Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computación

Curso: Bases de Datos I

Profesor: Ing. Franco Quirós Ramírez

Primera Tarea Programada: Prueba de Concepto

Estudiantes:
Andrés Baldi Mora
Elías Ramírez Hernández

II Semestre, 2025

Índice de contenidos

Índice de figuras

Capítulo 1: Introducción

En este documento se presenta toda la documentación e información necesaria sobre los resultados obtenidos de la primera tarea programada del curso de bases de datos I, realizada por los estudiantes Andrés Baldi Mora y Elías Ramírez Hernández. El principal objetivo del presente documento consiste en demostrar el cumplimiento de los requerimientos que se nos dieron del proyecto, a su vez detallar descripciones del ambiente de desarrollo y las métricas del proyecto.

El proyecto como tal, consistió en realizar una aplicación web que se conecte a una capa lógica y que está a su vez se conectará con la base de datos. Para lo que fue la aplicación web y más específicamente el frontend se emplearon lenguajes y tecnologías sencillas del desarrollo web como HTML, CSS y JavaScript. Para la capa lógica se empleó Python y por último para la base de datos se usó Microsoft SQL Server, tal y como se solicitó en los requerimientos del programa. Para crear la red se empleó el software llamado Hamachi para poder crear un ambiente de trabajo colaborativo.

A lo largo de este documento se detallarán los aspectos ya mencionados en el párrafo anterior y aspectos técnicos como el análisis de los resultados y las métricas empleadas en el proyecto, esto con el fin de demostrar que se cumplieron los requerimientos solicitados y un funcionamiento correcto de la solución implementada.

Capítulo 2: Contexto del desarrollo

Descripción del ambiente de desarrollo

En el ambiente de desarrollo empleado para realizar este proyecto se emplearon diferentes tecnologías y aplicaciones externas para poder realizarlo de acuerdo con los requerimientos indicados. Entre ellas, destacan GitHub (para manejar las versiones del código del proyecto), SQL Server Management Studio (para administrar la base de datos y ejecutar queries), Visual Studio Code (para edición general de código) y Discord (una red social mediante la cual realizamos las reuniones virtuales para trabajar). El repositorio de GitHub se puede encontrar en el enlace: https://github.com/EliPoli64/Tarea1BasesDatos . Además, utilizamos la herramienta Blogger para realizar la bitácora del proyecto, la cual se puede encontrar en este enlace: https://tarea1bases1.blogspot.com/

Para la arquitectura de red se utilizó el software gratuito llamado “Hamachi”, el uso de este software fue necesario para evitar alojar la base de datos en un servidor externo ya que nos pedía ingresar nuestra tarjeta de crédito o débito y nos dio cierto nivel de incertidumbre de si nos realizarían algún cobro a futuro, entonces para evitar los cobros y las incertidumbres decidimos emplear el “Hamachi”. Hamachi es un software que permite realizar virtualización de redes que permite emular una red local o también conocida como LAN entre varios dispositivos conectados a través de internet. También emplea la tecnología de redes privadas virtuales o VPN para garantizar una transmisión de información más segura y privada. Este software lo que nos permitió como ya se mencionó fue crear una red “local” para que otra máquina se conectará desde una red externa a la red local y a su vez lograra conectarse al servidor local donde se encontraba la base de datos alojada.

Arquitectura de aplicación

Este proyecto cuenta con 3 partes fundamentales para su correcto funcionamiento, las cuales son el frontend, backend y la base de datos. En la figura 1 podemos ver una representación gráfica de lo mencionado.

