

## API testing using postman screenshots:

### 1. Login API

The screenshot shows the Postman application interface. On the left, there's a file tree for a project named 'EBD\_PROJECT'. In the main area, a POST request is being made to `http://localhost:5000/api/users/login`. The 'Body' tab is selected, showing raw JSON input:

```
1 [{}]
2 ... "email": "malak@hotmail.com",
3 ... "password": "123456"
4 ]
```

Below the request, the response status is shown as `200 OK` with a time of `265 ms` and a size of `548 B`. The response body is displayed in JSON format:

```
1 [{"message": "Login successful", "user": {"_id": "69386d1a183e9a9df6f3a6a2", "username": "malak", "email": "malak@hotmail.com", "password": "$2a$10$AfA8CcMLyewfZ2dUXii.EcCuAeGiM3VLi.fszUchop0V85vd0zOr.", "createdAt": "2025-12-09T18:40:26.880Z", "updatedAt": "2025-12-09T18:40:26.880Z", "__v": 0}]}]
```

### 2. Create Gam3eya API

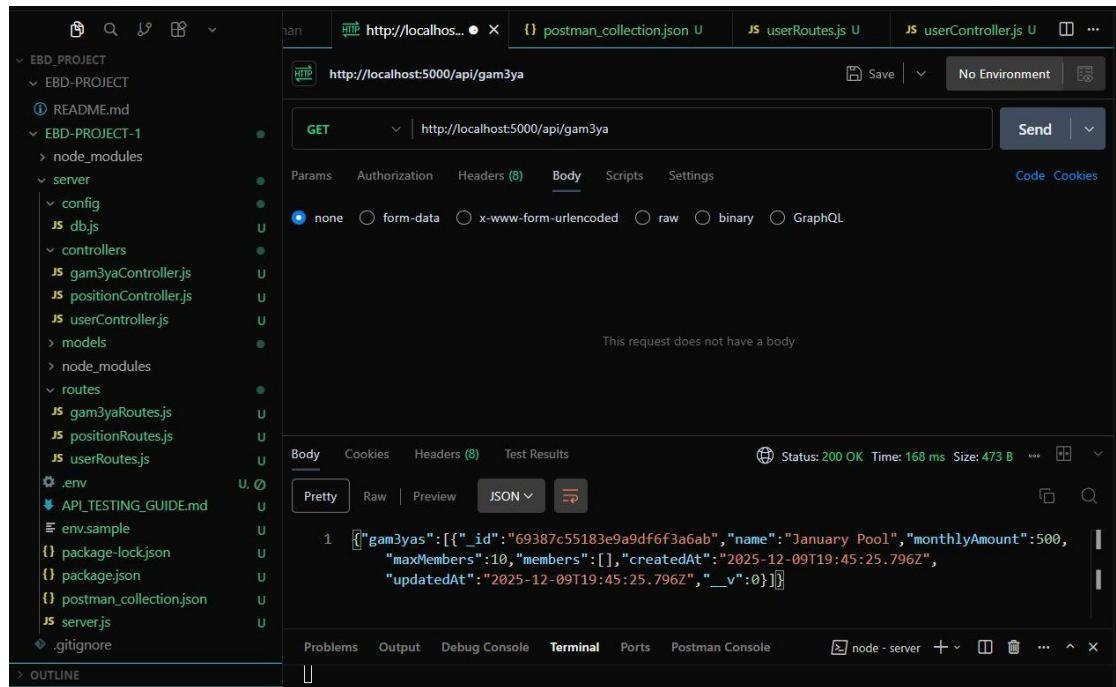
The screenshot shows the Postman application interface. On the left, there's a file tree for a project named 'EBD\_PROJECT'. In the main area, a POST request is being made to `http://localhost:5000/api/gam3ya`. The 'Body' tab is selected, showing raw JSON input:

```
1 [{}]
2 ... "name": "January Pool",
3 ... "monthlyAmount": 500,
4 ... "maxMembers": 10
5 ]
```

Below the request, the response status is shown as `201 Created` with a time of `287 ms` and a size of `502 B`. The response body is displayed in JSON format:

```
1 [{"message": "Gam3ya created", "gam3ya": {"name": "January Pool", "monthlyAmount": 500, "maxMembers": 10, "members": [], "_id": "69387c55183e9a9df6f3a6ab", "createdAt": "2025-12-09T19:45:25.796Z", "updatedAt": "2025-12-09T19:45:25.796Z", "__v": 0}}]
```

### 3. Get All Gam3yas API

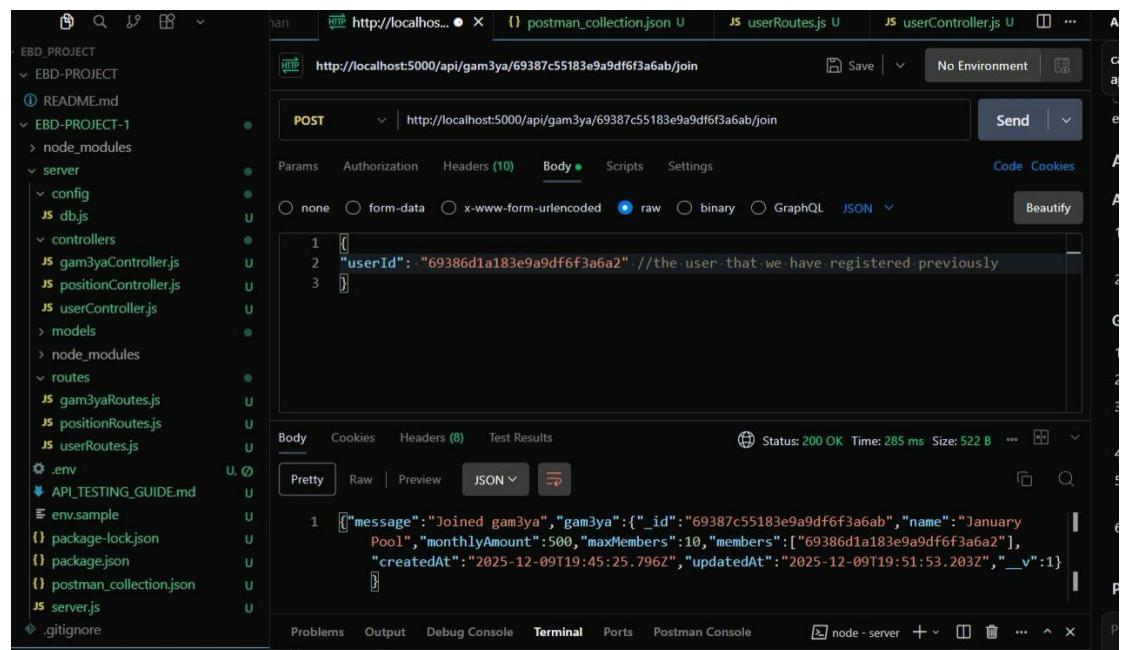


The screenshot shows the Postman application interface. On the left, there's a sidebar with project files: EBD\_PROJECT, EBD-PROJECT, README.md, EBD-PROJECT-1, node\_modules, server, config, db.js, controllers, gam3yaController.js, positionController.js, userController.js, models, routes, gam3yaRoutes.js, positionRoutes.js, userRoutes.js, .env, API\_TESTING\_GUIDE.md, env.sample, package-lock.json, package.json, postman\_collection.json, and server.js. The .gitignore file is also listed.

The main area shows a GET request to `http://localhost:5000/api/gam3ya`. The Body tab is selected, showing "none" as the type. The response status is 200 OK, time 168 ms, size 473 B. The response body is:

```
[{"gam3yas": [{"_id": "69387c55183e9a9df6f3a6ab", "name": "January Pool", "monthlyAmount": 500, "maxMembers": 10, "members": [], "createdAt": "2025-12-09T19:45:25.796Z", "updatedAt": "2025-12-09T19:45:25.796Z", "__v": 0}]}]
```

### 4. Join Gam3ya API

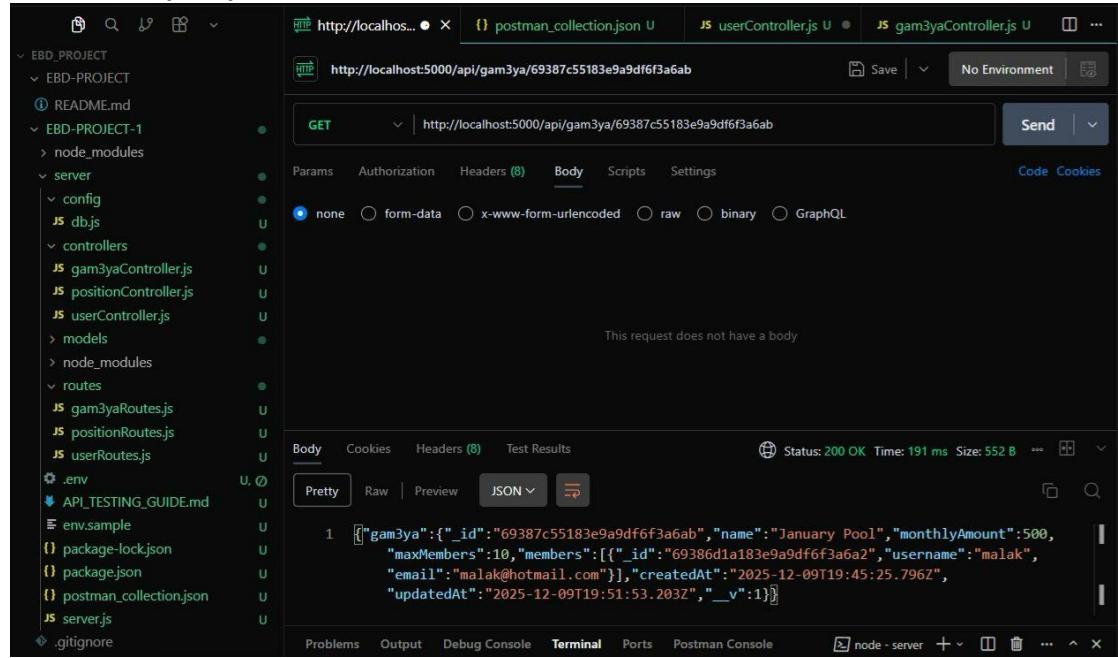


The screenshot shows the Postman application interface. The sidebar is identical to the previous screenshot, showing the same project files.

The main area shows a POST request to `http://localhost:5000/api/gam3ya/69387c55183e9a9df6f3a6ab/join`. The Body tab is selected, showing "raw" as the type. The response status is 200 OK, time 285 ms, size 522 B. The response body is:

```
{"message": "Joined gam3ya", "gam3ya": {"_id": "69387c55183e9a9df6f3a6ab", "name": "January Pool", "monthlyAmount": 500, "maxMembers": 10, "members": ["69386d1a183e9a9df6f3a6a2"], "createdAt": "2025-12-09T19:45:25.796Z", "updatedAt": "2025-12-09T19:51:53.203Z", "__v": 1}}
```

