Documentación Programa Administración De Estudiantes (Examen Final)

Docente:

Jeyson Alejandro Calvache

Integrantes:

Yan Carlos Cuaran Imbacuan

Edwar Andrés Hernández

Semestre: V

Instituto Tecnológico Del Putumayo

Sede Mocoa

Tecnología En Desarrollo De software

Lenguaje De Cuarta Generación

12/06/2021

Documentación Programa Administración De Estudiantes

Problema:

El problema consiste en pasar el Parcial 2 realizado a API REST el cual consistía en la administración de estudiantes.

Recomendaciones:

- Al ingresar los respectivos ID de las tablas en los campos con los que se quiere relacionar hay que tener en cuenta que el ID del objeto o dato debe existir de lo contrario generara un error.
- Ejemplo: al relacionar un dato de la tabla students con la tabla sessions, el ID del dato de la tabla students debe de existir para hacer la relación.
- Para que el programa se ejecute correctamente se debe realizar la instalación de la base de datos DB.
- En la función de ediciones en el programa se debe enviar el ID del dato a editar tanto en la URL como en el JSON esto solo aplica en la edición al pasar de los pasos se dará una explicación gráfica.

Pasos Para el funcionamiento del programa:

Paso 1 (cuatro métodos de periodos)

Endpoint: /periods

Este Endpoint tiene los métodos GET, POST, PUT, DELETE.

El método GET (visualizar periodos), no tiene ningún problema se ingresa el link en el cliente de API y nos devuelve una respuesta.

Link: http://localhost:5000/periods [‘GET’]

http://localhost:5000/periods

Respuesta:


```
{
  "message": "list periods",
  "periods": [
    [
      1,
      2021,
      1
    ]
  ]
}
```

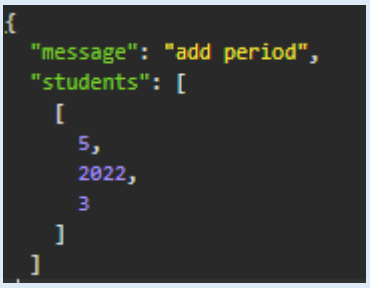
El método POST (crear periodos) tiene una serie de pasos extra se debe ingresar los campos de la tabla y los datos que se desea guardar para obtener respuesta.

Link: http://localhost:5000/periods [‘POST’]

http://localhost:5000/periods

Datos a ingresar en el JSON: ---------- > Respuesta:

```
{
        "year": "2022",
        "period": 3
}
```



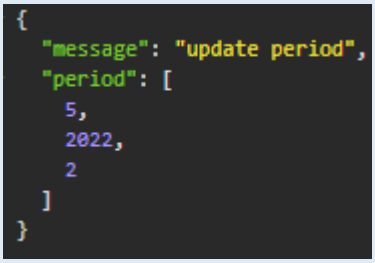
El método PUT (editar periodos) en este caso tiene un caso especial en cuanto al ID se bebe agregar tanto en la URL como en el JSON del dato que se desea actualizar.

Link: http://localhost:5000/periods/ID [‘PUT’]

Datos a ingresaren el JSON ejemplo: ----------> Respuesta:

http://localhost:5000/periods/5

```
{
        "id": 5,
        "year": 2022,
        "period": 2
}
```



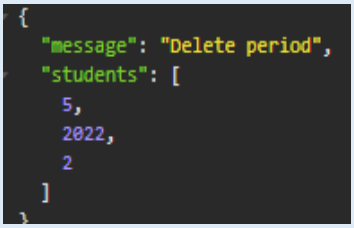
El método DELETE (eliminar periodos) en este caso el ID del periodo que se quiere eliminar se debe agregar en la URL.

Link: http://localhost:5000/periods/ID [‘DELETE’]

ejemplo:

http://localhost:5000/periods/5

Respuesta:



Paso 2 (cuatro métodos estudiantes)

Endpoint: /students

Este Endpoint tiene los métodos GET, POST, PUT, DELETE.

El método GET (visualizar estudiantes), no tiene ningún problema se ingresa el link en el cliente de API y nos devuelve una respuesta.

