Frontend Code:

1. src/App.js
```
import React, { useState } from 'react';
import Login from './Login';
import Dashboard from './Dashboard';

function App() {
  const [token, setToken] = useState(localStorage.getItem('token'));

  return (
    <div>
      {token ? <Dashboard token={token} /> : <Login setToken={setToken} />}
    </div>
  );
}

export default App;
```

2. src/Login.js
```
import React, { useState } from 'react';
import axios from 'axios';

function Login({ setToken }) {
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');

  const handleSubmit = async (e) => {
    e.preventDefault();
    try {
      const response = await axios.post('http://localhost:5000/api/auth/login', {
email, password });
      localStorage.setItem('token', response.data.token);
      setToken(response.data.token);
    } catch (error) {
      console.error('Login failed', error);
    }
  };

  return (
    <form onSubmit={handleSubmit}>
```

```
      <input type="email" value={email} onChange={(e) =>
setEmail(e.target.value)} placeholder="Email" required />
      <input type="password" value={password} onChange={(e) =>
setPassword(e.target.value)} placeholder="Password" required />
      <button type="submit">Login</button>
    </form>
  );
}

export default Login;
```

3. src/Dashboard.js

```
import React, { useEffect, useState } from 'react';
import axios from 'axios';

function Dashboard({ token }) {
  const [users, setUsers] = useState([]);

  useEffect(() => {
    const fetchUsers = async () => {
      try {
        const response = await axios.get('http://localhost:5000/api/users', {
          headers: { Authorization: `Bearer ${token}` },
        });
        setUsers(response.data);
      } catch (error) {
        console.error('Failed to fetch users', error);
      }
    };
    fetchUsers();
  }, [token]);

  return (
    <div>
      <h1>Dashboard</h1>
      <ul>
        {users.map((user) => (
          <li key={user._id}>{user.username} - {user.role}</li>
        ))}
      </ul>
    </div>
  );
}

export default Dashboard;
```

Backend Code:

1. server.js

```
const express = require('express');
const mongoose = require('mongoose');
const cors = require('cors');
const userRoutes = require('./routes/userRoutes');
const authRoutes = require('./routes/authRoutes');
const { authenticateToken } = require('./middleware/authMiddleware');

const app = express();
app.use(cors());
app.use(express.json());

mongoose.connect('your_mongodb_connection_string', {
  useNewUrlParser: true,
  useUnifiedTopology: true,
});

app.use('/api/auth', authRoutes);
app.use('/api/users', authenticateToken, userRoutes);

app.listen(5000, () => {
  console.log('Server running on port 5000');
});
```

2. models/User.js

```
const mongoose = require('mongoose');

const userSchema = new mongoose.Schema({
  username: { type: String, required: true },
  email: { type: String, required: true },
  password: { type: String, required: true },
  role: { type: String, enum: ['user', 'admin'], default: 'user' },
});

module.exports = mongoose.model('User', userSchema);
```

3. routes/authRoutes.js

```
const express = require('express');
const bcrypt = require('bcrypt');
const jwt = require('jsonwebtoken');
const User = require('../models/User');

const router = express.Router();

router.post('/register', async (req, res) => {
  const { username, email, password } = req.body;
  const hashedPassword = await bcrypt.hash(password, 10);
  const newUser = new User({ username, email, password: hashedPassword });
  await newUser.save();
  res.status(201).send('User registered');
});

router.post('/login', async (req, res) => {
  const { email, password } = req.body;
  const user = await User.findOne({ email });
  if (!user) return res.status(400).send('User not found');
  const valid = await bcrypt.compare(password, user.password);
  if (!valid) return res.status(400).send('Invalid credentials');
  const token = jwt.sign({ id: user._id, role: user.role }, 'your_jwt_secret');
  res.json({ token });
});

module.exports = router;
```

4. routes/userRoutes.js

```
const express = require('express');
const User = require('../models/User');

const router = express.Router();

router.get('/', async (req, res) => {
  const users = await User.find();
  res.json(users);
});

module.exports = router;
```

5. middleware/authMiddleware.js

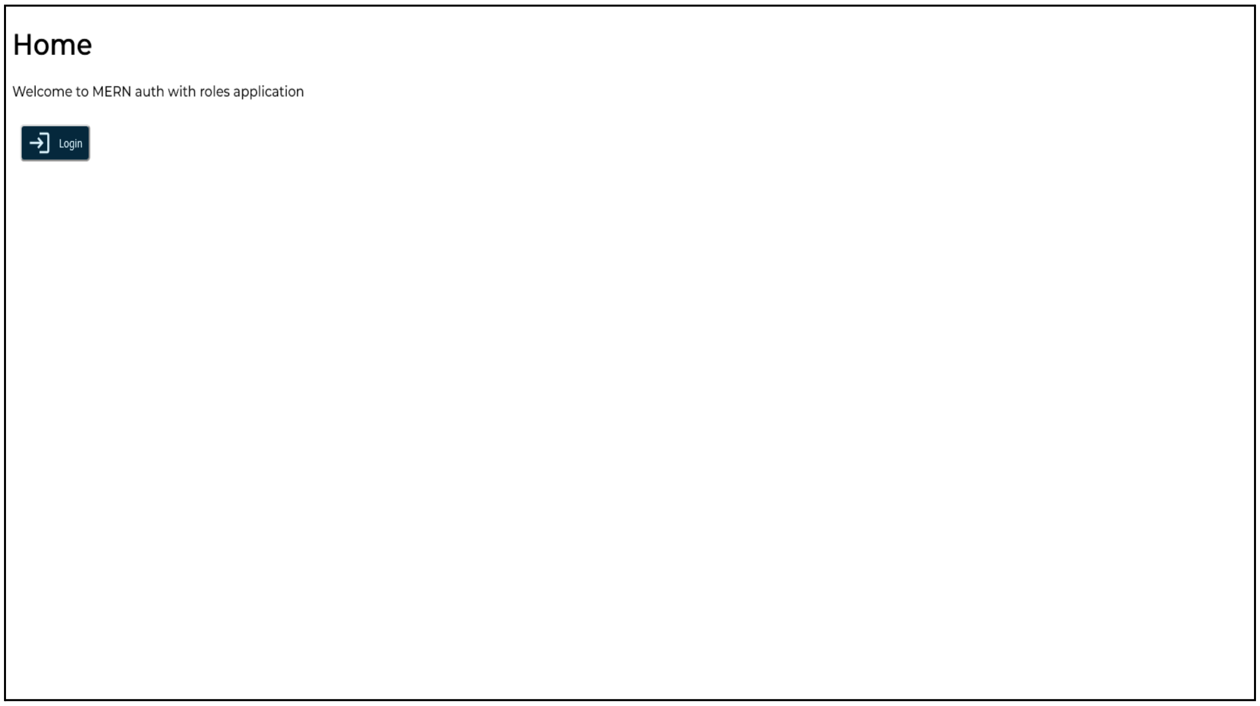
```
const jwt = require('jsonwebtoken');

function authenticateToken(req, res, next) {
  const token = req.header('Authorization')?.split(' ')[1];
  if (!token) return res.status(401).send('Access denied');
  jwt.verify(token, 'your_jwt_secret', (err, user) => {
    if (err) return res.status(403).send('Invalid token');
    req.user = user;
    next();
  });
}

module.exports = { authenticateToken };
```

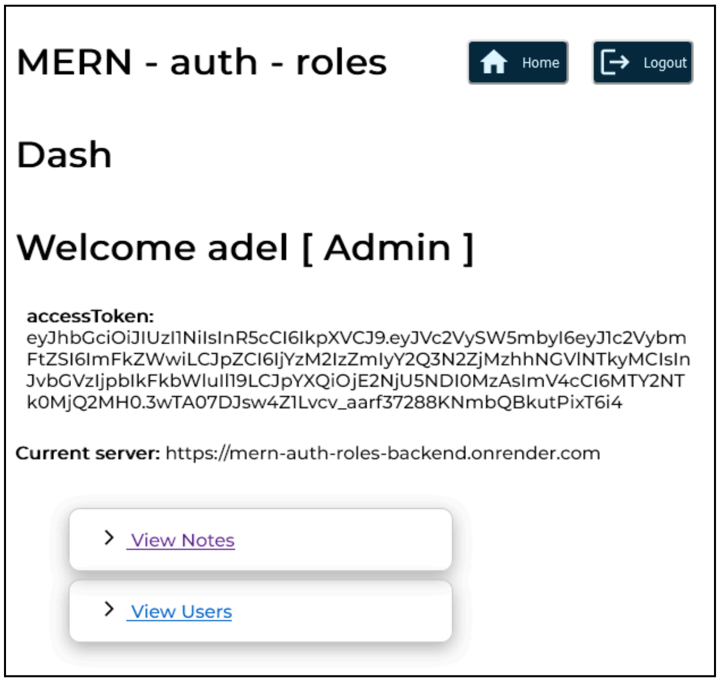
Output:

Home page


Home

Welcome to MERN auth with roles application

Dashboard:



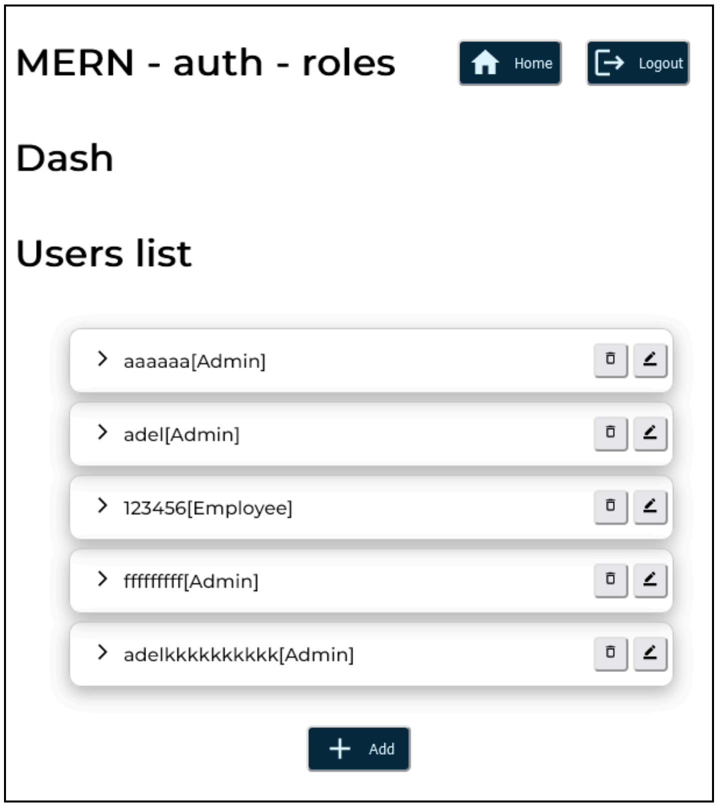
Dash

Welcome adel [Admin]

accessToken:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJVc2VySW5mbyI6eyJ1c2Vybm
FtZSI6ImFkZWwiLCJpZCI6IjYzM2IzZmIyY2Q3N2ZjMzhhNGVlNTkyMCIsIn
JvbGVzIjpbIkFkbWluIl19LCJpYXQiOjE2NjU5NDI0MzAsImV4cCI6MTY2NT
k0MjQ2MH0.3wTA07DJsw4Z1Lvcv_aarf37288KNmbQBkutPixT6i4

Current server: https://mern-auth-roles-backend.onrender.com

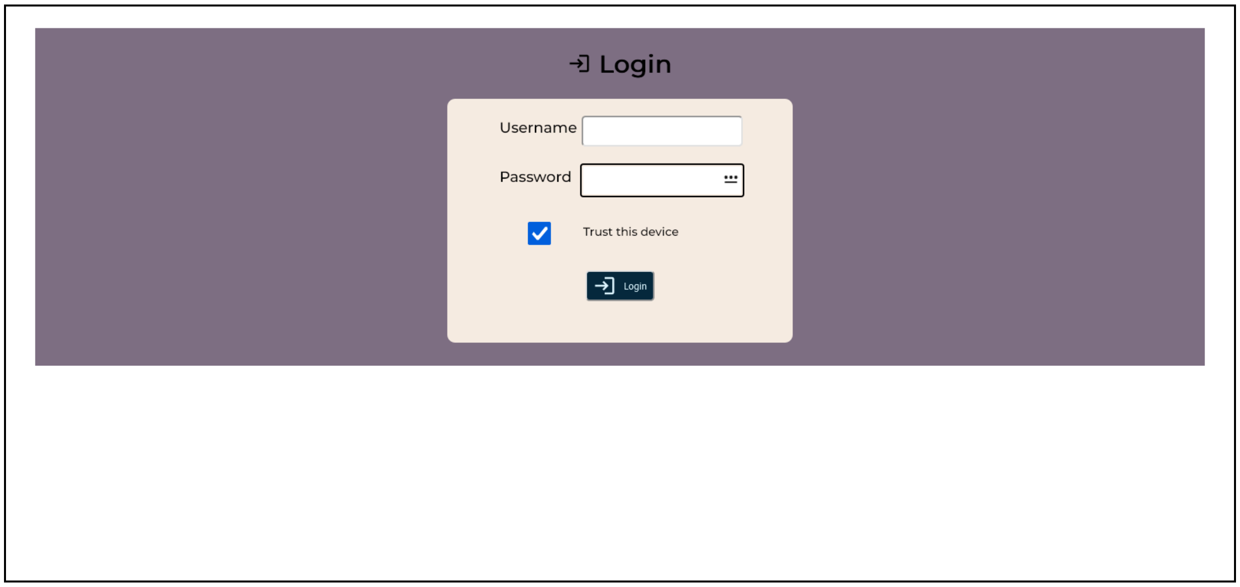
Users:



Dash

Users list

aaaaaa[Admin]
adel[Admin]
123456[Employee]
ffffffff[Admin]
adelkkkkkkkkk[Admin]

