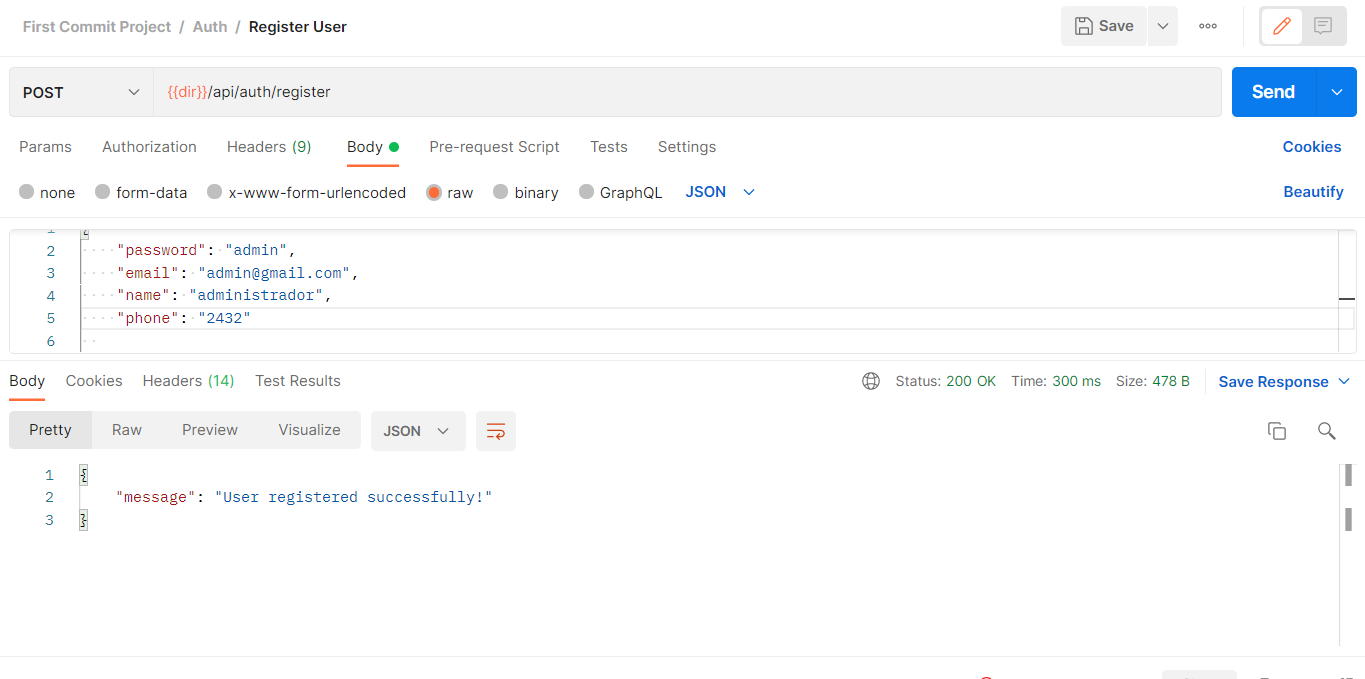
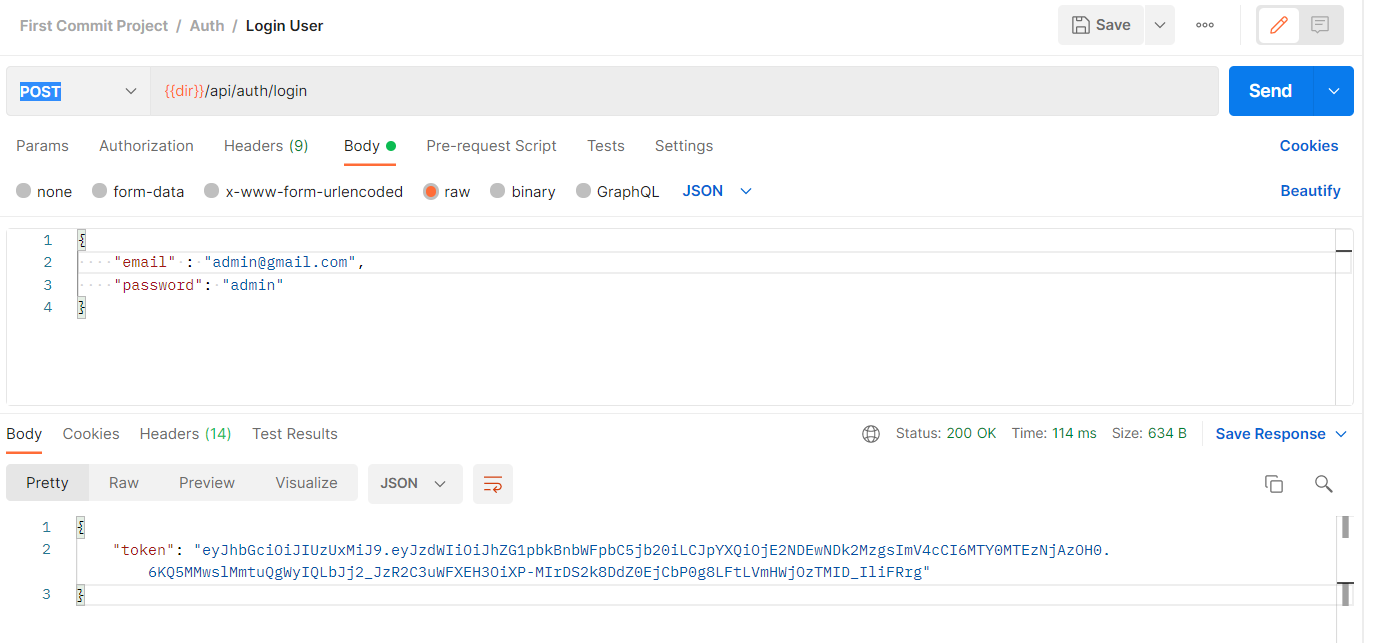
**Autorización**

