Texto

Descripción generada automáticamente

Universidad Politécnica Salesiana

Sede Guayaquil

CARRERA: COMPUTACIÓN

**ANTEPROYECTO**

Previo a la obtención del título de Ingeniería en Ciencias de la Computación

TITULO

“Implementación de un sistema web Odontológico para la administración de la Clínica Dental Odontomix”

AUTORES

Murillo Ramos Javier Enrique

González Parrales Carlos Jose

TUTOR  
Quiroz Martínez Miguel Ángel, Ing.

Guayaquil, septiembre de 2023

Índice

[1. Título 2](#_Toc146638014)

[2. Problema de Estudio (estado del arte) 2](#_Toc146638015)

[3. Justificación 2](#_Toc146638016)

[4. Objetivos 2](#_Toc146638017)

[4.1. Objetivo General 2](#_Toc146638018)

[4.2. Objetivos Específicos 2](#_Toc146638019)

[5. Marco Hipotético 3](#_Toc146638020)

[6. Marco Teórico Referencial 3](#_Toc146638021)

[6.1. HTML 3](#_Toc146638022)

[6.2. PHP 3](#_Toc146638023)

[6.3. Framework 3](#_Toc146638024)

[6.4. MySQL 4](#_Toc146638025)

[6.5. Tailwind Css 4](#_Toc146638026)

[6.6. Laravel 4](#_Toc146638027)

[6.7. React 5](#_Toc146638028)

[6.8. Railway 6](#_Toc146638029)

[6.9. API REST 6](#_Toc146638030)

[6.10. Arquitectura Cliente-Servidor 6](#_Toc146638031)

[6.11. Arquitectura de 3 capas 7](#_Toc146638032)

[6.12. MVC 8](#_Toc146638033)

[6.13. Metodología Cascada 8](#_Toc146638034)

[6.14. Historial Clínico 9](#_Toc146638035)

[6.15. Odontograma 9](#_Toc146638036)

[7. Metodología / Propuesta de desarrollo 9](#_Toc146638037)

[8. Cronograma y actividades a desarrollar 10](#_Toc146638038)

[9. Presupuesto 11](#_Toc146638039)

[10. Referencias bibliográficas 11](#_Toc146638040)

# Título

“Implementación de un sistema web Odontológico para la administración de la Clínica Dental Odontomix”

# Problema de Estudio (estado del arte)

Esta sección indica el problema de estudio que va a ser abordado mediante la opción de titulación escogido por el estudiante, además presenta si existen trabajos previos o cualquier otra información relevante para el trabajo a realizar.

Coloque mínimo tres párrafos en este apartado que sirva como el contexto de lo general al nivel específico. La portada. Incluye los datos personales del estudiante y los del anteproyecto. El título debe ser claro, conciso, viable, con enfoque acorde a la formación profesional del estudiante, que evidencie lo que se pretende hacer. Guardará estrecha relación con el objetivo general.

En este apartado se especificará de forma clara cómo nace la idea, si existen trabajos previos y cualquier otra información que permita conocer aspectos previos al trabajo propuesto. El artículo debe estar estructurado de acuerdo con el formato exigido por la revista científica en la que se proyecta publicarlo. En el caso de un artículo presentado como ponencia de un Congreso, se debe ajustar al formato exigido por el Congreso. En el caso de ser sometido a revisión interna por pares, podrá ser usado el formato de cualquiera de las revistas de la universidad.

El artículo académico expone el resultado total o parcial de un trabajo de investigación donde se define los niveles exploratorios, descriptivos y correlaciónales o explicativos dentro de las líneas de investigación establecidas por la Universidad.

# Justificación

El centro dental Odontomix carece de un software para la optimización de procesos y recolección de la información. El personal interno del centro dental Odontomix ocupan demasiado tiempo en buscar el historial clínico de un paciente, debido que usan documentos físicos para almacenar la información y ocupan mucho espacio físico.

El software será de ayuda para las gestiones y procesos tales como facturación, agendamiento de citas, historiales clínicos y los reportes que se realicen dentro del Centro Dental Odontomix.

# Objetivos

# Objetivo General

Desarrollar e implementar un aplicativo web que nos permita llevar el control del historial clínico de los pacientes, realizar agendamientos y permitir realizar cobros por medio de la facturación.

# Objetivos Específicos

* Controlar el registro del historial clínico
* Facilitar el agendamiento
* Implementar el módulo de facturación

# Marco Hipotético

Presenta las hipótesis o preguntas de investigación referente al tema propuesto, mínimo 3 preguntas de investigación.

Las preguntas que se plantean para el este estudio son:

* ¿Cómo se realizará el control de registro de los datos de los pacientes?
* ¿Cómo se manejará las citas de los pacientes?
* ¿De qué forma se va a facturar?

# Marco Teórico Referencial

# HTML

La versión actual de HTML es la 5 donde brinda una independencia del dispositivo, lo que significa que los sitios web se puede desarrollar en cualquier tipo de plataforma, no se necesita de instalar complementos adicionales para poder ejecutar un sitio web (Wilson, 2023)

En HTML tenemos varias etiquetas con la que se define la estructura como son <html>, <head> y <body>. No son las únicas, pero son las principales al generar un sitio web

# PHP

PHP es un lenguaje de programación robusto con todo los necesario para desarrollar sistemas web, en la actualidad se usa comúnmente para el desarrollo backend (de lado del servidor).

La versión 8 de PHP que es la actual se lanzó el 26 de noviembre de 2020 trayendo características importantes como es el compilador JIT, Tipos de unión, Operadores NullSafe, siendo el compilador JIT la característica más interesante en esta versión (Engebreth, 2021)

# Framework

Un framework es un conjunto premeditado de conceptos, módulos y criterios estandarizados que facilitan la tarea de desarrollar sitios y aplicaciones web. más fácil. Proporciona funcionalidad genérica con módulos ya escritos y componentes a medida creados de manera estándar. En resumen, es un entorno reutilizable que permite a los diseñadores y desarrolladores web construir sus proyectos y soluciones con un mínimo de codificación y sin preocuparse por los detalles de bajo nivel. Esto reduce el tiempo de desarrollo y de desarrollo y facilita el mantenimiento y las modificaciones siempre que sea necesario.

Normalmente, hay dos tipos de frameworks:

* Frameworks de front-end (frameworks CSS y JavaScript)
* Frameworks de programación back-end o del lado del servidor.

Los desarrolladores y programadores web utilizan los marcos de trabajo de back-end para crear aplicaciones en el servidor.

programadores web para crear aplicaciones en el lado del servidor, los son utilizados por diseñadores y desarrolladores web para implementar el lenguaje de hojas de estilo en cascada (Shenoy & Prabhu, 2018)

# MySQL

MySQL es un popular sistema de gestión de bases de datos SQL de código abierto desarrollado, distribuido y soportado por Oracle Corporation. MySQL gestiona una colección estructurada de datos. Una base de datos MySQL le ayuda a añadir, acceder y procesar los datos almacenados en la base de datos. MySQL almacena los datos en tablas separadas. Las estructuras de la base de datos están organizadas en archivos físicos optimizados para la velocidad. El modelo lógico, con objetos como bases de datos, tablas, vistas, filas y columnas, ofrece un entorno de programación flexible. La parte SQL de "MySQL" significa "Structured Query Language" (lenguaje de consulta estructurado), que es el lenguaje estandarizado más utilizado para acceder a las bases de datos. lenguaje estandarizado más utilizado para acceder a bases de datos. El software MySQL utiliza la licencia GPL (GNU General Public) y es software de código abierto (Christudas, 2019)

# Tailwind Css

Tailwind CSS es un Framework CSS que prioriza la capacidad de crear rápidamente experiencias de usuario únicas. un Framework CSS de bajo nivel que es extremadamente personalizable y proporciona todos los componentes básicos necesarios para crear diseños personalizados sin tener que esforzarse para superar estilos objetables. Lo mejor de Tailwind es que no impone pautas de diseño ni dicta la apariencia de su sitio web; en cambio, sólo necesitas combinar pequeñas piezas para crear una interfaz de usuario única. En pocas palabras, Tailwind toma un archivo CSS "sin procesar", lo procesa a través de un archivo de configuración y genera los resultados (Jeena et al., n.d.)

