

Assignment

ExpressJS Web Application

with MySQL and Mongo Databases

Contents

Introduction	3
MySQL Database.....	3
Mongo Database	4
Marks	4
Submission of the Project	5
Overview of Web App	5
GET / (Home Page)	5
GET /employees (Employees Page)	6
GET /employees/edit/:eid, POST /employees/edit/:eid (Edit Employee Page)	7
GET /depts (Departments Page)	8
GET / /depts/delete/:did (Delete Department)	9
GET /employeesMongoDB (Employees (MongoDB) Page)	10
GET /employeesMongoDB/add, POST / employeesMongoDB/add	11
(Add Employee (MongoDB) Page)	11

Introduction

Write an ExpressJS web application that queries and updates a MySQL database and a Mongo database.

The MySQL database is called *proj2022* and can be downloaded from the *Project* section of Moodle for this module (*proj2022.sql*).

MySQL Database

The *collegedb* database consists of 4 tables:

employee	
Column Name	Details
Eid	A code representing the Employee's ID
Ename	The Employee's name
Role	The Employee's name (Manager or Employee)
Salary	The Employee's salary

location	
Column Name	Details
Lid	A code representing the Location's ID
county	The county of the Location

dept	
Column Name	Details
Did	A code representing the Department's ID
dname	The Department's name
Lid	A code representing the Department's location
budget	The Department's budget

emp_dept	
Column Name	Details
Eid	A code representing the Employee's ID
Did	A code representing the Department's ID

Full details of the Primary and Foreign Keys as well as datatypes can be found in the database itself.

Import *proj2022.sql* into MySQL to setup the initial database.

Mongo Database

The Mongo database should be called *employeesDB* and consists of a collection called *employees*, which initially has 7 documents as follows:

```
{ "_id": "E001", "phone": "0863325785", "email": "tom@gmail.com" }
{ "_id": "E006", "phone": "0875454215", "email": "brianM@gmail.com" }
{ "_id": "E009", "phone": "0857845121", "email": "will@hotmail.com" }
{ "_id": "E012", "phone": "0835580145", "email": "alan@yahoo.com" }
{ "_id": "E020", "phone": "0869985147", "email": "pj123@hotmail.com" }
{ "_id": "E025", "phone": "0854545711", "email": "martina@gmail.com" }
{ "_id": "E026", "phone": "0887878965", "email": "anna@hotmail.com" }
```

Import *employeesDB.json* into MongoDB to setup the initial database as follows:

```
mongoimport --db=employeesDB --collection=employees
--file="<path employeesDB.json>"
```

Marks

This assignment is worth 50% of the marks for the module.

90% of the marks for this assignment will be for implementing the functionality in this document.

10% of the marks for this assignment will be for innovation, extra functionality, exceeding the requirements listed in this document.

Any innovation etc. must be clearly indicated in a file called *Innovation.pdf* stored in the root folder of your application.

NOTE: Students may be invited to an MS Teams meeting for a [viva](#) explanation of any or all parts of their submission.

Plagiarism will be dealt with in accordance with the university's [Student Code](#).

Submission of the Project

The zipped project (named GXXXXXXX.zip or GXXXXXXX.7z where GXXXXXXX is your student number) should be uploaded to the *Project* section of Moodle no later than **Monday, January 2nd, 2023 at 8:00am.**

You must have a file entitled *GitLocation.pdf* in the root folder of your application which contains a link to the GIT repository you used when developing your project.

Overview of Web App

The web app can handle the following HTTP methods and verbs:

GET / (Home Page)

The Home page consists of 3 links:

- One to the *Employees* page
- One to the *Departments* page
- One to the *Employees (MongoDB)* page

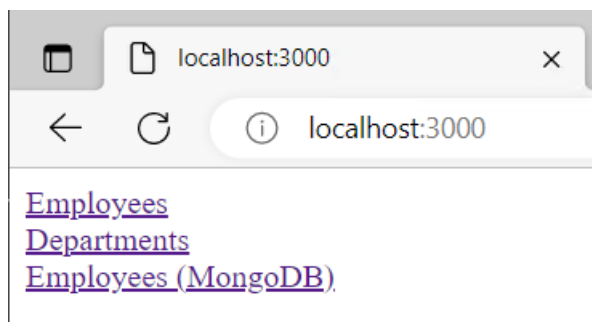
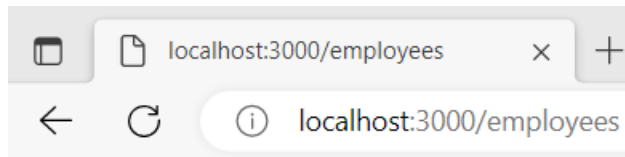


Figure 1 Main Page

GET /employees (Employees Page)

The *Employees* page:

- Shows details of all Employees
- Has an *Update* link for each Employee



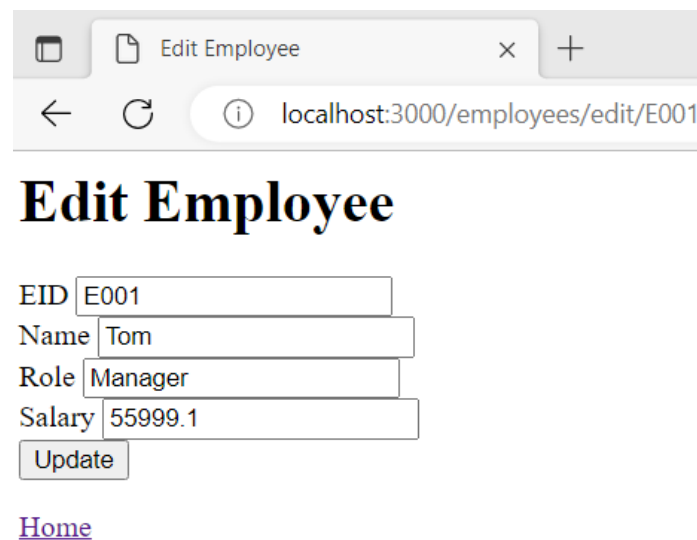
Employees

EID	Name	Role	Salary	Update
E001	Tom	Manager	55999.1	Update
E002	Anne	Employee	51100.09	Update
E003	Mary	Employee	52220.29	Update
E004	Alan	Employee	60220	Update
E005	Cathy	Manager	71500.12	Update
E006	Brian	Employee	71500.12	Update
E007	Pat	Employee	49500.83	Update
E008	Marion	Manager	69500.83	Update
E009	William	Employee	47700.22	Update
E010	Barry	Employee	45700.22	Update
E011	Damien	Employee	72750.51	Update
E012	Alan	Employee	52750.51	Update
E013	Thomas	Manager	72750.51	Update
E014	Mary-Ann	Manager	68751.53	Update
E015	Shane	Employee	49751.5	Update
E016	Seamus	Employee	47751.5	Update
E017	Fiona	Employee	47731.53	Update
E018	Deirdre	Employee	55731.63	Update
E019	David	Employee	59731.99	Update
E020	PT	Employee	55433.39	Update

Figure 2 *Employees Page*

GET /employees/edit/:eid, POST /employees/edit/:eid (Edit Employee Page)

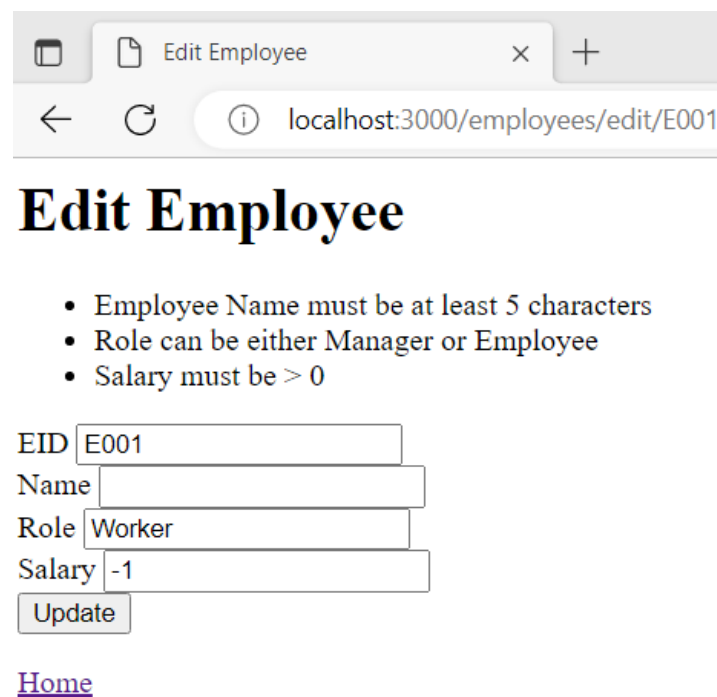
When the *Update* link is clicked beside an employee a GET request is sent to `/employees/edit/:eid` and the employee's details are shown.



The screenshot shows a web browser window with the title 'Edit Employee' and the URL 'localhost:3000/employees/edit/E001'. The page has a heading 'Edit Employee' and a form with the following fields: EID (E001), Name (Tom), Role (Manager), and Salary (55999.1). There is an 'Update' button below the form. A link labeled 'Home' is at the bottom left.

Figure 3 Edit Employee Page

- EID is not editable.
- Name should be a minimum of 5 characters.
- Role should be *Manager* or *Employee*.
- Salary should be > 0 .



The screenshot shows the same 'Edit Employee' page but with validation errors. The EID field is still 'E001'. The Name field is empty. The Role field is 'Worker'. The Salary field is '-1'. Red error messages are visible below the Name and Salary fields. The 'Update' button is still present, and the 'Home' link is at the bottom left.

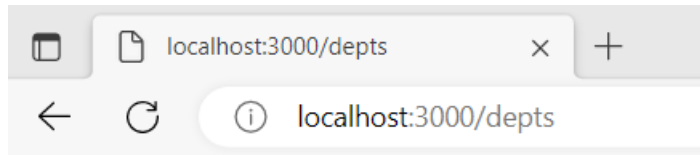
Figure 4 Edit Employee Page with Errors

When an Employee has been successfully updated in the MySQL database the user is returned to the *GET /employees* (Employees Page).

GET /depts (Departments Page)

The *Departments* page:

- Shows each department ID, name, budget and county the Department is in.
- Has a *Delete* action for each Department.
- Has a link back to the *GET / (Home Page)* page.



Departments

DID	Name	Budget	Location	Delete
OPS	Operations	1100332	Cork	Delete
R&D	Research & Devel	2000500	Dublin	Delete
FIN	Finance	1000000	Galway	Delete
HR	Human Resources	700500	Galway	Delete
SAL	Sales	900232	Galway	Delete

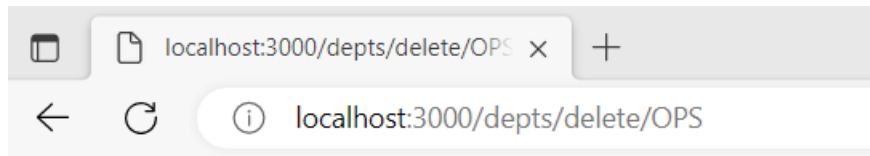
[Home](#)

Figure 5 *Departments Page*

GET //depts/delete/:did (Delete Department)

When the *Delete* link is clicked beside a department, a GET request is sent to */depts/delete/:did* and if there are no associated Employees, the Department is deleted from the MySQL database and the user is returned to **GET /depts (Departments Page)**.

If a Department has associated Employees, it cannot be deleted and an error message should be shown:



Error Message

OPS has Employees and cannot be deleted

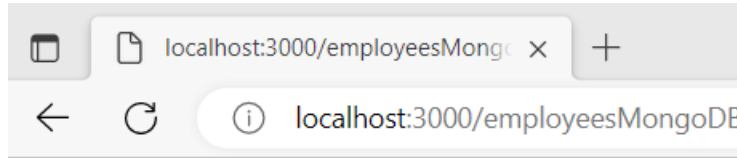
[Home](#)

Figure 6 Department not deleted

GET /employeesMongoDB (Employees (MongoDB) Page)

The *Employees (MongoDB)* page:

- Shows details of all Employees (from MongoDB).
- Has an *Add Employee (MongoDB)* link.
- Has a link back to the *Main* page.



Employees (MongoDB)

[Add Employee \(MongoDB\)](#)

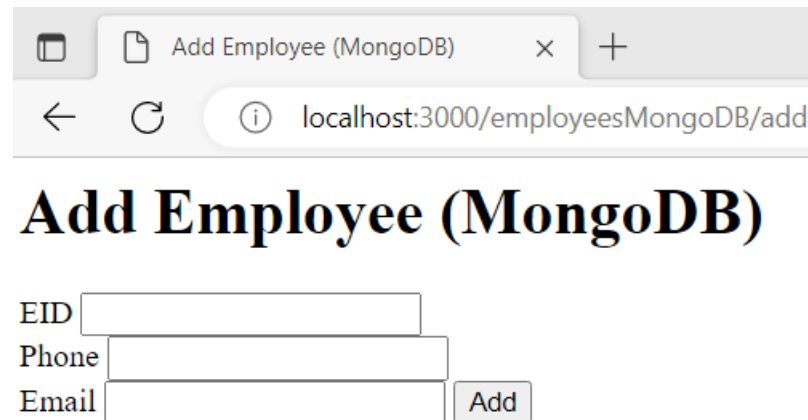
EID	Phone	Email
E001	0863325785	tom@gmail.com
E006	0875454215	brianM@gmail.com
E009	0857845121	will@hotmail.com
E012	0835580145	alan@yahoo.com
E020	0869985147	pj123@hotmail.com
E025	0854545711	martina@gmail.com
E026	0887878965	anna@hotmail.com

