EMPLOYEE DATABASE SYSTEM

A PROJECT REPORT

Submitted for

CSE3009-NOSQL DATABASES

By

Terala Srujan	(18mis7202)
D.Nishankh	(18mis7281)
V.Sridatta varma	(18mis7125)

SCHOOL OF COMPUTER SCIENCE ENGINEERING
VIT-AP UNIVERSITY
AMARAVATI- 522237
May 2021

ABSTRACT :

This project totally focuses on management of employee data of a particular organisation as Employees are the most valuable assets of an organization, and managing them properly helps to ensure the success of our business. Employee data has to be maintained carefully to make informed business decisions. The HR department is also legally mandated to manage and protect employee data, and failing to do so will lead to heavy penalties. The problem is that managing a large repository of employee data manually comes with many challenges. It can be time-consuming and error-prone, especially when your organization has a huge number of employees and there are many other vital tasks to take care of. Security is also a major concern associated with manual database management.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Employee Management System is a distributed application, developed to maintain the details of employees working in any organization. The Employee Management System has been developed to overcome the problems existing in the manual system. This project aims to simplify the task of maintaining records of the employees of Company. To develop a well-designed database to store employee information provides full functional reports to management of Company.

It is simple to understand and can be used by anyone who is not even familiar with simple employees' system. It is user friendly and just asks the user to follow step by step operations by giving him few options. It is fast and can perform many operations of a company. This application is helpful to department of the organization which maintains data of employees related to an organization.

1.1 Objectives

The objective of this project is to:

- Create new users to the system accordingly.
- provide a comprehensive approach towards the management of employee information.
- Helps in maintaining the computerized employee details.
- more efficient and reliable to use.
- can store huge data with less computer memory.

1.2 Background and Literature Survey

The proposed system is designed to eliminate all the drawbacks of the existing system. The system shall be responsible for maintaining information about employees, positions, company benefits, departments, warnings, administration.

This system will reduce the complexity of employee management. By using this system, we can easily maintain all the records about employees. It will reduce searching time. It can be easily handled by the person who have elementary knowledge of computer because it provides a user-friendly environment. In this world of growing technologies everything has been computerized. With large number of works opportunities, the Human workforce has increased. Thus, there is a need of a system which can handle the data of such a large number of Employees in an organization. This project simplifies the task of maintain records because of its user-friendly nature.

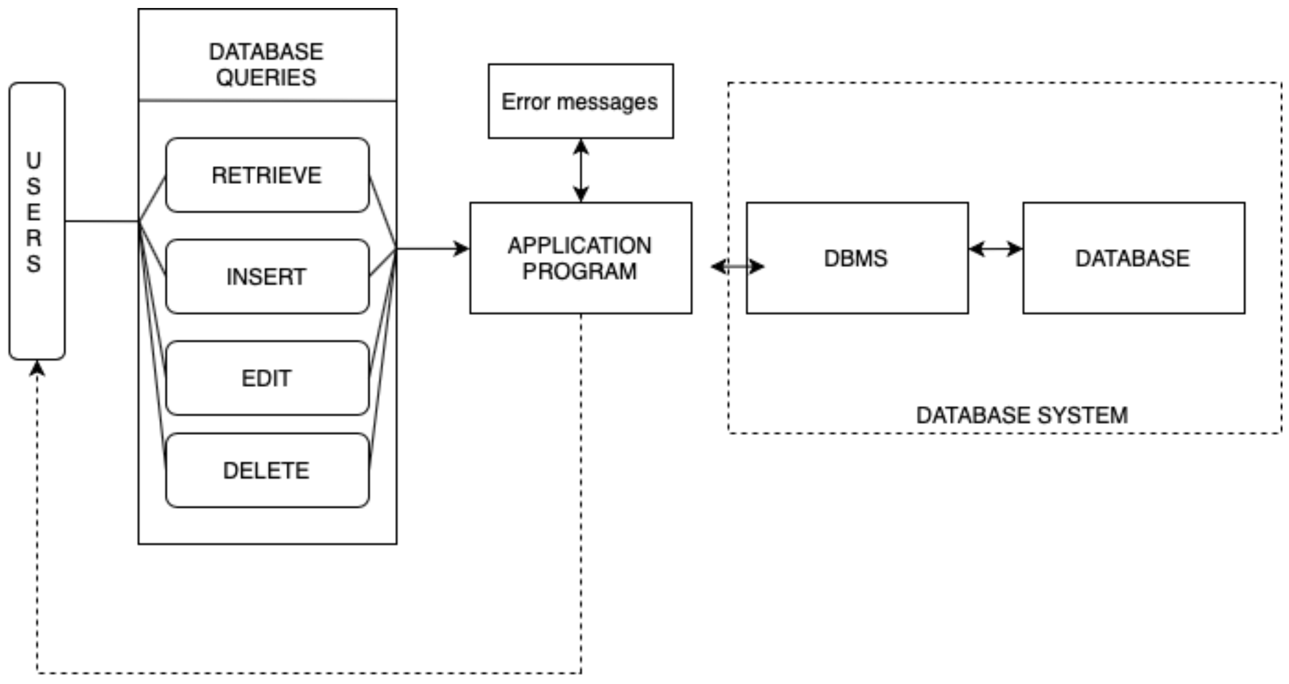
1.3 Organization of the Report

The remaining chapters of the project report are described as follows:

- Chapter 2 contains the proposed system, methodology and software details.
- Chapter 3 gives the cost involved in the implementation of the project.
- Chapter 4 discusses the results obtained after the project was implemented.
- Chapter 5 concludes the report.
- Chapter 6 consists of codes.
- Chapter 7 gives references.

