



# Acceso a datos

ActUT05. Creación de servicios web con Spring Boot

Francisco José García Cutillas | 2FPGS\_DAM



## Índice

Properties.....	3
Estructura de paquetes .....	3
Appinstituto.....	4
Controller.....	4
Model.....	6
Repository.....	10
Service .....	11
Dto .....	14
Utilidades.....	14
Controller.....	15

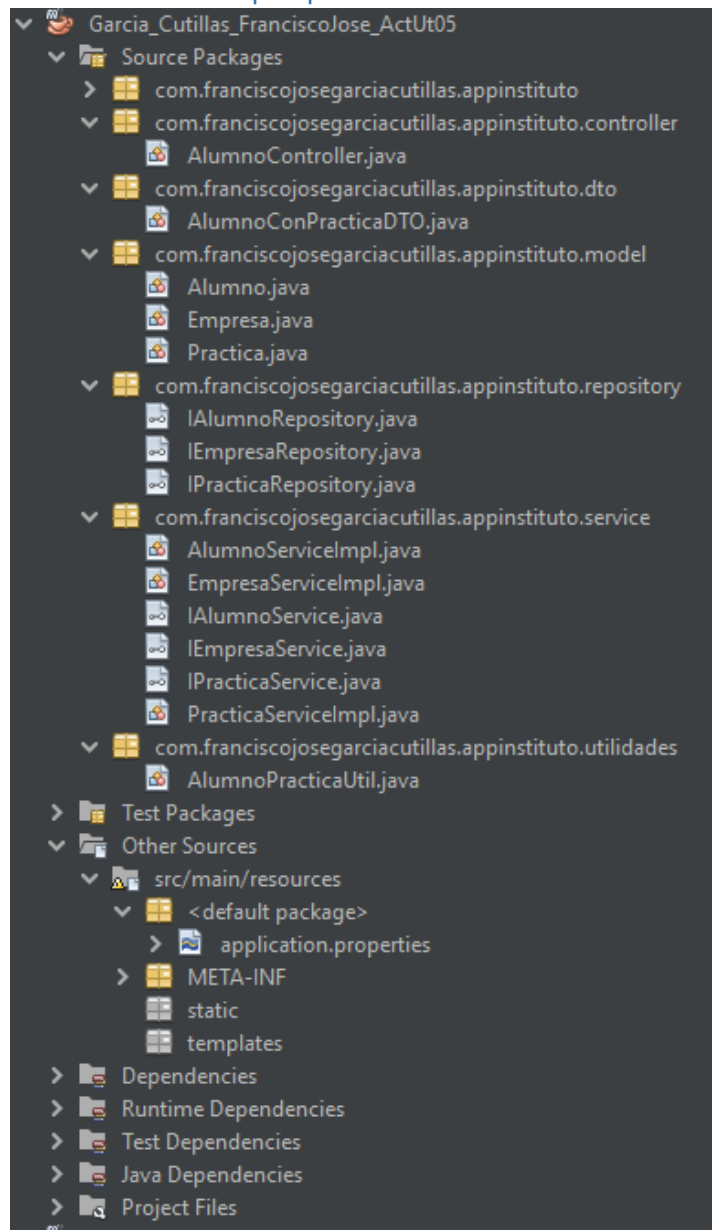
## Properties

```

1  spring.jpa.database=mysql
2  spring.jpa.hibernate.ddl-auto=none
3  spring.jpa.show-sql: true
4
5  spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
6  spring.datasource.url=jdbc:mysql://localhost:3306/fct
7  spring.datasource.username=root
8  spring.datasource.password=root
9

```

## Estructura de paquetes



## Appinstituto

```

1 package com.franciscojosegarciaacutillas.appinstituto;
2
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5
6 @SpringBootApplication
7 public class AppinstitutoApplication {
8
9     public static void main(String[] args) {
10         SpringApplication.run(AppinstitutoApplication.class, args);
11     }
12
13 }
14

```

## Controller

```

1 package com.franciscojosegarciaacutillas.appinstituto.controller;
2
3 import com.franciscojosegarciaacutillas.appinstituto.dto.AlumnoConPracticaDTO;
4 import com.franciscojosegarciaacutillas.appinstituto.model.Alumno;
5 import com.franciscojosegarciaacutillas.appinstituto.model.Practica;
6 import com.franciscojosegarciaacutillas.appinstituto.service.AlumnoServiceImpl;
7 import com.franciscojosegarciaacutillas.appinstituto.service.PracticaServiceImpl;
8 import com.franciscojosegarciaacutillas.appinstituto.utilidades.AlumnoPracticaUtil;
9 import java.util.ArrayList;
10 import java.util.Objects;
11 import org.springframework.beans.factory.annotation.Autowired;
12 import org.springframework.http.HttpStatus;
13 import org.springframework.http.ResponseEntity;
14 import org.springframework.web.bind.annotation.DeleteMapping;
15 import org.springframework.web.bind.annotation.GetMapping;
16 import org.springframework.web.bind.annotation.PathVariable;
17 import org.springframework.web.bind.annotation.PostMapping;
18 import org.springframework.web.bind.annotation.PutMapping;
19 import org.springframework.web.bind.annotation.RequestBody;
20 import org.springframework.web.bind.annotation.RequestMapping;
21 import org.springframework.web.bind.annotation.RestController;
22
23 /**
24  *
25  * @author Fran
26  */
27 @RestController
28 @RequestMapping("/api/v1")
29 public class AlumnoController {
30
31     @Autowired
32     AlumnoServiceImpl aluServ;
33
34     @Autowired
35     PracticaServiceImpl pracServ;
36
37     //1.Recuperar todos los alumnos
38     @GetMapping("/alumnos")
39     public ResponseEntity<ArrayList<AlumnoConPracticaDTO>> getAlumnosConPractica() {
40
41         ArrayList<Alumno> alumnos = aluServ.listar();
42         ArrayList<AlumnoConPracticaDTO> alumnosDevolver = new ArrayList<>();
43
44         for (Alumno alumno : alumnos) {
45
46             AlumnoConPracticaDTO nuevoAlumno = new AlumnoConPracticaDTO(alumno);
47             alumnosDevolver.add(nuevoAlumno);
48
49         }
50
51         return new ResponseEntity<> (body: alumnosDevolver, status: HttpStatus.OK);
52     }
53
54     //2.Recuperar los datos de un alumno en concreto
55     @GetMapping("/alumnos/{id}")
56     public ResponseEntity<Alumno> getAlumnoId(@PathVariable Integer id) {
57
58         Alumno alumno = aluServ.obtener(id);
59
60     }
61

```

```

61         if (alumno.getAlumnoId() == null) {
62             return new ResponseEntity<> (status: HttpStatus.NOT_FOUND);
63         } else {
64             return new ResponseEntity<> (body: alumno, status: HttpStatus.OK);
65         }
66     }
67 }
68
69 //3. Insertar un alumno sin práctica
70 @PostMapping("/alumnos")
71 public ResponseEntity<Alumno> postAlumnoSinPractica(@RequestBody Alumno alumno) {
72     if (alumno.getAlumnoId() != null) {
73         Alumno alumnoExiste = aluServ.obtener(id: alumno.getAlumnoId());
74         if (alumnoExiste.getAlumnoId() != null) {
75             return new ResponseEntity<> (status: HttpStatus.CONFLICT);
76         }
77     }
78     Alumno alumnoInsertar = aluServ.registrar(alumno);
79     return new ResponseEntity<> (body: alumnoInsertar, status: HttpStatus.CREATED);
80 }
81
82 //4. Actualizar datos de un alumno
83 @PutMapping("/alumnos/{id}")
84 public ResponseEntity<Alumno> putAlumno(@RequestBody Alumno alumno, @PathVariable Integer id) {
85     Alumno alumnoModificar = aluServ.obtener(id);
86     if (alumnoModificar.getAlumnoId() == null) {
87         return new ResponseEntity<> (status: HttpStatus.NOT_FOUND);
88     } else {
89         if (!Objects.equals(alumno.getAlumnoId(), id) && alumno.getAlumnoId() != null) {
90             return new ResponseEntity<> (status: HttpStatus.BAD_REQUEST);
91         } else {
92             alumnoModificar.setNombreAlumno(nombreAlumno: alumno.getNombreAlumno());
93             alumnoModificar.setEdad(edad: alumno.getEdad());
94             alumnoModificar.setTitulo(titulo: alumno.getTitulo());
95             alumnoModificar = aluServ.modificar(alumno: alumnoModificar);
96         }
97     }
98     return new ResponseEntity<> (body: alumnoModificar, status: HttpStatus.OK);
99 }

```

```

100 //5. Insertar una práctica para un alumno
101 @PutMapping("/alumnos/practica")
102 public ResponseEntity<Practica> putPracticaAlumno(@RequestBody AlumnoPracticaUtil alumnoPracticaUtil) {
103     Practica practicaInsertar = pracServ.obtener(id: alumnoPracticaUtil.getPracticaId());
104     Alumno alumnoPractica = aluServ.obtener(id: alumnoPracticaUtil.getAlumnoId());
105     if (practicaInsertar.getPracticaId() == null || alumnoPractica.getAlumnoId() == null) {
106         return new ResponseEntity<> (status: HttpStatus.NOT_FOUND);
107     }
108     practicaInsertar.setAlumnoId(alumnoPractica);
109     practicaInsertar = pracServ.modificar(practica: practicaInsertar);
110     return new ResponseEntity<> (body: practicaInsertar, status: HttpStatus.OK);
111 }
112
113 //6. Eliminar un alumno
114 @DeleteMapping("/alumnos/{id}")
115 public ResponseEntity<Alumno> deleteAlumno(@PathVariable Integer id) {
116     Alumno alumno = aluServ.obtener(id);
117     if (alumno.getAlumnoId() == null) {
118         return new ResponseEntity<> (status: HttpStatus.NOT_FOUND);
119     }
120     aluServ.eliminar(id);
121     return new ResponseEntity<> (status: HttpStatus.NO_CONTENT);
122 }
123
124 //7. Devolver todos los alumnos con su práctica de un determinado título
125 @GetMapping("/alumnos/titulo/{titulo}")
126 public ResponseEntity<ArrayList<AlumnoConPracticaDTO>> getAlumnoTitulo(@PathVariable String titulo) {
127     ArrayList<Alumno> alumnos = aluServ.mostrarAlumnosTitulo(titulo);
128     ArrayList<AlumnoConPracticaDTO> alumnosDevolver = new ArrayList<>();
129     if (alumnos.isEmpty()) {
130         return new ResponseEntity<> (status: HttpStatus.NOT_FOUND);
131     }
132     for (Alumno alumno : alumnos) {
133         AlumnoConPracticaDTO nuevoAlumno = new AlumnoConPracticaDTO(alumno);
134         alumnosDevolver.add(nuevoAlumno);
135     }
136     return new ResponseEntity<> (body: alumnosDevolver, status: HttpStatus.OK);
137 }

```

```

179 //8.Devolver todos los alumnos que tienen menos de cierta edad
180 @GetMapping("/alumnos/edad/{edad}")
181 public ResponseEntity<ArrayList<Alumno>> getAlumnoEdadMenorQue(@PathVariable Integer edad) {
182
183     ArrayList<Alumno> alumnos = aluServ.mostrarAlumnosEdadMenorQue(edad);
184
185     if (alumnos.isEmpty()) {
186
187         return new ResponseEntity<> (status:HttpStatus.NOT_FOUND);
188     }
189
190     return new ResponseEntity<> (body: alumnos, status:HttpStatus.OK);
191 }
192
193 //9.Obtener todos los alumnos que no tienen práctica
194 @GetMapping("/alumnos/sinpractica")
195 public ResponseEntity<ArrayList<Alumno>> getAlumnosSinPractica() {
196
197     ArrayList<Alumno> alumnos = aluServ.mostrarAlumnosSinPractica();
198
199     if (alumnos.isEmpty()) {
200
201         return new ResponseEntity<> (status:HttpStatus.NOT_FOUND);
202     }
203
204     return new ResponseEntity<> (body: alumnos, status:HttpStatus.OK);
205 }
206
207 //10.Obtener todos los alumnos que tienen práctica asignada
208 @GetMapping("/alumnos/conpractica")
209 public ResponseEntity<ArrayList<Alumno>> getAlumnosConPracticaAsignada() {
210
211     ArrayList<Alumno> alumnos = aluServ.mostrarAlumnosConPractica();
212
213     if (alumnos.isEmpty()) {
214
215         return new ResponseEntity<> (status:HttpStatus.NOT_FOUND);
216     }
217
218     return new ResponseEntity<> (body: alumnos, status:HttpStatus.OK);
219 }
220
221 }
222
223
224
225
226
227
228

```

## Model

```

1 package com.franciscojosegarciacutillas.appinstituto.model;
2
3 import com.fasterxml.jackson.annotation.JsonIgnore;
4 import java.io.Serializable;
5 import java.util.List;
6 import jakarta.persistence.*;
7
8 /**
9  *
10  * @author Fran
11  */
12 @Entity
13 @Table(name = "alumnos")
14 @NamedQueries({
15     @NamedQuery(name = "Alumno.findAll", query = "SELECT a FROM Alumno a"),
16     @NamedQuery(name = "Alumno.findById", query = "SELECT a FROM Alumno a WHERE a.alumnoId = :alumnoId"),
17     @NamedQuery(name = "Alumno.findByNameAlumno", query = "SELECT a FROM Alumno a WHERE a.nombreAlumno = :nombreAlumno"),
18     @NamedQuery(name = "Alumno.findByEdad", query = "SELECT a FROM Alumno a WHERE a.edad = :edad"),
19     @NamedQuery(name = "Alumno.findByTitulo", query = "SELECT a FROM Alumno a WHERE a.titulo = :titulo")})
20 public class Alumno implements Serializable {
21
22     private static final long serialVersionUID = 1L;
23     @Id
24     @GeneratedValue(strategy = GenerationType.IDENTITY)
25     @Basic(optional = false)
26     @Column(name = "alumno_id")
27     private Integer alumnoId;
28     @Basic(optional = false)
29     @Column(name = "nombre_alumno")
30     private String nombreAlumno;
31     @Basic(optional = false)
32     @Column(name = "edad")
33     private int edad;
34     @Basic(optional = false)
35     @Column(name = "titulo")
36     private String titulo;
37     @JoinTable(name = "alumnosconpracticas", joinColumns = {
38         @JoinColumn(name = "alumno_id", referencedColumnName = "alumno_id"), inverseJoinColumns = {
39             @JoinColumn(name = "practica_id", referencedColumnName = "practica_id")})
40     @ManyToMany
41     @JsonIgnore
42     private List<Practica> practicaList;
43     @OneToOne(mappedBy = "alumnoId")
44     @JsonIgnore
45     private Practica practica;
46
47     public Alumno() {
48     }
49
50     public Alumno(Integer alumnoId) {
51         this.alumnoId = alumnoId;
52     }
53
54     public Alumno(Integer alumnoId, String nombreAlumno, int edad, String titulo) {
55         this.alumnoId = alumnoId;
56         this.nombreAlumno = nombreAlumno;
57         this.edad = edad;
58         this.titulo = titulo;
59     }
60

```

```

61 public Alumno(String nombreAlumno, int edad, String titulo) {
62     this.nombreAlumno = nombreAlumno;
63     this.edad = edad;
64     this.titulo = titulo;
65 }
66
67 public Integer getAlumnoId() {
68     return alumnoId;
69 }
70
71 public void setAlumnoId(Integer alumnoId) {
72     this.alumnoId = alumnoId;
73 }
74
75 public String getNombreAlumno() {
76     return nombreAlumno;
77 }
78
79 public void setNombreAlumno(String nombreAlumno) {
80     this.nombreAlumno = nombreAlumno;
81 }
82
83 public int getEdad() {
84     return edad;
85 }
86
87 public void setEdad(int edad) {
88     this.edad = edad;
89 }
90
91 public String getTitulo() {
92     return titulo;
93 }
94
95 public void setTitulo(String titulo) {
96     this.titulo = titulo;
97 }
98
99 public List<Practica> getPracticaList() {
100     return practicaList;
101 }
102
103 public void setPracticaList(List<Practica> practicaList) {
104     this.practicaList = practicaList;
105 }
106
107 public Practica getPractica() {
108     return practica;
109 }
110
111 public void setPractica(Practica practica) {
112     this.practica = practica;
113 }
114
115 @Override
116 public int hashCode() {
117     int hash = 0;
118     hash += (alumnoId != null ? alumnoId.hashCode() : 0);
119     return hash;
120 }
121

```

```

122 @Override
123 public boolean equals(Object object) {
124     // TODO: Warning - this method won't work in the case the id fields are not set
125     if (!(object instanceof Alumno)) {
126         return false;
127     }
128     Alumno other = (Alumno) object;
129     if ((this.alumnoId == null && other.alumnoId != null) || (this.alumnoId != null && !this.alumnoId.equals(other.alumnoId))) {
130         return false;
131     }
132     return true;
133 }
134
135 @Override
136 public String toString() {
137     return "Alumno con ID = " + alumnoId + ", Nombre = " + nombreAlumno + ", Edad = " + edad + ", Titulo = " + titulo;
138 }
139
140 }
141

```

```

1 package com.franciscojosegarciaacutillas.appinstituto.model;
2
3 import com.fasterxml.jackson.annotation.JsonIgnore;
4 import java.io.Serializable;
5 import java.util.List;
6 import jakarta.persistence.*;
7
8 /**
9  *
10  * @author Fran
11  */
12 @Entity
13 @Table(name = "empresas")
14 @NamedQueries({
15     @NamedQuery(name = "Empresa.findAll", query = "SELECT e FROM Empresa e"),
16     @NamedQuery(name = "Empresa.findById", query = "SELECT e FROM Empresa e WHERE e.empresaId = :empresaId"),
17     @NamedQuery(name = "Empresa.findByNombreEmpresa", query = "SELECT e FROM Empresa e WHERE e.nombreEmpresa = :nombreEmpresa"),
18     @NamedQuery(name = "Empresa.findBySector", query = "SELECT e FROM Empresa e WHERE e.sector = :sector")}
19 public class Empresa implements Serializable {
20
21     private static final long serialVersionUID = 1L;
22     @Id
23     @GeneratedValue(strategy = GenerationType.IDENTITY)
24     @Basic(optional = false)
25     @Column(name = "empresa_id")
26     private Integer empresaId;
27     @Basic(optional = false)
28     @Column(name = "nombre_empresa")
29     private String nombreEmpresa;
30     @Basic(optional = false)
31     @Column(name = "sector")
32     private String sector;
33     @OneToMany(cascade = CascadeType.ALL, mappedBy = "empresa")
34     @JsonIgnore
35     private List<Practica> practicaList;
36
37     public Empresa() {
38     }
39
40     public Empresa(Integer empresaId) {
41         this.empresaId = empresaId;
42     }
43
44     public Empresa(Integer empresaId, String nombreEmpresa, String sector) {
45         this.empresaId = empresaId;
46         this.nombreEmpresa = nombreEmpresa;
47         this.sector = sector;
48     }
49
50     public Integer getEmpresaId() {
51         return empresaId;
52     }
53
54     public void setEmpresaId(Integer empresaId) {
55         this.empresaId = empresaId;
56     }
57
58     public String getNombreEmpresa() {
59         return nombreEmpresa;
60     }
61
62     public void setNombreEmpresa(String nombreEmpresa) {
63         this.nombreEmpresa = nombreEmpresa;
64     }
65
66     public String getSector() {
67         return sector;
68     }
69
70     public void setSector(String sector) {
71         this.sector = sector;
72     }
73
74     public List<Practica> getPracticaList() {
75         return practicaList;
76     }
77
78     public void setPracticaList(List<Practica> practicaList) {
79         this.practicaList = practicaList;
80     }
81
82     @Override
83     public int hashCode() {
84         int hash = 0;
85         hash += (empresaId != null ? empresaId.hashCode() : 0);
86         return hash;
87     }
88
89     @Override
90     public boolean equals(Object object) {
91         // TODO: Warning - this method won't work in the case the id fields are not set
92         if (!(object instanceof Empresa)) {
93             return false;
94         }
95         Empresa other = (Empresa) object;
96         return ((this.empresaId == null && other.empresaId != null) || (this.empresaId != null && !this.empresaId.equals(other.empresaId))) {
97             return false;
98         }
99         return true;
100     }
101
102     @Override
103     public String toString() {
104         return "Empresa con ID = " + empresaId + ", Nombre = " + nombreEmpresa + ", Sector = " + sector;
105     }
106
107 }
108