[Home](#)

Figure 7 List Employees (MongoDB)

***GET /employeesMongoDB/add, POST / employeesMongoDB/add
(Add Employee (MongoDB) Page)***

When the *Add Employee (MongoDB)* button is clicked a GET request is sent to /employeesMongoDB/add and the following page is displayed:



EID

Phone

Email

[Home](#)

[Home](#)
Figure 8 Add Employee (MongoDB)

The following conditions must be checked:

- EID must be 4 characters:
- Phone must be >5 characters.
- Email must be a valid email address.



EID

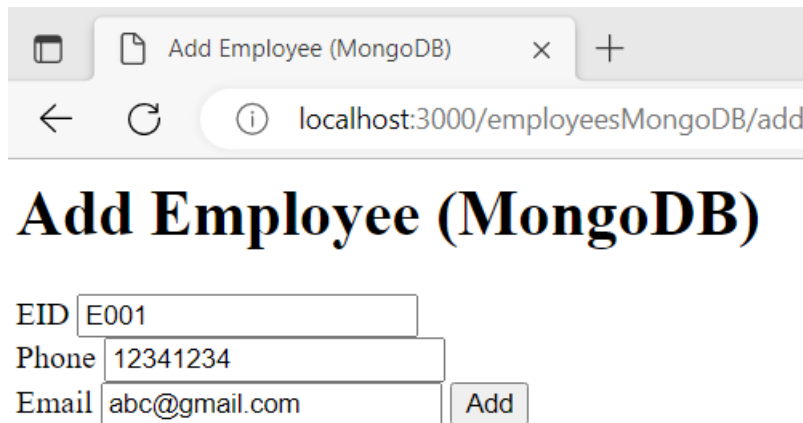
Phone

Email

[Home](#)

[Home](#)
Figure 9 Add Employee (MongoDB) - Errors

If an EID is entered that is already in the Mongo database, an error message should be displayed.



Add Employee (MongoDB)

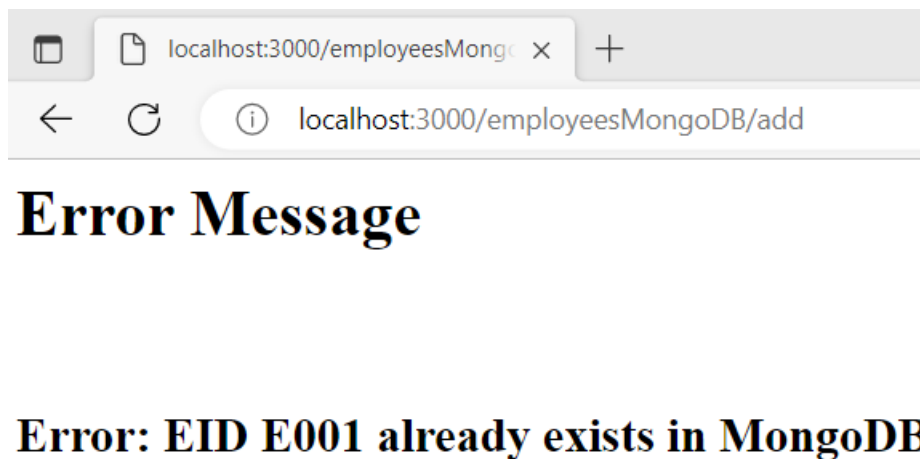
EID

Phone

Email

[Home](#)

Figure 10 Add Employee (MongoDB) – EID already in MongoDB



Error Message

Error: EID E001 already exists in MongoDB

[Home](#)

Figure 11 Add Employee (MongoDB) – Error: EID already exists in MongoDB.

If an EID is entered that is not in the MySQL database, an error message should be displayed.



