

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashslakhtariya/sem4practicals/tree/main/FET>

Practical - 17 (MEAN Stack – CRUD Operations)

Objective : To understand the concept of services and dependency injection in Angular.

Exercise :

To understand the concept of MEAN Stack and connect the database in MongoDB to Angular application.

Structure :

main project : angular_dbdemo

database : mongodb (local install)

JS (server.js) (backend) :

```
const express = require('express')
const path = require('path')
const mongoose = require('mongoose')
const cors = require('cors')
const bodyParser = require('body-parser')

// Connecting with mongo db
mongoose
  .connect('mongodb://yash:haribol@127.0.0.1:27017/?authMechanism=DEFAULT')
  //your connection string (locally installed mongodb)
  .then((x) => {
    console.log(`Connected to Mongo! Database name:
"${x.connections[0].name}"`)
  })
  .catch((err) => {
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashslakhtariya/sem4practicals/tree/main/FET>

```
    console.error('Error connecting to mongo', err.reason)
  })

// Setting up port with express js
const employeeRoute = require('../backend/routes/employee.route')
const app = express()
app.use(bodyParser.json())
app.use(
  bodyParser.urlencoded({
    extended: false,
  }),
)
app.use(cors())
app.use(express.static(path.join(__dirname, '/angular_dbdemo'))) //change
to project's name
app.use('/', express.static(path.join(__dirname, '/angular_dbdemo')))
//change to project's name
app.use('/api', employeeRoute)

// Create port
const port = process.env.PORT || 4000
const server = app.listen(port, () => {
  console.log('Connected to port ' + port)
})

// Find 404 and hand over to error handler
app.use((req, res, next) => {
  next(createError(404))
})
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashlakhtariya/sem4practicals/tree/main/FET>

```
// error handler
app.use(function (err, req, res, next) {
  console.error(err.message) // Log error message in our server's console
  if (!err.statusCode) err.statusCode = 500 // If err has no specified
  error code, set error code to 'Internal Server Error (500)'
  res.status(err.statusCode).send(err.message) // All HTTP requests must
  have a response, so let's send back an error with its status code and
  message
})
```

JS (Employee.js) (backend model) :

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

// Define collection and schema
let EmployeeSc = new mongoose.Schema({
  name: {
    type: String
  },
  email: {
    type: String
  },
  designation: {
    type: String
  },
  phoneNumber: {
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashslakhtariya/sem4practicals/tree/main/FET>

```
      type: Number
    }
  }, {
    collection: 'employees'
  })
module.exports = mongoose.model('Employee', EmployeeSc)
```

JS (employee.route.js) (backend route) :

```
const express = require('express');
const employeeRoute = express.Router();
// Employee model
let Employee = require('../models/Employee');
// Add Employee
employeeRoute.route('/create').post((req, res, next) => {
  Employee.create(req.body).then((result) => {
    res.json(result);
    console.log(result)})
  .catch((err) => {console.log(err)})
});
// Get All Employees
employeeRoute.route('/').get((req, res) => {
  Employee.find({}).then((result) => {res.json(result)})
  .catch((err) => console.log(err))
})
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashslakhtariya/sem4practicals/tree/main/FET>

```
// Get single employee

employeeRoute.route('/read/:id').get((req, res) => {

  Employee.findById(req.params.id).then((result) => {res.json(result)})

  .catch((err) => console.log(err))

})

// Update employee

employeeRoute.route('/update/:id').put((req, res, next) => {

  Employee.findByIdAndUpdate(req.params.id,

req.body).then((result) => {res.json(result)})

  .catch((err) => console.log(err))

})

// Delete employee

employeeRoute.route('/delete/:id').delete((req, res, next) => {

Employee.findOneAndDelete(req.params.id).then((result) => {res.json(result)})

)

  .catch((err) => console.log(err))

})

module.exports = employeeRoute;
```

TS (employee-create.component.ts) :

