#ashtag
REVISTA ESPECIALIZADA EN INGENIERÍA
ISSN electrónico: 2346 - 139X

Aplicación para control del gasto energético de los consultorios médicos del policlínico Orestes Falls Oñate

Application for the control of the energy expenditure of the medical offices of the polyclinic Orestes Falls Oñate

William Jiménez Puentes

Ingeniero en Informática, Cuba.
Universidad Isla de la Juventud: Jesús Montané Oropesa.
wjimenez@uij.edu.cu;
https://orcid.org/0009-0002-3150-3308

Jorge Unger Rodríguez

Ingeniero en Informática, Cuba.
Universidad Isla de la Juventud: Jesús Montané Oropesa.
junger@uij.edu.cu;
https://orcid.org/0009-0005-5282-7784

Cómo citar: Jiménez Puentes, W., y Unger Rodríguez, J. (2023). Aplicación para control del gasto energético de los consultorios médicos del policlínico Orestes Falls Oñate. *#ashtag*, 1(22), 21-42. https://doi.org/10.52143/2346139X.1069

Recibido: 18-03-2023
Aceptado: 01-06-2023
Publicado: 15-06-2023

10.52143/2346139X.1070

Resumen

En la presente investigación se desarrolló una aplicación web para controlar el gasto energético de los consultorios médicos del Policlínico Orestes Falls Oñate, mediante el uso de herramientas de *software* libre, basada en tecnología Python y JavaScript y el sistema de gestión de bases de datos relacional PostgreSQL. En el proceso de desarrollo del sistema utilizó la metodología programación extrema (XP), junto con el lenguaje unificado de modelado (UML). Se implementó una arquitectura cliente-servidor, en esta arquitectura, la comunicación entre el *frontend* y el backend se realiza a través de APIs REST. El sistema propuesto contribuye a que toda la información sea accesible y que se logre una reducción del tiempo en la búsqueda de posibles errores en el proceso, economice el tiempo de trabajo y el costo y perfeccione el sistema de trabajo permitiendo mejorar las condiciones del personal.

Palabras Clave: aplicación, control, programación, rest, sistema.

Abstract

In this research, a web application was developed to control the energy expenditure of the medical offices of the Orestes Falls Oñate Polyclinic, using free software tools, based on Python and JavaScript technology and the database management system. PostgreSQL relational. In the system development process, the Extreme Programming (XP) methodology was used, along with the Unified Modeling Language (UML). A client-server architecture was implemented, in this architecture, communication between the frontend and the backend is done through REST APIs. The proposed system contributes to making all information accessible and achieving a reduction in time searching for possible errors in the process, saving work time and cost, and perfecting the work system, allowing for improved staff conditions.

Keywords: *Application, Control, Programming, Rest, System.*

Introducción

El uso eficiente de la energía, a veces simplemente llamado, eficiencia energética o ahorro energético, tiene como objetivo reducir la cantidad de energía requerida para proporcionar productos y servicios (Canaviri, 2019; Barrera, 2021). Las mejoras en la eficiencia energética se logran generalmente mediante la adopción de una tecnología o un proceso de producción más eficientes o mediante la aplicación de métodos comúnmente aceptados para reducir las pérdidas de energía (Espinosa, 2023).

Diferentes autores a nivel internacional enuncian varias motivaciones para mejorar la eficiencia energética entre los que destacan Martino (2023); Stein *at al.* (2023); Zambrano y Real (2021). Varios de ellos coinciden en que la reducción del uso de energía reduce los costos de electricidad y puede generar un ahorro financiero para los consumidores si el ahorro de energía compensa cualquier costo adicional de implementar una tecnología de eficiencia energética.

Se coincide con Espinosa *at al.* (2020) en que, en el caso de Cuba, el Programa de Ahorro de Electricidad fue la primera acción de carácter integral que se llevó a cabo en el país para promocionar el ahorro de electricidad y la cultura energética en Cuba. Incluye diversas medidas que conducen al ahorro de energía eléctrica y es conocido por la sigla PAEC. Durante su aplicación se han obtenido significativos resultados.

Datos aportados por Hidalgo (2014) muestran cómo entre los años 1990 y 1993, como consecuencia del derrumbe del campo socialista y la crisis económica que comenzó a sufrir el país, la disponibilidad de generación decreció de un 78 % a valores inferiores al 50 %, lo cual trajo como consecuencia que se produjeran prolongados apagones y que el consumo de energía eléctrica decreciera a más del 6 % como promedio anual.

El propio autor resalta que a partir de 1993 se produce una recuperación en la generación de electricidad, con una tasa promedio anual del 6,6 %, por el incremento de la disponibilidad de generación hasta niveles muy cercanos al 65 % y la puesta en marcha de la unidad nro.1 de la termoeléctrica de Felton. Esta mejoría registrada provocó que el consumo de combustible, para generar electricidad, creciera a razón del 6,2 % como promedio anual.

Se parte entonces de reconocer que la energía eléctrica no se almacena, por lo que la capacidad de generación del sistema tiene que estar diseñada para resolver la demanda máxima. De ahí que ante el incremento del consumo de energía de los consumidores, tanto del sector residencial como del resto, se decide elaborar un programa para el ahorro de electricidad. Es así como en noviembre de 1997 comenzó su trabajo el PAEC, en momentos en que la economía cubana se encontraba en franco proceso de recuperación y, en consecuencia, el crecimiento en la demanda y el consumo de electricidad de este propio año cerraba con tasas de 4,9 % en la máxima demanda promedio y de un 7,8 % en la generación de electricidad con relación al año anterior.

Aunque el sector eléctrico cubano venía trabajando en el uso racional y el manejo de la demanda de electricidad desde principios de la década del 70, el PAEC marca el inicio de un tratamiento del tema con total integralidad, incorporando los resultados esperados, de las acciones del programa, en el estudio de Desarrollo Perspectivo del Sector (Planeamiento Directivo) como parte de las decisiones por garantizar para que la calidad del servicio alcance los niveles requeridos.

Reyes y Rodríguez (2018) enunciaban que el municipio especial Isla de la Juventud solo generaba con combustible fósil, pero a partir de la Revolución Energética, importante programa ideado por Fidel Castro, se convirtió desde hace más de diez años en polígono de prueba para el empleo de estas energías, que para su aumento ya se concretan acciones en pos de una superior eficiencia y calidad del servicio. En el policlínico Orestes Falls Oñate se está realizando un trabajo para contribuir al ahorro energético y para lograr mejorar el proceso de control del consumo energético de cada uno de los consultorios y así contribuir al país.

