

DOCUMENTATION OF THE TASK GIVEN

Step 1

Install the required node modules

Command

```
npm install
```

Step 2

Create the required database using mongoDB , given in the end

Step 3

after installing the required modules , start the server

command

```
node index.js
```

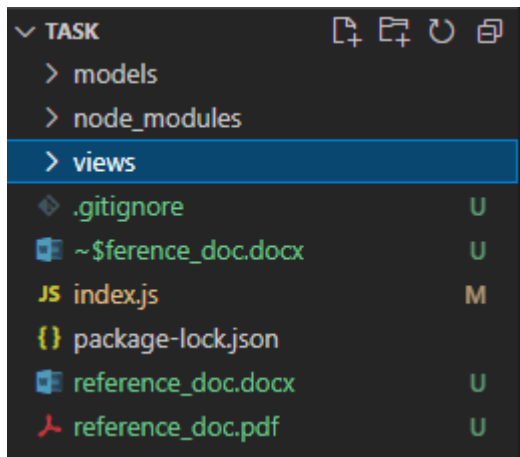
Step 4

Open any web browser

Navigate to localhost:8000

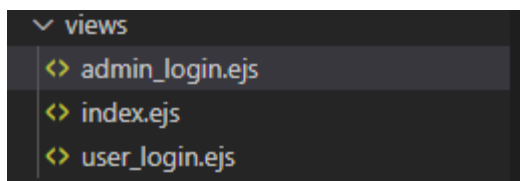


Folder structure



Index.js → main index node js with all logic

Views has 3 pages

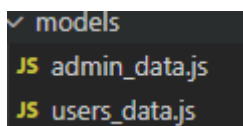


Index.ejs → Home page

Admin_login.ejs → login for admin page

User_login.ejs → login for normal users

Models has 2 schemas



Admin_data → schema for admin

Users_data → schema for normal users

HOME PAGE

HOME PAGE

[HOME](#)

[USER LOGIN](#)

[ADMIN LOGIN](#)

ADMIN LOGIN

<http://localhost:8000/admin>

Admin Login

UserId

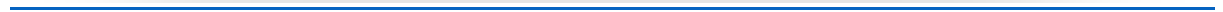


password



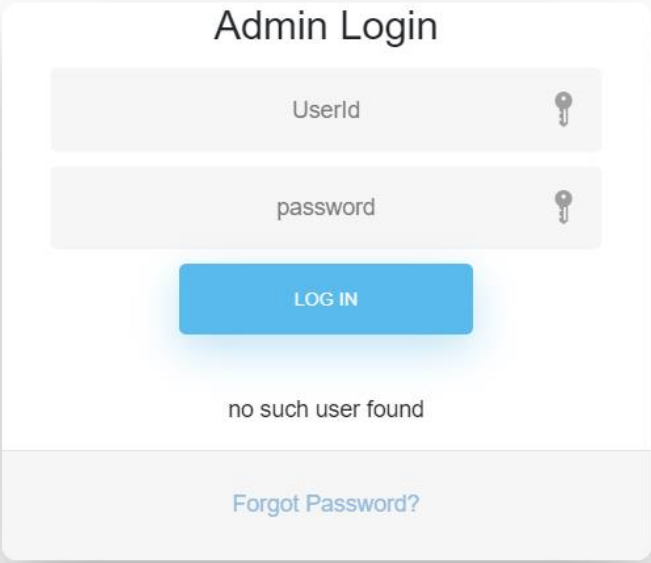
LOG IN

[Forgot Password?](#)