# Laravel

Laravel ofrece herramientas poderosas para crear aplicaciones grandes y confiables; Sin embargo, esto requiere comprender algunos conceptos clave, como el uso de un contenedor IoC o un contenedor de servicios para gestionar las dependencias de clases.

Laravel te ayuda a escribir código en un entorno poco acoplado. La plataforma se centra en desarrollar clases con una única responsabilidad, evitando la codificación en módulos de nivel superior y permitiendo que los contenedores IoC tengan abstracciones específicas. Los módulos de nivel superior son independientes de los módulos de nivel inferior, lo que hace que la programación sea divertida.

el patrón lógico MVC. contiene muchas otras características que han mejorado las características del patrón, elevándolos a un nuevo nivel.

Mecanismo de inyección de dependencia (DI) de forma predeterminada (Sinha, 2019)

Algunos de los aspectos relevantes de Laravel son:

* Migración:
* La migración es como el control de versiones de las bases de datos. Las migraciones permiten modificar la estructura de la base de datos de forma organizada y coherente.
* Puedes crear una nueva migración para crear o modificar tablas, y luego ejecutar la migración para aplicar los cambios.
* Si cometes un error, Laravel te permite "deshacer" la última migración para que sea fácil corregir el error.
* Middleware
* El middleware proporciona un mecanismo conveniente para filtrar las peticiones HTTP que llegan a tu aplicación.
* Por ejemplo, el middleware en Laravel puede comprobar si el usuario de la aplicación está autenticado. Si no es así, el middleware redirige al usuario a la página de inicio de sesión.
* Puedes crear middleware personalizado que se adapte a tus necesidades, como comprobar roles o permisos.
* ORM Eloquent:
* Eloquent es ORM (Object Relational Mapping) en Laravel. Proporciona una forma sencilla y positiva de interactuar con una base de datos utilizando objetos en lugar de escribir SQL directamente.
* Cada modelo Eloquent corresponde a una tabla en la base de datos y representa un tipo de entidad (por ejemplo, usuario).
* Eloquent proporciona una variedad de operaciones CRUD y relaciones (por ejemplo, uno-a-muchos, muchos-a-muchos, etc.) para facilitar la interacción con los datos.
* Rutas y Controladores
* Laravel proporciona una forma sencilla de definir rutas y asociarlas con controladores.
* Los controladores agrupan la lógica asociada con el manejo de peticiones HTTP y normalmente interactúan con modelos y vistas para responder a esas peticiones.
* Artisan:
* Artisan es la interfaz de línea de comandos que viene con Laravel. Proporciona una serie de comandos útiles para tareas comunes como la creación de controladores, modelos, migraciones, y más.

# React

React es una biblioteca de vistas de JavaScript muy popular que se usa frecuentemente junto con otras tecnologías para crear aplicaciones de JavaScript en una sola página. React, fue lanzado por Facebook en 2013, ha superado rápidamente a otros proyectos similares en popularidad debidoa su énfasis en la afirmación del estado, la ubicación de elementos específicos de puntos de vista y el peso relativo de las opiniones de los usuarios. Esto lo hace muy diferente de Angular y Ember. React se describe a sí mismo como una" biblioteca de JavaScript para la construcción de interfaces de usuario " y establece el siguiente en su página de inicio:

React facilita la creación de interfaces de usuario interactivas. Cree vistas simples para cada estado de la aplicación y React actualizará y mostrará los componentes apropiados cuando esos estados cambien (So, 2018)

