REST API

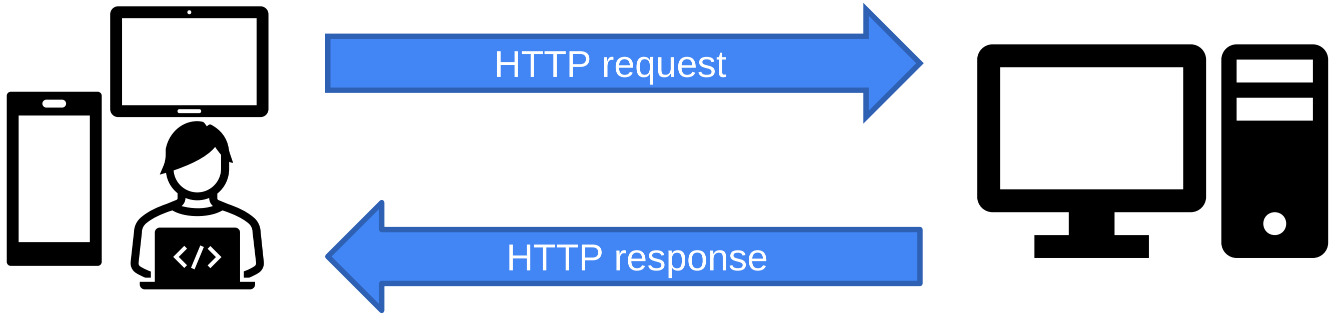
Asst.Prof.Dr.Nuengwong Tuaycharoen

Outline

1. REST API
2. API Development
3. API Security
4. OpenAPI with Swagger

REST API

The Client-Server Architecture

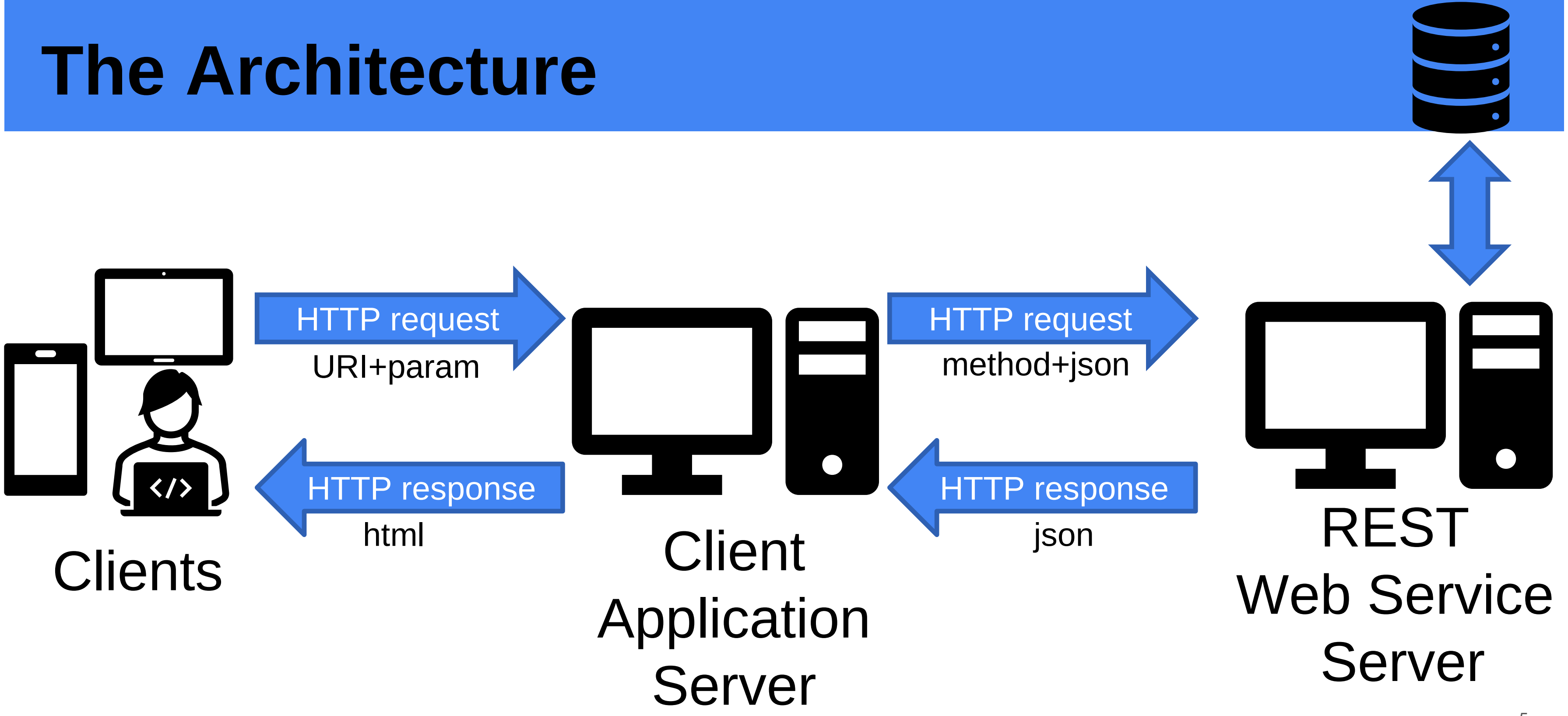


Clients

REST Web Service Server

https://ahmetozlu93.medium.com/mastering-rest-architecture-rest-architecture-details-e47ec659f6bc

The Architecture



HTTP Request Methods

GET	Retrieve Resource
POST	Submit Resource
PUT/PATCH	Update Resource
DELETE	Delete/Destroy Resource

RESTFUL API Standards

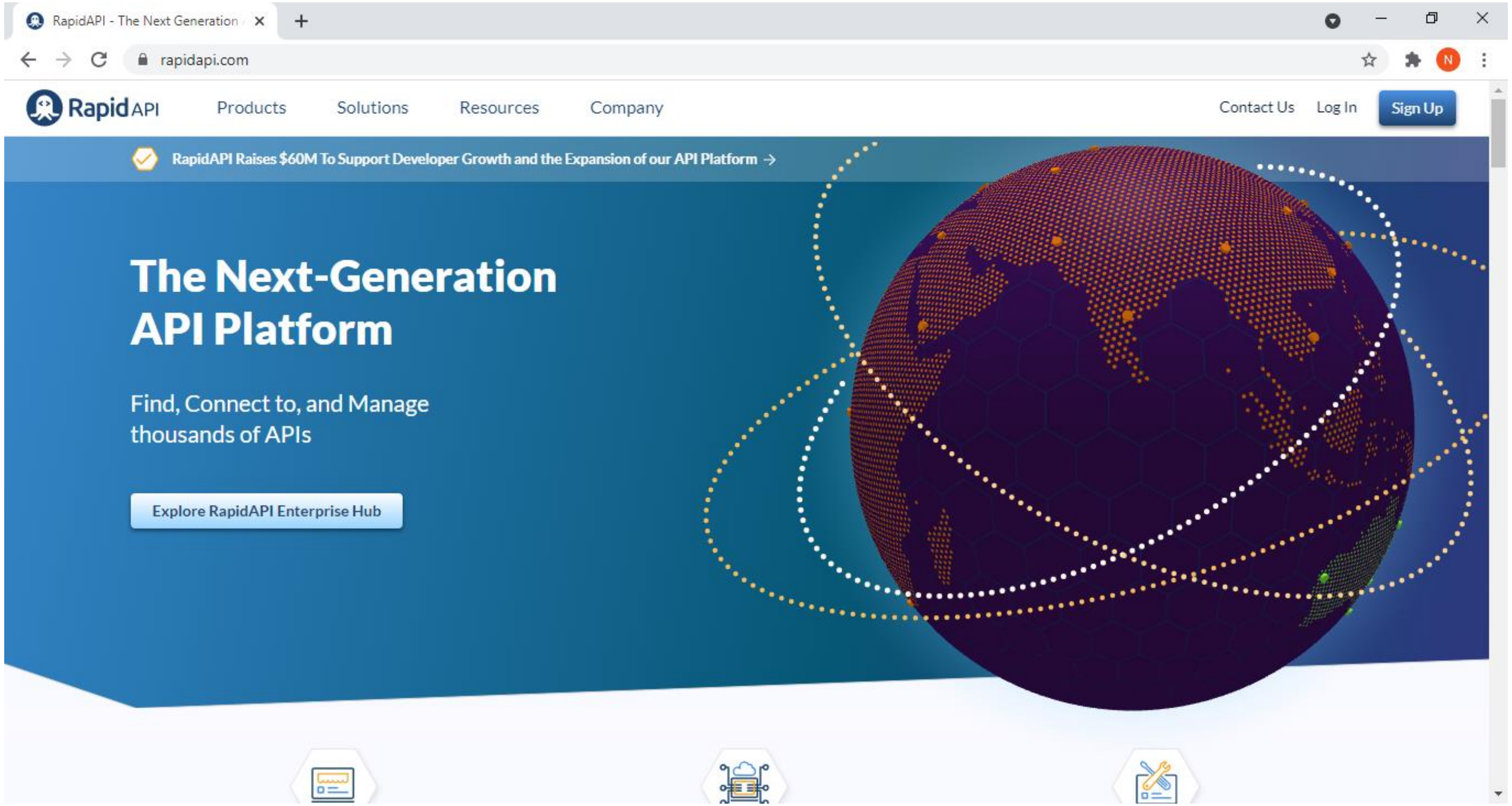
GET	/resources	Get resources
GET	/resources/1	Get resource with ID of 1
POST	/resources	Add a resource
PUT	/resources/1	Update resource with ID of 1
DELETE	/resources/1	Delete resource with ID of 1

HTTP response status codes

https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

- Informational responses (100–199)
- Successful responses (200–299)
  - 200 Success
  - 201 Created
  - 204 No Content
- Redirects (300–399)
  - 304 Not Modified
- Client errors (400–499)
  - 400 Bad Request
  - 401 Unauthorized
  - 404 Not Found
- Server errors (500–599)
  - 500 Internal Server Error

https://rapidapi.com/

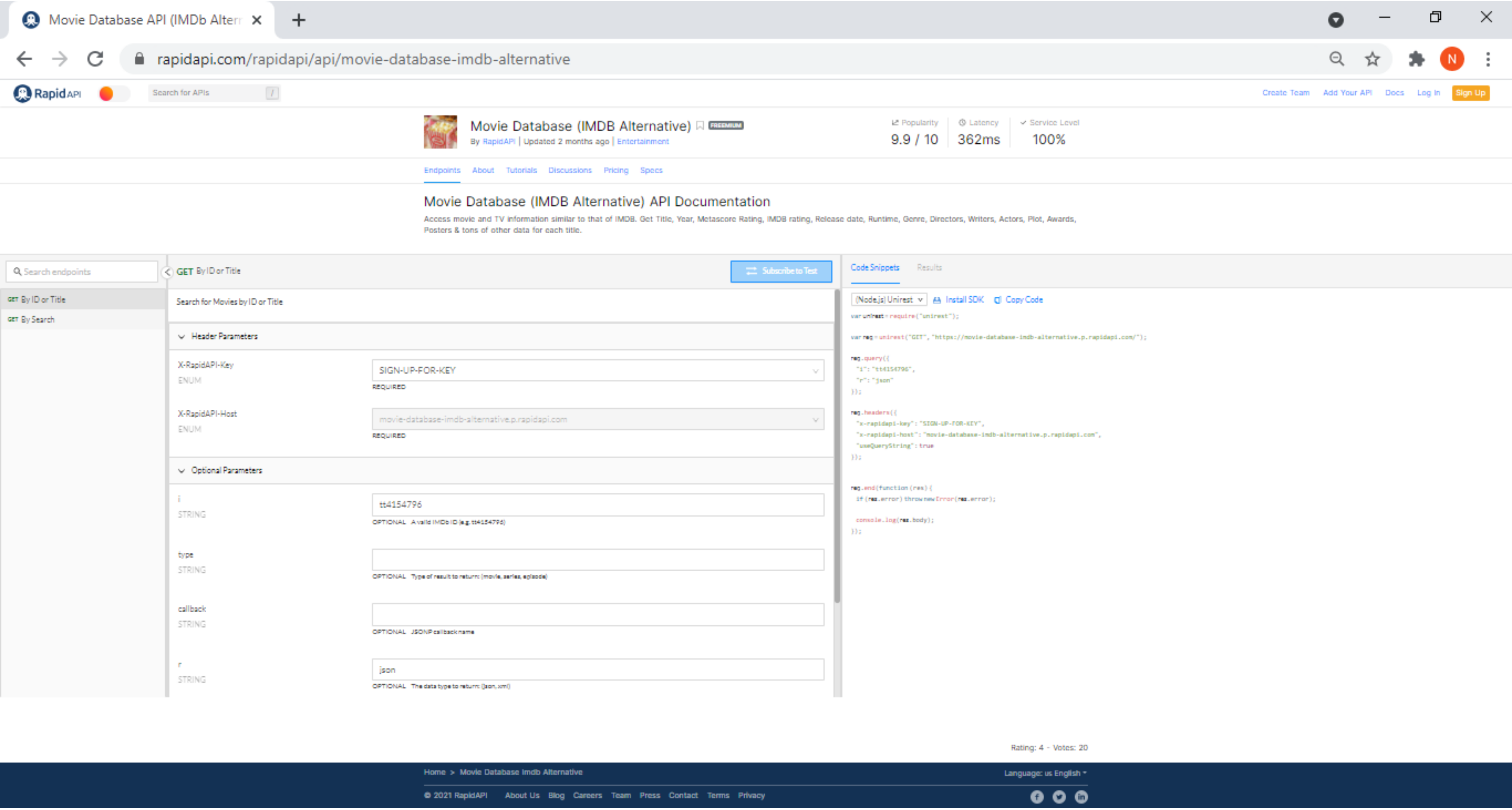


RapidAPI Raises $60M To Support Developer Growth and the Expansion of our API Platform →

The Next-Generation API Platform

Find, Connect to, and Manage thousands of APIs

Explore RapidAPI Enterprise Hub


Movie Database API (IMDb Altern
rapidapi.com/rapidapi/api/movie-database-imdb-alternative
Rapid API
Search for APIs
Create Team
Add Your API
Docs
Log In
Sign Up

Movie Database (IMDB Alternative) FREEMIUM

By RapidAPI | Updated 2 months ago | Entertainment

Popularity	Latency	Service Level
9.9 / 10	362ms	100%

Movie Database (IMDB Alternative) API Documentation

Access movie and TV information similar to that of IMDB. Get Title, Year, Metascore Rating, IMDB rating, Release date, Runtime, Genre, Directors, Writers, Actors, Plot, Awards, Posters & tons of other data for each title.

Search for Movies by ID or Title

Header Parameters

X-RapidAPI-Key ENUM	SIGN-UP-FOR-KEY REQUIRED
X-RapidAPI-Host ENUM	movie-database-imdb-alternative.p.rapidapi.com REQUIRED

Optional Parameters

i STRING	tt4154796 OPTIONAL A valid IMDb ID (e.g. tt4154796)
type STRING	 OPTIONAL Type of result to return: (movie, series, episode)
callback STRING	 OPTIONAL JSONP callback name
r STRING	json OPTIONAL The data type to return: (json, xml)

```
var unirest = require("unirest");

var req = unirest("GET", "https://movie-database-imdb-alternative.p.rapidapi.com/");

req.query({
  "i": "tt4154796",
  "r": "json"
});

req.headers({
  "x-rapidapi-key": "SIGN-UP-FOR-KEY",
  "x-rapidapi-host": "movie-database-imdb-alternative.p.rapidapi.com",
  "useQueryString": true
});


req.end(function (res) {
  if (res.error) throw new Error(res.error);

  console.log(res.body);
});
```

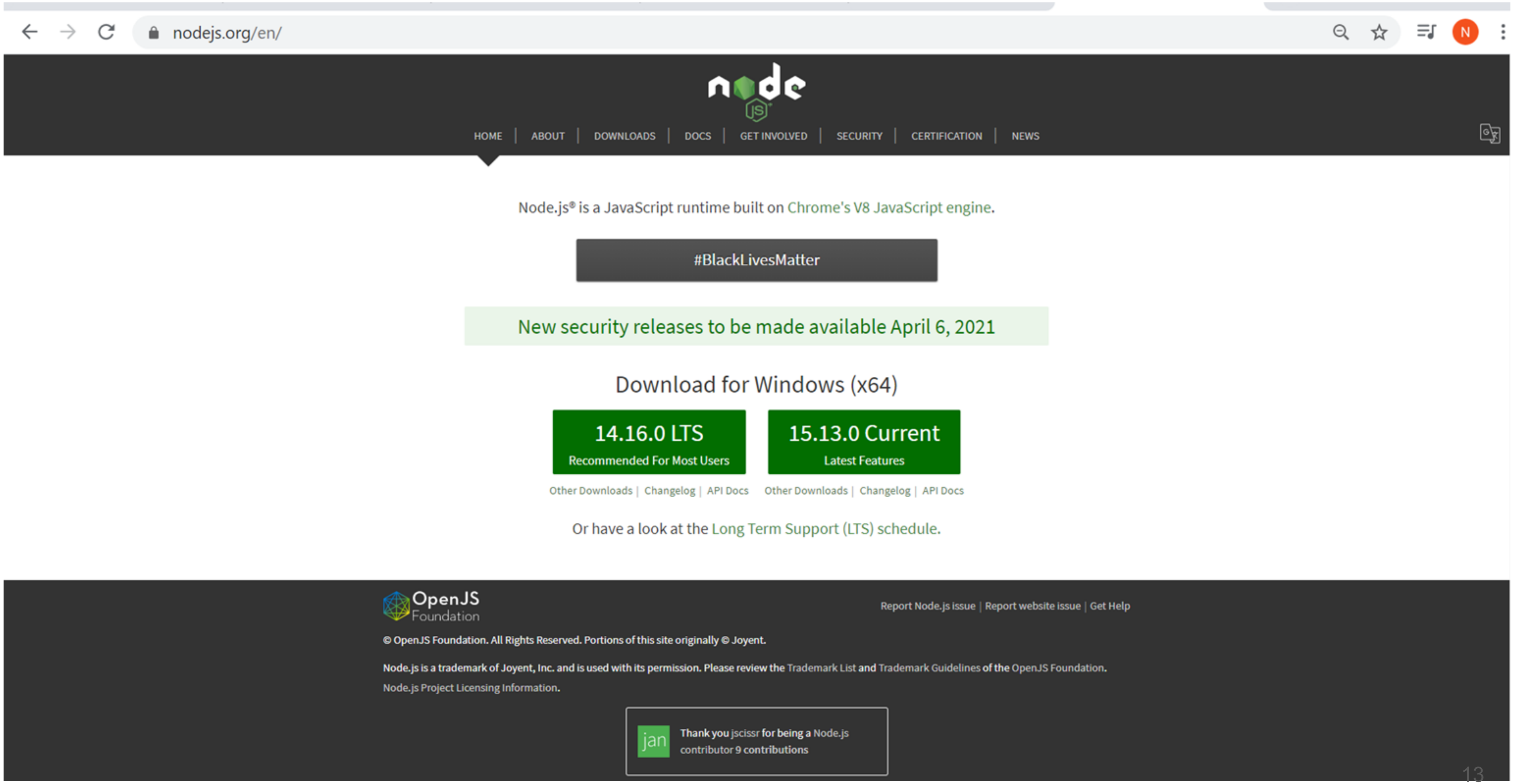
Rating: 4 - Votes: 20

API Development

Ref: https://github.com/WebDevSimplified/Your-First-Node-REST-API
https://www.youtube.com/watch?v=fgTGADljAeg

Environment Setup [35 min]

1. Node.js [nodejs.org]
2. VS Code [code.visualstudio.com/download]


nodejs.org/en/
HOME | ABOUT | DOWNLOADS | DOCS | GET INVOLVED | SECURITY | CERTIFICATION | NEWS

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine.

#BlackLivesMatter

New security releases to be made available April 6, 2021

Download for Windows (x64)

14.16.0 LTS
Recommended For Most Users

15.13.0 Current
Latest Features

Other Downloads | Changelog | API Docs   Other Downloads | Changelog | API Docs

Or have a look at the Long Term Support (LTS) schedule.

OpenJS Foundation

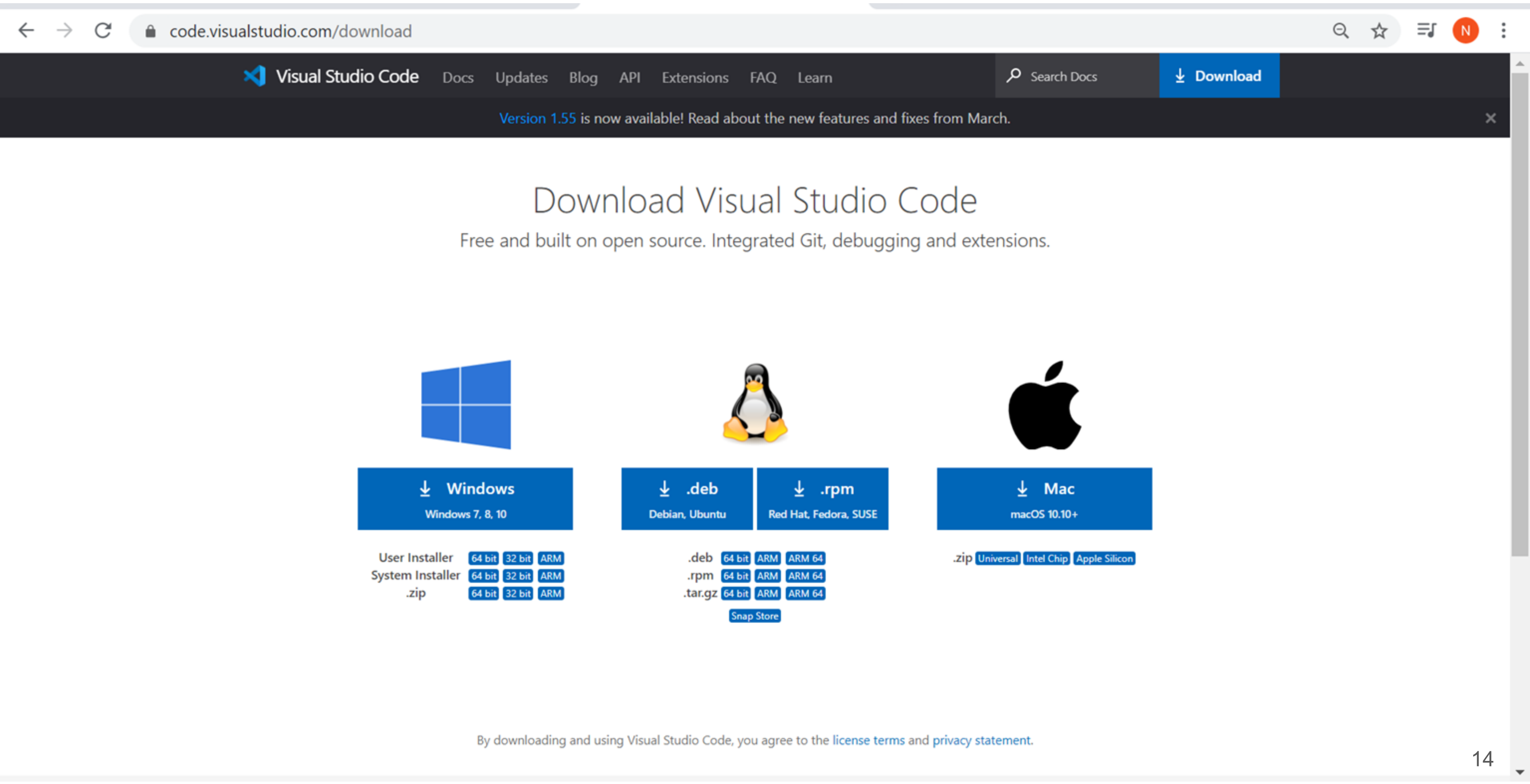
Report Node.js issue | Report website issue | Get Help

© OpenJS Foundation. All Rights Reserved. Portions of this site originally © Joyent.

Node.js is a trademark of Joyent, Inc. and is used with its permission. Please review the Trademark List and Trademark Guidelines of the OpenJS Foundation.

Node.js Project Licensing Information.

jan Thank you jscissr for being a Node.js contributor 9 contributions



code.visualstudio.com/download

Visual Studio Code Docs Updates Blog API Extensions FAQ Learn Search Docs Download

Version 1.55 is now available! Read about the new features and fixes from March.

Download Visual Studio Code

Free and built on open source. Integrated Git, debugging and extensions.

Windows
Windows 7, 8, 10

.deb
Debian, Ubuntu

.rpm
Red Hat, Fedora, SUSE

Mac
macOS 10.10+

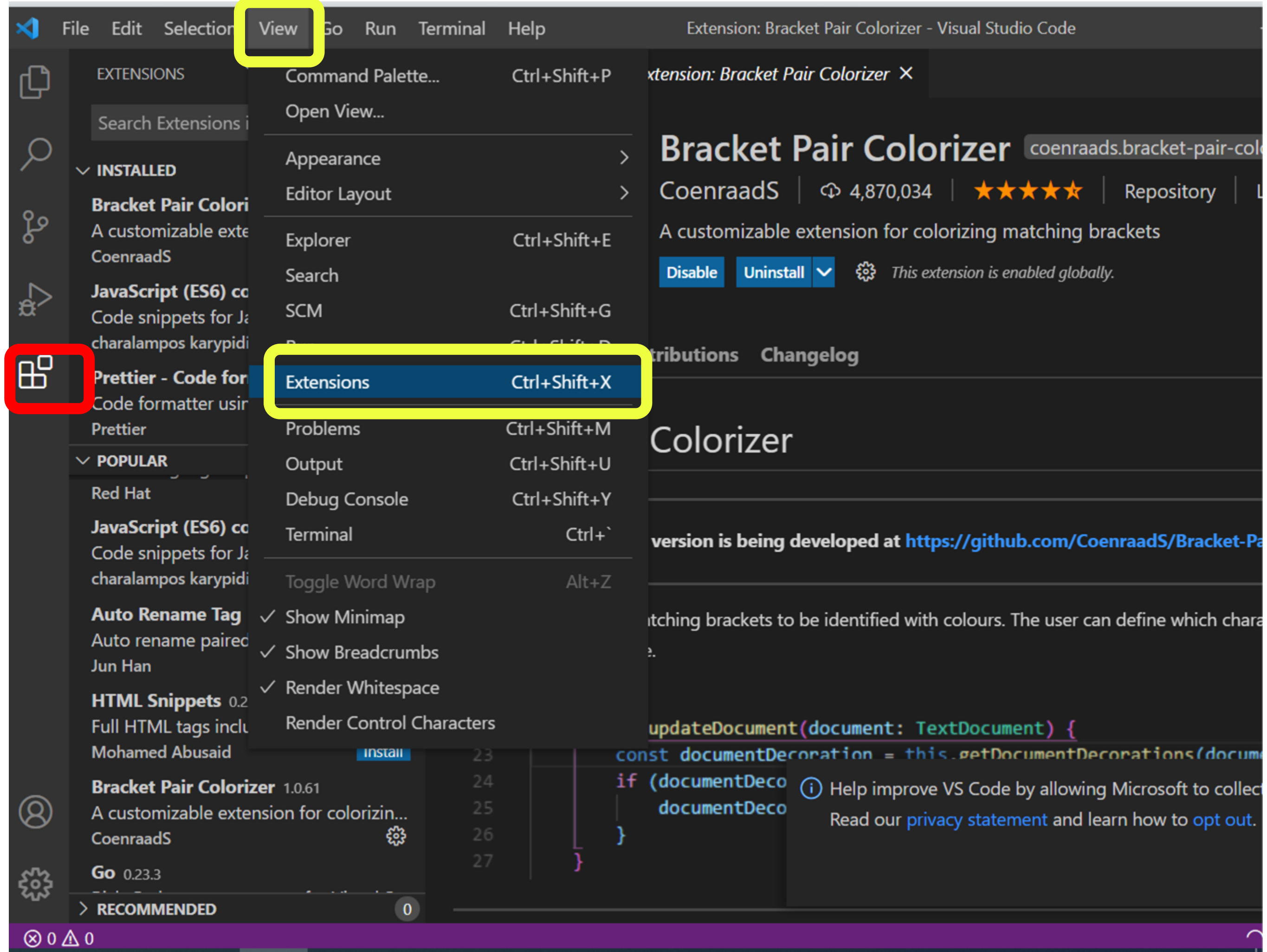
User Installer	64 bit	32 bit	ARM
System Installer	64 bit	32 bit	ARM
.zip	64 bit	32 bit	ARM

.deb	64 bit	ARM	ARM 64
.rpm	64 bit	ARM	ARM 64
.tar.gz	64 bit	ARM	ARM 64

Snap Store

.zip Universal Intel Chip Apple Silicon

By downloading and using Visual Studio Code, you agree to the license terms and privacy statement.



A customizable extension for colorizing matching brackets

This extension is enabled globally.

version is being developed at https://github.com/CoenraadS/Bracket-Pa

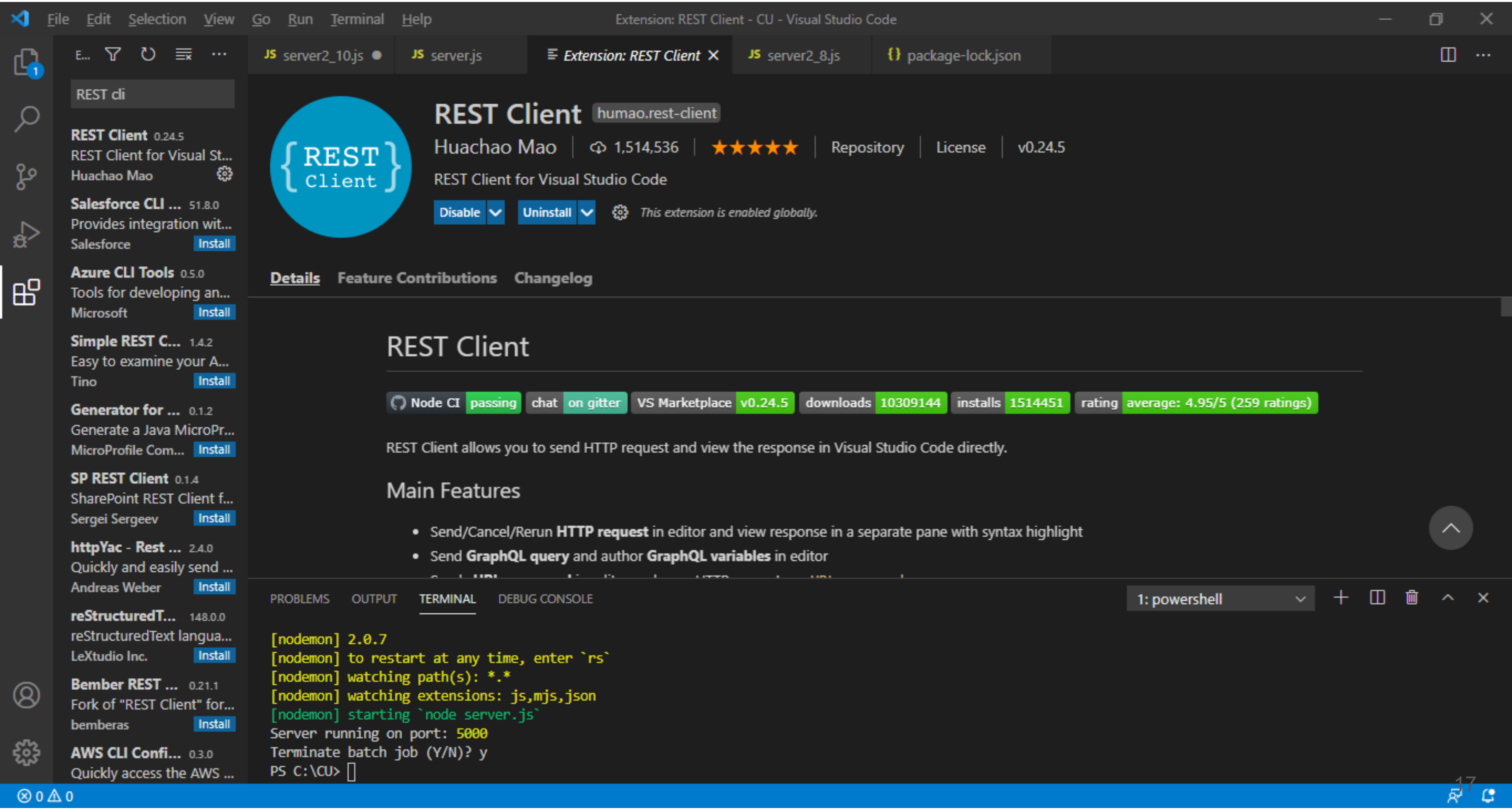
tching brackets to be identified with colours. The user can define which chara

```
updateDocument(document: TextDocument) {
    const documentDecoration = this.getDocumentDecorations(docum
    if (documentDeco
        documentDeco
    }
}
```

Help improve VS Code by allowing Microsoft to collec
Read our privacy statement and learn how to opt out.

Install VS Code Extensions

1. Bracket Pair Colorizer
2. DotENV
3. ExpressSnippet
4. JavaScript (ES6) code snippets
5. Prettier - Code formatter
6. REST Client


File Edit Selection View Go Run Terminal Help
Extension: REST Client - CU - Visual Studio Code
server2_10.js
server.js
Extension: REST Client
server2_8.js
package-lock.json
REST cli
REST Client 0.24.5
REST Client for Visual St...
Huachao Mao
Salesforce CLI ... 51.8.0
Provides integration wit...
Salesforce Install
Azure CLI Tools 0.5.0
Tools for developing an...
Microsoft Install
Simple REST C... 1.4.2
Easy to examine your A...
Tino Install
Generator for ... 0.1.2
Generate a Java MicroPr...
MicroProfile Com... Install
SP REST Client 0.1.4
SharePoint REST Client f...
Sergei Sergeev Install
httpYac - Rest ... 2.4.0
Quickly and easily send ...
Andreas Weber Install
reStructuredT... 148.0.0
reStructuredText langua...
LeXtudio Inc. Install
Bember REST ... 0.21.1
Fork of "REST Client" for...
bemberas Install
AWS CLI Confi... 0.3.0
Quickly access the AWS ...
REST Client humao.rest-client
Huachao Mao 1,514,536 ★★★★★ Repository License v0.24.5
REST Client for Visual Studio Code
Disable Uninstall This extension is enabled globally.
Details Feature Contributions Changelog
Node CI passing chat on gitter VS Marketplace v0.24.5 downloads 10309144 installs 1514451 rating average: 4.95/5 (259 ratings)
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
1: powershell
⊗ 0 ⚠ 0

REST Client

REST Client allows you to send HTTP request and view the response in Visual Studio Code directly.

