



**UNIVERSIDAD
DE GRANADA**

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA Y
TELECOMUNICACIONES

Seminario 3

Diseño y Desarrollo de Sistemas de Información

Santiago González Silot
Germán Calleja Rider
Alberto Cano Jaén
Alvaro Perea Vega

(2020-2021)

Índice

1. Historias de Usuario	4
1.1. Informes de departamento	4
1.2. Informes docencia	5
2. Requisitos funcionales	6
2.1. Mostrar acta consejo departamento dada una fecha	6
2.2. Listar actas de consejo por orden temporal	6
2.3. Mostrar acta junta dirección dada una fecha	6
2.4. Listar actas de la junta de dirección por orden temporal	6
2.5. Mostrar acta de comisiones dada una fecha	7
2.6. Listar actas de comisiones por orden temporal	7
2.7. Editar categoría informes	7
2.8. Añadir categoría informes	8
2.9. Eliminar categoría informes.	8
2.10. Editar un informe	8
2.11. Eliminar un informe	9
2.12. Subir un informe	9
2.13. Listar información asociada a un profesor en concreto	9
2.14. Listar información asociada a cada uno de los profesores	10
2.15. Imprimir lista de asignaturas de todo el departamento	10
3. Práctica 1	11
4. Esquema externo	16
4.1. DFD0 Armazón 1	16
4.2. DFD0 Armazón 2	17
4.3. DFD1 Informes de departamento 1	18
4.4. DFD1 Informes de departamento 2	20
4.5. DFD1 Informes de docencia 1	22
5. Diagramas E-R	24
5.1. Informes de departamento	24
5.2. Informes de docencia	25
5.3. Paso a Tablas	26
6. Interfaz de Usuario	27
6.1. Informes departamento	27
6.2. Informes docencia	28
7. Práctica 2	29
8. Seminario 2	30
8.1. Herramientas utilizadas	30
8.2. ¿Porqué estas herramientas?	30
8.3. Instalación de NodeJS y NPM	30
8.4. Instalación de MongoDB Community Server	33
8.5. Conexión a la base de datos	36
8.5.1. Creación del proyecto	36
8.5.2. Conexión desde NodeJS a MongoDB	36
8.5.3. Comprobamos la conexión a MongoDB	37
8.5.4. Archivos JSON (plantillas)	38
8.5.5. Bibliografía	39
8.5.6. Capturas Trello	39

9. Seminario 3	40
9.1. Herramientas utilizadas	40
9.2. Lista de asignaturas y contenidos vistos	40
9.3. Conclusión	40
9.4. Autoevaluación	41
9.4.1. Herramientas utilizadas	41
9.4.2. Instalación	41
9.4.3. Conexión con la BD	41
9.5. Seminario	41
9.6. Notas	41
9.7. Estructura de la Aplicación	42
9.7.1. Capturas del Sistema	45

Índice de figuras

1.	Mapa mental	11
2.	Caja negra (1)	12
3.	Caja negra (2)	12
4.	Armazón (1)	13
5.	Armazón (2)	13
6.	Informes departamento (1)	14
7.	Informes departamento (2)	15
8.	Informes docencia	15
9.	Esquema externo Armazon 1	16
10.	Esquema externo Armazon 1	17
11.	Esquema externo DFD1 Informes de departamento 1	19
12.	Esquema externo DFD1 Informes de departamento 2	21
13.	Esquema externo DFD1 Informes de docencia 1	23
14.	Diagram E-R Informes de departamento	24
15.	Diagram E-R Informes de docencia	25
16.	Dibujo tablas E-R	26
17.	Interfaz de Usuario de Informes de Departamento	27
18.	Interfaz de Usuario de Informes de Docencia	28
19.	Interfaz de Usuario de Informes de Docencia (Pop-up)	28
20.	Captura de Trello	29
21.	Vista de Informes de Departamento	46
22.	Añadiendo un fichero a un informe	46
23.	Añadido el fichero	46
24.	Fichero PDF desde el Servidor	47

Índice de cuadros

1.	Acceder actas de departamento	4
2.	Gestionar la categoría informes	4
3.	Gestionar los informes	4
4.	Acceder a la información asociada a un profesor	5
5.	Acceder a la información asociada a todos los profesores	5

1. Historias de Usuario

1.1. Informes de departamento

Nombre: Acceder actas departamento (consejo de departamento, junta de dirección y comisiones)	ID: 1
HU1: Como profesor del departamento quiero poder acceder a todos los informes del departamento.	
Estimación: Igual al número de actas	Dependencias:
Prioridad: 300	
Pruebas de aceptación: - Introducir una fecha posterior a la actual. - Introducir una fecha fuera del periodo docente. - Que no haya ningún acta todavía e imprima un mensaje de error.	

Cuadro 1: Acceder actas de departamento

Nombre: Gestionar la categoría informes	ID: 2
HU2: Como administrador del sistema quiero poder editar, eliminar y añadir categorías dentro de informes.	
Estimación:	Dependencias:
Prioridad: 300	

Cuadro 2: Gestionar la categoría informes

Nombre: Gestionar los informes	ID: 3
HU3: Como administrador del sistema quiero poder gestionar los informes, editar, subir y borrar.	
Estimación:	Dependencias:
Prioridad: 300	

Cuadro 3: Gestionar los informes

1.2. Informes docencia

Desde el punto de vista del administrador:

Nombre: Acceder a la información asociada a un profesor ID: 4	
HU4: Como administrador del departamento quiero poder ver la ordenación docente de cada profesor (los créditos que tiene asociados, grupos y subgrupos que imparten, compensación de créditos (si corresponde))	
Estimación: Igual al número de profesores en el departamento.	Dependencias:
Prioridad:	
Pruebas de aceptación: <ul style="list-style-type: none">- Introducir un profesor que no pertenezca al departamento.- Que el documento descargado no sea en formato csv.	

Cuadro 4: Acceder a la información asociada a un profesor

Nombre: Acceder a la información asociada a todos los profesores ID: 5	
HU5: Como administrador quiero poder acceder a la ordenación docente global del departamento.	
Estimación: Igual al número de profesores en el departamento.	Dependencias:
Prioridad:	
Pruebas de aceptación: <ul style="list-style-type: none">- Que no aparezcan todos los profesores pertenecientes al departamento.- Que el documento descargado no sea en formato csv.	

Cuadro 5: Acceder a la información asociada a todos los profesores

2. Requisitos funcionales

2.1. Mostrar acta consejo departamento dada una fecha

RF1: Mostrar acta consejo departamento dada una fecha.

Entrada: Agente externo: Profesor. Acción: Solicitar acta en concreto. Requisito de datos de entrada RDE1.

BD: Requisito de datos de lectura RDR1.

RDE1: Datos de entrada del acta.

Fecha:Cadena de caracteres(10)(dd/mm/aaaa).

RDR1: Fichero almacenado del acta.

Fichero en formato '.pdf'.

2.2. Listar actas de consejo por orden temporal

RF2: Listar actas de consejo por orden temporal.

Entrada: Agente externo: Profesor. Acción: Solicitar listado de acta.

BD: Requisito de datos de lectura RDR2.

RDR2: Ficheros almacenados del acta.

Conjunto de ficheros '.pdf' en formato '.zip'.

2.3. Mostrar acta junta dirección dada una fecha

RF3: Mostrar acta junta dirección dada una fecha.

Entrada: Agente externo: Profesor. Acción: Solicitar acta en concreto. Requisito de datos de entrada RDE3.

BD: Requisito de datos de lectura RDR1.

RDE3: Datos de entrada del acta.

Fecha:Cadena de caracteres(10)(dd/mm/aaaa).

RDR3: Fichero almacenado del acta.

Fichero en formato '.pdf'.

2.4. Listar actas de la junta de dirección por orden temporal

RF4: Listar actas de la junta de dirección por orden temporal.

Entrada: Agente externo: Profesor. Acción: Solicitar listado de actas.

BD: Requisito de datos de lectura RDR4.

RDR4: Fichero almacenado del acta.

Conjunto de ficheros '.pdf' en formato '.zip'.

2.5. Mostrar acta de comisiones dada una fecha

RF5: Mostrar acta de comisiones dada una fecha.

Entrada: Agente externo: Profesor. Acción: Solicitar acta en concreto. Requisito de datos de entrada RDE5.

BD: Requisito de datos de lectura RDR5.

RDE1: Datos de entrada del acta.

Fecha:Cadena de caracteres(10)(dd/mm/aaaa).

RDR1: Fichero almacenado del acta.

Fichero en formato '.pdf'.

2.6. Listar actas de comisiones por orden temporal

RF6: Listar actas de comisiones por orden temporal.

Entrada: Agente externo: Profesor. Acción: Solicitar listado de actas. Requisito de datos de entrada RDR6.

BD: Requisito de datos de lectura RDR1.

RDR6: Ficheros almacenados del acta.

Conjunto de ficheros '.pdf' en formato '.zip'.

2.7. Editar categoría informes

RF7: Editar categoría informes.

Entrada: Agente externo: Administrador. Acción: Editar categoría. Requisito de datos de entrada RDE7.

BD: Requisito de datos de lectura RDR7 y de escritura RDW7.

RDE7: Datos de entrada de editar categoría.

Nombre: Cadena de caracteres (40).

RDR7: Datos almacenados de la categoría.

Nombre: Cadena de caracteres(40).

RDW7: Datos a modificar de la categoría.

Nombre: Cadena de caracteres(40).

2.8. Añadir categoría informes

RF8: Añadir categoría informes.

Entrada: Agente externo: Administrador. Acción: Añadir categoría. Requisito de datos de entrada RDE8.

BD: Requisito de datos de escritura RDW8.

RDE8: Datos de entrada para añadir categoría.
Nombre: Cadena de caracteres (40).

RDW8: Datos almacenados de la categoría.
Nombre: Cadena de caracteres(40).

2.9. Eliminar categoría informes.

RF9: Eliminar categoría informes.

Entrada: Agente externo: Administrador. Acción: Eliminar categoría. Requisito de datos de entrada RDE9

BD: Requisito de datos de escritura RDW9.

RDE9: Datos de entrada de eliminar categoría.
Nombre: Cadena de caracteres (40).

RDW9: Datos eliminados de la categoría.
Nombre: Cadena de caracteres(40).

2.10. Editar un informe

RF10: Editar un informe.

Entrada: Agente externo: Administrador. Acción: Editar un informe en concreto. Requisito de datos de entrada RDE10.

BD: Requisito de datos de lectura RDR10 y de escritura RDW10.

RDE10: Datos de entrada del informe.
Nombre: Cadena de caracteres (40).

RDR10: Datos almacenados del informe.
Fichero en formato '.pdf'.

RDW10: Datos a modificar de la categoría.
Fichero en formato '.pdf'.

2.11. Eliminar un informe

RF11: Eliminar un informe.

Entrada: Agente externo: Administrador. Acción: Eliminar un informe existente. Requisito de datos de entrada RDE11.

BD: Requisito de datos de escritura RDW11.

RDE11: Datos de entrada para eliminar del informe.
Nombre: Cadena de caracteres (40).

RDW11: Datos eliminados del informe.
Fichero en formato ‘.pdf’.

2.12. Subir un informe

RF12: Subir un informe.

Entrada: Agente externo: Administrador. Acción: Subir informe. Requisito de datos de entrada RDE12.

BD: Requisito de datos de escritura RDW12.

RDE12: Datos de entrada de informe.
Nombre: Cadena de caracteres (40).

RDW11: Fichero almacenado del informe.
Fichero en formato ‘.pdf’.

2.13. Listar información asociada a un profesor en concreto

RF13: Listar información asociada a un profesor en concreto.

Entrada: Agente externo: Administrador. Acción: Solicitar listado. Requisito de datos de entrada RDE 13.

BD: Requisito de datos de lectura RDR13.

RDE13: Datos de entrada de un profesor.
DNI:Cadena de caracteres(8 dígitos + 1 letra).

RDR13: Datos almacenados del profesor.
Fichero ‘.pdf’ o ‘.csv’ el cual incluye:

- Nombre: Cadena de caracteres (20)
- Apellidos: Cadena de caracteres (30)
- DNI: Cadena de caracteres(8 dígitos + 1 letra)
- Teléfono: Cadena de caracteres (13, pudiendo incluir “+” al inicio de la cadena)
- Dirección: Cadena de caracteres (40)
- Cargo: Cadena de caracteres (20)
- Créditos: Cadena de caracteres (Número positivo (2))

- Grupos que imparten: (Lista de cadenas cada una de 30)
- Subgrupos que imparten: (Lista de cadenas cada una de 30)
- Compensación de créditos: (Número positivo(2))

2.14. Listar información asociada a cada uno de los profesores

RF14: Listar información asociada a cada uno de los profesores.

Entrada: Agente externo: Administrador. Acción: Solicitar listado.

BD: Requisito de datos de lectura RDR14.

RDR14: Un fichero ‘.pdf’ para cada profesor o un único fichero ‘.csv’ el cual incluye:

- Nombre: Cadena de caracteres (20)
- Apellidos: Cadena de caracteres (30)
- DNI: Cadena de caracteres(8 dígitos + 1 letra)
- Teléfono: Cadena de caracteres (13, pudiendo incluir “+” al inicio de la cadena)
- Dirección: Cadena de caracteres (40)
- Cargo: Cadena de caracteres (20)
- Créditos: Cadena de caracteres (Número positivo (2))
- Grupos que imparten: (Lista de cadenas cada una de 30)
- Subgrupos que imparten: (Lista de cadenas cada una de 30)
- Compensación de créditos: (Número positivo(2))

2.15. Imprimir lista de asignaturas de todo el departamento

RF15: Imprimir lista de asignaturas de todo el departamento.

Entrada: Agente externo: Administrador. Acción: Imprimir lista asignatura.

BD: Requisito de datos de lectura RDR15.

RDR15: Fichero almacenado de la lista de asignaturas.

Fichero en formato ‘.xlsx’ o ‘.pdf’.