**El usuario se registra**



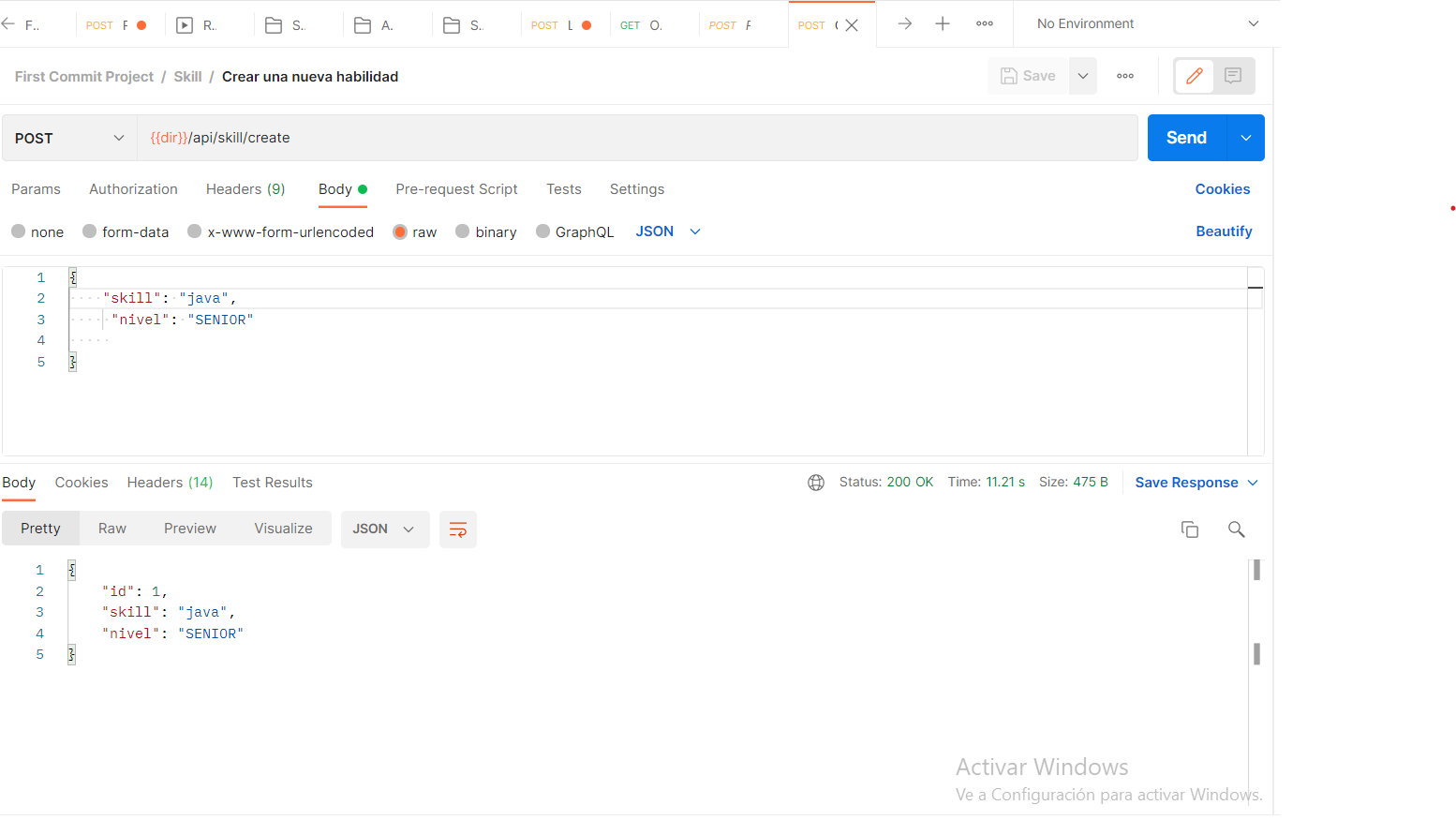
El teléfono no es obligatorio. Los otros datos si. El email se chequea no exista.

