

Homework2 – Part3 (Test cases for the REST APIs)

UNI – PKV2103

Implementation Notes:

- ❖ PUT and POST requires JSON objects to be defined in body
- ❖ GET and DELETE can be tested by URL with query params.

1. `@application.route("/api/databases", methods=["GET"])`

- GET list of databases in my local instance of mysql. No input parameters passed. The related return link is same as the original link. Please see below the screenshots of Postman and also console output. Test Successful

The screenshot shows the Postman interface with a successful response. The URL is `http://127.0.0.1:5000/api/databases`. The response body is a JSON object:

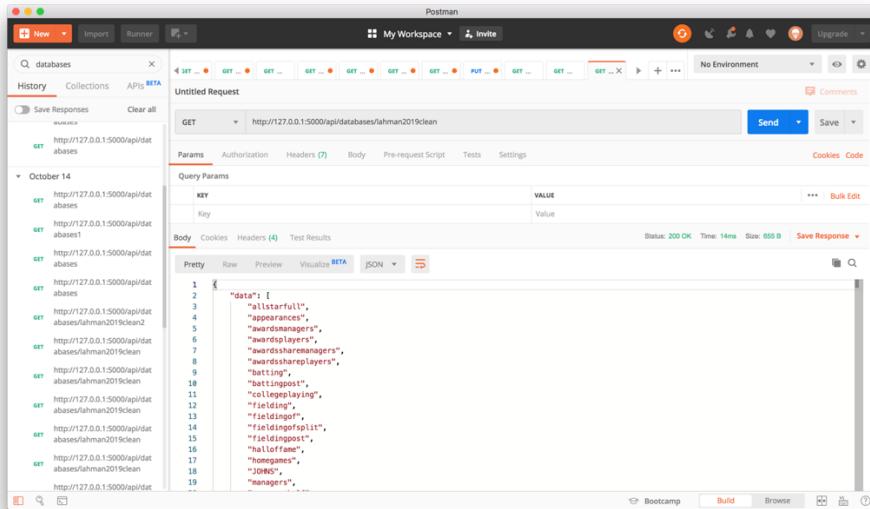
```
[{"data": [{"name": "information_schema", "host": "127.0.0.1", "port": 3306}, {"name": "sys", "host": "127.0.0.1", "port": 3306}, {"name": "performance_schema", "host": "127.0.0.1", "port": 3306}, {"name": "sys", "host": "127.0.0.1", "port": 3306}], "links": [{"rel": "current", "href": "http://127.0.0.1:5000/api/databases"}]}
```

The screenshot shows the PyCharm IDE with the `app.py` file open. The terminal window shows the server logs for a GET request to `/api/databases`:

```
DEBUG:root:2019-10-18 06:16:52.953727: Method GET received:
[{"path": "/api/databases", "method": "GET", "path_params": null, "query_params": {}, "headers": {"User-Agent": "PostmanRuntime/7.17.1", "Accept": "*/*", "Cache-Control": "no-cache", "Postman-Token": "869a9f3e-2543-4568-abf8-115b86bf1d4e", "Host": "127.0.0.1:5000", "Accept-Encoding": "gzip, deflate", "Connection": "keep-alive"}, "body": null, "url": "http://127.0.0.1:5000/api/databases", "base_url": "http://127.0.0.1:5000/api/databases"}]
DEBUG:root:Executing SQL: select group_concat(column_name order by ordinal_position) pk_cols from information_schema.key_column_usage where table_name = 'tables' and table_schema =
DEBUG:root:Executing SQL: select table_schema and constraint_name = 'primary' group by table_name
DEBUG:root:Executing SQL: select count(*) count from information_schema.tables
primary key: None
row count: 342
DEBUG:root:Executing SQL: select distinct table_schema from information_schema.tables
INFO:werkzeug:127.0.0.1 -- [18/Oct/2019 06:16:53] "GET /api/databases HTTP/1.1" 200 -
rows selected: 7
```

2. `@application.route("/api/databases/<dbname>", methods=["GET"])`

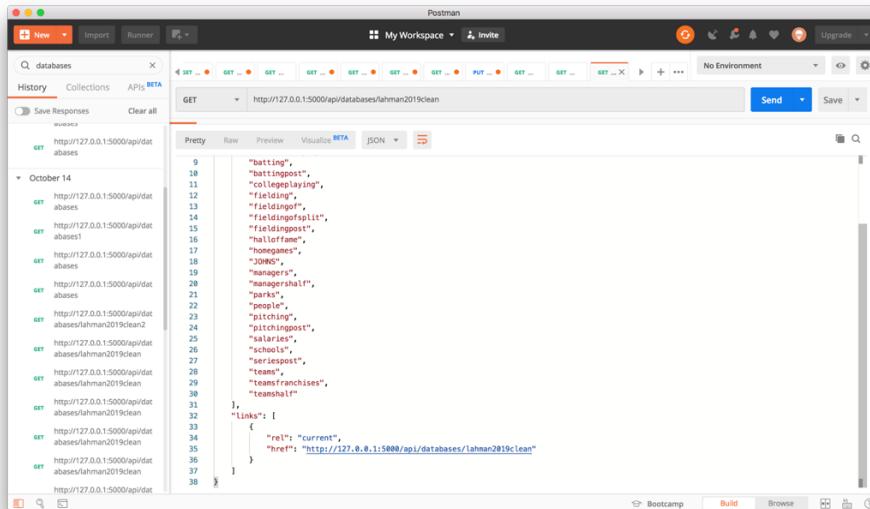
- GET list of tables in the given database. No query parameters passed. The related return link is same as the original link. Returns OK (200) status code. Please see below the screenshots of Postman and also console output



```

{
  "data": [
    "allstarfull",
    "appearances",
    "awardsmanagers",
    "awardsplayers",
    "awardsshareplayers",
    "batting",
    "battingpost",
    "collegeplaying",
    "collegepost",
    "fieldingoff",
    "fieldingsplit",
    "fieldingpost",
    "game",
    "homogames",
    "JOHNS",
    "managers"
  ]
}

