Tarea 2

Andrés Jiménez Valverde
Helena Vargas Quiros

Tecnológico de Costa Rica

IC 4301 : Bases de datos 1

Franco Quirós Ramírez

28 de abril del 2025

Índice de contenido

Índice de figuras

Capítulo 1. Introducción

El presente documento tiene como objetivo presentar el contexto, ambiente, resultados y métricas de la Tarea Programada 2 del curso de Bases de Datos 1 del I semestre del 2025. Esta segunda tarea programada consiste en una prueba de concepto más compleja en la que se deben realizar funcionalidades en SQL y correrlas en un web browser. Esta tarea tiene el fin de aprender técnicas más complejas que funcionarán en el mundo real, como el cargado masivo de datos, filtrado y CRUD completo de distintas tablas, así como transacciones.

Entre los objetivos y requerimientos del proyecto se encuentran: Implementar una aplicación sencilla que realiza: validaciones de entrada de datos, consultas sobre varias tablas en la BD con despliegue de información en una interfaz de usuario, la captura, edición y validación de valores que se salvan en una tabla, el registro de movimientos en una aplicación transaccional, el registro de actividades en la BD a través de una bitácora de eventos, la implementación de manejo de errores en procedimientos almacenados, y la escritura de transacciones de BD.

En las siguientes secciones, se detalla información importante sobre el ambiente de desarrollo, el proceso de programación de la Tarea, los resultados obtenidos y las métricas del proyecto, explicado por medio de tablas y gráficos que reflejan la realidad del ambiente de desarrollo y los resultados.

Capítulo 2. Descripción del ambiente de desarrollo

En esta sección de ambiente de desarrollo se definen las principales herramientas y tecnologías utilizadas en la implementación de esta Tarea Programada, así como la forma en que los componentes se conectan e interactúan entre sí.

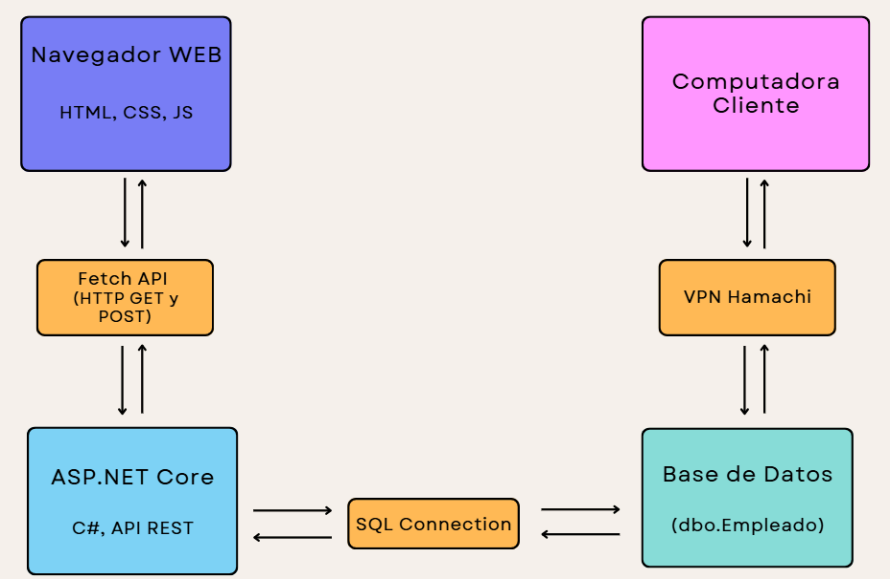
Herramientas y tecnologías utilizadas

- **IDE:** Visual Studio Community (Para desarrollo en C# y ASP.NET Core)

- **Base de Datos:** Microsoft SQL Server Community Edition
- **Lenguaje de capa lógica (backend):** C# con ASP.NET Core
- **Lenguajes de capa usuario (frontend):** HTML, CSS Y JavaScript
- **Cliente y Servidor:** El servidor de la base de datos se encuentra en la máquina de Helena Vargas. Para que la computadora cliente pueda acceder al servidor, se utiliza el VPN Hamachi, que permite establecer una red virtual privada.
- **Método de Comunicación:** El navegador web se comunica con el servidor a través de Fetch API llamando a endpoints HTTP de operaciones GET y POST. El backend se conecta a la base de datos usando SqlConnection y ejecuta consultas mediante SqlCommand.