Dentro del departamento de administración del Policlínico Orestes Falls Oñate, encargado de controlar a través de partes, las lecturas diarias del consumo eléctrico de los consultorios de médicos de familia (CMF), se obtuvo una visión individual de cada uno de los procesos y una visión general de la integración de todos ellos. Además, se evidenciaron dificultades que hacen poco eficiente dichos procesos, que se enumeran a continuación:

- El proceso se lleva a cabo de manera manual a través de un libro de Microsoft Excel, lo que limita la precisión y eficiencia en la recopilación y análisis de los datos.
- La información que se emite llega a ser insuficiente a la hora de sustentar el comportamiento del consumo eléctrico.
- El proceso de creación de un nuevo día de consumo es manual creando cada día una nueva hoja de cálculo y copiando una plantilla con las fórmulas que utiliza adecuándolas para el día en curso.
- Demoras en la recopilación de la información a causas de teléfono.
- Perdida de información, por errores humanos en el Excel,
- Falta de un sistema de copias de seguridad, lo que aumenta el riesgo de pérdida de información.
- La hoja de cálculo no permite que múltiples usuarios trabajen y analicen los datos.

Actualmente la práctica ha demostrado que la aplicación de las tecnologías de la información y las comunicaciones (TIC) en los distintos campos y esferas de la vida, mejora considerablemente su desempeño y desarrollo, así como la optimización de los procesos presentes en ellos. Por tanto, no es solo una necesidad, prácticamente es una obligación. Las ventajas que presenta la informatización ante el modo tradicional son infinitas; rapidez, mayor capacidad de almacenamiento en menor espacio, información versátil, mayor accesibilidad, facilita en gran medida el trabajo al personal responsable.

La presente investigación pretende realizar la implementación de una aplicación web con bondades y características que mejoren el control energético y facilite el trabajo de los trabajadores del departamento de administración y demás usuarios del sistema.

Materiales y métodos

Para el desarrollo de la investigación se declaran los siguientes métodos de investigación:

Métodos teóricos

- **Analítico-sintético:** Este método permite sintetizar la información necesaria para el desarrollo de la aplicación web de gestión de la información del consumo energético. Se utiliza al estudiar la documentación necesaria referente a la medición y consumo de energía, permitiendo la extracción de los elementos más importantes que se relacionan con el objeto de estudio.
- **Histórico lógico:** Este método es utilizado en la consulta de bibliografía referente al consumo energético. Permitió analizar los antecedentes y tendencias del control del consumo energético de los CMF del policlínico Orestes Falls Oñate.
- **Modelación:** Permitirá el diseño y posterior confección del *software* para la informatización de la solución propuesta a través de los diferentes artefactos que plantea la metodología seleccionada y su representación facilitando el estudio del objetivo de la investigación.

Métodos empíricos

- **Observación científica:** Mediante la observación se comprobó la necesidad de la investigación a partir de la información obtenida referente control y consumo energético de los CMF del policlínico Orestes Falls Oñate

La solución informática desarrollada contribuye significativamente al mejoramiento del sistema de control del consumo energético de los consultorios médicos. Al implementar una arquitectura cliente-servidor con comunicación a través de API REST entre el *frontend* y el backend, se logra una mayor accesibilidad a la información. Esto no solo facilita el trabajo del personal responsable, sino que también permite una reducción del tiempo en la búsqueda de posibles errores, economizando tiempo y costos.

Otro indicador importante es la independencia y optimización alcanzada en el trabajo del personal. La aplicación web está diseñada para ofrecer una experiencia fluida, permitiendo a los usuarios realizar sus tareas de manera eficiente y sin contratiempos. La optimización del proceso de control energético debería traducirse en una mayor independencia de los usuarios, liberándolos de tareas redundantes y permitiéndoles centrarse en actividades más críticas.

Se espera que la nueva forma de realizar el control energético genere un alto nivel de satisfacción entre el personal. La aplicación web no solo simplifica el proceso, sino que también introduce eficiencias que contribuyen directamente a la reducción del consumo energético y ahorro de costos. La retroalimentación positiva del personal en relación con esta mejora en el proceso debería reflejarse en una mayor satisfacción general.

- **Entrevista:** Esta técnica es utilizada para recopilar información referente a la energía eléctrica y al consumo energético. Esta entrevista fue dirigida a la planificadora del departamento de administración con el objetivo de obtener información sobre su flujo, definir la situación actual y deducir los requisitos e indicadores necesarios para cumplir con los objetivos de investigación. La entrevista proporcionó información importante sobre el registro y control del gasto energético en los consultorios médicos, así como las dificultades relacionadas con la recepción de información y el uso de documentos Excel. Además, se obtuvieron ideas y sugerencias sobre la necesidad de digitalizar los procesos. En general, la entrevista fue exitosa en términos de obtener datos relevantes para la investigación.

- **Análisis documental:** Posibilitó la revisión y el análisis de los documentos normativos y manuales relacionados con las normas y el proceso de control eléctrico de los CMF del policlínico Orestes Falls Oñate.

Método matemático

- **Análisis porcentual:** Con base en el cálculo de los puntos de caso de uso, se busca determinar el tiempo de desarrollo, el costo total y el esfuerzo estimado por cada persona involucrada en el proceso. Asimismo, se pretende identificar los beneficios tangibles e intangibles que aportará la solución propuesta y analizar los costos con el fin de establecer su viabilidad.

- **Población:** Está constituida por todos los trabajadores que integran el departamento administración que actualmente son 3 personas, el administrador, la planificadora y la secretaria y 25 consultorios del Policlínico Orestes Falls Oñate.

- **Muestra:** La muestra representa el 100 % de la población para que sea suficientemente representativa a la investigación y luego pueda generalizarse.

Justificación de la muestra: La selección de la muestra constituye el total de sujetos de la población porque al ser tan pequeña puede ser abarcada en su totalidad.

Se utilizó como base la metodología XP (Tabla 1), donde las fases de exploración, planificación y diseño definidas por la metodología ágil Programación Extrema (XP) presentada en Letelier y Penadés (2012).

Según la metodología se desarrollan los artefactos correspondientes como son: las historias de usuario (HU), el plan de iteraciones, el plan de entregas y las tarjetas CRC (Clase, Responsabilidad y Colaboración). También se definen los requisitos funcionales y no funcionales del sistema. Es argumentada la arquitectura y los patrones de diseño utilizados, así como se describe el modelo de datos. Las entregas son frecuentes, y existe una refactorización continua, lo que nos permite mejorar el diseño cada vez que se le añada una nueva funcionalidad.

