**Task: 🦋 Butterfly critique**

**Task 1: Allow a user to rate butterflies on a scale of 0 through 5**

**Task 2: Allow retrieval of a list of a user's rated butterflies, sorted by rating**

**Implementation**

**butterfly-homework\src\**

1. Index.js 🡪 Developed two APIs
   1. addRatings (POST)
   2. userRatings (GET)
2. validators.js 🡪 Implemented two validators
   1. validateRatings () 🡪 check the type of the input
   2. validateRatingRange () 🡪 validate the range of the input

**butterfly-homework\test**

1. index.test.js 🡪 implemented testing functionality for APIs
2. validators.test.js 🡪 implemented testing functionality for implemented validators in validators.js

**JSON File Name: butterflies.db.json**

**Existing JSON:**

The existing JSON contains butterflies entity and users entity. Here, we will add one more entity(table) that is “ratings” to fulfill the given tasks.

**Reason:** To allow user to rate from 0 to 5 ratings on the butterflies and retrieve the list of rated butterfliers of a particular user in a sorting manner.

Important keys of the ratings json:

"ratings": [ {

"user\_id": "OOWzUaHLsK",

"butterfly\_id": "GI9\_EuH8s1",

"rate": 4

}

**Tool used to test the APIs:** Postman

**Node version:** v14.15.3

**Npm version:** 6.14.9

**Task 1:**

**Assumption:** User has already logged-In and selected existing butterflies to give rating

**File:** index.js

**API:** app.post (‘/addRatings’, async (req, res) 🡺 {}).

**HTTP Method used:** POST

🡪Here, addRatings API is used to add the data to the database (JSON) in “ratings”.

🡪To Accomplish this API,

Validations are performed before adding the data to the database.

1. validate the type of the data
   1. User\_id: string
   2. Butterfly\_id: string
   3. Rate: number

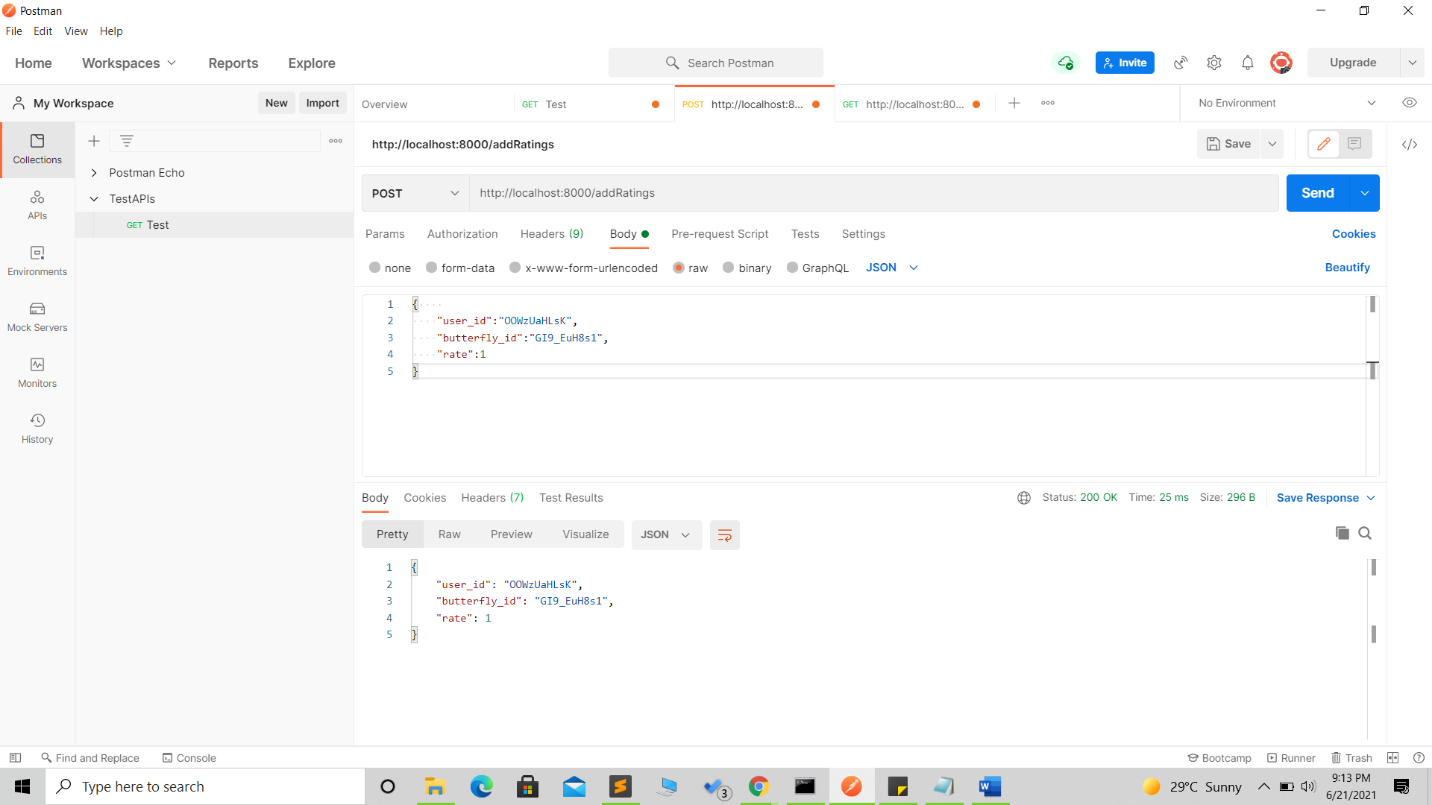
Function created: **validateRatings ()**

1. Validate the range of the rate (0 to 5)

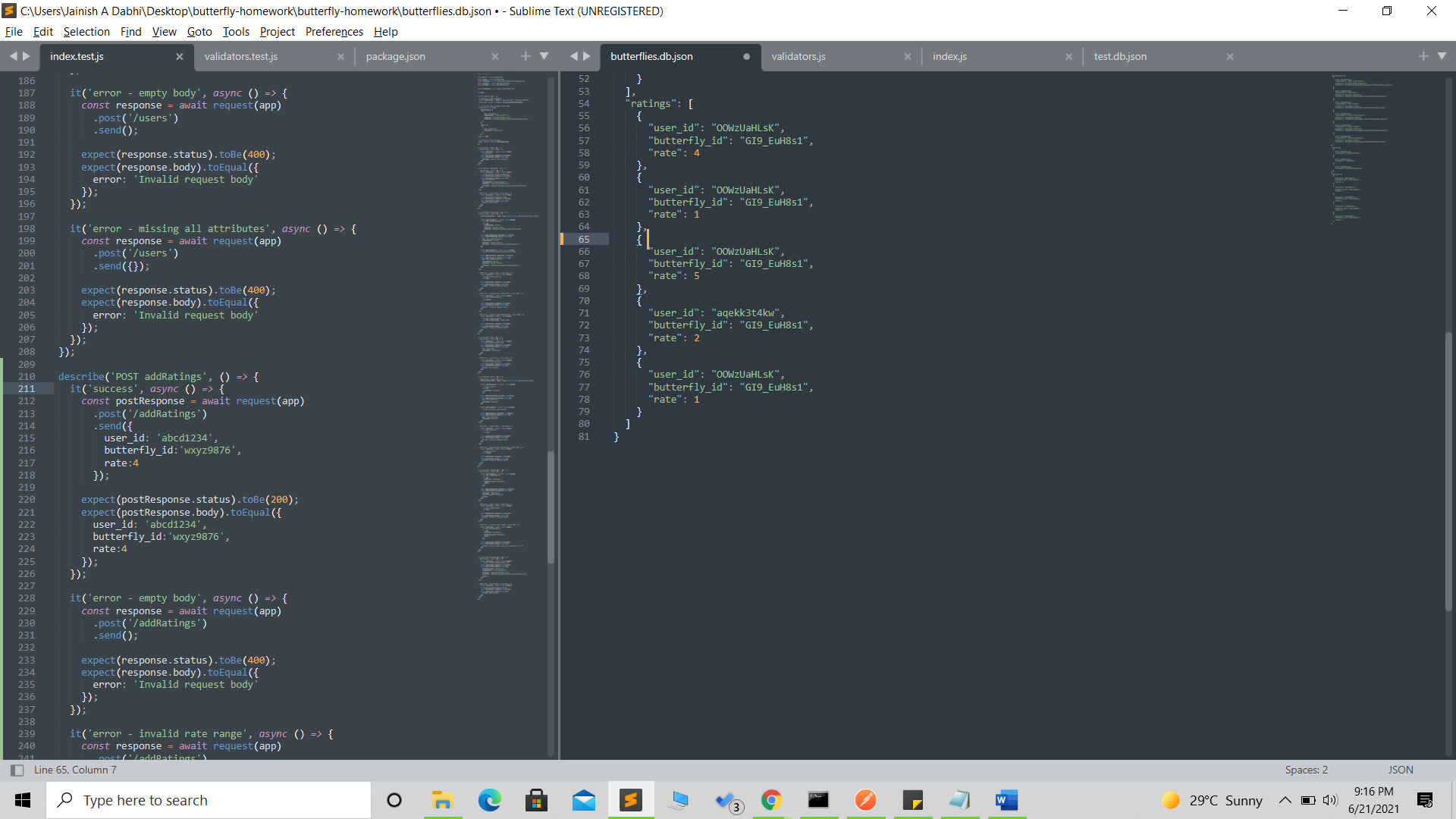
Function created: **validateRatingRange ()**

**Screenshots of the API**

Postman API Testing



Updated butterflies.db.json



**Task 2:**

**File:** index.js

**API:** app.get (‘/userRatings/:user\_id’, async (req, res) 🡺 {}).

**HTTP Method used:** GET

🡪Here, we will get the list of the butterflies in a sorting order based on ratings of the user.

**Logic:**

🡪 We will fetch the data of a particular user from ratings on user\_id provided in the URL.

🡪After that, we will sort the ratings in a descending order.

🡪 Mapping of ratings and butterflies to get the butterfly data.

**Ratings**

User\_id

Butterfly\_id

rate

**Butterflies**

Id

commonName

species

article

Mapping

🡪Once we map both the ids from the tables, we can get the details of the butterflies.

**Screenshots**

Unsorted JSON Data in “ratings”

**A computer screen capture

Description automatically generated with low confidence**

Postman API Testing to display the butterflies in a sorting order

Graphical user interface, text, application, email

Description automatically generated



**Testing**

🡪Implemented testing methods to validate the test cases for all the APIs and Validators.

**Screenshots of Testing**

**A screenshot of a computer

Description automatically generated with medium confidence**