

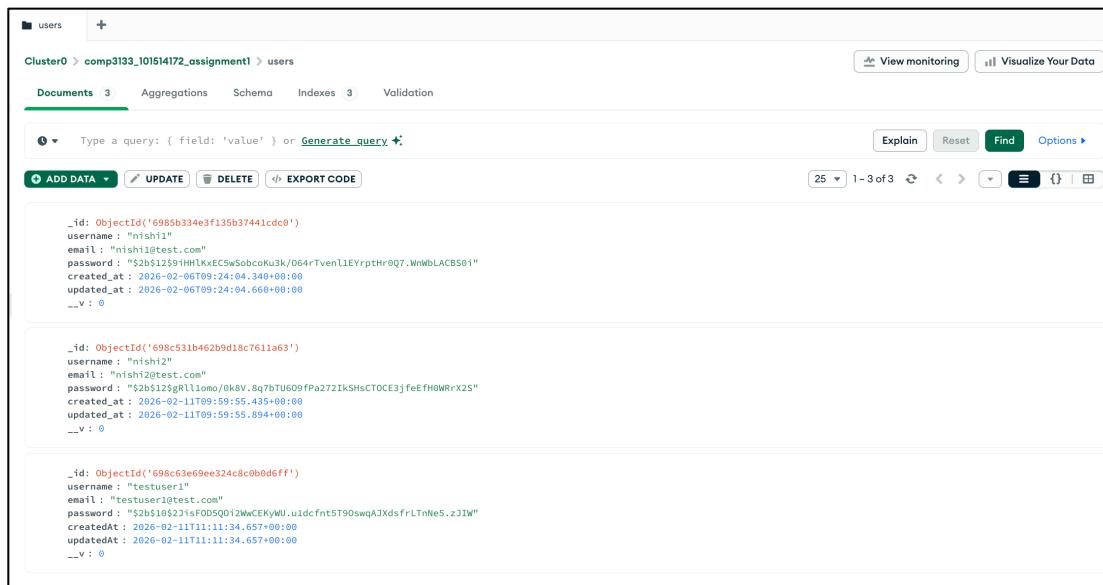
# COMP 3133 – FULL STACK DEVELOPMENT 2

## ASSIGNMENT 01 SNAPSHOTS

GITHUB REPO LINK: [https://github.com/inishimehta/101514172\\_COMP3133\\_Assignment01](https://github.com/inishimehta/101514172_COMP3133_Assignment01)