# Railway

La definición actual de computación en nube es "un modelo que permite un acceso cómodo y a la carta a través de la red a un conjunto compartido de recursos informáticos configurables (por ejemplo, redes, servidores, almacenamiento, aplicaciones y servicios) que pueden aprovisionarse y liberarse rápidamente con un mínimo esfuerzo de gestión o interacción del proveedor de servicios" (Bhatt & Acken, n.d.)Railway.app puede definirse como: "Una plataforma que facilita el desarrollo, despliegue y gestión de aplicaciones, entregando infraestructura como servicio a través de Internet, y eliminando la necesidad de gestionar el hardware y el sistema subyacente. En este contexto, los servicios ofrecidos por Railway.app pueden considerarse como Infraestructura como Servicio (IaaS) y Plataforma como Servicio (PaaS), permitiendo a los desarrolladores centrarse en el código y las funcionalidades de sus aplicaciones, mientras que la infraestructura y el backend son gestionados automáticamente por la plataforma."

# API REST

API REST en la actualidad se ha convertido en una arquitectura usada por mucha gente en las TI, esto gracias a la facilidad de interactuar con un sistema por HTTP y enviar la información con los datos por medio de un JSON.

El comienzo de REST fue por su autor Roy Fielding, en el año 2000 presentó su tesis doctoral “Diseño de la arquitectura de Software basada en Redes” para sistemas hipermedia distribuidos. La abreviatura de REST es “Transferencia de Estado Representacional” siendo una arquitectura que ayuda a crear y organizar sistema distribuidos (Doglio, 2018)

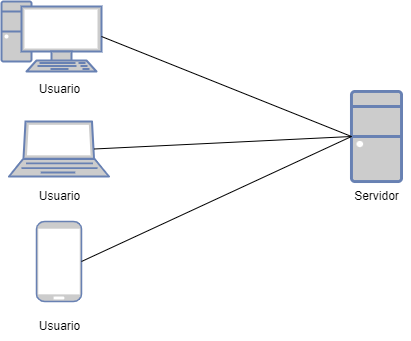
El rendimiento de REST es de forma eficiente y simple, gracias a su simplicidad tenemos una interfaz que permite interactuar entre sistemas. Se puede implementar y consumido por cualquier tipo de tecnología debido a que es una tecnología y lenguaje son independientes.

El sistema web se desarrollará y consumirá una API REST en Laravel mediante una capa de transformación que se encuentra entra en los modelos Eloquent y las respuestas JSON que devuelve el sistema.

# Arquitectura Cliente-Servidor

La arquitectura cliente-servidor es utilizado para describir un modelo informático para desarrollar los sistemas, donde se usa más comúnmente es en los sistemas web. La arquitectura se enfoca en la distribución en funciones de dos tipos: Cliente y servidor, funciona de tal manera que el cliente realiza cualquier petición al cliente y el servidor devuelve una respuesta para el cliente.

La arquitectura cliente-servidor sigue un estándar de configuración de maestro-esclavo, el cual el servidor es el maestro y los clientes son los esclavos. Además, este modelo lo adoptan mucho para los sistemas empresariales debido a su escalabilidad, disponibilidad, rendimiento, etc (Surianarayanan & Chelliah, 2023).

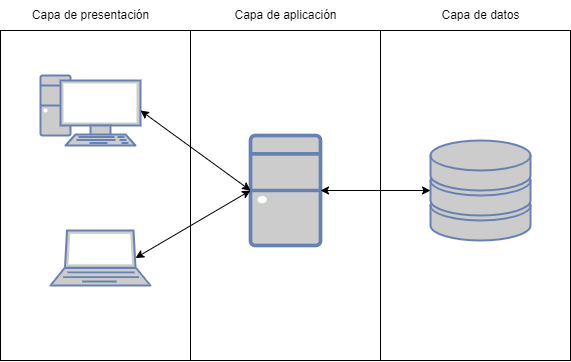


Ilustración

# Arquitectura de 3 capas

Es una arquitectura muy bien establecida para organizar las aplicaciones en 3 niveles informáticos lógicos y físicos como son: el nivel de presentación o también llamado interfaz de usuario, el nivel de aplicación donde se gestiona los datos y el nivel de datos, en el cual se almacenan y se administra los datos que se manejará el sistema.