CHAPTER-2

2.1 Proposed System



2.2 Working Methodology

The project is completely based on software with two sections, front-end and back-end. The front-end consists of an HTML(HyperText Markup Language) file , CSS(Cascading Style Sheets) file , javascript files , embedded java script files , node js and mongo db.

As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications Thread-based networking is relatively inefficient and very difficult to use. Furthermore, users of Node.js are free from worries of dead-locking the process, since there are no locks. Almost no function in Node.js directly performs I/O, so the process never blocks. Because nothing blocks, scalable systems are very reasonable to develop in Node.js.

MongoDB is an object-oriented, simple, dynamic, and scalable NoSQL database. It is based on the NoSQL document store model. The data objects are stored as separate documents inside a collection — instead of storing the data into the columns and rows of a traditional relational database

EJS simply stands for Embedded Javascript. It is a simple templating language/engine that lets its user generate HTML with plain javascript. EJS is mostly useful whenever you have to output HTML with a lot of javascript.

Vs code is the main code editor we have used.

CHAPTER-3

DATABASE ANALYSIS

The Database we used is mongo db which is a open source software and the attributes we have used are :

1. Name
2. Designation
3. Salary

We have inserted 523 rows into the database

CHAPTER 4

RESULTS AND DISCUSSIONS

Application consists of a home page and it navigates to :

1.	search bar	-search the user in database
2.	Edit bar	-provides a option to update elements In database
3.	Registration	-to add new user to database
4.	Delete	-To delete the user

When a invalid user is being searched it shows error message not found.
When a profile is updated a message pops up stating that profile is updated.

CHAPTER 5

CONCLUSION AND FUTURE WORK

This project is built keeping in mind that it is to be used by only one user that is the admin. It is built for use in small scale organisation where the number of employees is limited. According to the requested requirement the admin can add, manipulate, update and delete all employee data in his organisation. The admin can add new departments and delete them. The required records can be easily viewed by the admin anytime time he wants in an instant. The payment of the employee is based on monthly basis. Numerous validations implemented would enable the admin to enter accurate data. The main objective of this framework is to save time, make the system cost effective and management records efficiently. The project can be extended to do :

- Admin could accessible the data around the world using any device.
- Build a communication access for employee and admin to generate requests subject to approval.
- Add attendance system and payroll history tabs.
- Build cost effective with more features for smaller organisations.

CHAPTER 6

APPENDIX (ONLY NOSQL CODE)

app.js

```
const express = require('express');
const app = express();
const path = require('path');
const dotenv = require('dotenv');
const methodOverride = require('method-override');
const session = require('express-session') ;
const flash = require('connect-flash');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const employeeRoutes = require('./routes/employees');
dotenv.config({path : './config.env'});
// Connecting to mongodb database
mongoose.connect(process.env.DATABASE_LOCAL, {
    useNewUrlParser : true,
    useUnifiedTopology : true,
    useCreateIndex: true
});
app.use(bodyParser.urlencoded({extended:true}));
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'ejs');
app.use(express.static('public'));
//middleware for  method override
app.use(methodOverride('_method'));
//middleware for express session
app.use(session({
    secret : "nodejs",
    resave : true,
    saveUninitialized:true
}));
//middleware for connect flash
app.use(flash());
//Setting messages variables globally
app.use((req, res, next)=> {
    res.locals.success_msg = req.flash(('success_msg'));
    res.locals.error_msg = req.flash(('error_msg'));
    next();
});
app.use(employeeRoutes);
const port = process.env.PORT;
app.listen(port, ()=> {
    console.log('Server is started.');
});
```

