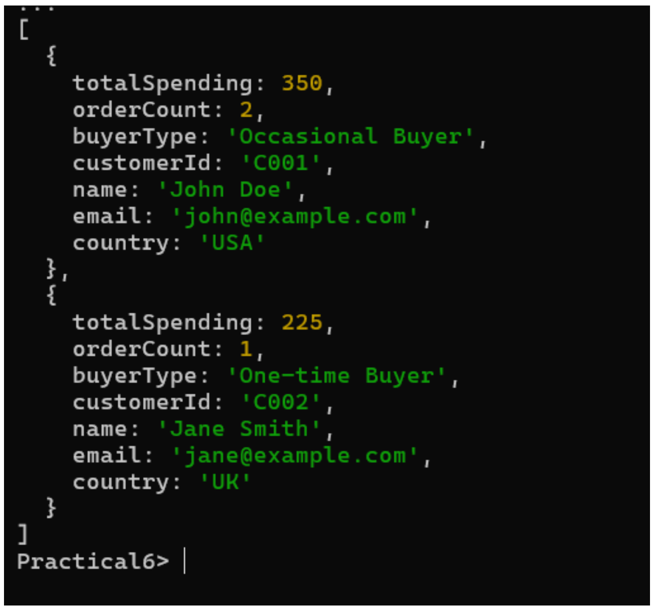
```
Microsoft Windows [Version
10.0.26200.7171]
(c) Microsoft Corporation. All rights
reserved.

C:\Users\Pabitra>mongosh
Current Mongosh Log ID:
691ab5423362b749b763b111
Connecting to:
mongodb://127.0.0.1:27017/?directConnecti
on=true&serverSelectionTimeoutMS=2000&
appName=mongosh+2.5.9
Using MongoDB:          8.0.13
Using Mongosh:          2.5.9

For mongosh info see:
https://www.mongodb.com/docs/mongodb-s
hell/

------
   The server generated these startup
warnings when booting
   2025-11-15T11:56:20.047+05:30: Access
control is not enabled for the database.
Read and write access to data and
configuration is unrestricted
------

test> use Practical6
switched to db Practical6
Practical6> db.orders.insertMany([
...   {
...     orderId: "ORD1001",
...     customerId: "C001",
...     orderDate:
ISODate("2025-09-01T10:30:00Z"),
...     items: [
...       { productId: "P001", price: 100,
quantity: 2 },
...       { productId: "P002", price: 50,
quantity: 1 }
...     ]
...   },
...   {
...     orderId: "ORD1002",
...     customerId: "C001",
...     orderDate:
ISODate("2025-09-05T12:00:00Z"),
...     items: [
...       { productId: "P001", price: 100,
quantity: 1 }
...     ]
...   },
...   {
...     orderId: "ORD1003",
...     customerId: "C002",
...     orderDate:
ISODate("2025-10-01T08:20:00Z"),
...     items: [
...       { productId: "P003", price: 75,
quantity: 3 }
...     ]
...   }
... ])
...
{
  acknowledged: true,
  insertedIds: {
    '0':
ObjectId('691ab6163362b749b763b112'),
    '1':
ObjectId('691ab6163362b749b763b113'),
    '2':
ObjectId('691ab6163362b749b763b114')
  }
}
Practical6> db.customers.insertMany([
...   {
...     customerId: "C001",
...     name: "John Doe",
...     email: "john@example.com",
...     country: "USA"
...   },
...   {
...     customerId: "C002",
...     name: "Jane Smith",
...     email: "jane@example.com",
...     country: "UK"
...   }
... ])
...
{
  acknowledged: true,
  insertedIds: {
    '0':
ObjectId('691ab61b3362b749b763b115'),
    '1':
ObjectId('691ab61b3362b749b763b116')
  }
}
Practical6> db.orders.aggregate([
...   {
...     $addFields: {
...       totalOrderAmount: {
...         $sum: {
...           $map: {
...             input: "$items",
...             as: "item",
...             in: { $multiply: [ "$$item.price",
"$$item.quantity" ] }
...           }
...         }
...       }
...     }
...   },
...   {
...     $group: {
...       _id: "$customerId",
...       totalSpending: { $sum:
"$totalOrderAmount" },
...       orderCount: { $sum: 1 },
...       orders: { $push: "$$ROOT" }
...     }
...   },
...   {
...     $lookup: {
...       from: "customers",
...       localField: "_id",
...       foreignField: "customerId",
...       as: "customerInfo"
...     }
...   },
...   { $unwind: "$customerInfo" },
...   {
...     $addFields: {
...       buyerType: {
...         $switch: {
...           branches: [
...             { case: { $gt: [ "$orderCount", 5 ]
}, then: "Frequent Buyer" },
...             { case: { $and: [ { $gte: [
"$orderCount", 2 ] }, { $lte: [ "$orderCount",
5 ] } ] }, then: "Occasional Buyer" }
...           ],
...           default: "One-time Buyer"
...         }
...       }
...     }
...   },
...   { $sort: { totalSpending: -1 } },
...   {
...     $project: {
...       _id: 0,
...       customerId: "$_id",
...       name: "$customerInfo.name",
...       email: "$customerInfo.email",
...       country: "$customerInfo.country",
...       totalSpending: 1,
...       orderCount: 1,
...       buyerType: 1
...     }
...   }
... ])
```



```
...
[
  {
    totalSpending: 350,
    orderCount: 2,
    buyerType: 'Occasional Buyer',
    customerId: 'C001',
    name: 'John Doe',
    email: 'john@example.com',
    country: 'USA'
  },
  {
    totalSpending: 225,
    orderCount: 1,
    buyerType: 'One-time Buyer',
    customerId: 'C002',
    name: 'Jane Smith',
    email: 'jane@example.com',
    country: 'UK'
  }
]
Practical6>
```

