



GOVERNMENT OF INDIA

MINISTRY OF SKILL DEVELOPMENT & ENTREPRENEURSHIP

DIRECTORATE GENERAL OF TRAINING

NATIONAL SKILL TRAINING INSTITUTE

NSTI(W) Trivandrum

PROJECT DOCUMENT

This is to certify that following trainee have completed their project titled

“Library Membership Management System”

For IBM Program – IT, Networking and Cloud (Technical Diploma)

ADIT-008 Bhavya B

Under the supervision of

Mr. Poovaragavan Velumani (Master Trainer, Edunet Foundation)

Abstract

Library Membership Management is a project that helps to keep track of library members based on their details. This project is designed to help the admin carry out operations in a smooth and error free manner. The “Library Membership Management” project has been developed to override the complications with the existing manual system. It eliminates the errors that might occur when the records are maintained manually and allows you to manage your workforce anytime.

CONTENTS

ABSTRACT

1. INTRODUCTION

2. SYSTEM REQUIREMENTS

2.1 SOFTWARE SPECIFICATION

2.2 HARDWARE SPECIFICATION

3. PROPOSED SYSTEM

4. APPENDICES

4.1 SOURCE CODE

4.2 SCREENSHOTS

5. CONCLUSION

6. REFERENCE

1. INTRODUCTION

This system provides admin the facility to make all the data saving process easy. Update, Add, Delete etc. This system provides to add information or members image from file. Systems avoid mistakes & error in maintaining the details of members. The main aim of our system is to provide a paper-less working of the library to 90 %. It also aims at providing low cost, reliable, automation of exiting system. All the details members are secured stored in database.

2. SYSTEM REQUIREMENTS

2.1 SOFTWARE REQUIREMENTS

Operating System : Windows 7

Front End : JavaScript

Back End : Nodejs, Mysql

Code Editor : Visual Code

Server : Web Browser:

2.2 HARDWARE REQUIREMENTS

RAM : 1 GB or above

Processor : 1 GHz or more

Hard Drive : 32 GB or above

Network Connectivity : LAN or Wi-Fi

3. PROPOSED SYSTEM

Library membership management system offers membership management for any library looking to more efficiently manage member records. Programming languages include JavaScript with Nodejs and MYSQL are used for developing the website. In this Library administrator can add, update, delete, view the member details.

This web application facilitates administrator can just enter the name and concerned details on the joining to the library using a computer. It will be stored in the database. Even if they want to see the members of the library, they can view it using this system. Editing processes are also easy. They can update all the details they have entered. Also the name and details of the members who left the club can also be removed easily using this system.

4. APPENDICES

4.1 SOURCE CODE

1. index.js

```
module.exports = {
  getHomePage: (req, res) => {
    let query = "SELECT * FROM `members` ORDER BY id ASC"; // query database
    to get all the members

    // execute query
    db.query(query, (err, result) => {
      if (err) {
        res.redirect('/');
      }
      res.render('index.ejs', {
        title: "Welcome to chapter library | View members"
        ,members: result
      });
    });
  },
};
```

2. member.js

```
module.exports = {
  addmemberPage: (req, res) => {
    res.render('add-member.ejs', {
      title: "Welcome to chapter library| Add a new member"
      ,message: ''
    });
  },
  addmember: (req, res) => {
    if (!req.files) {
      return res.status(400).send("No files were uploaded.");
    }

    let message = '';
    let first_name = req.body.first_name;
    let last_name = req.body.last_name;
```

```

    let phone = req.body.phone;
    let username = req.body.username;
    let uploadedFile = req.files.image;
let image_name = uploadedFile.name;
    let fileExtension = uploadedFile.mimetype.split('/')[1];
    image_name = username + '.' + fileExtension;

    let usernameQuery = "SELECT * FROM `members` WHERE username ="
+ username + ""

    db.query(usernameQuery, (err, result) => {
        if (err) {
            return res.status(500).send(err);
        }
        if (result.length > 0) {
            message = 'Username already exists';
            res.render('add-member.ejs', {
                message,
                title: "Welcome to chapter library| Add a new memr"
            });
        } else {
            // check the filetype before uploading it
            if (uploadedFile.mimetype === 'image/png' || uploadedFile.mimetype === 'image/jpeg' || uploadedFile.mimetype === 'image/gif')
            {
                // upload the file to the /public/assets/img directory
                uploadedFile.mv(`public/assets/img/${image_name}`,
                (err) => {
                    if (err) {
                        return res.status(500).send(err);
                    }
                    // send the member's details to the database
                    let query = "INSERT INTO `members` (first_name, last_name, phone, image, username) VALUES ('" +
                        first_name + "', '" + last_name + "', '" +
                        phone + "', '" + image_name + "', '" + username + "')";
                    db.query(query, (err, result) => {
                        if (err) {
                            return res.status(500).send(err);
                        }
                        res.redirect('/');
                    });
                });
            } else {