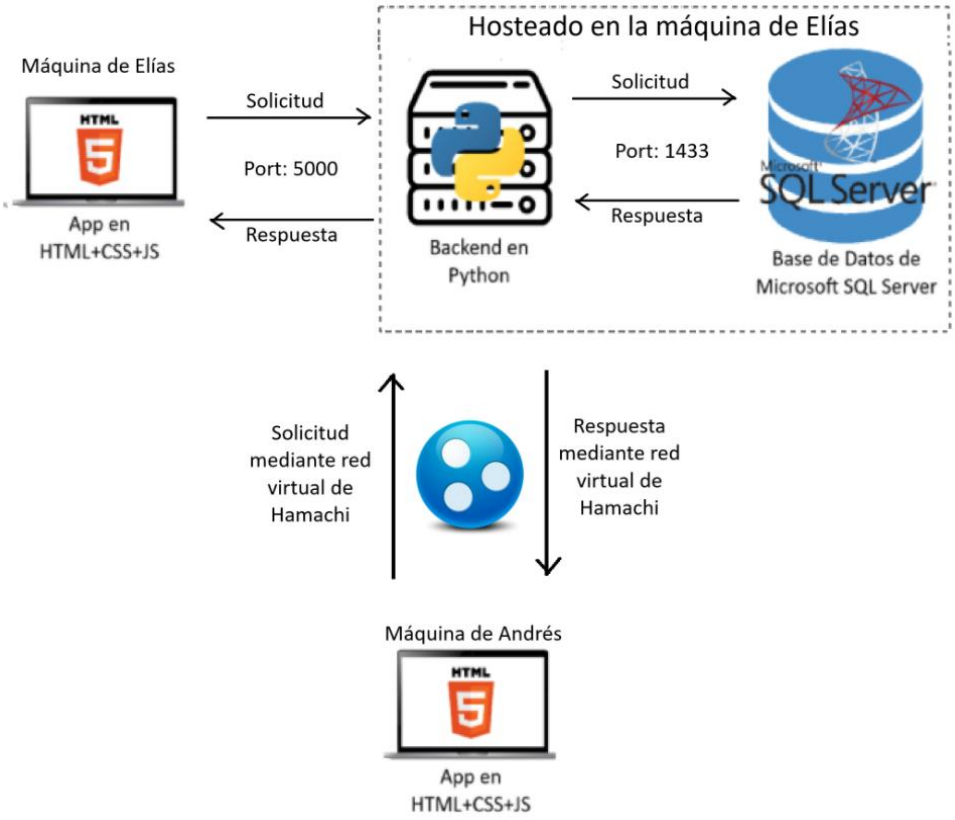

Figura 1: Arquitectura de aplicación del proyecto.

Frontend

Para la parte del frontend se emplearon tecnologías relativamente sencillas como lo vendrían siendo HTML5, CSS3 y JavaScript. La estructura como tal del frontend se divide en 4 archivos, dos HTML y dos JavaScript. El primer archivo de HTML es el “html.index” el cual es la página principal para visualizar y buscar los empleados, el segundo archivo es “insertar.html” donde se encuentra el formulario para insertar nuevos empleados. Pasando a los archivos JavaScript tenemos el “src.js” el cual cuenta con toda la lógica de la página principal, tanto de la consulta como de la visualización. Por último, tenemos el archivo “insertar.js” el cual contiene la lógica del formulario de inserción, se toman en cuenta las validaciones y el envío de datos.

Backend

El backend es un poco más extenso en cuanto a tecnologías comparado con el frontend, acá se emplearon tecnologías como Python, Flask, Flask-CORS y pyODBC. Contamos con un único archivo el cual aparece con el nombre de “backend.py” y en este mismo archivo podremos encontrar todo lo referente con la capa lógica y manejo de los datos que se extraen de la base de datos. Se hicieron un total de 3 endpoints los cuales llevan los siguientes nombres:

- “GET /proyecto/select/<nombre>”: Es el encargado de filtrar a los empleados mediante el nombre que el usuario digita.

- “GET /proyecto/selectTodos”: Se encarga de obtener a todos los empleados de la base de datos, para mostrarlos en el grid general

- “GET /proyecto/insert/<nombre>/<salario>”: Se encarga de insertar a los empleados a la base de datos con su respectivo salario.

Base de datos

Para la base de datos solamente se empleó una tecnología la cual es Microsoft SQL Server y su respectivo Management para poder crear la base de datos y las tablas solicitadas en los requerimientos. La tabla se llama “Empleado” y esta cuenta con tres campos, “ID (Pk, Identity)”, “Nombre (varchar)” y “Salario (money)”, a su vez, cuenta con dos Stored procedures que son los encargados de comunicarse con el backend para evitar tener código SQL en el backend y así aumentar la seguridad del programa. El primer Stored procedure se llama “sp_FiltrarEmpleados” y se encarga, como su nombre lo indica de filtrar empleados por patrón de nombre. El otro Stored Procedure se llama “sp_InsertarEmpleado” y lo que hace es insertar un nuevo empleado con validación de que este no exista. Contamos también con dos Scripts extras aparte de los Stored Procedures, los cuales son el Script de llenado de la base de datos y el script de creación de la

base de datos, pero estos no están conectados con el backend, se podría decir que son “independientes” pero fundamentales para el funcionamiento correcto de la base de datos.

Tecnologías empleadas y explicación

Frontend

- HTML: Es el encargado de darle el “esqueleto” a la página web, todos los botones, formularios, entradas de texto, etiquetas son gracias a este componente de estructura semántica de las páginas web

- CSS: Es el encargado de darle “estilo” a los archivos de HTML para que estos sean más atractivos para la vista del usuario dándole así una experiencia más amena con los estilos y diseños responsive que se pueden lograr con CSS.

- JavaScript: Fue empleado para la comunicación con el API mediante la realización de peticiones HTTP al servidor (comunicar el frontend con el backend) y para añadir interactividad en la parte del frontend.

Backend

- Python: Python fue el lenguaje seleccionado para programar la capa lógica que se comunicaría entre la base de datos y la página web, con dicho lenguaje se programó el servidor.

- Flask: Flask es un microframework que se usó para crear la API REST, algunas de las características que se emplearon fueron la definición de los endpoints API y la devolución de los datos en formato JSON.

- Flask-CORS: Es el manejo de políticas CORS para comunicación cross-origin que permite al frontend comunicarse con el backend sin restricciones del navegador.

- pyODBC: Es el conector Python para comunicación con SQL Server mediante ODBC, algunas de sus características son el soporte de transacciones y manejo de conexiones y la ejecución de stored procedures y queries parametrizadas.

Base de datos

- Microsoft SQL Server: Este es el motor de la base de datos empleada, el cual es un sistema de gestión de base de datos relacional.

- T-SQL: Es el lenguaje de Microsoft SQL Server el cual cumple con la función de realizar stored procedures y queries para consultar, modificar, eliminar o insertar datos dentro de la base de datos.

Capítulo 3: Resultados del proyecto

Elemento evaluado	Valoración	% de implementación	Comentarios
Documentación	Implementado en su mayoría bien	90%	No se documentaron a fondo los errores obtenidos, ni en el Blogger ni en la documentación.
Base de datos y diseño	Está implementado exitosamente	100%	No hubo que diseñar nada, solo implementar un diseño predeterminado.
SP creados	Está implementado exitosamente	100%	N/A
Datos de prueba	Está implementado exitosamente	100%	N/A

Elemento evaluado	Valoración	% de implementación	Comentarios
Conexión a BD	Está implementado exitosamente	100%	N/A
Listar empleados	Está implementado exitosamente	100%	N/A
Insertar empleado	Está implementado exitosamente	100%	N/A

Capítulo 4: Métricas del proyecto

Métrica	Total	Comentarios
Horas trabajadas	15h, 20m	No tuvimos mucho problema con errores. +1 hora para revisar todo finalmente y comparar con la rúbrica enviada por el profesor.
Número de sesiones de trabajo	4	N/A
Duración total de las pruebas (en horas)	3	N/A
Horas de resolución de errores	3	Esto fue por la máquina de Elías, cuyo driver de BD no funcionaba, y por Hamachi
Total de líneas de código	520	Líneas añadidas – líneas eliminadas según GitHub.
Número de commits GitHub	8	N/A
Cantidad de archivos	12	4 archivos de SQL (SPs, script de llenado, script creación BD), 1 archivo de Python, 2 archivos de JavaScript, 2 archivos de HTML, 1 archivo de CSS, 1 archivo .docx (documentación) y 1 archivo PDF (documentación exportada)

Métrica	Total	Comentarios
Cantidad de datos de prueba (registro de la tabla)	50	N/A
Cantidad de tablas creadas	1	Solo había una tabla en la especificación.
Cantidad de procedimientos almacenados	2	N/A
Cantidad de funciones	7	3 funciones creadas en Python (backend), 4 en JavaScript (frontend)

Figuras adicionales (Gráficos de GitHub)

Figura 2: Historial de clonaciones/pulls y visitantes del repositorio de GitHub.

Figura 3: Historial de commits del repositorio.