Config.env

DATABASE_LOCAL = mongodb://localhost:27017/employee

employee.js

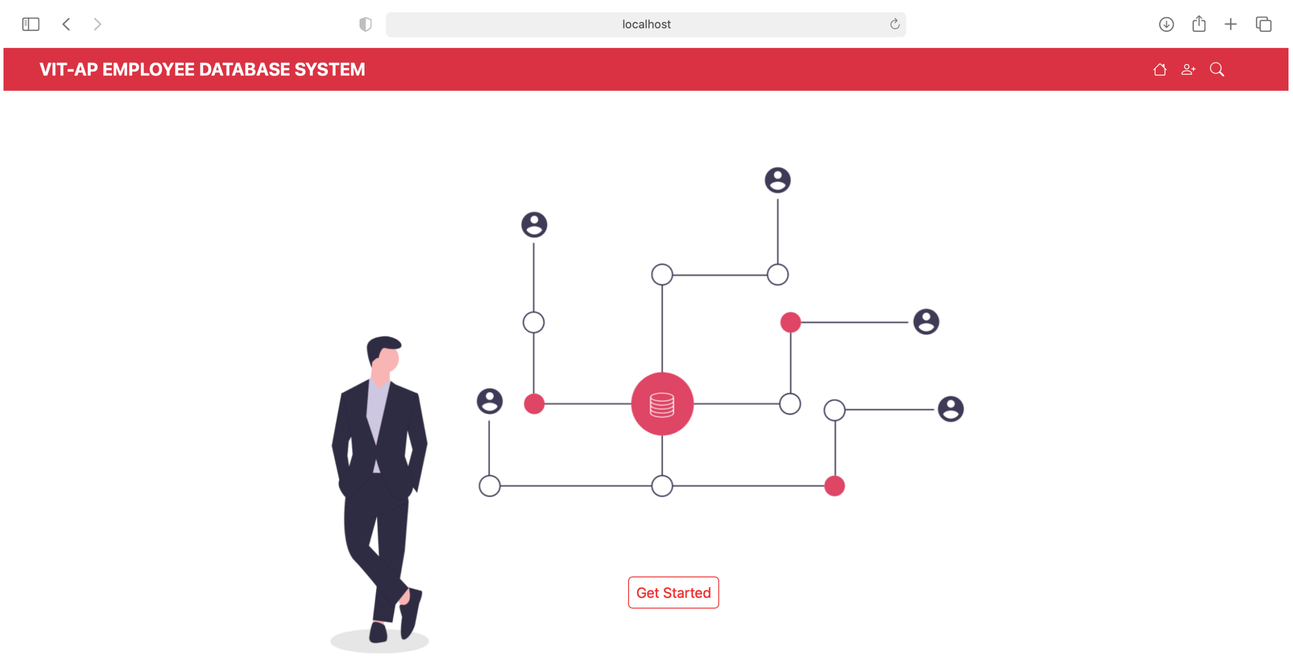
```
const express = require('express');
const router = express.Router();
const Employee = require('../models/employee');
//get routes starts here
router.get('/', (req, res)=> {
    Employee.find({})
        .then(employees => {
            res.render('index', {employees : employees});
        })
        .catch(err=> {
            req.flash('error_msg', 'ERROR: '+err)
            res.redirect('/');
        })

});
router.get('/employee/new', (req,res)=> {
    res.render('new');
});
router.get('/employee/search', (req,res)=> {
    res.render('search', {employee:""});
});
```

```
router.get('/employee/home', (req,res)=> {
    res.render('home');
});
router.get('/employee', (req,res)=> {
    let searchQuery = {name : req.query.name};
    Employee.findOne(searchQuery)
        .then(employee => {
            res.render('search', {employee:employee});
        })
        .catch(err => {
            req.flash('error_msg', 'ERROR: '+err)
            res.redirect('/');
        });
});
router.get('/edit/:id', (req, res)=> {
    let searchQuery = {_id : req.params.id};
    Employee.findOne(searchQuery)
        .then(employee => {
            res.render('edit', {employee:employee});
        })
        .catch(err => {
            req.flash('error_msg', 'ERROR: '+err)
            res.redirect('/');
        });
});
//get routes ends here
//post routes starts here
router.post('/employee/new', (req,res)=> {
    let newEmployee = {
        name : req.body.name,
        designation : req.body.designation,
        salary : req.body.salary
    };
    Employee.create(newEmployee)
        .then(employee => {
            req.flash('success_msg', 'Employee data added to 
database successfully.')
            res.redirect('/');
        })
        .catch(err => {
            req.flash('error_msg', 'ERROR: '+err)
            res.redirect('/');
        });
});
//post routes end here
//put routes starts here
router.put('/edit/:id', (req, res)=> {
    let searchQuery = {_id : req.params.id};
    Employee.updateOne(searchQuery, {$set: {
        name : req.body.name,
        designation : req.body.designation,
        salary : req.body.salary
    }})
    .then(employee => {
        req.flash('success_msg', 'Employee data updated 
successfully.')
        res.redirect('/');
    })
    .catch(err => {
        req.flash('error_msg', 'ERROR: '+err)
        res.redirect('/');
    });
});
//put routes ends here
//delete routes starts here
router.delete('/delete/:id', (req, res)=> {
    let searchQuery = {_id : req.params.id};
    Employee.deleteOne(searchQuery)
        .then(employee=>{
            req.flash('success_msg', 'Employee deleted 
successfully.')
            res.redirect('/');
        })
        .catch(err => {
            req.flash('error_msg', 'ERROR: '+err)
            res.redirect('/');
        });
});
//delete routes ends here
module.exports = router;
```

