Marwadi University	**Marwadi University** **Faculty of Technology** **Department of Information and Communication Technology**	
Subject: AWT (01CT1625)	**Aim: Develop a web application which involves database operations using NodeJS.**	
Experiment No: 2	**Date:**	**Enrollment No: 92200133005**

AIM: Develop a web application which involves database operations using NodeJS.

CODE:

```
// user.js
const mongoose = require('mongoose');
const userSchema = new mongoose.Schema({
  name: {
    type: String,
    required: true },
  email: {
    type: String,
    required: true,
    unique: true },
  age: {
    type: Number,
    required: true },
  createdAt: {
    type: Date,
    default: Date.now
  } });
module.exports = mongoose.model('User', userSchema);
//server.js
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
```

Marwadi University	**Marwadi University** **Faculty of Technology** **Department of Information and Communication Technology**	
Subject: AWT (01CT1625)	**Aim: Develop a web application which involves database operations using NodeJS.**	
Experiment No: 2	**Date:**	**Enrollment No: 92200133005**

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

const path = require('path');

// Import routes

const userRoutes = require('./routes/users');

// Initialize express app

const app = express();

const PORT = process.env.PORT || 3000;

// Middleware

app.use(cors());

app.use(bodyParser.json());

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

app.use(express.static(path.join(__dirname, 'public')));

// Connect to MongoDB

mongoose.connect('mongodb://localhost:27017/practical2', {

  useNewUrlParser: true,

  useUnifiedTopology: true

})

.then(() => console.log('MongoDB connected successfully'))

.catch(err => console.error('MongoDB connection error:', err));

// Routes

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

// Serve the main HTML file

app.get('/', (req, res) => {

  res.sendFile(path.join(__dirname, 'public', 'index.html'));
```

```
});

// Start the server

app.listen(PORT, () => {

  console.log(`Server running on port ${PORT}`);

});

//Index.html

<script>

        document.addEventListener('DOMContentLoaded', function() {

            const userForm = document.getElementById('userForm');

            const usersList = document.getElementById('usersList');

            const cancelBtn = document.getElementById('cancelBtn');

            // Load users on page load

            fetchUsers(); // Form submission

            userForm.addEventListener('submit', function(e) {

                e.preventDefault();

                const userId = document.getElementById('userId').value;

                const userData = {

                    name: document.getElementById('name').value,

                    email: document.getElementById('email').value,

                    age: document.getElementById('age').value

                };

