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

"Arts Club Management System"

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

Submitted by

Vinaya Vijayan

ADIT/TVM/19/017

ADIT(2019-2021)

Under the supervision of

Mr. Poovaragavan Velumani (Master Trainer, Edunet Foundation)

ABSTRACT

Arts Club Members Management System is a software that is used for managing the details of the pupils who are interested in participating and joining as the members of the Arts Club of any schools or institutions. It is necessary to keep track of the pupils who joined the club. In that case this system is very helpful. This is an online management system of the members of the Arts Club in different institutions.

CONTENTS

INTRODUCTION

1.1 ABOUT PROJECT

This is a computerised system for managing the details of the Arts Club members. The name and the details of the pupils joining the Arts Club are stored and can be managed using this system. Thus it makes the job of the administrator easy. Managing the details of the club members like who joined, who left, who changed the position and editing the mistakes or such things are really difficult in manual managing. So this data management system makes the job easy and user friendly. This software can be used in schools, colleges or any other institutions.

1.2 USERS

The users of this project are the administrators of Arts Club of any schools, colleges or other institutions.

2. SYSTEM ANALYSIS

2.1 SOFTWARE REQUIREMENTS

Operating System : Windows 7 or above

Text Editor : Visual Studio Code

Front End : HTML, CSS, JavaScript, Bootstrap

Back End : Node.js, MySQL

Node.js Packages : Express, MySQL, Morgan

Browser : Google Chrome/Microsoft Edge/Mozilla Firefox

2.2 HARDWARE REQUIREMENTS

- A working PC
- RAM 4GB or higher
- Processor 1GHz or more
- Hard drive 32 GB or above
- Stable network connectivity

3. PROPOSED SYSTEM

Usually schools, colleges or institutions record the information of various club members manually. It is a difficult process because it may have many unbiased errors due to manual entry. Even if the mistake is made it is difficult to rectify because they have to go through all the records to find it. Editing is also difficult in manual paper works. If someone wished to leave the club, then it is also difficult to erase the name from the manual entry. IF some pupil wanted to change the activity then also it will be really difficult to change the information and details in the manual paper entry by a person or administrator.

This system is a user-friendly system to be used in those situations. Here, the administrator or pupil can just enter the name and concerned details on the joining to the club using a computer. It will be stored in the database. Even if they want to see the members of the club, 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 pupils who left the club can also be removed easily using this system.

Moreover this system can also be used for any other clubs other than Arts Club upon further modifications. So this will be really useful for schools and colleges.

4. APPENDICES

4.1 SOURCE CODE

app.js

```
var createError = require('http-errors');
var express = require('express');
var path = require('path');
var cookieParser = require('cookie-parser');
var logger = require('morgan');

var flash = require('express-flash');
var session = require('express-session');
var mysql = require('mysql');
var connection  = require('./lib/db');

var indexRouter = require('./routes/index');
var usersRouter = require('./routes/users');
var membersRouter = require('./routes/members');

var app = express();

// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'ejs');

app.use(logger('dev'));
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));

app.use(session({
  cookie: { maxAge: 60000 },
  store: new session.MemoryStore,
  saveUninitialized: true,
  resave: 'true',
  secret: 'secret'
}))

app.use(flash());
app.use('/', indexRouter);
app.use('/users', usersRouter);
app.use('/members', membersRouter);

// catch 404 and forward to error handler
app.use(function(req, res, next) {
  next(createError(404));
});

// error handler
app.use(function(err, req, res, next) {
  // set locals, only providing error in development
  res.locals.message = err.message;
  res.locals.error = req.app.get('env') === 'development' ? err :
{};

  // render the error page
  res.status(err.status || 500);
  res.render('error');
});

module.exports = app;
```

package.json

```
{
  "name": "nodejs-crud",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "start": "node ./bin/www"
  },
  "dependencies": {
    "cookie-parser": "~1.4.4",
    "debug": "~2.6.9",
    "ejs": "~2.6.1",
    "express": "~4.16.1",
    "express-flash": "^0.0.2",
    "express-session": "^1.17.2",
    "http-errors": "~1.6.3",
    "method-override": "^3.0.0",
    "morgan": "~1.9.1",
    "mysql": "^2.18.1"
  }
}
```

index.js

```
var express = require('express');
var router = express.Router();

/* GET home page. */
router.get('/', function(req, res, next) {
  res.render('index', { title: 'Express' });
});

module.exports = router;
```

