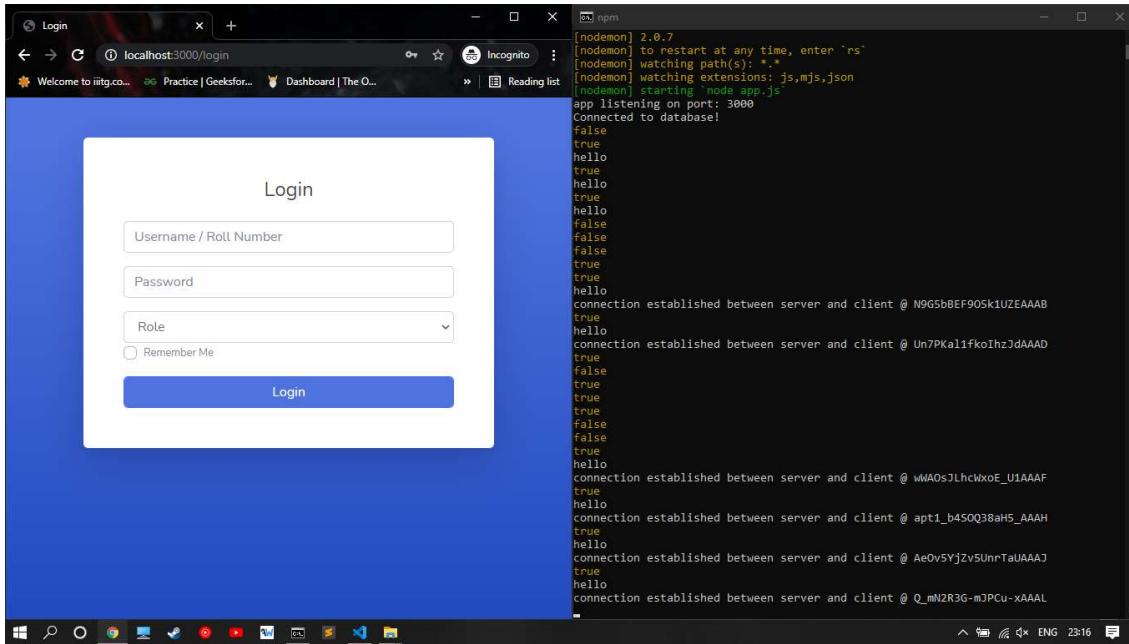
Role : Admin

Expected o/p : No Login

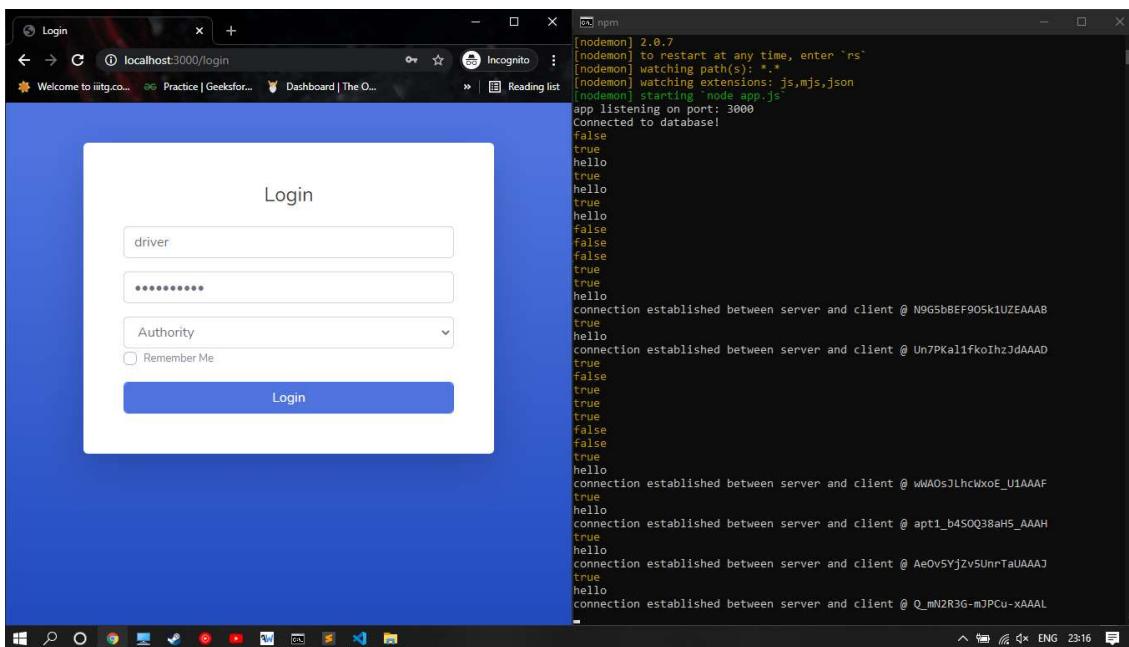
Output :



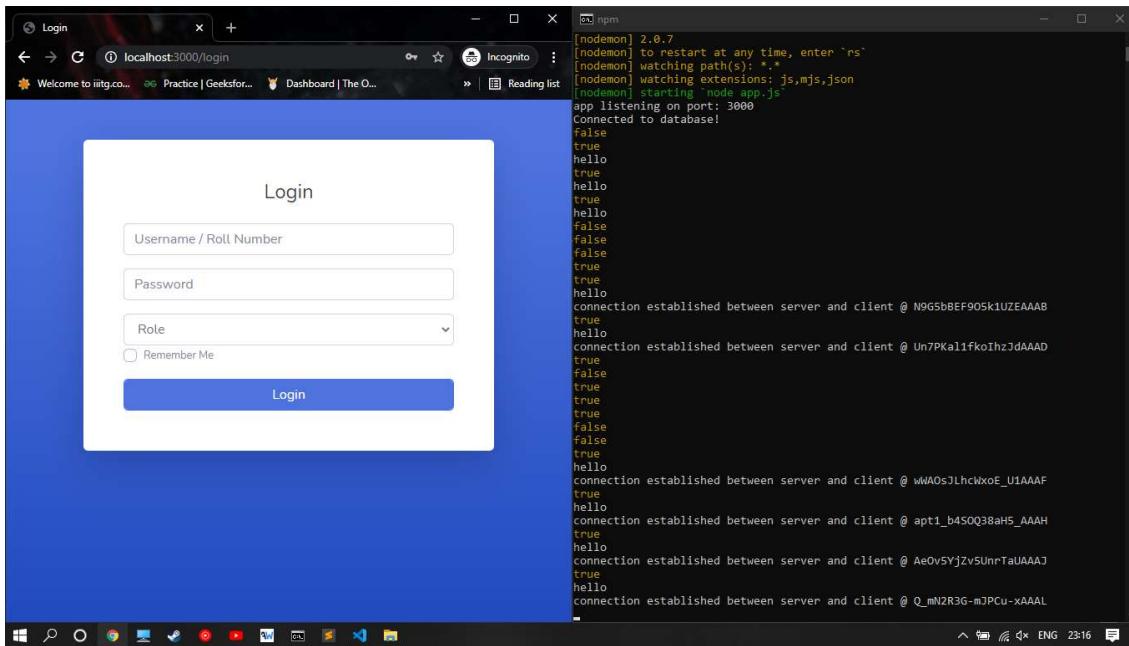
No Login :



Username : driver
Role : Authority
Expected o/p : No Login
Output :



No Login :

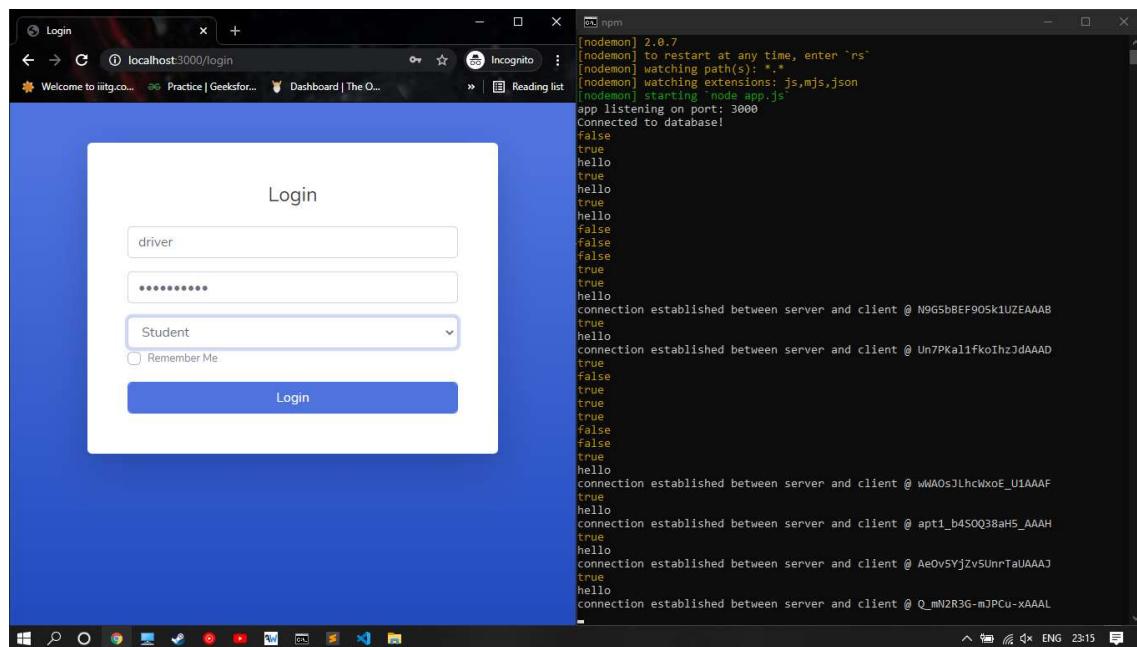


Username : driver

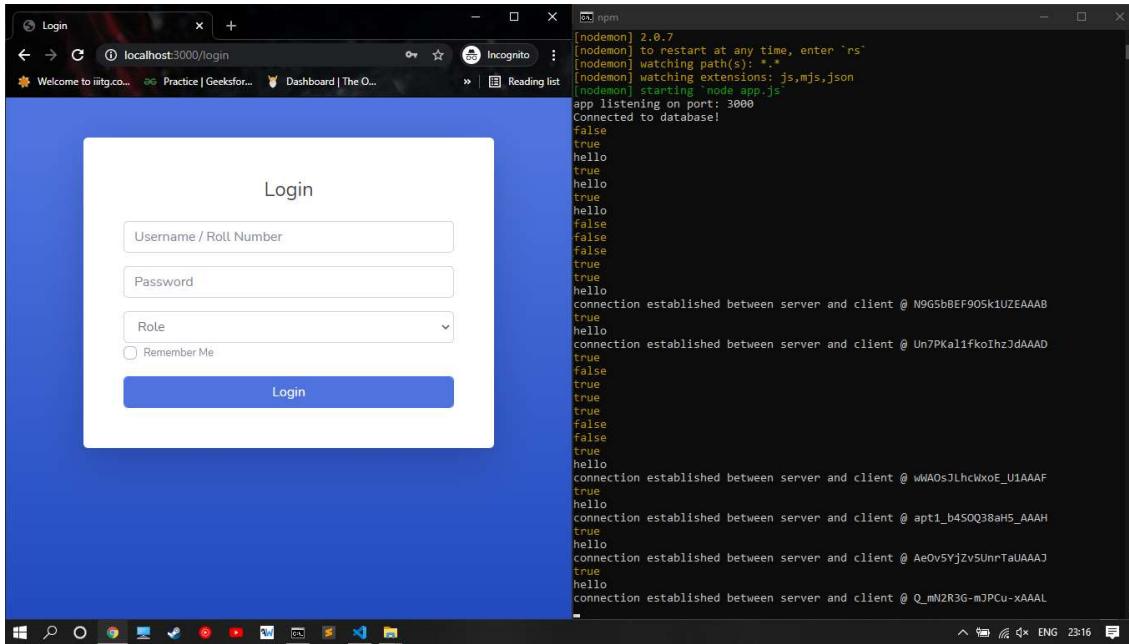
Role : Student

Expected o/p : No Login

Output :



No Login :