```



```

        message = "Invalid File format. Only 'gif', 'jpe'
and 'png' images are allowed.";
        res.render('add-member.ejs', {
            message,
            title: "Welcome to chapter library | Add a ne
w member"

        });
    }
}
});
},
editmemberPage: (req, res) => {
    let memberId = req.params.id;
    let query = "SELECT * FROM `members` WHERE id = '" + memberI
+ "' ";
    db.query(query, (err, result) => {
        if (err) {
            return res.status(500).send(err);
        }
        res.render('edit-member.ejs', {
            title: "Edit member"
            ,member: result[0]
            ,message: ''
        });
    });
},
editmember: (req, res) => {
    let memberId = req.params.id;
    let first_name = req.body.first_name;
    let last_name = req.body.last_name;
    let phone= req.body.phone;

    let query = "UPDATE `members` SET `first_name` = '" + first_n
ame + "', `last_name` = '" + last_name + "', `phone` = '" + phone+ "'
WHERE `members`.`id` = '" + memberId + "'";
    db.query(query, (err, result) => {
        if (err) {
            return res.status(500).send(err);
        }
        res.redirect('/');
    });
},
deletemember: (req, res) => {
    let memberId = req.params.id;

```

```

        let getImageQuery = 'SELECT image from `members` WHERE id = "'
        + memberId + '"';
        let deleteUserQuery = 'DELETE FROM members WHERE id = "' + me
        mberId + '"';

        db.query(getImageQuery, (err, result) => {
            if (err) {
                return res.status(500).send(err);
            }

            let image = result[0].image;

            fs.unlink(`public/assets/img/${image}`, (err) => {
                if (err) {
                    return res.status(500).send(err);
                }
                db.query(deleteUserQuery, (err, result) => {
                    if (err) {
                        return res.status(500).send(err);
                    }
                    res.redirect('/');
                });
            });
        });
    });
}
};

```

3. header.ejs

```

<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
        content="width=device-width, user-scalable=no, initial-
scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title><%= title %></title>
</head>
<style>
    .table-wrapper {
        margin-top: 50px;
        margin-left: 100px;
    }

```

```

    }
th{
    padding: 25px 30px;
    color: white;
    text-align: center;
    text-decoration: none;
    background-color: #333;
}
td{
    padding: 25px 30px;
}
.member-img {
    width: 40px;
    height: 40px;
}

.add-member-form {
    margin-top: 50px;
    margin-left: 400px;
}
nav{
    color:rgb(72, 72, 218);
    margin-left: 20px;
}
.button{
    text-align: center;
    margin-left: 50px;
    background-color: rgb(7, 172, 7);
}
.action{
    text-align: center;
    margin-left: 50px;
    background-color: rgb(247, 47, 47)
}
a{
    text-decoration: none;
}
#insert{
    margin-left: 1050px;
}
.form-control{
    margin-top: 25px;
    margin-left: 20px;
}

```

```

.col-md-12{
    margin-top: 20px;
    margin-left: 20px;
}
#add{
    margin-left: 250px;
}
#user{
    margin-left: 25px;
    margin-top: 25px;
}
#ph{
    margin-left: 50px;
    margin-top: 25px;
}
#update{
    margin-top: 25px;
    margin-left: 250px;
}
</style>
<body>
<div class="page-wrapper">
    <nav class="navbar navbar-light bg-light">
        <span class="navbar-brand mb-0 h1" ><h1>chapter library</h1></span>
        <a id="insert" href="/add" title="Add a New member">Add a member</a>

    </nav>