```



```

{
  "data": [
    "batting",
    "collegeplaying",
    "fielding",
    "fieldingoff",
    "fieldingofsplit",
    "fieldingpost",
    "halloffame",
    "homogames",
    "JOHNS",
    "managers",
    "managershalf",
    "parks",
    "people",
    "pitching",
    "rosterpost",
    "salaries",
    "schools",
    "seriespost",
    "teams",
    "teamfranchises",
    "teamshalf"
  ],
  "links": [
    {
      "rel": "current",
      "href": "http://127.0.0.1:5000/api/databases/lahman2019clean"
    }
  ]
}

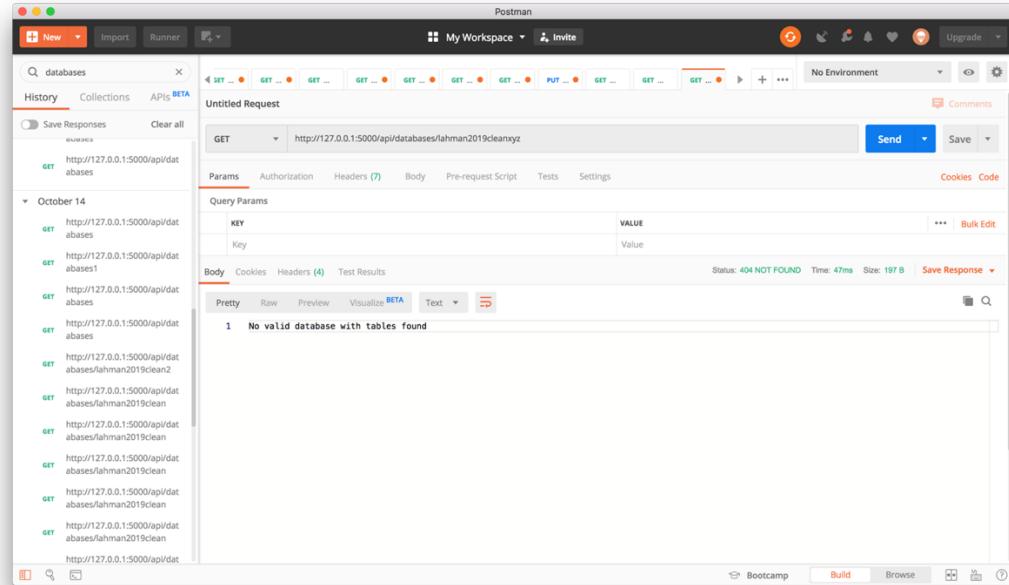
```

```

DEBUG[root@2019-10-06:27:05.926887]: Method GET received:
{
    "path": "/api/databases/lahman2019clean",
    "method": "GET",
    "path_params": null,
    "query_params": {},
    "headers": {
        "User-Agent": "PostmanRuntime/7.17.1",
        "Accept": "*/*",
        "Cache-Control": "no-cache",
        "Host": "127.0.0.1:5000",
        "Accept-Encoding": "gzip, deflate",
        "Connection": "keep-alive"
    },
    "body": null,
    "url": "https://127.0.0.1:5000/api/databases/lahman2019clean",
    "base_url": "https://127.0.0.1:5000/api/databases/lahman2019clean"
}
DEBUG[root]: Executing SQL: select distinct table_name from information_schema.tables where table_type = 'BASE TABLE' and table_schema = 'lahman2019clean'
rows selected: 28
INFO:werkzeug:127.0.0.1 -- [10/Oct/2019:06:27:45] "GET /api/databases/lahman2019clean HTTP/1.1" 200 -

```

- Edge Case:** Non-existent schema is passed to fetch the tables. Expectation is to return Not Found Error Message with 404 status code. Please see below the screenshots of Postman



3. `@application.route('/api/<dbname>/<resource>/<primary_key>', methods=['GET', 'PUT', 'DELETE'])`

GET Method

- GET by primary key (aardsda01_SEA_2010_1). Fields to be returned are playerid and yearid. No other query parameters passed. The related return link should be same as the original link. No prev and next links should be present. Returns OK (200) status code. Please see below the screenshots of Postman and also console output. *Test Successful.*

Postman screenshot showing a successful GET request to `http://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2010_1?fields=playerid,yearid`. The response is a JSON object with a single data entry containing playerid and yearid.

```

1 {
2   "data": [
3     {
4       "playerid": "aardsda01",
5       "yearid": "2010"
6     }
7   ],
8   "links": [
9     {
10       "rel": "current",
11       "href": "http://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2010_1?fields=playerid,yearid"
12     }
13   ]
14 }

