

Back-End Development

Exploring

#5 Loopback Module





Loopback

LoopBack is a highly-extensible, open-source Node.js framework that enables you to create dynamic end-to-end REST APIs with little or no coding, access databases and also incorporate model relationships and access controls for complex APIs (<http://loopback.io>).

■ Installation

```
$ npm install -g loopback-cli
```

■ Check version

```
$ npx loopback-cli -v
```

Create App



- On terminal, type:

\$ npx loopback-cli

- *What's the name of your application?*
lin_loopback
- *Enter name of the directory to contain the project:*
lin_loopback
- *Which version of LoopBack would you like to use?*
3.x (current)
- *What kind of application do you have in mind?*
api-server

- Go to project directory:

\$ cd lin_loopback

- Run the application:

\$ node .

- *Web server Listening at: <http://localhost:3000>*
- *Browse your REST API at <http://localhost:3000/explorer>*

http://localhost:3000/explorer

LoopBack API Explorer

Token Set: [Set Access Token](#)

User

Show/Hide | List Operations | Expand Operations

PATCH	/Users	Patch an existing model instance or insert a new one into the data source.
GET	/Users	Find all instances of the model matched by filter from the data source.
PUT	/Users	Replace an existing model instance or insert a new one into the data source.
POST	/Users	Create a new instance of the model and persist it into the data source.
PATCH	/Users/{id}	Patch attributes for a model instance and persist it into the data source.
GET	/Users/{id}	Find a model instance by {{id}} from the data source.
HEAD	/Users/{id}	Check whether a model instance exists in the data source.
PUT	/Users/{id}	Replace attributes for a model instance and persist it into the data source.
DELETE	/Users/{id}	Delete a model instance by {{id}} from the data source.
GET	/Users/{id}/accessTokens	Queries accessTokens of User.
POST	/Users/{id}/accessTokens	Creates a new instance in accessTokens of this model.
DELETE	/Users/{id}/accessTokens	Deletes all accessTokens of this model.
GET	/Users/{id}/accessTokens/{fk}	Find a related item by id for accessTokens.
PUT	/Users/{id}/accessTokens/{fk}	Update a related item by id for accessTokens.
DELETE	/Users/{id}/accessTokens/{fk}	Delete a related item by id for accessTokens.

Set App ★

- By default, we must set token for accessing loopback API. To use its API without any authentication, do these steps:

Go to **projectDir/server/boot/authentication.js**
Comment this line: `server.enableAuth();`

- Try to **POST** at **POST/Users!**

```
{  
  "email" : "lintang@purwadhika.com",  
  "password" : "123456"  
}
```

- Try to post custom data, then **GET** at **GET/Users!**

POST

LoopBack API ExplorerToken Not SetSet Access Token

POST /UsersCreate a new instance of the model and persist it into the data source.

Parameters

Parameter	Value	Description	Parameter Type	Data Type				
data	<pre>{ "email": "lin@aku.id", "password": "1234321"} </pre> <div>Parameter content type: application/json</div>	Model instance data	body	<table><thead><tr><th>Model</th><th>Example Value</th></tr></thead><tbody><tr><td></td><td><pre>{ "realm": "string", "username": "string", "email": "string", "emailVerified": true}</pre></td></tr></tbody></table>	Model	Example Value		<pre>{ "realm": "string", "username": "string", "email": "string", "emailVerified": true}</pre>
Model	Example Value							
	<pre>{ "realm": "string", "username": "string", "email": "string", "emailVerified": true}</pre>							

Try it out!

[Hide Response](#)

Curl

```
curl -X POST --header 'Content-Type: application/json' --header 'Accept: application/json' -d '{ \  "email": "lin@aku.id", \  "password": "1234321" \}' 'http://localhost:3000/api/Users'
```

Request URL

```
http://localhost:3000/api/Users
```

Response Body

```
{  "email": "lin@aku.id",  "id": 3}
```

Response Code

```
200
```

- By default, “email” & “password” have to be declared correctly!
- It will response an “id” automatically!
- You can type other JSON data after “email” & “password” too.

GET

LoopBack API Explorer

Token Not Set

Set Access Token

GET /Users

Find all instances of the model matched by filter from the data source.

Try it out!

