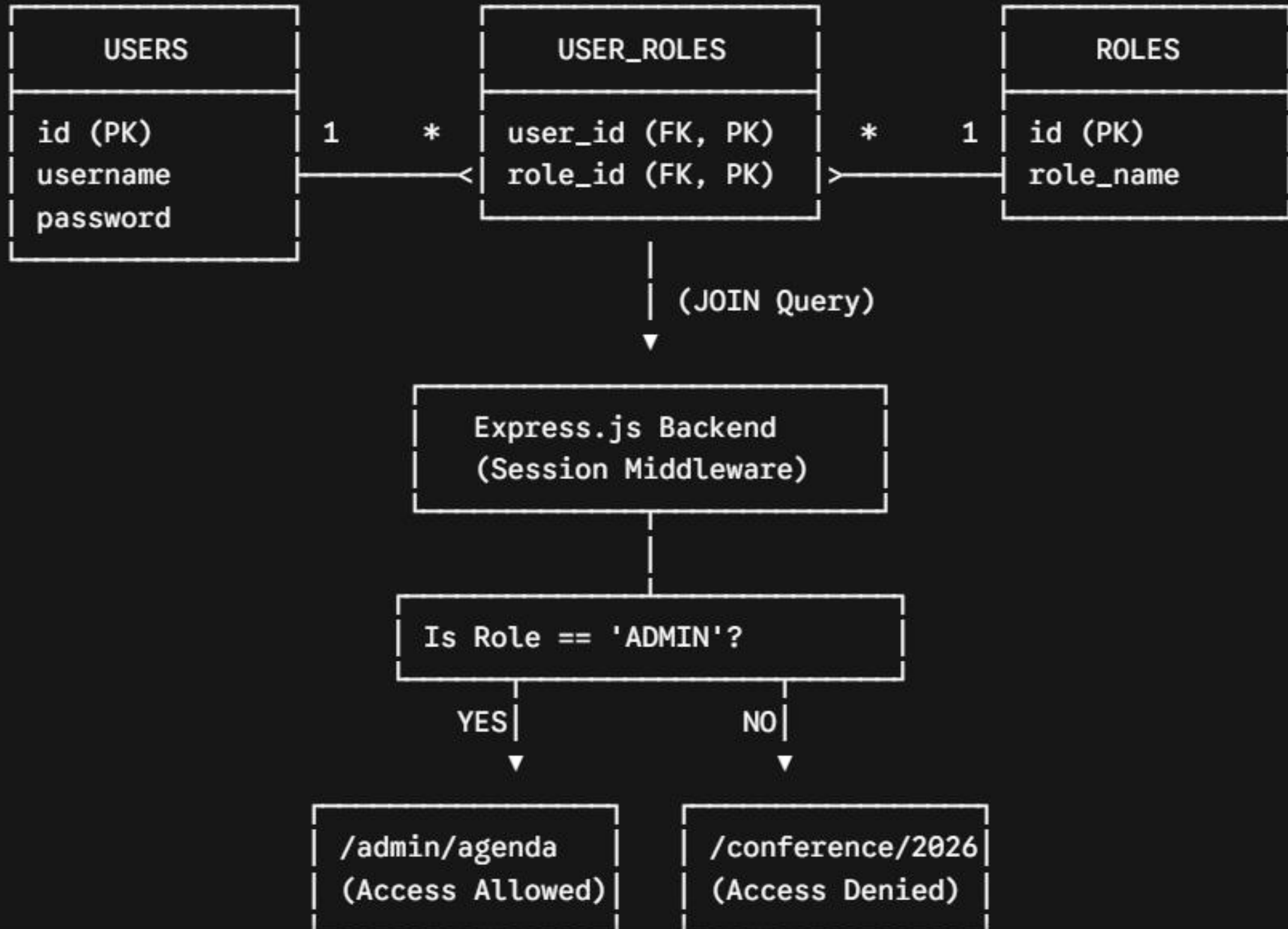


```
// Upload Route
router.post('/admin/upload-gallery', uploadGallery.array('galleryPhotos', 10), (req, res) => {
  res.redirect('/conference/2026');
});
```

```
index.html | menu.ejs | main.js | server.js | edit-agenda.ejs | conference.js | conference.ejs | index.ejs | cities-data
1 <%- include('../partials/header') %>
2 <%- include('../partials/menu') %>
3
4 <main class="container my-5">
5   <div class="row justify-content-center">
6     <div class="col-md-8">
7       <div class="card shadow-sm p-4">
8         <h3 class="mb-4 text-center">Edit Conference Agenda (2026)</h3>
9         <form action="/conference/admin/agenda" method="POST">
10
11           <div class="mb-3">
12             <label for="day1Text" class="form-label fw-bold">Day 1 Agenda Points:</label>
13             <textarea class="form-control" id="day1Text" name="day1Text" rows="6" placeholder="Keynote Speech&#10;Panel
14               Discussion"><%= day1Text %></textarea>
15           </div>
16
17           <div class="mb-3">
18             <label for="day2Text" class="form-label fw-bold">Day 2 Agenda Points:</label>
19             <textarea class="form-control" id="day2Text" name="day2Text" rows="6" placeholder="NIC Transport Showcase&#10;Awards
20               Ceremony"><%= day2Text %></textarea>
21           </div>
22
23           <button type="submit" class="btn btn-primary w-100">Save Agenda</button>
24         </form>
25       </div>
26     </div>
27   </div>
28 <%- include('../partials/footer') %>
```

```
index.html | menu.ejs | main.js | server.js | edit-agenda.ejs | conference.js | conference.ejs | index.ejs | cities-data.js

1 <%- include('../partials/header') %>
2 <%- include('../partials/menu') %>
3
4 <main class="container my-5">
5   <div class="row justify-content-center">
6     <div class="col-md-8">
7       <div class="card shadow-sm p-4">
8         <h3 class="mb-4 text-center">Edit Conference Agenda (2026)</h3>
9         <form action="/conference/admin/agenda" method="POST">
10
11           <div class="mb-3">
12             <label for="day1Text" class="form-label fw-bold">Day 1 Agenda Points:</label>
13             <textarea class="form-control" id="day1Text" name="day1Text" rows="6" placeholder="Keynote Speech&#10;Panel
14               Discussion"><%= day1Text %></textarea>
15           </div>
16
17           <div class="mb-3">
18             <label for="day2Text" class="form-label fw-bold">Day 2 Agenda Points:</label>
19             <textarea class="form-control" id="day2Text" name="day2Text" rows="6" placeholder="NIC Transport Showcase&#10;Awards
20               Ceremony"><%= day2Text %></textarea>
21           </div>
22
23           <button type="submit" class="btn btn-primary w-100">Save Agenda</button>
24         </form>
25       </div>
26     </div>
27   </div>
28 <%- include('../partials/footer') %>
```



Step-by-Step Flow Explanation

1. Tables ka Matlab:

- `users` **Table:** Isme sirf user ki personal details (username, password) honggi.
- `roles` **Table:** Isme system ke saare roles honggi (jaise: `ADMIN` , `USER` , `EDITOR`).
- `user_roles` **Table:** Yeh beech ki link/mapping table hai jo batati hai ki **kis User ID ke paas konsa Role ID hai.**

2. Login Kaise Kaam Karega?

- User form me username aur password daal kar submit karega.
- Node.js server database me `JOIN` query chala kar teenon tables ko check karega.
- Agar credentials sahi hain, toh server user ke roles ko fetch karke **Session Memory** me save kar lega (e.g.
`req.session.user = { username: 'admin', roles: ['ADMIN'] } ).`

3. Admin Route Check (`isAdmin` Middleware):

- Jab bhi koi `/conference/admin/agenda` par jayega, middleware check karega:

Session Available? → Role == 'ADMIN' Check → Grant Access

- Agar role `ADMIN` hai, toh admin page render hoga; warna Access Denied (403) ka message dikha dega.

SQL



```
CREATE TABLE roles (  
    id SERIAL PRIMARY KEY,  
    role_name VARCHAR(50) UNIQUE NOT NULL  
);  
  
CREATE TABLE users (  
    id SERIAL PRIMARY KEY,  
    username VARCHAR(50) UNIQUE NOT NULL,  
    password VARCHAR(255) NOT NULL,  
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
);  
  
CREATE TABLE user_roles (  
    user_id INT REFERENCES users(id) ON DELETE CASCADE,  
    role_id INT REFERENCES roles(id) ON DELETE CASCADE,  
    PRIMARY KEY (user_id, role_id)  
);
```

SQL



```
SELECT u.id, u.username, r.role_name
FROM users u
JOIN user_roles ur ON u.id = ur.user_id
JOIN roles r ON ur.role_id = r.id
WHERE u.username = $1 AND u.password = $2;
```

Kya aapko is RBAC Architecture ka complete Node.js + PostgreSQL Express code chahiye?

```
Connection-specific DNS Suffix . :  
Link-local IPv6 Address . . . . . : fe80::8ab6:e086:d4ea:2aef%18  
IPv4 Address. . . . . : 10.200.64.127  
Subnet Mask . . . . . : 255.255.255.0  
Default Gateway . . . . . : 10.200.64.1
```

line1000 LAN adapter Wi-Fi:



Not secure

10.200.64.127/parivahansewadbim/web/en/videos

The provided host name is not valid for this server.

CREATE TABLE

Query returned successfully in 184 msec.



Query

Query History

```
1 CREATE TABLE roles (  
2     id SERIAL PRIMARY KEY,  
3     role_name VARCHAR(50) UNIQUE NOT NULL  
4 );  
5  
6 CREATE TABLE users (  
7     id SERIAL PRIMARY KEY,  
8     username VARCHAR(50) UNIQUE NOT NULL,  
9     password VARCHAR(255) NOT NULL,  
10    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
11 );  
12  
13 CREATE TABLE user_roles (  
14     user_id INT REFERENCES users(id) ON DELETE CASCADE,  
15     role_id INT REFERENCES roles(id) ON DELETE CASCADE,  
16     PRIMARY KEY (user_id, role_id)  
17 );
```

transport_conference/postgres@PostgreSQL 16



Data Output Messages Notifications

CREATE TABLE

Query returned successfully in 184 msec.

Query Query History

```
1 CREATE TABLE roles (  
2     id SERIAL PRIMARY KEY,  
3     role_name VARCHAR(50) UNIQUE NOT NULL  
4 );  
5  
6 CREATE TABLE users (  
7     id SERIAL PRIMARY KEY,  
8     username VARCHAR(50) UNIQUE NOT NULL,  
9     password VARCHAR(255) NOT NULL,  
10    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
11 );  
12  
13 CREATE TABLE user_roles (  
14     user_id INT REFERENCES users(id) ON DELETE CASCADE,  
15     role_id INT REFERENCES roles(id) ON DELETE CASCADE,  
16     PRIMARY KEY (user_id, role_id)  
17 );
```

 public.users/transport_conference/postgres@PostgreSQL 16 

Data Output Messages Notifications



	id [PK] integer 	username character varying (50) 	password character varying (255) 	created_at timestamp without time zone 
1	1	admin	admin123	2026-09-09 08:40:47.326727

Query Query History

```
1 SELECT * FROM public.users
2 ORDER BY id ASC
```

public.roles/transport_conference/postgres@PostgreSQL 16



Data Output Messages Notifications

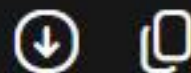


	id [PK] integer	role_name character varying (50)
1	1	ADMIN
2	2	USER

Query Query History

```
1 SELECT * FROM public.roles
2 ORDER BY id ASC
```

SQL



```
SELECT u.id, u.username, r.role_name  
FROM users u  
JOIN user_roles ur ON u.id = ur.user_id  
JOIN roles r ON ur.role_id = r.id  
WHERE u.username = 'admin';
```

Output: आपको screen par `username: admin` aur `role_name: ADMIN` dikhai dega.



