

PROJECTE

MÒDUL M12 / M13.	GRUP: DAW/DAM	DATA: 24/10/2023
Desenvolupament de l'activitat: En grups de 3		

Objectius:

1. Desenvolupar un projecte de software complert.
2. Entendre les fases que formen part d'un projecte.
3. Conèixer les diferents parts que té un projecte: Interfície, FrontEnd i BackEnd.
4. Desenvolupar amb els diferents llenguatges, llibreries i frameworks, segons la part del projecte.
5. Utilitzar eines col·laboratives de sistemes de control de versions pel codi.
6. Treballar amb les metodologies àgils a través de *Sprints*

Condicions d'entrega:

1. Les entregues sempre hauran de tenir:
 - a. Codi (sempre que *l'sprint* estigui relacionat amb el desenvolupament de codi)
 - b. Documentació:
 - i. PRODUCTE INCREMENTAL. Documentar totes aquelles parts relacionades amb el desenvolupament del software que s'ha realitzat durant *l'sprint*.
 - ii. KANBAN.
 1. Ha de tenir la imatge inicial i final de sprint de la pissarra del Kanban.
 2. S'haurà de mostrar l'estat de cada tasca realitzada en *l'sprint* (*to do, in progress, done*), qui la té assignada i les hores que s'han realitzat en cada tasca.
 3. Totes les tasques no finalitzades hauran d'anar acompanyades d'un comentari que expliqui el motiu pel qual no s'ha realitzat.
 2. Les entregues de cada sprint es podrà fer com a molt tard el diumenge.
 3. Totes les entregues es realitzaran a través de GitHub.

DESCRIPCIÓ

El centre IticBCN s'ha posat en contacte amb nosaltres degut a la necessitat que tenen actualment per a la gestió de la borsa de treball dels seus alumnes.

Per tal de poder agilitzar els processos que el centre ha de realitzar de forma manual, ens demanen el desenvolupament d'una aplicació, en versió web i per a dispositius mòbils, que automatitzi els processos més important per tal de guanyar eficiència a l'hora de buscar les empreses pels alumnes.

El que ens demanen és desenvolupar una aplicació que permeti que els alumnes puguin afegir el seu CV, buscar ofertes de pràctiques de les empreses i es puguin apuntar en aquelles ofertes que els hi interessi.

Quan un alumne s'apunti a una oferta, el sistema enviarà automàticament un email a l'empresa amb la informació de l'alumne i el seu CV (com a document adjunt).

Per altra banda, també hi haurà els administradors de l'aplicació, que seran els encarregats de gestionar tot allò que fa referència a les empreses, és a dir, podran afegir empreses a l'aplicatiu, modificar la seva informació o esborrar-la. També, seran els encarregats d'afegir les ofertes de les empreses per a què els alumnes després les puguin consultar.

A part, també gestionaran algunes aspectes relacionats amb els alumnes com per exemple validar que el C.V. que ha adjuntat l'alumne sigui correcte. Fins que aquest document no estigui validat els alumnes no podran apuntar-se a cap oferta de treball. També, tindrà la opció de donar d'alta a la plataforma de forma massiva als alumnes nous a través d'un fitxer csv.

Per tal de poder accedir a la plataforma, tant els administradors com els alumnes hauran de fer un login.

Durant l'execució del projecte, al treballar amb metodologies àgils, estarà obert a nous requeriments.

Sprint 1 - Inici de projecte

DURADA: Dilluns 23 d'octubre - Diumenge 29 d'octubre

HORES: 23 hores.

Tasques a realitzar:

1. Identificar i documentar tots els requisits indicats pel client.
2. Determinar les funcionalitats, és a dir, tot allò que farà l'usuari a través de l'aplicació..
3. Realitzar el Diagrama de casos d'ús.
4. Proposar les pantalles de l'aplicació a través d'un wireframe de baixa fidelitat **a boli i paper.**
Ha de tenir mínim 5 pàgines.
5. Documentar segons les condicions d'entrega.

Exercici 1:

1. Desenvolupar una aplicació web i mòbil per a la gestió de la borsa de treball del centre IticBCN.
2. Permetre als alumnes afegir el seu CV, buscar ofertes de pràctiques i apuntar-se a les que els interessin.
3. Enviar un email automàtic a l'empresa amb la informació i el CV de l'alumne quan s'apunti a una oferta.
4. Permetre als administradors afegir, modificar i esborrar empreses i ofertes de pràctiques.
5. Permetre als administradors validar els CV dels alumnes i donar d'alta als alumnes nous amb un fitxer csv.
6. Implementar un sistema de login per als administradors i els alumnes.
7. Treballar amb metodologies àgils i estar obert a nous requeriments.