[Hide Response](#)

Curl

```
curl -X GET --header 'Accept: application/json' 'http://localhost:3000/api/Users'
```

Request URL

```
http://localhost:3000/api/Users
```

Response Body

```
[
  {
    "email": "sasas@kokok.mama",
    "id": 1
  },
  {
    "email": "sasaas@kokoak.mama",
    "id": 2
  },
  {
    "email": "lin@aku.id",
    "id": 3
  }
]
```

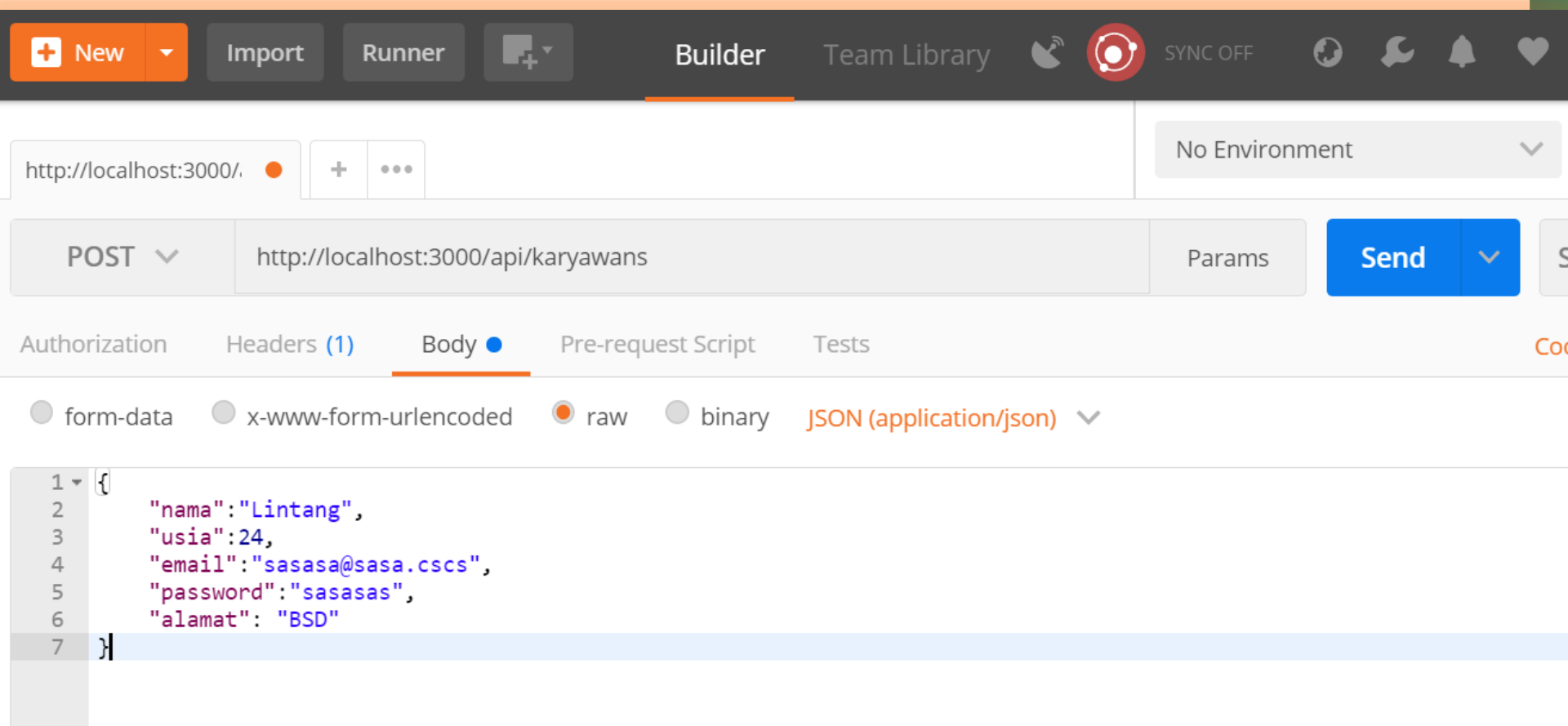
■ It will response an “id”
& hide the “password”
automatically!



- A powerful GUI platform to make API development faster & easier, from building API requests through testing, documentation & sharing.
- Postman has features for every API developer: request building, tests & pre-request scripts, variables, environments & request descriptions, designed to work seamlessly together.
- Download from getpostman.com

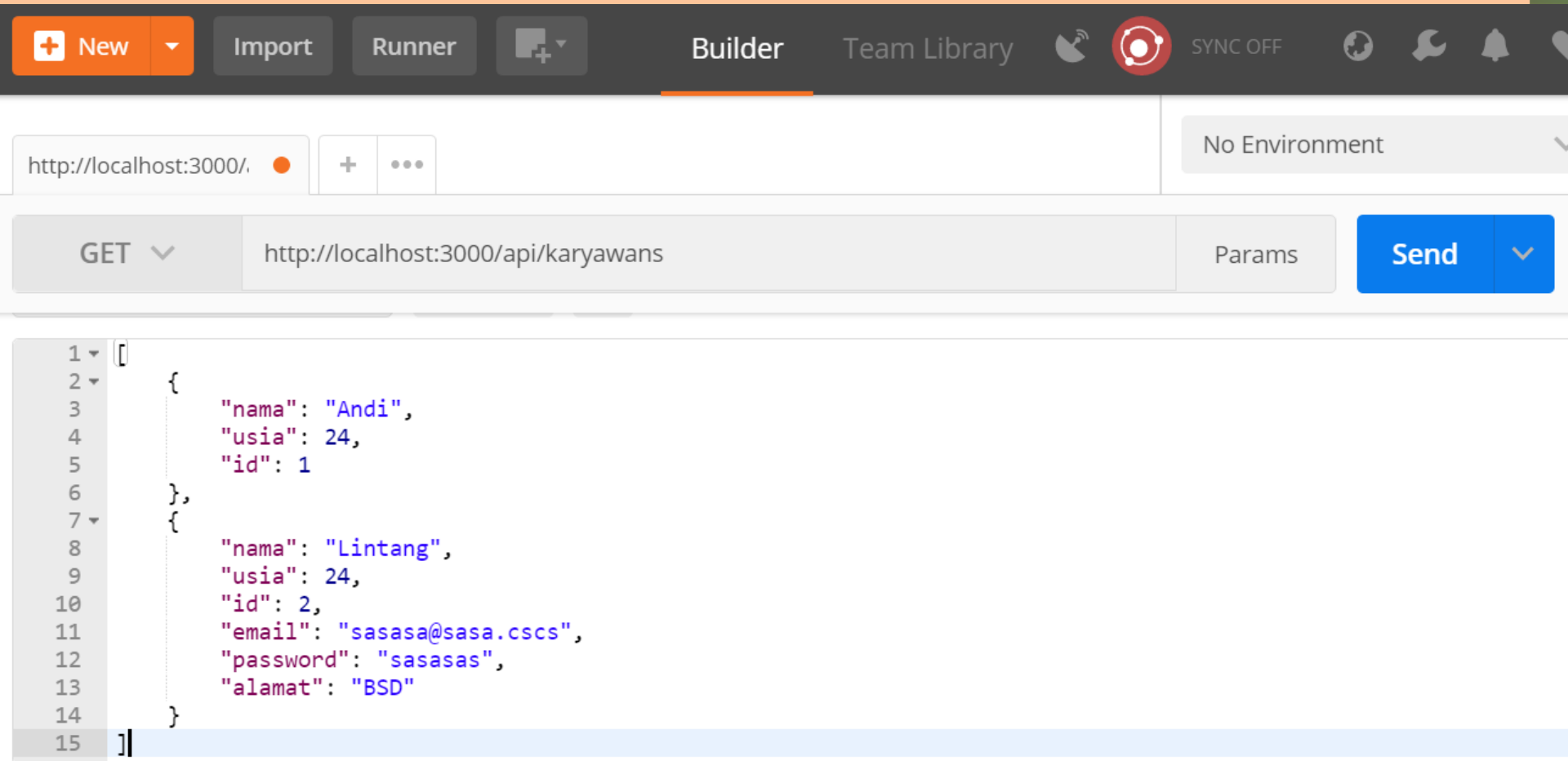
**Try to POST & GET
with POSTMAN!**

POST with POSTMAN



Set Header Request to
Content-Type: application/json

GET with POSTMAN



Create Model

- On directory project, type:

\$ npx loopback-cli model

- [?] Enter the model name: *karyawan*
- [?] Select the data-source to attach person to: *db*
- [?] Select model's base class (*PersistedModel*)
- [?] Expose person via the REST API? *Yes*
- [?] Custom plural form (used to build REST URL):
- [?] Common model or server only? *Common*

Let's add some person properties now.

