EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Ciudad Valles

REPORTE FINAL DE RESIDENCIA PROFESIONAL

Plataforma institucional de la formación docente y actualización profesional: enfoque en el backend

Ingeniería en Sistemas Computacionales

Luis Felipe Garcia Martínez **20690041**

M.E. Claudia Cruz Navarro
ASESOR INTERNO

DR .Jaime Jesús Delgado Meraz
ASESOR EXTERNO

Ciudad Valles, S.L.P. a 6 de diciembre del 2024

Dedicatoria o Agradecimiento

A Dios, mi guía celestial, por brindarme fortaleza, sabiduría y la oportunidad de enfrentar cada desafío con fe y esperanza.

A mis padres, por ser la base sólida de mi vida. A mi madre Lila Patricia Martínez Estrada, por su amor incondicional, sacrificios y enseñanzas, y a mi padre Tomás García García, por su apoyo constante, ejemplo de perseverancia y sus consejos que siempre me han guiado.

A mis abuelos paternos Vicenta García y Lucio García, quienes, aunque ya no están físicamente, permanecen en mi corazón como una fuente eterna de inspiración, amor y sabiduría. Sus enseñanzas y valores dejaron una huella imborrable en mi vida.

A mi novia Evelyn Naomi Balderas, por su compañía inquebrantable, su comprensión y por ser un pilar fundamental en los momentos de esfuerzo y superación.

A mis amigos, por su lealtad, ánimo y por hacer de este viaje uno lleno de experiencias compartidas que valoro profundamente.

Y a mis profesores, cuya dedicación y enseñanzas trascendieron lo académico, dejando en mí lecciones de vida que han forjado mi identidad y fortalecieron mi determinación para alcanzar mis metas.

A todos ellos, gracias por su amor, apoyo y confianza, que me han impulsado a alcanzar una de las metas más importantes de mi vida.

Resumen

El presente proyecto tiene como propósito desarrollar una plataforma web destinada al registro y gestión de cursos de capacitación dirigidos a los docentes del TecNM Campus Ciudad Valles. Este esfuerzo tiene como objetivo optimizar los procesos administrativos relacionados con la formación docente, mejorando la eficiencia, accesibilidad y organización de la información y los recursos asociados.

Para su desarrollo, se adoptó la metodología espiral, caracterizada por sus fases iterativas de comunicación, planeación, modelado, construcción y despliegue. Esta metodología permitió una ejecución flexible y adaptable a las necesidades específicas del proyecto. Asimismo, se emplearon tecnologías como Laravel, PHP y MySQL, que garantizaron la creación de una plataforma robusta, escalable y eficiente, capaz de responder a las demandas actuales del entorno educativo.

El proyecto está dirigido a los docentes del Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles, quienes se beneficiarán al contar con una herramienta tecnológica moderna que facilita su acceso y participación en los procesos de capacitación y actualización profesional. Con esta iniciativa, se busca fortalecer los procesos educativos institucionales y fomentar el desarrollo continuo del personal docente, contribuyendo así al mejoramiento de la calidad educativa.

ÍNDICE

1.- Introducción

El Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles, en su compromiso con la mejora continua y la excelencia educativa, cuenta con diversas áreas y departamentos que desempeñan funciones específicas. Entre ellos, destaca el Departamento de Desarrollo Académico (DA), encargado del diseño y desarrollo del proyecto Plataforma Institucional de la Formación Docente y Actualización Profesional.

Este proyecto surge como respuesta a la necesidad de abordar diversas problemáticas relacionadas con la gestión y administración de los cursos de capacitación dirigidos al personal docente. Actualmente, el campus carece de un sistema de información centralizado que permita un control y registro automatizado de los cursos. En su lugar, se utilizan formularios y documentos digitales dispersos, lo que genera problemas como la entrega tardía, incompleta o poco clara de información a los docentes, dificultando su acceso a los detalles de los cursos y afectando la eficiencia del proceso.

Además, la ausencia de un sistema centralizado genera esfuerzos duplicados en tareas como registros o búsquedas de información, al no estar disponible de manera inmediata y organizada. En este contexto, el proyecto de la plataforma tiene como objetivo ser una solución integral que optimice los procesos administrativos, mejore la comunicación entre los involucrados y facilite el acceso oportuno a la información, contribuyendo al fortalecimiento de la calidad educativa de la institución.

Para abordar estas problemáticas, se diseñó una interfaz de usuario intuitiva y accesible, permitiendo que tanto los docentes como los administradores puedan interactuar fácilmente con la plataforma. Entre sus principales funcionalidades, destacan el registro de docentes en los cursos de capacitación, la visualización de la oferta disponible, la consulta de historiales de capacitación y la automatización del proceso de gestión de cursos.

Esto último permite actualizar y administrar la información de manera eficiente, reduciendo la carga administrativa. Asimismo, se incorporó un sistema de generación de reportes e informes que facilita la toma de decisiones basada en datos confiables y organizados.

El desarrollo del proyecto fue posible gracias al uso de tecnologías avanzadas, que permitieron cumplir con los objetivos establecidos. Como resultado, se logró implementar una herramienta tecnológica moderna y eficiente que fortalece la formación continua del personal docente y mejora significativamente los procesos administrativos del Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles.

2.- Descripción de la empresa u organización y del puesto o área del trabajo del estudiante

2.1.- Antecedentes de la empresa

El Instituto Tecnológico de Ciudad Valles es una Institución de Educación Superior, dependiente del Tecnológico Nacional de México (TecNM), cuenta con 43 años de historia, brindando un servicio educativo en Ciudad Valles y la Región Huasteca; representa además la primer Institución de Educación Superior en la región, y a lo largo de estos años han egresado casi 6671 profesionistas de las diferentes carreras; como Institución pionera de la Educación Superior en la Huasteca Potosina, se ha consolidado en la región, gracias al fortalecimiento permanente por la estructura institucional que se lleva a cabo; organizacional y académica.

Su historia inicia el 06 octubre de 1980, fecha en que abre sus puertas para recibir a los primeros integrantes de lo que sería la Comunidad Tecnológica, que apuntaría al desarrollo agropecuario; y a raíz de una Conferencia Magistral realizada en las instalaciones del Cinema Valles 70, el día seis de octubre del año de 1980, a las 10:00 horas, y estando presente el Gobernador del estado de San Luis Potosí: Lic. Carlos Jonguitud Barrios, así como autoridades educativas del Estado, se consolida el proyecto de la creación del Instituto Tecnológico Agropecuario (ITA) No. 22 durante el sexenio del Presidente José López Portillo.

Durante aquellos dos años iniciales, de 1980 a 1982, el ITA No. 22 tenía oficinas administrativas en un local prestado de la Presidencia Municipal, donde el alcalde Rafael Piña González facilitó el mobiliario; y bajo la orientación del primer director y fundador, Ing. Miguel Ángel Villar, también trabajaron como catedráticos los profesionistas Ing. Gustavo Tirado Estrada, el MVZ. Juan Manuel Lumbreras Martínez, el Lic. Víctor Baños Acosta, el Lic. Pánfilo Abundis Flores, el Ing. Leopoldo Ocegueda

Altamirano y el Ing. Abel López Márquez. Los fundadores civiles del plantel fueron el presidente Municipal en turno, el C. Rafael Piña González, el Gobernador del Estado, Lic. Carlos Jonguitud Barrios, y el Lic. Abraham Chemas González, quien donó el terreno para los edificios.

En el año 1992 ante la necesidad de nuevas y mejores opciones educativas, el Gobierno Federal del Lic. Carlos Salinas de Gortari, a través de la Secretaria de Educación Pública dio acceso a la petición de un nuevo plantel y cambió el esquema académico de Instituto Tecnológico Agropecuario al de Instituto Tecnológico; con el nuevo sistema, el Instituto Tecnológico de Ciudad Valles (ITCV) abrió inicialmente la carrera de Ingeniería Industrial y Licenciatura en Administración con una matrícula de 23 y 29 alumnos respectivamente; y al siguiente año la matrícula incrementó a 92 y 141 estudiantes en las dos carreras mencionadas.

En el año 1994 se da inicio a la carrera de Licenciatura en Informática, la cual tuvo una excelente aceptación, pues se inscribieron 176 alumnos de nuevo ingreso en ese año y en el año 2002 la carrera de Agronomía entra en liquidación por la baja demanda que presentaba y se inició la carrera de Ingeniería en Industrias Alimentarias con 60 estudiantes; actualmente, son seis las carreras ofertadas: Ingeniería en Gestión Empresarial, Ingeniería Industrial, Ingeniería en Sistemas Computacionales, Ingeniería Ambiental e Ingeniería en Industrias Alimentarias, en agosto del 2020 se da la reapertura de Ingeniería en Agronomía, sumado a la modalidad mixta un nuevo programa para contar ya con dos programas de estudio en la modalidad semipresencial en el 2020, Ingeniería en Gestión Empresarial con apertura en el 2018 e Ingeniería Industrial y un posgrado ofertado en ingeniería con las líneas de investigación de: Producción más Limpia e Ingeniería en Software desde el 2017.

Cabe destacar que el día 23 de julio de 2014 fue publicado, en el Diario Oficial de la Federación, el Decreto Presidencial por el que se crea la Institución de Educación Superior Tecnológica más grande de nuestro país, el Tecnológico Nacional de México (TecNM).

De acuerdo con el Decreto citado, el TecNM se funda como un órgano desconcentrado de la Secretaría de Educación Pública, que sustituye a la unidad administrativa que se hacía cargo de coordinar este importante subsistema de Educación Superior; y es a partir de esa fecha que el Tecnológico Nacional de México, Tecnológico de Ciudad Valles, al igual que 254 Instituciones forman parte de la comunidad TecNM.

El esfuerzo colaborativo de personal directivo, docente, de apoyo a la educación y estudiantes; se refleja en los avances obtenidos, las oportunidades de mejora y en el uso de los recursos asignados a la Institución; siempre comprometidos para formar seres humanos y profesionistas íntegros, que construyan una sociedad más justa y equitativa en el México actual.

Por lo anterior, es importante destacar que en los últimos años se han obtenido resultados importantes, tales como: la obtención de recursos para equipamiento de laboratorios y talleres, la construcción de una Unidad Multifuncional, un Gimnasio Auditorio, una Unidad Académica Departamental Tipo II, un Edificio de Posgrado y un Edificio Multifuncional, el mantenimiento de la certificación del Sistema de Gestión de la Calidad ISO 9001:2015, la certificación en el Sistema de Gestión Ambiental ISO 14001:2015, la certificación en el Sistema de Gestión de la Seguridad y Salud en el Trabajo ISO 45001:2018, la certificación en el Sistema de Gestión de Energía ISO 50001:2018, la certificación en la Norma Mexicana de Igualdad Laboral y No Discriminación NMX-R-025-SCFI-2015 y la conformación de la Comisión de Seguridad y Salud en el Trabajo, Unidad Interna de Protección Civil, Comité de Gestión Ambiental y Comité de Gestión de la Energía; la participación del Instituto en la red ANUIES, el

reconocimiento de la Secretaría del Trabajo y Previsión Social como Institución capacitadora y de adiestramiento, el incremento en la participación de docentes y estudiantes en proyectos de investigación e innovación tecnológica; la acreditación por su buena calidad de los Planes y Programas de Estudio de las carreras de Ingeniería en Industrias Alimentarias e Ingeniería Industrial; así como la apertura en el 2017 del Programa de Posgrado con la Maestría en Ingeniería; logros que ayudan de manera significativa a mejorar el servicio educativo y que nos impulsan a engrandecer nuestras fortalezas institucionales.

Hoy, gracias a la certeza de que, en el TecNM, Instituto de Ciudad Valles, existe el compromiso entre estudiantes, académicos, personal de apoyo a la educación y directivos; se tiene la plena confianza de que la participación activa y dinámica, seguirá uniendo esfuerzos para enfrentar a los retos venideros, y así dar soluciones estratégicas e integrales a las necesidades de la sociedad. (Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles, 2024)

2.2.- Misión de la empresa

Formar ciudadanos del mundo de nivel licenciatura y posgrado, comprometidos con el desarrollo socioeconómico, cultural y ambiental, con amplio conocimiento científico-tecnológico, resilientes y capaces de dar respuesta a las necesidades del entorno globalizado. (Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles, 2024)

2.3.- Visión de la empresa

Mantener el liderazgo en educación superior tecnológica de calidad por el destacado desempeño de los egresados para enfrentar con responsabilidad social los retos del contexto global. (Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles, 2024)

2.4.- Organigrama

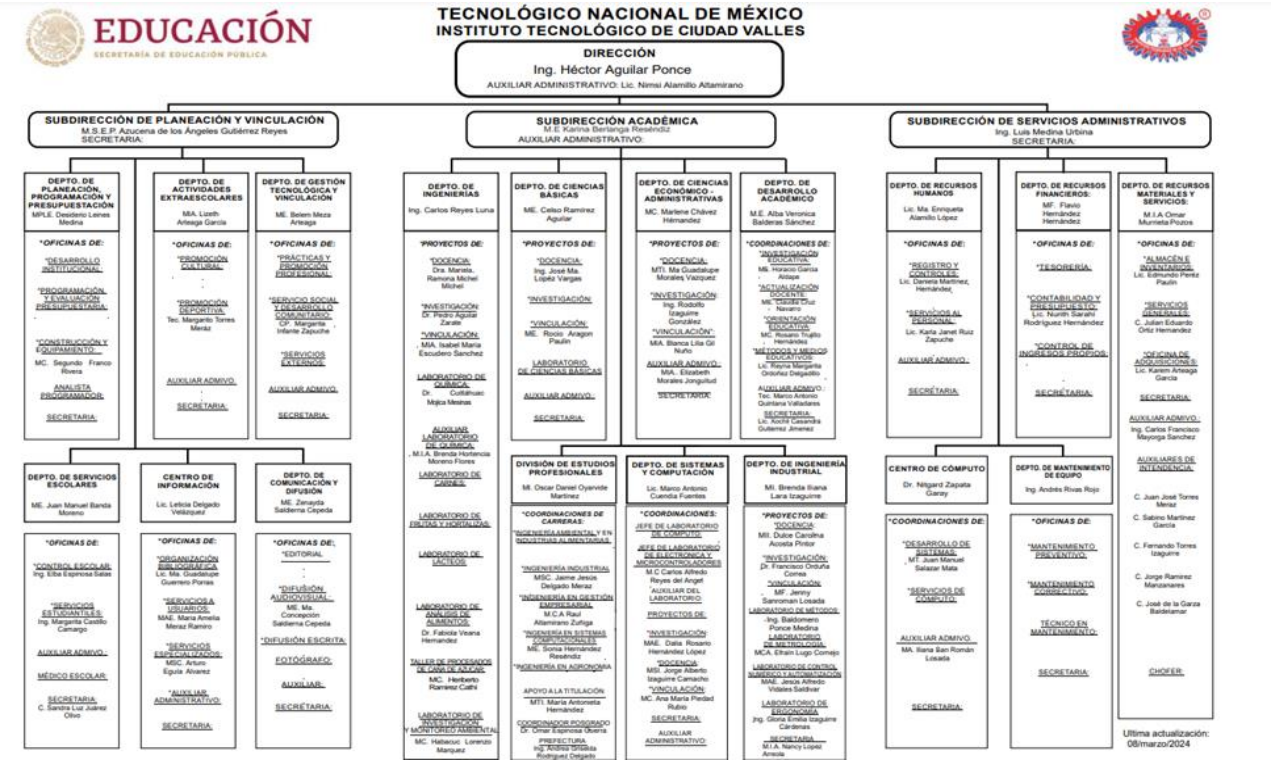

Ilustración 1 Organigrama del Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles

Tomado del sitio oficial TecNM / IT Ciudad Valles: ORGANIGRAMA – TEcNM | Instituto Tecnológico de Ciudad Valles. (2024). https://www.tecvalles.mx/wp/organigrama-4/

2.5.- Ubicación de la empresa (domicilio, teléfonos, nombre del Gerente, croquis para llegar a la empresa)

Domicilio: Carr. al Ingenio Plan de Ayala Km. 2, Col. Vista Hermosa. Cd. Valles, S.L.P. C.P. 79010

Teléfonos: (481) 381 20 44, 381 46 05 y 383 21 51

Nombre del Gerente: MAP. Héctor Aguilar Ponce

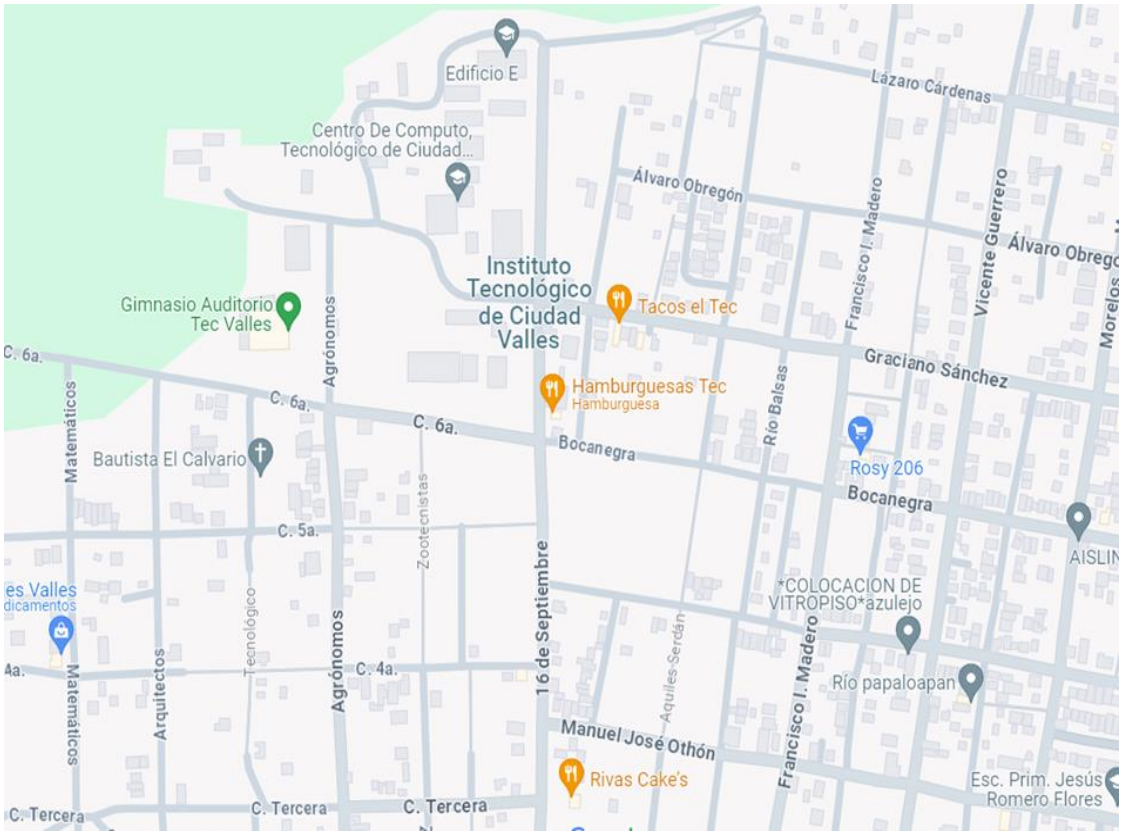

Ilustración 2 Ubicación del Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles

Tomado del sitio oficial TecNM / IT Ciudad Valles: Google Maps. (2024). Ubicación del TecNM / IT Ciudad Valles. Recuperado 30 de julio de 2024, de https://maps.app.goo.gl/RkhAtLsxs58m5m447

2.6 Área o Departamento donde se desarrolla la Residencia Profesional

Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles.

Coordinación de Actualización Docente.

Departamento de Desarrollo Académico.

3.- Problemas a resolver, priorizándolos

- Actualmente no se tiene un sistema de información con el cuál se lleve el control y registro de los cursos para los docentes de manera completamente automática, solo se utilizan algunos formularios, documentos digitales, etc.
- La información no llega de manera correcta en tiempo y forma a los docentes en algunos casos, por lo que se les dificulta saber más acerca de los cursos.
- Existen esfuerzos duplicados al realizar registro o búsquedas de la información, ya que no se cuenta con la información disponible de manera inmediata y ordenada.

4.- Objetivos (General y Específicos)

4.1.- Objetivo general

Desarrollar una plataforma web para el registro y gestión de cursos de capacitación para los docentes del Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles, con el fin de optimizar los procesos administrativos.

4.2.- Objetivos específicos

- Diseñar la interfaz de usuario que sea intuitiva y accesible que permita a los docentes y administradores navegar y utilizar la plataforma con facilidad.
- Implementar funcionalidades de registro que permitan a los docentes inscribirse en cursos de capacitación, visualizar los cursos disponibles y consultar su historial de capacitaciones.
- Automatizar el proceso de gestión de cursos para la actualización y administración de la información sobre los cursos, reduciendo la carga administrativa.
- Generación reportes y/o informes de resultados para la toma de decisiones.

5.- Justificación

La capacitación continua de los docentes es fundamental para asegurar la calidad de la enseñanza y el desarrollo profesional del personal académico. El Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles reconoce esta necesidad y busca optimizar el proceso de registro y seguimiento de los cursos de capacitación ofrecidos a sus docentes.

El proceso actual de registro de cursos de capacitación es semiautomatizado y presenta varios desafíos, como la duplicación de esfuerzos, errores en los registros, pérdida de información y una carga administrativa considerable para los docentes y el personal administrativo. Esto no solo afecta la eficiencia operativa, sino que también puede impactar negativamente en la motivación y participación de los docentes en las actividades de formación.

Para abordar estos desafíos, se propone el desarrollo de una plataforma web dedicada al registro y gestión de cursos de capacitación para los docentes del Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles. Esta plataforma permitirá automatizar el proceso de registro, reducir errores, y proporcionar una interfaz amigable y accesible para los docentes.

Este proyecto estará dividido en dos fases, la interfaz de gestión de la información y la lógica de negocios; el alcance del presente documento incluye la implementación de la lógica de negocios y el origen de datos.

6.- Marco Teórico (fundamentos teóricos)

Entre las metodologías que se pueden implementar para el desarrollo de una plataforma web se encuentran:

Metodología de cascada: también conocida como ciclo de vida de desarrollo de sistemas (SDLC, por sus siglas en inglés), es un proceso lineal en el que el trabajo se realiza de manera escalonada (similar a una cascada) y en orden secuencial. (asana, 2024)

Programación extrema (XP): se usa para gestionar proyectos dinámicos con plazos ajustados. Este enfoque se basa en la creación de ciclos de desarrollo cortos con muchas versiones. Esto genera procesos más rápidos y una mayor productividad. (asana, 2024)

Metodología Scrum: se basa en “sprints” cortos que se usan para crear un ciclo de proyecto. Estos ciclos duran de una a dos semanas y se organizan con equipos de hasta 10 personas. Este enfoque es diferente al modelo de cascada, donde las tareas individuales se dividen y relacionan por dependencias. (asana, 2024)

Metodología Kanban: representa las tareas pendientes del proyecto usando elementos visuales como tableros. Los equipos ágiles usan este enfoque para visualizar mejor los flujos de trabajo y el progreso de los proyectos. Además, ayuda a reducir la probabilidad de que se generen cuellos de botella. Por lo general, este método se aplica en una herramienta de software que te permite cambiar y arrastrar tableros dentro de los proyectos sin problemas, aunque esto no es absolutamente necesario. (asana, 2024)

Metodología en Espiral: La metodología en espiral, desarrollada por Barry Boehm en 1988, es un enfoque evolutivo para el desarrollo de software que combina

la repetición (iteraciones) con la gestión de riesgos. A diferencia de las metodologías lineales, la espiral permite revisitar fases previas del desarrollo conforme el proyecto avanza, ajustándose a nuevos requisitos o cambios en el entorno del proyecto. (Boehm, 2022)

6.1 Metodología en espiral: Para este proyecto se estará implementando la metodología en espiral (Pressman, 2010), a diferencia de las metodologías mencionadas anteriormente la espiral permite realizar iteraciones y revisiones constantes, lo cual es fundamental para adaptarse a los cambios en los requisitos del cliente, además la metodología espiral ofrece un enfoque más centrado en la gestión de riesgos y en la planificación a largo plazo, se divide en 5 etapas, la primera es la comunicación, que nos va a permitir poder estar en contacto con los asesores del proyecto, se podrá saber en todo momento si existe alguna modificación o para agregar algún extra al proyecto, después sigue la etapa de planeación, en la que se seleccionarán las herramientas que se van a utilizar, continua con el modelado, en la que se construirá el diseño del proyecto, agregando todas las funciones que este hará, al terminar el diseño se para a la construcción del proyecto, aquí es donde se comenzará a realizar el código, haciendo todas las pruebas necesarias para que todo funcione como debe, y para terminar llega la fase de despliegue, en donde se hace la entrega del proyecto y se recibe la retroalimentación, si hay algo que no quedó correctamente o se tiene alguna idea nueva para implementar se repite el espiral y comienza de nuevo la etapa de comunicación.

6.1.1 Características de la Metodología en Espiral

- **Iterativa y Evolutiva**: Permite el desarrollo en fases repetitivas (iteraciones), donde cada ciclo produce una versión más completa y refinada del software.

- **Gestión de Riesgos**: Cada ciclo comienza con la identificación y análisis de riesgos, permitiendo mitigar problemas potenciales antes de que afecten significativamente el proyecto.
- **Flexibilidad**: Es adaptable a cambios de requisitos o de entorno, lo que es crucial en proyectos donde las necesidades pueden evolucionar rápidamente.
- **Retroalimentación Continua**: Cada iteración se completa con la revisión y evaluación del progreso, proporcionando una oportunidad para obtener retroalimentación y realizar ajustes. (Boehm, 2022)

6.2 Plataforma web: es un entorno en el que los usuarios podemos llevar a cabo tareas, gestionar actividades, colaborar con otros usuarios e interactuar por medio de las herramientas y funcionalidades que ofrece dicha plataforma. Una plataforma digital es un entorno en el que los usuarios podemos llevar a cabo tareas, gestionar actividades, colaborar con otros usuarios e interactuar por medio de las herramientas y funcionalidades que ofrece dicha plataforma. (Coppola, 2023).

Visual Studio Code: Visual Studio Code es un editor de código fuente ligero pero eficaz que se ejecuta en el escritorio y está disponible para Windows, macOS y Linux. Incluye compatibilidad integrada con JavaScript, TypeScript y Node.js, y cuenta con un amplio ecosistema de extensiones para otros lenguajes y entornos de ejecución (como C++, C#, Java, Python, Go, .NET). (Microsoft, 2022)

6.3 Base de datos: Una base de datos es una recopilación organizada de información o datos estructurados, que normalmente se almacena de forma electrónica en un sistema informático. Normalmente, una base de datos está controlada por un sistema de gestión de bases de datos (DBMS). En conjunto, los datos y el DBMS, junto con las aplicaciones asociadas a ellos, reciben el nombre de sistema de bases de datos, abreviado normalmente a simplemente base de datos. (Oracle, 2020)

MySQL: es el sistema de gestión de bases de datos relacional más extendido en la actualidad al estar basada en código abierto. Desarrollado originalmente por MySQL AB, fue adquirida por Sun MicroSystems en 2008 y está su vez comprada por Oracle Corporation en 2010, la cual ya era dueña de un motor propio InnoDB para MySQL. Es un sistema de gestión de bases de datos que cuenta con una doble licencia. Por una parte, es de código abierto, pero por otra, cuenta con una versión comercial gestionada por la compañía Oracle. (Robledano, 2019)

6.4 Laravel: es un framework de PHP y es utilizado para desarrollar aplicaciones web. Laravel crea un entorno de trabajo y proporciona herramientas a los desarrolladores para ayudarles a desarrollar en PHP sus aplicaciones web. Lo que se busca con Laravel es construir aplicaciones sólidas y estables, que sean fáciles de desarrollar y la utilización de parte del código pre programada, para que pueda aprovecharse y reutilizarse, evitando así la reescritura del código en la misma aplicación. (AXARNET COMUNICACIONES S.L, 2023)

Framework: es un entorno de trabajo, que sigue un patrón o esquema estandarizado que se utiliza para desarrollar aplicaciones o cualquier tipo de software. Los framework facilitan la vida a los desarrolladores la creación de sus proyectos al facilitar un conjunto de herramientas para automatizar las tareas más comunes en la programación, aumentando así la velocidad cuando se está programando y facilitando la colaboración entre desarrolladores, al usar todas las mismas herramientas dentro del mismo framework. (AXARNET COMUNICACIONES S.L, 2023)

PHP: es el lenguaje de programación más utilizado en el mundo para desarrollar sitios web, aplicaciones web y los populares CMS, como WordPress o Joomla. (AXARNET COMUNICACIONES S.L, 2023)

Tailwind: en pocas palabras, es un framework CSS que da prioridad a la utilidad sobre el propio estilo, pero además a diferencia de otros frameworks CSS como

Bootstrap o Bulma, Tailwind no provee una serie de componentes predefinidos. En su lugar, este framework opera en un nivel inferior y te proporciona un conjunto de clases de ayuda para estructura y estilado, de forma que, usando dichas clases, puedas crear rápidamente diseños personalizados con facilidad. Además, no es opinado y gracias a su flexibilidad te permite crear un diseño realmente único. (Huet, 2022)

XAMPP: es un software gratuito de código abierto que proporciona la creación de un entorno de desarrollo local (servidor) para aplicaciones web. Es multiplataforma, trabaja tanto en Windows, Linux o Mac OS perfectamente. XAMPP es el servidor más utilizado y popular entre los desarrolladores web y programadores, porque permite instalar y configurar el entorno de un servidor local de manera sencilla. Esto es útil para el desarrollo, pruebas y depuración de aplicaciones web antes de desplegarlas en un servidor de producción en línea. (Peña, 2024)

6.5 Backend: es la parte invisible pero esencial de un sitio, encargada de manejar la lógica y el procesamiento de datos necesarios para que todo funcione de manera correcta y segura, se ocupa de tareas como almacenar y recuperar datos de una base de datos, procesar formularios, autenticar usuarios y gestionar la seguridad del sitio. Es responsable de toda la “magia” que sucede detrás de la interfaz visible, también accede al servidor, que es una aplicación especializada que entiende la forma en la que el navegador hace solicitudes. (Maldeadora, 2019)

Los desarrolladores backend: se encargan de hacer que la lógica del sitio funcione correctamente, que la información se transmita de manera segura y que el rendimiento de la aplicación no afecte la experiencia del usuario. (Maldeadora, 2019)

6.5.1 Lenguajes de programación más utilizados en el desarrollo backend: Algunos de los más populares y ampliamente adoptados son:

- **Python:** es un lenguaje de programación de alto nivel, interpretado y de propósito general. Se caracteriza por su legibilidad y simplicidad, lo que lo hace ideal tanto para principiantes como para expertos en el desarrollo de software. Python es ampliamente utilizado en desarrollo web, ciencia de datos, inteligencia artificial, y automatización. (Matthes, 2023)
- **JavaScript:** es un lenguaje de programación interpretado que se ejecuta principalmente en el lado del cliente en navegadores web, pero también se utiliza en el backend con entornos como Node.js. Es esencial para el desarrollo de aplicaciones web interactivas y dinámicas. (Crockford, 2022)
- **Ruby:** es un lenguaje de programación orientado a objetos, conocido por su elegancia y simplicidad. Es especialmente popular en el desarrollo web gracias al framework Ruby on Rails, que facilita la creación rápida de aplicaciones web robustas. (Thomas, 2022)
- **Java:** es un lenguaje de programación de propósito general, concurrente y orientado a objetos. Es ampliamente utilizado para desarrollar aplicaciones móviles (especialmente en Android), sistemas empresariales y aplicaciones web escalables. (Bloch, 2023)
- **PHP:** es un lenguaje de scripting diseñado principalmente para el desarrollo web. Permite la creación dinámica de contenido y es uno de los lenguajes más utilizados para el desarrollo de servidores web, como en sistemas de gestión de contenido (CMS) como WordPress. (Kunkel, 2023)

6.5.2 Principales tecnologías utilizadas en backend: el desarrollo backend utiliza una variedad de tecnologías para construir aplicaciones web robustas y escalables. (Maldeadora, 2019) Estas son algunas de las principales:

- **Frameworks de desarrollo:** como Django (Python), Ruby on Rails (Ruby), Laravel (PHP) y Express.js (JavaScript)

- **Bases de datos:** algunas bases de datos populares son MySQL, PostgreSQL, MongoDB y SQLite.
- **Servidores web:** como Apache, Nginx y Microsoft IIS.
- **APIs:** las APIs RESTful son comunes en el desarrollo backend y utilizan el protocolo HTTP para intercambiar información.
- **Contenedores y orquestación:** como Docker, y herramientas de orquestación, como Kubernetes, ha ganado popularidad en el desarrollo backend.

6.5.3 Importancia del Backend en Plataformas Web Educativas: El backend de una plataforma web es la parte del software que gestiona la lógica del servidor, bases de datos y la comunicación entre el servidor y los clientes. (SEOEstudios, 2022)

En el contexto de una plataforma educativa, el backend es crucial porque:

- **Gestiona el almacenamiento y acceso a los datos**: Asegura que toda la información, como datos de usuarios, cursos, y actividades, esté almacenada de forma segura y pueda ser recuperada eficientemente.
- **Soporta la autenticación y autorización de usuarios**: Implementa mecanismos para verificar la identidad de los usuarios y garantizar que solo accedan a los recursos permitidos.
- **Facilita la integración de servicios**: Permite la integración con otros sistemas educativos y herramientas externas, lo cual es esencial para un entorno de aprendizaje completo y moderno.
- **Asegura la escalabilidad y rendimiento**: Proporciona un sistema que puede manejar un gran número de usuarios simultáneos sin comprometer la velocidad y eficiencia.

6.5.4 Frameworks de Desarrollo Backend: son conjuntos de herramientas y librerías que facilitan el desarrollo de software proporcionando una estructura estándar y componentes reutilizables. En el desarrollo backend, los frameworks ayudan a

acelerar el desarrollo al ofrecer soluciones pre configuradas para tareas comunes, como la gestión de bases de datos, la seguridad y la autenticación de usuarios. Algunos frameworks populares para el desarrollo backend incluyen:

Django (Python): Un framework de alto nivel para el desarrollo rápido de aplicaciones web con Python. Django sigue el patrón de diseño Model-View-Template (MVT) y viene con muchas características incorporadas, como autenticación de usuarios, administración y protección contra ataques comunes. (Vincent, 2022)

Express.js (Node.js): Un framework minimalista para Node.js que proporciona una serie de herramientas y funcionalidades para construir aplicaciones web y APIs. Es altamente flexible y se utiliza a menudo en el desarrollo de aplicaciones rápidas y escalables.

Spring Boot (Java): Un framework de Java que facilita el desarrollo de aplicaciones independientes, basadas en Spring, con configuraciones mínimas. Es muy utilizado en aplicaciones empresariales y es conocido por su robustez y flexibilidad. (Musib, 2022)

Laravel (PHP): Un framework PHP con una sintaxis elegante y expresiva, diseñado para facilitar el desarrollo de aplicaciones web robustas. Laravel incluye una serie de características como autenticación, migraciones de base de datos y motor de plantillas, que simplifican el desarrollo backend. (Stauffer, 2023)

APIs y Servicios Web: Las APIs (Interfaz de Programación de Aplicaciones) son conjuntos de reglas y protocolos que permiten a diferentes sistemas comunicarse entre sí. En una plataforma educativa, las APIs permiten la integración de servicios externos y la comunicación entre el frontend y el backend. Los servicios web, que utilizan APIs, son esenciales para la arquitectura de la plataforma, ya que permiten el intercambio de datos y la ejecución de operaciones sobre los mismos. (Humble, 2010)

Bases de Datos: Las bases de datos son sistemas de almacenamiento que permiten organizar y gestionar grandes cantidades de información de manera eficiente. En una plataforma de formación docente, las bases de datos son fundamentales para almacenar la información de los usuarios, cursos, registros de actividades, evaluaciones y otros datos críticos. (Rendón, 2019)

Existen diferentes tipos de bases de datos que pueden ser utilizadas, dependiendo de los requerimientos específicos del proyecto:

- **Bases de Datos Relacionales:** Utilizan tablas para almacenar datos y permiten definir relaciones entre estas tablas. Son ideales para aplicaciones donde la consistencia y la integridad de los datos son cruciales. Ejemplos de sistemas de gestión de bases de datos relacionales (RDBMS) incluyen MySQL, PostgreSQL y Oracle.
- **Bases de Datos NoSQL:** Están diseñadas para manejar datos no estructurados o semi-estructurados, proporcionando flexibilidad y escalabilidad. Son útiles en situaciones donde se manejan grandes volúmenes de datos y se requieren esquemas flexibles. Ejemplos de bases de datos NoSQL incluyen MongoDB, Cassandra y DynamoDB.

6.5.5 Qué hace un desarrollador Backend: las tareas específicas de un/a desarrollador backend pueden variar según el proyecto, el tipo de aplicación o la empresa para la que trabajen, pero algunas de sus funciones más comunes son: (Canelo, 2024)

- **Gestión de bases de datos:** Se encarga de diseñar y mantener la estructura de las bases de datos utilizadas en la aplicación. Esto implica crear tablas, definir relaciones entre datos, optimizar consultas y garantizar que la información se almacene y recupere de manera eficiente.

- **Desarrollo de APIs:** Crean las APIs (Interfaces de Programación de Aplicaciones) que permiten que el frontend y otras aplicaciones se comuniquen con el servidor. Estas APIs definen cómo se deben solicitar y enviar datos entre el cliente y el servidor.
- **Lógica de negocio:** Implementan la lógica empresarial de la aplicación, que consiste en las reglas y algoritmos que gobiernan el comportamiento de la aplicación. Esto incluye procesamiento de datos, cálculos, validaciones y cualquier otra tarea relacionada con la funcionalidad principal de la aplicación.
- **Seguridad:** La seguridad es una parte crucial del trabajo de un desarrollador backend. Deben asegurarse de que los datos de los usuarios estén protegidos, evitar vulnerabilidades y prevenir ataques maliciosos, como inyecciones de código o intentos de acceso no autorizado.
- **Escalabilidad y rendimiento:** A medida que una aplicación crece en popularidad, el backend debe poder manejar un mayor número de usuarios y solicitudes sin perder rendimiento. Los desarrolladores backend trabajan para optimizar y escalar la infraestructura para garantizar una experiencia fluida para los usuarios.
- **Integración de servicios externos:** En muchas ocasiones, las aplicaciones necesitan comunicarse con servicios externos, como sistemas de pago, servicios de almacenamiento en la nube o redes sociales. Este rol se encarga de integrar estos servicios en la aplicación de manera segura y confiable.
- **Pruebas y depuración:** Realizan pruebas exhaustivas para identificar y solucionar errores y problemas de rendimiento. También colaboran con otros miembros del equipo para garantizar la calidad del software desarrollado.
- **Mantenimiento y actualizaciones:** A medida que la aplicación evoluciona, se requiere mantenerla y actualizarla para agregar nuevas características, solucionar problemas y mantenerla al día con los cambios tecnológicos.

6.5.6 El Backend permite: mejorar la experiencia de usuario, gracias al backend se puede personalizar el sitio, añadir nuevas funciones, hacerlo más óptimo y adecuarlo a las necesidades del usuario. (Ken, 2023)

1. **Intercambiar información.** Si no se cuenta con backend, el sitio no puede recibir información del usuario para ejecutar funciones como comprar, búsqueda, personalización o almacenar información de usuario.
2. **Mejorar la seguridad.** Ofrece a los usuarios una navegación en la que toda su información será resguardada, es decir, sus datos no serán vulnerados, en cuanto a los dueños de la página tendrán la tranquilidad de que al estar en línea no habrá complicaciones.
3. **Conectarse a través de múltiples dispositivos.** Los usuarios se conectan sin importar el tipo de hardware, navegador o sistema operativo que utilicen, debido a que el desarrollo backend ofrece la posibilidad de que los servidores completen los cálculos en el mismo lugar, sin necesidad de ejecutarse en la computadora del usuario, de tal forma que el funcionamiento es más fluido y efectivo.
4. **Configurar la aplicación conforme a las necesidades del usuario y negocio:** este proceso implica ajustar la aplicación de acuerdo a los requisitos específicos de los usuarios y las metas del negocio. El backend permite adaptar funciones como la personalización de la experiencia del usuario, la integración con sistemas externos y la configuración de permisos y accesos, lo cual asegura que la aplicación sea eficiente y relevante para las personas que la usan y para el negocio en su conjunto.

7.-Procedimiento y descripción de las actividades realizadas

Para este proyecto se estará implementando la metodología en espiral (Pressman, 2010),la cual permite entregar avances evolutivos, se divide en 5 etapas, la primera es la comunicación, que nos va a permitir poder estar en contacto con los asesores del proyecto, se podrá saber en todo momento si existe alguna modificación o para agregar algún extra al proyecto, después sigue la etapa de planeación, en la que se seleccionarán las herramientas que se van a utilizar, continua con el modelado, en la que se construirá el diseño del proyecto, agregando todas las funciones que este hará, al terminar el diseño se para a la construcción del proyecto, aquí es donde se comenzará a realizar el código, haciendo todas las pruebas necesarias para que todo funcione como debe, y para terminar llega la fase de despliegue, en donde se hace la entrega del proyecto y se recibe la retroalimentación, si hay algo que no quedó correctamente o se tiene alguna idea nueva para implementar se repite el espiral y comienza de nuevo la etapa de comunicación. Las etapas se describen a continuación:

7.1.- Comunicación con asesores

El día 18 de septiembre del 2024

Objetivos:

- Verificar la estructura de los proyectos relacionados.
- Establecer las mismas tecnologías y herramientas a utilizar en los proyectos.

Finalidad:

Alinear a todos los integrantes de los proyectos en cuanto a la estructura, tecnologías y herramientas a emplear, asegurando una mayor coherencia y eficiencia en el desarrollo de los mismos.

El día 7 de octubre del 2024

Objetivos:

- Definir los roles que cada integrante ocupará en los proyectos.
- Asegurar una distribución clara de responsabilidades entre los participantes.

Finalidad:

Establecer una estructura organizativa clara dentro de los proyectos, asignando roles específicos para optimizar el trabajo en equipo y garantizar una ejecución eficiente.

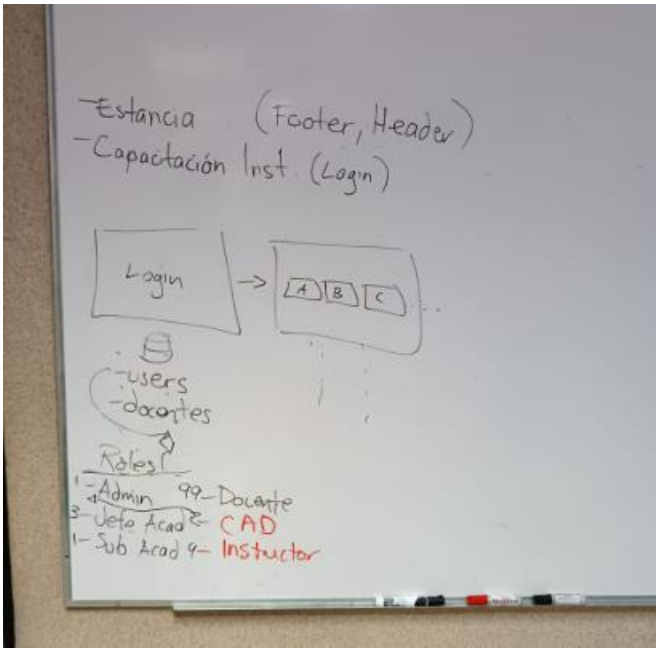

Ilustración 3 Reunión 7 Octubre

El día 17 de octubre del 2024

Objetivos:

- Restablecer la base de datos del proyecto.
- Agregar nuevas tablas con todos sus campos, asegurando la inclusión de llaves primarias y foráneas.

Finalidad:

Garantizar la integridad y organización de los datos dentro del proyecto mediante la actualización de la estructura de la base de datos.

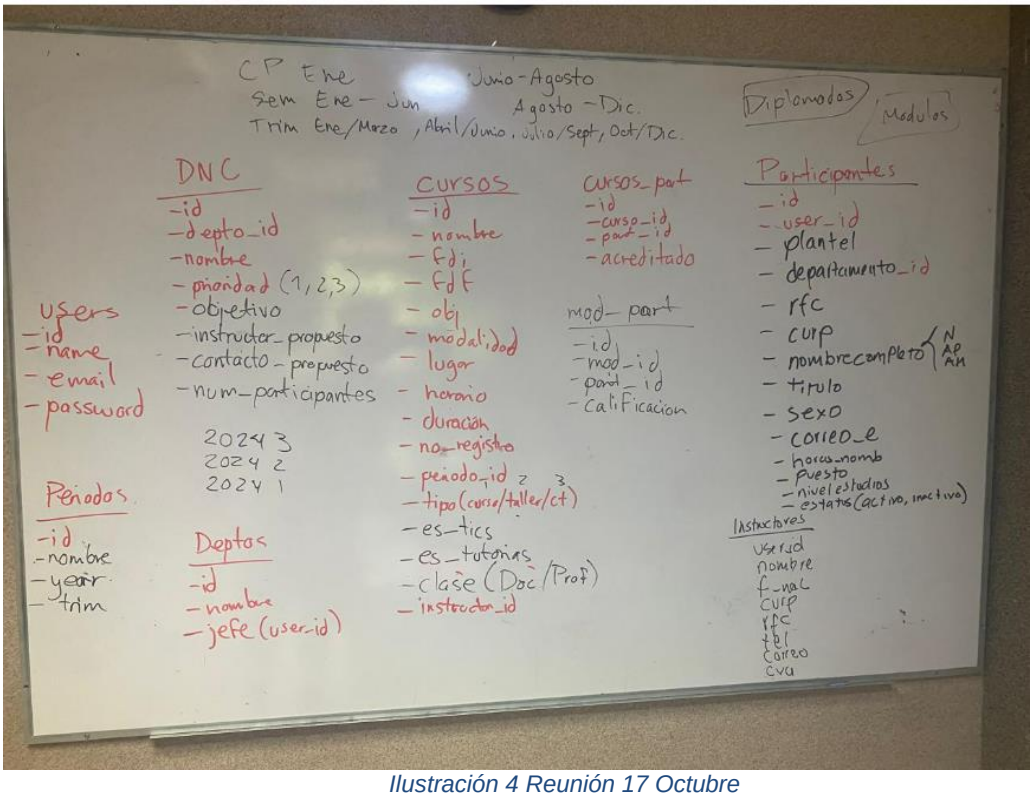

Ilustración 4 Reunión 17 Octubre

El día 4 de noviembre del 2024

Objetivos:

- Modificar la estructura para la selección de tipos y roles.
- Incluir las opciones de tipo: Docente, Administrativo, Otro.
- Incorporar los roles: Instructor, Subdirector, CAD, Jefe de Departamento.

Finalidad:

Optimizar la organización y asignación de roles en el sistema, asegurando que cada usuario pueda seleccionar correctamente su tipo y rol dentro del proyecto.

Ilustración 5 Reunión 4 Noviembre

7.2.- Planeación del proyecto

En esta etapa se discutirán las herramientas que se usarán en el proyecto, tales como lenguajes de programación, gestores de bases de datos, frameworks, arquitecturas de servidor, y estrategias de integración continua, etc.

Herramienta	Descripción	Versión
Visual Studio Code	Editor de código fuente ligero pero eficaz que se ejecuta en el escritorio y está disponible para Windows, macOS y Linux. Incluye compatibilidad integrada con JavaScript, TypeScript y Node.js, y cuenta con un amplio ecosistema de extensiones para otros lenguajes y entornos de ejecución (como C++, C#, Java, Python, Go, .NET).	1.94.2
MySQL	Es un sistema de gestión de bases de datos que cuenta con una doble licencia. Por una parte, es de código abierto, pero por otra, cuenta con una versión comercial gestionada por la compañía Oracle.	10.4.24-MariaDB
Laravel	Es un framework de PHP y es utilizado para desarrollar aplicaciones web. Laravel crea un entorno de trabajo y proporciona herramientas a los desarrolladores para ayudarles a desarrollar en PHP sus aplicaciones web.	10.31.0
PHP	Es el lenguaje de programación más utilizado en el mundo para desarrollar sitios web,	8.1.6

	aplicaciones web y los populares CMS, como WordPress o Joomla.	
Tailwind	Es un framework CSS que da prioridad a la utilidad sobre el propio estilo, pero además a diferencia de otros frameworks CSS como Bootstrap o Bulma, Tailwind no provee una serie de componentes predefinidos. En su lugar, este framework opera en un nivel inferior y te proporciona un conjunto de clases de ayuda para estructura y estilado, de forma que, usando dichas clases, puedas crear rápidamente diseños personalizados con facilidad.	3.3.5
XAMPP	Es un software gratuito de código abierto que proporciona la creación de un entorno de desarrollo local (servidor) para aplicaciones web. Es multiplataforma, trabaja tanto en Windows, Linux o Mac OS perfectamente. XAMPP es el servidor más utilizado y popular entre los desarrolladores web y programadores, porque permite instalar y configurar el entorno de un servidor local de manera sencilla.	3.3.0

7.3.- Modelado del proyecto

En esta etapa se establecerá el modelo del proyecto, es decir, las entidades y relaciones que formarán la base de datos, la lógica de negocio que gestionara las operaciones del sistema. Esto incluirá la configuración de autenticación, gestión de usuarios, almacenamiento de datos de registro y cursos, entre otros componentes esenciales.

El diagrama muestra un modelo entidad-relación que representa una base de datos diseñada para gestionar usuarios, roles, cursos, instructores, participantes y departamentos, con sus respectivas relaciones.

- Entidades principales y atributos:
  - Users: Tabla que almacena información de los usuarios, incluyendo atributos como name, email y password.
  - Roles: Define los roles disponibles en el sistema, identificados por id y nombre.
  - User_Roles: Relación de muchos a muchos entre usuarios y roles, representada mediante las llaves foráneas user_id y role_id.
  - Departamentos: Representa los departamentos dentro de la organización, con atributos como nombre.
  - Instructores: Contiene datos específicos de los instructores, como rfc, n_cedula y nombre.
  - Cursos: Tabla que define los cursos, incluyendo atributos como nombre, objetivo, duracion, y instructor.
  - Periodos: Almacena información sobre los periodos, con atributos como nombre y fechas (inicio y fin).
  - Participantes: Registra información de los participantes, incluyendo su departamento, curso inscrito, y su estado (estatus_activo_inactivo).

- DNC (Detección de Necesidades de Capacitación): Relaciona departamentos con cursos requeridos, con atributos como nombre, objetivo_propuesto y num_participantes.
- Cursos_Part: Relación entre cursos y participantes, que incluye atributos como calificacion y comentarios.

❖ Relaciones entre entidades:

- Users y Roles: Relación de muchos a muchos a través de la tabla intermedia User_Roles, permitiendo asignar múltiples roles a un usuario.
- Cursos e Instructores: Relación uno a muchos, donde un curso está asociado a un único instructor.
- Cursos y Participantes: Relación de muchos a muchos mediante la tabla Cursos_Part, que también almacena información adicional como calificaciones.
- Periodos y Cursos: Relación uno a muchos, asociando un periodo con múltiples cursos.
- Departamentos y Participantes: Relación uno a muchos, dado que un departamento puede tener múltiples participantes.
- DNC y Departamentos: Relación uno a uno o uno a muchos, vinculando necesidades de capacitación con los departamentos.
- Normalización: El modelo está diseñado para minimizar redundancias y garantizar la integridad referencial mediante el uso de llaves primarias (PK) y llaves foráneas (FK).

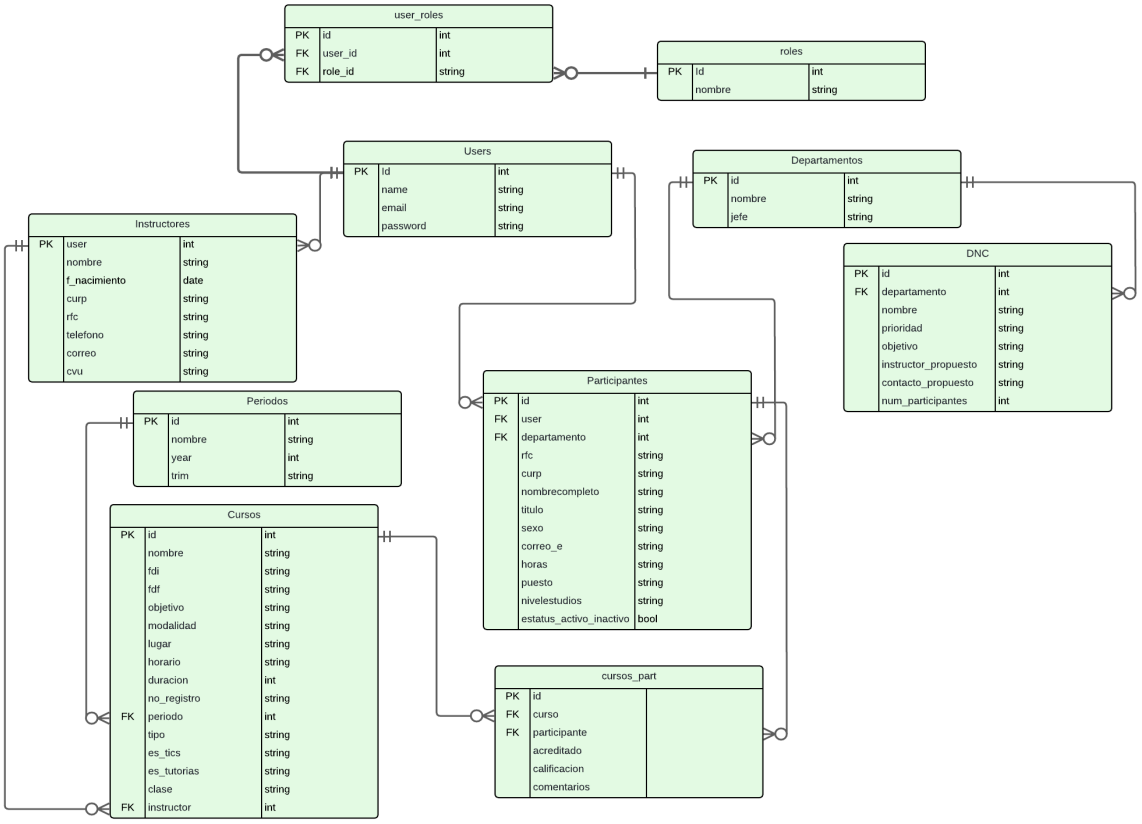

Ilustración 6 Diagrama de entidad-relación

El diagrama mostrado fue generado desde phpMyAdmin, una herramienta gráfica incluida en XAMPP, basada en la estructura de la base de datos configurada en el proyecto Laravel. Las tablas y relaciones reflejadas corresponden a las definidas mediante migraciones de Laravel. Estas relaciones fueron detectadas automáticamente por phpMyAdmin gracias a las llaves primarias y foráneas establecidas en las migraciones.

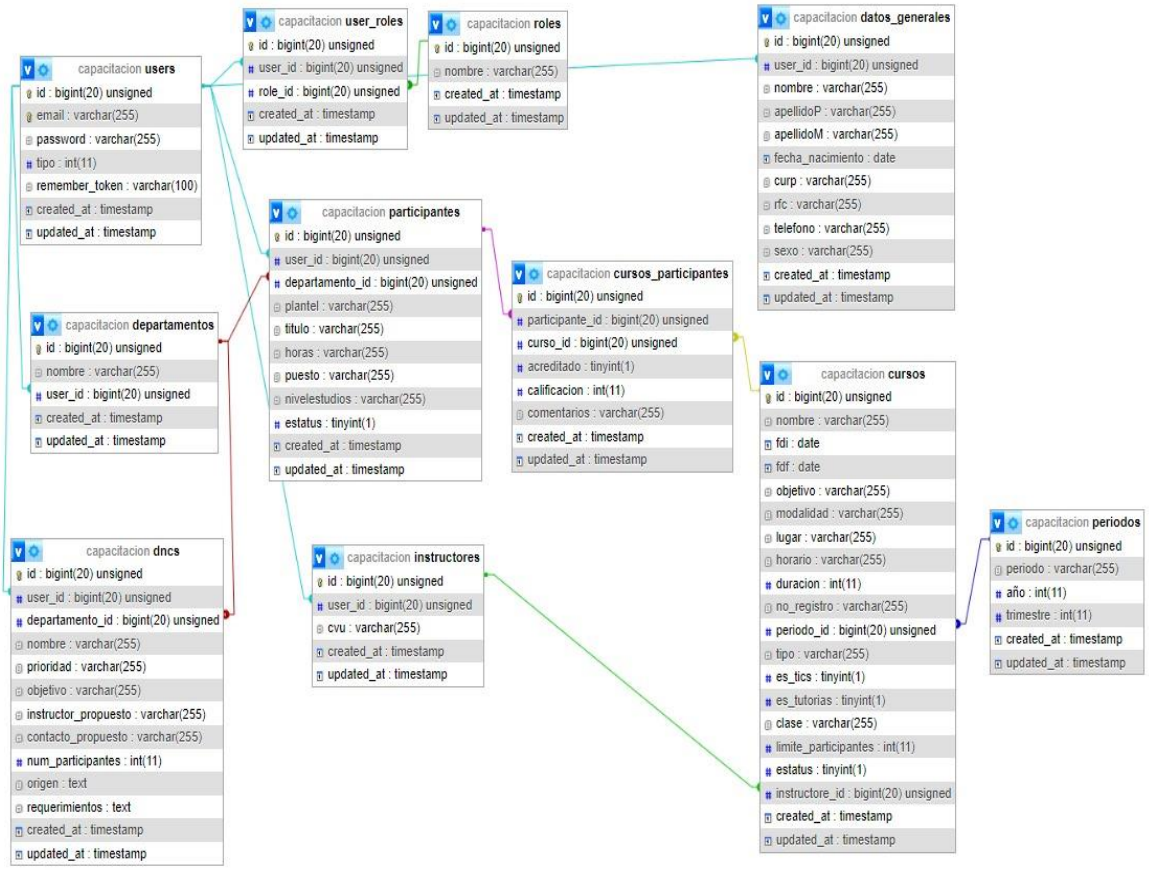

Ilustración 7 Diagrama de base de datos en Xampp desde phpMyAdmin

El diagrama es un flujo de trabajo que describe el proceso de planificación, ejecución y supervisión de cursos de formación y actualización profesional. Está estructurado en cinco áreas principales: Subdirección, Departamento de Desarrollo Académico, Departamentos Académicos, Departamento de Comunicación y Difusión, e Instructor. Cada columna indica las responsabilidades específicas de cada área en las diferentes etapas del proceso.

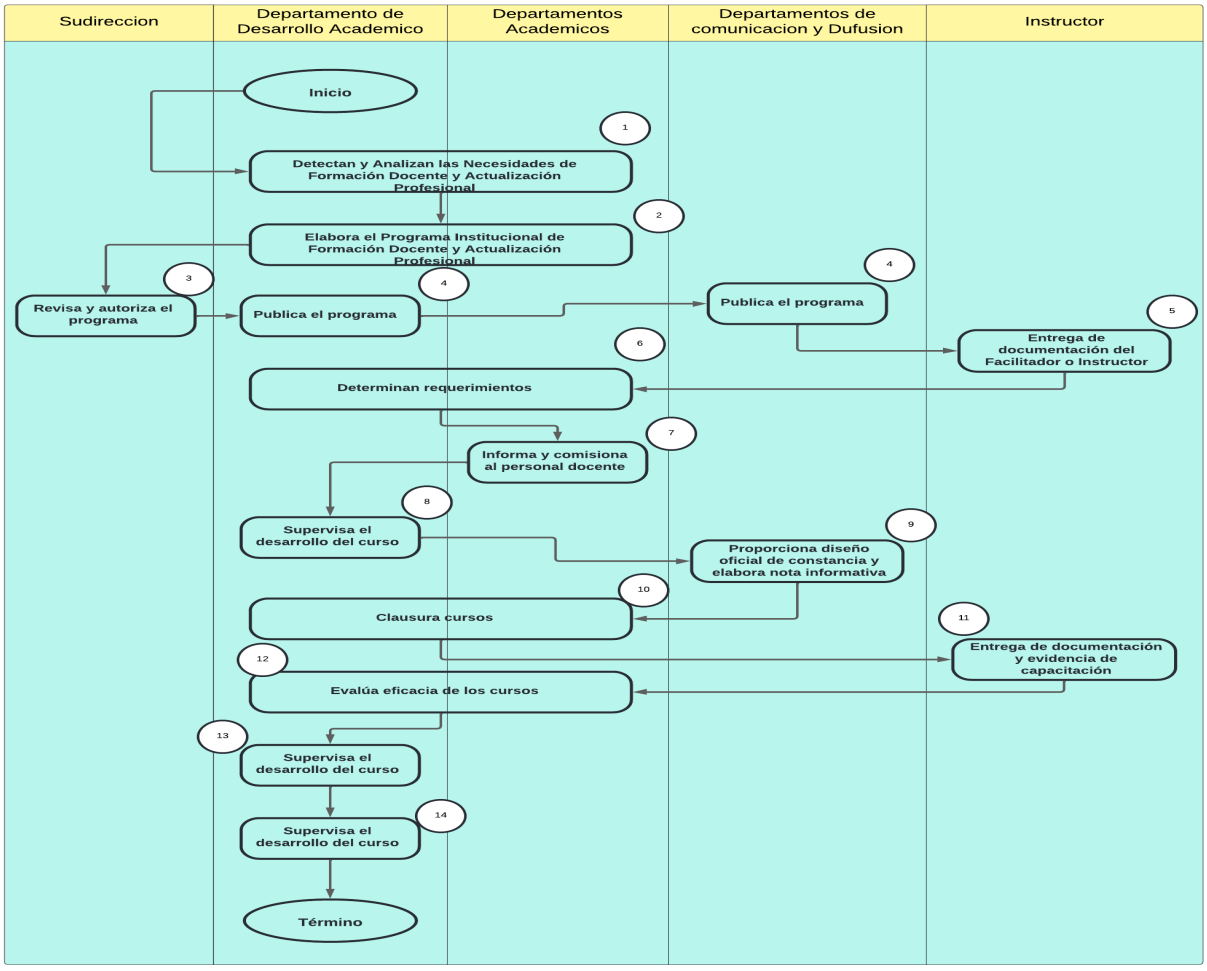

Ilustración 8 Diagrama de flujo

7. 4.- Construcción del proyecto

En esta etapa se comenzará a construir el código de todo el proyecto, teniendo en cuenta todo lo que se habló en las etapas anteriores.

7.4.1 Seeders

Los seeders en Laravel se utilizan para poblar la base de datos con datos iniciales o de prueba, lo cual es fundamental en la fase de desarrollo y en configuraciones iniciales de un proyecto.

RoleSeeder.php

El archivo RoleSeeder.php se utiliza en Laravel para poblar la tabla roles de la base de datos con un conjunto de roles predefinidos. Estos roles permiten definir los diferentes tipos de permisos y niveles de acceso dentro del sistema, asegurando que cada usuario tenga asignadas las responsabilidades y permisos correspondientes.

- La función run dentro del seeder RoleSeeder realiza los siguientes pasos:

Definición de los Roles: Se define un arreglo roles que contiene un conjunto de roles básicos que el sistema debe reconocer. Cada rol está representado como un arreglo asociativo con la clave nombre, que define el tipo de rol. Ejemplos de roles definidos incluyen "admin", "Jefe departamento", y "Instructor".

```php
public function run(): void
{
    $roles = [
        ['nombre' => 'admin'],
        ['nombre' => 'Jefe departamento'],
        ['nombre' => 'Subdirector Academico'],
        ['nombre' => 'CAD'],
        ['nombre' => 'Instructor']
    ];
```

Ilustración 9 RoleSeeder

- Inserción de Roles en la Tabla roles

A través de un ciclo foreach, cada elemento en el arreglo roles se inserta en la tabla roles usando el modelo Role. Este paso asegura que cada rol esté registrado en la base de datos y disponible para ser asignado a usuarios en el sistema.

```
    foreach($roles as $role){
        role::create($role);
    }

    $this->command->info('Roles creados');
}
```

Ilustración 10 Roles

- Importancia en el Proyecto

El seeder RoleSeeder.php es fundamental en la fase de configuración inicial del sistema, ya que establece la estructura básica de roles que permitirán la gestión de usuarios y permisos. Los roles definen los distintos niveles de acceso y funciones dentro del sistema, como administración, dirección académica y capacitación. Con esta estructura de roles, el sistema puede controlar de manera efectiva quién tiene acceso a qué funcionalidades, asegurando un flujo de trabajo organizado y seguro.

userSeeder.php

El archivo UserSeeder.php se utiliza en Laravel para poblar la tabla users en la base de datos con datos de usuarios predefinidos. Este seeder facilita la creación de cuentas de usuario básicas para pruebas y desarrollo, lo que permite verificar el funcionamiento de autenticación y manejo de usuarios en el sistema.

- ❖ Descripción de la Función run

La función run dentro del seeder UserSeeder realiza los siguientes pasos:

Definición de Usuarios: Se define un arreglo users que contiene información de usuarios iniciales que el sistema debe reconocer. Cada usuario está representado como un arreglo asociativo con claves como email, name y password, donde:

- email: El correo electrónico del usuario.
- name: El nombre de usuario.
- password: La contraseña del usuario en texto plano (en un entorno real, las contraseñas deben estar cifradas).

En este caso, se define un usuario con el correo admin@tecvalles.mx, el nombre admin, y la contraseña 12345678.

```php
public function run(): void
{
    $users = [
        ['email'=> 'admin@tecvalles.mx', 'name' => 'admin', 'password' => '12345678'],

    ];
```

Ilustración 11 Función run

Inserción de Usuarios en la Tabla users: A través de un ciclo foreach, cada elemento del arreglo users se inserta en la tabla users usando el modelo User. Esto garantiza que los usuarios predefinidos estén registrados en la base de datos y listos para iniciar sesión en el sistema.

```
    foreach($users as $user){
        User::create($user);
    }

    $this->command->info('Usuarios creados');
}
```

Ilustración 12 Usuarios

- Importancia en el Proyecto

El seeder UserSeeder.php es importante en la fase de desarrollo y configuración del sistema, ya que establece cuentas de usuario iniciales necesarias para acceder al sistema y probar funcionalidades relacionadas con autenticación, permisos y manejo de usuarios. Estos datos iniciales aseguran que los desarrolladores tengan un usuario administrador para probar configuraciones y gestionen roles y permisos dentro del sistema.

User_RoleSeeder.php

El archivo User_RoleSeeder.php en Laravel se usa para poblar la tabla user_roles de la base de datos con datos predefinidos que asignan roles específicos a usuarios. Este seeder permite definir relaciones entre usuarios y roles, asegurando que cada usuario tenga los permisos y el nivel de acceso adecuado.

- Descripción de la Función run

La función run dentro del seeder User_RoleSeeder realiza los siguientes pasos:

Definición de las Relaciones Usuario-Rol: Se define un arreglo user_roles que contiene las asignaciones de roles a usuarios. Cada asignación está representada como un arreglo asociativo con claves como user_id y role_id, que indican la relación entre un usuario y un rol específico. En este caso, el usuario con ID 1 está asociado al rol con ID 1.

```php
class User_RoleSeeder extends Seeder
{
    /**
     * Run the database seeds.
     */
    public function run(): void
    {
        $user_roles = [
            ['user_id'=> '1','role_id' => '1'],
        ];
```

Ilustración 13 Función run

Inserción de Relaciones en la Tabla user_roles: A través de un ciclo foreach, cada elemento en el arreglo user_roles se inserta en la tabla user_roles utilizando el

modelo user_role. Este paso asegura que cada relación usuario-rol se registre correctamente en la base de datos.

```
    foreach($user_roles as $user_role){
        user_role::create($user_role);
    }

    $this->command->info('Usuarios con roles creados');
}
```

Ilustración 14 roles

- Importancia en el Proyecto

El seeder User_RoleSeeder.php es crucial en la fase de configuración inicial, ya que establece las relaciones entre usuarios y roles, lo que permite gestionar permisos y accesos en el sistema. Estas asignaciones son esenciales para establecer el control de acceso y asegurar que cada usuario tenga acceso a las funcionalidades permitidas según su rol, como administración, visualización o edición de recursos específicos.

InitialSeeder.php

Este archivo seeder se encarga de insertar datos iniciales en varias tablas de la base de datos para establecer un conjunto básico de registros en el sistema. Esto permite que el sistema cuente con roles, usuarios, departamentos y otros datos necesarios para su funcionamiento desde el inicio.

- Descripción de la Función run

Usuarios (users): Se define un arreglo users que contiene un usuario inicial (el administrador) con un correo electrónico, una contraseña y un tipo de usuario. A través de un ciclo foreach, se recorre este arreglo y se crean los registros en la tabla users mediante el modelo User.

```
$users = [
    ['email'=> 'admin@tecvalles.mx', 'password' => '12345678', 'tipo' => '1']
];

foreach($users as $user){
    User::create($user);
}
```

Ilustración 15 usuarios

Roles (roles): Este bloque define un arreglo roles con diferentes roles que los usuarios pueden tener, como "admin", "Docente", "Jefe departamento", entre otros. Cada rol es insertado en la tabla roles usando el modelo Role.

```
$roles = [
    ['nombre' => 'admin'],
    ['nombre' => 'Docente'],
    ['nombre' => 'Jefe departamento'],
    ['nombre' => 'Subdirector Academico'],
    ['nombre' => 'CAD'],
    ['nombre' => 'Instructor']
];

foreach($roles as $role){
    role::create($role);
}

$this->command->info('Roles creados');
```

Ilustración 16 arreglo de roles

Relación Usuario-Rol (user_roles): Aquí se establece una relación entre el usuario creado y el rol de administrador. Esto se logra mediante la tabla user_roles, que vincula el user_id con el role_id.

```php
$user_roles = [
    ['user_id'=> '1','role_id' => '1'],
];

foreach($user_roles as $user_role){
    user_role::create($user_role);
}
```

Ilustración 17 relación usuario-rol

Departamentos (departamentos): Este bloque crea registros en la tabla departamentos con nombres de departamentos como "Ciencias Básicas" y "Sistema y computación". Esto permite que el sistema tenga áreas académicas definidas desde el inicio.

```php
$departamentos = [
    ['nombre' => 'Ciencias Basicas'],
    ['nombre' => 'Ciencias Economico - Administrativas'],
    ['nombre' => 'Sistema y computacion'],
    ['nombre' => 'Industrial'],
    ['nombre' => 'Ingenierias'],
    ['nombre' => 'Agronomia']
];

foreach($departamentos as $departamento){
    Departamento::create($departamento);
}

$this->command->info('Departamentos creados');
```

Ilustración 18 registro departamentos

Datos Generales (datos_generales): Este bloque inserta datos generales asociados al usuario administrador. Estos datos pueden incluir el nombre completo u otra información relevante del usuario.

```php
$datos_generales = [
    ['user_id' => '1', 'nombre' => 'admin']
];

foreach($datos_generales as $datos){
    Datos_generale::create($datos);
}

$this->command->info('Datos generales creados');
```

Ilustración 19 registro datos generales

Participantes (participantes): Por último, se crea un registro en la tabla participantes para el usuario administrador, incluyendo detalles como el departamento, horas, y puesto.

```php
$participantes = [
    ['user_id' => '1', 'departamento_id' => '1', 'plantel' => '.', 'horas' => '0', 'puesto' => 'admin']
];

foreach($participantes as $participante){
    Participante::create($participante);
}

$this->command->info('Participantes creados');
```

Ilustración 20 registro participantes

- Importancia en el Proyecto

Este archivo InitialSeeder.php facilita la creación de datos iniciales para comenzar el desarrollo del sistema con elementos fundamentales como roles, departamentos y datos de usuario. Esto asegura que el sistema cuente con los datos mínimos requeridos para su operación y pruebas, permitiendo a los desarrolladores trabajar con una configuración predefinida y coherente.

DatabaseSeeder.php

El DatabaseSeeder se utiliza para definir y ejecutar los seeders de todas las tablas de la base de datos. Esta clase permite especificar qué seeders se deben ejecutar en el método run(), lo que facilita el control centralizado sobre los datos iniciales que se requieren en la aplicación.

- Método run()

public function run(): void: El método run() es el núcleo del DatabaseSeeder. Laravel llama a este método cuando se ejecuta el comando php artisan db:seed. El propósito de run() es ejecutar todos los seeders necesarios para poblar la base de datos con datos iniciales o de prueba.

```
public function run(): void
{
    // \App\Models\User::factory(10)->create();

    // \App\Models\User::factory()->create([
    //     'name' => 'Test User',
    //     'email' => 'test@example.com',
    // ]);
}
```

Ilustración 21 Función run

- Código de Ejemplo (Comentado)

En el archivo, se incluye una línea de código comentada que crea diez usuarios de ejemplo en la base de datos. Esto resulta útil en etapas de prueba, ya que permite generar rápidamente varios usuarios con datos ficticios.

En la segunda línea comentada, se muestra cómo crear un usuario con atributos específicos (name y email).

7.4.2 Models

En Laravel, los models son una parte fundamental del patrón MVC (Modelo-Vista-Controlador). Representan la lógica de negocio y actúan como una abstracción de las tablas de la base de datos. Conectan tu aplicación a la base de datos, definen relaciones entre tablas, y simplifican la lógica de negocio al trabajar con los datos como si fueran objetos.

- Cada modelo está vinculado a una tabla de la base de datos.
- Permiten trabajar con datos como objetos, lo que facilita la interacción con las tablas sin escribir SQL manualmente.

User.php

El archivo User.php es el modelo de Eloquent que representa la tabla users en la base de datos dentro de un proyecto de Laravel. Este modelo gestiona las relaciones y atributos asociados a los usuarios del sistema, definiendo cómo interactúan con otras entidades de la base de datos.

- Descripción de las Funcionalidades del Modelo
- Atributos Asignables: Se especifican los campos que pueden ser rellenados masivamente mediante el atributo fillable. En este caso:
  - email: Correo electrónico del usuario.
  - password: Contraseña del usuario.
  - tipo: Tipo de usuario (puede indicar su rol o categoría).
- Atributos Ocultos: El atributo hidden define qué campos no deben ser visibles en las respuestas JSON, como:
  - password: Para proteger la contraseña del usuario.

- remember_token: Usado por Laravel para recordar sesiones de inicio de sesión.

app > Models > User.php

```php
<?php

namespace App\Models;

// use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Notifications\Notifiable;
use Laravel\Sanctum\HasApiTokens;
use App\models\Curso;
use App\models\CursoInscrito;

class User extends Authenticatable
{
    use HasApiTokens, HasFactory, Notifiable;

    /**
     * The attributes that are mass assignable.
     *
     * @var array<int, string>
     */
    protected $fillable = [
        'email',
        'password',
        'tipo'
    ];

    /**
     * The attributes that should be hidden for serialization.
     *
     * @var array<int, string>
     */
    protected $hidden = [
        'password',
        'remember_token',
    ];
```

Ilustración 22 Descripción de las Funcionalidades del Modelo

❖ Casting de Atributos: El atributo casts convierte automáticamente ciertos campos a tipos específicos. Ejemplo:

- email_verified_at se convierte a un tipo datetime.
- password se encripta automáticamente

❖ Relaciones del Modelo

- Relación hasMany con cursos_participante: Define que un usuario puede estar inscrito en varios cursos.
- Relación belongsTo con departamento: Indica que un usuario pertenece a un departamento específico.
- Relación hasOne con Participante: Define que un usuario puede tener un único registro asociado como participante.

- Relación belongsToMany con Role: Establece que un usuario puede tener múltiples roles mediante una tabla pivote user_roles.
- Relación belongsTo con Instructore: Define que un usuario puede estar vinculado a un registro en la tabla de instructores.
- Relación hasOne con Datos_generale: Un usuario tiene datos generales únicos asociados.

```php
protected $casts = [
    'email_verified_at' => 'datetime',
    'password' => 'hashed',
];
public function cursosinscritos() {
    return $this ->hasMany(cursos_participante::class);
}
public function departamento(){
    return $this ->belongsTo(departamento::class);
}
public function participante()
{
    return $this->hasOne(Participante::class, 'user_id');
}
public function roles()
{
    return $this->belongsToMany(Role::class, 'user_roles', 'user_id', 'role_id');
}
public function user_roles()
{
    return $this->belongsToMany(Role::class, 'user_roles');
}
public function instructor(){
    return $this ->belongsTo(Instructore::class);
}
public function datos_generales()
{
    return $this->hasOne(Datos_generale::class);
}
```

Ilustración 23 Relaciones del modelo

Role.php

El archivo Role.php es un modelo de Eloquent que representa la tabla roles en la base de datos de un proyecto Laravel. Este modelo define cómo interactúa la entidad "Rol" con el sistema y sus atributos principales.

- Descripción del Modelo
- Atributos Asignables: El atributo fillable define los campos que pueden ser asignados de manera masiva (mass-assignment) cuando se crean o actualizan registros. En este caso:
  - nombre: Es el nombre del rol, como "admin", "docente", "jefe de departamento", etc.

Ilustración 24 Descripción del modelo Role

- Importancia del Modelo

El modelo Role es crucial para la gestión de roles dentro del sistema. Cada rol define un conjunto de permisos o responsabilidades que un usuario puede tener en la plataforma. Este modelo se utiliza generalmente en una relación de muchos a muchos con el modelo User, permitiendo asignar uno o varios roles a cada usuario.

user_role.php

El archivo user_role.php es un modelo de Eloquent que representa una relación muchos a muchos entre los modelos User (Usuario) y Role (Rol) en un proyecto Laravel. Este modelo gestiona los registros en la tabla pivote user_roles de la base de datos.

- ❖ Atributos Asignables: La propiedad fillable define los campos de la tabla que pueden ser asignados de forma masiva (mass-assignment):
  - user_id: Representa la clave foránea que apunta al usuario.
  - role_id: Representa la clave foránea que apunta al rol.
- ❖ Relaciones Definidas

Relación con User: Define que cada registro de user_role pertenece a un único usuario, utilizando la clave foránea user_id.

Relación con Role: Define que cada registro de user_role pertenece a un único rol, utilizando la clave foránea role_id.

app > Models > user_role.php

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class user_role extends Model
{
    use HasFactory;

    protected $fillable = [
        'user_id',
        'role_id'
    ];
    public $timestamps = true;
    public function user()
    {
        return $this->belongsTo(User::class);
    }

    public function role()
    {
        return $this->belongsTo(Role::class);
    }
}

```

Ilustración 25 Descripción del modelo user_rol

Departamento.php

El archivo Departamento.php define un modelo de Eloquent que representa la tabla departamentos en el proyecto Laravel. Este modelo se utiliza para gestionar los registros de los departamentos en el sistema y establecer relaciones con otros modelos.

- Descripción del Modelo
- Atributos Asignables: La propiedad fillable define los campos de la tabla que pueden ser asignados de forma masiva (mass-assignment). En este caso:
  - nombre: Nombre del departamento.
  - user_id: Representa la clave foránea que enlaza el departamento con un usuario (como un jefe de departamento).
- Relación Definida

Relación con User: Esto indica que cada departamento pertenece a un usuario específico (por ejemplo, un jefe de departamento).

app > Models > Departamento.php

```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Departamento extends Model
{
    use HasFactory;

    protected $fillable = [
        'nombre',
        'user_id'
    ];

    public function user()
    {
        return $this->belongsTo(User::class, 'user_id');
    }
}
```

Ilustración 26 Descripción del modelo departamento

Clave foránea: El atributo 'user_id' en la tabla departamentos enlaza al usuario correspondiente en la tabla users.

Curso.php

El archivo Curso.php define un modelo de Eloquent en Laravel que representa la tabla cursos. Este modelo es esencial para gestionar los cursos que se ofrecen dentro del sistema, incluyendo sus características, relaciones con participantes, instructores y períodos.

- Atributos Asignables: La propiedad fillable permite la asignación masiva (mass-assignment) de los siguientes atributos:
  - nombre: Nombre del curso.
  - fdi: Fecha de inicio del curso.
  - fdf: Fecha de finalización del curso.
  - objetivo: Descripción del objetivo del curso.
  - modalidad: Modalidad del curso (presencial, en línea, etc.).
  - lugar: Ubicación donde se llevará a cabo el curso.
  - horario: Horario del curso.
  - duracion: Duración total del curso en horas.
  - no_registro: Número de registro del curso.
  - periodo_id: Relación con el período académico.
  - tipo: Tipo de curso.
  - es_tics: Indica si el curso está relacionado con TICs.
  - es_tutorias: Indica si el curso está relacionado con tutorías.
  - clase: Clase o categoría del curso.
  - instructor_id: Relación con el instructor que imparte el curso.
  - limite_participantes: Número máximo de participantes permitidos.
  - estatus: Estado actual del curso (activo, inactivo, etc.).

```php
protected $fillable = [
    'nombre',
    'fdi',
    'fdf',
    'objetivo',
    'modalidad',
    'lugar',
    'horario',
    'duracion',
    'no_registro',
    'periodo_id',
    'tipo',
    'es_tics',
    'es_tutorias',
    'clase',
    'instructor_id',
    "limite_participantes",
    "estatus",
];
```

Ilustración 27 Atributos Asignables

- Ejemplo de uso:

```php
Curso::create([
    'nombre' => 'Introducción a Laravel',
    'fdi' => '2024-01-15',
    'fdf' => '2024-01-20',
    'objetivo' => 'Aprender los fundamentos de Laravel',
    'modalidad' => 'En línea',
    'lugar' => 'Plataforma Zoom',
    'horario' => '10:00 - 13:00',
    'duracion' => 15,
    'no_registro' => 'C001',
    'periodo_id' => 1,
    'tipo' => 'Técnico',
    'es_tics' => true,
    'es_tutorias' => false,
    'clase' => 'Curso',
    'instructor_id' => 2,
    'limite_participantes' => 30,
    'estatus' => 'Activo',
]);
```

Ilustración 28 Ejemplo de uso

- Relaciones Definidas

Relación con Cursos_Participante: Define una relación de uno a muchos con el modelo Cursos_Participante, que actúa como tabla intermedia entre cursos y participantes.

Relación con Participante: Define una relación de muchos a muchos con el modelo Participante, a través de la tabla intermedia cursos_participantes.

Relación con Periodo: Define una relación de uno a muchos (inversa), ya que un curso pertenece a un período

Relación con Instructore: Define una relación de uno a muchos (inversa) con el modelo Instructore. Un curso está asociado a un instructor específico.

Relación con User: Define una relación de uno a muchos (inversa) con el modelo User. Esta relación es útil si se asocia un curso a un usuario (por ejemplo, quien lo creó).

```php
public function cursos_participantes()
{
    return $this->hasMany(Cursos_Participante::class, 'curso_id');
}
public function participantes()
{
    return $this->belongsToMany(Participante::class, 'cursos_participantes', 'curso_id', 'participante_id');
}
public function periodo()
{
    return $this->belongsTo(periodo::class);
}
public function instructore()
{
    return $this->belongsTo(Instructore::class);
}
public function user()
{
    return $this->belongsTo(User::class, 'user_id');
}
}
```

Ilustración 29 Relaciones definidas

Instructore.php

El archivo Instructore.php define el modelo de Eloquent que representa a los instructores en el sistema. Este modelo permite gestionar la relación entre los usuarios y sus datos específicos como instructores, asegurando que puedan ser vinculados correctamente con cursos y otros elementos del sistema.

- Descripción del Modelo
- Atributos Asignables: La propiedad fillable define los atributos que pueden ser asignados de manera masiva (mass-assignment). Estos son:
  - user_id: Referencia al usuario que actúa como instructor.
  - cvu: Información curricular (CVU) del instructor.
- Relaciones Definidas

Relación con User:

- Define una relación de uno a muchos (inversa) con el modelo User, indicando que cada instructor está asociado a un único usuario.
- Esta relación utiliza la clave foránea user_id.

Ilustración 30 Descripción del modelo instructor

Participante.php

El archivo Participante.php define el modelo Eloquent que representa a los participantes en el sistema. Este modelo permite gestionar a los usuarios que toman cursos y registrar información relevante, como su departamento, estudios y estado en el sistema.

- Atributos Asignables: La propiedad fillable define los atributos que pueden ser asignados masivamente (mass-assignment). Estos incluyen:
  - user_id: Referencia al usuario asociado.
  - departamento_id: Identificador del departamento al que pertenece.
  - plantel: Plantel del participante.
  - titulo: Título académico del participante.
  - horas: Número de horas acumuladas en cursos.
  - puesto: Puesto que ocupa el participante.
  - nivelestudios: Nivel de estudios del participante.
  - estatus: Estado actual del participante.

```php
protected $fillable = [
    'user_id',
    'departamento_id',
    'plantel',
    'titulo',
    'horas',
    'puesto',
    'nivelestudios',
    'estatus'
];
```

Ilustración 31 Atributos asignables

❖ Ejemplo de uso:

```php
Participante::create([
    'user_id' => 1, // ID del usuario existente
    'departamento_id' => 2, // ID del departamento
    'plantel' => 'Plantel Central',
    'titulo' => 'Licenciado en Informática',
    'horas' => 40,
    'puesto' => 'Docente',
    'nivelestudios' => 'Maestría',
    'estatus' => 'Activo'
]);
```

Ilustración 32 Ejemplo de uso

❖ Relaciones Definidas

Relación con User:

- Define una relación de uno a muchos (inversa) con el modelo User, donde un participante está vinculado a un único usuario.
- Utiliza la clave foránea user_id.

❖ Relación con Departamento:

Define una relación de uno a muchos (inversa) con el modelo Departamento, indicando que un participante pertenece a un departamento específico.

Utiliza la clave foránea departamento_id.

❖ Relación con Curso:

Define una relación de muchos a muchos con el modelo Curso a través de la tabla pivote cursos_participantes.

❖ En esta relación:

- participante_id es la clave foránea del participante.
- curso_id es la clave foránea del curso.

```php
public function user()
{
    return $this->belongsTo(User::class, 'user_id');
}

public function departamento()
{
    return $this->belongsTo(Departamento::class, 'departamento_id');
}

public function cursos()
{
    return $this->belongsToMany(Curso::class, 'cursos_participantes', 'participante_id', 'curso_id');
}
```

Ilustración 33 Relaciones definidas

❖ Importancia del Modelo

El modelo Participante permite:

- Administrar los datos personales y académicos de los usuarios que participan en cursos.
- Relacionar participantes con los departamentos a los que pertenecen.
- Registrar la participación en cursos, junto con información complementaria como horas acumuladas o nivel de estudios.

cursos_participante.php

Este modelo almacena información adicional sobre la relación entre un participante y un curso, como si fue acreditado, la calificación obtenida y comentarios relacionados.

- Atributos Asignables: La propiedad fillable define los atributos que pueden ser asignados masivamente (mass-assignment). En este caso:
  - participante_id: ID del participante inscrito en el curso.
  - curso_id: ID del curso en el que está inscrito.
  - acreditado: Indica si el participante acreditó el curso (valor booleano o similar).
  - calificacion: Calificación obtenida por el participante en el curso.
  - comentarios: Comentarios relacionados con el desempeño del participante.

```php
protected $fillable = [
    'participante_id',
    'curso_id',
    'acreditado',
    'calificacion',
    'comentarios'
];
```

Ilustración 34 Atributos Asignables

- Ejemplo de uso:

```php
cursos_participante::create([
    'participante_id' => 1, // ID del participante
    'curso_id' => 3,        // ID del curso
    'acreditado' => true,
    'calificacion' => 95,
    'comentarios' => 'Excelente participación en el curso'
]);
```

Ilustración 35 Ejemplo de uso

DNC.php

El modelo dnc representa una tabla en la base de datos que se utiliza para gestionar las Detecciones de Necesidades de Capacitación (DNC). Este archivo está configurado con Laravel Eloquent y contiene la definición de los atributos y relaciones necesarios para su funcionalidad.

- Atributos Asignables: La propiedad fillable contiene los atributos que pueden ser asignados masivamente (mass-assignment). Los atributos del modelo son:
  - user_id: ID del usuario que generó la solicitud de DNC.
  - departamento_id: ID del departamento asociado a la solicitud.
  - nombre: Nombre o título de la necesidad detectada.
  - prioridad: Nivel de prioridad asignado a la solicitud.
  - objetivo: Propósito principal de la capacitación.
  - instructor_propuesto: Nombre del instructor sugerido para impartir el curso.
  - contacto_propuesto: Detalle del contacto propuesto para el instructor.
  - num_participantes: Número de participantes previstos para la capacitación.
  - origen: Fuente u origen de la solicitud de DNC.

- requerimientos: Detalles adicionales o necesidades específicas para llevar a cabo la capacitación.

```php
protected $fillable = [
    'user_id',
    'departamento_id',
    'nombre',
    'prioridad',
    'objetivo',
    'instructor_propuesto',
    'contacto_propuesto',
    'num_participantes',
    'origen',
    'requerimientos'
];
```

Ilustración 36 Atributos asignables

❖ Ejemplo de uso:

```php
dnc::create([
    'user_id' => 1,
    'departamento_id' => 3,
    'nombre' => 'Capacitación en Metodologías Ágiles',
    'prioridad' => 'Alta',
    'objetivo' => 'Implementar metodologías ágiles en proyectos internos',
    'instructor_propuesto' => 'Juan Pérez',
    'contacto_propuesto' => 'juan.perez@example.com',
    'num_participantes' => 15,
    'origen' => 'Encuesta interna',
    'requerimientos' => 'Sala con proyector y conexión a internet'
]);
```

Ilustración 37 Ejemplo de uso

❖ Relaciones Definidas

Relación con User:

- Se establece una relación de uno a muchos con el modelo User, a través de la clave foránea user_id.

Relación con Departamento:

Se establece una relación de uno a muchos con el modelo Departamento, a través de la clave foránea departamento_id.

```php
public function user()
{
    return $this->hasMany(User::class, 'user_id');
}

public function departamento()
{
    return $this->hasMany(Departamento::class, 'departamento_id');
}
```

Ilustración 38 Relaciones definidas

❖ Importancia del Modelo

El modelo dnc es clave en el sistema para gestionar las necesidades de capacitación de los departamentos. Cada registro de esta tabla almacena información crítica como el usuario que lo creó, el departamento involucrado, el objetivo de la capacitación y otros detalles esenciales.

7.4.3 Controllers

InicioController.php

El archivo InicioController.php es un controlador de Laravel, y su función principal es manejar las solicitudes relacionadas con la vista inicial de tu aplicación.

- Definición de la Clase

La clase InicioController extiende de Controller, lo que significa que hereda las funcionalidades básicas que Laravel ofrece para controladores, como middleware o métodos comunes.

InicioController.php

app > Http > Controllers > InicioController.php

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Models\User;

use Illuminate\Support\Facades\Auth;

class InicioController extends Controller
{
    public function index(){
        if (Auth::check()) {
            return view('layouts.dashboard');
        }
    }
}
```

Ilustración 39 InicioController

- Método index

Este método es el encargado de manejar las solicitudes que apuntan a la ruta inicial o de inicio del sistema.

UsuarioController.php

El controlador UsuarioController se encarga de gestionar las operaciones relacionadas con los usuarios en el sistema. Está diseñado para realizar funciones de administración, como listar usuarios, y borrar usuarios, además de condicionar el acceso según los roles del usuario autenticado.

- Método index

Función: Muestra una lista de todos los usuarios registrados, accesible solo para los roles admin y CAD.

Lógica de Control:

- Verifica si el usuario está autenticado (Auth::id()).
- Obtiene los roles del usuario autenticado y los transforma en un arreglo.
- Si el usuario tiene el rol admin o CAD, se recuperan todos los usuarios con User::all() y se muestra la vista vistas.usuarios.admin.index.
- Si el usuario no tiene los permisos adecuados, es redirigido a la página principal (definida en redirect).

Devolución:

Vista: vistas.usuarios.admin.index, con la variable usuarios que contiene la lista de usuarios.

```php
public function index()
{
    if (Auth::id()) {
        $userRole = Auth::user()->roles->pluck('nombre')->toArray();
        if(in_array('admin', $userRole) or in_array('CAD', $userRole))
        {
            $usuarios = User::all();

            return view('vistas.usuarios.admin.index', compact('usuarios'));
        }
        return redirect(RouteServiceProvider::HOME);
    }
}
```

Ilustración 40 Método index

- Método create

Función: Presenta un formulario para crear un nuevo usuario.

- Método store

Función: Procesa y guarda un nuevo usuario en la base de datos a partir de los datos enviados desde un formulario.

- Método show

Función: Muestra la información de un usuario específico según su ID.

- Método edit

Función: Presenta un formulario para editar los datos de un usuario existente.

❖ Método update

Función: Procesa los datos enviados desde el formulario de edición y actualiza la información de un usuario en la base de datos.

```php
public function create()
{
    //
}
/**
 * Store a newly created resource in storage.
 */
public function store(Request $request)
{
    //
}
/**
 * Display the specified resource.
 */
public function show(string $id)
{
    //
}
/**
 * Show the form for editing the specified resource.
 */
public function edit(string $id)
{
    //
}
/**
 * Update the specified resource in storage.
 */
public function update(Request $request, string $id)
{
    //
}
```

Ilustración 41 Métodos

❖ Método destroy

Función: Elimina un usuario específico de la base de datos.

Parámetro:

User usuario: Usa inyección de dependencias para recibir directamente el modelo del usuario a eliminar.

Lógica:

- El método delete() se llama sobre el modelo usuario, eliminándolo de la base de datos.
- Después, redirige a la lista de usuarios (usuarios.index).

Devolución:

Redirección a la vista que lista los usuarios.

```php
public function destroy(User $usuario)
{
    $usuario->delete();
    return redirect(route('usuarios.index'));
}
```

Ilustración 42 Método destroy

CursoController.php

El controlador CursoController gestiona todas las operaciones relacionadas con los cursos, incluyendo su creación, edición, listado, inscripción de participantes, inicio y finalización.

❖ Métodos Principales

Listar Cursos

- index: Lista todos los cursos con información de su instructor y periodo para cualquier usuario autenticado.
- admin_index: Muestra los cursos en una vista exclusiva para administradores.
- docente_index: Lista cursos donde un docente no está inscrito.

```php
public function index(Request $request)
{
    if (Auth::id()) {
        $cursos = Curso::with(['periodo', 'instructore'])->get();
        return view('vistas.cursos.index', compact('cursos'));
    }
}

public function admin_index(Request $request)
{
    if (Auth::id()) {
            $cursos = Curso::with(['periodo', 'instructore'])->get();
            return view('vistas.cursos.admin.index', compact('cursos'));
    }
}

public function docente_index(Request $request)
{
    if (Auth::id()) {
        $participante = $request->user()->participante;

        // Obtener cursos donde el docente no está inscrito
        $cursos = Curso::whereDoesntHave('cursos_participantes', function ($query) use ($participante) {
            $query->where('participante_id', $participante->id);
        })->get();
        return view('vistas.cursos.docente.index', compact('cursos'));
    }
}
```

Ilustración 43 Listar cursos

❖ Crear Cursos

- create: Muestra el formulario de creación de cursos, cargando datos como periodos, instructores y departamentos relacionados.
- store: Valida los datos enviados por el formulario, crea un nuevo curso y lo guarda en la base de datos.

```php
public function create($id)
{
    $solicitarcurso = dnc::find($id);
    $departamento = Departamento::find($solicitarcurso->id);
    $instructores = Instructore::all();
    $periodos = Periodo::orderBy('created_at')->get();

    return view('vistas.cursos.admin.create', compact('solicitarcurso', 'periodos', 'departamento', 'instructores'));
}
```

Ilustración 44 Create

```php
public function store(Request $request)
{
    // Validación de formularios
    $request->validate([
        'nombre' => 'required',
        'fecha_inicio' => 'required|date',
        'fecha_final' => 'required|date',
        'objetivo' => 'required',
        'modalidad' => 'required',
        'lugar' => 'required',
        'horario' => 'required',
        'duracion' => 'required|integer|min:1',
        'no_registro' => 'required',
        'periodo' => 'required',
        'tipo' => 'required',
        'clase' => 'required',
        'limite_participantes' => 'required',
        'es_tics' => 'sometimes|boolean',
        'es_tutorias' => 'sometimes|boolean'
    ]);
```

Ilustración 45 Store

❖ Editar Cursos

- edit: Carga los datos del curso seleccionado para mostrarlos en un formulario de edición.
- update: Actualiza los datos de un curso existente con la información del formulario enviado.

```php
public function edit($id)
{
    $curso = Curso::with(['instructore','periodo'])->find($id);
    $instructores = Instructore::all();
    $periodos = Periodo::orderBy('created_at')->get();
    return view('vistas.cursos.admin.edit', compact('curso', 'periodos', 'instructores'));
}

/**
 * Update the specified resource in storage.
 */
public function update(Request $request, Curso $curso)
{
    $curso->nombre = $request->nombre;
    $curso->fdi = $request->fdi;
    $curso->fdf = $request->fdf;
    $curso->objetivo = $request->objetivo;
    $curso->modalidad = $request->modalidad;
    $curso->lugar = $request->lugar;
    $curso->horario = $request->horario;
    $curso->duracion = $request->duracion;
    $curso->no_registro = $request->no_registro;
    $curso->periodo_id = $request->periodo;
    $curso->tipo = $request->tipo;
    $curso->clase = $request->clase;
    $curso->limite_participantes = $request->limite_participantes;
    $curso->es_tics = $request->es_tics;
    $curso->es_tutorias = $request->es_tutorias;
    $curso->instructore_id = $request->instructor;
    $curso->save();
    return redirect(route('admin_cursos.index'));
}
```

Ilustración 46 Edit-Update

❖ Visualizar Detalles

- show: Muestra información detallada de un curso, incluyendo los participantes inscritos.

- admin_show: Similar a show, pero para administradores.

```php
public function show($id)
{
    if (Auth::id()) {
        $curso = Curso::find($id);
        $ParticipantesInscritos = cursos_participante::where('curso_id', $id)->with(['participante'])->get();
        return view('vistas.cursos.show', compact('curso', 'ParticipantesInscritos'));
    }
}

public function admin_show($id)
{
    if (Auth::id()) {
        $curso = Curso::find($id);
        $ParticipantesInscritos = cursos_participante::where('curso_id', $id)->with(['participante'])->get();
        return view('vistas.cursos.admin.show', compact('curso', 'ParticipantesInscritos'));
    }
}

/**
 * Show the form for editing the specified resource.
 */
public function edit($id)
{
    $curso = Curso::with(['instructore','periodo'])->find($id);
    $instructores = Instructore::all();
    $periodos = Periodo::orderBy('created_at')->get();
    return view('vistas.cursos.admin.edit', compact('curso', 'periodos', 'instructores'));
}
```

Ilustración 47 Visualizar detalles

❖ Cambiar Estado de un Curso

iniciar_curso: Cambia el estado del curso a "activo" (estatus = 1).

terminar_curso: Cambia el estado del curso a "finalizado" (estatus = 0).

```php
public function terminar_curso(Curso $curso)
{
    $curso->estatus = '0';
    $curso->save();
    return back();
}

public function iniciar_curso(Curso $curso)
{
    $curso->estatus = '1';
    $curso->save();
    return back();
}
```

Ilustración 48 Iniciar curso-terminar

- Eliminar Cursos

destroy: Elimina un curso específico de la base de datos.

```
    */
    public function destroy(Curso $curso)
    {
        $curso->delete();
        return redirect(route('admin_cursos.index'));
    }
}
```

Ilustración 49 destroy

CursoInscritoController.php

Este controlador gestiona las inscripciones de los usuarios en los cursos, permitiendo consultar, registrar y eliminar cursos inscritos.

- ❖ Métodos Principales
- ❖ Mostrar Cursos Inscritos o Finalizados

index:

- Si la acción es inscrito, muestra los cursos en los que el usuario está actualmente inscrito.
- Si la acción es cursado, lista los cursos ya terminados por el usuario.
- Las vistas varían según el tipo de acción:
- Cursos inscritos: vistas.docente.cursos.inscritos.index.
- Cursos terminados: vistas.docente.cursos.terminados.index.

❖ Registrar Inscripción a un Curso

```php
public function index(Request $request)
{
    if ($request->input('action') === 'inscrito') {
        $cursosInscritos = $request->user()->cursosInscritos;
        return view('vistas.docente.cursos.inscritos.index', compact('cursosInscritos'));
    }
    if ($request->input('action') === 'cursado') {
        $cursosInscritos = $request->user()->cursosInscritos;
        return view('vistas.docente.cursos.terminados.index', compact('cursosInscritos'));
    }
}
```

Ilustración 50 Index

store:

- Crea una nueva inscripción del usuario autenticado en un curso específico.
- Se utiliza el modelo cursos_participante para registrar:
- El ID del usuario.
- El ID del curso.
- Guarda el registro en la base de datos y redirige de vuelta.

```php
public function store(Request $request)
{
    $cursoPartii = new cursos_participante();
    $cursoPartii->user_id = auth()->user()->id;
    $cursoPartii->curso_id = $request->curso_id;

    $cursoPartii->save();
    return back();
}
```

Ilustración 51 Store

❖ Eliminar Inscripción

destroy:

- Busca un curso inscrito por su ID y lo elimina.
- Después, redirige al usuario a la página anterior.

```php
public function destroy($id)
{
    $cursoInscrito = CursoInscrito::findOrFail($id);
    $cursoInscrito->delete();

    return back();
}
```

Ilustración 52 Destroy

CursoParticipanteController.php

Este controlador gestiona las relaciones entre los participantes y los cursos, como inscripciones, visualización de cursos en progreso o terminados, y la eliminación de inscripciones.

- ❖ Mostrar Cursos por Estado
  - cursando_index:
  - Recupera los cursos en los que el participante está actualmente inscrito y en progreso (estatus = 1).
- ❖ Filtra usando:
  - participante_id: Relacionado al usuario autenticado.
  - whereHas: Confirma que el curso relacionado tiene el estado activo.
  - Retorna una vista: vistas.cursos.docente.cursando.index.
- ❖ terminados_index:
  - Similar a cursando_index, pero filtra los cursos terminados (estatus = 0).
  - Vista asociada: vistas.cursos.docente.terminados.index.

```php
public function cursando_index(Request $request)
{
    $participante = $request->user()->participante->id;
    $cursosCursando = cursos_participante::where('participante_id', $participante)
                        ->whereHas('curso', function ($query) {
                            $query->where('estatus', '1');
                        })
                        ->get();

    return view('vistas.cursos.docente.cursando.index', compact('cursosCursando'));
}

public function terminados_index(Request $request)
{
    $participante = $request->user()->participante->id;
    $cursosTerminados = cursos_participante::where('participante_id', $participante)
                        ->whereHas('curso', function ($query) {
                            $query->where('estatus', '0');
                        })
                        ->get();
    return view('vistas.cursos.docente.terminados.index', compact('cursosTerminados'));
}
```

Ilustración 53 Cursos por estado

❖ Inscripción a un Curso

store:

- Crea una nueva relación entre un participante y un curso.
- Usa el modelo cursos_participante para almacenar:
- El ID del participante.
- El ID del curso.
- Guarda los datos en la base y redirige a la página anterior.

❖ Detalles de un Curso Terminado

show:

- Busca un curso específico por su ID usando el modelo Curso.
- Retorna una vista para mostrar los detalles del curso: vistas.cursos.docente.terminados.show.

```php
public function store(Request $request)
{
    $cursoParticipante = new cursos_participante();
    $cursoParticipante->participante_id = $request->user()->participante->id;
    $cursoParticipante->curso_id = $request->curso_id;

    $cursoParticipante->save();
    return back();
}

/**
 * Display the specified resource.
 */
public function show($id)
{
    $curso = Curso::find($id);
    return view('vistas.cursos.docente.terminados.show', compact('curso'));
}
```

Ilustración 54 Store-show

❖ Eliminación de Inscripción

destroy:

- Elimina la inscripción de un participante a un curso específico.
- Usa el modelo cursos_participante.
- Redirige de vuelta a la página anterior.

```php
public function destroy(cursos_participante $participanteInscrito)
{
    $participanteInscrito->delete();

    return back();
}
```

Ilustración 55 Destroy

SolicitarCursoController.php

Este controlador gestiona las solicitudes de cursos (DNC - Detección de Necesidades de Capacitación) realizadas por los usuarios, permitiendo a diferentes roles interactuar con las solicitudes según sus permisos.

❖ Roles y Vistas Asociadas

admin_index:

- Recupera todas las solicitudes de cursos para el admin.
- Vista: vistas.DNC.admin.index.

jefe_departamento_index:

- Recupera las solicitudes de cursos creadas por el jefe de departamento autenticado.
- Vista: vistas.DNC.jefe_departamento.index.

```php
public function admin_index()
{

    if (Auth::id()) {

        $solicitarCursos = dnc::get();
        return view('vistas.DNC.admin.index', compact('solicitarCursos'));

    }
}

public function jefe_departamento_index()
{

    if (Auth::id()) {

        $solicitarCursos = dnc::where('user_id', auth()->user()->id)->get();
        return view('vistas.DNC.jefe_departamento.index', compact('solicitarCursos'));

    }
}
```

Ilustración 56 Roles y vistas

❖ Crear Nueva Solicitud

create:

- Muestra el formulario para crear una nueva solicitud.
- Vista: vistas.DNC.create.

store:

- Valida los datos del formulario:
- Campos obligatorios: nombre, objetivo, instructor, contacto_instructor, participantes, prioridad, origen, requerimientos.
- Validación especial: participantes debe ser un número entero mayor o igual a 1.
- Crea una nueva solicitud (modelo dnc) con los datos proporcionados y el ID del usuario autenticado.
- Redirige a la lista de solicitudes del jefe de departamento.

```php
public function store(Request $request)
{
    $request->validate([
        'nombre' => 'required',
        'objetivo' => 'required',
        'instructor' => 'required',
        'contacto_instructor' => 'required',
        'participantes' => 'required|integer|min:1',
        'prioridad' => 'required',
        'origen' => 'required',
        'requerimientos' => 'required',
    ]);
    $solicitud = new dnc();
    $solicitud->user_id = auth()->user()->id;
    $solicitud->departamento_id = 1;
    $solicitud->nombre = $request->nombre;
    $solicitud->objetivo = $request->objetivo;
    $solicitud->instructor_propuesto = $request->instructor;
    $solicitud->contacto_propuesto = $request->contacto_instructor;
    $solicitud->num_participantes = $request->participantes;
    $solicitud->prioridad = $request->prioridad;
    $solicitud->origen = $request->origen;
    $solicitud->requerimientos = $request->requerimientos;


    $solicitud->save();
    return redirect(route('jefe_solicitarcursos.index'));
}
```

Ilustración 57 Store

❖ Mostrar Detalles de una Solicitud

show:

- Recupera y muestra los detalles de una solicitud específica.
- Vista: vistas.dnc.admin.show.

❖ Eliminar Solicitud

destroy:

- Busca y elimina una solicitud de curso por su ID.
- Redirige a la lista de solicitudes del jefe de departamento.

```php
public function destroy($id)
{

    $solicitarcurso = dnc::find($id);

    $solicitarcurso->delete();
    return redirect(route('jefe_solicitarcursos.index'));
}
```

Ilustración 58 Destroy

7.5.- Despliegue del proyecto

Para la entrega del proyecto se publicó el repositorio a través de la plataforma GitHub en la ruta:

https://github.com/20690446-ADRIAN-MARTINEZ-LARA/capacitacion_interno

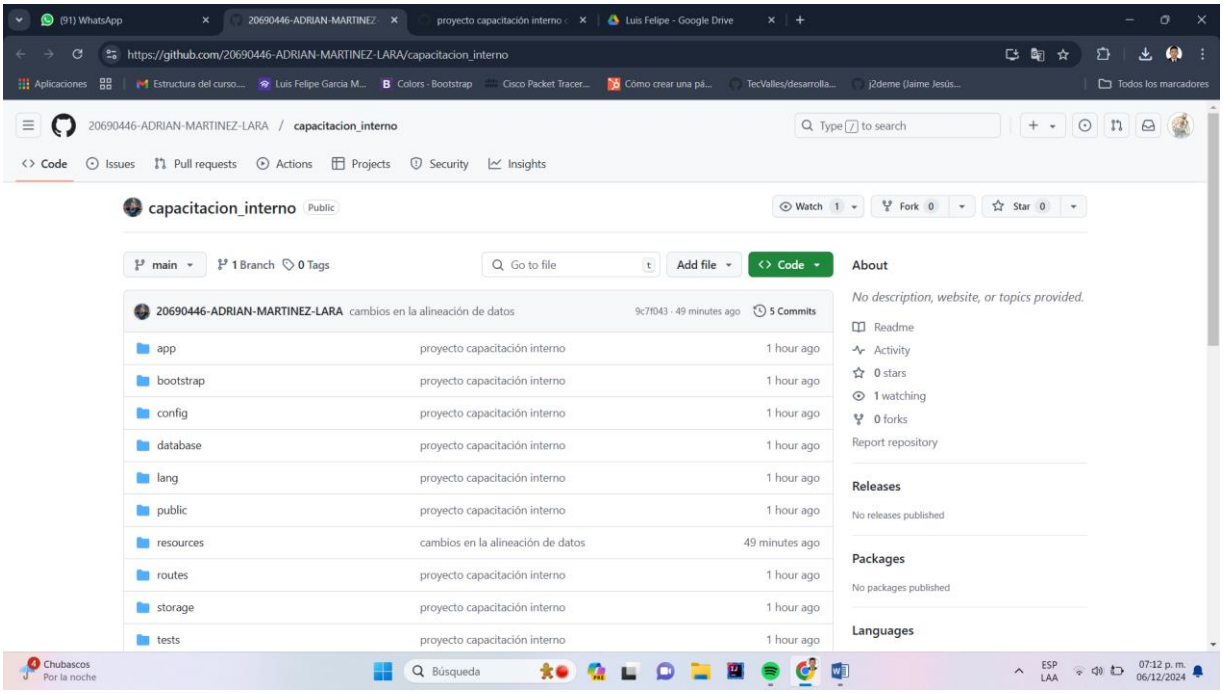

Ilustración 59 GitHub ruta del repositorio

8.- Resultados, planos, gráficas, prototipos, manuales, programas, análisis estadísticos, modelos matemáticos, simulaciones, normatividades, regulaciones y restricciones, entre otros

Como pantalla de inicio, representada en la ilustración 58, se muestra el usuario administrador, el cual tiene todos los permisos de la plataforma, a partir de ahí se irán creando los usuarios con el formulario de registro, cada formulario será rellenado con los datos del usuario, el cual puede llegar a ser docente, administrativo, jefe de departamento, subdirector académico, CAD (Coordinador de Actualización Docente) e instructor, cada uno con su respectiva interfaz y permisos para que puedan realizar sus registros y/o demás acciones.

Ilustración 60 Pantalla Inicio

Ilustración 61 Pantalla Administrador

El CAD tendrá acceso a las mismas funciones que el administrador, con esto se espera que lleven el control de la plataforma.

Ilustración 62 Pantalla CAD

El subdirector académico, el cual es un rol de consulta, tendrá acceso a todo en modo de vista, ya sea ver información de usuarios, cursos, departamentos, periodos, etc., pero no podrá registrar, editar ni eliminar nada de información.

Ilustración 63 Pantalla Subdirector

El de Jefe departamento tendrá como función solicitar cursos, los cuales pueden ser aceptados o denegados por el administrador y el CAD podrá ver dichas solicitudes con toda la información y si es necesario tendrá la opción de eliminarlas.

Ilustración 64 Pantalla Jefe de Departamento

El instructor tendrá como función calificar a los docentes en sus cursos, con esto se sabrá qué docentes han aprobado y reprobado, para posteriormente generar un concentrado con los docentes aprobados para solicitar sus constancias de capacitación.

Ilustración 65 Pantalla Instructor

El docente tendrá como función registrarse a los cursos disponibles, si por algún motivo tiene que salir del curso tendrá la opción, al finalizar un curso podrá contestar la encuesta de satisfacción, los datos recabados de dicha encuesta serán utilizados para la elaboración de gráficas, con esto se sabrá si el curso ha sido del agrado del docente.

Ilustración 66 Pantalla Docente

Cada usuario podrá tener uno o más roles, al seleccionar varios roles se le agregarán las funciones correspondientes para cada uno, por ejemplo, si un usuario cuenta con el rol de docente y jefe de departamento, podrá solicitar cursos, ver sus solicitudes, los cursos, etc., más aparte podrá unirse a los cursos que haya disponibles en ese momento, dependiendo los roles que tenga el usuario será lo que podrá hacer dentro de la plataforma.

Además, los roles de administrador y CAD podrán editar los datos, roles y estatus de los usuarios, por ejemplo, si existe un cambio de jefe de departamento se podrá eliminar el rol del actual jefe y agregarlo al nuevo, en el caso del estatus funcionará para cuando algún docente esté inactivo, con ese estatus sólo tendrá acceso a sus cursos terminados, no podrá inscribirse ni realizar otra acción hasta que este sea reactivado.

Cada usuario cuenta con su perfil, en dado caso de que él mismo quiera editar su información personal.

9.- Conclusiones de Proyecto, recomendaciones y experiencia personal profesional adquirida

9.1. - Conclusiones

El desarrollo del sistema se completó con éxito, diseñada para atender los requerimientos de docentes y administradores del Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles, la plataforma mejora significativamente la organización y disponibilidad de la información, fomentando procesos más eficientes de actualización profesional.

La conclusión de este proyecto representa un paso importante hacia la modernización de los procesos institucionales, facilitando la formación continua del personal docente y fortaleciendo la calidad educativa del Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles en línea con sus principios y metas estratégicas.

9.2. -Recomendaciones

La recomendación principal es que se continúe con el desarrollo del proyecto, con el fin de mejorar la plataforma, agregando nuevas funciones que faciliten el trabajo de los usuarios o con un diseño más intuitivo.

9.3. - Experiencia personal profesional adquirida

Durante el desarrollo de este proyecto, he adquirido una experiencia significativa en diferentes aspectos tanto profesionales como personales. Uno de los aprendizajes más importantes fue consolidar mis conocimientos técnicos en el desarrollo de sistemas y aplicaciones web. Aprendí a trabajar con Laravel, un framework que me permitió estructurar y gestionar roles, vistas y funcionalidades de manera eficiente. Además, profundicé en el uso de herramientas para gestionar bases de datos, aplicando buenas prácticas en el diseño de tablas con claves primarias y foráneas que garantizan la integridad de los datos.

La implementación de diferentes roles en el sistema, como el de administrador, subdirector académico, jefe académico, docente e instructor, me permitió comprender la importancia de analizar las necesidades específicas de cada usuario y adaptar las funcionalidades para brindarles una experiencia adecuada. Esto incluyó desde la creación de formularios de registro y encuestas hasta la generación de estadísticas y reportes. También desarrollé habilidades en la validación de datos y la prevención de errores, asegurando que las acciones en el sistema fueran seguras y funcionales.

En el ámbito profesional, este proyecto también me permitió trabajar en equipo de manera efectiva. Colaborar con otros integrantes del proyecto implicó una constante comunicación, intercambio de ideas y una división clara de responsabilidades. Aprendí a escuchar diferentes perspectivas, resolver conflictos y construir soluciones que beneficiaran al equipo y al proyecto en general.

Otro aspecto fundamental fue el aprendizaje de la gestión del tiempo. Cumplir con un horario establecido y atender a reuniones con asesores, compañeros, maestros, me ayudó a mejorar mi disciplina y organización personal. Esto incluyó planificar mis actividades, priorizar tareas críticas y ajustarme a los plazos establecidos para cumplir con los objetivos del proyecto.

Finalmente, puedo decir que esta experiencia no solo fortaleció mis habilidades técnicas, sino también mi capacidad para adaptarme a un entorno profesional real. Aprendí la importancia del compromiso, la responsabilidad y la proactividad, valores esenciales que aplicaré en mi futuro profesional.

10.- Competencias desarrolladas y/o aplicadas

Algunas de estas competencias son:

- Implementa aplicaciones computacionales para solucionar problemas de diversos contextos, integrando diferentes tecnologías, plataformas o dispositivos.
- Coordina y participa en equipos multidisciplinarios para la aplicación de soluciones innovadoras en diferentes contextos.
- Diseñar, desarrollar y administrar bases de datos conforme a requerimientos definidos, normas organizacionales de manejo y seguridad de la información, utilizando tecnologías emergentes.

11.- Fuentes de información

asana. (19 de febrero de 2024). *Las 12 metodologías más populares para la gestión de proyectos*. Obtenido de https://asana.com/es/resources/project-management-methodologies

AXARNET COMUNICACIONES S.L. (2023). *Laravel: Qué es y para qué sirve 【Características】*. Obtenido de https://axarnet.es/blog/que-es-laravel

Bloch, J. (2023). *Effective Java*. Obtenido de Addison-Wesley Professional.: https://www.pearson.com/

Boehm, B. W. (2022). *A spiral model of software development and enhancement.* Obtenido de ACM SIGSOFT Software Engineering Notes: https://dl.acm.org/doi/10.1145/12944.12948

Canelo, M. M. (2024). *Backend Developer: Qué es, funciones y cómo convertirse en uno/a*. Obtenido de Marketing & Communications Manager en Profile.: https://profile.es/blog/backend-developer/

Coppola, M. (21 de 1 de 2023). *HubSpot*. Obtenido de Qué es una plataforma digital, qué tipos existen y ejemplos: https://blog.hubspot.es/website/que-es-plataforma-digital

Crockford, D. (2022). *JavaScript: The Good Parts*. Obtenido de https://www.oreilly.com/

Huet, P. (21 de enero de 2022). *Qué es Tailwind CSS y por qué deberías usarlo*. Obtenido de https://openwebinars.net/blog/que-es-tailwind-css-y-por-que-deberias-usarlo/

Humble, J. &. (2010). *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation.* Obtenido de https://aws.amazon.com/es/what-is/api/#:~:text=Una%20API%20web%20o%20API,las%20API%20son%20servicios%20web.

Ken, A. (3 de 10 de 2023). *Backend: ¿Qué es y para qué sirve?* Obtenido de https://www.gluo.mx/blog/backend-que-es-y-para-que-sirve

Kunkel, C. (2023). *Modern PHP: New Features and Good Practices*. Obtenido de https://www.oreilly.com/

Maldeadora, N. (2019). *Qué es Frontend y Backend: características, diferencias y ejemplos*. Obtenido de https://aws.amazon.com/es/compare/the-difference-between-frontend-and-backend/#:~:text=El%20front%20end%20es%20aquello,permiten%20que%20la%20aplicaci%C3%B3n%20funcione.

Matthes, E. (2023). *Python Crash Course: A Hands-On*. Obtenido de Project-Based Introduction to Programming: https://www.penguinrandomhousegrupoeditorial.com/

Microsoft. (2022). *Visual Studio: IDE y Editor de código para desarrolladores de software*. Obtenido de https://visualstudio.microsoft.com/es/

Musib, S. (2022). *Spring Boot in Practice*. Obtenido de https://www.ibm.com/mx-es/topics/java-spring-boot

Oracle. (24 de noviembre de 2020). *¿Qué es una base de datos?* Obtenido de https://www.oracle.com/mx/database/what-is-database/

Peña, V. (2024). *¿Qué Es Xampp?* Obtenido de https://norvicsoftware.com/que-es-xampp/

Pressman, R. S. (2010). Ingeniería del software Un enfoque práctico. En R. S. Pressman, *Ingeniería del software Un enfoque práctico* (Vol. 7, pág. 810). McGraw-Hill.

Pressman, R. S. (2010). Ingenieria del software, un enfoque practico. En R. S. Pressman, *Ingenieria del software, un enfoque practico* (pág. 810). México: McGraw-Hil.

Rendón, Y. A. (28 de mayo de 2019). *Bases de datos relacionales vs. no relacionales*.

Robledano, A. (24 de septiembre de 2019). *Qué es MySQL: Características y ventajas*. Obtenido de https://openwebinars.net/blog/que-es-mysql/

SEOEstudios, P. (27 de enero de 2022). *Qué es backend y por qué es tan importante para tu web*. Obtenido de https://www.seoestudios.es/que-es-backend-web/

Stauffer, M. (2023). *Laravel: Up & Running*. Obtenido de https://axarnet.es/blog/que-es-laravel

Tecnológico Nacional de México / Instituto Tecnológico de Ciudad Valles. (2024). *Tecnológico Nacional de México Campus Ciudad Valles*. Obtenido de https://www.tecvalles.mx/wp/historia-del-tecnologico-nacional-de-mexico-campus-ciudad-valles/

Thomas, D. &. (2022). *Programming Ruby 3.2: The Pragmatic Programmers'*. Obtenido de https://pragprog.com/

Vincent, W. S. (2022). *Django for Professionals: Production websites with Python & Django*. Obtenido de https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Introduction

12.- Anexos

Anexo 1: Carta de Liberación por parte de la empresa

Educación
Secretaría de Educación Pública
TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Ciudad Valles
Departamento de Desarrollo Académico

Ciudad Valles, San Luis Potosí, 06/diciembre/2024
Oficio No. 069.15/173/2024
ASUNTO: Carta de Terminación de Residencia Profesional

MAP. HÉCTOR AGUILAR PONCE
DIRECTOR DEL INSTITUTO TECNOLÓGICO
DE CIUDAD VALLES
PRESENTE

ATN': BELEM MEZA ARTEAGA
JEFA DEL DEPARTAMENTO DE GESTIÓN
TECNOLÓGICA Y VINCULACIÓN

Por este medio me permito informarle que el (la) C. **LUIS FELIPE GARCÍA MARTÍNEZ** estudiante de la carrera **INGENIERÍA EN SISTEMAS COMPUTACIONALES** con número de control **20690041**, ha realizado satisfactoriamente su RESIDENCIA PROFESIONAL en las oficinas del departamento de DESARROLLO ACADÉMICO, en el cuál desarrolló el proyecto **PLATAFORMA INSTITUCIONAL DE LA FORMACIÓN DOCENTE Y PROFESIONAL: ENFOQUE EN EL BACKEND**, durante el período comprendido del 26 de agosto de 2024 al 06 de diciembre de 2024.

Se extiende la presente para los fines legales que al interesado (a) convengan, en Ciudad Valles, San Luis Potosí a los seis días del mes de diciembre de dos mil veinticuatro.

ATENTAMENTE
Excelencia en Educación Tecnológica®

DR. JAIME JESÚS DELGADO MERAZ
JEFE DEL DEPARTAMENTO DE DESARROLLO ACADÉMICO

EDUCACIÓN SECRETARÍA DE EDUCACIÓN PÚBLICA — TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE CIUDAD VALLES
DEPARTAMENTO DE DESARROLLO ACADÉMICO

ccp. Archivo

DR'JJDM

100% PLÁSTICO

Carr. Al Ingenio Plan de Ayala Km. 2 Col. Vista Hermosa. Ciudad Valles, S.L.P. C.P. 79010 Tel. 01 (481) 381 20 44 Ext. 132
e-mail: dda@tecvalles.mx https://www.tecvalles.mx/wp/

2024 Felipe Carrillo PUERTO

Ilustración 67 Carta de terminación

Anexo 2: Cronograma de actividades

INSTITUTO TECNOLÓGICO DE CIUDAD VALLES

Estudiante: Luis Felipe García Martínez. **No. De Control:** 20690041
Nombre del Proyecto: Plataforma institucional de la formación docente y actualización profesional: enfoque en el backend
Empresa: Instituto Tecnológico de Ciudad Valles.
Asesor Externo: DR. Jaime Jesús Delgado Meraz **Asesor Interno:** ME. Claudia Cruz Navarro.
Periodo de Realización: **Agosto-Diciembre 2024.**

ACTIVIDAD		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Comunicación con asesores	P															
	R															
Planeación del proyecto	P															
	R															
Modelado del proyecto	P															
	R															
Construcción del proyecto	P															
	R															
Despliegue del proyecto	P															
	R															

Anexo 3: