**Aplicación web con arquitectura híbrida (MVC y modelo en capas) para promover el marketing digital y emprendimiento en la Universidad Popular del Cesar.**

**López Baleta Calletana**1,2**, Mejia Rodriguez Carlos Alberto**1,3**, González Mora Fernando**4 **and García Molina Laura Fernanda** 5

1Grupo de Investigación en Desarrollo y Aplicación de Tecnologías de la Información y la Comunicación - GIDEATIC, Universidad Popular del Cesar, Aguachica, Colombia.

2Especialista en TIC para el diseño de estrategias didácticas en educación. Universidad Popular del Cesar seccional Aguachica, ORCID: 0000-0003-2220-1199, calletanalopez@unicesar.edu.co

3 Magister en E-learning, Magister en Big Data. Universidad Popular del Cesar seccional Aguachica, carlosmejia@unicesar.edu.co

4Especialista en TIC. Universidad Popular del Cesar seccional Aguachica, fernando.gonzalez@unicesar.edu.co

5Ingeniera de Sistemas. Universidad Popular del Cesar seccional Aguachica, lfernandagarcia@unicesar.edu.co

**Resumen.** El proyecto tiene como principal objetivo la creación de una plataforma web integral destinada a impulsar el marketing digital y facilitar la divulgación de proyectos de emprendimiento. Esta iniciativa se enfoca en la Universidad Popular del Cesar, específicamente en su seccional de Aguachica, con el propósito de obtener financiamiento tanto desde el ámbito interno de la institución como a través de la colaboración con empresas y organizaciones externas. La plataforma web resultante permitirá la administración de datos de participantes, tanto estudiantes como docentes emprendedores, la gestión detallada de los proyectos emprendedores, el seguimiento y actualización del estado de cada proyecto, la publicación de proyectos aprobados en la plataforma, y la generación de informes y estadísticas para evaluar el progreso y el impacto de los proyectos. La arquitectura del sistema se basará en un enfoque híbrido, combinando elementos del patrón Modelo-Vista-Controlador (MVC) con una estructura de capas para garantizar una organización efectiva de datos, lógica empresarial y presentación en la plataforma.

***Palabras claves****: MVC, modelo en capas, promoción, marketing digital, emprendimiento.*

**Web application with hybrid architecture (MVC and layered model) to promote digital marketing and entrepreneurship at the Universidad Popular del Cesar.**

**Abstract**. The main objective of the project is the creation of a comprehensive web platform aimed at promoting digital marketing and facilitating the dissemination of entrepreneurial projects. This initiative focuses on the Popular University of Cesar, specifically in its Aguachica branch, with the purpose of obtaining financing both from within the institution and through collaboration with external companies and organizations. The resulting web platform will allow the administration of participant data, both students and entrepreneurial teachers, the detailed management of entrepreneurial projects, the monitoring and updating of the status of each project, the publication of approved projects on the platform, and the generation of reports. and statistics to evaluate the progress and impact of projects. The system architecture will be based on a hybrid approach, combining elements of the Model-View-Controller (MVC) pattern with a layered structure to ensure effective organization of data, business logic and presentation on the platform.

***Keywords*:** MVC, layered model, promotion, digital marketing, entrepreneurship.

**1. Introducción**

En un mundo cada vez más digitalizado, la aplicación del marketing digital y el fomento del espíritu emprendedor son cruciales. Para cubrir esta necesidad, se plantea el desarrollo de una aplicación web con una arquitectura híbrida que combina de manera eficaz el patrón Modelo-Vista-Controlador (MVC) y el modelo en capas. Esta arquitectura proporciona una base robusta y organizada para la plataforma, permitiendo la fácil gestión de datos, la lógica empresarial y la presentación. La aplicación tiene como objetivo facilitar la publicación de proyectos de emprendimiento en la Universidad Popular del Cesar Seccional Aguachica, promoviendo la colaboración o patrocinio interno y externo. A través de esta plataforma, se busca estimular el desarrollo de habilidades digitales y la difusión de ideas innovadoras en diversos ámbitos y contextos.

**2. Estado del arte**

Los trabajos de titulación presentados abordan el desarrollo de soluciones tecnológicas para diferentes problemas o casos de uso. Los primeros tres trabajos tienen en común la utilización del patrón de arquitectura modelo-vista-controlador (MVC) para el desarrollo de sus aplicaciones web. El primero crea una aplicación para gestión administrativa de un comité, el segundo desarrolla una aplicación para manejo de información de parque automotor, y el tercero propone un asistente virtual para optimizar procesos en una empresa.

Por otro lado, los tres últimos trabajos presentados comparten el uso de una arquitectura multicapa o en capas para sus desarrollos. El cuarto proyecto plantea una arquitectura multicapa basada en microservicios para aplicaciones de salud. El quinto valida una arquitectura flexible en capas orientadas a servicios para un repositorio educativo. Finalmente, el sexto implementa componentes como servicios web y monitoreo en una arquitectura en capas utilizando la metodología ágil Scrum. A continuación, se amplía detalladamente el contexto de los 6 artículos:

El trabajo de Rosales (2023) presenta el desarrollo de una aplicación web para gestionar los procesos administrativos del comité de seguro campesino en la comuna Cerezal Bellavista. Actualmente llevan los procesos de forma manual generando grandes volúmenes de registros físicos, inconsistencias e ineficiencias. La aplicación web propuesta incluiría módulos de seguridad, registro de pagos, asistencia e informes, permitiendo un mejor control y agilidad en los procesos.

La investigación de Chugá & Hernández (2023) detalla el desarrollo de un prototipo de aplicación para mejorar el manejo de información del parque automotor del GAD Municipal de San Pedro de Huaca. Mediante encuestas y entrevistas se determinaron los procesos y problemas actuales para luego crear la aplicación web buscando mejoras en órdenes, mantenimiento, inventario y reportes. La aplicación centraliza la información para una mejor gestión y toma de decisiones.

El trabajo de Vera (2023) propone desarrollar un asistente virtual o chatbot para mejorar el proceso de devoluciones de repuestos en la empresa COJAPAN Ltda. Plantea una solución tecnológica ante los problemas actuales de demoras e ineficiencias. Describir la metodología de investigación, el diseño propuesto y el desarrollo mediante herramientas de código abierto, buscando agilizar y optimizar el proceso.

La investigación de Calderón (2023) tiene como objetivo proponer una arquitectura multicapa para soportar la interoperabilidad entre aplicaciones de salud tradicionales y modernas usando microservicios. Se basa en una gran base de datos heterogénea y se adapta a un caso de uso real para la detección temprana de enfermedades. Definir características de software para televigilancia médica, evaluando ventajas y desventajas de los microservicios.

En la tesis de maestría de Luna (2016) se desarrolla y valida una arquitectura flexible orientada a servicios para implementar un repositorio de recursos educativos. Permite recuperar y almacenar recursos multimedia, además de visualizarlos en línea sin descargar. Se analizaron tecnologías y estándares, y se implementó un prototipo que cumplió con los objetivos planteados.

En el trabajo de investigación de Ortega (2015) se desarrollan servicios web y monitoreo como parte de la arquitectura de una aplicación universitaria usando Scrum. Implementa servicios SEC y SEP con Java, REST y JSON para consultas y procedimientos dinámicos. También desarrolla un sistema de monitoreo ELK para recolectar y visualizar bitácoras. Provee una capacidad de servicios reutilizable y un sistema de monitoreo útil para el desarrollo de aplicaciones.

**3. Materiales y métodos**

La ejecución del trabajo se llevó a cabo mediante la implementación de la metodología SCRUM, ya que permite garantizar el desarrollo óptimo del sistema. Considerando el alcance del proyecto y sus componentes, se optó por su utilización ya que describe un enfoque de trabajo ágil que auxilia a individuos, equipos y organizaciones en la generación de valor a través de soluciones adaptables para problemas complejos (Francia, n.d.).

Esta metodología ágil promueve la colaboración en el trabajo, estableciendo roles definidos y tareas específicas para los involucrados. Los procesos se desarrollan de forma iterativa a través de reuniones semanales llamadas "Sprints", lo que permite un seguimiento constante del progreso. Además, SCRUM permite fijar objetivos específicos y alcanzables dentro del proyecto y detectar posibles deficiencias a través de un proceso continuo de retroalimentación (Schwaber & Sutherland, 2020a).

