

Manual Funcionamiento de la Aplicación.

Antes de compilar y ejecutar la aplicación se debe asegurar que el archivo `application.properties` ubicado en la carpeta `src/main/resources` contenga los datos adecuados para la conexión a su base de datos creada previamente.

Los datos a modificar en el archivo `application.properties` son:

```
Spring.datasource.url
Spring.datasource.username
Spring.datasource.password
```

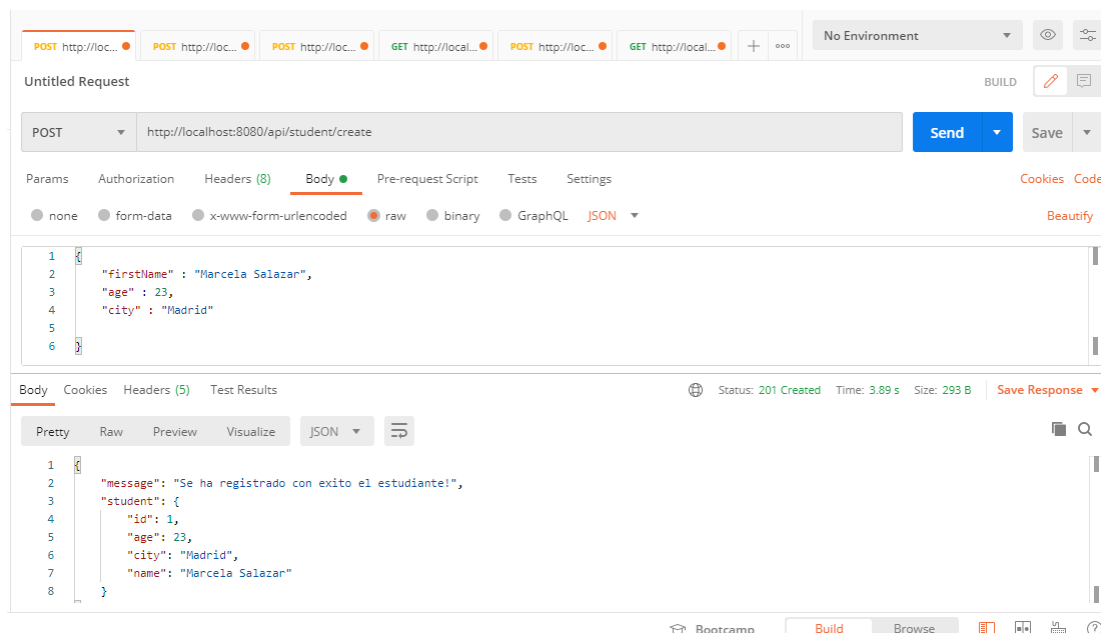
Para una base de datos de Mysql que corresponderán a la URL , nombre de usuario y password de la base de datos, según la configuración de su entorno.

Una vez configurada la conexión a la base de datos ejecutamos la aplicación como una aplicación de springboot.

La colección de peticiones se encontrarán en el repositorio enviado.

1. Crear un conjunto de estudiantes.

Por medio de la petición que se muestra en la imagen se puede crear varios estudiantes modificando los campos que se muestran en el json a conveniencia.



The screenshot shows a SQL IDE interface with a script editor on the left and a results/output pane on the right. The script editor contains the following SQL code:

```

30
31 CREATE TABLE question_exam(
32   id MEDIUMINT NOT NULL AUTO_INCREMENT PRIMARY KEY,
33   id_question MEDIUMINT NOT NULL,
34   id_exam MEDIUMINT NOT NULL,
35   option_selected VARCHAR(150)
36 );
37
38
39 SELECT * FROM student;
40 SELECT * FROM question;

```

The results pane shows the output of the SQL execution. The first query, `SELECT * FROM student;`, returned 3 rows of data:

id	firstName	age	city
1	Marcela Salazar	23	Madrid
2	Andres Mendoza	27	Medellin
3	Maria Perez	32	Call

The second query, `SELECT * FROM question;`, returned 0 rows of data.

The output pane also shows the execution log with the following entries:

#	Time	Action	Message	Duration / F
10	14:50:09	CREATE TABLE question(id MEDIUMINT NOT NULL AUTO_INCREMENT PRIMARY KE...	0 row(s) affected	0.203 sec
11	14:52:08	CREATE TABLE question_exam(id MEDIUMINT NOT NULL AUTO_INCREMENT PRIMA...	0 row(s) affected	0.125 sec
12	14:58:46	SELECT * FROM student LIMIT 0, 1000	1 row(s) returned	0.015 sec /
13	14:59:55	SELECT * FROM student LIMIT 0, 1000	3 row(s) returned	0.000 sec /

2. Crear un conjunto de preguntas.

Por medio de la petición que se muestra en la imagen se puede crear una pregunta y registrarla modificando los campos que se muestran en el json a conveniencia.

