

Georgia Sugisandhea_535230080

Bruno and MongoDB screenshots

1. Pagination and Filter

Ada 11 user dalam repository

The screenshot shows the MongoDB Compass interface connected to the 'demo-untar' database. The 'users' collection is selected, displaying 11 documents. Each document represents a user with fields like _id, name, email, password, attempt, and last_attempt. The interface includes a search bar, a toolbar with ADD DATA, EXPORT DATA, UPDATE, and DELETE buttons, and a results panel showing the document details.

Default url, menampilkan semua data dengan filter default descending

The screenshot shows the Postman application interface. A GET request is made to the URL `http://localhost:5000/api/users`. The response shows a paginated list of users with the following data:

```
1 "page_number": 1,
2 "page_size": 11,
3 "count": 11,
4 "total_pages": 1,
5 "has_previous_page": false,
6 "has_next_page": false,
7 "data": [
8     {
9         "id": "662a3f03659781cd71197ab9",
10        "name": "Administrator",
11        "email": "admin@example.com"
12    },
13    {
14        "id": "66376dc51c91362a95f2cdf1",
15        "name": "towelandsoap",
16        "email": "bathandbodyworks@gmail.com"
17    },
18    {
19        "id": "663636e76d2ef93470a6e2e9"
20    }
21 ]
```

url dengan custom page number dan size, filter email default descending, default search off

The screenshot shows the Bruno API testing tool interface. The left sidebar lists collections: backend, midterm, and polocash. Under midterm, there are three POST requests: login, getuser, and create user. The getuser request is selected. The right panel shows a GET request to `http://localhost:5000/api/users?page_number=2&page_size=4`. The query parameters are listed in a table:

Name	Value
page_number	2
page_size	4

The response pane displays the JSON data returned by the API:

```
1 "page_number": 2,
2 "page_size": 4,
3 "count": 4,
4 "total_pages": 3,
5 "has_previous_page": true,
6 "has_next_page": true,
7 "data": [
8   {
9     "id": "66376d5f1c91362a95f2cdf7",
10    "name": "Captain Man",
11    "email": "moms@gmail.com"
12  },
13  {
14    "id": "66376546c7371a801f6e391b",
15    "name": "Dorothy",
16    "email": "oc@gmail.com"
17  },
18  {
19    }
```

Url dengan search email dan page number dan size

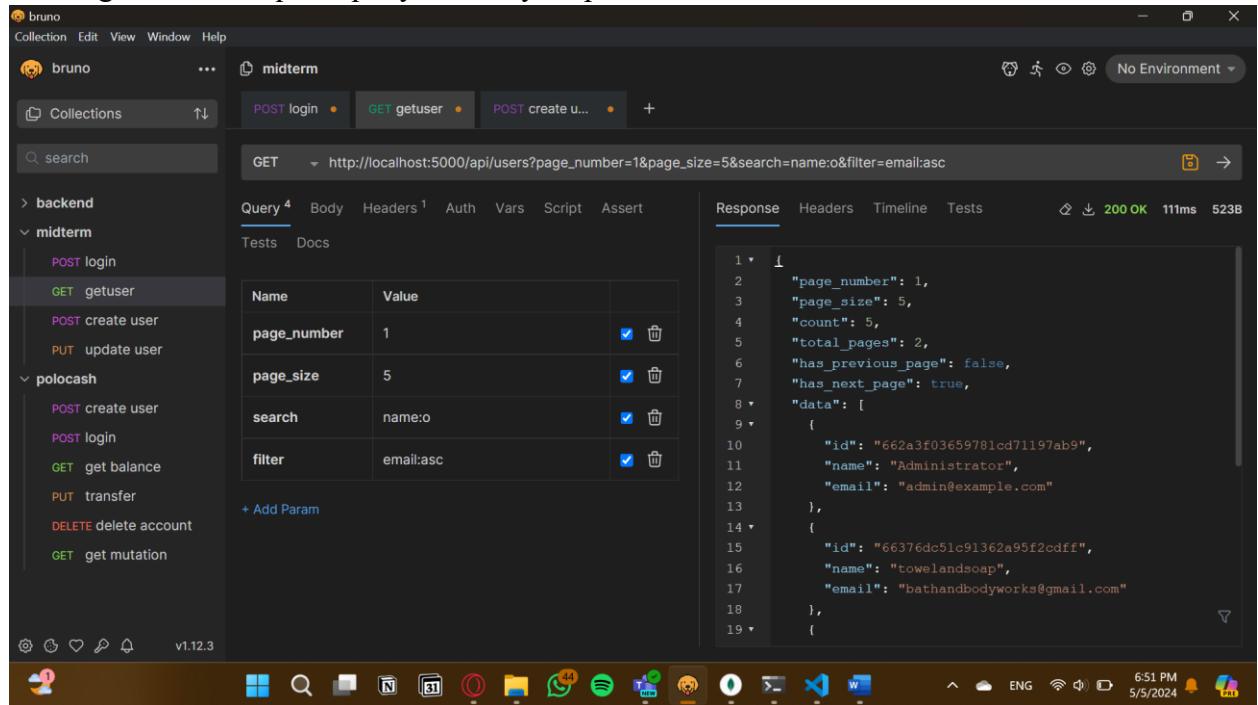
The screenshot shows the Bruno API testing tool interface. The left sidebar lists collections: backend, midterm, and polocash. Under midterm, there are three POST requests: login, getuser, and create user. The getuser request is selected. The right panel shows a GET request to `http://localhost:5000/api/users?page_number=1&page_size=5&search=email:admin`. The query parameters are listed in a table:

Name	Value
page_number	1
page_size	5
search	email:admin

The response pane displays the JSON data returned by the API:

```
1 "page_number": 1,
2 "page_size": 5,
3 "count": 1,
4 "total_pages": 1,
5 "has_previous_page": false,
6 "has_next_page": false,
7 "data": [
8   {
9     "id": "662a3f03659781cd7119ab9",
10    "name": "Administrator",
11    "email": "admin@example.com"
12  }
13 ]
14
15 ]
```

url dengan semua request query semuanya dipakai



The screenshot shows the Postman application interface. On the left, there's a sidebar with collections like 'backend' and 'midterm'. In the main area, a 'midterm' collection is selected. A 'GET getuser' request is highlighted. The 'Body' tab shows the following query parameters:

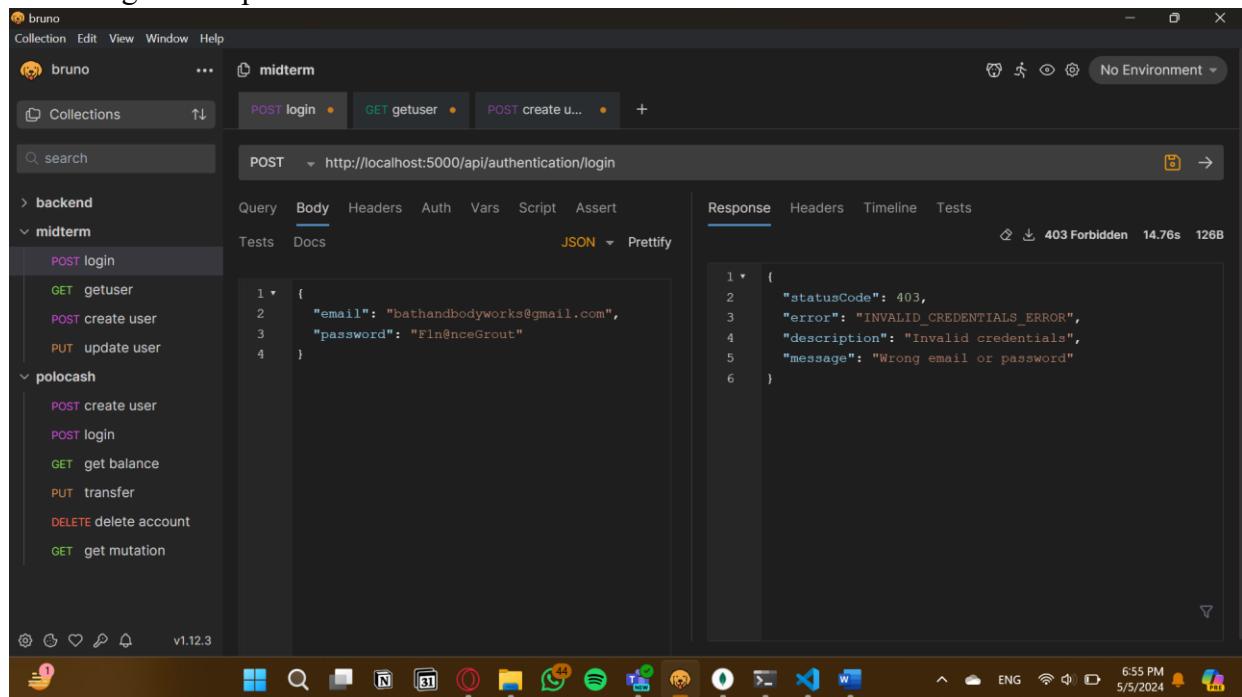
Name	Value
page_number	1
page_size	5
search	name:o
filter	email:asc

The 'Response' tab displays the JSON response:

```
1 "page_number": 1,
2 "page_size": 5,
3 "count": 5,
4 "total_pages": 2,
5 "has_previous_page": false,
6 "has_next_page": true,
7 "data": [
8   {
9     "id": "662a3f03659781cd71197ab5",
10    "name": "Administrator",
11    "email": "admin@example.com"
12  },
13  {
14    "id": "66376dc51c91362a95f2cdf1",
15    "name": "towlandsop",
16    "email": "bathandbodyworks@gmail.com"
17  },
18  {
19    }
```

2. Login Attempt

Failed login attempt 1



The screenshot shows the Postman application interface. A 'POST login' request is highlighted. The 'Body' tab shows the following JSON body:

```
1 {
2   "email": "pathandbodyworks@gmail.com",
3   "password": "Fln@nceGrout"
4 }
```

The 'Response' tab displays the JSON response:

```
1 {
2   "statusCode": 403,
3   "error": "INVALID_CREDENTIALS_ERROR",
4   "description": "Invalid credentials",
5   "message": "Wrong email or password"
6 }
```

Failed login attempt 2

The screenshot shows the Bruno API testing tool interface. The left sidebar contains collections: 'backend' and 'midterm'. Under 'midterm', there are several endpoints: POST login, GET getuser, POST create user, PUT update user, and others under 'polocash'. The main area shows a POST request to 'http://localhost:5000/api/authentication/login'. The 'Body' tab is selected, showing the JSON payload:

```
1  {
2    "email": "bathandbodyworks@gmail.com",
3    "password": "Fln@nceGroup"
4 }
```

The 'Response' tab shows the error response in JSON format:

```
1  {
2    "statusCode": 403,
3    "error": "INVALID_CREDENTIALS_ERROR",
4    "description": "Invalid credentials",
5    "message": "Wrong email or password"
6 }
```

The status bar at the bottom indicates v1.12.3 and the system tray shows various icons.

Success login attempt

The screenshot shows the Bruno API testing tool interface, identical to the previous one but with a successful login attempt. The 'Body' tab shows the same JSON payload as before:

```
1  {
2    "email": "bathandbodyworks@gmail.com",
3    "password": "Fln@nceGroup"
4 }
```

The 'Response' tab shows the success response in JSON format, which is very long and truncated at the end:

```
1  [
2    {
3      "email": "bathandbodyworks@gmail.com",
4      "name": "towelandscap",
5      "user_id": "66376dc51c91362a95f2cdff",
6      "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJl
bWFpbCI6ImJhdGhhbmRib2R5d29ya3NAZ21haWwuY29tLiwidXNlc
klkIjoiNjYzNzZkYzUxYzkxMzYyYTklZjJzGZmIiwiwaWF0IjoxNz
E0OTEwMTk1LC1eHA10)E3MTQ5OTY1OTV9.bf_oNuS_5dIfxiFwoN
E7u0phznNZSP2Hbw72pZYrWY"
7    }
8  ]
```

The status bar at the bottom indicates v1.12.3 and the system tray shows various icons.