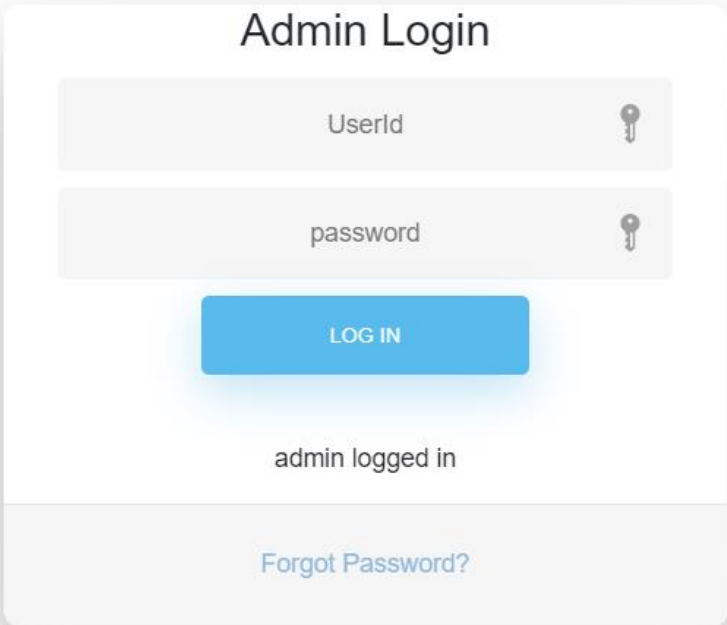
Enter the required userId and password

If such user does not exist in database



The image shows a web form titled "Admin Login". It contains two input fields: "UserId" and "password", each with a key icon on the right. Below the fields is a blue "LOG IN" button. Under the button, the text "no such user found" is displayed. At the bottom of the form, there is a link that says "Forgot Password?".

If user found

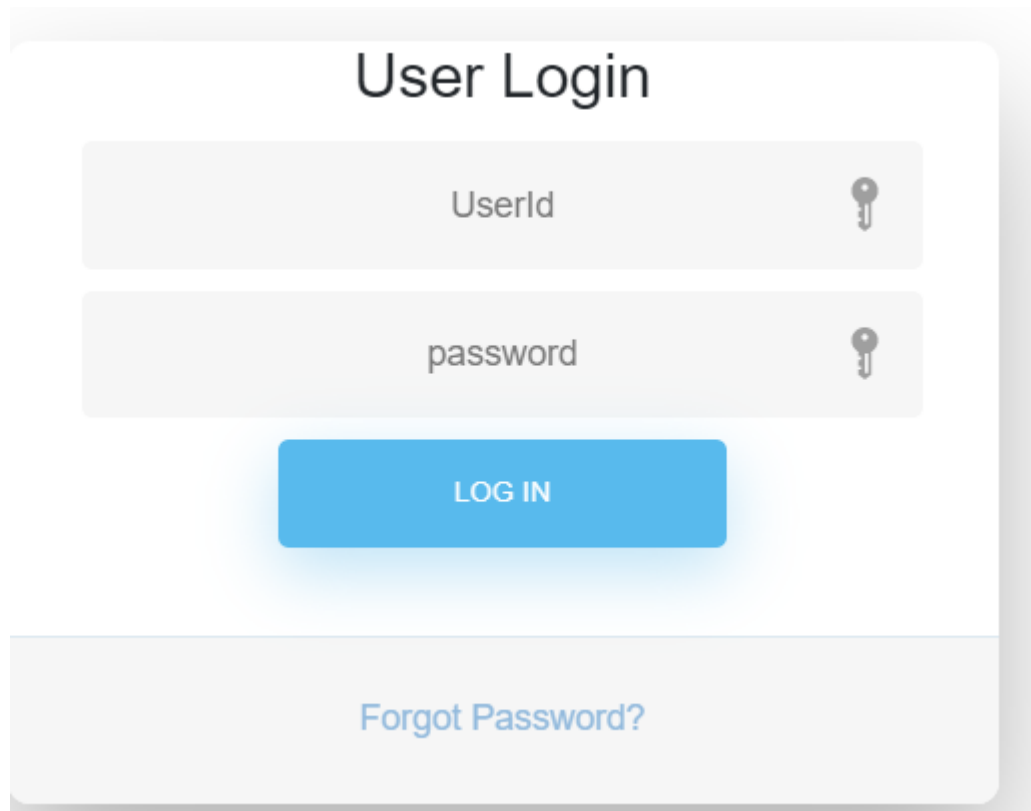


The image shows the same "Admin Login" form as above. In this state, the text "admin logged in" is displayed below the "LOG IN" button. The "Forgot Password?" link remains at the bottom.

After the user has logged in we can redirect to the required page, but for now only a message is shown if the user has successfully logged in or not

Similarly done for users login

<http://localhost:8000/user>



The image shows a 'User Login' form with a light gray background and rounded corners. At the top, the title 'User Login' is centered in a large, dark gray font. Below the title are two input fields, each with a light gray background and rounded corners. The first field is labeled 'UserId' and the second is labeled 'password'. Both fields have a small key icon on the right side, indicating they are required. Below the input fields is a blue button with the text 'LOG IN' in white, uppercase letters. At the bottom of the form is a light gray bar with the text 'Forgot Password?' in a blue, sans-serif font.