Tabla 1. Plan de iteración de la metodología XP

Fases metodológicas	Actividades	Recursos
I Planificación del proyecto	Descripción del sistema Roles Historias de usuario. Requisitos no funcionales. Estimación de esfuerzos Plan de iteración Plan de entrega.	- Servidor web - Computadora del cliente - Teléfono móvil
II Diseño	Modelo de datos Tarjetas CRC Arquitectura del *Software* Patrones Arquitectónicos Patrones de Diseño	- Computadora del cliente - Teléfono móvil
III Implementación, prueba y validación	Tareas de ingeniería Estándar de codificación Diagrama de despliegue Prueba de aceptación Detección y corrección de errores Evaluación de la eficiencia	- Servidor web - Computadora del cliente - Teléfono móvil

Resultados

Fase de planificación del proyecto

A grandes rasgos, en la fase inicial los clientes plantean las historias de usuarios que son de interés para la primera entrega del producto. Además, el equipo de desarrollo se va familiarizando con las herramientas, tecnología y práctica que se utilizan en el proyecto. En la tabla 2 se muestran los roles o grupos del sistema.

Tabla 1. Plan de iteración de la metodología XP

Rol	Responsabilidad
Administrador (administrador)	Es el que gestiona el funcionamiento del *software*, como la creación, actualización y deshabilitación de usuarios, cambios de contraseña, nomencladores y crea planificaciones, entrena modelos y ejecuta predicciones, filtrar y realizar operaciones CRUD a todos los metrocontadores y exportar registros Excel. realiza copias de seguridad, envía notificaciones a usuarios
Cliente (usuario)	Puede filtrar y realizar operaciones CRUD a su metrocontador y exportar registros Excel.

Además, se debe tener en cuenta que, las historias de usuarios (HU) sustituyen a los documentos de especificación funcional, y a los casos de uso. Estas historias son escritas por el cliente, en su propio lenguaje, como descripciones cortas de lo que el sistema debe realizar. La diferencia más importante entre estas historias y los tradicionales documentos de especificación funcional se encuentra en el nivel de detalle requerido. Las historias de usuario deben tener el detalle mínimo como para que los programadores puedan realizar una estimación poco riesgosa del tiempo que llevará su desarrollo. Cuando llegue el momento de la implementación, los desarrolladores dialogarán directamente con el cliente para obtener todos los detalles necesarios. Las historias de usuarios deben poder ser programadas en un tiempo entre una y tres semanas. Si la estimación es superior a tres semanas, debe ser dividida en dos o más historias. Si es menos de una semana, se debe combinar con otra historia.

Las HU se representan mediante tablas, las cuales se dividen en varias secciones denominadas:

- Número: identificador de la HU.
- Nombre: nombre que identifica a la HU.
- Usuario: involucrados en la ejecución de la HU.
- Iteración asignada: iteración en que se implementará la HU.
- Prioridad en el negocio: prioridad de la HU con respecto al resto de las HU (alta, media o baja).
- Riesgo en desarrollo: riesgo en la implementación de la HU (alto, medio o bajo)
- Puntos estimados: estima el esfuerzo asociado a la implementación de la HU. Un punto equivale a una semana ideal de programación, siendo utilizados, generalmente, de uno a tres puntos.
- Descripción: descripción sintetizada de la HU.
- Observaciones: información adicional.

Como resultado del trabajo realizado durante la fase de exploración se identificaron un total de siete (7) historias de usuario que están listadas a continuación:

- Gestionar Usuarios
- Gestionar metrocontadores
- Gestionar Planes de Consumo
- Gestionar Registros
- Gestionar copias de seguridad
- Gestionar mensajes y notificaciones
- Gestionar predicción de consumo eléctrico

En la Tabla 3 se presenta la HU "Registros" ya que es una de las principales funcionalidades del sistema.

Tabla 3. Historia de usuario gestionar registros

Historia de usuario	
Número: 4	**Nombre:** Gestionar Registros
Usuario: Administrador y Cliente	**Iteración asignada:** 2
Prioridad en negocio: Alta	**Puntos estimados**: 1
Riesgo en desarrollo: Medio	**Puntos reales**: 2
Descripción: El usuario gestiona los detalles del proceso correspondientes a su rol con las operaciones de insertar, listar, actualizar y eliminar. Realizar filtros de los registros	
Observaciones: • El rol de cliente tiene todos los permisos al metrocontador asignado. • El rol de Administrador tiene permiso a todos los metrocontadores del sistema y puede exportar los registros.	

Requisitos no funcionales

Los requisitos no funcionales (RNF) detallan las propiedades o cualidades que el producto debe tener, aumentándole funcionalidad al sistema, pues hacen al producto atractivo, fácil de usar, rápido y confiable. A continuación, se presentan los requisitos no funcionales definidos para la aplicación. (Eriksson, 2017)

Requisitos de usabilidad: RNF1: Los servicios que brindará la aplicación deben estar de acorde al poco conocimiento de la informática que pueda tener el usuario. La aplicación web debe poseer un diseño adaptativo, a fin de garantizar la adecuada visualización en múltiples computadores personales, dispositivos tableta y teléfonos inteligentes.

Requisito de soporte: RNF2: se mantendrá un sistema de codificación estándar siguiendo las normas establecidas.

Requisitos de interfaz: RNF3: el sistema debe ofrecer una interfaz atractiva, amigable y fácil de operar, la tipografía debe ser uniforme, de un tamaño adecuado y con un contraste que resalte los textos. Todas las interfaces deben estar en idioma español.

Requisitos de portabilidad: RNF4: el sistema debe desarrollarse sin basarse en características propias de algún sistema operativo, para lograr un producto multiplataforma.

Requisitos de seguridad:

Autenticación: RNF5: los usuarios se autenticarán haciendo uso de la aplicación para autenticación integrada en Quasar con la dependencia de Pinia y Axios.

Confiabilidad: RNF6: la información manejada por el sistema estará protegida de accesos no autorizados previamente definido por los roles.

Integridad: RNF7: la información que es manejada por el sistema permanecerá inalterada, a menos que sea modificada por el personal autorizado.

Disponibilidad: RNF8: solo tendrá acceso al sistema el personal autorizado.

Requisitos de ambiente de ejecución:

En el cliente:

RNF9: *Software*: navegador web con soporte para HTML 5 y CSS 3.

RNF10: *Hardware*: como mínimo 256 MB de memoria RAM, como mínimo un microprocesador a 550 MHz, conexión de red.