Employee.js

```
const mongoose = require('mongoose');
let employeeScheme = new mongoose.Schema({
    name : String,
    designation : String,
    salary : Number
});
module.exports = mongoose.model('Employee', employeeScheme);
```

home.ejs



Code

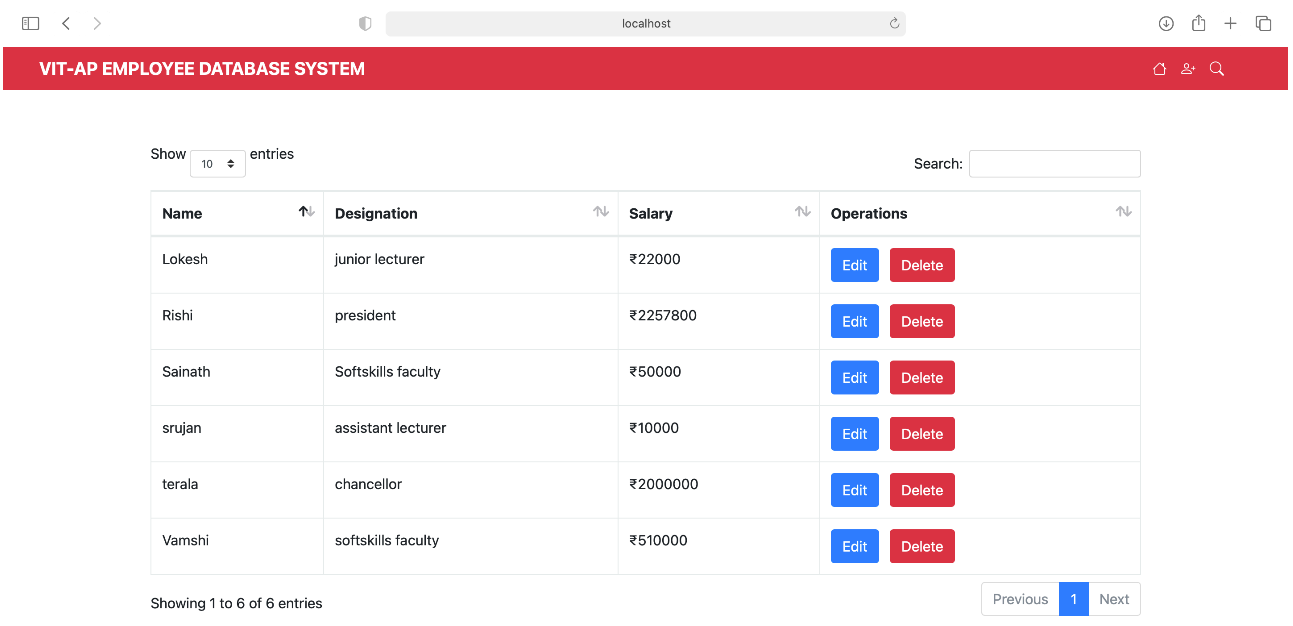
```
<%- include('partials/header') -%>
<style>
    body{
 background-image: url("https://static.wixstatic.com/media/
ee5f36_0110f5d09c4a43c39a280cb4ca82f552~mv2.png/v1/fill/
w_2304,h_1149,fp_0.50_0.50/
ee5f36_0110f5d09c4a43c39a280cb4ca82f552~mv2.png");
    background-repeat: no-repeat;
    background-size: cover;
    }
    .button {
  background-color: white;
  color: rgb(240, 47, 47);
  border: 1px solid rgb(240, 47, 47);
  border-radius: 5px;
  padding: 8px;
}
.button:hover{
    background-color:rgb(240, 47, 47);
  color: white;
  border: 1px solid rgb(240, 47, 47);
  text-decoration: none;
  border-radius: 5px;
  padding: 8px; 
}
.loc1{
    padding-left: 700px;
    padding-top: 550px;
}
</style>
<body>
    <div class="loc1">
    <a class="button" href="/">Get Started </a></div>
    </body>
    <%- include('partials/footer') -%>
```

header.ejs

```
<!DOCTYPE html>
<html>
<head>
    <title> VIT-AP Employee Database System</title>
    <link rel="stylesheet" type="text/css" href="../libs/
bootstrap.css">
```