Link: http://localhost:5000/students [‘GET’]

http://localhost:5000/students

Respuesta:


```
{
  "message": "list students",
  "students": [
    [
      1,
      "1006789060",
      "yan",
      "cuaran",
      "31343443333",
      "yan@gmail.com",
      "5",
      "2021(1)"
    ]
  ]
}
```


El método POST (crear students) tiene una serie de pasos extra se debe ingresar los campos de la tabla y los datos que se desea guardar para obtener respuesta.

Link: http://localhost:5000/students [‘POST’]

http://localhost:5000/students

Datos a ingresar en el JSON: ---------- > Respuesta:


```
{
    "identification": "1001909019",
    "name": "juan",
    "surname": "portilla",
    "phone": "30899987878",
    "email": "juan@gmail.com",
    "semester": "3",
    "period_id": 2
}
```



```
{
  "message": "add student",
  "students": [
    [
      37,
      "1001909019",
      "juan",
      "portilla",
      "30899987878",
      "juan@gmail.com",
      "3",
      "2021(2)"
    ]
  ]
}
```

El método PUT (editar students) en este caso tiene un caso especial en cuanto al ID se bebe agregar tanto en la URL como en el JSON del dato que se desea actualizar.

Link: http://localhost:5000/students/ID [‘PUT’]

Datos a ingresaren el JSON ejemplo: ----------> Respuesta:

http://localhost:5000/students/37


```
{
        "id": 37,
        "identification": "1007866567",
        "name": "alexander",
        "surname": "mejia",
        "phone": "32156277272",
        "email": "alex@gmail.com",
        "semester": "5",
        "period_id": 4
}
```


El método DELETE (eliminar students) en este caso el ID del estudiante que se quiere eliminar se debe agregar en la URL.

Link: http://localhost:5000/students/ID [‘DELETE’]

ejemplo:

http://localhost:5000/students/31

Respuesta:

Paso 3 (cuatro métodos espacios académicos)

Endpoint: /spaces

Este Endpoint tiene los métodos GET, POST, PUT, DELETE.

El método GET (visualizar spaces), no tiene ningún problema se ingresa el link en el cliente de API y nos devuelve una respuesta.

Link: http://localhost:5000/spaces [‘GET’]

http://localhost:5000/spaces

Respuesta:


```
{
  "message": "list academic spaces",
  "students": [
    [
      1,
      "2021(1)",
      "Electiva",
      "5"
    ]
  ]
}
```


El método POST (crear spaces) tiene una serie de pasos extra se debe ingresar los campos de la tabla y los datos que se desea guardar para obtener respuesta.

Link: http://localhost:5000/spaces [‘POST’]

http://localhost:5000/spaces

Datos a ingresar en el JSON: ----------> Respuesta:


```
{
        "period_id": 1,
        "name": "Ingles II333",
        "semester": "4"
}
```



```
{
  "message": "add space",
  "spaces": [
    [
      9,
      "2021(1)",
      "Ingles II333",
      "4"
    ]
  ]
}
```

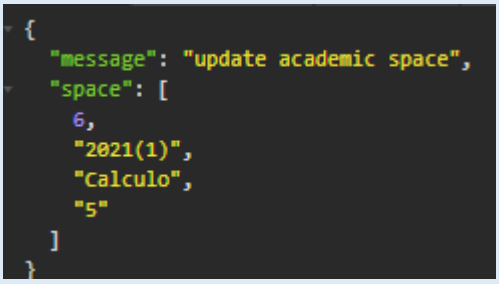
El método PUT (editar spaces) en este caso tiene un caso especial en cuanto al ID se bebe agregar tanto en la URL como en el JSON del dato que se desea actualizar.

Link: http://localhost:5000/spaces/ID [‘PUT’]

Datos a ingresaren el JSON ejemplo: -----> Respuesta:

http://localhost:5000/spaces/6


```
{
    "id": 6,
    "period_id": 1,
    "name": "Calculo",
    "semester": "5"
}
```

El método DELETE (eliminar spaces) en este caso el ID del espacio académico que se quiere eliminar se debe agregar en la URL.

Link: http://localhost:5000/spaces/ID [‘DELETE’]

ejemplo:

http://localhost:5000/spaces/9

Respuesta:

Paso 4 (cuatro métodos sesiones de clases)

Endpoint: /sessions

Este Endpoint tiene los métodos GET, POST, PUT, DELETE.

El método GET (visualizar sessions), no tiene ningún problema se ingresa el link en el cliente de API y nos devuelve una respuesta.

