

Elizabeth Thomas
ID: 101097106

COMP 3133 – Full Stack Development – II

Assignment – I

Submission Checklist:

Make single docx file with all labeled screenshots.

MongoDB Console Screenshots: o Provide screenshots of your MongoDB database showing collections.

The screenshot displays the MongoDB Atlas web interface. The top navigation bar includes the Atlas logo, a user profile dropdown for 'Elizabeth', and links for 'Access Manager' and 'Billing'. The main header shows 'Project 0' and 'Data Services'. The left sidebar contains a navigation menu with categories like Overview, DATABASE, Clusters, SERVICES, SECURITY, and Goto. The main content area is titled 'ELIZABETH'S ORG - 2024-10-11 > PROJECT 0 > DATABASES' and shows 'Cluster0' with version 8.0.4 in the AWS Oregon (us-west-2) region. The 'Collections' tab is active, displaying a list of collections under the 'comp3133fullstack' database. The 'employees' collection is selected, showing its storage and logical data sizes, total documents, and index sizes. A search bar is present, and a sample document is displayed with fields like _id, firstname, lastname, email, gender, city, designation, salary, created, and updatedat.

Cluster0 Data | Cloud: MongoDB | x

cloud.mongodb.com/v2/67094e055b6ce7532af523f1#/metrics/replicaSet/67094ee7672de12d3ec58099/explorer/comp3133fullstack/employees/find

Atlas Elizabeth's ... Access Manager Billing All Clusters Get Help Elizabeth

Project 0 Data Services Charts

Overview DATABASE Clusters SERVICES Atlas Search Stream Processing Triggers Migration Data Federation SECURITY Quickstart Backup Database Access Network Access Advanced Goto

ELIZABETH'S ORG - 2024-10-11 > PROJECT 0 > DATABASES

Cluster0 VERSION 8.0.4 REGION AWS Oregon (us-west-2)

Overview Real Time Metrics Collections Atlas Search Performance Advisor Online Archive Cmd Line

DATABASES: 11 COLLECTIONS: 29

+ Create Database

Search Namespaces

comp3133fullstack employees users sample_airbnb sample_analytics sample_geospatial sample_guides sample_mflix sample_restaurants sample_supplies sample_training sample_weatherdata test

comp3133fullstack.employees

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 4.3KB TOTAL DOCUMENTS: 21 INDEXES TOTAL SIZE: 36KB

Find Indexes Schema Anti-Patterns Aggregation Search Indexes

Generate queries from natural language in Compass

INSERT DOCUMENT

Filter Type a query: { field: 'value' } Reset Apply Options

_id: ObjectId('67b241f89e0d8047b0ef6081')
firstname: "Bob"
lastname: "Brown"
email: "bob.brown@example.com"
gender: "Male"
city: "Chicago"
designation: "DevOps Engineer"
salary: 85000
created: 2025-02-16T19:52:24.690+00:00
updatedat: 2025-02-16T19:52:24.690+00:00
_v: 0

_id: ObjectId('67b2532cf459ba916b167a83')

Elizabeth Thomas
ID: 101097106

2. Postman API Collection: o Export your Postman collection and include it in your submission.

✓ Submitted in D2L

4. Project ZIP File:

✓ Submitted in D2L

5. GitHub Project Link: Provide the link to your GitHub repository.

https://github.com/eliza2526/comp3133_assignment1

6. Sample User Detail: Include a sample user's details for testing login.

- username: user1 password: password1
- username: user2 password: password2
- username: newuser password: password123

o Add any comments or notes that could be helpful.

Employee Ids for CRUD Operations

- 67b1880b6e2a5d4d5d96f88a
- 67b2532cf459ba916b167a83
- 67b1880b6e2a5d4d5d96f88b
- 67b1880b6e2a5d4d5d96f88c
- 67b1880b6e2a5d4d5d96f88d
- 67b188e16e523fd9ee8c3819
- 67b191d9af7d514b043db89c

API endpoint

- <http://localhost:4000/graphql>

Elizabeth Thomas
ID: 101097106

3. Screenshots: Include screenshots of each API being tested along with responses.

GraphQL Implementation

Following is the list of to develop which accept all data as JSON Object whenever needed:

1 **Mutation** Signup Allow user to create new account

The screenshot displays the GraphQL IDE interface. The left pane contains the following query:

```
1 mutation {  
2   signup(username: "newuser1", email: "newuser1@example.com", password: "password123") {  
3     id  
4     username  
5     email  
6     created  
7     updatedAt  
8   }  
9 }  
10
```

The right pane shows the JSON response:

```
{  
  "data": {  
    "signup": {  
      "id": "67b2a80345b4c7dd7d648374",  
      "username": "newuser1",  
      "email": "newuser1@example.com",  
      "created": "1739761667039",  
      "updatedAt": null  
    }  
  }  
}
```

On the far right, the 'Documentation Explorer' sidebar is visible, showing a search bar and a list of root types: 'query: Query' and 'mutation: Mutation'.