Query

Query History



```
1  -- 1. Basic Roles Insert Karein
2  INSERT INTO roles (role_name) VALUES ('ADMIN'), ('USER');
3
4  -- 2. Test Admin User Insert Karein (Password: admin123)
5  INSERT INTO users (username, password) VALUES ('admin', 'admin123');
6
7  -- 3. 'admin' User ko 'ADMIN' Role Assign Karein
8  -- Assuming: 'admin' user ki ID = 1 aur 'ADMIN' role ki ID = 1
9  INSERT INTO user_roles (user_id, role_id)
10 VALUES (
11     (SELECT id FROM users WHERE username = 'admin'),
12     (SELECT id FROM roles WHERE role_name = 'ADMIN')
13 );
```

transport_conference/postgres@PostgreSQL 16



Data Output Messages Notifications



	id integer	username character varying (50)	role_name character varying (50)
1	1	admin	ADMIN

Query Query History

```
1 SELECT u.id, u.username, r.role_name
2 FROM users u
3 JOIN user_roles ur ON u.id = ur.user_id
4 JOIN roles r ON ur.role_id = r.id
5 WHERE u.username = 'admin';
```

```

app.post('/login', async (req, res) => {
  const { username, password } = req.body;

  try {
    const query = `
      SELECT u.id, u.username, r.role_name
      FROM users u
      JOIN user_roles ur ON u.id = ur.user_id
      JOIN roles r ON ur.role_id = r.id
      WHERE u.username = $1 AND u.password = $2
    `;

    const result = await pool.query(query, [username, password]);

    if (result.rows.length > 0) {
      // User and Roles array Extract Karein
      const user = {
        id: result.rows[0].id,
        username: result.rows[0].username,
        roles: result.rows.map(row => row.role_name) // e.g. ['ADMIN']
      };

      req.session.user = user; // Session me store hua

      if (user.roles.includes('ADMIN')) {
        return res.redirect('/conference/admin/agenda');
      } else {
        return res.redirect('/conference/2026');
      }
    } else {
      res.send('Invalid Username or Password');
    }
  } catch (err) {
    console.error(err);
    res.status(500).send('Server Error');
  }
});

```



```
1 <%- include('partials/header') %>
2 <%- include('partials/menu') %>
3
4 <main class="container my-5">
5   <div class="row justify-content-center">
6     <div class="col-md-5">
7       <div class="card shadow-sm p-4">
8         <h3 class="text-center mb-4">Account Login</h3>
9
10        <!-- Error Alert -->
11        <% if (typeof error !== 'undefined' && error) { %>
12          <div class="alert alert-danger" role="alert">
13            <%= error %>
14          </div>
15        <% } %>
16
17        <form action="/login" method="POST">
18          <div class="mb-3">
19            <label for="username" class="form-label fw-bold">Username</label>
20            <input type="text" class="form-control" id="username" name="username" placeholder="Enter username" required>
21          </div>
22          <div class="mb-3">
23            <label for="password" class="form-label fw-bold">Password</label>
24            <input type="password" class="form-control" id="password" name="password" placeholder="Enter password" required>
25          </div>
26          <button type="submit" class="btn btn-primary w-100">Login</button>
27        </form>
28      </div>
29    </div>
30  </div>
31 </main>
32
33 <%- include('partials/footer') %>
```

```

83
84
85
86
87
88  /*****Login *****/
89
90  //
91
92
93
94  app.post('/login', async (req, res) => {
95    const { username, password } = req.body;
96
97    try {
98      const query = `
99        SELECT u.id, u.username, r.role_name
100        FROM users u
101        JOIN user_roles ur ON u.id = ur.user_id
102        JOIN roles r ON ur.role_id = r.id
103        WHERE u.username = $1 AND u.password = $2
104      `;
105
106      const result = await pool.query(query, [username, password]);
107
108      if (result.rows.length > 0) {
109        // User and Roles array Extract Karein
110        const user = {
111          id: result.rows[0].id,
112          username: result.rows[0].username,
113          roles: result.rows.map(row => row.role_name) // e.g. ['ADMIN']
114        };
115
116        req.session.user = user; // Session me store hua
117
118        if (user.roles.includes('ADMIN')) {
119          return res.redirect('/conference/admin/agenda');
120        } else {
121          return res.redirect('/conference/2026');
122        }
123      } else {
124        res.send('Invalid Username or Password');
125      }
126    } catch (err) {
127      console.error(err);
128      res.status(500).send('Server Error');
129    }
130  });
131
132
133