Servidor web:

RNF11: *Software*: Servidor HTTP Nginx 1.24.0 o superior

RNF12: *Hardware*: como mínimo 4096 MB de memoria RAM, con un microprocesador a 2.0 GHz.

Servidor de bases de datos:

RNF13: *Software*: Gestor de Base de Datos PostgreSQL 14.10 o superior.

RNF14: *Hardware*, como mínimo 256 MB de memoria RAM, con un microprocesador a 1.0 GHz y 10 Gb de Disco Duro.

La metodología de desarrollo de *software*, muestra la planificación como un permanente diálogo entre el usuario final y el equipo de desarrollo, definiéndose lo que el *software* tiene que resolver para que genere algún valor. En esta fase el cliente establece la prioridad de cada HU y los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Típicamente esta fase consiste en una o varias reuniones grupales de planificación. El resultado de esta fase es un Plan de Entrega. Una vez acordado este cronograma, comienza una fase de iteraciones, en dónde en cada una de ellas se desarrolla, prueba e instala unas pocas historias de usuarios (HU).

Estimación de esfuerzo por Historia de Usuario

Para el desarrollo de la solución, se realizó la estimación de esfuerzo para cada una de las HU, como se puede apreciar en la tabla 4.

Tabla 4. Puntos de estimación por HU

Nro.	Historia de usuario	Estimación (semana)
1	Gestionar Usuarios	1
2	Gestionar metrocontadores	1
3	Gestionar planes de consumo	2
4	Gestionar registros	4
5	Gestionar copias de seguridad	1
6	Generar notificaciones	1
7	Gestionar predicción de consumo eléctrico	8
Total		**18**

Plan de iteración

A continuación, se describen cada una de las iteraciones propuestas, donde la duración total de iteraciones se obtiene a partir del esfuerzo estimado por el desarrollador:

Iteración 1: en esta iteración se implementan las HU necesarias, relacionadas con los metrocontadores, condiciones de estos, y registros

Iteración 2: en esta iteración se implementan las HU relacionadas con prioridad alta en el negocio, la cual tiene como finalidad realizar el inicio, procesos y actualización.

Iteración 3: en esta iteración se implementan las HU relacionadas con las predicciones de consumo. Se obtendrá la versión 1.0 del sistema de forma tal que los usuarios puedan probar todas las funcionalidades y explotar el sistema a plenitud.

A modo de resumen se presenta una Tabla 5 que muestra la duración de cada iteración, así como el orden en que serán implementadas las HU en cada una de ellas según la prioridad asignada por el cliente:

Tabla 5. Plan de duración de las iteraciones

Iteración	Historia de usuario	Duración
1	Gestionar metrocontadores	5
	Gestionar registros	
2	Gestionar usuarios	5
	Gestionar planes de consumo	
	Gestionar copias de seguridad	
	Generar notificaciones	
3	Gestionar predicción de consumo eléctrico	8

Plan de entrega

Dado el plan de iteraciones realizado anteriormente, el plan de entregas acordado con el cliente se resume en la Tabla 6:

Tabla 6. Plan de entrega

Nro.	Historia de Usuario	Estimación (semanas)	Fecha (inicio - fin)
1	Gestionar metrocontadores	1	19/06/23 - 25/06/23
2	Gestionar registros	4	26/06/23 - 26/07/23
3	Gestionar usuarios	1	31/07/23 - 07/08/23
4	Gestionar planes de consumo	2	07/08/23 - 14/08/23
5	Gestionar copias de seguridad	1	14/08/23 - 18/08/23
6	Gestionar predicción de consumo eléctrico	8	21/08/23 - 21/10/23
7	Generar notificaciones y mensajes	1	23/10/23 - 27/10/23

Fase de diseño

La metodología de desarrollo de *software* XP sugiere la elaboración de diseños sencillos para lograr un fácil entendimiento en la fase de desarrollo, permitiendo la reducción del tiempo de elaboración de la tarea asignada al desarrollador. Se deben evitar a toda costa la complejidad innecesaria y el código extra. El *software* que posee un diseño adecuado es aquel que refleja claramente la intención de implementación de los programadores, supera con éxito todas las pruebas, no tiene lógica duplicada y tiene el menor número posible de clases y métodos.

XP construye un proceso de diseño evolutivo, que se basa en describir la modificación del código fuente sin cambiar el comportamiento del sistema y lograr así que sea simple en cada iteración. En el diseño no se realiza nada de forma anticipada para necesidades futuras, solo se centra en la iteración actual. Se obtiene como resultado un proceso de diseño disciplinado, que combina la disciplina con la adaptabilidad y logra que sea una de las más desarrolladas de entre todas las metodologías ágiles (Canós, Letelier y Penadés, 2011).

Modelo de red LSTM

A partir del estudio se procedió a analizar la implementación de la red neuronal LSTM en la vista de entrenamiento de modelos de Django, utilizada por el usuario en cuestión.

- Generalidades: La implementación de la red LSTM se lleva a cabo en una vista de Django denominada TrainModelView. En este contexto, se destaca la utilización de TensorFlow y Keras para construir y entrenar la red, así como la adopción de un enfoque de serialización para preparar los datos provenientes de la base de datos mediante el RegistryPredictionSerializer.

- Preprocesamiento de datos: El código demuestra un sólido preprocesamiento de datos, incluyendo la conversión de fechas, la creación de nuevas características y la gestión de tipos de datos. La eliminación de columnas no relevantes y la reorganización estructurada de los datos contribuyen a la preparación eficiente para el modelado.

- Configuración del modelo: La arquitectura del modelo LSTM se define adecuadamente en la sección correspondiente, con capas de entrada, LSTM y capas densas. La elección de hiperparámetros es transparente, destacando valores cruciales como *sequence length, batch size, epochs y patience*. Además, se especifica la función de pérdida como el error cuadrático medio (mse) y se utiliza el optimizador Adam con una tasa de aprendizaje de 5e-4.

- Entrenamiento y evaluación: Se implementan generadores de series temporales para el entrenamiento y la evaluación, y se incorpora Early Stopping para evitar el sobreajuste. La persistencia de modelos en archivos facilita su posterior recuperación y uso. El seguimiento y registro de métricas de entrenamiento, validación y prueba proporciona una evaluación comprehensiva del rendimiento del modelo.

- En la vista denominada PredictView, se lleva a cabo la utilización de modelos previamente entrenados para realizar predicciones sobre el consumo eléctrico del próximo mes. Este análisis se centra en la estructura y funcionalidades implementadas en dicho código.

