

Friend Management

Table of Contents

<u>I.</u>	<u>PROJECT STRUCTURE</u>	<u>1</u>
<u>II.</u>	<u>WORKFLOW</u>	<u>3</u>
<u>III.</u>	<u>TECHNOLOGIES.....</u>	<u>3</u>
<u>IV.</u>	<u>SETUP APPLICATION</u>	<u>3</u>
<u>V.</u>	<u>RUNNING</u>	<u>3</u>
<u>VI.</u>	<u>LIST APIS</u>	<u>8</u>

I. Project structure

▼ S3_FriendManagementAPI_N... ●

- > .git
- > config
- > db/migrations
- > docs
- ▼ internal
 - > controllers
 - > models

> repository

> vendor ●

🔗 .DS_Store

≡ .env.dev

🔗 .gitignore

🚢 docker-compose.yaml

🚢 Dockerfile

≡ go.mod 9+

≡ go.sum

🚢 main.go

M Makefile

📘 README.md

⚙️ sqlboiler.toml

II. Workflow

Request => Controller => Repository => model => Database

- Controller: Get request from http request, decode, validate, call controller, write http response.
- Controller: Handler business logic, call repositories.
- Model: Data access layer.

III. Technologies

- Golang: go module, go-chi, sqlboiler, db-migration, mockery
- Postgres
- Docker
- Makefile

IV. Setup application

- Make sure that your Docker is up and running
- From folder `S3_FriendManagementAPI_NhutTo/`, run command `make setup`

V. Running

- Run app:
Start the server with command `make run`
Server running on: `https://localhost:8080`
- Run test:
Run unit test with command `make test`
Unit test results:

```
? github.com/ToTranMinhNhut/S3_FriendManagementAPI_NhutTo [no test files]
? github.com/ToTranMinhNhut/S3_FriendManagementAPI_NhutTo/config [no test files]
=== RUN    TestControllers_GetFriends
=== RUN    TestControllers_GetFriends/success_with_an_input
=== RUN
TestControllers_GetFriends/failed_with_an_unknow_format_input
--- PASS: TestControllers_GetFriends (0.00s)
    --- PASS: TestControllers_GetFriends/success_with_an_input (0.00s)
    --- PASS:
TestControllers_GetFriends/failed_with_an_unknow_format_input (0.00s)
=== RUN    TestControllers_CreateFriends
=== RUN    TestControllers_CreateFriends/success_with_an_input
=== RUN
TestControllers_CreateFriends/failed_with_an_unknow_format_input
--- PASS: TestControllers_CreateFriends (0.00s)
    --- PASS: TestControllers_CreateFriends/success_with_an_input (0.00s)
    --- PASS:
TestControllers_CreateFriends/failed_with_an_unknow_format_input (0.00s)
=== RUN    TestControllers_GetCommonFriends
```

```
=== RUN    TestControllers_GetCommonFriends/success_with_an_input
=== RUN
TestControllers_GetCommonFriends/failed_with_an_unknow_format_input
--- PASS: TestControllers_GetCommonFriends (0.00s)
    --- PASS: TestControllers_GetCommonFriends/success_with_an_input
(0.00s)
    --- PASS:
TestControllers_GetCommonFriends/failed_with_an_unknow_format_input
(0.00s)
=== RUN    TestControllers_CreateSubscription
=== RUN    TestControllers_CreateSubscription/success_with_an_input
=== RUN
TestControllers_CreateSubscription/failed_with_an_unknow_format_input
--- PASS: TestControllers_CreateSubscription (0.00s)
    --- PASS: TestControllers_CreateSubscription/success_with_an_input
(0.00s)
    --- PASS:
TestControllers_CreateSubscription/failed_with_an_unknow_format_input
(0.00s)
=== RUN    TestControllers_CreateUserBlocks
=== RUN    TestControllers_CreateUserBlocks/success_with_an_input
=== RUN
TestControllers_CreateUserBlocks/failed_with_an_unknow_format_input
--- PASS: TestControllers_CreateUserBlocks (0.00s)
    --- PASS: TestControllers_CreateUserBlocks/success_with_an_input
(0.00s)
    --- PASS:
TestControllers_CreateUserBlocks/failed_with_an_unknow_format_input
(0.00s)
=== RUN    TestControllers_GetRecipientEmails
=== RUN    TestControllers_GetRecipientEmails/success_with_an_input
=== RUN
TestControllers_GetRecipientEmails/failed_with_an_unknow_format_input
--- PASS: TestControllers_GetRecipientEmails (0.00s)
    --- PASS: TestControllers_GetRecipientEmails/success_with_an_input
(0.00s)
    --- PASS:
TestControllers_GetRecipientEmails/failed_with_an_unknow_format_input
(0.00s)
=== RUN    TestControllers_GetUsers
=== RUN    TestControllers_GetUsers/success_with_an_empty_input
=== RUN    TestControllers_GetUsers/failed_with_an_unknow_format_input
--- PASS: TestControllers_GetUsers (0.00s)
    --- PASS: TestControllers_GetUsers/success_with_an_empty_input
(0.00s)
    --- PASS:
TestControllers_GetUsers/failed_with_an_unknow_format_input (0.00s)
PASS
```