Main Features

- Send/Cancel/Rerun **HTTP request** in editor and view response in a separate pane with syntax highlight
- Send **GraphQL query** and author **GraphQL variables** in editor

```
[nodemon] 2.0.7
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node server.js`
Server running on port: 5000
Terminate batch job (Y/N)? y
PS C:\CU>
```

Download the sample project

- https://drive.google.com/file/d/1cwIyVacJFeCyvIwZVUZP61VEpUOte_Pd/view?usp=sharing
- Unzip the file (i.e. C:\CU\)
- In VS Code, open the folder
- Run these commands in your VS Code terminal

1. npm init
2. npm install -D nodemon
3. npm install express dotenv mongoose
4. npm start

route.rest: Route Structure

GET/POST/PUT/DELETE

```
GET http://localhost:5000/subscribers
GET http://localhost:5000/subscribers/:id
POST http://localhost:5000/subscribers
DELETE http://localhost:5000/subscribers/:id
PATCH  http://localhost:5000/subscribers/:id
```

Package.json

```json
{
  "name": "firstapiproject",
  "version": "1.0.0",
  "main": "server.js",
  "scripts": {
    "devStart": "nodemon server.js",
    "start": "nodemon server.js"
  },
  "author": "ohm",
  "license": "ISC",
  "dependencies": {
    "express": "^4.18.1",
    "mongoose": "^6.5.4"
  },
  "devDependencies": {
    "dotenv": "^8.6.0",
    "nodemon": "^1.19.4"
  },
  "description": ""
}
```

server.js

```javascript
require('dotenv').config({path:'./config.env'});
const express = require('express')
const app = express()
const mongoose = require('mongoose’)
mongoose.set('strictQuery', true);//suppress deprecation warning
mongoose.connect(process.env.DATABASE_URL)

const db = mongoose.connection
db.on('error', (error) => console.error(error))
db.once('open', () => console.log('Connected to Database'))

app.use(express.json())

const subscribersRouter = require('./routes/subscribers')
app.use('/subscribers', subscribersRouter)

app.listen(process.env.PORT, () => console.log('Server Started'))
```

config.env

```
NODE_ENV=development
PORT=5000

DATABASE_URL=mongodb+srv://ohm123:ohm123@traversymedia.o0h
i2.mongodb.net/firstAPIproj?retryWrites=true&w=majority
```

models/subscriber.js

```
const mongoose = require('mongoose')

const subscriberSchema = new mongoose.Schema({
  name: {
    type: String,
    required: true
  },
  subscribedToChannel: {
    type: String,
    required: true
  },
  subscribeDate: {
    type: Date,
    required: true,
    default: Date.now
  }
})

module.exports = mongoose.model('Subscriber', subscriberSchema)
```

routes/subscribers.js (1/6)

```
const express = require('express')
const router = express.Router()
const Subscriber = require('../models/subscriber')

// Getting all
router.get('/', async (req, res) => {
  try {
    const subscribers = await Subscriber.find()
    res.json(subscribers)
  } catch (err) {
    res.status(500).json({ message: err.message })
  }
})
```

routes/subscribers.js (2/6)

```
// Getting One
router.get('/:id', getSubscriber, (req, res) => {
  res.json(res.subscriber)
})
```

routes/subscribers.js (3/6)

```
// Creating one
router.post('/', async (req, res) => {
  const subscriber = new Subscriber({
    name: req.body.name,
    subscribedToChannel: req.body.subscribedToChannel
  })
  try {
    const newSubscriber = await subscriber.save()
    res.status(201).json(newSubscriber)
  } catch (err) {
    res.status(400).json({ message: err.message })
  }
})
```

routes/subscribers.js (4/6)

```
async function getSubscriber(req, res, next) {
  let subscriber
  try {
    subscriber = await Subscriber.findById(req.params.id)
    if (subscriber == null) {
      return res.status(404).json({ message: 'Cannot find subscriber' })
    }
  } catch (err) {
    return res.status(500).json({ message: err.message })
  }

  res.subscriber = subscriber
  next()
}

module.exports = router
```

routes/subscribers.js (5/6)

```
// Deleting One
router.delete('/:id', getSubscriber, async (req, res) => {
  try {
    await res.subscriber.remove()
    res.json({ message: 'Deleted Subscriber' })
  } catch (err) {
    res.status(500).json({ message: err.message })
  }
})
```

routes/subscribers.js (6/6)

```
// Updating One
router.patch('/:id', getSubscriber, async (req, res) => {
  if (req.body.name != null) {
    res.subscriber.name = req.body.name
  }
  if (req.body.subscribedToChannel != null) {
    res.subscriber.subscribedToChannel = req.body.subscribedToChannel
  }
  try {
    const updatedSubscriber = await res.subscriber.save()
    res.json(updatedSubscriber)
  } catch (err) {
    res.status(400).json({ message: err.message })
  }
})
```

route.rest : Call the API

```
GET http://localhost:5000/subscribers

###
GET http://localhost:5000/subscribers/6
09bd8671452242d88d36e36

###
POST http://localhost:5000/subscribers
Content-Type: application/json

{
    "name": "Ken Nakarin",
    "subscribedToChannel":"THE STANDARD
"
}
```

```
###
DELETE http://localhost:5000/subsc
ribers/609bd8671452242d88d36e36

###
PATCH http://localhost:5000/subscr
ibers/609bd8671452242d88d36e36
Content-Type: application/json

{
    "name": "Jane Dawn",
    "subscribedToChannel":"THE STA
NDARD"
}
```

API Security

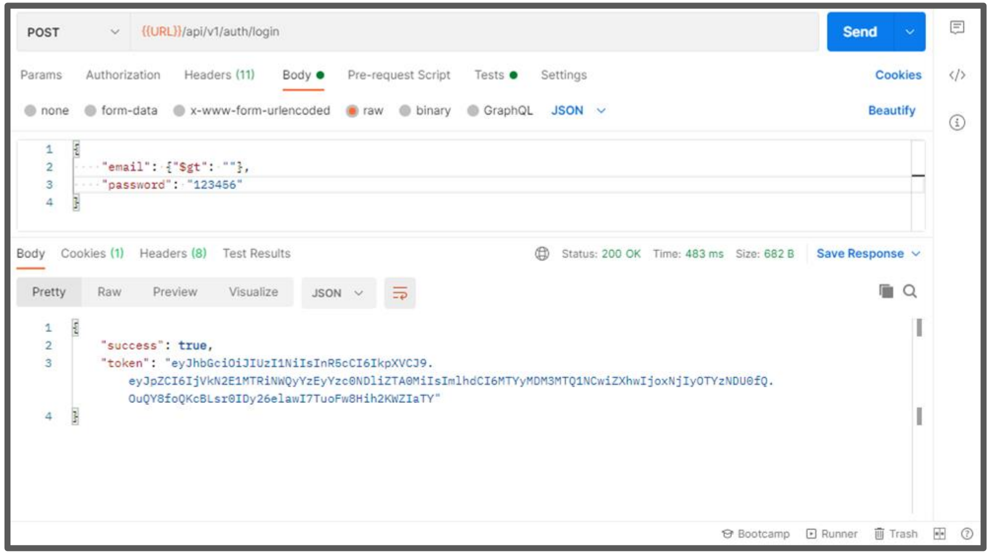
Ref: https://www.udemy.com/course/nodejs-api-masterclass/

1. Logout To Clear Token Cookie

```
//@desc     Log user out / clear cookie
//@route    GET /api/v1/auth/logout
//@access   Private
exports.logout=asyncHandler(async(req,res,next)=>{
    res.cookie('token','none',{
        expires: new Date(Date.now()+ 10*1000),
        httpOnly:true
    });

    res.status(200).json({
        success:true,
        data:{}
    });
});
```

2. Prevent NoSQL Injection & Sanitize Data

> npm i express-mongo-sanitize



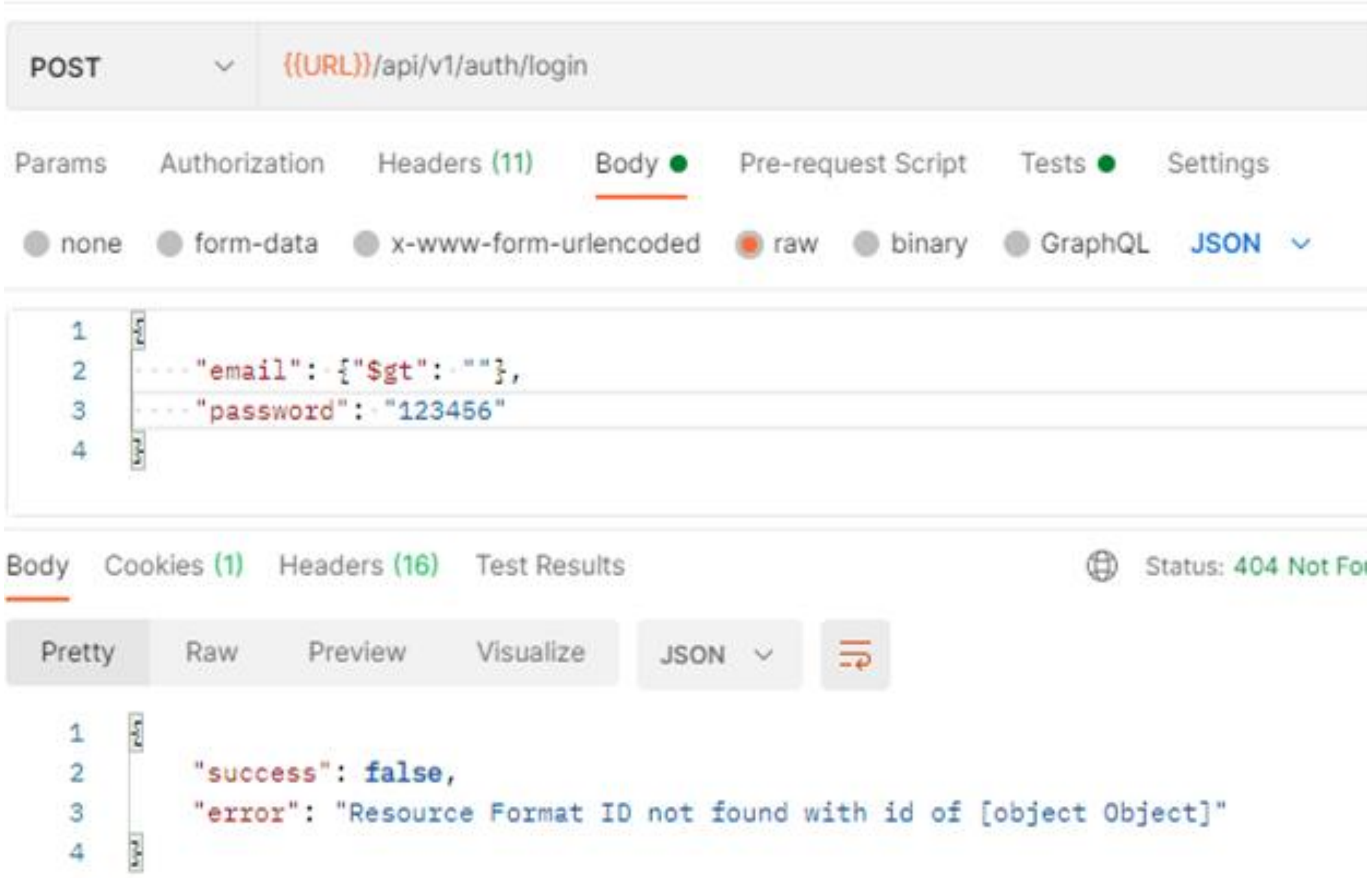
```
{{URL}}/api/v1/auth/login
```

```
{
    "email": {"$gt": ""},
    "password": "123456"
}
```