El siguiente diagrama Figura 1 “Diagrama de ambiente de desarrollo” representa cómo los elementos del sistema interactúan en el desarrollo y ejecución de la aplicación:

Figura 1.
Diagrama de ambiente de desarrollo



Nota: Esta figura muestra las diferentes conexiones en el ambiente de desarrollo.

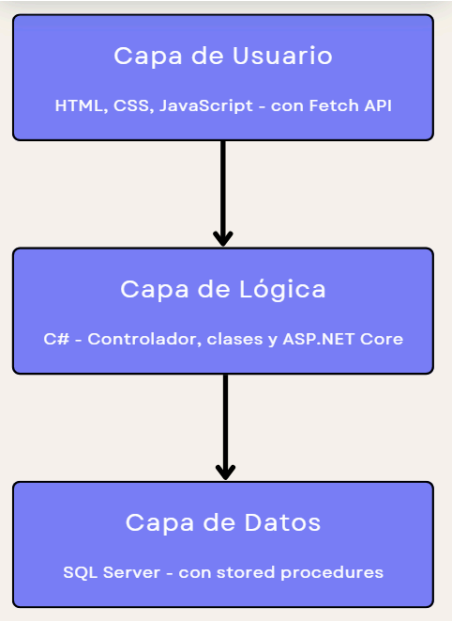
Arquitectura de Aplicación

En esta subsección se describe cómo está estructurado el código y cómo interactúan sus diferentes componentes. Este proyecto sigue una arquitectura en capas, donde cada capa tiene una responsabilidad específica.

- **Capa de Datos (BD y acceso a BD):**
    - La BD contiene distintas tablas, incluyendo catálogos como tablas más variables a las que se le aplica CRUD de manera más frecuente.
    - Se usan stored procedures para las operaciones principales: CRUD de empleados y movimientos, filtrado, detección de errores, entre otros.
    - La clase AccesarBD.cs se encarga de la conexión a la base de datos y la ejecución de los stored procedures, así como la recuperación del dataset.
- **Capa de Lógica (Backend y API en ASP.NET Core):**
    - El backend crea dos APIs mediante la clase BDController.cs, GET para obtener la listas y POST para insertar parámetros.
    - Se utiliza SqlConnection para conectarse a la base de datos, y SqlCommand para ejecutar las stored procedures.
    - La API devuelve respuestas en formato json donde se encuentran los datos o los códigos de error dependiendo de la solicitud.
- **Capa de Usuario (Frontend . HTML, CSS, JavaScript):**
    - La interfaz de usuario se encuentra en la carpeta wwwroot, donde están los archivos principales html para las diferentes fases del CRUD de empleados y movimientos.
    - Los archivos css se encargan del diseño y posicionamiento de elementos estáticos.
    - Los archivos JavaScript manejan la comunicación con el backend usando Fetch API. GET muestra la listas o recupera datos, POST envía los datos del formulario y muestra éxito o error. Además se incluyen las acciones de los botones, y validaciones de forma.

Nota: No se utiliza MVC, solo una arquitectura por capas que se comunican mediante APIs. El backend expone APIs REST y el frontend las recibe mediante Fetch API.

Figura 2.
Gráfico de arquitectura de aplicación



Nota: Esta figura muestra las capas de arquitectura de aplicación.

Capítulo 3. Análisis de resultados

En este capítulo mostramos y comentamos los porcentajes obtenidos de satisfacción a la hora de completar cada métrica del proyecto, con comentarios de si faltaron funciones, errores sin terminar, funcionalidad y los porcentajes correspondientes obtenidos según los resultados de las pruebas.

La siguiente tabla “Tabla 1. Tabla de resultados” nos permite ver la evaluación de cada una de las partes del proyecto y el porcentaje atribuido a cada una según el funcionamiento observado en la tarea

Tabla 1.

Tabla de resultados

Item	**Resultado**
1.Documentación	% 15 / 15% Completa de acuerdo a la plantilla
2.BD y diseño	%7 / 7% Base de datos está creada de manera correcta siguiendo los estándares vistos en clase
3.Script XML Carga de datos	%15 / 15% Script XML funciona correctamente y obtiene los datos provenientes del mismo
4.Código SP	%23/ 23% Stored procedures siguen los estándares y funcionan correctamente
5.Funcionalidad	%40/40% Funciona y sigue instrucciones
5.1 Login/Logout	%6/6% Funciona correctamente
5.2 Lista empleado con filtrado	%6/6% Funciona correctamente
5.3 Insertar empleado	%5/5% Funciona correctamente
5.4 Insertar movimientos	%7/7% Funciona correctamente
5.5 Listar Movimientos	%5/5% Funciona correctamente
5.6 Trazabilidad	%6/6% Funciona correctamente
5.7 Mensajes y códigos de error	%5/5% Funciona correctamente

Nota: Esta es una tabla que contiene los resultados de la métrica de cada parte del proyecto su % y un comentario del porque el valor correspondiente, utiliza los valores de % por cada ítem

Capítulo 4. Métricas de proyecto

En este capítulo veremos las mediciones de tiempo, pruebas, entradas, horas trabajadas y otros que nos permitan medir el esfuerzo y tiempo que se le dedicó al proyecto desde el momento en el que fue asignado. En la siguiente tabla, “Tabla 2. Tabla con las métricas del proyecto”

Tabla 2.

Tabla con las métricas de la tarea

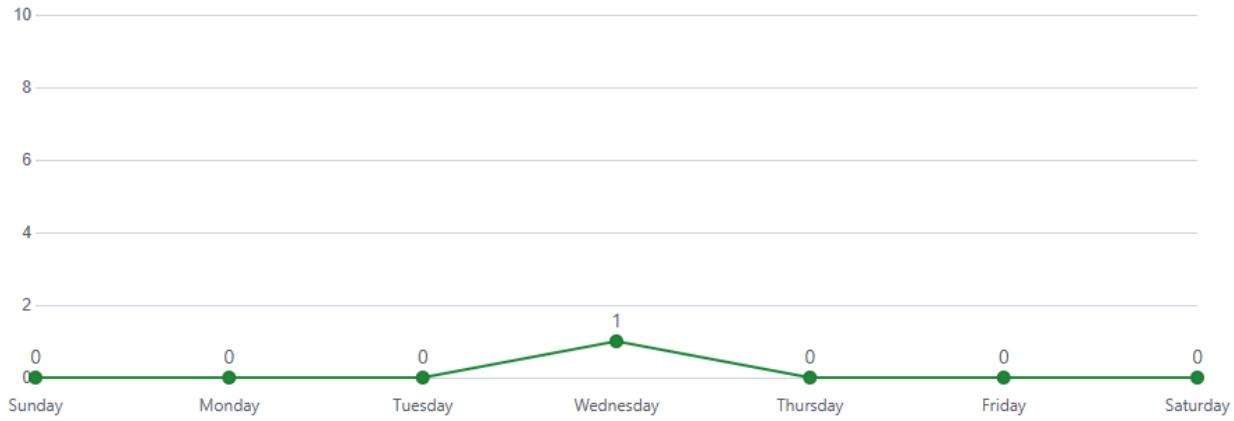
Métrica	Valor	Comentarios
Fecha de Inicio	2/04/25	Fecha inicial
Total de horas proyecto	15 horas	A lo largo de 5 semanas
Consultas al profesor	2 virtuales	Recordatorios de subir clase con información aclaratoria relacionada al proyecto
Entradas de blog	8	Cantidad de entradas de blog
Actualizaciones de github	41	Cantidad total de commits
Cantidad de tablas	10	Tablas creadas en SQL
Cantidad de SP	15	Stored procedures creados
Cantidad de pruebas	20 al día	Se hacían pruebas para verificar funcionalidades específicas continuamente, y pruebas que abarcaban todas las funcionalidades
Tiempo de ejecución del script	10	Helena es la computadora host y Andrés la computadora cliente
Cantidad de datos procesados	380~	Variaba según se iban agregando más datos a la tabla
Cantidad líneas de código (SQL)	1000~1100	Divididos en cada uno de los SP y tablas
Cantidad líneas de código (C#)	1567	C# esta dividido en códigos controladores
Cantidad líneas de código	1429	Cada página HTML cuenta

