

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**  
**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**  
**“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**  
**імені ІГОРЯ СІКОРСЬКОГО”**

Інститут прикладного системного аналізу  
кафедра системного проектування

**Контрольна робота №6**  
з дисципліни «Проектування інформаційних систем»

Виконав:  
студент 4 курсу  
групи ДА-72  
Кулик В.О.

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

**Задача:**

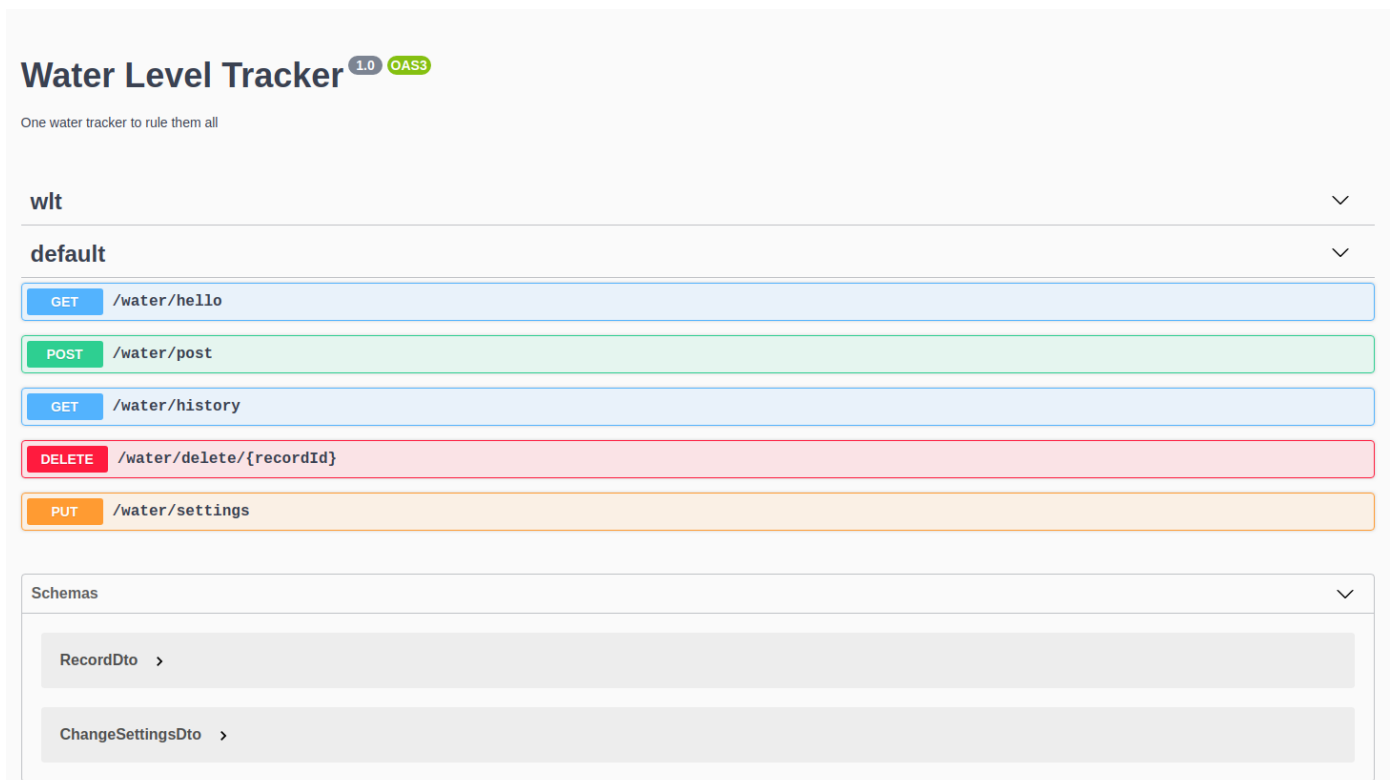
1. Вивчити теги системи генерації керівництва користувача.
2. Створити опис для всіх класів API з описом призначення кожного класу, методів класу і членів класу.
3. Згенерувати документацію у форматах PDF, HTML.

**Хід роботи:**

**1. Документація користувача**

Для генерування документації користувача було використано інструмент Swagger, що має надбудови до Nest.js, на якому проводилося написання коду програми.

Вид документації у згорнутому стані:



Більш детально роздивимося деякі кінцеві точки у контролері:

**WaterLevelTracker**

**default**

**GET** /water/hello

**POST** /water/post

Parameters

No parameters

Request body *required*

application/json

Example Value | Schema

```
{
  "time": "string",
  "water": 0
}
```

Responses

Code	Description	Links
201	Adding a new record of water drunk to the history	No links

**DELETE** /water/delete/{recordId}

Parameters

Name	Description
<b>recordId</b> <i>required</i>	ID of particular record to find one
number	
(path)	

recordId - ID of particular record to find one

Responses

Code	Description	Links
200	Deleting the water record	No links

**PUT** /water/settings

Schemas

RecordDto

```
{
  time* string
    Time when the water portion is taken
  water* number
    Amount of water (in milliliters) that is taken
}
```

Як бачим, документація детально відображає усі необхідні дані для використання API програми.

Вказані параметри, вигляд та опис компонентів програми.

## 2. Coding convention – ESLint/AirBnb.

Даний конвеншн був обраний серед інших цечerez свою популярність на ринку, практичність та актуальність.

Приклади з конвершном та без:

## 1) Без конвенції коду:

```
src > app.service.ts > AppService > findRecordById
1  import { Injectable } from '@nestjs/common';
2  import { RecordDto } from './record.dto';
3
4  @Injectable()
5  export class AppService {
6    getGreeting(str: string): string { return str }
7
8    getHistory(): RecordDto[] {
9      const arr = [];
10     const recordOne = {time: '14:00', water: 300}
11     const recordTwo = {time: '15:00', water: 250}
12     arr.push(RecordDto.from(recordOne));
13     arr.push(RecordDto.from(recordTwo));
14     return arr
15   }
16
17   findRecordById(recordId): RecordDto {const record = new RecordDto();return record; }
18 }
19
```

## 2) З конвенцією коду:

```
src > app.service.ts > AppService > getHistory
1  import { Injectable } from '@nestjs/common';
2  import { RecordDto } from './record.dto';
3
4  @Injectable()
5  export class AppService {
6    getGreeting(str: string): string {
7      return str;
8    }
9
10   getHistory(): RecordDto[] {
11     const arr = [];
12     const recordOne = {
13       time: '14:00',
14       water: 300,
15     };
16     const recordTwo = {
17       time: '15:00',
18       water: 250,
19     };
20
21     arr.push(RecordDto.from(recordOne));
22     arr.push(RecordDto.from(recordTwo));
23
24     return arr;
25   }
26
27   findRecordById(recordId): RecordDto {
28     const record = new RecordDto();
29     return record;
30   }
31 }
32
```

**3. Висновок:** в ході виконання лабораторної роботи було вивчено теги для генерації документації користувача для коду програми (ендпоінтів), що містить детальні пояснення та вказівки щодо використання. Було також підібрано конвенцію коду для подальшого створення проекту, проаналізовано її переваги та обгрунтовано даний вибір.