```



```

1 package com.franciscojosegarciacutillas.appinstituto.Model;
2
3 import com.fasterxml.jackson.annotation.JsonIgnore;
4 import java.io.Serializable;
5 import java.util.Date;
6 import java.util.List;
7 import jakarta.persistence.*;
8
9 /**
10  *
11  * @author Fran
12  */
13 @Entity
14 @Table(name = "practicas")
15 @NamedQueries({
16     @NamedQuery(name = "Practica.findAll", query = "SELECT p FROM Practica p"),
17     @NamedQuery(name = "Practica.findById", query = "SELECT p FROM Practica p WHERE p.practicaId = :practicaId"),
18     @NamedQuery(name = "Practica.findByIdFechaInicio", query = "SELECT p FROM Practica p WHERE p.fechaInicio = :fechaInicio"),
19     @NamedQuery(name = "Practica.findByIdFechaFin", query = "SELECT p FROM Practica p WHERE p.fechaFin = :fechaFin"),
20     @NamedQuery(name = "Practica.findByIdDescripcion", query = "SELECT p FROM Practica p WHERE p.descripcion = :descripcion"))
21 public class Practica implements Serializable {
22
23     private static final long serialVersionUID = 1L;
24     @Id
25     @GeneratedValue(strategy = GenerationType.IDENTITY)
26     @Basic(optional = false)
27     @Column(name = "practica_id")
28     private Integer practicaId;
29     @Basic(optional = false)
30     @Column(name = "fecha_inicio")
31     @Temporal(TemporalType.DATE)
32     private Date fechaInicio;
33     @Basic(optional = false)
34     @Column(name = "fecha_fin")
35     @Temporal(TemporalType.DATE)
36     private Date fechaFin;
37     @Basic(optional = false)
38     @Column(name = "descripcion")
39     private String descripcion;
40     @ManyToMany(mappedBy = "practicaList")
41     @JsonIgnore
42     private List<Alumno> alumnoList;
43     @JoinColumn(name = "alumno_id", referencedColumnName = "alumno_id")
44     @ManyToOne
45     @JsonIgnore
46     private Alumno alumnoId;
47     @JoinColumn(name = "empresa_id", referencedColumnName = "empresa_id")
48     @ManyToOne(optional = false)
49     private Empresa empresa;
50
51     public Practica() {
52     }
53
54     public Practica(Integer practicaId) {
55         this.practicaId = practicaId;
56     }
57
58     public Practica(Integer practicaId, Date fechaInicio, Date fechaFin, String descripcion) {
59         this.practicaId = practicaId;
60         this.fechaInicio = fechaInicio;
61         this.fechaFin = fechaFin;
62         this.descripcion = descripcion;
63     }
64

```

```

65     public Integer getPracticaId() {
66         return practicaId;
67     }
68
69     public void setPracticaId(Integer practicaId) {
70         this.practicaId = practicaId;
71     }
72
73     public Date getFechaInicio() {
74         return fechaInicio;
75     }
76
77     public void setFechaInicio(Date fechaInicio) {
78         this.fechaInicio = fechaInicio;
79     }
80
81     public Date getFechaFin() {
82         return fechaFin;
83     }
84
85     public void setFechaFin(Date fechaFin) {
86         this.fechaFin = fechaFin;
87     }
88
89     public String getDescripcion() {
90         return descripcion;
91     }
92
93     public void setDescripcion(String descripcion) {
94         this.descripcion = descripcion;
95     }
96
97     public List<Alumno> getAlumnoList() {
98         return alumnoList;
99     }
100
101     public void setAlumnoList(List<Alumno> alumnoList) {
102         this.alumnoList = alumnoList;
103     }
104
105     public Alumno getAlumnoId() {
106         return alumnoId;
107     }
108
109     public void setAlumnoId(Alumno alumnoId) {
110         this.alumnoId = alumnoId;
111     }
112
113     public Empresa getEmpresa() {
114         return empresa;
115     }
116
117     public void setEmpresa(Empresa empresaId) {
118         this.empresa = empresaId;
119     }
120
121     @Override
122     public int hashCode() {
123         int hash = 0;
124         hash += (practicaId != null ? practicaId.hashCode() : 0);
125         return hash;
126     }
127

```

```

128     @Override
129     public boolean equals(Object object) {
130         // TODO: Warning - this method won't work in the case the id fields are not set
131         if (!(object instanceof Practica)) {
132             return false;
133         }
134         Practica other = (Practica) object;
135         if ((this.practicaId == null && other.practicaId != null) || (this.practicaId != null && !this.practicaId.equals(other.practicaId))) {
136             return false;
137         }
138         return true;
139     }
140
141     @Override
142     public String toString() {
143         return "Practica con ID = " + practicaId + ", Fecha de inicio = " + fechaInicio + ", Fecha de fin = " + fechaFin +
144             ", Descripción = " + descripcion;
145     }
146 }
147
148

```

## Repository

```

1  package com.franciscojosegarciacutillas.appinstituto.repository;
2
3  import com.franciscojosegarciacutillas.appinstituto.model.Alumno;
4  import java.util.List;
5  import org.springframework.data.jpa.repository.JpaRepository;
6  import org.springframework.data.jpa.repository.Query;
7
8  /**
9   *
10   * @author Fran
11   */
12  public interface IAlumnoRepository extends JpaRepository<Alumno, Integer> {
13
14      List<Alumno> findByTitulo(String titulo);
15      List<Alumno> findByEdadLessThan(Integer edad);
16      @Query("FROM Alumno a WHERE a.practica is null")
17      List<Alumno> findByPracticaNull();
18      @Query("FROM Alumno a WHERE a.practica is not null")
19      List<Alumno> findByPracticaNotNull();
20
21  }

```

```

1  package com.franciscojosegarciacutillas.appinstituto.repository;
2
3  import com.franciscojosegarciacutillas.appinstituto.model.Empresa;
4  import org.springframework.data.jpa.repository.JpaRepository;
5
6  /**
7   *
8   * @author Fran
9   */
10  public interface IEmpresaRepository extends JpaRepository<Empresa, Integer> {
11
12  }
13

```

```

1  package com.franciscojosegarciacutillas.appinstituto.repository;
2
3  import com.franciscojosegarciacutillas.appinstituto.model.Practica;
4  import org.springframework.data.jpa.repository.JpaRepository;
5
6  /**
7   *
8   * @author Fran
9   */
10  public interface IPracticaRepository extends JpaRepository<Practica, Integer> {
11
12  }
13

```

## Service

```

1 package com.franciscojosegarciaacutillas.appinstituto.service;
2
3 import com.franciscojosegarciaacutillas.appinstituto.model.Alumno;
4 import java.util.ArrayList;
5
6 /**
7  *
8  * @author Fran
9  */
10 public interface IAlumnoService {
11
12     Alumno registrar(Alumno alumno);
13     Alumno modificar(Alumno alumno);
14     ArrayList<Alumno> listar();
15     Alumno obtener(Integer id);
16     void eliminar(Integer id);
17     ArrayList<Alumno> mostrarAlumnosTitulo(String titulo);
18     ArrayList<Alumno> mostrarAlumnosEdadMenorQue(Integer edad);
19     ArrayList<Alumno> mostrarAlumnosSinPractica();
20     ArrayList<Alumno> mostrarAlumnosConPractica();
21
22 }
23

