

| 科目名 | 年度 | レポート番号 | クラス | 学籍番号 | 名前 |
|--------|------|--------|-----|----------|--------------|
| API 実習 | 2023 | 2 | B | 20122077 | Roger Marvin |

レポートは最大 5 ページ以内とします。ページ数や文字数よりも、わかりやすく書けているかどうか、点数アップの分かれ目です。

Google スプレッドシートをもとに API を作成し、下記を行ってください。

1. Google スプレッドシートをもとに作成した API について、以下を報告すること。

(ア) 作成した API の概要

→ カメラの情報の API です

(イ) どんなことに役立つかの説明 << 読んだ人が具体的なイメージを思い浮かべられるように。

→ カメラの初心者、好きな人がカメラについて調べて、情報を得られる

(ウ) 作成した Google スプレッドシートの URL

→ https://docs.google.com/spreadsheets/d/19chCp2Thtkh9pE2oc4_aOqh5X0t_dnGWvLTQAxgtyJY/edit#gid=0

(エ) API エンドポイントの URL

→ https://api.sssapi.app/2cJOTICQZYXjm9H3_1ekS

2. Microsoft Learn の「Node.js と Express を使用して Web API を構築する」に取り組み、以下を報告する。

(ア) 作成したものの説明

- Github Codespaces で簡単な WEB API を作成しました。「Training | Microsoft Learn」に基づいて index.js(app.js) と client.js を作成しました。Index.js ではメインとしてリクエストとレスポンスを管理するコードです。例えば、「**app.get("/users", (req,res)) =>**」と「**app.get("/products", (req,res)=>**」を使ってそれぞれの中身呼び出します。

```

← → ↺ //animated-dollop-699r6gp5j94r24pqq-3000.app.github.dev/users ↗

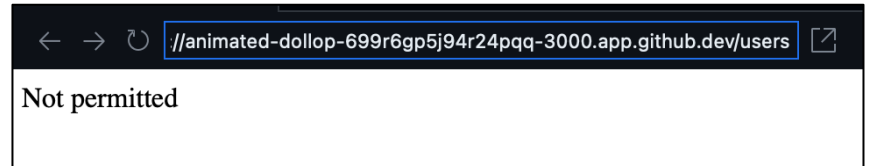
[{"id":1,"name":"Roger Marvin"}, {"id":2,"name":"Kevin Collin"}, {"id":3,"name":"Louis Martius"}]

← → ↺ imated-dollop-699r6gp5j94r24pqq-3000.app.github.dev/products ↗

[{"id":1,"name":"Katana"}, {"id":2,"name":"Magnum"}, {"id":3,"name":"Grimoire"}]
```

- また、function の isAuthorized を使って、暗語化を作りました。例えば、get /users のところで、isAuthorized をつけたら、ページをアクセスできなくなります。もちろん get/products にも isAuthorized をつけたら、ページをアクセスできなくなります。

```
20 app.get("/users", isAuthorized, (req,res)=>{
21   res.json([
22     {
23       id: 1,
24       name: "Roger Marvin"
25     },
26     {
27       id: 2,
28       name: "Kevin Collin"
29     },
30     {
31       id : 3,
32       name: "Louis Martius"
33     }
34   ]);
35 });
```



(イ) 自分が理解したこと →

「Training | Microsoft Learn」に基づいて、簡単な WEB API を作りました。このコードを書いた後、分かったことは API の基本のリクエストやレスポンスのコードの書き方、この API を暗号化する方法、REST のメソッドの「get」の使い方についてよく理解できました。しかし、この API のレスポンスは自分で決める（コードで書くこと ; データベースを使わずに）ために、まだ公開できない/他の人に使えないことを思っています。

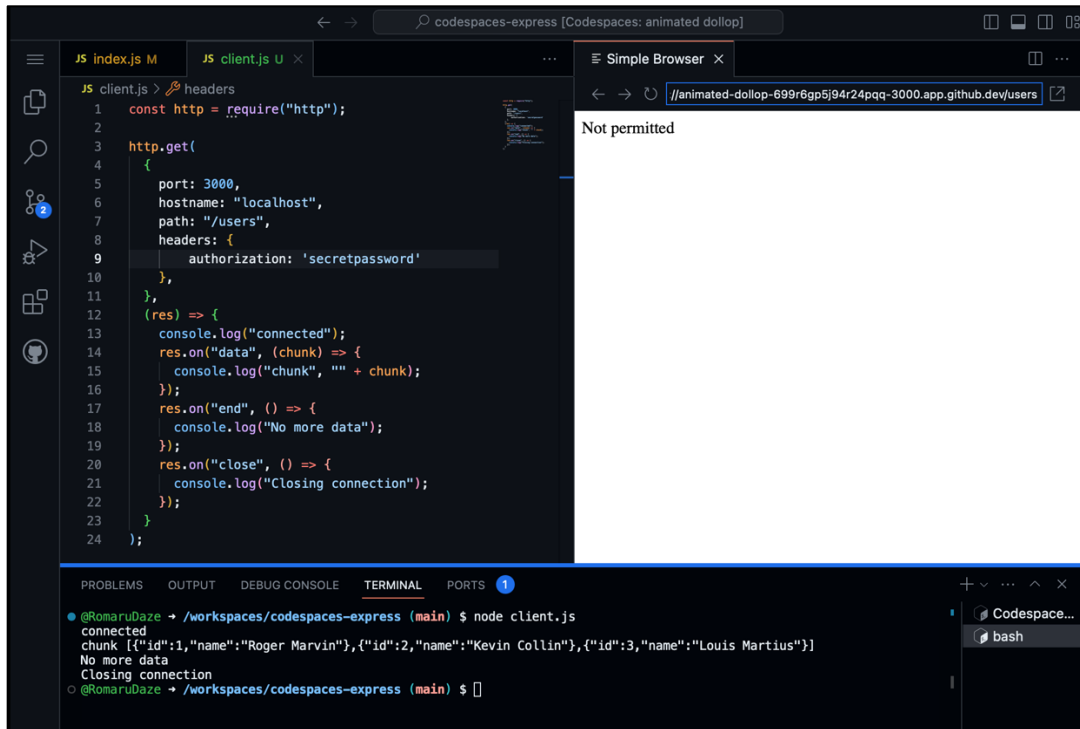
(ウ) どんなことに役立つか →

もしもデータベースからデータを取れて、見やすい場面で表示できたら、どんな情報（食べ物、ポケモン、ゲームなど）を他の人に役に立ちます。また、API 自体を作れば、色んなものを作ることができます。例えば、WEB 上の図鑑やゲームなどに作れます。

また、この知識を使って、日常的な問題（大学の課題、予定管理など）をアプリ作って、人生を楽しめます。

(工) 作成した WebAPI が動いていることがわかる画面ショットを貼り付けること →

Index.js



The screenshot shows a VS Code editor with two tabs: 'JS index.js M' and 'JS client.js U'. The 'client.js' tab is active, displaying the following code:

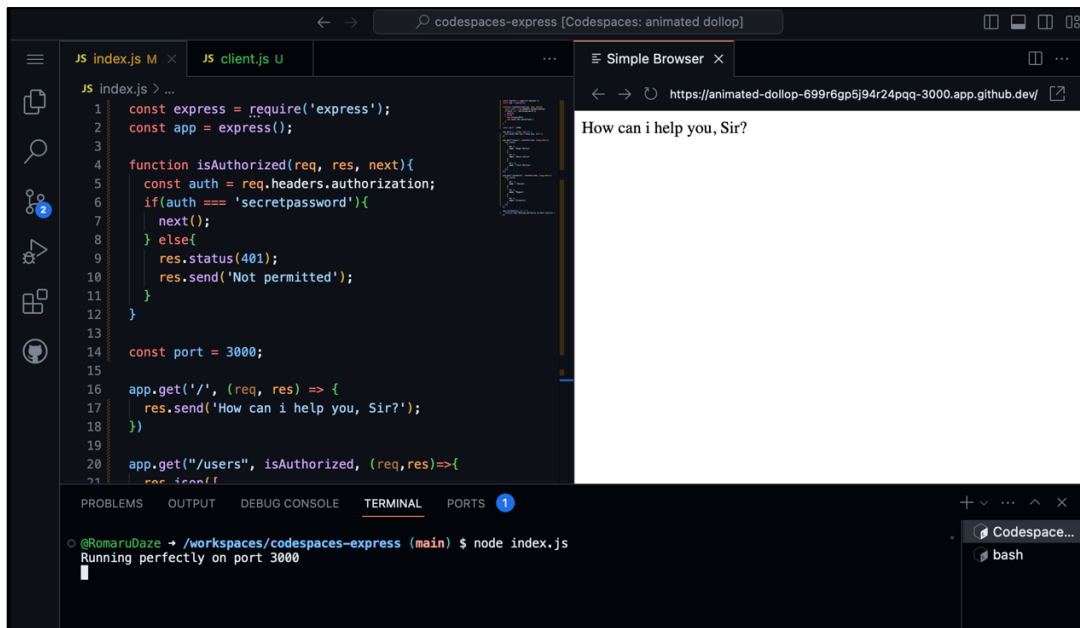
```
1 const http = require("http");
2
3 http.get(
4   {
5     port: 3000,
6     hostname: "localhost",
7     path: "/users",
8     headers: {
9       authorization: 'secretpassword'
10    },
11  },
12  (res) => {
13    console.log("connected");
14    res.on("data", (chunk) => {
15      console.log("chunk", "" + chunk);
16    });
17    res.on("end", () => {
18      console.log("No more data");
19    });
20    res.on("close", () => {
21      console.log("Closing connection");
22    });
23  }
24 );
```

Below the editor is a terminal window showing the command `node client.js` and its output:

```
@RomaruDaze + /workspaces/codespaces-express (main) $ node client.js
connected
chunk [{"id":1,"name":"Roger Marvin"},{"id":2,"name":"Kevin Collin"},{"id":3,"name":"Louis Martius"}]
No more data
Closing connection
@RomaruDaze + /workspaces/codespaces-express (main) $
```

To the right of the editor is a 'Simple Browser' window with the address `https://animated-dollop-699r6gp5j94r24pq-3000.app.github.dev/users`. The browser displays the message 'Not permitted'.

Client.js



The screenshot shows a VS Code editor with two tabs: 'JS index.js M' and 'JS client.js U'. The 'index.js' tab is active, displaying the following code:

```
1 const express = require('express');
2 const app = express();
3
4 function isAuthorized(req, res, next){
5   const auth = req.headers.authorization;
6   if(auth === 'secretpassword'){
7     next();
8   } else{
9     res.status(401);
10    res.send('Not permitted');
11  }
12 }
13
14 const port = 3000;
15
16 app.get('/', (req, res) => {
17   res.send('How can i help you, Sir?');
18 })
19
20 app.get("/users", isAuthorized, (req,res)=>{
21   res.json([
22     {id:1, name: "Roger Marvin"},
23     {id:2, name: "Kevin Collin"},
24     {id:3, name: "Louis Martius"}
25   ]);
26 });
```

Below the editor is a terminal window showing the command `node index.js` and its output:

```
@RomaruDaze + /workspaces/codespaces-express (main) $ node index.js
Running perfectly on port 3000
```

To the right of the editor is a 'Simple Browser' window with the address `https://animated-dollop-699r6gp5j94r24pq-3000.app.github.dev/`. The browser displays the message 'How can i help you, Sir?'.

(オ)「知識チェック」の結果について、画面ショットを貼り付けること →

知識チェック

200 XP

2分

それぞれの質問に最も適した回答を選んでください。その後、[回答を確認] を選択します。

1. Express を使用して Web アプリケーションを構築するために必要な手順は、次のどれですか? *

☐

アプリをインスタンス化し、ルートを作成し、ミドルウェアを設定し、エラー ハンドラーを設定し、サーバーをリスンする

☒

アプリをインスタンス化し、サーバーをリスンする

☐

✓ 正解です。アプリを起動して実行するには、これらの手順だけが必要です。ルートをいくつか構成することを強くお勧めします。

☐

アプリをインスタンス化し、ルートを作成し、サーバーをリスンする

☐

アプリをインスタンス化し、ルートを作成し、ミドルウェアを設定し、サーバーをリスンする

2. Express アプリから JSON 応答を送信する方法として、推奨されるのは次のどれですか? *

☒

応答オブジェクトで `json()` ヘルパー メソッドを呼び出す: `res.json({ content: ' ' })`

☐

✓ 正解です。JSON として応答を送信する方法はいくつもありますが、この方法が最も一般的であり、簡単に使用できます。

☐

`res.send({ content: ' ' })` を呼び出す

☐

`res.send(JSON.stringify({ content: ' ' }))` を呼び出す

☐

次のいずれかの方法を使用する: `res.type('json')`、`res.type('application/json')`、`res.contentType('application/json')`、`res.format({ 'application/json': function() { res.send({}) } })`

3. JSON データを含む Post 要求を処理するように Express を設定するにはどうすればよいですか? *

☐

`app.post(<route>, () =>{})` のように `post` メソッドを使用してルートを登録し、`req.body` オブジェクトから読み取る

☐

本文解析ミドルウェアを作成し、`app.post(<route>, () =>{})` のように `post` メソッドを使用してルートを登録して、`req.data` オブジェクトから読み取る

☒

先頭で `app.use(bodyParser.json())` を呼び出し、`app.post(<route>, () =>{})` のように `post` メソッドを使用してルートを登録して、`req.body` オブジェクトから読み取る

☐

✓ 正解。この呼び出しでは、受信データを JSON として解釈するように `bodyParser` が構成されます。

☐

先頭で `app.use(bodyParser.urlencoded({ extended: false }))` を呼び出し、`app.post(<route>, () =>{})` のように `post` メソッドを使用してルートを登録して、`req.body` オブジェクトから読み取る

次のユニット: まとめ