UNIVERSIDAD NACIONAL AUTÓNOMA DE NICARAGUA
UNAN – LEÓN
FACULTAD DE CIENCIAS Y TECNOLOGÍA
INGENIERÍA EN SISTEMAS DE LA INFORMACIÓN

AÑO LECTIVO: 2025
SEMESTRE: II
Proyecto Final (Avances)

Componente Curricular: # PROGRAMACION ORIENTADA A LA WEB II

Grupo: GP1
Profesor(a): Lic. Juan Carlos Leyton Briones
Autor:

1. Francisco José Jarquín Briceño.

León, Nicaragua, 10 de septiembre de 2025.

"¡A la Libertad por la Universidad!"

Introducción

Esta aplicación es una aplicación para llenar una encuesta y poder visualizar las respuestas que han sido enviadas.

Aplicación

La aplicación es una página web SPA que nos permite iniciar sesión ya sea como un encuestado (respondent), o como el administrador (admin), llenar una encuesta y enviar nuestras respuestas, las cuales se guardan en una base de datos.

Esta aplicación consta de todos los elementos por defecto al crear un nuevo proyecto usando la plantilla de React y ASP.NET Core que se encuentra dentro de Visual Studio (habiendo instalado el paquete de desarrollo web).

En esta aplicación se ha creado una API en el backend, el cual usa ASP.NET Core para ello, esta API se encarga de comunicarse con la base de datos para realizar operaciones CRUD en ella, para esto se ha instalado un conector de .NET con MySQL proporcionado por Devart, el cual tiene funcionalidad ORM, permitiéndonos trabajar con la BD usando clases y métodos; estos datos son enviados al frontend hecho en React para poder mostrárselos al usuario.

Aparte de esto, se han agregado los siguientes archivos al proyecto:

Backend

`BackendController.cs`

Es un controlador que posee los siguientes elementos:

- `_context`: Es una variable que contiene el contexto de la base de datos. Es del tipo `ProyectoEncuestaContext`.

- `BackendController()`: Constructor del controlador en cuestión, durante este momento se inicializa el contexto de la BD.

- `Ping()`: Es un método `HTTP GET` que simplemente nos devuelve una respuesta `HTTP OK`. Su ruta es “`/api/ping`”.

- `Register(user)`: Es un método `HTTP POST` que registra al usuario proporcionado, guardando sus datos en las tablas `Respondents` y `Auth` de la BD. Nos retorna una respuesta `HTTP Created`. Su ruta es "`/api/register`".

- `LogIn(user)`: Es un método `HTTP POST` que extrae los datos del usuario del cual se desea crear una sesión. Retorna un JSON con el estado de la operación ("ok" o "error"), y su rol ("admin", "respondent", o "null" en caso de error). Su ruta es "`/api/login`".

- `Survey()`: Es un método `HTTP GET` que extrae los datos de la encuesta a responder. Retorna un JSON con los datos de la encuesta, como el título de la encuesta, preguntas con sus posibles respuestas, etc. Su ruta es "`/api/survey`".

- `SubmitAnswers(form)`: Es un método `HTTP POST` que guarda nuestras respuestas en la BD. Retorna una respuesta `HTTP Created`. Su ruta es "`/api/submit-answers`".

- `GetAnswers()`: Es un método `HTTP GET` que extrae las respuestas que se han enviado de la encuesta de la BD. Retorna un JSON con los datos de las respuestas. Su ruta es "`/api/get-answers`".