Failed login attempt 1

The screenshot shows the Postman application interface. The left sidebar has a collection named "bruno" expanded, showing categories "backend", "midterm", and "polocash". Under "midterm", there is a "POST login" request selected. The "Body" tab is active, showing a JSON payload:

```
1  {
2    "email": "bathandbodyworks@gmail.com",
3    "password": "Fln@nceGrout"
4 }
```

The "Response" tab shows the API response in JSON format:

```
1  {
2    "statusCode": 403,
3    "error": "INVALID_CREDENTIALS_ERROR",
4    "description": "Invalid credentials",
5    "message": "Wrong email or password"
6 }
```

The status bar at the bottom indicates "v1.12.3". The taskbar at the bottom right shows various icons and the date/time "5/5/2024 6:59 PM".

Failed login attempt 2

The screenshot shows the Postman application interface, identical to the first one but with a later timestamp. The left sidebar has a collection named "bruno" expanded, showing categories "backend", "midterm", and "polocash". Under "midterm", there is a "POST login" request selected. The "Body" tab is active, showing the same JSON payload as before:

```
1  {
2    "email": "bathandbodyworks@gmail.com",
3    "password": "Fln@nceGrout"
4 }
```

The "Response" tab shows the API response in JSON format:

```
1  {
2    "statusCode": 403,
3    "error": "INVALID_CREDENTIALS_ERROR",
4    "description": "Invalid credentials",
5    "message": "Wrong email or password"
6 }
```

The status bar at the bottom indicates "v1.12.3". The taskbar at the bottom right shows various icons and the date/time "5/5/2024 7:00 PM".

Failed login attempt 3

The screenshot shows the Postman application interface. The left sidebar has a tree view with collections: 'backend' (expanded), 'midterm' (selected), and 'polocash'. Under 'midterm', there are several requests: 'POST login' (selected), 'GET getuser', 'POST create user', 'PUT update user', 'POST create user', 'POST login', 'GET get balance', 'PUT transfer', 'DELETE delete account', and 'GET get mutation'. The main area shows a POST request to 'http://localhost:5000/api/authentication/login'. The 'Body' tab is selected, showing a JSON payload:

```
1 ▾ {  
2   "email": "bathandbodyworks@gmail.com",  
3   "password": "Fln@nceGrout"  
4 }
```

The 'Response' tab shows the following JSON response:

```
1 ▾ {  
2   "statusCode": 403,  
3   "error": "INVALID_CREDENTIALS_ERROR",  
4   "description": "Invalid credentials",  
5   "message": "Wrong email or password"  
6 }
```

The status bar at the bottom indicates v1.12.3.

Failed login attempt 4

The screenshot shows the Postman application interface, identical to the previous one but with a later timestamp. The left sidebar shows the same tree structure with 'midterm' selected. The main area shows the same POST request to 'http://localhost:5000/api/authentication/login' with the same JSON payload. The 'Response' tab shows the same JSON response as before:

```
1 ▾ {  
2   "statusCode": 403,  
3   "error": "INVALID_CREDENTIALS_ERROR",  
4   "description": "Invalid credentials",  
5   "message": "Wrong email or password"  
6 }
```

The status bar at the bottom indicates v1.12.3.

Failed login attempt 5

The screenshot shows the Postman application interface. The left sidebar has a collection named "bruno" expanded, showing categories "backend", "midterm", and "polocash". Under "midterm", there is a "POST login" request selected. The "Body" tab shows a JSON payload:

```
1  {
2     "email": "bathandbodyworks@gmail.com",
3     "password": "Fln@nceGrout"
4 }
```

The "Response" tab displays the API response:

```
1  {
2     "statusCode": 403,
3     "error": "INVALID_CREDENTIALS_ERROR",
4     "description": "Invalid credentials",
5     "message": "Wrong email or password, limit reached"
6 }
```

The status bar at the bottom indicates v1.12.3.

Blocked attempt

The screenshot shows the Postman application interface. The left sidebar has a collection named "bruno" expanded, showing categories "backend", "midterm", and "polocash". Under "midterm", there is a "POST login" request selected. The "Body" tab shows a JSON payload:

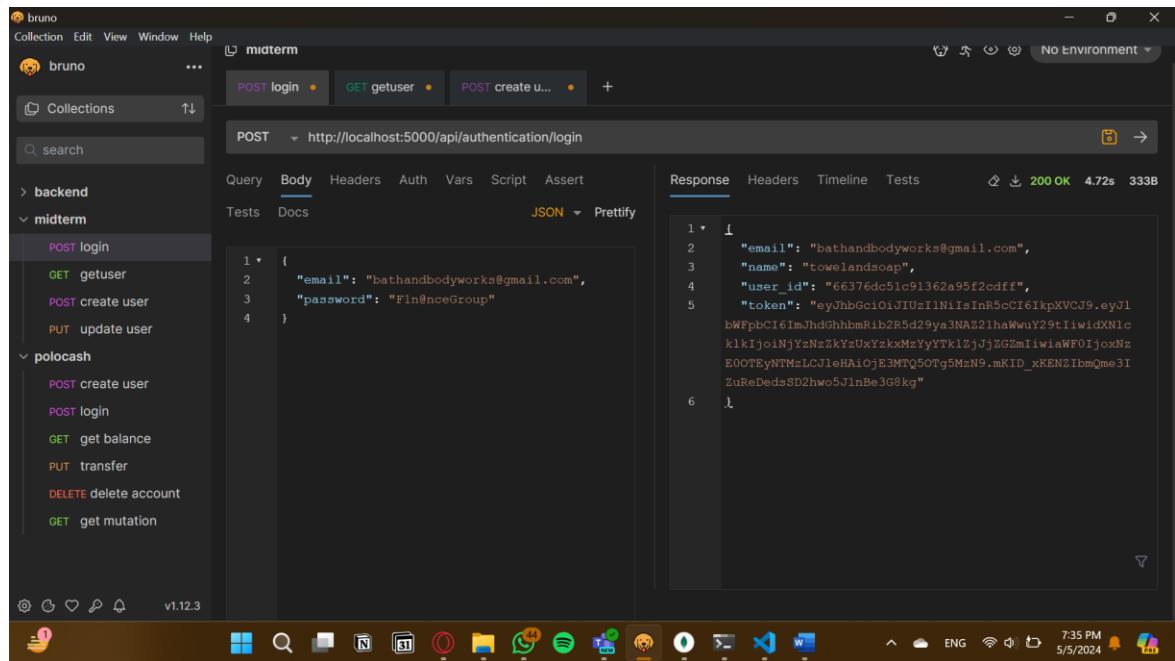
```
1  {
2     "email": "bathandbodyworks@gmail.com",
3     "password": "Fln@nceGrout"
4 }
```

The "Response" tab displays the API response:

```
1  {
2     "statusCode": 403,
3     "error": "FORBIDDEN_ERROR",
4     "description": "Access forbidden",
5     "message": "Too many attempts"
6 }
```

The status bar at the bottom indicates v1.12.3.

After >30 minutes



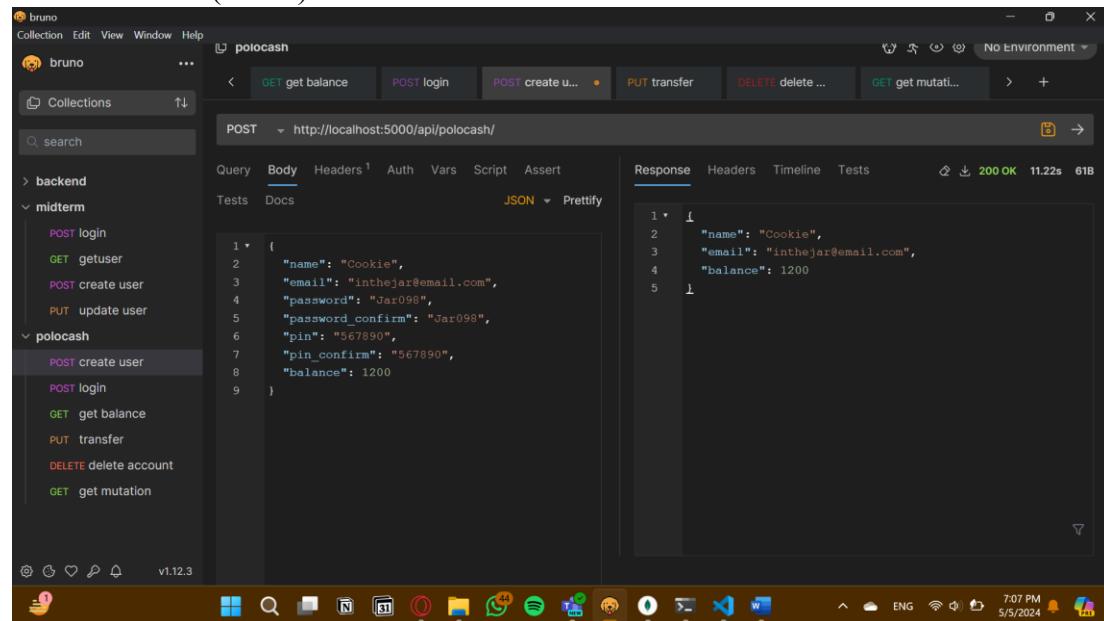
The screenshot shows the Postman application interface. The left sidebar lists collections: 'backend' and 'midterm'. Under 'midterm', there are several requests: 'POST login', 'GET getuser', 'POST create user', and 'PUT update user'. The 'polocash' collection is also visible with requests like 'POST create user', 'POST login', 'GET get balance', 'PUT transfer', 'DELETE delete account', and 'GET get mutation'. The main workspace shows a POST request to 'http://localhost:5000/api/authentication/login'. The 'Body' tab contains the following JSON payload:

```
1 {
2   "email": "bathandbodyworks@gmail.com",
3   "name": "towelandsop",
4   "user_id": "66376dc51c91362a95f2cdff",
5   "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJl
bWFpbCI6ImJhdGhhbmRib2R5d29ya3NAZ21haWwuY29tIiwidXNlc
klikjoiNjYzNzKyzUxYzkkMzTyYTkIzJjZGZmliviaWF0IjoxNz
E0OTExNTMzLC1leHA1OjE3MTQ50Tg5MzN9.mKID_xKENZIbmQme61B
ZuReDedsSD2hwo5JlnBe3G8kg"
```

The 'Response' tab shows a successful 200 OK response with a content length of 333B. The response body is identical to the payload above. The bottom status bar indicates v1.12.3.

3. M-Banking called polocash

a. Create account (create)