```

```
1 <%- include('partials/header') %>
2 <%- include('partials/menu') %>
3
4 <main class="container my-5">
5   <div class="row justify-content-center">
6     <div class="col-md-5">
7       <div class="card shadow-sm p-4">
8         <h3 class="text-center mb-4">Account Login</h3>
9
10        <!-- Error Alert -->
11        <% if (typeof error !== 'undefined' && error) { %>
12          <div class="alert alert-danger" role="alert">
13            <%= error %>
14          </div>
15        <% } %>
16
17        <form action="/login" method="POST">
18          <div class="mb-3">
19            <label for="username" class="form-label fw-bold">Username</label>
20            <input type="text" class="form-control" id="username" name="username" placeholder="Enter username" required>
21          </div>
22          <div class="mb-3">
23            <label for="password" class="form-label fw-bold">Password</label>
24            <input type="password" class="form-control" id="password" name="password" placeholder="Enter password" required>
25          </div>
26          <button type="submit" class="btn btn-primary w-100">Login</button>
27        </form>
28      </div>
29    </div>
30  </div>
31 </main>
32
33 <%- include('partials/footer') %>
```

Step 2: `server.js` me `res.locals` middleware add karein

Apni main `server.js` file me `express-session` jahan `app.use` kiya hai, uske thik niche yeh 3 lines add kar dein:

JavaScript



```
app.use((req, res, next) => {  
  res.locals.user = req.session ? req.session.user : null;  
  next();  
});
```

```
96
97
98 // 1. Login Page open karne ke liye (GET Route)
99 router.get('/login', (req, res) => {
100   if (req.session && req.session.user && req.session.user.roles.includes('ADMIN')) {
101     return res.redirect('/conference/admin/agenda');
102   }
103   res.render('login', { error: null });
104 });
105
106 // 2. Login submit hone par (POST Route - 'app' ki jagah 'router' karein)
107 router.post('/login', async (req, res) => {
108   const { username, password } = req.body;
109
110   try {
111     const query = `
112       SELECT u.id, u.username, r.role_name
113       FROM users u
114       JOIN user_roles ur ON u.id = ur.user_id
115       JOIN roles r ON ur.role_id = r.id
116       WHERE u.username = $1 AND u.password = $2
117     `;
118
119     const result = await pool.query(query, [username, password]);
120
121     if (result.rows.length > 0) {
122       const user = {
123         id: result.rows[0].id,
124         username: result.rows[0].username,
125         roles: result.rows.map(row => row.role_name)
126       };
127
128       req.session.user = user;
129
130       if (user.roles.includes('ADMIN')) {
131         return res.redirect('/conference/admin/agenda');
132       } else {
133         return res.redirect('/conference/2026');
134       }
135     } else {
136       res.render('login', { error: 'Invalid Username or Password' });
137     }
138   } catch (err) {
139     console.error(err);
140     res.status(500).send('Server Error');
141   }
142 });
143
144 // 3. Logout Route
145 router.get('/logout', (req, res) => {
146   req.session.destroy(() => {
147     res.redirect('/conference/login');
148   });
149 });
```

Step 1: Installation Command

Apne project ke root terminal/command prompt me yeh command run karein:

Bash



```
npm install express-session
```

Step 2: Server File me Usage (`server.js` ya `app.js`)

Package install karne ke baad apni main file (`server.js`) me isko is tarah import aur configure karein:

JavaScript



```
const session = require('express-session');

// Express Session Middleware Setup
app.use(session({
  secret: 'my_secret_key_123', // Apni secret key rakhein
  resave: false,
  saveUninitialized: false,
  cookie: {
    maxAge: 24 * 60 * 60 * 1000 // Session 1 din tak valid rahega
  }
}));

// Session user ko saare EJS views/menu me accessible banane ke liye middleware
app.use((req, res, next) => {
  res.locals.user = req.session ? req.session.user : null;
  next();
});
```

1. `res.locals.user` wali 3 lines ka kya matlab hai?

Aapne jab user ko login karwaya, toh uski details backend par `req.session.user` me save ho gayi.

Lekin Express ka rule yeh hai ki backend ka data har ek EJS view file (`menu.ejs` , `index.ejs` , `conference.ejs`) ko tab tak nahi milta jab tak aap use bhejte nahi.

JavaScript



```
app.use((req, res, next) => {  
  res.locals.user = req.session ? req.session.user : null;  
  next();  
});
```

- `res.locals` : Express ka ek special object hai. Isme aap jo bhi variable daalte hain (jaise humne `res.locals.user` daala), woh saare EJS views aur partials (`menu.ejs`) me automatically pohench jata hai.

```
33 packages are looking for funding
  run `npm fund` for details
```

```
found 0 vulnerabilities
```

```
PS C:\Drupal-Miscellaneous\Drupal Study\Node JS\transport_conference> ^C
```

```
PS C:\Drupal-Miscellaneous\Drupal Study\Node JS\transport_conference> node server.js
```

```
C:\Drupal-Miscellaneous\Drupal Study\Node JS\transport_conference\server.js:9
```

```
const session = require('express-session');
      ^
```

```
SyntaxError: Identifier 'session' has already been declared
```

```
    at wrapSafe (node:internal/modules/cjs/loader:1638:18)
```

```
    at Module._compile (node:internal/modules/cjs/loader:1680:20)
```

```
    at Object..js (node:internal/modules/cjs/loader:1839:10)
```

```
    at Module.load (node:internal/modules/cjs/loader:1441:32)
```

```
    at Function._load (node:internal/modules/cjs/loader:1263:12)
```

```
    at TracingChannel.traceSync (node:diagnostics_channel:328:14)
```

```
    at wrapModuleLoad (node:internal/modules/cjs/loader:237:24)
```

```
    at Function.executeUserEntryPoint [as runMain] (node:internal/modules/run_main:171:5)
```

```
    at node:internal/main/run_main_module:36:49
```

```
Node.js v22.22.0
```

```
PS C:\Drupal-Miscellaneous\Drupal Study\Node JS\transport_conference> |
```



← → ↻ ⓘ localhost:3000/login

Cannot POST /login



minister of transport and high



My Links Page



localhost:3000/login

Cannot POST /login



localhost:3000/conference/login

Server Error

FOLDERS

- ▼ transport_conference
 - ▼ db
 - /* agenda.json
 - /* index.js
 - ▶ node_modules
 - ▼ public
 - ▼ conference
 - ▶ photogallery
 - ▼ resources
 - ▼ 2026
 - ▶ css

```
untitled | conference.js | index.js | server.js
1  const { Pool } = require('pg');
2  require('dotenv').config();
3
4  const pool = new Pool({
5    user: process.env.DB_USER || 'postgres',
6    host: process.env.DB_HOST || 'localhost',
7    database: process.env.DB_NAME || 'transport_conference',
8    password: process.env.DB_PASSWORD || 'kamlu',
9    port: process.env.DB_PORT || 5433,
10  });
11
12  module.exports = pool;
```

e-Transport Conference

Government of India Initiative

Register Now

[Download Report](#)

Admin Panel

Logout (admin)

Edit Conference Agenda (2026)

Day 1 Agenda Points:

02:00 PM – 03:00 PM

Registration & Lunch

03:00 PM – 03:30 PM — Inaugural Session

Arrival of Chief Guest

Presentation of bouquet

Lighting of Lamp

Day 2 Agenda Points:

09:30 AM – 09:45 AM

Welcome Speech by NIC

09:45 AM – 10:00 AM

Presentation on Dashboard & Analytics Portal – by NIC

10:00 AM – 10:15 AM

Presentation on new initiatives in eDetection – by NIC

Save Agenda

Boht badhiya! Login flow aur redirection bilkul properly kaam kar rahe hain.

Admin page par User Info Bar aur Live Session Countdown Timer add karne ke liye baseline logic is prakar hai:

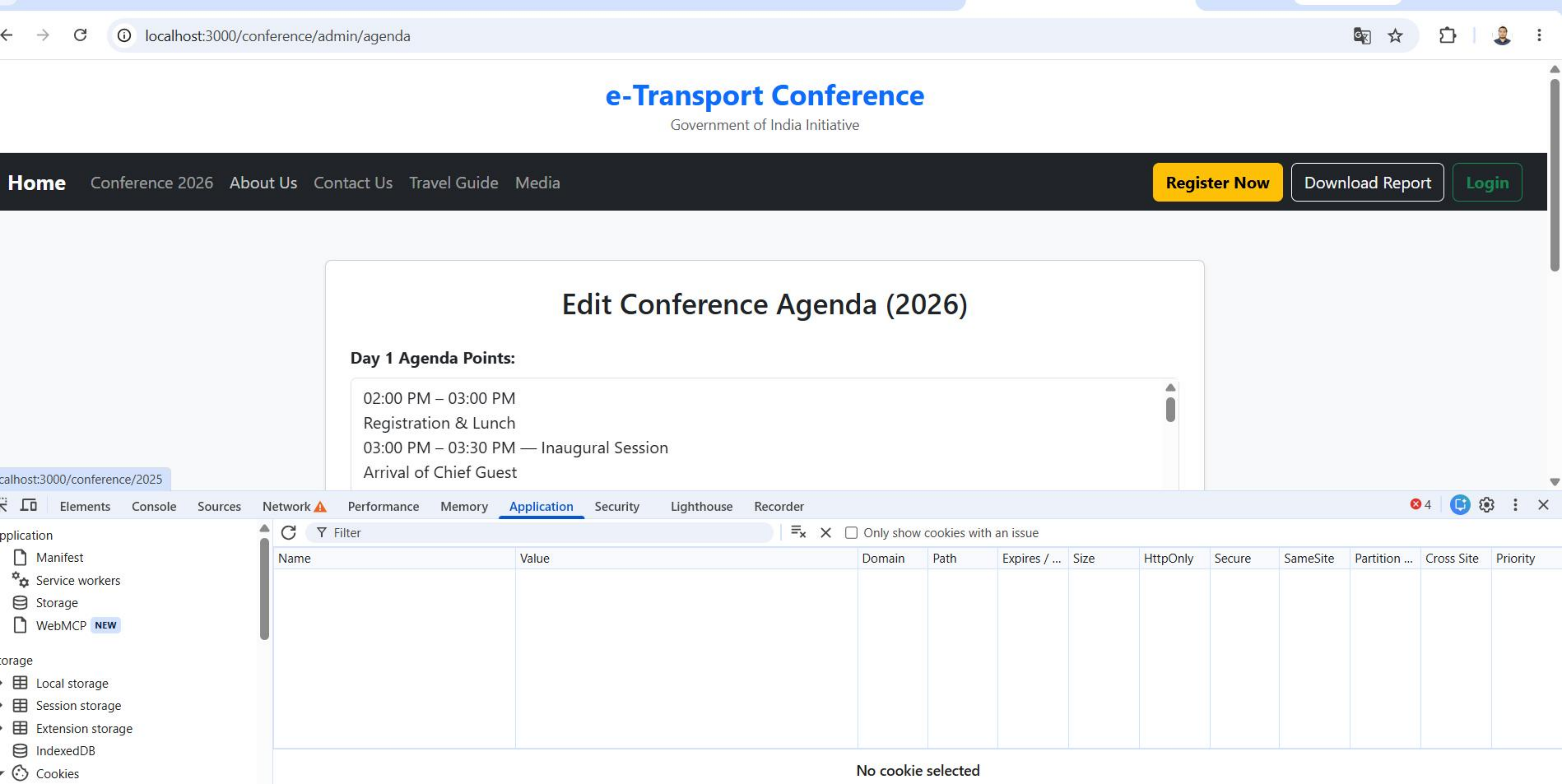
Step 1: `server.js` me Session Expiry Time Pass Karein

Countdown timer ko yeh pata hona chahiye ki session kab expire hoga. Iske liye `server.js` me `res.locals` middleware ko update karein:

JavaScript



```
app.use((req, res, next) => {  
  res.locals.user = req.session ? req.session.user : null;  
  // Session expiry timestamp (milliseconds) client ko bhejne ke liye  
  res.locals.sessionExpiresAt = req.session && req.session.cookie ? req.session.cookie.expires : null;  
  next();  
});
```



e-Transport Conference

Government of India Initiative

Edit Conference Agenda (2026)

Day 1 Agenda Points:

- 02:00 PM – 03:00 PM
Registration & Lunch
- 03:00 PM – 03:30 PM — Inaugural Session
- Arrival of Chief Guest

Application

- Manifest
- Service workers
- Storage
- WebMCP NEW

Storage

- Local storage
- Session storage
- Extension storage
- IndexedDB
- Cookies

Filter

Name	Value	Domain	Path	Expires / ...	Size	HttpOnly	Secure	SameSite	Partition ...	Cross Site	Priority
No cookie selected											