- Método predict_with_model: En esta instancia, se define un método denominado predict_with_model, el cual recibe un modelo, datos de entrada, sequence length y batch size como parámetros. Este método utiliza la función tf.keras.preprocessing.timeseries_dataset_from_array para preparar los datos y, posteriormente, realiza predicciones con el modelo proporcionado. La salida del modelo es retornada por el método.

- Método get: La vista PredictView se presenta como una clase que hereda de APIView. En este contexto, se emplea la función load_model de Keras para cargar modelos previamente entrenados desde la carpeta "apps/predictions/models-keras". Además, se obtiene la fecha actual y se calcula el mes siguiente para generar registros ficticios de consumo eléctrico para dicho período.

- Generación de registros ficticios: Se procede a la generación de registros ficticios para cada metrocontador, incorporando información relevante como día, mes, días transcurridos y día de la semana. Este proceso se realiza en concordancia con la estructura de los modelos previamente entrenados.

- Predicciones y Almacenamiento en la Base de Datos: Previamente a la realización de nuevas predicciones, se elimina cualquier predicción existente en la base de datos. Para cada metrocontador y su respectivo modelo, se lleva a cabo la predicción utilizando el método predict_with_model. Los resultados de las predicciones se suman para obtener el consumo total estimado para el próximo mes. Se procede a la creación de instancias de la clase Predictions con los datos obtenidos, las cuales son posteriormente almacenadas en la base de datos.

Modelo de datos

El modelo de datos correspondiente a los módulos desarrollados utiliza llaves subrogadas para facilitar la manipulación de la información, así como tablas nomencladoras para el almacenamiento estructurado de datos. Está compuesto por un total de 29 tablas: 9 de ellas son principales y fueron modeladas directamente por el desarrollador; otras 9 son generadas por la biblioteca *Django Simple History*, la cual almacena el estado del modelo en cada operación de creación, actualización o eliminación, permitiendo incluso recuperar versiones anteriores y registrar qué usuario realizó cada cambio. Las 11 tablas restantes pertenecen al *framework* Django y se encargan de gestionar aspectos como migraciones, registros de actividad (*logs*) y tokens de autenticación para el funcionamiento general del sistema.

Tarjetas CRC

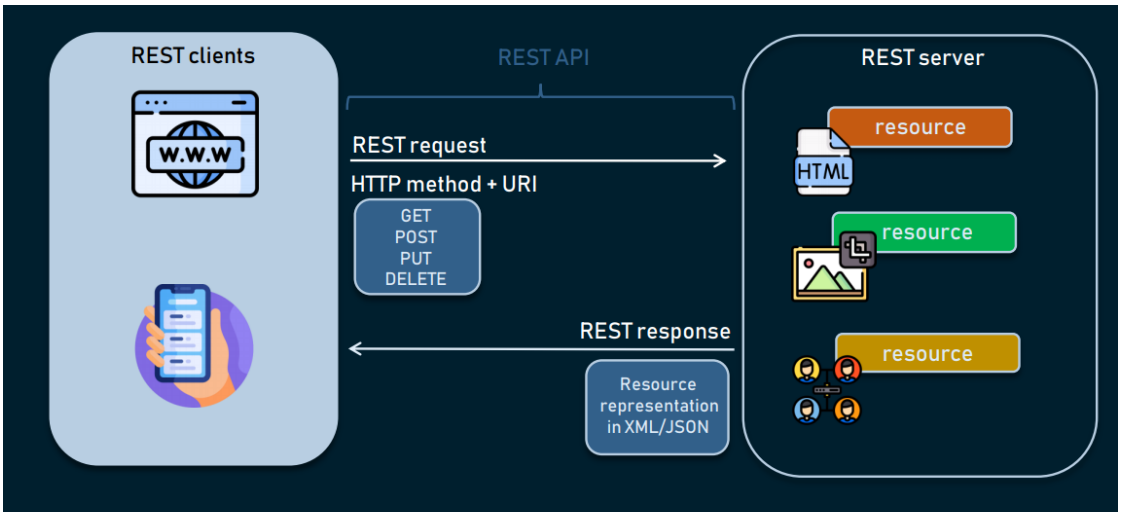
En la tabla 7 se muestra la estructura de la tarjeta CRC de la clase Registros ya que es una de las clases principales del sistema.

Clase: Registro	
Responsabilidades:	Colaboradores:
Gestionar los Procesos por Registros: • Crear • Listar: • Actualizar • Eliminar	Metrocontador

Arquitectura del *software*

En el desarrollo de la aplicación, se adoptó una arquitectura Cliente-Servidor, siguiendo el patrón arquitectónico Restful orientado al recurso (Resource-Oriented Architecture [ROA]). Este enfoque se implementó mediante dos aplicaciones individuales que colaboran para el funcionamiento del sistema. En el backend, se utilizó una aplicación servidor desarrollada en Django, haciendo uso de la biblioteca especializada en el desarrollo de API Restful, Django Rest *Framework*. Se integra con el enfoque MVT de Django. Sin embargo, a nivel de API REST, DRF sigue los principios de la arquitectura orientada a recursos (ROA) y utiliza patrones RESTful para diseñar y exponer API web (Figura 1).

Figura 1 Componentes de RESTful



En el contexto específico de Django y Django Rest *Framework*, el modelo arquitectónico REST se basa en 3 conceptos: Acciones, Recursos y Representaciones. Los recursos de API se asignan a sus Modelos a través de serializadores.

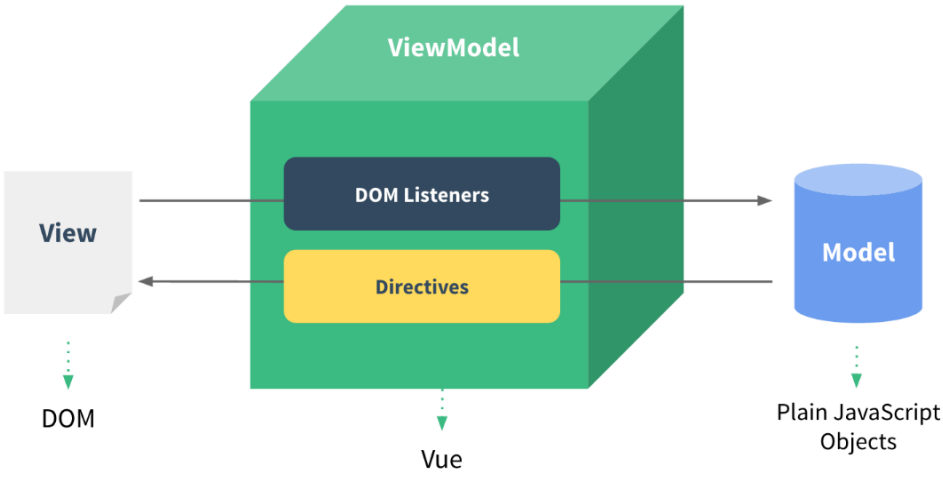
- El Comportamiento las acciones determinan lo que el cliente que consume la API desea hacer con un recurso dado. Las acciones se asignan a VERBOS HTTP. El marco REST agrega soporte para el enrutamiento automático de URL a Django y le proporciona una forma simple, rápida y consistente de conectar su lógica de vista a un conjunto de URL. Incluye rutas para el conjunto estándar de *list, create, retrieve, update, partial_update y destroy*.