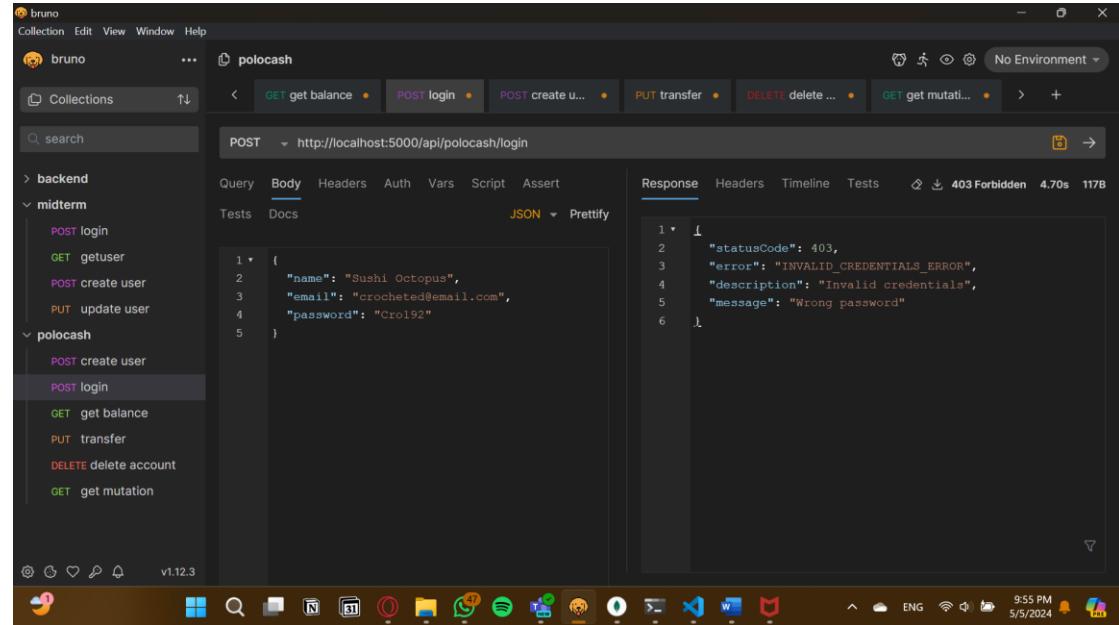
The screenshot shows the Postman application interface. The left sidebar lists collections: 'backend' and 'midterm'. Under 'midterm', there are several requests: 'POST login', 'GET getuser', 'POST create user', and 'PUT update user'. The 'polocash' collection is also visible with requests like 'POST create user', 'POST login', 'GET get balance', 'PUT transfer', 'DELETE delete account', and 'GET get mutation'. The main workspace shows a POST request to 'http://localhost:5000/api/polocash/'. The 'Body' tab contains the following JSON payload:

```
1 {
2   "name": "Cookie",
3   "email": "inthejar@email.com",
4   "password": "Jar098",
5   "password_confirm": "Jar098",
6   "pin": "567890",
7   "pin_confirm": "567890",
8   "balance": 1200
9 }
```

The 'Response' tab shows a successful 200 OK response with a content length of 61B. The response body is identical to the payload above. The bottom status bar indicates v1.12.3.

b. Login

Failed attempt 1



The screenshot shows the Postman application interface. On the left, the sidebar displays collections: 'backend', 'midterm', and 'polocash'. Under 'polocash', there are several requests: 'POST create user', 'POST login' (which is selected), 'GET get balance', 'PUT transfer', 'DELETE delete account', and 'GET get mutation'. The main panel shows a POST request to 'http://localhost:5000/api/polocash/login'. The 'Body' tab contains the following JSON payload:

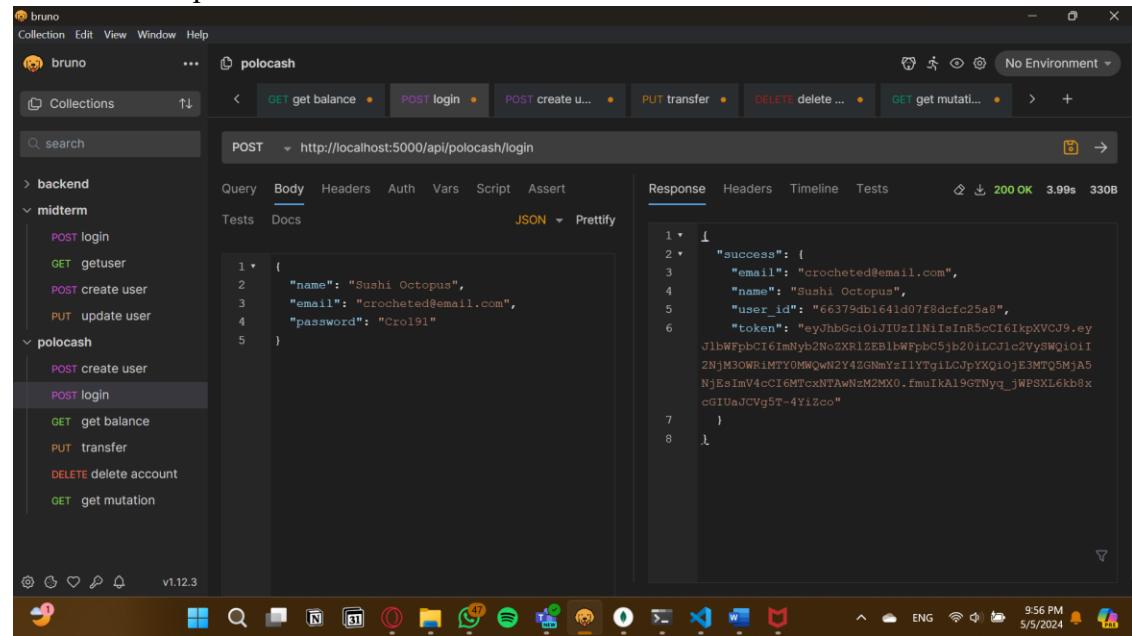
```
1 * {
2   "name": "Sushi Octopus",
3   "email": "crocheted@email.com",
4   "password": "Crol92"
5 }
```

The 'Response' tab shows the following JSON response:

```
1 * {
2   "statusCode": 403,
3   "error": "INVALID_CREDENTIALS_ERROR",
4   "description": "Invalid credentials",
5   "message": "Wrong password"
6 }
```

The status bar at the bottom indicates 'v1.12.3'.

Success attempt



The screenshot shows the Postman application interface, identical to the previous one but with a successful login attempt. The 'Body' tab now contains a different JSON payload:

```
1 * {
2   "name": "Sushi Octopus",
3   "email": "crocheted@email.com",
4   "password": "Crol91"
5 }
```

The 'Response' tab shows the following JSON response, indicating success:

```
1 * {
2   "success": {
3     "email": "crocheted@email.com",
4     "name": "Sushi Octopus",
5     "user_id": "66379db1641d07f8dcfc25a8",
6     "token": "eyJhbGciOiJIUzI1NiIsInRcIjE1kpgXVCJ9eyJlbWFpbCI6ImNyb2NoZXR1ZEBlbWFpbC5jb20iLCJlc2VySWQioIIZNjM3OWR1MTY0MWQwN2Y4ZGNmYzI1YTgjLCLUpYXQ1oJE3MTQ5MjA5NjEsImV4cI6MTcxNTAwNzM2MX0.fmuIkAl9GTNyq_jWPSXL6kb8xcGIUuaJCVg5T-4Yi2co"
7   }
8 }
```

The status bar at the bottom indicates 'v1.12.3'.