La capa de presentación muestra al usuario una entrada amigable para comunicarse con el sistema por otro lado la capa de aplicación realiza las funciones de control y manipulación de la conexión lógica de los flujos de información y por último la capa de datos donde está indexada la base de datos para modelar la información necesaria para el sistema (Chen’ et al., n.d.).



Ilustración

# MVC

MVC (Modelo-Vista-Controlador) es un patrón de diseño de software basado en la combinación de tres tipos de componentes principales en lenguajes de programación como PHP, a menudo con un fuerte enfoque orientado a objetos. principios en lenguajes de programación como PHP, a menudo enfatizando el paradigma orientado a objetos (OOP). Estos tres tipos de componentes se denominan modelos, vistas y controladores.

* El modelo en un patrón de diseño MVC alberga la lógica de negocio de una aplicación. Esta lógica puede abarcar desde cómo se almacenan los datos hasta la interacción con servicios externos. Si una aplicación requiere acceder a una base de datos o interactuar con servicios como consultar existencias o publicar en redes sociales, ese código se sitúa en el modelo.
* La vista en el patrón MVC contiene los elementos de la interfaz de usuario, incluyendo HTML, CSS y JavaScript. Es lo que el usuario ve e interactúa. A veces, lo que se muestra es una combinación de varias vistas en una sola solicitud.
* El controlador en el patrón MVC sirve como puente entre modelos y vistas. Aísla la lógica de negocio del modelo de la interfaz de usuario en la vista. Gestiona las respuestas de la aplicación ante interacciones del usuario. Es el primer componente en recibir una solicitud, y se encarga de activar los modelos y vistas adecuados para atender dicha petición (Pitt, 2012)

# Metodología Cascada

El modelo en cascada trata de que el proyecto se planifica y se ejecuta dentro de un periodo de tiempo para lograr los objetivos finales (Bibik, 2018).

Es una metodología de desarrollo de software tradicional que va por fases, se debe de analizar los requerimientos, dibujar la arquitectura que usara el sistema, desarrollar el sistema para luego finalmente realizar las pruebas de usuario e implementarlo donde el usuario.

# Historial Clínico

El historial clínico es una estructura que se realiza para identificar la salud y los problemas de salud de un paciente, este incluye una evaluación de problemas de salud actuales y anteriores del paciente, tratamientos médicos actuales y previos, la salud del paciente en general (Bilaney et al., 2015).

En el historial clínico se incluye los datos personales del paciente, información de exámenes físicos como: altura y peso, y si el paciente sufre de alguna alergia.

# Odontograma

La odontograma es algo esencial en la odontología ayuda a identificar cuales tratamientos se debe aplicar en cada diente y proporcionar una base solidad de cómo se llevará el caso (Daza-Cardona et al., 2021).Con la odontograma puede identificar cual es el estado de los dientes del paciente sin tener que realizar una revisión.

La forma en que se realiza por lo general es a mano con lápiz y pluma, pero con los avances de tecnologías se han adoptado mejores maneras de realizar mediante la computadora. En la actualidad todo sistema odontológico debe de incluir de manera simple y eficiente realizar la odontograma por parte del doctor.

# Metodología / Propuesta de desarrollo

La metodología usada para el desarrollo del sistema web es la metodología en cascada.

La metodología de desarrolla en 5 fases las cuales son: análisis, diseño, implementación, verificación y mantenimiento, cada fase se ejecutó secuencialmente como indica la metodología. Usando esta metodología realizaremos varias iteraciones hasta la implementación realizando pruebas y verificando con el usuario para que todo esté correcto.

Fase análisis

En esta etapa se realiza el levantamiento de información con los actores, definiendo cual es el alcance del sistema. Determinando los objetivos que cumplirá dentro del sistema.