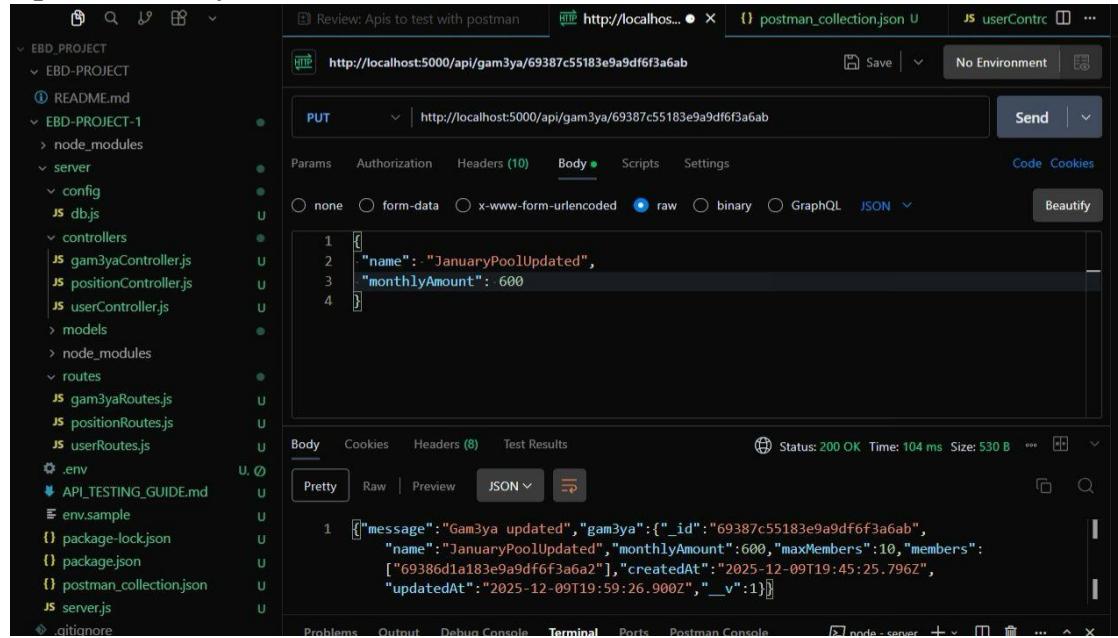
## 5. Get Gam3ya By ID API



The screenshot shows the Postman interface with a GET request to `http://localhost:5000/api/gam3ya/69387c55183e9a9df6f3a6ab`. The response status is 200 OK, and the response body is a JSON object:

```
[{"gam3ya": {"_id": "69387c55183e9a9df6f3a6ab", "name": "January Pool", "monthlyAmount": 500, "maxMembers": 10, "members": [{"_id": "69386d1a183e9a9df6f3a6a2", "username": "malak", "email": "malak@hotmail.com"}], "createdAt": "2025-12-09T19:45:25.796Z", "updatedAt": "2025-12-09T19:51:53.203Z", "__v": 1}]
```

## 6. Update Gam3ya API



The screenshot shows the Postman interface with a PUT request to `http://localhost:5000/api/gam3ya/69387c55183e9a9df6f3a6ab`. The response status is 200 OK, and the response body is a JSON object:

```
{"message": "Gam3ya updated", "gam3ya": {"_id": "69387c55183e9a9df6f3a6ab", "name": "JanuaryPoolUpdated", "monthlyAmount": 600, "maxMembers": 10, "members": [{"_id": "69386d1a183e9a9df6f3a6a2"}], "createdAt": "2025-12-09T19:45:25.796Z", "updatedAt": "2025-12-09T19:59:26.900Z", "__v": 1}}
```

## 7. Delete Gam3eya API

The screenshot shows the Postman interface with a successful DELETE request. The URL is `http://localhost:5000/api/gam3ya/69388081183e9a9df6f3a6b3`. The response status is 200 OK, and the body contains the message `{"message": "Gam3ya deleted"}`.

### Confirmation of deletion (Delete Gam3eya API)

The screenshot shows the Postman interface with a failed GET request. The URL is `http://localhost:5000/api/gam3ya/69388081183e9a9df6f3a6b3`. The response status is 404 Not Found, and the body contains the message `{"message": "Gam3ya not found"}`.

## 8. Create Position API

The screenshot shows the Postman interface with a collection named "EBD\_PROJECT". A POST request is being made to `http://localhost:5000/api/positions`. The request body is set to "raw" JSON, containing the following data:

```
[{"gam3yaId": "69387c55183e9a9df6f3a6ab", "userId": "69386d1a183e9a9df6f3a6a2", "positionNumber": 1}]
```

The response status is 201 Created, with a response body indicating the position was created successfully.

## 9. List Specific Position For Trade API

The screenshot shows the Postman interface with the same "EBD\_PROJECT" collection. A POST request is being made to `http://localhost:5000/api/positions/69388758183e9a9df6f3a6b7/list`. The request body is set to "raw" JSON, containing the following data:

```
{"_id": "69388758183e9a9df6f3a6b7"}
```

The response status is 200 OK, with a response body indicating the position was listed for trade successfully.

