|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **NOMBRE Y APELLIDOS:**  **Miguel Ramírez Ramos** | | | | **FECHA: 25-08-2023** | | |
| **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: | | UA2 | Duración: | 3 h |
| UF: | 1844 |
| PRÁCTICA Nº: | E2 |
| DENOMINACIÓN: **Principios de la orientación a objetos** | | | | | | |
| **DESCRIPCIÓN**  1.- El alumno de forma individual deberá en base a la planificación realizada en la prueba E1 de la UF1844, realizar lo siguiente:   * Adaptar la documentación basándose en las herramientas vistas en clase (Framework), tomando como referencia la programación por componentes del FrameWork de desarrollo. * Crear el proyecto de aplicación con sus componentes definido en la documentación. * Incluir el documento creado para la planificación de la aplicación en formato .md dentro del directorio raíz del proyecto.   Pegar en este Archivo el resultado de la prueba y convertir en pdf. Enviar o Subir a Github.  La práctica se realizará de manera individual.  **MEDIOS PARA SU REALIZACIÓN**   * Equipo informático. * Aplicación Visual Code Studio instalada en el equipo. * Navegadores actualizados   **PAUTAS DE ACTUACIÓN DEL FORMADOR**  *Al inicio de la práctica, que se desarrollará de manera individual por cada uno de los alumnos, el formador/a realizará las siguientes actuaciones:*  - 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.  Durante la realización de la práctica el formador/a supervisará el desarrollo de esta para evaluar tanto los procedimientos como el resultado final.  Al finalizar la práctica el formador examinará el desarrollo que han realizado los/as alumnos/as, proponiendo las medidas de corrección, en caso necesario.  ESPECIFICACIONES PARA LA EVALUACIÓN DE LA PRÁCTICA | | | | | | |
| **Resultados a comprobar** | | | **Indicadores de logro** | | | |
| 1. Crea objetos, clases y métodos adecuados a la funcionalidad del componente software a desarrollar utilizando lenguajes de programación orientados a objetos.   Conforme el criterio de evaluación CE 1.2 | | | * 1. Crea objetos adecuados a la funcionalidad del componente software a desarrollar utilizando lenguajes de programación orientados a objetos | | | |
| * 1. Crea clases adecuadas a la funcionalidad del componente software a desarrollar utilizando lenguajes de programación orientados a objetos | | | |
| * 1. Crea métodos adecuados a la funcionalidad del componente software a desarrollar utilizando lenguajes de programación orientados a objetos | | | |

**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 en base a la planificación realizada en la prueba E1 de la UF1844, realizar lo siguiente:

* Adaptar la documentación basándose en las herramientas vistas en clase (Framework), tomando como referencia la programación por componentes del FrameWork de desarrollo.
* Crear el proyecto de aplicación con sus componentes definido en la documentación.
* Incluir el documento creado para la planificación de la aplicación en formato .md dentro del directorio raíz del proyecto.

Pegar en este Archivo el resultado de la prueba y convertir en pdf. Enviar o Subir a Github.

La práctica se realizará de manera individual.

app.component.html

<div id="containerInicio">

  <ul>

      <li><a href=""></a></li>

      <li><a routerLink="">Inicio</a></li>

      <li><a routerLink="/empleados">Empleados</a></li>

      <li><a routerLink="/jefes">Jefes</a></li>

      <li><a routerLink="/tareas">Tareas</a></li>

      <li><a routerLink="/app-formulario-acceso">Acceso</a></li>

  </ul>

</div>

<router-outlet></router-outlet>

<ng-template>

  <app-empleados></app-empleados>

  <app-jefes></app-jefes>

  <app-tareas></app-tareas>

</ng-template>

app-routing.module.ts

import { NgModule } from '@angular/core';

import { RouterModule, Routes } from '@angular/router';

import { FormsModule } from '@angular/forms';

import { AppComponent } from './app.component';

import { EmpleadosComponent } from './empleados/empleados.component';

import { JefesComponent } from './jefes/jefes.component';

import { TareasComponent } from './tareas/tareas.component';

const routes: Routes = [

  { path: '', component: AppComponent },

  { path: 'empleados', component: EmpleadosComponent },

  { path: 'jefes', component: JefesComponent },

  { path: 'tareas', component: TareasComponent },

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

app.component.css

\*{

    padding: 25px;

}

#containerInicio {

    background-color: lightcoral;

}

Modelo empleados.ts

export class empleadosBD{

    constructor(

        public nombreE: String,

        public apellidoE: String,

        public nombreDepartamentoE: String,

        public anioEnLaEmpresaE: number,

        public salarioE: number,

        public correoE: String,

        public telefonoE: String,

        public fotoE: String,

        public estadoE: boolean

    ){}

}

empleados.ts

import { Component, OnInit } from '@angular/core';

import { empleadosBD } from '../models/empleados';

@Component({

  selector: 'app-empleados',

  templateUrl: './empleados.component.html',

  styleUrls: ['./empleados.component.css']

})

export class EmpleadosComponent implements OnInit{

  public empleadosAll: empleadosBD[];

  public nombreEmpleados: string;

  public apellidoEmpleados: string;

  public nombreDepartamento: string;

  public estadoDeLaTarea: boolean;

  constructor(){

    this.nombreEmpleados="";

    this.apellidoEmpleados="";

    this.nombreDepartamento= "";

    this.estadoDeLaTarea= false;

    this.empleadosAll=[]

  }

  ngOnInit(){

    console.log("OnInit Ejecutado")

    this.inicializarEmpleados();

  }

  private inicializarEmpleados(){

    this.empleadosAll =[

      new empleadosBD( "Juan", "Pérez", "Ventas", 3, 5000, "juan.perez@example.com", "123456789", "ruta/a/la/foto.jpg", true),

      new empleadosBD( "María", "González","Recursos Humanos", 5, 6000, "maria.gonzalez@example.com", "987654321", "ruta/a/la/foto.jpg", true),

      new empleadosBD( "Ana", "López","Marketing", 2, 4500, "ana.lopez@example.com", "555555555", "ruta/a/la/foto.jpg", true),

      new empleadosBD( "Pedro", "Ramírez","Desarrollo de Productos", 4, 7000, "pedro.ramirez@example.com", "999999999", "ruta/a/la/foto.jpg", false),

    ];

  }

}

empleados.html

<div id="empleadosContainer">

    <div class="empleadosFicha">

        <h1 class="empleadosEnunciado">Ficha de los Jefes de la Empresa</h1>

        <div \*ngFor="let empleados of empleadosAll">

            <h3 class="nombresJefes">{{ empleados.nombre }} {{ empleados.apellidos}}</h3>

            <p class="atributosJefes">Nombre Departamento: {{ empleados.nombreDepartamento }}</p>

            <p class="atributosJefes">Estado De la Tarea: {{empleados.estadoDeLaTarea}}</p>

        </div>

    </div>

</div>

# Ejemplo:

* Interacción del usuario:
  + A través de botones en la parte superior – derecha:
    - Botón 1 – Acceder
    - .
    - .
    - .
  + A través de formularios de pedidos donde se recogerán los siguientes datos
    - Formulario de pedido
      * Nombre
      * Correo electrónico
      * Teléfono
    - Formulario de consulta
    - .
    - .
    - .

**SISTEMAS DE VALORACIÓN MF 0492\_3 – UF1844 – E2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **RESULTADOS A COMPROBAR** | **INDICADORES DE LOGRO** | **ESCALA DE MEDIDAS** | |  |
| 1. Crea objetos, clases y métodos adecuados a la funcionalidad del componente software a desarrollar utilizando lenguajes de programación orientados a objetos.   Conforme el criterio de evaluación CE 1.2 | Crea objetos adecuados a la funcionalidad del componente software a desarrollar utilizando lenguajes de programación orientados a objetos | - Crea objetos adecuados a la funcionalidad del componente software entre un 75% y 100%  - Crea objetos adecuados a la funcionalidad del componente software entre un 50 % y 75%  - Crea objetos adecuados a la funcionalidad del componente software por debajo de un 50 % | B  R  M | 40  20  0 |
| Crea clases adecuadas a la funcionalidad del componente software a desarrollar utilizando lenguajes de programación orientados a objetos | - Crea clases adecuadas a la funcionalidad del componente software entre un 75% y 100%.  - Crea clases adecuadas a la funcionalidad del componente software entre un 50% y 75%.  - Crea clases adecuadas a la funcionalidad del componente software por debajo de un 50%. | B  R  M | 30  15  0 |
| Crea métodos adecuados a la funcionalidad del componente software a desarrollar utilizando lenguajes de programación orientados a objetos | - Crea métodos adecuados a la funcionalidad del componente software entre un 75% y 100%.  - Crea métodos adecuados a la funcionalidad del componente software entre un 50% y 75%.  - Crea métodos adecuados a la funcionalidad del componente software por debajo de un 50%. | B  R  M | 30  15  0 |
|  | **Valor mínimo exigible: 50** | **Valor máximo: 100** | |  |