Members.js

```
var express = require('express');
var router = express.Router();
var dbConn  = require('../lib/db');

// display books page
router.get('/', function(req, res, next) {

    dbConn.query('SELECT * FROM members ORDER BY id 
desc',function(err,rows)     {

        if(err) {
            req.flash('error', err);
            // render to views/books/index.ejs
            res.render('members',{data:''});
        } else {
            // render to views/books/index.ejs
            res.render('members',{data:rows});
        }
    });
});

// display add book page
router.get('/add', function(req, res, next) {
    // render to add.ejs
    res.render('members/add', {
        name: '',
        field: ''
    })
})

// add a new book
router.post('/add', function(req, res, next) {

    let name = req.body.name;
    let field = req.body.field;
    let errors = false;

    if(name.length === 0 || field.length === 0) {
        errors = true;

        // set flash message
        req.flash('error', "Please enter name and interested
field");
        // render to add.ejs with flash message
        res.render('members/add', {
            name: name,
            field: field
        })
    }

    // if no error
    if(!errors) {

        var form_data = {
            name: name,
            field: field
        }

        // insert query
        dbConn.query('INSERT INTO members SET ?', form_data,
function(err, result) {
            //if(err) throw err
            if (err) {
                req.flash('error', err)

                // render to add.ejs
                res.render('members/add', {
                    name: form_data.name,
                    field: form_data.field
                })
            } else {
                req.flash('success', ' Added successfully');
                res.redirect('/members');
            }
        })
    }
})

// display edit book page
router.get('/edit/(:id)', function(req, res, next) {

    let id = req.params.id;

    dbConn.query('SELECT * FROM members WHERE id = ' + id,
function(err, rows, fields) {
        if(err) throw err

        // if user not found
        if (rows.length <= 0) {
            req.flash('error', 'Member not found with id = ' +
id)
            res.redirect('/members')
        }
        // if book found
        else {
            // render to edit.ejs
            res.render('members/edit', {
                title: 'Edit',
                id: rows[0].id,
                name: rows[0].name,
                field: rows[0].field
            })
        }
    })
})

// update book data
router.post('/update/:id', function(req, res, next) {

    let id = req.params.id;
    let name = req.body.name;
    let field = req.body.field;
    let errors = false;

    if(name.length === 0 || field.length === 0) {
        errors = true;

        // set flash message
        req.flash('error', "Please enter name and interested
field");
        // render to add.ejs with flash message
        res.render('members/edit', {
            id: req.params.id,
            name: name,
            field: field
        })
    }

    // if no error
    if( !errors ) {

        var form_data = {
            name: name,
            field: field
        }
        // update query
        dbConn.query('UPDATE members SET ? WHERE id = ' + id,
form_data, function(err, result) {
            //if(err) throw err
            if (err) {
                // set flash message
                req.flash('error', err)
                // render to edit.ejs
                res.render('members/edit', {
                    id: req.params.id,
                    name: form_data.name,
                    field: form_data.field
                })
            } else {
                req.flash('success', 'Updated successfully !');
                res.redirect('/members');
            }
        })
    }
})

// delete member
router.get('/delete/(:id)', function(req, res, next) {
    let id = req.params.id;

    dbConn.query('DELETE FROM members WHERE id = ' + id,
function(err, result) {
        //if(err) throw err
        if (err) {
            // set flash message
            req.flash('error', err)
            // redirect to members page
            res.redirect('/members')
        } else {
            // set flash message
            req.flash('success', 'Deleted successfully!!')
            // redirect to members page
            res.redirect('/members')
        }
    })
})

module.exports = router;
```

Users.js

```
var express = require('express');
var router = express.Router();

/* GET users listing. */
router.get('/', function(req, res, next) {
  res.send('respond with a resource');
});

module.exports = router;
```