### MONGODB COLLECTIONS

#### users

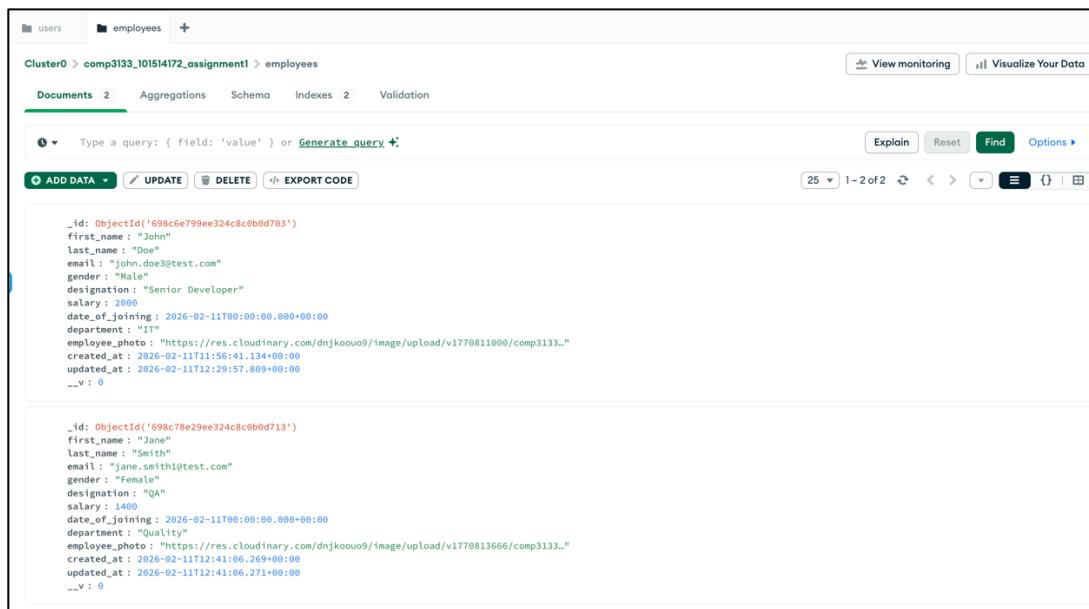


```
_id: ObjectId('6985b334e3f135b37441cdc0')
username : "nish1"
email : "nish1@test.com"
password : "$2b$12$gPl1omo/0k8V.8q7btU609fPa272IkSHsCT0CE3jfeEfH0WRrX2S"
created_at : 2026-02-06T09:24:04.349+0:00
updated_at : 2026-02-06T09:24:04.666+0:00
__v : 0

_id: ObjectId('698c531b462b9d18c7611a63')
username : "nish12"
email : "nish12@test.com"
password : "$2b$12$gPl1omo/0k8V.8q7btU609fPa272IkSHsCT0CE3jfeEfH0WRrX2S"
created_at : 2026-02-11T09:59:55.435+0:00
updated_at : 2026-02-11T09:59:55.894+0:00
__v : 0

_id: ObjectId('698c63e69ee324c8c0b0d6ff')
username : "testuser1"
email : "testuser1@test.com"
password : "$2b$10$232fcd050012WwCEKvWUuldcfn5T90swqAJxsfrLtnNe5.zJIW"
createdAt : 2026-02-11T11:11:34.657+0:00
updatedAt : 2026-02-11T11:11:34.657+0:00
__v : 0
```

#### employees



```