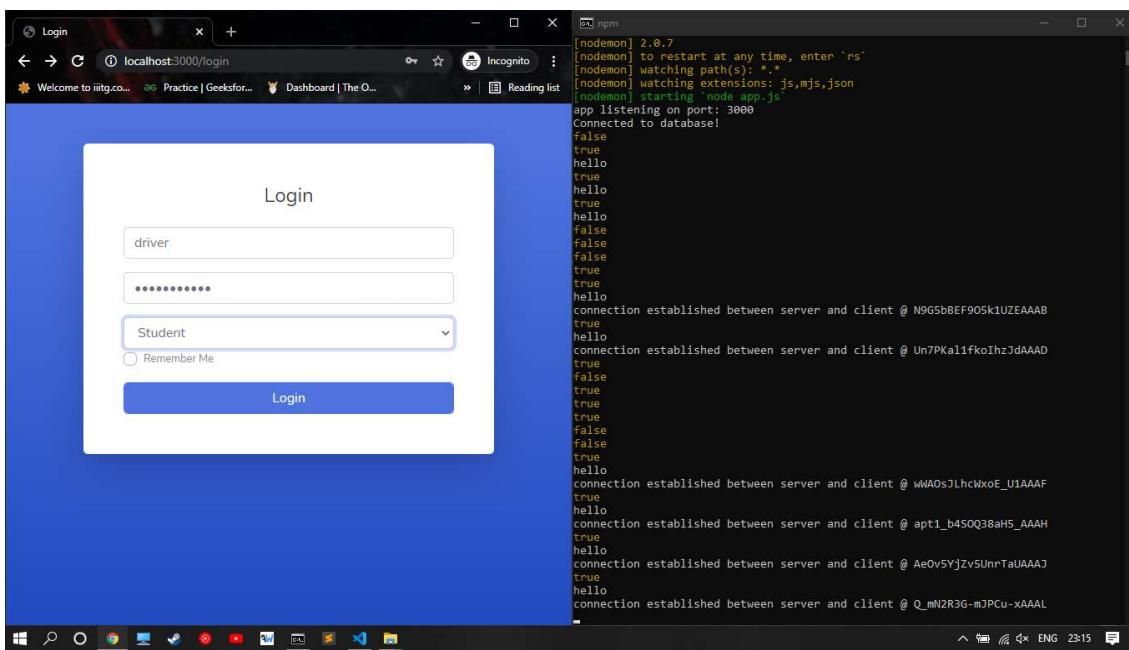
Case 3: Correct Username but Incorrect password and role

Username : driver

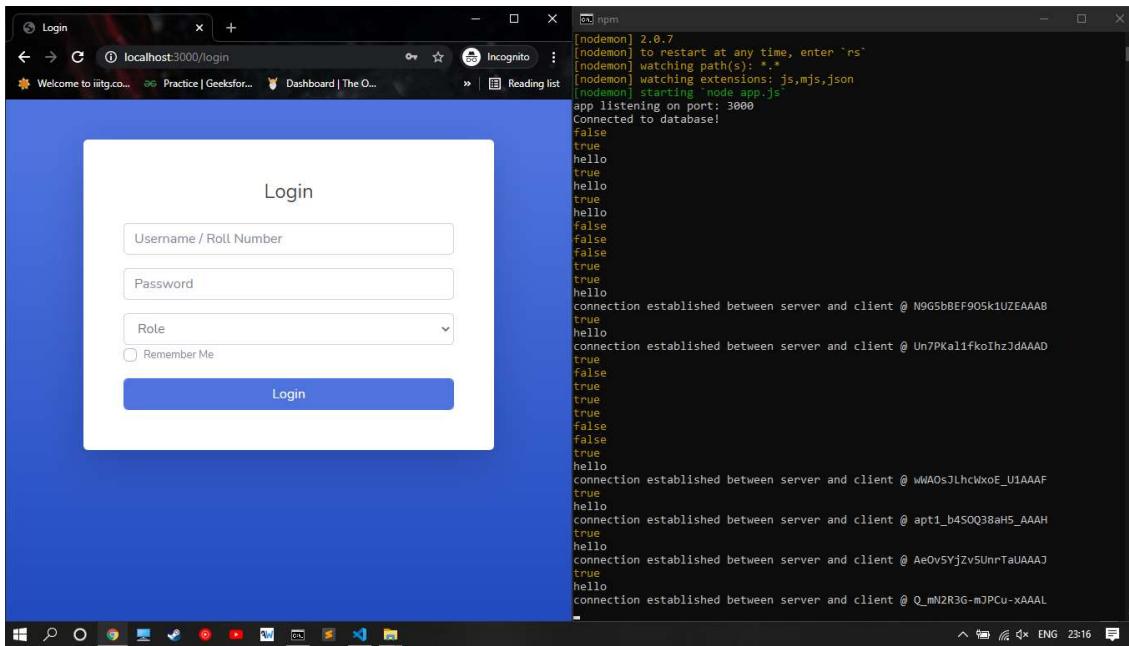
Role : Student

Expected o/p : No Login

Output :

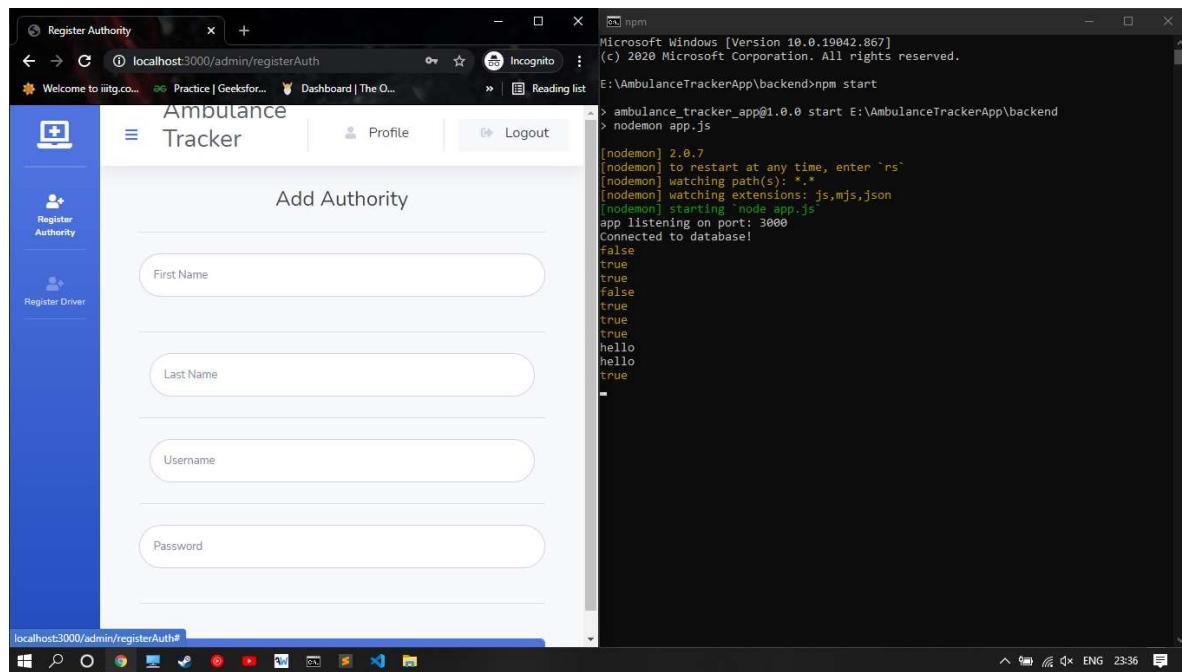


No Login :

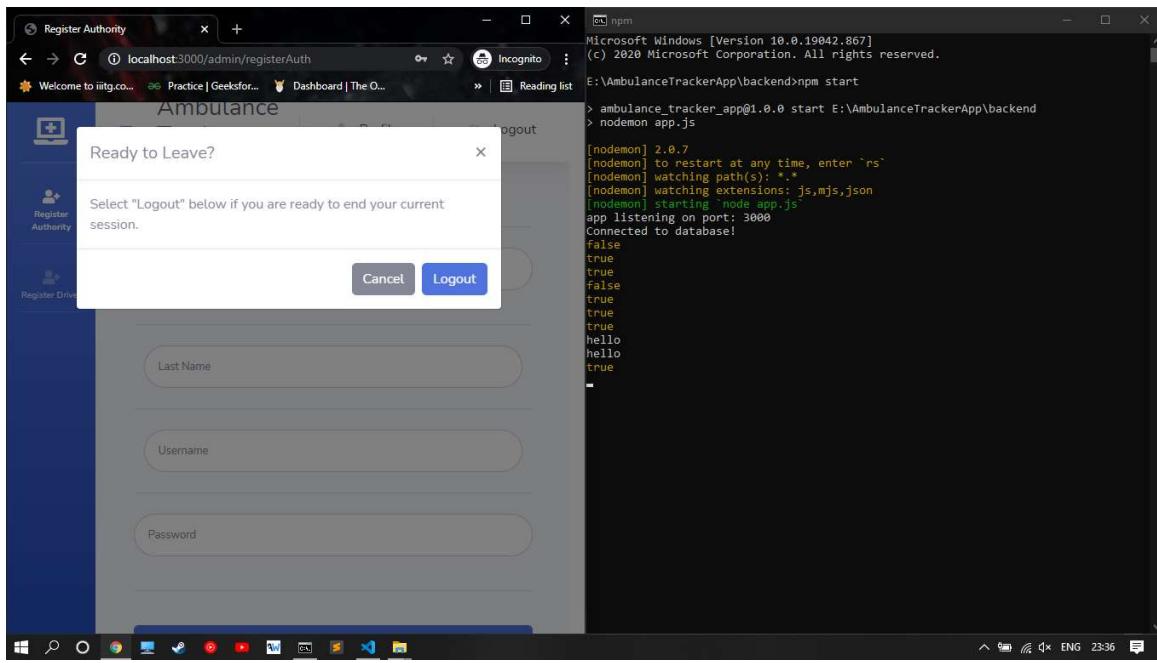


Logout -

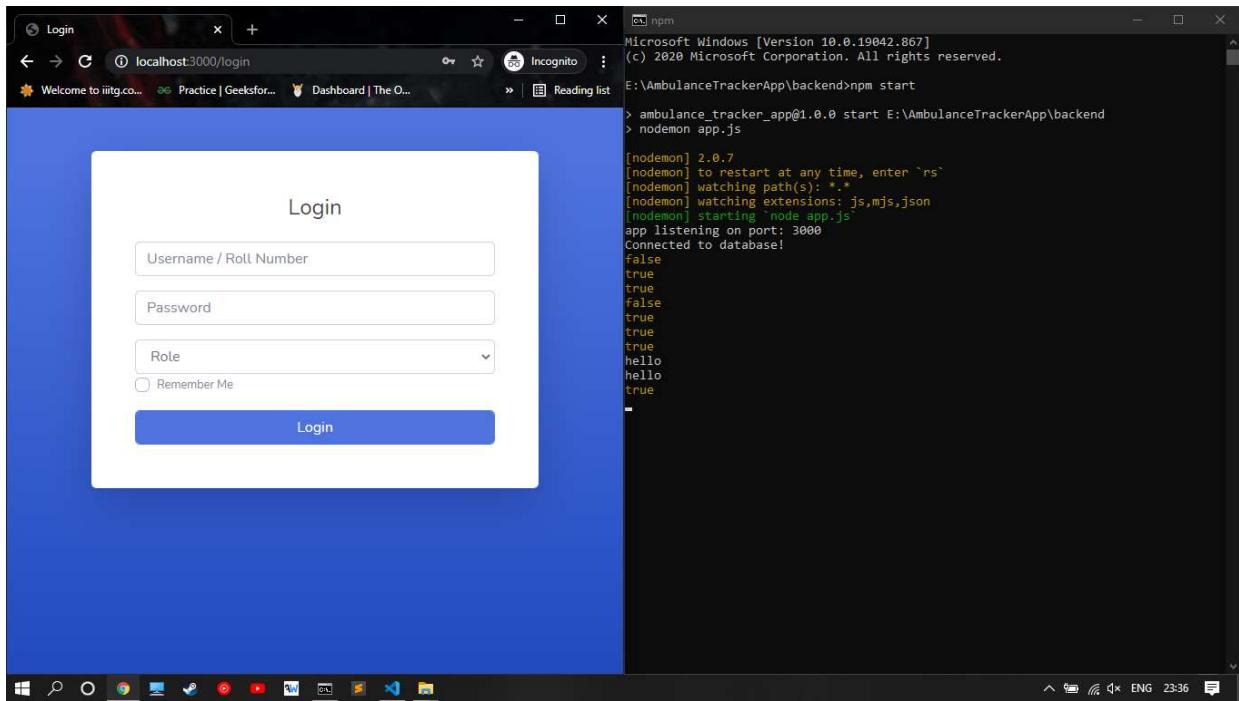
Logout of Admin :



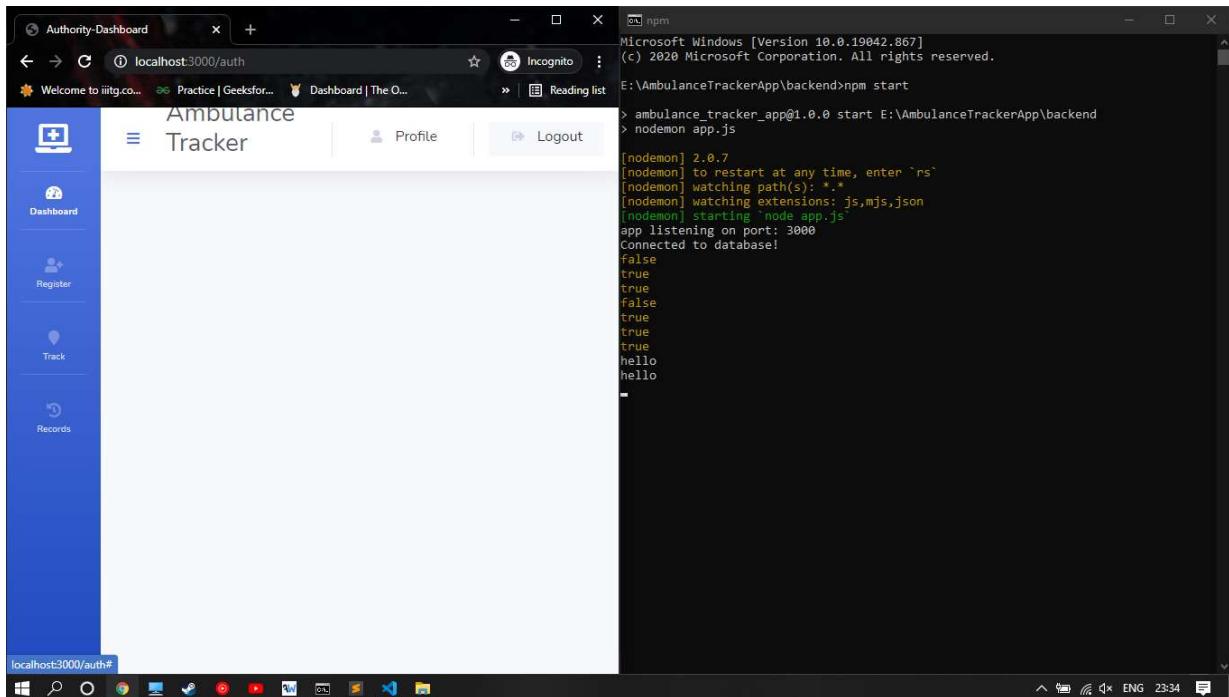
Logout Popup :



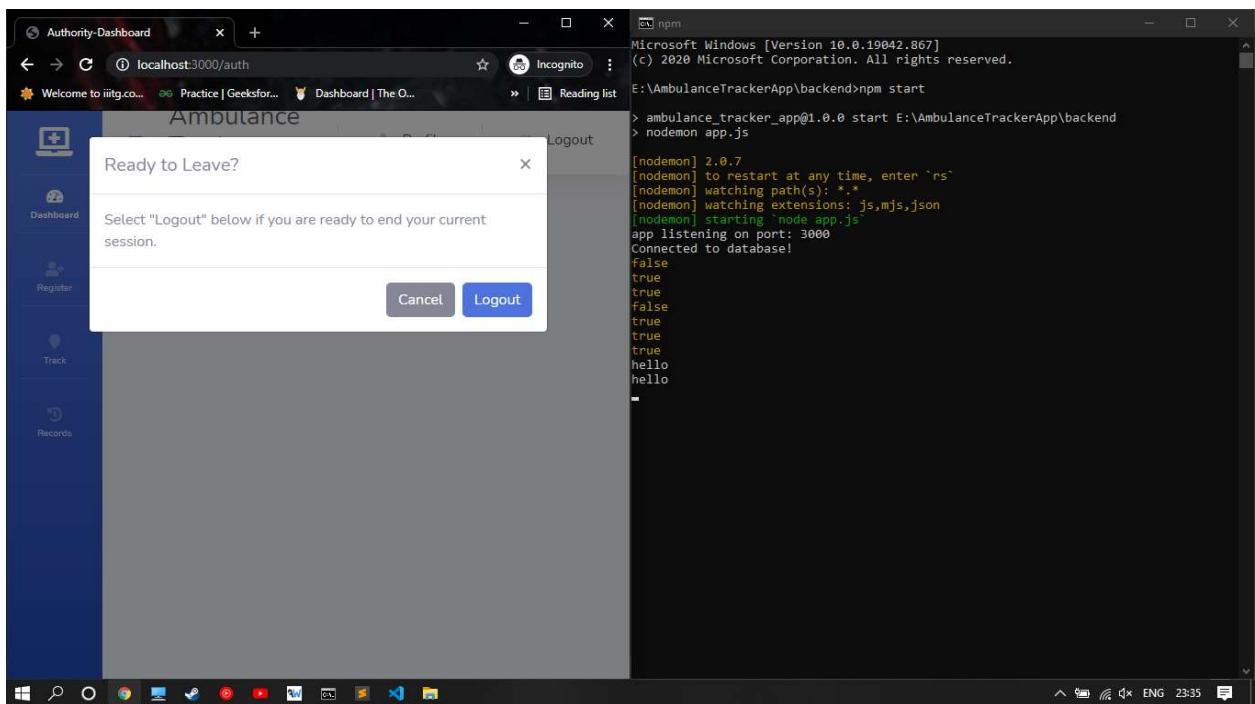
Successfully Logged Out :



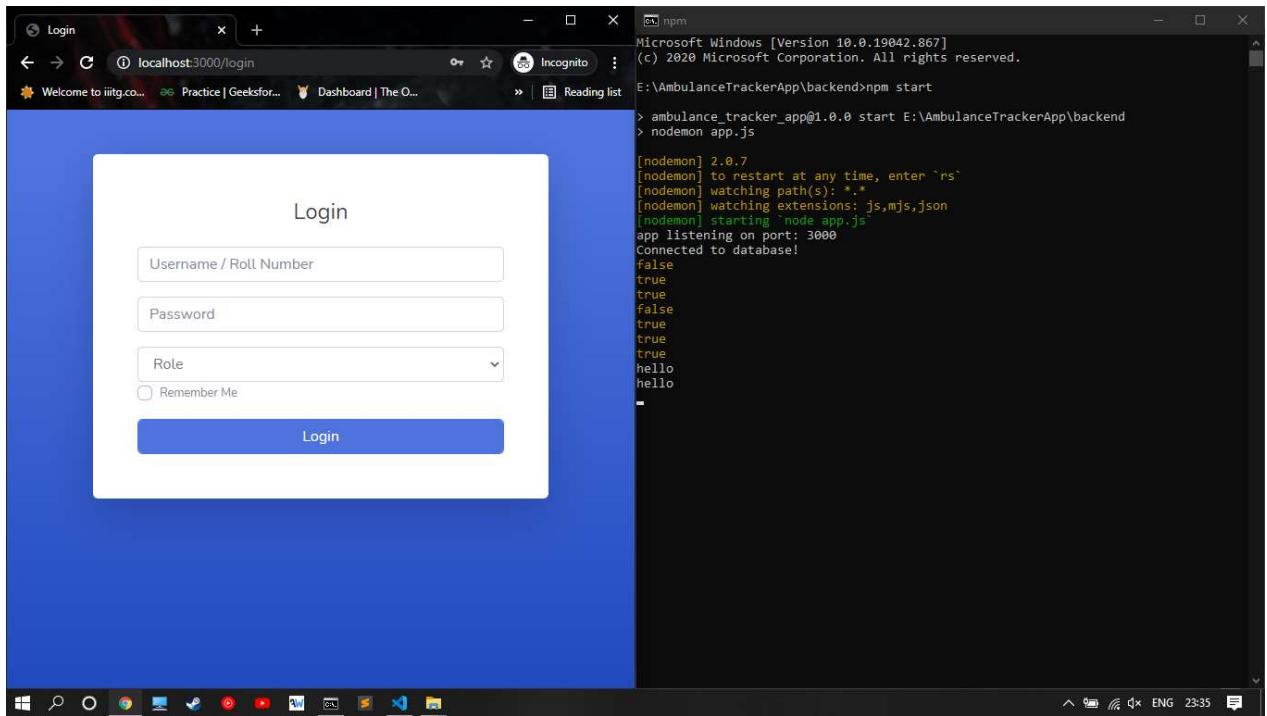
Logout of Authority :



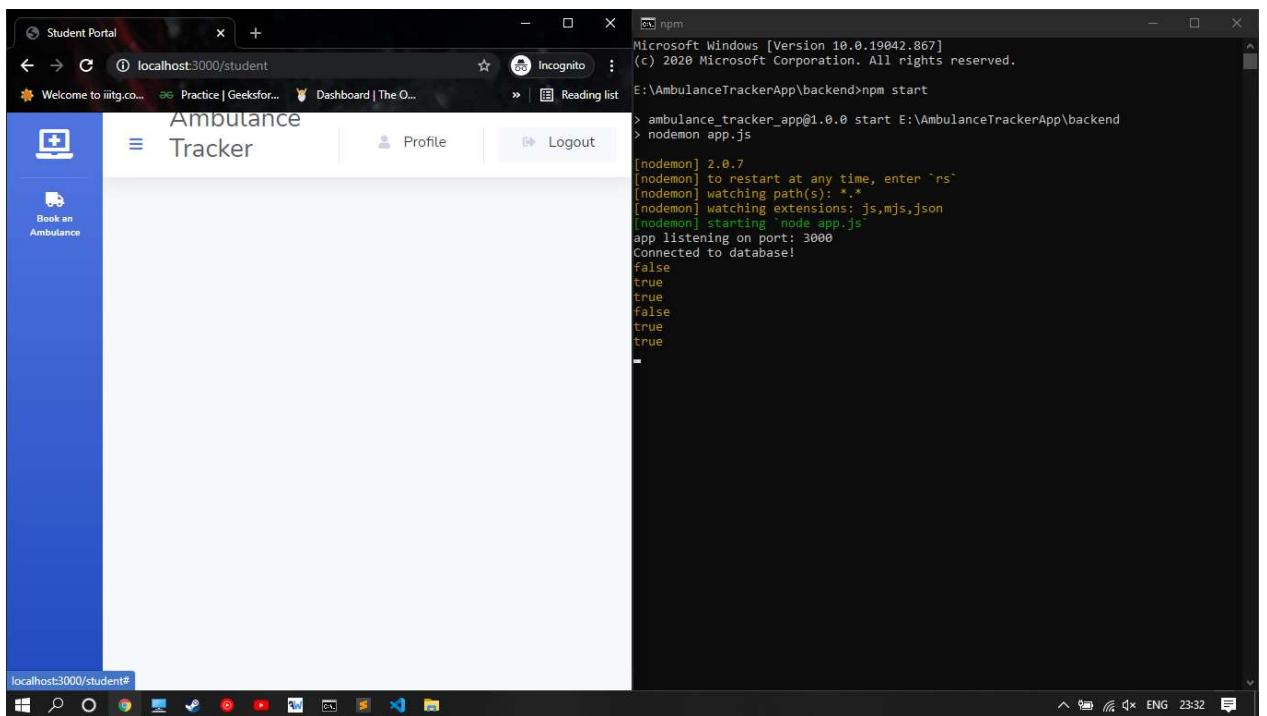
Logout Popup :