(JS)		con su programación en JS
Cantidad líneas de código (HTML)	225	Las páginas para dar datos al usuario
Cantidad líneas de código (CSS)	455	Cada página HTML tiene su código CSS con formato
Cantidad líneas de código (Total)	4676 en promedio	Total de lineas de código

Nota: Aquí se indican varias de las cantidades medibles que se tengan que tratar del proyecto ya sean horas o cantidades de pruebas y comentarios acerca de cada una

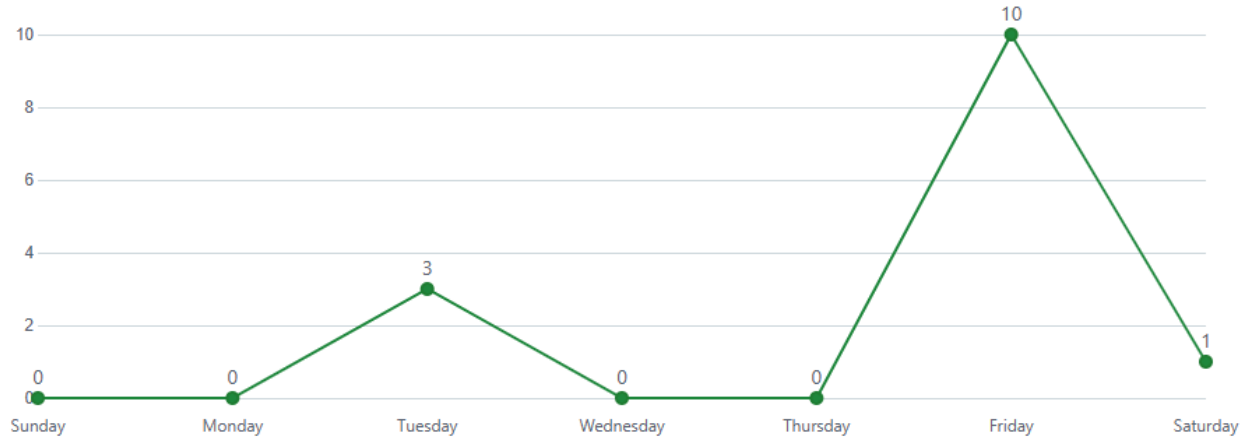
Las siguientes figuras nos permiten tener una visualización más detallada de los días en los que se trabajó y se subió la información al repositorio, esto permite que podamos ver la actividad del proyecto y el progreso realizado de manera semanal para la realización de la tarea

Figura 3.
Cantidad de entradas en la semana del 29 de marzo en github



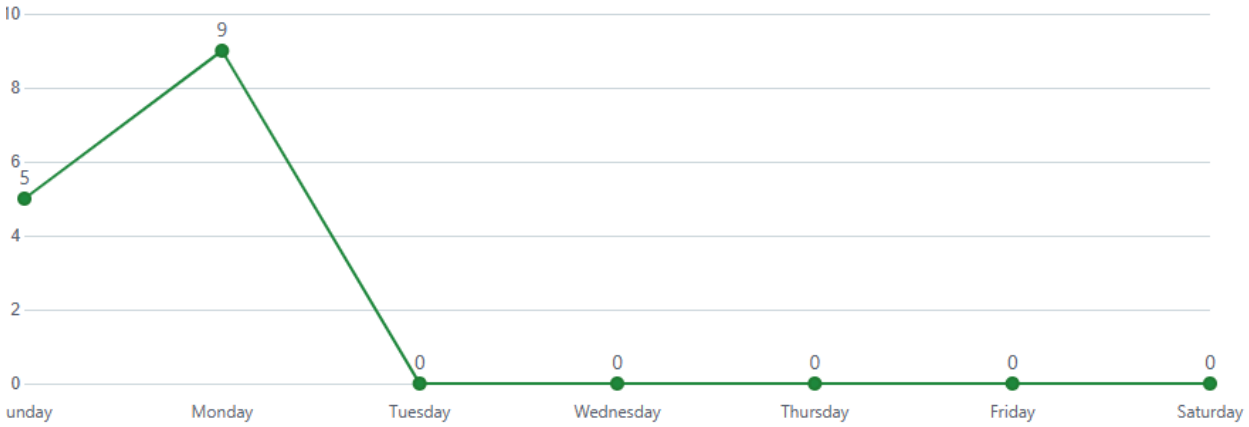
Nota: Esta figura muestra los días en los que se actualizó la base de datos de github con nuevas entradas en la semana del 29 de marzo, la tabla viaja del 29 de marzo hasta el 5 por lo cual el primer dia mostrado es 2 de marzo