Link: http://localhost:5000/sessions [‘GET’]

http://localhost:5000/sessions

Respuesta:

El método POST (crear sessions) tiene una serie de pasos extra se debe ingresar los campos de la tabla y los datos que se desea guardar para obtener respuesta.

Link: http://localhost:5000/sessions [‘POST’]

http://localhost:5000/sessions

Datos a ingresar en el JSON: ---------- > Respuesta:

{
"academic_space_id": 3,
"cut": 3,
"date": "06-06-2021",
"start_time": "08:00",
"end_time": "00:00"
}

El método PUT (editar sessions) en este caso tiene un caso especial en cuanto al ID se bebe agregar tanto en la URL como en el JSON del dato que se desea actualizar.

Link: http://localhost:5000/sessions/ID [‘PUT’]

Datos a ingresaren el JSON ejemplo: ----------> Respuesta:

http://localhost:5000/sessions/9


```
{
        "id": 9,
        "academic_space_id": 3,
        "cut": 2,
        "date": "06-06-2022",
        "start_time": "08:00",
        "end_time": "07:00"
}
```



```
{
  "message": "update session",
  "session": [
    9,
    "Electiva II",
    "cut (2)",
    "06-06-2022",
    "08:00",
    "07:00"
  ]
}
```


El método DELETE (eliminar sessions) en este caso el ID de la sesión que se quiere eliminar se debe agregar en la URL.

Link: http://localhost:5000/sessions/ID [‘DELETE’]

ejemplo:

http://localhost:5000/sessions/9

Respuesta:


```
{
  "message": "Delete session",
  "session": [
    9,
    "Electiva II",
    "cut (2)",
    "06-06-2022",
    "08:00",
    "07:00"
  ]
}
```


Paso 5 (cuatro métodos actividades realizadas)

Endpoint: /activities

Este Endpoint tiene los métodos GET, POST, PUT, DELETE.

El método GET (visualizar activities), no tiene ningún problema se ingresa el link en el cliente de API y nos devuelve una respuesta.

Link: http://localhost:5000/activities [‘GET’]

http://localhost:5000/activities

Respuesta:


```
{
  "message": "list performed activities",
  "pf_activities": [
    [
      1,
      "Electiva",
      "cut (2)",
      "exposicion",
      "3.78"
    ]
  ]
}
```


El método POST (crear activities) tiene una serie de pasos extra se debe ingresar los campos de la tabla y los datos que se desea guardar para obtener respuesta.

Link: http://localhost:5000/activities [‘POST’]

http://localhost:5000/activities

Datos a ingresar en el JSON: ---------- > Respuesta:


```
{
    "academic_space_id": 4,
    "cut": 3,
    "name": "pronunciacion",
    "average_cut": "3.9"
}
```



```
{
  "message": "add performed activities",
  "pf_activities": [
    [
      7,
      "Integrales",
      "cut (3)",
      "pronunciacion",
      "3.9"
    ]
  ]
}
```


El método PUT (editar activities) en este caso tiene un caso especial en cuanto al ID se bebe agregar tanto en la URL como en el JSON del dato que se desea actualizar.

Link: http://localhost:5000/activities/ID [‘PUT’]

Datos a ingresaren el JSON ejemplo: ----------> Respuesta:

http://localhost:5000/activities/7


```
{
    "id": 7,
    "academic_space_id": 5,
    "cut": 1,
    "name": "pronunciacion",
    "average_cut": "4.0"
}
```



```
{
  "message": "update activity",
  "pf_activity": [
    7,
    "Ingles III",
    "cut (1)",
    "pronunciacion",
    "4.0"
  ]
}
```

El método DELETE (eliminar activities) en este caso el ID de la actividad que se quiere eliminar se debe agregar en la URL.

Link: http://localhost:5000/activities/ID [‘DELETE’]

ejemplo:

http://localhost:5000/activities/7

Respuesta:


```
{
  "message": "Delete activity",
  "pf_activity": [
    7,
    "Ingles III",
    "cut (1)",
    "pronunciacion",
    "4.0"
  ]
}
```


Paso 6 (cuatro métodos de notas de estudiantes)

Endpoint: /notes

Este Endpoint tiene los métodos GET, POST, PUT, DELETE.

El método GET (visualizar notes), no tiene ningún problema se ingresa el link en el cliente de API y nos devuelve una respuesta.

Link: http://localhost:5000/notes [‘GET’]

http://localhost:5000/notes

Respuesta:


```
{
  "message": "list students notes",
  "notes": [
    [
      1,
      "exposicion",
      "yan",
      "3.5",
      "mejorar presentacion"
    ]
  ]
}
```

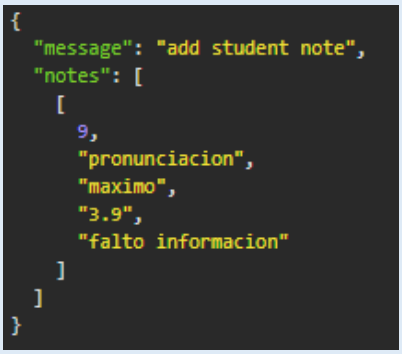
El método POST (crear notes) tiene una serie de pasos extra se debe ingresar los campos de la tabla y los datos que se desea guardar para obtener respuesta.

Link: http://localhost:5000/notes [‘POST’]

http://localhost:5000/notes

Datos a ingresar en el JSON: ---------- > Respuesta:

```
{
        "performed_activity_id": 5,
        "student_id": 29,
        "note": "3.9",
        "observation": "falto informacion"
}
```



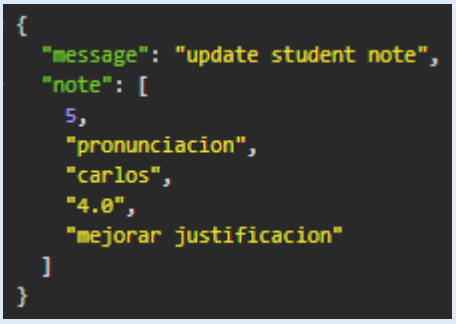
El método PUT (editar notes) en este caso tiene un caso especial en cuanto al ID se bebe agregar tanto en la URL como en el JSON del dato que se desea actualizar.

Link: http://localhost:5000/notes/ID [‘PUT’]

Datos a ingresaren el JSON ejemplo: ----------> Respuesta:

http://localhost:5000/notes/5

```
{
        "id": 5,
        "performed_activity_id": 3,
        "student_id": 28,
        "note": "4.0",
        "observation": "mejorar justificacion"
}
```



El método DELETE (eliminar notes) en este caso el ID de la nota del estudiante que se quiere eliminar se debe agregar en la URL.

Link: http://localhost:5000/notes/ID [‘DELETE’]

ejemplo:

http://localhost:5000/notes/8

Respuesta:


```
{
  "message": "Delete student note",
  "note": [
    8,
    "pronunciacion",
    "maximo",
    "3.9",
    "falto informacion"
  ]
}
```


Paso 7 (cuatro métodos de asistencias estudiantes)

Endpoint: /assists

Este Endpoint tiene los métodos GET, POST, PUT, DELETE.

El método GET (visualizar assists), no tiene ningún problema se ingresa el link en el cliente de API y nos devuelve una respuesta.

Link: http://localhost:5000/assists [‘GET’]

http://localhost:5000/assists

Respuesta:


```
{
  "message": "list assists",
  "s_assists": [
    [
      1,
      "10-03-2021",
      "yan",
      1
    ]
  ]
}
```


El método POST (crear assists) tiene una serie de pasos extra se debe ingresar los campos de la tabla y los datos que se desea guardar para obtener respuesta.

Link: http://localhost:5000/assists [‘POST’]

http://localhost:5000/assists

Datos a ingresar en el JSON ejemplo: ----------> Respuesta:

```
{
        "session_id": 2,
        "student_id": 1,
        "assistance": 1
}
```

El método PUT (editar assists) en este caso tiene un caso especial en cuanto al ID se bebe agregar tanto en la URL como en el JSON del dato que se desea actualizar.

Link: http://localhost:5000/assists/ID ['PUT']

Datos a ingresaren el JSON ejemplo: ---------- > Respuesta:

http://localhost:5000/assists/8

```
{
        "id": 8,
        "session_id": 3,
        "student_id": 28,
        "assistance": 1
}
```

El método DELETE (eliminar assists) en este caso el ID de la asistencia que se quiere eliminar se debe agregar en la URL.

Link: http://localhost:5000/assists/ID ['DELETE']

ejemplo:

http://localhost:5000/assists/8

Respuesta: