

The screenshot shows a web application on the left and the MongoDB Atlas interface on the right. The web application is displaying a 400 Bad Request error with the message "Username has been taken". The MongoDB Atlas interface shows the 'Assignment01.users' collection with a document for 'bret' already existing.

**Web Application (Left):**

- Environment: COMP3123Assignment01
- Endpoint: POST localhost:8081/api/v1/user/signup
- Request Body (JSON):

```
{  "username": "bret",  "password": "bret@123",  "email": "bret123@georgebrown.ca"}
```
- Response (JSON):

```
{  "message": "Username has been taken"}
```

**MongoDB Atlas (Right):**

- Database: Assignment01
- Collection: users
- Document:

```
{  "_id": ObjectId('678a801c580c0e7293952d'),  "username": "bret",  "email": "bret123@georgebrown.ca",  "password": "$2b$10$c1WhQ0Za3qJH00Tr9ZVAn...",  "created_at": 2024-10-12T13:59:16.204+00:00,  "updated_at": 2024-10-12T13:59:16.204+00:00,  "__v": 0}
```

Figure 1 Sign up 01 checks for existing user

The screenshot shows a web application on the left and the MongoDB Atlas interface on the right. The web application is displaying a 200 OK response with a success message. The MongoDB Atlas interface shows the 'Assignment01.users' collection with a new document for 'saltedfish' added.

**Web Application (Left):**

- Environment: COMP3123Assignment01
- Endpoint: POST localhost:8081/api/v1/user/signup
- Request Body (JSON):

```
{  "username": "saltedfish",  "password": "bret@123",  "email": "saltedfish@georgebrown.ca"}
```
- Response (JSON):

```
{  "message": "User created successfully.",  "user_id": "678a801c580c0e72939527",  "Email": "saltedfish@georgebrown.ca",  "Password": "bret@123"}
```

**MongoDB Atlas (Right):**

- Database: Assignment01
- Collection: users
- Document:

```
{  "_id": ObjectId('678a801c580c0e72939527'),  "username": "saltedfish",  "email": "saltedfish@georgebrown.ca",  "password": "$2b$10$gF6r4M0Xucv4p0hgz...",  "created_at": 2024-10-12T13:56:44.771+00:00,  "updated_at": 2024-10-12T13:56:44.771+00:00,  "__v": 0}
```

Figure 2 Sign up 02 successful passwords hash

Home Workspaces API Network Upgrade

My Workspace New Import

Collections

- COMP3095
- COMP3123Assignment01
  - POST register
  - POST login
  - GET logout
  - GET getAllUsers
  - GET getAllEmployees
  - POST CreateNewEmployee
  - GET getEmployee
  - PUT updateEmployee
  - DEL deleteEmployee
  - GET New Request Copy
- Fullstack

Environments

History

HTTP COMP3123Assignment01 / login

POST localhost:8081/api/v1/user/login

Send

Body raw JSON Beautify

```
1 {
2   "username": "saltedfish",
3   "password": "bret@123"
4 }
```

Body 200 OK 90 ms 282 B

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "saltedfish has login successfully"
3 }
```

Console Postbot Runner Vault

Figure 3 Login success

The screenshot shows a development environment with VS Code on the left and a web browser on the right. In VS Code, a REST client is configured with a GET request to `localhost:8081/api/v1/emp/employees`. The response is a 200 OK status with a JSON array of two employee objects. The browser shows the MongoDB Atlas interface, displaying the `Assignment01.employees` collection with details like storage size, document count, and a list of indexes.

```
GET localhost:8081/api/v1/emp/employees
```

```
{
  "_id": "670a810bbc508c0e72939539",
  "firstname": "alice",
  "lastname": "johnson",
  "email": "alice.johnson@example.com",
  "position": "designer",
  "salary": 85000,
  "date_of_joining": "2023-08-10T00:00:00.000Z",
  "department": "design",
  "created_at": "2024-10-12T14:00:43.263Z",
  "updated_at": "2024-10-12T14:00:43.263Z"
}, {
  "_id": "670a814cbc508c0e7293953c",
  "firstname": "haoyun",
  "lastname": "yang",
  "email": "haoyun.yang@example.com",
  "position": "back-end dev",
  "salary": 77777,
  "date_of_joining": "2023-08-10T00:00:00.000Z",
  "department": "it",
  "created_at": "2024-10-12T14:01:48.662Z",
  "updated_at": "2024-10-12T14:01:48.662Z"
}
```

Figure 4 Get employee list

The screenshot shows the same development environment. In VS Code, a REST client is configured with a POST request to `localhost:8081/api/v1/emp/employees`. The request body is a JSON object representing a new employee. The response is a 200 OK status with a JSON object containing a success message and the new employee's ID. The browser shows the MongoDB Atlas interface, displaying the `Assignment01.employees` collection with details like storage size, document count, and a list of indexes.