Fase diseño

En la etapa de diseño definimos el alcance del sistema web que se desarrollara, los módulos que se crearan e interactuaran los usuarios son:

* Agendamiento de citas
* Registro de usuario
* Registro de pacientes
* Registro de clientes
* Registro de historial clínico
* Creación de odontograma
* Módulo de facturación

La arquitectura en que se basa el sistema web es de cliente servidor, con el modelo de 3 capas, este modelo nos da una flexibilidad al momento de querer modificar algún dato sin tener que comprometer la vista del usuario. Y utilizamos un patrón de diseño como es el Modelo Vista Controlador

Las tecnologías que se definieron para el desarrollo del sistema web fueron:

* Servidor: Railway
* Arquitectura: Cliente-Servidor
* Lenguaje: PHP
* Framework: Laravel
* Base de datos: MySQL

Fase implementación

En esta fase aplicamos todos los conocimientos adquiridos durante la carrera de computación para desarrollar el sistema, aplicando las tecnologías antes mencionadas en la fase diseño y de acuerdo al análisis que se había definido con el levantamiento de información.

Cada módulo que se termine realizaremos pruebas las pruebas unitarias y revisaremos las fases anteriores para ver si está todo correcto.

Fase verificación

Una vez el sistema web ha sido desarrollado en casi su totalidad lanzaremos pruebas para que el usuario interactúe con el sistema y revisar si todos los requerimientos y objetivos se hayan cumplido.

Fase mantenimiento

Cuando haya finalizado el sistema web y todos los objetivos y requerimientos han sido cumplidos se lanza a producción y el sistema tendrá constantes mejoras de software y mantenimiento a cualquier error que suceda durante el uso del sistema.

# Cronograma y actividades a desarrollar

Detalla las actividades generales que se realizarán, de acuerdo con la Unidad de Titulación del proyecto respectivo. Utilice el siguiente esquema:

Tabla . Ejemplo de distribución del tiempo, se puede utilizar un diagrama de Gantt con el software de project

|  |  |  |  |
| --- | --- | --- | --- |
| Nombre de tarea | Horas | Comienzo | Fin |
| Artículo Académico |  | lun 01/10/18 | vie 01/02/19 |
| FASE 1 |  | lun 01/10/18 | mar 16/10/18 |
| Realizar un plan inicial de investigación. | 34 | lun 01/10/18 | lun 08/10/18 |
| Plantear tema y método. | 34 | mar 09/10/18 | mar 16/10/18 |
| FASE 2 |  | mié 17/10/18 | mar 06/11/18 |
| Investigar sobre estudios vinculados con el tema | 70 | mié 17/10/18 | mar 06/11/18 |
| FASE 3 |  | mié 07/11/18 | mar 11/12/18 |
| Redactar introducción, usando las fuentes relacionadas. | 28 | mié 07/11/18 | jue 15/11/18 |
| Establecer antecedentes del problema. | 32 | vie 16/11/18 | mar 27/11/18 |
| Elaborar el artículo de acuerdo con el formato de la Unidad de Titulación | 40 | mié 28/11/18 | mar 11/12/18 |
| FASE 4 |  | jue 13/12/18 | lun 07/01/19 |
| Construir la sección de materiales y métodos. Recolectar datos. | 16 | jue 13/12/18 | mar 18/12/18 |
| Organizar el orden de imágenes y tablas. | 18 | mié 19/12/18 | jue 20/12/18 |
| Redactar los resultados y representarlos en gráficos. | 20 | vie 21/12/18 | jue 27/12/18 |
| Desarrollar las conclusiones. | 20 | vie 28/12/18 | jue 03/01/19 |
| Enlistar las referencias de los estudios referentes al tema. | 18 | vie 04/01/19 | lun 07/01/19 |
| FASE 5 |  | mar 08/01/19 | lun 04/02/19 |
| Realizar una revisión del artículo. | 18 | mar 08/01/19 | vie 11/01/19 |
| Reformular el título, de ser necesario. | 19 | lun 14/01/19 | mar 15/01/19 |
| Elaborar versión final tomando en cuenta las observaciones del revisor. | 27 | mié 16/01/19 | mié 23/01/19 |
| Revisar rigurosamente el documento según el formato. | 19 | jue 24/01/19 | vie 25/01/19 |
| Correcciones del artículo, por observaciones realizadas por parte de los revisores. | 18 | lun 28/01/19 | jue 31/01/19 |
| Presentación del artículo o exposición frente al tribunal de grado. | 19 | vie 01/02/19 | lun 04/02/19 |
|  | 400 |  |  |