Add Employee (MongoDB)

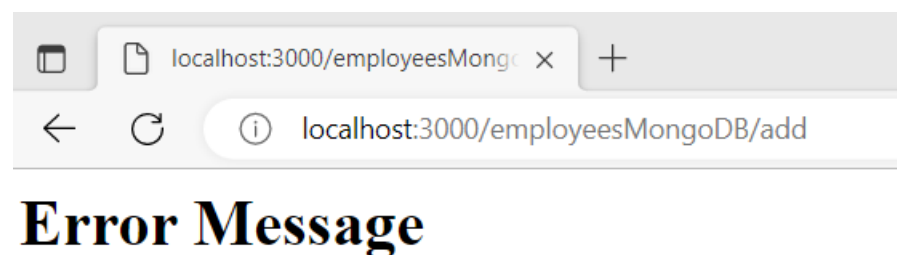
EID

Phone

Email

[Home](#)

Figure 12 Add Employee (MongoDB) – EID not in MySQL database



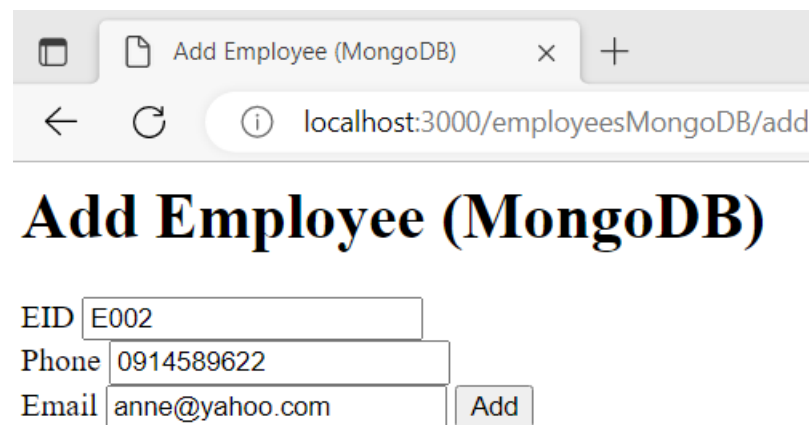
Error Message

Employee E999 doesn't exist in MySQL DB

[Home](#)

Figure 13 Add Employee (MongoDB) – Error: EID does not exist in MySQL database

When valid data has been entered, a new document should be added to the Mongo database, and user redirected to **GET /employeesMongoDB (Employees (MongoDB) Page)**.



Add Employee (MongoDB)

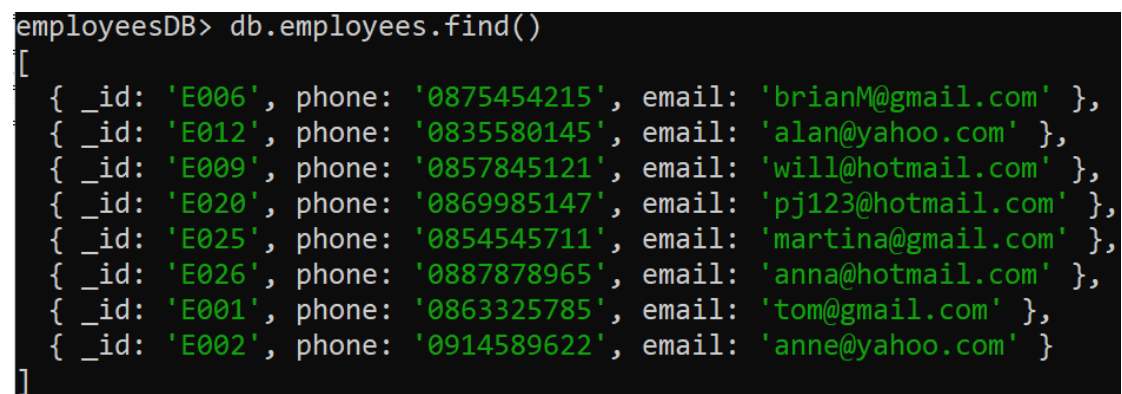
EID

Phone

Email

[Home](#)

Figure 14 Valid Employee (MongoDB) data



```
employeesDB> db.employees.find()
[
  { _id: 'E006', phone: '0875454215', email: 'brianM@gmail.com' },
  { _id: 'E012', phone: '0835580145', email: 'alan@yahoo.com' },
  { _id: 'E009', phone: '0857845121', email: 'will@hotmail.com' },
  { _id: 'E020', phone: '0869985147', email: 'pj123@hotmail.com' },
  { _id: 'E025', phone: '0854545711', email: 'martina@gmail.com' },
  { _id: 'E026', phone: '0887878965', email: 'anna@hotmail.com' },
  { _id: 'E001', phone: '0863325785', email: 'tom@gmail.com' },
  { _id: 'E002', phone: '0914589622', email: 'anne@yahoo.com' }
]
```

Figure 15 New document added to Mongo database.