## 10. Get All Positions Listed For Trade API

The screenshot shows the Postman interface with a successful API call. The URL is `http://localhost:5000/api/positions/trade`. The response status is `200 OK`, time `1731 ms`, size `768 B`. The response body is a JSON array of positions:

```
[{"positions": [{"_id": "69388758183e9a9df6f3a6b7", "gam3yaId": {"_id": "69387c55183e9a9df6f3a6ab", "name": "JanuaryPoolUpdated", "monthlyAmount": 600, "maxMembers": 10, "members": ["69386d1a183e9a9df6f3a6a2"], "createdAt": "2025-12-09T19:45:25.796Z", "updatedAt": "2025-12-09T19:59:26.900Z", "__v": 1}, "userId": {"_id": "69386d1a183e9a9df6f3a6a2", "username": "malak", "email": "malak@hotmail.com"}, "positionNumber": 1, "isForTrade": true, "createdAt": "2025-12-09T20:32:24.568Z", "updatedAt": "2025-12-09T20:40:26.572Z", "__v": 0}]}]
```

## 11. Get Positions API

The screenshot shows the Postman interface with a successful API call. The URL is `http://localhost:5000/api/positions`. The response status is `200 OK`, time `594 ms`, size `768 B`. The response body is a JSON array of positions:

```
[{"positions": [{"_id": "69388758183e9a9df6f3a6b7", "gam3yaId": {"_id": "69387c55183e9a9df6f3a6ab", "name": "JanuaryPoolUpdated", "monthlyAmount": 600, "maxMembers": 10, "members": ["69386d1a183e9a9df6f3a6a2"], "createdAt": "2025-12-09T19:45:25.796Z", "updatedAt": "2025-12-09T19:59:26.900Z", "__v": 1}, "userId": {"_id": "69386d1a183e9a9df6f3a6a2", "username": "malak", "email": "malak@hotmail.com"}, "positionNumber": 1, "isForTrade": true, "createdAt": "2025-12-09T20:32:24.568Z", "updatedAt": "2025-12-09T20:40:26.572Z", "__v": 0}]}]
```

## 12. Get Positions By ID API

The screenshot shows the Postman interface with a successful GET request to `/api/positions/69388758183e9a9df6f3a6b7`. The response status is `200 OK` with a time of `136 ms` and a size of `765 B`. The response body is a JSON object representing a position:

```
{"position": {"_id": "69388758183e9a9df6f3a6b7", "gam3yaId": "69387c55183e9a9df6f3a6ab", "name": "JanuaryPoolUpdated", "monthlyAmount": 600, "maxMembers": 10, "members": ["69386d1a183e9a9df6f3a6a2"], "createdAt": "2025-12-09T19:45:25.796Z", "updatedAt": "2025-12-09T19:59:26.900Z", "__v": 1}, "userId": {"_id": "69386d1a183e9a9df6f3a6a2", "username": "malak", "email": "malak@hotmail.com"}, "positionNumber": 1, "isForTrade": true, "createdAt": "2025-12-09T20:32:24.568Z", "updatedAt": "2025-12-09T20:40:26.572Z", "__v": 0}}
```

## 13. Update Position API

The screenshot shows the Postman interface with a successful PUT request to `/api/positions/69388758183e9a9df6f3a6b7`. The response status is `200 OK` with a time of `102 ms` and a size of `542 B`. The response body is a JSON object containing a message:

```
{"message": "Position updated", "position": {"_id": "69388758183e9a9df6f3a6b7", "gam3yaId": "69387c55183e9a9df6f3a6ab", "userId": "69386d1a183e9a9df6f3a6a2", "positionNumber": 5, "isForTrade": true, "createdAt": "2025-12-09T20:32:24.568Z", "updatedAt": "2025-12-09T20:50:16.787Z", "__v": 0}}
```

## 14. Delete Positions API

The screenshot shows the Postman interface with a successful DELETE request to `http://localhost:5000/api/positions/69388758183e9a9df6f3a6b7`. The response status is 200 OK, and the body contains the message `{"message": "Position deleted"}`.