Status: 200 OK Time: 483 ms Size: 682 B

```
{
    "success": true,
    "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
        eyJpZCI6IjVkN2E1MTRiNWQyYzEyYzc0NDliZTA0MiIsImlhdCI6MTYyMDM3MTQ1NCwiZXhwIjoxNjIyOTYzNDU0fQ.
        OuQY8foQKcBLsr0IDy26elawI7TuoFw8Hih2KWZIaTY"
}
```

```
//server.js
const mongoSanitize=require('express-mongo-sanitize');
…

//Sanitize data
app.use(mongoSanitize());
```



```
{{URL}}/api/v1/auth/login
```

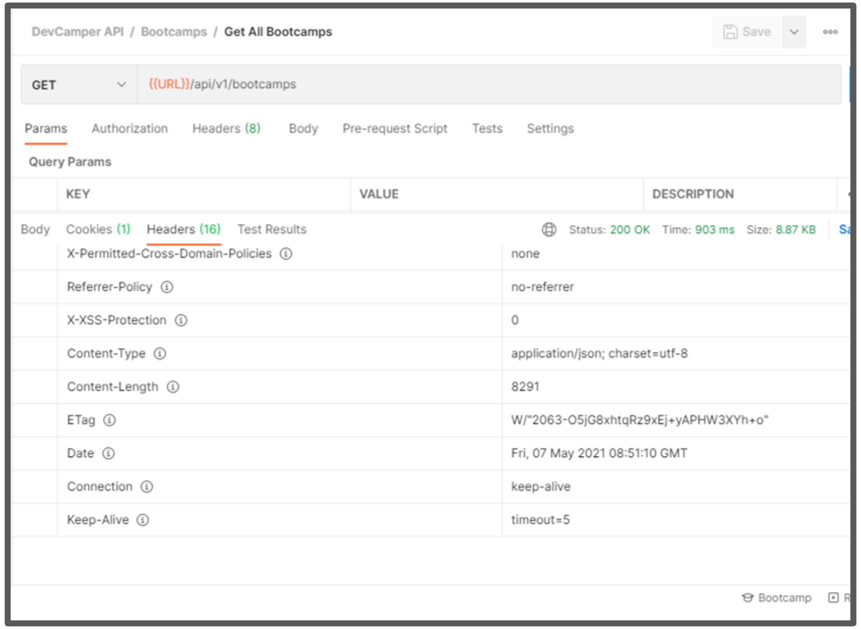
```
{
    "email": {"$gt": ""},
    "password": "123456"
}
```

Status: 404 Not Fo

```
{
    "success": false,
    "error": "Resource Format ID not found with id of [object Object]"
}
```

3. Security Headers

```
> npm i helmet
```



DevCamper API / Bootcamps / Get All Bootcamps

GET {{URL}}/api/v1/bootcamps

Params Authorization Headers (8) Body Pre-request Script Tests Settings

Query Params

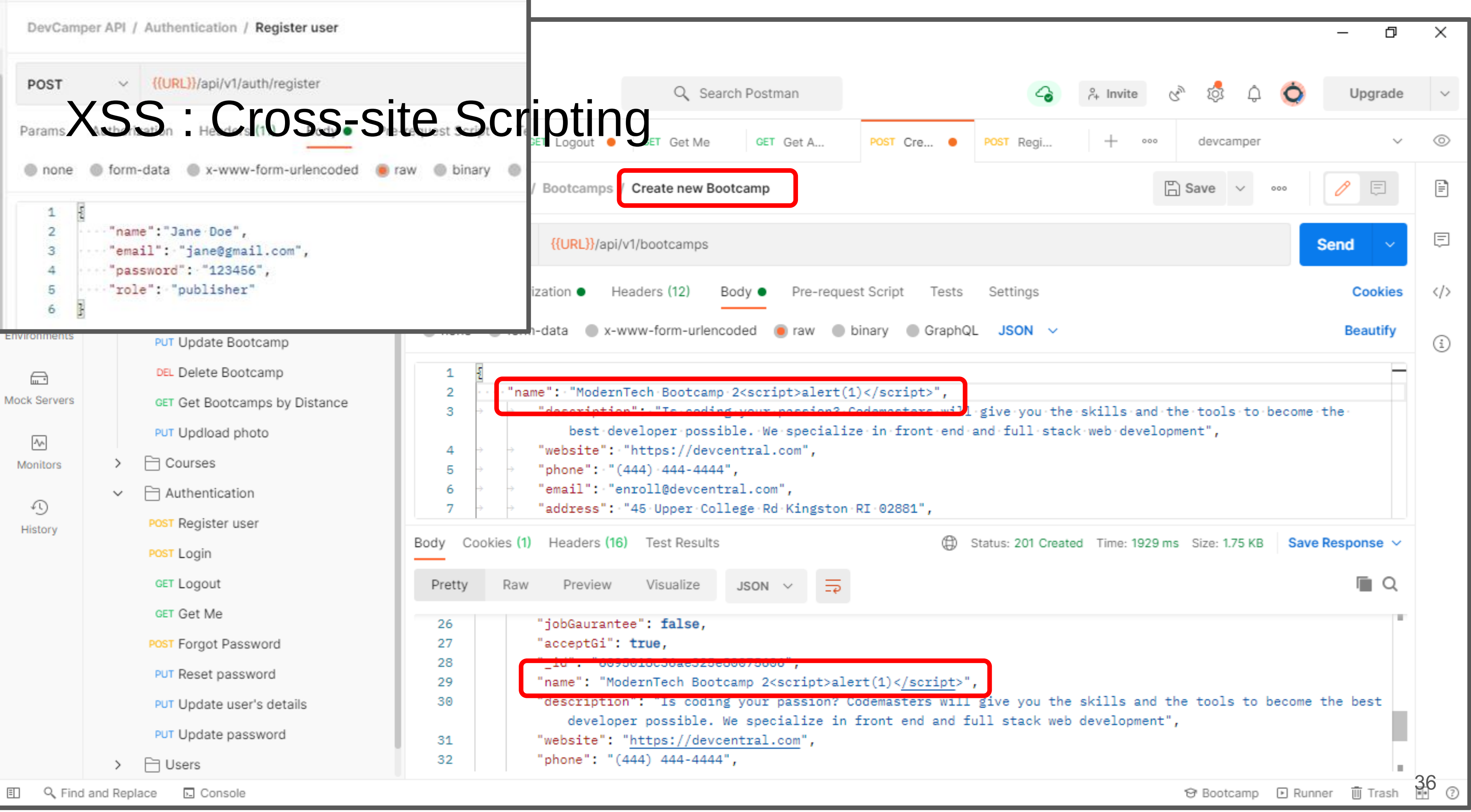
KEY	VALUE	DESCRIPTION

Body Cookies (1) Headers (16) Test Results — Status: 200 OK Time: 903 ms Size: 8.87 KB

X-Permitted-Cross-Domain-Policies	none
Referrer-Policy	no-referrer
X-XSS-Protection	0
Content-Type	application/json; charset=utf-8
Content-Length	8291
ETag	W/"2063-O5jG8xhtqRz9xEj+yAPHW3XYh+o"
Date	Fri, 07 May 2021 08:51:10 GMT
Connection	keep-alive
Keep-Alive	timeout=5

```
//server.js
const helmet=require('helmet');
...
//Set security headers
app.use(helmet());
```

XSS : Cross-site Scripting



DevCamper API / Authentication / Register user

POST {{URL}}/api/v1/auth/register

Body: raw

```
{
    "name":"Jane Doe",
    "email": "jane@gmail.com",
    "password": "123456",
    "role": "publisher"
}
```

Bootcamps / Create new Bootcamp

POST {{URL}}/api/v1/bootcamps

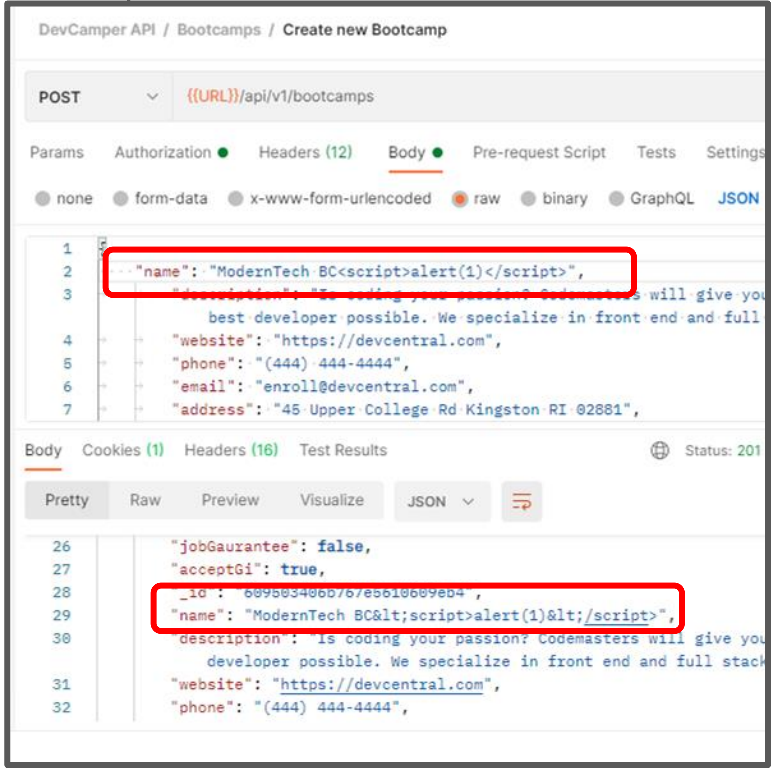
```
{
    "name": "ModernTech Bootcamp 2<script>alert(1)</script>",
        "description": "Is coding your passion? Codemasters will give you the skills and the tools to become the
            best developer possible. We specialize in front end and full stack web development",
        "website": "https://devcentral.com",
        "phone": "(444) 444-4444",
        "email": "enroll@devcentral.com",
        "address": "45 Upper College Rd Kingston RI 02881",
```

Status: 201 Created Time: 1929 ms Size: 1.75 KB

```
        "jobGaurantee": false,
        "acceptGi": true,
        "_id": "[illegible]",
        "name": "ModernTech Bootcamp 2<script>alert(1)</script>",
        "description": "Is coding your passion? Codemasters will give you the skills and the tools to become the best
            developer possible. We specialize in front end and full stack web development",
        "website": "https://devcentral.com",
        "phone": "(444) 444-4444",
```

4. XSS Protection

>npm i xss-clean



DevCamper API / Bootcamps / **Create new Bootcamp**

POST {{URL}}/api/v1/bootcamps

```
{
    "name": "ModernTech BC<script>alert(1)</script>",
        "description": "Is coding your passion? Codemasters will give you
            best developer possible. We specialize in front end and full
        "website": "https://devcentral.com",
        "phone": "(444) 444-4444",
        "email": "enroll@devcentral.com",
        "address": "45 Upper College Rd Kingston RI 02881",
```

```
        "jobGaurantee": false,
        "acceptGi": true,
        "_id": "609503406b767e5610609eb4",
        "name": "ModernTech BC&lt;script>alert(1)&lt;/script>",
        "description": "Is coding your passion? Codemasters will give you
            developer possible. We specialize in front end and full stack
        "website": "https://devcentral.com",
        "phone": "(444) 444-4444",
```

server.js

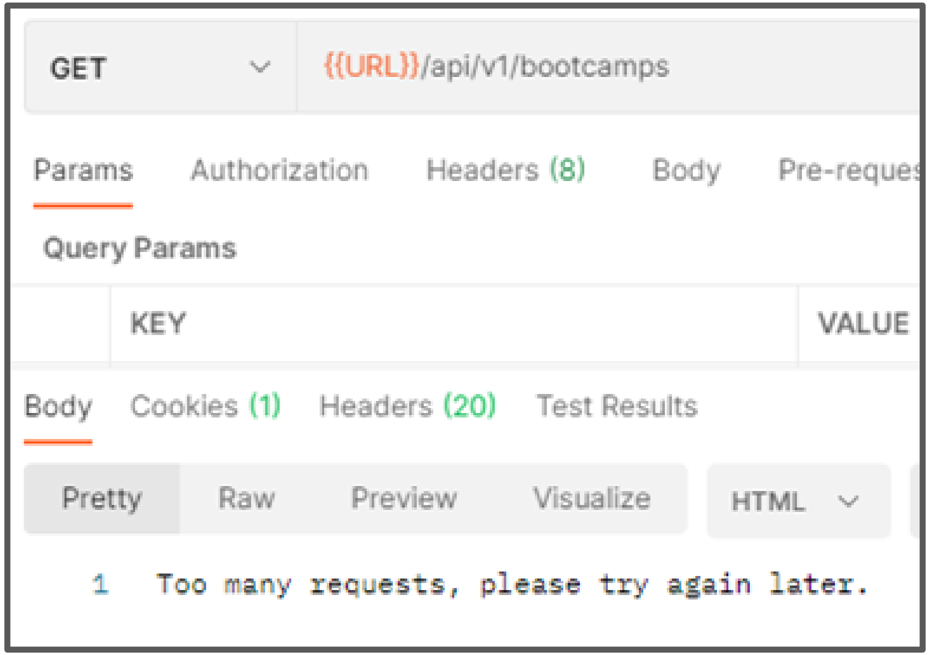
```
const xss=require('xss-clean');


//Prevent XSS attacks
app.use(xss());
```

5. Rate Limiting & HPP

HPP: prevent HTTP parameter polution attack

http://www.aaa.com/search?p=cat&p=dog&p=pig&...

> npm i express-rate-limit hpp



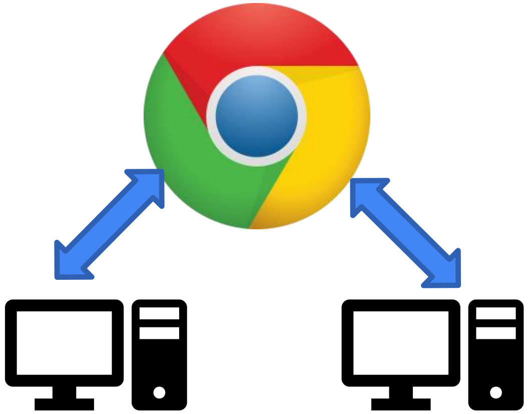
```
Too many requests, please try again later.
```

server.js

```
const rateLimit=require('express-
rate-limit');
const hpp=require('hpp');
…
//Rate Limiting
const limiter=rateLimit({
    windowsMs:10*60*1000,//10 mins
    max: 100
});
app.use(limiter);
```

6. CORS: Cross-Origin Resource Sharing

```
> npm i cors
```



```
http://www.A.com
```

```
Access-Control-Allow-Origin: http://www.A.com
```

server.js

```
const cors=require('cors');

…

//Enable CORS

app.use(cors());
```

https://expressjs.com/en/resources/middleware/cors.html

Creating OpenAPI Document with Swagger

https://github.com/satansdeer/swagger-api-library
https://www.youtube.com/watch?v=S8kmHtQeflo

OpenAPI

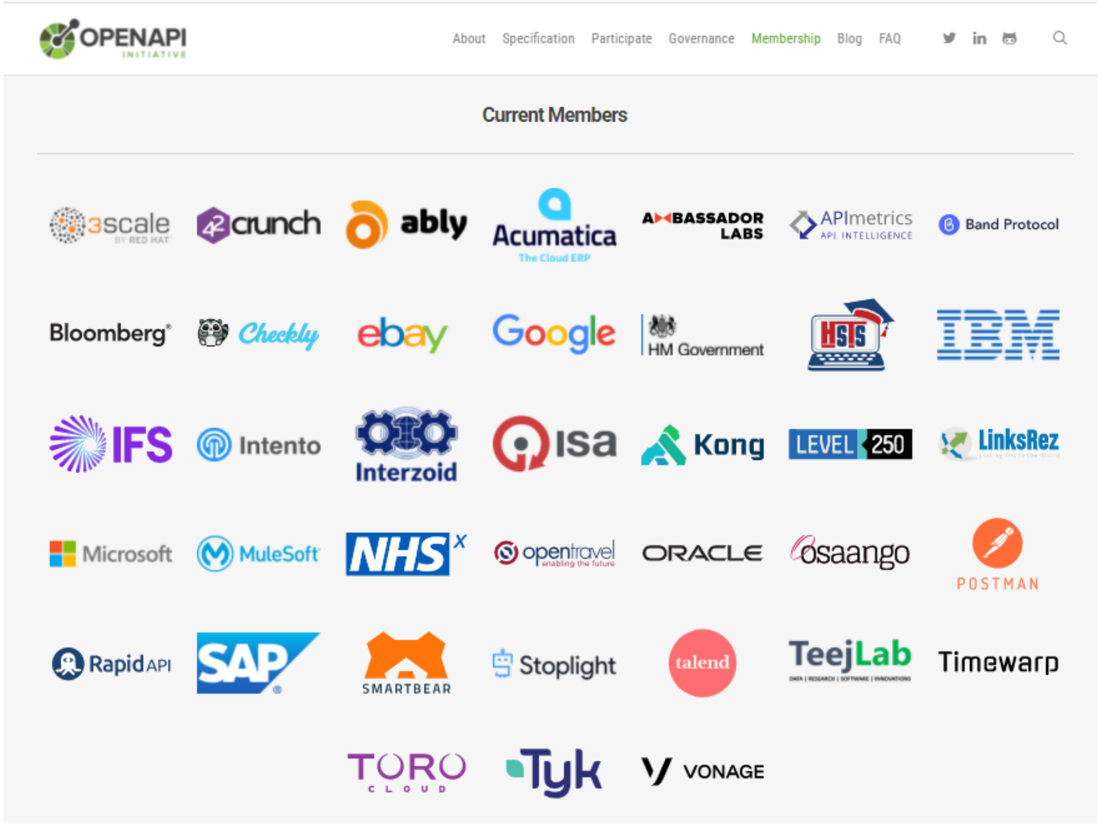
- A **standard, language-agnostic** interface to **REST APIs** which allows both **humans** and **computers** to **discover** and **understand** the capabilities of the service **without access** to source code documentation or through network traffic inspection.

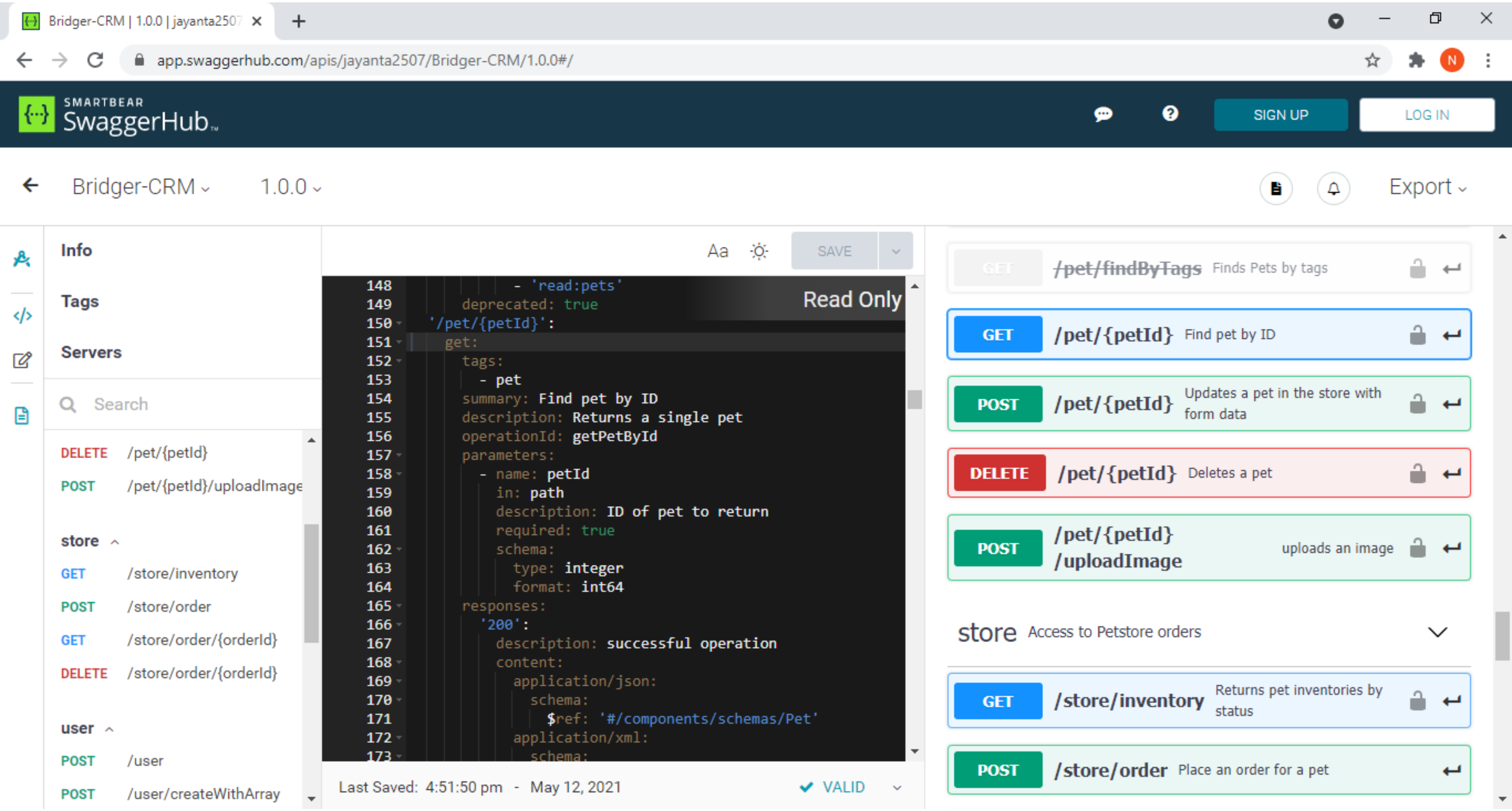
OpenAPI Specification

- Allow machines/tools to integrate with API
- Allow humans to implement API code
- Allow humans to read and generate API documentation and test case

- OpenAPI = Specification
- Swagger = tools

OpenAPI Initiative: https://www.openapis.org/





DELETE	/pet/{petId}
POST	/pet/{petId}/uploadImage

store

GET	/store/inventory
POST	/store/order
GET	/store/order/{orderId}
DELETE	/store/order/{orderId}

user

POST	/user
POST	/user/createWithArray

```
          - 'read:pets'
      deprecated: true
  '/pet/{petId}':
    get:
      tags:
        - pet
      summary: Find pet by ID
      description: Returns a single pet
      operationId: getPetById
      parameters:
        - name: petId
          in: path
          description: ID of pet to return
          required: true
          schema:
            type: integer
            format: int64
      responses:
        '200':
          description: successful operation
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Pet'
            application/xml:
              schema:
```

Last Saved: 4:51:50 pm - May 12, 2021 ✓ VALID

GET	~~/pet/findByTags~~	Finds Pets by tags
GET	/pet/{petId}	Find pet by ID
POST	/pet/{petId}	Updates a pet in the store with form data
DELETE	/pet/{petId}	Deletes a pet
POST	/pet/{petId}/uploadImage	uploads an image

store Access to Petstore orders

GET	/store/inventory	Returns pet inventories by status
POST	/store/order	Place an order for a pet

How to create OpenAPI document with swagger

- > npm install --save swagger-jsdoc@6.0.0
- > npm install --save swagger-ui-express

Ref: https://www.npmjs.com/package/swagger-jsdoc

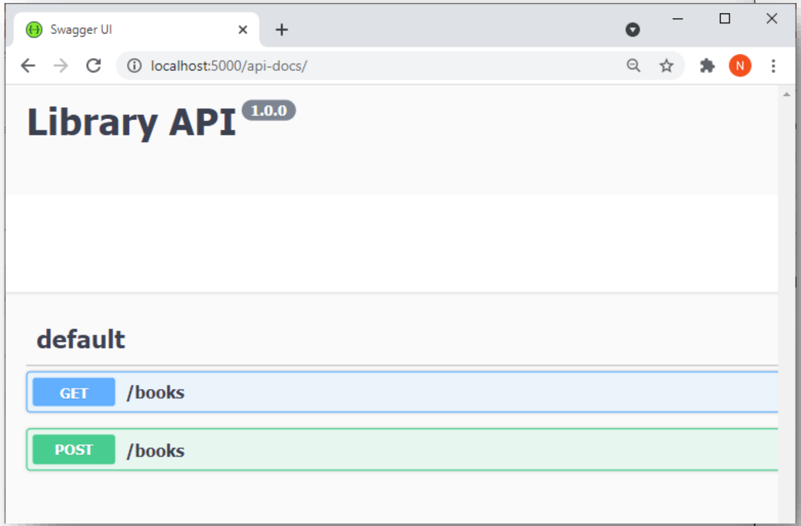
- Add swagger comment in your server.js and route files

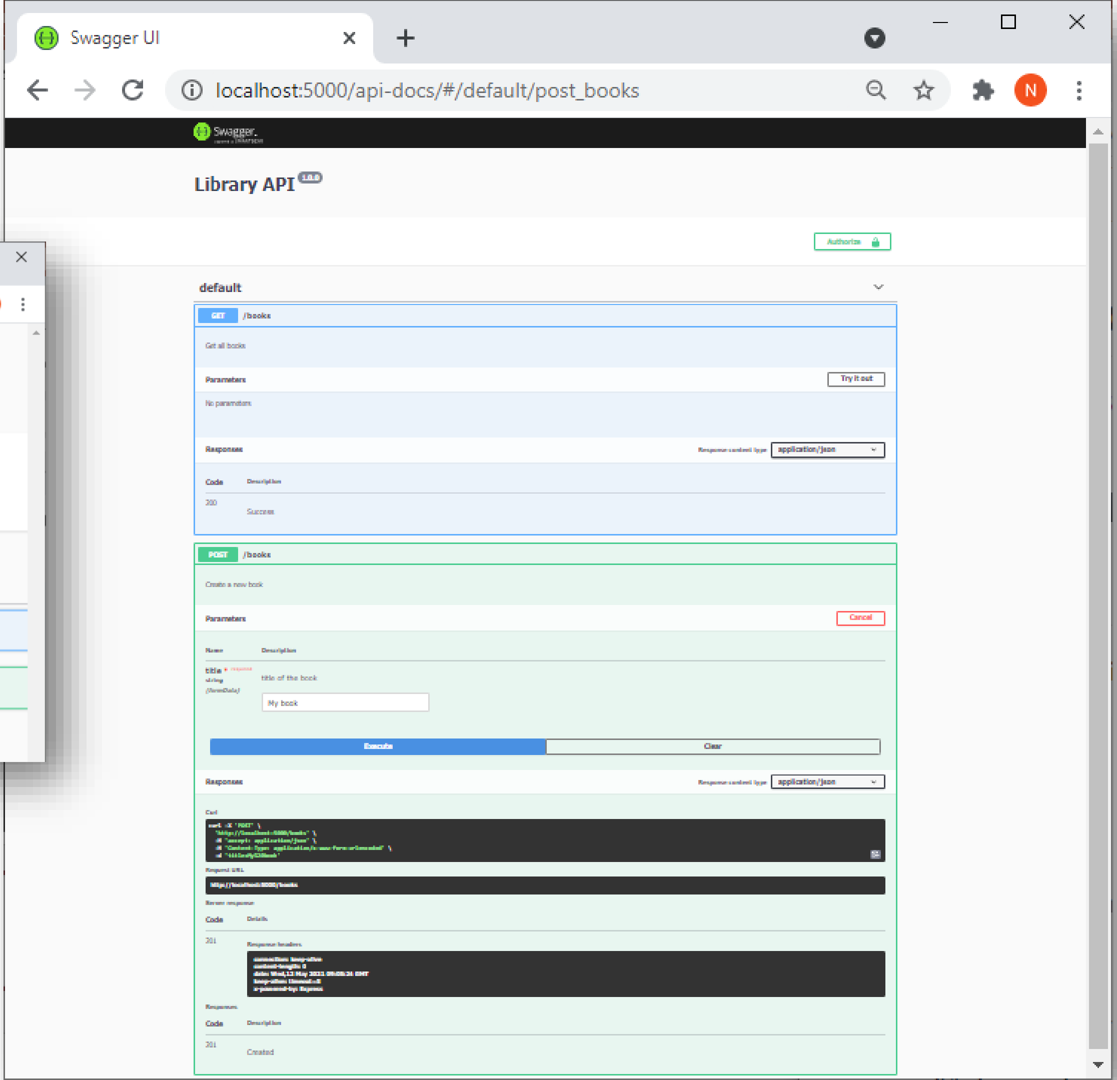
Ref: https://swagger.io/docs/specification/about/

server.js

```
const express =require('express');
const swaggerJsDoc = require('swagger-jsdoc');
const swaggerUI = require('swagger-ui-express');

const app = express();

const swaggerOptions={
    swaggerDefinition:{
        info: {
            title: 'Library API',
            version: '1.0.0'
        }
    },
    apis:['server.js'],
};

const swaggerDocs=swaggerJsDoc(swaggerOptions);
app.use('/api-docs',swaggerUI.serve, swaggerUI.setup(swaggerDocs));
```

```
/**
 * @swagger
 * /books:
 *  get:
 *    description: Get all books
 *    responses:
 *      200:
 *        description: Success
 *
 */

app.get('/books',(req,res)=>{
    …
});
```





```
curl -X 'POST' \
  'http://localhost:5000/books' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/x-www-form-urlencoded' \
  -d 'title=My%20book'
```

```
http://localhost:5000/books
```

```
connection: keep-alive
content-length: 0
date: Wed,12 May 2021 [illegible] GMT
keep-alive: timeout=5
x-powered-by: Express
```

Sample code

- https://drive.google.com/file/d/1cNbdCAiUTLNnfZqaqSbf-2ZDtHC9UmED/view?usp=sharing
- Unzip & open the folder with VS Code

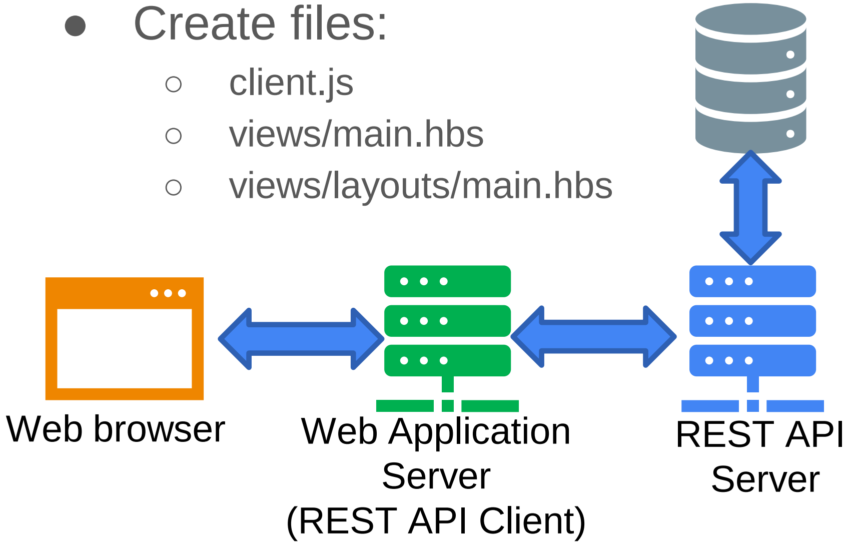
1. npm init
2. npm install -D nodemon
3. npm install express dotenv mongoose
4. npm install --save swagger-jsdoc@6.0.0
5. npm install --save swagger-ui-express
6. npm start
7. Go to: http://localhost:5000/api-docs/

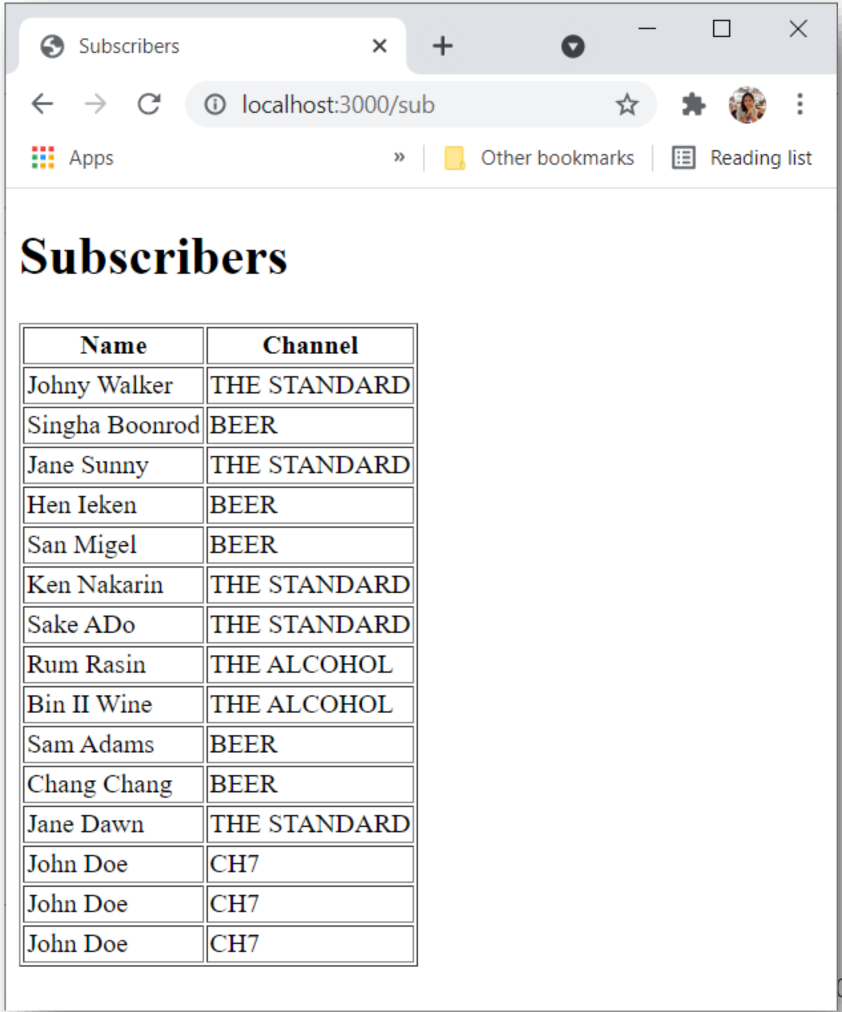
The (Simple) Client

Ref: https://www.npmjs.com/package/unirest

How to create the API Client

- > npm i unirest
- > npm i express-handlebars@6.0.6
- Create files:
  - client.js
  - views/main.hbs
  - views/layouts/main.hbs