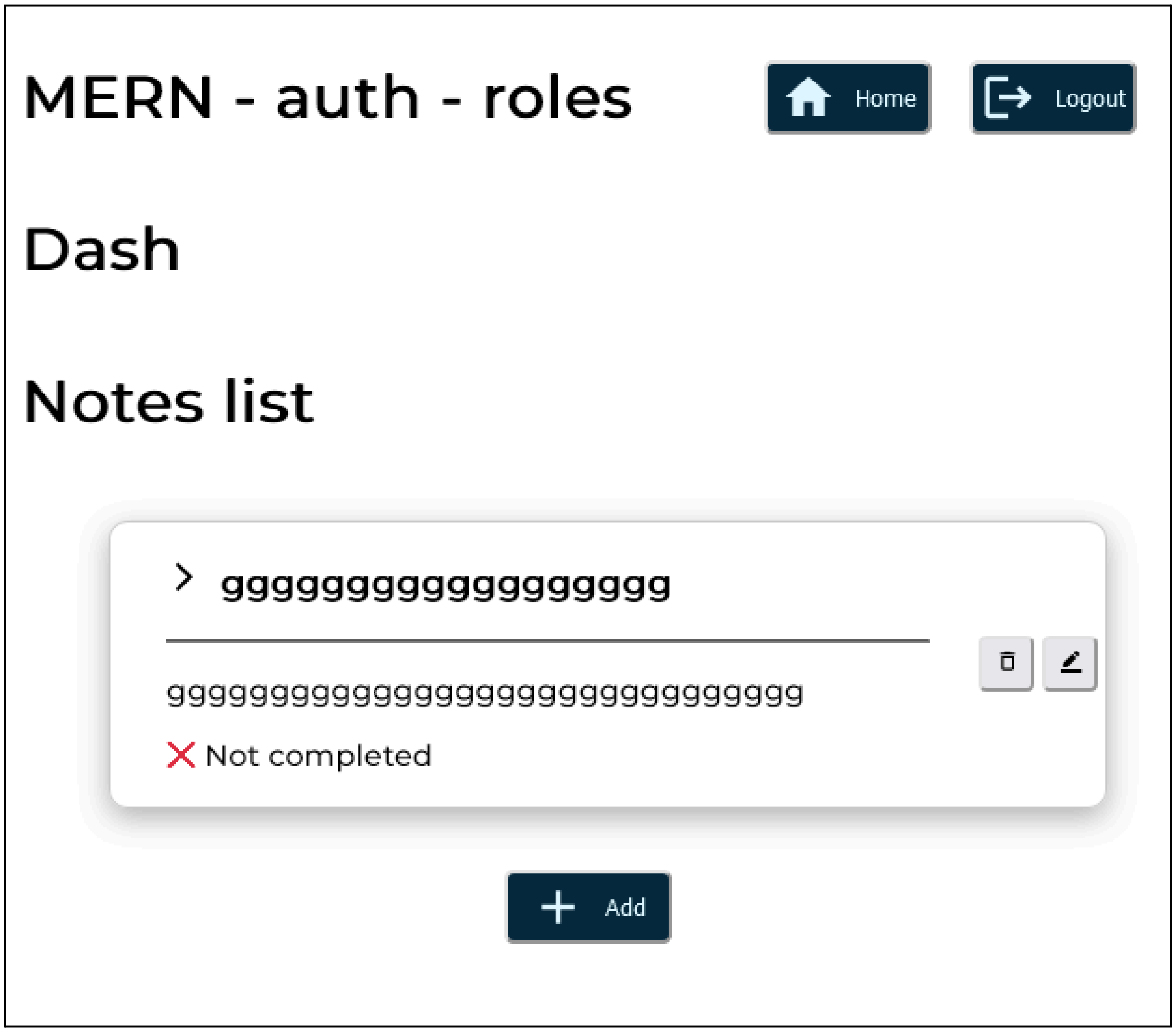

Login

Username

Password

Trust this device

Notes:


MERN - auth - roles

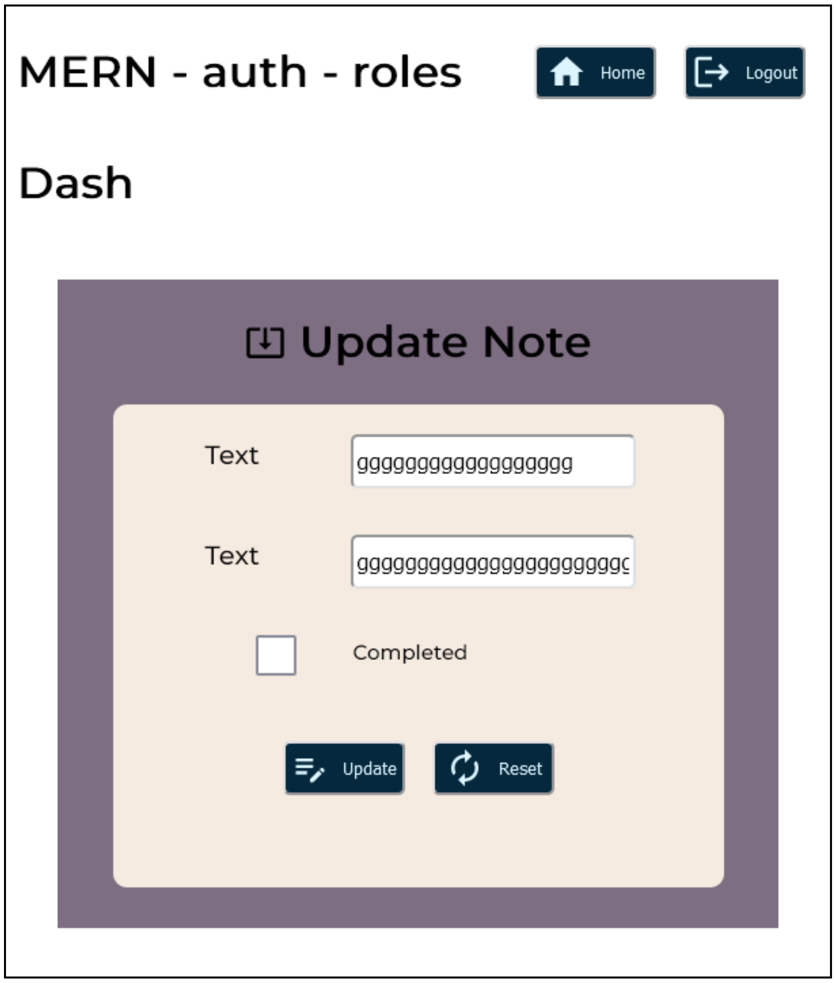
Dash

Notes list

gggggggggggggggggg

gggggggggggggggggggggggggggggggg

❌ Not completed



Text: gggggggggggggggggg

Text: gggggggggggggggggggggggg

☐ Completed