

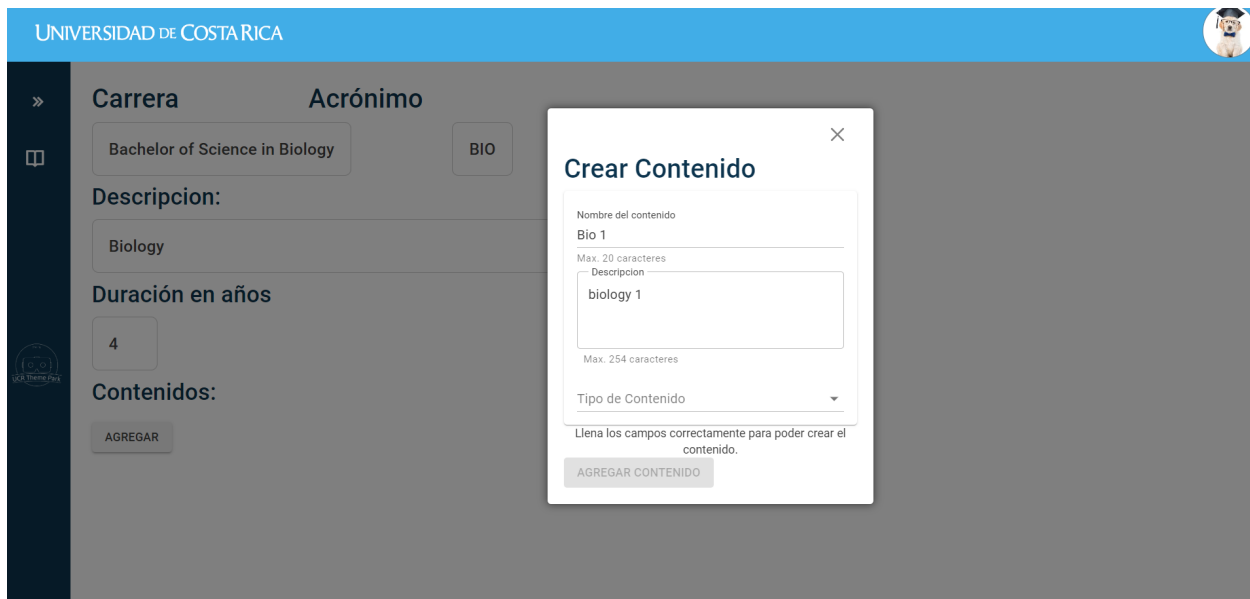
Evaluación técnica 2.

Entregables

Se entregara un archivo editable tipo google doc/word con las respuestas de la parte escrita, ademas de eso se entregara un pdf de lo mismo como respaldo. Para la parte programada se entregara un archivo de texto con el link del repositorio de github que se uso para la realización del proyecto, ademas de mi auto-evaluación.

1. Crear los proyectos para la aplicación en la solución. Programar la funcionalidad necesaria de la aplicación en el branch devCARNET. Realice commits al branch devCARNET e integre cada vez que tenga parte de la funcionalidad finalizada y funcionando al branch main, lo que implica realizar los procesos de integración. Aplique todas las buenas prácticas de la ingeniería de software y las recomendaciones de la ET1.

[x] Agregar contenidos a una carrera (5%), tarea realizada en frontend y backend, se puede agregar un contenido a una carrera existente o crear una carrera nueva y agregarle un contenido.



The screenshot displays the 'UNIVERSIDAD DE COSTARICA' web application interface. On the left, a sidebar contains navigation icons. The main content area shows a form for adding content to a career. The form has two tabs: 'Carrera' and 'Acrónimo'. Under 'Carrera', there is a text input field containing 'Bachelor of Science in Biology' and a button labeled 'BIO'. Below this, the 'Descripción:' section has a text input field with 'Biology'. The 'Duración en años' section has a text input field with '4'. The 'Contenidos:' section has a button labeled 'AGREGAR'. A modal window titled 'Crear Contenido' is open in the center, featuring a close button (X) in the top right corner. The modal contains a 'Nombre del contenido' text input field with 'Bio 1', a 'Max. 20 caracteres' label, a 'Descripción' text input field with 'biology 1', a 'Max. 254 caracteres' label, and a 'Tipo de Contenido' dropdown menu. At the bottom of the modal, there is a message 'Llena los campos correctamente para poder crear el contenido.' and a button labeled 'AGREGAR CONTENIDO'.