Subscribers

Name	Channel
Johny Walker	THE STANDARD
Singha Boonrod	BEER
Jane Sunny	THE STANDARD
Hen Ieken	BEER
San Migel	BEER
Ken Nakarin	THE STANDARD
Sake ADo	THE STANDARD
Rum Rasin	THE ALCOHOL
Bin II Wine	THE ALCOHOL
Sam Adams	BEER
Chang Chang	BEER
Jane Dawn	THE STANDARD
John Doe	CH7
John Doe	CH7
John Doe	CH7

client.js (1/2)

```
const exphbs=require('express-handlebars')
const express = require('express')
const unirest = require('unirest')
const app = express()

//Handlebars
app.engine('.hbs',exphbs.engine({defaultLayout:'main',extname:'.hbs'}));
app.set('view engine','.hbs')

const PORT=3000;

app.get('/sub',async function(req,res){
    var Request = unirest.get('http://localhost:5000/subscribers').then((response) => {
        //console.log(response.body)
        var datafromserver = response.body
        res.render('main',{datafromserver})
      });

})

app.listen(PORT, () => console.log('Client Started'))
```

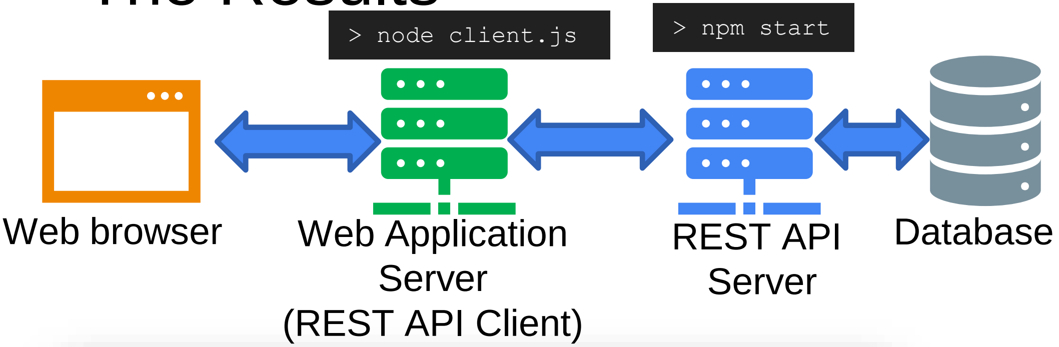
views/main.hbs

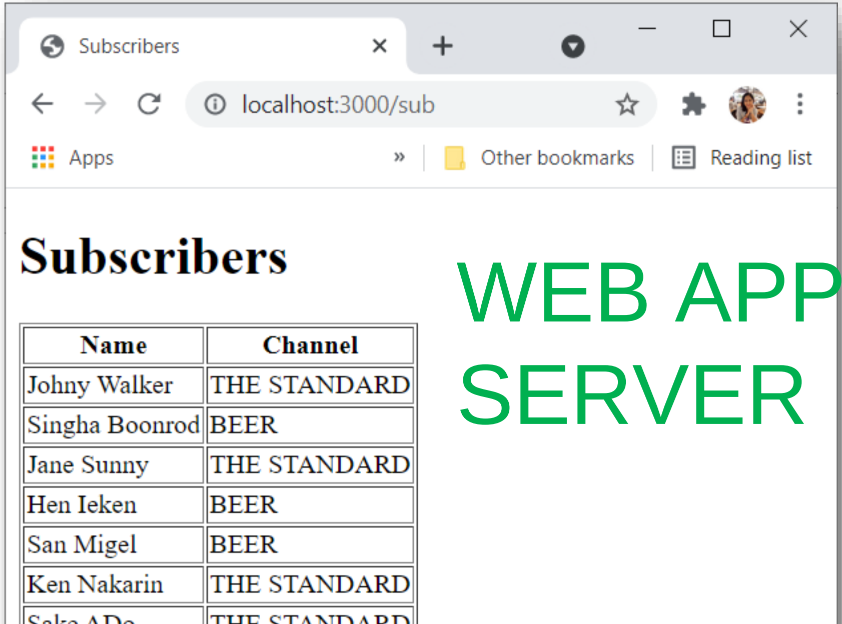
```
<h1>Subscribers</h1>
<table border=1>
    <tr>
        <th>Name</th>
        <th>Channel</th>
    </tr>
    {{#each datafromserver}}
    <tr>
        <td>{{name}}</td>
        <td>{{subscribedToChannel}}</td>
    </tr>
     {{/each}}
</table>


</div>
```

views/layouts/main.hbs

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content
="width=device-width, initial-
scale=1.0">
        <title>Subscribers</title>
    </head>
    <body>
        {{{body}}}
    </body>
</html>
```

The Results





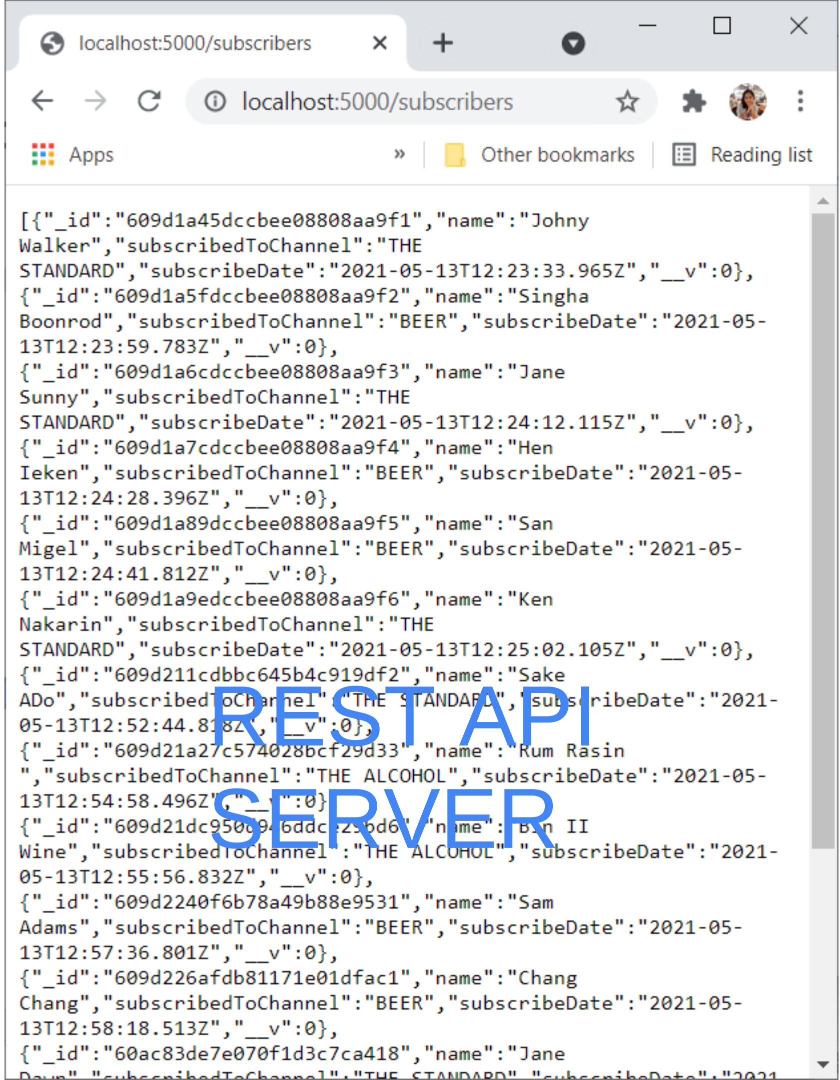
Subscribers

localhost:3000/sub

Subscribers

Name	Channel
Johny Walker	THE STANDARD
Singha Boonrod	BEER
Jane Sunny	THE STANDARD
Hen Ieken	BEER
San Migel	BEER
Ken Nakarin	THE STANDARD
Sake ADo	THE STANDARD

WEB APP SERVER



localhost:5000/subscribers

```
[{"_id":"609d1a45dccbee08808aa9f1","name":"Johny
Walker","subscribedToChannel":"THE
STANDARD","subscribeDate":"2021-05-13T12:23:33.965Z","__v":0},
{"_id":"609d1a5fdccbee08808aa9f2","name":"Singha
Boonrod","subscribedToChannel":"BEER","subscribeDate":"2021-05-
13T12:23:59.783Z","__v":0},
{"_id":"609d1a6cdccbee08808aa9f3","name":"Jane
Sunny","subscribedToChannel":"THE
STANDARD","subscribeDate":"2021-05-13T12:24:12.115Z","__v":0},
{"_id":"609d1a7cdccbee08808aa9f4","name":"Hen
Ieken","subscribedToChannel":"BEER","subscribeDate":"2021-05-
13T12:24:28.396Z","__v":0},
{"_id":"609d1a89dccbee08808aa9f5","name":"San
Migel","subscribedToChannel":"BEER","subscribeDate":"2021-05-
13T12:24:41.812Z","__v":0},
{"_id":"609d1a9edccbee08808aa9f6","name":"Ken
Nakarin","subscribedToChannel":"THE
STANDARD","subscribeDate":"2021-05-13T12:25:02.105Z","__v":0},
{"_id":"609d211cdbbc645b4c919df2","name":"Sake
ADo","subscribedToChannel":"THE STANDARD","subscribeDate":"2021-
05-13T12:52:44.818Z","__v":0},
{"_id":"609d21a27c574028bcf29d33","name":"Rum Rasin
","subscribedToChannel":"THE ALCOHOL","subscribeDate":"2021-05-
13T12:54:58.496Z","__v":0},
{"_id":"609d21dc950[illegible]6ddc[illegible]bd6","name":"B[illegible]n II
Wine","subscribedToChannel":"THE ALCOHOL","subscribeDate":"2021-
05-13T12:55:56.832Z","__v":0},
{"_id":"609d2240f6b78a49b88e9531","name":"Sam
Adams","subscribedToChannel":"BEER","subscribeDate":"2021-05-
13T12:57:36.801Z","__v":0},
{"_id":"609d226afdb81171e01dfac1","name":"Chang
Chang","subscribedToChannel":"BEER","subscribeDate":"2021-05-
13T12:58:18.513Z","__v":0},
{"_id":"60ac83de7e070f1d3c7ca418","name":"Jane
Dawn","subscribedToChannel":"THE STANDARD","subscribeDate":"2021
```

REST API SERVER

REST API’s Assignment

1. Re-create a REST API from the sample code for a restaurant’s member system.
2. Create your own database. It can be on mongodb.com. (Please do not use my MongoDB database.)
3. The data for a member includes name, address, telephone number, e-mail address, and member start date.
4. The REST API must be able to perform all CRUD operations with related methods (GET/POST/PUT/DELETE).
5. Record your demonstration of all methods with a route.rest file. The video clip should be less than 5 minutes in mp4 format.
6. Submit your Video clip to MCV.