

Entry-Level Software Engineer

Practical Assessment

HealthTrack — Water Intake Tracking API

This documentation is written by Aulia Nadhirah Yasmin Badrulkamal

DEPENDENCIES NEEDED

- typescript
- npm
- @types/express@5.0.5
- @types/uuid@10.0.0
- express@5.1.0
- uuid@13.0.0

HOW TO RUN

**To run this on your machine, make sure you all the dependencies needed before tunning*

1. run npm install (this will install all the relevant packages)
2. run npm run dev
3. when you see **API listening on port \${port}** on your terminal, the API is running
4. use software such as **Postman** or **Insomnia** to use the API endpoints

API ENDPOINTS

1. LOG WATER INTAKE

POST /water-logs	
Request	
Header	Content-Type: application/json
Body	{

	<pre> "amount" : 100, "note" : "Drank water today!" } </pre>
Response	
<pre> { "status": "success", "message": "water intake successfully logged!", "id": "37d385da-8cf5-4cab-a192-60f055df1fe2", "data": { "amount": 100, "timestamp": "2025-11-11T10:00:33.346Z", "note": "Drank water today!" } } </pre>	
Error Response (amount is not provided)	
<pre> { "status": "error", "message": "Water amount is not provided!", "id": null, "data": null } </pre>	

In this API endpoint, I generated an UUID to identify each log. Since it's a very unique ID that is not easily replicated. Using regular numbers that are based on how many logs there are can get messy when some logs are deleted, creating the risk of duplicate IDs.

2. GET TODAY'S WATER INTAKE SUMMARY

GET /water-logs/today	
Request	
Header	Content-Type: application/json
Body	
Response	

```
{
  "status": "success",
  "message": "successfully returned summary",
  "id": "11/11/2025",
  "data": {
    "dailyGoal": 2000,
    "totalAmount": 100,
    "progressPercentage": 5,
    "logs": {
      "37d385da-8cf5-4cab-a192-60f055df1fe2": {
        "amount": 100,
        "timestamp": "2025-11-11T10:00:33.346Z",
        "note": "Drank water today!"
      }
    }
  }
}
```

Error Response (No water logs that day)

```
{
  "status": "error",
  "message": "No water intake log today",
  "id": null,
  "data": null
}
```

3. GET LOGS FOR A SPECIFIC DATE

GET /water-logs?date=YYYY-MM-DD

Request

Header	Content-Type: application/json
---------------	--------------------------------

Body

Response

```
{
```

```

    "status": "success",
    "message": "successfully returned summary",
    "id": "11/11/2025",
    "data": {
      "37d385da-8cf5-4cab-a192-60f055df1fe2": {
        "amount": 100,
        "timestamp": "2025-11-11T10:00:33.346Z",
        "note": "Drank water today!"
      }
    }
  }
}

```

Error Response (No water logs that day)

```

{
  "status": "error",
  "message": "No water intake on 13/11/2025",
  "id": null,
  "data": null
}

```

4. SET DAILY WATER INTAKE GOAL

POST /daily-goal

Request

Header	Content-Type: application/json
---------------	--------------------------------

Body	<pre> { "newDailyGoal": 3000 } </pre>
-------------	---

Response

```

{
  "status": "success",
  "message": "returned daily goal",
  "id": "11/11/2025",
  "data": 2000
}

```

}
Error Response (newDailyGoal is not provided)
<pre>{ "status": "error", "message": "New daily goal is not provided!", "id": null, "data": null }</pre>

Here I have to make an assumption for the definition of “Daily Goal”. From the example and the name of the variable itself, I assumed it is a value that could change each day. Since if daily goal was a global variable for every single day, major recalculation for “**progressPercentage**” variable for every single day would be needed.

5. GET CURRENT DAILY GOAL

GET /daily-goal	
Request	
Header	Content-Type: application/json
Body	
Response	
<pre>{ "status": "success", "message": "changed daily goal", "id": "11/11/2025", "data": { "dailyGoal": 3000, "totalAmount": 100, "progressPercentage": 5, "logs": { "37d385da-8cf5-4cab-a192-60f055df1fe2": { "amount": 100, "timestamp": "2025-11-11T10:00:33.346Z", </pre>	

```

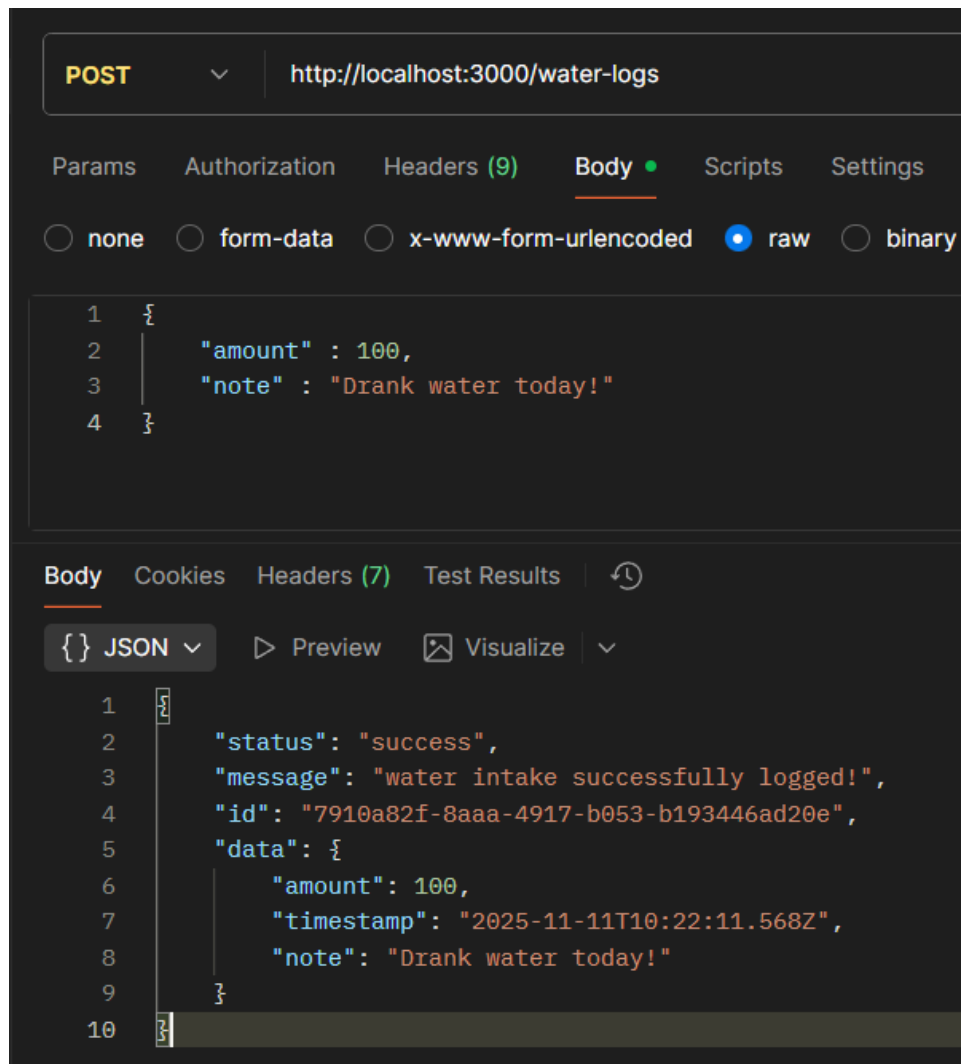
    "note": "Drank water today!"
  }
}
}

```

Here I assumed that if there is no water log that day, this API would still make a dailyLog with an empty water intake log and a default daily goal of 2000. This goes the same with the **POST /daily-goal** API endpoint

TESTING

TC01	
Testcase	Logging water intake successfully
Instructions	Use the POST <code>http://localhost:3000/water-logs</code> API by using Postman / Insomnia
Input	<pre> { "amount": 100, "note": "Drank water today!" } </pre>
Expected Behavior	it would accept the user's input and save the new water intake into the database. It would then send a response that shows the new water intake data.
Actual Output	<pre> { "status": "success", "message": "water intake successfully logged!", "id": "7910a82f-8aaa-4917-b053-b193446ad20e", "data": { "amount": 100, "timestamp": "2025-11-11T10:22:11.568Z", "note": "Drank water today!" } } </pre>
Status	PASS
Testing Documentation	



TC02	
Testcase	Getting today's total water intake
Instructions	Use the GET <code>http://localhost:3000/water-logs/today</code> API by using Postman / Insomnia
Input	-
Expected Behavior	it would retrieve daily log data from the database and returns it to the user. The daily log data contains : <ul style="list-style-type: none">dailyGoal

	<ul style="list-style-type: none">• totalAmount• progressPercentage• logs
Actual Output	<pre>{ "status": "success", "message": "successfully returned summary", "id": "11/11/2025", "data": { "dailyGoal": 3000, "totalAmount": 200, "progressPercentage": 6.666666666666667, "logs": { "37d385da-8cf5-4cab-a192-60f055df1fe2": { "amount": 100, "timestamp": "2025-11-11T10:00:33.346Z", "note": "Drank water today!" }, "7910a82f-8aaa-4917-b053-b193446ad20e": { "amount": 100, "timestamp": "2025-11-11T10:22:11.568Z", "note": "Drank water today!" } } } }</pre>
Status	PASS
Testing Documentation	


```
GET http://localhost:3000/water-logs/today

Body
JSON
1 {
2   "status": "success",
3   "message": "successfully returned summary",
4   "id": "11/11/2025",
5   "data": {
6     "dailyGoal": 3000,
7     "totalAmount": 200,
8     "progressPercentage": 6.666666666666667,
9     "logs": {
10      "37d385da-8cf5-4cab-a192-60f055df1fe2": {
11        "amount": 100,
12        "timestamp": "2025-11-11T10:00:33.346Z",
13        "note": "Drank water today!"
14      },
15      "7910a82f-8aaa-4917-b053-b193446ad20e": {
16        "amount": 100,
17        "timestamp": "2025-11-11T10:22:11.568Z",
18        "note": "Drank water today!"
19      }
20    }
21  }
22 }
```

TC03	
Testcase	Calculating progress toward daily goal
Instructions	Use the GET <code>http://localhost:3000/water-logs/today</code> API by using Postman / Insomnia
Input	-
Expected Behavior	it would retrieve daily log data from the database and return it to the user. Progress towards the daily goal can be seen from the daily log data. The number shown in the daily goal should be $(totalAmount - dailyGoal) * 100$
Actual Output	{

	<pre>"status": "success", "message": "successfully returned summary", "id": "11/11/2025", "data": { "dailyGoal": 3000, "totalAmount": 300, "progressPercentage": 10, "logs": { "37d385da-8cf5-4cab-a192-60f055df1fe2": { "amount": 100, "timestamp": "2025-11-11T10:00:33.346Z", "note": "Drank water today!" }, "7910a82f-8aaa-4917-b053-b193446ad20e": { "amount": 100, "timestamp": "2025-11-11T10:22:11.568Z", "note": "Drank water today!" }, "09ba86bd-864a-411d-adec-38c7719a6379": { "amount": 100, "timestamp": "2025-11-11T10:29:38.579Z", "note": "Drank water today!" } } }</pre>
Status	PASS
Testing Documentation	

GET

http://localhost:3000/water-logs/today

Params

Authorization

Headers (6)

Body

Scripts

Settings

Body

Cookies

Headers (7)

Test Results



{ } JSON

Preview

Visualize



```

1  {
2    "status": "success",
3    "message": "successfully returned summary",
4    "id": "11/11/2025",
5    "data": {
6      "dailyGoal": 3000,
7      "totalAmount": 300,
8      "progressPercentage": 10,
9      "logs": {
10       "37d385da-8cf5-4cab-a192-60f055df1fe2": {
11         "amount": 100,
12         "timestamp": "2025-11-11T10:00:33.346Z",
13         "note": "Drank water today!"
14       },
15       "7910a82f-8aaa-4917-b053-b193446ad20e": {
16         "amount": 100,
17         "timestamp": "2025-11-11T10:22:11.568Z",
18         "note": "Drank water today!"
19       },
20       "09ba86bd-864a-411d-adec-38c7719a6379": {
21         "amount": 100,
22         "timestamp": "2025-11-11T10:29:38.579Z",
23         "note": "Drank water today!"
24       }
25     }
26   }
27 }
```