Exercici 2:

Depenent de l'usuari, cadascun tindrà una funcionalitat.

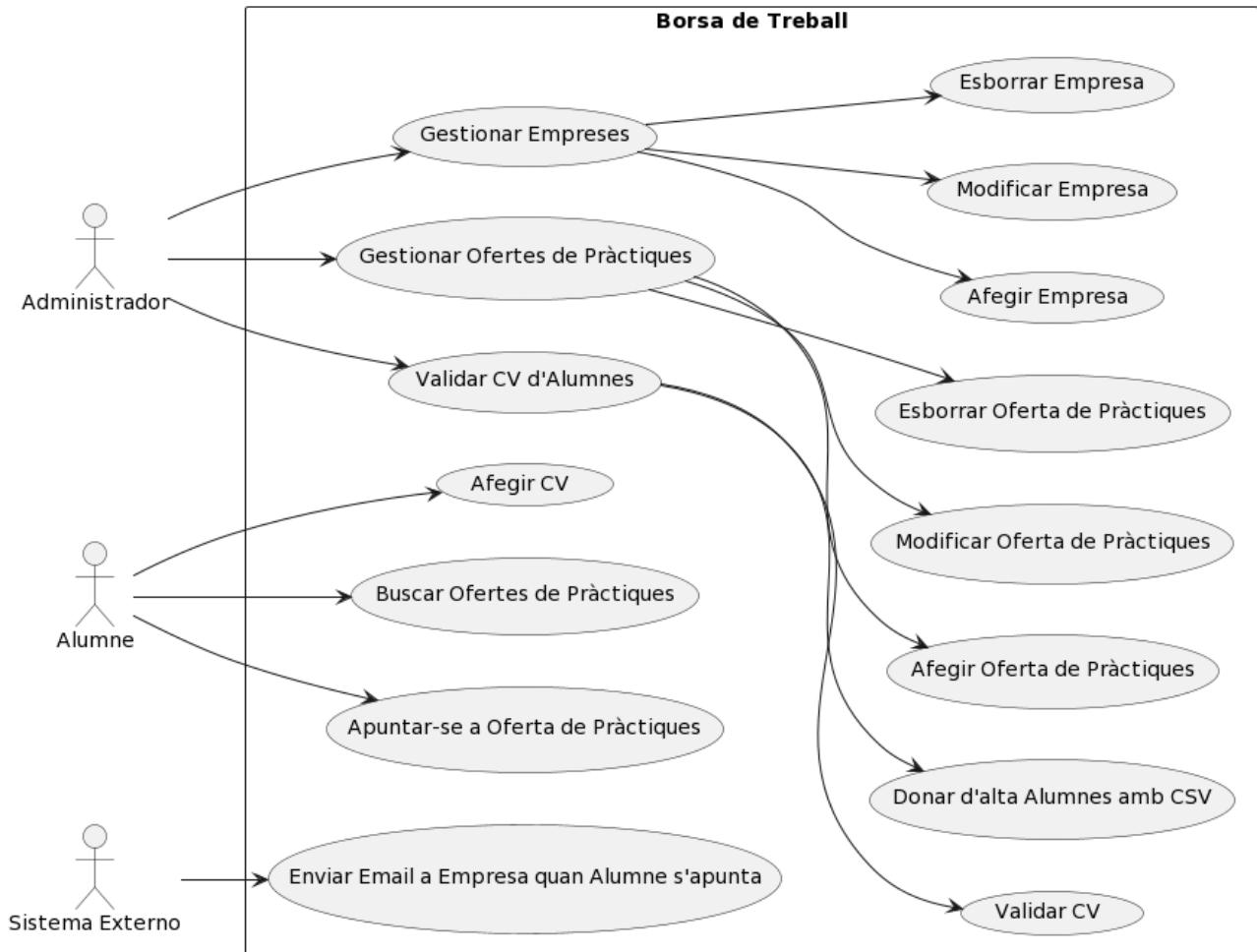
Si es estudiant:

- **Afegir CV:** Els estudiants poden carregar els seus CV a la plataforma.
- Cercar Ofertes de Pràctiques: Els estudiants poden cercar ofertes de pràctiques publicades per les empreses.
- **Sol·licitar Pràctiques:** Els estudiants poden sol·licitar les posicions de pràctiques que els interessin.
- **Notificació Automàtica per Correu Electrònic:** Quan un estudiant sol·licita una pràctica, el sistema enviarà automàticament un correu electrònic a l'empresa amb la informació de l'estudiant i el CV adjunt.

Si es administrador:

- **Gestió d'Empreses:** Els administradors poden afegir, modificar o esborrar perfils d'empresa dins de l'aplicació.
- **Gestió d'Ofertes de Pràctiques:** Els administradors poden afegir ofertes de pràctiques en nom de les empreses.
- **Validació del CV dels Estudiants:** Els administradors validen els CV dels estudiants abans que puguin sol·licitar pràctiques.
- **Inscripció Massiva:** Els administradors tenen l'opció d'inscriure nous estudiants a la plataforma utilitzant un fitxer CSV.

Exercici 3:



Exercici 4:

<p>1) Página principal / Fórmula Express</p>	<p>2) Login</p> <p>Inicia sessió</p> <p>EMAIL: _____</p> <p>CORREU: _____</p>	<p>3) Página alumno</p> <p>ITIC Ofertas Missatges <input checked="" type="checkbox"/> nom</p> <p>nom <input checked="" type="checkbox"/> Paràgraf/ descripció</p> <p>Ofertas <input checked="" type="checkbox"/></p> <p><input checked="" type="checkbox"/> Nom Empresa Nom Empresa</p>	<p>4) Página professor</p> <p>ITIC Acords Ofertas Administració <input checked="" type="checkbox"/> nom</p> <p>nom <input checked="" type="checkbox"/> Paràgraf/ descripció</p> <p>Aconsejaments Paràgraf nom nom Paràgraf nom nom nom nom nom nom</p>
<p>5) Ofertas alumno</p> <p>ITIC Ofertas Missatges <input checked="" type="checkbox"/> nom</p> <p>des més ofertas <input checked="" type="checkbox"/> nom nom nom</p> <p>Recomanacions <input checked="" type="checkbox"/> nom ampliació nom ampliació</p>	<p>6) Ofertas profesor</p> <p>ITIC Acords Ofertas Administració <input checked="" type="checkbox"/> nom</p> <p>Ofertas <input checked="" type="checkbox"/></p> <p>nom empresa <input checked="" type="checkbox"/> nom nom nom</p>	<p>7) Missatges</p> <p>ITIC Ofertas Missatges <input checked="" type="checkbox"/> nom</p> <p>nom <input checked="" type="checkbox"/> nom <input checked="" type="checkbox"/> nom <input checked="" type="checkbox"/> nom <input checked="" type="checkbox"/></p>	
<p>8) Acords</p> <p>ITIC Ofertas administració <input checked="" type="checkbox"/> nom</p> <p>Acords estaberts <input checked="" type="checkbox"/> nom nom nom nom nom nom nom nom nom</p> <p>Acords pendents <input checked="" type="checkbox"/> nom nom nom nom nom nom nom nom nom</p> <p>es un botó que obre aquest finestra</p>	<p>9) Administració</p> <p>ITIC Ofertas administració <input checked="" type="checkbox"/> nom</p> <p>llista ofertas <input checked="" type="checkbox"/> nom nom nom nom nom nom nom nom nom</p> <p>modificar elimitar <input checked="" type="checkbox"/> nom nom nom nom nom nom</p>		
<p>10) Afegit uscar</p> <p>ITIC <input checked="" type="checkbox"/> nom</p> <p>Afegit uscar <input checked="" type="checkbox"/> nom nom nom nom nom nom nom nom nom</p> <p>(Conformatri d'alta)</p>			

Exercici 5.

No hi ha codi perque es de moment tota la part abans.

Totes les tasques de kanban que ens vam repartir estan fetes, ja que hem repartit un exercici cadascun menys l'exercici del wireframe, on vam posar les idees comuns.

Sprint 2 - Primera API-rest**DURADA: Dilluns 30 d'octubre - Diumenge 5 de novembre****HORES: 20 hores**

Tasques a realitzar:

1. Entendre què és una API-rest: <https://spring.io/guides/tutorials/rest/>
2. Començar l'API-rest de Borsa de Treball amb Spring Boot (Web, JPA, H2) amb les funcionalitats (Històries d'Usuari, US) següents
 - a. US: Com a usuari puc consultar una empresa l'API.
 - b. US: Com a usuari puc afegir una empresa de l'API
 - c. US: Com a usuari puc modificar una empresa l'API.
 - d. US: Com a usuari puc eliminar una empresa l'API.
3. Comprovar que totes les funcionalitats funcionen correctament
 - a. Podeu fer servir aquest plugin de Chrome



Yet Another REST Client

yet-another-rest-client.com  Destacades

4. Documentar segons les condicions d'entrega.

El controlador es troba al github que te el seu link en la entrega.

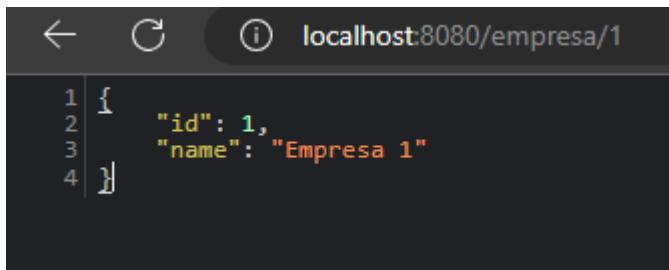
Aquestes son les funcionalitats:

- Aquí tenim la búsqueda de totes les empreses:



```
← ⌂ ⓘ localhost:8080/empresas
1 [
2   {
3     "id": 1,
4     "name": "Empresa 1"
5   },
6   {
7     "id": 2,
8     "name": "Empresa 2"
9   }
10 ]
```

- Aquí tenim la búsqueda per una unica ID



```
← ⌂ ⓘ localhost:8080/empresa/1
1 {
2   "id": 1,
3   "name": "Empresa 1"
4 }
```

- Aquí afegim la empresa amb id 6:

Request Settings

URL: POST ▾

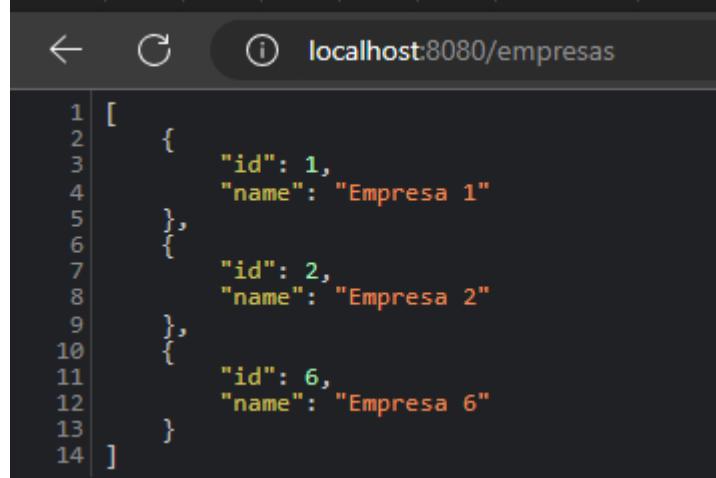
Payload:

```
{
  "id": 6,
  "name": "Empresaaaa"
}
```

Custom Headers

Authentication

Send Request



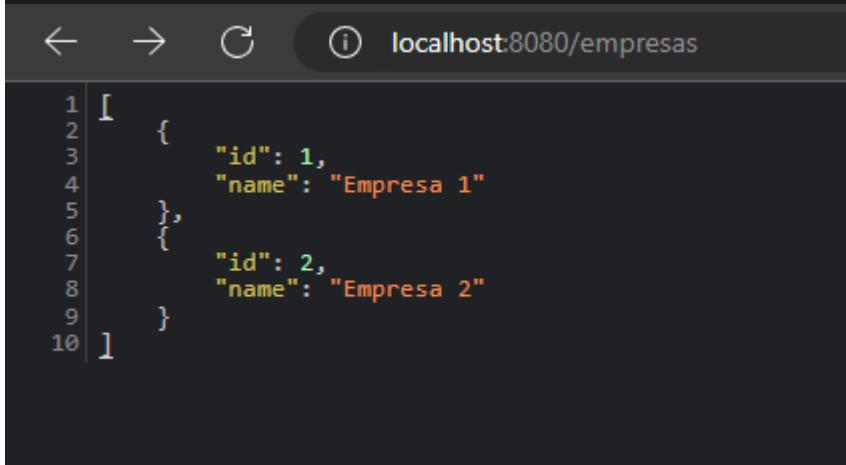
```
1 [ 
2   {
3     "id": 1,
4     "name": "Empresa 1"
5   },
6   {
7     "id": 2,
8     "name": "Empresa 2"
9   },
10  {
11    "id": 6,
12    "name": "Empresa 6"
13  }
14 ]
```

- Aquí tenim la funció de eliminar la empresa:



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

- Request Settings**: URL: <http://localhost:8080/empresaEliminar/6>, Method: DELETE, Payload: application/json.
- Custom Headers**: None listed.
- Authentication**: None listed.
- Send Request** button.



The browser window displays the JSON response from the server at localhost:8080/empresas. The response is:

```
1 [ 2 { 3   "id": 1, 4   "name": "Empresa 1" 5 }, 6 { 7   "id": 2, 8   "name": "Empresa 2" 9 } 10 ]
```

Sprint 3 - Primera API-rest

DURADA: Dilluns 6 de Novembre - Diumenge 21 de novembre

HORES: 24 hores

Tasques a realitzar:

1. Entendre com funcionen les relacions 1-N mitjançant anotacions de JPA (Hibernate)

[Activitat Relacions 1 Spring Boot: introducció guiada a les relacions](#)

[Activitat Relacions 2 Spring Boot: introducció guiada a les relacions 2](#)

[relacion-muchos-a-uno-con-manytoone](#)

<https://spring.io/projects/spring-data-jpa>

2. Començar l'API-rest de Borsa de Treball amb Spring Boot (Web, JPA, H2) amb les fu

a. US: Com a usuari puc afegir una oferta d'una empresa a l'API

b. US: Com a usuari puc consultar ofertes a l'API.

c. US: Com a usuari puc consultar ofertes d'una empresa donada a l'API.

d. US: Com a usuari puc actualitzar una oferta d'una empresa donada l'API.

e. US: Com a usuari puc eliminar una oferta d'una empresa donada l'API.

f. Escolliu una de les dues opcions:

i. US: Com a usuari, si elimino una empresa s'eliminen automàticament totes les ofertes de l'empresa.

ii. US: Com a usuari, no puc eliminar una empresa si té ofertes de feina.

3. Comprovar que totes les funcionalitats funcionen correctament

a. Heu de fer servir Postman i documentar-ho.

4. Documentar segons les condicions d'entrega.

a. Com a usuari puc afegir una oferta d'una empresa a l'API

Request Settings

URL: <http://localhost:8080/empresa/1/ofertas> POST

Payload:

```
{  
    "id": 5,  
    "titul": "Desenvolupament de Xarxes"  
}
```

Response

Request URL: <http://localhost:8080/empresa/1/ofertas>
Request Method: POST
Response Time: 0.438 seconds
Response Status: 200 - OK

200

Response Body Response Body (RAW) Response Headers Request Details

```
{  
    "id": 5,  
    "titul": "Desenvolupament de Xarxes"  
}
```

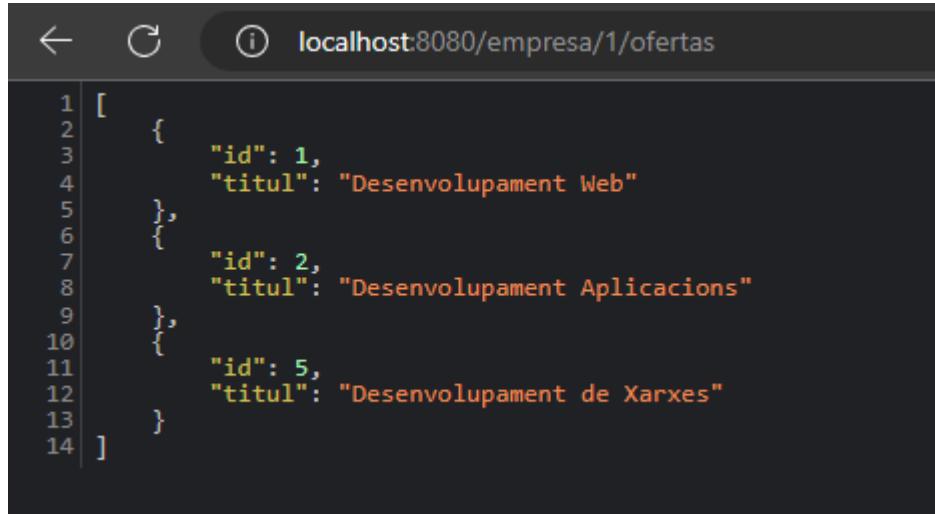
 

b. Com a usuari puc consultar ofertes a l'API.

localhost:8080/ofertas

```
1 [  
2     {  
3         "id": 1,  
4         "titul": "Desenvolupament Web"  
5     },  
6     {  
7         "id": 2,  
8         "titul": "Desenvolupament Aplicacions"  
9     },  
10    {  
11        "id": 3,  
12        "titul": "Desenvolupament Web"  
13    },  
14    {  
15        "id": 4,  
16        "titul": "Desenvolupament Aplicacions"  
17    }  
18 ]
```

c. Com a usuari puc consultar ofertes d'una empresa donada a l'API.



A screenshot of a web browser window. The address bar shows "localhost:8080/empresa/1/ofertas". The main content area displays a JSON array of three objects, each representing an offer from a company. The objects are numbered 1, 2, and 5, with titles "Desenvolupament Web", "Desenvolupament Aplicacions", and "Desenvolupament de Xarxes" respectively. The JSON code is as follows:

```
1 [  
2   {  
3     "id": 1,  
4     "titul": "Desenvolupament Web"  
5   },  
6   {  
7     "id": 2,  
8     "titul": "Desenvolupament Aplicacions"  
9   },  
10  {  
11    "id": 5,  
12    "titul": "Desenvolupament de Xarxes"  
13  }  
14 ]
```

d. Com a usuari puc actualitzar una oferta d'una empresa donada l'API.

Request Settings

URL: <http://localhost:8080/empresa/1/ofertas/5> PUT

Payload:

```
{  
    "id": 5,  
    "titul": "Administracio de sistemes i xarxes"  
}
```

Response

Request URL: <http://localhost:8080/empresa/1/ofertas/5>
Request Method: PUT
Response Time: 0.858 seconds
Response Status: 200 - OK

200

Response Body Response Body (RAW) Response Headers Request Details

```
{  
    "id": 5,  
    "titul": "Administracio de sistemes i xarxes"  
}
```

localhost:8080/empresa/1/ofertas

```
1 [  
2     {  
3         "id": 1,  
4         "titul": "Desenvolupament Web"  
5     },  
6     {  
7         "id": 2,  
8         "titul": "Desenvolupament Aplicacions"  
9     },  
10    {  
11        "id": 5,  
12        "titul": "Administracio de sistemes i xarxes"  
13    }  
14 ]
```

- e. Com a usuari puc eliminar una oferta d'una empresa donada l'API.

Request Settings

URL: <http://localhost:8080/empresa/1/ofertas/5> **DELETE**

Payload:

Response

Request URL: <http://localhost:8080/empresa/1/ofertas/5>
Request Method: DELETE
Response Time: 0.049 seconds
Response Status: 200 - OK

Response Body **Response Body (RAW)** **Response Headers** **Request Details**

(No Response Data)

localhost:8080/empresa/1/ofertas

```
1 [ 
2   { 
3     "id": 1,
4     "titul": "Desenvolupament Web"
5   },
6   { 
7     "id": 2,
8     "titul": "Desenvolupament Aplicacions"
9   }
10 ]
```

- f. Com a usuari, si elimino una empresa s'eliminen automàticament totes les ofertes de l'empresa.

```
private String titul,
@ManyToOne(fetch = FetchType.LAZY, cascade = CascadeType.REMOVE)
@JsonBackReference
private Empresa empresa;
```

Response

Request URL: <http://localhost:8080/empresaEliminar/1>
 Request Method: DELETE
 Response Time: 0.037 seconds
 Response Status: 200 - OK

200

[Response Body](#) [Response Body \(RAW\)](#) [Response Headers](#) [Request Details](#)

(No Response Data)

[Copy Request](#) [Copy Response](#)

← ⌂ ⓘ localhost:8080/empresas

```
1 [ 
2   { 
3     "id": 2,
4     "name": "Empresa 2",
5     "ofertas": [
6       {
7         "id": 3,
8         "titul": "Desenvolupament Web"
9       },
10      {
11        "id": 4,
12        "titul": "Desenvolupament Aplicacions"
13      }
14    ]
15  }
16 ]
```

← ⌂ ⓘ localhost:8080/ofertas

```
1 [ 
2   { 
3     "id": 3,
4     "titul": "Desenvolupament Web"
5   },
6   {
7     "id": 4,
8     "titul": "Desenvolupament Aplicacions"
9   }
10 ]
```

Sprint 4 - Primera API-rest

DURADA: Dilluns 20 de Novembre - Diumenge 3 de desembre

HORES: 48 hores

Tasques a realitzar:

Els inversors propietaris de l'aplicació que esteu desenvolupant s'ha mostrat preocupats per la qualitat d'alguns dels projectes que esteu elaborant i han decidit que, abans de continuar, heu d'introduir testos unitaris i/o d'integritat per tal de garantir amb més seguretat el funcionament correcte del vostre codi, i més tenint en compte que Spring Boot incorpora per defecte la dependència *l'Spring-boot-starter-test*. Ens demanen que:

1. Afegiu testos unitaris a la capa de persistència.
 - a. <https://openwebinars.net/academia/aprende/testear-persistencia-app-spring-boot>
2. Afegiu una capa de servei i afegiu test amb Junit i Mockito
 - a. <https://openwebinars.net/academia/portada/testing-unitario-spring-boot-junit-mockito>
3. Afegiu tests a les altres capes (es pot deixar per l'Sprint següent)
4. Afegiu més consultes bàsiques a banda del CRUD
 - a. <https://openwebinars.net/academia/portada/consultas-basicas-spring-data-jpa>
5. Milloreu la documentació de la vostra API amb OpenApi
 - a. <https://openwebinars.net/academia/portada/documentar-api-rest-open-api-3-0/>

D'altra banda, s'ha detectat la necessitat de millorar la comprensió d'alguns conceptes bàsics d'Spring i, per això, recomanen ampliar abans la vostra formació en aquests punts.

1. Millorar la comprensió de com Spring Boot introduceix les dependències
 - a. <https://openwebinars.net/academia/aprende/introduccion-spring-framework/>
 - b. Valoreu l'opció de canviar l'IDE i utilitzar **IntelliJ**, amb el plugin de **Spring**
2. Millorar la comprensió d'Spring Data JPA i de les opcions del fitxer de properties
 - a. <https://openwebinars.net/academia/aprende/introduccion-spring-data-jpa/>
3. Documenteu la feina feta segons les condicions d'entrega.

-Test empresa

eclipse-workspace - projecte-sprint/src/test/java/com/example/demo/TestsEmpresa.java - Eclipse IDE

```

File Edit Source Refactor Navigate Search Project Run Window Help
Debug Project Explorer JUnit
Finished after 4,925 seconds
Runs: 5/5 Errors: 0 Failures: 0
Tests Empresa [Runner: JUnit 5] (0,098 s)
  testUpdateEmpresa() (0,075 s)
  testGetAllEmpresas() (0,005 s)
  testFindById() (0,003 s)
  testDeleteEmpresa() (0,003 s)
  testSubmitForm() (0,004 s)

Failure Trace

```

```

1 package com.example.demo;
2
3 import static org.junit.jupiter.api.Assertions.assertEquals;
4
5 @DataJpaTest
6 public class TestsEmpresa {
7
8     @InjectMocks
9     private EmpresaService empresaService;
10
11     @Mock
12     private EmpresaRepository empresaRepository;
13
14     //Test per crear una empresa i buscar si existeix.
15     @Test
16     public void testSubmitForm() {
17         Empresa empresa = new Empresa(3L, "Empresa 3", new ArrayList<>());
18         Oferta oferta1 = new Oferta(1L, "Desenvolupament Web", empresa);
19         Oferta oferta2 = new Oferta(2L, "Desenvolupament Aplicacions", empresa);
20         empresa.getOfertas().add(oferta1);
21         empresa.getOfertas().add(oferta2);
22
23         when(empresaRepository.save(empresa)).thenReturn(empresa);
24         when(empresaRepository.findById(3L)).thenReturn(Optional.of(empresa)); // Add this line
25
26         String result = empresaService.submitForm(empresa);
27
28         assertEquals("Correcte", result);
29
30         verify(empresaRepository).save(empresa);
31
32         Optional<Empresa> savedEmpresa = empresaService.findById(3L);
33
34         assertEquals(true, savedEmpresa.isPresent());
35         assertEquals("Empresa 3", savedEmpresa.get().getName());
36
37         List<Oferta> savedOfertas = savedEmpresa.get().getOfertas();
38         assertEquals(2, savedOfertas.size());
39
40         Oferta savedOferta1 = savedOfertas.get(0);
41         assertEquals(1L, savedOferta1.getId());
42         assertEquals("Desenvolupament Web", savedOferta1.getTitul());
43
44     }
45
46 }

```

-Test oferta amb Mockito:

eclipse-workspace - projecte-sprint/src/test/java/com/example/demo/TestsOferta.java - Eclipse IDE

```

File Edit Source Refactor Navigate Search Project Run Window Help
Debug Project Explorer JUnit
Finished after 5,075 seconds
Runs: 5/5 Errors: 0 Failures: 0
Tests Oferta [Runner: JUnit 5] (0,130 s)
  testUpdateOferta() (0,105 s)
  testDeleteOferta() (0,005 s)
  testAddOferta() (0,004 s)
  testGetOfertasByEmpresa() (0,004 s)
  test GetAllOfertas() (0,003 s)

Failure Trace

