

Práctica 3 (C1_3)

DISEÑO Y DESARROLLO DE SISTEMAS DE INFORMACIÓN



UNIVERSIDAD DE GRANADA

Autores:

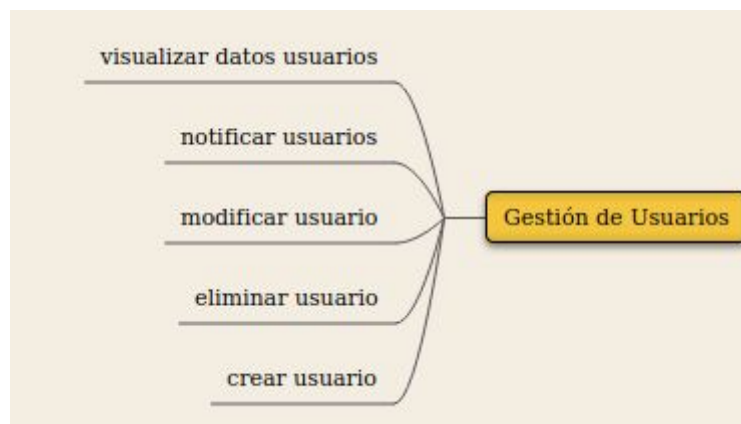
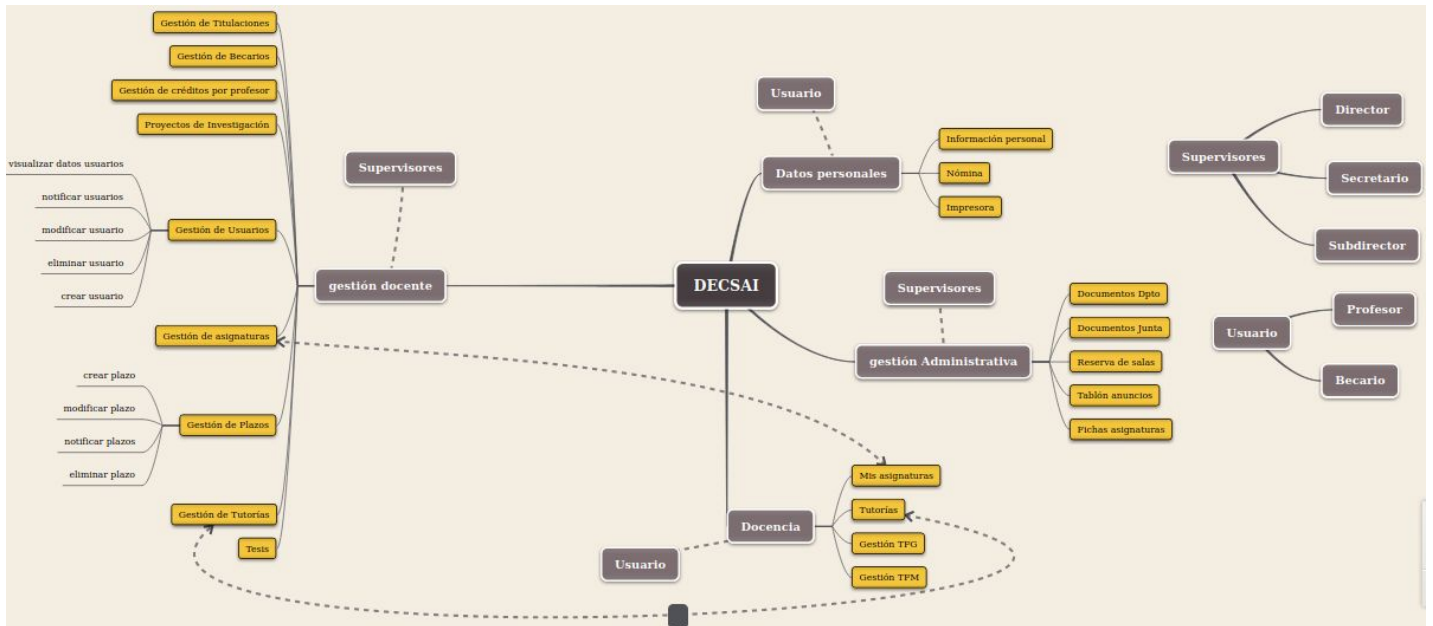
BENITEZ ALGUACIL, MIGUEL ANGEL
GALLARDO MOLINA, FRANCISCO JAVIER
GALLARDO RUIZ, ANGELA
JIMENEZ PIÑERO, SAMUEL
PAREJO BELLIDO, JOSE ANTONIO

Índice

1. Mapa mental del Sistema
2. Historias de Usuario
3. Listado de Requisitos y Restricciones
4. Diagrama de Flujos de Datos (DFD)
 - 4.1. Gestión de Usuarios
 - 4.2. Gestión de Plazos
5. Esquemas externos
 - 5.1. Gestión de Usuarios
 - 5.2. Gestión de Plazos
6. Esquemas Entidad/Relación
 - 6.1. Gestión de Usuarios
 - 6.2. Gestión de Plazos
7. Paso a tablas
 - 7.1. Gestión de Usuarios
 - 7.2. Gestión de Plazos
8. Refinamiento del diseño lógico
 - 8.1. Dependencias Funcionales
 - 8.2. Normalización.
9. Sentencias noSQL y SQL
 - 9.1. Gestión de Usuarios
 - 9.2. Gestión de Plazos
10. Diseño de la Interfaz
11. Uso de Mongodb
 - 11.1. Introducción
 - 11.2. Diferencias entre tablas SQL y colecciones en Mongodb
 - 11.3. Crear una cuenta
 - 11.4. Instalar el shell de Mongodb

- 11.5. Crear una base de datos
- 11.6. Dar acceso a programadores
- 11.7. Conectar la base de datos de MongoDB con nuestra aplicación
- 12. Ejemplo: ¿Cómo hacer un get de los datos?
- 13. Tablas asignaturas
- 14. Capturas de pantalla del sistema
- 15. Servidor web para alojar la página web
- 16. Trello (P1, P2, S2, S3 y P3)

1. Mapa mental del Sistema



2. Historias de Usuario

*Supervisor: director de departamento, subdirector de departamento y/o secretario

*Usuario: persona que imparte docencia.

- HU de Gestión de Usuario

HU	Historia de Usuario	Descripción	Prior.	R. F.	R. D.	R. S.
HU-1.1	COMO Supervisor QUIERO gestionar usuarios PARA añadir o eliminar usuarios del sistema y consultar sus datos.	Un supervisor puede gestionar los usuarios del sistema, puede darlos de alta y de baja en el sistema, y consultar sus datos personales.	Alta	RF-1.1 RF-1.2 RF-1.4	RDE-1.1 RDE-1.2 RDW-1.1 RDW-1.2 RDR-1.4 RDS-1.4	RS-1.1 RS-1.2 RS-1.3
HU-1.2	COMO Usuario QUIERO gestionar mis datos personales PARA añadirlos o actualizarlos en el sistema.	Un usuario puede visualizar y modificar sus datos personales.	Media	RF-1.3 RF-1.4	RDE-1.3 RDW-1.3 RDR-1.4 RDS-1.4	RS-1.4 RS-1.5 RS-1.6 RS-1.7 RS-1.8
HU-1.3	COMO Supervisor General QUIERO dar o quitar permisos de supervisor PARA modificar los permisos de un usuario del sistema.	Un supervisor general puede dar o quitar permisos de supervisor a un usuario para convertirlo en un supervisor normal.	Alta	RF-1.5	RDW-1.5 RDS-1.5	RS-1.9

- HU de Gestión de Plazos

HU	Historia de Usuario	Descripción	Prior.	R. F.	R. D.	R. S.
HU-2.1	COMO Supervisor General QUIERO poder gestionar una categoría PARA darla de alta en el sistema	Un supervisor general puede añadir una nueva categoría al sistema, dar acceso a otros supervisores a ella y eliminarla.	Alta	RF-2.1 RF-2.3 RF-2.4	RDE-2.1 RDW-2.1 RDR-2.3 RDS-2.3 RDW-2.4	RS-2.1 RS-2.2
HU-2.2	COMO Supervisor QUIERO poder gestionar las fechas de un plazo PARA actualizar el plazo en el sistema	Un supervisor puede visualizar y modificar las fechas de una categoría a la que tenga acceso.	Alta	RF-2.2 RF-2.3	RDE-2.2 RDW-2.2 RDR-2.3 RDS-2.3	RS-2.3 RS-2.4 RS-2.5
HU-2.3	COMO Usuario QUIERO recibir información de los plazos PARA tener constancia de un plazo que me concierne.	Cuando se modifica la fecha de un plazo, se notifica a los usuarios la información del plazo.	Alta	RF-2.5	RDR-2.5 RDS-2.5	-

3. Listado de Requisitos y Restricciones

RF-1. Gestión de usuarios: Se permitirá dar de alta/baja a un usuario en el sistema, así como consultar y/o modificar cualquiera de sus datos y listarlos.

RF-1.1: Alta de usuario. Se registrará un nuevo usuario en el sistema, usando su DNI o Correo electrónico.

Entrada: Agente externo: supervisor. Acción: Crear usuario. Requisitos de datos de entrada: RDE-1.1.

BD: Requisitos de datos de escritura RDW-1.1.

Salida: Agente externo: supervisor. Acción: Confirmación resultado. Requisitos de datos de salida: ninguno.

RDE-1.1: Datos de entrada de alta de contacto

D.N.I./Pasaporte: Cadena de caracteres (9)

Correo: Cadena de caracteres (40)

Fecha Entrada Departamento: Dato tipo Date(1)

RDW-1.1: Datos almacenados de contacto.

Los mismos datos que RDE-1.1.

RF-1.2 Baja de usuario. Se eliminará del sistema toda la información relativa a un usuario.

Entrada: Agente externo: supervisor. Acción: Eliminar usuario. Requisito de datos de entrada: RDE-1.2.

BD: Requisito de datos de escritura RDW-1.2.

Salida: Agente externo: supervisor. Acción: Confirmación resultado. Requisitos de datos de salida: ninguno.

RDE-1.2: Datos de entrada de baja de contacto

D.N.I./Pasaporte: Cadena de caracteres (9)

RDW-1.2: Datos almacenados de contacto.

Los mismos datos que RDW-1.1.

RF-1.3 Modificar datos de usuario. Se podrán alterar los datos almacenados de un usuario.

Entrada: Agente externo: usuario. Acción: Modificar datos usuario. Requisitos de datos de entrada: RDE-1.3.

BD: Requisitos de datos de escritura RDW-1.3.

Salida: Agente externo: supervisor y usuario. Acción: Confirmación datos. Requisitos de datos de salida: ninguno.

RDE-1.3: Datos de entrada de modificación de contacto.

D.N.I./Pasaporte: Cadena de caracteres (9)

Nombre: Cadena de caracteres (20)

Apellidos: Cadena de caracteres (40)

Teléfono: Cadena de caracteres (20)
Nombre Abreviado: Cadena de caracteres (15)
Correo: Cadena de caracteres (40)
Login: Cadena de caracteres (20)
Password: Cadena de caracteres (40)
Foto: Archivo de imagen (1)
Tipo de Usuario: Dato tipo entero (1)
Despacho: Dato tipo entero
Teléfono Despacho: Cadena de caracteres (9)
Teléfono Personal: Cadena de caracteres (9)
Imparte Docencia: Dato tipo bool
Miembro Actual: Dato tipo bool
Miembro Total: Dato tipo bool
Miembro Consejo: Dato tipo bool

RDW-1.3: Datos almacenados de contacto.

Los mismo datos que RDW-1.1.

RF-1.4 Consultar datos de usuario. Se mostrarán todos los datos relativos a un determinado usuario.

Entrada: Agente externo:supervisor y usuario. Acción: consultar datos. Requisitos de datos de entrada: ninguno.

BD: Requisitos de datos de lectura: RDR-1.4.

Salida: Agente externo: supervisor y usuario. Acción: confirmar datos. Requisitos de datos de salida: RDS-1.4.

RDR-1.4: Datos de lectura de consulta de usuario

D.N.I./Pasaporte: Cadena de caracteres (9)
Nombre: Cadena de caracteres (20)
Apellidos: Cadena de caracteres (40)
Teléfono: Cadena de caracteres (20)
Nombre Abreviado: Cadena de caracteres (15)
Correo: Cadena de caracteres (40)
Login: Cadena de caracteres (20)
Password: Cadena de caracteres (40)
Foto: Archivo de imagen (1)
Tipo de Usuario: Dato tipo entero (1)
Despacho: Dato tipo entero
Teléfono Despacho: Cadena de caracteres (9)
Teléfono Personal: Cadena de caracteres (9)
Imparte Docencia: Dato tipo bool
Miembro Actual: Dato tipo bool
Miembro Total: Dato tipo bool
Miembro Consejo: Dato tipo bool
Fecha Entrada Departamento: Dato tipo Date(1).

RDS-1.4: Datos de salida de consulta de usuario.

Los mismos datos que RDR-1.4.

RF-1.5 Dar o quitar permisos de supervisor. Se podrán dar o quitar permisos de supervisor a los usuarios del sistema.

Entrada: Agente externo: supervisor Acción: se cambiarán los permisos de usuarios del sistema. Requisitos de datos de entrada: RDE-1.5.

BD: Requisitos de datos de lectura RDW-1.5.

Salida: Agente externo: supervisor Acción: confirmación resultado. Requisitos de datos de salida: RDS-1.5.

RDW-1.5: Datos almacenados de contacto.

Tipo de Usuario: Dato tipo entero (1).

RDS-1.5: Datos almacenados de contacto.

Los mismos datos que RDE-1.1.

Restricciones semánticas relacionadas con RF-1.1

RS-1.1: El campo *DNI* o el campo *Correo* no pueden ser vacíos.

RF: RF-1.1

RD(s): RDE-1.1

Descripción: "Al menos un campo (DNI o correo) debe tener información referente al usuario, si ambos están vacíos, no se da de alta al usuario."

Restricciones semánticas relacionadas con RF-1.2

RS-1.2: El formato del D.N.I./Pasaporte debe ser el correcto.

RF: RF-1.2

RD(s): RDE-1.2

Descripción: "Si el formato del D.N.I./Pasaporte no es correcto, no se elimina el usuario y se pide que se corrija."

RS-1.3: El D.N.I./Pasaporte corresponde a un único usuario existente.

RF: RF-1.2

RD(s): RDE-1.2

Descripción: "Si no hay ningún usuario asociado a ese D.N.I./Pasaporte, no se elimina ningún usuario y se devuelve un error"

Restricciones semánticas relacionadas con RF-1.3

RS-1.4: Los campos *DNI*, *Nombre*, *Apellidos*, *Correo*, *Login* y *Password* no pueden ser vacíos.

RF: RF-1.3

RD(s): RDE-1.3

Descripción: "Si *DNI*, *Nombre*, *Apellidos*, *Correo*, *Login* o *Password* son vacíos, no se modifica el usuario y se pide que se corrija."

RS-1.5: El formato de DNI, *Teléfono Despacho* y *Teléfono Personal* debe ser el correcto.

RF: RF-1.3

RD(s): RDE-1.3

Descripción: “Si el formato del DNI o del teléfono (en el caso de que se haya introducido) no son correctos, no se da de alta al usuario y se pide que se corrija.”

RS-1.6: *Foto* debe ser un archivo de imagen JPEG o PNG.

RF: RF-1.3

RD(s): RDE-1.3

Descripción: “Si el tipo de archivo de *Foto* no es JPEG o PNG, no se cargará y dará mensaje de error.”

RS-1.7: *Confirme Password* debe ser idéntico a *Password*.

RF: RF-1.3

RD(s): RDE-1.3

Descripción: “Si el contenido de *Password* y *Confirme Password* no coinciden, no se da de alta al usuario y se pide que se corrija.”

RS-1.8: Los campos *DNI*, *Correo* y *Login* deben ser únicos en el sistema.

RF: RF-1.3

RD(s): RDE-1.3

Descripción: “Si el *DNI*, *Correo* o el *Login* introducidos ya aparecen en el sistema vinculados a otro usuario, no se da de alta al usuario y se pide que se corrija.”

Restricciones semánticas relacionadas con RF-1.5

RS-1.9: El campo *Tipo de Usuario* debe ser correcto

RF: RF-1.5

RD(s): RDE-1.5

Descripción: “Si el contenido de *Tipo de Usuario* no es correcto, no se modifica al usuario y se pide que se corrija.”

RF-2. Gestión de plazos: Se permitirá crear un nuevo plazo, así como eliminar, consultar y/o modificar un plazo ya existente.

RF-2.1: Añadir nuevo plazo. Se registrará un nuevo plazo en el sistema, este será, a ojos del sistema, una nueva categoría.

Entrada: Agente externo: supervisor general. Acción: Crear plazo. Requisitos de datos de entrada: RDE-2.1.

BD: Requisitos de datos de escritura RDW-2.1.

Salida: Agente externo: supervisor general. Acción: Confirmación resultado. Requisitos de datos de salida: ninguno.

RDE-2.1: Datos de entrada de creación de categoría (Nueva categoría)

Nombre: Cadena de caracteres (30)

Descripción: Cadena de caracteres (140)

Acceso: Lista de cadena de caracteres (9) ← DNI de los supervisores

RDW-2.1: Datos almacenados de categoría.

Los mismos datos que RDE-2.1.

RF-2.2: Modificar un plazo. Se podrá alterar la fecha de una categoría.

Entrada: Agente externo: supervisor con acceso. Acción: Modificar fecha plazo. Requisitos de datos de entrada: RDE-2.2.

BD: Requisitos de datos de escritura RDW-2.2.

Salida: Agente externo: supervisor. Acción: Confirmación datos. Requisitos de datos de salida: ninguno.

RDE-2.2: Datos de entrada de modificación de plazo

Fecha de inicio: Dato tipo date

Fecha de fin: Dato tipo date

RDW-2.2: Datos almacenados de plazo.

Los mismos datos que RDE-2.2.

RF-2.3: Visualizar un plazo. Se podrán visualizar los datos de un plazo.

Entrada: Agente externo: supervisor. Acción: visualizar plazos. Requisitos de datos de entrada: ninguno.

BD: Requisitos de datos de lectura: RDR-2.3.

Salida: Agente externo: usuario. Acción: confirmar datos. Requisitos de datos de salida: RDS-2.3.

RDR-2.3: Visualización del plazo

Nombre: Cadena de caracteres (30)

Descripción: Cadena de caracteres (140)

Fecha de inicio: Dato tipo date

Fecha de fin: Dato tipo date

RDS-2.3: Confirmación del plazo.

Los mismos datos que RDR-2.3.

RF-2.4: Eliminar un plazo. Se eliminará del sistema la información relativa a una categoría.

Entrada: Agente externo: supervisor general. Acción: Eliminar plazo. Requisito de datos de entrada: ninguno.

BD: Requisitos de datos de escritura RDW-2.4.

Salida: Agente externo: supervisor. Acción: confirmación resultado. Requisitos de datos de salida: ninguno.

RDW-2.4: Datos almacenados de plazo.
Los mismos datos que RDR-2.3.

RF-2.5: Notificar un plazo. Se notificará por correo la información referente a un plazo a todos los usuarios.

Entrada: Agente externo: sistema. Acción: notificar plazos. Requisitos de datos de entrada: ninguno.

BD: Requisitos de datos de lectura: RDR-2.5.

Salida: Agente externo: usuario. Acción: ninguna. Requisitos de datos de salida: RDS-2.5.

RDR-2.5: Visualización del plazo
Los mismos datos que RDR-2.3

RDS-2.5: Notificación del plazo.
Los mismos datos que RDR-2.5.

Restricciones semánticas relacionadas con RF-2.1

RS-2.1: Los campos *Nombre* y *Descripción* no pueden ser vacíos.

RF: RF-2.1

RD(s): RDE-2.1

Descripción: “Si *Nombre* o *Descripción* son vacíos, no se crea la categoría y se pide que se corrija.”

RS-2.2: No pueden existir dos categorías con el mismo nombre.

RF: RF-2.1

RD(s): RDE-2.1

Descripción: “Si el campo *Nombre* ya pertenece a otra categoría ya creada, no se crea la categoría y se pide que se corrija”

Restricciones semánticas relacionadas con RF-2.2

RS-2.3: Los campos *Fecha Inicio* y *Fecha Fin* no pueden ser vacíos.

RF: RF-2.2

RD(s): RDE-2.2

Descripción: “Si *Fecha Inicio* o *Fecha Fin* son vacíos, no se modifica el plazo y se pide que se corrija.”

RS-2.4: *Fecha Fin* no puede ser anterior a *Fecha Inicio*.

RF: RF-2.2

RD(s): RDE-2.2

Descripción: “Si *Fecha Fin* < *Fecha Inicio*, no se modifica el plazo y se pide que se corrija.”

RS-2.5: No puede existir una categoría con el mismo *Nombre*.

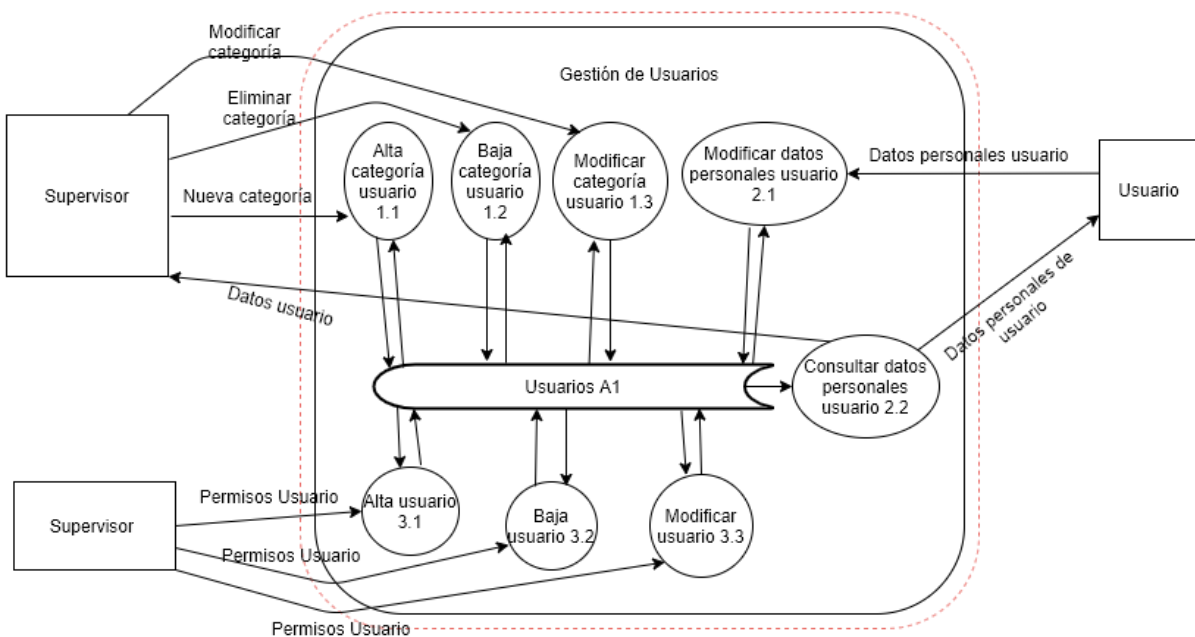
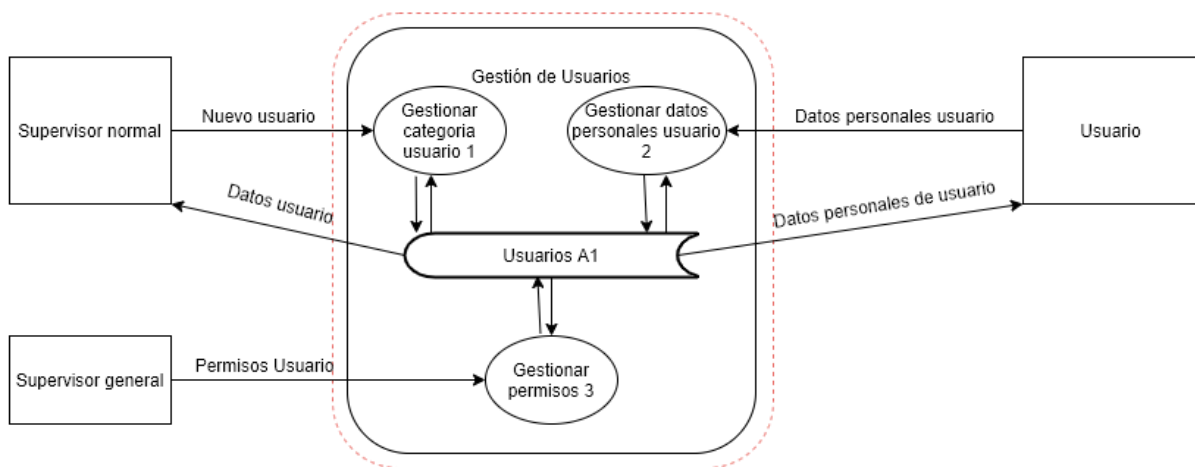
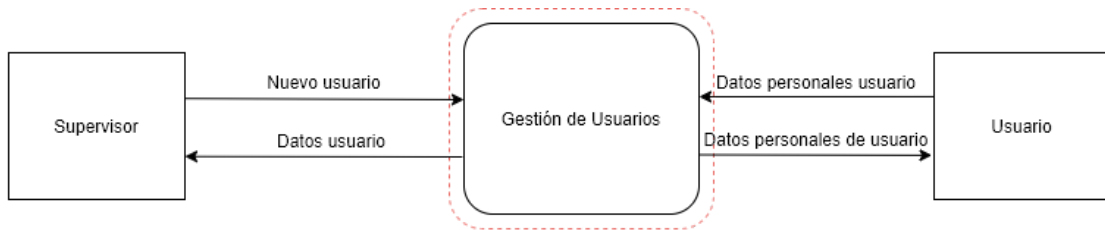
RF: RF-2.2

RD(s): RDE-2.2

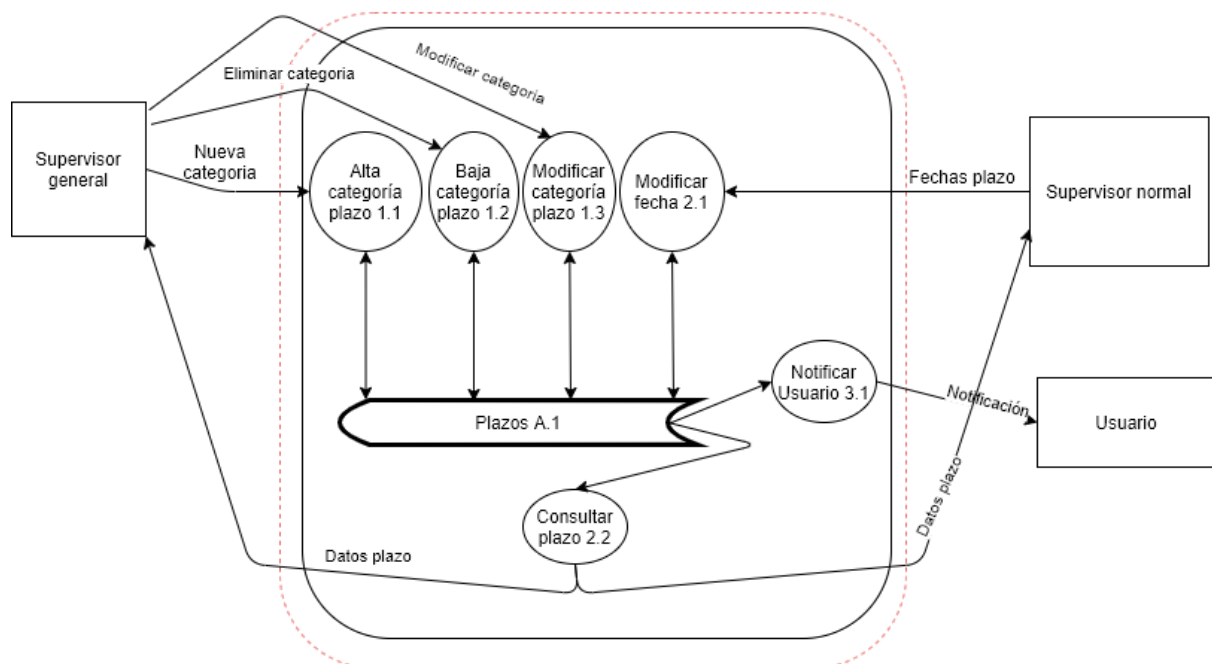
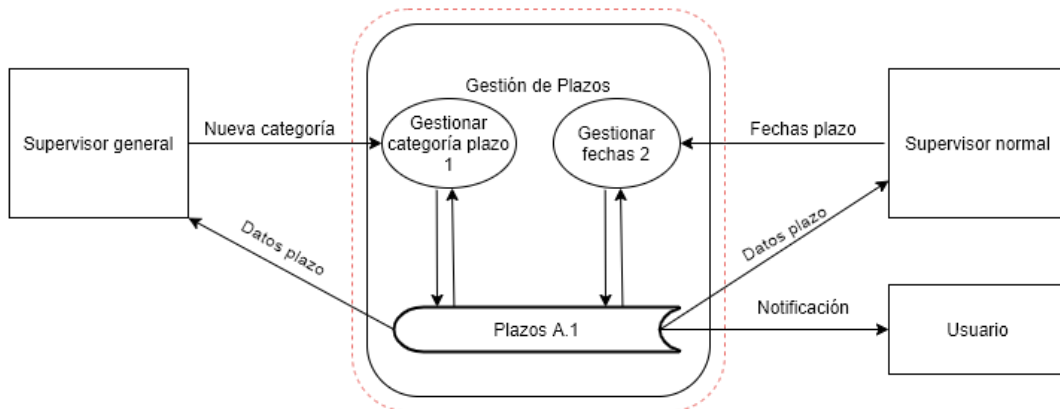
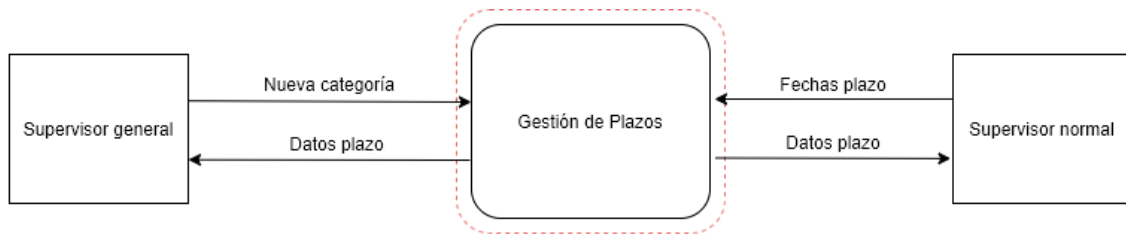
Descripción: “Si el campo *Nombre* coincide con otro plazo ya creado, no se modifica el plazo y se pide que se corrija.

4. Diagramas de Flujos de Datos (DFD)

4.1. Gestión de Usuarios



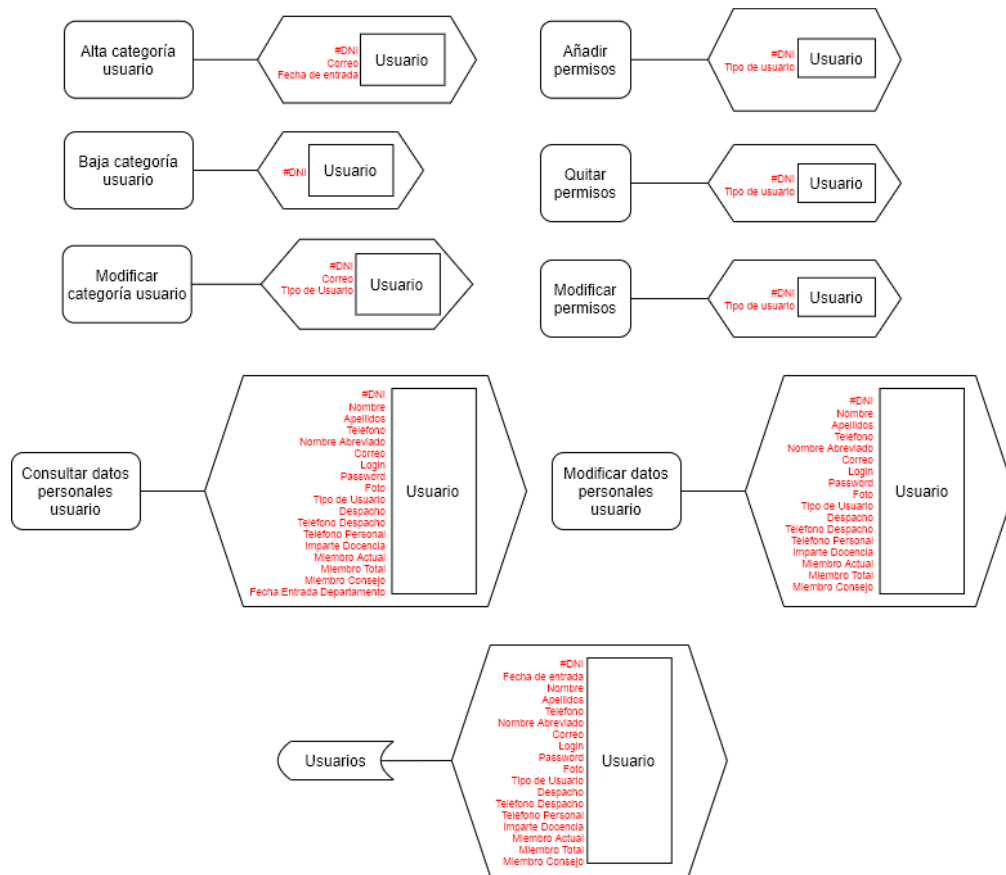
4.2. Gestión de Plazos



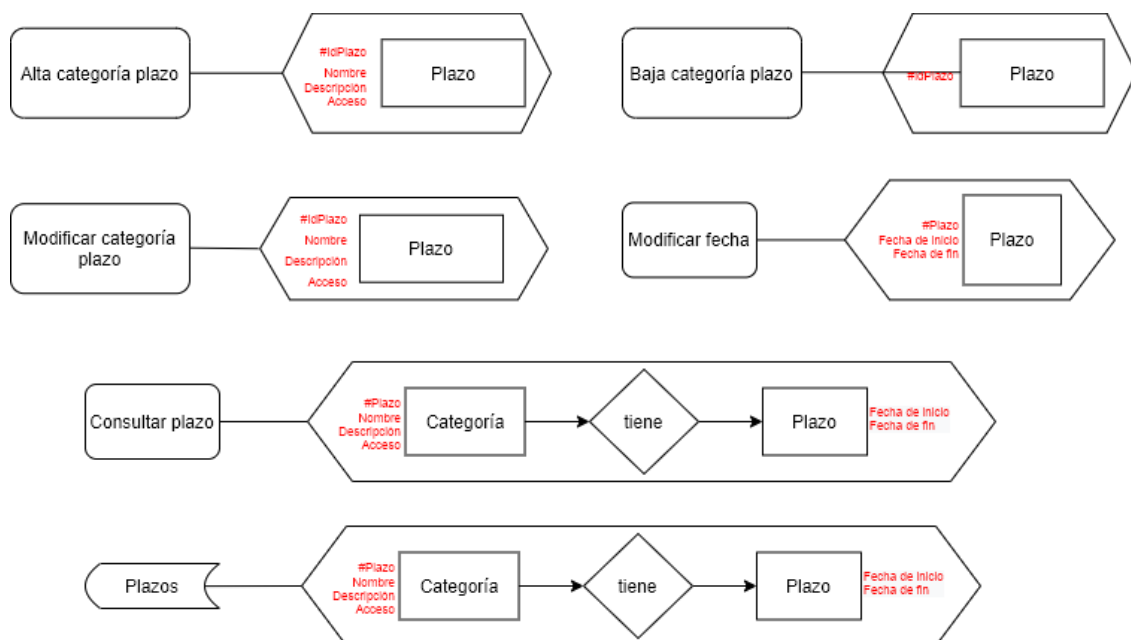
<https://app.diagrams.net/#G1KR36slw8poXdyE8vOA6QILvcQ25RjDVP>

5. Esquemas externos

5.1. Gestión de Usuarios

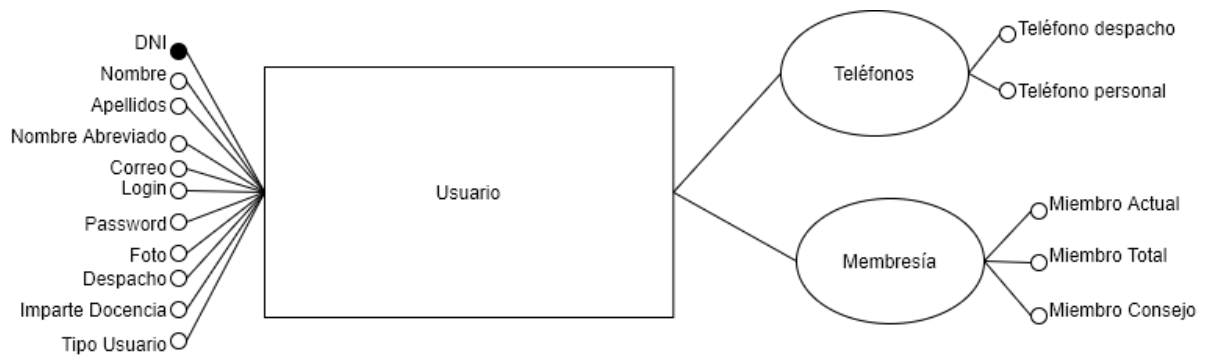


5.2. Gestión de Plazos



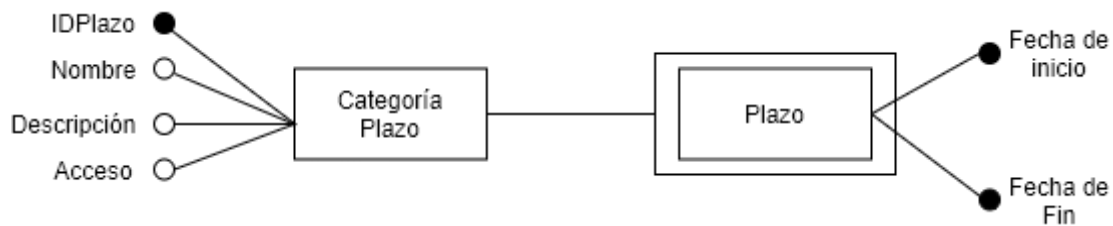
6. Esquemas Entidad/Relación

6.1. Gestión de Usuarios



Como en esta funcionalidad lo que se hace es gestión y no se almacenan datos del administrador ni del usuario, solo nos queda la entidad usuario (acordado con el profesor)

6.2. Gestión de Plazos



7. Paso a tablas

7.1. Gestión de usuarios

- Paso a Tablas con atributos compuestos

1.Usuario(**DNI**, Nombre, Apellidos, Nombre Abreviado, Correo, Login, Password, Foto, Despacho, Imparte Docencia, Tipo Usuario, Fecha Entrada Departamento, Teléfonos, Membresía)

- Paso a Tablas con atributos simples

1.Usuario(**DNI**, Nombre, Apellidos, Nombre Abreviado, **Login**, Foto, Despacho, Imparte Docencia, Tipo Usuario, Fecha Entrada Departamento, Teléfono Despacho, Teléfono Personal, Miembro Actual, Miembro Total, Miembro Consejo)

Cuenta(**Login**, correo, password)

La tabla cumple FNBC porque todos los determinantes son CC

7.2. Gestión de plazos

1.Categoría plazo(**IDPlazo**, Nombre, Descripción)



2.Fecha plazo(**IDPlazo**, **Fecha de inicio**, **Fecha de fin**)

CP1

8. Refinamiento del diseño lógico

8.1. Dependencias funcionales

- 8.1.1 Gestión de usuarios

DNI → Nombre, Apellidos, Nombre Abreviado, Foto, Despacho, Teléfono Despacho, Teléfono Personal, Imparte Docencia, Tipo Usuario, Fecha Entrada Departamento, Miembro Actual, Miembro Total, Miembro Consejo, Login.

Login → Correo, Password.

- 8.1.2 Gestión de plazos

IDPlazo → Nombre, Descripción, Acceso

8.2. Normalización

Las tablas ya se encuentran normalizadas.

9. Sentencias noSQL y SQL

*Para noSQL hemos decidido usar mongodb. Además añadimos su contrapartida en SQL por aun no tener muy claro que usar.

9.1. Gestión de Usuarios

Para crear la base de datos

```
use GestionUsuarios;
```

Para conectarnos a la base de datos

```
conn = new Mongo();
```

```
db = conn.getDB("GestionUsuarios");
```

Para crear una colección con sus distintas restricciones

```
db.createCollection("Usuarios",
    {
        autoIndexId:      false,
        Nombre:           <string>,
        Apellidos:        <string>,
        Nombre_Abreviado: <string>,
        validator:        { Correo: { $regex: /@flares\.$/ } },
        Login:            <string>,
        Password:         <string>,
        Foto:              <document>,
        Despacho:         <number>,
        Imparte_Docencia: <boolean>,
        Tipo_Usuario:     <number>,
        Teléfono_Despacho: <string>,
        Miembro_Actual:   <boolean>,
        Miembro_Total:    <boolean>,
        Miembro_Consejo:  <boolean>
    }
);
```

Para añadir lo hacemos desde shell

Shell:

*Pongo Many y no one para insertar varios
db.Usuario.insertOne()

```
{
    _id      :      12345678A,
    Nombre   :      "Angela",
```

```

    Apellidos      :      " Gallardo Ruiz",
    Nombre_Abreviado :      "Angy ",
    Correo         :      "angela@correo.com",
    Login          :      "AngelaGR",
    Password       :      "cumpleangela",
    Foto           :
    Despacho       :      "2D",
    Imparte_Docencia :      true,
    Tipo_Usuario   :      123456789,
    Teléfono_Despacho :      123456789,
    Miembro_Actual :      true,
    Miembro_Total  :      false,
    Miembro_Consejo :      false
}

```

Para crear la tabla con sus restricciones en SQL

```

CREATE TABLE usuarios(
    DNI varchar2(9) PRIMARY KEY,
    Nombre varchar2(20),
    Apellidos varchar2(40),
    Nombre_Abreviado varchar2(40),
    Correo varchar2(40) CONSTRAINT HA_DE_TENER_ARROBA CHECK( Correo
    LIKE '%@%' ) NOT NULL,
    Login varchar2(20),
    Password varchar2(40) DEFAULT("contraseña"),
    Foto varbinary(max),
    Despacho int,
    Imparte_Docencia bit,
    Tipo_Usuario int,
    Teléfono_Despacho varchar(9),
    Teléfono_Personal varchar(9),
    Miembro_Actual bit,
    Miembro_Total bit,
    Miembro_Consejo bit,
    Fecha_Entrada_Departamento date NOT NULL
);

```

9.2. Gestión de Plazos

Para crear la base de datos

use GestionPlazos

Para conectarnos a la base de datos

```
conn = new Mongo();
```

```
db = conn.getDB("GestionPlazos");
```

Para crear una colección con sus distintas restricciones

```
db.createCollection("Plazos",  
    {  
        Nombre:      <string>,  
        Descripción  <string>,  
        Fecha_ini    <date>,  
        Fecha_fin     <date>  
    });
```

Crear colección Plazos sin shell

```
    "roles": [  
    {  
        "name": "Owner",  
        "apply_when": {},  
        "insert": true,  
        "delete": true,  
        "search": true,  
        "write": true,  
        "fields": {  
            "Descripción": {},  
            "Fecha Fin": {},  
            "Fecha Inicio": {},  
            "Nombre Plazo": {}  
        },  
        "additional_fields": {}  
    },  
    {  
        "name": "User",  
        "apply_when": {},  
        "insert": false,  
        "delete": false,  
        "search": true,  
        "read": true,  
        "fields": {  
            "Descripción": {},  
            "Fecha Fin": {},  
            "Fecha Inicio": {},  
            "Nombre Plazo": {}  
        },  
        "additional_fields": {}  
    }  
    ]
```

```
],  
"filters": [],  
"schema": {}  
}
```


Insertar plazos

```
db.Plazo.insertOne(  
    {  
        Nombre      :    "Entrega actas"  
        Descripción  :    "Día límite para la entrega de las actas"  
        Fecha inicio :    "1/02/2021"  
        Fecha fin    :    "13/02/2021"  
    }  
)
```

Para crear la tabla con sus restricciones en SQL

```
CREATE TABLE "PLAZOS" (  
    "NOMBRE PLAZO" VARCHAR2(20) NOT NULL ENABLE,  
    "DESCRIPCION" VARCHAR2(50),  
    "FECHA_INICIO" DATE,  
    "FECHA_FIN" DATE,  
    CONSTRAINT "PLAZOS_PK" PRIMARY KEY ("NOMBRE PLAZO")  
    USING INDEX ENABLE,  
    CONSTRAINT "FECHAS_RS" CHECK ( "FECHA_INICIO" <=FECHA_FIN) ENABLE  
)
```


10. Diseño de la Interfaz



UNIVERSIDAD
DE GRANADA

INICIO

MENÚ

LOGIN
LOGOUT

GESTIÓN DE USUARIOS

Filtrar por DNI


Filtrar por Email

ELIMINAR
SELECCIONADOS

AÑADIR USUARIO

SELECCIONAR	DNI	EMAIL	PERMISOS	FECHA DE ALTA	
<input type="checkbox"/>	DNI	Email	Supervisor	Fecha de alta en el sistema	MODIFICAR
<input type="checkbox"/>	DNI	Email	Usuario	Fecha de alta en el sistema	MODIFICAR
<input type="checkbox"/>	DNI	Email	Usuario	Fecha de alta en el sistema	MODIFICAR
<input type="checkbox"/>	DNI	Email	Supervisor	Fecha de alta en el sistema	MODIFICAR
<input type="checkbox"/>	DNI	Email	Usuario	Fecha de alta en el sistema	MODIFICAR
<input type="checkbox"/>	DNI	Email	Usuario	Fecha de alta en el sistema	MODIFICAR


Vista Gestión de Usuarios



UNIVERSIDAD
DE GRANADA

INICIO

MENÚ

LOGIN
LOGOUT

GESTIÓN DE PLAZOS

Fecha de inicio

Select

Fecha de fin

Select

ELIMINAR
SELECCIONADOS

AÑADIR PLAZO

SELECCIONAR	NOMBRE PLAZO	DESCRIPCIÓN	FECHA DE INICIO	FECHA DE INICIO
<input type="checkbox"/>	Nombre	Descripción del plazo	FechaIni	FechaFin
<input type="checkbox"/>	Nombre	Descripción del plazo	FechaIni	FechaFin
<input type="checkbox"/>	Nombre	Descripción del plazo	FechaIni	FechaFin
<input type="checkbox"/>	Nombre	Descripción del plazo	FechaIni	FechaFin
<input type="checkbox"/>	Nombre	Descripción del plazo	FechaIni	FechaFin
<input type="checkbox"/>	Nombre	Descripción del plazo	FechaIni	FechaFin

Vista Gestión de Plazos

Modificar Datos Personales



DNI	EMAIL			
<input type="text"/>	<input type="text"/>			
NOMBRE	APELLIDOS			
<input type="text"/>	<input type="text"/>			
NOMBRE ABREVIADO	LOGIN	CONTRASEÑA		
<input type="text"/>	<input type="text"/>	<input type="password"/>		
TELÉFONO DESPACHO	TELÉFONO PERSONAL			
<input type="text"/>	<input type="text"/>			
DESPACHO	TIPO USUARIO	FECHA ENTRADA DPTO.	M. ACTUAL	M. CONSEJO
<input type="text" value="2.2"/>	<input type="text" value="Profesor"/>	<input type="text" value="11/04/2020"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
			M.TOTAL	IMPORTE
			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="button" value="GUARDAR"/>				

Vista de Modificar Datos Personales por supervisor

Modificar Mis Datos Personales



DNI	EMAIL			
<input type="text"/>	<input type="text"/>			
NOMBRE	APELLIDOS			
<input type="text"/>	<input type="text"/>			
NOMBRE ABREVIADO	LOGIN	CONTRASEÑA		
<input type="text"/>	<input type="text"/>	<input type="password"/>		
TELÉFONO DESPACHO	TELÉFONO PERSONAL			
<input type="text"/>	<input type="text"/>			
DESPACHO	TIPO USUARIO	FECHA ENTRADA DPTO.	M. ACTUAL	M. CONSEJO
<input type="text"/>	<input type="text" value="Profesor"/>	<input type="text" value="01/09/2015"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
			M.TOTAL	IMPORTE
			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="button" value="GUARDAR"/>				

Vista de Modificar Datos Personales por Usuario

Dar de alta usuario

SIGN UP

Vista de Dar de alta a Usuario

Añadir Plazo



Supervisores con acceso

Supervisor 1

Supervisor 2

Supervisor 3

AÑADIR

Vista de Añadir Plazo

Modificar Plazo

Nombre del plazo

Descripción

Supervisores con acceso

Fecha Inicio

Fecha Fin

Supervisor 1
Supervisor 2
Supervisor 3

11/03/2020

11/20/2020

GUARDAR

Vista de Modificar Plazo

11. Uso de MongoDB

11.1. Introducción

Nosotros hemos decidido que nuestra base de datos sea organizada como un NoSQL. La herramienta que vamos a usar es MongoDB, también conocida como Mongo. Es básicamente una base de datos de documentos similares a JSON cuya estructura es dinámica.

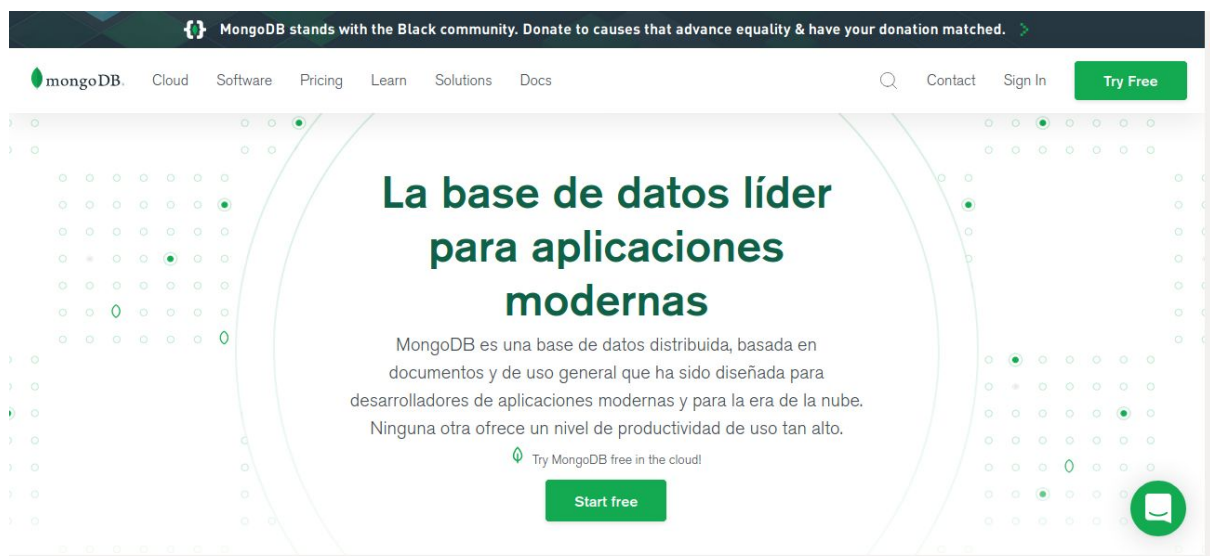
11.2. Diferencias entre tablas SQL y colecciones en MongoDB

Las bases de datos de MongoDB son no relacionales, por lo que no poseen relaciones. Esto hace que puedas modificar los datos en cualquier momento si lo necesitas. Con esto se consigue que las consultas sean más rápidas que en SQL. Esto tiene de inconveniente que si queremos modificar un dato que está en varias colecciones de tablas, habremos de hacerlo de forma manual.

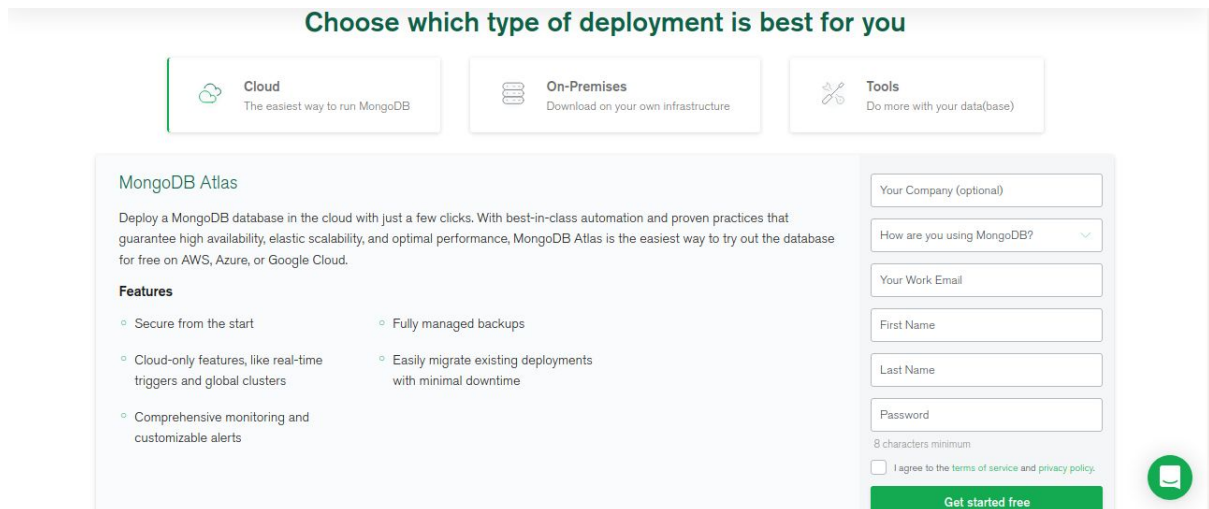
Por otro lado, los datos SQL se manejan con un esquema bien definido y rígido, por lo que todos los datos de una misma tabla tienen la misma estructura, cantidad de campos y tipo de dato. Eso conlleva que tenga una estructura compleja mejor organizada pero es mucho más tedioso de modificar que una tabla NoSQL de Mongo. Y los datos SQL son relacionales, así que existen relaciones entre tablas con claves externas.

En resumen, los datos de una tabla SQL estarían estructurados y guardados en tablas con un esquema estático mientras que en una colección NoSQL no se estructuran en tablas y presenta un esquema dinámico.

11.3. Crear una cuenta



Esta es la página principal. Para crear la cuenta, se selecciona *Try Free* en la esquina superior derecha.



Se elige una de las tres opciones que indican arriba y luego se introducen los datos personales para inscribirse. Nosotros escogemos la opción Cloud con MongoDB Atlas.

Y con esto ya estaría la cuenta creada. Ahora debemos instalar la shell de mongo. Para ello necesitamos Ubuntu, concretamente la versión 20.04.

11.4. Instalar el shell

Instalamos curl que es una herramienta de línea de comandos utilizada para transferir datos. Seguidamente usamos el siguiente comando que lee cualquier dato almacenado en la URL e imprime el contenido del archivo de claves GPG y luego lo canaliza a la lista de claves de confianza con el sudo apt-key add. Devuelve OK.

```
$ curl -fsSL https://www.mongodb.org/static/pgp/server-4.4.asc | sudo apt-key add -
```

```
angela@angela:~$ sudo apt install curl
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes NUEVOS:
  curl
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 38 no actualizados.
Se necesita descargar 161 kB de archivos.
Se utilizarán 411 kB de espacio de disco adicional después de esta operación.
Des:1 http://es.archive.ubuntu.com/ubuntu focal-updates/main amd64 curl amd64 7.68.0-1ubuntu2.2 [161
kB]
Descargados 161 kB en 0s (482 kB/s)
Seleccionando el paquete curl previamente no seleccionado.
(Leyendo la base de datos ... 207208 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../curl_7.68.0-1ubuntu2.2_amd64.deb ...
Desempaquetando curl (7.68.0-1ubuntu2.2) ...
Configurando curl (7.68.0-1ubuntu2.2) ...
Procesando disparadores para man-db (2.9.1-1) ...
angela@angela:~$ curl -fsSL https://www.mongodb.org/static/pgp/server-4.4.asc | sudo apt-key add -
OK
```

Seguidamente comprobamos que la clave se añadió bien en nuestras claves de confianza. Vemos que si es en la primera de ellas donde en la parte de 'uid' pone MongoDB

\$ apt-key list

```
angela@angela:~$ apt-key list
/etc/apt/trusted.gpg
-----
pub   rsa4096 2016-04-12 [SC]
      EB4C 1BFD 4F04 2F6D DDCC EC91 7721 F63B D38B 4796
uid   [desconocida] Google Inc. (Linux Packages Signing Authority) <linux-packages-keymaster@google.com>
sub   rsa4096 2019-07-22 [S] [caduca: 2022-07-21]

pub   rsa4096 2019-05-28 [SC] [caduca: 2024-05-26]
      2069 1EEC 3521 6C63 CAF6 6CE1 6564 08E3 90CF B1F5
uid   [desconocida] MongoDB 4.4 Release Signing Key <packaging@mongodb.com>

/etc/apt/trusted.gpg.d/ubuntu-keyring-2012-archive.gpg
-----
pub   rsa4096 2012-05-11 [SC]
      790B C727 7767 219C 42C8 6F93 3B4F E6AC C0B2 1F32
uid   [desconocida] Ubuntu Archive Automatic Signing Key (2012) <ftpmaster@ubuntu.com>

/etc/apt/trusted.gpg.d/ubuntu-keyring-2012-cdimage.gpg
-----
pub   rsa4096 2012-05-11 [SC]
      8439 38DF 228D 22F7 B374 2BC0 D94A A3F0 EFE2 1092
uid   [desconocida] Ubuntu CD Image Automatic Signing Key (2012) <cdimage@ubuntu.com>

/etc/apt/trusted.gpg.d/ubuntu-keyring-2018-archive.gpg
-----
pub   rsa4096 2018-09-17 [SC]
      F6EC B376 2474 EDA9 D21B 7022 8719 20D1 991B C93C
uid   [desconocida] Ubuntu Archive Automatic Signing Key (2018) <ftpmaster@ubuntu.com>
```

Creamos un directorio sources.list.d con las siguientes lineas.

\$ echo "deb [arch=amd64,arm64] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.4.list

```
angela@angela:~$ echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.4.list
deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4 multiverse
```


Finalmente instalamos el paquete mongodb-org ubicado en el servidor apt.

```
$ sudo apt install mongodb-org
```

```
angela@angela:~$ sudo apt install mongodb-org
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  mongodb-database-tools mongodb-org-database-tools-extra mongodb-org-mongos mongodb-org-server
  mongodb-org-shell mongodb-org-tools
Se instalarán los siguientes paquetes NUEVOS:
  mongodb-database-tools mongodb-org mongodb-org-database-tools-extra mongodb-org-mongos
  mongodb-org-server mongodb-org-shell mongodb-org-tools
0 actualizados, 7 nuevos se instalarán, 0 para eliminar y 41 no actualizados.
Se necesita descargar 104 MB de archivos.
Se utilizarán 200 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s
Des:1 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4/multiverse amd64 mongodb-database-too
ls amd64 100.2.1 [54,4 MB]
Des:2 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4/multiverse amd64 mongodb-org-shell am
d64 4.4.2 [13,2 MB]
Des:3 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4/multiverse amd64 mongodb-org-server a
md64 4.4.2 [20,4 MB]
Des:4 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4/multiverse amd64 mongodb-org-mongos a
md64 4.4.2 [15,7 MB]
Des:5 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4/multiverse amd64 mongodb-org-database
-tools-extra amd64 4.4.2 [5.636 B]
Des:6 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4/multiverse amd64 mongodb-org-tools am
d64 4.4.2 [2.892 B]
Des:7 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4/multiverse amd64 mongodb-org amd64 4.
4.2 [3.520 B]
Descargados 104 MB en 5s (18,9 MB/s)
Seleccionando el paquete mongodb-database-tools previamente no seleccionado.
(Leyendo la base de datos ... 207215 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../0-mongodb-database-tools_100.2.1_amd64.deb ...
Desempaquetando mongodb-database-tools (100.2.1) ...
Seleccionando el paquete mongodb-org-shell previamente no seleccionado.
Preparando para desempaquetar .../1-mongodb-org-shell_4.4.2_amd64.deb ...
Desempaquetando mongodb-org-shell (4.4.2) ...
Progreso: [ 14%] [#####.....]
```

INICIAR SERVICIO Y PROBAR LA BASE DE DATOS

Ya hemos terminado con la instalación ahora debemos iniciar servicio en MongoDB desde la shell y probar la base de datos.

Para ello empezaremos ejecutando el comando systemctl para iniciar el servicio

```
$ sudo systemctl start mongod.service
```

Lo habilitamos con el comando:

```
$ sudo systemctl enable mongod
```

Verificamos que la base está operativa conectando con el servidor. Sabemos que lo está por la devolución del "ok" : 1.

```
$ mongo --eval 'db.runCommand({ connectionStatus: 1 })'
```

```

angela@angela:~$ sudo systemctl start mongod.service
angela@angela:~$ sudo systemctl enable mongod
Created symlink /etc/systemd/system/multi-user.target.wants/mongod.service → /lib/systemd/system/mongod.service.
angela@angela:~$ mongo --eval 'db.runCommand({ connectionStatus: 1 })'
MongoDB shell version v4.4.2
connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongod
Implicit session: session { "id" : UUID("f9b45597-1f32-4d76-afa4-b474ddf22f4b") }
MongoDB server version: 4.4.2
{
  "authInfo" : {
    "authenticatedUsers" : [ ],
    "authenticatedUserRoles" : [ ]
  },
  "ok" : 1
}

```

A continuación comprobamos el estado del servicio con el comando:

`$ sudo systemctl status mongod`

```

angela@angela:~$ sudo systemctl status mongod
● mongod.service - MongoDB Database Server
   Loaded: loaded (/lib/systemd/system/mongod.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2020-12-09 16:38:50 CET; 1min 42s ago
     Docs: https://docs.mongodb.org/manual
    Main PID: 16370 (mongod)
      Memory: 57.5M
    CGroup: /system.slice/mongod.service
           └─16370 /usr/bin/mongod --config /etc/mongod.conf

dic 09 16:38:50 angela systemd[1]: /lib/systemd/system/mongod.service:12: PIDFile= references a path
dic 09 16:38:50 angela systemd[1]: Started MongoDB Database Server.
dic 09 16:39:02 angela systemd[1]: /lib/systemd/system/mongod.service:12: PIDFile= references a path
lines 1-12/12 (END)

```

Volvemos a la página web de mongo y habilitamos el cluster para ser usado desde la shell dándole a connect to Cluster

×

Connect to Cluster0

✓ Setup connection security

✓ Choose a connection method

Connect

I do not have the mongo shell installed

I have the mongo shell installed

1

Select your mongo shell version

4.4

(To check your shell version, run `mongo --version`)

2

Run your connection string in your command line

Use this connection string in your application:

```
mongo "mongodb+srv://cluster0.zv7bh.mongodb.net/<dbname>" --user
```

Copy

Replace **<dbname>** with the name of the database that connections will use by default. You will be prompted for the password for the Database User, **migue**. When entering your password, make sure all special characters are [URL encoded](#).

Having trouble connecting? [View our troubleshooting documentation](#)

Go Back

Close

Copiamos el comando que nos ha puesto con mongo delante y el nombre de nuestra base de datos donde pone <dbname>

```
angela@angela:~$ mongo "mongodb+srv://cluster0.zv7bh.mongodb.net/MiDecsai" --username admin
MongoDB shell version v4.4.2
Enter password:
connecting to: mongodb://cluster0-shard-00-01.zv7bh.mongodb.net:27017,cluster0-shard-00-00.zv7bh.mongodb.net/?ssl=true&gssapiServiceName=mongodb&replicaSet=atlas-5h6b0o-shard-0&ssl=true
Implicit session: session { "id" : UUID("d3d2c937-46fa-4459-8333-912f995fddef") }
MongoDB server version: 4.2.11
WARNING: shell and server versions do not match
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
  https://docs.mongodb.com/
Questions? Try the MongoDB Developer Community Forums
  https://community.mongodb.com
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> MiDecsaio.g_usuario.insert
uncaught exception: ReferenceError: MiDecsaio is not defined :
@(shell):1:1
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> --help
NaN
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> youdb
uncaught exception: ReferenceError: youdb is not defined :
@(shell):1:1
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> find
uncaught exception: ReferenceError: find is not defined :
@(shell):1:1
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY>
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY>
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY>
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY>
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.admincommand(show)
uncaught exception: ReferenceError: show is not defined :
```

Comprobamos que nos hemos conectado bien con un show dbs y parece que así es.

```
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> show dbs
MiDecsai  0.000GB
admin     0.000GB
local     1.283GB
```

Hacemos un db.NombreDeColeccion.find() y efectivamente nos sale un usuario que habíamos añadido anteriormente y eureka nos lo devuelve.

```
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.G_Usuarios.find()
{ "_id" : "12345678A", "Nombre" : "Angela", "Apellidos" : "Gallardo Ruiz", "Correo" : "angela@correo.com" }
```

Probamos insertando usuarios con el comando insert e intentamos introducir un usuario con un mismo id, que es el DNI, y debería de devolvernos un error.

```
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.G_Usuarios.insert({Apellido:"Benitez"})
WriteResult({ "nInserted" : 1 })
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.G_Usuarios.find()
{ "_id" : "12345678A", "Nombre" : "Angela", "Apellidos" : "Gallardo Ruiz", "Correo" : "angela@correo.com" }
{ "_id" : ObjectId("5fd0f7687aa604372c33ff52"), "Apellido" : "Benitez" }
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.G_Usuarios.insert({_id:"12345678A",Apellido:"Benitez"})
```

El error:

```
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.G_Usuarios.insert({_id:"12345678A",Apellido:"Benitez"})
WriteResult({
  "nInserted" : 0,
  "writeError" : {
    "code" : 11000,
    "errmsg" : "E11000 duplicate key error collection: MiDecsal.G_Usuarios index: _id_ dup key: { _id: \"12345678A\" }"
  }
})
```

Adjuntamos más pruebas de comandos que realizamos:

```
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> show dbs
MiDecsal 0.000GB
admin 0.000GB
local 1.283GB
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.g_usuarios.dataSize
function() {
  return this.stats().size;
}
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> g_usuarios.dataSize
uncaught exception: ReferenceError: g_usuarios is not defined :
@(shell):1:1
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> MiDecsal.g_usuarios.dataSize
uncaught exception: ReferenceError: MiDecsal is not defined :
@(shell):1:1
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.G_Usuarios.dataSize()
102
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.G_Usuarios.dataFind()
uncaught exception: TypeError: db.G_Usuarios.dataFind is not a function :
@(shell):1:1
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.G_Usuarios.find()
{ "_id" : "12345678A", "Nombre" : "Angela", "Apellidos" : "Gallardo Ruiz", "Correo" : "angela@correo.com" }
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.G_Usuarios.insert()
uncaught exception: Error: no object passed to insert! :
DBCollection.prototype.insert@src/mongo/shell/collection.js:275:15
@(shell):1:1
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.G_Usuarios.insert({Apellido:"Benitez"})
WriteResult({ "nInserted" : 1 })
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.G_Usuarios.find()
{ "_id" : "12345678A", "Nombre" : "Angela", "Apellidos" : "Gallardo Ruiz", "Correo" : "angela@correo.com" }
{ "_id" : ObjectId("5fd0f7687aa604372c33ff52"), "Apellido" : "Benitez" }
MongoDB Enterprise atlas-5h6b0o-shard-0:PRIMARY> db.G_Usuarios.insert({_id:"12345678A",Apellido:"Benitez"})
WriteResult({
  "nInserted" : 0,
  "writeError" : {
    "code" : 11000,
    "errmsg" : "E11000 duplicate key error collection: MiDecsal.G_Usuarios index: _id_ dup key: { _id: \"12345678A\" }"
  }
})
```

11.5. Crear una base de datos

Para poder crear una base de datos en mongoDB, debemos de tener el shell de mongo instalado y conectado a nuestro cluster. Una vez hecho, comenzamos nuestra base de datos usando el comando use. Cuando hayamos creado la tabla, nos conectamos a ella y comenzamos a añadir datos. Ejemplo:

use MiTabla;

```
mongo = new Mongo();
```

```
db = mongo.getDB("MiTabla");
```

```
db.createCollection("UnaTabla",
```

```
{
```

```
  autoIndexId:      false,
```

```
  Nombre:           <string>,
```

```
  Apellidos:        <string>,
```

```
);

db.UnaTabla.insertOne(
{
    _id      :      12345678A,
    Nombre   :      " UnNombre",
    Apellidos :      "UnApellido"
}
```

Si lo comparamos con un ejemplo en SQL sería algo como:

```
CREATE TABLE MiTabla(
    DNI varchar2(9) PRIMARY KEY,
    Nombre varchar2(15),
    Apellidos varchar2(35),
);
```

11.6. Dar acceso a programadores

Una vez creadas las colecciones, nos vamos a la pestaña de 'Access Manager' (arriba a la izquierda) y nos aparece esta vista. Le damos a 'Invite User' (en verde arriba a la derecha). Seguidamente añadimos su correo electrónico donde le llegará la petición para unirse al proyecto con el cluster previamente creado. Una vez que acepte tenemos que volver a esta pantalla de 'Access Manager' donde tendremos que cambiar los permisos en 'Edit Permission' para que tenga total acceso.

The screenshot shows the MongoDB Atlas 'Organization Access Manager' interface. The left sidebar contains navigation links: ORGANIZATION, Projects, Alerts, Activity Feed, Settings, Access Manager (highlighted), Billing, and Support. The main content area is titled 'Organization Access Manager' and includes a search bar and tabs for Users, Teams, and API Keys. A table lists the following users:

Display Name	Email/Username	Organization Role	Projects	Last Login Date	Actions
Charts User	charts+5fbc752de1b0c0273a80e1e1@mongodb.com	Organization Member	1	Has not logged in yet	EDIT PERMISSIONS
Javier Gallardo	e.fgallardo00@go.ugr.es	Organization Member	0	12/12/20 - 09:45 AM	EDIT PERMISSIONS
JOSE ANTONIO PAREJO BELLIDO	e.parejo@go.ugr.es	Organization Owner	0	12/12/20 - 10:05 AM	EDIT PERMISSIONS
ANGELA GALLARDO RUIZ	e.angelagr@go.ugr.es	Organization Owner	0	12/11/20 - 10:39 PM	EDIT PERMISSIONS
MIGUEL ANGEL	e.miguelaguacil@go.ugr.es	Organization Owner	0	12/12/20 - 11:09 AM	EDIT PERMISSIONS
SAMUEL JIMENEZ PIÑERO	e.sjp1906@go.ugr.es	Organization Owner	1	12/12/20 - 11:25 AM	EDIT PERMISSIONS

At the bottom of the page, there is a status bar showing 'System Status: All Good', 'Last Login: 85:49:195.42', and copyright information for MongoDB, Inc. A 'Feature Requests' link is also present.

Una vez incluido con los correctos permisos en el proyecto, tendremos que ir a la pantalla principal:

The screenshot shows the MongoDB Atlas interface. The left sidebar has a 'SECURITY' section with 'Network Access' highlighted. The main panel shows the 'Clusters' page for 'Cluster0'. It includes a search bar, tabs for 'CONNECT', 'METRICS', and 'COLLECTIONS', and various performance graphs. A 'Create a New Cluster' button is in the top right corner.

Y seleccionar 'Network Access' donde cada usuario tendrá que añadir su IP para conectarse al cluster de forma remota.

The screenshot shows the 'Network Access' page in MongoDB Atlas. The left sidebar has 'Network Access' selected. The main panel shows the 'IP Access List' tab. It includes a message: 'You will only be able to connect to your cluster from the following list of IP Addresses:'. Below this is a table with columns for IP Address, Comment, Status, and Actions. The table lists two IP addresses: '85.49.195.42/32' and '0.0.0.0/0', both with a status of 'Active'. A '+ ADD IP ADDRESS' button is in the top right corner.

11.7. Conectar la base de datos de Mongodb con nuestra aplicación

1º Hace falta crear un archivo Json que conecte con nuestra base de datos, como el siguiente:

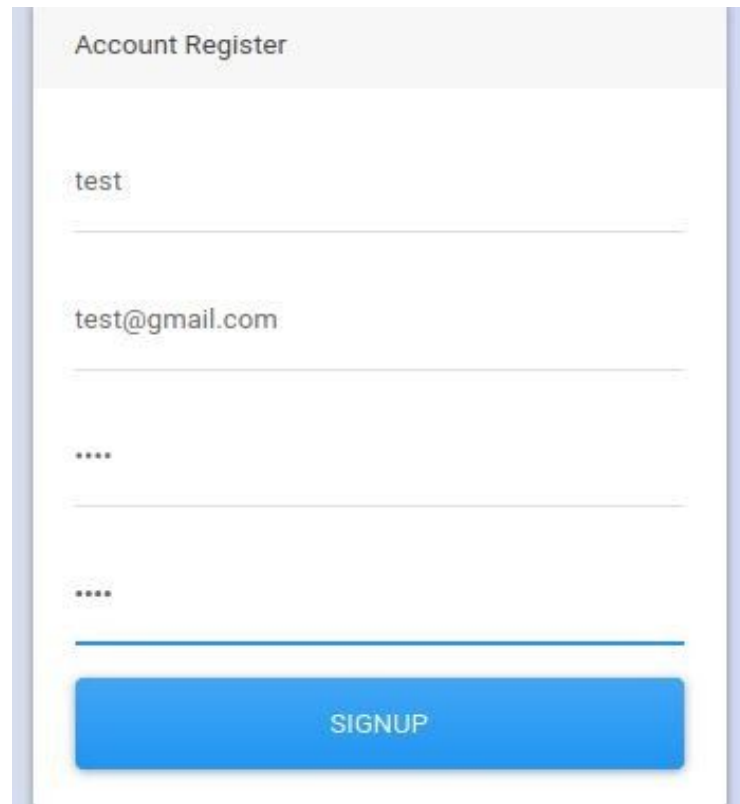
```
JS database.js •
1  const mongoose = require("mongoose");
2  const config = require("./config");
3
4  const MONGODB_URI = `mongodb+srv://parejo:<password>@realmcluster.l27da.mongodb.net/<dbname>?retryWrites=true&w=majority`;
5
6  mongoose
7    .connect(MONGODB_URI, {
8      useNewUrlParser: true,
9      useUnifiedTopology: true,
10     useFindAndModify: false,
11     useCreateIndex: true,
12   })
13   .then((db) => console.log("Mongodb is connected to", db.connection.host))
14   .catch((err) => console.error(err));
15
```

A continuación entraremos a nuestra cuenta de mongo y en nuestro cluster nos aparecerá la opción connect, la seleccionamos y elegimos la opción "Connect your application" y copiamos el string de conexión que se nos facilita, en nuestro código lo pegamos en la constante MONGODB_URI y nos faltaría introducir la contraseña en <password> y el nombre de nuestra base de datos en <dbname>, si no has creado ninguna base de datos te la crea automáticamente con el nombre que pongas .

Una vez enlazado con nuestra base de datos solamente debemos iniciar nuestro sistema y comprobar que se insertan los datos en nuestra base de datos, en nuestro caso no necesitamos crear ninguna tabla ni nada parecido, si no que mongo genera documentos en los que guarda cada entrada de datos.


A continuación se mostrarán algunas imágenes para que comprobéis como aparecen los datos insertados en la BD, para ello haremos:

El registro de un usuario “test” con un email test@gmail.com”



The image shows a mobile application screen titled "Account Register". It features four input fields: the first contains "test", the second contains "test@gmail.com", and the third and fourth are masked with four dots each. A blue "SIGNUP" button is positioned at the bottom of the form.

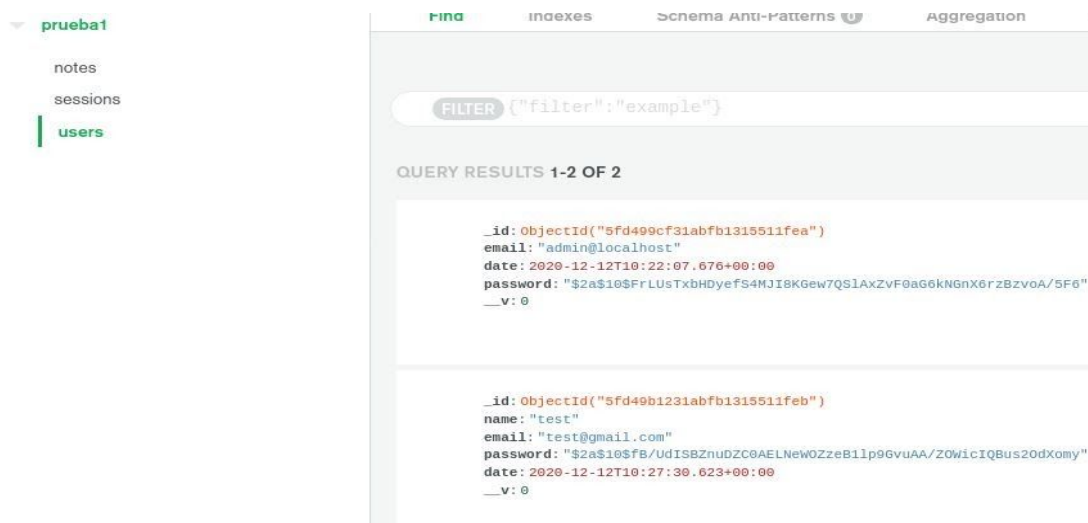
Crearemos una nota que diga “Hola”:



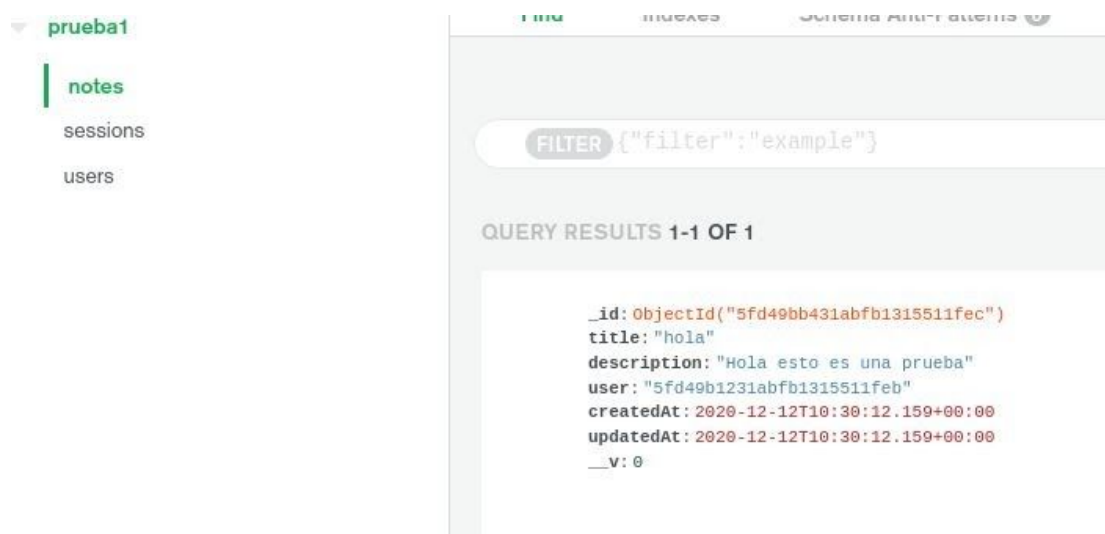
The image shows a mobile application screen titled "New Note". It has two text input fields. The first field contains the word "hola". The second field contains the text "Hola esto es una prueba", with the word "una" underlined in red. A blue "SAVE" button is located at the bottom of the screen.

Vemos en la web de MongoDB como se han creado ambas cosas correctamente en sus respectivas colecciones:

En la colección de users:



En la colección de notes:



Para realizar este ejemplo con el que nos estamos ayudando para realizar nuestra conexión a nuestra base de datos, hemos utilizado el código que se proporciona en el siguiente repositorio de github: <https://github.com/FaztTech/nodejs-notes-app>

Y también gracias a los videos del siguiente canal:

<https://www.youtube.com/channel/UCX9NJ471o7Wie1DQe94RVlg>

En dicho canal se explica paso a paso cómo se crea el código que se encuentra en el repositorio anteriormente mencionado y algunos tutoriales sobre MongoDB.

12. Ejemplo: ¿Cómo hacer un get de los datos?

Una de las funciones básicas de una base de datos es poder obtener los datos para poder trabajar con ellos en nuestra aplicación.

En este ejemplo vamos a ver cómo funciona la página de “Gestión de Plazos”, para esta página necesitamos que se muestren los datos de la colección “G_Plazos” y que se muestren los ‘atributos’ de “nombre”, “descripcion”, “fecha_ini” y “fecha_fin”.

MiDecsai

GESTIÓN DE USUARIOS

GESTIÓN DE PLAZOS

Gestión De Plazos

dd/mm/aaaa

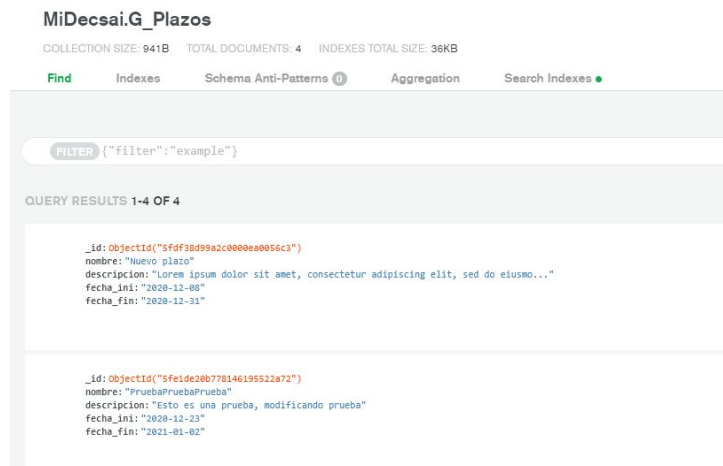
dd/mm/aaaa

AÑADIR PLAZO

Nombre	Descripción	Fecha De Inicio	Fecha De Fin		
Nuevo Plazo	<div><div><div>Lorem Ipsum Dolor Sit Amet, Consectetur Adipiscing Elit, Sed Do Eiusmod Tempor Incididunt Ut Labore Et Dolor Magna Aliqua. Ut Enim Ad Minim Veniam, Quis Nostrud Exercitation Ullamco Laboris Nisi Ut Aliquip Ex Ea Commodo Consequat. Duis Aute Irure Dolor In Reprehenderit In Voluptate Velit Esse Cillum Dolor Eu Fugiat Nulla Pariatur. Excepteur Sint Occaecat Cupidatat Non Proident, Sunt In Culpa Qui Officia Deserunt Mollit Anim Id Est Laborum</div></div></div>	2020-12-08	2020-12-31	Modificar	Eliminar
PruebaPruebaPrueba	Esto Es Una Prueba, Modificando Prueba	2020-12-23	2021-01-02	Modificar	Eliminar
Empezar Seminario	Asd Asd Asd As Da	2021-01-15	2021-01-29	Modificar	Eliminar
Nuevo Modificado	Asda Sds Da			Modificar	Eliminar

Para poder obtener los datos, lo primero es conectarse a nuestra base de datos, en nuestro caso, MiDecsai.

Ahora, tenemos que tener en cuenta dónde están nuestros datos, es decir, en qué colección.



Para conectarnos a la base de datos hemos usado lo siguiente:

- “conexionMongo(\$db)” en esta función establecemos la conexión con mongo.
- “getCollection(\$coleccion)” aquí introducimos como argumento el nombre de la colección que queremos obtener y nos proporciona todos los datos que hay en esa colección usando find().


```

1 <?php
2     require_once __DIR__ . '/../vendor/autoload.php';
3     function conexionMongo($db){
4         $client = new MongoClient(
5             'mongodbsrv://admin:admin@cluster0.zv7bh.mongodb.net/' . $db . 'MiDecsai?retryWrites=true&w=majority');
6
7         return $client->getDb();
8     }
9
10    function getCollection($coleccion){
11        $db = conexionMongo('MiDecsai');
12        $collection = $db->$coleccion;
13        $cursor = $collection->find();
14
15        return $cursor;
16    }
17 }>

```

Lo primero que tenemos que tener en cuenta es cómo devuelve los datos MongoDB desde la aplicación. Para hacer esto vamos a añadir lo siguiente a nuestro archivo `gestion_plazos.php`:

```

<?php
include 'get.php';

$plazos = getCollection('G_Plazos')->toArray();

?>

```

De esta manera estamos metiendo toda la información que hay en la colección `G_Plazos`, convertida en un array, en la variable `$plazos`. Este array se ve de la siguiente forma:

```

Array ( [0] => MongoDB\Model\BSONDocument Object ( [Storage:ArrayObject:private] => Array ( [_id] => MongoDB\BSON\ObjectId Object ( [Oid] => 5fdf38d99a2c0000ea0056c3 ) [Nombre] => Nuevo Plazo [Descripcion] => Lorem Ipsum Dolor Sit
Amet, Consectetur Adipiscing Elit, Sed Do Eiusmod Tempor Incididunt Ut Labore Et Dolore Magna Aliqua. Ut Enim Ad Minim Veniam, Quis Nostrud Exercitation Ullamco Laboris Nisi Ut Aliquip Ex Ea Commodore Consequat.
Duis Aute Irure Dolor In Reprehenderit In Voluptate Velit Esse Cillum Dolor Eu Fugiat Nulla Pariatur. Excepteur Sint Occaecat Cupidatat Non Proident, Sunt In Culpa Qui Officia Deserunt Mollit Anim Id Est Laborum [Fecha_ini] =>
2020-12-08 [Fecha_fin] => 2020-12-31 ) ) [1] => MongoDB\Model\BSONDocument Object ( [Storage:ArrayObject:private] => Array ( [_id] => MongoDB\BSON\ObjectId Object ( [Oid] => 5f5e20b778146195522a72 ) [Nombre] =>
PruebaPruebaPrueba [Descripcion] => Esto Es Una Prueba, Modificando Prueba [Fecha_ini] => 2020-12-23 [Fecha_fin] => 2021-01-02 ) ) [2] => MongoDB\Model\BSONDocument Object ( [Storage:ArrayObject:private] => Array ( [_id] =>
MongoDB\BSON\ObjectId Object ( [Oid] => 5fff0c43ee670000bf006963 ) [Nombre] => Empezar Seminario [Descripcion] => Asd Asd Asd As Da [Fecha_ini] => 2021-01-15 [Fecha_fin] => 2021-01-29 ) ) [3] => MongoDB\Model\BSONDocument
Object ( [Storage:ArrayObject:private] => Array ( [_id] => MongoDB\BSON\ObjectId Object ( [Oid] => 5fff0eb5ee670000bf006964 ) [Nombre] => Nuevo Modificado [Descripcion] => Asda Sds Da ) ) )

```

Hemos hecho un `print_r($plazos)` para ver lo que tenía dentro esta variable, como podemos ver, no es un array de string normal y corriente sino que hay más cosas. Vamos a intentar separarlo por ‘documentos’, lo que sería el equivalente a nuestras filas de SQL, cada posición de nuestro array nos dará un documento.

```

MongoDB\Model\BSONDocument Object ( [Storage:ArrayObject:private] => Array ( [_id] => MongoDB\BSON\ObjectId Object ( [Oid] => 5fdf38d99a2c0000ea0056c3 ) [Nombre] => Nuevo Plazo [Descripcion] => Lorem Ipsum Dolor Sit
Amet, Consectetur Adipiscing Elit, Sed Do Eiusmod Tempor Incididunt Ut Labore Et Dolore Magna Aliqua. Ut Enim Ad Minim Veniam, Quis Nostrud Exercitation Ullamco Laboris Nisi Ut Aliquip Ex Ea Commodore Consequat. Duis
Aute Irure Dolor In Reprehenderit In Voluptate Velit Esse Cillum Dolor Eu Fugiat Nulla Pariatur. Excepteur Sint Occaecat Cupidatat Non Proident, Sunt In Culpa Qui Officia Deserunt Mollit Anim Id Est Laborum [Fecha_ini] => 2020-
12-08 [Fecha_fin] => 2020-12-31 ) )

MongoDB\Model\BSONDocument Object ( [Storage:ArrayObject:private] => Array ( [_id] => MongoDB\BSON\ObjectId Object ( [Oid] => 5f5e20b778146195522a72 ) [Nombre] => PruebaPruebaPrueba [Descripcion] => Esto Es Una Prueba,
Modificando Prueba [Fecha_ini] => 2020-12-23 [Fecha_fin] => 2021-01-02 ) )

MongoDB\Model\BSONDocument Object ( [Storage:ArrayObject:private] => Array ( [_id] => MongoDB\BSON\ObjectId Object ( [Oid] => 5fff0c43ee670000bf006963 ) [Nombre] => Empezar Seminario [Descripcion] => Asd Asd Asd As Da
[Fecha_ini] => 2021-01-15 [Fecha_fin] => 2021-01-29 ) )

MongoDB\Model\BSONDocument Object ( [Storage:ArrayObject:private] => Array ( [_id] => MongoDB\BSON\ObjectId Object ( [Oid] => 5fff0eb5ee670000bf006964 ) [Nombre] => Nuevo Modificado [Descripcion] => Asda Sds Da ) )

```

Si nos fijamos ahora cada documento tiene la forma de:

```

MongoDB\Model\BSONDocument Object ( [Storage:ArrayObject:private] => Array (
[_id] => MongoDB\BSON\ObjectId Object ( [Oid] => ID_DEL_DOCUMENTO)
[Atributo_1] => Contenido Atributo_1 [Atributo_2] => Contenido Atributo_2 .... ) )

```

OJO! Tiene esta forma porque previamente lo hemos convertido en una array con el método “`toArray()`”

Para poder acceder a cada uno de esos atributos, hacemos `$plazo->nombre_atributo`.

En nuestro caso, para poder poner los datos en la tabla de Gestión de Plazos, mezclamos el html con el php para acceder a cada atributo de cada plazo introducido:

```
<?php $contador_plazo = -1; ?>

<?php foreach ($plazos as $plazo): ?>
    <tr>
        <?php $contador_plazo = $contador_plazo + 1; ?>

        <td style="vertical-align:middle;">
            <?php if(isset($plazo->nombre)) echo $plazo->nombre ?>
        </td>
        <td style="width: 40%; vertical-align:middle;">
            <?php if(isset($plazo->descripcion)) echo $plazo->descripcion ?>
        </td>
        <td style="vertical-align:middle;">
            <?php if(isset($plazo->fecha_ini)) echo $plazo->fecha_ini ?>
        </td>
        <td style="vertical-align:middle;">
            <?php if(isset($plazo->fecha_fin)) echo $plazo->fecha_fin ?>
        </td>
    </tr>
</foreach>
```

Si nos fijamos, cada dato tiene un `if(isset(...)) echo ...`, esto nos sirve para comprobar que el dato con el nombre de atributo que hemos introducido **exista y no esté vacío**, de esta forma, si no tiene datos válidos, no lo escribe.

Esto es especialmente importante en mongo ya que un documento puede tener cualquier cosa, a diferencia de sql que tendrá null si el dato no está relleno.

Así terminaría nuestro ejemplo de cómo obtener los datos de G_Plazos y mostrarlos.

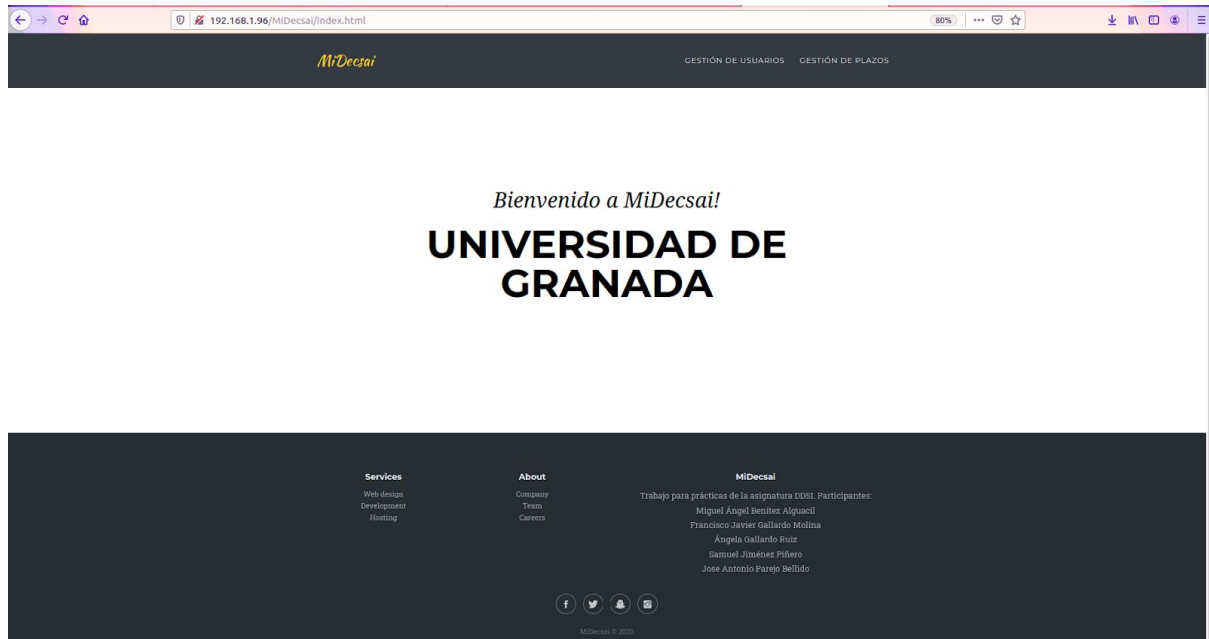
13. Tabla asignaturas.

En esta tabla mostraremos las asignaturas que hemos cursado, junto los conceptos claves aprendidos en dichas asignaturas, que tengan relación con Diseño y Desarrollo de Sistemas de Información.

Nombre Asignatura	Conceptos aprendidos relacionados con DDSI
Programación Web	Uso de lenguaje HTML5 (conceptos básicos)
Desarrollo de Aplicaciones para Internet	Uso de lenguaje HTML5 (más avanzado), PHP (muy básico) y Javascript (muy básico)
Metodologías de Desarrollo Ágil	Uso de scrum
Fundamento de Bases de Datos	Conceptos básicos de bases de datos (MySQL, SQL developer), esquemas entidad relación y paso a tablas,
Administración de Bases de Datos	Continuación de FBD (conceptos más avanzados)
Transmisión de datos y redes de computadores	Puesta en marcha de un servicio telemático en Internet
Fundamentos de Redes	Configuración de servicios de Red, gestión de Red
Ingeniería de Servidores	Configuración y desarrollo de un servidor web
Servidores Web de Altas Prestaciones	Continuación de ISE

14. Capturas de pantalla del sistema

A continuación vamos a mostrar unas capturas de nuestro sistema. En el pie de cada imagen explicaremos un poco de cómo está implementado.



Aquí solo hay código html. Quizás lo más destacable sean los dos botones de gestión de usuarios y gestión de plazos que básicamente nos redirigen a la url de uno o de otro.



Dentro de gestión de usuarios tenemos, además del html, un bucle for escrito en javascript que recorre la colección de MongoDB de usuarios mostrando el DNI, email, tipo de usuario y fecha de alta. Además de mostrar todo esto nos permite ir a otra url para modificar cada uno o eliminarlo. Tenemos además dos botones más para o bien añadir usuarios o bien crear nuevos tipos de los mismos. Los campos donde pone "filtrar" están actualmente inoperativos.




Modificar Datos Personales

DNI: 11233455Y Correo: modificacion@gmail.com

Fecha Entrada Departamento: 2020/12/23

Guardar

Si le damos a modificar los datos personales nos lleva a esta url. Aquí en el front end la mejor solución que hemos ideado es que los datos del usuario se borran y luego se crea un usuario con esos mismos datos o con sus modificaciones si estas han sido realizadas.



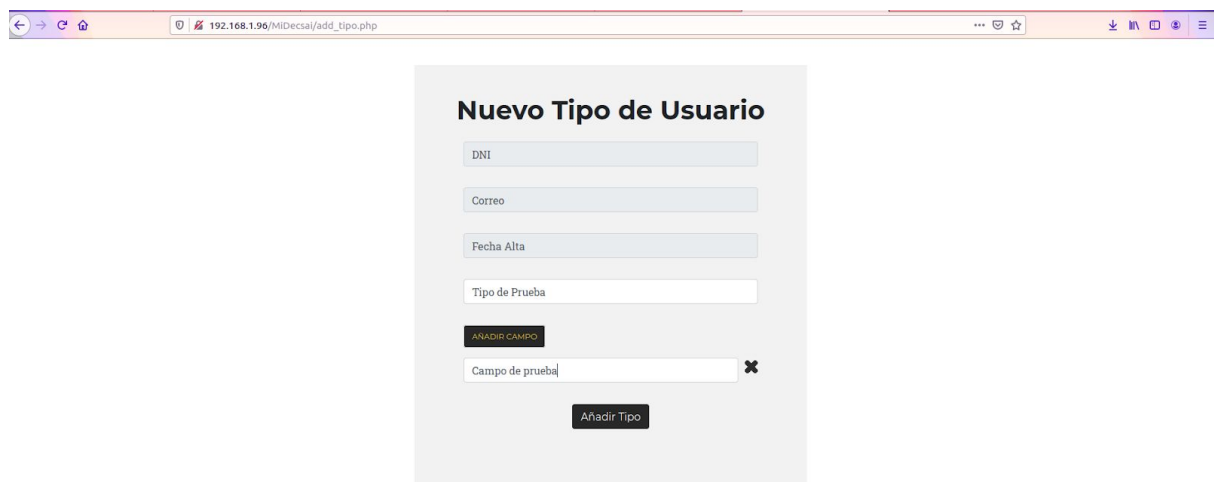
Dar de Alta Usuario

DNI: 1234567K Correo: prueba3@gmail.com

Profesor

Dar de Alta

Si le damos a dar de alta nos lleva a esta pantalla. Aquí lo que hacemos es realizar un script con los campos de usuario DNI, correo y tipo y seguidamente se lo mandamos al cluster.



Nuevo Tipo de Usuario

DNI: Correo: Fecha Alta: Tipo de Prueba:

AÑADIR CAMPO

Campo de prueba

Añadir Tipo

Clicando en añadir tipo de usuario nos sale esta pantalla en la que podemos añadir tipos de usuarios con más o menos campos. Utilizamos bastante javascript para que eso se pueda llevar a cabo.

Nombre	Descripción	Fecha De Inicio	Fecha De Fin		
Nuevo Plazo	Lorem Ipsum Dolor Sit Amet, Consectetur Adipiscing Elit. Sed Do Eiusmod Tempor Incididunt Ut Labore Et Dolor Magna Aliqua. Ut Enim Ad Minim Veniam, Quis Nostrud Exercitation Ullamco Laboris Nisi Ut Aliquip Ex Ea Commodore Consequat. Duis Aute Irure Dolor In Reprehenderit In Voluptate Velit Esse Cillum Dolor Eu Fugiat Nulla Pariatur. Excepteur Sint Occaecat Cupidatat Non Proident, Sunt In Culpa Qui Officia Deserunt Mollit Anim Id Est Laborum	2020-12-08	2020-12-31	Modificar	Eliminar
PruebaPruebaPrueba	Esto Es Una Prueba	2020-12-23	2021-01-02	Modificar	Eliminar

Si nos vamos a gestión de plazos funciona de igual modo que la de gestión de usuarios.

Funciona igual que modificar usuarios.

Funciona de igual modo que añadir usuarios.

15. Servidor web para alojar la página web

Para alojar la página web hemos montado nuestro propio servidor web en una Raspberry Pi 4. Para ello, instalamos una imagen de Ubuntu Server 20.04.1 LTS con Apache, PHP 7. También utilizamos Composer como administrador de dependencias para PHP y poder instalar las extensiones de mongo para PHP.



Por otro lado, hemos obtenido el dominio gratuito registrándonos en www.noip.com y hemos abierto el puerto 80 del router.

Editar

Nombre	Protocolo	Puerto/Rango Externo	Puerto/Rango Interno	Direccion IP	Activar
Raspberry servidor	TCP ▼	80:80	80:80		<input checked="" type="checkbox"/> ON

El proyecto se encuentra en la carpeta MiDecsai dentro del directorio /var/www/html/

```
ubuntu@ubuntu: /var/www/html$ ls
MiDecsai  composer.json  composer.lock  index.html  phpinfo.php  vendor
```

La dirección en la que encontraremos nuestro proyecto es <http://midecsai.hopto.org/>

En caso de que el servidor deje de estar disponible, hemos preparado un vídeo en el que podemos ver el funcionamiento de la página web. Se encuentra en:

<https://drive.google.com/file/d/128M-CVjJCKML7wclXAU7juxzOyJnMK7F/view?usp=sharing>

16. Trello

Práctica 1

Practica 1

- Terminar el documento y enviar (4 de nov.)

En proceso (Práctica 1)

- Hacer pantallazos a Trello

Hecho Practica 1

- Terminar DFD1 (2 de nov.)
- Preguntar Dudas
- DFD de refinamiento
- Gestión de Usuarios (30 de oct.)
- Plazos del Sistema (30 de oct.)
- Esquema Armazón
- Esquema de Caja Negra (27 de oct.)
- Completar Mapa Mental (27 de oct.)
- Dividir la práctica
- Mandar borrador de la práctica
- Gestión de Usuarios (2/2)
- Plazos del Sistema (2/2)
- Listado de Restricciones Semánticas

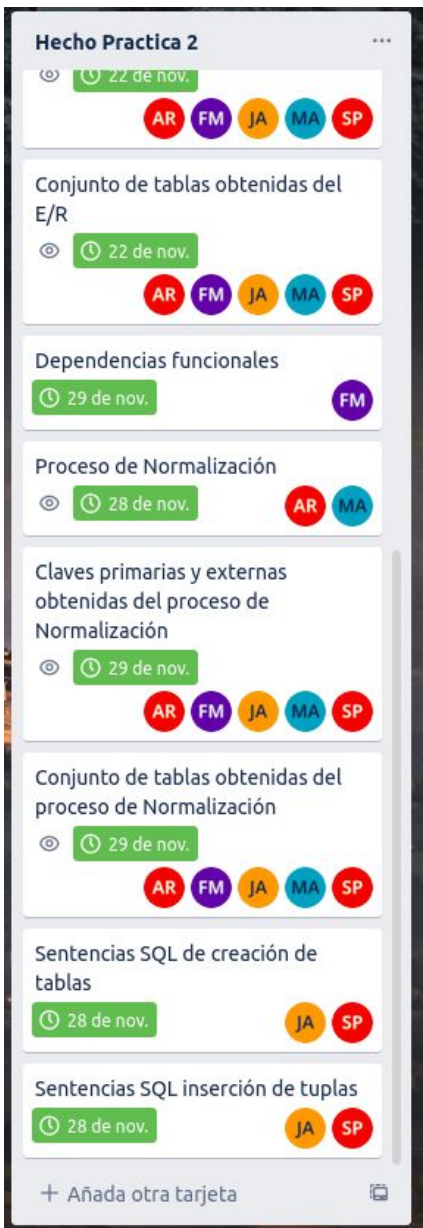
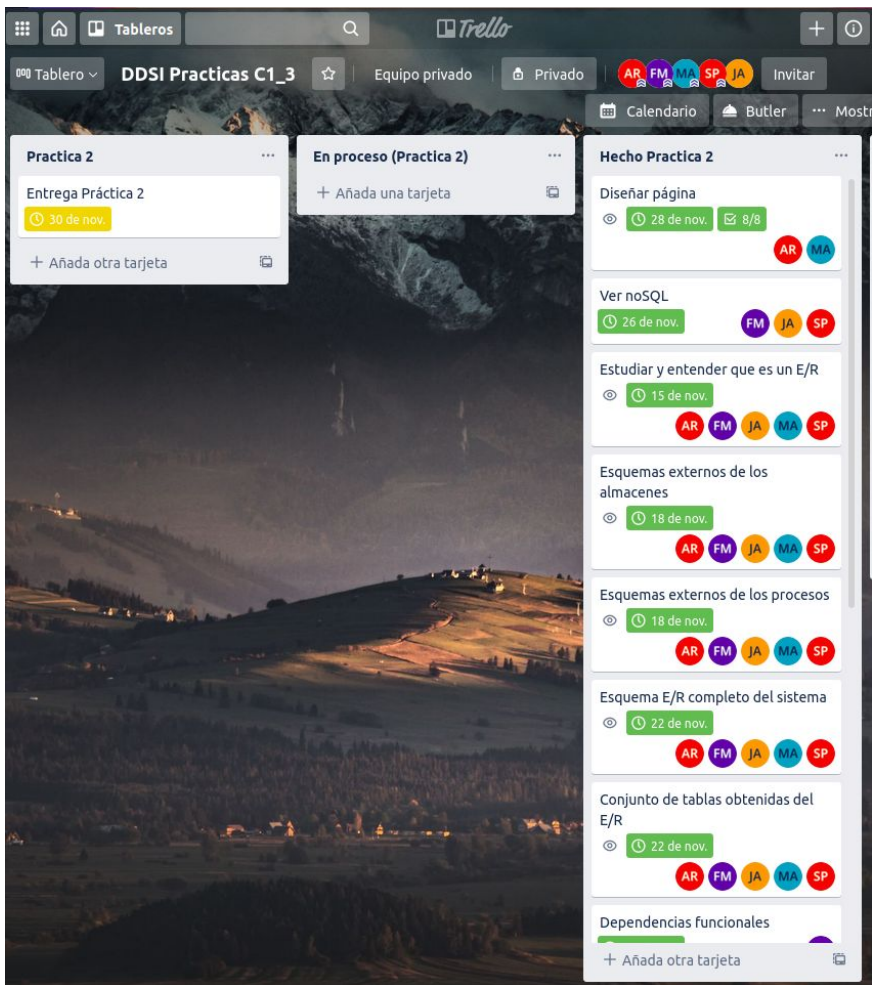
octubre 2020

LUN.	MAR.	MIÉ.	JUE.	VIE.	SÁB.	DOM.
12	13	14	15	16	17	18
19	20	21 1 elemento Gestión de Usuario Plazos del Sistema Primera aproximación	22	23	24	25 2 elementos Listado de Requisitos Funcionales Gestión de Usuario
26	27 2 elementos Esquema de Caja Negra Completar Mapa Mental	28	29	30 1 elemento Gestión de Usuario Plazos del Sistema Esquema Armazón	31 de oct. 1 elemento Primera aproximación Seminario	1 de nov.
2 1 elemento Terminar DFD1	3 1 elemento Buscar información	4 1 elemento Terminar el documento y enviar	5	6	7	8

Hecho Practica 1

- Gestión de Usuarios (2/2)
- Plazos del Sistema (2/2)
- Listado de Restricciones Semánticas
- Gestión de Usuarios (2/2)
- Plazos del Sistema (2/2)
- Listado de Requisitos de Datos
- Gestión de Usuarios (2/2)
- Plazos del Sistema (2/2)
- Listado de Requisitos Funcionales
- Gestión de Usuarios (2/2)
- Plazos del Sistema (2/2)
- Historias de usuario
- Mapa mental (Primer borrador)
- Gestión de Usuarios (2/2)
- Plazos del Sistema (2/2)
- Primera aproximación

Práctica 2



noviembre 2020						
LUN.	MAR.	MIÉ.	JUE.	VIE.	SÁB.	DOM.
9	10	11	12	13	14	15 1 elemento Estudiar y entender que es un E/R
16	17	18 2 elementos Esquemas externos de los almacenes Esquemas	19	20	21	22 2 elementos Esquema E/R completo del sistema Conjunto de
23	24	25	26 1 elemento Ver noSQL	27	28 4 elementos Diseñar página Proceso de Normalización	29 3 elementos Dependencias funcionales Claves primarias y
30 de nov. Entrega Práctica 2	1 de dic.	2	3	4	5	6

Seminario 2

Hecho Seminario 2

Buscar informacion

👁

🕒 9 de dic.

📧 4/4

AR FM JA MA SP

Hacer documento

🔔 2

👁

🕒 12 de dic.

AR FM JA MA SP

Crear base de datos

👁

AR FM JA MA SP

Crear tablas

👁

AR FM JA MA SP

Vincular base de datos

👁

AR FM JA MA SP

+ Añada otra tarjeta

📅

diciembre 2020						
LUN.	MAR.	MIÉ.	JUE.	VIE.	SÁB.	DOM.
30 de nov. 1 elemento <i>Entrega Práctica-2</i>	1 de dic.	2	3	4	5	6
7	8	9 1 elemento <i>Buscar informacion</i>	10	11	12 1 elemento <i>Hacer documento</i>	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27

Seminario 3

Hecho Seminario 3

- Terminar html
17 de dic. MA
- Crear servidor web
23 de dic. AR
- Conectar
17 de dic. AR
- Crear servicio php
1 🔔 1 👁 AR FM JA MA SP
- Eliminar usuario
1 🔔 1 👁 23 de dic. FM
- Modificar usuario
23 de dic. JA

+ Añada otra tarjeta

Hecho Seminario 3

- Modificar usuario
23 de dic. JA
- Añadir tipo de usuario
23 de dic. SP
- Añadir plazo
19 de dic. MA
- Eliminar plazo
19 de dic. MA
- Modificar plazo
19 de dic. MA
- Tabla asignaturas
23 de dic. JA

+ Añada otra tarjeta

Hecho Seminario 3

- Añadir plazo
19 de dic. MA
- Eliminar plazo
19 de dic. MA
- Modificar plazo
19 de dic. MA
- Tabla asignaturas
23 de dic. JA
- Terminar documento
2 🔔 1 👁 23 de dic. AR FM JA MA SP

+ Añada otra tarjeta

dicembre 2020						
LUN.	MAR.	MIÉ.	JUE.	VIE.	SÁB.	DOM.
14	15	16	17 2 elementos Terminar.html Conectar	18	19 3 elementos Añadir-plazo Eliminar-plazo Modificar-plazo	20
21	22	23 6 elementos Crear-servidor-web Eliminar-usuario Modificar-usuario	24	25	26	27
28	29	30	31 de dic.	1 de ene.	2	3
4	5	6	7	8	9	10

Práctica 3

Práctica 3

Crear nuevo tipo usuario (con html "dinámico")

30 de dic. de 2020

MA

Modificar Usuario (Arreglar)

8 de ene.

ARJA

Abrir puertos servidor web

12 de ene.

AR

Pulir documento

16 de ene.

FMSP

Crear video de página web

17 de ene.

AR

Ejemplo de uso de get en mongo

14 de ene.

JA

+ Añada otra tarjeta

enero 2021							Reserva tus comentarios x		Semana	Mes	⚙	✕
LUN.	MAR.	MIÉ.	JUE.	VIÉ.	SÁB.	DOM.						
28	29	30 1 elemento Crear nuevo tipo usuario (con html "dinámico")	31 de dic.	1 de ene.	2	3						
4	5	6	7	8 1 elemento Modificar Usuario (Arreglar)	9	10						
11	12 1 elemento Abrir puertos servidor web	13	14 1 elemento Ejemplo de uso de get en mongo	15	16 1 elemento Pulir documento	17 1 elemento Crear video de página web						