*3.1 Metodología*

Según Schwaber & Sutherland (2020), SCRUM, es un marco de trabajo ágil, utilizado para gestionar proyectos complejos, enfocado en entregar valor de forma iterativa e incremental. Es agile con un ciclo de vida adaptativo, por tal razón para los autores no es considerada una metodología sino un marco de trabajo; debido a que se basa en la teoría empírica y el pensamiento Lean (mejora continua), con pilares de transparencia, inspección y adaptación (Schwaber & Sutherland, 2020b).

Cada proyecto SCRUM, realiza un número de sprints, cada “Sprint” es el término utilizado para una “iteración”. Se utiliza el Product Backlog (Una pila de producto) para así definir el alcance del resto del trabajo. Se van seleccionando los primeros elementos, tareas o actividades de la pila, se realiza el Sprint Planning, donde el Scrum Team selecciona los objetivos y se añaden al Sprint Backlog (Pila del Sprint), al ser considerado terminado, van a Increment y al Sprint Review para la inspección del resultado, que es el penúltimo evento del Sprint a fin de concluir en Spring Retrospective; y así sucesivamente hasta que suceda una de dos cosas, se lleva a feliz culminación el proyecto o se termina porque no es justificable o no es viable su finalización. Para todo este proceso, se tiene en cuenta lo siguiente:

*Valores clave:* compromiso, enfoque, franqueza, respeto y coraje.

*Roles:* Product Owner, Scrum Master y Developers.

*Artefactos clave:* Product Backlog, Sprint Backlog e Increment.

*Eventos:* Sprint (1-4 semanas), Sprint Planning, Daily Scrum, Sprint Review y Sprint Retrospective.

*El Product Owner* gestiona el Product Backlog priorizando los requisitos.

*El Scrum Master* facilita el proceso y elimina impedimentos.

*Los Developers* auto-gestionan su trabajo en Sprints para entregar incrementos de valor.

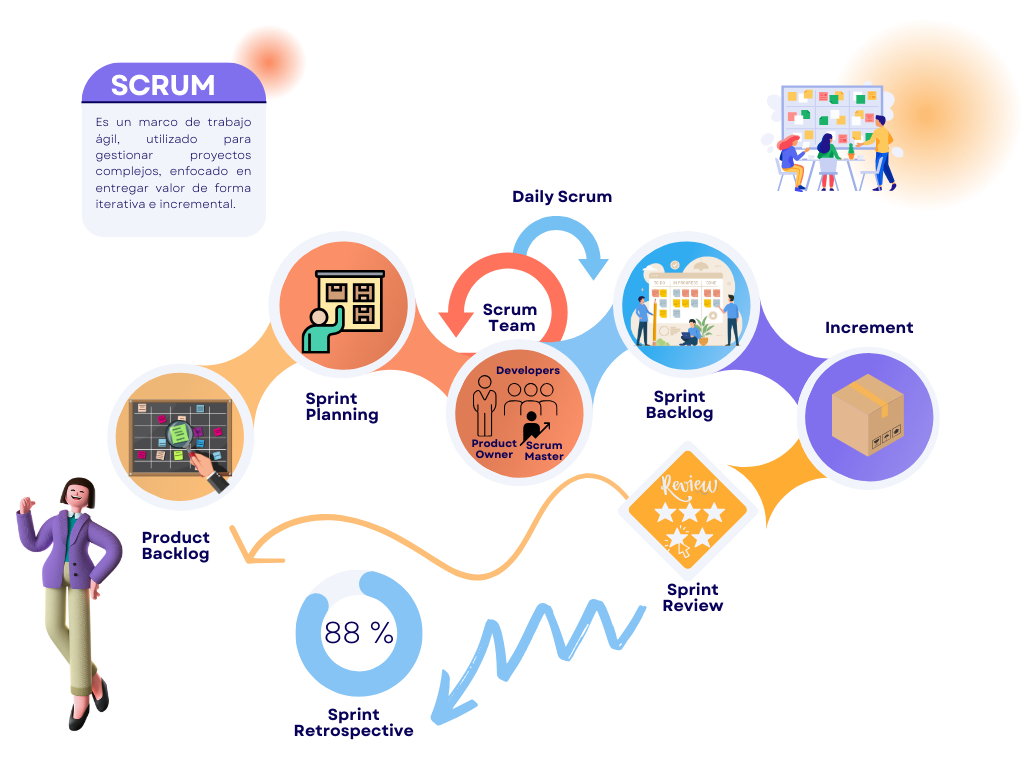
*El Sprint Backlog* detalla los elementos seleccionados del Product Backlog para el Sprint actual.

Al final de cada Sprint se obtiene un Incremento de producto potencialmente entregable, se realizan reuniones cortas diarias para inspección y adaptación. Se hace la revisión y retrospectiva al finalizar el Sprint o para evaluar e incorporar mejoras.

Para mayor comprensión se realiza la siguiente esquematización sobre el marco de trabajo SCRUM.

Figura 1

*Marco de trabajo SCRUM*



*Nota.* Adaptación del marco de trabajo SCRUM (Schwaber & Sutherland, 2020b)

*3.2 Definición de la Arquitectura*

Para el desarrollo de software se utilizan dos arquitecturas: Modelo – Vista – Controlador (MVC) y la arquitectura en Capas lo cual conllevó a una arquitectura hibrida resultante de las dos arquitecturas mencionadas, lo cual implica la combinación de diferentes enfoques arquitectónicos o la integración de múltiples tecnologías para abordar de manera efectiva los requisitos y desafíos específicos de un sistema o una infraestructura, aprovechando lo mejor de cada enfoque para lograr los objetivos deseados (Cartés, 2011).

*3.2.1 FrontEnd – Arquitectura MVC*

Arquitectura del Frontend: Modelo Vista Controlador (MVC)

Para la implementación del frontend en el proyecto, se eligió adoptar la arquitectura de software conocida como Modelo Vista Controlador (MVC), un patrón de diseño ampliamente reconocido y consolidado. Este enfoque estructurado y eficiente para el diseño y desarrollo de aplicaciones se fundamenta en la clara y distinta separación de responsabilidades dentro de un sistema, dividiendo la aplicación en tres componentes clave: Modelo, Vista y Controlador.

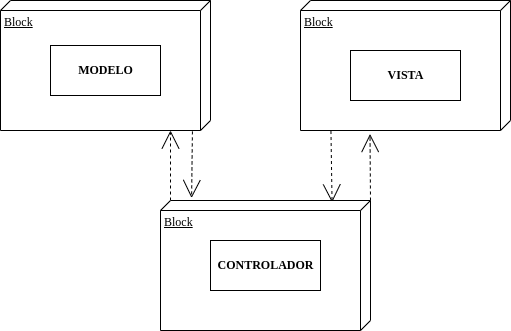
*Modelo:* El componente del Modelo se encarga de gestionar los datos y la lógica de negocio de la aplicación. Aquí es donde se almacena y manipula la información esencial para el sistema. Esta separación de datos permite una gestión más eficiente y facilita la modificación y actualización de la información sin afectar directamente a la interfaz de usuario.

*Vista:* La Vista representa la interfaz gráfica de usuario con la que los usuarios interactúan. Se encarga de mostrar los datos del Modelo y de capturar las acciones del usuario, proporcionando una experiencia visual y de usuario atractiva e intuitiva. Al separar la lógica de presentación de la lógica de negocio, la Vista puede adaptarse y cambiarse sin afectar la manipulación de datos subyacente.

*Controlador:* El Controlador actúa como intermediario entre el Modelo y la Vista. Gestiona las solicitudes del usuario, interpreta las acciones realizadas en la interfaz gráfica y actualiza el Modelo en consecuencia. Al hacerlo, mantiene la comunicación fluida entre la Vista y el Modelo sin que ambos estén directamente acoplados. Esta separación de responsabilidades facilita la escalabilidad, la mantenibilidad y la colaboración en el desarrollo del software.

Figura 2

*Arquitectura MVC (Front-End)*



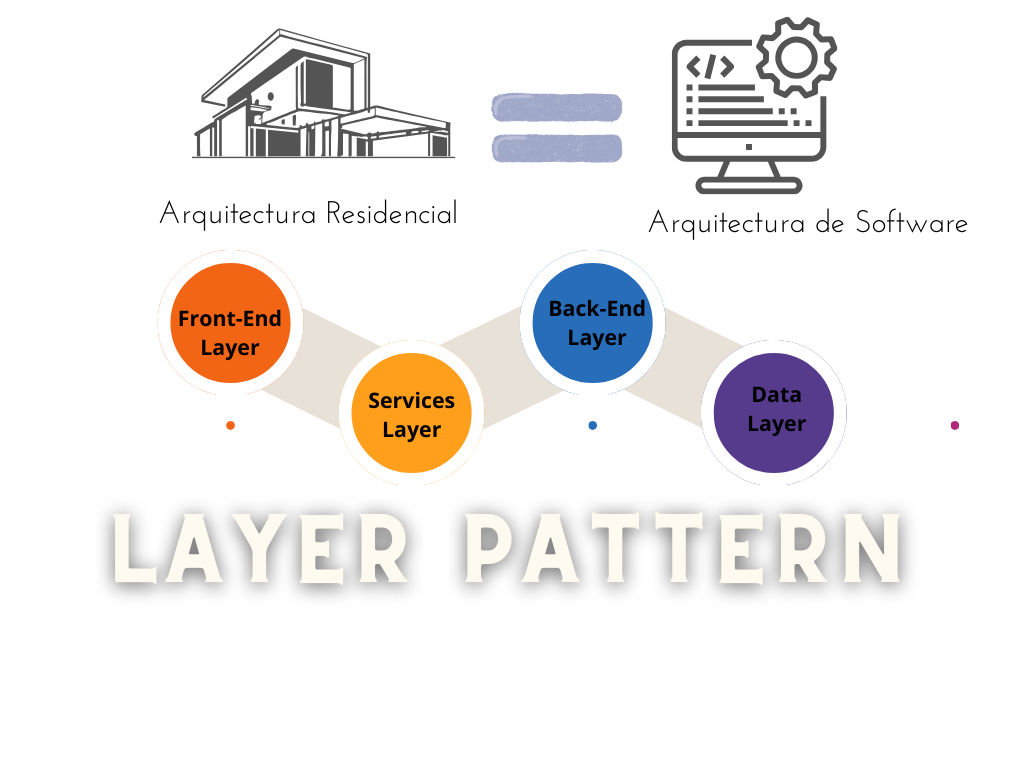
En resumen, la arquitectura MVC, que es un patrón de diseño probado y adoptado ampliamente, proporciona una estructura organizada y modular para el frontend. Esto permite una fácil gestión de datos, una presentación atractiva y una interacción intuitiva para los usuarios. Al adoptar este enfoque, se garantiza el desarrollo de un software robusto y adaptable que cumpla con las necesidades de los usuarios y clientes, al tiempo que facilita el mantenimiento y las futuras expansiones del sistema.

*3.2.2 BackEnd - Arquitectura en Capas*

La arquitectura de software es prácticamente la misma que la arquitectura residencial, ya que permite definir el diseño y la colección de componentes individuales que conforman el software completo.

Figura 3

*Arquitectura de Software*



*Nota.* Elaboración Propia

Los patrones de diseño son importantes para mejorar la eficiencia, facilitar la depuración, coordinar proyectos sin problemas y agregar nuevas funciones. El patrón de diseño en capas o n niveles, se asemeja a las estructuras organizativas y de comunicación de tecnologías de la información TI tradicionales, donde cada capa desempeña una función específica dentro de la aplicación, a continuación, se describen las cuatro capas estándares para el patrón de diseño de capas:

*Layer Pattern*

1. *Capa de presentación.* Se define como la capa de interface de usuario (IU) (Front-End Layer)
2. *Capa de Aplicación o Servicios.* Contiene la lógica de la aplicación; se manejan las solicitudes del usuario y se comunica con la capa lógica.
3. *Capa de lógica de negocio o Dominio.* Se implementan las operaciones y procesos clave que definen la funcionalidad central de la aplicación (Back-End Layer)
4. *Capa de acceso a Datos o Persistencia.* Su principal función es recuperar y almacenar datos, además de gestionar la conexión y las consultas a las fuentes de datos (Data Layer)

Para el presente proyecto de software “Aplicación web para promover el marketing digital mediante la publicación de proyectos de emprendimiento en la Universidad Popular del Cesar seccional Aguachica”, se implementa el patrón de diseño de capas en la parte del Back-End de la aplicación, viéndose cada una de las capas a implementar reflejadas en el siguiente diagrama.