App. js

```
import React, { useState } from "react";
import "./App.css";

function App() {
  const [email, setEmail] = useState("");
  const [password, setPassword] =
useState("");
  const [error, setError] = useState("");

  const handleSubmit = (e) => {
    e.preventDefault();

    if (!email || !password) {
      setError("Please fill in both fields");
      return;
    }

    // Example check login (you can replace
with real API call)
    if (email === "user@example.com" &&
password === "123456") {
      alert("Login successful!");
      setError("");
    } else {
setError("Invalid email or password");
    }
  };

  return (
    <div className="login-container">
      <form className="login-form"
onSubmit={handleSubmit}>
        <h2>Login</h2>
        {error && <p
className="error">{error}</p>}

        <label>Email</label>
        <input
          type="email"
          placeholder="Enter your email"
          value={email}
          onChange={(e) =>
setEmail(e.target.value)}
        />

        <label>Password</label>
        <input
          type="password"
          placeholder="Enter your password"
          value={password}
          onChange={(e) =>
setPassword(e.target.value)}
        />

        <button type="submit">Login</button>
      </form>
    </div>
  );
}

export default App;
```

App. css

```
body {
  font-family: Arial, sans-serif;
  background: #f0f2f5;
  margin: 0;
  display: flex;
  height: 100vh;
  justify-content: center;
  align-items: center;
}

.login-container {
  background: white;
  padding: 40px;
  border-radius: 10px;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
  width: 350px;
}

.login-form h2 {
  text-align: center;
  margin-bottom: 20px;
}

.login-form label {
  display: block;
  margin-top: 10px;
  font-weight: bold;
}

.login-form input {
  width: 100%;
  padding: 10px;
  margin-top: 5px;
  border: 1px solid #ccc;
  border-radius: 5px;
}

.login-form button {
  width: 100%;
  padding: 10px;
  background-color: #007bff;
  color: white;
  border: none;
  border-radius: 5px;
  margin-top: 15px;
  cursor: pointer;
  font-size: 16px;
}

.login-form button:hover {
  background-color: #0056b3;
}

.error {
  color: red;
  text-align: center;
  margin-bottom: 10px;
}
```

8

In command prompt

```
mkdir registration-app

cd registration-app

npm init -y

npm install express mongoose
body-parser cors
```

User. js

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

const userSchema = new
mongoose.Schema({
  name: { type: String, required: true,
minlength: 2 },
  email: { type: String, required: true, unique:
true, match: /^\S+@\S+\.\S+$/ },
  password: { type: String, required: true,
minlength: 6 }
});
module.exports = mongoose.model('User',
userSchema);
```

Server.js

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');
const User = require('./models/user');

const app = express();
const PORT = 3000;

app.use(cors());
app.use(bodyParser.json());
app.use(express.static('public'));

mongoose.connect('mongodb://127.0.0.1:27
017/registrationdb', {
  useNewUrlParser: true,
  useUnifiedTopology: true
}).then(() => console.log('✅ MongoDB
connected'))
  .catch(err => console.log(err));

app.post('/register', async (req, res) => {
  try {
    const { name, email, password } =
req.body;
    if (!name || !email || !password) {
      return res.status(400).json({ error: 'All
fields are required.' });
    }
    const user = new User({ name, email,
password });
    await user.save();
    res.status(200).json({ message: '🎉 User
registered successfully!' });
  } catch (err) {
    let msg = 'Error registering user: ';
    if (err.code === 11000) { // Duplicate
email
      msg += 'Email already exists.';
    } else {
      msg += err.message;
    }
    res.status(400).json({ error: msg });
  }
});

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

Index. Html