                if (userId) {

                    // Update existing user

                    updateUser(userId, userData);
```

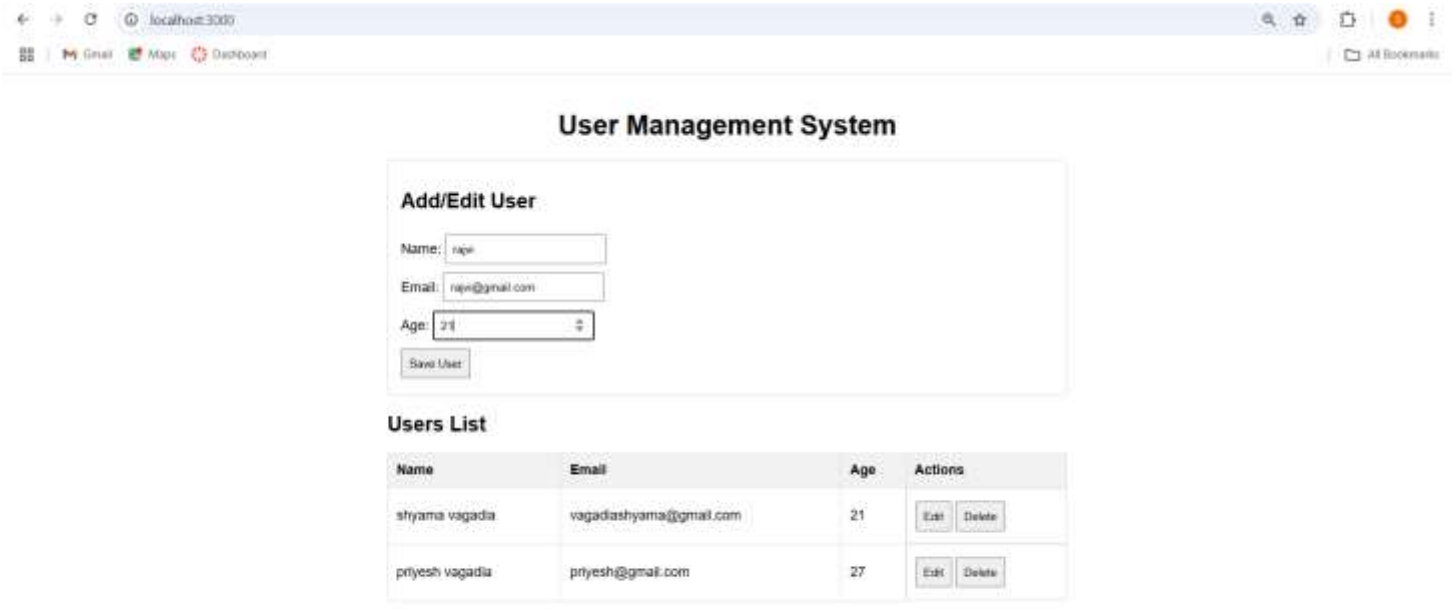
```
    } else {
        // Create new user
        createUser(userData);
    }
}); // Cancel edit
cancelBtn.addEventListener('click', function() {
    resetForm();
}); // Fetch all users
function fetchUsers() {
    fetch('/api/users')
        .then(response => response.json())
        .then(users => {
            usersList.innerHTML = '';
            users.forEach(user => {
                const row = document.createElement('tr');
                row.innerHTML = `
                    <td>${user.name}</td>
                    <td>${user.email}</td>
                    <td>${user.age}</td>
                    <td class="actions">
                        <button onclick="editUser('${user._id}')">Edit</button>
                        <button onclick="deleteUser('${user._id}')">Delete</button>
                    </td>`;
                usersList.appendChild(row);
```

