ℹ *Every route has a api prefix, for example: [/auth/register] -> [api/auth/register]*

Unauthenticated Routes

The user is not expected to provide an ID token in each request header

Route:

POST /auth/register

Description:

Register a user account

Body:

```
{
   "email",
   "password",
   "code",
   "first_name",
   "mid_initial,
   "last_name"
}
```

Route:

POST /auth/login

Description:

Login user account. Return the user ID token

Body:

```
{
   "email",
   "password"
}
```

Route:

GET /regist-code/verify/:code

Description:

Validate if the registration code exist in the CRT database

Route:

POST /change-request/mail-submit/:key

Description:

Take incoming emails from Mailgun service

Body:

```
{
   Mail_JSON
}
```

Route:

POST /change-request/mail-request-info/:key

Description:

Process request from email and return change request information

Body:

```
{
   Mail_JSON
}
```

Authenticated Routes

The user is expected and required to provide jwt token in each request header, or else the request will be denied

Route:

GET /util/flag

Description:

Return list of a flagged item owned by the current user (e.g. flagged change request)

Route:

GET /util/notification

Description:

Return all unread notifications and notifications from the last 3 days of the current user

Route:

GET /util/msg

Description:

Return all unread and marked messages of the current user

Route:

POST /util/notification/paginate

Description:

Return paginated notifications list of the current user. Used in notification datatable

Body:

```
{
   DataTable_JSON
}
```

Route:

PATCH /util/notification/clear-new/:target

Description:

Set the select notification from unread to read

Route:

GET /user

Description:

Return information of current user

Route:

GET /user/:email

Description:

Return a user by email

Route:

POST /user/search/:role

Description:

Return list of users filtered by the search data

Body:

```
{
   "term",
   "page"
}
```

Route:

POST /user/datatable

Description:

Return list of users. Used to perform datatable server process

Body:

```
{
   DataTable_JSON
}
```

Route:

DELETE /user/:id

Description:

Delete user by id

Route:

PATCH /user/:id

Description:

Update user by id

Body:

```
{
   "role": "Client || Admin || Developer"
}
```

Route:

POST /regist-code

Description:

Create new registration code

Body:

```
{
   "allowEdit",
   "content",
   "email",
   "first_name",
   "last_name",
   "mid_initial",
   "role": "Developer || Admin || Client"
}
```

Route:

GET /message/:id

Description:

Return message by id

Route:

POST /message/list

Description:

Return all message of the current user in the paginated formula

Body:

```
{
   "limit",
   "page",
   "search",
   "type": "inbox || sent || archive"
}
```

Route:

POST /message

Description:

Create new message

Body:

```
{
   "content",
   "title",
   "receiver": "[ "Name (Email)" ]"
}
```

Route:

PATCH /message/clear-new

Description:

Mark the message as read

Route:

PATCH /message/archive

Description:

Toggle the message archive status

Body:

```
{
   "isArchived",
   "list"
}
```

Route:

PATCH /message/:id

Description:

Update message by id

Body:

```
{
   "isRead",
   "isArchived",
   "isBookmark"
}
```

Route:

POST /change-request/list

Description:

Return change request list of the current user filtered by request tabs

Body:

```
{
   "method": "tab",
   "tab": "active || all || Cancelled || To Do || In Progress || Complete"
}
```

Route:

POST /change-request/admin/list

Description:

Return all change request in the system and filter by the request

Body:

```
{
   "method",
   "date",
   "id",
   "clientsName": "[ Name ]",
   "status": "Cancelled || To Do || In Progress || Complete",
   "tab": "active || all || Cancelled || To Do || In Progress || Complete"
}
```

Route:

GET /change-request/chart/:range

Description:

Return change request status ratio of requested date range. Used in ChartJS

Route:

GET /change-request/:id

Description:

Return change request by id

Route:

POST /change-request/

Description:

Create new change request

Body:

```
{
   "client",
   "message",
   "details",
   "title"
}
```

Route:

PATCH /change-request/:id

Description:

Update change request content by id

Body:

```
{
   "status": "Cancelled || To Do || In Progress || Complete",
   "details",
   "title"
}
```

Route:

POST /change-request/search/:target

Description:

Return list of change requests filtered by the search data

Body:

```
{
   "term",
   "page"
}
```

Route:

DELETE /change-request/:id/unflag

Description:

Delete change request in flagged list

Route:

POST /change-request/:id/flag

Description:

Add change request into flag list

Route:

GET /change-request/:id/msg/:num

Description:

Return requested number of messages in requested change request id

Route:

POST /change-request/:id/msg

Description:

Create a message for a change request

Body:

```
{
   "content"
}
```

Route:

DELETE /change-request/msg/:id

Description:

Delete change request message by message id

Route:

GET /change-request/:id/hist

Description:

Return history of requested change request id

Route:

GET /dev

Description:

Return all developer to-do groups

Route:

POST /dev/todo

Description:

Create new to-do group

Body:

```
{
   "content",
   ",description"
}
```

Route:

DELETE /dev/todo/:id

Description:

Delete to-do group by id

Route:

PATCH /dev/todo/:id

Description:

Update to-do group by id

Body:

```
{
   "content"
}
```

Route:

POST /dev/todo/:id/task

Description:

Add a task to to-do group by id

Body:

```
{
   "content",
   ",description"
}
```

Route:

DELETE /dev/task/:id

Description:

Delete task by id

Route:

PATCH /dev/task/:id

Description:

Update task by id

Body:

```
{
   "content"
}
```

Route:

PATCH /dev/task/complete/:id

Description:

Set task to complete

Body:

```
{
   "isCompleted"
}
```

Route:

DELETE /dev/todo/:id

Description:

Delete to-do group

Route:

GET /test/generate/user/:num

Description:

Generate number of dummy users

Route:

GET /test/generat/cr/:num

Description:

Generate number of dummy change requests

Route:

GET /test/correctCR

Description:

Refresh change request data field after the data got mess up by testing