```

```

1 package com.franciscojosegarciaacutillas.appinstituto.service;
2
3 import com.franciscojosegarciaacutillas.appinstituto.model.Alumno;
4 import com.franciscojosegarciaacutillas.appinstituto.repository.IAlumnoRepository;
5 import java.util.ArrayList;
6 import java.util.Optional;
7 import org.springframework.beans.factory.annotation.Autowired;
8 import org.springframework.stereotype.Service;
9
10 /**
11  *
12  * @author Fran
13  */
14 @Service
15 public class AlumnoServiceImpl implements IAlumnoService {
16
17     @Autowired
18     private IAlumnoRepository iRepository;
19
20     @Override
21     public Alumno registrar(Alumno alumno) {
22         return iRepository.save(entity: alumno);
23     }
24
25     @Override
26     public Alumno modificar(Alumno alumno) {
27         return iRepository.save(entity: alumno);
28     }
29
30     @Override
31     public ArrayList<Alumno> listar() {
32         return (ArrayList<Alumno>) iRepository.findAll();
33     }
34
35     @Override
36     public Alumno obtener(Integer id) {
37         Optional<Alumno> op = iRepository.findById(id);
38         return op.isPresent() ? op.get() : new Alumno();
39     }
40
41     @Override
42     public void eliminar(Integer id) {
43         iRepository.deleteById(id);
44     }
45
46     @Override
47     public ArrayList<Alumno> mostrarAlumnosTitulo(String titulo) {
48         return (ArrayList<Alumno>) iRepository.findByTitulo(titulo);
49     }
50
51     @Override
52     public ArrayList<Alumno> mostrarAlumnosEdadMenorQue(Integer edad) {
53         return (ArrayList<Alumno>) iRepository.findByEdadLessThan(edad);
54     }
55
56     @Override
57     public ArrayList<Alumno> mostrarAlumnosSinPractica() {
58         return (ArrayList<Alumno>) iRepository.findByPracticaNull();
59     }
60
61     @Override
62     public ArrayList<Alumno> mostrarAlumnosConPractica() {
63         return (ArrayList<Alumno>) iRepository.findByPracticaNotNull();
64     }
65
66 }
67

```

```

1 package com.franciscojosegarciacutillas.appinstituto.service;
2
3 import com.franciscojosegarciacutillas.appinstituto.model.Empresa;
4 import java.util.ArrayList;
5
6 /**
7  *
8  * @author Fran
9  */
10 public interface IEmpresaService {
11
12     Empresa registrar(Empresa empresa);
13     Empresa modificar(Empresa empresa);
14     ArrayList<Empresa> listar();
15     Empresa obtener(Integer id);
16     void eliminar(Integer id);
17
18 }

```

```

1 package com.franciscojosegarciacutillas.appinstituto.service;
2
3 import com.franciscojosegarciacutillas.appinstituto.model.Empresa;
4 import com.franciscojosegarciacutillas.appinstituto.repository.IEmpresaRepository;
5 import java.util.ArrayList;
6 import java.util.Optional;
7 import org.springframework.beans.factory.annotation.Autowired;
8 import org.springframework.stereotype.Service;
9
10 /**
11  *
12  * @author Fran
13  */
14 @Service
15 public class EmpresaServiceImpl implements IEmpresaService {
16
17     @Autowired
18     private IEmpresaRepository iRepository;
19
20     @Override
21     public Empresa registrar(Empresa empresa) {
22         return iRepository.save(entity: empresa);
23     }
24
25     @Override
26     public Empresa modificar(Empresa empresa) {
27         return iRepository.save(entity: empresa);
28     }
29
30     @Override
31     public ArrayList<Empresa> listar() {
32         return (ArrayList<Empresa>)iRepository.findAll();
33     }
34
35     @Override
36     public Empresa obtener(Integer id) {
37         Optional<Empresa> op = iRepository.findById(id);
38         return op.isPresent() ? op.get() : new Empresa();
39     }
40
41     @Override
42     public void eliminar(Integer id) {
43         iRepository.deleteById(id);
44     }
45
46 }

```

```

1 package com.franciscojosegarciacutillas.appinstituto.service;
2
3 import com.franciscojosegarciacutillas.appinstituto.model.Practica;
4 import java.util.ArrayList;
5
6 /**
7  *
8  * @author Fran
9  */
10 public interface IPracticaService {
11
12     Practica registrar(Practica practica);
13     Practica modificar(Practica practica);
14     ArrayList<Practica> listar();
15     Practica obtener(Integer id);
16     void eliminar(Integer id);
17
18 }
19

```

```

1 package com.franciscojosegarciacutillas.appinstituto.service;
2
3 import com.franciscojosegarciacutillas.appinstituto.model.Practica;
4 import com.franciscojosegarciacutillas.appinstituto.repository.IPracticaRepository;
5 import java.util.ArrayList;
6 import java.util.Optional;
7 import org.springframework.beans.factory.annotation.Autowired;
8 import org.springframework.stereotype.Service;
9
10 /**
11  *
12  * @author Fran
13  */
14 @Service
15 public class PracticaServiceImpl implements IPracticaService {
16
17     @Autowired
18     private IPracticaRepository iRepository;
19
20     @Override
21     public Practica registrar(Practica practica) {
22         return iRepository.save(entity: practica);
23     }
24
25     @Override
26     public Practica modificar(Practica practica) {
27         return iRepository.save(entity: practica);
28     }
29
30     @Override
31     public ArrayList<Practica> listar() {
32         return (ArrayList<Practica>) iRepository.findAll();
33     }
34
35     @Override
36     public Practica obtener(Integer id) {
37         Optional<Practica> op = iRepository.findById(id);
38         return op.isPresent() ? op.get() : new Practica();
39     }
40
41     @Override
42     public void eliminar(Integer id) {
43         iRepository.deleteById(id);
44     }
45
46 }
47

```

## Dto

```

1 package com.franciscojosegarciacutillas.appinstituto.dto;
2
3 import com.franciscojosegarciacutillas.appinstituto.model.Alumno;
4 import com.franciscojosegarciacutillas.appinstituto.model.Practica;
5 import lombok.AllArgsConstructor;
6 import lombok.Getter;
7 import lombok.NoArgsConstructor;
8 import lombok.Setter;
9
10 /**
11  *
12  * @author Fran
13  */
14 @Getter @Setter @AllArgsConstructor @NoArgsConstructor
15 public class AlumnoConPracticaDTO {
16
17     private String nombre;
18     private int edad;
19     private String titulo;
20     private Practica practica;
21
22     public AlumnoConPracticaDTO(Alumno alumno) {
23
24         this.nombre = alumno.getNombreAlumno();
25         this.edad = alumno.getEdad();
26         this.titulo = alumno.getTitulo();
27         this.practica = alumno.getPractica();
28     }
29
30 }