The screenshot shows a REST client interface with a POST request to `http://localhost:8080/api/question/create`. The request body is a JSON object:

```

1 {
2   "description": "En que año nacio facebook",
3   "optionA": "2006",
4   "optionB": "2014",
5   "optionC": "2007",
6   "optionD": "1990",
7   "optionOk": "C"
8 }

```

The response body is a JSON object:

```

1 {
2   "message": "Se ha creado con exito la pregunta!",
3   "question": {
4     "id": 1,
5     "option_a": "2006",
6     "option_b": "2014",
7     "option_c": "2007"
8   }
9 }

```

The status bar at the bottom indicates the request was successful with a status of 201 Created, a time of 153 ms, and a size of 387 B.

scripts base de datos

```

32 id MEDIUMINT NOT NULL AUTO_INCREMENT PRIMARY KEY,
33 id_question MEDIUMINT NOT NULL,
34 id_exam MEDIUMINT NOT NULL,
35 option_selected VARCHAR(150)
36 );
37
38
39 • SELECT * FROM student;
40 • SELECT * FROM question;
41 • SELECT * FROM question_exam;
42 • SELECT * FROM exam;

```

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

id	description_q	option_a	option_b	option_c	option_d	correct_option
1	En que año nacio facebook	2006	2014	2007	1990	C
2	Que lenguaje maneja spring framework	Java	Python	C++	.Net	A
3	Cual es la capital de españa	Bogota	Asuncion	Cal	Madrid	D

question 5 x

Output

#	Time	Action	Message	Duration / Fetch
13	14:59:55	SELECT * FROM student LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec
14	15:04:04	SELECT * FROM student LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec
15	15:04:16	SELECT * FROM question LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec
16	15:07:35	SELECT * FROM question LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec

3. Crear un examen.

Por medio de la petición que se muestra en la imagen se puede crear un examen, registrarlo y asignarlo a un estudiante modificando los campos que se muestran en el json , donde la lista question debe contener los ids de cada una de las preguntas que desee asignar a estudiante, idEstudiante corresponde al id del estudiante asignado.

POST http://localhost:8080/api/exam/create

Send Save

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

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```

1 {
2   "idEstudiante": 1,
3   "questions": [2,3,5]
4 }

```

Body Cookies Headers (5) Test Results

Status: 201 Created Time: 893 ms Size: 311 B Save Response

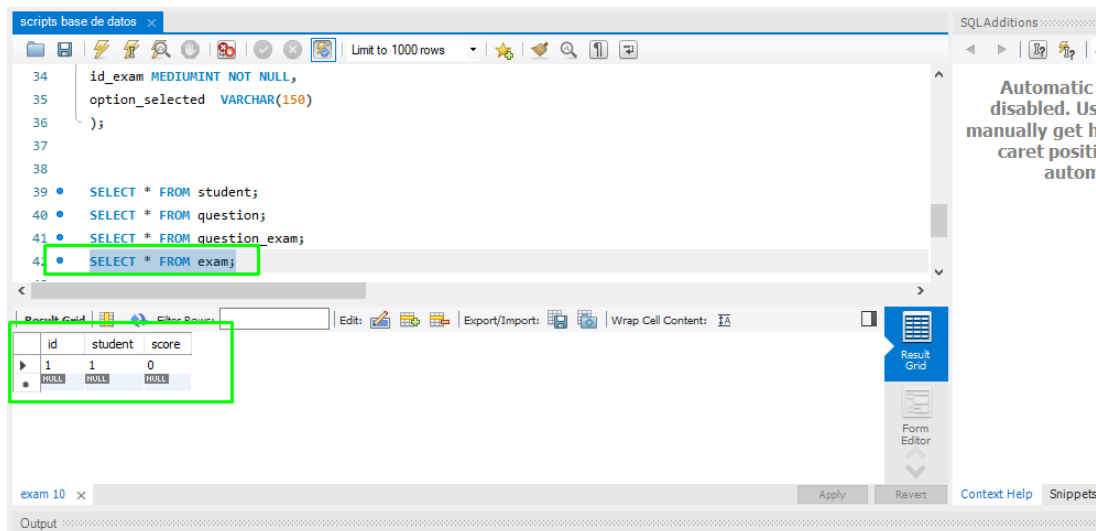
Pretty Raw Preview Visualize JSON

```

3   "exam": {
4     "id": 1,
5     "student": {
6       "id": 1,
7       "age": 23,
8       "city": "Madrid",
9       "name": "Marcela Salazar"
10    },
11    "score": 0
12  }