Enter an empty property name when done.

- [?] Property name: *id*
- [?] Property type: (Use arrow keys) *number*
- [?] Required? (y/N) *y*

- Property *“id”* is a must! After done, try to run its app via terminal: **\$ node .**

Create Model

LoopBack API Explorer

Token Not Set

accessToken

lin_loopback

karyawan

Show/Hide | List Operations

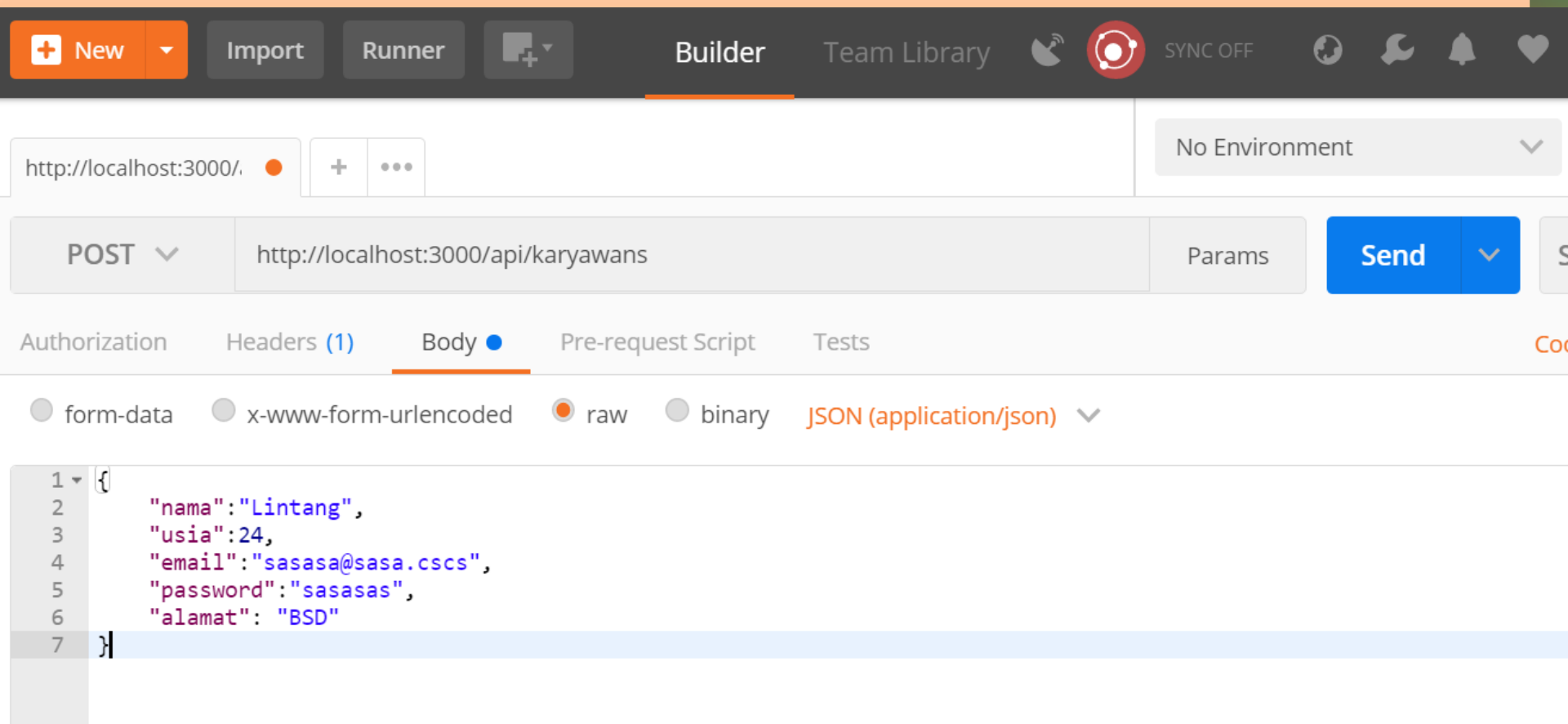
User

Show/Hide | List Operations

[BASE URL: /api , API VERSION: 1.0.0]