```
import { Router } from '@angular/router';
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashlakhtariya/sem4practicals/tree/main/FET>

```
import { ApiService } from '../service/api.service';
import { Component, OnInit, NgZone } from '@angular/core';
import { FormGroup, FormBuilder, Validators } from '@angular/forms';
@Component({
  selector: 'app-employee-create',
  templateUrl: './employee-create.component.html',
  styleUrls: ['./employee-create.component.css'],
})
export class EmployeeCreateComponent implements OnInit {
  submitted = false;
  employeeForm!: FormGroup;
  EmployeeProfile: string[] = ['Finance', 'BDM', 'HR', 'Sales', 'Admin'];
  constructor(
    public fb: FormBuilder,
    private router: Router,
    private ngZone: NgZone,
    private apiService: ApiService
  ) {
    this.mainForm();
  }
  ngOnInit() {}
  mainForm() {
    this.employeeForm = this.fb.group({
      name: ['', [Validators.required]],
      email: [
        '',

```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashslakhtariya/sem4practicals/tree/main/FET>

```
[
    Validators.required,
    Validators.pattern('[a-z0-9._%+-]+@[a-z0-9.-]+.[a-z]{2,3}$'),
],
],
designation: ['', [Validators.required]],
phoneNumber: ['', [Validators.required,
Validators.pattern('^[0-9]+$')]],
});
}

// Choose designation with select dropdown
updateProfile(e:string) {
    this.employeeForm.get('designation')?.setValue(e, {
        onlySelf: true,
    });
}

// Getter to access form control
get myForm() {
    return this.employeeForm.controls;
}

onSubmit() {
    this.submitted = true;
    if (!this.employeeForm.valid) {
        return false;
    } else {
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashslakhtariya/sem4practicals/tree/main/FET>

```
        return

    this.apiService.createEmployee(this.employeeForm.value).subscribe({
        complete: () => {
            console.log('Employee successfully created!'),
            this.ngZone.run(() =>
this.router.navigateByUrl('/employees-list'));
        },
        error: (e) => {
            console.log(e);
        },
    });
}
}
```

TS (employee-edit.component.ts) :

```
import { Employee } from './../../model/employee';
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';
import { ApiService } from './../../service/api.service';
import { FormGroup, FormBuilder, Validators } from '@angular/forms';

@Component({
    selector: 'app-employee-edit',
    templateUrl: './employee-edit.component.html',
```


Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashslakhtariya/sem4practicals/tree/main/FET>

```
styleUrls: ['./employee-edit.component.css'],
}))

export class EmployeeEditComponent implements OnInit {
  submitted = false;
  editForm!: FormGroup;
  employeeData!: Employee[];
  EmployeeProfile: any = ['Finance', 'BDM', 'HR', 'Sales', 'Admin'];
  constructor(
    public fb: FormBuilder,
    private actRoute: ActivatedRoute,
    private apiService: ApiService,
    private router: Router
  ) { }
  ngOnInit() {
    this.updateEmployee();
    let id = this.actRoute.snapshot.paramMap.get('id');
    this.getEmployee(id);
    this.editForm = this.fb.group({
      name: ['', [Validators.required]],
      email: [
        '',
        [
          Validators.required,
          Validators.pattern('[a-z0-9._%+-]+@[a-z0-9.-]+.[a-z]{2,3}$'),
        ],
      ],
    });
  }
}
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashslakhtariya/sem4practicals/tree/main/FET>

```
    ],
    designation: ['', [Validators.required]],
    phoneNumber: ['', [Validators.required,
Validators.pattern('^[0-9]+$')]],
  });
}

// Choose options with select-dropdown
updateProfile(e: any) {
  this.editForm.get('designation')?.setValue(e, { onlySelf: true, });
}

// Getter to access form control
get myForm() {
  return this.editForm.controls;
}

getEmployee(id: any) {
  this.apiService.getEmployee(id).subscribe((data) => {
    this.editForm.setValue({
      name: data['name'],
      email: data['email'],
      designation: data['designation'],
      phoneNumber: data['phoneNumber'],
    });
  });
}

updateEmployee() {
  this.editForm = this.fb.group({
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashslakhtariya/sem4practicals/tree/main/FET>

```
name: ['', [Validators.required]],
email: ['',
  [
    Validators.required,
    Validators.pattern('[a-z0-9._%+-]+@[a-z0-9.-]+.[a-z]{2,3}$'),
  ],
],
designation: ['', [Validators.required]],
phoneNumber: ['', [Validators.required,
Validators.pattern('^[0-9]+$')]],
});
}
onSubmit(): void {
  this.submitted = true;
  if (!this.editForm.valid) {
    alert("Invalid attempt !!!!")
  } else {
    if (window.confirm('Are you sure?')) {
      let id = this.actRoute.snapshot.paramMap.get('id');
      this.apiService.updateEmployee(id, this.editForm.value).subscribe({
        complete: () => {
          this.router.navigateByUrl('/employees-list');
          console.log('Content updated successfully!');
        },
        error: (e) => {
          console.log(e);
        }
      });
    }
  }
}
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashlakhtariya/sem4practicals/tree/main/FET>

```
    },  
    });  
  }  
}  
}  
}
```

TS (employee-list.component.ts) :

```
import { Component, OnInit } from '@angular/core';  
import { ApiService } from './../../service/api.service';  
  
@Component({  
  selector: 'app-employee-list',  
  templateUrl: './employee-list.component.html',  
  styleUrls: ['./employee-list.component.css']  
})  
  
export class EmployeeListComponent implements OnInit {  
  Employee:any = [];  
  
  constructor(private apiService: ApiService) {  
    this.readEmployee();  
  }  
  
  ngOnInit() {}  
  
  readEmployee(){  
    this.apiService.getEmployees().subscribe((data) => {  
      this.Employee = data;  
    })  
  }  
}
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashslakhtariya/sem4practicals/tree/main/FET>

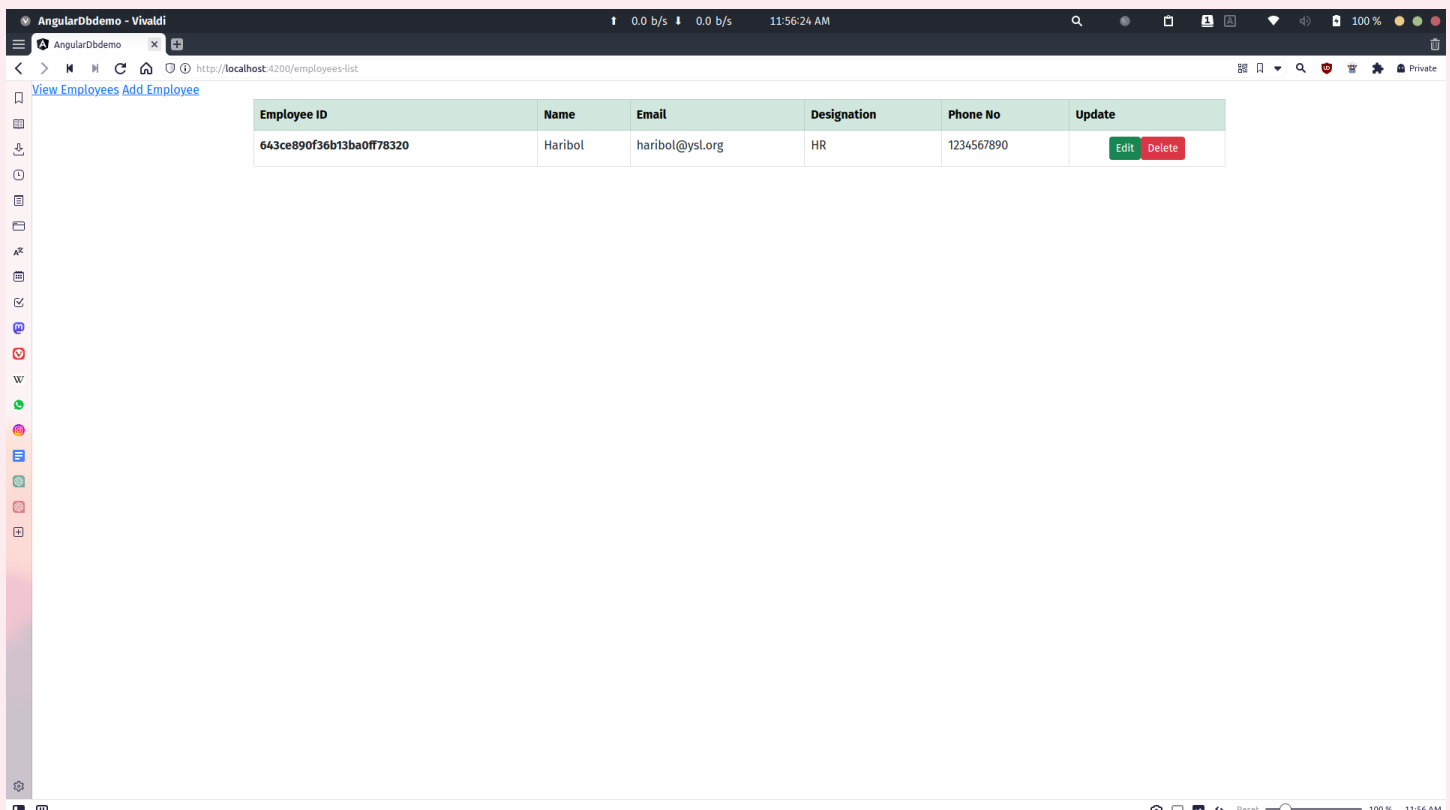
```
}

removeEmployee(employee:any, index:any) {
  if(window.confirm('Are you sure?')) {
    this.apiService.deleteEmployee(employee._id).subscribe((data) => {
      this.Employee.splice(index, 1);
    })
  }
}

}
```

Output :

1. Initial list



Employee ID	Name	Email	Designation	Phone No	Update
643ce890f36b13ba0ff78320	Haribol	haribol@ysl.org	HR	1234567890	<button>Edit</button> <button>Delete</button>

[View Employees](#) [Add Employee](#)

Name - Yash Lakhtariya

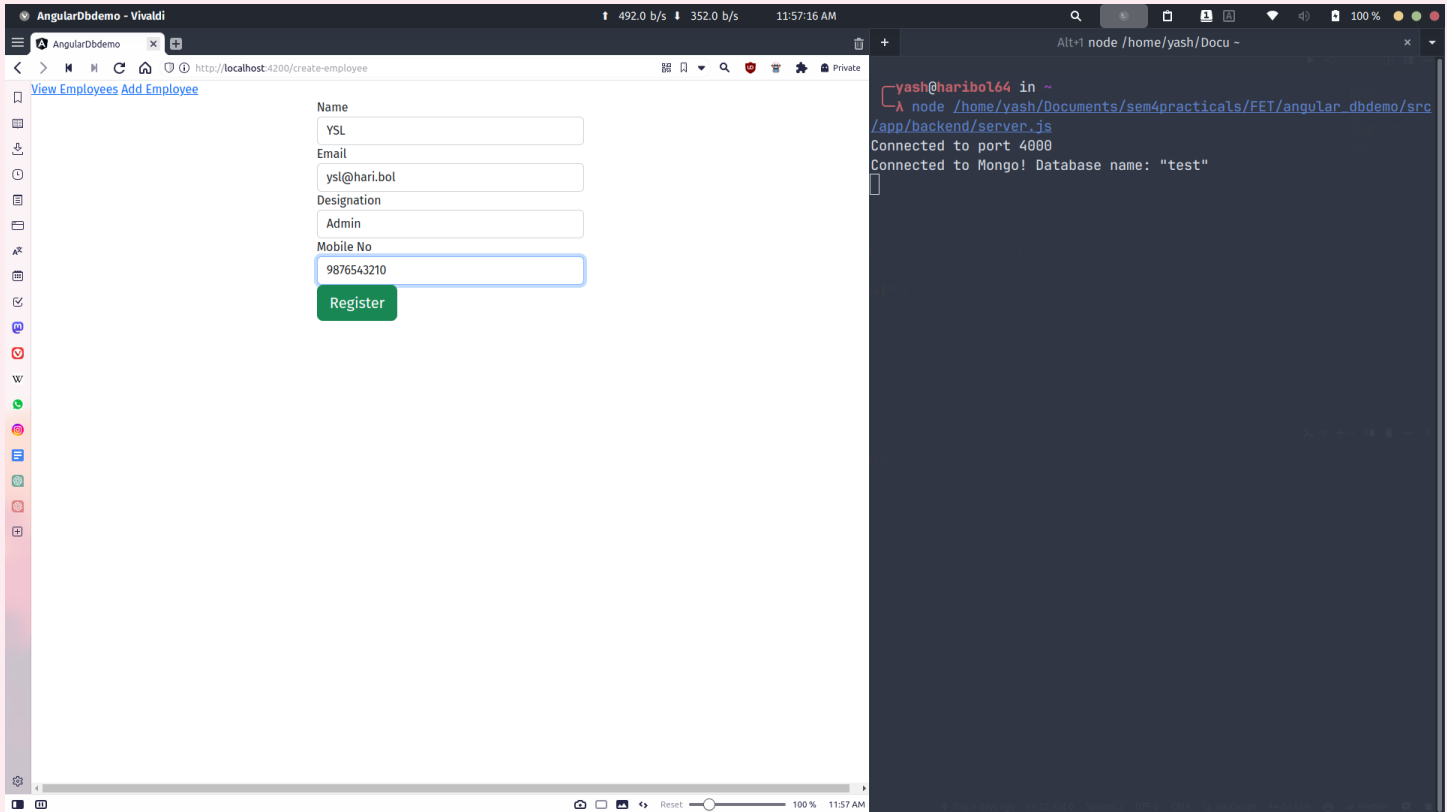
Enrollment number - 21162101012

Branch - CBA Batch - 41

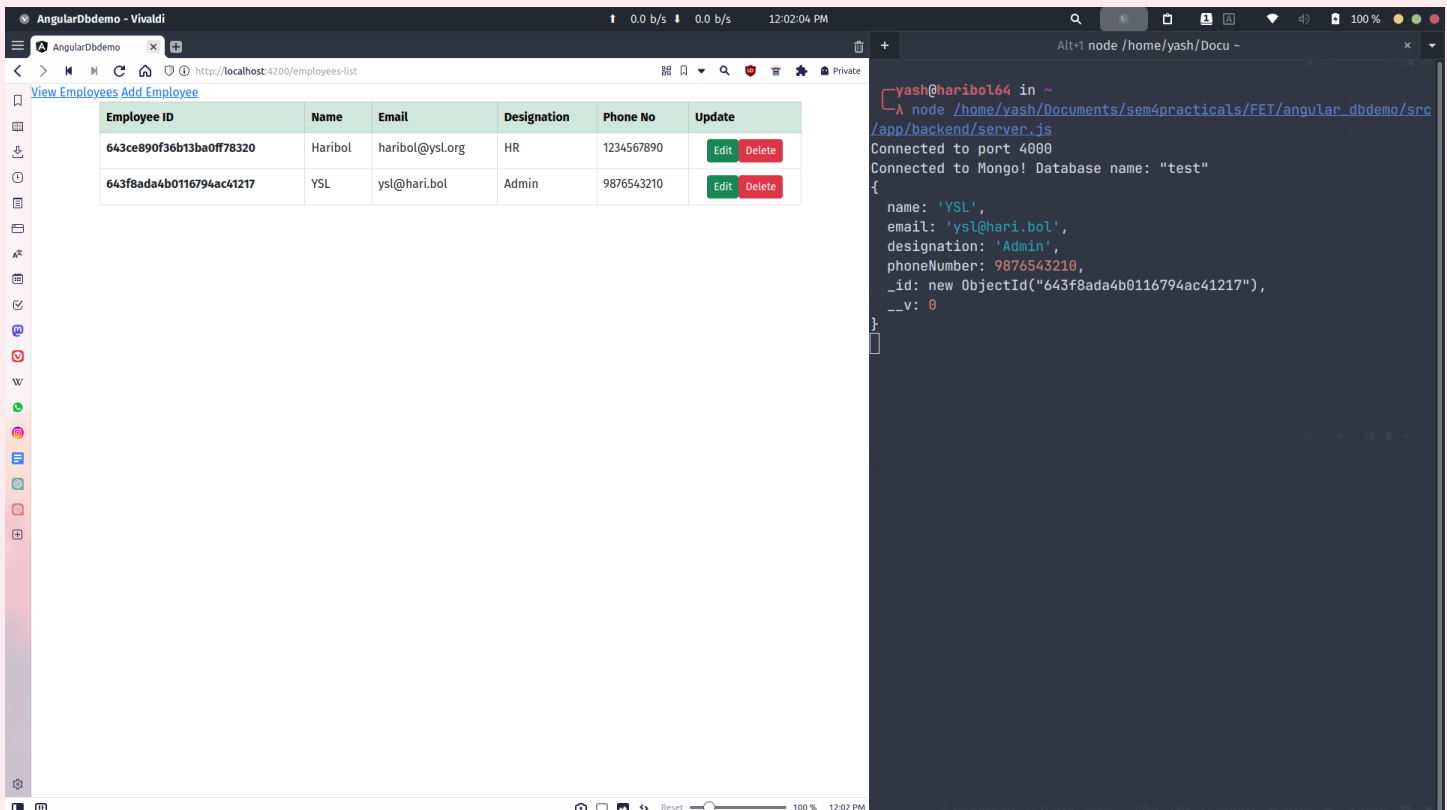
FET Practical 17

Code for all practicals : <https://github.com/yashslakhtariya/sem4practicals/tree/main/FET>

2. Adding details



3. Details inserted



Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashslakhtariya/sem4practicals/tree/main/FET>

4. Update details

The screenshot shows the AngularDbdemo application in a web browser. The 'Edit Employee' form is displayed, with the following fields:

- Name: YSL
- Email: ysl@hari.bol
- Designation: Admin
- Mobile No: 6432150789

The 'Update' button is visible. To the right, the terminal output shows the server.js file running, connected to port 4000 and the MongoDB database 'test'. The output displays the employee details:

```
{
  name: 'YSL',
  email: 'ysl@hari.bol',
  designation: 'Admin',
  phoneNumber: 9876543210,
  _id: new ObjectId("643f8ada4b0116794ac41217"),
  __v: 0
}
```

5. Details updated

The screenshot shows the AngularDbdemo application in a web browser. The 'employees-list' table is displayed, showing the updated details of the employee:

Employee ID	Name	Email	Designation	Phone No	Update
643ce890f36b13ba0ff78320	Haribol	haribol@ysl.org	HR	1234567890	Edit Delete
643f8ada4b0116794ac41217	YSL	ysl@hari.bol	Admin	6432150789	Edit Delete

To the right, the terminal output shows the server.js file running, connected to port 4000 and the MongoDB database 'test'. The output displays the employee details:

```
{
  name: 'YSL',
  email: 'ysl@hari.bol',
  designation: 'Admin',
  phoneNumber: 9876543210,
  _id: new ObjectId("643f8ada4b0116794ac41217"),
  __v: 0
}
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 41

FET Practical 17

Code for all practicals : <https://github.com/yashslakhtariya/sem4practicals/tree/main/FET>

6. Delete a record

The screenshot shows the AngularDbdemo application running in a browser. A confirmation dialog box is displayed over the table, asking "Are you sure?" with "OK" and "Cancel" buttons. The table has columns: Employee ID, Phone No, and Update. The first row has Employee ID 643ce890f36b13ba0ff78320 and Phone No 1234567890. The second row has Employee ID 643f8ada4b0116794ac41217 and Phone No 6432150789.

Employee ID	Phone No	Update
643ce890f36b13ba0ff78320	1234567890	<button>Edit</button> <button>Delete</button>
643f8ada4b0116794ac41217	6432150789	<button>Edit</button> <button>Delete</button>

```
yash@haribol64 in ~  
└─$ node /home/yash/Documents/sem4practicals/FET/angular_dbdemo/src/app/backend/server.js  
Connected to port 4000  
Connected to Mongo! Database name: "test"  
{  
  name: 'YSL',  
  email: 'ysl@hari.bol',  
  designation: 'Admin',  
  phoneNumber: 9876543210,  
  _id: new ObjectId("643f8ada4b0116794ac41217"),  
  __v: 0  
}
```

7. Record deleted

The screenshot shows the AngularDbdemo application running in a browser. The table now only contains one record. The terminal on the right shows the same MongoDB query result as in the previous screenshot.

Employee ID	Name	Email	Designation	Phone No	Update
643ce890f36b13ba0ff78320	Haribol	haribol@ysLorg	HR	1234567890	<button>Edit</button> <button>Delete</button>

```
yash@haribol64 in ~  
└─$ node /home/yash/Documents/sem4practicals/FET/angular_dbdemo/src/app/backend/server.js  
Connected to port 4000  
Connected to Mongo! Database name: "test"  
{  
  name: 'YSL',  
  email: 'ysl@hari.bol',  
  designation: 'Admin',  
  phoneNumber: 9876543210,  
  _id: new ObjectId("643f8ada4b0116794ac41217"),  
  __v: 0  
}
```