```
POST localhost:8081/api/v1/emp/employees
```

```
{
  "firstname": "Alice",
  "lastname": "Johnson",
  "email": "alice.johnson@example.com",
  "position": "Designer",
  "salary": 85000,
  "date_of_joining": "2023-08-10T00:00:00.000Z",
  "department": "Design"
}
```

```
{
  "message": "Employee created successfully.",
  "employee_id": "670a810bbc508c0e72939539"
}
```

Figure 5 Create a new employee

The screenshot shows the Postman interface with a workspace named "My Workspace". The left sidebar displays a collection of API requests under "COMP3123Assignment01". The selected request is "getEmployee". The main panel shows the request details for a GET request to the endpoint `localhost:8081/api/v1/emp/employees/670a810bbc508c0e72939539`. The "Query Params" section is empty. The "Body" tab is selected, showing a JSON response with employee details. The status is "200 OK" with a response time of 35 ms and a body size of 540 B.

**Request:** GET `localhost:8081/api/v1/emp/employees/670a810bbc508c0e72939539`

**Query Params:**

Key	Value	Description
Key	Value	Description

**Body (JSON):**

```
1 {
2   "id": "670a810bbc508c0e72939539",
3   "firstname": "alice",
4   "lastname": "johnson",
5   "email": "alice.johnson@example.com",
6   "position": "designer",
7   "salary": 85000,
8   "date_of_joining": "2023-08-10T00:00:00.000Z",
9   "department": "design",
10  "created_at": "2024-10-12T14:09:43.203Z",
11  "updated_at": "2024-10-12T14:09:43.203Z",
12  "__v": 0
13 }
```

Figure 6 Get employee details by id

The screenshot shows the Postman interface with a workspace named "My Workspace". The left sidebar displays a collection of API requests under "COMP3123Assignment01". The selected request is "updateEmployee". The main panel shows the request details for a PUT request to the endpoint `localhost:8081/api/v1/emp/employees/670a810bbc508c0e72939539`. The "Body" tab is selected, showing a JSON payload to update the employee's position and salary. The status is "200 OK" with a response time of 41 ms and a body size of 629 B.

**Request:** PUT `localhost:8081/api/v1/emp/employees/670a810bbc508c0e72939539`

**Body (JSON):**

```
1 {
2   "position": "Senior Designer",
3   "salary": 95000
4 }
```

**Response (JSON):**

```
1 {
2   "message": "Employee ID 670a810bbc508c0e72939539 has been updated!",
3   "infoUpdate": {
4     "id": "670a810bbc508c0e72939539",
5     "firstname": "alice",
6     "lastname": "johnson",
7     "email": "alice.johnson@example.com",
8     "position": "senior designer",
9     "salary": 95000,
10    "date_of_joining": "2023-08-10T00:00:00.000Z",
11    "department": "design",
12    "created_at": "2024-10-12T14:09:43.203Z",
13    "updated_at": "2024-10-12T14:09:43.203Z",
14    "__v": 0
15  }
16 }
```

Figure 7 Update employee details by id

The image shows a split-screen view of a development environment. The left pane displays an API client interface with a workspace named 'My Workspace'. A collection of endpoints is listed on the left, including 'DELETE deleteEmployee'. The main area shows the details for this endpoint, including a 'Send' button and a 'Body' tab displaying a JSON response. The response indicates a successful deletion with a message and an ID. The right pane shows the MongoDB Atlas web interface. The 'Data Services' section is active, showing the 'Assignment01' database and the 'employees' collection. The 'Find' tab is selected, displaying a query result for one document. The document contains fields like '\_id', 'firstname', 'lastname', 'email', 'position', 'salary', 'date\_of\_joining', 'department', 'created\_at', and 'updated\_at'. The status bar at the bottom indicates 'System Status: All Good'.

**API Client Response Body:**

```
1 {
2   "message": "Employee has been deleted",
3   "emp_delete": {
4     "_id": "670a814cbc508c0e7293953c",
5     "firstname": "haoyun",
6     "lastname": "yang",
7     "email": "haoyun.yang@example.com",
8     "position": "back-end dev",
9     "salary": 77777,
10    "date_of_joining": "2023-08-10T00:00:00.000Z",
11    "department": "it",
12    "created_at": "2024-10-12T14:01:48.662Z",
13    "updated_at": "2024-10-12T14:01:48.662Z",
14    "__v": 0
15  }
16 }
```

**MongoDB Atlas Query Result:**

```
{
  "_id": ObjectId('670a810bbc508c0e72939539'),
  "firstname": "alice",
  "lastname": "johnson",
  "email": "alice.johnson@example.com",
  "position": "senior designer",
  "salary": 95000,
  "date_of_joining": "2023-08-10T00:00:00.000Z",
  "department": "design",
  "created_at": "2024-10-12T14:00:43.203+00:00",
  "updated_at": "2024-10-12T14:00:43.203+00:00",
  "__v": 0
}
```

Figure 8 Delete Employee by id