```

## Utilidades

```

1 package com.franciscojosegarciacutillas.appinstituto.utilidades;
2
3 import lombok.AllArgsConstructor;
4 import lombok.Getter;
5 import lombok.Setter;
6
7
8 /**
9  *
10  * @author Fran
11  */
12 @Getter @Setter @AllArgsConstructor
13 public class AlumnoPracticaUtil{
14
15     private Integer alumnoId;
16     private Integer practicaId;
17
18
19 }
20

```

## Controller

```

1 package com.franciscojosegarciacutillas.appinstituto.controller;
2
3 import com.franciscojosegarciacutillas.appinstituto.dto.AlumnoConPracticaDTO;
4 import com.franciscojosegarciacutillas.appinstituto.model.Alumno;
5 import com.franciscojosegarciacutillas.appinstituto.model.Practica;
6 import com.franciscojosegarciacutillas.appinstituto.service.AlumnoServiceImpl;
7 import com.franciscojosegarciacutillas.appinstituto.service.PracticaServiceImpl;
8 import com.franciscojosegarciacutillas.appinstituto.utilidades.AlumnoPracticaUtil;
9 import java.util.ArrayList;
10 import java.util.Objects;
11 import org.springframework.beans.factory.annotation.Autowired;
12 import org.springframework.http.HttpStatus;
13 import org.springframework.http.ResponseEntity;
14 import org.springframework.web.bind.annotation.DeleteMapping;
15 import org.springframework.web.bind.annotation.GetMapping;
16 import org.springframework.web.bind.annotation.PathVariable;
17 import org.springframework.web.bind.annotation.PostMapping;
18 import org.springframework.web.bind.annotation.PutMapping;
19 import org.springframework.web.bind.annotation.RequestBody;
20 import org.springframework.web.bind.annotation.RequestMapping;
21 import org.springframework.web.bind.annotation.RestController;
22
23 /**
24  *
25  * @author Fran
26  */
27 @RestController
28 @RequestMapping("/api/v1")
29 public class AlumnoController {
30
31     @Autowired
32     AlumnoServiceImpl aluServ;
33
34     @Autowired
35     PracticaServiceImpl pracServ;
36

```

Tomando como referencia dicha base de datos deberás generar un API Rest que proporcione los siguientes endpoints

1. Consultar todos los alumnos con su práctica (si la tuviera asignada)

```

//Recuperar todos los alumnos
@GetMapping("/alumnos")
public ResponseEntity<ArrayList<AlumnoConPracticaDTO>> getAlumnosConPractica() {

    ArrayList<Alumno> alumnos = aluServ.listar();
    ArrayList<AlumnoConPracticaDTO> alumnosDevolver = new ArrayList<>();

    for (Alumno alumno : alumnos) {

        AlumnoConPracticaDTO nuevoAlumno = new AlumnoConPracticaDTO(alumno);
        alumnosDevolver.add(e: nuevoAlumno);

    }

    return new ResponseEntity<> (body: alumnosDevolver, status: HttpStatus.OK);

}

```

HTTP <http://localhost:8080/api/v1/alumnos> Save Send

GET <http://localhost:8080/api/v1/alumnos> Send

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

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (5) Test Results 200 OK 13 ms 890 B Save as example

Pretty Raw Preview Visualize JSON Copy Search

```

1  [
2    {
3      "nombre": "Marina García",
4      "edad": 21,
5      "titulo": "DAM",
6      "practica": {
7        "practicaId": 1,
8        "fechaInicio": "2022-02-01",
9        "fechaFin": "2022-04-30",
10       "descripcion": "Desarrollo Web en Tech Innovators",
11       "empresa": {
12         "empresaId": 1,
13         "nombreEmpresa": "NTT DATA",
14         "sector": "Tecnología"
15       }
16     },
17   },
18   {
19     "nombre": "Alejandro Ruiz",
20     "edad": 43,
21     "titulo": "DAW",
22     "practica": {
23       "practicaId": 2,
24       "fechaInicio": "2022-03-15",
25       "fechaFin": "2022-06-15",
26       "descripcion": "Análisis de Datos en Data Wizards",
27       "empresa": {
28         "empresaId": 2,
29         "nombreEmpresa": "NEORIS",
30         "sector": "Data Science"
31       }
32     },
33   },
34   {
35     "nombre": "Laura Martínez",
36     "edad": 32,
37     "titulo": "DAW",
38     "practica": null
39   },
40   {
41     "nombre": "Javier López",
42     "edad": 24,
43     "titulo": "DAM",
44     "practica": null
45   },
46   {
47     "nombre": "Sara Rodríguez",
48     "edad": 21,
49     "titulo": "ASIR",
50     "practica": null
51   }
52 ]

```



## 2. Consultar los datos de un alumno concreto

```
//Recuperar los datos de un alumno en concreto
@GetMapping("/alumnos/{id}")
public ResponseEntity<Alumno> getAlumnoId(@PathVariable Integer id) {

    Alumno alumno = aluServ.obtener(id);

    if (alumno.getAlumnoId() == null) {

        return new ResponseEntity<> ( status:HttpStatus.NOT_FOUND);

    } else {

        return new ResponseEntity<> ( body: alumno, status:HttpStatus.OK);

    }

}
```

HTTP <http://localhost:8080/api/v1/alumnos/3> Save

GET <http://localhost:8080/api/v1/alumnos/3> Send

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

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body Cookies Headers (5) Test Results 200 OK 6 ms 236 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "alumnoId": 3,
3   "nombreAlumno": "Laura Martínez",
4   "edad": 32,
5   "titulo": "DAW"
6 }
```

HTTP <http://localhost:8080/api/v1/alumnos/8> Save

GET <http://localhost:8080/api/v1/alumnos/8> Send

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

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body Cookies Headers (4) Test Results 404 Not Found 6 ms 130 B Save as example

Pretty Raw Preview Visualize Text

```
1
```

## 3. Insertar un alumno (sin práctica)

```
//Insertar un alumno sin práctica
@PostMapping("/alumnos")
public ResponseEntity<Alumno> postAlumnoSinPractica(@RequestBody Alumno alumno) {

    if (alumno.getAlumnoId() != null) {
        Alumno alumnoExiste = aluServ.obtener(id: alumno.getAlumnoId());
        if (alumnoExiste.getAlumnoId() != null) {
            return new ResponseEntity<> (status: HttpStatus.CONFLICT);
        }
    }

    Alumno alumnoInsertar = aluServ.registrar(alumno);

    return new ResponseEntity<> (body: alumnoInsertar, status: HttpStatus.CREATED);
}
```

HTTP <http://localhost:8080/api/v1/alumnos> Save

**POST** <http://localhost:8080/api/v1/alumnos> Send

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

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** Beautify

```
1 {
2   "nombreAlumno": "Juan Palomo",
3   "edad": 28,
4   "titulo": "DAM"
5 }
```

**Body** Cookies Headers (5) Test Results 201 Created 8 ms 237 B Save as example

**Pretty** Raw Preview Visualize **JSON**