User Login

UserId

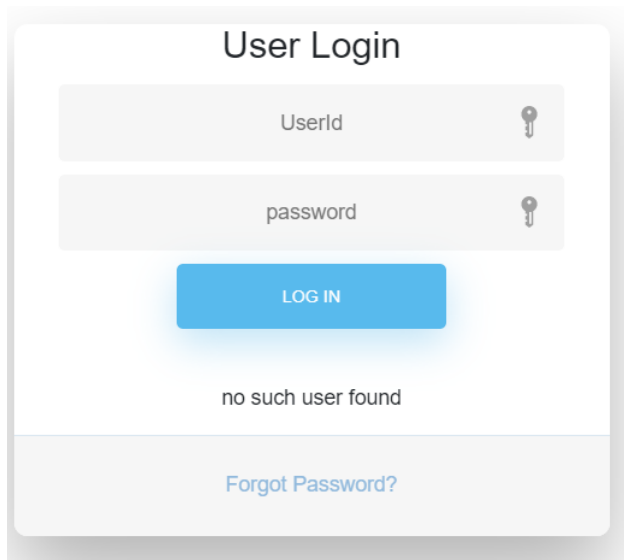
password

LOG IN

Forgot Password?

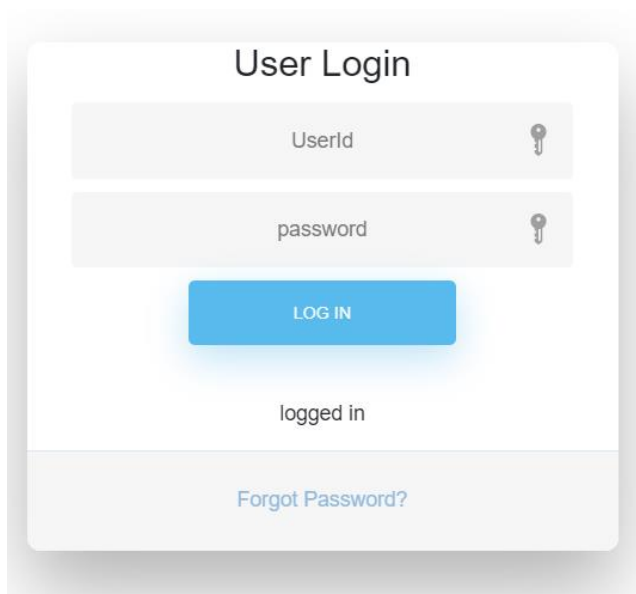
Enter username and password

If no such user found



The image shows a 'User Login' form with two input fields: 'UserId' and 'password', each with a key icon on the right. Below the fields is a blue 'LOG IN' button. Under the button, the text 'no such user found' is displayed. At the bottom of the form is a link that says 'Forgot Password?'.

If user found

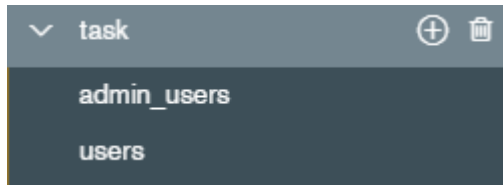


The image shows the same 'User Login' form as above, but with the text 'logged in' displayed below the 'LOG IN' button. The 'Forgot Password?' link remains at the bottom.

After the user has logged in we can redirect to the required page, but for now only a message is shown if the user has successfully logged in or not

DATABASE (made using mongoDB , connected locally)

Collection Name ^	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
admin_users	1	59.0 B	59.0 B	1	20.0 KB	
users	1	56.0 B	56.0 B	1	20.0 KB	



Admin_users db

task.admin_users Documents

task.admin_users

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER

ADD DATA VIEW

```
{
  "_id": ObjectId("613475f7b125d6897aedf1ca"),
  "userId": "bhavin",
  "password": "123"
}
```

Users db

task.users

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER

ADD DATA VIEW

```
{
  "_id": ObjectId("61348263b125d6897aedf1cd"),
  "userId": "bha",
  "password": "123"
}
```

Passwords can be hashed/encrypted and stored in later stages of the project