```

4. add-member.ejs

```

<%- include ('partials/header.ejs'); -%>
<div class="container">
    <% if (message != '') { %>
        <p class="text-center text-danger"><%= message %></p>
    <% } %>
    <form class="add-member-form" action="" method="post" enctype="multipart/form-data">
        <div class="form-row">
            <div class="form-group col-md-4">
                <input type="text" class="form-control" name="first_name" id="first-name" placeholder="First Name" required>
            </div>

```

```

        <div class="form-group col-md-4">
            <input type="text" class="form-
control" name="last_name" id="last-name" placeholder="Last Name" required>
        </div>
        <div class="form-group col-md-4">
            <input type="text" class="form-
control" name="username" id="username" placeholder="Username" required>
        </div>
    </div>
    <div class="form-row">
        <div class="form-group col-md-6">
            <input type="number" class="form-
control" name="phone" id="phone" placeholder="phone" required>
        </div>

        <div class="col-md-12">
            <label for="member-img"><b>member Image</b></label><br>
            <input type="file" name="image" id="member-
img" class="" required>
        </div>
    </div>
    <div id="add">
        <button type="submit" id="btn btn-primary float-
right">Add Member</button>
    </div>
</form>
</div>
</div>
</body>
</html>

```

5. edit-member.ejs

```

<%- include ('partials/header.ejs'); -%>
<div class="container">
    <% if (message) { %>
        <p class="text-center text-danger"><%= message %></p>
    <% } %>

    <% if (member) { %>
        <form class="add-member-
form" action="" method="post" enctype="multipart/form-data">
            <div class="form-row">
                <div class="form-group col-md-4">

```

```

        <label for="first-name">First Name</label>
        <input type="text" class="form-
control" name="first_name" id="first-
name" value="<%= member.first_name %>" required>
    </div>
    <div class="form-group col-md-4">
        <label for="last-name">Last Name</label>
        <input type="text" class="form-
control" name="last_name" id="last-
name" value="<%= member.last_name %>" required>
    </div>
    <div class="form-group col-md-4">
        <label for="username">Username</label>
        <input type="text" id="user" name="username" id="username" va
lue="<%= member.username %>" required disabled title="Username cannot be edited."
    >
    </div>
</div>
</div>
<div class="form-row">
    <div class="form-group col-md-6">
        <label for="phone">phone</label>
        <input type="number" id="ph" name="phone" id="phone" placehol
der="phone" value="<%= member.phone %>" required>
    </div>
</div>
<div id=update>
    <button type="submit" class="btn btn-success float-
right">Update member</button>
</div>
</form>
<% } else { %>
    <p class="text-
center">Player Not Found. Go <a href="/add">here</a> to add members.</p>
    <% } %>
</div>
</div>
</body>
</html>

```

6.index.ejs

```

<%- include ('partials/header.ejs'); -%>

```

```

<div class="container">
  <% if (message) { %>
    <p class="text-center text-danger"><%= message %></p>
  <% } %>

  <% if (member) { %>
    <form class="add-member-form" action="" method="post" enctype="multipart/form-data">
      <div class="form-row">
        <div class="form-group col-md-4">
          <label for="first-name">First Name</label>
          <input type="text" class="form-control" name="first_name" id="first-name" value="<%= member.first_name %>" required>
        </div>
        <div class="form-group col-md-4">
          <label for="last-name">Last Name</label>
          <input type="text" class="form-control" name="last_name" id="last-name" value="<%= member.last_name %>" required>
        </div>
        <div class="form-group col-md-4">
          <label for="username">Username</label>
          <input type="text" id="user" name="username" id="username" value="<%= member.username %>" required disabled title="Username cannot be edited."
        >
        </div>
      </div>
      <div class="form-row">
        <div class="form-group col-md-6">
          <label for="phone">phone</label>
          <input type="number" id="ph" name="phone" id="phone" placeholder="phone" value="<%= member.phone %>" required>
        </div>
      </div>
      <div id="update">
        <button type="submit" class="btn btn-success float-right">Update member</button>
      </div>
    </form>
  <% } else { %>
    <p class="text-center">Player Not Found. Go <a href="/add">here</a> to add members.</p>
  <% } %>
</div>

```

```
</div>
</body>
</html>
```

7. app.js

```
const express = require('express');
const fileUpload = require('express-fileupload');
const bodyParser = require('body-parser');
const mysql = require('mysql');
const path = require('path');
const app = express();

const {getHomePage} = require('./routes/index');
const {addmemberPage, addmember, deletemember, editmember, editmemberPage} = require('./routes/member');
const port = 5000;

// create connection to database
// the mysql.createConnection function takes in a configuration object which contains host, user, password and the database name.
const db = mysql.createConnection ({
  host: 'localhost',
  user: 'root',
  password: '',
  database: 'chapter'
});

// connect to database
db.connect((err) => {
  if (err) {
    throw err;
  }
  console.log('Connected to database');
});
global.db = db;

// configure middleware
app.set('port', process.env.port || port); // set express to use this port
app.set('views', __dirname + '/views'); // set express to look in this folder to render our view
app.set('view engine', 'ejs'); // configure template engine
app.use(bodyParser.urlencoded({ extended: false }));
app.use(bodyParser.json()); // parse form data client
```



```

app.use(express.static(path.join(__dirname, 'public'))); // configure express to
use public folder
app.use(fileUpload()); // configure fileupload

// routes for the app

app.get('/', getHomePage);
app.get('/add', addmemberPage);
app.get('/edit/:id', editmemberPage);
app.get('/delete/:id', deletemember);
app.post('/add', addmember);
app.post('/edit/:id', editmember);

// set the app to listen on the port
app.listen(port, () => {
  console.log(`Server running on port: ${port}`);
});

```

8. package.json

```

{
  "name": "em1",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "body-parser": "^1.19.0",
    "ejs": "^3.1.6",
    "express": "^4.17.1",
    "express-fileupload": "^1.2.1",
    "mysql": "^2.18.1",
    "req-flash": "0.0.3"
  },
  "devDependencies": {},
  "description": ""
}

```










4.2 SCREENSHOTS

em1 docs - bhavyabhasu x CONCLUSION FOR MEM x Project proposal of Libr x Welcome to chapter libr x gopika full name - Goog x

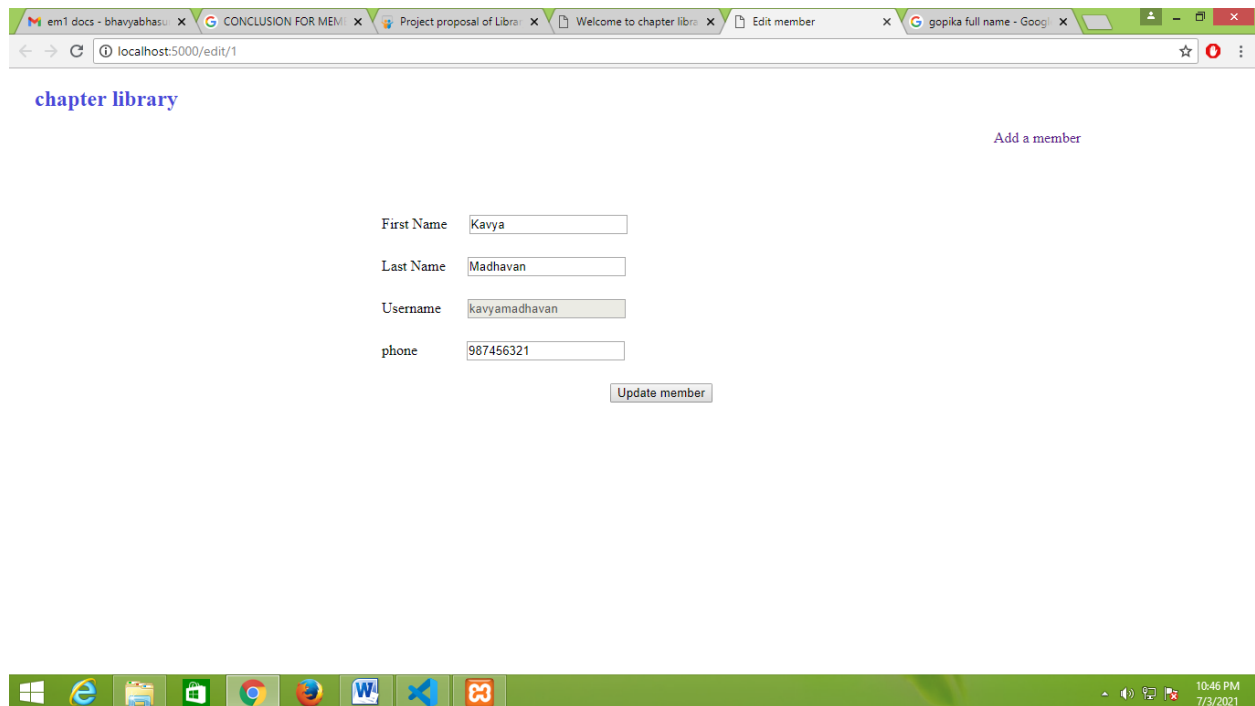
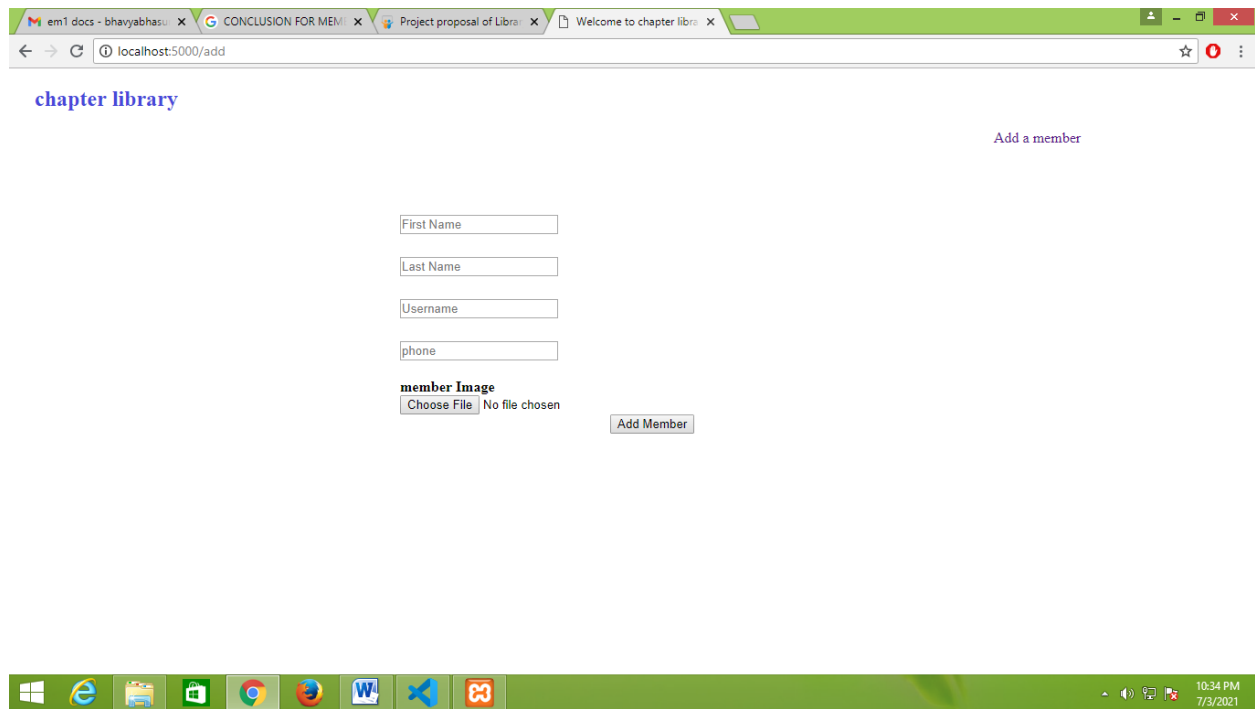
localhost:5000

chapter library

Add a member

ID	Image	First Name	Last Name	phone	Username	Action
1		Kavya	Madhavan	987456321	@kavyamadhavan	 
2		Navya	Nair	123456789	@navyanair	 
3		Gopika	Krishnan	1597536842	@gopikakrishnan	 

Windows taskbar: 10:45 PM 7/3/2021



5. CONCLUSION

The library membership management system needs to be computerized to reduce human errors and increase the efficiency. The proposed library membership management system will be computerized management system developed to maintain all the daily work of library. Library membership management system are designed to store all the information about members. The main focus of this project is to lessen human effort and encourage efficient record keeping.

6. REFFERNCE

1. <https://www.geeksforgeeks.org/>
2. <https://www.w3schools.com/>