```

```

1 package com.example.demo;
2
3
4 import static org.junit.jupiter.api.Assertions.assertEquals;
5
6 @DataJpaTest
7 public class TestsOferta {
8
9     @InjectMocks
10    private OfertaService ofertaService;
11
12    @InjectMocks
13    private EmpresaService empresaService;
14
15    @Mock
16    private OfertaRepository ofertaRepository;
17
18    @Mock
19    private EmpresaRepository empresaRepository;
20
21    //Test per veure totes les ofertes
22    @Test
23    public void testGetAllOfertas() {
24        Oferta oferta1 = new Oferta(1L, "Desenvolupament Web", null);
25        Oferta oferta2 = new Oferta(2L, "Desenvolupament Aplicacions", null);
26        List<Oferta> ofertas = Arrays.asList(oferta1, oferta2);
27
28        when(ofertaRepository.findAll()).thenReturn(ofertas);
29
30        Iterable<Oferta> result = ofertaService.getAllOfertas();
31
32        List<Oferta> resultList = StreamSupport.stream(result.spliterator(), false).collect(Collectors.toList());
33
34        assertEquals(2, resultList.size());
35        assertEquals(oferta1, resultList.get(0));
36        assertEquals(oferta2, resultList.get(1));
37    }
38
39    //Test per afegir una oferta
40    @Test
41    public void testAddOferta() {
42        Empresa empresa = new Empresa(2L, "Empresa 2", new ArrayList<>());
43        Oferta oferta = new Oferta(1L, "Desenvolupament Web", empresa);
44
45        when(empresaRepository.findById(2L)).thenReturn(empresa);
46        when(ofertaRepository.save(oferta)).thenReturn(oferta);
47
48        ofertaService.addOferta(oferta);
49
50        assertEquals(1, empresa.getOfertas().size());
51        assertEquals(oferta, empresa.getOfertas().get(0));
52    }
53
54 }

```

-Capa Service d'empresa i Oferta

```

eclipse-workspace - projecte-sprint/src/main/java/com/example/demo/repository/EmpresaService.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Debug Project Explorer JUnit
src/main/java com.example.demo.repository;
import org.springframework.stereotype.Service;
@Service
public class EmpresaService {
    private final EmpresaRepository empresaRepository;
    public EmpresaService(EmpresaRepository empresaRepository) {
        this.empresaRepository = empresaRepository;
    }
    public Optional<Empresa> findById(Long id) {
        return empresaRepository.findById(id);
    }
    public Empresa save(Empresa empresa) {
        return empresaRepository.save(empresa);
    }
    public Iterable<Empresa> getAllEmpresas() {
        return empresaRepository.findAll();
    }
    public String submitForm(Empresa empresa) {
        Empresa savedEmpresa = empresaRepository.save(empresa);
        return savedEmpresa != null ? "Correcte" : "Error";
    }
    public Empresa updateEmpresa(Empresa newEmpresa, Long id) {
        return empresaRepository.findById(id)
            .map(empre -> {
                empre.setName(newEmpresa.getName());
                return empresaRepository.save(empre);
            })
            .orElseGet(() -> {
                newEmpresa.setId(id);
                return empresaRepository.save(newEmpresa);
            });
    }
    public void deleteEmpresa(Long id) {
        empresaRepository.deleteById(id);
    }
}

eclipse-workspace - projecte-sprint/src/main/java/com/example/demo/repository/OfertaService.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Debug Project Explorer JUnit
src/main/java com.example.demo.repository;
import org.springframework.beans.factory.annotation.Autowired;
@Service
public class OfertaService {
    private final OfertaRepository ofertaRepository;
    private final EmpresaRepository empresaRepository;
    @Autowired
    public OfertaService(OfertaRepository ofertaRepository, EmpresaRepository empresaRepository) {
        this.ofertaRepository = ofertaRepository;
        this.empresaRepository = empresaRepository;
    }
    public Optional<Oferta> findById(Long id) {
        return ofertaRepository.findById(id);
    }
    public Oferta save(Oferta oferta) {
        return ofertaRepository.save(oferta);
    }
    public Iterable<Oferta> getAllOfertas() {
        return ofertaRepository.findAll();
    }
    public Oferta addOferta(Long empresaid, Oferta oferta) {
        return empresaRepository.findById(empresaid)
            .map(empre -> {
                oferta.setEmpresa(empre);
                return ofertaRepository.save(oferta);
            })
            .orElse(null);
    }
    public Oferta updateOferta(Long empresaid, Long ofertaid, Oferta ofertarequest) {
        if(!empresaRepository.existsById(empresaid)) {
            return null;
        }
        return ofertaRepository.findById(ofertaid)
            .map(empre -> {
                empre.setNombre("Nuevo nombre");
                return empre;
            })
            .orElse(null);
    }
}

```

Syntax error on token "Rpackage" package expected