**Steps to run code**

1. Download / Clone “imda\_assessment” folder from Github
2. Navigate to q1/application (e.g. cd YOUR\_PATH/q1/application)
3. Install dependencies with “pip install -r requirements.txt”
4. Run app.py
5. Test endpoints with Postman **(PROVIDED, unless you want to test other cases)**

**Design Rationale**

* ID given in csv is not unique (e.g. multiple reviews with ID 2601), hence generated db\_ID as key in db.
* Likewise, as ID is not unique, JSON value has to input column values of ramen review to retrieve specific ramen value
* Assumption: Review with same info can be created 🡪 different people might submit the same information, hence did not error handle this
* Note: To handle null values, query has to be modified, hence used for loop to query “IS NULL” as cannot query “= null”

**REST API Endpoints (please refer to postman link given)**

| **Request** | **API** | **Function** | **Input JSON** | **Expected Response** |
| --- | --- | --- | --- | --- |
| GET | http://127.0.0.1:5000/getAllRamen | Get all reviews | nil | [      [          1,          1,          "IDN",          "Brand A",          "Seaweed Instant Noodle",          "Cup",          5.0  ],  …  ] |
| POST | http://127.0.0.1:5000/createReview | Create Review | {          "ID":2601,          "Country":"IDN",          "Brand":"Brand A",          "Type":"Seaweed Instant Noodle",          "Package":"Cup",          "Rating":50123  } | {      "content": "new review is added",      "result": "Success"  } |
| POST | http://127.0.0.1:5000/modifyReview | Modify Review | {          "ID":2,          "Country":"KOR",          "Brand":"Brand B",          "Type":"Beef Ramen",          "Package":"Cup",          "Rating":3.3,          "new\_ID":26667,          "new\_Country":"JPJP",          "new\_Brand":"Test Modify Brand",          "new\_Type":"Test Modify Type",          "new\_Package":"Cup",          "new\_Rating":5  } | //modify success (row exists)  {      "content": "ramen updated successfully",      "result": "Success"  }  //modify fail (row does not exist)  {      "content": "no rows exist",      "result": "Error"  }  // modify fail (duplicates exist for after-modified info)  {      "content": "An existing row already exist with the updated values, please check db / contact support",      "result": "Error"  } |
| POST | http://127.0.0.1:5000/deleteReview | Delete review | {            "ID":1,          "Country":"IDN",          "Brand":"Brand A",          "Type":"Seaweed Instant Noodle",          "Package":"Cup",          "Rating":5  } | //delete success(row exists)  {      "content": "review deleted successfully",      "result": "Success"  }  //delete fail (row does not exist)  {      "content": "no rows exist",      "result": "Error"  }  // delete fail (duplicates exist)  {      "content": "Duplicated rows exist, please check db / contact support",      "result": "Error"  } |
| POST | http://127.0.0.1:5000/countryReview | Extract review by country | {          "Country":"THA"  } | //Extract success(row exists)  {      "content": [          [              7,              7,              "THA",              "Brand G",              "Tom Yum Ramen",              "Cup",              5.0          ],  …  }  //Extract success(row does not exist)  {      "content": "no rows exist for specified country",      "result": "Success"  } |
| POST | http://127.0.0.1:5000/typeReview | Extract review by text | {          "Type":"ins "  } | //Extract success(row exists)  {      "content": [          [              907,              907,              "CHN",              "Brand G",              "Sea Tangle & Sparerins Flavor Noodles",              "Bowl",              3.8          ],  …  }  //Extract success(row does not exist)  {      "content": "no rows exist for specified type",      "result": "Success"  } |