ok

github.com/ToTranMinhNhut/S3_FriendManagementAPI_NhutTo/internal/c
ontrollers 0.012s
?

github.com/ToTranMinhNhut/S3_FriendManagementAPI_NhutTo/internal/m
odels [no test files]

=== RUN TestRepository_CreateFriend

=== RUN

TestRepository_CreateFriend/success_with_adding_input_of_userIds

=== RUN

TestRepository_CreateFriend/query_by_an_unknown_input_userIds

--- PASS: TestRepository_CreateFriend (0.10s)

--- PASS:

TestRepository_CreateFriend/success_with_adding_input_of_userIds
(0.06s)

--- PASS:

TestRepository_CreateFriend/query_by_an_unknown_input_userIds (0.05s)

=== RUN TestRepository_IsExistedFriend

=== RUN

TestRepository_IsExistedFriend/success_with_adding_input_of_userIds

=== RUN

TestRepository_IsExistedFriend/query_by_an_unknown_input_userIds

--- PASS: TestRepository_IsExistedFriend (0.09s)

--- PASS:

TestRepository_IsExistedFriend/success_with_adding_input_of_userIds
(0.05s)

--- PASS:

TestRepository_IsExistedFriend/query_by_an_unknown_input_userIds
(0.04s)

=== RUN TestRepository_IsBlockedUser

=== RUN

TestRepository_IsBlockedUser/success_with_adding_input_of_userIds

=== RUN

TestRepository_IsBlockedUser/query_by_an_unknown_input_userIds

--- PASS: TestRepository_IsBlockedUser (0.09s)

--- PASS:

TestRepository_IsBlockedUser/success_with_adding_input_of_userIds
(0.05s)

--- PASS:

TestRepository_IsBlockedUser/query_by_an_unknown_input_userIds (0.05s)

=== RUN TestRepository_GetFriendsByID

=== RUN

TestRepository_GetFriendsByID/query_by_an_unknown_input_userId

=== RUN

TestRepository_GetFriendsByID/success_with_adding_input_of_userId

--- PASS: TestRepository_GetFriendsByID (0.09s)

--- PASS:

TestRepository_GetFriendsByID/query_by_an_unknown_input_userId (0.04s)

```
    --- PASS:
TestRepository_GetFriendsByID/success_with_adding_input_of_userId
(0.04s)
=== RUN    TestRepository_GetUserBlocksByID
=== RUN
TestRepository_GetUserBlocksByID/query_by_an_unknown_input_userId
=== RUN
TestRepository_GetUserBlocksByID/success_with_adding_input_of_userId
--- PASS: TestRepository_GetUserBlocksByID (0.07s)
    --- PASS:
TestRepository_GetUserBlocksByID/query_by_an_unknown_input_userId
(0.04s)
    --- PASS:
TestRepository_GetUserBlocksByID/success_with_adding_input_of_userId
(0.04s)
=== RUN    TestRepository_CreateSubscription
=== RUN
TestRepository_CreateSubscription/success_with_adding_input_of_userIds
=== RUN
TestRepository_CreateSubscription/query_by_an_unknown_input_userIds
--- PASS: TestRepository_CreateSubscription (0.08s)
    --- PASS:
TestRepository_CreateSubscription/success_with_adding_input_of_userIds
(0.04s)
    --- PASS:
TestRepository_CreateSubscription/query_by_an_unknown_input_userIds
(0.04s)
=== RUN    TestRepository_GetRecipientEmails
=== RUN
TestRepository_GetRecipientEmails/query_by_an_unknown_input_userId
=== RUN
TestRepository_GetRecipientEmails/success_with_adding_input_of_userId
--- PASS: TestRepository_GetRecipientEmails (0.08s)
    --- PASS:
TestRepository_GetRecipientEmails/query_by_an_unknown_input_userId
(0.04s)
    --- PASS:
TestRepository_GetRecipientEmails/success_with_adding_input_of_userId
(0.04s)
=== RUN    TestRepository_CreateUserBlock
=== RUN
TestRepository_CreateUserBlock/query_by_an_unknown_input_userIds
=== RUN
TestRepository_CreateUserBlock/success_with_adding_input_of_userIds
--- PASS: TestRepository_CreateUserBlock (0.08s)
    --- PASS:
TestRepository_CreateUserBlock/query_by_an_unknown_input_userIds
(0.04s)
```

```

    --- PASS:
TestRepository_CreateUserBlock/success_with_adding_input_of_userIds
(0.04s)
=== RUN    TestRepository_IsSubscribedFriend
=== RUN
TestRepository_IsSubscribedFriend/success_with_adding_input_of_userIds
=== RUN
TestRepository_IsSubscribedFriend/query_by_an_unknown_input_userIds
--- PASS: TestRepository_IsSubscribedFriend (0.08s)
    --- PASS:
TestRepository_IsSubscribedFriend/success_with_adding_input_of_userIds
(0.04s)
    --- PASS:
TestRepository_IsSubscribedFriend/query_by_an_unknown_input_userIds
(0.04s)
=== RUN    TestRepository_GetUserIDByEmail
=== RUN
TestRepository_GetUserIDByEmail/query_by_an_unknown_input_email
=== RUN
TestRepository_GetUserIDByEmail/success_with_adding_input_of_email
--- PASS: TestRepository_GetUserIDByEmail (0.07s)
    --- PASS:
TestRepository_GetUserIDByEmail/query_by_an_unknown_input_email
(0.04s)
    --- PASS:
TestRepository_GetUserIDByEmail/success_with_adding_input_of_email
(0.04s)
=== RUN    TestRepository_GetEmailsByUserIDs
=== RUN
TestRepository_GetEmailsByUserIDs/success_with_adding_input_of_userIds
=== RUN
TestRepository_GetEmailsByUserIDs/query_by_an_unknown_input_userIds
--- PASS: TestRepository_GetEmailsByUserIDs (0.08s)
    --- PASS:
TestRepository_GetEmailsByUserIDs/success_with_adding_input_of_userIds
(0.03s)
    --- PASS:
TestRepository_GetEmailsByUserIDs/query_by_an_unknown_input_userIds
(0.04s)
=== RUN    TestRepository_GetUsers
=== RUN    TestRepository_GetUsers/successfully_get_all_users
--- PASS: TestRepository_GetUsers (0.04s)
    --- PASS: TestRepository_GetUsers/successfully_get_all_users
(0.04s)
PASS

```

ok

github.com/ToTranMinhNhut/S3_FriendManagementAPI_NhutTo/internal/repository

VI. List APIs

1. Get users

Get: <http://localhost:8080/v1/users>

Parameter request: none

1.1 Success:

Status code 200 OK

```
{
  "count": 1,
  "success": true
  "users": [
    "lisa@example.com"
  ]
}
```

1.2 Error: body request invalid

Status code: 400 Bad Request

```
{
  "message": "Body request invalid format",
  "success": false
}
```

2. Create friend

POST: <http://localhost:8080/v1/friends>

Parameter request:

```
{
  "friends": [
    "andy@example.com",
    "john@example.com"
  ]
}
```