```

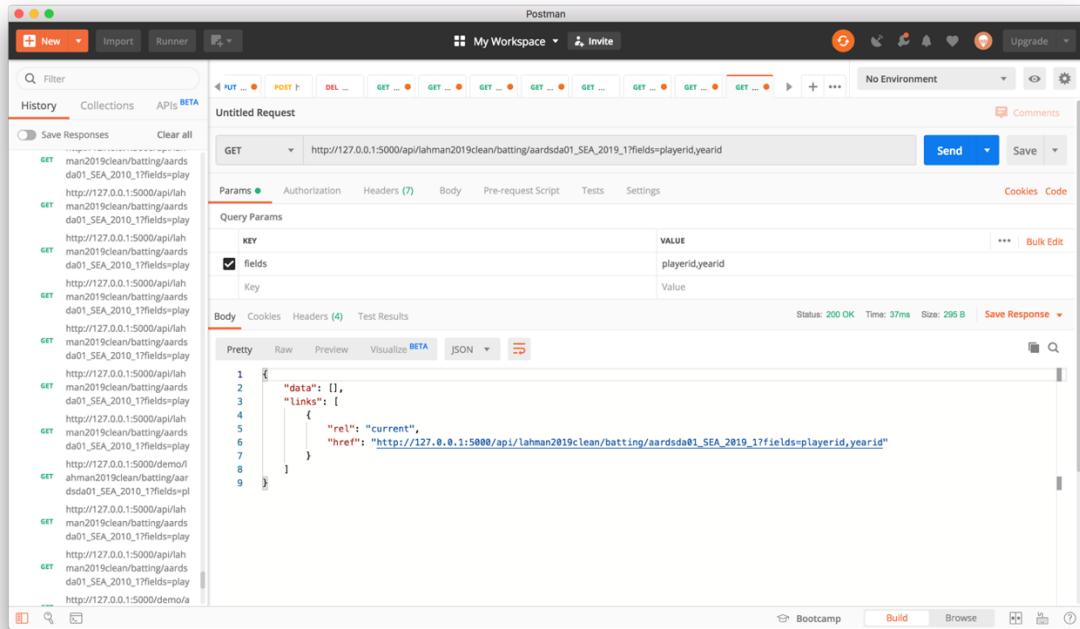
PyCharm screenshot showing the application code for HW2F19-Template. The code defines a main function that constructs a query string and a body for a POST request to the /batting endpoint, specifying playerid and yearid as fields.

```

1 path = "/api/lahman2019clean/batting/aardsda01_SEA_2010_1",
2 method = "GET",
3 params = {
4     "teamID": "Lahman2019Clean",
5     "batting",
6     "aardsda01_SEA_2010_1"
7 },
8 query_params = {},
9 headers = {
10     "User-Agent": "PostmanRuntime/7.17.1",
11     "Accept": "*/*",
12     "Cache-Control": "no-cache",
13     "Postman-Token": "87e47a9d-22f0-4d78-9ebf-6cbe34756b8d",
14     "Host": "127.0.0.1:5000",
15     "Accept-Encoding": "gzip, deflate",
16     "Connection": "keep-alive"
17 },
18 body = null,
19 url = "http://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2010_1?fields=playerid,yearid",
20 base_url = "http://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2010_1",
21 fields_params = "playerid, yearid",
22 fields = [
23     "playerid",
24     "yearid"
25 ]
26
27 DEBUG:root:Executing SQL = select group concat(column_name order by ordinal_position) pk_cols from information_schema.key_column_usage where table_name = 'batting' and table_schema = 'lahman2019clean' and constraint_name = 'primary' group by table_name
28 primary key [ 'playerID', 'teamID', 'stint' ]
29 DEBUG:root:Executing SQL = select count(*) count from lahman2019clean.batting
30 root@ip-172-31-10-10:~/Desktop/HW2F19-Template$ DEBUG:root:Executing SQL = select playerid, yearid from lahman2019clean.batting WHERE playerID='aardsda01' AND teamID='SEA' AND yearID='2010' AND stint='1'
31 DEBUG:root:Executing SQL selected: 1
32 INFO[werkzeug]:[127.0.0.1] -- [10/Oct/2019:20:36:01] "GET /api/lahman2019clean/batting/aardsda01_SEA_2010_1?fields=playerid,yearid HTTP/1.1" 200 -
33 templates: {'playerID': 'aardsda01', 'teamID': 'SEA', 'yearID': '2010', 'stint': '1'}

```

- Edge case: GET by primary key (aardsda01_SEA_2019_1). Zero rows returned by this key. Fields to be returned are playerid and yearid. No other query parameters passed. The related return link should be same as the original link. No prev and next links should be present. Returns OK (200) status code. Please see below the screenshots of Postman and also console output. *Test Successful*.



```

DEBUG:root:2019-10-18 20:53:27.065972: Method GET received:
{
  "path": "/api/lahman2019clean/batting/aardsda01_SEA_2019_1",
  "method": "GET",
  "path_params": [
    "lahman2019clean",
    "batting",
    "aardsda01_SEA_2019_1"
  ],
  "query_params": {},
  "headers": {
    "User-Agent": "PostmanRuntime/7.17.1",
    "Accept": "*/*",
    "Cache-Control": "no-cache",
    "Postman-Token": "cb5855d6-66cf-4f65-91d1-006cce84e5bd",
    "Host": "127.0.0.1:5000",
    "Accept-Encoding": "gzip, deflate",
    "Connection": "keep-alive"
  },
  "body": null,
  "url": "http://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2019_1?fields=playerid,yearid",
  "base_url": "http://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2019_1",
  "fields_param": "playerid,yearid",
  "fields": [
    "playerid",
    "yearid"
  ]
}
DEBUG:root:Executing SQL = select playerid,yearid from lahman2019clean.batting WHERE playerID='aardsda01' AND teamID='SEA' AND yearID='2019' AND stint='1'
template: {'playerID': 'aardsda01', 'teamID': 'SEA', 'yearID': '2019', 'stint': '1'}
DEBUG:root:Rows selected: 0
INFO:werkzeug:127.0.0.1 - - [18/Oct/2019 20:53:27] "GET /api/lahman2019clean/batting/aardsda01_SEA_2019_1?fields=playerid,yearid HTTP/1.1" 200 -