Figura 4

*Patrón de capas para la aplicación web (Back-End)*

**

*Nota.* Elaboración Propia

*3.3 Tecnologías*

Para el desarrollo de la aplicación web, se ve la necesidad de definir a los usuarios involucrados en el proceso a realizar en el área de Emprendimiento de la Universidad Popular del Cesar seccional Aguachica, soportado en el de apoyo del área de emprendimiento. Este entorno de usuario es una comunidad diversa y dinámica que se extiende más allá de los confines de las aulas universitarias. En él, convergen una variedad de actores, cada uno con su conjunto único de intereses, aspiraciones y necesidades en el ámbito del marketing digital y el emprendimiento.

Bajo este contexto, se explora en detalle la conformación del entorno de usuario, destacando cómo la aplicación web se convertirá en un punto de encuentro y enriquecimiento mutuo para estos diversos grupos. Desde estudiantes ávidos de conocimientos hasta empresarios visionarios, desde docentes apasionados hasta inversores potenciales, todos se reúnen en esta plataforma con un objetivo común: aprovechar las oportunidades que el marketing digital y el emprendimiento ofrecen en la sociedad actual.

En las secciones que siguen, desglosamos en detalle quiénes son estos usuarios, cuáles son sus expectativas y cómo la aplicación web contribuirá a satisfacer sus necesidades y aspiraciones, promoviendo el aprendizaje continuo, la colaboración y la innovación en la Universidad Popular del Cesar, Seccional Aguachica.

La aplicación web se desarrollará adoptando un enfoque de arquitectura cliente-servidor (MVC), donde el backend se construirá usando Java con el framework Spring Boot para la lógica de negocio, mientras que el frontend se implementará mediante HTML, CSS y JavaScript, aprovechando la biblioteca React para crear una interfaz de usuario dinámica y eficiente. Esta combinación permitirá una separación clara de responsabilidades, facilitando el desarrollo colaborativo y la escalabilidad del proyecto.

*3.4 Entorno de desarrollo*

En toda creación de software, resulta imprescindible la selección de las herramientas apropiadas que se adecuen tanto a la metodología como a las tecnologías que conforman el desarrollo. Iniciando por aquellas utilidades que facilitarán una gestión eficiente de las etapas, tales como:

Trello, una herramienta eficaz para la ejecución de proyectos, en la cual podemos establecer las fases correspondientes a cada etapa de desarrollo, historias de usuario, actividades, gestión de tareas, objetivos, Sprint y demás procesos de la metodología SCRUM, al tiempo que se mantiene un control detallado de las iteraciones.

Otro recurso esencial en cualquier proyecto software es el sistema de control de versiones, razón por la cual se optó por la implementación de GitHub, una plataforma que permite llevar a cabo proyectos colaborativos que involucran diversas tecnologías, operando directamente con Git. De esta manera, se logra un control exhaustivo de las distintas versiones y ramificaciones que el proyecto requiere. Además, esta herramienta se adapta de manera sencilla a distintos entornos de desarrollo y entornos de desarrollo integrados (IDE) como Atom, Visual Studio y Visual Studio Code, Sublime Text, Netbeans, entre otros.

En lo que respecta al entorno de desarrollo integrado (IDE), se determinó que la elección más adecuada, tomando en consideración las características propias de la tecnología y las recomendaciones oficiales, es el empleo de Apache NetBeans en su edición Latest version of the IDE, released on September 1, 2023.

**4. Resultados**

*4.1 requisitos funcionales*

La arquitectura de desarrollo de la aplicación se definirá a partir de los requisitos funcionales del sistema, los cuales se presentan a continuación.

Tabla 1. Matriz de Requerimiento

|  |  |
| --- | --- |
| **Requerimiento** | Prioridad |
| El sistema debe permitir al Área de emprendimiento consultar la documentación suministrada por cada participante sobre su proyecto emprendedor. | 1 |
| El sistema debe permitir al Área de emprendimiento gestionar la escala Likert. Donde cada escala está representada de la siguiente manera: 1 es "Totalmente en desacuerdo", 2 es "En desacuerdo", 3 es "Algo de acuerdo", 4 es "De Acuerdo" y 5 es "Totalmente de acuerdo". | 4 |
| El sistema debe permitir al Área de emprendimiento gestionar la rúbrica de evaluación de proyectos. | 1 |
| El sistema debe permitir al Área de emprendimiento consultar una rúbrica o lista de chequeo predefinida para evaluar los proyectos de emprendimiento. | 1 |
| El sistema debe permitir al Área de emprendimiento poder asignar un estado a cada proyecto emprendedor. El estado puede ser "Aprobado", "No aprobado" o "En revisión". | 1 |
| El sistema debe permitir al Área de emprendimiento poder agregar comentarios a los proyectos emprendedores que se encuentren con estado "En revisión". | 3 |
| El sistema deberá permitir al Área de emprendimiento consultar aquellos inversionistas interesados en cada proyecto emprendedor. | 3 |
| El sistema debe permitir al Área de emprendimiento generar reportes consolidados sobre los proyectos evaluados. Los informes deben incluir estadísticas sobre proyectos aprobados, no aprobados o en revisión. | 3 |
| El sistema debe permitir al Área de emprendimiento la gestión de usuarios (crear, consultar, actualizar o deshabilitar). | 6 |
| El sistema debe permitir a los participantes registrarse mediante un formulario con campos de información básica como nombre, apellidos, correo, nombre de usuario, contraseña, tipo de identificación y tipo de usuario (estudiante o docente). | 5 |
| El sistema debe permitir a los participantes gestionar sus proyectos emprendedores (registrar, consultar, actualizar o deshabilitar). | 1 |
| El sistema debe permitir a los participantes consultar el estado de sus proyectos emprendedores. | 3 |
| El sistema debe permitir a los participantes consultar a los inversionistas interesados en cada uno de sus proyectos. | 4 |
| El sistema debe permitir a los participantes consultar indicadores de evaluación en cada uno de los proyectos asignados por los inversionistas. | 4 |
| El sistema debe permitir a los inversionistas registrarse a través de un formulario con campos de información básica como nombre, apellidos, correo, nombre de usuario, contraseña, tipo de identificación y tipo de usuario (inversionista). | 5 |
| El sistema debe permitir a los inversionistas consultar el listado de proyectos publicados. | 3 |
| El sistema debe permitir a los Inversionistas evaluar proyectos publicados mediante un indicador de evaluación definido mediante la escala Likert. | 4 |
| El sistema debe permitir a los Inversionistas seleccionar proyectos publicados a financiar. | 4 |

Con base en los requerimientos funcionales se define una arquitectura de software que combina el patrón Modelo-Vista-Controlador (MVC) en el FrontEnd y una arquitectura en Capas en el BackEnd, como se ha mencionado previamente.

*4.2 Implementación de MVC para el FrontEnd*

Modelo (Model): El modelo contendría las representaciones de los datos necesarios para los participantes, los inversionistas y los proyectos emprendedores. Se incluirían las estructuras de datos para usuarios, proyectos, estados, rúbricas, comentarios y escalas Likert.

Vista (View): Las vistas serían las interfaces de usuario que permiten a los usuarios interactuar con el sistema. Se crearían vistas para que los participantes registren y gestionen sus proyectos emprendedores, vean el estado de sus proyectos y consulten la información relacionada con los inversionistas. También se diseñarían vistas para que los inversionistas evalúen y seleccionen proyectos a financiar, y para que el Área de Emprendimiento genere informes consolidados.

Controlador (Controller): Los controladores gestionarían las acciones realizadas por los usuarios y comunicarían con el BackEnd. Se encargarían de procesar las solicitudes de los usuarios, validar datos, interactuar con el modelo para acceder o modificar información y mostrar las vistas adecuadas.

*4.3 Implementación de Arquitectura en Capas para el BackEnd*

Capa de Presentación: Esta capa sería responsable de gestionar las interacciones con el FrontEnd. Aquí se validarían las solicitudes de los usuarios y se llamaría a la capa de lógica de negocios para procesar la información.

Capa de Lógica de Negocios: En esta capa se implementaría la lógica de la aplicación, incluyendo la gestión de proyectos, rúbricas, estados, comentarios, escalas Likert y evaluaciones de inversionistas. Se aseguraría la coherencia y consistencia de los datos.

Capa de Acceso a Datos: La capa de acceso a datos se encargaría de interactuar con la base de datos para almacenar y recuperar información sobre usuarios, proyectos y evaluaciones. Se garantizaría la persistencia de los datos.

*4.4 Resultados esperados con la arquitectura propuesta*

Se estima que la arquitectura planteada permite separar el trabajo de desarrollo y mantener un diseño modular y escalable. El FrontEnd se basaría en el patrón MVC, mientras que el BackEnd seguiría una arquitectura en capas para asegurar un procesamiento eficiente y coherente de la información. Esto facilita el desarrollo, el mantenimiento y la escalabilidad de la aplicación web.

**5. Conclusiones**

La "Aplicación Web con Arquitectura Híbrida MVC y Modelo en Capas para Promover el Marketing Digital y la Publicación de Proyectos de Emprendimiento en la Universidad Popular del Cesar Seccional Aguachica" ofrece una solución integral para impulsar el emprendimiento y el marketing digital en el ámbito universitario. La arquitectura híbrida seleccionada combina de manera eficaz el patrón Modelo-Vista-Controlador (MVC) en el FrontEnd y una arquitectura en capas en el BackEnd, lo que proporciona modularidad, escalabilidad y una clara separación de responsabilidades.

Los requerimientos funcionales identificados se alinean con los objetivos de la aplicación, permitiendo la gestión eficiente de proyectos emprendedores, evaluaciones de inversionistas, generación de informes y más. Esto no solo beneficia a la universidad y su comunidad emprendedora, sino que también fomenta el desarrollo de habilidades digitales y la generación de ideas innovadoras. En cuanto al desarrollo de la aplicación la propuesta de la separación de capas facilitará el mantenimiento y el crecimiento futuro de la aplicación.

**Referencias**

Calderón G., H. (2023). *Diseño e implementación de una arquitectura de capas enfocada a microservicios en el contexto EHEALTH*. https://ebuah.uah.es/dspace/handle/10017/56890

Cartés, J. (2011). *Arquitectura de software híbrida basada en spring framework y ejb 3* [Escuela de Ingeniería Informática]. http://opac.pucv.cl/pucv\_txt/txt-5500/UCF5503\_01.pdf

Chugá C., M. F. & Hernández Q., R. F. (2023). *Tecnologías de la Información y Comunicación en el manejo del parque automotor*. http://repositorio.upec.edu.ec/handle/123456789/1741

Francia, J. (n.d.). *¿Qué es SCRUM?* https://www.scrum.org/resources/blog/que-es-scrum

Luna P., M. (2016). Modelo de recuperación de recursos educativos con tecnología libre e implementando una arquitectura en capas orientada a servicios. http://hdl.handle.net/11317/913

Ortega G., H. (2015). Estudio de factibilidad para la propuesta “framework de trabajo para proyectos de titulación aplicando metodologí a scrum en la ingeniería de software”.. http://repositorio.ug.edu.ec/handle/redug/11560

Rosales M., Evelyn L. (2023). *Aplicación web para la gestión de registro de pagos y control de recibos del Comité Seguro Campesino de la comuna-Cerezal Bellavista*. La Libertad. UPSE, Matriz. Facultad de Sistemas y Telecomunicaciones. 51p

Schwaber, K., & Sutherland, J. (2020a). La Guía Definitiva de Scrum: Las Reglas del Juego. *UMV*. https://repositorio.uvm.edu.ve/handle/123456789/59

Schwaber, K., & Sutherland, J. (2020b). *La Guía Definitiva de Scrum: Las Reglas del Juego*. https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Spanish-Latin-South-American.pdf

Vera P., J. A. (2023). *Desarrollo de un asistente virtual para fortalecer la gestión de devoluciones de repuestos en la Empresa Comercial Japonesa Automotriz Cía*. Tesis. Recuperado a partir de http://repositorio.ug.edu.ec/handle/redug/68525