aiolambda

Release 0.2.6

Dec 20, 2018

Contents:

1 Quickstart **1**
- 1.1 Requirements: . . . 1
- 1.2 Create *project*: . . . 1
- 1.3 OpenAPI specs: . . . 1
- 1.4 Code . . . 3
- 1.5 Useful commands: . . . 3

2 Template init **5**

3 Template update **7**

4 Database **9**

5 AMQP **11**

6 Roadmap **13**

7 Aiolambda-cli **15**

8 Indices and tables **17**

CHAPTER 1

Quickstart

1.1 Requirements:

- **docker**
- **python3**
- **pip3**
- **aiolambda-cli:** *pip install -U aiolambda*

1.2 Create *project*:

```
$ aiolambda-cli init jwt
```

1.3 OpenAPI specs:

Define microservice API specs, for example:

```
openapi: 3.0.0
info:
  title: JWT Example
  version: '1.0'
servers:
  - url: /v1
tags:
- name: auth
  description: "Generate token"
- name: secret
  description: "Return secret response"
```

(continues on next page)

(continued from previous page)

```
- name: ping
  description: "Test service"
paths:
  /auth/{user_id}:
    get:
      tags:
      - auth
      summary: Return JWT token
      operationId: jwt.handlers.auth_handler
      parameters:
      - name: user_id
        description: User unique identifier
        in: path
        required: true
        example: 12
        schema:
          type: integer
      responses:
        '200':
          description: JWT token
          content:
            'application/json':
              schema:
                type: string
  /secret:
    get:
      tags:
      - secret
      summary: Return secret string
      operationId: jwt.handlers.secret_handler
      responses:
        '200':
          description: secret response
          content:
            'application/json':
              schema:
                $ref: '#/components/schemas/Secret'

      security:
      - jwt: ['secret']
  /ping:
    get:
      tags:
      - ping
      summary: "Test service"
      description: ""
      operationId: "jwt.handlers.ping_handler"
      responses:
        '200':
          description: successful operation

components:
  schemas:
    Secret:
      type: "object"
      properties:
        user_id:
```

(continues on next page)

(continued from previous page)

```
          type: "string"
          example: "12"
        secret:
          type: "string"
          example: "wbevuec"
        token_info:
          $ref: '#/components/schemas/Token info'
    Token info:
      type: object
      properties:
        iss:
          type: string
          example: aiolambda
        iat:
          type: integer
          example: 1545340057
        exp:
          type: integer
          example: 1545340657
        sub:
          type: string
  securitySchemes:
    jwt:
      type: http
      scheme: bearer
      bearerFormat: JWT
      x-bearerInfoFunc: aiolambda.security.decode_token
```

1.4 Code

Define your custom handlers referenced in *openAPI specs*. For example, *auth_handler*:

```
async def auth_handler(user_id: str) -> Response:
    return compose(
        generate_token,
        logger.debug,
        return_200)(user_id)
```

1.5 Useful commands:

See example *jwt* README.md

CHAPTER 2

Template init

TODO

CHAPTER 3

Template update

CHAPTER 4

Database

TODO

CHAPTER 5

AMQP

TODO

CHAPTER 6

Roadmap

- **Add more examples:**
  - **crud_users: postgres. Postgres calls.** *Document project: [**page:database:postgres**]
  - **user_settings: mq. Send and subscribe AMQP messages.** *Document project: [**page:amqp**]
  - ***jwt*: ci integration. Basic CI integration with travis.** *Document project: [**page:ci:travisci**]
  - ***jwt*: container integration. Basic container integration.** *Document project: [**page:containers:docker:**]
  - ***jwt*: functional implementations. Async compose, types: errors, ROP documentation.** *Document project: [**page:functional**]
- **Change command definitions:**
  - **inputs:** Command defined with env vars, not cli commands (only use cli commands for short commands [big execution number of times]).
  - **updates:** Makes with inline, replace based ansible filters. (same things: number_of_postgres versions, etc… with default values)
  - **.env:** Recomend .env for auto updates and load envars in each directory (*projects*).
- **Contributions:**
  - **databases:**
    - **redis**…
    - **mongo**…
  - **containers:**
    - **firecracker**…
  - **code analysis:**
    - **code climate**
  - **automated api tests:**

    - **dredd**
  - aiolambda functional

CHAPTER 7

Aiolambda-cli

```
Usage:
    aiolambda-cli init [--db=<name>] [--mq] [--ci] <project-name>
    aiolambda-cli [-h]
Options:
    -h --help       Show this screen.
    --version       Show version.
    --db=<name>     Database to include: [postgresql]. [default: postgresql]
    --mq            Include RabbitMQ/AMQP client library.
    --ci            Include Travis-CI configuration.
```

CHAPTER 8

Indices and tables

- genindex
- search