Failed login attempt 1

The screenshot shows the Bruno API testing interface. On the left, the sidebar lists collections: backend, midterm, and polocash. Under polocash, there are several endpoints: POST create user, POST login, GET get user, POST create user, PUT update user, and others. The main panel shows a POST request to `http://localhost:5000/api/polocash/login`. The Body tab contains the following JSON payload:

```
1  {
2   "name": "Scarlett",
3   "email": "milkandhoney@email.com",
4   "password": "Wine26"
5 }
```

The Response tab shows the following JSON error response:

```
1  {
2   "statusCode": 403,
3   "error": "INVALID_CREDENTIALS_ERROR",
4   "description": "Invalid credentials",
5   "message": "Wrong password"
6 }
```

Failed login attempt 2

This screenshot is identical to the first one, showing a failed login attempt. The sidebar, request details, and error response are all the same.

Failed login attempt 3

The screenshot shows the Bruno API client interface. On the left, the sidebar lists collections: backend, midterm, and polocash. Under polocash, the POST login endpoint is selected. The main area shows a POST request to `http://localhost:5000/api/polocash/login`. The Body tab contains the following JSON payload:

```
1  {
2    "name": "Scarlett",
3    "email": "milkandhoney@email.com",
4    "password": "Wine26"
5 }
```

The Response tab displays the following JSON response:

```
1  {
2    "statusCode": 403,
3    "error": "INVALID_CREDENTIALS_ERROR",
4    "description": "Invalid credentials",
5    "message": "Wrong password"
6 }
```

The status bar at the bottom indicates v1.12.3.

Failed login attempt 4

The screenshot shows the Bruno API client interface. On the left, the sidebar lists collections: backend, midterm, and polocash. Under polocash, the POST login endpoint is selected. The main area shows a POST request to `http://localhost:5000/api/polocash/login`. The Body tab contains the following JSON payload:

```
1  {
2    "name": "Scarlett",
3    "email": "milkandhoney@email.com",
4    "password": "Wine26"
5 }
```

The Response tab displays the following JSON response:

```
1  {
2    "statusCode": 403,
3    "error": "INVALID_CREDENTIALS_ERROR",
4    "description": "Invalid credentials",
5    "message": "Wrong password"
6 }
```

The status bar at the bottom indicates v1.12.3.

Failed login attempt 5

The screenshot shows the Bruno API testing tool interface. The left sidebar contains a tree view of API endpoints under 'backend' and 'midterm'. The 'polocash' endpoint is selected. The top navigation bar shows the URL as `http://localhost:5000/api/polocash/login`. The main area displays a POST request with the following JSON body:

```
1  {
2     "name": "Scarlett",
3     "email": "milkandhoney@email.com",
4     "password": "Wine26"
5 }
```

The response tab shows the following JSON response:

```
1  {
2     "statusCode": 403,
3     "error": "INVALID_CREDENTIALS_ERROR",
4     "description": "Invalid credentials",
5     "message": "Wrong password, too much attempt, your account is blocked, please go to nearest bank"
6 }
```

The status code is 403 Forbidden, and the message indicates that the account is blocked due to too many failed attempts.

Blocked

The screenshot shows the Bruno API testing tool interface, identical to the previous one but with a different response message. The body of the POST request remains the same:

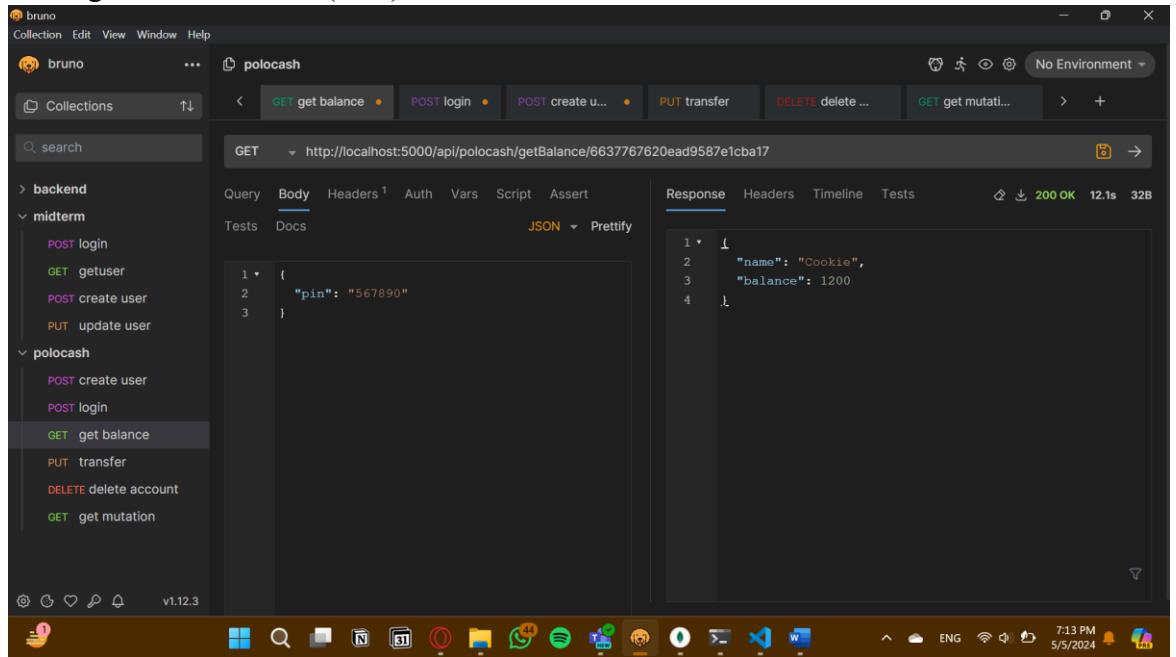
```
1  {
2     "name": "Scarlett",
3     "email": "milkandhoney@email.com",
4     "password": "Wine26"
5 }
```

