WEEK-3

AIM: Write a program for session management using cookies and sessions

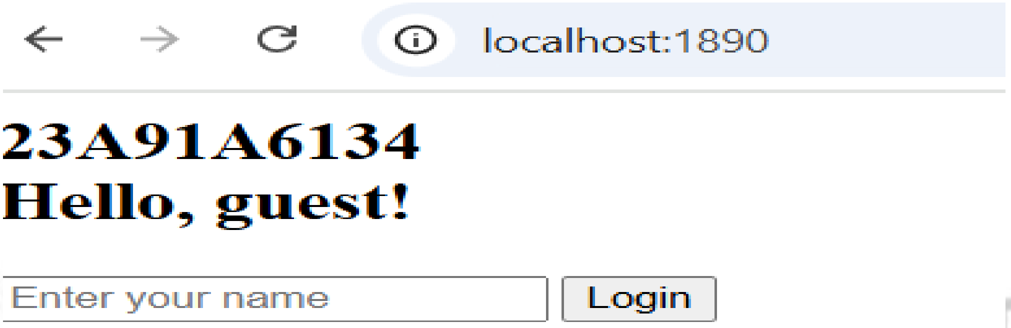
DESCRIPTION: This program demonstrates session management in a web application using cookies and server-side sessions. It allows a user to log in with a username, maintains the user's state across multiple requests using a session, and also sets a cookie in the browser to store basic user information.

PROGRAM:

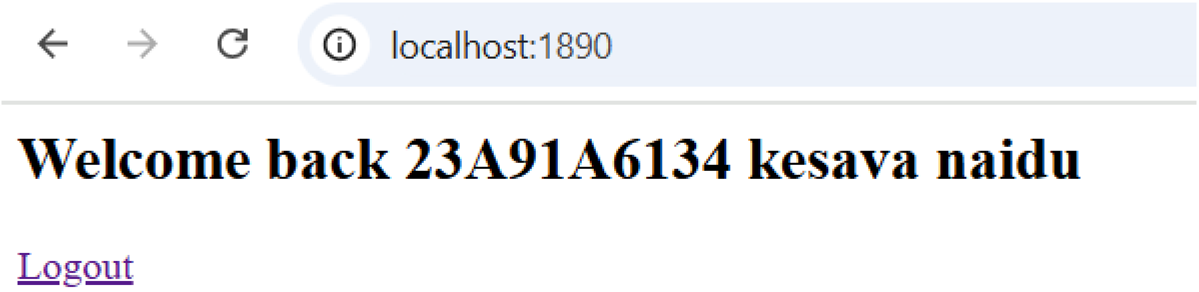
```
const express = require('express');
const session = require('express-session');
const cookieParser = require('cookie-parser');
const app = express();
const port = 1890;

app.use(cookieParser());
app.use(express.urlencoded({ extended: true }));

app.use(session({
  secret: 'mysecret',
  resave: false,
  saveUninitialized: true,
  cookie: { maxAge: 60000 } // 1 minute
}));

// Home route
app.get('/', (req, res) => {
  if (req.session.username) {
    res.send(`
      <h2>Welcome back 23A91A6134  ${req.session.username}</h2>
      <p><a href="/logout">Logout</a></p>
    `);
  } else {
    res.send(`
      <h2>Hello, 23A91A6134!</h2>
      <form method="POST" action="/login">
        <input type="text" name="username" placeholder="Enter your name" required>
        <button type="submit">Login</button>
      </form>
    `);
  }
});

// Login route
app.post('/login', (req, res) => {
  const username = req.body.username;
  req.session.username = username;
  res.cookie('username', username); // set cookie
  res.redirect('/');
});
```

```
// Logout route
app.get('/logout', (req, res) => {
   res.clearCookie('username'); // clear cookie
   req.session.destroy(err => {
      res.redirect('/');
   });
});

app.listen(port, () => {
   console.log(`Server is running on http://localhost:${port}`);
});
```

OUTPUT:



23A91A6134
Hello, guest!



Welcome back 23A91A6134 kesava naidu

Logout

B) AIM: Write a program for user authentication.

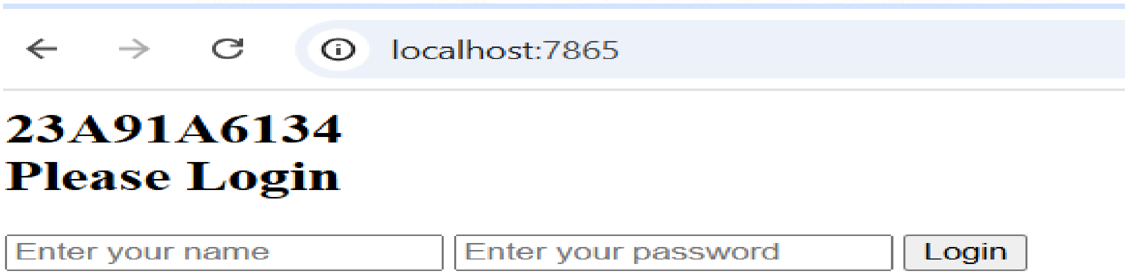
DESCRIPTION: This program implements a basic user authentication system using a server-side web application. It allows users to log in with a valid username and password and manages their authenticated session until they log out.

PROGRAM:

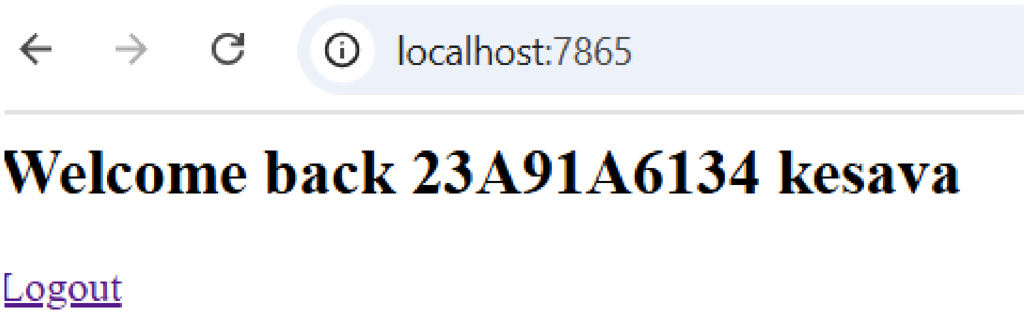
```
const express = require('express');
const session = require('express-session');
const bodyParser = require('body-parser');
const app = express();
const port = 7865;

const users = [
   { username: 'kesava', password: '1234' },
   { username: 'aditya', password: '1431' },
    { username: 'veerana', password: '5678' },
   { username: 'venkat', password: '9876' },
];

app.use(bodyParser.urlencoded({ extended: true }));
app.use(session({
  secret: 'mysecret',
  resave: false,
  saveUninitialized: true,
  cookie: { maxAge: 60000 } // session expires in 1 minute
}));

// Home route
app.get('/', (req, res) => {
  if (req.session.username) {
     res.send(`
        <h2>Welcome back 23A91A6114 ${req.session.username}</h2>
        <p><a href="/logout">Logout</a></p>
     `);
  } else {
     res.send(`
        <h2>23A91A6114 <br> Please Login</h2>
        <form method="POST" action="/login">
           <input type="text" name="username" placeholder="Enter your name" required>
           <input type="password" name="password" placeholder="Enter your password"
required>
           <button type="submit">Login</button>
        </form>
     `);
  }
});

// Login handler
```

```
app.post('/login', (req, res) => {
    const { username, password } = req.body;
    const user = users.find(u => u.username === username && u.password === password);
    if (user) {
        req.session.username = username;
        res.redirect('/');
    } else {
        res.send('<h2>Invalid credentials</h2><a href="/">Try again</a>');
    }
});

// Logout route
app.get('/logout', (req, res) => {
    req.session.destroy(err => {
        if (err) {
            return res.send('<h2>Error logging out</h2>');
        }
        res.redirect('/');
    });
});

app.listen(port, () => {
    console.log(`Server is running on http://localhost:${port}`);
});
```

OUTPUT:



23A91A6134
Please Login



Welcome back 23A91A6134 kesava

Logout