Figura 4.
Cantidad de entradas en la semana del 6 de abril en github



Nota: Esta figura muestra los días en los que se actualizó la base de datos de github con nuevas entradas en la semana del 6 de abril hasta el 12 de abril

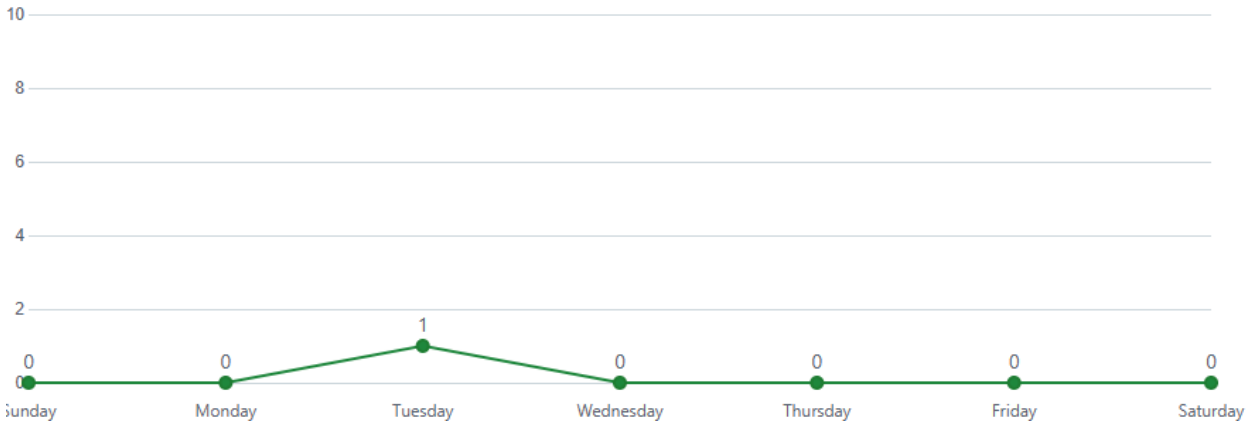
Figura 5.
Cantidad de entradas en la semana del 13 de abril en github



Nota: Esta figura muestra los días en los que se actualizó la base de datos de github con nuevas entradas en la semana del 13 de abril hasta el 19 de abril

Figura 6.

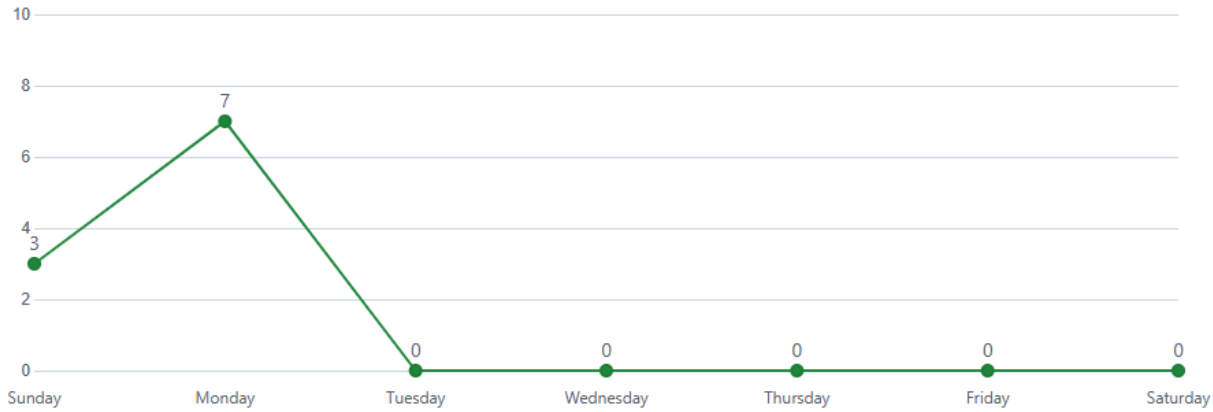
Cantidad de entradas en la semana del 20 de abril en github