```
<!DOCTYPE html>
```

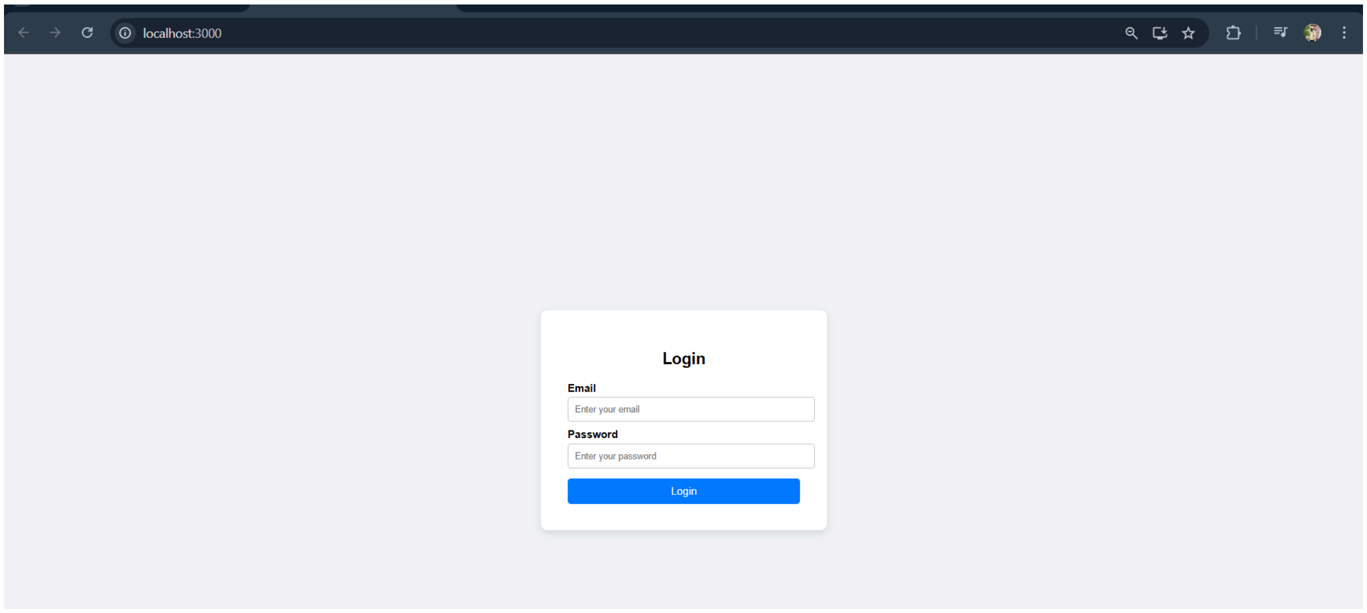
```
<html lang="en" ng-app="registrationApp">
<head>
  <meta charset="UTF-8">
  <title>User Registration</title>
  <meta name="viewport"
content="width=device-width,
initial-scale=1">
  <script
src="https://ajax.googleapis.com/ajax/libs/a
ngularjs/1.8.3/angular.min.js"></script>
  <script src="app.js"></script>
  <style>
    body {
      font-family: 'Segoe UI', 'Inter', Arial,
sans-serif;
      background: linear-gradient(135deg,
#667eea 0%, #764ba2 100%);
      margin: 0;
      padding: 0;
      display: flex;
      align-items: center;
      justify-content: center;
      height: 100vh;
    }
    .form-container {
      background: white;
      padding: 40px 30px;
      border-radius: 12px;
      box-shadow: 0 4px 20px
rgba(102,126,234,0.15);
      min-width: 320px;
      width: 400px;
    }
    h2 {
      color: #667eea;
      text-align: center;
      margin-bottom: 20px;
      font-weight: 700;
    }
    form {
      display: flex;
      flex-direction: column;
      gap: 14px;
    }
    input {
      padding: 12px;
      border: 1px solid #ccc;
      border-radius: 6px;
      font-size: 16px;
      transition: border 0.2s;
    }
    input:focus {
      border-color: #764ba2;
      outline: none;
    }
    button {
      padding: 12px;
      background: linear-gradient(135deg,
#667eea 0%, #764ba2 100%);
      color: white;
      border: none;
      border-radius: 6px;
      font-size: 17px;
      font-weight: 600;
      cursor: pointer;
      transition: background 0.2s;
    }
    button:hover {
      background: linear-gradient(135deg,
#764ba2 0%, #667eea 100%);
    }
    .success {
      color: #1a8c1a;
      background: #e8fae8;
      border-radius: 4px;
      text-align: center;
      padding: 8px;
      margin-top: 10px;
      font-weight: 500;
      animation: fadeIn 0.7s;
    }
    .error {
      color: #c33;
      background: #fee;
      border-radius: 4px;
      text-align: center;
      padding: 8px;
      margin-top: 10px;
      font-weight: 500;
      animation: shake 0.4s;
    }
    @keyframes shake {
      0%,100%{transform:translateX(0);}
20%{transform:translateX(-5px);}
60%{transform:translateX(5px);}