```

- Negative test case: GET by primary key (aardsda01_SEA_2010_1). Invalid field names to return. Returns Internal Error (500) status code along with the reason of incorrect column name. Please see below the screenshots of Postman and also console output. *Test Successful.*

Postman screenshot showing a failed API call. The URL is `http://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2019_1?fields=playerid,yearid1`. The response status is 500 INTERNAL SERVER ERROR with the message "Internal Error: (1054, "Unknown column 'yearid1' in 'field list'"').

```

template: {'playerID': 'aardsda01', 'teamID': 'SEA', 'yearID': '2019', 'stint': '1'}
DEBUG:root:2019-10-10 21:12:49,97338: Method GET received:
{
    "path": "/api/lahman2019clean/batting/aardsda01_SEA_2019_1",
    "method": "GET",
    "path_params": [
        "lahman2019clean",
        "batting",
        "aardsda01_SEA_2019_1"
    ],
    "query_params": {},
    "headers": {
        "User-Agent": "PostmanRuntime/7.17.1",
        "Accept": "*/*",
        "Cache-Control": "no-cache",
        "Postman-Token": "aa6b6b3a-27e3-49a1-93fc-a688282dd18d",
        "Host": "127.0.0.1:5000",
        "Accept-Encoding": "gzip, deflate",
        "Connection": "keep-alive"
    },
    "body": null,
    "url": "http://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2019_1?fields=playerid,yearid1",
    "base_url": "https://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2019_1",
    "fields_param": "playerid,yearid1",
    "fields": [
        "playerid",
        "yearid1"
    ]
}
DEBUG:root:Executing SQL = select playerid,yearid1 from lahman2019clean.batting WHERE playerID='aardsda01' AND teamID='SEA' AND yearID='2019' AND stint='1'
Exception e = (1054, "Unknown column 'yearid1' in 'field list'"')
(1054, "Unknown column 'yearid1' in 'field list'"')
INFO:werkzeug:127.0.0.1 - - [10/Oct/2019:21:12:49] "GET /api/lahman2019clean/batting/aardsda01_SEA_2019_1?fields=playerid,yearid1 HTTP/1.1" 500 -

```

- GET by primary key (aardsda01_SEA_2010_1). No return fields passed explicitly. The related return link should be same as the original link. No prev and next links should be present. Returns OK (200) status code and the data for all fields. Please see below the screenshots of Postman and also console output. *Test Successful*

Postman

Untitled Request

GET http://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2010_1

Params Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize **BETA** JSON

```

1 {
2   "data": [
3     {
4       "playerID": "aardsda01",
5       "yearID": "2010",
6       "stint": "1",
7       "teamID": "SEA",
8       "lgID": "AL",
9       "G": "53",
10      "AB": "0",
11      "R": "0",
12      "H": "0",
13      "B2": "0",
14      "B3": "0",
15      "HR": "0",
16      "RB1": "0",
17      "SB": "0",
18      "CS": "0",
19      "BB": "0"
20    }
21  ]
22 }
  
```

Status: 200 OK Time: 10ms Size: 550 B Save Response

PUT Method

- PUT by primary key. I chose the same batting table. The values to be updated are passed in the body. It should return the number of records updated. The related return link should contain the link with which you can verify the updated (i.e., by primary_key). No prev and next links should be present. Returns OK (200) status code. Please see below the screenshots of Postman and also console output. *Test Successful.*

Postman

Untitled Request

PUT http://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2010_1?fields=playerid,yearid,stint,R

Params ● Authorization Headers (9) **Body** ● Pre-request Script Tests Settings

Body

```

1 - {
2   "R": "32",
3   "H": "32",
4   "B2": "32",
5   "B3": "32",
6   "HR": "32",
7   "RB1": "32",
8   "SB": "32",
9   "CS": "32",
10  "BB": "32"
11 }
  
```

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize **BETA** JSON

```

1 {
2   "Rows Updated": 1,
3   "links": [
4     {
5       "rel": "current",
6       "href": "http://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2010_1"
7     }
8   ]
9 }
  
```

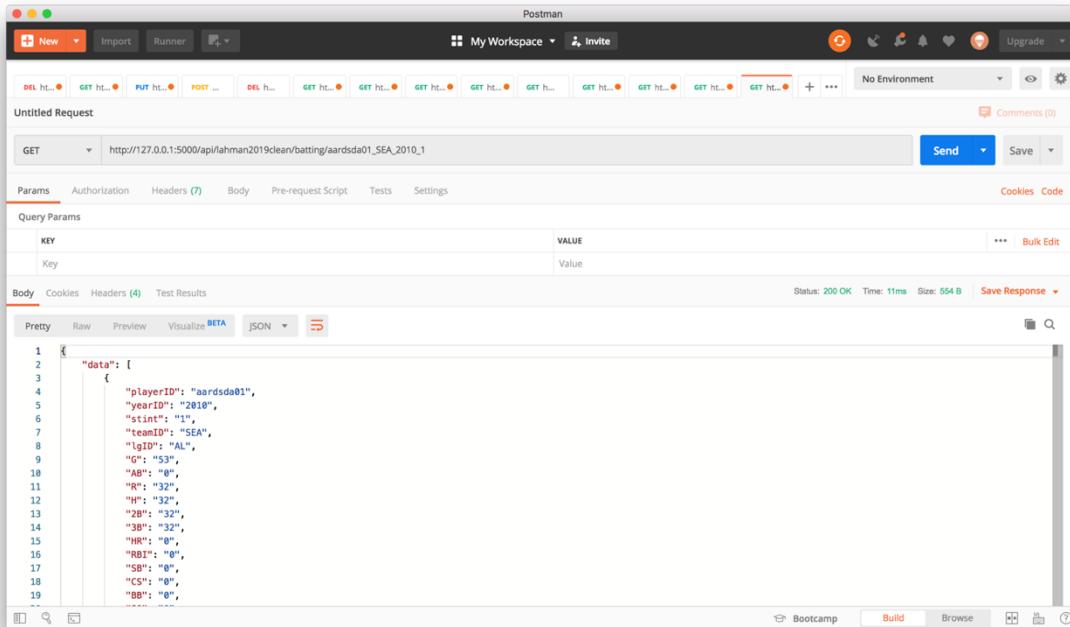
Status: 200 OK Time: 38ms Size: 279 B Save Response

```

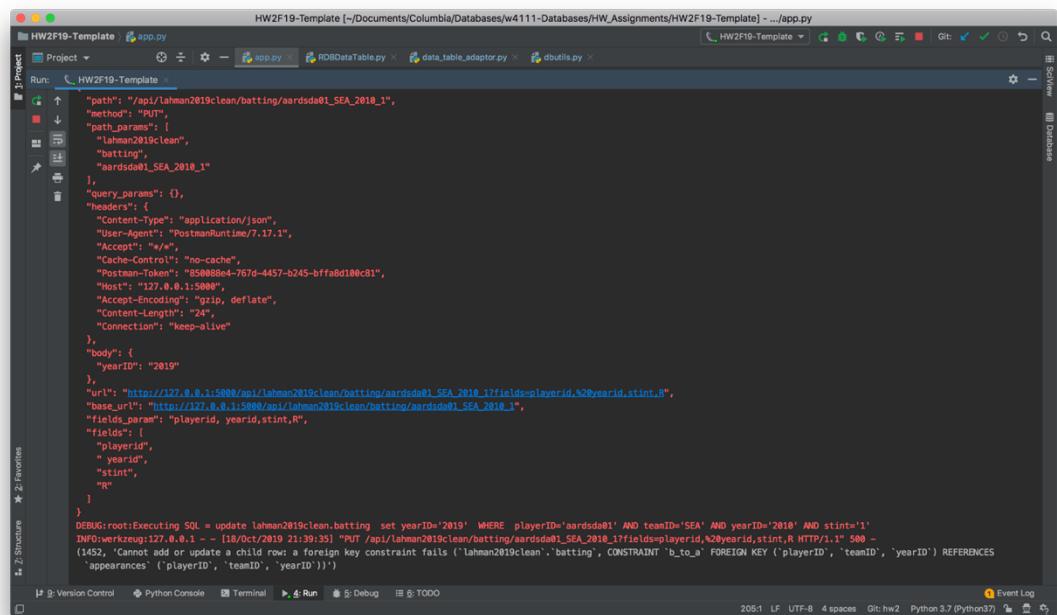
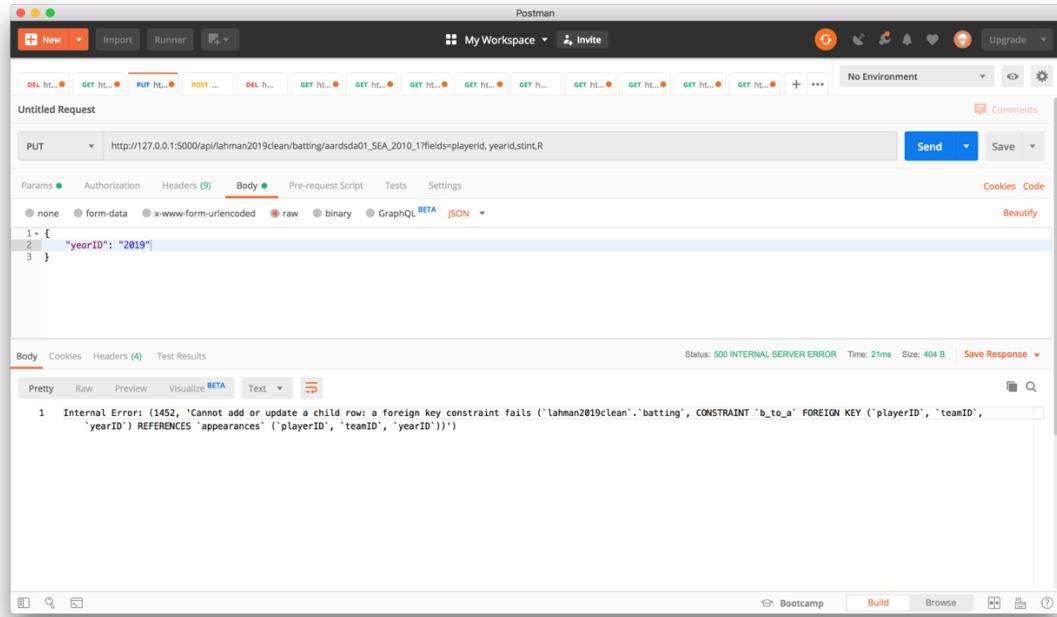
    "path_params": [
        "lahman2019clean",
        "batting",
        "aardsda01_SEA_2010_1"
    ],
    "query_params": {},
    "headers": {
        "Content-Type": "application/json",
        "User-Agent": "PostmanRuntime/7.17.1",
        "Accept": "*/*",
        "Cache-Control": "no-cache",
        "Postman-Token": "3af2a53d-1e71-4c81-a92c-d2a74e462a96",
        "Host": "127.0.0.1:5000",
        "Accept-Encoding": "gzip, deflate",
        "Content-Length": "62",
        "Connection": "keep-alive"
    },
    "body": {
        "R": "32",
        "H": "32",
        "2B": "32",
        "3B": "32"
    },
    "url": "http://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2010_1?fields=playerid,%20yearid,stint,R",
    "base_url": "http://127.0.0.1:5000/api/lahman2019clean/batting/aardsda01_SEA_2010_1",
    "fields": [
        "playerid",
        "yearid",
        "stint",
        "R"
    ]
}
DEBUG:root:Executing SQL = update lahman2019clean.batting set R='32',H='32',2B='32',3B='32' WHERE playerID='aardsda01' AND teamID='SEA' AND yearID='2010' AND stint='1'
DEBUG:root:Rows updated: 1
INFO:werkzeug:127.0.0.1 - - [18/Oct/2019 21:31:45] "PUT /api/lahman2019clean/batting/aardsda01_SEA_2010_1?fields=playerid,%20yearid,stint,R HTTP/1.1" 200 -