_id: ObjectId('698c6e799ee324c8c0b0d703')
first_name : "John"
last_name : "Doe"
email : "john.doe@test.com"
gender : "Male"
designation : "Senior Developer"
salary : 2000
date_of_joining : 2026-02-11T00:00:00.000+0:00
department : "IT"
employee_photo : "https://res.cloudinary.com/dnjkoouo9/image/upload/v1770811000/comp3133.."
created_at : 2026-02-11T11:56:41.134+0:00
updated_at : 2026-02-11T12:29:57.809+0:00
__v : 0

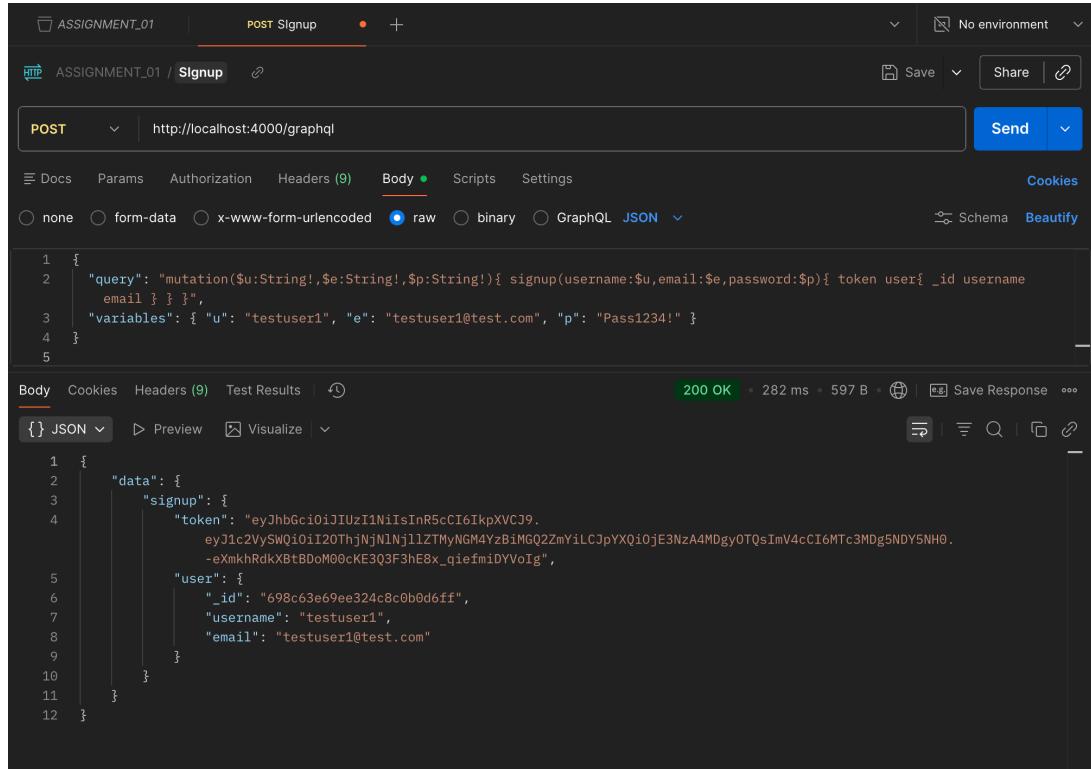
_id: ObjectId('698c78e29ee324c8c0b0d713')
first_name : "Jane"
last_name : "Smith"
email : "jane.smith1@test.com"
gender : "Female"
designation : "QA"
salary : 1400
date_of_joining : 2026-02-11T00:00:00.000+0:00
department : "Quality"
employee_photo : "https://res.cloudinary.com/dnjkoouo9/image/upload/v1770813666/comp3133.."
created_at : 2026-02-11T12:41:06.269+0:00
updated_at : 2026-02-11T12:41:06.271+0:00
__v : 0
```

**SAMPLE USER DETAILS FOR LOGIN/TESTING**

```
1  {
2    "query": "mutation($u:String!,$e:String!,$p:String!){ signup(username:$u,email:$e,password:$p){ token user{ _id username email } } }",
3    "variables": {
4      "u": "testuser1",
5      "e": "testuser1@test.com",
6      "p": "Pass1234!"
7    }
8 }
```

## API TESTING SCREENSHOTS

### 1. SignUp (Mutation)



The screenshot shows a Postman interface with the following details:

- Request URL:** `http://localhost:4000/graphql`
- Method:** POST
- Body (raw JSON):**

```

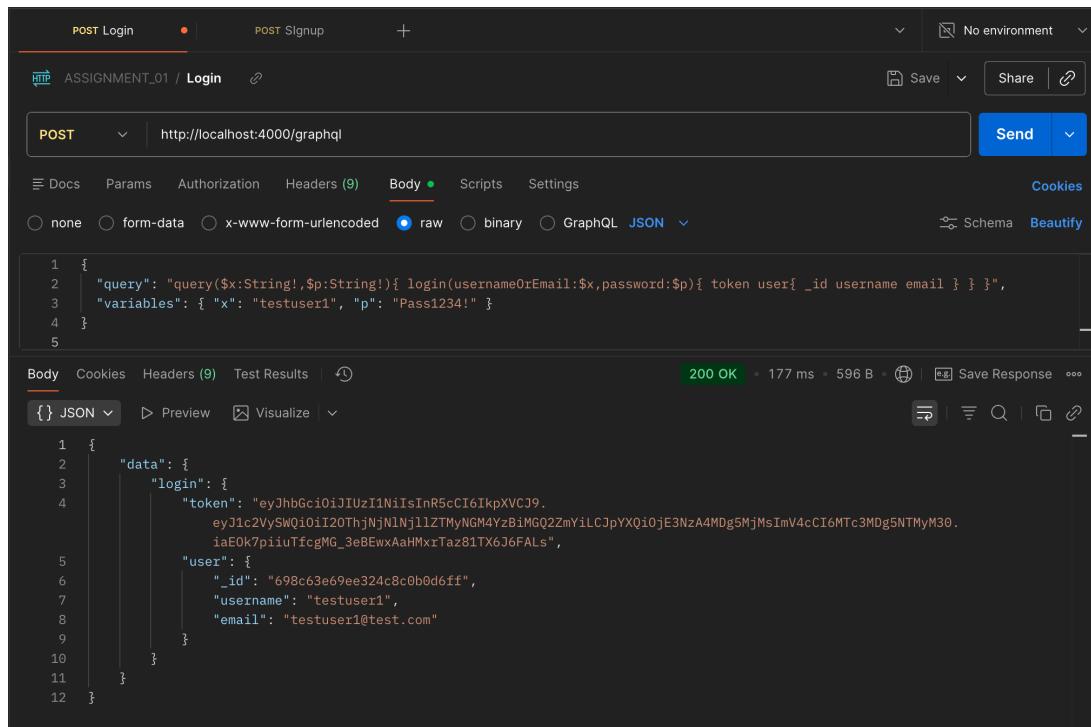
1 {
2   "query": "mutation($u:String!,$e:String!,$p:String!){ signup(username:$u,email:$e,password:$p){ token user{ _id username
3     email } } }",
4   "variables": { "u": "testuser1", "e": "testuser1@test.com", "p": "Pass1234!" }
5 }
```

- Response Status:** 200 OK
- Response Body (JSON):**

```

1 {
2   "data": {
3     "signup": {
4       "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
5         eyJic2VySWQiOii20ThjNjNlNjlZTMyNGM4YzBiMGQ2ZmYiLCJpYXQiOjE3NzA4MDgyOTQsImV4cCI6MTc3MDg5NDY5NH0.
6         -eXnhRdkXBtBDM00ckE3Qf3hE8x_qiefmiDYYoIg",
7       "user": {
8         "_id": "698c63e69ee324c8c0b0d6ff",
9         "username": "testuser1",
10        "email": "testuser1@test.com"
11      }
12    }
13  }
14 }
```

### 2. Login (Query)



The screenshot shows a Postman interface with the following details:

- Request URL:** `http://localhost:4000/graphql`
- Method:** POST
- Body (raw JSON):**

```

1 {
2   "query": "query($x:String!,$p:String!){ login(usernameOrEmail:$x,password:$p){ token user{ _id username email } } }",
3   "variables": { "x": "testuser1", "p": "Pass1234!" }
4 }
```

- Response Status:** 200 OK
- Response Body (JSON):**

```

1 {
2   "data": {
3     "login": {
4       "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
5         eyJic2VySWQiOii20ThjNjNlNjlZTMyNGM4YzBiMGQ2ZmYiLCJpYXQiOjE3NzA4MDg5MjMsImV4cCI6MTc3MDg5NTMyM30.
6         iaE0k7piiuTfcgMG_3eBEwxAaHmxrTaz81TX6J6fALs",
7       "user": {
8         "_id": "698c63e69ee324c8c0b0d6ff",
9         "username": "testuser1",
10        "email": "testuser1@test.com"
11      }
12    }
13  }
14 }
```

### 3. Get All Employees (Query)

The screenshot shows the GraphQL playground interface at <http://localhost:4000/graphq>. A POST request is being sent to the same URL. The body contains the following GraphQL query:

```
1 {
2   "query": "query{ getAllEmployees{ _id first_name last_name email designation department salary gender date_of_joining employee_photo } }"
3 }
```

The response status is 200 OK, with a response time of 51 ms and a response size of 324 B. The JSON response is:

```
1 {
2   "data": {
3     "getAllEmployees": []
4   }
5 }
```

The screenshot shows the GraphQL playground interface at <http://localhost:4000/graphq>. A POST request is being sent to the same URL. The body contains the following GraphQL query:

```
1 {
2   "query": "query{ getAllEmployees{ _id first_name last_name email designation department salary gender date_of_joining employee_photo } }"
3 }
```

The response status is 200 OK, with a response time of 45 ms and a response size of 999 B. The JSON response is a list of two employee objects:

```
1 {
2   "data": {
3     "getAllEmployees": [
4       {
5         "_id": "698c6f899ee324c8c0b0d706",
6         "first_name": "Jane",
7         "last_name": "Smith",
8         "email": "jane.smith1@test.com",
9         "designation": "QA",
10        "department": "Quality",
11        "salary": 1400,
12        "gender": "Female",
13        "date_of_joining": "1770768000000",
14        "employee_photo": "https://res.cloudinary.com/dnjkouuo9/image/upload/v1770811273/comp3133/employees/e38d9djhf1jxohq4pv6.jpg"
15      },
16      {
17        "_id": "698c6e799ee324c8c0b0d703",
18        "first_name": "John",
19        "last_name": "Doe",
20        "email": "john.doe3@test.com",
21        "designation": "Developer",
22        "department": "IT",
23        "salary": 1200,
24        "gender": "Male",
25        "date_of_joining": "1770768000000",
26        "employee_photo": "https://res.cloudinary.com/dnjkouuo9/image/upload/v1770811000/comp3133/employees/eamyugnzjlqvbnm37h.jpg"
27      }
28    ]
29  }
30 }
```

#### 4. Add New Employee (Mutation)

The screenshot shows the GraphQL playground interface with the following details:

- Environment:** ASSIGNMENT\_01 / Add New Employee
- Method:** POST
- URL:** http://localhost:4000/graphql
- Body:** Raw JSON (selected)
- Variables:** JSON object containing employee details.
- Response:** JSON object showing the added employee's details.

```
1 {  
2   "query": "mutation($in:EmployeeInput!){ addNewEmployee(input:$in){ _id first_name last_name email employee_photo designation department salary  
      gender date_of_joining }}",  
3   "variables": {  
4     "in": {  
5       "first_name": "John",  
6       "last_name": "Doe",  
7       "email": "john.doe3@test.com",  
8       "gender": "Male",  
9       "designation": "Developer",  
10      "salary": 1200,  
11      "date_of_joining": "2026-02-11",  
12      "department": "IT",  
13      "employee_photo": "https://plus.unsplash.com/premium_photo-1739786996022-5ed5b56834e2?q=80&w=1480&auto=format&fit=crop&ixlib=rb-4.1.0&  
14      ixid=M3wxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8fA%3D%3D"  
15    }  
16  }  
17  
18  {  
19    "data": {  
20      "addNewEmployee": {  
21        "_id": "698c6e799ee324c8cb0b0d703",  
22        "first_name": "John",  
23        "last_name": "Doe",  
24        "email": "john.doe3@test.com",  
25        "employee_photo": "https://res.cloudinary.com/dnjkoouo9/image/upload/v1770811000/comp3133/employees/eamyugnzjlvgbvamn37h.jpg",  
26        "designation": "Developer",  
27        "department": "IT",  
28        "salary": 1200,  
29        "gender": "Male",  
30        "date_of_joining": "1770768000000"  
31      }  
32    }  
33  }  
34  
35  [200 OK] 824 ms 657 B [Save Response] [Runner] [Start Proxy] [Cookies] [Vault] [Trash]
```

## 5. Search Employee with Id (Query)

The screenshot shows the GraphQL playground interface with a successful query execution. The query retrieves an employee by ID, returning details like first name, last name, email, designation, department, salary, gender, date of joining, and employee photo URL.

```

query($id:ID!){ searchEmployeeById(eid:$id){ _id first_name last_name email designation department salary gender date_of_joining employee_photo } }
variables: { "id": "698c6e799ee324c8c0b0d703" }