3. Práctica 1

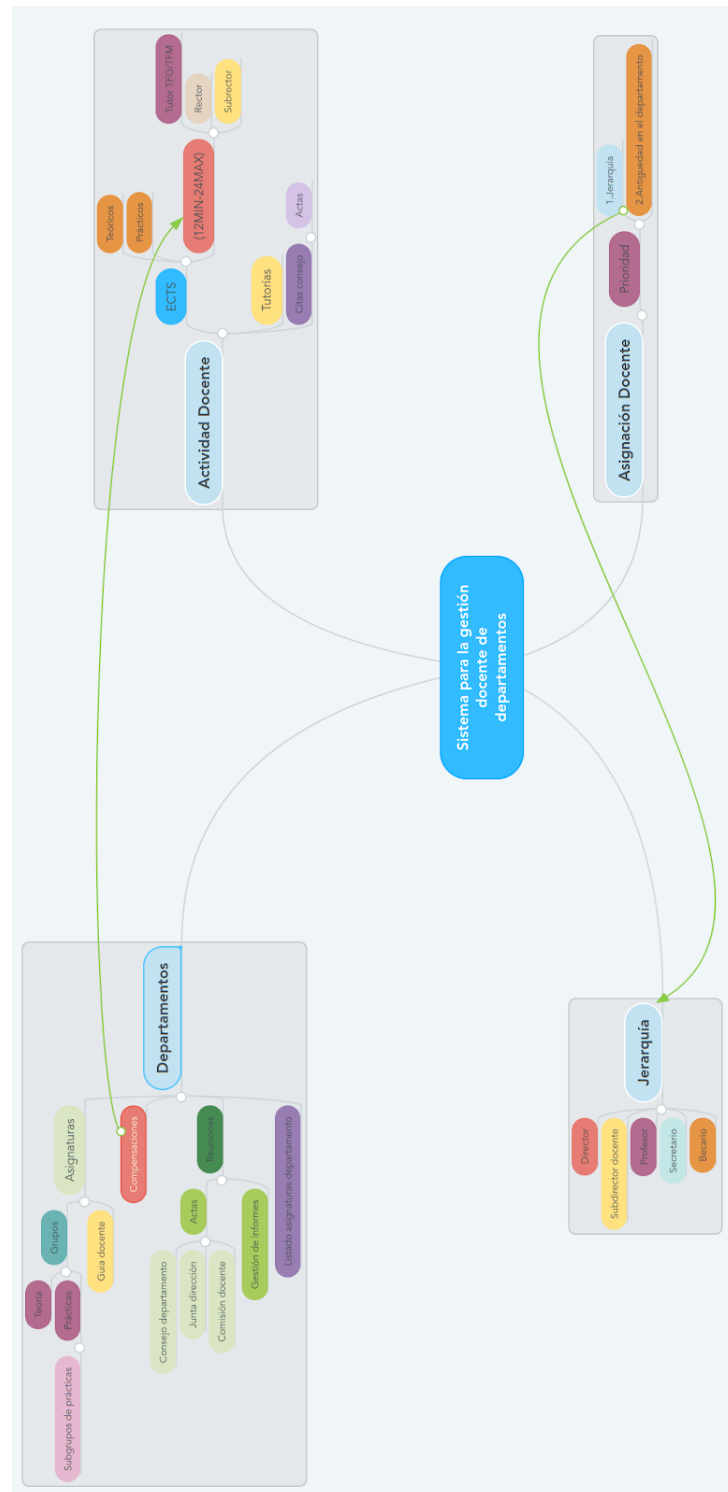


Figura 1: Mapa mental

Caja Negra

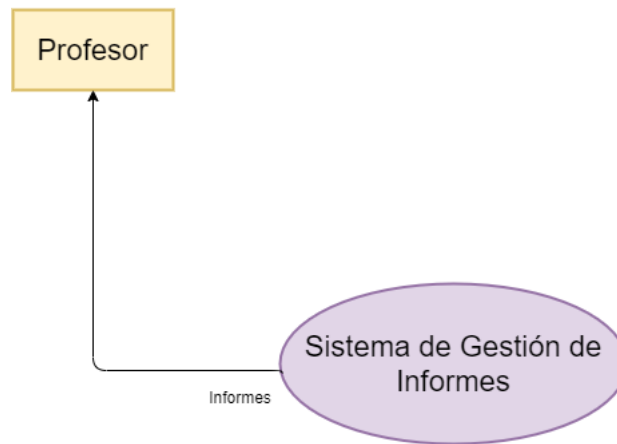


Figura 2: Caja negra (1)

Caja Negra

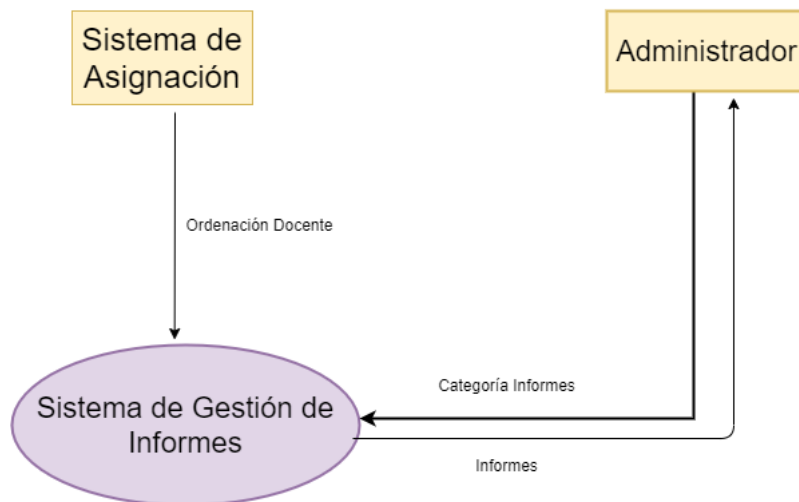


Figura 3: Caja negra (2)

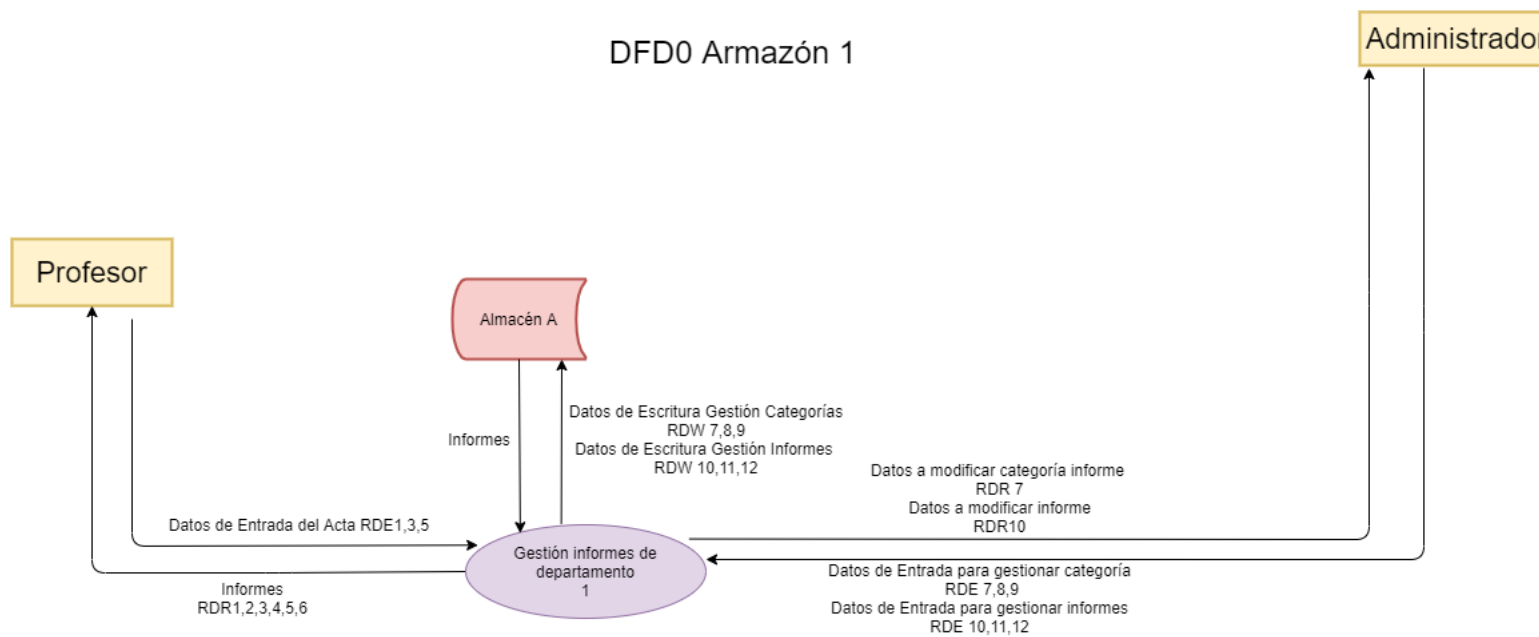


Figura 4: Armazón (1)

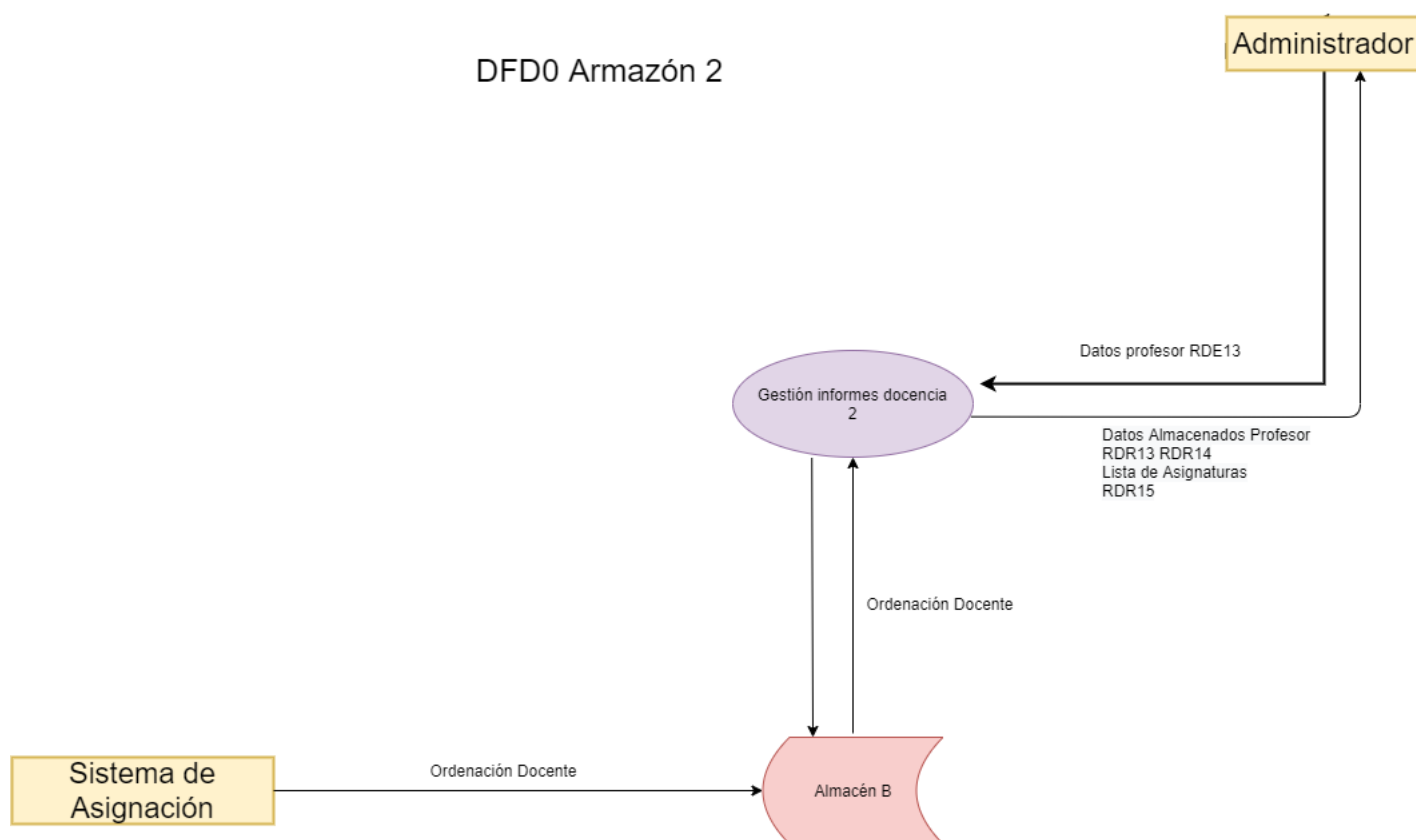


Figura 5: Armazón (2)

DFD1 INFORMES DEPARTAMENTO 1

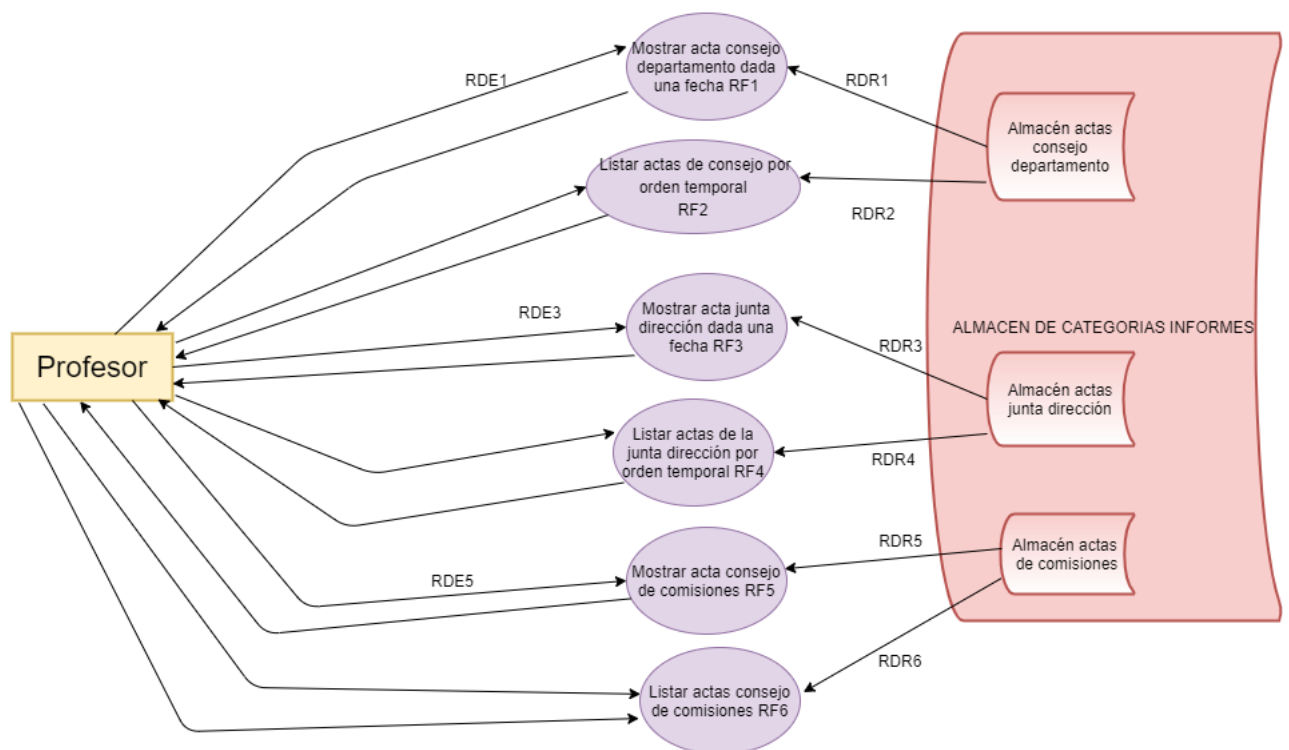


Figura 6: Informes departamento (1)

