

NOMBRE Y APELLIDOS:			FECHA: 18-09-2023		
DOCENTE: MANUEL MACÍAS PÉREZ			NOTA:		
(IFCD0210) DESARROLLO DE APLICACIONES CON TECNOLOGÍAS WEB.			Nº CURSO: 22-35/008902		
MF:	0492	UNIDADES DE APRENDIZAJE A LAS QUE RESPONDE:	UA	Duración:	3 h
UF:	1845				
PRÁCTICA Nº:	E2				
DENOMINACIÓN: Ficheros de datos.					
<p><u>DESCRIPCIÓN</u></p> <p>1.- El alumno de forma individual deberá realizar lo siguiente:</p> <ul style="list-style-type: none"> • Crear un componente en Angular para enviar y recibir datos desde la base de datos creada en la prueba E1. • Crear su vista plantilla en html • Dar estilos con CSS o utilizar Angular Material. <p>Pegar en este Archivo el pantallazo de cómo se vería la vista en un navegador.</p> <p>Pegar en este Archivo el pantallazo de la base creada en E1 con al menos un objeto guardado.</p> <p>Adjuntar el código del archivo TypeScript del componente creado y convertir este documento en pdf. Enviar o Subir a Github.</p> <p>La práctica se realizará de manera individual.</p> <p><u>MEDIOS PARA SU REALIZACIÓN</u></p> <ul style="list-style-type: none"> - Equipo informático. - Aplicación Visual Code Studio instalada en el equipo. - Navegadores actualizados <p><u>PAUTAS DE ACTUACIÓN DEL FORMADOR</u></p> <p><i>Al inicio de la práctica, que se desarrollará de manera individual por cada uno de los alumnos, el formador/a realizará las siguientes actuaciones:</i></p> <ul style="list-style-type: none"> - Fijará los objetivos de la práctica. - Aportará las instrucciones necesarias a los alumnos/as para la realización de la misma, haciendo hincapié en aquellos aspectos más relevantes. - Facilitará a cada alumno/a la documentación necesaria para el desarrollo de la práctica. - Resolverá las dudas que se planteen durante el transcurso de la práctica, con objeto de que el alumnado aprenda y pueda concluir la realización de la misma. <p>Durante la realización de la práctica el formador/a supervisará el desarrollo de esta para evaluar tanto los procedimientos como el resultado final.</p> <p>Al finalizar la práctica el formador examinará el desarrollo que han realizado los/as alumnos/as, proponiendo las medidas de corrección, en caso necesario.</p> <p style="text-align: center;">ESPECIFICACIONES PARA LA EVALUACIÓN DE LA PRÁCTICA</p>					

Resultados a comprobar	Indicadores de logro
1. Integrar sentencias en los componentes software para acceder y manipular la información ubicada en bases de datos Conforme el criterio de evaluación CE 1.2	1.1 Integra sentencias en los componentes software para acceder a la información ubicada en bases de datos
	1.2 Manipula la información ubicada en bases de datos
2. En un supuesto práctico en el que se pide construir componentes de software que accedan a datos soportados en bases de datos u otras estructuras de almacenamiento. Conforme el criterio de evaluación CE 1.3	2.1 Identifica los elementos y estructuras contenidas en una base de datos.
	2.2 Utiliza los objetos, conectores y middleware necesarios en la construcción del componente para realizar los accesos a los datos soportados en la base de datos
	2.3 Realiza operaciones de definición y manipulación de informaciones soportadas en bases de datos

Sistema de valoración

Definición de indicadores y escalas de medida

Los indicadores que se van a establecer, será una hoja de chequeo, sistema de valoración, que complementa a este documento, donde se evalúan todos los resultados a comprobar (tareas). En este documento, se establecerán a su vez los indicadores de logro que se han de tener en cuenta, para conseguir los resultados a comprobar.

Mínimo exigible

El mínimo exigible para la superación de la práctica es de 50 puntos sobre 100 puntos

SUPUESTO PRÁCTICO

1.- El alumno de forma individual deberá realizar lo siguiente:

- Crear un componente en Angular para enviar y recibir datos desde la base de datos creada en la prueba E1.
- Crear su vista plantilla en html
- Dar estilos con CSS o utilizar Angular Material.

Pegar en este Archivo el pantallazo de cómo se vería la vista en un navegador.

Pegar en este Archivo el pantallazo de la base creada en E1 con al menos un objeto guardado.

Adjuntar el código del archivo TypeScript del componente creado y convertir este documento en pdf. Enviar o Subir a Github.

La práctica se realizará de manera individual.

JS tareasModels.js X

EvaluableProyectoAngular > servenode > app > models > JS tareasModels.js > TareasSchema

You, hace 2 horas | 1 author (You)

```
1  const mongoose = require('mongoose');
2
3  const TareasSchema = mongoose.Schema({
4    _id: {
5      type: number,
6      required: true
7    },
8    nombre: {
9      type: String,
10     required: true
11   },
12
13   descripcion: {
14     type: String,
15     required: true
16   },
17   tareaIniciada: {
18     type: boolean,
19     required: true
20   },
21   //Puntuación
22   tareaPendiente: {
23     type: boolean,
24     required: true
25   },
26   // Estado
27   tareaFinalizada: {
28     type: boolean,
29     required: true
30   }
31 });
32
33 module.exports = mongoose.model('Tareas', TareasSchema);
```

```
Go to component
1 <!-- <form [formGroup]="crearForm" (ngSubmit)="onSubmit()" -->
2 <form (ngSubmit)="onSubmit()"
3   <div class="form-group">
4     <label for="name">Nombre</label>
5     <input type="text" formControlName="name" class="form-control">
6   </div>
7   <div class="form-group">
8     <label for="duration">Duración</label>
9     <input type="number" formControlName="duration" class="form-control">
10  </div>
11  <div class="form-group">
12    <label for="description">Descripción</label>
13    <textarea formControlName="description" class="form-control"></textarea>
14  </div>
15  <div class="form-group">
16    <label for="category">Categoría</label>
17    <input type="text" formControlName="category" class="form-control">
18  </div>
19  <div class="form-group">
20    <label for="fotoPortada">Foto de portada</label>
21    <input type="file" formControlName="fotoPortada" class="form-control">
22  </div>
23  <button type="submit" class="btn btn-primary">Crear tarea</button>
24 </form>
25
26
```

crear-tarea.component.css U X

EvaluableProyectoAngular > src > app > crear-tarea > # crear-tarea.component.css > .form-control

```
1  *{
2
3      background-color: #00214A;
4  }
5
6  form{
7      width: 500px;
8      /* margin: 0 auto; */
9      padding: 20px;
10     margin-top: 10vh;
11     margin-left: 25vw;
12     width: 50vw;
13 }
14
15
16 .form-group {
17
18     width: 20vw;
19     height: 20vh;
20     margin-bottom: 20px;
21 }
22
23 .form-group label{
24     font-weight: bold;
25     padding: 8px;
26 }
27
28 /* Form input */
29 .form-control {
30     width: 100%;
31     padding: 10px;
32     border: 1px solid #ccc;
33     border-radius: 5px;
34 }
35
36 /* Form button */
37 .btn-primary {
38     background-color: #007bff;
39     color: white;
40     border: 1px solid #007bff;
```

TS crear-tarea.component.ts X

EvaluableProyectoAngular > src > app > crear-tarea > TS crear-tarea.component.ts > CrearTareaComponent

```
1  import { Component, OnInit } from '@angular/core';
2  import { FormBuilder, FormGroup, Validators } from '@angular/forms';
3  import { GestionMensaje } from '../models/gestion-mensaje';
4
5  @Component({
6    selector: 'app-crear-tarea',
7    templateUrl: './crear-tarea.component.html',
8    styleUrls: ['./crear-tarea.component.css']
9  })
10
11  export class CrearTareaComponent implements OnInit {
12    crearForm: FormGroup; // Definir FormGroup para el formulario reactivo
13
14
15    constructor(private fb: FormBuilder, private gestionMensaje: GestionMensaje) { // Inyectar FormBuilder
16      this.crearForm = this.fb.group({
17        name: ['', Validators.required],
18        duration: ['', [Validators.required, Validators.email]],
19        description: ['', [Validators.required, Validators.minLength(20)]],
20        category: ['', [Validators.required, Validators.minLength(8)]],
21        fotoPortada: ['']
22      });
23    }
24  }
25
26  ngOnInit(): void {
27
28  }
29
30  onSubmit() {
31    const mensaje = new GestionMensaje(
32
33      this.crearForm.value.name,
34      this.crearForm.value.duration,
35      this.crearForm.value.description,
36      this.crearForm.value.category,
37      this.crearForm.value.fotoPortada
38    );
39  }
```

JS server.js X

EvaluableProyectoAngular > servenode > app > JS server.js > ...

```
16 // Conexión con Base datos
17 // mongoose.Promise = global.Promise
18 // mongoose.connect('mongodb://127.0.0.1:27017/Appweb', {
19 //   useNewUrlParser: true,
20 // })
21 // .then(() => {
22 //   console.log("Conexión establecida...")
23 // })
24 // .catch(err => console.log(err))
25
26 // Conectamos DB Cursos
27 conectarDB();
28
29 app.use(cors())
30
31 app.use(express.json());
32 You, hace 6 días • Holi ...
33 app.use('/api/usuarios', require('./app/usuariosroutes'));
34 app.use('/api/tareas', require('./app/tareasroutes'));
35 app.use('/api/lstServicios', require('lstServiciosroutes'));
36
37
38 v app.listen(3700, () => {
39   console.log('El servidor esta corriendo perfectamente')
40 })
```


JS usuariosModels.js X

EvaluableProyectoAngular > servenode > app > models > JS usuariosModels.js > ...

You, hace 2 horas | 1 author (You)

```
1 const mongoose = require('mongoose');    You, la semana pasada • holi ...
2
3 const UsuariosSchema = mongoose.Schema({
4   _id: {
5     type: String,
6     required: true
7   },
8   userName: {
9     type: String,
10    required: true
11  },
12  //Código de la tarea a realizar
13  CodeOfTheTaskToBePerformed: {
14    type: String,
15    required: true
16  },
17  DescriptionofTheTaskToBePerformed: {
18    type: String,
19    required: true
20  },
21  //Puntuación
22  Score: {
23    type: number,
24    required: true
25  },
26  // Estado
27  State: {
28    type: String,
29    required: true
30  }
31 });
32
33 module.exports = mongoose.model('Usuarios', UsuariosSchema);
```

✓ controller

JS IstServiciosController.js

JS tareasController.js

JS usuariosController.js

> models

Ejemplo:

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
import { GCurso } from '../Models/gcurso';
@Injectable({
  providedIn: 'root'
})
export class GcursoService {
  url= 'http://localhost:3700/api/cursos/'
  constructor(private http: HttpClient) { }
  getCursos(): Observable<any>{
    return this.http.get(this.url)}
  eliminarCurso(id: string): Observable<any> {
    return this.http.delete(this.url + id);}
  guardarCurso(curso: GCurso): Observable<any> {
    return this.http.post(this.url, curso);}
  obtenerCurso(id: string): Observable<any> {
    return this.http.get(this.url + id);}
  editarCurso(id: string, curso: GCurso): Observable<any>{
    return this.http.put(this.url + id, curso)}
}
```

Thank You for Downloading Vis... MySQL: Begin Your Download Descargar .NET 7.0 SDK (v7.0.40) Cursos.mdev

localhost:4200/cursos

Accesos Administración Blogging CloudMM Clientes Consulta Desarrollo web Formación Empresa Ideas Productividad Redes Sociales Varios Bookmarks Paralelizar Publicaciones Proyectos Otros marcadores

Inicio Acceso Cursos Alumnos Registro Contacto Tareas

LISTADO DE CURSOS

NUEVO

Curso	Descripción	Categoría	Duración	Temas
IFCD0210	Desarrollo de aplicaciones con tecnologías web	Nivel 3	510 horas	MF0491, MF0492, MF0493

SISTEMAS DE VALORACIÓN MF 0492_3 – UF1845 – E2

RESULTADOS A COMPROBAR	INDICADORES DE LOGRO	ESCALA DE MEDIDAS	
3. Integrar sentencias en los componentes software para acceder y manipular la información ubicada en bases de datos Conforme el criterio de evaluación CE 1.2	Integra sentencias en los componentes software para acceder a la información ubicada en bases de datos	- Integra sentencias en los componentes software para acceder a la información ubicada en bases de datos entre un 75% y 100%	B 20
		- Integra sentencias en los componentes software para acceder a la información ubicada en bases de datos entre un 50 % y 75%	R 10
		- Integra sentencias en los componentes software para acceder a la información ubicada en bases de datos por debajo de un 50 %	M 0
	Manipula la información ubicada en bases de datos	- Manipula la información ubicada en bases de datos entre un 75% y 100%.	B 20
		- Manipula la información ubicada en bases de datos entre un 50% y 75%.	R 10
		- Manipula la información ubicada en bases de datos por debajo de un 50%.	M 0
4. En un supuesto práctico en el que se pide construir componentes de software que accedan a datos soportados en bases de datos u otras estructuras de almacenamiento. Conforme el criterio de evaluación CE 1.3	2.1 Identifica los elementos y estructuras contenidas en una base de datos.	- Identifica los elementos y estructuras contenidas en una base de datos entre un 75% y 100%.	B 20
		- Identifica los elementos y estructuras contenidas en una base de datos entre un 50% y 75%.	R 10
		- Identifica los elementos y estructuras contenidas en una base de datos por debajo de un 50%.	M 0
	2.2 Utiliza los objetos, conectores y middleware necesarios en la construcción del componente para realizar los accesos a los datos soportados en la base de datos	- Utiliza los objetos, conectores y middleware necesarios en la construcción del componente entre un 75% y 100%.	B 20
		- Utiliza los objetos, conectores y middleware necesarios en la construcción del componente entre un 50% y 75%.	R 10
		- Utiliza los objetos, conectores y middleware necesarios en la construcción del componente por debajo de un 50%.	M 0
	2.3 Realiza operaciones de definición y manipulación de informaciones soportadas en bases de datos	- Realiza operaciones de definición y manipulación de informaciones soportadas en bases de datos entre un 75% y 100%.	B 20
		- Realiza operaciones de definición y manipulación de informaciones soportadas en bases de datos entre un 50% y 75%.	R 10
		- Realiza operaciones de definición y manipulación de informaciones soportadas en bases de datos por debajo de un 50%.	M 0
	Valor mínimo exigible: 50	Valor máximo: 100	