Nota: Esta figura muestra los días en los que se actualizó la base de datos de github con nuevas entradas en la semana del 20 al 26

Figura 7.

Cantidad de entradas en la semana del 27 de abril en github

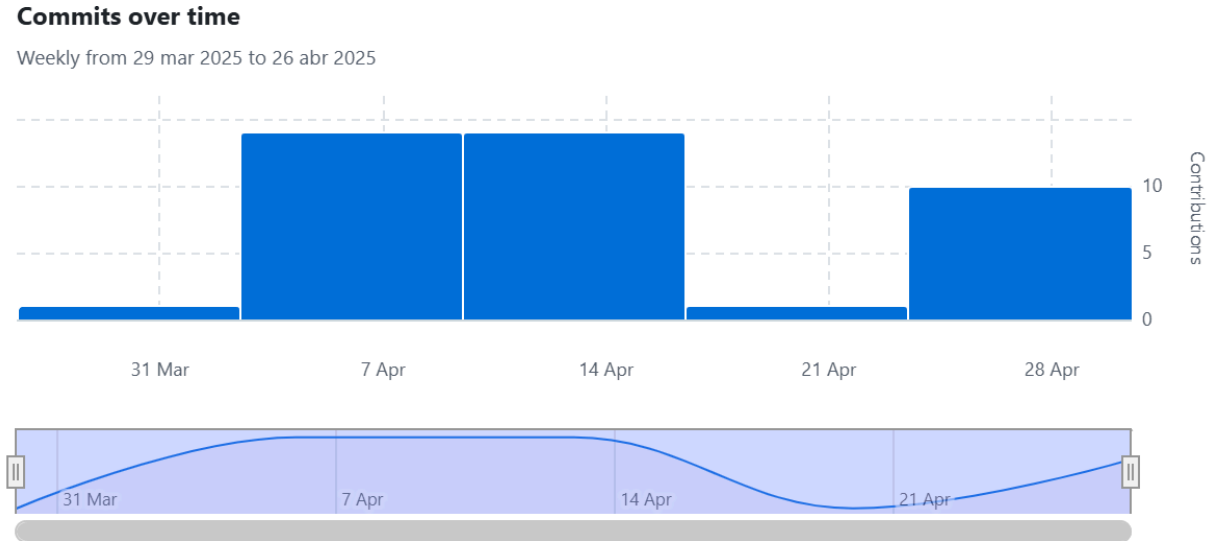


Nota: Esta figura muestra los días en los que se actualizó la base de datos de github con nuevas entradas en la semana del 27 hasta la entrega del proyecto

Mientras tanto la siguiente figura “Figura 8. *Contribuciones totales a lo largo del proyecto de manera semanal desde el 2 de abril hasta el 28 de abril” es una figura resumen que nos permite ver de manera compacta el progreso realizado semanalmente en github y la cantidad de entradas realizadas en cada lapso de tiempo*

Figura 8.

Contribuciones totales a lo largo del proyecto de manera semanal desde el 2 de abril hasta el 28 de abril



Nota: Esta figura muestra de manera compresa las entradas al github a lo largo del tiempo según las semanas de trabajo

Links adicionales

Link al repositorio GitHub:
LilyJmz/Tarea-2

Link al blogger:
Tarea-2