**El usuario se loguea**

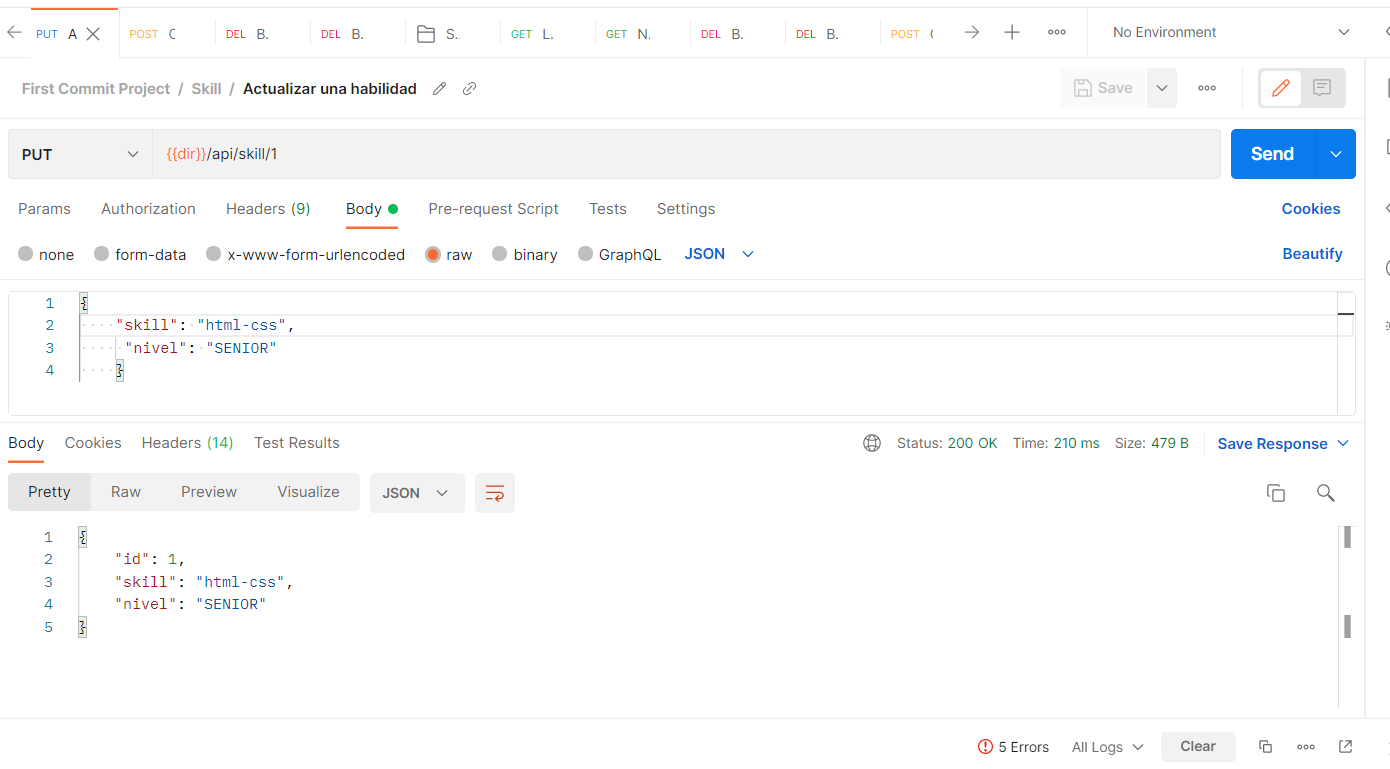


**Habilidades**

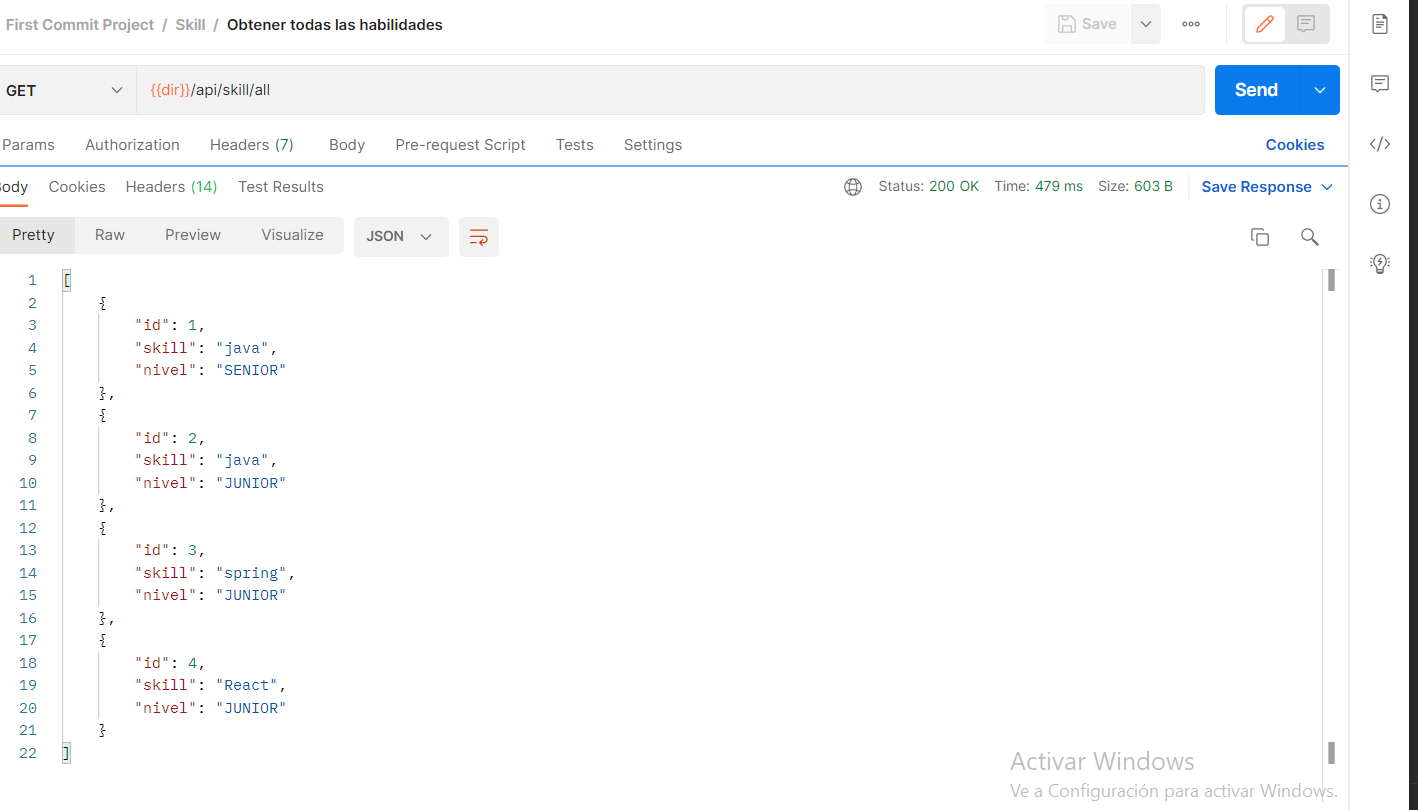
**Se crean**



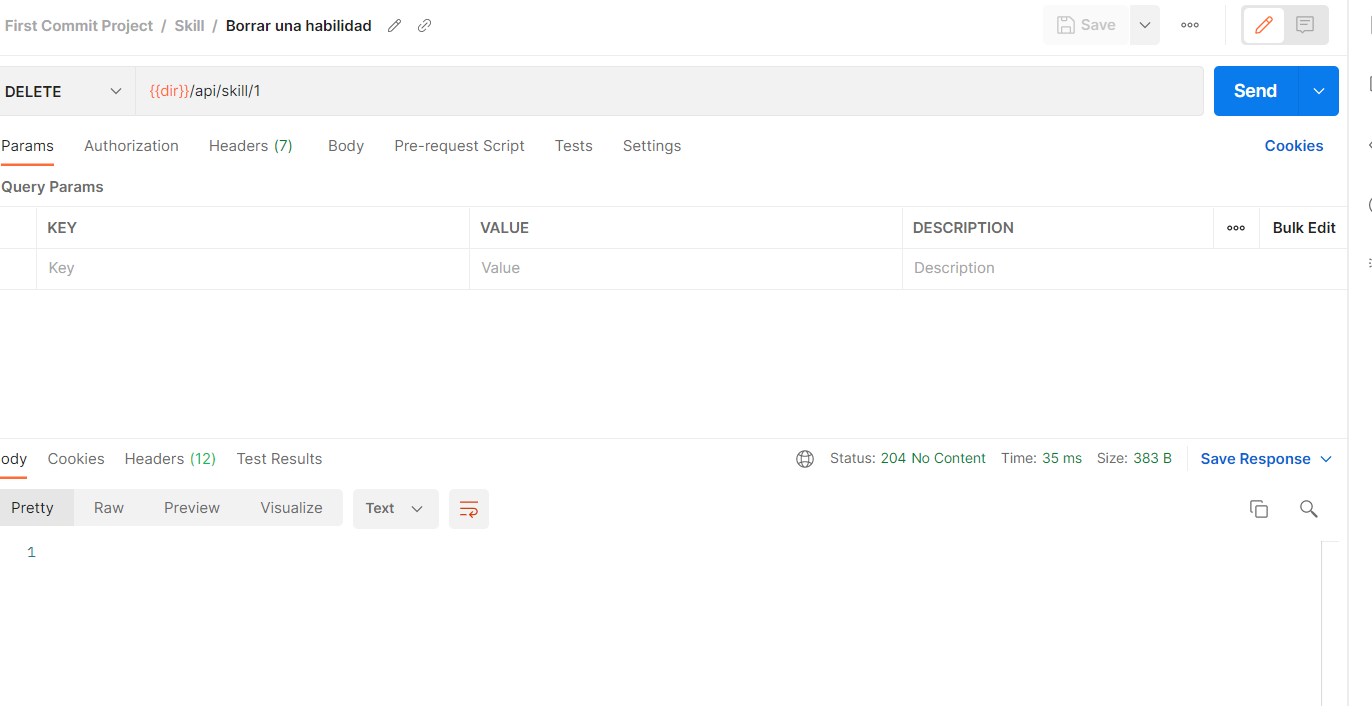
**Se actualizan**

****

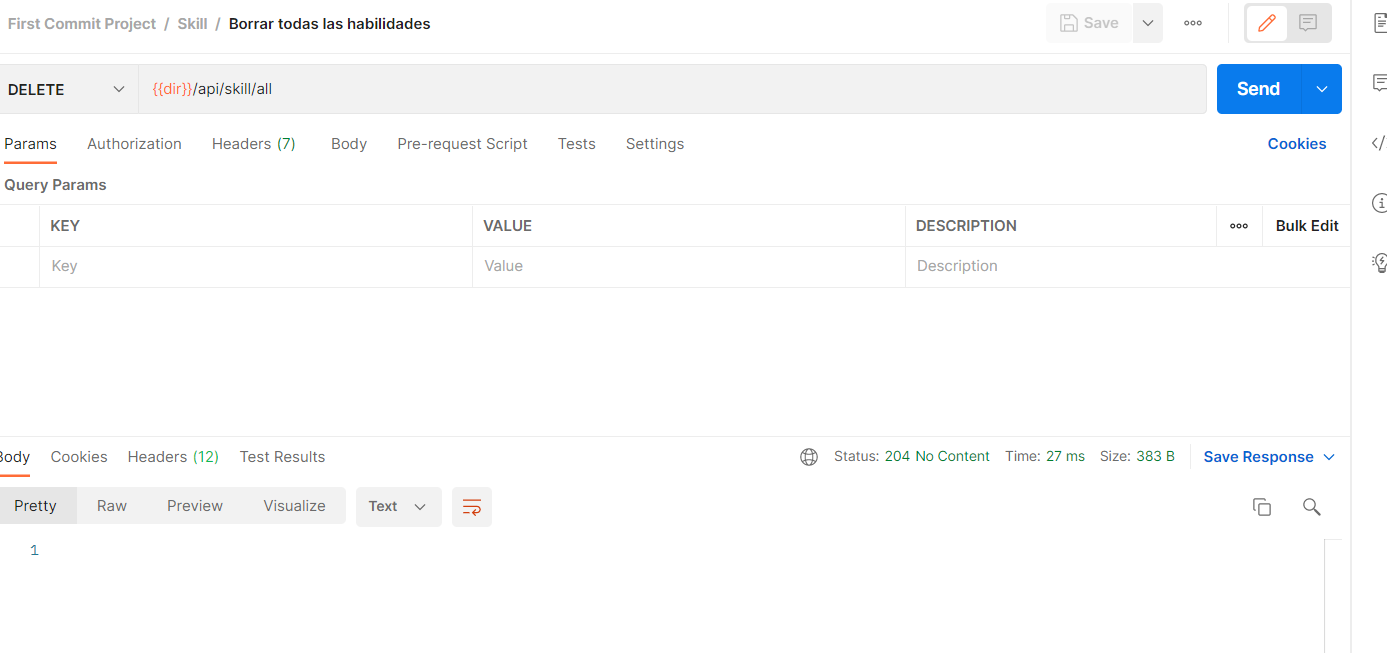
**Se muestran**



**Se borran**

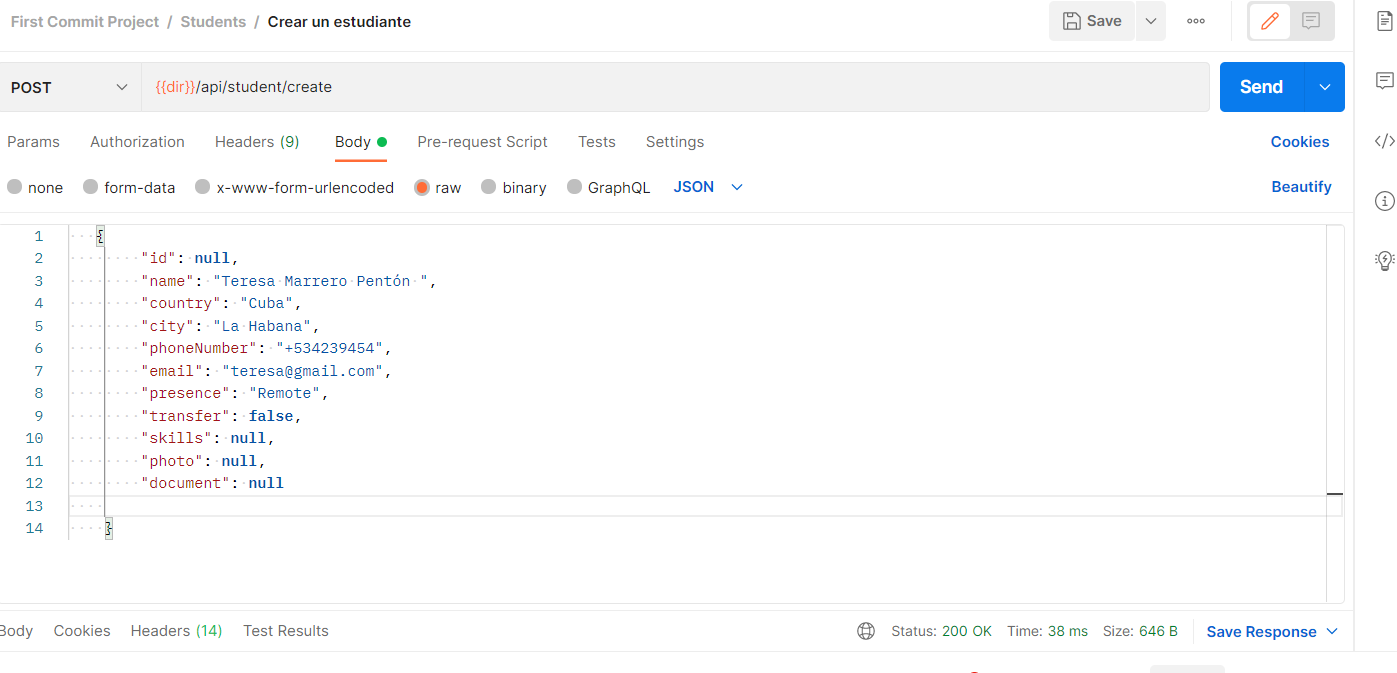


