МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ”
ІПСА
Кафедра Системного проектування

Лабораторна робота 6

з дисципліни: «ПРОЕКТУВАННЯ ІНФОРМАЦІЙНИХ СИСТЕМ»

Система автоматичного створення довідника користувача та оформлення коду за допомогою Coding Convention.

Виконав:

студент групи ДА-71

Павлюк Вадим

Мета роботи: за допомогою системи генерації довідника користувача створити документ у форматі PDF і HTML для архітектурної програмної моделі.

Оформлення документації в коді

```
const axios = require('axios');
/**
 * @class
 * @description Description: This class has methods for CRUD API
 * @constructor
 * @param {string} url - this is Endpoint link
 * @throws {string} Error if typeof url is not string
 * @example
 * const api = new Api('http://localhost:3000/Storage')
 */
class Api {
  constructor(url) {
    if (typeof url !== 'string') {
      throw new Error('Api constructor() url expected to be typeof String');
    }
    Object.defineProperties(this, {
      url: { get: () => url },
      status: {
        get: () => (obj) => {
          console.log(`Status code: ${obj.status}`);
          console.log(`Status text: ${obj.statusText}`);
          console.log(`Request method: ${obj.request.method}`);
          console.log(`Path: ${obj.request.path}`);
          console.log(`Date: ${obj.headers.date}`);
        }
      }
    })
    Object.freeze(this)
  }
/**
 * @param {string} params To get all data, set this value in ".To get a certain see example
 * @param {boolean} isStatus If needed to look at status, set this value in true
 * @description Description: This method is used to make GET request
 * @throws {string} Error if link is not valid
 * @returns {array} returns array of object that were requested
 * @example
 * api.makeGet('/?id=2', true) // getting Object where id = 2
 */
  async makeGet(params='', isStatus=false) {
    try {
      const link = this.url + params;
      const response = await axios.get(link);
```

```
      if (isStatus) {
        this.status(response);
      }
      return response.data
    }
    catch (error) {
      console.error(error);
    }
  }

/**
 *
 * @param {object} params Contains properties
 * @param {boolean} isStatus if needed to look at status, set this value in true
 * @description Description: This method is used to make POST request
 * @throws {string} Error if link is not valid
 * @returns Nothing
 * @example
 * api.makePost({id: 4, user: tes1}, true) // creating new object where id = 4
 */
  async makePost(params={}, isStatus=false) {
    try {
      const response = await axios.post(this.url, params);
      if (isStatus) {
        this.status(response);
      }
    }
    catch (error) {
      console.error(error);
    }
  }
/**
 *
 * @param {string} params
 * @param {boolean} isStatus if needed to look at status, set this value in true
 * @description Description: This method is used to make DELETE request
 * @throws {string} Error if link is not valid
 * @returns Nothing
 * @example
 * api.makeDelete('/2', true) // deleting object where id = 2
 */
  async makeDelete(params='', isStatus=false) {
    try {
      const link = this.url + params;
      const response = await axios.delete(link);
      if (isStatus) {
        this.status(response);
      }
    }
    catch (error) {
```

```
      console.error(error);
    }
  }
/**
 *
 * @param {object} params
 * @param {boolean} isStatus if needed to look at status, set this value in true
 * @description Description: This method is used to make PATCH request
 * @throws {string} Error if link is not valid
 * @returns Nothing
 * @example
 * api.makeUpdate({path: '/2', config: {user: test2}}, true) // updating object where id = 2
 */
  async makeUpdate(params={}, isStatus=false) {
    try {
      const link = this.url + params.path;
      const response = await axios.patch(link, params.config);
      if (isStatus) {
        this.status(response);
      }
    }
    catch (error) {
      console.error(error);
    }
  }
}
```

Для автоматичного створення документації використовував jsdoc

Результат

Class: Api

Api(url)

new Api(url)

Description: This class has methods for CRUD API

Parameters:

Name	Type	Description
url	string	this is Endpoint link

Source: apiDocumentation.js, line 11

Throws:

Error if typeof url is not string

Type
string

Example

```
const api = new Api('http://localhost:3000/Storage')
```

Home

Classes

Api

Methods

(async) makeDelete(params, isStatus)

Description: This method is used to make DELETE request

Parameters:

Name	Type	Default	Description
params	string		
isStatus	boolean	false	if needed to look at status, set this value in true

Source: apiDocumentation.js, line 84

Throws:

Error if link is not valid

Type
string

Returns:

Nothing

Example

```
api.makeDelete('/2', true) // deleting object where id = 2
```

(async) makeGet(params, isStatus) → {array}

Використовувались теги:

@description тег дозволяє надати загальний опис символу, який ви документуєте

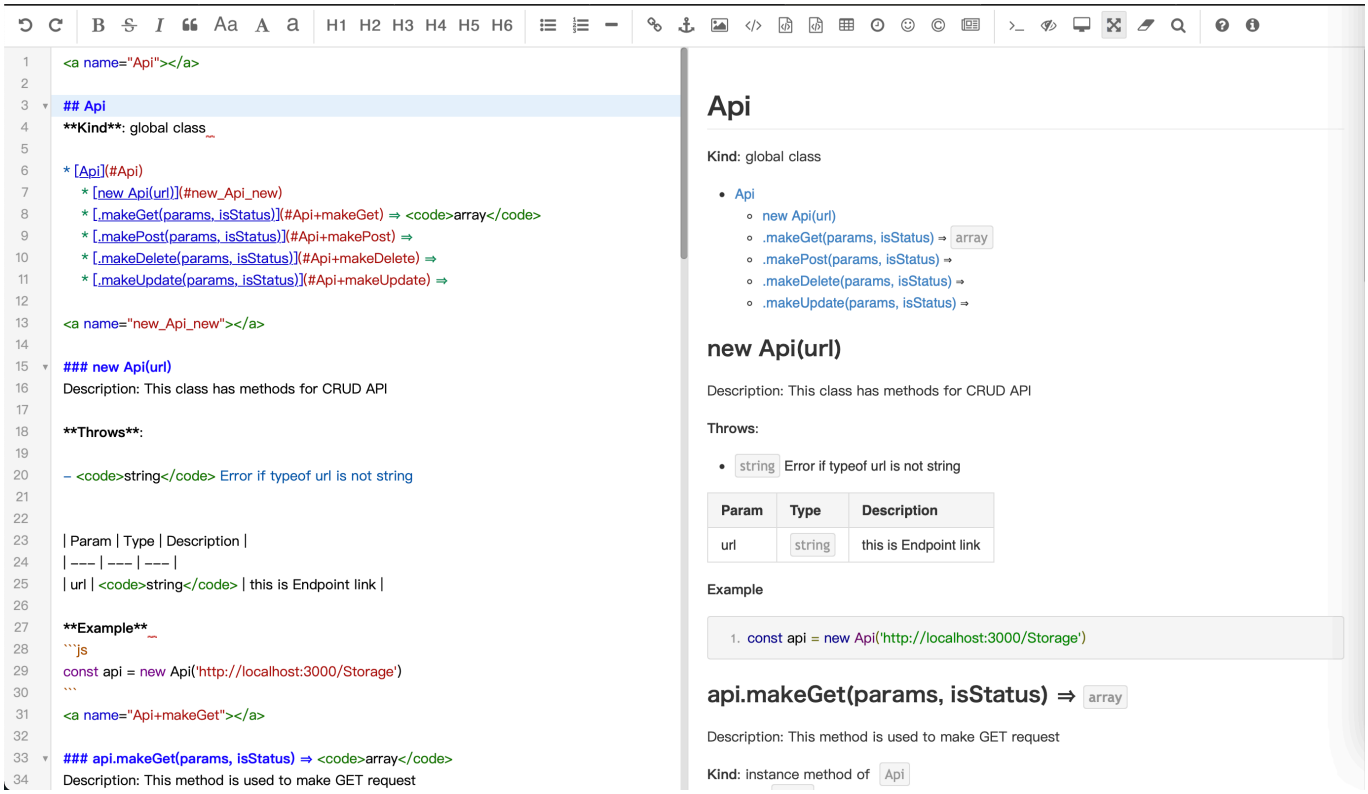
@param тег містить назву, тип та опис параметра функції.