```

```

{
  "data": {
    "searchEmployeeById": {
      "_id": "698c6e799ee324c8c0b0d703",
      "first_name": "John",
      "last_name": "Doe",
      "email": "john.doe3@test.com",
      "designation": "Developer",
      "department": "IT",
      "salary": 1200,
      "gender": "Male",
      "date_of_joining": "1770768000000",
      "employee_photo": "https://res.cloudinary.com/dnjkooouo9/image/upload/v1770811000/comp3133/employees/eamyugnzjlvqbvamn37h.jpg"
    }
  }
}

```

## 6. Update Employee with Id (Mutation)

The screenshot shows the GraphQL playground interface with a successful mutation execution. The mutation updates an employee's salary and designation.

```

mutation($id:ID!,$in:EmployeeUpdateInput!){ updateEmployeeById(eid:$id,input:$in){ _id salary designation department updated_at employee_photo } }
variables: { "id": "698c6e799ee324c8c0b0d703", "in": { "salary": 2000, "designation": "Senior Developer" } }

```

```

{
  "data": {
    "updateEmployeeById": {
      "_id": "698c6e799ee324c8c0b0d703",
      "salary": 2000,
      "designation": "Senior Developer",
      "department": "IT",
      "updated_at": "1770812050293",
      "employee_photo": "https://res.cloudinary.com/dnjkooouo9/image/upload/v1770811000/comp3133/employees/eamyugnzjlvqbvamn37h.jpg"
    }
  }
}

```

8.

## 8. Search Employee with Designation or Department (Query)

The screenshot shows the GraphQL playground interface with a successful query execution. The query is:

```

1  {
2    "query": "query($d:String,$dep:String){ searchEmployeeByDesignationOrDepartment(designation:$d,department:$dep){ _id first_name designation
3      department } }",
4    "variables": { "d": "Senior Developer", "dep": null }
5  }

```

The response body is:

```

1  {
2    "data": {
3      "searchEmployeeByDesignationOrDepartment": [
4        {
5          "_id": "698c6e799ee324c8c0b0d703",
6          "first_name": "John",
7          "designation": "Senior Developer",
8          "department": "IT"
9        }
10      ]
11    }
12  }

```

The status is 200 OK with a response time of 66 ms and a size of 454 B.

## 7. Delete Employee by Id (Mutation)

The screenshot shows the GraphQL playground interface with a successful mutation execution. The mutation is:

```

1  {
2    "query": "mutation($id:ID!){ deleteEmployeeById(eid:$id) }",
3    "variables": { "id": "698c6f899ee324c8c0b0d706" }
4  }

```

The response body is:

```

1  {
2    "data": {
3      "deleteEmployeeById": true
4    }
5  }

```

The status is 200 OK with a response time of 59 ms and a size of 330 B.

## Employees after updating and deleting (Query)

The screenshot shows the GraphQL playground interface with the following details:

- HTTP Method:** POST
- URL:** http://localhost:4000/graphql
- Body Type:** raw
- Query:**

```
1 {
2   "query": "query{ getAllEmployees{ _id first_name last_name email designation department salary gender date_of_joining employee_photo } }"
3 }
```

- Response Status:** 200 OK
- Time:** 38 ms
- Size:** 667 B
- Content:** JSON
- Result:** A JSON object representing the list of employees.

```
1 {
2   "data": {
3     "getAllEmployees": [
4       {
5         "_id": "698c6e799ee324c8c0b0d703",
6         "first_name": "John",
7         "last_name": "Doe",
8         "email": "john.doe3@test.com",
9         "designation": "Senior Developer",
10        "department": "IT",
11        "salary": 2000,
12        "gender": "Male",
13        "date_of_joining": "1770768000000",
14        "employee_photo": "https://res.cloudinary.com/dnjkoouo9/image/upload/v1770811000/comp3133/employees/eamyugnzjlvqbavnn37h.jpg"
15      }
16    ]
17  }
18 }
```