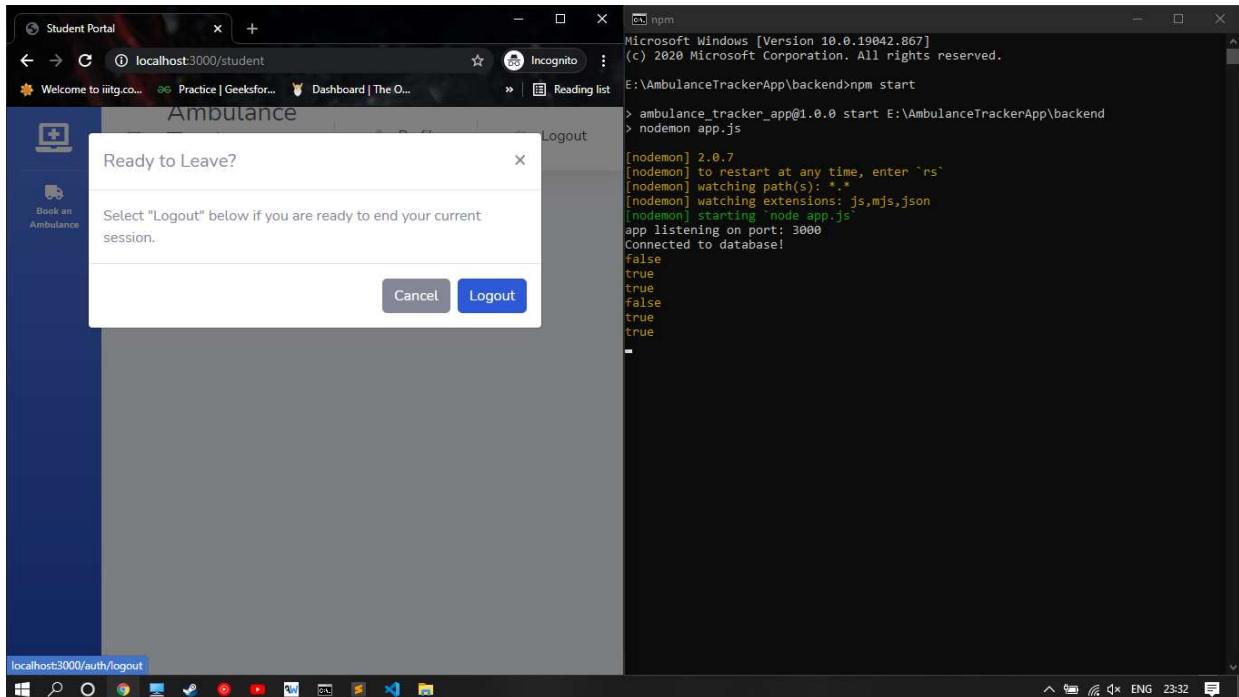
Successfully Logged Out :



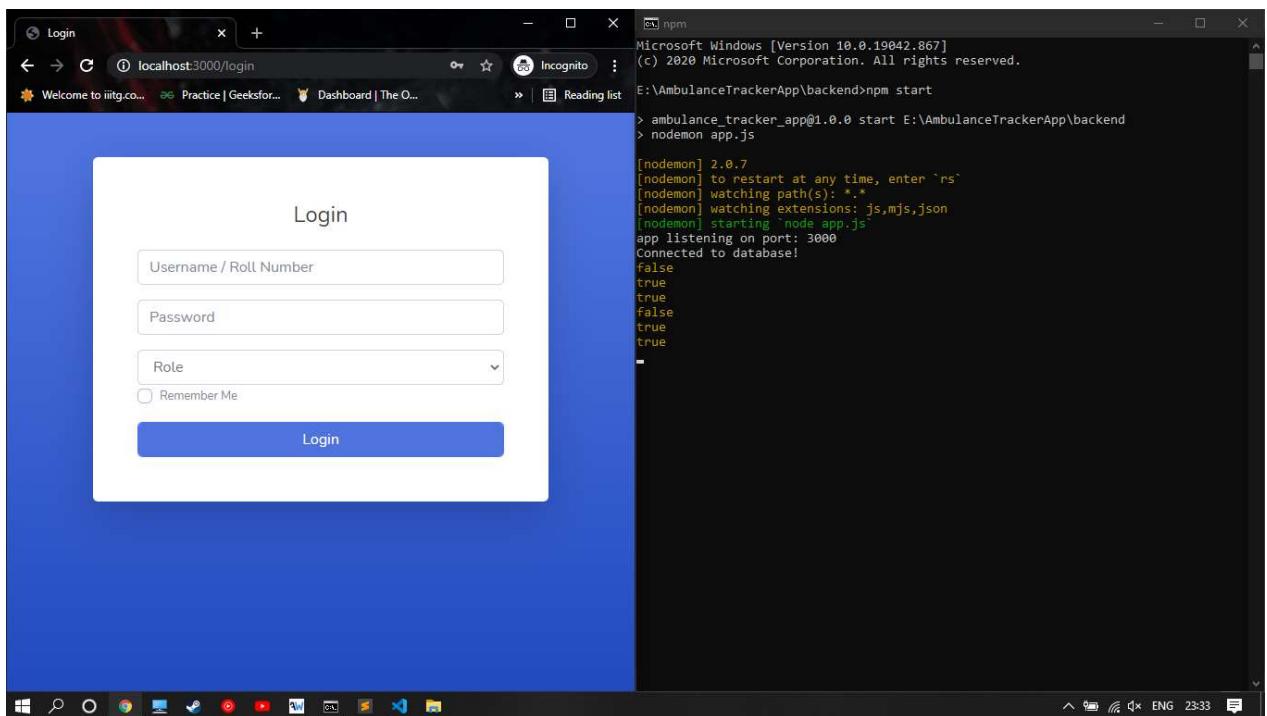
Logout of Student :



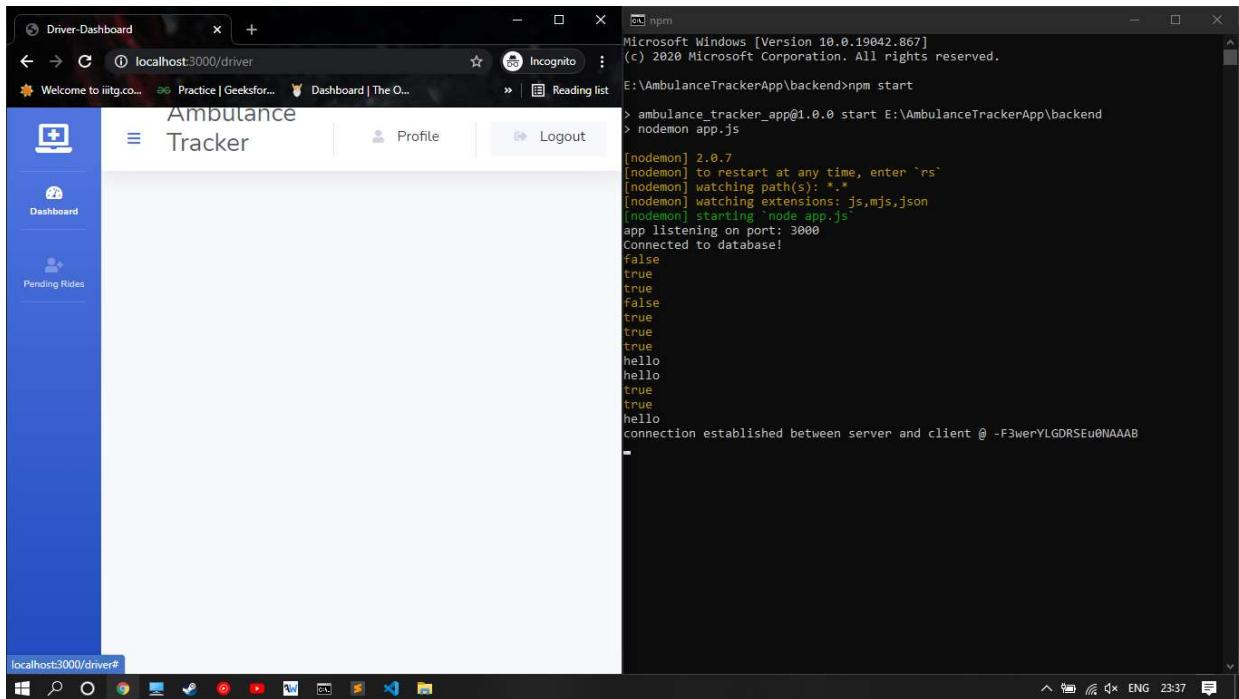
Logout Popup :



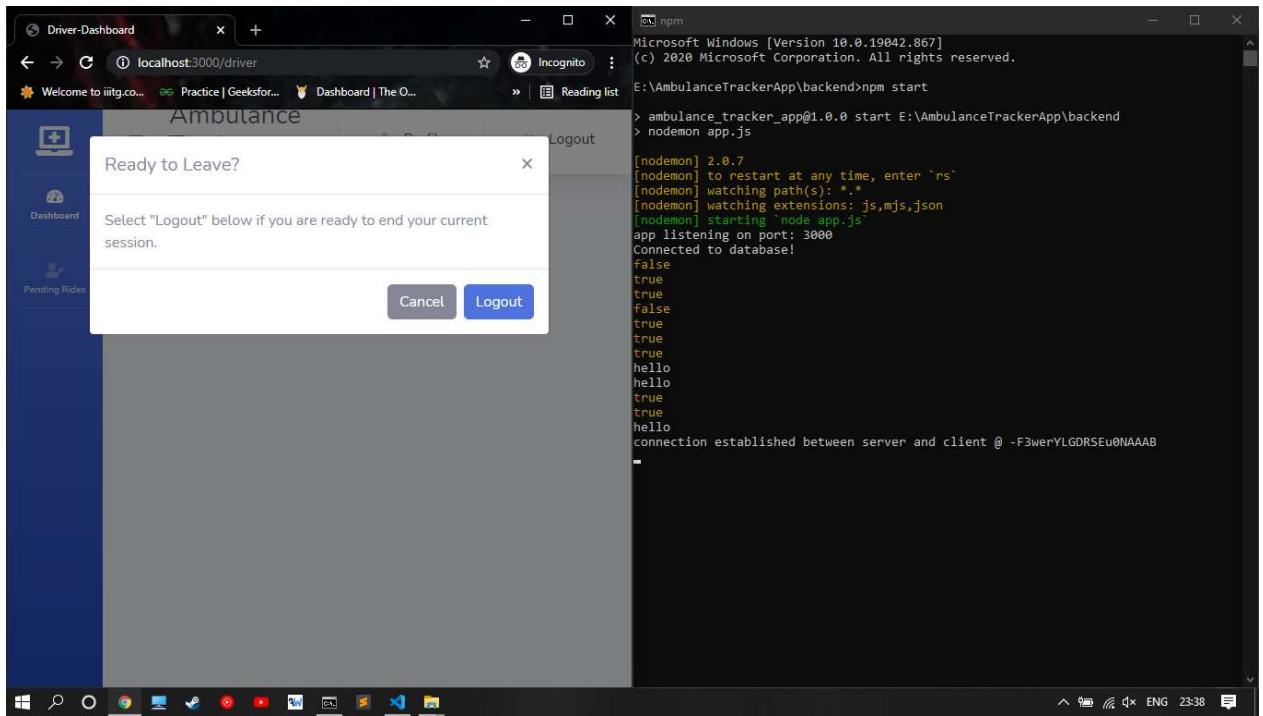
Successfully Logged Out :



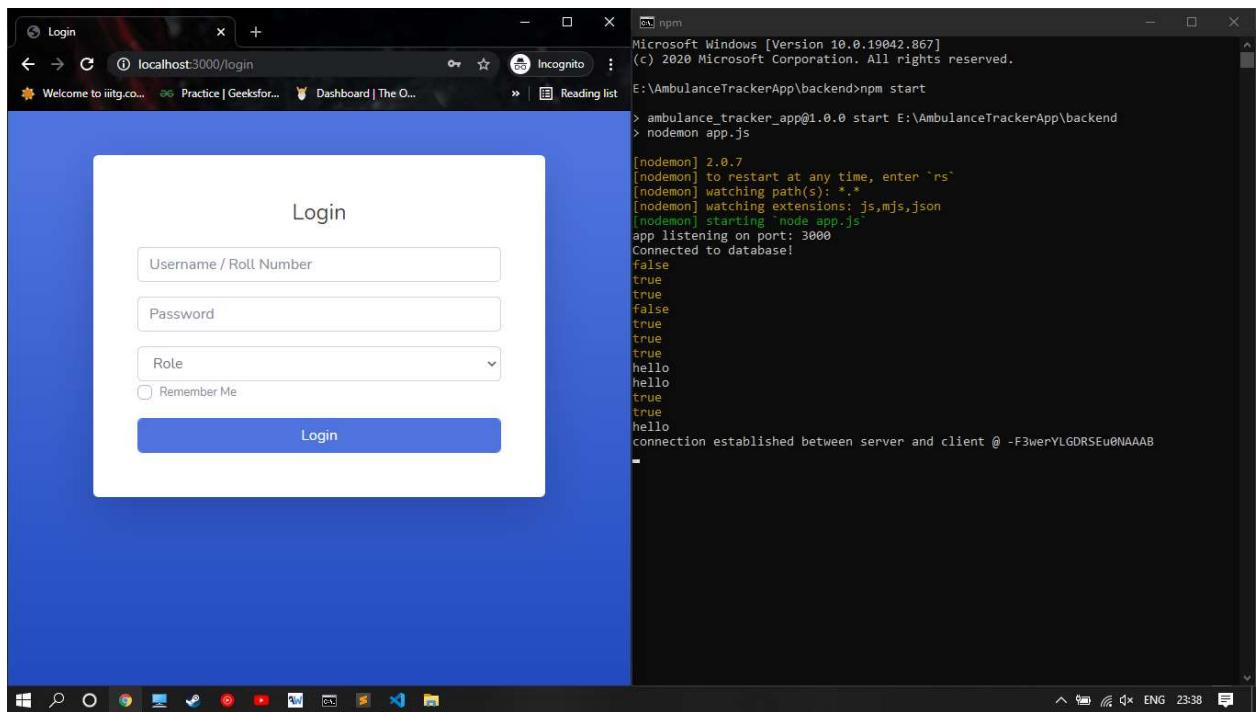
Logout of Driver :



Logout Popup :



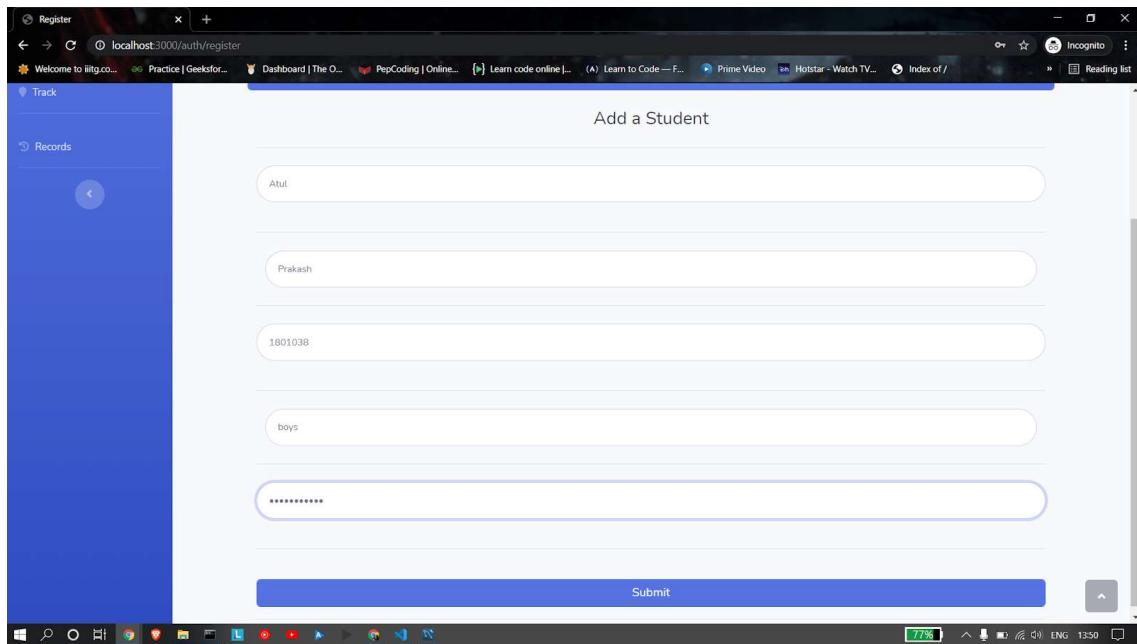
Successfully Logged Out :



2. Authorities can register new users :

Case 1: Correct Input

Input :



Expected Result : User should be successfully registered

Result :

User Added -

The screenshot shows the MySQL Workbench interface. On the left, there's a sidebar with various management and performance tabs like 'Server Status', 'Performance Reports', and 'Schemas'. The main area has a 'Query 1' tab open with the following SQL query:

```
1 * select * from student;
```

The results grid shows the following data:

| rollnumber | fname | lname | hostel | passer |
|------------|---------|---------|--------|-----------|
| 1801038 | Atul | Prakash | boys | \$20\$10 |
| 1801045 | student | student | boys | \$20\$11C |
| 18001109 | naveen | goel | boys | \$20\$11C |
| 18001669 | shobhit | pryag | boys | \$20\$11C |

The 'Output' pane at the bottom shows the execution log:

| # | Action | Message | Duration / Fetch |
|---|-------------------------------------|-------------------|-----------------------|
| 1 | use ambulance | 0 row(s) affected | 0.047 sec |
| 2 | show tables | 6 row(s) returned | 0.047 sec / 0.000 sec |
| 3 | select * from student LIMIT 0, 1000 | 3 row(s) returned | 0.046 sec / 0.000 sec |
| 4 | select * from student LIMIT 0, 1000 | 4 row(s) returned | 0.047 sec / 0.000 sec |

Input :

The screenshot shows a web browser window with the URL `localhost:3000/auth/register#`. The page title is 'Add a Student'. There are five input fields filled with the following values:

- fname: Diwakar
- lname: Bharti
- rollnumber: 1801059
- hostel: boys
- passer: *****

A large blue 'Submit' button is located at the bottom of the form.

Before Adding :

The screenshot shows the MySQL Workbench interface. The left sidebar has a 'Registers' section with 'Records'. The main area shows the 'student' table in the 'student' schema. A query window displays the following SQL and results:

```
1 * select * from student;
```

| rollnumber | fname | lname | hostel | password |
|------------|---------|---------|--------|---|
| 1801038 | Atul | Prakash | boys | \$2b\$10\$G2tohB7Kvbg69N6nop7.RV75nZ6z8... |
| 1801045 | student | student | boys | \$2b\$10\$KWCqfFjQprJMtz4tUmPE.WzBsp6g7f... |
| 1801109 | namit | goel | boys | \$2b\$10\$Amo.eCgbXlaaa4iyOs.hQygpa7m... |
| 1801169 | shobhit | singh | boys | \$2b\$10\$PdvU8Lopmg7tQnTodBe54Wiv2m8... |
| 1801170 | ss | ss | ss | ss |

The status bar at the bottom right shows '70%' and 'ENG 14:02'.

Expected Result : User should be successfully registered

Result :

User Added -

The screenshot shows the MySQL Workbench interface after a new user has been added. The left sidebar now includes a 'Register' section under 'User'. The main area shows the 'student' table with the new user added:

| rollnumber | fname | lname | hostel | password |
|------------|---------|---------|--------|---|
| 1801038 | Atul | Prakash | boys | \$2b\$10\$G2tohB7Kvbg69N6nop7.RV75nZ6z8... |
| 1801045 | student | student | boys | \$2b\$10\$KWCqfFjQprJMtz4tUmPE.WzBsp6g7f... |
| 1801109 | namit | goel | boys | \$2b\$10\$Amo.eCgbXlaaa4iyOs.hQygpa7m... |
| 1801169 | shobhit | singh | boys | \$2b\$10\$PdvU8Lopmg7tQnTodBe54Wiv2m8... |
| 1801170 | ss | ss | ss | ss |
| 1801171 | ss | ss | ss | ss |

The status bar at the bottom right shows '69%' and 'ENG 14:04'.

Case 2: Wrong Input

Input :

The screenshot shows a web browser window with the URL `localhost:5000/auth/register`. The page title is "Add a Student". There are five input fields:

- rollnumber: 1801027
- fname: xyz
- lname: abc
- gender: boys
- password: *****

A blue "Submit" button is located at the bottom of the form.

Expected Result : User should not be added

Result :

User not added -

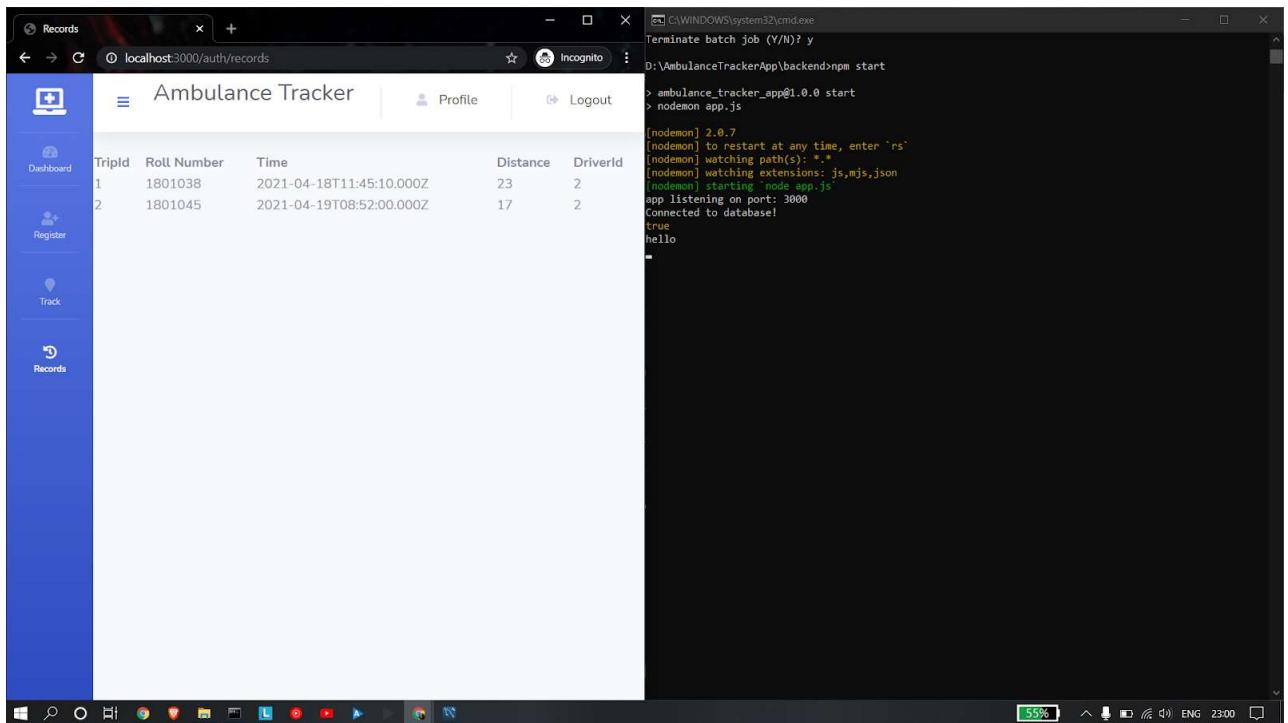
The screenshot shows the MySQL Workbench interface. In the top navigation bar, the database is set to "Ambulance". The left sidebar shows various management and performance tabs. The main area displays a query editor with the SQL command:

```
1 * select * from student;
```