**Se borran todas**

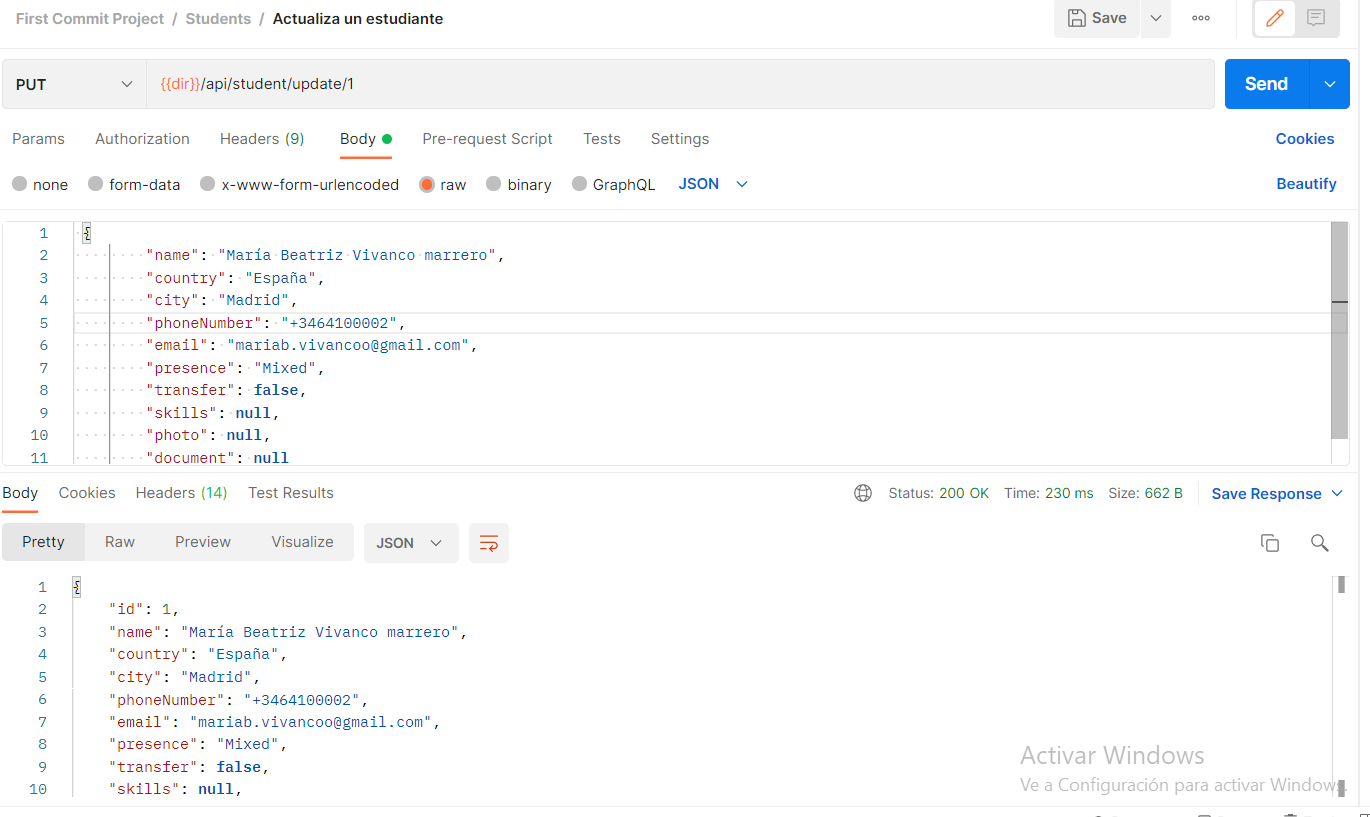


**Estudiantes** (Se hacen todas las operaciones en función del usuario que esté logueado)

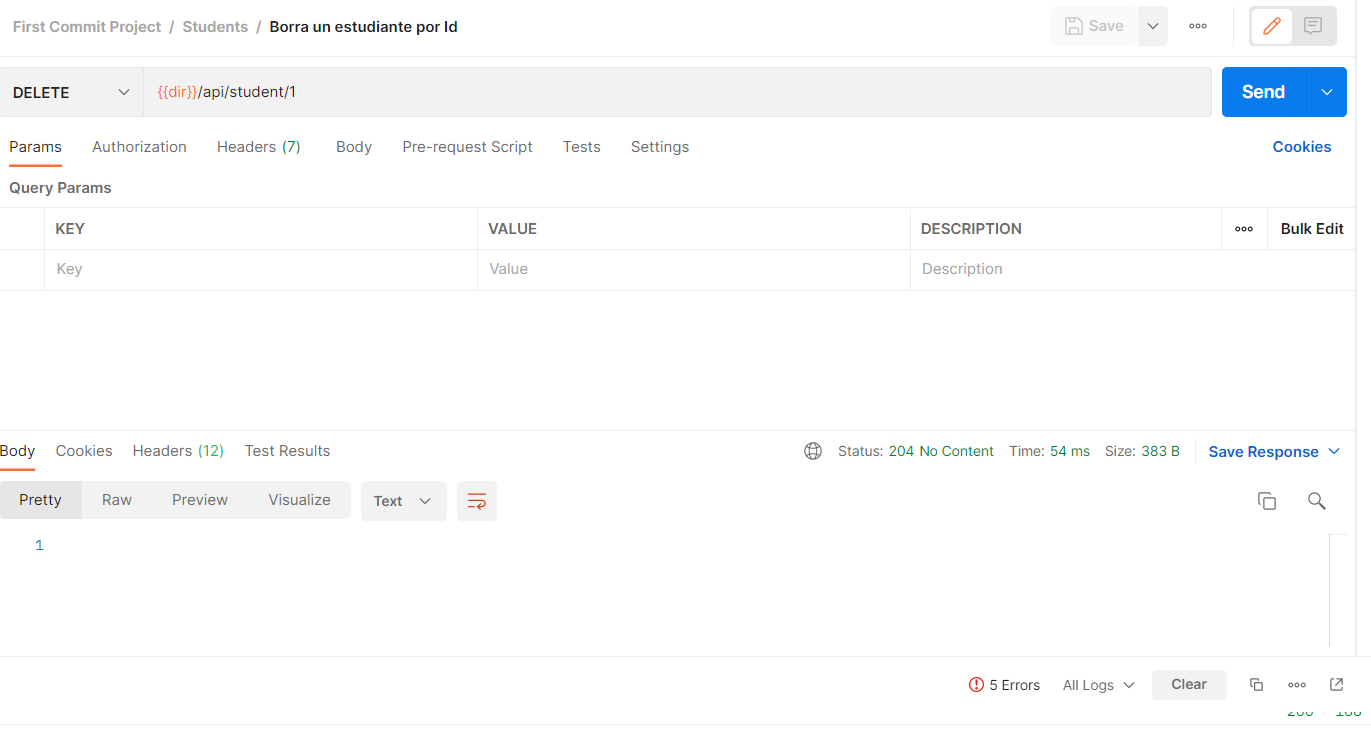
**Se crean**

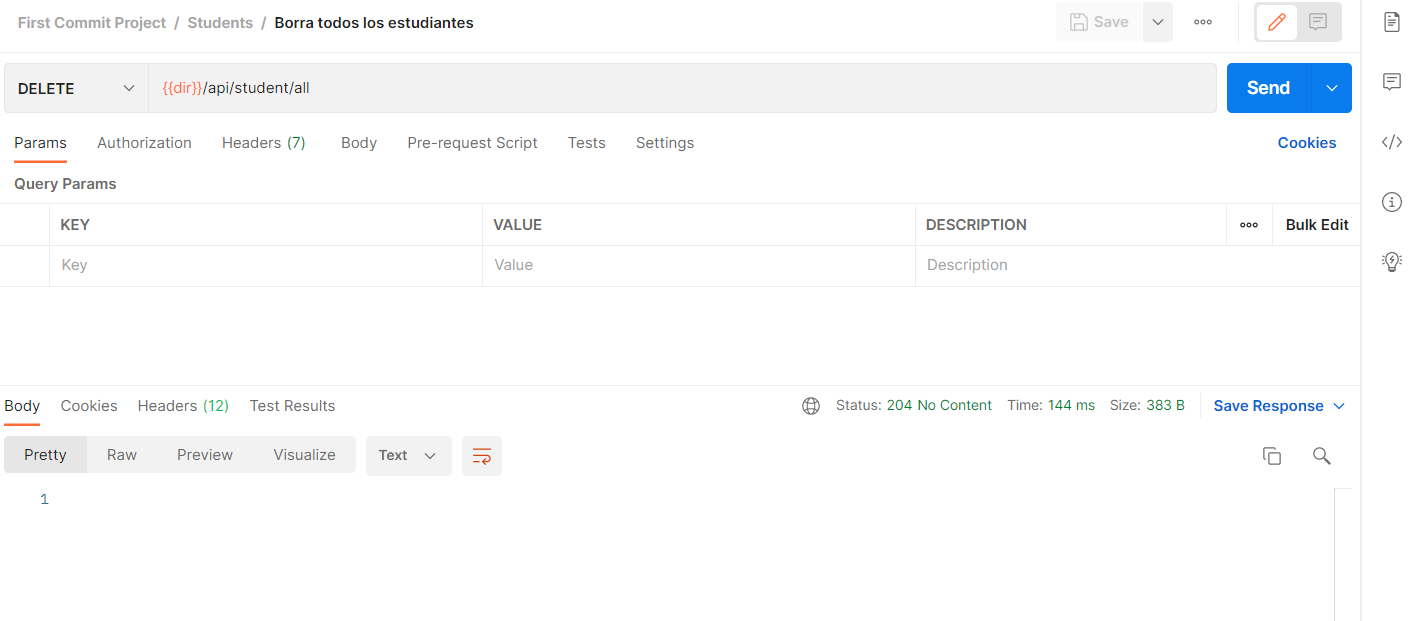


**Se actualizan**

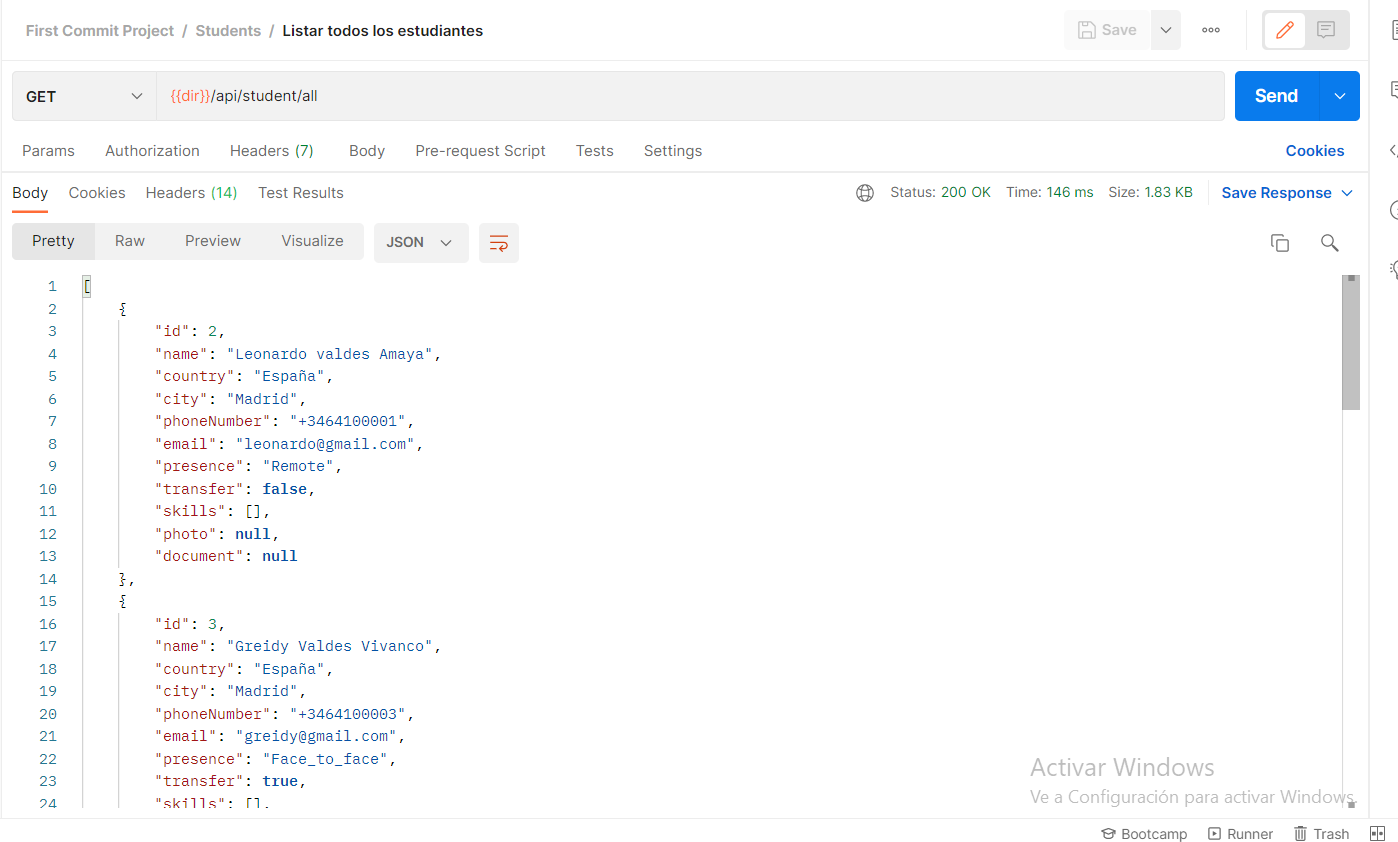


**Se borra uno**

**Se borran todos**



**Se obtienen todos**



También se pueden obtener estos mismos datos por los diferentes filtros que aún no los tengo implementados

@GetMapping("/all")  
public List<Student> getAllStudent() {  
 return studentRepository.findAll();  
}  
  
@GetMapping("/city/{nameCity}")  
public List<Student> getStudentByCity(@PathVariable String nameCity) {  
   
}  
  
@GetMapping("/presence/{typePresence}")  
public List<Student> getStudentByPresence(@PathVariable Presence typePresence) {  
   
}  
  
@GetMapping("/country/{nameCountry}")  
public List<Student> getStudentByCountry(@PathVariable String nameCountry) {  
   
}  
  
@GetMapping("/transfer/typePresence}")  
public List<Student> getStudentByTransfery(@PathVariable Boolean typeTransfer) {  
   
}  
  
@GetMapping("/skills")  
public List<Student> getStudentBySkylls(@RequestBody Set<Skill> skills) {  
   
}

**Cosas Importantes a tener en cuenta que aún no he implementado:**

* Recordar contraseña.
* El envío de email y lograr la recuperación de la contraseña del usuario.
* Validar los datos de entrada
* Actualizar solo los datos que me proporcionen en los casos de los put
* Implementar los filtros de búqueda de la manera más eficiente (Hacer consulta a Alan si por Spring o query de hibernate)
* Implementar la salva y actualización del documento e imagen usando Cloudinary.

Ejemplo de no autorizado