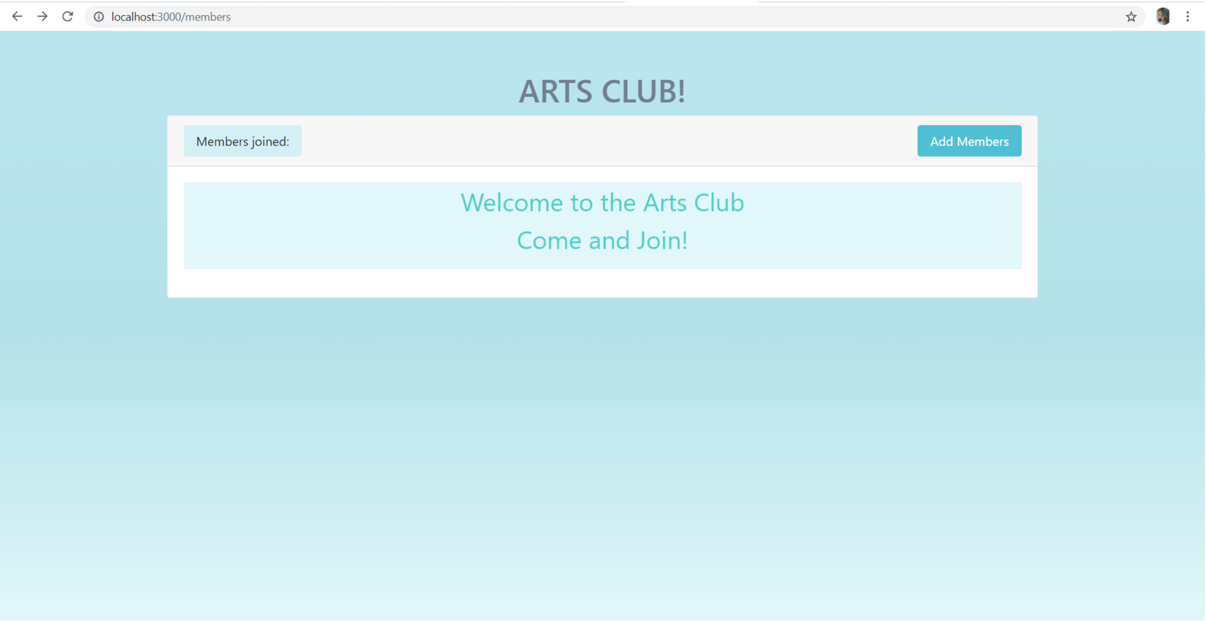
Db.js

```
var mysql = require('mysql');
var connection = mysql.createConnection({
    host:'localhost',
    user:'root',
    password:'',
    database:'artsclub'
});
connection.connect(function(error){
    if(!!error) {
        console.log(error);
    } else {
        console.log('Connected..!');
    }
});

module.exports = connection;
```

4.2 SCREENSHOTS

home



localhost:3000/members

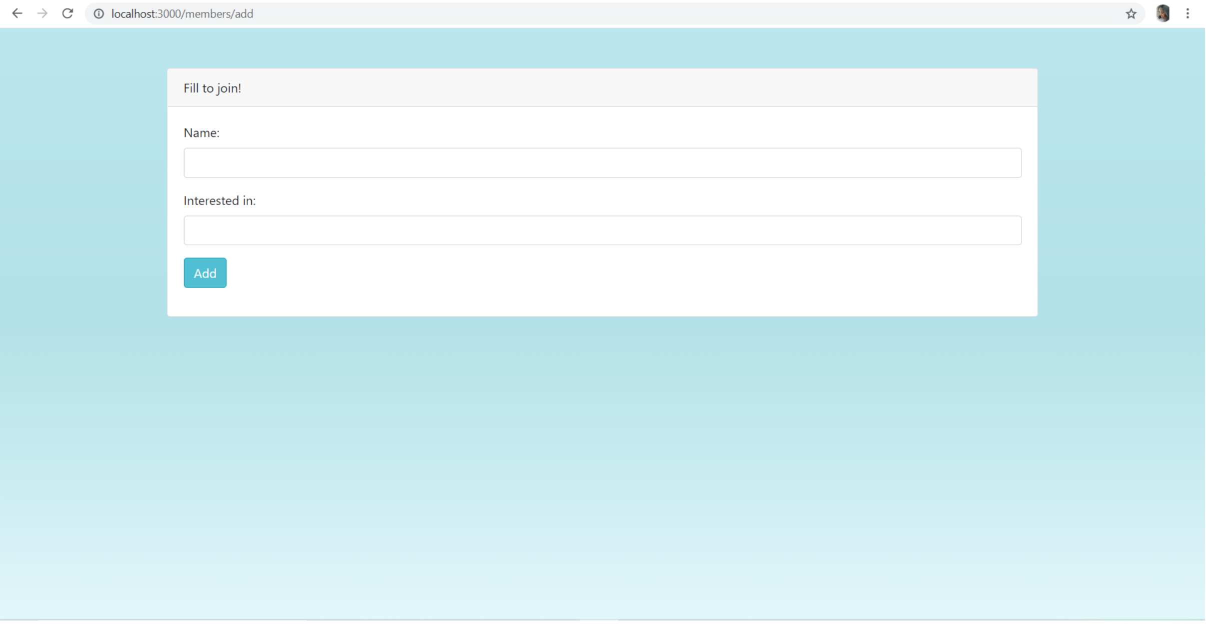
ARTS CLUB!