The response tab shows the following JSON response:

```
1  {
2     "statusCode": 403,
3     "error": "FORBIDDEN_ERROR",
4     "description": "Access forbidden",
5     "message": "too much attempt, your account is blocked, please go to nearest bank"
6 }
```

The status code is 403 Forbidden, and the message indicates that the account is blocked due to too many failed attempts.

c. Getting account balance (read)



The screenshot shows the Postman application interface. On the left, the sidebar displays collections: 'backend' and 'midterm'. Under 'midterm', there are several endpoints: 'POST login', 'GET getuser', 'POST create user', 'PUT update user', 'polocash' (which is expanded), 'POST create user', 'POST login', 'GET get balance' (which is selected), 'PUT transfer', 'DELETE delete account', and 'GET get mutation'. The main panel shows a GET request to 'http://localhost:5000/api/polocash/getBalance/6637767620ead9587e1cba17'. The 'Body' tab is selected, showing a JSON payload:

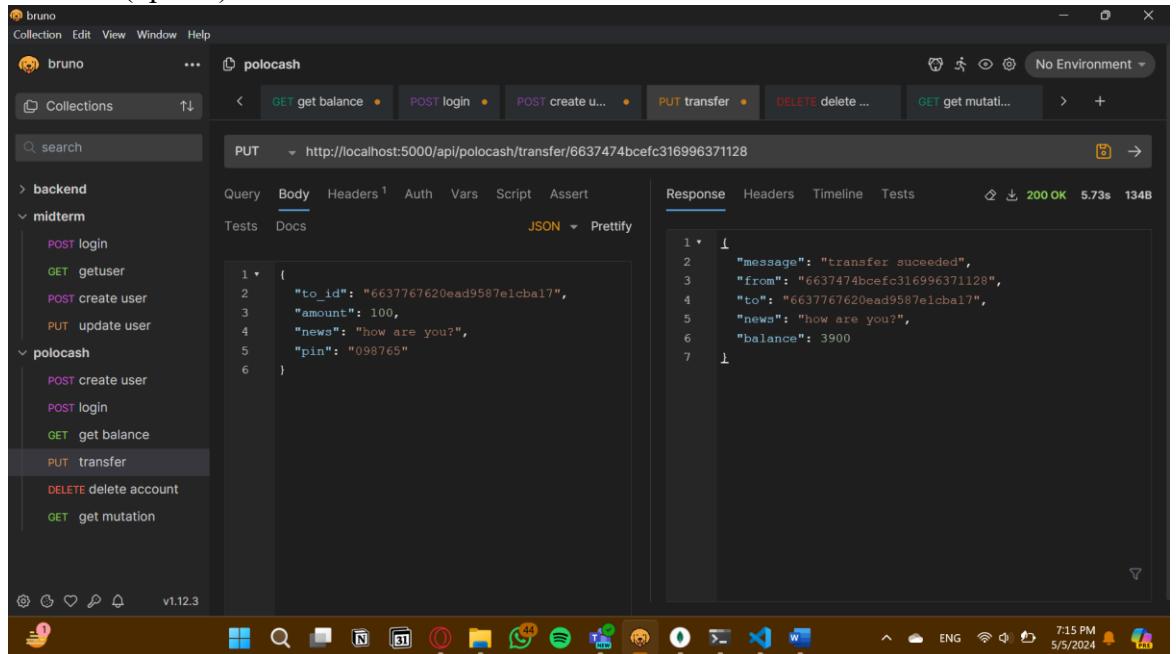
```
1 {  
2   "pin": "567890"  
3 }
```

The 'Response' tab shows the JSON response:

```
1 {  
2   "name": "Cookie",  
3   "balance": 1200  
4 }
```

The status bar at the bottom indicates v1.12.3.

d. Transfer (update)



The screenshot shows the Postman application interface. The sidebar and request URL are identical to the previous screenshot. The 'Body' tab is selected, showing a JSON payload for a transfer:

```
1 {  
2   "to_id": "6637767620ead9587e1cba17",  
3   "amount": 100,  
4   "news": "how are you?",  
5   "pin": "098765"  
6 }
```

The 'Response' tab shows the JSON response:

```
1 {  
2   "message": "transfer suceeded",  
3   "from": "6637767620ead9587e1cba17",  
4   "to": "6637767620ead9587e1cba17",  
5   "news": "how are you?",  
6   "news": "how are you?",  
7   "balance": 3900  
8 }
```

The status bar at the bottom indicates v1.12.3.

After transfer (before, refer to getting account balance picture)

The screenshot shows the Bruno API client interface. On the left, the sidebar lists collections: backend, midterm, and polocash. Under polocash, the GET get balance endpoint is selected. The main panel shows a GET request to `http://localhost:5000/api/polocash/getBalance/6637767620ead9587e1cba17`. The response is a JSON object:

```
1  {
2     "name": "Cookie",
3     "balance": 1300
4 }
```

If balance not enough

The screenshot shows the Bruno API client interface. The sidebar lists collections: backend, midterm, and polocash. Under polocash, the PUT transfer endpoint is selected. The main panel shows a PUT request to `http://localhost:5000/api/polocash/transfer/66377474bcfc316996371128`. The response is a JSON error object:

```
1  {
2     "statusCode": 422,
3     "error": "UNPROCESSABLE_ENTITY_ERROR",
4     "description": "Unprocessable entity",
5     "message": "Not enough balance"
6 }
```

e. Delete account (admin only) (delete)

The screenshot shows the Bruno API client interface. The left sidebar lists collections: 'backend', 'midterm', and 'polocash'. Under 'polocash', the 'DELETE delete account' endpoint is selected. The top bar shows the URL: `DELETE http://localhost:5000/api/polocash/deleteAcc/663730bctfabcdb5dff1f71`. The response tab shows a 200 OK status with the message:

```
1 "msg": "Account deleted"
```

.