Elizabeth Thomas
ID: 101097106

2 Query Login Allow user to access the system

The screenshot shows a web browser with a GraphQL client interface. The address bar shows the URL: `localhost:4000/graphql?query=query%20%7B%0A%20%20%20%20login(username%3A%20"newuser"%2C%20password%3A%20"password123")%20%7B%0A%20%20%20%20%20%20id%0A%...`. The interface has a top bar with "Cluster0 Data | Cloud: MongoD" and "GraphQL". Below the bar are buttons for "Prettify", "Merge", "Copy", and "History". The main editor is split into two panes. The left pane contains a GraphQL query:

```
1 query {  
2   login(username: "newuser", password: "password123") {  
3     id  
4     username  
5     email  
6     created  
7     updatedAt  
8   }  
9 }
```

The right pane shows the JSON response:

```
{  
  "data": {  
    "login": {  
      "id": "67b29a96ecaa923267dd6a6e",  
      "username": "newuser",  
      "email": "newuser@example.com",  
      "created": "1739758230587",  
      "updatedAt": null  
    }  
  }  
}
```

At the bottom left, there is a section labeled "QUERY VARIABLES". On the right side, there is a "Documentation Explorer" panel with a search bar "Search Schema...", a description "A GraphQL schema provides a root type of operation.", and a section "ROOT TYPES" listing "query: Query" and "mutation: Mutation".

ID: 101097106

3 Query

Get all employees User can get all employee list

The screenshot displays the GraphQL IDE interface. The top bar shows the browser address bar with the URL `localhost:4000/graphql?query=query%20%7B%6A%20%20%20%20employees%20%7B%6A%20%20%20%20%20id%6A%20%20%20%20%20firstname%6A%20%20%20%20%20%20`. Below the address bar, the GraphQL editor contains a query:

```
1 query {  
2   employees {  
3     id  
4     firstname  
5     lastname  
6     email  
7     gender  
8     city  
9     designation  
10    salary  
11    created  
12    updatedAt  
13  }  
14 }  
15
```

To the right of the editor are buttons for "Prettify", "Merge", "Copy", and "History". Below the editor, the "QUERY VARIABLES" section is visible.

The JSON response is displayed on the right side of the editor:

```
{  
  "data": {  
    "employees": [  
      {  
        "id": "67b1880b6e2a5d4d5d96f88a",  
        "firstname": "Jay",  
        "lastname": "Jones",  
        "email": "jay.jones@example.com",  
        "gender": "Female",  
        "city": "Los Angeles",  
        "designation": "Senior Software Engineer",  
        "salary": 90000,  
        "created": "1739687947562",  
        "updatedAt": "1739761911888"  
      },  
      {  
        "id": "67b1880b6e2a5d4d5d96f88b",  
        "firstname": "Alice",  
        "lastname": "Smith",  
        "email": "alice.smith@example.com",  
        "gender": "Female",  
        "city": "San Francisco",  
        "designation": "Product Manager",  
        "salary": 95000,  
        "created": "1739687947562",  
        "updatedAt": "1739761911888"  
      },  
      {  
        "id": "67b1880b6e2a5d4d5d96f88c",  
        "firstname": "Bob",  
        "lastname": "Brown",  
        "email": "bob.brown@example.com",  
        "gender": "Male",  
        "city": "Chicago",  
        "designation": "DevOps Engineer",  
        "salary": 85000,  
        "created": "1739687947562",  
        "updatedAt": "1739761911889"  
      },  
      {  
        "id": "67b1880b6e2a5d4d5d96f88d",  
        "firstname": "Mike",  
        "lastname": "Stevens",  

```

On the far right, the "Documentation Explorer" panel is open, showing a search bar labeled "Search Schema..." and a description: "A GraphQL schema provides a ro kind of operation." Below this, the "ROOT TYPES" section lists "query: Query" and "mutation: Mutation".

Elizabeth Thomas
ID: 101097106

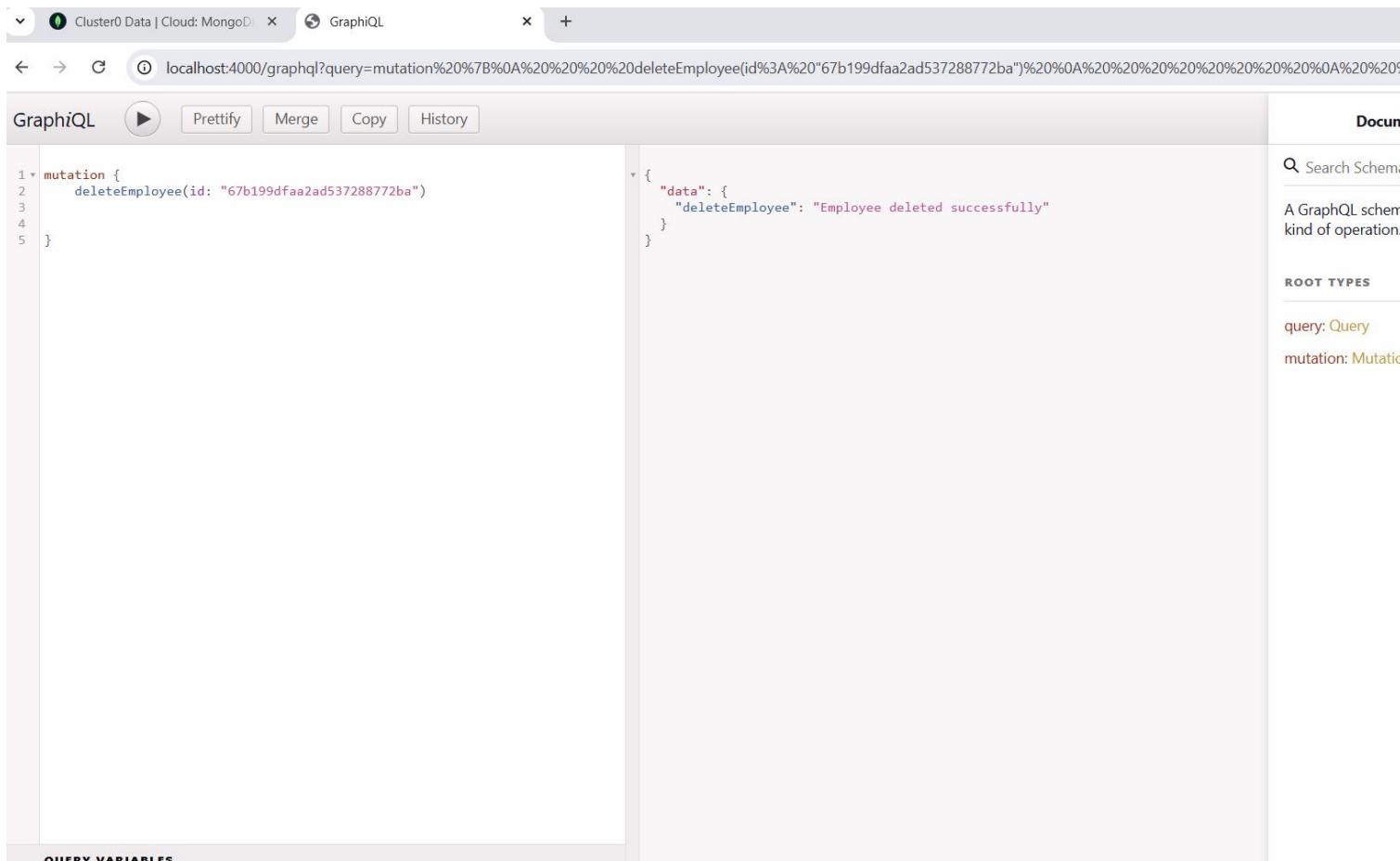
6 Mutation Update employee by eid User can update employee details

[illegible]

Elizabeth Thomas
ID: 101097106

7 Mutation

Delete employee by eid User can delete employee by employee id



Elizabeth Thomas
ID: 101097106

8 Query Search Employee by designation or department User can get employee list by designation or department

Cluster0 Data | Cloud: MongoD x GraphQL x +

localhost:4000/graphql?query=query%20%7B%0A%20%20%20%20employeesByDesignation(designation%3A%20"DevOps%20Engineer")%20%7B%0A%20

GraphQL

Prettify Merge Copy History

```
1 query {
2   employeesByDesignation(designation: "DevOps Engineer") {
3     id
4     firstname
5     lastname
6     email
7     gender
8     city
9     designation
10    department
11    salary
12    created
13    updatedAt
14  }
15 }
16
```

```
{
  "data": {
    "employeesByDesignation": [
      {
        "id": "67b1880b6e2a5d4d5d96f88c",
        "firstname": "Bob",
        "lastname": "Brown",
        "email": "bob.brown@example.com",
        "gender": "Male",
        "city": "Chicago",
        "designation": "DevOps Engineer",
        "department": null,
        "salary": 85000,
        "created": "1739687947562",
        "updatedAt": "1739762669604"
      },
      {
        "id": "67b191d9af7d514b043db89f",
        "firstname": "Bob",
        "lastname": "Brown",
        "email": "bob.brown@example.com",
        "gender": "Male",
        "city": "Chicago",
        "designation": "DevOps Engineer",
        "department": null,
        "salary": 85000,
        "created": "1739690457208",
        "updatedAt": "1739762669605"
      },
      {
        "id": "67b241f89e0d8047b0ef6081",
        "firstname": "Bob",
        "lastname": "Brown",
        "email": "bob.brown@example.com",
        "gender": "Male",
        "city": "Chicago",
        "designation": "DevOps Engineer",
        "department": null,
        "salary": 85000,
        "created": "1739735544690",
        "updatedAt": "1739762669605"
      }
    ]
  }
}
```