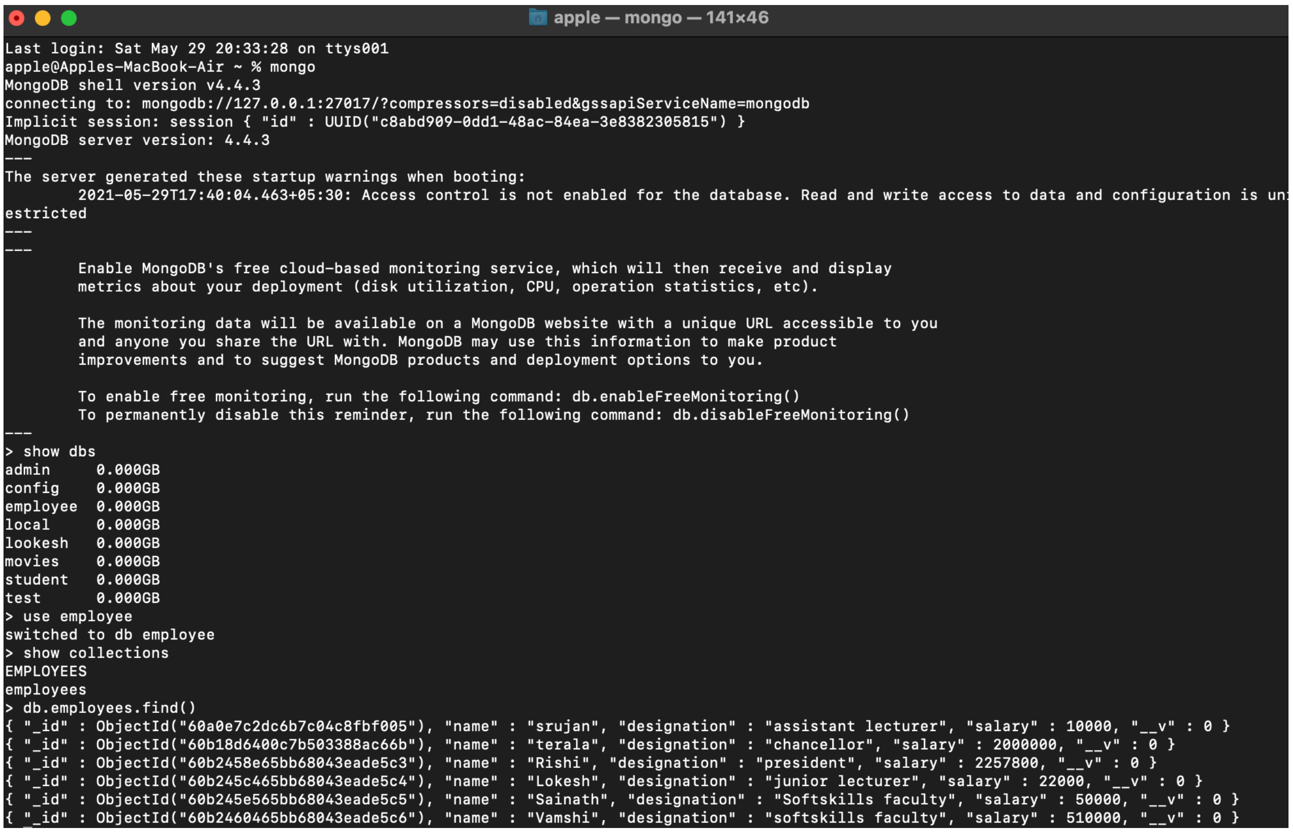
```html
    <link href="../libs/dataTables.bootstrap4.min.css"
rel="stylesheet">
    <link rel="stylesheet" type="text/css" href="../stylesheets/
design.css">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/
bootstrap-icons@1.5.0/font/bootstrap-icons.css">
    <script src="../libs/jquery.min.js"></script>
    <script src="../libs/jquery.dataTables.min.js"></script>
    <script src="../libs/dataTables.bootstrap4.min.js"></script>
</head>
<body>
    <nav class="py-1 navbar navbar-expand-lg navbar-light bg-
danger">
        <a class="navbar-brand text-white ml-4" href="/"><b>VIT-AP
EMPLOYEE DATABASE SYSTEM</b></a>
        <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarSupportedContent"
            aria-controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse justify-content-end"
id="navbarSupportedContent">
            <div class="mr-5">
                <ul class="navbar-nav mr-auto mr-0">
                    <li class="nav-item">
                        <a class="nav-link text-white" href="/
employee/home"><i class="bi bi-house"></i> </a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link text-white" href="/
employee/new"><i class="bi bi-person-plus"></i></b></a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link text-white" href="/
employee/search"><i class="bi bi-search"></i> </a>
                    </li>
                </ul>
    </nav>
```

Index.ejs



Name	Designation	Salary	Operations
Lokesh	junior lecturer	₹22000	Edit Delete
Rishi	president	₹2257800	Edit Delete
Sainath	Softskills faculty	₹50000	Edit Delete
srujan	assistant lecturer	₹10000	Edit Delete
terala	chancellor	₹2000000	Edit Delete
Vamshi	softskills faculty	₹510000	Edit Delete