    }
    @keyframes fadeIn {
      0% { opacity: 0; }
      100% { opacity: 1; }
    }
    @media (max-width: 500px) {
      .form-container { width: 98vw; padding:
18px 6vw; }
    }
  </style>
</head>
<body
ng-controller="RegistrationController">
  <div class="form-container">
    <h2>Register Account</h2>
    <form ng-submit="registerUser()"
id="registrationForm" autocomplete="off"
novalidate>
      <input type="text"
ng-model="user.name" placeholder="Full
Name" required>
      <input type="email"
ng-model="user.email" placeholder="Email
Address" required>
      <input type="password"
ng-model="user.password"
placeholder="Password (min 6 chars)"
required minlength="6">
      <button
type="submit">Register</button>
    </form>
    <p class="success"
ng-if="successMessage">{{successMessag
e}}</p>
    <p class="error"
ng-if="errorMessage">{{errorMessage}}</p>
  </div>
</body>
</html>
```

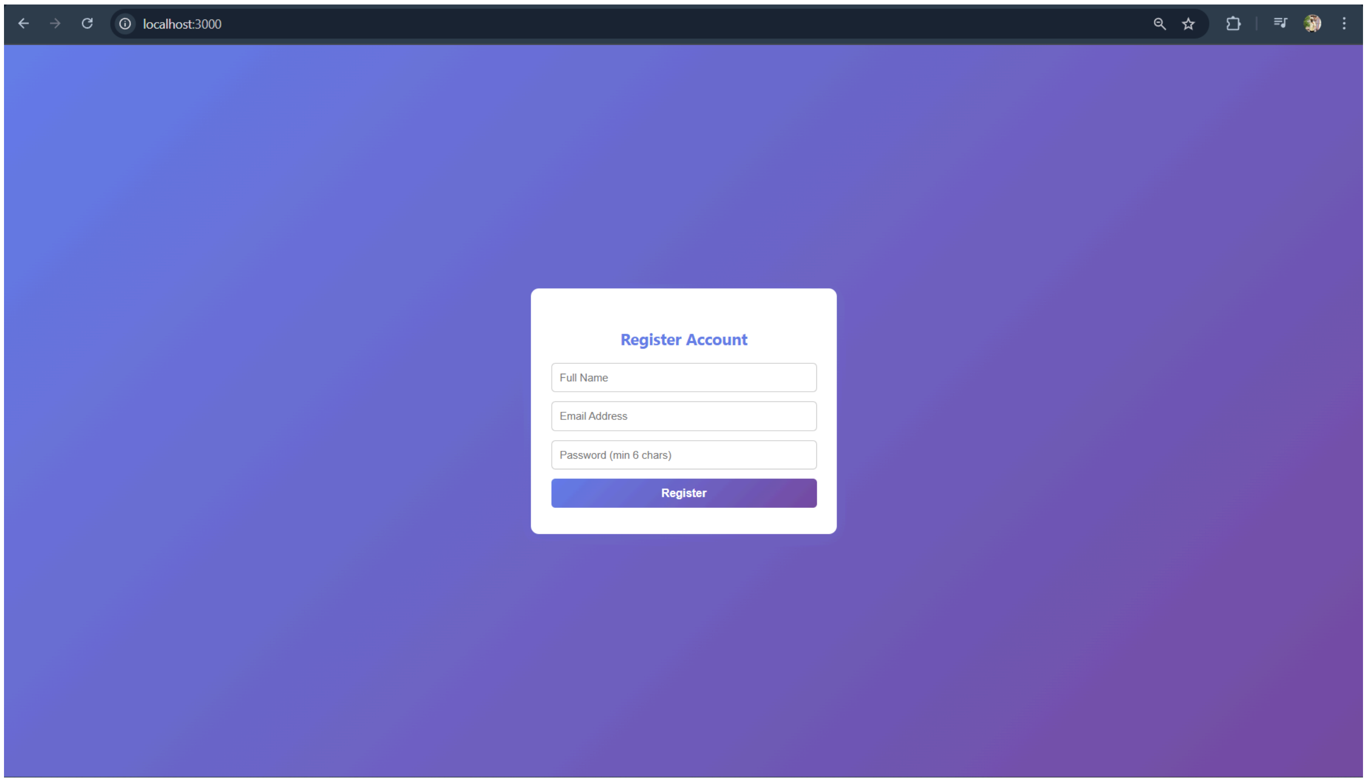
App. js

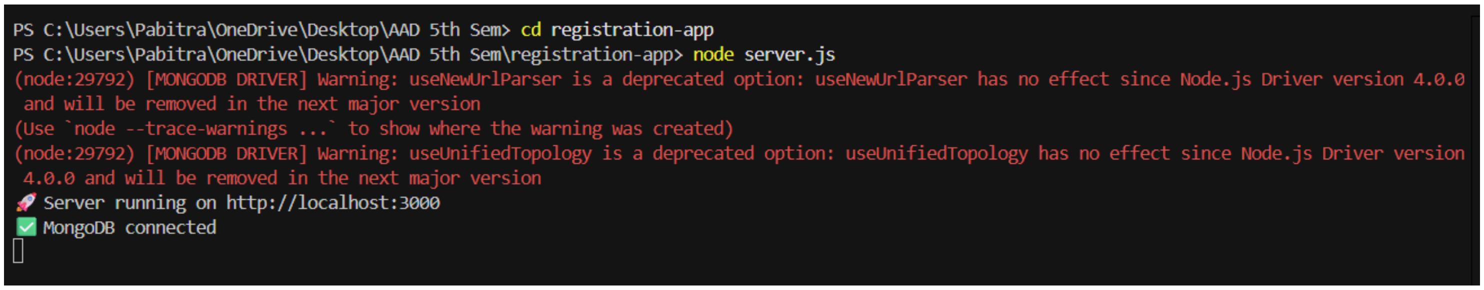
```
var app = angular.module('registrationApp',
[]);

app.controller('RegistrationController',
function($scope, $http) {
  $scope.user = {};
  $scope.successMessage = '';
  $scope.errorMessage = '';

  $scope.registerUser = function() {
    $http.post('/register', $scope.user)
      .then(function(response) {
        $scope.successMessage =
response.data.message;
        $scope.errorMessage = '';
        $scope.user = {};
      }, function(error) {
        $scope.errorMessage =
error.data.error;
        $scope.successMessage = '';
      });
  };
});
```







```
PS C:\Users\Pabitra\OneDrive\Desktop\AAD 5th Sem> cd registration-app
PS C:\Users\Pabitra\OneDrive\Desktop\AAD 5th Sem\registration-app> node server.js
(node:29792) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option: useNewUrlParser has no effect since Node.js Driver version 4.0.0
 and will be removed in the next major version
(Use `node --trace-warnings ...` to show where the warning was created)
(node:29792) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated option: useUnifiedTopology has no effect since Node.js Driver version
 4.0.0 and will be removed in the next major version
🚀 Server running on http://localhost:3000
✅ MongoDB connected
```