- Las representaciones REST son la forma en que se muestran los datos de un recurso a quien realiza una solicitud en un punto final específico. En general, estas funciones se muestran en formato JSON. Básicamente, todos los datos en sus modelos, después de ser serializados, se representan como objetos JSON al devolver la solicitud. En caso de DRF los Viewsets le permite combinar la lógica de un conjunto de vistas relacionadas en una sola clase. Las acciones previstas por el ModelViewSet clase son. list(), .retrieve(), .create(), .update(), .partial_update(), y .destroy().

Por otro lado, el *frontend* consta de una aplicación cliente en el navegador de los usuarios, desarrollada con Quasar *Framework*, que opera sobre Vue.js y este implementa principalmente el patrón de arquitectura de *software* Model – View – ViewModel o sus más conocidas siglas MVVM. Patrón que tiene como propósito separar el código de la UI y el código que no es de la UI.

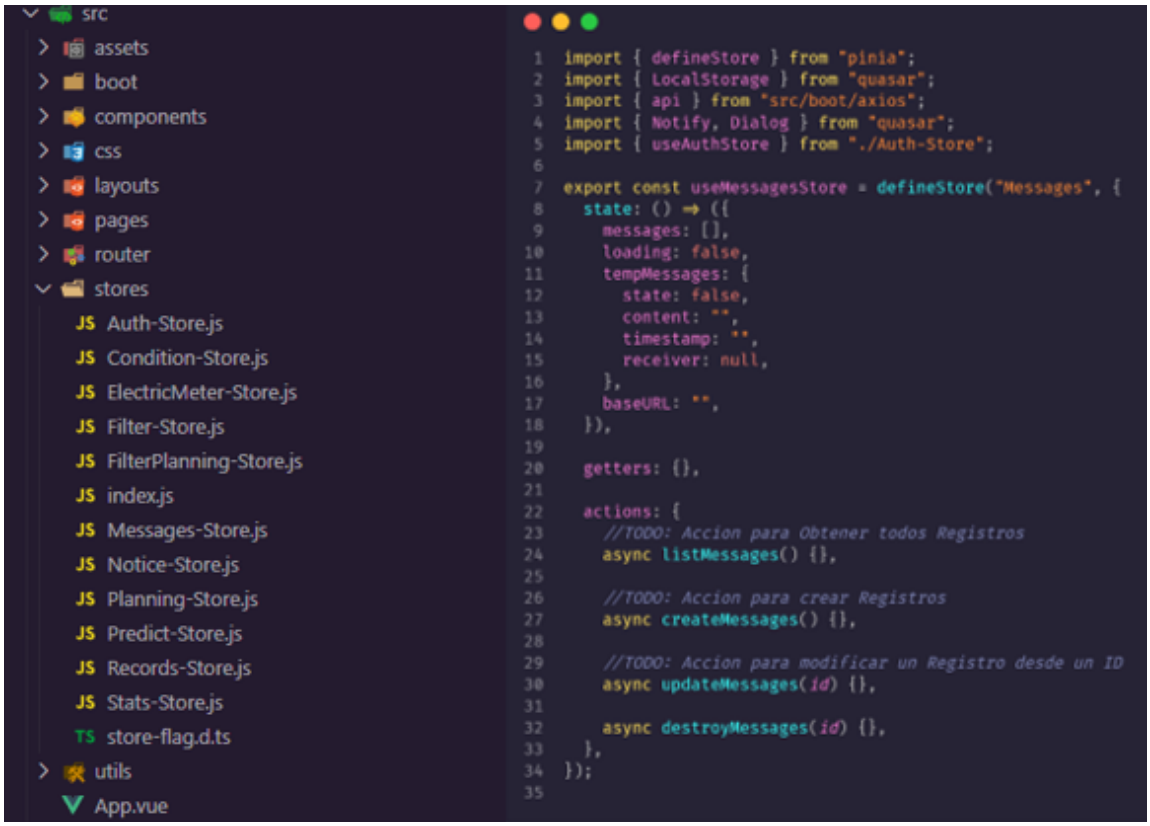
En el contexto específico de Vue.js y Quasar *Framework*, la estructura del componente se desglosa tal y como se presenta en la Figura 2.

Figura 2 Estructura MVVM



Modelo (Store de Pinia): En esta implementación, el modelo corresponde al store de Pinia. Este componente gestiona el estado de la aplicación y define acciones que interactúan con la API para obtener y gestionar la información (Figura 3).

Figura 3 tore de estados en Pinia



Vista (Bloque <template>): La vista se define dentro del bloque <template>, representando la estructura y presentación de la interfaz de usuario (Figura 4).

Figura 4 Bloque Template Componente de Quasar *Framework*

```
<template>
  <q-page padding>
    <div class="row justify-center">
      <div
        class="text-primary text-h5 text-bold text-center col-12 q-pa-sm q-animate--scale"
      >
        Copias de Seguridad y Restauración
      </div>
      <DownloadBackup />
      <RestoreBackup />
    </div>
  </q-page>
</template>
```

ViewModel o lógica del componente (Bloque <script>): La lógica del componente se encuentra en el bloque <script>, el cual no solo incluye la lógica del componente sino también la instancia de Vue que define dicho componente. Este bloque puede considerarse como el controlador o la lógica detrás de la vista (Figura 5).

Figura 3 Bloque Script Componente de Quasar *Framework*

```
<script setup>
import { onBeforeMount } from "vue";
import { useAuthStore } from "src/stores/Auth-Store";
import DownloadBackup from "src/components/cards/DownloadBackup.vue";
import RestoreBackup from "src/components/cards/RestoreBackup.vue";

const { verifyToken } = useAuthStore();

onBeforeMount(async () => {
  await verifyToken();
});
</script>
```

- Fase de implementación, prueba y validación

Una vez identificadas las HU, el programador procede a descomponer cada una en tareas específicas para ser implementadas (las denominadas tareas de ingeniería), que están escritas técnicamente y que darán solución a la HU correspondiente.