```
        });})
      .catch(error => console.error('Error fetching users:', error));
} // Create a new user
function createUser(userData) {
  fetch('/api/users', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json' },
    body: JSON.stringify(userData)
  })
  .then(response => response.json())
  .then(() => {
    resetForm();
    fetchUsers();
  })
  .catch(error => console.error('Error creating user:', error));
}// Update an existing user
function updateUser(userId, userData) {
  fetch(`/api/users/${userId}`, {
    method: 'PUT',
    headers: {
      'Content-Type': 'application/json' },
    body: JSON.stringify(userData)
  })
```

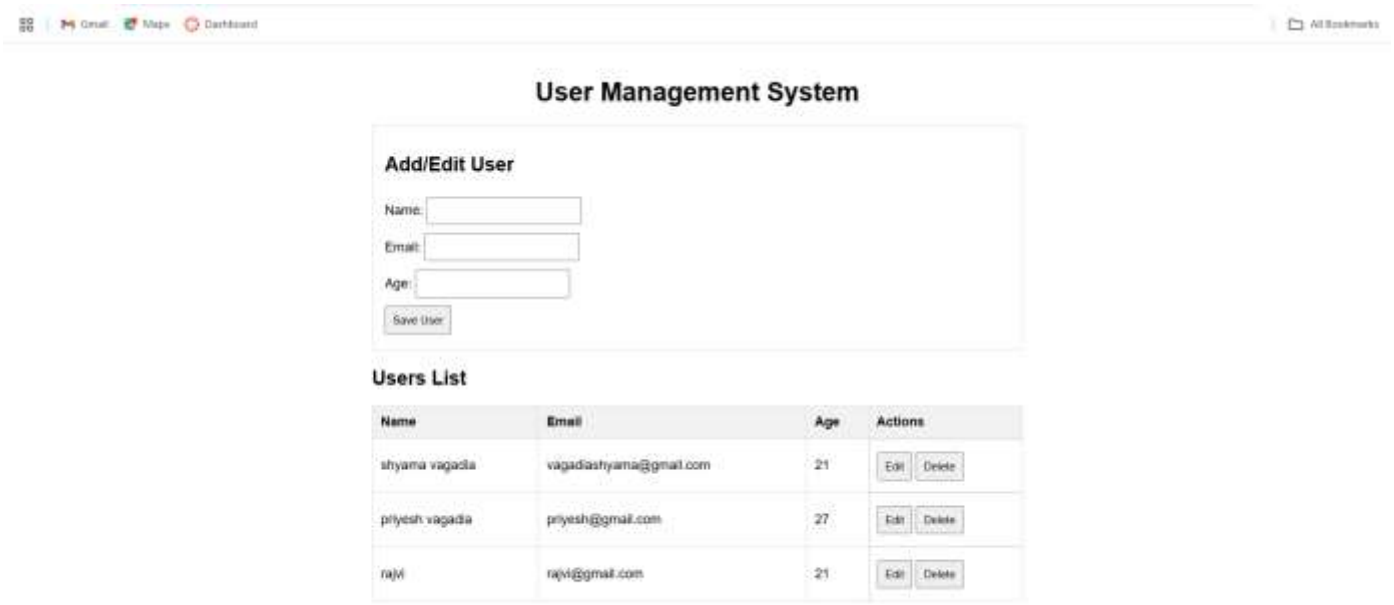
```
    .then(response => response.json())
    .then(() => {
        resetForm();
        fetchUsers();})
    .catch(error => console.error('Error updating user:', error));
} // Reset the form
function resetForm() {
    document.getElementById('userId').value = '';
    document.getElementById('name').value = '';
    document.getElementById('email').value = '';
    document.getElementById('age').value = '';
    document.getElementById('cancelBtn').style.display = 'none';
} // Make these functions available globally
window.editUser = function(userId) {
    fetch(`/api/users/${userId}`)
        .then(response => response.json())
        .then(user => {
            document.getElementById('userId').value = user._id;
            document.getElementById('name').value = user.name;
            document.getElementById('email').value = user.email;
            document.getElementById('age').value = user.age;
            document.getElementById('cancelBtn').style.display = 'inline';
        })
        .catch(error => console.error('Error fetching user:', error));
```

```
    };
    window.deleteUser = function(userId) {
      if (confirm('Are you sure you want to delete this user?')) {
        fetch(`/api/users/${userId}`, {
          method: 'DELETE'
        })
        .then(() => {
          fetchUsers();})
        .catch(error => console.error('Error deleting user:', error));
      }};});
</script>
```

OUTPUT:



Marwadi University	**Marwadi University** **Faculty of Technology** **Department of Information and Communication Technology**	
Subject: AWT (01CT1625)	**Aim: Develop a web application which involves database operations using NodeJS.**	
Experiment No: 2	**Date:**	**Enrollment No: 92200133005**




```
PS C:\Users\Priyesh Vagadia\Desktop\Semester 6\AWT\practical-2> node server.j
s
(node:14412) [MONGODB DRIVER] Warning: useNewUrlParser is a deprecated option
: useNewUrlParser has no effect since Node.js Driver version 4.0.0 and will b
e removed in the next major version
(Use `node --trace-warnings ...` to show where the warning was created)
(node:14412) [MONGODB DRIVER] Warning: useUnifiedTopology is a deprecated opt
ion: useUnifiedTopology has no effect since Node.js Driver version 4.0.0 and
will be removed in the next major version
Server running on port 3000
MongoDB connected successfully
```