If not admin

The screenshot shows the Bruno API client interface. The left sidebar lists collections: 'backend', 'midterm', and 'polocash'. Under 'polocash', the 'DELETE delete account' endpoint is selected. The top bar shows the URL: `DELETE http://localhost:5000/api/polocash/deleteAcc/6637767620ead9587e1cba17`. The response tab shows a 422 Unprocessable Entity status with the message:

```
1 "statusCode": 422,
2 "error": "UNPROCESSABLE_ENTITY_ERROR",
3 "description": "Unprocessable entity",
4 "message": "Sorry, only administrator can delete accounts"
5
6
```

.

f. Getting account mutation (read)

bruno

Collection Edit View Window Help

bruno ... polocash No Environment

Collections GET http://localhost:5000/api/polocash/getMutation/6637287b78475faae1d5103d

Query Body Headers¹ Auth Vars Script Assert Tests Docs Response Headers Timeline Tests 200 OK 5.98s 212B

JSON Prettify

```

mutation: [
  {
    "date": "2024-05-05T09:34:29.658Z",
    "from": "Pomegranate",
    "to": "Sudsy",
    "amount": 1000,
    "news": ""
  },
  {
    "date": "2024-05-05T09:35:56.988Z",
    "from": "Pomegranate",
    "to": "Sudsy",
    "amount": 1000,
    "news": "commission"
  }
]

```

v1.12.3

*Anything that includes putting in pin and password has only 5 failed attempts, not only login, then it's blocked. And for security reasons, account is blocked and not able to retry even after 30 minutes.

Failed attempt 1

bruno

Collection Edit View Window Help

bruno ... polocash No Environment

Collections GET http://localhost:5000/api/polocash/getBalance/6637474bcfc316996371128

Query Body Headers¹ Auth Vars Script Assert Tests Docs Response Headers Timeline Tests 403 Forbidden 4.87s 112B

JSON Prettify

```

{
  "statusCode": 403,
  "error": "INVALID_CREDENTIALS_ERROR",
  "description": "Invalid credentials",
  "message": "Wrong pin"
}

```

v1.12.3

Failed attempt 2

The screenshot shows the Bruno API testing tool interface. The left sidebar lists collections: backend, midterm, and polocash. Under polocash, there are several endpoints: POST login, GET getuser, POST create user, PUT update user, POST create user, POST login, GET get balance, PUT transfer, DELETE delete account, and GET get mutation. The main panel shows a PUT request to `http://localhost:5000/api/polocash/transfer/6637474bcefc316996371128`. The Body tab contains the following JSON payload:

```
1 ▶  {
2     "to_id": "663747eccefc316996371132",
3     "amount": 500,
4     "news": "please last chance",
5     "pin": "098764"
6 }
```

The Response tab shows the following JSON error response:

```
1 ▶  {
2     "statusCode": 403,
3     "error": "INVALID_CREDENTIALS_ERROR",
4     "description": "Invalid credentials",
5     "message": "Wrong pin"
6 }
```

The status bar at the bottom indicates the response was a 403 Forbidden error with a duration of 4.73s and 112B.

Failed attempt 3

The screenshot shows the Bruno API testing tool interface. The left sidebar lists collections: backend, midterm, and polocash. Under polocash, there are several endpoints: POST login, GET getuser, POST create user, PUT update user, POST create user, POST login, GET get balance, PUT transfer, DELETE delete account, and GET get mutation. The main panel shows a GET request to `http://localhost:5000/api/polocash/getMutation/6637474bcefc316996371128`. The Body tab contains the following JSON payload:

```
1 ▶  {
2     "pin": "098763"
3 }
```

The Response tab shows the following JSON error response:

```
1 ▶  {
2     "statusCode": 422,
3     "error": "UNPROCESSABLE_ENTITY_ERROR",
4     "description": "Unprocessable entity",
5     "message": "Wrong pin"
6 }
```

The status bar at the bottom indicates the response was a 422 Unprocessable Entity error with a duration of 4.9s and 114B.

Failed attempt 4

The screenshot shows the Bruno API testing tool interface. On the left, the sidebar lists collections: backend, midterm, and polocash. Under polocash, the GET method for 'get balance' is selected. The main panel shows a GET request to `http://localhost:5000/api/polocash/getBalance/6637474bccef316996371128`. The response is a 403 Forbidden error with the following JSON body:

```
1 ▶  {
2     "statusCode": 403,
3     "error": "INVALID_CREDENTIALS_ERROR",
4     "description": "Invalid credentials",
5     "message": "Wrong pin"
6 }
```

Failed attempt 5

The screenshot shows the Bruno API testing tool interface. On the left, the sidebar lists collections: backend, midterm, and polocash. Under polocash, the PUT method for 'transfer' is selected. The main panel shows a PUT request to `http://localhost:5000/api/polocash/transfer/6637474bccef316996371128`. The response is a 403 Forbidden error with the following JSON body:

```
1 ▶  {
2     "statusCode": 403,
3     "error": "INVALID_CREDENTIALS_ERROR",
4     "description": "Invalid credentials",
5     "message": "Wrong pin, too much attempt, your account is blocked, please go to nearest bank"
6 }
```

Blocked

The screenshot shows the Bruno API testing tool interface. The left sidebar lists collections: backend, midterm, and polocash. Under polocash, there are several endpoints: POST create user, POST login, GET get balance, PUT transfer, DELETE delete account, and GET get mutation. The main area shows a PUT request to `http://localhost:5000/api/polocash/transfer/6637474bcefc316996371128`. The Body tab contains the following JSON payload:

```
1  {
2    "to_id": "6637474bcefc316996371132",
3    "amount": 100,
4    "pin": "098765"
5 }
```

The Response tab shows the following JSON error response:

```
1  {
2    "statusCode": 403,
3    "error": "FORBIDDEN_ERROR",
4    "description": "Access forbidden",
5    "message": "too much attempt, your account is blocked, please go to nearest bank"
6 }
```

The status bar at the bottom indicates v1.12.3 and the system tray shows the date and time as 9:48 PM 5/5/2024.