Las tareas de ingeniería consisten en la división de cada historia de usuario en varias tareas que luego son implementadas por el programador. En la Tabla 8 se encuentran las tareas de ingeniería definidas por HU.

Tabla 8 Crear interfaz para filtrar registros de consumo eléctrico

Tarea de ingeniería	
Número tarea: 8	**Número HU:** 4
Nombre tarea: Crear interfaz para visualizar y filtrar registros de consumo reportadas eléctrico de los metrocontadores	
Tipo tarea: Desarrollo	**Puntos estimados:** 3 días
Fecha inicio: 17/07/23	**Fecha fin**: 20/07/23
Programador responsable: William Jiménez Puentes	
Descripción: esta tarea facilita la creación de una interfaz para filtrar los registros de acuerdo a los metrocontadores, consumo, año, mes, fecha, rango de fecha y rango de consumo	

En el desarrollo de la aplicación, se decide usar el estándar de codificación de JavaScript llamado Standard Style, una guía de estilo y conjunto de reglas que promueve prácticas consistentes y legibles en el desarrollo de aplicaciones JavaScript. Además de algunas recomendaciones propias. De forma general estas incluyen:

- Uso de punto y coma o una coma al final de cada declaración en dependencia.
- Indentación (sangría, *indentation*) con dos espacios.
- Uso de comillas dobles para cadenas de texto.
- La llave de apertura de las clases debe ir en la misma línea que el nombre de la clase.
- Variables y funciones escritas en camelCase.
- Uso de *const* o *let* en lugar de *var* para declarar variables.
- Espacios alrededor de operadores y después de comas.
- Uso de *arrow functions* (funciones flecha) cuando sea posible.
- Uso de *async/await* para manejar promesas.
- Nombres de archivos y carpetas en minúsculas y separados por guiones bajos.
- Uso de comentarios descriptivos para explicar el código y su funcionamiento.
- Convenciones de nomenclatura específicas para diferentes componentes de la aplicación. Por ejemplo, los modelos se nombran en singular en PascalCase (por ejemplo, *user*), las tablas de la base de datos se nombran en plural en snake case (por ejemplo, *users*), y los componentes en PascalCase (por ejemplo, EssentialLink).
- Utiliza migraciones de base de datos para gestionar los cambios en la estructura de la base de datos. Las migraciones siguen una convención de nomenclatura específica y proporcionan métodos para crear, modificar y eliminar tablas y columnas.

Utiliza una convención de enrutamiento basada en REST para definir las rutas de la aplicación. Se recomienda seguir esta convención para mantener una estructura coherente y predecible en las rutas de la aplicación.

Se define además el modelo de despliegue, el cual describe la configuración de los nodos del sistema en tiempo de ejecución, los enlaces de comunicación entre ellos y otros dispositivos sin función de cómputo que residen en la misma red. Los requerimientos de *software* y *hardware* asociados a los nodos de procesamiento del modelo se presentan en la Tabla 9.

Tabla 9 Requisitos de despliegue

Servidor Web:	Servidor API	Servidor de Bases de Datos:	PC Cliente:
1. Procesador 1.2 GHz 2. 512 MB de RAM. 3. Servidor HTTP Nginx 1.18.0	1. Procesador 2.0 GHz 2. 2048 MB de RAM. 3. Python 3.10.12	1. Procesador 1.2 GHz 2. 512 MB de RAM. 3. 40 GB de Disco Duro. 4. Gestor de Base de Datos PostgreSQL 14.9 o superior	1. Procesador 1.2 GHz 2. 512 MB de RAM. 3. Navegador Web con JavaScript habilitado y soporte HTML5 y CSS3.

La estrategia de pruebas de *software* describe el enfoque y los objetivos generales de las actividades de prueba. Incluye los niveles de prueba, el tipo de prueba a ser ejecutada y los casos de prueba diseñados para lograr los objetivos.

• Prueba de aceptación: Las pruebas de aceptación se derivan de las historias de usuario que se han implementado como parte de la liberación del *software* para verificar que este cumple con lo especificado por el cliente (Pressman, 2010). A continuación, se muestra el caso de prueba de aceptación diseñado para la HU Gestionar necesidades de capacitación debido a que es una funcionalidad básica del sistema.

A continuación, se muestra la prueba de aceptación (HU4P1), correspondiente a la HU 4 (Tabla 10).

Tabla 10 Muestra la prueba de aceptación (HU4P1)

Caso de prueba de aceptación	
Código: HU4P1	**Nro. Historia de usuario: 4**
Nombre: Gestionar Registros	
Descripción: Verificar que el usuario (administrador o cliente) pueda gestionar los detalles del proceso de registros, incluyendo las operaciones de insertar, listar, actualizar y eliminar, así como realizar filtros de los registros.	
Condiciones de ejecución: El usuario debe estar autenticado en el sistema.	
Entrada/pasos de ejecución: **1. El usuario se dirige al menú "Enlaces" y selecciona la opción "Registros".** **2. Selecciona el botón de añadir correspondiente a la tabla Registros del Sistema.** **3. Completa todos los campos del formulario en la ventana emergente y guarda la información para registrar los datos en el sistema.** **4. Selecciona la opción de añadir (+) para registrar más datos y mantener la ventana emergente abierta.** **5. Selecciona la opción de cancelar para cerrar la ventana emergente.** **6. En la lista de registros, prueba las acciones de editar y eliminar.** **7. Realiza filtros en los registros según requiera el usuario por (metrocontador, año, mes, consumo Kwh, rango de consumo Kwh, fecha, rango de fechas).**	
Resultado esperado: **• Los datos se insertan correctamente en la tabla de registros.** **• La ventana emergente se cierra correctamente al seleccionar cancelar.** **• Las acciones de editar y eliminar funcionan correctamente.** **• Los filtros de registros se aplican correctamente según los parámetros seleccionados por el usuario.**	
Resultado obtenido: Se realizaron correctamente las operaciones solicitadas por el usuario.	
Evaluación de la prueba: Prueba satisfactoria.	

Resultado de las pruebas de aceptación

Para validar que la salida emitida por el sistema informático fuese el resultado esperado por el cliente se desarrollaron ## pruebas de aceptación. Se realizó un total de 2 iteraciones para lograr alcanzar los resultados satisfactorios desde el punto de vista funcional de la aplicación.

- En la primera iteración, de 9 pruebas realizadas, 7 coincidieron con los resultados esperados por el cliente, esto representa un 78 % de satisfacción, mientras que ## pruebas resultaron fallidas que representan un 22 % de insatisfacción.