DFD1 INFORMES DEPARTAMENTO 2

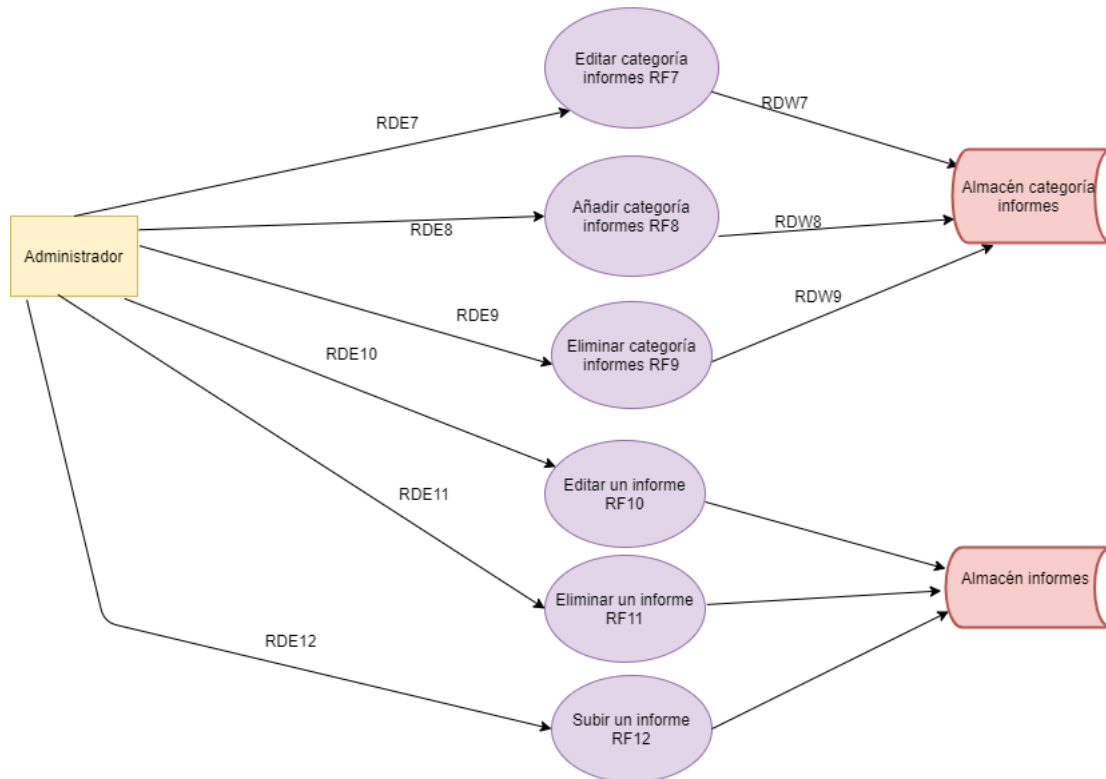


Figura 7: Informes departamento (2)

INFORMES DOCENCIA

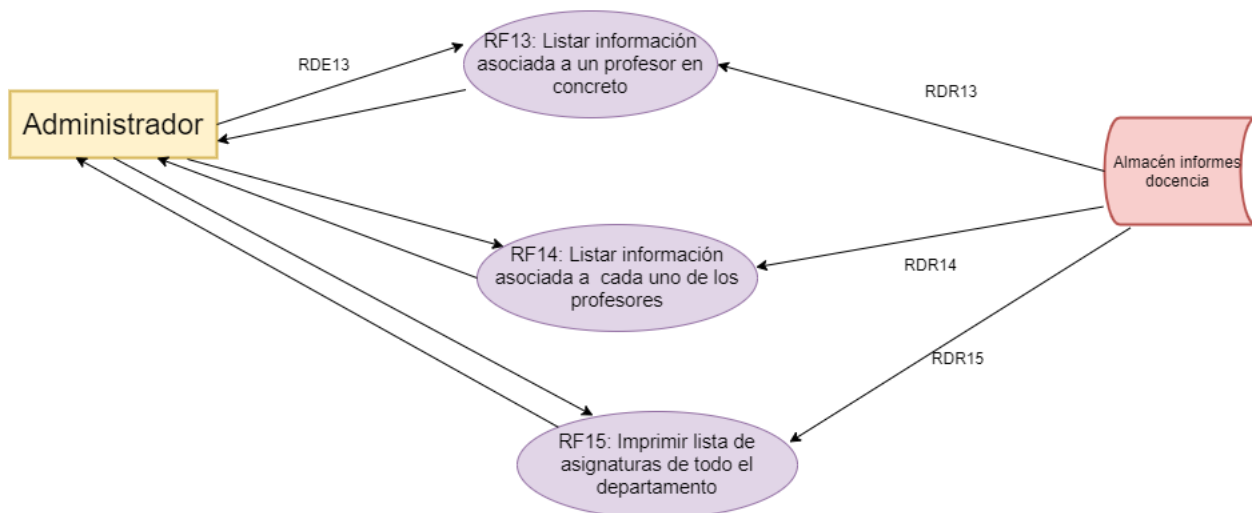


Figura 8: Informes docencia

4. Esquema externo

4.1. DFD0 Armazón 1

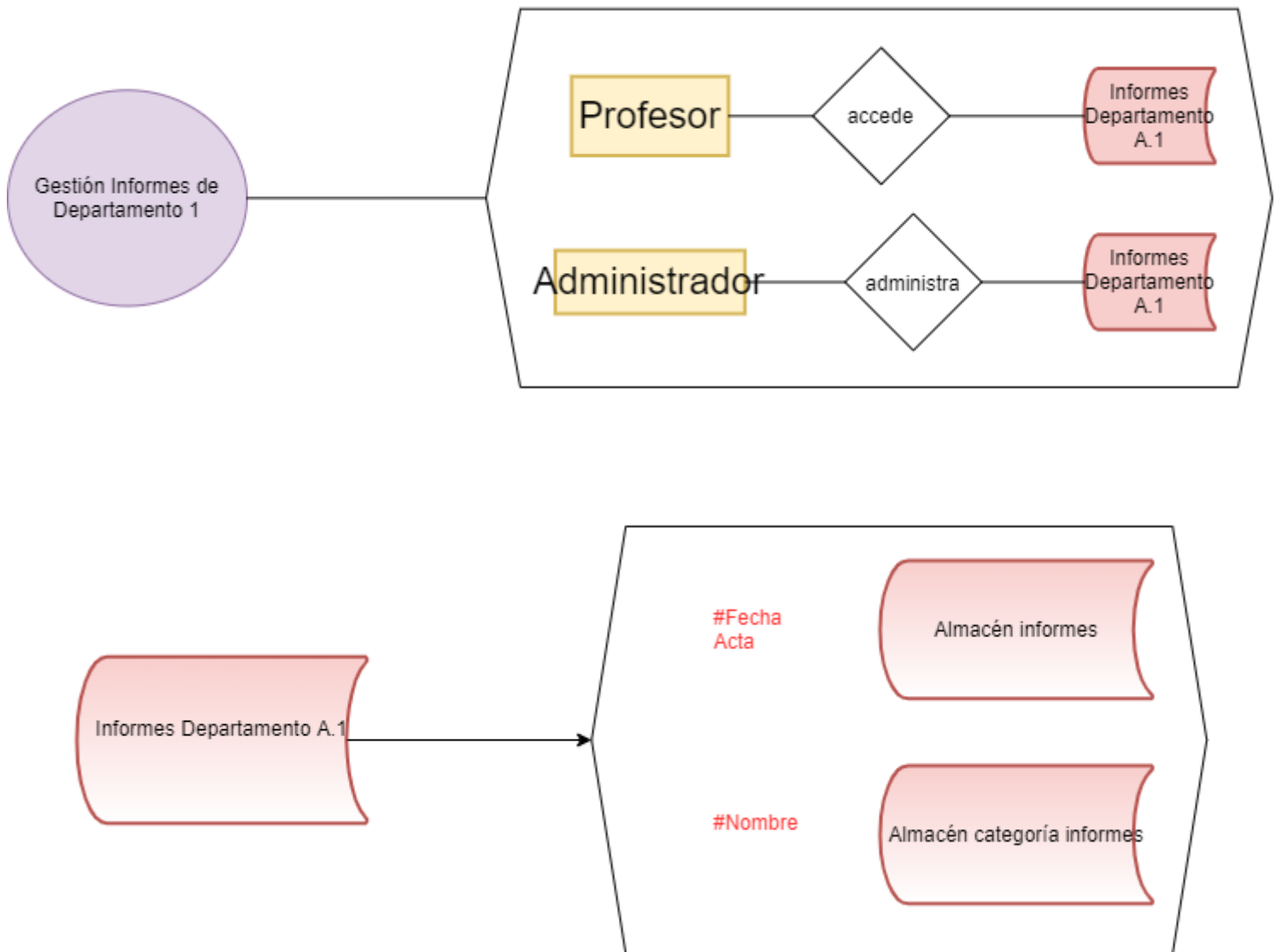


Figura 9: Esquema externo Armazon 1

Almacenes:

Almacén informes de departamento: #Fecha, Acta

Flujos de datos:

Id-Profesor: #DNI, Nombre, Apellidos

4.2. DFD0 Armazón 2

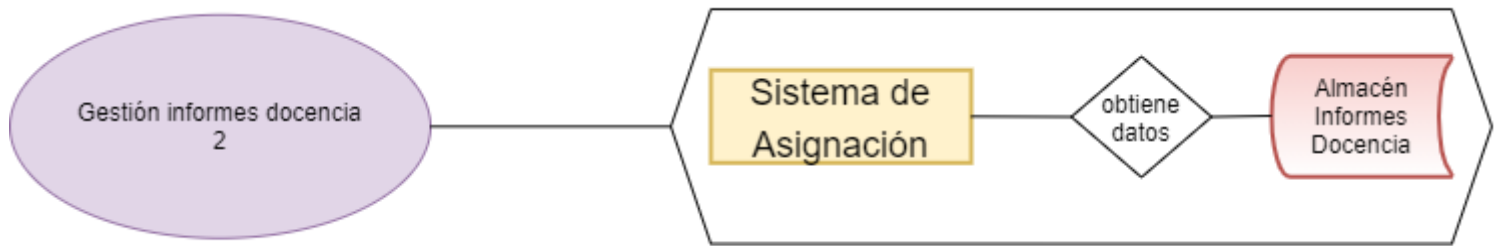
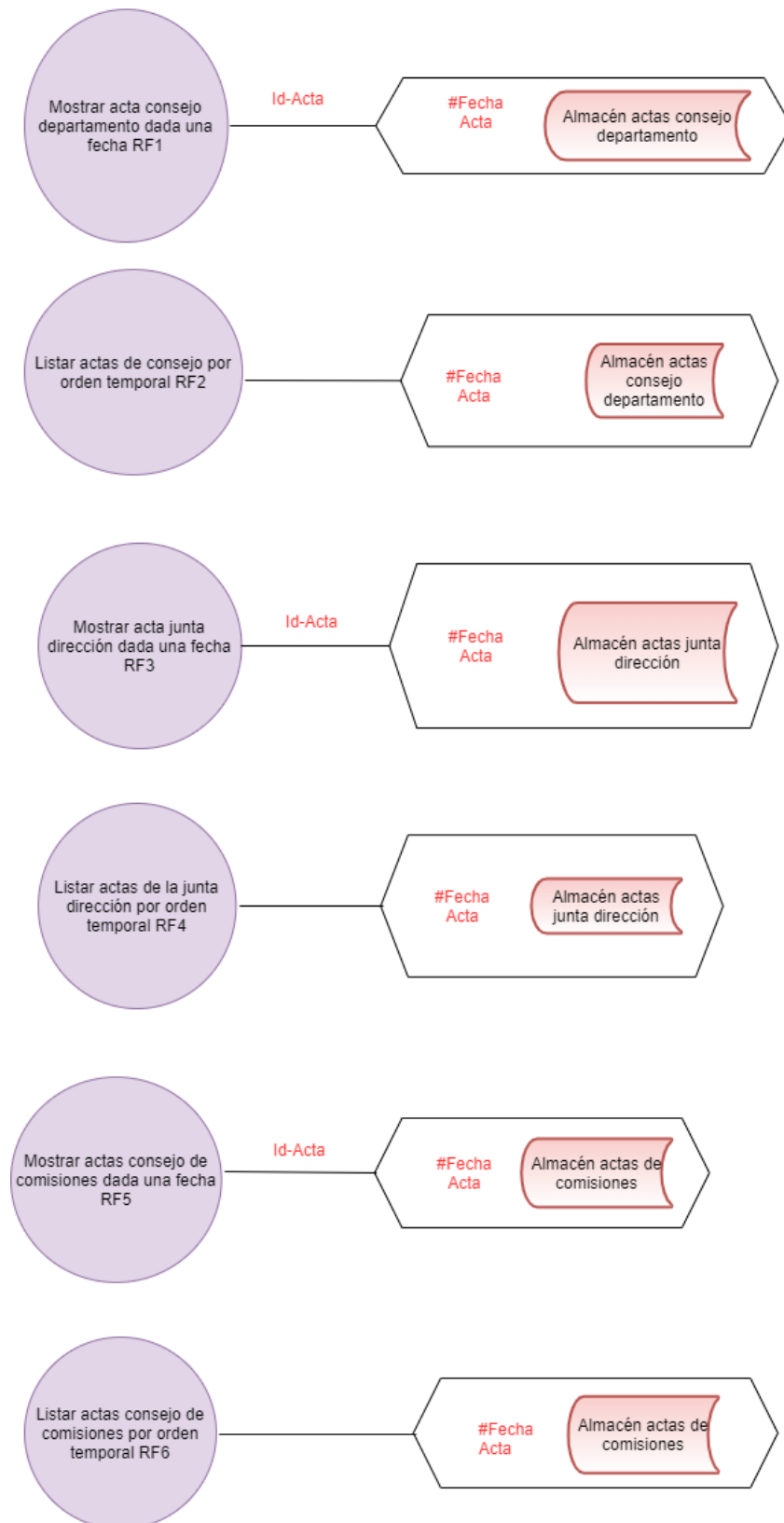


Figura 10: Esquema externo Armazon 1

Almacenes:

Almacén informes de docencia: #Miembro, Creditos, Minimo_Creditos, Pasa_Turno, Entra_F1

4.3. DFD1 Informes de departamento 1



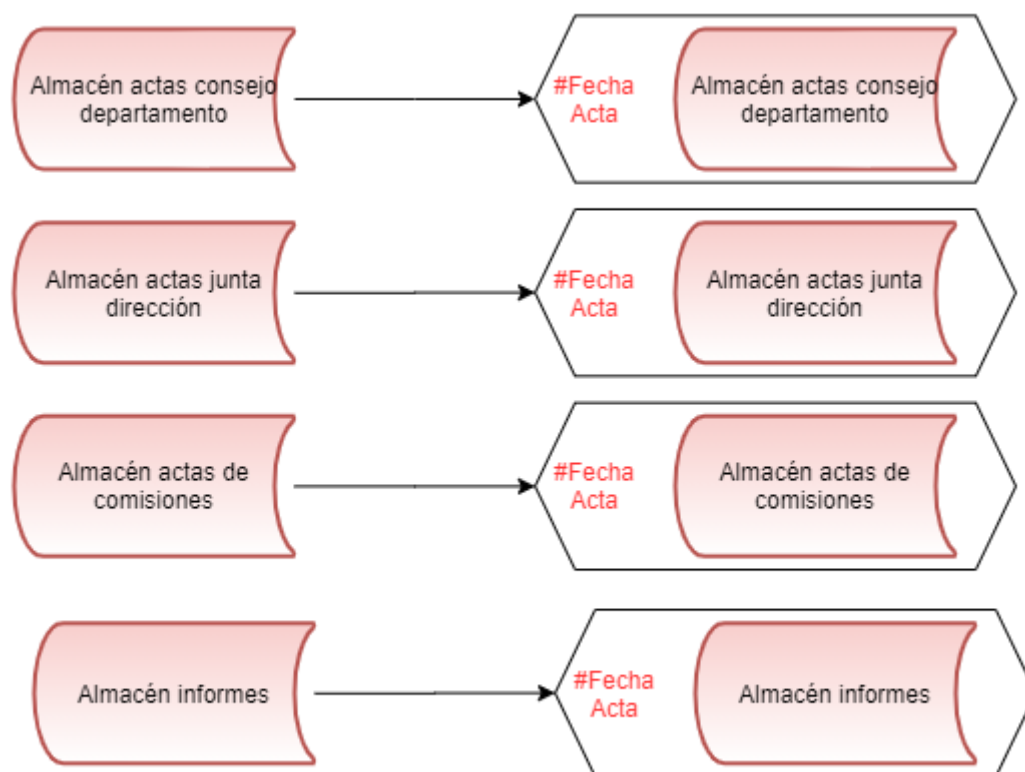


Figura 11: Esquema externo DFD1 Informes de departamento 1

Almacenes:

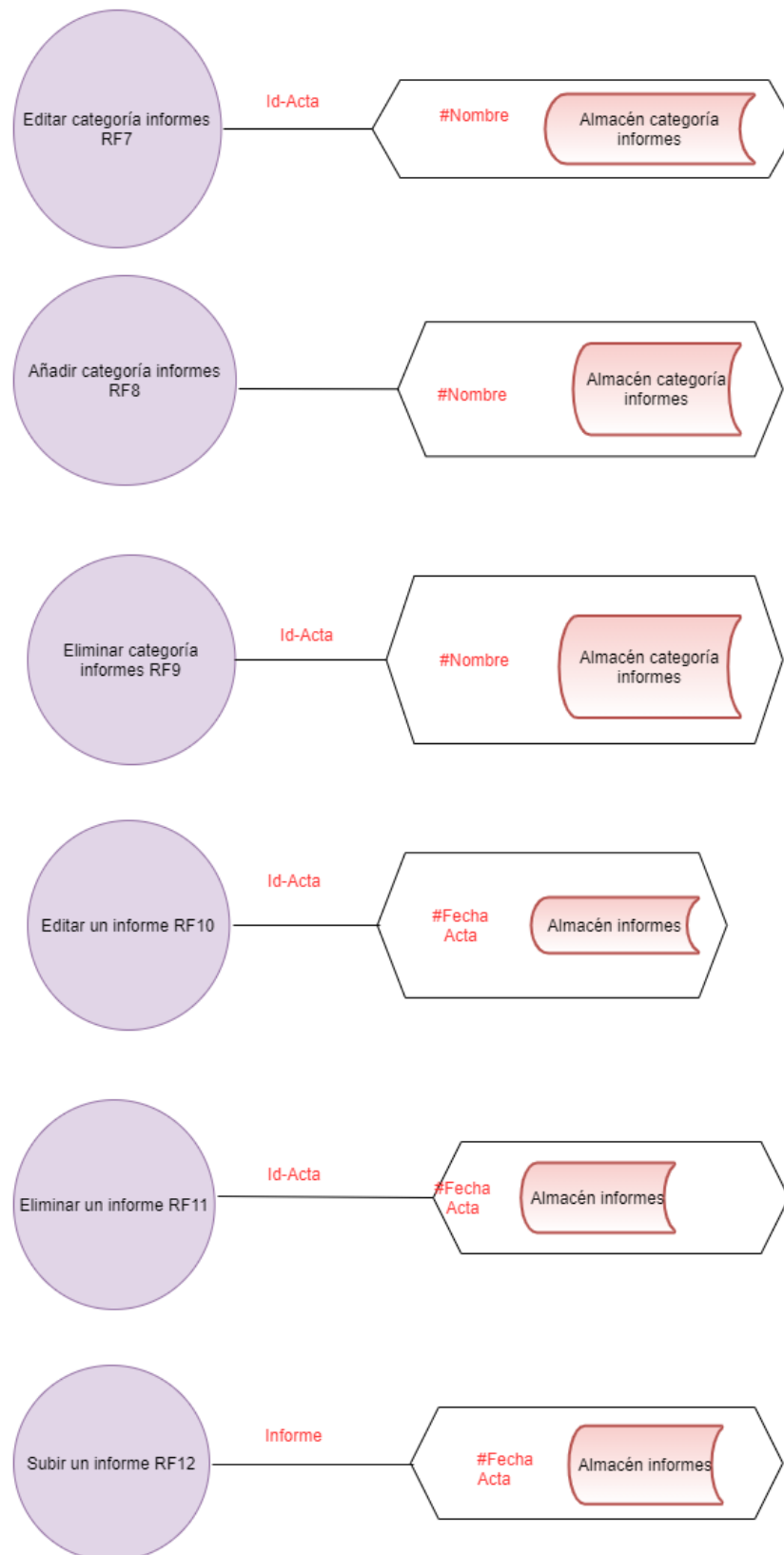
Almacén actas consejo departamento: #Fecha, Acta

Flujos de datos:

Acta: #Fecha

Id-Acta: #Fecha

4.4. DFD1 Informes de departamento 2



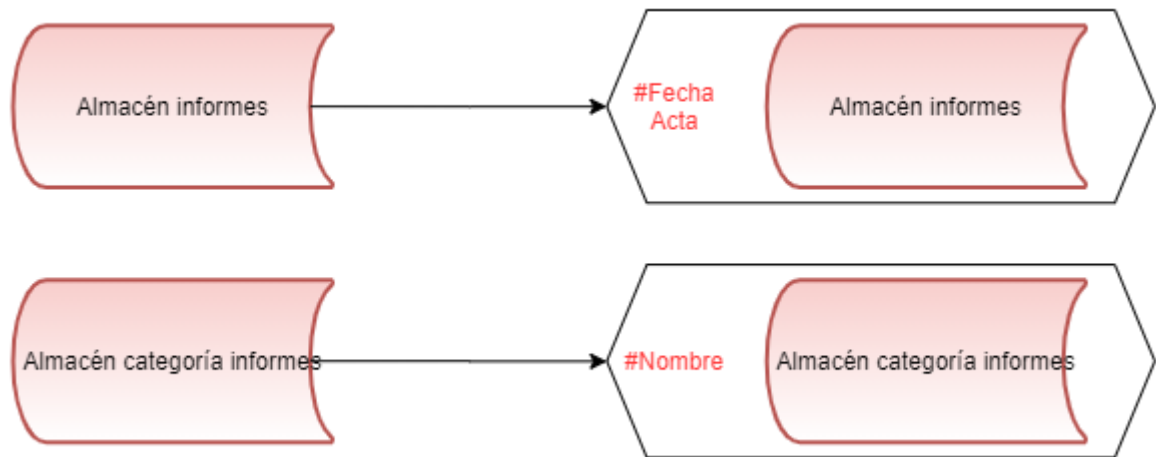


Figura 12: Esquema externo DFD1 Informes de departamento 2

Almacenes:

Almacén categoría informes: #Nombre

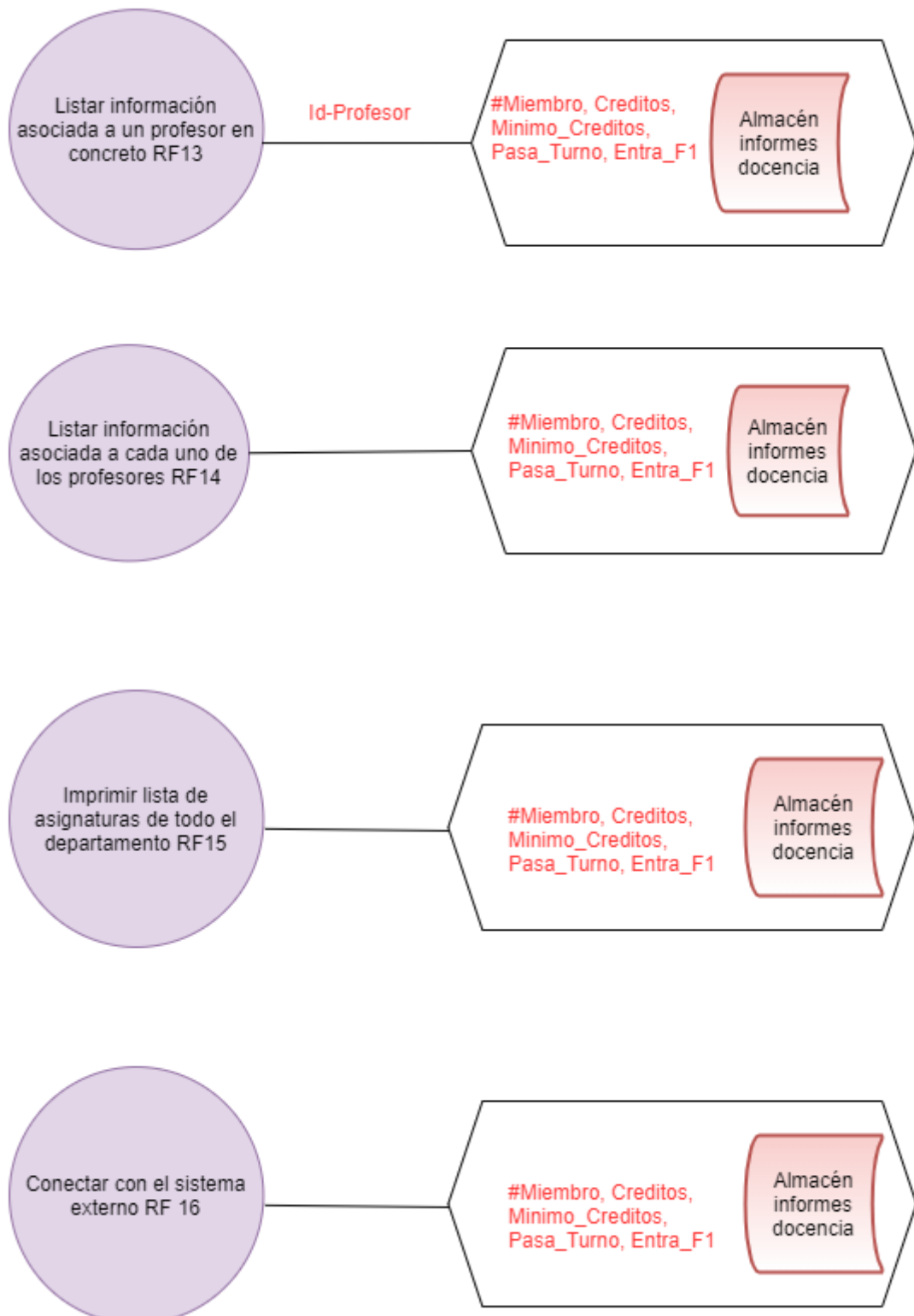
Almacén informes: #Fecha, Acta

Flujos de datos:

Acta: #Fecha

Id-Acta: #Fecha

4.5. DFD1 Informes de docencia 1



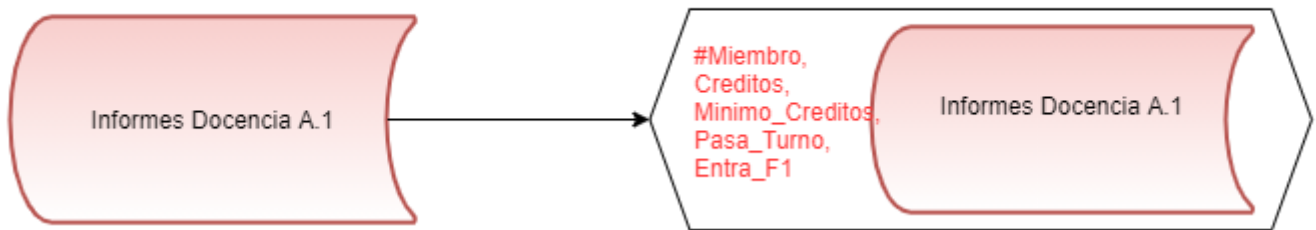


Figura 13: Esquema externo DFD1 Informes de docencia 1

Almacenes:

Almacén informes de docencia: #Miembro, Creditos, Minimo_Creditos, Pasa_Turno, Entra_F1

Flujos de datos:

Id-Profesor: #DNI, Nombre, Apellidos

5. Diagramas E-R

5.1. Informes de departamento

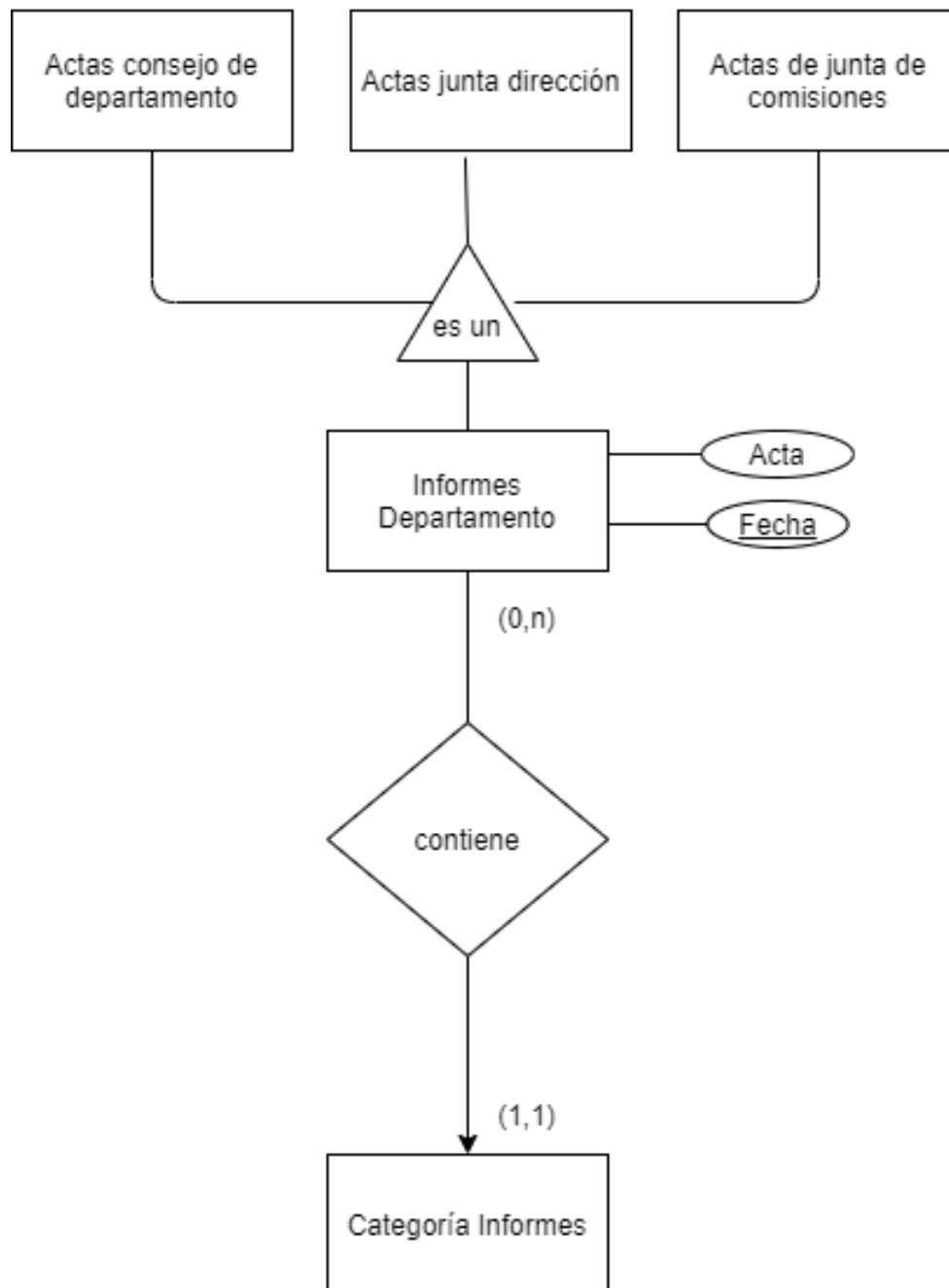


Figura 14: Diagram E-R Informes de departamento

5.2. Informes de docencia

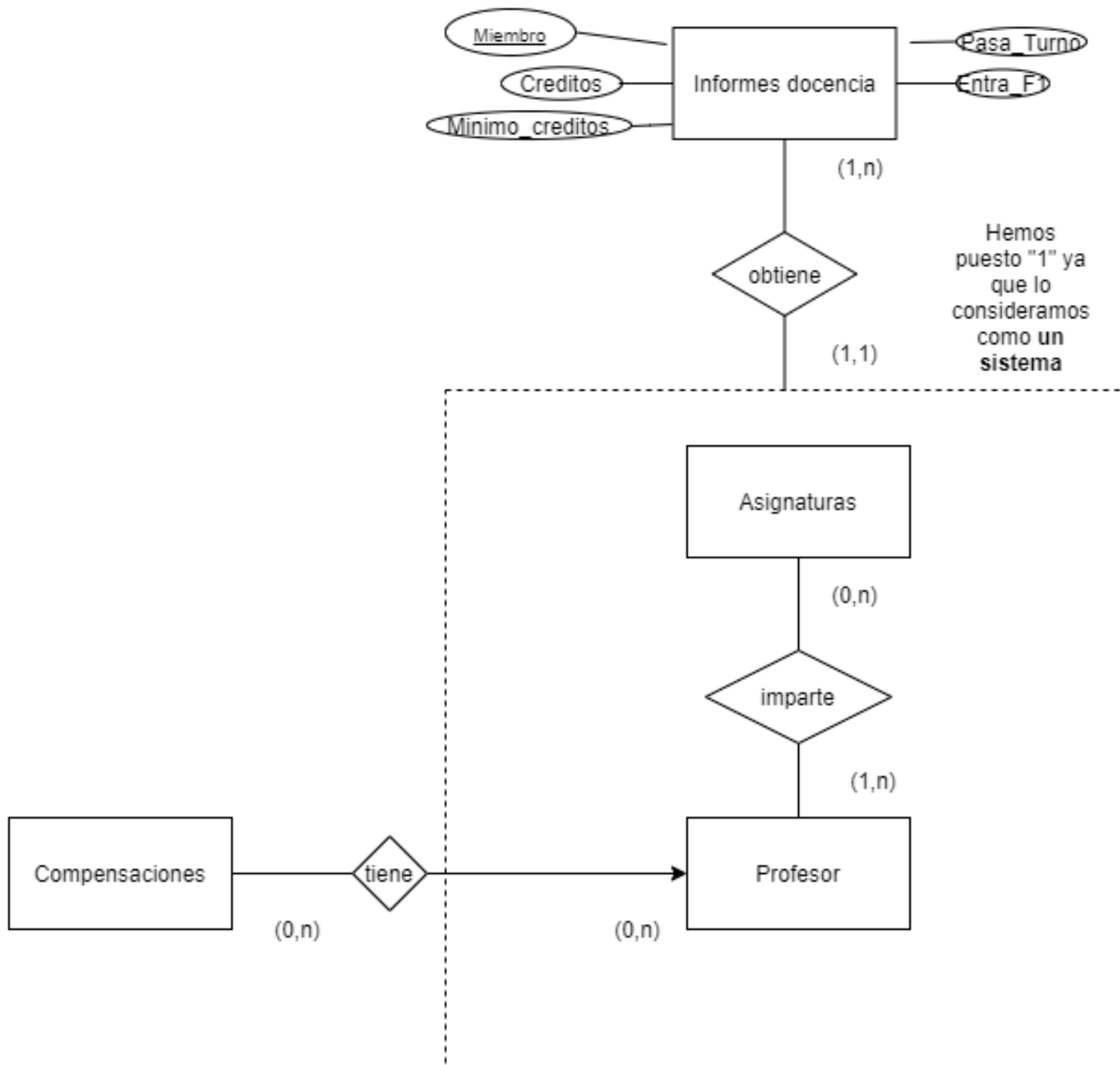


Figura 15: Diagram E-R Informes de docencia

5.3. Paso a Tablas

Tabla: Informes departamento		
fecha	nombre	URL
19/11/2020	Acta del Consejo de Departamento N27	https://ugr.es/home/acta_n27.pdf
20/11/2020	Acta del Consejo de Departamento N28	https://ugr.es/home/acta_n28.pdf
....
12/12/2020	Acta del Consejo de Departamento N56	https://ugr.es/home/acta_n56.pdf


Tabla: Categoría informes	
nombre	ruta
Consejo de Departamento	https://ugr.es/categorias/consejo_departamento.html
Junta dirección	https://ugr.es/categorias/junta_direccion.html
Junta de Comisiones	https://ugr.es/categorias/junta_comisiones.html

Tabla: Informes docencia					
orden	miembro	creditos	minimo_creditos	pasa_turno	entra_F1
1	Carlos Corona Cruz	12	6	No	Si
2	Carlos Corona Cruz	15	12	No	Si
....			
545	Carlos Corona Cruz	12	12	Si	No

Figura 16: Dibujo tablas E-R

6. Interfaz de Usuario

6.1. Informes departamento

**DECSAI**
Departamento de Ciencias de la Computación e I.A.
UNIVERSIDAD DE CORDOBA

Inicio

Informes

Docencia

Informes departamento

Informes docencia

FECHA

NOMBRE

TIPO

TIPO FICHERO

BUSCAR

AÑADIR ACTA

introducir fecha

introducir nombre


Cons. departamento

PDF

FECHA	NOMBRE	TIPO	FORMATO	VISUALIZAR	ULTIMA MODIFICACION
26/11/2020	Acta de departamento n200	Junta de direccion	PDF	DESCARGAR Tam. fichero 200MB	Carlos Corona Cruz 16:48:25 26/11/2020
26/11/2020	Acta de departamento n200	Junta de direccion	PDF	DESCARGAR Tam. fichero 200MB	Carlos Corona Cruz 16:48:25 26/11/2020
26/11/2020	Acta de departamento n200	Junta de direccion	PDF	DESCARGAR Tam. fichero 200MB	Carlos Corona Cruz 16:48:25 26/11/2020
26/11/2020	Acta de departamento n200	Junta de direccion	PDF	DESCARGAR Tam. fichero 200MB	Carlos Corona Cruz 16:48:25 26/11/2020

Figura 17: Interfaz de Usuario de Informes de Departamento


6.2. Informes docencia

 **DECSAI**
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

Inicio Informes Docencia
Informes departamento Informes docencia

ORDEN	MIEMBRO	CRÉDITOS	CR. MIN	PASA TURNO	ENTRA F1	
1	Carlos Cruz Corona	16	12	SI	SI	VER MAS
2	Carlos Cruz Corona	16	12	NO	SI	VER MAS
3	Carlos Cruz Corona	16	12	SI	NO	VER MAS
4	Carlos Cruz Corona	16	12	SI	NO	VER MAS
	Carlos Cruz Corona	16	12	SI	NO	VER MAS

Figura 18: Interfaz de Usuario de Informes de Docencia

 **DECSAI**
Departamento de Ciencias de la Computación e I.A.
Universidad de Granada

Inicio Informes Docencia
Informes departamento Informes docencia

ORDEN	MIEMBRO	CRÉDITOS	CR. MIN	PASA TURNO	ENTRA F1	
1	Carlos Cruz Corona	16	12	SI	SI	VER MAS
2	Carlos Cruz Corona	16	12	NO	SI	VER MAS
3	Carlos Cruz Corona	16	12	SI	NO	VER MAS
4	Carlos Cruz Corona	16	12	SI	NO	VER MAS
	Carlos Cruz Corona	16	12	SI	NO	VER MAS

COMPENSACIONES DOCENTES

COMPENSACIONES POR PROYECTO

Título	Código	Créditos
"ANALISIS DE"	TYKSC-2343	1
Total		1 Crds

COMPENSACIONES POR OTROS CONCEPTOS

Conceptos	Créditos
Reduccion por Edad	1
Total	1 Crds

ASIGNATURAS 2021/2020

Asignatura	Titulación	Tipo	Grupo	Créditos	Fase
Algoritmica	Ingeniería Informática	Prácticas	4	1.5	2ª fase
Total				1.5 Crds	

Figura 19: Interfaz de Usuario de Informes de Docencia (Pop-up)

7. Práctica 2

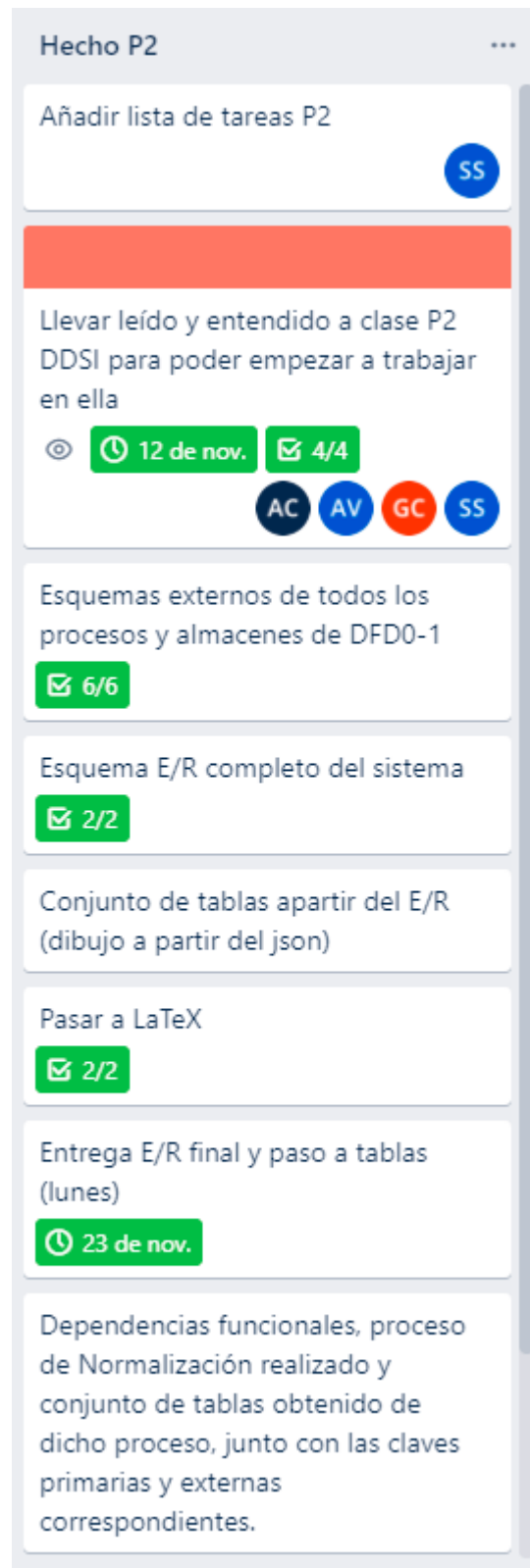


Figura 20: Captura de Trello

8. Seminario 2

8.1. Herramientas utilizadas

- Github
- Visual Studio Code
- MongoDB Community Server + MongoDB Compass
- Node.js + NPM

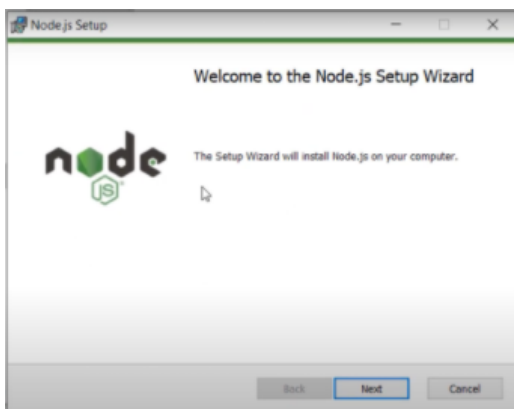
8.2. ¿Porqué estas herramientas?

- Hemos utilizado **Git/Github** ya que dada la situación es la mejor manera de organizar los proyectos grupales y poder trabajar todos a la vez añadiendo nuevas funcionalidades. Otra de las razones por las cuales lo hemos escogido es que como muchas empresas utilizan esta herramienta, pues nos ha parecido bastante interesante empezar a aprender y a acostumbrarnos a trabajar con esta herramienta.
- Hemos escogido **MongoDB** ya que es el SGBD perfecto para bases de datos no relacionales para documentos como es nuestro caso (Gestión de informes y de departamentos). Además el trabajo se facilita con el uso de herramientas como MongoDB Community Server (el cual instala el servidor MongoDB) y más en concreto MongoDB Compass (como SGBD).
- Hemos utilizado **Node.js** ya que es uno de los más conocidos, además de disponer de una gran cantidad de frameworks que facilitan el trabajo (como por ejemplo Express el cual estamos utilizando para la interfaz gráfica en el seminario 3).
- Además un punto a favor de todas estas herramientas es que tienen compatibilidad para diferentes sistemas operativos, por lo que no es necesario que todos los integrantes del equipo tengan el mismo sistema.

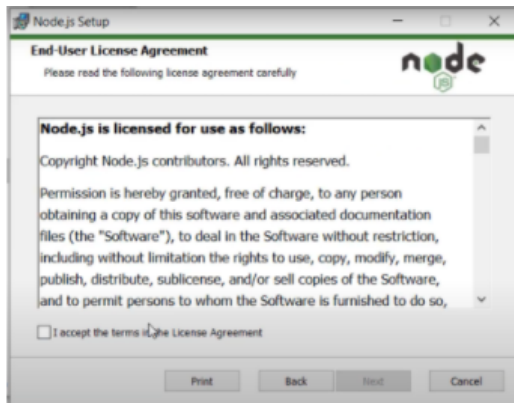
8.3. Instalación de NodeJS y NPM

A continuación explicaremos como instalar node.js junto con npm en Windows 10. Para ello, debemos seguir los siguientes pasos:

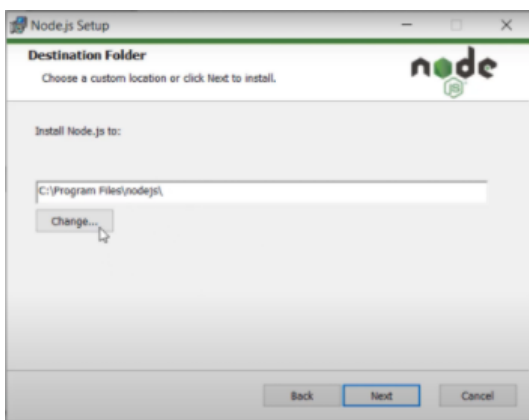
1. Ir a la página web de Node.js y descargamos el instalador.
2. Ejecutamos el Node.js Setup Wizard



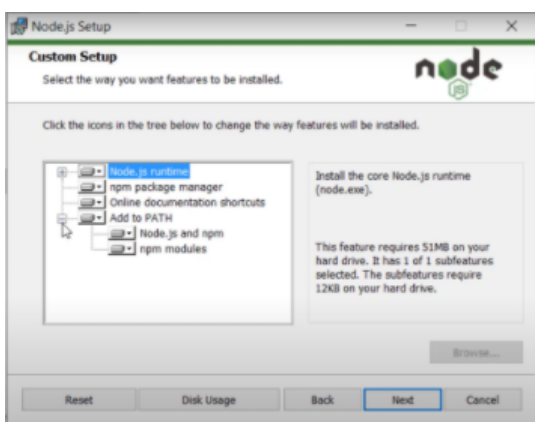
3. Aceptamos los términos y condiciones de uso:



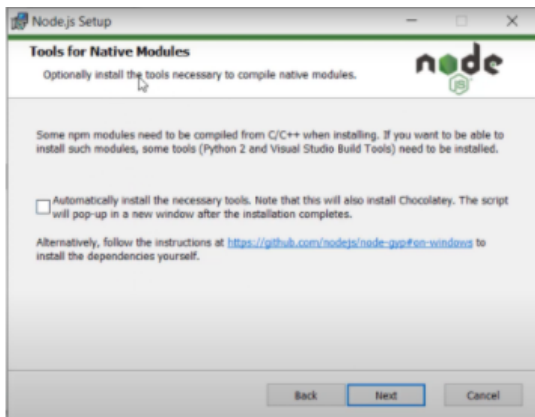
4. Indicamos en qué carpeta queremos instalar Node.js:



5. Escogemos los paquetes que deseamos instalar, incluyendo npm:



6. Marcamos la casilla para instalar automáticamente las herramientas necesarias y pulsamos siguiente hasta que acabe:



7. Finalmente, comprobamos que se ha instalado correctamente node.js y npm.

```
Simbolo del sistema
Microsoft Windows [Versión 10.0.18363.1256]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Users\alvaritoactivated>node -v
v14.15.1

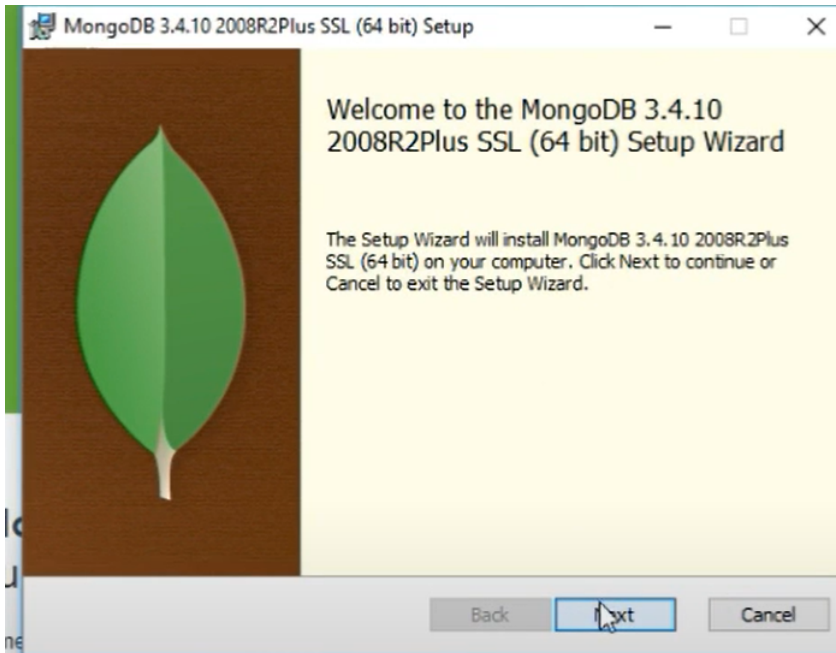
C:\Users\alvaritoactivated>npm -v
6.14.8

C:\Users\alvaritoactivated>
```

8.4. Instalación de MongoDB Community Server

Para instalar MongoDB Community Server debemos seguir los siguientes pasos:

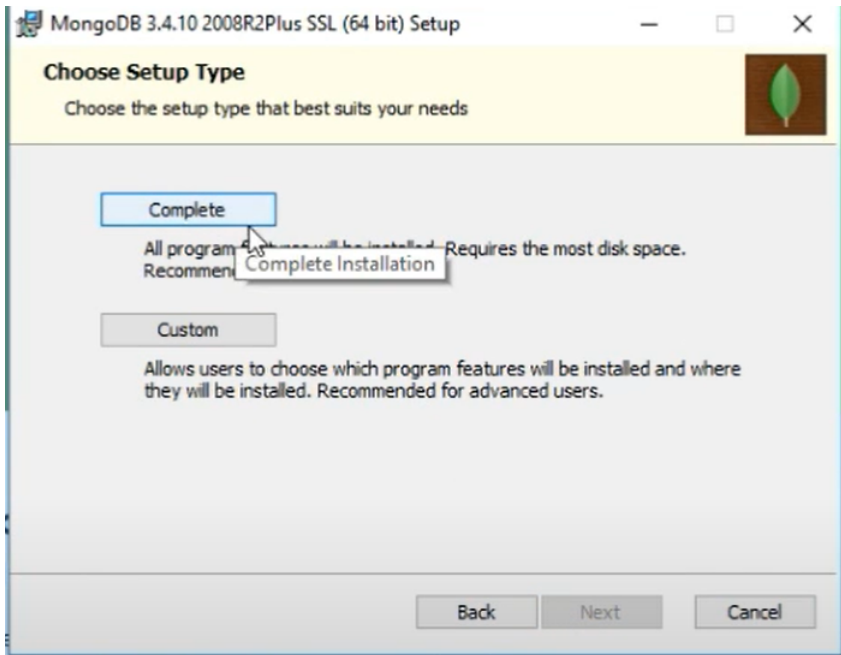
1. En primer lugar, nos dirigimos a la web de MongoDB para descargarnos el instalador.
2. Ejecutamos el instalador:



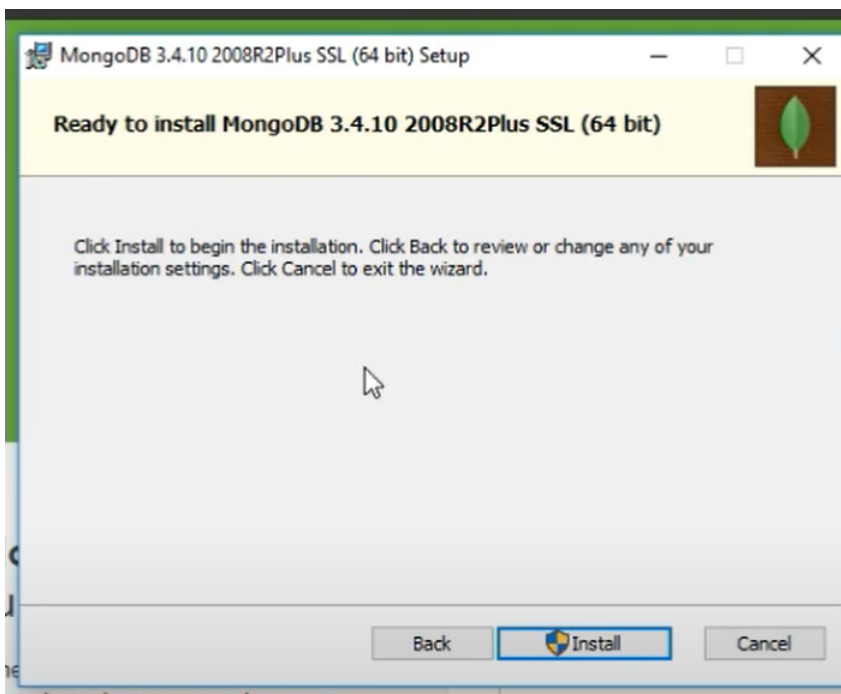
3. Aceptamos los términos y condiciones:



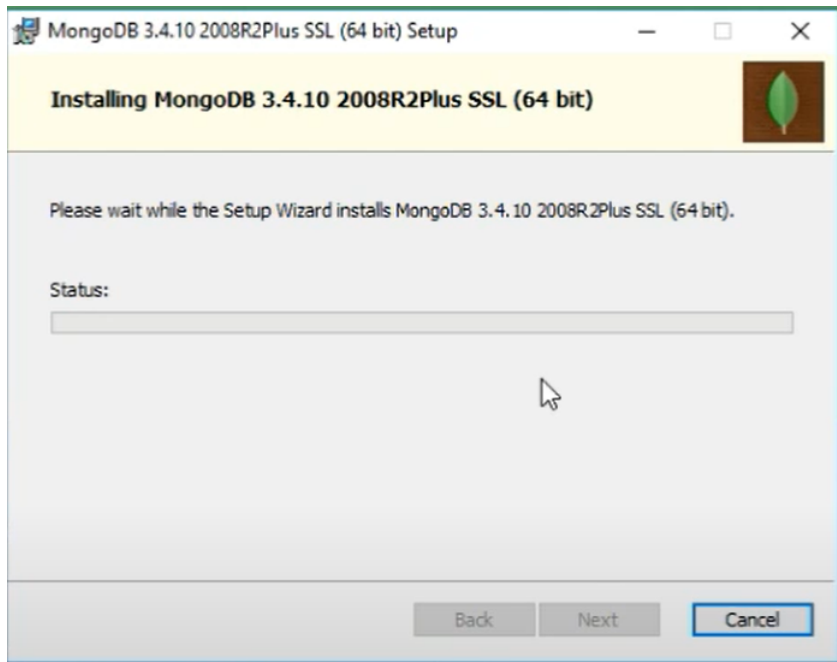
4. Escogemos la instalación completa:



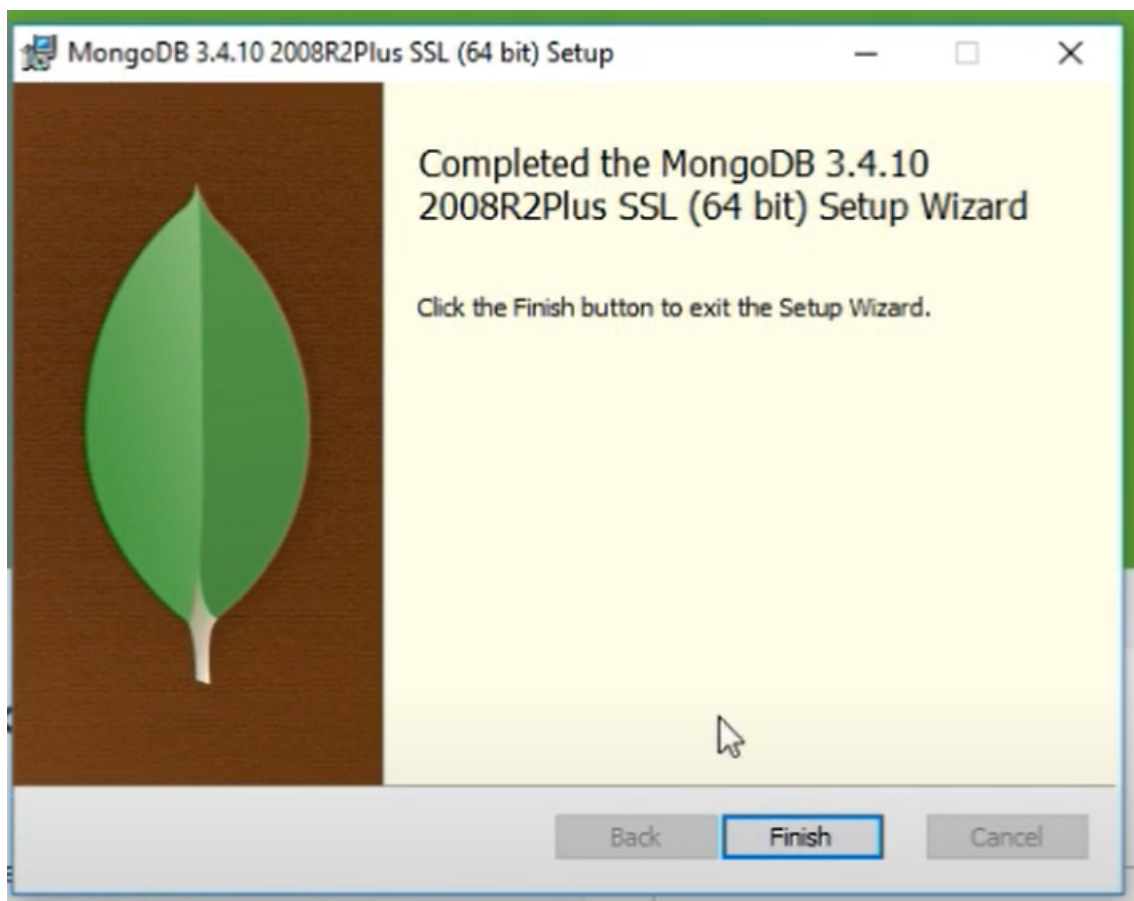
5. Hacemos clic en instalar:



6. Esperamos a que se termine la instalación:



7. Finalmente, ya tenemos instalado MongoDB Community Server:



8.5. Conexión a la base de datos

Es importante crear primero un proyecto de node.js desde la terminal de Windows mediante el comando:

```
npm init -f
```

8.5.1. Creación del proyecto

Una vez ejecutada la anterior orden, en nuestro proyecto se creará automáticamente el archivo **package.json**. Este archivo es la configuración fundamental de nuestro proyecto y es el siguiente:

```
{
  "name": "ddsic2_7",
  "version": "1.0.0",
  "description": "*_Alberto\\r*_Santiago\\r*_Germ n\\r*_ lvaro ",
  "main": "server.js",
  "dependencies": {
    "express": "^4.17.1",
    "mongodb": "^3.6.3",
    "mongoose": "^5.11.2",
    "pug": "^3.0.0"
  },
  "devDependencies": {
    "browser-sync": "^2.26.13",
    "nodemon": "^2.0.6"
  },
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "nodemon ./server.js",
    "ui": "browser-sync start --config bs-config.js"
  },
  "repository": {
    "type": "git",
    "url": "git+https://github.com/acanojaen/ddsic2_7.git"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "bugs": {
    "url": "https://github.com/acanojaen/ddsic2_7/issues"
  },
  "homepage": "https://github.com/acanojaen/ddsic2_7#readme"
}
```

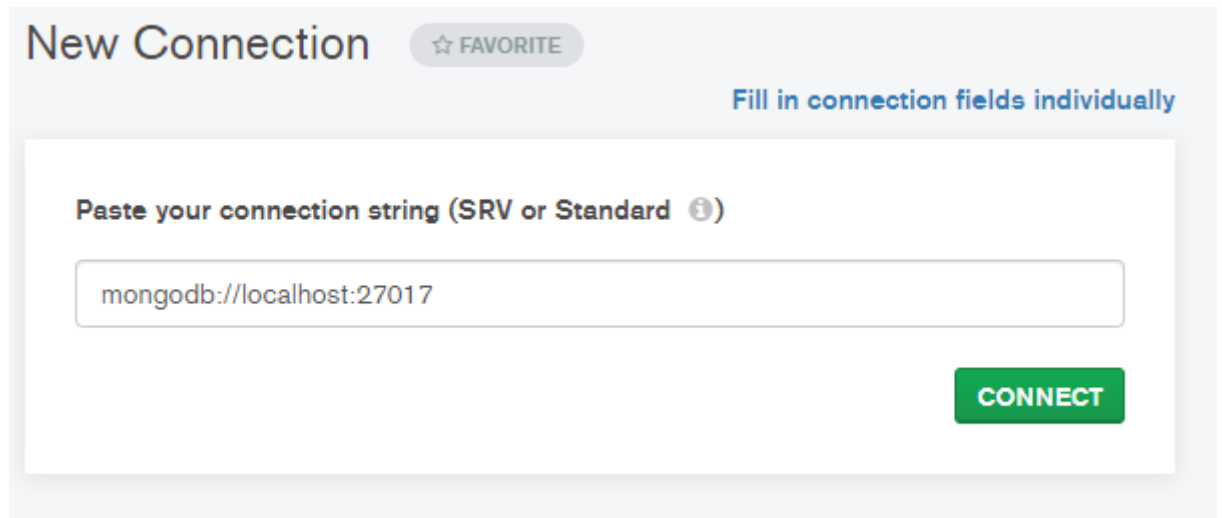
8.5.2. Conexión desde NodeJS a MongoDB

```
const bd = require('mongoose');
const mongo_port = process.env.PORT || "27017";
bd.Promise = global.Promise;
bd.connect('mongodb://localhost:' + mongo_port + '/almacen').then(() => {
  // exito
  console.log('La conexión a MongoDB se ha realizado correctamente');
}).catch(err => console.log(err)); // error
```

Para que se entienda un poco más el fragmento de código anterior, el proceso sería el siguiente: El primer paso es incluir el módulo mongoose, que se realiza a través de la función require. Una vez instalado este módulo, podemos utilizar las funciones necesarias para poder crear la conexión con la base de datos. A continuación, especificamos nuestra cadena después del connect para realizar la conexión a la base de datos. En la cadena de conexión, hay 1 valores clave que se pasa (que es el puerto para la conexión con MongoDB).

8.5.3. Comprobamos la conexión a MongoDB

Para comprobar que la conexión se ha establecido correctamente, desde MongoDB Compass (que es un Sistema de Gestión de Bases de datos) y accedemos a localhost en el puerto establecido en el paso anterior: Una vez que estamos conectados podemos ver las distintas bases de datos creadas,



New Connection ☆ FAVORITE

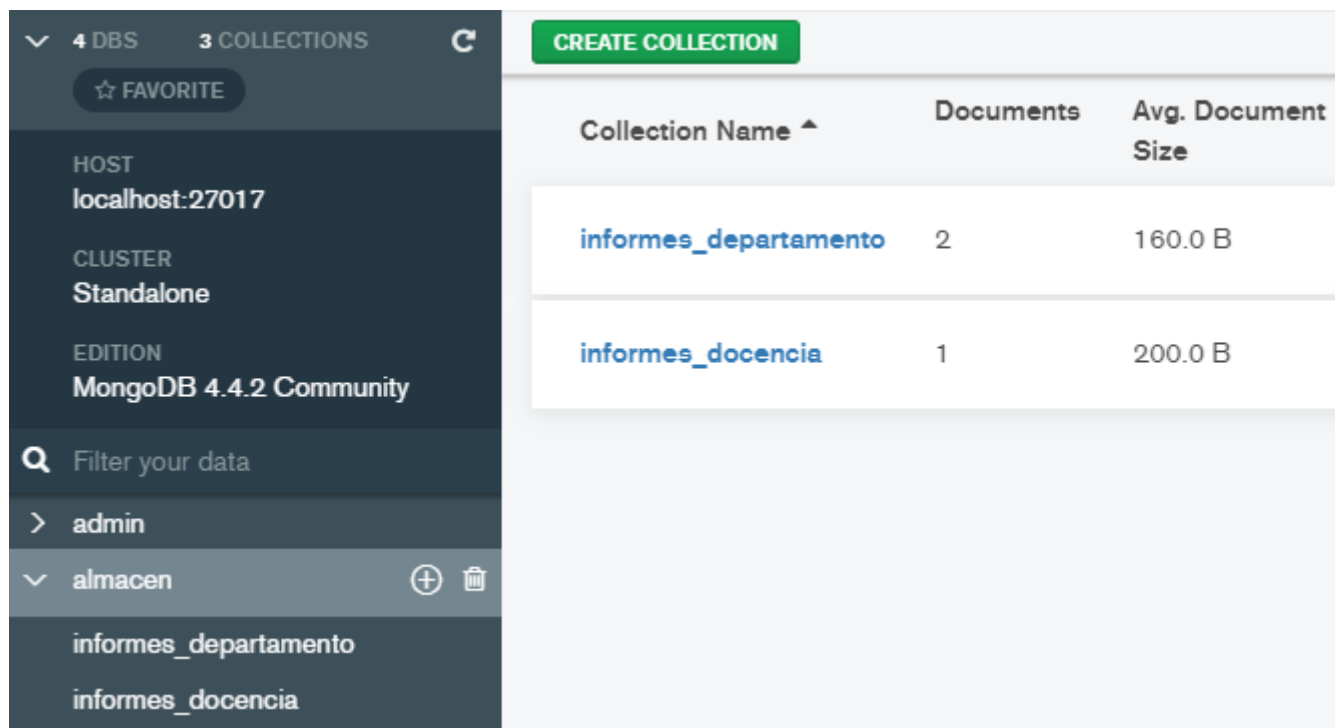
Fill in connection fields individually

Paste your connection string (SRV or Standard ⓘ)

mongodb://localhost:27017

CONNECT

nosotros hemos creado una base de datos llamada almacen, en la que se incluyen dos tablas:



4 DBS 3 COLLECTIONS ⌂

☆ FAVORITE

HOST
localhost:27017

CLUSTER
Standalone

EDITION
MongoDB 4.4.2 Community

Filter your data

> admin

▼ almacen ⊕ 🗑️

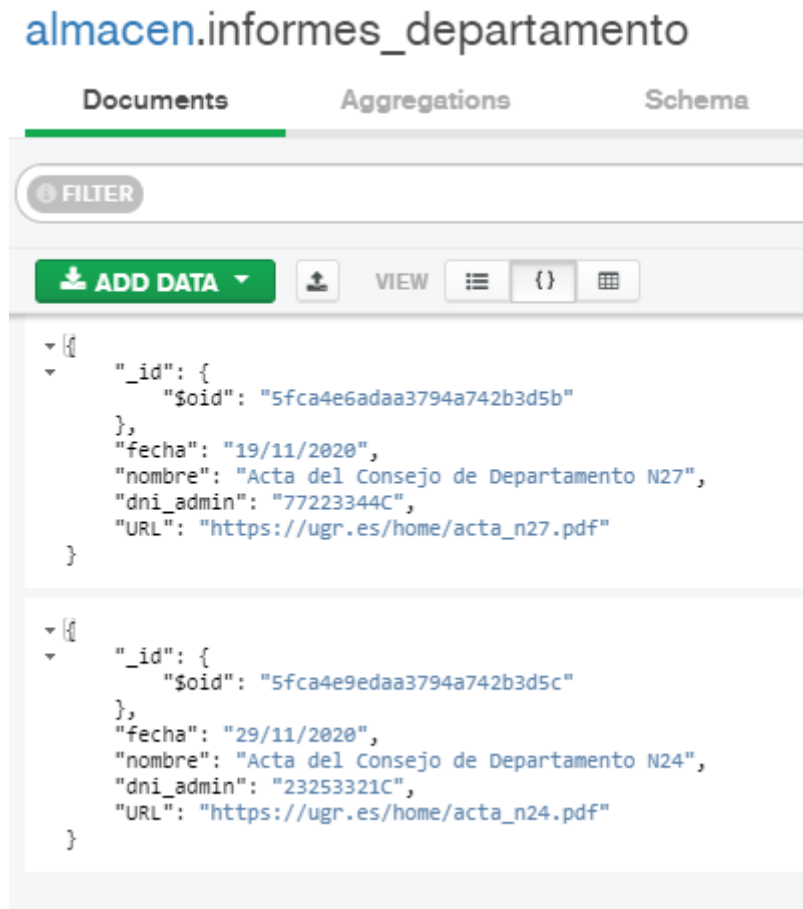
informes_departamento

informes_docencia

CREATE COLLECTION

Collection Name ^	Documents	Avg. Document Size
informes_departamento	2	160.0 B
informes_docencia	1	200.0 B

Dentro de cada tabla introducimos los ficheros json que habíamos creado anteriormente (y que están adjuntos más abajo):



8.5.4. Archivos JSON (plantillas)

1. Archivo informes de departamento

```
[{
  "_id": {
    "$oid": "5fca4e6adaa3794a742b3d5b"
  },
  "fecha": "19/11/2020",
  "nombre": "Acta del Consejo de Departamento N27",
  "dni_admin": "77223344C",
  "URL": "https://ugr.es/home/acta_n27.pdf"
},{
  "_id": {
    "$oid": "5fca4e9edaa3794a742b3d5c"
  },
  "fecha": "29/11/2020",
  "nombre": "Acta del Consejo de Departamento N24",
  "dni_admin": "23253321C",
  "URL": "https://ugr.es/home/acta_n24.pdf"
}]
```

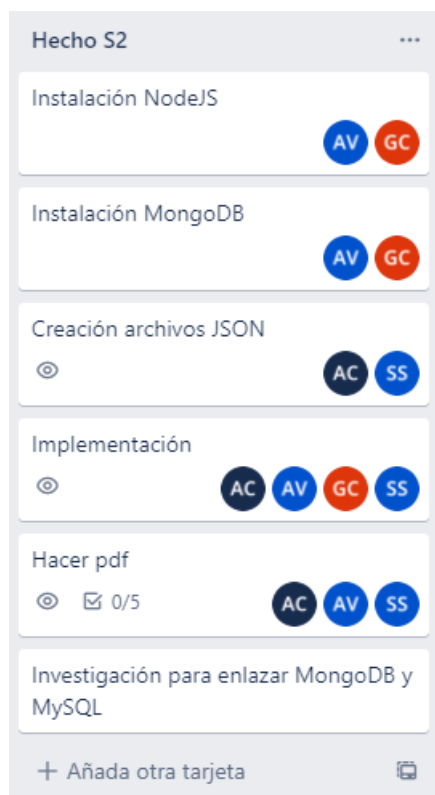
2. Archivo **informes de docencia**

```
[{
  "_id": {
    "$oid": "5fca50cadaa3794a742b3d5d"
  },
  "orden": "1",
  "miembro": [
    {
      "nombre": "Franciso_Jose",
      "apellido": "Cubero_Bon",
      "dni": "23541100C"
    }
  ],
  "creditos": "12",
  "minimo_creditos": "9",
  "pasa_turno": "si",
  "entra_F1": "no"
}]
```

8.5.5. Bibliografía

<https://victorroblesweb.es/2018/01/09/conexion-a-mongodb-desde-nodejs-api-restful-node/>
<https://www.mongodb.com/products/compass>
https://www.youtube.com/watch?v=2KMQdqDk9e8&ab_channel=Fazt
https://www.youtube.com/watch?v=2KMQdqDk9e8&ab_channel=Fazt

8.5.6. Capturas Trello



9. Seminario 3

9.1. Herramientas utilizadas

Ademas de todas las herramientas utilizadas en el seminario anterior vamos a contar tambien con:

- **PUG** - es una librería para hacer templates basada en Javascript, y muy usado para el desarrollo web. Lo hemos elegido ya que su sintaxis es muy sencilla (no necesita etiquetas, solo indentaciones).
- **Multer** - es un middleware para Express y NodeJS que hace que sea fácil manipular este multipart/form-data cuando tus usuarios suben archivos.

9.2. Lista de asignaturas y contenidos vistos

Fundamentos de Redes:	<ul style="list-style-type: none">• Desarrollo de programas básicos de transmisión de datos• Práctica 1: Configuración de servicios de acceso remoto, transmisión de ficheros y acceso web (1 Ejercicio)
Ingeniería de Servidores:	<ul style="list-style-type: none">• Práctica 2 Sesión 3: Instalación servidor web básico usando LAMP
Inteligencia Artificial	<ul style="list-style-type: none">• Práctica 1: bot conversacional con AIML basado en XML.

9.3. Conclusión

En nuestro grupo somos 4 personas, 2 de ellas (Álvaro y Santiago) están en su tercer año de carrera y no han visto nada o de forma muy escasa conceptos relacionados con la programación web, frontend, backend, HTML,CSS y el resto de conceptos que se mencionan en el seminario. Lo unico que hemos visto ha sido:

- En Fundamentos de Redes hemos abierto puertos HTTP, y luego mostrado un servidor realmente muy básico en el que se nos pide un login y una contraseña para acceder.
- En Ingeniería de Servidores también mostramos un servidor (utilizando LAMP) con una página en HTML aún más básica la cual muestra una frase según si se conecta correctamente o no. Además accedemos al frontend de una app desde el servidor HTTP para monitorizar el PC.
- En inteligencia artificial trabajamos en una práctica con AIML el cual es un lenguaje de marcado basado en XML.

Por otro lado Germán que está en la especialidad de Sistemas de Información ha visto:

- Learning Analytics (Erasmus) (Implementar un sistema de recomendación para Netflix con su correspondiente página web para la visualización)

Alberto, que está en la especialidad también de Sistemas de Información ha visto:

- Recuperación de Información (donde se ha trabajado con archivos .json)
- Programming Web Services (asignatura del erasmus convalidada por Programación Web) donde se ha trabajado en programación de una API REST con mongoDB y nodeJS.

9.4. Autoevaluación

9.4.1. Herramientas utilizadas

Entre todos hemos discutido acerca de las herramientas que usar basándonos en nuestra experiencia previa en otras asignaturas o proyectos.

9.4.2. Instalación

Hemos buscado la forma de instalar las herramientas seleccionadas compartiendo con el resto de compañeros la instalación para así poder tener cada uno todas las herramientas que vamos a utilizar.

9.4.3. Conexión con la BD

Alberto ha sido quien nos ha guiado con la creación de la base de datos y su conexión, ya que previamente ha trabajado con MongoDB y NodeJS.

9.5. Seminario

Respecto al resto de tareas que conlleva el seminario, todos hemos aportado, repartiendo pequeñas tareas para trabajar de forma eficiente.

Cabe destacar que hemos aprovechado al máximo todas las horas de trabajo en clase, así como las tutorías con el profesor, por lo que ha sido un trabajo llevadero en el que cada participante ha trabajado y aportado al grupo.

9.6. Notas

- Santiago González Silot: 9
- Germán Calleja Rider: 9
- Alberto Cano Jaen: 10
- Alvaro Perea Vega: 9

9.7. Estructura de la Aplicación

Nuestra aplicación se estructura entorno al documento **server.js** en el cual en el seminario anterior, creamos una conexión con la base de datos establecida y un servidor HTTP en el puerto 8080.

```
'use strict'

/**
 * Modulos externos requeridos
 */
const bodyParser = require("body-parser");
const express = require("express");
const path = require("path");
const mongodb = require('mongodb');
const app = express();
var router = express.Router();
var common = require('./common');

/**
 * Configuracion
 */
var config = {
  mongo_port: 8080, dbURL: "mongodb://localhost:27017", dbName: "almacen"
}

/**
 * Configuracion de la App
 */
app.listen(config.mongo_port, function() {
  console.log('Servidor HTTP en la url http://localhost:' + config.mongo_port + '/')
});

app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: true }));
app.use(express.static(path.join(__dirname, "public")));
app.use('/pdf', express.static(__dirname + '/public/pdf'));
app.set("views", path.join(__dirname, "views"));
app.set("view_engine", "pug");

/**
 * Inicializar base de datos
 */
mongodb.connect(config.dbURL, function(err, con){
  if(err){
    console.log("Conexión a la base de datos fallida");
    process.exit(0);
  }

  console.log("Conexión a la base de datos correcta: " + config.dbURL);
  var db = con.db(config.dbName)

  common.informes_departamento = db.collection("informes_departamento");
  common.informes_docencia = db.collection("informes_docencia");
});

/**
 * Vistas
 */
app.get("/", (req, res) => {
  res.render("index", { title: "Inicio" });
});

require('./routers')(app);
```

En este archivo inicializamos todos los Frameworks que vamos a utilizar aunque muchos de ellos están incluidos directamente en Express, pero es necesaria su inicialización. Además también hemos creado una API RES que está conectada a este documento y a su vez está enlazada con el sistema de vistas (basado en PUG).

La conexión con el sistema de vistas (rutas) se hace en la siguiente línea:

```
require('./routers')(app);
```

A continuación incluimos el archivo (**routers.js**) que hace como API RES donde se definen las diferentes acciones (de momento todas son de tipo GET y POST):

```
module.exports = function(app) {
  var departamento = require('./controllers/departamento.js');
  var multer = require('multer');
  var storage = multer.diskStorage({
    destination: function (req, file, cb) {
      cb(null, 'public/pdf/')
    },
    filename: function (req, file, cb) {
      cb(null, file.originalname)
    }
  });
  var upload = multer({ storage: storage })

  app.get('/departamento', departamento.findAll);
  app.post('/departamento', departamento.findAll);

  app.get('/nuevo_informe', departamento.nuevoInforme);
  app.post('/nuevo_informe', departamento.anadirInforme);

  app.post('/subir_archivo/:dd/:mm/:aaaa', upload.single('imageupload'),function (
    req, res) {
    departamento.subirArchivo(req, res)
  });

  app.post('/departamento/:dd/:mm/:aaaa', departamento.delete);
}
```

Por ejemplo:

```
app.get('/nuevo_informe', departamento.nuevoInforme);
```

Cuando desde el front-end se hace una llamada de tipo GET a la ruta **"/nuevo_informe"** se llamaría automáticamente al método **departamento.nuevoInforme** que se encuentra dentro del fichero que se llama en la segunda línea de este fichero.

El fichero `./controllers/departamento.js` contiene todas las funciones que realizan consultas a la base de datos y es el siguiente:

```
const express = require('express');
const common = require('../common');
const path = require('path');
const router = express.Router();
const methodOverride = require('method-override');
router.use(methodOverride('_method'));

// Listar todos los informes de departamento:
exports.findAll = function(req, res){
  common.informes_departamento.find({}).toArray(function(err, inf) {
    if (err) {
      res.send(err);
    } else if (inf.length) {
      res.render('departamento', {
        'informes': inf,
      });
    } else {
      res.send('No documents found');
    }
  });
};

exports.anadirInforme = function(req, res){
  common.informes_departamento.findOne({"fecha": req.body.fecha}).then(
    function(informe) {
      if(informe != null)
        res.send('Ya hay un informe con esa fecha');
      else {
        if(req.body.nombre != null && req.body.dni != null){
          common.informes_departamento.insertOne({"fecha": req.body.fecha,
            "nombre": req.body.nombre, "dni_admin": req.body.dni, "URL"
            : ""}, function(err, inserted) {
            res.redirect('/departamento');
          });
        } else {
          res.send('Campos incorrectos');
        }
      }
    },
    function(err) { console.log(err) }
  );
};

exports.delete = function(req, res){
  deleteParam = req.params.dd + "/" + req.params.mm + "/" + req.params.aaaa;
  common.informes_departamento.deleteOne({"fecha": deleteParam}, function(err,
    results) {
    if (err){
      console.log("failed");
      throw err;
    }
    res.redirect('/departamento');
  });
};

exports.nuevoArchivo = function(req, res){
  console.log(req)
  fecha_nuevo = req.params.dd + "/" + req.params.mm + "/" + req.params.aaaa;
  common.informes_departamento.findOne({"fecha": fecha_nuevo}, function(err,
    results) {
    if (err){
      console.log("failed");
      throw err;
    }
  }

  res.render('subir_archivo', {title: 'Aadir informe departamento', fech:
    fecha_nuevo});
};
```

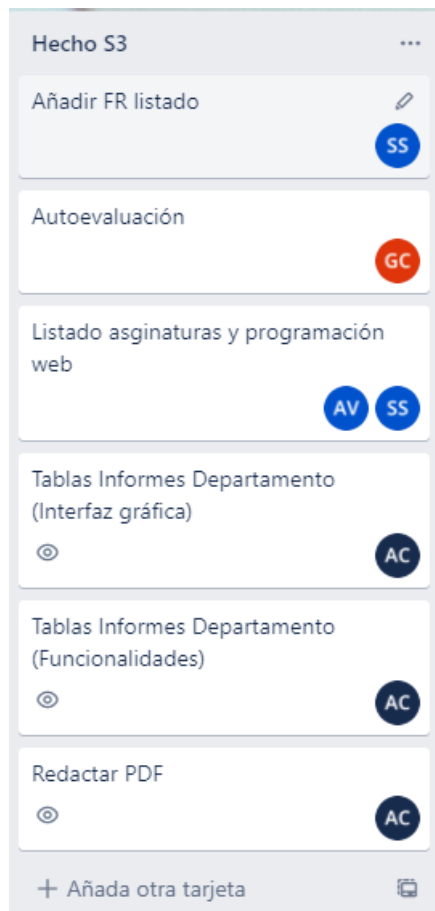
```

    });
  });
exports.subirArchivo = function(req, res){
  fecha_nuevo = req.params.dd + "/" + req.params.mm + "/" + req.params.aaaa;
  console.log(req.body)
  var url = "http://localhost:8080/pdf/" + req.file.filename;
  var newvalue = { $set: {URL: url} };
  common.informes_departamento.findOneAndUpdate({"fecha": fecha_nuevo}, newvalue,
    function(err, updated) {
      if(err) {
        console.log("failed");
        throw err;
      } else {
        res.redirect('/departamento');
      }
    }
  );
}

exports.nuevoInforme = function(req, res){
  res.render('nuevo-informe', {title: 'Añadir informe departamento'});
}

```

Todos los ficheros se encuentran en: https://github.com/acanojaen/ddsic2_7



9.7.1. Capturas del Sistema

Figura 21: Vista de Informes de Departamento

Figura 22: Añadiendo un fichero a un informe

Fecha	Nombre	DNI	Enlace
11/20/2020	Informe De Departamento N20	27233443R	Seleccionar
12/20/2020	Informe De Departamento N21	24242139R	Ver

Figura 23: Añadido el fichero

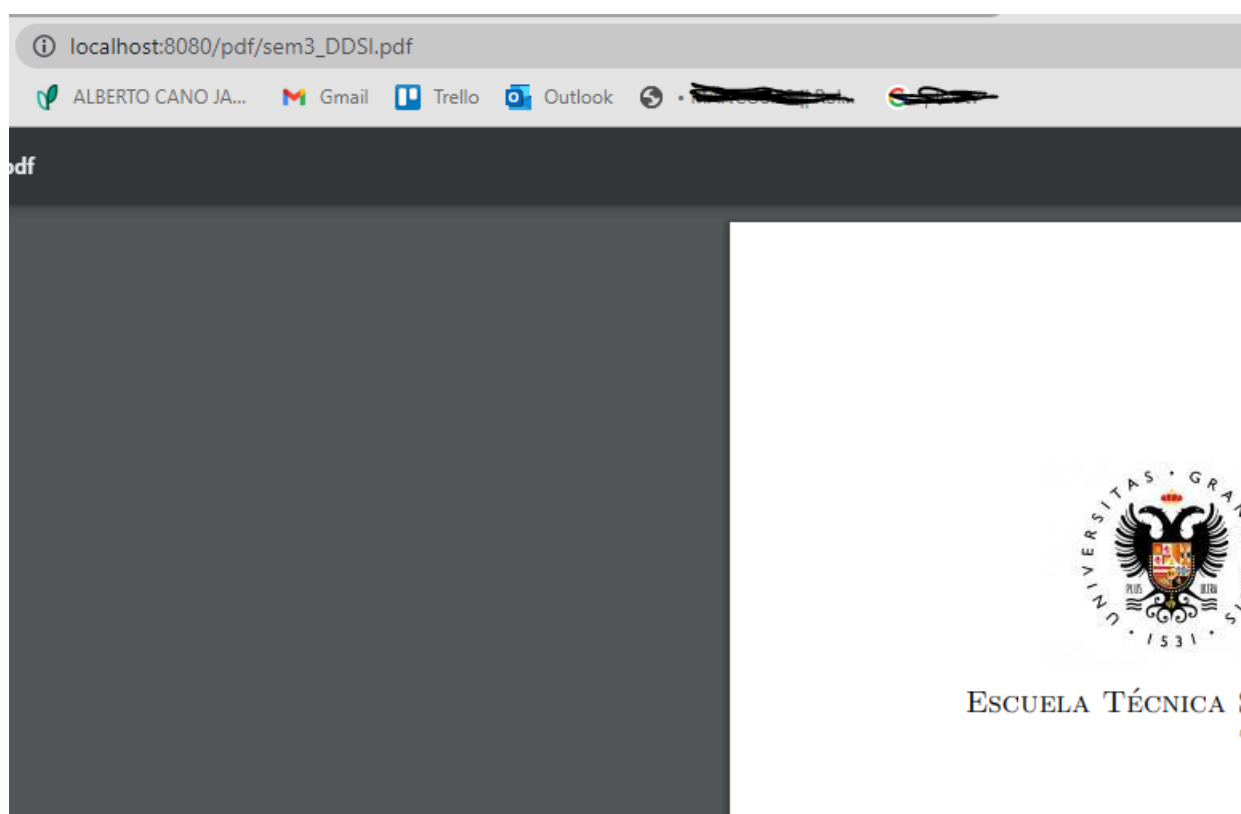


Figura 24: Fichero PDF desde el Servidor