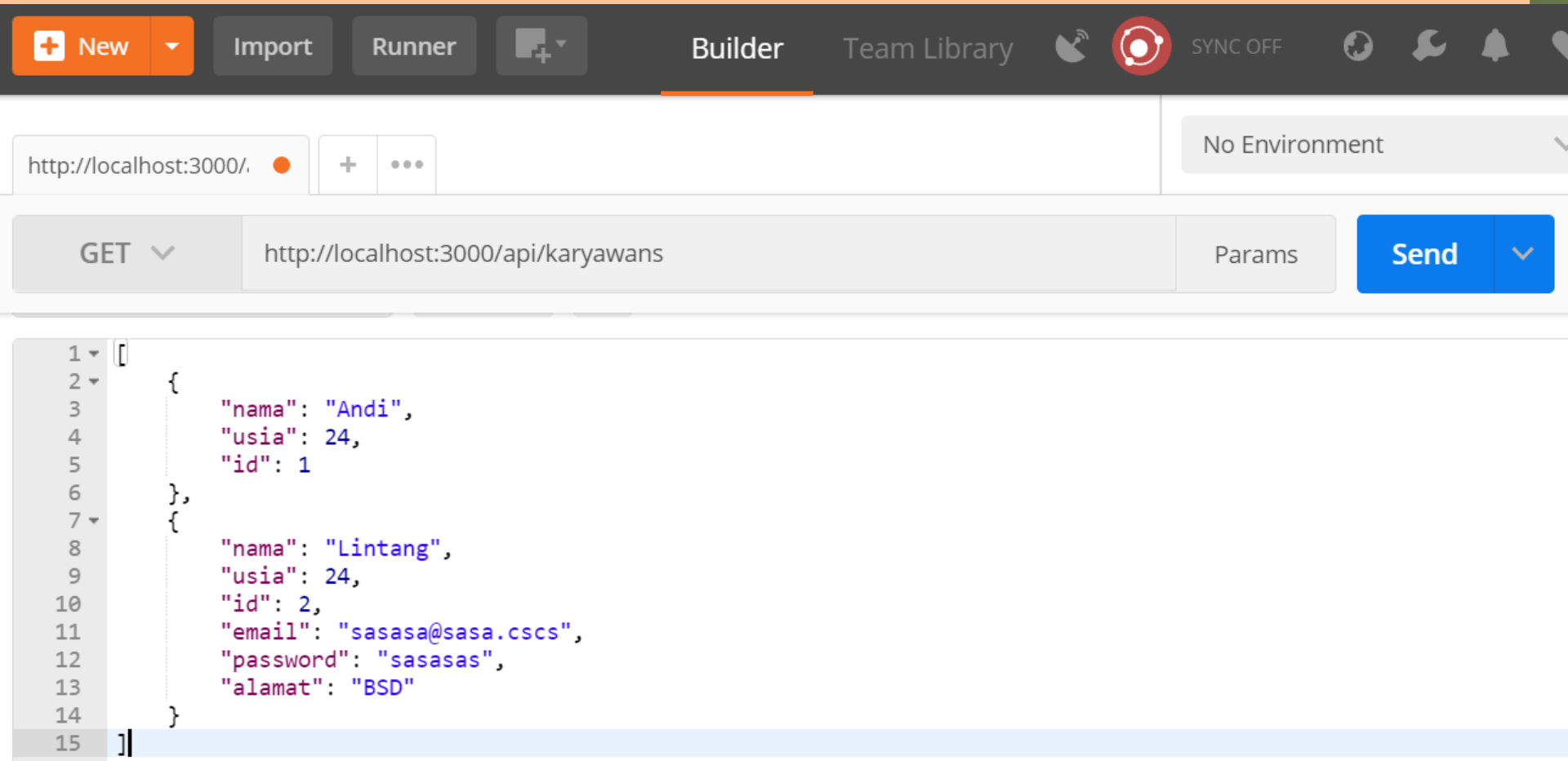
**Try to POST & GET
with POSTMAN!**

POST with POSTMAN



Set Header Request to
Content-Type: application/json

GET with POSTMAN



Back-End Development

Exploring

#5 Loopback Module