Script de la base de datos:

```
drop database if exists Proyecto_Encuesta;
create database Proyecto_Encuesta;
use Proyecto_Encuesta;


-- Cada encuesta
create table Surveys (
    ID int auto_increment,
    Title varchar(50),
    Description varchar(200),
    Is_anonymous bool,
    Status varchar(50),
    Opens_at datetime,
    Closes_at datetime,
    Created_at datetime,

    constraint Surveys_PK primary key(ID)
);
```

```
-- Cada pregunta de la encuesta
create table Questions (
    ID int auto_increment,
    Survey_ID int,
    Position int,
    Text varchar(200),
    Type varchar(50),
    Is_required bool,
    Min_value float,
    Max_value float,

    constraint Questions_PK primary key(ID),

    -- 1 encuesta puede tener muchas preguntas, 1 pregunta solamente
pertenece a 1 encuesta
    constraint Questions_FK1 foreign key(Survey_ID) references
Surveys(ID)
    on update cascade on delete cascade
);

-- Cada persona que responde la encuesta
create table Respondents (
    ID int auto_increment,
    External_ID varchar(50),
    Name varchar(100),
    Email varchar(254),
    Created_at datetime,

    constraint Respondents_PK primary key(ID)
);

-- Cada envio como tal, que se realiza al finalizar la encuesta
create table Submissions (
    ID int auto_increment,
    Survey_ID int,
    Respondent_ID int,
    Started_at datetime,
    Submitted_at datetime,

    constraint Submissions_PK primary key(ID),
```

```
    -- 1 encuesta puede tener varios envios, 1 envio solamente tiene 1
encuesta
    constraint Submissions_FK1 foreign key(Survey_ID) references
Surveys(ID)
    on update cascade on delete cascade,

    -- 1 usuario puede tener varios envios, 1 envio solamente tiene 1
usuario
    constraint Submissions_FK2 foreign key(Respondent_ID) references
Respondents(ID)
    on update cascade on delete cascade
);

-- El banco de opciones o posibles respuestas
create table Choices (
    ID int auto_increment,
    Question_ID int,
    Position int,
    Label varchar(100),
    Value varchar(100),

    constraint Choices_PK primary key(ID),

    -- 1 pregunta puede tener varias opciones, 1 opcion solamente
tiene 1 pregunta
    constraint Choices_FK1 foreign key(Question_ID) references
Questions(ID)
    on update cascade on delete cascade
);

-- Las respuestas que el encuestado da como tal
create table Answers (
    ID int auto_increment,
    Submission_ID int,
    Question_ID int,
    Answer_text varchar(100),
    Answer_number float,
    Answer_date datetime,
    Selected_Choice_ID int,

    constraint Answers_PK primary key(ID),
```

```
    -- 1 envio puede tener varias respuestas, 1 respuesta solamente
tiene 1 envio
    constraint Answers_FK1 foreign key(Submission_ID) references
Submissions(ID)
    on update cascade on delete cascade,

    -- 1 pregunta puede tener varias respuestas, 1 respuesta solamente
tiene 1 pregunta
    constraint Answers_FK2 foreign key(Question_ID) references
Questions(ID)
    on update cascade on delete cascade,

    -- 1 opcion puede estar seleccionada por varias respuestas, 1
respuesta solamente tiene asociada 1 opcion
    constraint Answers_FK3 foreign key(Selected_Choice_ID) references
Choices(ID)
    on update cascade on delete cascade
);

-- Tabla para unir Answers y Choices en una relacion muchos a muchos
(N:N)
create table Answers_Choices (
    Answer_ID int,
    Choice_ID int,

    constraint Answers_Choices_PK primary key(Answer_ID, Choice_ID),

    constraint Answers_Choices_FK1 foreign key(Answer_ID) references
Answers(ID)
    on update cascade on delete cascade,
    constraint Answers_Choices_FK2 foreign key(Choice_ID) references
Choices(ID)
    on update cascade on delete cascade
);

/**/
/* Added at a later date */
/**/
-- Asegurar unicidad de correo para autenticación
alter table Respondents add unique key uq_respondents_email (Email);

-- Tabla de credenciales/roles vinculada a Respondents
```

```
create table if not exists Auth (
    Respondent_ID int primary key,
    Password_Hash varchar(255) not null,
    Role enum('admin','respondent') default 'respondent',
    Created_at datetime default current_timestamp,
    constraint Auth_FK1 foreign key (Respondent_ID) references
Respondents(ID)
        on update cascade on delete cascade
);

-- Defaults útiles
alter table Surveys modify Created_at datetime default
current_timestamp;
alter table Respondents modify Created_at datetime default
current_timestamp;
alter table Submissions modify Started_at datetime default
current_timestamp;



-- Inserción de datos
insert into Respondents values
    (NULL, NULL, 'Liam', 'liam@gmail.com', current_timestamp()),
    (NULL, NULL, 'Francisco', 'francisco@gmail.com',
current_timestamp());

insert into Auth values
    ((select R.ID from Respondents as R where R.Name like 'Liam'),
'123qwe', 'admin', current_timestamp()),
    ((select R.ID from Respondents as R where R.Name like
'Francisco'), '123qwe', 'respondent', current_timestamp());



-- Agregar una encuesta y preguntas
-- 1. Insertar la encuesta
INSERT INTO Surveys
  (Title,
   Description,
   Is_anonymous,
   Status,
   Opens_at,
   Closes_at,
```

```
   Created_at)
VALUES
  ('Encuesta sobre uso y percepción de la IA',
   'Objetivo: Conocer las percepciones, experiencias y preocupaciones
de las personas sobre el uso excesivo de la IA en la vida cotidiana y
el trabajo.',
   TRUE,
   'open',
   '2025-09-01 00:00:00',
   '2025-11-28 23:59:59',
   '2025-08-25 10:00:00');


-- 2. Insertar las preguntas (Survey_ID = 1)
INSERT INTO Questions
  (Survey_ID,
   Position,
   Text,
   Type,
   Is_required,
   Min_value,
   Max_value)
VALUES
  (1, 1, 'Edad:', 'single', TRUE, NULL, NULL),
  (1, 2, 'Ocupación:', 'single', TRUE, NULL, NULL),
  (1, 3, '¿Con qué frecuencia utilizas herramientas basadas en IA
(ChatGPT, Copilot, asistentes virtuales, etc.)?', 'single', TRUE,
NULL, NULL),
  (1, 4, '¿En qué ámbitos usas más la IA? (Puedes marcar varias)',
'multi', TRUE, NULL, NULL),
  (1, 5, '¿Consideras que dependes demasiado de la IA?', 'single',
TRUE, NULL, NULL),
  (1, 6, '¿Cuáles crees que son los riesgos del uso excesivo de la IA?
(Marca los que consideres)', 'multi', TRUE, NULL, NULL),
  (1, 7, '¿Crees que la IA debería regularse para evitar abusos?',
'single', TRUE, NULL, NULL),
  (1, 8, 'Describe en una frase qué piensas del impacto de la IA en la
sociedad:', 'text', TRUE, NULL, NULL),
  (1, 9, '¿Qué solución propondrías para evitar un uso excesivo de la
IA?', 'text', TRUE, NULL, NULL);
```

```
-- Opciones para la pregunta 1 (Edad)
INSERT INTO Choices (Question_ID, Position, Label, Value) VALUES
  (1, 1, 'Menos de 18 años', 'menos_18'),
  (1, 2, '18-25 años', '18_25'),
  (1, 3, '26-35 años', '26_35'),
  (1, 4, '36-50 años', '36_50'),
  (1, 5, 'Más de 50 años', 'mas_50');

-- Opciones para la pregunta 2 (Ocupación)
INSERT INTO Choices (Question_ID, Position, Label, Value) VALUES
  (2, 1, 'Estudiante', 'estudiante'),
  (2, 2, 'Empleado/a', 'empleado'),
  (2, 3, 'Empresario/a', 'empresario'),
  (2, 4, 'Desempleado/a', 'desempleado'),
  (2, 5, 'Otro', 'otro');

-- Opciones para la pregunta 3 (Frecuencia de uso de IA)
INSERT INTO Choices (Question_ID, Position, Label, Value) VALUES
  (3, 1, 'Varias veces al día', 'varias_dia'),
  (3, 2, 'Una vez al día', 'una_dia'),
  (3, 3, 'Varias veces por semana', 'varias_semana'),
  (3, 4, 'Casi nunca', 'casi_nunca');

-- Opciones para la pregunta 4 (Ámbitos de uso de IA)
INSERT INTO Choices (Question_ID, Position, Label, Value) VALUES
  (4, 1, 'Trabajo/estudios', 'trabajo_estudios'),
  (4, 2, 'Entretenimiento', 'entretenimiento'),
  (4, 3, 'Creatividad (arte, música, escritura)', 'creatividad'),
  (4, 4, 'Comunicación', 'comunicacion'),
  (4, 5, 'Otro', 'otro');

-- Opciones para la pregunta 5 (Dependencia de la IA)
INSERT INTO Choices (Question_ID, Position, Label, Value) VALUES
  (5, 1, 'Sí', 'si'),
  (5, 2, 'No', 'no'),
  (5, 3, 'No estoy seguro/a', 'no_seguro');

-- Opciones para la pregunta 6 (Riesgos del uso excesivo de IA)
INSERT INTO Choices (Question_ID, Position, Label, Value) VALUES
  (6, 1, 'Pérdida de creatividad humana', 'perdida_creatividad'),
  (6, 2, 'Desplazamiento laboral', 'desplazamiento_laboral'),
  (6, 3, 'Falta de habilidades propias', 'falta_habilidades'),
```

```
  (6, 4, 'Dependencia tecnológica', 'dependencia_tecnologica'),
  (6, 5, 'Privacidad y seguridad de datos', 'privacidad_seguridad'),
  (6, 6, 'Otro', 'otro');

-- Opciones para la pregunta 7 (Regulación de la IA)
INSERT INTO Choices (Question_ID, Position, Label, Value) VALUES
  (7, 1, 'Sí, con leyes estrictas', 'si_estrictas'),
  (7, 2, 'Sí, pero con flexibilidad', 'si_flexibles'),
  (7, 3, 'No, debe ser de libre uso', 'no_libre_uso'),
  (7, 4, 'No sé', 'no_se');



-- select * from Respondents;
-- select * from Auth;
-- select * from Surveys;
-- select * from Questions;
-- select * from Choices;
-- select * from Answers;
-- select * from Answers_Choices;
-- select * from Submissions;
```

Models

Es una carpeta dentro del proyecto de backend (ASP.NET Core) que contiene los modelos de las tablas de la BD, así como también el archivo del contexto de la BD (`ProyectoEncuestaContext.cs`).

Dichos modelos son los siguientes:

- `Answer.cs`: Representa cada respuesta de cada pregunta.
- `Auth.cs`: Contiene los datos de autenticación de cada usuario.
- `Choice.cs`: Representa cada una de las posibles opciones de una pregunta.
- `Question.cs`: Representa cada pregunta de la encuesta.
- `Respondent.cs`: Representa cada usuario.

- `Submission.cs`: Representa cada envío de la encuesta, o sea, cada encuesta completamente respondida y enviada por el usuario

- `Survey.cs`: Representa cada una de las encuestas almacenadas.

Frontend

`/.env`

Este archivo define variables de entorno, en este caso se guarda una variable de entorno que contiene la URL del servidor que ejecuta el backend, el propósito de esta variable es minimizar el código duplicado al hacer peticiones al backend.

`/src/App.jsx`

Debido a que la parte de frontend es una Single Page App (SPA), solamente se tiene una página cuyo contenido se reemplaza dinámicamente, este cambio de contenido se realiza en este archivo, `App.jsx`. Es el orquestador del frontend.

`/src/assets`

Esta carpeta contiene los recursos estáticos utilizados por la aplicación, en nuestro caso solamente contiene un archivo `loader.svg`, el cual es un icono de carga de la aplicación.

`/src/components`

Esta carpeta contiene todas los componentes y pantallas de React usados en la aplicación. Estos son:

- `AnswersScreen.jsx`: En esta pantalla se muestran las respuestas que se han enviado de la encuesta. Esta página solo es visible para el administrador.

- `LoadingScreen.css`: Es el archivo de estilos del componente `LoadingScreen.jsx`, permite que el icono de carga de vuelta.

- `LoadingScreen.jsx`: En esta pantalla se muestra una rueda girando, para esperar a que se inicialize la aplicación (frontend + backend), después de ser inicializada la aplicación, somos redirigidos a la pantalla de login (`LoginScreen.jsx`) o a la pantalla con el menú principal (`MainMenuScreen.jsx`), dependiendo si aún tenemos una sesión valida.

- `LoginScreen.jsx`: En esta pantalla se pide al usuario que inicie sesión o que se registre, al registrarse se crea un usuario con los roles por defecto (“respondent”), y al iniciar sesión se crea una nueva sesión que dura aproximadamente 1 día o 24 horas.

- `MainMenuScreen.jsx`: En esta pantalla se muestran varias opciones, entre las cuales están **Cerrar sesión** (la cual destruye nuestra sesión actual y nos lleva de regreso a la pantalla de inicio de sesión), **Llenar encuesta** (que nos lleva a la pantalla de la encuesta para responderla), y **Mirar respuestas** (solamente disponible para el administrador, el cual nos lleva a la pantalla de respuestas enviadas para esta encuesta)

- `SurveyAnsweredScreen.jsx`: Esta pantalla se muestra cuando un usuario termina de responder una encuesta, le agradece su participación y le ofrece regresar al menú principal a través de un botón.

- `SurveyScreen.jsx`: En esta pantalla se muestra la encuesta como tal, para ser respondida por el usuario, al enviar la encuesta, se nos manda a la pantalla de `SurveyAnsweredScreen.jsx`.