Showing 1 to 6 of 6 entries

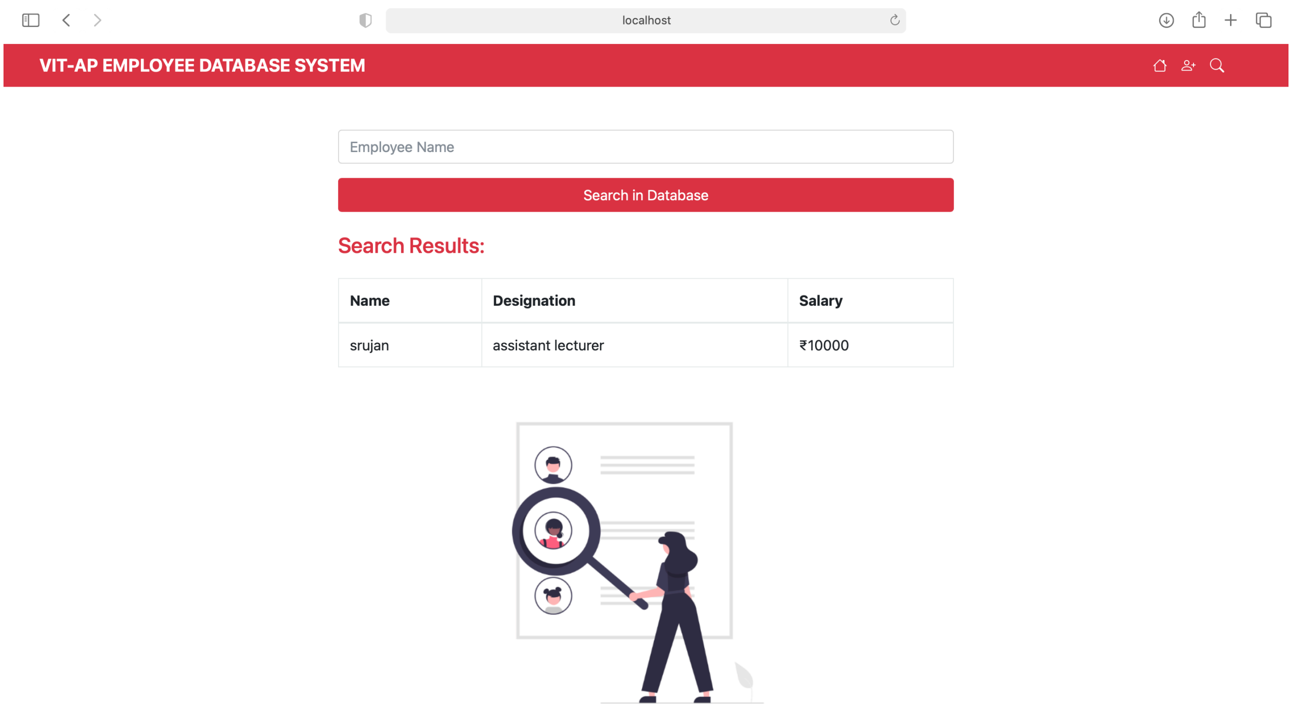


```
Last login: Sat May 29 20:33:28 on ttys001
apple@Apples-MacBook-Air ~ % mongo
MongoDB shell version v4.4.3
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("c8abd909-0dd1-48ac-84ea-3e8382305815") }
MongoDB server version: 4.4.3
---
The server generated these startup warnings when booting:
        2021-05-29T17:40:04.463+05:30: Access control is not enabled for the database. Read and write access to data and configuration is un
estricted
---
---
        Enable MongoDB's free cloud-based monitoring service, which will then receive and display
        metrics about your deployment (disk utilization, CPU, operation statistics, etc).

        The monitoring data will be available on a MongoDB website with a unique URL accessible to you
        and anyone you share the URL with. MongoDB may use this information to make product
        improvements and to suggest MongoDB products and deployment options to you.

        To enable free monitoring, run the following command: db.enableFreeMonitoring()
        To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
---
> show dbs
admin     0.000GB
config    0.000GB
employee  0.000GB
local     0.000GB
lookesh   0.000GB
movies    0.000GB
student   0.000GB
test      0.000GB
> use employee
switched to db employee
> show collections
EMPLOYEES
employees
> db.employees.find()
{ "_id" : ObjectId("60a0e7c2dc6b7c04c8fbf005"), "name" : "srujan", "designation" : "assistant lecturer", "salary" : 10000, "__v" : 0 }
{ "_id" : ObjectId("60b18d6400c7b503388ac66b"), "name" : "terala", "designation" : "chancellor", "salary" : 2000000, "__v" : 0 }
{ "_id" : ObjectId("60b2458e65bb68043eade5c3"), "name" : "Rishi", "designation" : "president", "salary" : 2257800, "__v" : 0 }
{ "_id" : ObjectId("60b245c465bb68043eade5c4"), "name" : "Lokesh", "designation" : "junior lecturer", "salary" : 22000, "__v" : 0 }
{ "_id" : ObjectId("60b245e565bb68043eade5c5"), "name" : "Sainath", "designation" : "Softskills faculty", "salary" : 50000, "__v" : 0 }
{ "_id" : ObjectId("60b2460465bb68043eade5c6"), "name" : "Vamshi", "designation" : "softskills faculty", "salary" : 510000, "__v" : 0 }
```

Code

```
<%- include('partials/header') -%>
<div class="container mt-5">
    <%- include('partials/messages') -%>
    <table class="table table-bordered" id="dataTable"
width="100%" cellspacing="0">
        <thead>
            <tr>
                <th>Name</th>
                <th>Designation</th>
                <th>Salary</th>
                <th>Operations</th>
            </tr>
        </thead>
        <tbody>
            <% employees.forEach((employee)=> { %>
                    <tr>
                        <td><%= employee.name %></td>
                        <td><%= employee.designation %></td>
                        <td>₹<%= employee.salary %></td>
                        <td>
                            <a class="btn btn-primary" href="/
edit/<%= employee.id %>">Edit</a>
                            <form style="display: inline;"
action="/delete/<%= employee.id %>?_method=DELETE" method="POST">
                                <input type="hidden"
name="_method" value="DELETE">
                                <button style="cursor: pointer;"
type="submit" class="btn btn-danger ml-2">Delete</button>
                            </form>
                        </td>
                    </tr>
               <% }) %>

            </tbody>
        </table>
    </div>
<%- include('partials/footer') -%>
```