Members joined:

Welcome to the Arts Club

Come and Join!

add members



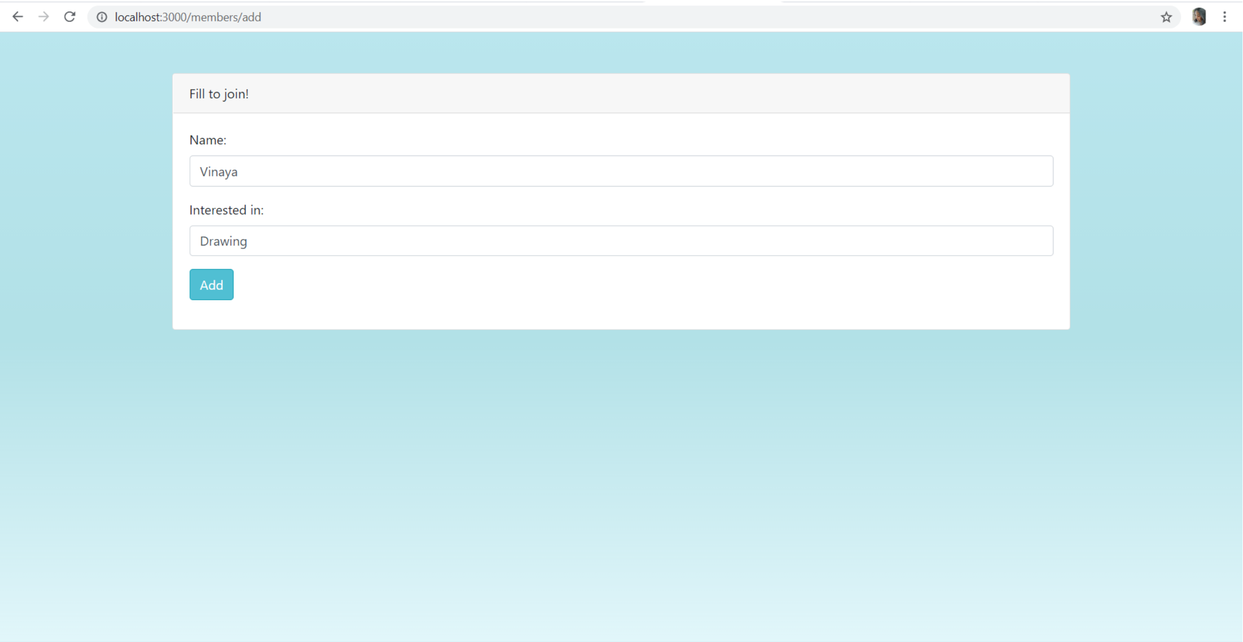
localhost:3000/members/add

Fill to join!

Name:

Interested in:

entering details



localhost:3000/members/add

Fill to join!

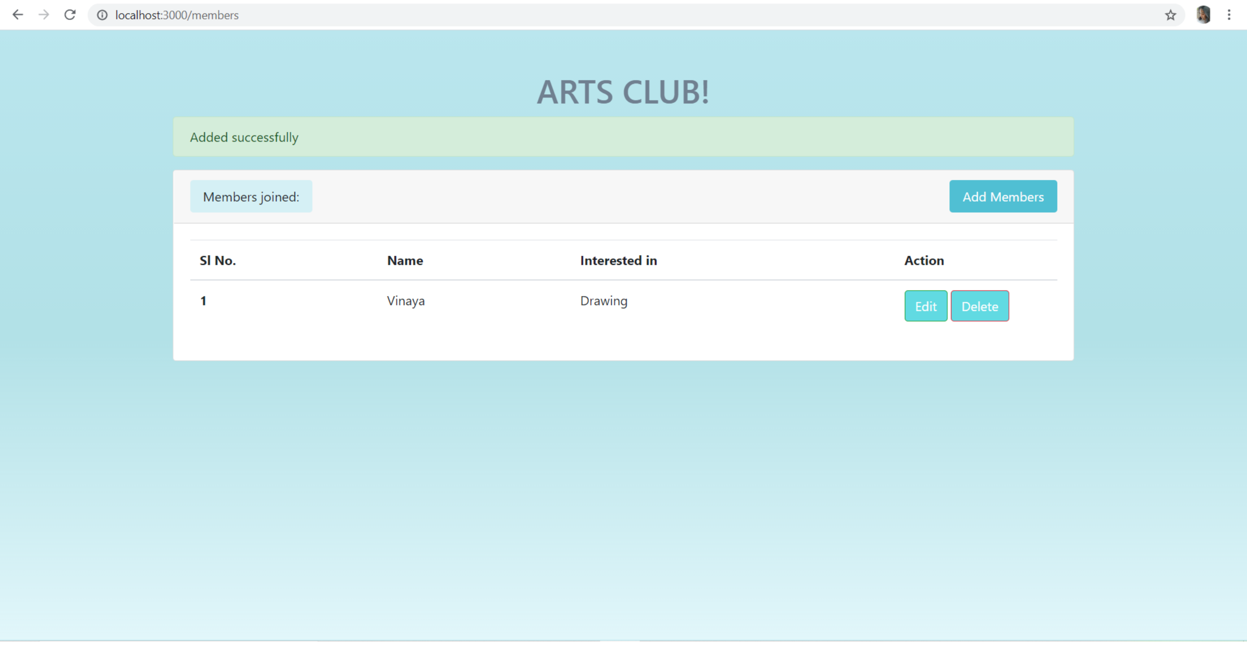
Name:

Vinaya

Interested in:

Drawing

details added



localhost:3000/members

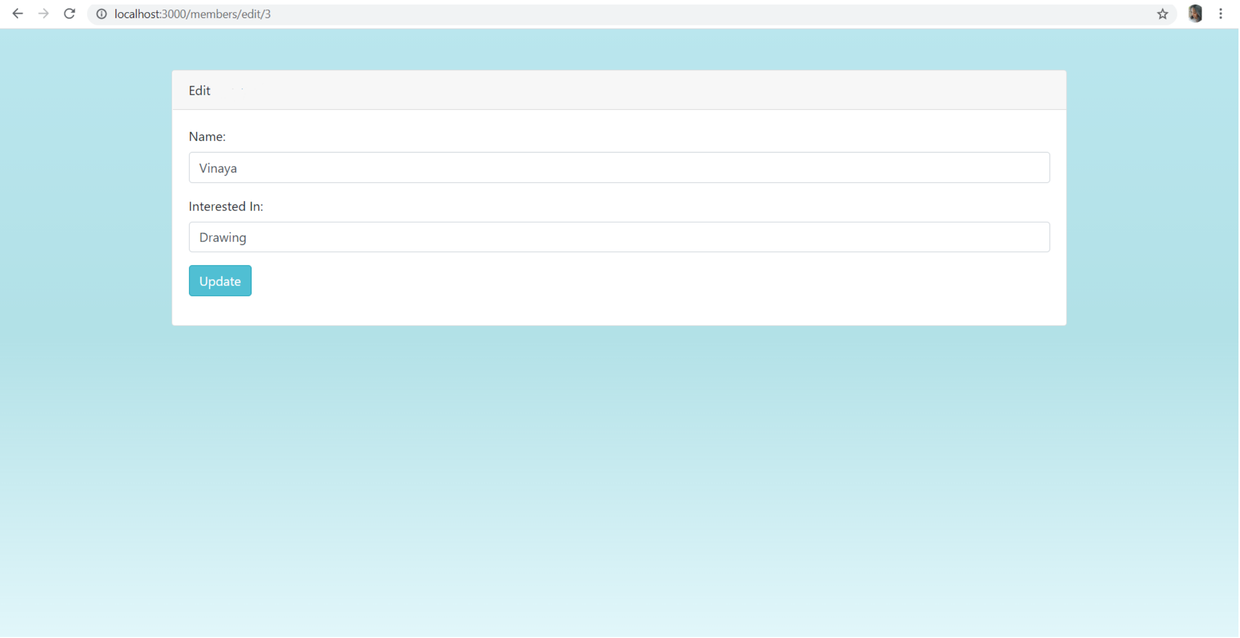
ARTS CLUB!

Added successfully

Members joined:

Sl No.	Name	Interested in	Action
1	Vinaya	Drawing	Edit Delete

edit details



localhost:3000/members/edit/3

Edit

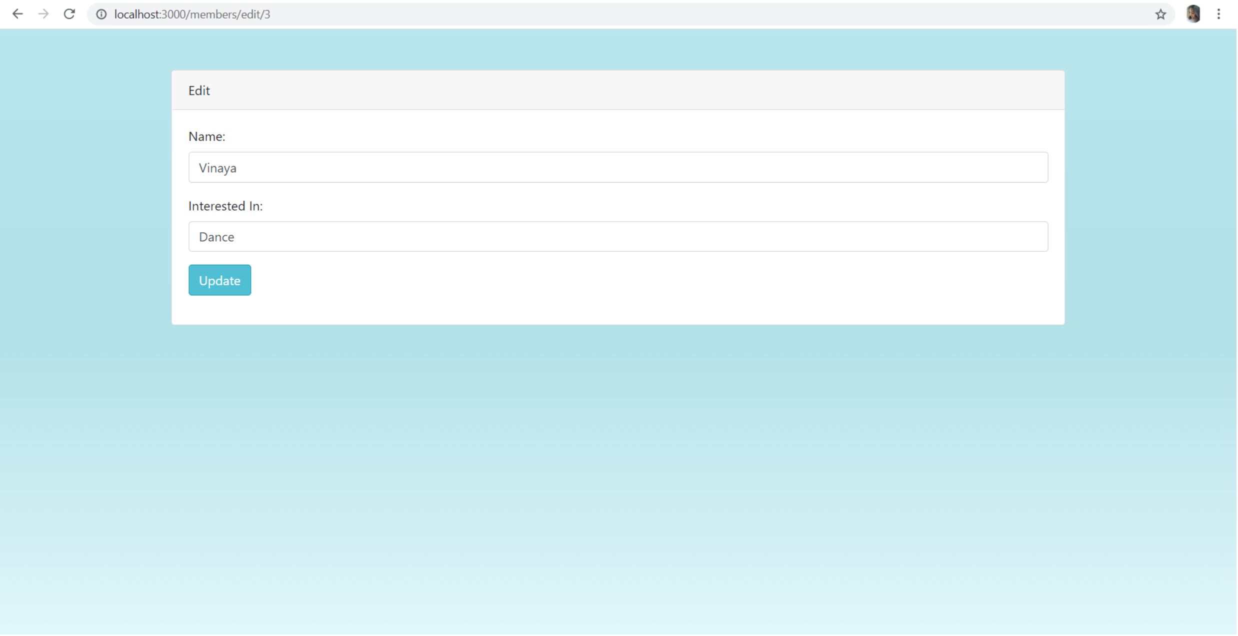
Name:

Vinaya

Interested In:

Drawing

entering new details



localhost:3000/members/edit/3

Edit

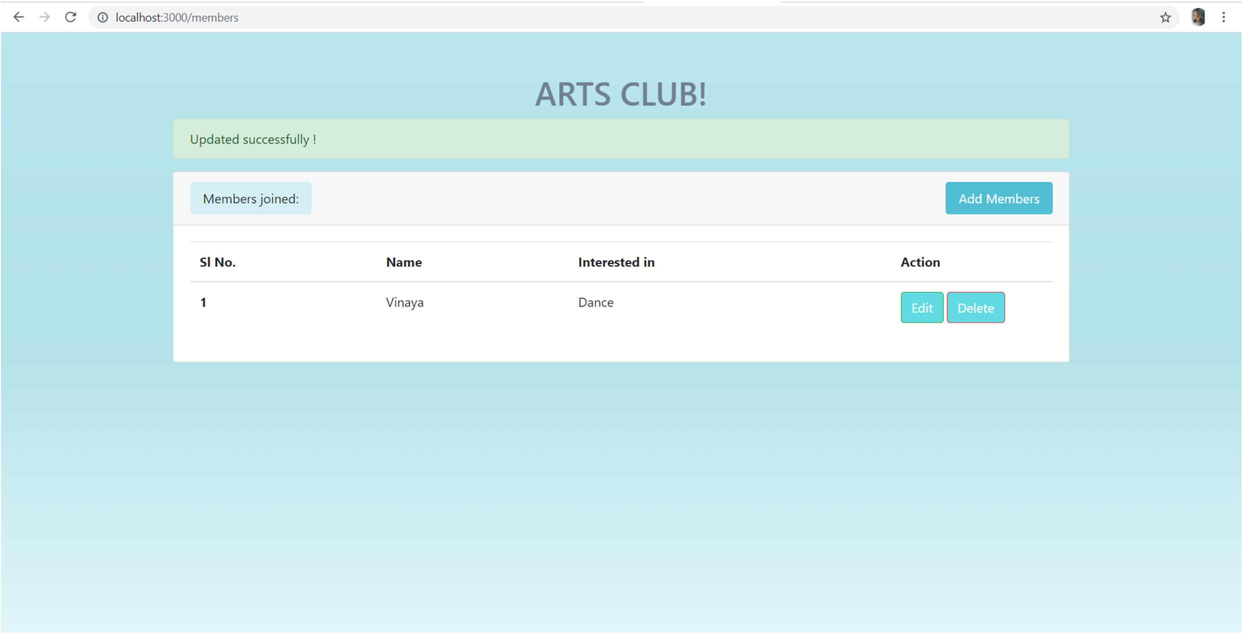
Name:

Vinaya

Interested In:

Dance

details updated



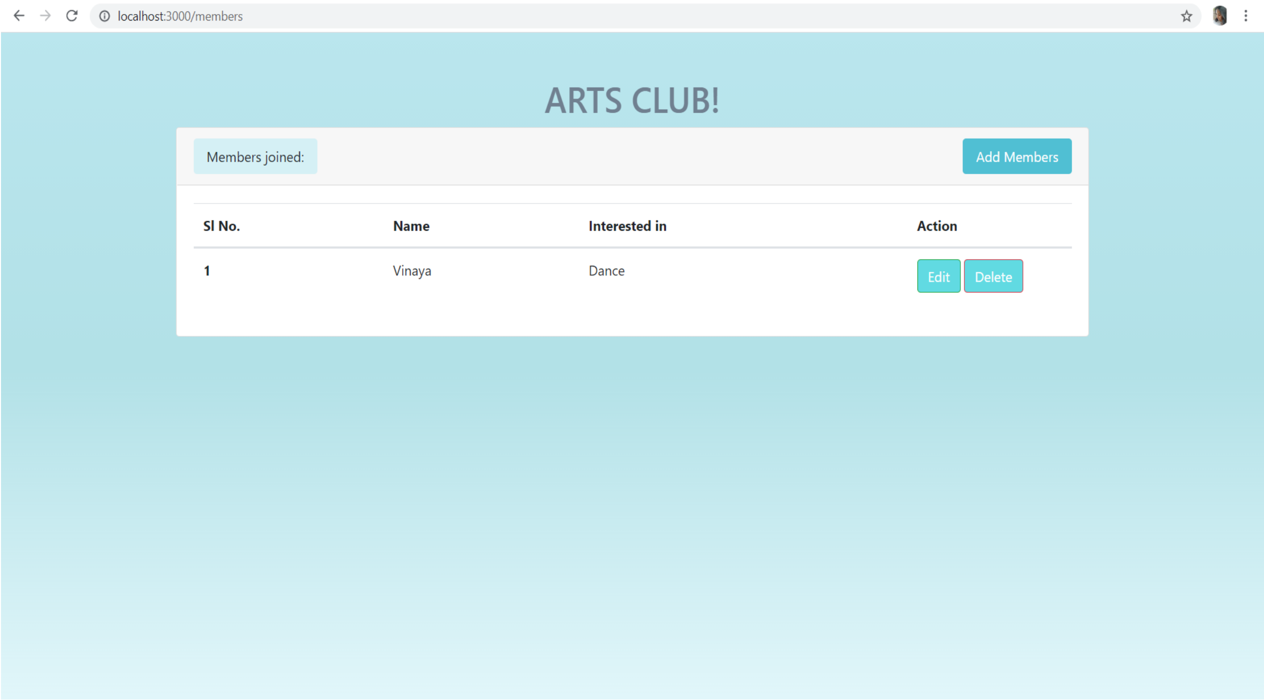
ARTS CLUB!

Updated successfully !

Members joined:

Sl No.	Name	Interested in	Action
1	Vinaya	Dance	Edit Delete

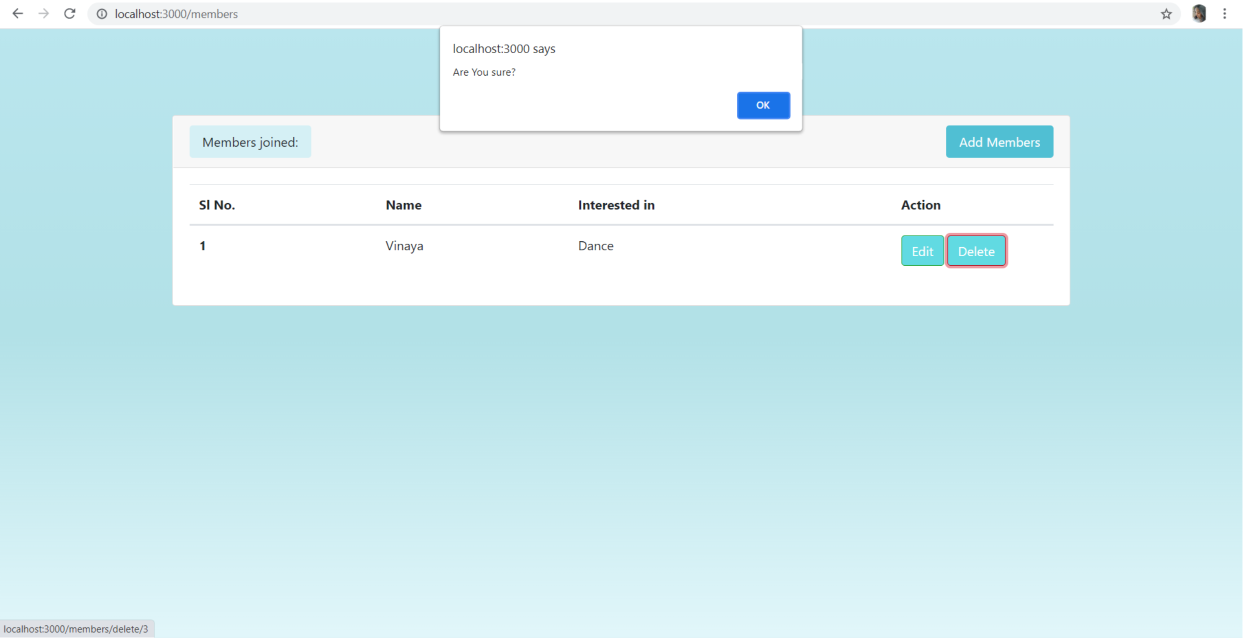
delete members



ARTS CLUB!

Members joined:

Sl No.	Name	Interested in	Action
1	Vinaya	Dance	Edit Delete

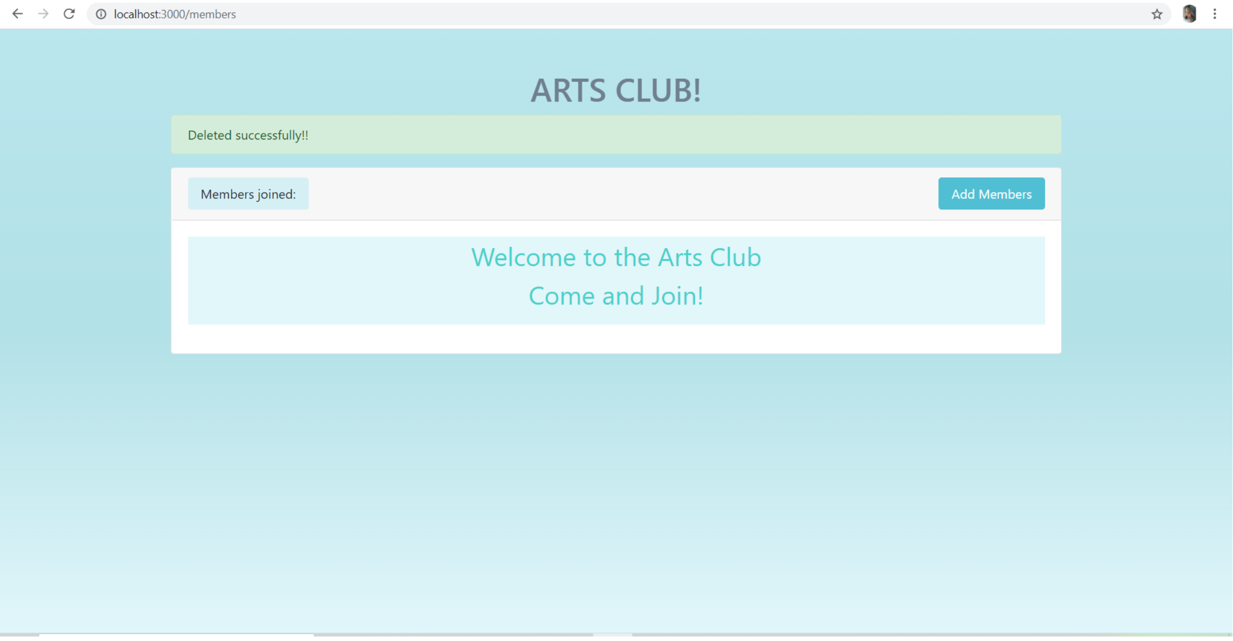


localhost:3000 says

Are You sure?

Sl No.	Name	Interested in	Action
1	Vinaya	Dance	Edit Delete

deleted



ARTS CLUB!

Deleted successfully!!

Welcome to the Arts Club

Come and Join!

5. CONCLUSION

The purpose of this project was to build an Arts Club Members Management System to make a computerised system for managing and monitoring the details and data corresponding to the members who are associated with the Arts Club of the school, college or other educational institutions. Successfully created the Arts Club Members Management System. The results are satisfactory. This project can be given additional updations and can be used for the management of members in other clubs also.

6. REFERENCE

- https://www.geeksforgeeks.org/
- https://way2tutorial.com/