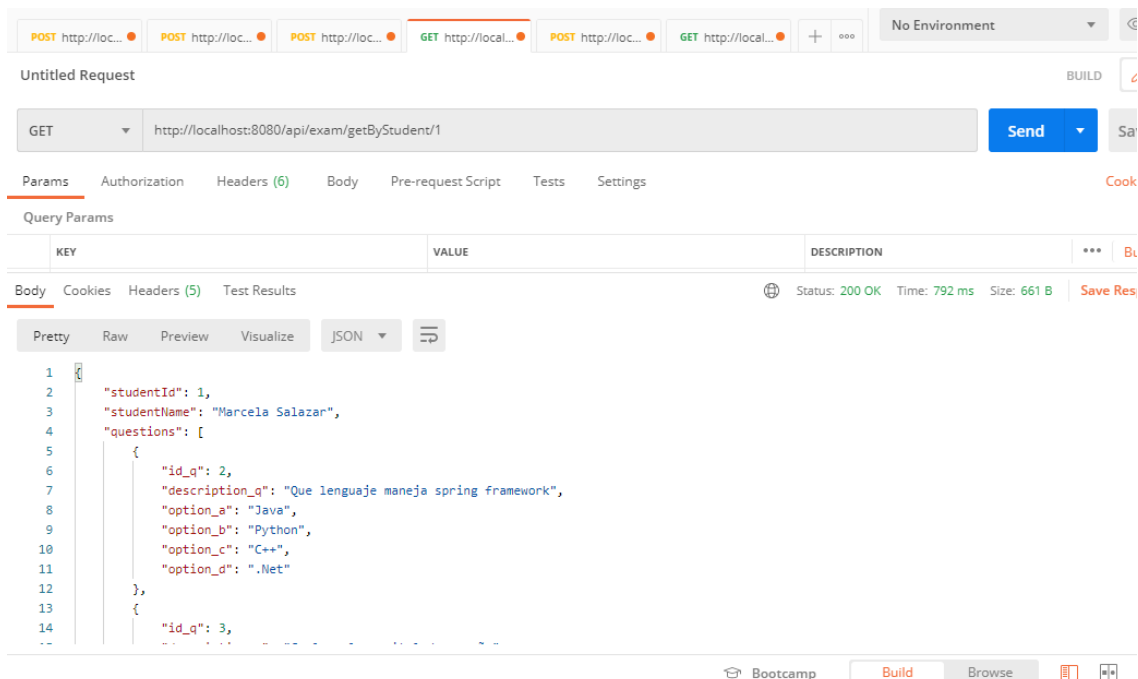
```

Bootcamp Build Browse



4. Obtener un examen creado especificando un id de estudiante.

Por medio de la petición que se muestra en la imagen se puede consultar el examen registrado enviando como parámetro en la URL el id del estudiante a consultar.



5. Responder un examen.

Por medio de la petición que se muestra en la imagen se puede responder el examen registrado, nótese que el Json enviado contiene el id del estudiante y una lista que contiene las distintas preguntas asignadas, estas contienen el id de la pregunta y la opción que seleccionó el estudiante.

El servicio responde con el nombre del estudiante y la calificación del examen.

POST http://localhost:8080/api/exam/answerExam

Send Save

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

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "idStudent": 1,
3   "answers": [
4     {
5       "idQuestion": 2,
6       "selected": "C"
7     },
8     {
9       "idQuestion": 3,
10      "selected": "D"
11    }
12  ]
13 }
```

Body Cookies Headers (5) Test Results

Status: 200 OK Time: 168 ms Size: 220 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "idStudent": 1,
3   "firstName": "Marcela Salazar",
4   "score": 66
5 }
```

Bootcamp Build Browse

scripts base de datos

Limit to 1000 rows

```
33 id_question MEDIUMINT NOT NULL,
34 id_exam MEDIUMINT NOT NULL,
35 option_selected VARCHAR(150)
36 );
37
38
39 • SELECT * FROM student;
40 • SELECT * FROM question;
41 • SELECT * FROM question_exam;
```

Result Grid

	id	id_question	id_exam	option_selected
▶	1	2	1	C
	2	3	1	D
	3	5	1	D
	4	1	2	NULL
	5	2	2	NULL
	6	1	3	NULL
	7	2	3	NULL

question_exam 18

Output

SQLAdditions

Automatic context h disabled. Use the tool manually get help for th caret position or to automatic help.

Result Grid Form Editor Context Help Snippets

scripts base de datos

Limit to 1000 rows

```
34 id_exam MEDIUMINT NOT NULL,
35 option_selected VARCHAR(150)
36 );
37
38
39 • SELECT * FROM student;
40 • SELECT * FROM question;
41 • SELECT * FROM question_exam;
42 • SELECT * FROM exam;
```

Result Grid

	id	student	score
▶	1	1	66
	2	2	0
	3	3	0

exam 19

Output

SQLAdditions

A dis manu c

Result Grid Form Editor Context Help

6. Recopilar las respuestas enviadas por un estudiante.

Por medio de la petición que se muestra en la imagen se puede obtener todas las respuestas seleccionadas por un estudiante, enviando como parámetro en la URL el id del estudiante.

El servicio responde con el nombre del estudiante, un conjunto de preguntas y la opción seleccionada, la calificación del examen.