2.1 Success:

Status code: 200 OK

```
{
  "success": true
}
```

2.2 Error: friendship is existing

Status code: 500 Internal Server

```
{
  "message": "The friend relationship has been existed",
  "success": false
}
```


2.3 Error: user is not existing

Status code: 500 Internal Server

```
{
  "message": "test@example.com is not exists",
  "success": false
}
```

2.4 Error: users have blocked by each other

Status code: 500 Internal Server

```
{
  "message": "The users have blocked each other",
  "success": false
}
```

2.5 Error: body request invalid

Status code: 400 Bad Request

```
{
  "message": "Body request invalid format",
  "success": false
}
```

3. List Friends

GET: http://localhost:8080/v1/friends

Parameter request:

```
{
  "Email": "andy@example.com"
}
```

3.1 Success

Status code: 200 OK

```
{
  "count": 1,
  "friends": [
    "lisa@example.com"
  ],
  "success": true
}
```

3.2 Error: user is not existing

Status code: 500 Internal Server

```
{
  "message": "test@example.com is not exists",
}
```

```
"success": false
}
```

3.3 Error: body request invalid

Status code: 400 Bad Request

```
{
  "message": "Body request invalid format",
  "success": false
}
```

4. Get common friends

GET: <http://localhost:8080/v1/commonFriends>

Parameter request:

```
{
  "friends": [
    "andy@example.com",
    "john@example.com"
  ]
}
```

4.1 Success

Status code: 200 OK

```
{
  "count": 1,
  "friends": [
    "common@example.com"
  ],
  "success": true
}
```

4.2 Error: user is not existing

Status code: 500 Internal Server

```
{
  "message": "test@example.com is not exists",
  "success": false
}
```

4.3 Error: body request invalid

Status code: 400 Bad Request

```
{
  "message": "Body request invalid format",
  "success": false
}
```

5. Create subscription

POST: <http://localhost:8080/v1/subscription>

Parameter request:

```
{
  "requestor": "andy@example.com",
  "target": "lisa@example.com"
}
```

5.1 Success

Status code: 200 OK

```
{
  "success": true
}
```

5.2 users have blocked by each other

Status code: 500 Internal Server

```
{
  "message": "The users have blocked each other",
  "success": false
}
```

5.3 Error: users have subscribed by each other

Status code: 500 Internal Server

```
{
  "message": "The users have subscribed each other",
  "success": false
}
```

5.4 Error: body request invalid

Status code: 400 Bad Request

```
{
  "message": "Body request invalid format",
  "success": false
}
```

5.5 Error: user is not existing

Status code: 500 Internal Server

```
{
  "message": "test@example.com is not exists",
  "success": false
}
```

6 Create user block

POST: http://localhost:8080/v1/blocking

Parameter request:

```
{
  "requestor": "common@example.com",
  "target": "kate@example.com"
}
```

6.1 Success

Status code: 200 OK

```
{
  "success": true
}
```

6.2 Error: users have blocked by each other

Status code: 500 Internal Server

```
{
  "message": "The users have blocked each other",
  "success": false
}
```

6.3 Error: body request invalid

Status code: 400 Bad Request

```
{
  "message": "Body request invalid format",
  "success": false
}
```

6.4 Error: user is not existing

Status code: 500 Internal Server

```
{
  "message": "test@example.com is not exists",
  "success": false
}
```

7 Get Recipients

GET: http://localhost:8080 /v1/recipients

Parameter request:

```
{
  "sender": "lisa@example.com",
  "text": "Hello World! kate@example.com"
}
```

7.1 Success

Status code: 200 OK

```
{
  "recipients": [
    "common@example.com",
    "kate@example.com"
  ],
  "success": true
}
```

7.2 Error: sender is not existing

Status code: 500 Internal Server

```
{
  "message": "test@example.com is not exists",
  "success": false
}
```

7.3 Error: body request invalid

Status code: 400 Bad Request

```
{
  "message": "Body request invalid format",
  "success": false
}
```

7.4 Error: user is not existing

Status code: 500 Internal Server

```
{
  "message": "test@example.com is not exists",
  "success": false
}
```