QUERY VARIABLES

Elizabeth Thomas
ID: 101097106

Validation Errors:

User already exists:

The screenshot displays the GraphQL Playground web application. The browser's address bar shows the URL: localhost:4000/graphql?query=m%20mutation%20{sign%20username%3A%20newuser%2C%20email%3A%20newuser%40example.com%2C%20password%3A%20password123}. The interface includes a top toolbar with buttons for Prettify, Merge, Copy, and History. The main area is split into two panes. The left pane contains a GraphQL mutation query:

```
1 mutation {  
2   signup(username: "newuser", email: "newuser@example.com", password: "password123") {  
3     id  
4     username  
5     email  
6     created  
7     updatedAt  
8   }  
9 }  
10
```

. The right pane shows the JSON response from the server:

```
{  
  "errors": [  
    {  
      "message": "Error signing up user: E11000 duplicate key error collection: comp3133fullstack.users index: username_1 dup key: { username: 'newuser' }",  
      "locations": [  
        {  
          "line": 2,  
          "column": 5  
        }  
      ],  
      "path": [  
        "signup"  
      ]  
    },  
  ],  
  "data": {  
    "signup": null  
  }  
}
```

. On the far right, there is a sidebar titled "Documentation Exp..." which contains a search box labeled "Search Schema...", a paragraph stating "A GraphQL schema provides a r... kind of operation.", and a section titled "ROOT TYPES" listing "query: Query" and "mutation: Mutation". At the bottom left, a partially visible section is labeled "QUERY VARIABLES".

ID: 101097106

Email wrong format

The screenshot displays the GraphQL Playground interface. The top bar shows the URL: `localhost:4000/graphql?query=mutation%20%7B%0A%20%20addEmployee(%0A%20%20%20%20firstname%3A%20"Lovie"%2C%0A%20%20%20%20lastname%3A`. Below the URL bar are buttons for "Prettify", "Merge", "Copy", and "History".

The left pane contains the following GraphQL query:

```
1 mutation {  
2   addEmployee(  
3     firstname: "Lovie",  
4     lastname: "Jackson",  
5     email: "lovie.jackson",  
6     gender: "Female",  
7     city: "Mississauga",  
8     designation: "Supervisor",  
9     department: "IT",  
10    salary: 10000  
11  ) {  
12    id  
13    firstname  
14    lastname  
15    email  
16    gender  
17    city  
18    designation  
19    department  
20    salary  
21  }  
22 }  
23 }
```

The right pane shows the JSON response, which indicates an error:

```
{  
  "errors": [  
    {  
      "message": "Error adding employee: Employee validation failed: email:  
Path `email` is invalid (lovie.jackson).",  
      "locations": [  
        {  
          "line": 2,  
          "column": 3  
        }  
      ],  
      "path": [  
        "addEmployee"  
      ]  
    },  
  ],  
  "data": {  
    "addEmployee": null  
  }  
}
```

ID: 101097106

Validation: salary less than zero

The screenshot displays the GraphQL Playground interface. The top bar shows the URL: `localhost:4000/graphql?query=mutation%20%7B%0A%20%20addEmployee(%0A%20%20%20%20firstname%3A%20"Lovie"%2C%0A%20%20%20%20lastname%3A%20"Jacks`. Below the URL bar, there are buttons for "Prettify", "Merge", "Copy", and "History".

The left pane contains the following GraphQL query:

```
1 mutation {  
2   addEmployee(  
3     firstname: "Lovie",  
4     lastname: "Jackson",  
5     email: "lovie.jackson@test.com",  
6     gender: "Female",  
7     city: "Missisauga",  
8     designation: "Supervisor",  
9     department: "IT",  
10    salary: -10000  
11  ) {  
12    id  
13    firstname  
14    lastname  
15    email  
16    gender  
17    city  
18    designation  
19    department  
20    salary  
21  }  
22 }  
23 }
```

The right pane shows the JSON response, which includes an error message:

```
{  
  "errors": [  
    {  
      "message": "Error adding employee: Employee validation failed: salary:  
Path `salary` (-10000) is less than minimum allowed value (0).",  
      "locations": [  
        {  
          "line": 2,  
          "column": 3  
        }  
      ],  
      "path": [  
        "addEmployee"  
      ]  
    },  
  ],  
  "data": {  
    "addEmployee": null  
  }  
}
```