\* Las horas del anteproyecto inician cuando se lo aprueba, bajo una resolución, no debe colocar fechas anteriores a la fecha actual, por eso son estimadas a partir de la convocatoria (después). Un sistema informático lo registrará.

# Presupuesto

|  |  |
| --- | --- |
| **Gastos del trabajo de investigación** | **Valores ($)** |
| *Plan de Railway* | $ 10 |
| TOTAL | **$ 900** |

# Referencias bibliográficas

Bhatt, N. P., & Acken, J. (n.d.). *Cloud Computing withhSecurity and Scalability Concepts and Practices ThirddEdition*.

Bibik, I. (2018). How to Kill the Scrum Monster: Quick Start to Agile Scrum Methodology and the Scrum Master Role. In *How to Kill the Scrum Monster: Quick Start to Agile Scrum Methodology and the Scrum Master Role*. Apress Media LLC. https://doi.org/10.1007/978-1-4842-3691-8

Bilaney, J., Walker, B., Smith, L.-J., & Gamble Helen, D. (2015). *MBBS Year 4 GUIDE TO HISTORY TAKING AND EXAMINATION*.

Chen’, S.-C., Gulatit, S., Hamidt, S., Huang’, X., Luo, L., Morisseau-Leroy, N., Powellq, M. D., Zhan’, C., & Zhang’, C. (n.d.). *A Three-Tier System Architecture Design and Development for Hurricane 0 c currenc e Simulation*.

Christudas, B. (2019). Practical Microservices Architectural Patterns. In *Practical Microservices Architectural Patterns*. Apress. https://doi.org/10.1007/978-1-4842-4501-9

Daza-Cardona, J. A., Vargas-Ramírez, J., & Guapacha-Sánchez, M. A. (2021). Doing odontograms and dentists in the classroom. Materiality and affect in dental education. *Tapuya: Latin American Science, Technology and Society*, *4*(1). https://doi.org/10.1080/25729861.2021.1968635

Doglio, F. (2018). REST API Development with Node.js. In *REST API Development with Node.js*. Apress. https://doi.org/10.1007/978-1-4842-3715-1

Engebreth, G. (2021). PHP 8 Revealed: Use Attributes, the JIT Compiler, Union Types, and More for Web Development? In *PHP 8 Revealed: Use Attributes, the JIT Compiler, Union Types, and More for Web Development?* Apress Media LLC. https://doi.org/10.1007/978-1-4842-6818-6

Jeena, I., Selvanayaki, J., & Shanmugam, K. (n.d.). *Lecture Notes in Networks and Systems 673 Expert Clouds and Applications Proceedings of ICOECA 2023*.

Pitt, Chris. (2012). *Pro PHP MVC*. Apress.

Shenoy, A., & Prabhu, A. (2018). CSS Framework Alternatives. In *CSS Framework Alternatives*. Apress. https://doi.org/10.1007/978-1-4842-3399-3

Sinha, S. (2019). Beginning Laravel. In *Beginning Laravel*. Apress. https://doi.org/10.1007/978-1-4842-4991-8

So, P. (2018). Decoupled Drupal in Practice. In *Decoupled Drupal in Practice*. Apress. https://doi.org/10.1007/978-1-4842-4072-4

Surianarayanan, C., & Chelliah, P. R. (2023). *Essentials of Cloud Computing*. Springer International Publishing. https://doi.org/10.1007/978-3-031-32044-6

Wilson, K. (2023). The Absolute Beginner’s Guide to HTML and CSS. In *The Absolute Beginner’s Guide to HTML and CSS*. Apress. https://doi.org/10.1007/978-1-4842-9250-1