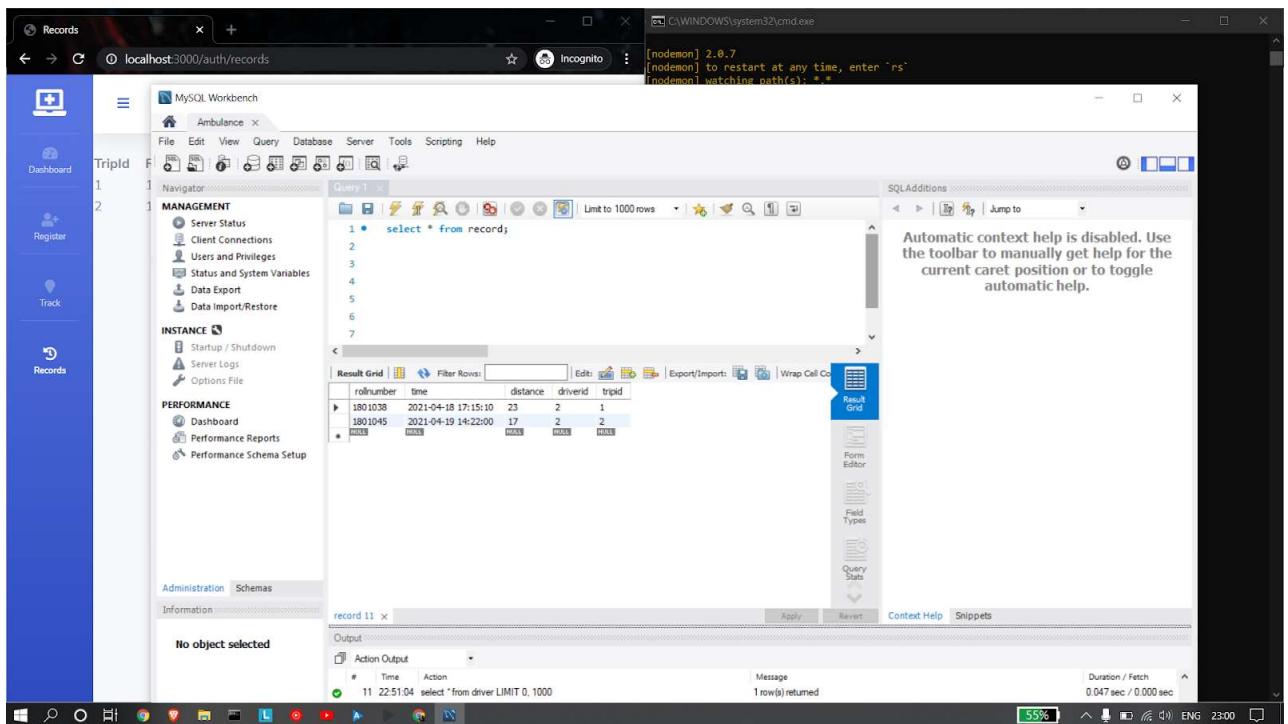
The results grid shows 18 rows of student data. The "Output" tab at the bottom displays the execution history:

| Action | Time | Message | Duration / Fetch |
|--------|----------|---------------------------------------|-----------------------|
| 10 | 13:59:58 | select * from driver LIMIT 0, 1000 | 0.032 sec / 0.000 sec |
| 11 | 14:01:22 | select * from authority LIMIT 0, 1000 | 0.047 sec / 0.000 sec |
| 12 | 14:01:59 | select * from student LIMIT 0, 1000 | 0.047 sec / 0.000 sec |
| 13 | 14:04:26 | select * from student LIMIT 0, 1000 | 0.046 sec / 0.000 sec |

3. Authorities will have full access to the records :



Records table :



4. Separate interface for Authority, Students and Drivers :

Admin :

Ambulance
Tracker

[Logout](#)

Add Authority

First Name

Last Name

Username

Password

Submit

Ambulance
Tracker

[Logout](#)

Add Driver

First Name

Last Name

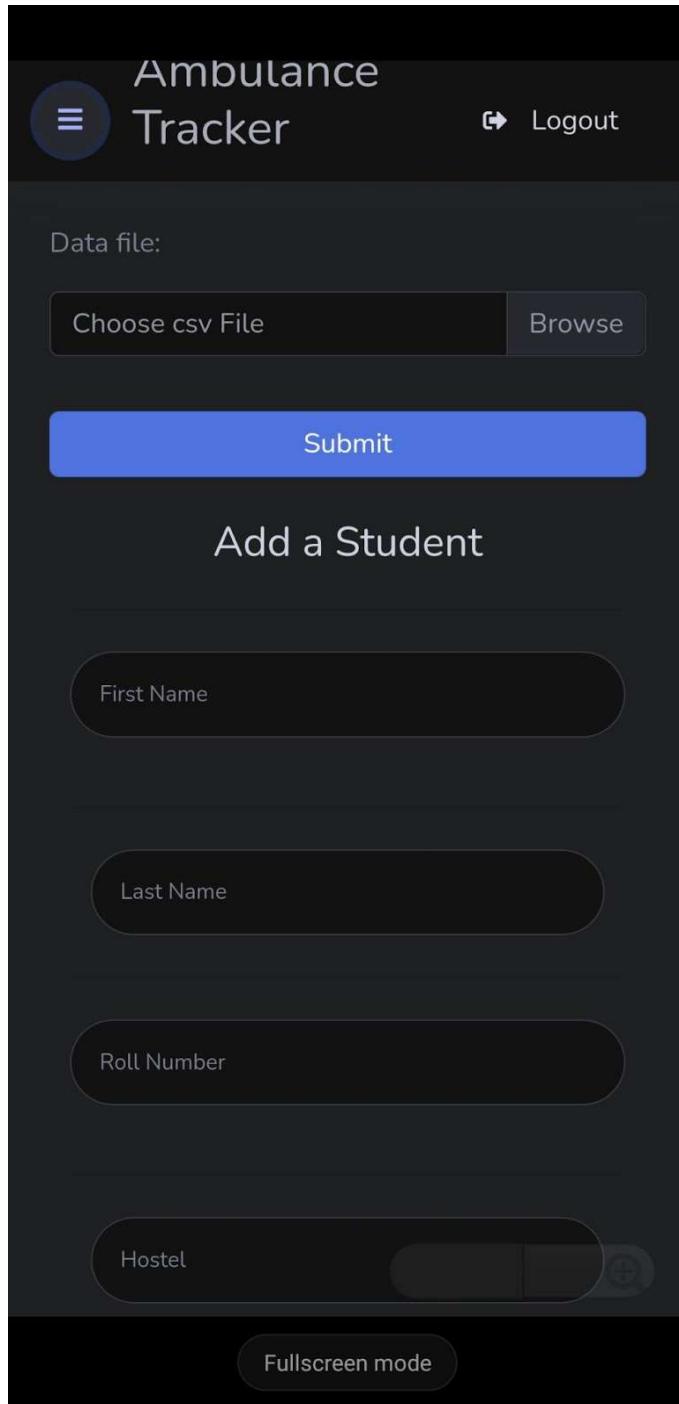
Username

Password

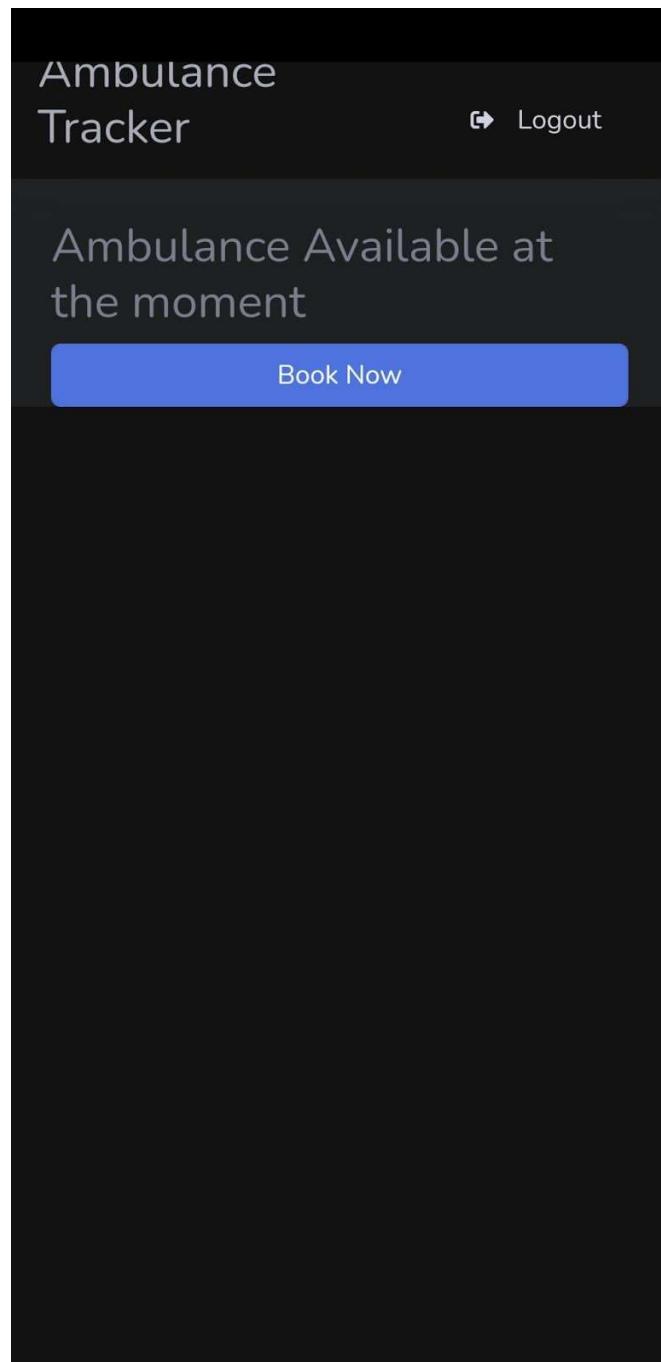
Mobile

Submit

Authority :



Student :

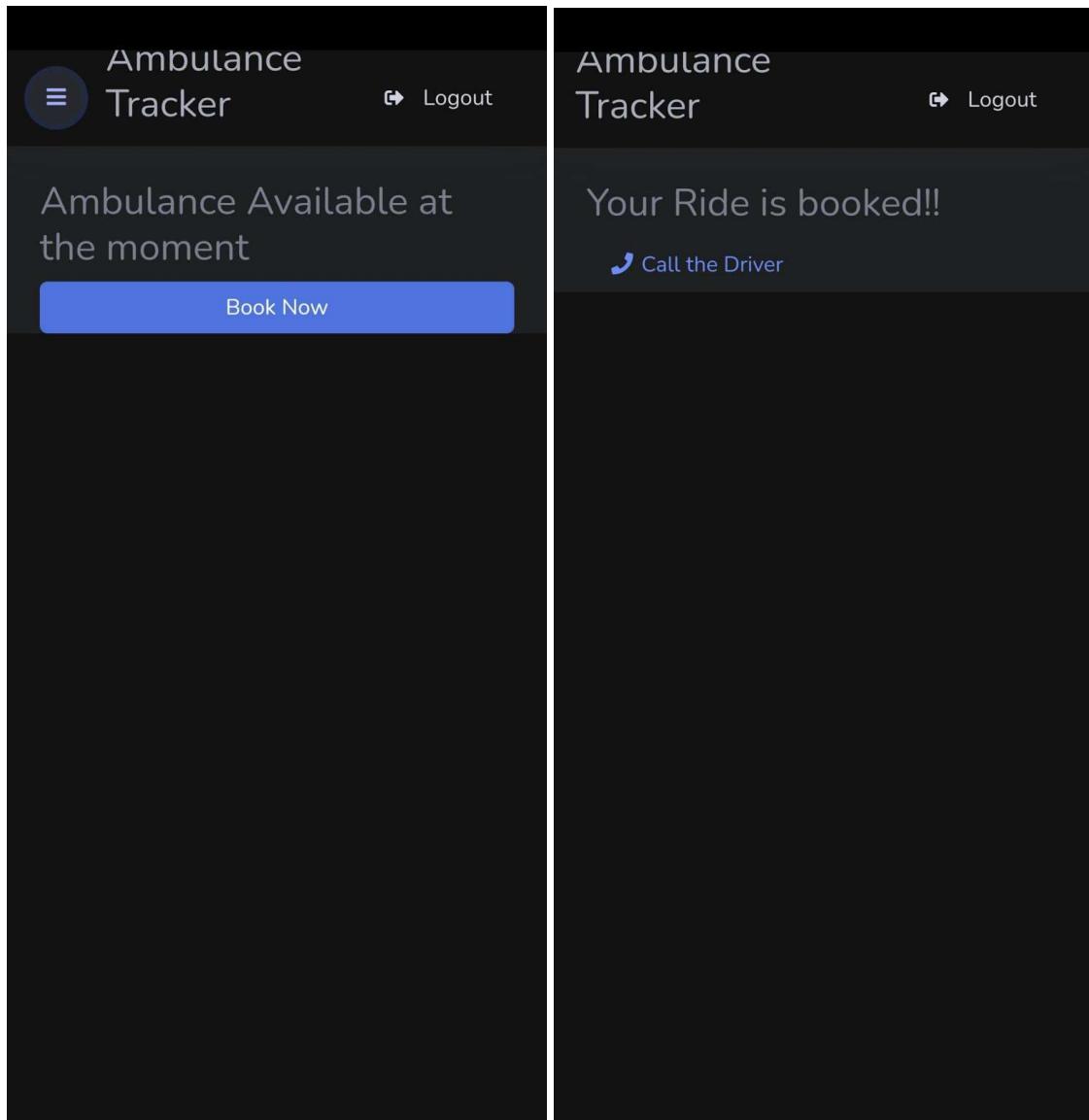


Driver :

The screenshot shows a mobile application titled "Ambulance Tracker". The top navigation bar includes the app's name and a "Logout" button. Below this is a table with columns: Name, Hostel, Roll no., Mobile, and a blank column represented by "...". A single row is visible, showing "chayan" in the Name column, "boys" in the Hostel column, "1801045" in the Roll no. column, and "7894561230" in the Mobile column. To the right of this row are three icons: a blue dot, a blue phone receiver, and the word "Call". At the bottom of the screen is a horizontal search bar containing a magnifying glass icon and a plus sign icon.

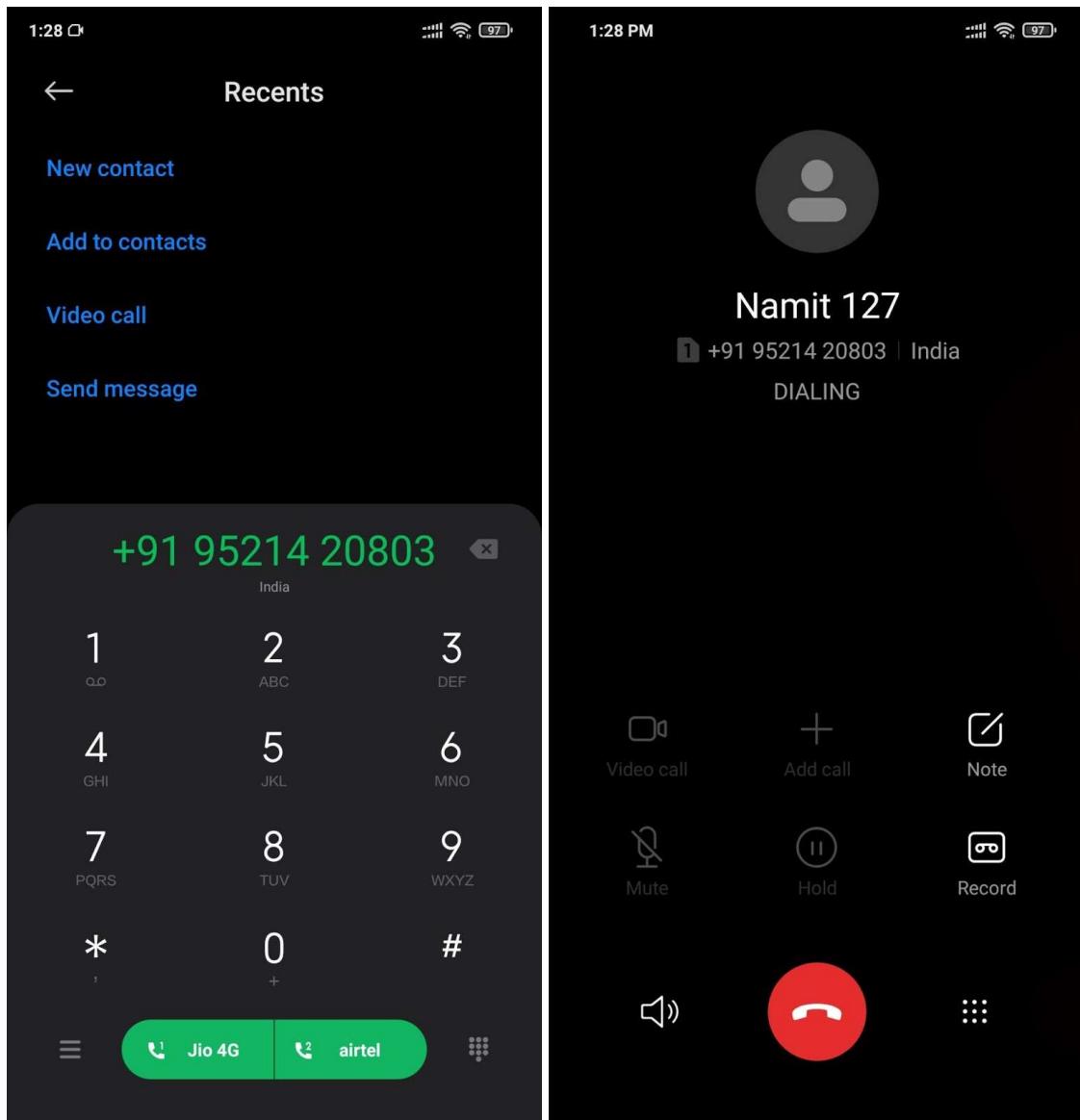
| Name | Hostel | Roll no. | Mobile | |
|--------|--------|----------|------------|-----------|
| chayan | boys | 1801045 | 7894561230 | • Call |

5. Students can check the availability of the ambulance in case of an emergency and book it or call the driver :



After clicking the Book now button, The option to call the driver will be available if the ambulance is available.

After Call the Driver, it will automatically redirect to the calling option to call the driver.



Checking the Availability of the Ambulance :

Ambulance Tracker

Logout

Check Availability

Ambulance Tracker

Logout

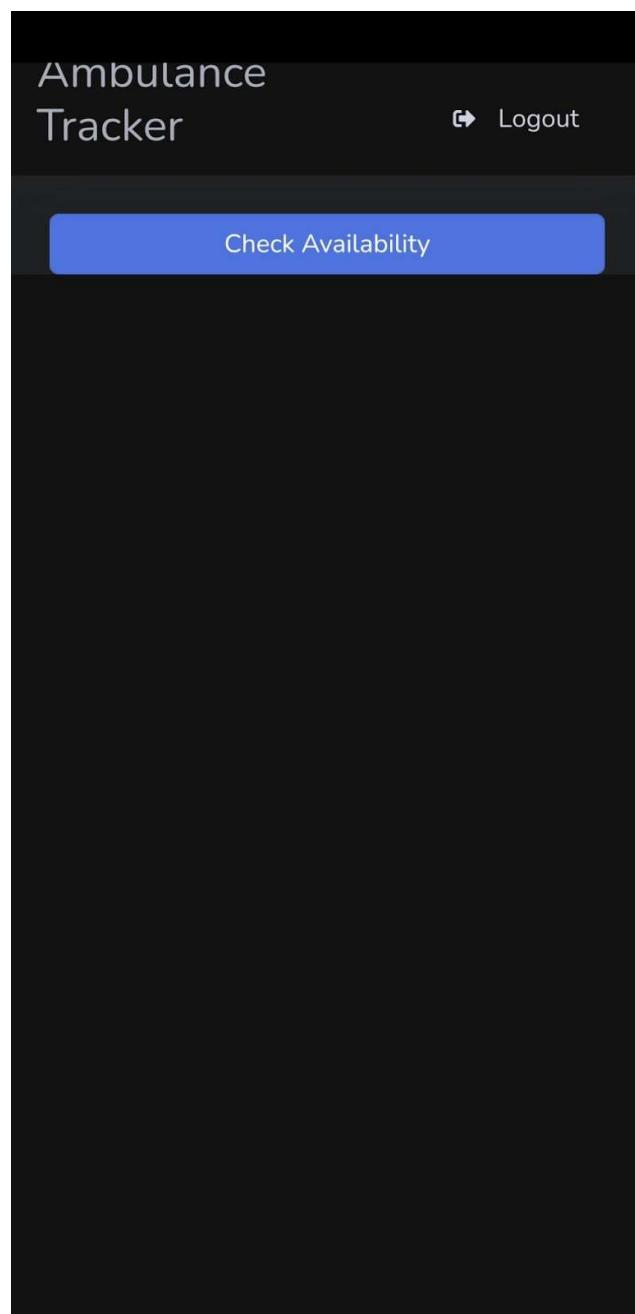
Home

Ambulance Available at the moment

Book Now

6. After a student's request for an ambulance, the driver can initiate and end the ride upon the completion of which a POST request would be generated to the server with appropriate information accordingly :

Student :

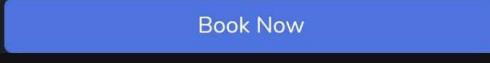


Ambulance Tracker

 Logout

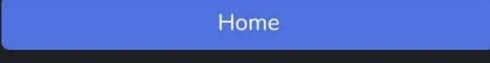
 Home

Ambulance Available at
the moment

 Book Now

Ambulance Tracker

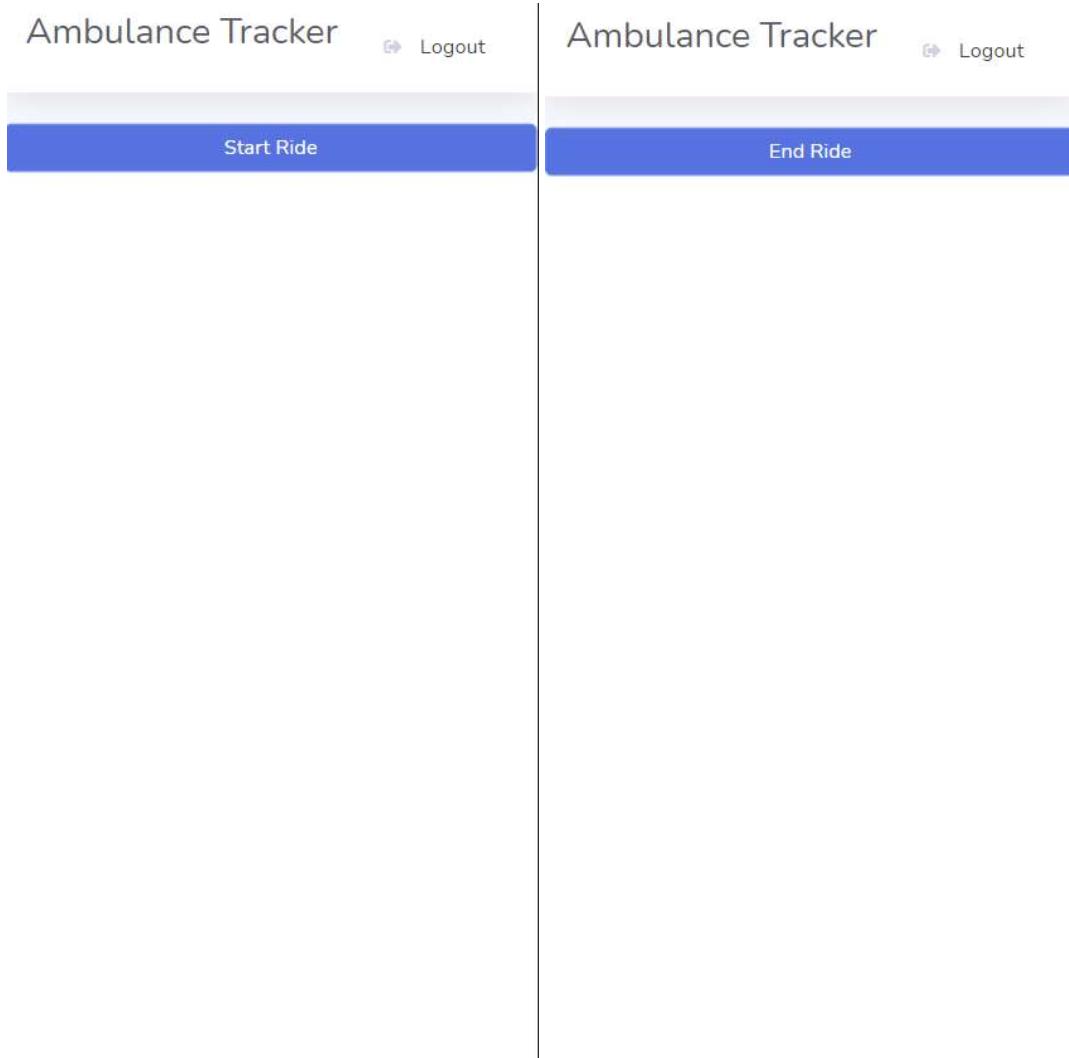
 Logout

 Home

Your Ride is booked!!

 Call the Driver

Driver :



If the ambulance is available then the Driver will get the option to start the ride and as soon as the Driver presses End Ride, he can start a new ride from the pending rides.

When the ambulance is unavailable, the student will be displayed the unavailability of the ambulance as shown below :

Student :

Ambulance Tracker Logout

[Home](#)

Ambulance is not Available at the moment

[Call the Driver](#)

[Book For Later](#)

Ambulance Tracker Logout

[Home](#)

Atul

1801038

boys

7992299306

[Book](#)

Driver :

The image displays two side-by-side screenshots of a mobile application titled "Ambulance Tracker".

Screenshot 1 (Left): This screen shows a list of passengers. The columns are labeled "Name", "Hostel", "Roll no.", "Mobile", and ".....". A single row is visible for "Atul", who is listed as "boys" in the Hostel column, with "1801038" in Roll no., "7992299306" in Mobile, and a red dot icon in the column. To the right of the mobile number is a blue phone icon with the word "Call" below it.

Screenshot 2 (Right): This screen shows the same passenger information for "Atul". Below the list is a large blue button with the white text "Start Ride".

Ambulance Tracker

 Logout

Name Hostel Roll no. Mobile
Atul boys 1801038 7992299306 •



 End Ride

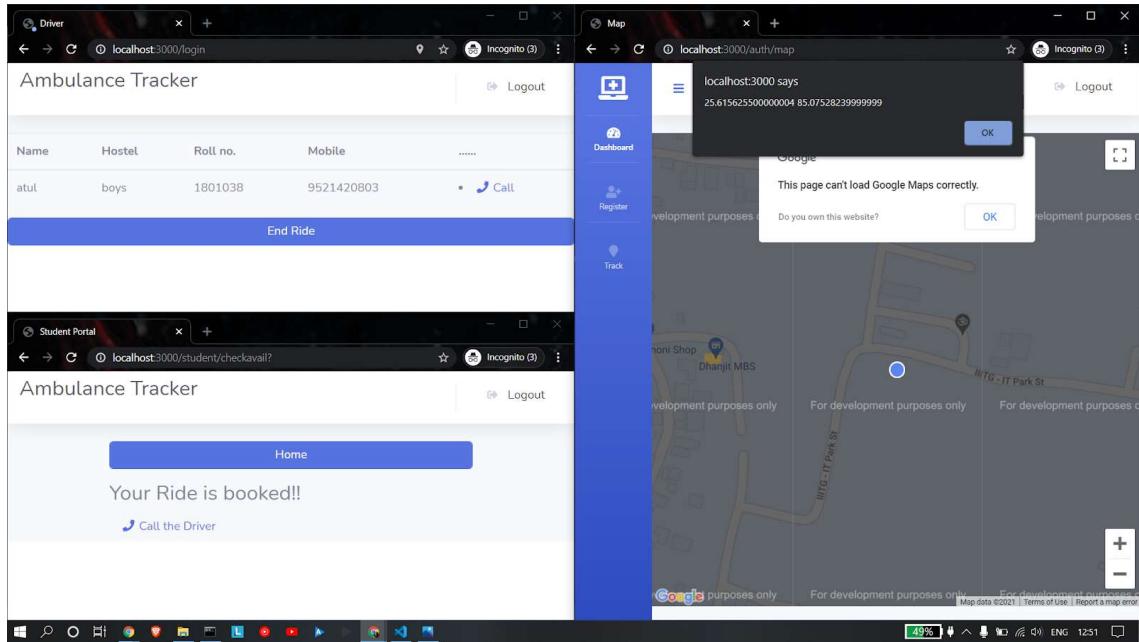
Ambulance Tracker

 Logout

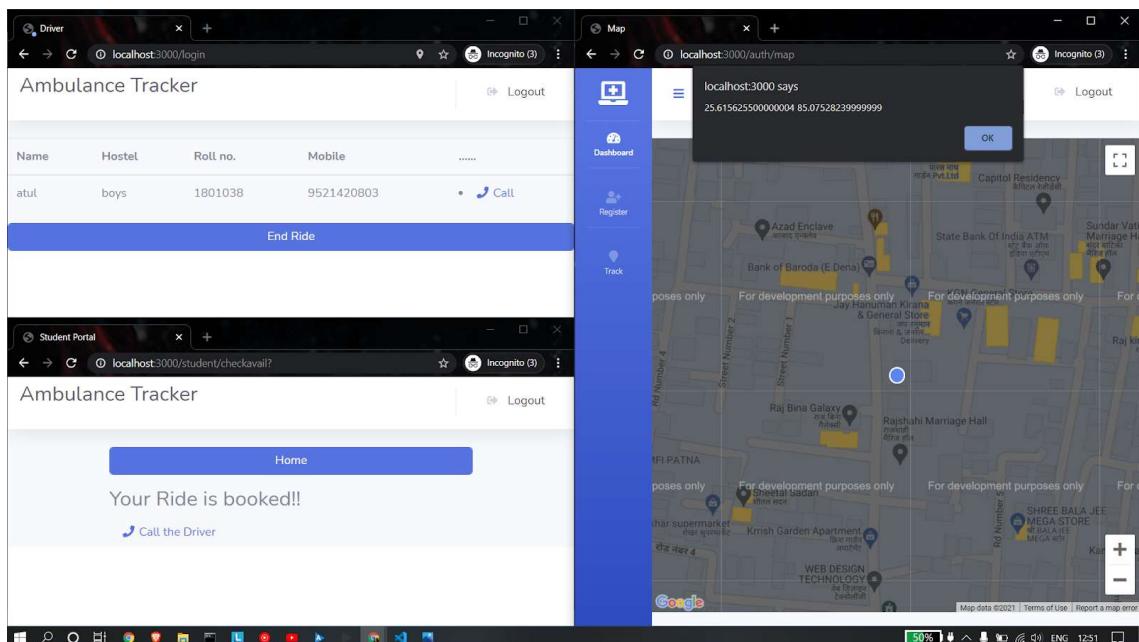
No Rides For Now!!!!!!

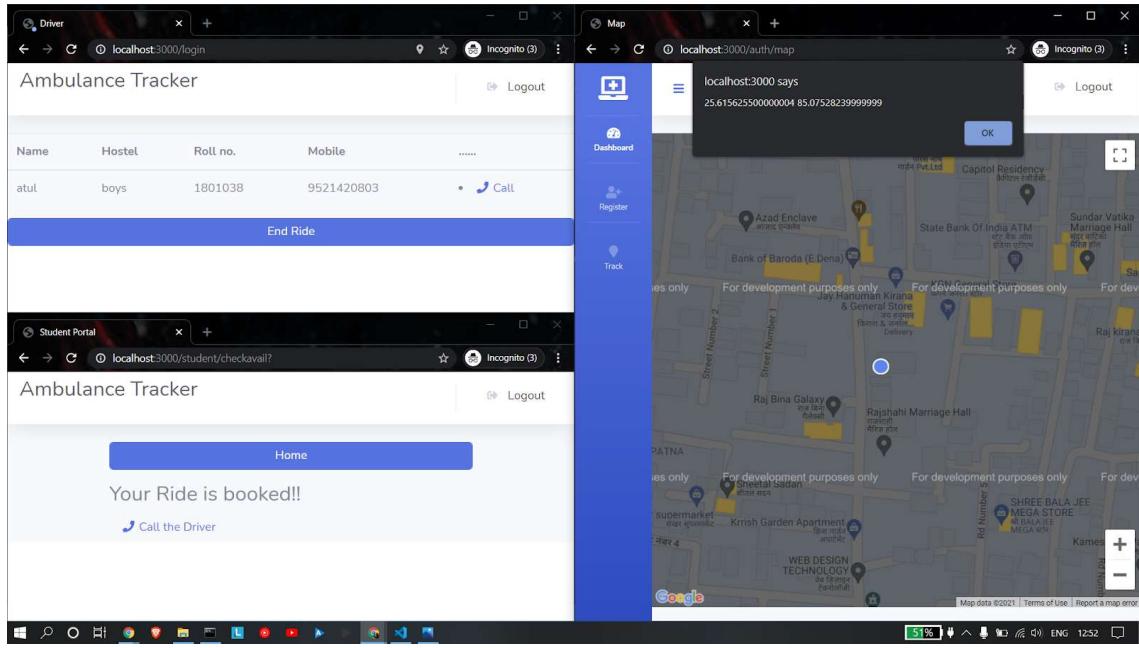
7. Location tracking facility is available for authorities :

After starting ride by driver :

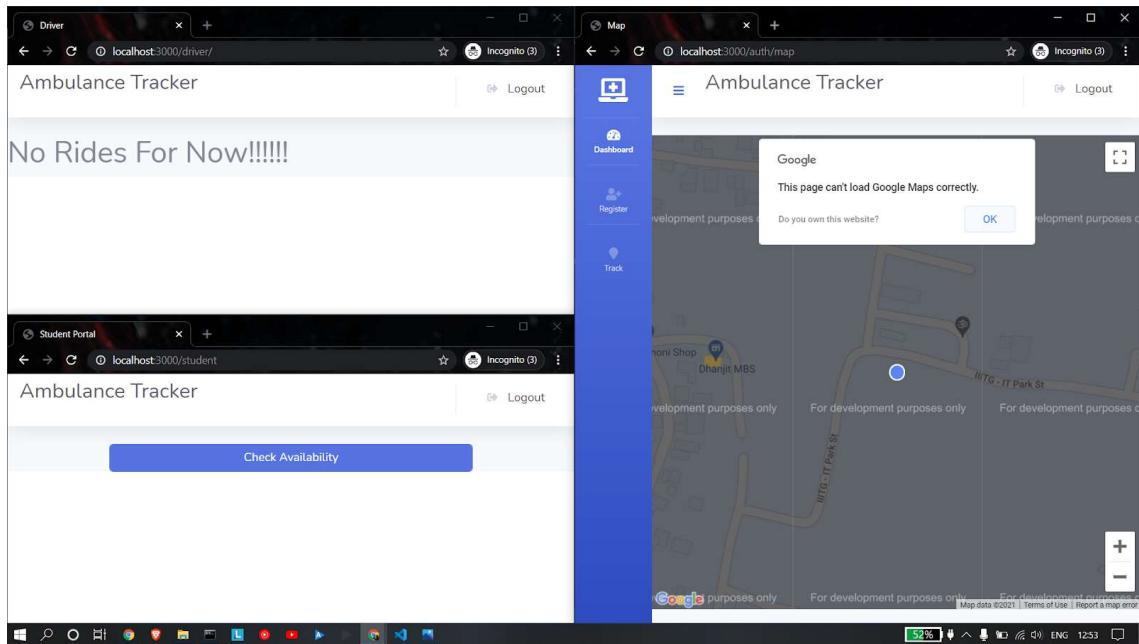


Location Displayed on the authority page :





After ending ride :



Lastly Checking the validity of all the functions in our project :

The screenshot shows the Visual Studio Code interface with the title "login.test.js - AmbulanceTrackerApp - Visual Studio Code". The left sidebar displays the file structure of the "AMBULANCETRACKERAPP" project, specifically the "backend" directory which contains "controllers" and "functesting" sub-directories. The "functesting" directory is currently selected. The right side of the screen shows the terminal window with the following output:

```
Test Suites: 5 passed, 5 total
Tests: 17 passed, 17 total
Snapshots: 0 total
Time: 9.323 s
Ran all test suites.
PS E:\AmbulanceTrackerApp\backend\functesting>
```

This was all the parts regarding the testing of our project.