```
1 {
2   "alumnoId": 8,
3   "nombreAlumno": "Juan Palomo",
4   "edad": 28,
5   "titulo": "DAM"
6 }
```

HTTP <http://localhost:8080/api/v1/alumnos> Save

**POST** <http://localhost:8080/api/v1/alumnos> Send

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

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON** Beautify

```
1 {
2   "alumnoId": 7,
3   "nombreAlumno": "Juan Palomo",
4   "edad": 28,
5   "titulo": "DAM"
6 }
```

**Body** Cookies Headers (4) Test Results 409 Conflict 35 ms 129 B Save as example

**Pretty** Raw Preview Visualize **Text**

```
1
```

## 4. Actualizar datos de un alumno

```
//Actualizar datos de un alumno
@PutMapping("/alumnos/{id}")
public ResponseEntity<Alumno> putAlumno(@RequestBody Alumno alumno, @PathVariable Integer id) {

    Alumno alumnoModificar = aluServ.obtener(id);

    if (alumnoModificar.getAlumnoId() == null) {

        return new ResponseEntity<> ( status:HttpStatus.NOT_FOUND);

    } else {

        if (!Objects.equals(a: alumno.getAlumnoId(), b: id) && alumno.getAlumnoId() != null) {

            return new ResponseEntity<> ( status:HttpStatus.BAD_REQUEST);

        } else {

            alumnoModificar.setNombreAlumno( nombreAlumno: alumno.getNombreAlumno());
            alumnoModificar.setEdad( edad: alumno.getEdad());
            alumnoModificar.setTitulo( titulo: alumno.getTitulo());

            alumnoModificar = aluServ.modificar( alumno: alumnoModificar);

        }

    }

    return new ResponseEntity<> ( body: alumnoModificar, status:HttpStatus.OK);

}
```

Le cambiamos de ASIR a DAM.

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:8080/api/v1/alumnos/7`
- Method:** PUT
- Body (JSON):**

```
{
  "nombreAlumno": "Juan Palomo",
  "edad": 28,
  "titulo": "DAM"
}
```
- Response (JSON):**

```
{
  "alumnoId": 7,
  "nombreAlumno": "Juan Palomo",
  "edad": 28,
  "titulo": "DAM"
}
```
- Status:** 200 OK, 51 ms, 232 B

## 5. Insertar una práctica para un alumno

```
//Insertar una práctica para un alumno
@PutMapping("/alumnos/practica")
public ResponseEntity<Practica> putPracticaAlumno(@RequestBody AlumnoPracticaUtil alumnoPracticaUtil){

    Practica practicaInsertar = pracServ.obtener(id:alumnoPracticaUtil.getPracticaId());
    Alumno alumnoPractica = aluServ.obtener(id:alumnoPracticaUtil.getAlumnoId());

    if (practicaInsertar.getPracticaId() == null || alumnoPractica.getAlumnoId() == null) {

        return new ResponseEntity<>(status:HttpStatus.NOT_FOUND);

    }

    practicaInsertar.setAlumnoId(alumnoId:alumnoPractica);
    practicaInsertar = pracServ.modificar(practica:practicaInsertar);

    return new ResponseEntity<>(body:practicaInsertar, status:HttpStatus.OK);

}
```

HTTP <http://localhost:8080/api/v1/alumnos/practica> Save Send

PUT <http://localhost:8080/api/v1/alumnos/practica>

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

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

```
1 {
2   "alumnoId": 4,
3   "practicaId": 5
4 }
```

Body Cookies Headers (5) Test Results 200 OK 19 ms 357 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "practicaId": 5,
3   "fechaInicio": "2022-06-15",
4   "fechaFin": "2022-09-15",
5   "descripcion": "Cloud Computing en Cloud Systems",
6   "empresa": {
7     "empresaId": 5,
8     "nombreEmpresa": "AYESA",
9     "sector": "Cloud Computing"
10  }
11 }
```

```
{
  "nombre": "Javier López",
  "edad": 24,
  "titulo": "DAM",
  "practica": {
    "practicaId": 5,
    "fechaInicio": "2022-06-15",
    "fechaFin": "2022-09-15",
    "descripcion": "Cloud Computing en Cloud Systems",
    "empresa": {
      "empresaId": 5,
      "nombreEmpresa": "AYESA",
      "sector": "Cloud Computing"
    }
  }
}
```

HTTP <http://localhost:8080/api/v1/alumnos/practica> Save

PUT <http://localhost:8080/api/v1/alumnos/practica> Send

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

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

```

1 {
2   "alumnoId": 9,
3   "practicaId": 5
4 }

```

Body Cookies Headers (4) Test Results 404 Not Found 8 ms 130 B Save as example

## 6. Eliminar un alumno.

```

//Eliminar un alumno
@DeleteMapping("/alumnos/{id}")
public ResponseEntity<Alumno> deleteAlumno(@PathVariable Integer id) {

    Alumno alumno = aluServ.obtener(id);
    if (alumno.getAlumnoId() == null) {

        return new ResponseEntity<> (status: HttpStatus.NOT_FOUND);

    }

    aluServ.eliminar(id);
    return new ResponseEntity<> (status: HttpStatus.NO_CONTENT);

}

```

HTTP <http://localhost:8080/api/v1/alumnos/7> Save

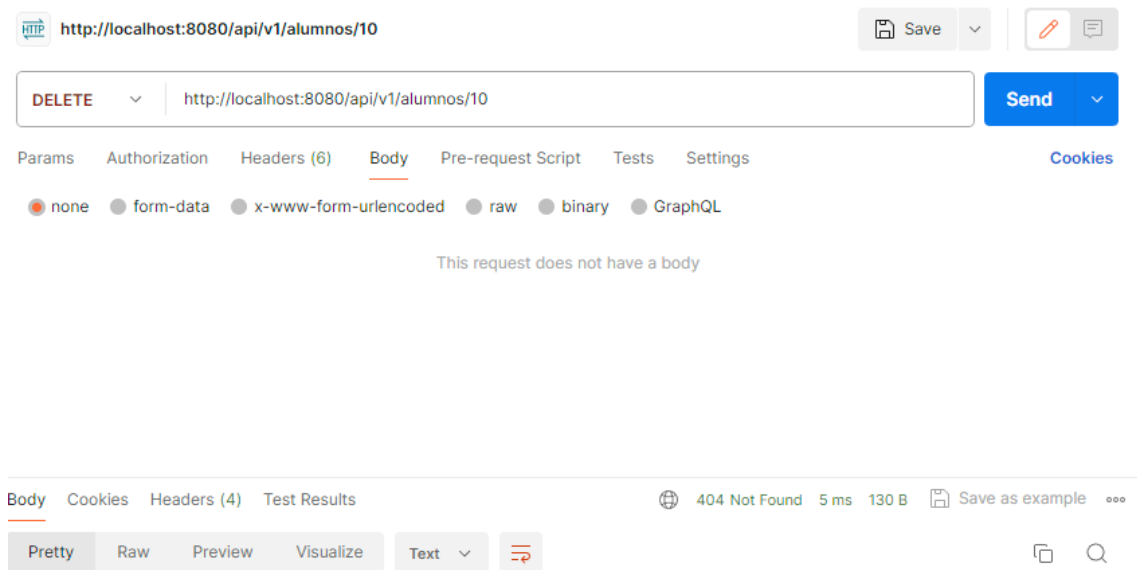
DELETE <http://localhost:8080/api/v1/alumnos/7> Send