@throws тег дозволяє задокументувати помилку, яку може викинути функція.

@example наводить приклад того, як використовувати документований предмет.

@returns тег документує значення, яке повертає функція.

Створити документацію в PDF-форматі невдалося оскільки в jsdoc немає такої можливості, тому створив в MD форматі.



```
<a name="Api"></a>

## Api
**Kind**: global class

* [Api](#Api)
    * [new Api(url)](#new_Api_new)
    * [.makeGet(params, isStatus)](#Api+makeGet) ⇒ <code>array</code>
    * [.makePost(params, isStatus)](#Api+makePost) ⇒
    * [.makeDelete(params, isStatus)](#Api+makeDelete) ⇒
    * [.makeUpdate(params, isStatus)](#Api+makeUpdate) ⇒

<a name="new_Api_new"></a>

### new Api(url)
Description: This class has methods for CRUD API

**Throws**:

- <code>string</code> Error if typeof url is not string


| Param | Type | Description |
| --- | --- | --- |
| url | <code>string</code> | this is Endpoint link |

**Example**
```js
const api = new Api('http://localhost:3000/Storage')
```


api.makeGet(params, isStatus) ⇒ <code>array</code>
Description: This method is used to make GET request
```

Api

Kind: global class

- Api
  - new Api(url)
  - .makeGet(params, isStatus) ⇒ array
  - .makePost(params, isStatus) ⇒
  - .makeDelete(params, isStatus) ⇒
  - .makeUpdate(params, isStatus) ⇒

new Api(url)

Description: This class has methods for CRUD API

Throws:

- string Error if typeof url is not string

Param	Type	Description
url	string	this is Endpoint link

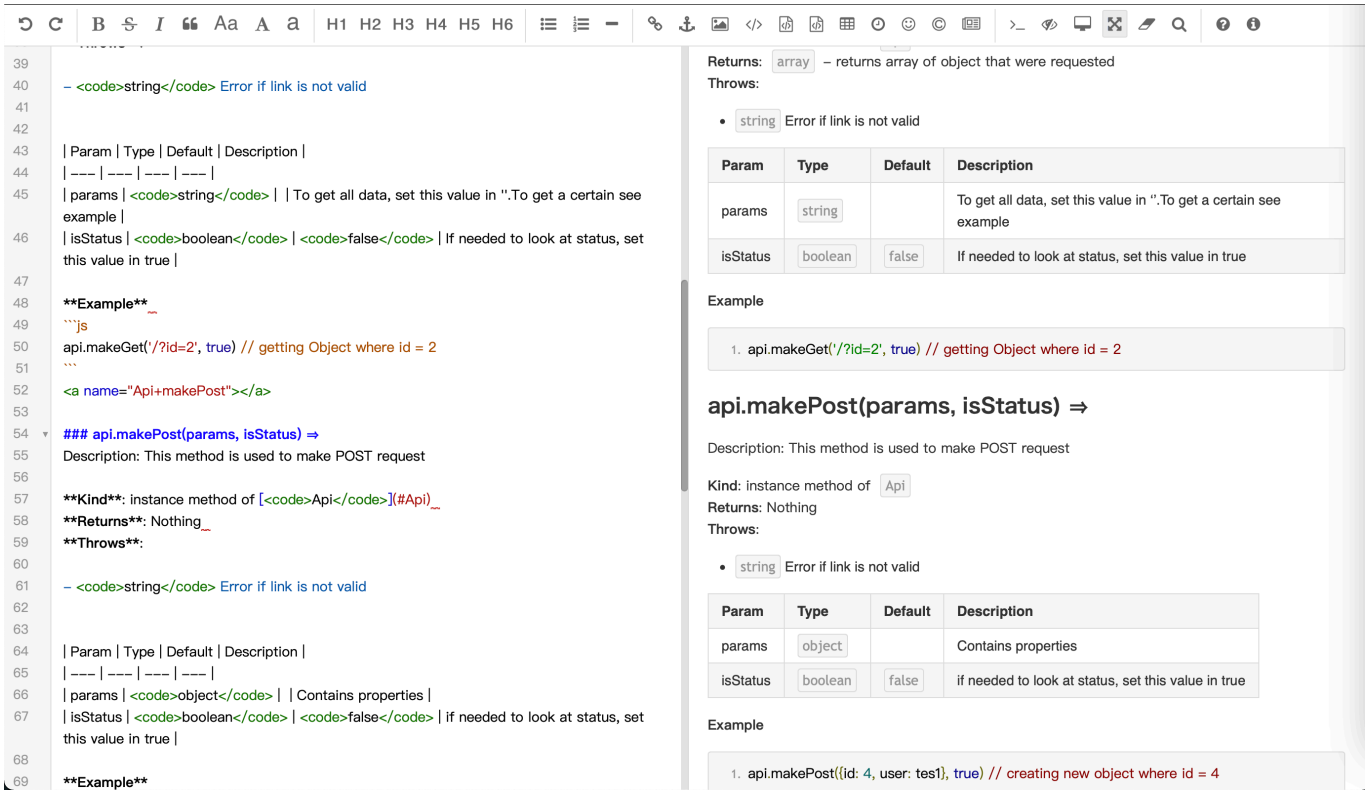
Example

```
const api = new Api('http://localhost:3000/Storage')
```

api.makeGet(params, isStatus) ⇒ array

Description: This method is used to make GET request

Kind: instance method of Api



```
39
40  – <code>string</code> Error if link is not valid
41
42
43  | Param | Type | Default | Description |
44  | --- | --- | --- | --- |
45  | params | <code>string</code> |  | To get all data, set this value in ''.To get a certain see example |
46  | isStatus | <code>boolean</code> | <code>false</code> | If needed to look at status, set this value in true |
47
48  **Example**
49  ```js
50  api.makeGet('/?id=2', true) // getting Object where id = 2
51  ```
52  <a name="Api+makePost"></a>
53
54  ### api.makePost(params, isStatus) ⇒
55  Description: This method is used to make POST request
56
57  **Kind**: instance method of [<code>Api</code>](#Api)
58  **Returns**: Nothing
59  **Throws**:
60
61  – <code>string</code> Error if link is not valid
62
63
64  | Param | Type | Default | Description |
65  | --- | --- | --- | --- |
66  | params | <code>object</code> |  | Contains properties |
67  | isStatus | <code>boolean</code> | <code>false</code> | if needed to look at status, set this value in true |
68
69  **Example**
```

Returns: array – returns array of object that were requested
Throws:

- string Error if link is not valid

Param	Type	Default	Description
params	string		To get all data, set this value in ''.To get a certain see example
isStatus	boolean	false	If needed to look at status, set this value in true

Example

```
1. api.makeGet('/?id=2', true) // getting Object where id = 2
```

api.makePost(params, isStatus) ⇒

Description: This method is used to make POST request

Kind: instance method of Api
Returns: Nothing
Throws:

- string Error if link is not valid

Param	Type	Default	Description
params	object		Contains properties
isStatus	boolean	false	if needed to look at status, set this value in true

Example

```
1. api.makePost({id: 4, user: tes1}, true) // creating new object where id = 4
```

Висновки

У даній лабораторній роботі навчився автоматично створювати документацію за допомогою jsdoc, у форматах HTML та MD. Також оформив код програми згідно W3Schools Coding Convention.

W3Schools https://www.w3schools.com/js/js_conventions.asp