## 15. Buy Position API

The screenshot shows the Postman interface with a successful POST request to `http://localhost:5000/api/positions/69388f894328f9a2bf3636da/buy`. The request body is a JSON object with a single key-value pair: `{"newUserId": "69386d1a183e9a9df6f3a6a2"}`. The response status is 200 OK, and the body contains a detailed message about the purchased position.

```
1 {"message": "Position purchased", "position": {"_id": "69388f894328f9a2bf3636da", "gam3yaId": {"_id": "69387c55183e9a9df6f3a6ab", "name": "JanuaryPoolUpdated"}, "userId": {"_id": "69386d1a183e9a9df6f3a6a2", "username": "malak", "email": "malak@hotmail.com"}, "positionNumber": 1, "isForTrade": false, "createdAt": "2025-12-09T21:07:21.280Z", "updatedAt": "2025-12-09T21:45:31.041Z", "_v": 0}, "transactionDetails": {"position": {"positionId": "69388f894328f9a2bf3636da", "positionNumber": 1, "gam3yaId": "69387c55183e9a9df6f3a6ab", "gam3yaName": "JanuaryPoolUpdated"}, "previousOwner": {"userId": "69386d1a183e9a9df6f3a6a2", "username": "malak", "email": "malak@hotmail.com"}, "newOwner": {"userId": "69386d1a183e9a9df6f3a6a2", "username": "malak", "email": "malak@hotmail.com"}}}
```

## 16. Swap Position API

The screenshot shows the Postman application interface. On the left, there is a sidebar with project files: EBD\_PROJECT, EBD-PROJECT, README.md, EBD-PROJECT-1, node\_modules, server, config, db.js, controllers, gam3yaCont..., positionCon..., userControll..., models, node\_modules, routes, gam3yaRout..., positionRout..., userRoutes.js, .env, APITESTING..., env.sample, package-lock..., package.json, postman.coll..., server.js, and .gitignore. Below these are OUTLINE and TIMELINE.

The main area shows a POST request to <http://localhost:5000/api/positions/swap>. The Body tab is selected, showing raw JSON input:

```
1 [{"positionId": "69388f894328f9a2bf3636da", "positionNumber": 1, "gam3yaId": "69387c55183e9a9df6f3a6ab", "gam3yaName": "JanuaryPoolUpdated", "previousOwner": {"userId": "69386d1a183e9a9df6f3a6a2", "username": "malak", "email": "malak@hotmail.com"}, "newOwner": {"userId": "69389075e3f3901a71816fea", "username": "leila", "email": "leila@hotmail.com"}}, {"positionId": "693899cfa...d06adf9fe7a382", "positionNumber": 2, "gam3yaId": "69387c55183e9a9df6f3a6ab", "gam3yaName": "JanuaryPoolUpdated", "previousOwner": {"userId": "69389075e3f3901a71816fea", "username": "leila", "email": "leila@hotmail.com"}, "newOwner": {"userId": "69386d1a183e9a9df6f3a6a2", "username": "malak", "email": "malak@hotmail.com"}}, {"positions": [{"id": "69388f894328f9a2bf3636da", "gam3yaId": "69387c55183e9a9df6f3a6ab", "name": "JanuaryPoolUpdated"}, {"id": "693899cfa...d06adf9fe7a382", "gam3yaId": "69387c55183e9a9df6f3a6ab", "name": "JanuaryPoolUpdated"}], "userId": {"id": "69386d1a183e9a9df6f3a6a2", "username": "malak", "email": "malak@hotmail.com"}, "positionNumber": 2, "isForTrade": false, "createdAt": "2025-12-09T21:07:21.280Z", "updatedAt": "2025-12-09T22:03:41.355Z", "__v": 0}, {"id": "693899cfa...d06adf9fe7a382", "gam3yaId": "69387c55183e9a9df6f3a6ab", "name": "JanuaryPoolUpdated"}, {"userId": {"id": "69386d1a183e9a9df6f3a6a2", "username": "malak", "email": "malak@hotmail.com"}, "positionNumber": 2, "isForTrade": false, "createdAt": "2025-12-09T21:51:11.125Z", "updatedAt": "2025-12-09T22:03:41.414Z", "__v": 0}]}
```

The response status is 200 OK, time 473 ms, size 1.64 KB. The response body is identical to the input JSON.