Agregar contenido a una carrera

[x] Buscar una carrera por nombre (5%), tarea realizada en frontend, al estar en la pagina de listar carreras se puede buscar una carrera por su nombre en la barra de busqueda que se encuentra en la parte superior de la pagina.

UNIVERSIDAD DE COSTA RICA				
Carreras				
Search				
Nombre	Acronimo	Duracion	Presupuesto de Beca	
Bachelor of Science in Biology	BIO	4	0	+
Bachelor of Science in Computer Science	CS	4	0	+
Bachelor of Science in Engineering	ENG	5	0	+
Doctor of Medicine	MED	7	0	+
1-4 of 4 < < > >				

Buscar carrera por nombre

[x] CRUDL de contenidos de una carrera (5%), al encontrarse en la pagina de mostrar carrera se pueden agregar, editar y eliminar contenidos de la carrera, ademas del listado visual de los contenidos.

UNIVERSIDAD DE COSTA RICA	
»	Carrera Acrónimo
□	<div>Bachelor of Science in Biology</div> <div>BIO</div>
	Descripcion: <div>Biology</div>
	Duración en años <div>4</div>
	Contenidos: <div>AGREGAR</div> <div> <div>Bio 1</div> <div>biology 1</div> <div>tecnologías</div> </div>

Listar de contenidos de una carrera

UNIVERSIDAD DE COSTA RICA

»

Carrera

Acrónimo

Bachelor of Science in Biology

BIO

Descripción:

Biology

Duración en años

4

Contenidos:

AGREGAR

Bio 1

biology 1

Modificar Contenido

Nombre del contenido

Bio 2

Max. 20 caracteres

Descripción

biology 2

Max. 254 caracteres

Tipo de Contenido

Tecnológico

Llena los campos correctamente para poder modificar el contenido.

MODIFICAR CONTENIDO

Editar contenido de una carrera

Como se puede apreciar, al crear un contenido se puede seleccionar el tipo de contenido que se esta creando, y al editar un contenido se puede modificar el tipo de contenido que se esta editando, de misma manera, si se modifica se ira actualizando el budget de la carrera en tiempo real.

Crear Contenido

Nombre del contenido

Bio 1

Max. 20 caracteres

Descripcion

biology 1

Max. 254 caracteres

Tecnológico

Ambiental


Social

Otros

Tipos de contenido de una carrera


[x] Mostrar información de una carrera con su presupuesto de becas (30%), gracias a un nuevo servicio implementado en backend, y en frontend, se puede apreciar el budget de cada carrera al listarlas. Ese budget es dinamico, es decir que si agrego o quito nuevo contenido a una carrera, el budget se actualiza automaticamente.

UNIVERSIDAD DE COSTA RICA



»

☰



Carreras

🔍 Search

Nombre	Acronimo	Duracion	Presupuesto de Beca	
Bachelor of Science in Biology	BIO	4	0	+
Bachelor of Science in Computer Science	CS	4	0	+
Bachelor of Science in Engineering	ENG	5	0	+
Doctor of Medicine	MED	7	0	+

1-4 of 4 |< < > >|

Listar carreras con budget

Por ejemplo, si se le agrega un contenido de tipo tecnico a la carrera de Ingenieria en Computacion, el budget de becas de esa carrera se actualizara automaticamente.

Nombre	Acronimo	Duracion	Presupuesto de Beca	
Bachelor of Science in Biology	BIO	4	240	+
Bachelor of Science in Computer Science	CS	4	0	+
Bachelor of Science in Engineering	ENG	5	0	+
Doctor of Medicine	MED	7	0	+

1-4 of 4 |< < > >|

Listar carreras con budget

[x] Mostrar información de una carrera con sus contenidos (5%), al estar en la pagina de mostrar carrera se puede ver la descripcion de la carrera y los contenidos asociados a esta, ademas de eso se agregaron en esa misma pagina las opciones para modificar y eliminar los contenidos de la carrera. Al introducir el concepto de tipo de contenido, se puede ver el tipo de contenido que se esta mostrando en la pagina de mostrar carrera.

Pruebas unitarias de la capa de dominio

[x] Infrastructure

[-] UCR.ECCLIS.TECH_EVALUATION_1.Infrastructure	156	132	288	566	54.1%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Infrastructure.ApplicationDbContext	3	16	19	52	15.7%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Infrastructure.Career.EntityConfigurations.CareerEntityConfiguration	0	36	36	62	0%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Infrastructure.Career.Repositories.SqlCareerRepository	153	38	191	356	80.1%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Infrastructure.InfrastructureLayerDependencyInjection	0	6	6	31	0%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Infrastructure.career.EntityConfigurations.ContentsEntityConfiguration	0	36	36	65	0%	<div><div></div></div>
+ UCR.ECCLIS.TECH_EVALUATION_1.Infrastructure.Tests.Unit	309	0	309	529	100%	<div><div></div></div>

Pruebas unitarias de la capa de infraestructura

[x] Application

Name	Covered	Uncovered	Coverable	Total		Line coverage
[-] UCR.ECCLIS.TECH_EVALUATION_1.Application	49	7	56	203	87.5%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Application.ApplicationLayerDependencyInjection	0	7	7	21	0%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Application.Services.Classes.CreateCareerService	12	0	12	39	100%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Application.Services.Classes.CreateContentsService	16	0	16	44	100%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Application.Services.Classes.DeleteContentService	7	0	7	38	100%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Application.Services.Classes.ListCareerService	7	0	7	30	100%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Application.Services.Classes.ListContentsService	7	0	7	31	100%	<div><div></div></div>

Pruebas unitarias de la capa de aplicacion

Por cuestiones de tiempo solo se pudieron realizar pruebas de integracion de la capa de infraestructura con la base de datos, pero visto de un punto de vista de ingenieria de software, es posiblemente la capa mas importante para realizar pruebas de integracion, ya que es la capa que se encarga de la comunicacion entre la base de datos y el resto del sistema, y por lo tanto es la capa que mas errores puede tener.

[-] UCR.ECCLIS.TECH_EVALUATION_1.Infrastructure	221	67	288	566	76.2%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Infrastructure.ApplicationDbContext	9	10	19	52	47.3%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Infrastructure.Career.EntityConfigurations.CareerEntityConfiguration	36	0	36	62	100%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Infrastructure.Career.Repositories.SqlCareerRepository	140	31	191	356	73.2%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Infrastructure.InfrastructureLayerDependencyInjection	0	6	6	31	0%	<div><div></div></div>
UCR.ECCLIS.TECH_EVALUATION_1.Infrastructure.career.EntityConfigurations.ContentsEntityConfiguration	36	0	36	65	100%	<div><div></div></div>

Pruebas de integracion de la capa de infraestructura

En general se logro una cobertura de pruebas muy alta, pero en algunos casos no se logro una cobertura del 100%. Pero se pudieron probar muchos de los casos de uso del sistema.

3. Aseguramiento de la Calidad. Investigue e instale una extensión|paquete en VS que le permita analizar el código (están en el video de laboratorio: Sonarlint, SonarAnalyzer.CSharp, SonarAnalyzer.CSharp.Styling). Analice los resultados del análisis y realice y justifique, al menos, tres mejoras significativas en el código (10%).

Para realizar lo anterior se instalaron las herramientas propuestas en el video y se usaron a lo largo del desarrollo para mejorar la calidad del código. Las principales mejoras que se realizaron fueron:

- Se disminuyo la cantidad de nuevos warnings, como el problema mencionado en el PI, donde muchos warnings "antiguos" se solucionaron, de igual manera se mantuvo una cantidad baja de warnings de 18.

▲ CS1908	This async method lacks 'await' operators and will run synchronously. Consider using the 'await' operator to await non-blocking API calls, or 'await Task.Run(...)' to do CPU-bound work on a background thread.	UCRECCJUS.TECH_EVALUATION_1.Infrastructure	SqlCareerRepository.cs	101	Active
▲ CS9002	Dereference of a possibly null reference.	UCRECCJUS.TECH_EVALUATION_1.Infrastructure	SqlCareerRepository.cs	244	Active
▲ CS8618	Non-nullable property 'Acronym' must contain a non-null value when exiting constructor. Consider declaring the property as nullable.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	CreateCareer.razor	140	Active
▲ CS8618	Non-nullable property 'Description' must contain a non-null value when exiting constructor. Consider declaring the property as nullable.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	CreateCareer.razor	143	Active
▲ CS8618	Non-nullable property 'Name' must contain a non-null value when exiting constructor. Consider declaring the property as nullable.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	CreateCareer.razor	166	Active
▲ CS8618	Non-nullable field 'career' must contain a non-null value when exiting constructor. Consider declaring the field as nullable.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	ViewCareer.razor	113	Active
▲ CS8618	Non-nullable property 'inbox' must contain a non-null value when exiting constructor. Consider declaring the property as nullable.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	ViewCareer.razor	116	Active
▲ CS8618	Non-nullable field 'content_rephased' must contain a non-null value when exiting constructor. Consider declaring the field as nullable.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	ViewCareer.razor	120	Active
▲ CS9169	The field 'ViewCareer.content_rephased' is never used	UCRECCJUS.TECH_EVALUATION_1.Presentation...	ViewCareer.razor	120	Active
▲ CS9601	Possible null reference assignment.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	ViewCareer.razor	135	Active
▲ CS9602	Dereference of a possibly null reference.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	ViewCareer.razor	136	Active
▲ CS8618	Non-nullable property 'ContentName' must contain a non-null value when exiting constructor. Consider declaring the property as nullable.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	ViewCareer.razor	166	Active
▲ CS8618	Non-nullable property 'Description' must contain a non-null value when exiting constructor. Consider declaring the property as nullable.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	ViewCareer.razor	170	Active
▲ CS8618	Non-nullable property 'CareerAcronym' must contain a non-null value when exiting constructor. Consider declaring the property as nullable.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	ViewCareer.razor	172	Active
▲ CS8618	Non-nullable property 'ContentType' must contain a non-null value when exiting constructor. Consider declaring the property as nullable.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	ViewCareer.razor	175	Active
▲ CS8618	Non-nullable field 'contentRephase' must contain a non-null value when exiting constructor. Consider declaring the field as nullable.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	ViewCareer.razor	182	Active
▲ CS8618	Non-nullable field 'description' must contain a non-null value when exiting constructor. Consider declaring the field as nullable.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	ViewCareer.razor	183	Active
▲ CS8618	Non-nullable property 'CareerAcronym' must contain a non-null value when exiting constructor. Consider declaring the property as nullable.	UCRECCJUS.TECH_EVALUATION_1.Presentation...	ViewCareer.razor	223	Active

Warnings

- Se mantuvo una cantidad baja de código comentado, ya que es considerado como un code smell. Antes de esa evaluación se tendía a comentar mucho el código, para "tenerlo a mano", pero se aprendió que es mejor tener un código limpio y fácil de leer, que tener un código con muchos comentarios que no aportan mucho al proceso de desarrollo.
- Se mejoró la cobertura de pruebas, ya que al inicio teníamos una cobertura de pruebas de 0%, y al final se logró una cobertura de pruebas de 70-100% en las distintas capas del sistema. Aunque siento que existe bastante margen de mejora en este aspecto para proteger al sistema de posibles errores en el futuro. Como por ejemplo pruebas de posible "nullable" en algunas clases, pero en general siento que se logró un buen trabajo en este aspecto.

4. Extra. Investigue e instale una extensión|paquete|herramienta en VS que le permita calcular la cobertura de pruebas detalladas por capa y clases. Detalle el reporte de cobertura y analice los resultados y siguientes pasos (10%).

Como se mencionó anteriormente, se utilizó la extensión de VS Studio "fine code coverage", la cual nos permite ver el porcentaje de cobertura de las pruebas realizadas. Muchas de las pruebas realizadas pudieron alcanzar una cobertura "acceptable" por la industria, es decir entre 70% y 100%, pero como ya mencionado, dudo de la veracidad de la herramienta, ya que en muchas clases se alcanzó una cobertura del 100%, lo cual debería ser complicado de lograr.

Una de las pruebas más interesantes y que mostro el mayor challenge fueron las pruebas de unidad en la capa de infraestructura, ya que algunas de las funciones tenían muchos condicionales y era difícil de probar todos los casos posibles. Pero en general se alcanzó una cobertura alta de las funciones, con principal "debilidad" las funciones con muchos condicionales y con throw exceptions que encontré complicado de provocar.

Evaluación técnica 1.

Enunciado & rubros de puntaje

Parte 1. Fundamentos de la ingeniería de software, análisis y discusión (9%). Una página para la respuesta de cada pregunta.

- 1 Como ingeniero/ingeniera de software. El equipo de desarrollo al que usted pertenece en su organización utiliza la metodología Scrum como la metodología para el desarrollo de software. Actualmente la gerencia técnica está planeando cambiar la metodología a XP. Describa en detalle un caso a favor de continuar con la metodología Scrum, y un caso a favor de cambiar a la metodología XP. Para su respuesta considere los fundamentos de cada metodología, las similitudes y las diferencias, los roles, sesiones y artefactos, y las reglas del proceso de cada uno de ellos. Finalmente, de una recomendación justificada técnicamente sobre la selección de la metodología a la gerencia técnica de su organización (4.5%).

- 2 Como ingeniero/ingeniera de software. El equipo de desarrollo al que usted pertenece en su organización no ha utilizado hasta el momento arquitecturas limpias, código limpio y principios SOLID en el desarrollo de las aplicaciones. Explique los principales beneficios (con ejemplos) de cada una de estas prácticas e indique cuál es su recomendación para implementar la utilización de estas como parte de su proceso de desarrollo. Sea puntual mencionando cada concepto, las ventajas que ofrece y porqué (con ejemplos) (4.5%).

En mediación: adjunte los artefactos/documentos desarrollados en formato editable y PDF **No subir archivos comprimidos**. El subir un archivo comprimido representa un cero en la nota respectiva.

Parte 2. Requerimientos (15%).

Importante: La parte 2 y 3 se desarrolla a partir del siguiente caso de estudio.

Caso de estudio.

La Universidad ha empezado a implementar un Campus de Realidad Virtual (VR), recientemente cuatro equipos de desarrollo han trabajado en funcionalidades para ofrecer interacción a través de edificios y aulas/laboratorios virtuales. La universidad ha determinado que para atraer personas jóvenes que buscan seleccionar una carrera profesional, debe apoyar las ferias vocacionales que se realizan regularmente con tecnologías de VR. Por tanto, se requiere que se trabaje en funcionalidades que permitan ofrecer funcionalidades para las distintas carreras universitarias, y que sean utilizadas en las ferias vocacionales. En primera instancia se requiere que las personas puedan ingresar a una carrera para obtener información novedosa de la misma. El sistema debe permitir buscar y seleccionar una carrera, y ofrecer información básica de la misma, empezando por la aplicación web para las personas administradoras de las ferias.

De acuerdo con lo descrito en el caso de estudio realice los siguientes ejercicios. Para cada uno de ellos usted como ingeniero/ingeniera de software debe tomar decisiones sobre los flujos recomendados para cada funcionalidad:

- 1 Especifique al menos un tema, un epic y tres historias de usuario debidamente documentadas sobre cualquiera de las funcionalidades del sistema (seleccionadas por prioridad según su criterio profesional). Describa al menos dos criterios de aceptación por historia de usuario (uno funcional y uno no funcional). Justifique el cumplimiento de los criterios INVEST para cada una de las historias (8%).
- 2 Especifique tres requerimientos no funcionales prioritarios para el sistema, clasifíquelos y cuantifíquelos. Justifique la selección (7%).

En mediación: adjunte los artefactos/documentos desarrollados en PDF. No subir archivos comprimidos. El subir un archivo comprimido representa un cero en la nota respectiva.

Parte 3. Repositorios y herramientas de desarrollo (70%).

La parte 3 se desarrolla a partir del caso de estudio anterior y la especificación adicional del sistema que se provee a continuación.

Utilice el siguiente link de GitHub Classroom para crear su repositorio de trabajo:
<https://classroom.github.com/a/PFe5gw02>

Continuación del caso de estudio.

La universidad requiere que la aplicación tenga la información de las carreras en una base de datos relacional:

- Carreras que se ofrecen.
- Contenidos que se muestran por cada carrera. Cada carrera puede tener uno más contenidos.

Requerimientos prioritarios para la aplicación:

Desarrolle una aplicación que permita realizar las operaciones básicas prioritarias (mínimo producto viable) para:

- Agregar contenidos a una carrera.

- Agregar una carrera.
- Buscar una carrera por nombre.
- Mostrar información de una carrera.

Debe ofrecer la funcionalidad como servicios web / backend y página web, que en un futuro puedan ser utilizados por un ambiente de realidad virtual.

Para desarrollar la aplicación utilice las tecnologías utilizadas en el curso hasta el momento (C#, SQL Server, EF Core, Web API, Blazor y otros). Es obligatorio el uso de repositorios de código y las buenas prácticas de la ingeniería del software aprendidas en el curso, por ejemplo: principios de la metodología XP y Scrum [incluyendo la programación en parejas], principios de arquitecturas limpias y sus reglas de dependencia, cada uno de los principios SOLID y principios de código limpio, sin limitarse a estos. Todo lo anterior debe ser considerado cuando diseñe e implemente la solución. La evaluación se realiza utilizando la técnica de programación en parejas.

Debe definir los requisitos mínimos que debe cumplir para decidir si el producto está listo (DoD, por ejemplo, los del proyecto del Sprint en ejecución). Desarrolle solo el mínimo producto viable de la funcionalidad prioritaria indicada. Se espera una solución limpia, respecto al código y la arquitectura, que considere el cumplimiento de los principios SOLID.

El producto y proceso se evalúa de acuerdo con: la completitud del desarrollo del proyecto de datos, la completitud del desarrollo del proyecto y su solución limpia, las buenas prácticas de ingeniería de software que incluyen la calidad del proceso y del producto, el valor de negocio que provee y si es un producto es potencialmente entregable. Los docentes consideran cada uno de estos aspectos para la asignación de la nota de cada parte que se detalla a continuación.

Se espera que realice las siguientes actividades en orden:

- 1 Utilizar un repositorio para la administración del código del proyecto de datos y del código de la aplicación (solución y sus proyectos). Configurar el archivo gitignore. (4%).
- 2 Crear la solución y el proyecto de datos en el main del repositorio y clonar en un branch con el nombre devCARNET. Programar lo necesario para administrar los datos de su aplicación en una base de datos SQL Server en el branch devCARNET. Realizar commits al branch devCARNET e integre cada vez que tenga parte de la funcionalidad finalizada y funcionando al branch main, lo que implica realizar los procesos de integración (10%).
- 3 Crear los proyectos para la aplicación en la solución (frontend y backend). Programar la funcionalidad necesaria de la aplicación en el branch devCARNET. Realice commits al branch devCARNET e integre cada vez que tenga parte de la funcionalidad finalizada y funcionando al branch main, lo que implica realizar los procesos de integración. Aplique todas las buenas prácticas de la ingeniería de software (30%).
- 4 Análisis y discusión. Basado en los principios de arquitecturas limpias y SOLID, y la funcionalidad de la aplicación web desarrollada, realice un análisis donde indique que decisiones de diseño recomienda para cumplir cada uno de los principios y la arquitectura limpia. Para esto justifique como cumplir las reglas de dependencia y cada uno de los principios SOLID según la implementación de la solución y ejemplificando sus argumentos. El análisis debe agregarlo a un archivo readme en el directorio raíz del repositorio (.md) (12%).

Indique en el archivo readme qué partes logró entregar y cuáles no. Indique si está entregando versiones funcionales. Si no cumplió con lo requerido, describa los principales problemas que se le presentaron y cómo los resolvería en una siguiente iteración. Antes de finalizar la evaluación debe hacer los commits and push de lo realizado, para asegurar que la solución está en el repositorio remoto.

En mediación: adjunte el link al repositorio.

Coevaluación: realice la distribución del 100% de la nota entre las personas participantes de acuerdo con su contribución en la evaluación.

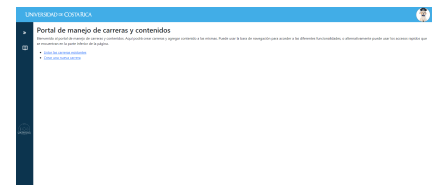
Guía de uso de la aplicación

Nuestra aplicación propone cuatro servicios web que permiten realizar las operaciones básicas prioritarias para la gestión de carreras y contenidos. Estos servicios son:

- Agregar contenidos a una carrera.
- Agregar una carrera.
- Listar carreras.
- Mostrar información de una carrera.

La aplicación para proponer estos servicios usa una multitud de paginas tales como:

- la pagina principal, propone un acceso rapido a listar y crear carreras:



Pagina principal

- la pagina de crear carrera, permite agregar una nueva carrera a la base de datos:
Pagina de crear carrera
- la pagina de listar carreras, permite ver todas las carreras en la base de datos:
Pagina de listar carreras
- la pagina de mostrar carrera, permite ver los detalles de una carrera en particular, siendo la descripcion y los contenidos asociados a esta:
Pagina de mostrar carrera
- dentro de la pagina anteriormente mencionada se puede crear un nuevo componente a la dada carrera:

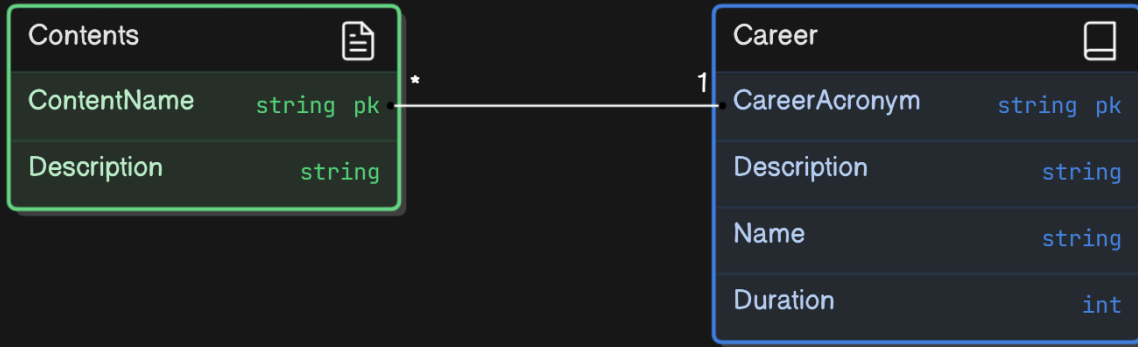
Pagina de crear contenido

Diseño de la base de datos

La base de datos esta compuesta de una total de 2 tablas, las cuales son:

- Carreras
- Contenidos Y siguen la siguiente estructura de entidades y relaciones:

Career and Content Relationship



Estructura de la base de datos