```

When you click on the related link provide, it will provide the data based on the primary key. Check that the updated values are reflected. *Test Successful.*



- Negative Test Case: PUT/Update by primary key. I chose the same batting table. Update the values such that it violates key constraints. Returns Internal Error (500) status code along with the reason of Key Violation. Please see below the screenshots of Postman and also console output. *Test Successful.*



DELETE Method

- DELETE by primary key. The row should not be referenced by another child table. For this purpose, I inserted a record into teams table with teamID = SEA and yearID = 2019. And then deleted the record with this API. Return the number of records deleted (i.e., 1). And also return the link with which the data can be validated for that key. *Test Successful.*

Postman - Untitled Request

DELETE http://127.0.0.1:5000/api/lahman2019clean/teams/SEA_2019

Body

```

1 <!
2   "teamID": "SEA",
3   "yearID": "2019"
4 }

```

Body

```

1 <!
2   "Rows Deleted": 1,
3   "links": [
4     {
5       "rel": "current",
6       "href": "http://127.0.0.1:5000/api/lahman2019clean/teams/SEA_2019"
7     }
8   ]
9

```

Status: 200 OK Time: 23ms Size: 265 B Save Response

HW2F19-Template [~/Documents/Columbia/Databases/w4111-Databases/HW_Assignments/HW2F19-Template] - .../app.py

Run: HW2F19-Template

```

DEBUG:root:2019-10-18 22:02:24.362391: Method DELETE received:
{
    "path": "/api/lahman2019clean/teams/SEA_2019",
    "method": "DELETE",
    "path_params": {
        "lahman2019clean": "lahman2019clean",
        "team": "SEA",
        "year": "2019"
    },
    "query_params": {},
    "headers": {
        "Content-Type": "application/json",
        "User-Agent": "PostmanRuntime/7.17.1",
        "Accept": "*/*",
        "Cache-Control": "no-cache",
        "Postman-Token": "f431153c-c2a3-4f7f-bae9-0ef811e4269b",
        "Host": "127.0.0.1:5000",
        "Accept-Encoding": "gzip, deflate",
        "Content-Length": "43",
        "Connection": "keep-alive"
    },
    "body": {
        "teamID": "SEA",
        "yearID": "2019"
    },
    "url": "http://127.0.0.1:5000/api/lahman2019clean/teams/SEA_2019",
    "base_url": "http://127.0.0.1:5000/api/lahman2019clean/teams/SEA_2019"
}

```

```

DEBUG:root:Executing SQL = delete from lahman2019clean.teams WHERE teamID='SEA' AND yearID='2019'
DEBUG:root:Row deleted: 1
INFO:werkzeug:127.0.0.1 - - [18/Oct/2019 22:02:24] "DELETE /api/lahman2019clean/teams/SEA_2019 HTTP/1.1" 200 -

```

Click on the link to check that the data has been deleted and doesn't exist anymore. *Test Successful.*

Postman

Untitled Request

GET http://127.0.0.1:5000/api/lahman2019clean/teams/SEA_2019

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE
Key	Value

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize **BETA** JSON

```

1 {
2   "data": [],
3   "links": [
4     {
5       "rel": "current",
6       "href": "http://127.0.0.1:5000/api/lahman2019clean/teams/SEA_2019"
7     }
8   ]
9 }
```

Status: 200 OK Time: 10ms Size: 258 B Save Response

- DELETE by non-existent key; It should give 0 rows deleted with OK status code. *Test Successful.*