Params Authorization Headers (6) **Body** Pre-request Script Tests Settings Cookies

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

This request does not have a body

Body Cookies Headers (3) Test Results 204 No Content 28 ms 112 B Save as example



### Haciendo uso de las consultas automatizadas de Spring Data

7. Devolver todos los alumnos con su práctica (si la tuviera asignada) de un determinado título

```
//Devolver todos los alumnos con su práctica de un determinado título
@GetMapping("/alumnos/titulo/{titulo}")
public ResponseEntity<ArrayList<AlumnoConPracticaDTO>> findByTitulo(@PathVariable String titulo){

    ArrayList<Alumno> alumnos = aluServ.mostrarAlumnosTitulo(titulo);
    ArrayList<AlumnoConPracticaDTO> alumnosDevolver = new ArrayList<>();

    if (alumnos.isEmpty()) {

        return new ResponseEntity<> ( status:HttpStatus.NOT_FOUND);

    }

    for (Alumno alumno : alumnos) {

        AlumnoConPracticaDTO nuevoAlumno = new AlumnoConPracticaDTO(alumno);
        alumnosDevolver.add( e: nuevoAlumno);

    }

    return new ResponseEntity<> ( body: alumnosDevolver, status:HttpStatus.OK);

}
```

HTTP

http://localhost:8080/api/v1/alumnos/titulo/daw

Save

GET

http://localhost:8080/api/v1/alumnos/titulo/daw

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body

Cookies

Headers (5)

Test Results

200 OK 13 ms 493 B

Save as example

Pretty

Raw

Preview

Visualize

JSON

```
1 [
2   {
3     "nombre": "Alejandro Ruiz",
4     "edad": 43,
5     "titulo": "DAW",
6     "practica": {
7       "practicaId": 2,
8       "fechaInicio": "2022-03-15",
9       "fechaFin": "2022-06-15",
10      "descripcion": "Análisis de Datos en Data Wizards",
11      "empresa": {
12        "empresaId": 2,
13        "nombreEmpresa": "NEORIS",
14        "sector": "Data Science"
15      }
16    }
17  },
18  {
19    "nombre": "Laura Martinez",
20    "edad": 32,
21    "titulo": "DAW",
22    "practica": null
23  }
24 ]
```

HTTP

http://localhost:8080/api/v1/alumnos/titulo/estetica

Save

GET

http://localhost:8080/api/v1/alumnos/titulo/estetica

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body

Cookies

Headers (4)

Test Results

404 Not Found 5 ms 130 B

Save as example

## 8. Devolver todos los alumnos que tienen menos de cierta edad

```
//8.Devolver todos los alumnos que tienen menos de cierta edad
@GetMapping("/alumnos/edad/{edad}")
public ResponseEntity<ArrayList<Alumno>> getAlumnoEdadMenorQue(@PathVariable Integer edad) {

    ArrayList<Alumno> alumnos = aluServ.mostrarAlumnosEdadMenorQue(edad);

    if (alumnos.isEmpty()) {

        return new ResponseEntity<> ( status:HttpStatus.NOT_FOUND);

    }

    return new ResponseEntity<> ( body: alumnos, status:HttpStatus.OK);

}
```

HTTP <http://localhost:8080/api/v1/alumnos/edad/22> Save

GET ▼ <http://localhost:8080/api/v1/alumnos/edad/22> Send ▼

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

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body Cookies Headers (5) Test Results 200 OK 8 ms 311 B Save as example ...

Pretty Raw Preview Visualize JSON ▼

```

1  [
2    {
3      "alumnoId": 1,
4      "nombreAlumno": "Marina García",
5      "edad": 21,
6      "titulo": "DAM"
7    },
8    {
9      "alumnoId": 5,
10     "nombreAlumno": "Sara Rodríguez",
11     "edad": 21,
12     "titulo": "ASIR"
13   }
14 ]
```

HTTP <http://localhost:8080/api/v1/alumnos/edad/20> Save

GET ▼ <http://localhost:8080/api/v1/alumnos/edad/20> Send ▼

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

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body Cookies Headers (4) Test Results 404 Not Found 5 ms 130 B Save as example ...



## Haciendo uso de consultas JPQL

### 9. Obtener todos los alumnos que no tienen práctica

```
//9.Obtener todos los alumnos que no tienen práctica
@GetMapping("/alumnos/sinpractica")
public ResponseEntity<ArrayList<Alumno>> getAlumnosSinPractica() {

    ArrayList<Alumno> alumnos = aluServ.mostrarAlumnosSinPractica();

    if (alumnos.isEmpty()) {

        return new ResponseEntity<> ( status:HttpStatus.NOT_FOUND);

    }

    return new ResponseEntity<> ( body:alumnos, status:HttpStatus.OK);

}
```

http://localhost:8080/api/v1/alumnos/sinpractica

GET

http://localhost:8080/api/v1/alumnos/sinpractica

Send

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body

Cookies

Headers (5)

Test Results

200 OK

10 ms

383 B

Save as example

Pretty

Raw

Preview

Visualize

JSON

```
1 [
2   {
3     "alumnoId": 3,
4     "nombreAlumno": "Laura Martinez",
5     "edad": 32,
6     "titulo": "DAW"
7   },
8   {
9     "alumnoId": 4,
10    "nombreAlumno": "Javier López",
11    "edad": 24,
12    "titulo": "DAM"
13  },
14  {
15    "alumnoId": 5,
16    "nombreAlumno": "Sara Rodríguez",
17    "edad": 21,
18    "titulo": "ASIR"
19  }
20 ]
```

## 10. Obtener todos los alumnos que tienen práctica asignada.

```
//10.Obtener todos los alumnos que tienen práctica asignada
@GetMapping("/alumnos/conpractica")
public ResponseEntity<ArrayList<Alumno>> getAlumnosConPracticaAsignada(){

    ArrayList<Alumno> alumnos = aluServ.mostrarAlumnosConPractica();

    if (alumnos.isEmpty()) {

        return new ResponseEntity<> ( status:HttpStatus.NOT_FOUND);

    }

    return new ResponseEntity<>( body: alumnos, status:HttpStatus.OK);

}
```

HTTP <http://localhost:8080/api/v1/alumnos/conpractica> Save

GET  <http://localhost:8080/api/v1/alumnos/conpractica> Send

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

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body Cookies Headers (5) Test Results 200 OK 117 ms 309 B Save as example

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "alumnoId": 1,
4     "nombreAlumno": "Marina García",
5     "edad": 21,
6     "titulo": "DAM"
7   },
8   {
9     "alumnoId": 2,
10    "nombreAlumno": "Alejandro Ruiz",
11    "edad": 43,
12    "titulo": "DAW"
13  }
14 ]
```