Search.ejs



VIT-AP EMPLOYEE DATABASE SYSTEM

Employee Name

Search in Database

Search Results:

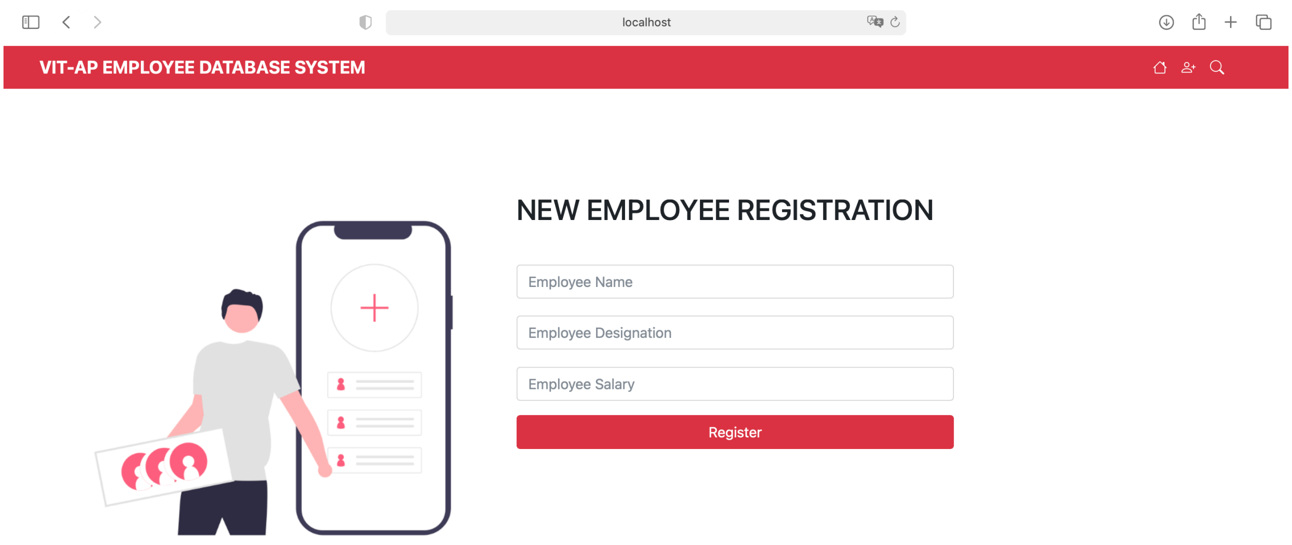
Name	Designation	Salary
srujan	assistant lecturer	₹10000

Code

```
<%- include('partials/header') -%>
<style>
    body{
 background-image: url("https://static.wixstatic.com/media/
ee5f36_33dc59b4441c4aa9b2735860043164b6~mv2.png/v1/fill/
w_2274,h_1172,fp_0.50_0.50/
ee5f36_33dc59b4441c4aa9b2735860043164b6~mv2.png");
    background-repeat: no-repeat;
    background-size: cover;
    }
</style>
    <div class="container mt-5 w-50">
        <form action="/employee" method="GET">
            <input type="text" name="name" class="form-control"
placeholder="Employee Name">
            <button type="submit" class="btn btn-danger btn-block
mt-3">Search in Database</button>
```

```
        </form>

        <% if(employee) { %>
            <h4 class="text-danger my-4">Search Results: </h4>
            <table class="table table-bordered" width="100%"
cellspacing="0">
                <thead>
                    <tr>
                        <th>Name</th>
                        <th>Designation</th>
                        <th>Salary</th>
                    </tr>
                </thead>
                <tbody>
                    <tr>
                        <td>
                            <%= employee.name %>
                        </td>
                        <td>
                            <%= employee.designation %>
                        </td>
                        <td>₹<%= employee.salary %>
                        </td>
                    </tr>
                </tbody>
            </table>
            <% } else if(employee !='' ){ %>
                <div class="alert alert-danger mt-4">Employee does
not exist with this name.</div>
                <% } %>
    </div>
    <%- include('partials/footer') -%>
```

new.ejs


localhost

VIT-AP EMPLOYEE DATABASE SYSTEM

NEW EMPLOYEE REGISTRATION

Employee Name

Employee Designation

Employee Salary

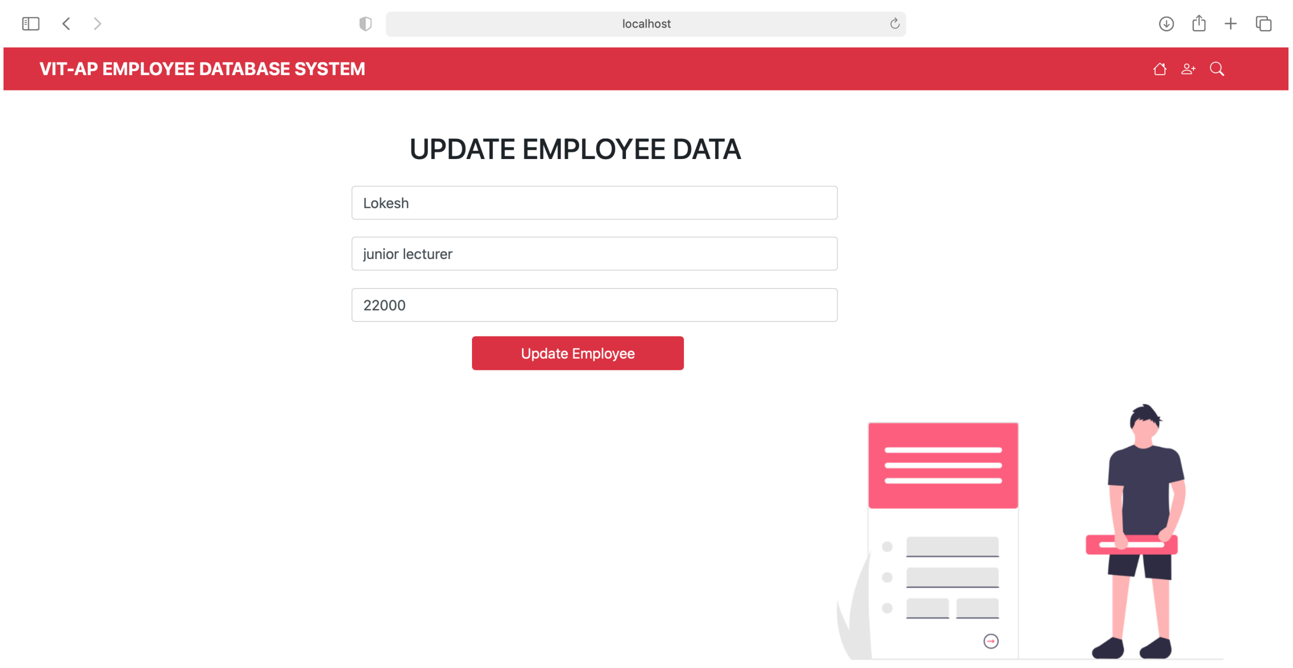
Register

Code

```
<%- include('partials/header') -%>
<style>
    body{
 background-image: url("https://static.wixstatic.com/media/
ee5f36_9393e8a269194b6fb8c5a963e5cdbf1d~mv2.png/v1/fill/
w_2274,h_1172,fp_0.50_0.50/
ee5f36_9393e8a269194b6fb8c5a963e5cdbf1d~mv2.png");
    background-repeat: no-repeat;
    background-size: cover;
    }
</style>
<div class="container mt-5 w-50">
    <h2 class="mb-4" style="padding-left: 200px;padding-top: 70px;
padding-bottom: 20px; "> NEW EMPLOYEE REGISTRATION</h2>
<div class="col-md-13" style="padding-left: 200px;">
    <form action="/employee/new" method="POST">
        <input type="text" name="name" class="form-control"
placeholder="Employee Name">
```

```
        <input type="text" name="designation" class="form-control"
placeholder="Employee Designation">
        <input type="text" name="salary" class="form-control"
placeholder="Employee Salary">
        <button type="submit" class="btn btn-danger btn-block
mt-3">Register</button>
    </div>
    </form>
</div>
<%- include('partials/footer') -%>
```

Edit.ejs



localhost

VIT-AP EMPLOYEE DATABASE SYSTEM

UPDATE EMPLOYEE DATA

Lokesh

junior lecturer

22000

Update Employee

Code

```
<%- include('partials/header') -%>
<style>
    body{
 background-image: url("https://static.wixstatic.com/media/
ee5f36_b694f6a93588436b83d2db70a28f5c6b~mv2.png/v1/fill/
w_2256,h_1125,fp_0.50_0.50/
ee5f36_b694f6a93588436b83d2db70a28f5c6b~mv2.png");
```

```
    background-repeat: no-repeat;
    background-size: cover;
    }
</style>
<body>
    <div class="container mt-5 w-50">
        <h2 class="mb-4" style="padding-left: 80px;">UPDATE 
EMPLOYEE DATA</h2>
        <div class="col-md-10">
        <form action="/edit/<%= employee.id %>?_method=PUT" 
method="POST">
            <input type="hidden" name="_method" value="PUT" >
            <input type="text" value="<%= employee.name %>" 
name="name" class="form-control"
                placeholder="Employee Name">
            <input type="text" value="<%= employee.designation %>" 
name="designation" class="form-control"
                placeholder="Employee Designation">
            <input type="text" value="<%= employee.salary %>" 
name="salary" class="form-control"
                placeholder="Employee Salary"></div>
    <div class="col-md-7" style="padding-left: 150px;">
            <button type="submit" class="btn btn-danger btn-block 
mt-3">Update Employee</button>
    </div>
        </form>

    </div>
    </body>
    <%- include('partials/footer') -%>
```

References

- https://www.udemy.com/course/nodejs-express-mongodb-bootcamp/learn/lecture/15080916?start=15#overview
- https://docs.mongodb.com/manual/tutorial/query-embedded-documents/
- https://www.w3schools.com/nodejs/nodejs_mongodb.asp
- https://www.guru99.com/node-js-mongodb.html
- https://www.coursera.org/learn/server-side-nodejs