Postman

DELETE http://127.0.0.1:5000/api/lahman2019clean/teams/SEA_201911

Params Authorization Headers (9) Body ● Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize **BETA** JSON

```

1 {
2   "Rows Deleted": 0,
3   "Links": [
4     {
5       "rel": "current",
6       "href": "http://127.0.0.1:5000/api/lahman2019clean/teams/SEA_201911"
7     }
8   ]
9 }
```

Status: 200 OK Time: 10ms Size: 267 B Save Response

- DELETE a record from the parent table. It should give Internal Error (500) complaining that child is referencing this record. *Test Successful.*

```
4. @application.route('/api/<dbname>/<resource_name>', methods=['GET', 'POST'])
```

GET Method with Pagination

- GET data by template: No pagination/limits. Should return all records and only has one current link (url). Below are two snapshots. One on table parks with no return fields specified. The other on teams with return fields as yearID. *Test Successful.*

Postman

Untitled Request

GET http://127.0.0.1:5000/api/lahman2019clean/parks?state=NY

Params Auth Headers (9) Body Pre-req. Tests Setting Cookies Code

Status: 200 OK Time: 54ms Size: 5.31 KB

Query Params

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> state	NY	

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize **BETA** JSON

```
1 {
  "data": [
    {
      "park.key": "ALB01",
      "park.name": "Riverside Park",
      "park.alias": "",
      "city": "Albany",
      "state": "NY",
      "country": "US"
    },
    {
      "park.key": "BUF01",
      "park.name": "Riverside Grounds",
      "park.alias": "",
      "city": "Buffalo",
      "state": "NY",
      "country": "US"
    },
    {
      "park.key": "BUF02",
      "park.name": "Olympic Park I",
      "park.alias": "",
      "city": "Buffalo",
      "state": "NY",
      "country": "US"
    }
  ]
}
```

Bootcamp Build Browse

Postman

Untitled Request

GET http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&fields=yearID

Params Auth Headers (7) Body Pre-req. Tests Setting Cookies Code

Status: 200 OK Time: 15ms Size: 1.09 KB

Query Params

KEY	VALUE
<input checked="" type="checkbox"/> teamID	SEA
<input checked="" type="checkbox"/> fields	yearID

Body Cookies Headers (4) Test Results

Pretty Raw Preview Visualize **BETA** JSON

```
113   },
114   {
115     "yearID": "2014"
116   },
117   {
118     "yearID": "2015"
119   },
120   {
121     "yearID": "2016"
122   },
123   {
124     "yearID": "2017"
125   },
126   {
127     "yearID": "2018"
128   },
129   ],
130   "links": [
131     {
132       "rel": "current",
133       "href": "http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&fields=yearID"
134     }
135   ]
136 }
```

Bootcamp Build Browse

```

HW2F19-Template [~/Documents/Columbia/Databases/w4111-Databases/HW_Assignments/HW2F19-Template] .../app.py
Project  HW2F19-Template ->/Documents/Columbia/Databases/w4111-Databases/HW_Assignments/HW2F19-Template
Run: HW2F19-Template
Run: HW2F19-Template
template: {'state': 'NY'}
DEBUG:root:Rows selected: 40
INFO:werkzeug:127.0.0.1 - - [18/Oct/2019 22:29:54] "GET /api/lehman2019clean/parks?state=NY HTTP/1.1" 200 -
INFO:werkzeug:127.0.0.1 - - [18/Oct/2019 22:33:35.981884] "Method GET received:
{
  "path": "/api/lehman2019clean/teams",
  "method": "GET",
  "path_params": [
    "lehman2019clean",
    "teams"
  ],
  "query_params": {
    "teamID": "SEA"
  },
  "headers": {
    "User-Agent": "PostmanRuntime/7.17.1",
    "Accept": "*/*",
    "Cache-Control": "no-cache",
    "Postman-Token": "fd2f787a-3ccf-4099-ad34-254eb3c93ae5",
    "Host": "127.0.0.1:5000",
    "Accept-Encoding": "gzip, deflate",
    "Connection": "keep-alive"
  },
  "body": null,
  "url": "http://127.0.0.1:5000/api/lehman2019clean/teams?teamID=SEA&fields=yearID",
  "baseURL": "http://127.0.0.1:5000/api/lehman2019clean/teams",
  "fields_param": "yearID",
  "fields": [
    "yearID"
  ]
}
DEBUG:root:Executing SQL: select yearID from lehman2019clean.teams WHERE teamID='SEA'
DEBUG:root:Rows selected: 42
INFO:werkzeug:127.0.0.1 - - [18/Oct/2019 22:33:35] "GET /api/lehman2019clean/teams?teamID=SEA&fields=yearID HTTP/1.1" 200 -
|
```

- GET Data by template. This time with a limit of 3 records. Should return only 3 rows and should show the link for next set of data. Should not have previous link. *Test Successful.*

Key	Value
teamID	SEA
fields	yearID
limit	3

```

1   "data": [
2     {
3       "yearID": "1977"
4     },
5     {
6       "yearID": "1978"
7     },
8     {
9       "yearID": "1979"
10    }
11  ],
12  "links": [
13    {
14      "rel": "current",
15      "href": "http://127.0.0.1:5000/api/lehman2019clean/teams?teamID=SEA&fields=yearID&limit=3"
16    },
17    {
18      "rel": "next",
19      "href": "http://127.0.0.1:5000/api/lehman2019clean/teams?teamID=SEA&fields=yearID&offset=3&limit=3"
20    }
21  ]
22 }
```

- Now click on the next url link. It should give next three records along with previous and next urls. *Test Successful.*