- En una última iteración, de 9 pruebas realizadas, 9 coincidieron con los resultados esperados por el cliente, por lo que se obtuvo el 100 % de pruebas satisfactorias, de manera que ninguna de las pruebas realizadas fue fallida lo que representa un 0 % de insatisfacción.

Evaluación de la mejora

Teniendo en cuenta que la mejora tiene que ver con la capacidad de aumentar la productividad o la calidad de un sistema, los indicadores de mejora están relacionados con las razones que indican los cambios en los recursos invertidos en la consecución de tareas y/o trabajos. A partir de los parámetros definidos para garantizar que dicho proceso sea eficiente, se realizó el siguiente estudio para analizar su comportamiento antes de la implementación de la mejora y luego de su utilización en la gestión de proyectos (Tabla 11).

Tabla 11 Indicadores para evaluar la mejora

Parámetro	Indicadores	Antes	Después
Tiempo de recopilación de datos	Tiempo promedio para recopilar los datos de consumo eléctrico	Aproximadamente 2 horas	Aproximadamente 30 minutos
Calidad de los datos	Porcentaje de datos completos y precisos	60 %	95 %
Acceso a los datos	Número de usuarios que pueden acceder a los datos	1	Varios cientos (depende del sistema)
Control y seguimiento de los procesos por especialistas	Calidad del control	Regular	Bueno
Recopilación de datos generales para elaborar informes	Calidad de los datos	Malo	Muy Bueno
Satisfacción de los clientes	Nivel de satisfacción	Regular	Muy Bueno

Con los resultados obtenidos, queda validada la mejora en el proceso como resultado de haber sido implementado el sistema propuesto.

Conclusiones

El desarrollo de los fundamentos teóricos que sustentan la investigación permitió profundizar en los conocimientos acerca del control energético de los CMF del policlínico Orestes Falls Oñate. Con la realización del diagnóstico sobre la situación actual del proceso de control energético de los CMF se evidenció la necesidad de la implementación de un sistema para el control energético con el objetivo de realizar un ciclo de vida del proceso de forma eficiente.

Además, mediante el diseño e implementación de las funcionalidades identificadas los autores obtuvieron un sistema que brinda todas las herramientas requeridas por el usuario. Se verificó y validó el sistema a partir de pruebas permitiendo comprobar el adecuado diseño del sistema, el cumplimento de los requisitos acordados con el cliente y el alto grado de satisfacción de los usuarios.

Con la aplicación del estudio comparativo se evidenció que el sistema CECMF permite el control eficiente del consumo energético de los CMF en la entidad objeto de estudio.

Además, como resultado del proceso de investigación y realización de la presente investigación se sugieren aspectos que serían importantes a tener en cuenta para el futuro perfeccionamiento del sistema:

- Compilar una aplicación nativa para dispositivos móviles que permita a un trabajador realizar todas las funcionalidades definidas desde su dispositivo móvil de forma más ágil.
- Ampliar los horizontes de la aplicación para trabajar con más de un centro.
- Para mantener los resultados obtenidos, se recomienda realizar un seguimiento del proceso de mejora a lo largo del tiempo. Esto permitirá identificar cualquier área que necesite ser mejorada o ajustada.

Referencias

Barrera, J. J. (2021). *Propuesta de un plan de eficiencia energética en el hotel Chrisban Hotel Boutique.* (Tesis de Grado, Universidad Antonio Nariño). https://repositorio.uan.edu.co/500

Canaviri, C. G. (2019) *Protocolos de eficiencia energética en instalaciones eléctricas y redes informáticas para el proyecto Mega Centro de llamadas DATACOM-ENTEL.* (Tesis Doctoral, Universidad Mayor de San Andrés). https://repositorio.umsa.bo/xmlui/handle/123456789/32646.

Canós, J. H., Letelier, P. y Penadés, M. C. (2011). *Métodologías Ágiles en el Desarrollo de Software.* DSIC -Universidad Politécnica de Valencia.

Espinosa, M del P.., Castro, C., Sánchez, R., y Oviedo, F. (2020). La relación sujeto-objeto durante el proceso de educación energética en estudiantes de técnico medio en electricidad en Cuba. *Revista Científica de FAREM-Estelí: Medio ambiente, tecnología y desarrollo humano, (35),* 15-30. https://dialnet.unirioja.es/servlet/articulo?codigo=7623840

Espinoza De La Grecca, L. A. (2023). *Implementación de un sistema fotovoltaico y su influencia en la eficiencia energética del alumbrado interno de la Universidad Ricardo Palma en el 2020.* (Tesis de maestría, Universidad Ricardo Palma). https://repositorio.urp.edu.pe/handle/20.500.14138/6443

Hidalgo, J. M. (2014). *Aplicación de la Prueba de la Necesidad para el diagnóstico energético del Hospital Clínico Quirúrgico Dr. Carlos Font Pupo de Banes.* (Tesis Doctoral, Instituto Superior Minero Metalúrgico De Moa). http://ninive.ismm.edu.cu/handle/123456789/2185

Letelier, P. y Penadés, M. C. (2012). Metodologías ágiles para el desarrollo de *software*: eXtreme Programming (XP). *Universidad Politécnica de Valencia, 17.* https://www.researchgate.net/profile/Patricio-Letelier/publication/28109707_Metodologias_agiles_para_el_desarrollo_de_*software*_eXtreme_Programming_XP/links/54ad00f10cf2479c2ee86820/Metodologias-agiles-para-el-desarrollo-de-*software*-eXtreme-Programming-XP.pdf

Martino, H. (2023). *Manual sobre la Aplicación de Medidas de Eficiencia Energética en Edificios Municipales.* https://sedici.unlp.edu.ar/handle/10915/148968

Pressman, R. S. (2002). *Ingenieria del software: un enfoque practico* (5.a. ed). Mcgraw-Hill Interamericana.

Stein, N., Consoli, E., Cohendoz, L. y Vereertbrugghen, D. (2023). Red de aprendizaje en eficiencia energética de cooperativas y empresas recuperadas (UNPAZ-Madygraf). In XXVIII *Reunión Anual Red Pymes Mercosur*. Universidad Nacional de Córdoba. https://www.aacademica.org/natalia.stein/32

Zambrano, E. E. A. y Real-Pérez, G. L. (2021). Las PYMES y la eficiencia energética con la ISO 50001. *Polo del Conocimiento: Revista científico-profesional, 6*(6), 674-694. https://dialnet.unirioja.es/servlet/articulo?codigo=8016995