Postman

Untitled Request

GET http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&fields=yearID&offset=3&limit=3

Params: teamID=SEA, fields=yearID, offset=3, limit=3

Body:

```
{
  "data": [
    {
      "yearID": "1980"
    },
    {
      "yearID": "1981"
    },
    {
      "yearID": "1982"
    }
  ],
  "links": [
    {
      "rel": "current",
      "href": "http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&fields=yearID&offset=3&limit=3"
    },
    {
      "rel": "next",
      "href": "http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&fields=yearID&offset=6&limit=3"
    },
    {
      "rel": "previous",
      "href": "http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&fields=yearID&offset=0&limit=3"
    }
  ]
}
```

```
INFO:werkzeug:127.0.0.1 -- [18/Oct/2019 22:37:26] "GET /api/lahman2019clean/teams?teamID=SEA&fields=yearID&limit=3 HTTP/1.1" 200 -
DEBUG:root:2019-10-18 22:38:48.638492: Method GET received:
{
  "path": "/api/lahman2019clean/teams",
  "method": "GET",
  "path_params": {
    "lahman2019clean",
    "teams"
  },
  "query_params": {
    "teamID": "SEA"
  },
  "headers": {
    "User-Agent": "PostmanRuntime/7.17.1",
    "Accept": "*/*",
    "Cache-Control": "no-cache",
    "Postman-Token": "d383fe4e-7f8a-42cc-b443-9dae602415fd",
    "Host": "127.0.0.1:5000",
    "Accept-Encoding": "gzip, deflate",
    "Connection": "keep-alive"
  },
  "body": null,
  "url": "http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&fields=yearID&limit=3",
  "base_url": "http://127.0.0.1:5000/api/lahman2019clean/teams",
  "fields_param": "yearID",
  "fields": [
    "yearID"
  ],
  "limit": 3,
  "offset": 3
}
DEBUG:root:Executing SQL = select yearID from lahman2019clean.teams WHERE teamID='SEA' limit 3 offset 3
DEBUG:root:Rows selected: 3
INFO:werkzeug:127.0.0.1 -- [18/Oct/2019 22:38:48] "GET /api/lahman2019clean/teams?teamID=SEA&fields=yearID&limit=3 HTTP/1.1" 200 -
```

- Edge Case: Now test the GET API with an input on offset and limit. The previous, current and next links should be generated automatically. *Test Successful.*

The screenshot shows the Postman application interface. A GET request is made to `http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&fields=yearID&offset=5&limit=2`. The response status is 200 OK, time taken is 11ms, and size is 569 B. The response body is a JSON object:

```

1 {
2   "data": [
3     {
4       "yearID": "1982"
5     },
6     {
7       "yearID": "1983"
8     }
9   ],
10  "links": [
11    {
12      "rel": "current",
13      "href": "http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&fields=yearID&offset=5&limit=2"
14    },
15    {
16      "rel": "next",
17      "href": "http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&fields=yearID&offset=7&limit=2"
18    },
19    {
20      "rel": "previous",
21      "href": "http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&fields=yearID&offset=3&limit=2"
22    }
23  ]
24 }

```

- Edge Case: GET API when the query params provided doesn't retrieve any data. Status code 200, with no data. There shouldn't be next and previous URLs. *Test Successful.*

The screenshot shows the Postman application interface. A GET request is made to `http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&fields=yearID&limit=2`. The response status is 200 OK, time taken is 15ms, and size is 283 B. The response body is a JSON object:

```

1 {
2   "data": [],
3   "links": [
4     {
5       "rel": "current",
6       "href": "http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&fields=yearID&limit=2"
7     }
8   ]
9 }

```

POST Method

- Insert new record. Json captured in body for new record attributes. Return a message saying Row inserted successfully and also give a url for the inserted record in response header. Return status code is 200 (OK), as mentioned in the HW2 specifications (this could also be done as return code 201) – *Test Successful.*

The screenshot shows the Postman application interface. A POST request is being made to the URL `http://127.0.0.1:5000/api/lahman2019clean/teams`. The request body is JSON, containing the following data:

```
1 - {  
2   "teamID": "SEA",  
3   "yearID": "2019"  
4 }
```

The response status is 200 OK, with a response body indicating the record was inserted successfully:

```
1 - {  
2   "Record Inserted": "Successful"  
3 }
```

Now check the response header for the URL. *Test Successful.*

The screenshot shows the Postman application interface again, displaying the response headers for the previous POST request. The headers are:

KEY	VALUE
Content-Type	application/json
Content-Length	33
Location	http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&yearID=2019
Server	Werkzeug/0.16.0 Python/3.7.4
Date	Sat, 19 Oct 2019 16:53:27 GMT

Retrieve data using URL to confirm the data. *Test Successful.*

The screenshot shows the Postman application interface. A GET request is made to `http://127.0.0.1:5000/api/lahman2019clean/teams?teamID=SEA&yearID=2019`. The response status is 200 OK, time 10ms, and size 933 B. The response body is a JSON object with a single key "data": [] containing 24 empty objects, each representing a team record for year 2019.

- Insert a new record which violates the primary key. Should return internal error with the error message. *Test Successful.*

The screenshot shows a POST request to `http://127.0.0.1:5000/api/lahman2019clean/teams`. The request body is a JSON object with two keys: "teamID" and "yearID", both set to "SEA" and "2019" respectively. The response status is 500 INTERNAL SERVER ERROR, with a message indicating a duplicate entry for the primary key "SEA-2019".