



Facultad de Ciencias y Tecnologías Computacionales

Sistema para la gestión de contenido de la Preparación para la Defensa en la Educación Superior

Trabajo de diploma para optar por el título de
Ingeniero en Ciencias Informáticas

Autor: Dariel López Vera

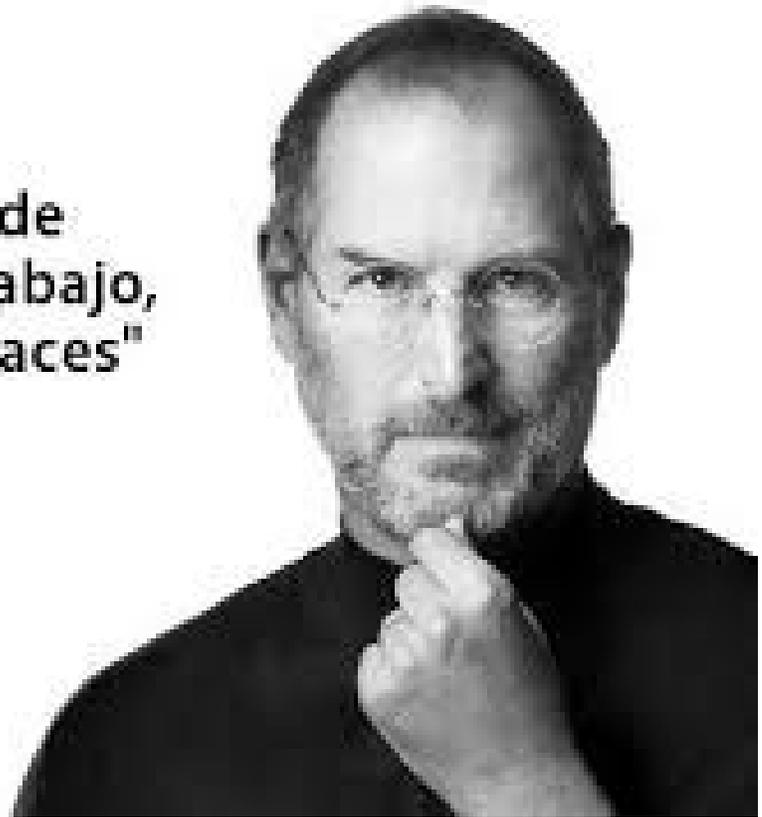
Tutor(es): Dr. C. Armando Leyva Pérez.
Ing. Luis Rubén Lima Mateo.

La Habana, septiembre de 2022

Año 63 de la Revolución

"La única forma de
hacer un gran trabajo,
es amar lo que haces"

—Steve Jobs



DECLARACIÓN DE AUTORÍA

El autor del trabajo de diploma con título “***Sistema para la gestión de contenido de la Preparación para la Defensa en la Educación Superior***” concede a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la investigación, con carácter exclusivo. De forma similar se declara como único autor de su contenido. Para que así conste firma la presente a los <día> días del mes de <mes> del año <año>.

<Dariel López Vera>

Firma del Autor

< Armando Leyva Pérez >

Firma del Tutor

< Luis Rubén Lima Mateo >

Firma del Tutor

DATOS DE CONTACTO

Dr.C Armando Leyva Pérez

Nombres y apellidos: Armando Leyva Pérez

Fecha de nacimiento: 15 de septiembre de 1960

Lugar de nacimiento: Contramaestre

Número de identidad permanente: 60091512145

Estado civil: casado

Militancia: PCC

Título de pregrado: Ingeniero Radiotécnico (1984); en el Instituto Técnico Militar “José Martí”

Categoría científica: Doctor en Ciencias Militares (2009)

Categoría docente: Profesor Auxiliar (2011)

Formación académica:

Especialista de Primer Grado de Mando y Estado Mayor Táctico de la Logística de Tropas Generales (2002); en la Academia de las FAR “General Máximo Gómez”

Especialista de Segundo Grado de Mando y Estado Mayor Táctico – Operativo Interarmas de Tropas Generales (2012); en la Academia de las FAR “General Máximo Gómez”

Cargos ocupados:

Jefe Sección Radiotécnica GCAA UM 1573 – 3710 (1984 – 1985)

Jefe Batería Radiotécnica GCAA UM 1573 – 3710 (1985 – 1986)

Jefe Batería Radiotécnica GCAA UM 4000 – 3710 (1986 – 1988)

Jefe Grupo Coheteril Antiaéreo – Asesor en misión internacionalista en la República Popular de Angola (1988 – 1989)

Jefe Plana Mayor GCAA UM 4000 – 3710 (1989 – 1993)

Jefe Centro Mando Combativo UM 4000 – 3710 (1993 – 1996)

Jefe de Estado Mayor, Segundo Jefe de Regimiento UM 4000 – 3710 (1996 – 1997)

Primer Oficial de Armamento Estado Mayor Provincial Holguín – Ejército Oriental (1997 – 2000)

Primer Oficial de Organización y Planificación de la Jefatura Logística Región Militar Holguín – Ejército Oriental (2000 – 2002)

Oficial de Organización y Planificación de la Jefatura Logística Ejército Oriental (2002 – 2004)

Primer Oficial de Organización y Planificación de la Jefatura Logística Ejército Oriental (2004 – 2009)

Profesor Cátedra Logística Academia de las FAR “General Máximo Gómez” (2009 - 2013)

Primer Profesor Cátedra Logística Academia de las FAR "General Máximo Gómez" (2013 - 2016)

Profesor Departamento Enseñanza Militar de la Universidad de las Ciencias Informáticas (2016 – 2018)

Jefe Departamento Enseñanza Militar de la Universidad de las Ciencias Informáticas (2018)

Cursos de posgrado recibidos:

1. Especialidad de Primer Grado de Mando y Estado Mayor Táctico de las Tropas Terrestres (2002)
2. Idioma Inglés (2006)
3. Problemas sociales de la Ciencia y la Tecnología (2006)
4. Problemas matemáticos aplicados a las Ciencias Militares (2007)
5. Arte Operativo (2007)
6. Pedagogía para nuevos profesores (2009)
7. Teoría y diseño curricular (2011)
8. El tutor en la formación de doctores (2011)
9. Curso básico de Idioma Ruso (2011)
10. Especialidad de Segundo Grado de Mando y Estado Mayor Táctico – Operativo Interarmas de Tropas Generales (2012)
11. Lucha radioelectrónica (2013)
12. Tecnologías avanzadas de la Informática y las Comunicaciones (2014)
13. Curso precongreso "Migraciones externas y jóvenes. Enfoque psicosocial" (2017)
14. La Tarea Vida o el Plan del estado para el enfrentamiento al cambio climático (2018)
15. Historia de la hegemonía imperial sobre América Latina (2018)

Publicaciones:

1. Trabajo teórico "Empleo de la minindustria como fuente de abastecimiento" (2009)
2. Trabajo teórico "Herramienta para determinar el despliegue de la minindustria como fuente de abastecimiento" (2009)
3. Trabajo teórico "Medidas para la vitalidad de la minindustria" (2009)
4. Trabajo teórico "Caracterización de la influencia de las condiciones de la situación en el empleo de la minindustria" (2009)
5. Tesis doctoral "La minindustria como fuente de abastecimiento a las tropas que defienden las regiones montañosas" (2009)

6. Artículo para la Revista Militar Cubana “Evolución histórica y estado actual del empleo de la minindustria como fuente de abastecimiento a las unidades que defienden las regiones montañosas” (2008)
7. Artículo para la Revista Militar Cubana “Algunas consideraciones sobre el empleo de la minindustria como fuente de abastecimiento a las unidades que defienden las regiones montañosas” (2009)
8. Artículo para la Revista Militar Cubana “Influencia de los factores físico – geográficos en la viabilidad del despliegue de la minindustria de montaña” (2009)
9. Artículo para la Revista Militar Cubana “Empleo de la minindustria como fuente de abastecimiento a las tropas en áreas urbanas” (2010)
10. Trabajo final de la especialidad de segundo grado de mando y estado mayor táctico – operativo interarmas (2012) “Aseguramiento logístico de las fuerzas y medios de la región militar en el despliegue operativo – combativo para la BDTCCDS”.
11. Trabajo teórico “Aseguramiento logístico de las fuerzas y medios de la región militar” (2013)
12. Trabajo teórico “Aseguramiento logístico del sector militar en el despliegue combativo” (2014)
13. Trabajo teórico “Aseguramiento logístico de la brigada de infantería en el despliegue combativo” (2014)
14. Artículo para la Revista Militar Cubana “Aseguramiento logístico de las tropas en el despliegue combativo” (2014)
15. Metodología para la creación y despliegue de la minindustria (2014)
16. Trabajo teórico “Aseguramiento logístico del sector militar” (2015)
17. Trabajo teórico “Aseguramiento logístico del batallón de infantería” (2015)
18. Trabajo teórico “Aseguramiento logístico en la zona de responsabilidad de la defensa antiaérea en la BDTCCDS” (2015)
19. Artículo para la Revista Militar Cubana “Influencia de las condiciones de la situación en el despliegue de los elementos tácticos de la logística” (2015)
20. Trabajo teórico “Aseguramiento logístico de la brigada de infantería” (2016)
21. Trabajo teórico “Aseguramiento logístico de la región militar” (2016)
22. Participación como ponente y jurado en los talleres científico - metodológicos de carácter nacional “Victoria 2008”; “Victoria 2009”; “Victoria 2010”; “Victoria 2011”; “Victoria 2012”; “Victoria 2013”; “Victoria 2014”; “Victoria 2015”.
23. La economía y la defensa nacional (2017)

24. El trabajo patriótico – militar e internacionalista en el proyecto “Marabana” de la Universidad de las Ciencias Informáticas.

Reconocimientos recibidos:

1. Medalla “Combatiente internacionalista de segunda clase” (1989)
2. Medalla “Por la victoria Cuba – RPA” (1989)
3. Distinción “Por el servicio en las FAR (10 años)” (1994)
4. Distinción “Por el servicio en las FAR (15 años)” (1994)
5. Distinción “Servicio Distinguido en las FAR” (1995)
6. Medalla conmemorativa “40 Aniversario de las FAR” (1996)
7. Distinción “Destacado en la preparación para la defensa” (1997)
8. Distinción “Servicio Distinguido en las FAR” (2002)
9. Distinción “Por el servicio en las FAR (20 años)” (2004)
10. Medalla conmemorativa “50 Aniversario de las FAR” (2006)
11. Condición 8 de octubre (2009)
12. Destacado en la actividad de Ciencia e Innovación Tecnológica en la Academia de las FAR (2009)
13. Mejor resultado de Ciencia e Innovación Tecnológica en la Academia de las FAR (2010)
14. Medalla “Ignacio Agramonte III Clase” (2010)
15. Talento de oro de dos estrellas (2011)
16. Mejor resultado de Ciencia e Innovación Tecnológica en las FAR (2012)
17. Medalla “Ignacio Agramonte II Clase” (2014)
18. Talento de oro de dos estrellas (2015)
19. Medalla conmemorativa “60 Aniversario de las FAR” (2016)

Ing. Luis Rubén Lima Mateo

AGRADECIMIENTOS

A mi familia en especial a mi madre por todo su apoyo y ayuda incondicional en todo momento sin importar las adversidades de la vida, por todo ese amor que solo una madre sabe dar, por ser la mejor madre del mundo. A mi padre por su apoyo en todo momento y quía.

A mi hermano por ser la persona que me insito a ser un profesional y a no darme por vencido.

A mi abuela, tíos y primos por su apoyo en todos estos años.

A mi cuñada y sobrino mi más reciente familia.

A mi novia Yinet que me dio su apoyo en todo momento.

Todos mis amigos: muchas gracias por todo lo vivido juntos, sus chistes, las fiestas en general, nunca los olvidare.

A Miquel Medina Ramírez por su ayuda y apoyo que me han ayudado a ser un mejor profesional.

Armando y Luis Rubén: muchas gracias por todo su apoyo incondicional, por esos momentos que compartimos juntos, por todos los consejos y lecciones dadas. Por sus enseñanzas quiero dedicarle estas frases: “Los grandes objetivos no se logran con dinero ni con talento, se logran con pasión, paciencia y perseverancia”, “ Si quieres llegar donde la mayoría no llega, debes hacer lo que la mayoría no hace” Profesores: muchas gracias por todas sus enseñanzas y lecciones, gracias a ustedes puedo ser el profesional que soy hoy.

DEDICATORIA

A mi madre Grisel, mi padre David, mi hermano Divisito, mi abuela Onelia, mis tíos Javier y Natacha, mi prima Merlin, mi novia Yinet, mis familiares y amigos y en especial a mi abuelo Carlos Manuel.

RESUMEN

Las universidades cubanas han sido baluartes de la defensa de la Revolución, y los estudiantes sus protagonistas fundamentales. El sistema de gestión de contenido de la Preparación para Defensa que se desea desarrollar posibilitará la gestión de contenido de la asignatura a nivel nacional. La investigación cuyos resultados se describen en este informe, tuvo como propósito desarrollar un sistema cubano para la gestión de contenido de la Preparación para la Defensa en respuesta a la escasa accesibilidad de sus contenidos. El empleo de las herramientas y tecnologías seleccionadas para la implementación de la solución propició la correspondencia entre los resultados obtenidos y los esperados. Las aplicaciones de diversas pruebas de software demostraron la efectividad de la solución implementada y su correspondencia con estándares internacionales de calidad, seguridad y usabilidad. El sistema para la gestión de contenido de la Preparación para la Defensa ofrece mayores niveles de racionalidad y eficiencia en los procesos de gestión de contenido, es soberana ante ambientes jurídicos de terceros países, incorpora los principios que rigen el proceso de informatización y refrenda la Constitución de la República.

Palabras Claves: Sistema, Preparación para la Defensa, Educación Superior, Gestión de Contenido, Accesibilidad

ABSTRACT

Cuban universities have been bastions in the defense of the Revolution, and students have been its fundamental protagonists. The Defense Preparation content management system that is to be developed will enable the content management of the subject at the national level. The research whose results are described in this report, had the purpose of developing a Cuban system for the content management of Defense Preparedness in response to the poor accessibility of its contents. The use of the tools and technologies selected for the implementation of the solution led to the correspondence between the results obtained and those expected. The applications of various software tests demonstrated the effectiveness of the implemented solution and its correspondence with international standards of quality, security and usability. The Defense Preparation content management system offers higher levels of rationality and efficiency in content management processes, is sovereign before legal environments of third countries, incorporates the principles that govern the computerization process and endorses the Constitution of the Republic.

Keywords: System, Defense Readiness, Higher Education, Content Management, Accessibility

TABLA DE CONTENIDOS

Contenido

DECLARACIÓN DE AUTORÍA.....	I
DATOS DE CONTACTO.....	II
AGRADECIMIENTOS.....	VI
DEDICATORIA.....	VII
RESUMEN.....	VIII
TABLA DE CONTENIDOS.....	IX
ÍNDICE DE TABLAS.....	XII
ÍNDICE DE FIGURAS.....	XIII
OPINIÓN DEL(OS) TUTOR(ES).....	14
AVAL DEL CLIENTE.....	15
INTRODUCCIÓN.....	16
CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA DE LA INVESTIGACIÓN.....	21
1.1 Conceptos asociados.....	21
1.1.1 Enseñanza militar.....	21
1.1.2 Universidad.....	21
1.1.3 Repositorio.....	21
1.1.4 Entornos Virtuales de Enseñanza y de Aprendizaje.....	21
1.2 Análisis de soluciones existentes.....	22
1.2.1 UdeMy.....	22
1.2.2 DigitalChalk.....	23
1.2.3 Chamilo.....	23
1.3 Metodología de desarrollo de software.....	24
1.3.1 Programación extrema o XP.....	25
1.4 Lenguajes.....	27
1.4.1 Lenguaje de modelado.....	27
1.4.2 Marco de trabajo.....	27
1.5 Herramientas utilizadas para el desarrollo del sistema.....	28
1.5.1 Herramienta de modelado.....	28
1.5.2 Herramientas de gestión de base de datos.....	30
1.5.3 Herramientas de Contenedores.....	31
1.6 IDE.....	34
1.6.1 Visual Studio Code.....	35

1.6.2 LiClipse.....	36
1.6.3 Selección del IDE.....	36
1.7 Conclusiones del capítulo.....	36
CAPÍTULO II: DISEÑO DE LA SOLUCIÓN PROPUESTA AL PROBLEMA CIENTÍFICO.....	38
2.1 Fase 1: Exploración.....	38
2.1.1 Modelo conceptual.....	38
2.1.2 Determinación de requisitos.....	39
2.1.3 Técnica de captura de requisitos.....	39
2.1.4 Selección de la técnica.....	40
2.1.5 Requisitos funcionales.....	40
2.1.6 Requisitos no funcionales.....	42
2.2 Fase 2: Planificación de las entregas.....	43
2.2.1 Historias de usuario.....	43
2.2.2 Estimación de tiempo por historia de usuario.....	44
2.2.3 Plan de iteraciones.....	46
2.2.4 Plan de entrega.....	49
2.2.5 Reuniones diarias de seguimiento.....	49
2.3 Fase 3: Etapa de diseño.....	50
2.3.1 Tarjetas CRC.....	51
2.3.2 Arquitectura Modelo-Template-Controlador.....	51
2.3.3 Prototipo de interfaz.....	53
2.4 Fase 4: Implementación.....	55
2.4.1 Estándares de codificación.....	55
2.4.2 Patrones de diseño.....	56
2.4.3 Modelo de datos.....	58
2.5 Conclusiones del capítulo.....	59
CAPÍTULO III: VALIDACIÓN DE LA SOLUCIÓN PROPUESTA.....	60
3.1 Diagrama de despliegue.....	60
3.2 Fase 5: Pruebas.....	60
3.2.1 Estrategia de pruebas.....	61
3.2.2 Pruebas unitarias.....	61
3.2.3 Pruebas de Caja Negra.....	63
3.2.4 Pruebas de Seguridad.....	64
3.2.5 Pruebas de aceptación.....	65
3.3 Resultados de las pruebas.....	66
3.4 Validación de las variables.....	67
3.5 Conclusiones del capítulo.....	68
CONCLUSIONES GENERALES.....	69
RECOMENDACIONES.....	70

REFERENCIAS BIBLIOGRÁFICAS.....	71
ANEXOS.....	73
ANEXO 1. GLOSARIO DE TÉRMINOS.....	73
ANEXO 2. ENTREVISTA.....	73

ÍNDICE DE TABLAS

- Tabla 1 Comparación sistemas homólogos
- Tabla 2 Comparación metodologías de software
- Tabla 3 Requisitos Funcionales Fuente: Elaboración Propia
- Tabla 4 Historia de usuario
- Tabla 5 Estimación de tiempo
- Tabla 6 Estimación del tiempo de HU
- Tabla 7 Plan de iteraciones
- Tabla 8 Plan de entrega
- Tabla 9 Reuniones de seguimiento
- Tabla 10 Tarjeta CRC
- Tabla 11 Juego de Datos Fuente: Elaboración Propia
- Tabla 12 Clases de equivalencia

- Tabla 13 Prueba de Aceptación Historia de Usuario Fuente: Elaboración Propia

- Tabla 14 Validación de variables

ÍNDICE DE FIGURAS

FIG 1 CICLO DE VIDA DE XP.(*IEL CICLO DE VIDA DE XP. - BÚSQUEDA DE GOOGLE, 2022*)...25

FIG 2 MODELO CONCEPTUAL FUENTE: ELABORACIÓN PROPIA.....39

FIG 3 MODELO VISTA PLANTILLA.....52

FIG 4 CAPA DE TEMPLATE.....53

FIG 5 CAPA DE MODELO.....53

FIG 6 CAPA DE VIEWS.....53

FIG 7 PROTOTIPO DE INTERFAZ DE PORTADA.....54

FIG 8 PROTOTIPO DE INTERFAZ DE SÍMBOLOS PATRIOS.....54

FIG 9 PROTOTIPO DE INTERFAZ DE AUTENTICACIÓN.....55

FIG 10 EJEMPLO DE PATRONES GRASP (CREADOR, BAJO ACOPLAMIENTO, ALTA COHESIÓN Y CONTROLADOR) FUENTE: ELABORACIÓN PROPIA.....57

FIG 11 EJEMPLO DE PATRONES GOF (DECORADOR)FUENTE: ELABORACIÓN PROPIA.....58

FIG 12 MODELO DE BASE DE DATOS FUENTE: ELABORACIÓN PROPIA.....59

FIG 13 DIAGRAMA DE DESPLIEGUE FUENTE: ELABORACIÓN PROPIA.....60

FIG 14 CAMINO BÁSICO FUENTE: ELABORACIÓN PROPIA.....62

FIG 15 GRÁFICO DE NO CONFORMIDADES FUENTE: ELABORACIÓN PROPIA.....67

OPINIÓN DEL(OS) TUTOR(ES)

El estudiante ha trabajado con responsabilidad, de forma independiente, con excelente calidad, en cada etapa del proceso investigativo ha demostrado conocimientos profundos de la especialidad y de la metodología de la investigación científica.

El resultado obtenido (Sistema para la gestión de contenido de la Preparación para la Defensa en la Educación Superior) garantiza la solución del problema planteado de cómo gestionar el contenido de la Preparación para la Defensa en la Educación Superior para mejorar su accesibilidad.

Este trabajo fue presentado en el Grupo de Tecnologías Educativas del Departamento Independiente de Enseñanza Militar del Ministerio de Educación Superior, en el Taller Nacional Científico – Metodológico de los Jefes de Departamentos de Enseñanza Militar de las Instituciones de Educación Superior 2022, con una excelente aceptación; además participó en el XXVIII Taller Regional Científico – Metodológico de Educación Patriótica – Militar e Internacionalista 2022 (con carácter nacional) donde obtuvo Premio Relevante.

Por todo lo anterior consideramos que el estudiante debe ser evaluado con calificación de Sobresaliente (5 puntos) y obtener el título de Ingeniero en Ciencias Informáticas.

Dr. C. Profesor Titular Armando Leyva Pérez

Ing. Profesor Instructor Luis Rubén Lima Mateo

AVAL DEL CLIENTE

El resultado obtenido (Sistema para la gestión de contenido de la Preparación para la Defensa en la Educación Superior) garantiza la solución del problema planteado de cómo gestionar el contenido de la Preparación para la Defensa en la Educación Superior para mejorar su accesibilidad.

Este trabajo fue presentado en el Grupo de Tecnologías Educativas del Departamento Independiente de Enseñanza Militar del Ministerio de Educación Superior, en el Taller Nacional Científico – Metodológico de los Jefes de Departamentos de Enseñanza Militar de las Instituciones de Educación Superior 2022, con una excelente aceptación; participó en el XXVIII Taller Regional Científico – Metodológico de Educación Patriótica – Militar e Internacionalista 2022 (con carácter nacional) donde obtuvo Premio Relevante; además está en proceso de montaje en la red de MES, por lo cual consideramos que cumple con nuestras expectativas y da solución al problema planteado.

Dr. C. Profesor Titular Reimundo Quesada Romero

Jefe del Departamento Independiente de Enseñanza Militar del Ministerio de Educación Superior

INTRODUCCIÓN

Las universidades cubanas han sido baluartes de la defensa de la Revolución, y los estudiantes sus protagonistas fundamentales. En 1975 se inició, como parte de la formación de los profesionales, la preparación militar. Con la creación del Ministerio de Educación Superior (MES) se extendió a todas las universidades y se impartió en concentrados militares, donde se formaron como oficiales decenas de miles de graduados hasta 1995. (Leyva et al., 2020)

Las condiciones económicas, la tensión a que fue sometido el país, los mandos militares y las propias universidades; las demandas y exigencias de formación del profesional, acorde a los nuevos escenarios, determinó que se emitiera la Directiva 29 del año 1995 del Ministro de las Fuerzas Armadas Revolucionarias, la cual estableció que la preparación para la defensa se impartiría por los propios profesores universitarios, mediante la disciplina Preparación para la Defensa (PPD) con un contenido teórico y práctico, vinculado con el perfil del profesional. (Leyva et al., 2020)

El Ministerio de Educación Superior no cuenta con recursos para poder impartir cursos online a nivel nacional por problemas económicos y por la situación actual de la pandemia y el bloqueo.

En la preparación del profesional cubano, la disciplina Preparación para la Defensa es una exigencia imprescindible para su futuro desempeño en las diferentes áreas del desarrollo económico, político y social, lo que se corresponde con la política económica y social del Partido y la Revolución, por su contribución en su formación revolucionaria, patriótica y el desarrollo de los valores que debe caracterizar a un graduado universitario.

La disciplina Preparación para la Defensa posibilita adquirir los conocimientos y herramientas para interpretar adecuadamente los riesgos, amenazas y agresiones a que está expuesto nuestro país; así como las vulnerabilidades de la sociedad cubana; las dimensiones y potenciales de la seguridad nacional; el sistema de seguridad y protección de la información oficial de la República de Cuba; el sistema de medidas de la Defensa Civil cubana y los problemas ambientales globales, el contenido de la agenda 2030 y las “acciones estratégicas” y “tareas” que conforman el “Plan del estado para el enfrentamiento al cambio climático (Tarea Vida)”.

Además, se aborda la concepción y las formas para defender el país y la Revolución; así como las ideas generales del derecho internacional humanitario (DIH), a través del papel, lugar y misión del egresado en las tareas de la defensa, bajo la idea expresada por el general de ejército Raúl Castro: *“En el escenario actual y previsible, conserva total vigencia la concepción estratégica de “guerra de todo el pueblo”, la cual se enriquece y perfecciona de modo constante”.* (Leyva et al., 2020)

Lo anteriormente planteado se complementa de manera extracurricular con las actividades de las milicias de tropas territoriales (MTT), la preparación y realización de los bastiones estudiantiles universitarios y la educación patriótica - militar e internacionalista (EPMI).

Las universidades del país no poseen los recursos para comunicar las actividades que realizan como son los bastiones; intercambios de información; actividades referentes a las asignaturas y competencias que pudieran realizarse entre los mismos estudiantes.

La disciplina articula un sistema de conocimientos, habilidades y valores, con un enfoque holístico que emana del aparato categorial y sus fundamentos; los cuales se articulan de manera coherente con el perfil profesional y su modo de actuación en los procesos de informatización de la sociedad cubana, facilitando vínculos intra, inter, multi y transdisciplinarios, que contribuyen a formar un profesional con una visión integral, creativo; gestor de su propio aprendizaje, con el valor agregado de defender el proyecto socialista desde su profesión

El surgimiento de Internet marcó un hito histórico en el desarrollo de la sociedad y la tecnología de la información. Desde entonces han ocurrido profundas transformaciones en el marco de la Educación. El desarrollo científico y técnico actual ha permitido automatizar los procesos de distribución y comercialización de contenido digital.

Nuestro país no cuenta con un sitio donde se publique documentos e información de las asignaturas vinculadas a la Preparación para la Defensa, lo cual ha debilitado la accesibilidad de los contenidos para sus estudiantes en las universidades y de personas jurídicas. Además, no cuenta con ninguna vía por la cual los profesores y estudiantes puedan intercambiar información ni consultar la misma de forma rápida y segura.

La conectividad dejará de ser un mito también en Cuba mediante el avance de la informatización de la sociedad, especialmente las telecomunicaciones están marcando el sendero del salto tecnológico, podemos afirmar que el 2020 puso a prueba los avances en materia de informatización alcanzados en Cuba. La pandemia aceleró de manera exponencial el uso de Internet y términos como *apk*, comercio electrónico y teletrabajo se enraizaron en el habla popular. El llamado era a quedarse en casa, para propiciar el aislamiento social, por lo que actividades presenciales migraron a las redes.

Como nota positiva, se anuncia que, al cierre del primer trimestre de 2021, el 100% de los municipios del país ya tienen cobertura 4G y también tienen esta posibilidad el 39% de los consejos populares. A tiempo que el 77% de los Consejos populares cuentan con al menos una solución de navegación ya sea por cobertura móvil, sitios Wifi o nauta Hogar. (Oscar Figueredo Reinaldo, 2021)

Actualmente, más del 68% de los cubanos accede a Internet, y la telefonía móvil tienen 6.6 millones de usuarios, crecen las redes 4G y el 76% de la población tiene cobertura a la señal de televisión digital.

El país cuenta con aplicaciones como Todus, Banca Móvil, Apklis, las cuales están marcando una nueva “era de la Internet en Cuba”.

El Ministerio de Educación Superior no está ajeno a este proceso y quiere explotar al máximo todos los potenciales anteriores. La información está dispersa en diferentes sitios web y pudiera agruparse de forma general en un solo sitio para la consulta de todos.

Puesto que nuestro país está en proceso de informatización, el Ministerio de Educación Superior desea implementar un sitio web en la cual se pueda gestionar los contenidos para la Preparación para la Defensa en la Educación Superior.

Luego de analizar la situación problemática se identifica el siguiente **problema a resolver**:

¿Cómo gestiona el contenido para la Preparación para la Defensa en la Educación Superior para mejorar la accesibilidad de sus contenidos?

El **objeto de estudio** es las herramientas para la gestión de contenidos, y el **campo de acción** es las herramientas de gestión de contenidos para la Preparación para la Defensa en la Educación.

Para darle solución al problema anterior se plantea como **objetivo general** desarrollar un sistema para la gestión de contenidos que contribuya a la accesibilidad del contenido de la Preparación para la Defensa en la Educación Superior.

Proponiendo como **idea a defender**: si se desarrolla un sistema para la gestión de contenido de la Preparación de la Defensa en la Educación Superior en Cuba se contribuirá a la accesibilidad de sus contenidos. Desglosándose en los siguientes **objetivos específicos**:

- Caracterizar los procesos de negocio y soluciones tecnológicas relacionadas con la gestión de los contenidos.
- Diseñar los artefactos e implementar la propuesta de solución para la gestión de los contenidos de la Preparación para la Defensa en la Educación Superior en Cuba.
- Validar la solución informática propuesta aplicando diferentes pruebas.

Para dar cumplimiento al objetivo planteado se proponen las siguientes **tareas de la investigación**:

- Análisis de los principales conceptos, contribuciones y tendencias sobre la gestión de contenido digital.

- Identificación de las tecnologías, herramientas y técnicas disponibles para la gestión de contenido digital mediante soluciones de software.
- Definición de los aspectos que distinguirán la propuesta de solución.
- Implementación de la solución informática.
- Evaluación técnica de la propuesta de solución.

El **resultado esperado** es obtener un Sistema para la gestión de contenido de la Preparación para la Defensa en la Educación Superior en Cuba, la cual contribuirá a la accesibilidad de sus contenidos.

Los Métodos teóricos utilizados para cumplir las tareas son los siguientes:

- Histórico-Lógico: Permitted realizar un estudio de los principales conceptos asociados a la distribución de contenido de la Preparación para la Defensa en la Universidades.
- Hipotético-deductivo: se utilizó para guiar la investigación desde el planteamiento del problema hasta la verificación de la solución a partir de las validaciones, orientando la secuencia lógica de las tareas que se realizaron.
- Análisis y Síntesis: Se utiliza para identificar y analizar las diversas funcionalidades de las plataformas a nivel internacional que pueden ser aplicadas en la solución.
- Modelación: Permitted la creación del modelo conceptual para entender el contexto en el que se enmarca la investigación.

Los **Métodos empíricos** utilizados para cumplir las tareas fueron:

- Entrevista: Permitted obtener la información necesaria relacionada con los problemas presentes en las universidades de nuestro país.

El presente documento se estructura en **tres capítulos**:

Capítulo 1. Fundamentación teórica de la Investigación. En este capítulo se definió algunos conceptos importantes para la posterior comprensión de la investigación. Se realizó una breve reseña y valoración sobre las soluciones existentes de los sistemas de distribución de contenido y se explican las herramientas, metodologías y lenguajes que serán utilizadas en la construcción de la solución.

Capítulo 2. Análisis y diseño de la plataforma para la gestión de la Preparación para la Defensa en la Educación Superior. En este capítulo se definió la solución que se propone y se realizó la selección de los requerimientos del sistema que se pretende implementar. Se hizo el análisis de la solución que se propone, se modelan y describen los recursos ingenieriles necesario acorde a la metodo-

logía XP que representan las funcionalidades del sistema, aplicando los patrones de arquitectura y diseño seleccionados.

Capítulo 3. Evaluación de la solución propuesta. En este capítulo se describió la implementación y posterior validación realizada al producto obtenido como solución.

CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA DE LA INVESTIGACIÓN

En este capítulo se expone la fundamentación teórica del trabajo, incluyéndose el estudio de otras soluciones informáticas existentes en la actualidad, de las metodologías, modelos de desarrollo de software y herramientas a utilizar.

1.1 Conceptos asociados

Los conceptos asociados son de vital importancia para una mejor comprensión del negocio, para ello se tuvo en cuenta los conceptos de Enseñanza militar, Universidad, repositorio y Entornos Virtuales de Enseñanza y de Aprendizaje.

1.1.1 Enseñanza militar

Actividad que se realiza en las Fuerzas Armadas y cuya finalidad es proporcionar a sus miembros la formación requerida para el ejercicio profesional en los diferentes cuerpos, escalas y especialidades, con objeto de atender las necesidades derivadas de la organización y preparación de las unidades y de su empleo en las operaciones. (RAE, 2022)

1.1.2 Universidad

Institución de enseñanza superior que comprende distintos centros (facultades, escuelas, institutos y otros) y que confiere los títulos académicos correspondientes. (RAE, 2022)

1.1.3 Repositorio

Es un espacio centralizado donde se almacena, organiza, mantiene y difunde información digital, habitualmente archivos informáticos, que pueden contener trabajos científicos, conjuntos de datos o software. (*Definición de repositorio — Definicion.de*, 2022)

1.1.4 Entornos Virtuales de Enseñanza y de Aprendizaje

Son plataformas informáticas cuyo propósito es orientar la comunicación pedagógica entre los participantes que intervienen en el proceso educativo y crear espacios o comunidades organizadas en torno al aprendizaje. (*Entornos Virtuales de Enseñanza y Aprendizaje (evea). Nivel 1 | Citep,2022.*)

Luego de analizar los principales conceptos se pudo entender con mayor profundidad el proceso de negocio que se quiere desarrollar en función de la problemática identificada y el contexto de uso.

1.2 Análisis de soluciones existentes

En el mundo, principalmente en Estados Unidos, América Latina y Europa, existen diferentes sistemas o herramientas que permiten dar solución a diversos problemas relacionados con la distribución de contenido.

El software libre es aquel que ofrece la libertad de utilizar, compartir, estudiar y modificar el código del programa. Al utilizar software libre, los recursos que se dejan de gastar en licencias pueden ser dedicados a personalizar la solución con total libertad y a integrarlo con otras herramientas informáticas. Esta adaptabilidad es una de las mayores ventajas del software libre frente al software privativo, cuyo código, cerrado y rígido por definición, no admite modificaciones significativas. (*Significado de Software libre*, 2022)

A pesar de las buenas intenciones de algunas fundaciones que promueven el software libre, el software propietario o comercial es el más recurrido debido al ingreso que este representa. («Software Propietario ¿Qué es el Software Propietario?», 2016)

El coste del software propietario tiende a ser considerablemente mayor que el de las soluciones de código abierto, por lo cual merece la pena intentar buscar opciones libres que además garanticen una gran calidad para los procesos de gestión documental por su capacidad multiplataforma y su sencillo manejo.

Las características tomadas en cuenta para el análisis fueron: tipo de software o código, de pago y navegadores.

A continuación, se expone el análisis de los sistemas homólogos a nivel internacional:

1.2.1 Udemy

Es una plataforma de aprendizaje en línea. Está dirigido a adultos profesionales. A diferencia de los programas académicos MOOC conducidos por tradicionales cursos de trabajo creados por las universidades, Udemy utiliza contenido de creadores en línea para vender y así conseguir ganancias. (*Online Education Platform Udemy Claims 10 Million Users | Time*, 2022)

Los instructores crean, poseen y gestionan cada curso de Udemy. La base de cada curso de Udemy son sus clases, que pueden incluir videos, diapositivas y texto. Además, pueden agregar recursos y diferentes tipos de actividades prácticas, como una manera de aprendizaje de los estudiantes.

Una de las grandes ventajas de aprender en Udemy es que puedes acceder a los cursos desde diferentes dispositivos y plataformas, incluidos ordenadores de mesa, portátiles o la aplicación móvil.

Actualmente, Udemy admite varios métodos de pago diferentes, en función del país y la ubicación de tu cuenta. (¿Cómo funciona Udemy?, 2022)

1.2.2 DigitalChalk

DigitalChalk es un completo software de gestión de aprendizaje en línea que proporciona un paquete de edición en línea llave en mano. Además, la plataforma totalmente integrada ofrece una solución de comercio electrónico completa que le ayuda a cobrar el pago de sus cursos. Además, ofrece funciones normativas y de cumplimiento, todas integradas en la aplicación. (*DigitalChalk: precios, funciones y opiniones | GetApp España, 2022*)

DigitalChalk cree en una revolución en la que cada empleado y alumno puede aprender y acceder fácilmente a los materiales educativos. Su sistema de gestión de aprendizaje (*Learning Management System, LMS*) permite a las organizaciones ofrecer contenido más significativo y atractivo. DigitalChalk proporciona una solución *LMS* centralizada que ofrece herramientas regulatorias y de cumplimiento integradas. Educa a toda la fuerza laboral o desarrolla soluciones de comercio electrónico; DigitalChalk ofrece un aprendizaje flexible y asequible para cualquier organización. (*DigitalChalk - Opiniones de usuarios - Pros y Contras, 2022*)

1.2.3 Chamilo

Chamilo es un campus virtual de código libre que se distribuye bajo licencia GNU/GPLv3, y que cualquier persona, institución o empresa puede usar libremente para la impartición de acciones formativas a través de internet.

Es interesante porque proporciona una interfaz muy sencilla para docentes y alumnos al tiempo que conserva una estructura dinámica para aquellos desarrolladores que quieren hacer modificaciones en el código. Cuenta con un gran número de herramientas encaminadas a facilitar el aprendizaje entre las que se encuentra la herramienta de creación de wikis, espacios para trabajo en grupo con recursos colaborativos grupales, blogs de aula con tareas asignables, foros puntuables, red social interna para el fomento del intercambio informal de conocimiento.

Esta plataforma permite a docentes poder crear y administrar el sitio web de un curso a través de un navegador de Internet (Mozilla Firefox, Internet Explorer, Safari, Opera, etc.). (*CHAMILO LMS: el e-learning ético - OpenExpo Europe, 2022*)

A continuación, se muestra una tabla comparativa de los sistemas detallados anteriormente, ver tabla 1:

Plataforma	Tipo de Software o Código	De Pago	Navegadores
Udemy	Privativo	Sí	Mozilla Firefox, Internet Explorer, Safari, Opera, etc.
	Libre	Sí	Mozilla Firefox, Internet Explorer, Opera, etc.
Chamilo	Libre	Sí	Mozilla Firefox, Internet Explorer, Safari, Opera, etc.

Tabla 1 Comparación sistemas homólogos Fuente: Elaboración propia

Luego de analizados los diferentes sistemas se llegó a la conclusión de que el sistema a desarrollar debe ser capaz de utilizarse en cualquier Navegador, las herramientas utilizadas para su desarrollo deberán ser de código abierto, lo cual cumple con lo planteado en la Constitución de la República en cuanto a la informatización del país. Además, debe ser capaz de gestionar contenidos, noticias y otras informaciones referentes a la Preparación de la Defensa en la Educación Superior y de libre costo.

1.3 Metodología de desarrollo de software

Piattini define a la metodología de desarrollo de software como “un conjunto de procedimientos, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevo software”. (Piattini, 2019)

La metodología indica cómo hay que obtener los diferentes productos parciales y finales.

Las metodologías se clasifican en 2 grupos: ágiles y tradicionales. A continuación, se muestran algunas características:(Series, 2018)

Metodologías ágiles	Metodologías tradicionales
Preparadas para cambios en el proyecto.	Cierta resistencia a los cambios.
No existe un contrato tradicional o al menos es bastante flexible.	Existe un contrato prefijado.
El cliente es parte del equipo de desarrollo.	El cliente interactúa con el equipo de desarrollo mediante reuniones.

Tabla 2 Comparación metodologías de software (Metodologías ágiles frente a las tradicionales en el proceso de desarrollo de software,2021)

Dadas las características de la aplicación, se decidió escoger una metodología ágil: el cliente es parte del proceso de desarrollo, la plataforma está sujeta a cambios y el equipo de trabajo es pequeño y con pocos roles.

1.3.1 Programación extrema o XP

Es una metodología centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. XP se basa en retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. (Metodologías ágiles frente a las tradicionales en el proceso de desarrollo de software,2022)

Véase en la Ilustración 1 el ciclo de vida de XP.

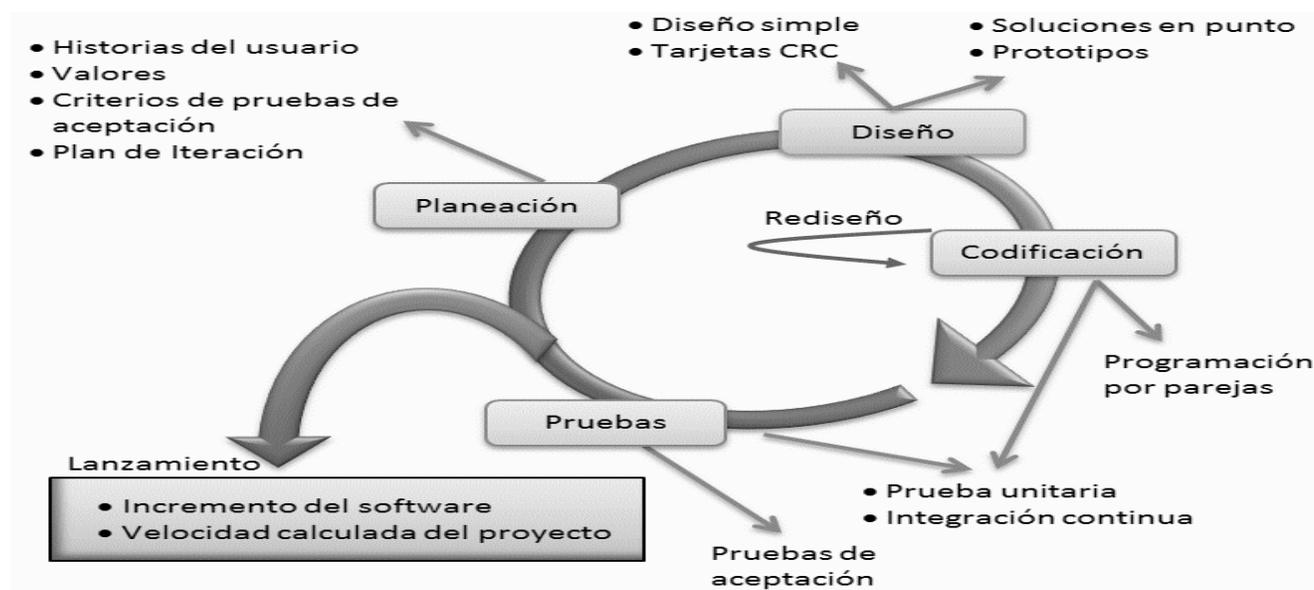


Fig 1 Ciclo de vida de XP.(el ciclo de vida de XP. - Búsqueda de Google, 2022)

El ciclo de vida ideal de XP consiste de seis fases:

Fase I: Exploración

En esta fase, los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. (*Ciclo de vida de un proyecto XP,2018*)

Fase II: Planificación de la Entrega

En esta fase el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses. Esta fase dura unos pocos días. (*Ciclo de vida de un proyecto XP*, 2018)

Fase III: Iteraciones

Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El Plan de Entrega está compuesto por iteraciones de no más de tres semanas. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Esto se logra escogiendo las historias que fueren la creación de esta arquitectura, sin embargo, esto no siempre es posible, ya que es el cliente quien decide qué historias se implementarán en cada iteración (para maximizar el valor de negocio). Al final de la última iteración el sistema estará listo para entrar en producción. (*Ciclo de vida de un proyecto XP*, 2018)

Fase IV: Producción

La fase de producción requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase.

Es posible que se rebaje el tiempo que toma cada iteración, de tres a una semana. Las ideas que han sido propuestas y las sugerencias son documentadas para su posterior implementación (por ejemplo, durante la fase de mantenimiento). (*Ciclo de vida de un proyecto XP*, 2018)

Fase V: Mantenimiento

Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente. De esta forma, la velocidad de desarrollo puede bajar después de la puesta del sistema en producción. La fase de mantenimiento puede requerir nuevo personal dentro del equipo y cambios en su estructura. (*Ciclo de vida de un proyecto XP*, 2018)

Fase VI: Muerte del Proyecto

Es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sis-

tema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo. (*Ciclo de vida de un proyecto XP*, 2018)

1.4 Lenguajes

1.4.1 Lenguaje de modelado

El lenguaje de modelado a utilizar será UML, el cual es un lenguaje diseñado para visualizar, especificar, construir y documentar software orientados a objeto, ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados. Se utiliza para especificar o para describir métodos o procesos, para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir.

1.4.2 Marco de trabajo

Django es un marco web Python de alto nivel que fomenta un desarrollo rápido y un diseño limpio y pragmático. Creado por desarrolladores experimentados, se ocupa de gran parte de las molestias del desarrollo web, por lo que puede concentrarse en escribir su aplicación sin necesidad de reinventar la rueda. Es gratis y de código abierto. (*Django overview | Django*, 2020)

A continuación, se muestran algunos de sus beneficios:

- **Rápido.**

Django fue diseñado para ayudar a los desarrolladores a llevar las aplicaciones desde el concepto hasta su finalización lo más rápido posible.

- **Cargado.**

Django incluye docenas de extras que puede usar para manejar tareas comunes de desarrollo web. Django se encarga de la autenticación de usuarios, la administración de contenido, los mapas del sitio, las fuentes RSS y muchas más tareas, desde el primer momento.

- **Seguro.**

Django se toma la seguridad muy en serio y ayuda a los desarrolladores a evitar muchos errores de seguridad comunes, como la inyección de SQL, las secuencias de comandos entre sitios, la falsificación de solicitudes entre sitios y el secuestro de clics. Su sistema de

autenticación de usuarios proporciona una forma segura de administrar las cuentas y contraseñas de los usuarios.

- **Escalable.**

Algunos de los sitios más concurridos de la web aprovechan la capacidad de Django para escalar de forma rápida y flexible.

- **Versátil.**

Empresas, organizaciones y gobiernos han utilizado Django para crear todo tipo de cosas, desde sistemas de gestión de contenido hasta redes sociales y plataformas informáticas científicas.

1.5 Herramientas utilizadas para el desarrollo del sistema

1.5.1 Herramienta de modelado

Las herramientas de modelado de sistemas informáticos, se emplean para la creación de modelos de sistemas que ya existen o que se desarrollarán, permitiendo crear un modelo del sistema a bajo costo y riesgo mínimo. (*Visual Paradigm User's Guides*, 2017)

Las buenas herramientas de modelado cumplen con las siguientes características:

- Permiten una visión descendente del sistema.
- Poseen componentes gráficos con algo de apoyo textual.
- El modelo resultante debe ser transparente (fácil de comprender).
- Poseen mínima redundancia (el aumento de redundancia, disminuye la transparencia del modelo y aumenta las tareas de mantenimiento).

La herramienta de modelado seleccionada fue Visual Paradigm para UML en su versión 8.0, debido a que es una herramienta para el desarrollo de aplicaciones utilizando modelado UML. Esta herramienta es ideal para ingenieros de software, analistas de sistemas y arquitectos de sistemas que están interesados en la construcción de sistemas a gran escala y necesitan confiabilidad y estabilidad en el desarrollo orientado a objetos. (*visualparadigm.com - visualparadigm Recursos e información.*, 2017)

Visual Paradigm ha sido concebido para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas y constituye una herramienta disponible en varias ediciones.

Se caracteriza por:

- Disponibilidad en múltiples plataformas (Windows, Linux).
- Diseño centrado en casos de uso.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Licencia: gratuita y comercial.
- Varios idiomas.
- Generación de código para Java y exportación como HTML.
- Diagramas de Procesos de Negocio - Proceso, Decisión, Actor de negocio, Documento.
- Generación de código - Modelo a código, diagrama a código.
- Editor de detalles de casos de uso - Entorno todo-en-uno para la especificación de los detalles de los casos de uso, incluyendo la especificación del modelo general y de las descripciones de los casos de uso.
- Soporte ORM - Generación de objetos Java desde bases de datos.
- Generación de bases de datos - Transformación de diagramas de Entidad-Relación en tablas de base de datos.
- Distribución automática de diagramas - Reorganización de las figuras y conectores de los diagramas UML.
- Editor de figuras.

La utilización de esta herramienta permite aumentar la calidad del software, a través de la mejora de la productividad en el desarrollo y mantenimiento del software, así como aumenta el conocimiento informático de una empresa ayudando así a la búsqueda de soluciones para los requisitos. También permite la reutilización del software, portabilidad y estandarización de la documentación, además del uso de las distintas metodologías propias de la Ingeniería de Software.

1.5.2 Herramientas de gestión de base de datos

Un sistema de gestor de bases de datos (DBMS), algunas veces llamada simplemente un administrador de base de datos, es un programa que permite a uno o más usuarios de computadoras crear y acceder a los datos en una base de datos. El DBMS gestiona las peticiones del usuario para que estos sean libres de entender que los datos se encuentran físicamente en los medios de almacenamiento. En el manejo de solicitudes de los usuarios, el DBMS asegura la integridad de los datos y seguridad (asegurándose de que sólo aquellos con privilegios de acceso pueden acceder a los datos). El más típico DBMS es un sistema de base de datos relacional. La selección adecuada del DBMS proveerá a la solución informática que se desea desarrollar numerosas herramientas, además de seguridad a los datos que se almacenan y mejorará considerablemente el desempeño de la aplicación final. (*Oracle Corporation.*, 2018)

La herramienta de Gestor de Base de Datos seleccionada es MySQL. Es el más popular sistema de gestión de base de datos multiplataforma de código abierto. Se caracteriza fundamentalmente por su velocidad, rendimiento y fiabilidad, motivo por el cual es ampliamente utilizado en sitios web y aplicaciones de escritorio de baja y mediana complejidad.

Además, cumple con las siguientes características:

- MySQL es de código abierto.
- Velocidad al realizar las operaciones, lo que le hace uno de los gestores con mejor rendimiento.
- Bajo costo en requerimientos para la elaboración de bases de datos, ya que debido a su bajo consumo puede ser ejecutado en una máquina con escasos recursos sin ningún problema.
- Facilidad de configuración e instalación.
- Soporta gran variedad de Sistemas Operativos.
- Baja probabilidad de corromper datos, incluso si los errores no se producen en el propio gestor, sino en el sistema en el que se encuentra integrado.
- Ofrece un sistema de contraseñas y privilegios seguros de verificación basada en el host y tráfico de contraseñas encriptado al conectarse a un servidor.
- Soporta gran cantidad de datos, incluso con más de 50 millones de registros.

1.5.3 Herramientas de Contenedores

El empaquetado de aplicaciones es el proceso que utilizan las empresas y organizaciones para obtener el software necesario en los dispositivos de los usuarios relevantes con facilidad. Implica la creación de un paquete de archivos para cada tipo o pieza de software que utiliza una empresa. Los beneficios de este método son enormes e implican ahorros de costos y una mayor eficiencia para la organización. Les brinda a los departamentos de TI la capacidad de ofrecer acceso al software (incluidas las últimas versiones) de una manera más fácil y rápida. (*Empaquetado de la aplicación: qué es y por qué es importante*, 2022).

1.5.3.1 Docker

Docker es un conjunto de herramientas para configurar, ejecutar y administrar contenedores. La herramienta de línea de comandos principal, Docker, se puede usar para configurar e iniciar rápidamente contenedores usando imágenes preconstruidas. La suite también incluye herramientas como docker-compose, que se utiliza para iniciar y detener rápidamente una configuración específica de varios contenedores.

Software Container tampoco es un concepto nuevo, pero trabajar en él es un trabajo difícil de bajo nivel para la mayoría de los ingenieros. Docker, sin embargo, resulta ser una herramienta rápida, fácil de usar y poderosa para la contenedorización de su software. Con Docker, se definen imágenes que son descripciones de la configuración y los comandos de un entorno de software. Y a partir de esas imágenes, se puede ejecutar contenedores que son el paquete ejecutable real.

Docker es un tipo especial de virtualización. La ventaja sobre VMWare, Hyper-V, KVM o Xen es que Docker usa de forma nativa el kernel del sistema operativo y no tiene ningún hipervisor en el medio. Eso hace que Docker sea muy rápido, potente y eficaz. (*Contenedores de Docker | ¿Qué es Docker?*, 2022).

1.5.3.2 Beneficios de Docker:

La ejecución de aplicaciones en contenedores en lugar de máquinas virtuales está cobrando impulso rápidamente en el mundo de TIC. Se considera que la tecnología es una de las de más rápido crecimiento en la historia reciente gracias a que fue adoptada por nombres clave en la industria junto con muchos proveedores de software. En el corazón de este ecosistema se encuentra Docker, una plataforma que permite a los usuarios empaquetar, distribuir y administrar aplicaciones Linux dentro de contenedores. La empresa y su software han crecido enormemente desde el momento en que penetraron en el mercado, y ahora buscan invertir para mejorar la usabilidad de la tecnología de contenedores. (apiumhub, 2022).

- **Retorno de la inversión y ahorro de costos**

El mayor impulso de la mayoría de las decisiones de gestión al seleccionar un nuevo producto es el retorno de la inversión. Cuanto más una solución pueda reducir los costos y, al mismo tiempo, aumentar las ganancias, mejor será una solución, especialmente para las grandes compañías establecidas que necesitan generar ingresos estables a largo plazo. En este sentido, Docker puede ayudar a facilitar este tipo de ahorro al reducir drásticamente los recursos de infraestructura. La naturaleza de Docker es que se necesitan menos recursos para ejecutar la misma aplicación. Debido a los requisitos de infraestructura reducidos que tiene Docker, las organizaciones pueden ahorrar en todo, desde los costos del servidor hasta los empleados necesarios para mantenerlos. Docker permite que los equipos de ingeniería sean más pequeños y más efectivos. (apiumhub, 2022)

- **Estandarización y productividad**

Los contenedores Docker garantizan la coherencia en múltiples ciclos de desarrollo y liberación, estandarizando su entorno. Una de las mayores ventajas de una arquitectura basada en Docker es en realidad la estandarización. Docker proporciona entornos repetibles de desarrollo, construcción, prueba y producción. La estandarización de la infraestructura de servicio en todo el proceso permite que cada miembro del equipo trabaje en un entorno de paridad de producción. Al hacer esto, los ingenieros están más equipados para analizar y corregir errores de manera eficiente dentro de la aplicación. Esto reduce la cantidad de tiempo desperdiciado en defectos y aumenta la cantidad de tiempo disponible para el desarrollo de características.

Como mencionamos, los contenedores Docker le permiten realizar cambios en sus imágenes Docker y la versión los controla. Por ejemplo, si realiza una actualización de componente que rompe todo su entorno, es muy fácil retroceder a una versión anterior de su imagen Docker. Todo este proceso puede probarse en unos minutos. Docker es rápido, lo que le permite realizar repeticiones rápidamente y lograr redundancia. Además, el lanzamiento de imágenes Docker es tan rápido como ejecutar un proceso de máquina. (apiumhub,2022)

- **Eficiencia de CI**

Docker permite construir una imagen de contenedor y usar esa misma imagen en cada paso del proceso de implementación. Una gran ventaja de esto es la capacidad de separar los pasos no dependientes y ejecutarlos en paralelo. El tiempo que lleva desde la construcción hasta la producción puede acelerarse notablemente. (apiumhub, 2022)

- **Compatibilidad y mantenibilidad**

Uno de los beneficios que todo el equipo apreciará es la paridad. La paridad, en términos de Docker, significa que las imágenes se ejecutan igual sin importar en qué servidor o en qué computadora portátil se ejecutan. Para los desarrolladores, esto significa menos tiempo dedicado a configurar entornos, depurar problemas específicos del entorno y una base de código más portátil y fácil de configurar. La paridad también significa que tu infraestructura de producción será más confiable y más fácil de mantener. (apiumhub, 2022)

- **Simplicidad y configuraciones más rápidas**

Uno de los beneficios clave de Docker es la forma en que simplifica las cosas. Los usuarios pueden tomar su propia configuración, ponerla en el código y desplegarla sin ningún problema. Como Docker se puede utilizar en una amplia variedad de entornos, los requisitos de la infraestructura ya no están vinculados con el entorno de la aplicación. (apiumhub, 2022)

- **Despliegue rápido**

Docker logra reducir la implementación a segundos. Esto se debe al hecho de que crea un contenedor para cada proceso y no arranca un sistema operativo. Los datos se pueden crear y destruir sin preocuparse de que el costo de volver a hacerlo sea más alto que asequible. (apiumhub, 2022)

- **Despliegue continuo y pruebas**

Si necesita realizar una actualización durante el ciclo de lanzamiento de un producto, puede realizar fácilmente los cambios necesarios en los contenedores Docker, probarlos e implementar los mismos cambios en sus contenedores existentes. Este tipo de flexibilidad es otra ventaja clave del uso de Docker. Docker realmente le permite construir, probar y lanzar imágenes que pueden implementarse en múltiples servidores. Incluso si hay un nuevo parche de seguridad disponible, el proceso sigue siendo el mismo. Puede aplicar el parche, probarlo y lanzarlo a la producción. (apiumhub, 2022)

- **Plataformas multi-nube**

Uno de los mayores beneficios de Docker es la portabilidad. En los últimos años, todos los principales proveedores de computación en nube, incluidos Amazon Web Services (AWS) y Google Compute Platform (GCP), han adoptado la disponibilidad de Docker y han agregado soporte individual. Los contenedores acoplables se pueden ejecutar dentro de una instancia de Amazon EC2, instancia de Google Compute Engine, servidor de Rackspace o VirtualBox, siempre que el sistema operativo host sea compatible con Docker. Si este es el caso, un contenedor que se ejecuta en una instancia de Amazon EC2 se puede portar fácilmente entre entornos, por ejemplo, a VirtualBox, logrando una co-

herencia y funcionalidad similares. Además, Docker funciona muy bien con otros proveedores como Microsoft Azure y OpenStack, y se puede usar con varios administradores de configuración como Chef, Puppet y Ansible, etc. (apiumhub, 2022)

- **Aislamiento**

Docker garantiza que sus aplicaciones y recursos estén aislados y segregados. Docker se asegura de que cada contenedor tenga sus propios recursos que están aislados de otros contenedores. Puede tener varios contenedores para aplicaciones separadas que ejecutan pilas completamente diferentes. Docker te ayuda a garantizar la eliminación de aplicaciones limpias ya que cada aplicación se ejecuta en su propio contenedor. Si ya no necesita una aplicación, simplemente puede eliminar su contenedor. No dejará ningún archivo temporal o de configuración en su sistema operativo anfitrión.

Además de estos beneficios, utilizar Docker también asegura que cada aplicación solo use los recursos que se les han asignado. Una aplicación en particular no usará todos sus recursos disponibles, lo que normalmente llevaría a la degradación del rendimiento o al tiempo de inactividad completo para otras aplicaciones. (apiumhub, 2022)

- **Seguridad**

Desde un punto de vista de seguridad, utilizar Docker garantiza que las aplicaciones que se ejecutan en contenedores estén completamente segregadas y aisladas entre sí, lo que le otorga un control total sobre el flujo y la administración del tráfico. Ningún contenedor Docker puede ver los procesos que se ejecutan dentro de otro contenedor. Desde un punto de vista arquitectónico, cada contenedor obtiene su propio conjunto de recursos que van desde el procesamiento hasta las pilas de la red. (apiumhub, 2022)

1.6 IDE

IDE es el acrónimo del término inglés *Integrated Development Environment* o, lo que es lo mismo, Entorno de Desarrollo Integrado. Es el escenario digital utilizado en programación para desarrollar aplicaciones, juegos... Es imprescindible tanto en el ámbito del Desarrollo de Aplicaciones Web (DAW) como en el Desarrollo de Aplicaciones Multiplataforma (DAM). Hace que la tarea del programador sea más sencilla, gracias a las herramientas que tiene incorporadas, como compiladores, depuradores o bibliotecas, y esto se traduce en un aumento de la productividad. (*¿Qué es un IDE en programación?*, 2019)

1.6.1 Visual Studio Code

Visual Studio Code es un IDE de desarrollo muy maduro que ha existido durante mucho tiempo. Es un editor de código fuente ligero pero potente que se ejecuta en su escritorio y está disponible para Windows, macOS y Linux. Viene con soporte incorporado para JavaScript, TypeScript y Node.js y tiene un rico ecosistema de extensiones para otros lenguajes (como C++, C#, Java, Python, PHP, Go) y tiempos de ejecución (como .NET y Unity).

En primer lugar, es un editor que se quita de en medio el ciclo de edición-construcción-depuración de-liciosamente fluido significa menos tiempo jugando con su entorno y más tiempo ejecutando sus ideas. (*Documentation for Visual Studio Code*, 2022)

Funciones de Visual Studio:

- Desarrolle: navegue, escriba y corrija su código rápidamente.
- Depurar: depurar, perfilar y diagnosticar con facilidad.
- Prueba: escriba código de alta calidad con herramientas de prueba completas.
- Colaborar: utilice herramientas de control de versiones como Perforce, Git. Sea ágil y colabore de manera eficiente.
- Ampliar: elija entre miles de extensiones para personalizar su IDE.
- Soporte: soporte empresarial y una gran comunidad.
- Windows: desarrolle aplicaciones y juegos para cualquier dispositivo Windows.
- Aplicaciones móviles: cree aplicaciones nativas o híbridas para Android, iOS y Windows.
- Aplicaciones de Azure: cree, administre e implemente aplicaciones de escala de nube en Azure.
- Aplicaciones web: desarrolle aplicaciones web modernas y aplicaciones web progresivas (PWA).
- Office: utilice potentes herramientas para el desarrollo de Office.
- Juegos: diseñe, codifique y depure juegos con gráficos de vanguardia.
- Extensiones: escriba sus propias extensiones de Visual Studio.
- Base de datos: desarrolle e implemente bases de datos SQL Server y Azure SQL.

1.6.2 LiClipse

LiClipse es un conjunto de complementos para mejorar Eclipse y mejorar la experiencia general de Eclipse. Incluye temas IDE mejorados, editores integrados para muchos idiomas (y una forma de crear fácilmente nuevos editores sin profundizar en la codificación Java real), mejoras de usabilidad para todos los editores de Eclipse, paquetes de algunos complementos e instaladores comunes existentes que se integran de forma nativa en cada plataforma. (*LiClipse by brainwy*, 2022)

Con él, los usuarios salen de la caja:

- Un editor rápido que admite muchos idiomas desde el primer momento.
- Compatibilidad con paquetes de TextMate.
- Una forma sencilla de agregar soporte para un nuevo idioma
- Mejoras de usabilidad para todos los editores de Eclipse que incluyen:
 - Múltiples cursores
 - Guías de sangría vertical
 - Barras de desplazamiento temáticas
 - Capacidades de búsqueda de texto mejoradas (con búsqueda basada en índices de Lucene, soporte para carpetas externas, editores abiertos y filtrado adicional en la página de resultados)
- Vista previa de HTML para los editores RST, Markdown y HTML
- Instaladores nativos
- Soporte mejorado de temas basado en las mejoras de Eclipse 4

1.6.3 Selección del IDE

Para el desarrollo del sistema se decidió utilizar el Visual Studio Code ya que el editor entiende profundamente su estructura, apoya a todas las características del lenguaje Python para proyectos modernos y antiguos. Combina la simplicidad de un editor de código fuente con potentes herramientas para desarrolladores, como la finalización y depuración de código de IntelliSense. Los tipos de proyectos que admite Visual Studio Code y las plantillas disponibles lo hacen muy atractivo como IDE para equipos grandes y pequeños. Además, proporciona varias ediciones, según sus necesidades y presupuesto, para que los equipos pequeños puedan acceder a la edición comunitaria gratuita.

1.7 Conclusiones del capítulo

Los resultados expuestos en este capítulo permitieron:

- El análisis de los conceptos fundamentales asociados al proceso de gestión de contenido para la preparación para la defensa permitió identificar la accesibilidad como aspecto fundamental en las soluciones informáticas para la promoción y visualización de contenido.
- El estudio comparativo de soluciones similares para la distribución de contenido resaltó la normalización, organización, clasificación, segmentación, disponibilidad e interacción de los contenidos como características más comunes, mientras que el uso de pagos por cursos destaca los fines de lucro y expansión de los organismos a nivel global y, por tanto, sus principales limitaciones para el desarrollo.
- El análisis de las características que distinguen a los sistemas de gestión de contenido permitió reorientarlas a las Universidades y las premisas del proceso de informatización de la sociedad cubana.

- La identificación de las características propuestas para la solución, el análisis de las tecnologías que distinguen soluciones para la gestión de contenido, contribuyó a la selección de las herramientas y tecnologías más adecuadas para desarrollar la propuesta de solución, destacando como metodología de desarrollo XP, como marco de trabajo Django, como herramienta de modelado Visual Paradigm usando como lenguaje UML y herramienta de empaquetado Docker Compose.

CAPÍTULO II: DISEÑO DE LA SOLUCIÓN PROPUESTA AL PROBLEMA CIENTÍFICO

En este capítulo se aborda la propuesta de solución a la problemática planteada. Se describen los requisitos funcionales y no funcionales, para dar cumplimiento a los objetivos planteados. Se realizan las historias de usuario, plan de iteraciones, tarjetas CRC y demás artefactos exigidos por la Metodología XP y sus fases.

2.1 Fase 1: Exploración

Se cumple con lo establecido en la **Fase 1** de la Metodología XP, donde el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.

2.1.1 Modelo conceptual

Un modelo conceptual es una descripción general de cómo un sistema se organiza y opera. Su tarea principal es especificar y describir los conceptos base. Este es un paso muy importante en cualquier metodología de análisis y diseño orientada a objetos, pues de aquí parte el diagrama de clases que modela la aplicación. Los errores que se cometan en la modelación de este diagrama pueden dar al traste con una aplicación incompleta o que no cumpla las expectativas de los clientes. («Modelos conceptuales para definir las plataformas digitales | IDA Chile», 2016)

El sistema cuenta con un perfil para cada rol del sistema (administrador, usuario), los cuales pueden interactuar con las funcionalidades del mismo acorde a su rol, las principales funcionalidades están enmarcadas, en la publicación de contenido y visualización de contenido.

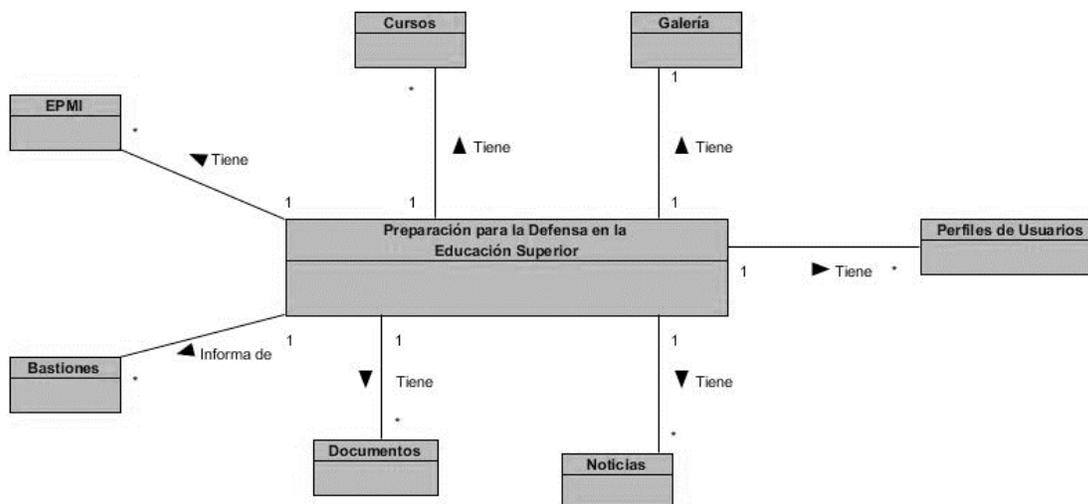


Fig 2 Modelo conceptual Fuente: Elaboración Propia

2.1.2 Determinación de requisitos

El análisis de requisitos es una de las tareas más importantes en el ciclo de vida del desarrollo de software, puesto que en ella se determinan los “planos” de la nueva aplicación. El análisis de requisitos se puede definir como el proceso del estudio de las necesidades de los usuarios para llegar a una definición de los requisitos del sistema, hardware o software, así como el proceso de estudio y refinamiento de dichos requisitos.

2.1.3 Técnica de captura de requisitos

En el proceso de desarrollo de software los analistas emplean varias técnicas para obtener los requisitos del cliente. Históricamente, esto ha incluido técnicas tales como las entrevistas, o talleres con grupos para crear listas de requisitos. Cuando sea necesario, el analista empleará una combinación de estos métodos para establecer los requisitos exactos de las personas implicadas, para producir un sistema que resuelva las necesidades del cliente. A continuación, se muestran algunas de estas técnicas:

2.1.3.1 Entrevista

Las entrevistas son un método común. Por lo general no se entrevista a toda la gente que se relacionará con el sistema, sino a una selección de personas que represente a todos los sectores críticos de la organización, con el énfasis puesto en los sectores más afectados o que harán un uso más frecuente del nuevo sistema.

2.1.3.2 Lluvia de ideas

Todos los participantes pueden aportar distintas ideas en un ambiente libre de prejuicios. Ningún participante debe juzgar negativamente la propuesta de otros, sino que se anotan todas las ideas en una pizarra y serán evaluadas al final de la sesión. El principio básico es no descartar de manera apresurada ningún planteo, de modo que existe la posibilidad de que surjan otras ideas derivadas de la idea original y se generan varios puntos de vista del problema.

2.1.3.3 Diseño de aplicación conjunta

Se trabaja directamente sobre los documentos a generar, las temáticas que se tratan durante las reuniones siguen un esquema y se busca que la misma sea ordenada y racional. Se define una agenda con los puntos principales a tratar durante la jornada. Este tipo de taller tiene el inconveniente de que es muy difícil poder reunir a todos los participantes, es costoso y generalmente es necesaria más de una reunión para establecer los requisitos del sistema. (Mesias Torres, 2018)

2.1.3.4 Forma de contrato

En lugar de una entrevista, se pueden llenar formularios o contratos indicando los requisitos. En sistemas muy complejos éstos pueden tener centenares de páginas. (Mesias Torres, 2018)

2.1.3.5 Objetivos medibles

Los requisitos formulados por los usuarios se toman como objetivos generales, a largo plazo, y en cambio se los debe analizar una y otra vez desde el punto de vista del sistema hasta determinar los objetivos críticos del funcionamiento interno que luego darán forma a los comportamientos apreciables por el usuario. Luego, se establecen formas de medir el progreso en la construcción, para evaluar en cualquier momento qué tan avanzado se encuentra el proyecto. (Mesias Torres, 2018)

2.1.4 Selección de la técnica

Luego de analizadas las posibles técnicas a utilizar se decidió emplear la entrevista y las lluvias de ideas, de conjunto con los objetivos medibles debido a la dinámica de desarrollo del software y la constante interacción del equipo con el Ministerio de Educación Superior.

2.1.5 Requisitos funcionales

Los requisitos funcionales describen la interacción entre el sistema y su ambiente independientemente de su implementación. El ambiente incluye al usuario y cualquier otro sistema externo que interactúa con el sistema. Los requisitos definidos por el cliente se listan a continuación:

Los requisitos funcionales del sistema son los siguientes:

Sistema para la gestión de contenido de la Preparación para la Defensa en la Educación Superior

No.	Requisito Funcional
RF_1	Crear Usuario
RF_2	Autenticar Usuario
RF_3	Modificar Usuario
RF_4	Eliminar Usuario
RF_5	Listar/Mostrar Usuario
RF_6	Buscar usuario
RF_7	Eliminar contenido
RF_8	Modificar contenido
RF_9	Buscar contenido
RF_10	Listar/Mostrar contenido
RF_11	Agregar contenido
RF_12	Agregar noticia
RF_13	Eliminar noticia
RF_14	Modificar noticia
RF_15	Buscar noticia
RF_16	Mostrar/Listar noticia
RF_17	Informar Bastión
RF_18	Modificar Información del Bastión
RF_19	Eliminar Información del Bastión
RF_20	Divulgar Información de posibles Cursos

RF_21	Modificar Información de posibles Cursos
RF_22	Eliminar Información de posibles Cursos
RF_23	Listar Información de posibles Cursos

Tabla 3 Requisitos Funcionales Fuente: Elaboración Propia

2.1.6 Requisitos no funcionales

Los requisitos no funcionales describen aspectos del sistema que son visibles por el usuario que no incluyen una relación directa con el comportamiento funcional del sistema. Los requerimientos no funcionales incluyen restricciones como el tiempo de respuesta (desempeño), la precisión, recursos consumidos, seguridad, etc.

Usabilidad:

- RNF_1 El sistema será una aplicación web con una curva de aprendizaje baja que pueda ser usada por cualquier persona que posea un nivel básico de conocimientos de computación.
- RNF_2 Deberá utilizar nombres sugerentes para lograr que el usuario encuentre lo que busca en el menor tiempo posible, las acciones a realizar serán fáciles de acceder.
- RNF_3 El sistema debe proporcionar mensajes de error que sean informativos y orientados a usuario final.
- RNF_4 La aplicación web debe poseer un diseño adaptable con el fin de garantizar la adecuada visualización en múltiples computadores personales, dispositivos tableta y teléfonos inteligentes.
- RNF_5 El sistema debe poseer interfaces gráficas bien formadas.

Seguridad:

- RNF_6 El sistema debe usar roles para especificar los privilegios de cada usuario.
- RNF_7 Los permisos de acceso al sistema podrán ser cambiados solamente por el administrador de acceso a datos.
- RNF_8 Todos los sistemas deben respaldarse cada 24 horas. Los respaldos deben ser almacenados en una localidad segura.

Hardware:

- RNF_9 Luego de instalar la solución en hardware con microprocesadores distintos (1era – Intel Pentium, 2da – AMD A8, 3era – Core I3) el sistema respondió más favorable a las peticiones en un servidor de aplicaciones web y de base de datos que cuente como mínimo con microprocesador Core i3 a 2,4 GHz o superior.
- RNF_10 Teniendo en cuenta el peso de la distribución de Visual Studio Code, el gestor de base de datos de MySQL y de la aplicación se debe disponer de aproximadamente 4GB de espacio de almacenamiento en disco como mínimo en el hardware de despliegue.
- RNF_11 Para el uso en las máquinas clientes, los requerimientos serán menores. Es necesario que estas posean al menos 1GB de memoria RAM y podrán visualizarse desde cualquier sistema de cómputo (móvil o de escritorio) con el software necesario.
- RNF_12 Los hardware clientes y servidor deben poseer una tarjeta de interconexión de red que permita acceder y responder a las peticiones respectivamente.

2.2 Fase 2: Planificación de las entregas

Se cumple con la **Fase 2** de la Metodología XP donde el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres semanas.

2.2.1 Historias de usuario

Las historias de usuario son la técnica utilizada para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas.

Se realiza la selección del RF_2 por el nivel de prioridad para ser desarrollados a continuación:

Historia de Usuario #2: Autenticar usuario	
Número: 2	Nombre del requisito: Autenticar usuario
Programador: Dariel López Vera	Iteración Asignada: 1

Prioridad: Alta	Tiempo Estimado:
Riesgo en Desarrollo: N/A	Tiempo Real:
Descripción: Una vez creado el usuario (ver HU # 2), le permite iniciar sesión y acceder a las funcionalidades que le fueron asignadas según el rol que desempeñe el mismo.	
Observaciones: N/A	

Tabla 4 Historia de usuario

2.2.2 Estimación de tiempo por historia de usuario

La programación bajo la metodología XP basa sus procesos de planificación en estimaciones temporales de las historias de usuario, las cuáles deben ser realizadas por los desarrolladores durante las diversas reuniones de planificación.

Una historia de usuario es lo suficientemente pequeña como para que el equipo la desarrolle durante una entrega de una a tres semanas; más de tres semanas implica que se debe señalar al cliente que debe dividir una historia de usuario y menos de una semana implica que la historia es demasiado sencilla y por lo que se deben unir dos o más de ellas para su mejor interpretación.

Las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto de estimación. Un punto de estimación, equivale a una semana ideal de programación. Las historias generalmente valen de 1 a 3 puntos de estimación. Por otra parte, el equipo de desarrollo mantiene un registro de la "velocidad" de desarrollo, establecida en puntos por iteración, basándose principalmente en la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración.

En la siguiente tabla de estimación del tiempo de las historias de usuario, se usó la denotación de punto de estimación por semana y la denotación de puntos flotantes para indicar días.

Ejemplo:

Denotación de punto de estimación	Interpretación

1 punto	1 día (8 horas laborales)
1.5 puntos	1 día y 4 horas

Tabla 5 Estimación de tiempo

Historia de usuario	Punto estimado
Crear Usuario	8
Eliminar Usuario	3
Modificar Usuario	2
Autenticar Usuario	2
Listar/Mostrar Usuario	2
Agregar contenido	5
Eliminar contenido	2
Modificar contenido	3
Buscar contenido	2
Listar/Mostrar contenido	2
Autenticar usuario	8
Agregar noticia	7
Eliminar noticia	5
Modificar noticia	6
Buscar noticia	3

Mostrar/Listar noticia	2
Informar Bastión	2
Modificar Información del Bastión	2
Eliminar Información del Bastión	2
Divulgar Información de posibles Cursos	3
Modificar Información de posibles Cursos	3
Eliminar Información de posibles Cursos	3
Listar Información de posibles Cursos	3

Tabla 6 Estimación del tiempo de HU

Las estimaciones realizadas permitieron confeccionar una evaluación puntual del tiempo de implementación de cada historia de usuario para la posterior elaboración del plan de iteración. Una vez realizadas las estimaciones fue preciso construir un plan de iteración donde se pueden agrupar estas historias y dar su cumplimiento de manera paulatina.

2.2.3 Plan de iteraciones

Un plan de iteraciones está constituido por un conjunto secuencial de actividades y tareas, cada una tiene recursos asignados y pueden depender a su vez de otras tareas. El plan de iteración se realiza una vez por cada iteración.

El contenido de la iteración puede variar, dependiendo de la posición dentro del ciclo de vida y de la naturaleza del proyecto. El plan de iteración incluye porciones relevantes de todas las disciplinas para cada iteración en particular, la prioridad de implementación se evalúa en base al cliente y el equipo de desarrollo, y el impacto del riesgo en base al juicio de experto. Los planes de iteración por lo general muestran un planeamiento detallado de quién va a realizar una tarea/actividad de acuerdo en conformidad a qué criterios de evaluación.

No. Iteración	Historia de Usuario	Prioridad	Esfuerzo Estimado
---------------	---------------------	-----------	-------------------

1	Crear Usuario	Alta	8	25
	Eliminar Usuario	Media	3	
	Modificar Usuario	Baja	2	
	Buscar Usuario	Media	2	
	Listar/Mostrar Usuario	Baja	2	
	Autenticar usuario	Alta	8	

2	Agregar contenido	Alta	5	14
	Eliminar contenido	Alta	2	
	Modificar contenido	Media	3	
	Buscar contenido	Alta	2	
	Listar/Mostrar contenido	Alta	2	
	Agregar noticia	Alta	7	38
	Eliminar noticia	Media	5	
	Modificar noticia	Media	6	

3	Buscar noticia	Media	3
	Mostrar/Listar noticia	Media	2
	Informar Bastión	Media	2
	Modificar Información del Bastión	Media	2
	Eliminar Información del Bastión	Media	2
	Divulgar Información de posibles Cursos	Media	3
	Modificar Información de posibles Cursos	Media	3
	Eliminar Información de posibles Cursos	Media	3
	Listar Información de posibles Cursos	Media	3

Tabla 7 Plan de Iteraciones

Una vez realizado el plan de iteraciones se pudo agrupar las diferentes historias de usuario en 3 iteraciones teniendo en cuenta las características que rigen la metodología XP. Con este plan de iteraciones ya es posible realizar un plan de entrega que será entregado al cliente y que el grupo de desarrollo está obligado a hacer cumplir.

2.2.4 Plan de entrega

Determinada la duración de cada iteración. Se presenta el plan de entrega elaborado para la fase de implementación teniendo en cuenta que el desarrollo del proyecto inicia el 1/06/2022 y concluirá el 6/09/2022:

	Iteración No.1	Iteración No.2	Iteración No.3
Cantidad de HU	6	5	12
Fecha de inicio	01/06/2022	7/07/2022	28/07/2022
Fecha de entrega	6/07/2022	27/07/2022	6/09/2022

Tabla 8 Plan de entrega

Una vez realizado el plan de entrega se puede confirmar con el cliente que el proyecto durará 16 semanas aproximadamente, lo que significa que tendrá un tiempo de desarrollo de 4 meses aproximadamente.

2.2.5 Reuniones diarias de seguimiento

El planeamiento es esencial para cualquier tipo de metodología, es por ello que XP requiere de una revisión continua del plan de trabajo. A pesar de ser una metodología que evita la documentación exagerada, es muy estricta en la organización del trabajo. Para la misma, el equipo de desarrollo definió desde el inicio aquellos espacios en los que el equipo y el cliente realizarían encuentros semanales para la evaluación del cumplimiento de los resultados de las diferentes iteraciones, así como el cumplimiento parcial, total o nuevas redefiniciones a los conceptos establecidos. Las pequeñas entregas facilitarán el trabajo de los desarrolladores, haciendo posible suplir las necesidades del cliente.

Reunión de	Fecha	Descripción
Plan de entregas	1/5/2022	Se realiza entre el equipo de trabajo y los clientes y se define el marco temporal de la realización del sistema. El cliente expone las historias de usuario a los integrantes de grupo, quienes estimarán el grado de dificultad de la implementación de cada historia. A partir de las historias de usuario, el cliente plantea las pruebas de aceptación con las cuales se comprueba que cada una de estas ha sido correctamente implementada.

Inicial de Iteración	01/06/2022 7/07/2022 28/07/2022	Esta reunión es realizada previo a iniciar una iteración donde se organizan las actividades de programación a realizar. Las historias de usuario son traducidas a tareas y asignadas a los desarrolladores.
Diarias	Al inicio de cada jornada de trabajo	Estas reuniones se realizan al comenzar la jornada laboral, donde todo el equipo de desarrollo se reúne para exponer los problemas e ideas que se estén presentando. Es de vital importancia evitar las discusiones largas, ya que se está utilizando tiempo laboral que puede ser destinado a la construcción del sistema
Fin de iteración	6/07/2022 27/07/2022 6/09/2022	Estas reuniones se realizan al finalizar cada iteración en conjunto con los clientes para presentar los avances en cada iteración y demostrar la aceptación por los mismos como muestra de una correcta implementación.

Tabla 9 Reuniones de seguimiento

Con el plan de reuniones queda determinado la secuencia de las distintas actividades a realizar dentro del proceso de desarrollo de la solución, contribuyendo a la calidad del producto deseado. Una vez determinado los artefactos organizativos, se procede a la fase de diseño de la solución propuesta.

2.3 Fase 3: Etapa de diseño

Teniendo en cuenta lo que plantea la metodología XP, en esta fase se confeccionan las tarjetas ClaseResponsabilidad-Colaborador (CRC) para la descripción de las principales clases de los módulos desarrollados. Se define la arquitectura del sistema y los estándares de codificación, así como los patrones de diseño utilizados en el desarrollo de la propuesta.

2.3.1 Tarjetas CRC

Las tarjetas CRC se elaboran durante la fase de diseño de la metodología XP para describir las entidades existentes en la aplicación. El uso de este tipo de tarjetas es una técnica de modelado que permite identificar las clases, responsabilidades y colaboraciones. El objetivo es obtener un diseño simple, elegante y fácil de comprender por parte de los programadores.

Nombre clase: Noticia.py	
Superclase: Controller.py	
Responsabilidad:	Colaborador:
Crear Noticia	Administrador
Modificar Noticia	
Eliminar Noticia	
Mostrar Noticia	

Tabla 10 Tarjeta CRC

Con la realización de las tarjetas **CRC** se evidencia la interrelación existente entre cada clase de la solución y sus diferentes funcionalidades. Teniendo en cuenta la interrelación de las entidades se hace necesario realizar prototipos que evidencien al cliente y al equipo de desarrollo el resultado que se obtendría con la solución.

2.3.2 Arquitectura Modelo-Template-Controlador

Django sigue el patrón MVC tan al pie de la letra que puede ser llamado un *framework* MVC. Somera-mente, la M, V y C se separan en Django de la siguiente manera:

M significa "Model" (Modelo), la porción de acceso a la base de datos, es manejada por la capa de la base de datos de Django, la cual describiremos en este capítulo.

V significa "View" (Vista), la porción que selecciona qué datos mostrar y cómo mostrarlos, es maneja-da por la vista y las plantillas.

C significa "Controller" (Controlador), la porción que delega a la vista dependiendo de la entrada del usuario, es manejada por el framework mismo siguiendo tu URLconf y llamando a la función apropia-da de Python para la URL obtenida.(5.2. El patrón de diseño MTV (El libro de Django 1.0), 2017)

Debido a que la "C" es manejada por el mismo framework y la parte más importante se produce en los modelos, las plantillas y las vistas, Django es conocido como un Framework MTV. En el patrón de diseño MTV.(5.2. El patrón de diseño MTV (El libro de Django 1.0), 2017)

M significa "Model" (Modelo), la capa de acceso a la base de datos. Esta capa contiene toda la información sobre los datos: cómo acceder a estos, cómo validarlos, cuál es el comportamiento que tiene, y las relaciones entre los datos.

T significa "Template" (Plantilla), la capa de presentación. Esta capa contiene las decisiones relacionadas a la presentación: como algunas cosas son mostradas sobre una página web u otro tipo de documento.

V significa "View" (Vista), la capa de la lógica de negocios. Esta capa contiene la lógica que accede al modelo y la delega a la plantilla apropiada: puedes pensar en esto como un puente entre los modelos y las plantillas. (5.2. El patrón de diseño MTV (El libro de Django 1.0), 2017)

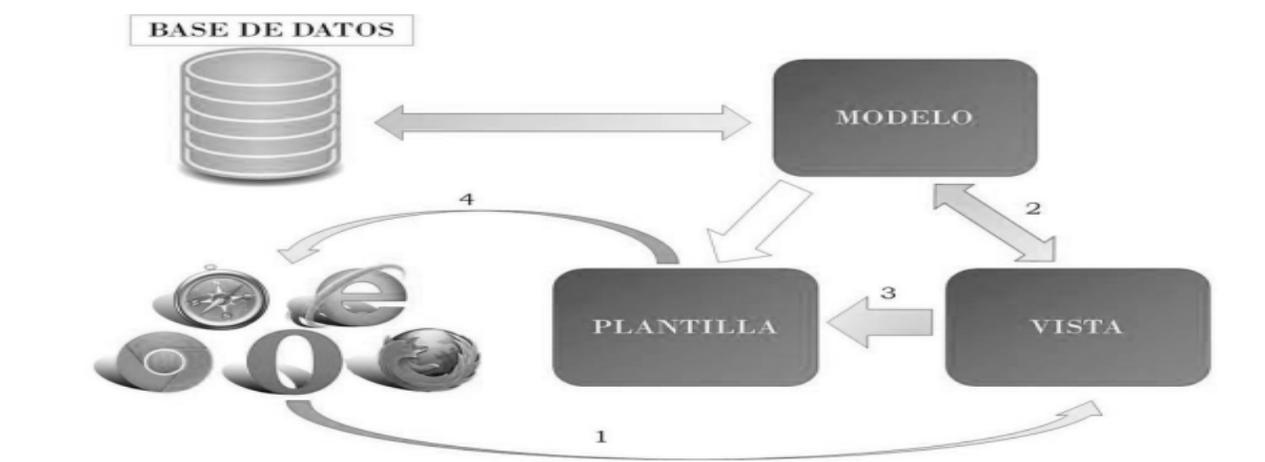


Fig 3 Modelo Vista Plantilla

Plantilla

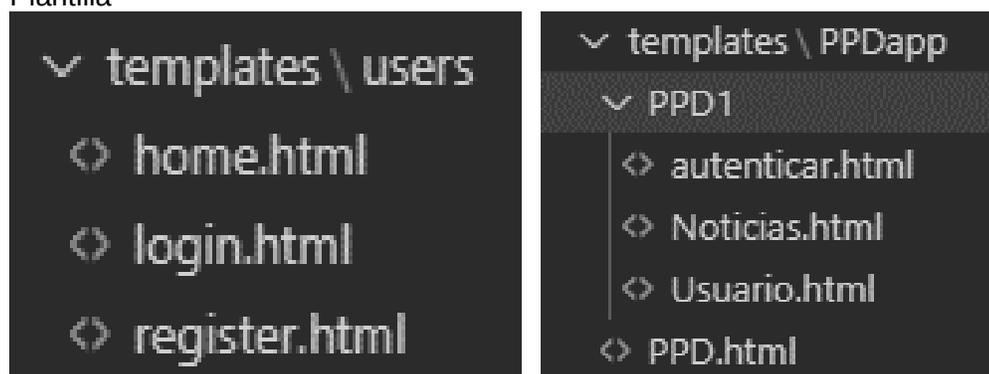


Fig 4 Capa de Template

Modelo

```
# Create your models here.
class Noticia(models.Model): ...

class Documentos(models.Model):|...
```

Fig 5 Capa de Modelo

Vistas

```
# Create your views here.
> def PPD (request): ...

> def Listado_de_Noticia (request): ...

> def autenticar(request): ...

> def login_user(request): ...

> def logout_user(request): ...

> def home(request): ...

> def register(request): ...
```

Fig 6 Capa de Views

2.3.3 Prototipo de interfaz

Los prototipos ayudan a identificar, comunicar y probar un producto antes de crearlo. Al tratarse de una Sistema para la gestión de contenido para la Preparación de la Defensa en la Educación Superior, se ha de conseguir que la atención del usuario que entra en la web se centre mayoritariamente en la información que se presenta dependiendo de intereses personales.



Fig 7 Prototipo de interfaz de Portada

Símbolos Patrios



EL ESCUDO DE LA PALMA REAL

El escudo se presenta sostenido por un haz de varas, unido por una banda estrecha de color rojo cruzada en equis que indica la unión de los cubanos. La corona del escudo está cubierta por un gorro frigio de color rojo, con una estrella blanca de cinco puntas en el centro, vuelto hacia la derecha.



EL HIMNO DE BAYAMO

Es un himno de combate, surgido en el fragor de la lucha subversiva contra el poder de la metrópoli. Llama a defender la Patria en el combate y ofender la propia vida en busca de la ansiada libertad. El Himno de Bayamo es el símbolo de la nación cubana, compuesto por Pedro Figueredo en agosto de 1867.



LA BANDERA DE LA ESTRELLA SOLITARIA

La bandera consta de tres franjas azules, que representan a los departamentos en los que se dividía la isla por aquel entonces; dos franjas blancas, simbolizando la fuerza del ideal independentista; y un triángulo rojo, que simboliza la igualdad, la libertad y la fraternidad; en cuyo centro se ubica una estrella

Fig 8 Prototipo de interfaz de Símbolos Patrios



Fig 9 Prototipo de interfaz de Autenticación

2.4 Fase 4: Implementación

El desarrollo o la codificación es un proceso que se realiza en forma paralela con el diseño y la cual está sujeta a varias observaciones por parte de XP. Dentro de esta fase se describen los estándares de codificación, así como los patrones de diseño que se emplearon dentro del sistema.

2.4.1 Estándares de codificación

En el proceso de desarrollo de un software siguiendo la metodología XP, es necesario que exista una adecuada comunicación entre los programadores del equipo de desarrollo. Para lograr esto, se establecen un conjunto de estándares definidos por las convenciones de escritura de código Python abarcando el marco de trabajo Django.

La razón para formular estándares de codificación es que podemos mantener el estilo de código consistente durante el desarrollo y mantenimiento. En el ciclo de vida de un proyecto, diferentes personas pueden participar en el desarrollo y mantenimiento. Los estándares de codificación que aseguran un estilo consistente sin duda evitarán muchos Los riesgos mantienen el proyecto en un estado razonable y saludable. *(Estándares de codificación - programador clic, 2021)*

Los estándares de codificación deben entenderse antes de aprender un idioma. Python usa PEP8 como estándar de codificación. PEP es la abreviatura de Python Enhancement Proposal y 8 representa la guía de estilo para el código Python.

- Codificación

El núcleo ha admitido Unicode desde Python 2.4. En cualquier caso UsoUTF-8

```
#-*- coding:utf-8 -*-
```

- Importar

En general, debe usar las importaciones en líneas separadas.

Por ejemplo:

1. No: import sys, os
2. Yes: import sys
3. import os

Pero esto también es posible:

```
from types import StringType, ListType
```

2.4.2 Patrones de diseño

Si bien la elaboración de un buen diseño del software contribuye directamente a la calidad del producto final, gran parte de esta yace en la utilización adecuada de los patrones de diseño y arquitectónicos existentes en la actualidad. Los patrones de diseño son soluciones habituales a problemas que ocurren con frecuencia en el diseño de software. Son como planos prefabricados que se pueden personalizar para resolver un problema de diseño recurrente en tu código. (*¿Qué es un patrón de diseño?*,2022)

Con empleo del marco de trabajo Django se garantiza la utilización de varios de estos patrones, permitiendo a los desarrolladores utilizar buenas prácticas de programación y ahorrar tiempo y recursos. A continuación, se describen los principales patrones empleados en el diseño e implementación de la solución.

2.4.2.1 Patrones GRASP

Los patrones GRASP (por sus siglas en inglés, *General Responsibility Assignment Software Patterns*) describen los principios fundamentales de la asignación de responsabilidades a objetos. El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el software orientado a objetos.

Los patrones GRASP utilizados en la solución propuesta fueron:

Sistema para la gestión de contenido de la Preparación para la Defensa en la Educación Superior

- Creador: este patrón es quien guía el proceso de asignación de responsabilidades relacionadas con la creación de objetos. Tiene el objetivo de asignarle a la clase B la responsabilidad de crear una instancia de la clase A.
- Bajo acoplamiento: Plantea que se debe poder reutilizar las funcionalidades de las distintas clases, con un nivel de dependencia mínima. Este patrón se evidencia en todas las aplicaciones web que funcionen sobre Django, pues cada pieza de las aplicaciones tiene un propósito clave, que puede modificarse sin afectar otras piezas. Por ejemplo, se puede cambiar la URL de cierta parte de la aplicación sin tener que afectar la implementación subyacente o se puede modificar el HTML de una página sin tener que tocar el código Python que la renderiza.
- Alta cohesión: Asigna responsabilidades de manera tal que la cohesión siga siendo alta, o sea que las funcionalidades de las clases estén altamente relacionadas de forma tal que exista una colaboración entre ellas para compartir el esfuerzo y no caiga todo el peso sobre una única clase. Usar este patrón simplifica el mantenimiento y favorece el bajo acoplamiento.
- Controlador: Permite manejar todos los eventos del sistema, al servir de intermediario entre las interfaces y el algoritmo que las implementa.

```
def Crear_Noticia (request):
    datos = request.POST
    titulo = request.POST['titulo']
    descripcion = request.POST['descripcion']
    autor = request.user
    Noticia.objects.create(titulo=titulo, descripcion=descripcion, autor=autor)
    noticias = Noticia.objects.all().order_by('created')
    data = {
        'noticias': Noticia.objects.filter(autor=request.user)
    }

    return render(request, "PPDapp/PPD1/Administrador.html", data)
```

Fig 10 Ejemplo de Patrones GRASP (Creador, Bajo Acoplamiento, Alta Cohesión y Controlador)

Fuente: Elaboración Propia

2.4.2.2 Patrones GOF

Patrones publicados por Gamma, Helm, Johnson y Vlossodes en 1995: patrones de la banda de los

cuatro (del inglés, *Gang of Four*). Esta serie de patrones permiten ampliar el lenguaje, aprender nuevos estilos de diseño y además introducir más notación UML. Existen 23 patrones GoF de los cuales 15 se utilizan con frecuencia. Los patrones de diseño del grupo GoF se clasifican en tres grandes categorías basadas en su propósito: creacionales, estructurales y de comportamiento. A continuación, se describe el patrón GoF utilizado en la solución propuesta:

- Decorador: Patrón estructural que extiende la funcionalidad de un objeto dinámicamente de manera tal que es transparente a sus clientes, utiliza una instancia de una subclase de la clase original que delega las operaciones al objeto original. Este patrón se evidencia en la clase Login require (decorador que trae Django por defecto, para acceder a una clase).

```

• def login_user(request):
•     username = request.POST['username']
•     password = request.POST['password']
•     message = ""
•     user = authenticate(request, username=username, password=password)
•     if user is not None:
•         login(request, user)
•         if user.is_superuser:
•             noticias = Noticia.objects.all().order_by('created')
•             return render(request, "PPDapp/PPD1/Administrador.html", {'noticias':noticias})
•
•         else:
•             noticias = Noticia.objects.all().order_by('created')
•             return render(request, "PPDapp/PPD1/Usuario.html", {'noticias':notivias})
•     else:
•         messages.add_message(request , messages.INFO, 'Usuario no Válido' )
•         return render(request, "PPDapp/PPD1/autenticar.html", {'message':message}

```

Fig 11 Ejemplo de Patrones GoF (Decorador)Fuente: Elaboración Propia

2.4.3 Modelo de datos

EL modelo de datos es un lenguaje orientado a hablar de una base de datos. Típicamente un modelo de datos permite describir:

- Las estructuras de datos de la base: El tipo de los datos que hay en la base y la forma en que se relacionan.
- Las restricciones de integridad: Un conjunto de condiciones que deben cumplir los datos para reflejar la realidad deseada.

- Operaciones de manipulación de los datos: típicamente, operaciones de agregado, borrado, modificación y recuperación de los datos de la base.

•

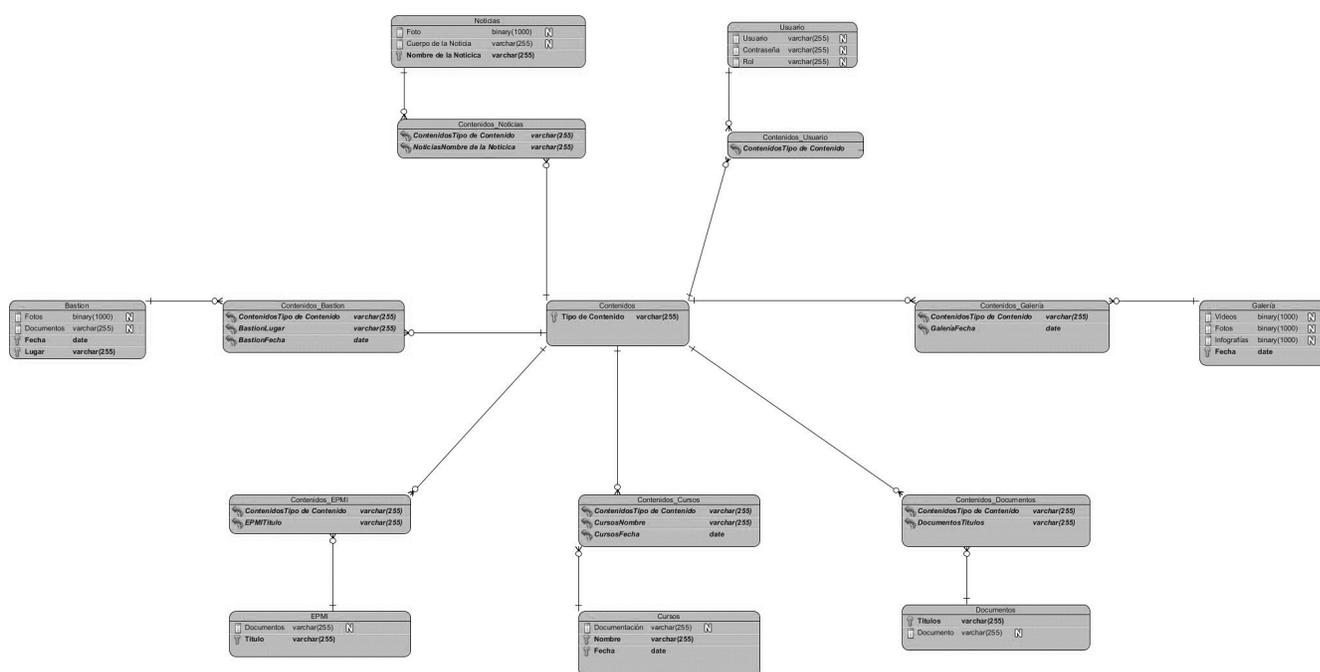


Fig 12 Modelo de Base de Datos Fuente: Elaboración propia

2.5 Conclusiones del capítulo

- La generación de los artefactos en la etapa de implementación de la plataforma siguiendo la Metodología XP, facilitaron las tareas de implementación, la reducción de errores y riesgos.
- El uso del modelo conceptual permitió describir el contexto para la gestión de contenido de la Preparación para la Defensa en la Educación Superior en Cuba e identificar la complejidad de las características para la propuesta de solución

- Se realizó la captura de los principales requerimientos que contribuyó a la confiabilidad de los datos recolectados y a satisfacer las expectativas de los usuarios mediante las características de la aplicación.
- Se realizaron las tarjetas CRC para describir las entidades existentes y tener un mayor dominio de las diferentes clases a desarrollar dentro de la solución.

CAPÍTULO III: VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

En este capítulo se aborda las diferentes pruebas realizadas al software y que son exigidas por la metodología XP para el buen funcionamiento y calidad de la propuesta desarrollada. Se detalla con exactitud los diferentes resultados alojados por cada prueba y la respuesta a dichas no conformidades.

3.1 Diagrama de despliegue

Los diagramas de despliegue se utilizan para visualizar los procesadores/ nodos/dispositivos de hardware de un sistema, los enlaces de comunicación entre ellos y la colocación de los archivos de software en ese hardware. («Tutorial de Diagrama de Despliegue | ¿Qué es un Diagrama de Despliegue», 2020)

Los elementos usados por este tipo de diagrama son nodos (representados como un prisma), componentes (representados como una caja rectangular con dos protuberancias del lado izquierdo) y asociaciones.

A continuación, se representa el ambiente donde debe ser desplegada la solución:

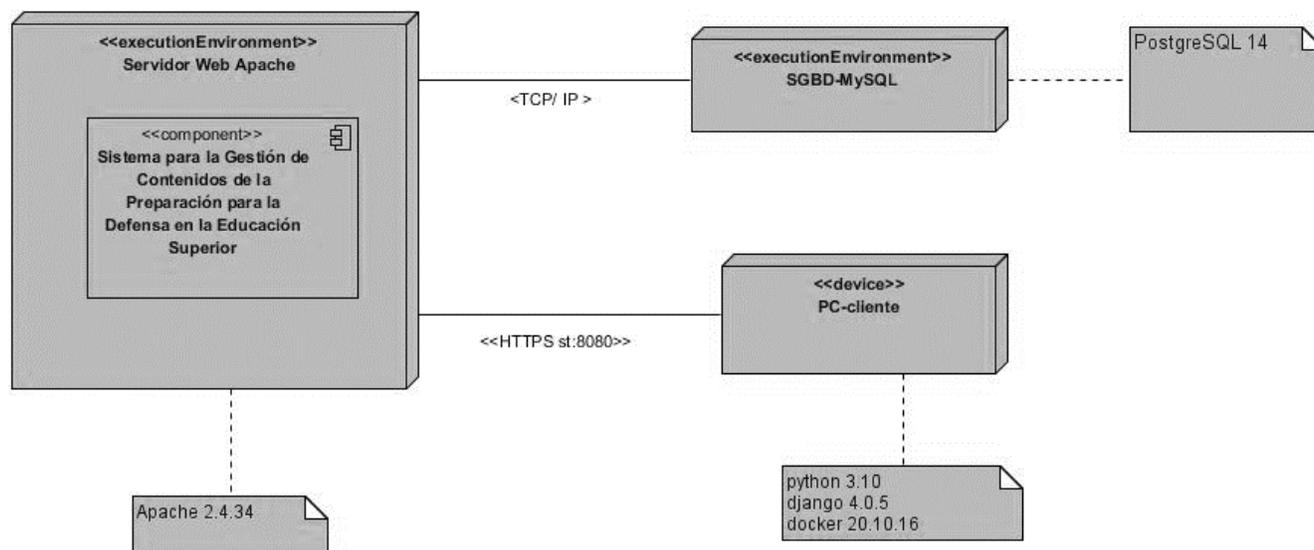


Fig 13 Diagrama de Despliegue Fuente: Elaboración Propia

3.2 Fase 5: Pruebas

Las pruebas de software forman parte de la última fase que propone XP, con el objetivo de lograr una herramienta que cumpla con los requisitos previamente identificados. En el presente subcapítulo se describen las etapas de codificación y de pruebas de la propuesta de solución.

Sistema para la gestión de contenido de la Preparación para la Defensa en la Educación Superior

Las pruebas de software son un conjunto de procesos con los que se pretende probar un sistema o aplicación en diferentes momentos para comprobar su correcto funcionamiento. Este tipo de pruebas abarca cualquier estadio del desarrollo del sistema, desde su creación hasta su puesta en producción. Lo interesante de las pruebas es que se puedan ejecutar de manera automática, para determinar en cualquier momento si tenemos una aplicación estable o si, por el contrario, un cambio en una parte ha afectado a otras partes sin que nos demos cuenta.(Turrado,2016)

Por lo general, las pruebas involucran operaciones del sistema evaluando los resultados bajo condiciones controladas, lo que hace que la realización de pruebas al software sea un factor de vital importancia.

3.2.1 Estrategia de pruebas

La Metodología XP propone que las pruebas de software sean realizadas al término de cada iteración, garantizando el funcionamiento deseado y la aceptabilidad por el cliente para realizar una entrega funcional y acorde a las exigencias de un producto con calidad. Las dos pruebas exigidas por la metodología, por su importancia y agilidad en el proceso; son las pruebas unitarias y de aceptación. Se hicieron las pruebas unitarias al código al finalizar cada iteración e igualmente se realizaron las pruebas de aceptación.

3.2.2 Pruebas unitarias

Las pruebas unitarias son **pruebas de caja blanca**, una técnica de prueba de software en la que se prueba la estructura interna, el diseño y la codificación del software para verificar el flujo de entrada y salida y para mejorar el diseño, la usabilidad y la seguridad. En una prueba de caja blanca, los probadores aparecen en el código, por lo que se denomina prueba de caja limpia, prueba de caja abierta, prueba de caja transparente, prueba basada en código y prueba de caja de vidrio.(admin, 2020a)

Estas pruebas se basan en el minucioso examen de los detalles procedimentales, donde se comprueban los caminos lógicos del software proponiendo casos de prueba que examinen que todas las condiciones y/o bucles estén correctas para determinar si el estado real coincide con el esperado o afirmado. (Pressman,2021)

La prueba de caja blanca implica probar el código del software para lo siguiente:

- Agujeros de seguridad internos
- Rutas rotas o mal estructuradas en los procesos de codificación
- Las entradas específicas fluyen a través del código

- Rendimiento esperado
- Funcionalidad de bucle condicional
- Pruebe cada enunciado, objeto y función individualmente

Para realizar la prueba se decidió utilizar el Camino Básico lo que demuestra un conjunto de pasos base del programa, lo que se quiere lograr es que cada sentencia de código se ejecute mínimo una vez.

Para calcularla, se tiene tres opciones:

- Restar las aristas menos los nodos y sumar 2:
 - o $V(G) = \text{Aristas} - \text{Nodos} + 2$
- Sumar 1 al número de nodos predicados
 - o $V(G) = \text{Nodos predicados} + 1$
- Contar el número de regiones (espacios «encerrados entre nodos y aristas», también se tiene en cuenta el espacio «exterior» a todos los nodos y aristas).
 - o $V(G) = \text{Regiones}$

Luego de realizar la prueba de caja blanca a la función `login_user(request)` se determinó que la complejidad ciclomática era 3 por lo que el método es sencillo y de poco riesgo para el sistema.

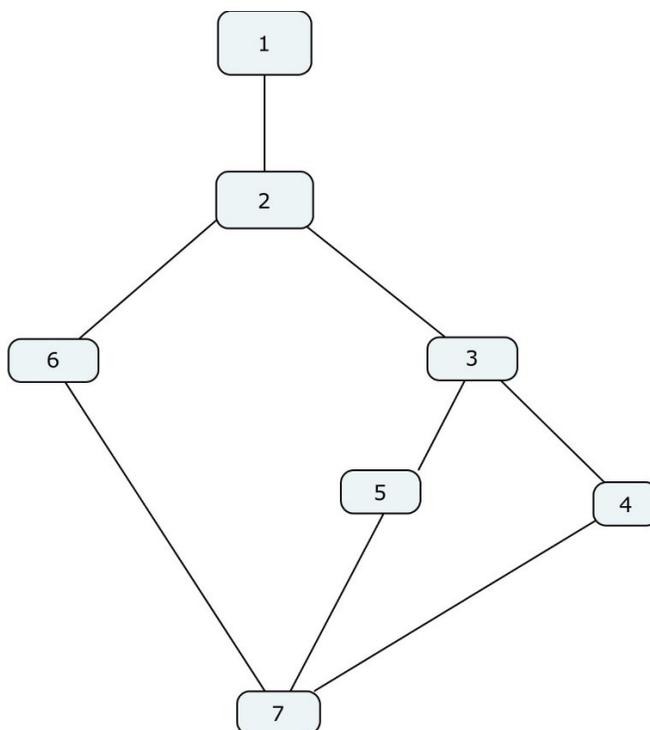


Fig 14 Camino Básico Fuente: Elaboración propia

Camino:

1,2,3,4,7

1,2,3,5,7

1,2,6,7

Camino	Usuario	Contraseña
1,2,3,4,7	Superusuario	Válida
1,2,3,5,7	Usuario	Válida
1,2,6,7	No está Registrado	Inválida

Tabla 11 Juego de Datos Fuente: Elaboración Propia

3.2.3 Pruebas de Caja Negra

Las pruebas de caja negra, conocidas también como *black box testing*, pueden definirse como una técnica donde se busca la verificación de las funcionalidades del *software* o aplicación analizada, sin tomar como referente la estructura del código interno, las rutas de tipo internas ni la información referente a la implementación. Esto quiere decir que la prueba se lleva a cabo con desconocimiento del

funcionamiento del sistema interno, debido a que se enfoca en la entrada y salida de un *software*, tomando como base sus especificaciones y requisitos. De manera que se puede asegurar que el objetivo de las pruebas de caja negra está relacionado con la validación de los recursos funcionales del *software* o aplicación que se busca examinar. (KeepCoding, 2022)

Mientras más amplio sea el espectro de elementos de entrada para realizar la prueba, las probabilidades de encontrar problemas en el software aumentan y, por lo tanto, será más confiable la calidad del software.

Este tipo de pruebas permite encontrar:

- Funciones incorrectas o ausentes
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación.

Para preparar los casos de prueba es necesario un número de datos que ayuden a la ejecución de estos casos y que permitan que el sistema se ejecute en todas sus variantes. Estos datos pueden ser válidos o no para el software y son seleccionados atendiendo a las especificidades de la funcionalidad a probar. Para realiza estas pruebas existen varias técnicas:

- **Partición de equivalencia:** divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software. Esta técnica es muy efectiva puesto que permite examinar los valores válidos e inválidos de las entradas existentes en el producto, descubre de forma inmediata una clase de errores que, de otro modo, requerirían la realización de muchos casos antes de detectar el error genérico. La partición de equivalente se dirige a la definición de casos de pruebas que descubran clases de errores, reduciendo de este modo el número de pruebas a realizar.

Se realizaron 6 casos de prueba a continuación se muestra un ejemplo de los mismos:

Identificador	Entrada(clase)	Clases válidas	Clases inválidas
Usuario	Usuario	Usuario=a-z	Usuario=0-9, caracteres especiales
Contraseña	Contraseña	Contraseña= a-z ,0-9, caracteres especiales	Contraseña=""

Tabla 12 Clases de equivalencia

Id del escenario	Escenario	Usuario	Contraseña	Respuesta del sistema	Resultado de la prueba
EC1	Autenticarse	dariellv V	Yinetmivida.9 8 V	El usuario se autentico correctamente	Satisfactorio
EC2	Autenticarse	ybolano V	Mi vida.980316 i	La contraseña no puede contener espacios en blanco	No satisfactorio
EC3	Autenticarse	dariellv234 i	Miamor.2345* V	El usuario no cumple los requisitos	No satisfactorio
EC4	Autenticarse	Ybolano23. i	Nioshf ndoahefo i	El usuario y contraseña no son válidos	No satisfactorio

3.2.4 Pruebas de Seguridad

Las Pruebas de Seguridad son un tipo de prueba de software que revela vulnerabilidades, amenazas, riesgos en una aplicación de software y previene ataques maliciosos de intrusos. El propósito de las Pruebas de Seguridad es identificar todas las lagunas y debilidades potenciales en el sistema de software que podrían resultar en la pérdida de información, ingresos, reputación de los empleados o personas ajenas a la Organización.(admin, 2020b)

Para el desarrollo de las mismas se utilizó la herramienta Netsparker que encuentra e informa de las vulnerabilidades de una aplicación web, como la inyección de SQL y de cross-site scripting (XSS) en todos los tipos de aplicaciones web, independientemente de la plataforma y de la tecnología con la que están construidos.

La tecnología única y absolutamente precisa Proof-Based Scanning™ de Netsparker no sólo informa de las vulnerabilidades, también produce una prueba de concepto para confirmar que no sean falsos positivos. Lo que le libera de tener que comprobar las vulnerabilidades identificadas. («Netsparker Web Application Security Scanner * QMA MSS», 2019)

Se realizó un análisis del sistema con la herramienta, los resultados del mismo están reflejados en el informe emitido por la misma herramienta.

3.2.5 Pruebas de aceptación

La prueba de aceptación del usuario (UAT), también conocida como prueba beta o de usuario final, se define como la prueba del software por parte del usuario o cliente para determinar si puede ser aceptado o no. Esta es la prueba final que se realiza una vez que se completan las pruebas funcionales, de sistema y de regresión. (¿Qué es la prueba de aceptación del usuario (UAT), 2019)

Estas pruebas son cruciales para determinar si el software cumple las expectativas y es viable para sus usuarios, es decir, si es aceptable y apto para salir al mercado. Sólo si lo es, se añade a un entorno de producción y se utiliza en el curso normal de la actividad empresarial. (Digité, 2022)

En el desarrollo ágil, las pruebas de aceptación forman parte del proceso y no son una ocurrencia tardía. Sin embargo, la intención sigue siendo la misma: verificar que el software cumple las expectativas desde el punto de vista del cliente y de los usuarios finales. (Digité, 2022) (Digité, 2022)

Caso de Prueba de Aceptación

Código: HU1_P1

HU: 1. Autenticar Usuario

Responsable: Dariel López Vera

Descripción: Prueba de funcionalidad que permite que el usuario pueda autenticarse en el sistema e interactuar con él.

En el caso de que sea administrador pueda autenticarse en el sistema, para que posteriormente pueda realizar las operaciones que tengan que ver con gestionar algún elemento del mismo.

Condiciones de ejecución: el usuario debe poner su usuario y contraseña y luego dar clic en el botón Autenticarse. Sino esta autenticado, dar clic en el botón Registrarse

Entrada/Pasos de ejecución:

- Se introduce el usuario en el campo Usuario.
- Se introduce la clave en el campo Contraseña.
- Presionar el botón Autenticarse.

- Sino presionar el botón Registrarse.

Resultados esperados: Debe entrar al sistema y posteriormente se debe mostrar las funcionalidades con las que puede interactuar.

Evaluación de la prueba: Satisfactoria.



Tabla 13 Prueba de Aceptación Historia de Usuario Fuente: Elaboración Propia

3.3 Resultados de las pruebas

A continuación, se muestra un gráfico final con los resultados arrojados en las cuatro iteraciones de pruebas donde se obtuvo un total de 20 no conformidades (NC) significativas: 8 con prioridad normal, 3 con prioridad baja, 4 con prioridad alta y 5 informativas. Los resultados obtenidos de las pruebas de aceptación por cada iteración pueden verse en la siguiente figura de este documento.

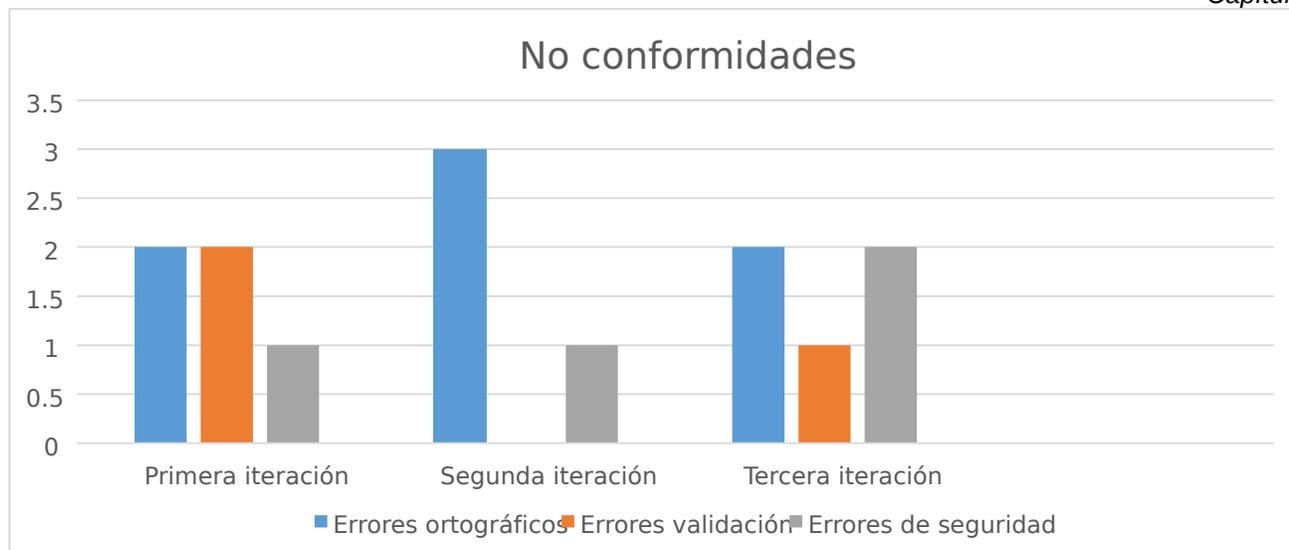


Fig 15 Gráfico de no conformidades Fuente: Elaboración Propia

Las principales NC no significativas encontradas fueron errores ortográficos, omisiones de tildes y cambio de mayúsculas por minúsculas. También se mostraban algunos mensajes innecesarios en pantalla y elementos de interfaz no sugerentes para un usuario con poca experiencia. Las principales NC significativas encontradas fueron errores de validación, errores en validaciones de los campos de los formularios. Después de concluida cada iteración se resolvieron las NC arrojadas quedando en un estado cerrado.

3.4 Validación de las variables

Luego del análisis, diseño e implementación de la solución se puede demostrar que:

Características	Antes de la solución	Después de la solución
Contenido Dispersos en Diferentes Sitios Web	Cuba no contaba con un Sistema donde se agrupará todo el contenido de la Preparación para la Defensa en la Educación Superior	Posee un sistema para la gestión de contenido de la Preparación para la Defensa en la Educación Superior

Intercambio entre profesores y alumnos a nivel nacional	Solo se hacía a nivel de Universidad	Ahora se hace a nivel de país
Posibilidad de dar noticias a nivel nacional	La forma en la que se divulgaban las noticias era de forma manual	Ahora se hace a través del sistema
Utilización de la conectividad para promoción y consumo de contenido	No se utilizaba	En Cuba existen más de 5 millones de líneas celulares, más de 2 millones de conexiones diarias, y todo lo expuesto está integrado en un solo sistema, dominio, permitiendo su rápida indexación, accesibilidad, tráfico y posicionamiento en motores de búsqueda.

Tabla 14 Validación de variables

Por tanto, se puede afirmar que con el desarrollo del sistema para la gestión de contenido de la Preparación para la Defensa en la Educación Superior contribuirá a la accesibilidad de los contenidos de la disciplina Preparación para la Defensa en la Educación Superior.

3.5 Conclusiones del capítulo

- Los diferentes métodos de pruebas aplicados a la solución durante el ciclo de vida del software permitieron comprobar los errores existentes y mejorar la calidad de los resultados.
- Se realizaron prueba de aceptación que pudo evidenciar el correcto funcionamiento y la aceptabilidad de los requisitos descritos de conjunto con el cliente en cada iteración en el proceso de desarrollo de la metodología XP.
- Los resultados de las pruebas demostraron su efectividad para ser desplegada.

CONCLUSIONES GENERALES

Considerando los resultados descritos en este informe, la necesidad y el objetivo planteado por la investigación se arriba a las siguientes conclusiones:

- El análisis de las diferentes fuentes, herramientas y soluciones tecnológicas para la distribución de contenido permitió identificar las características claves que distinguen al sistema propuesto y su adaptación al contexto nacional.
- El empleo de las herramientas y tecnologías seleccionadas para la implementación de la solución propició la correspondencia entre los resultados obtenidos y los esperados, lo cual pudo asegurar el nivel de precisión en el análisis y diseño del sistema.
- La aplicación de diversas pruebas de software arrojó resultados satisfactorios en relación al código y el conjunto de interfaces implementadas, las cuales demostraron, además, la invulnerabilidad en el código y su capacidad técnica en función del entorno para el cual fue desarrollado.
- Los resultados de la investigación que describe este informe demostraron la viabilidad de la solución propuesta para la gestión de contenido de la Preparación para la Defensa en la Educación Superior en Cuba, cuyo desarrollo contribuye a la accesibilidad de sus contenidos.

RECOMENDACIONES

Los resultados obtenidos en este trabajo sientan las bases para la realización de proyectos futuros que permitan enriquecer y escalar la solución propuesta. Por lo que se recomienda que:

- Integrar al sistema para la gestión de contenido de la Preparación para la Defensa en la Educación Superior un motor de búsqueda.
- Desarrollar complemento de analítica web para monitorear el tráfico y comportamiento de los usuarios.
- Integrar al sistema para la gestión de contenido de la Preparación para la Defensa en la Educación Superior con repositorios universitarios y el sistema de catálogos de las bibliotecas.
- Crear una apk para el uso en teléfonos móviles.
- Integración con otros sistemas relacionados con la Preparación para la Defensa

REFERENCIAS BIBLIOGRÁFICAS

1. 5.2. *El patrón de diseño MTV (El libro de Django 1.0)*. (s. f.). Recuperado 6 de septiembre de 2022, de 5.2. *El patrón de diseño MTV (El libro de Django 1.0)*. (s. f.). Recuperado 6 de septiembre de 2022, de <https://uniwebsidad.com/libros/django-1-0/capitulo-5/el-patron-de-diseno-mtv>
2. admin. (2020a, enero 1). ¿Qué es una prueba de caja BLANCA? Técnicas, muestras y tipos. *Ebooks Online*. <https://ebooksonline.es/que-es-una-prueba-de-caja-blanca-tecnicas-muestras-y-tipos/>
3. admin. (2020b, enero 1). ¿Qué es una prueba de seguridad? Tipos con ejemplo. *Ebooks Online*. <https://ebooksonline.es/que-es-una-prueba-de-seguridad-tipos-con-ejemplo/>
4. apiumhub. (s. f.). *Utilizar Docker: Beneficios, estadísticas* | Apiumhub. Recuperado 9 de junio de 2022, de <https://apiumhub.com/es/tech-blog-barcelona/beneficios-de-utilizar-docker/>
5. *CHAMILO LMS: el e-learning ético—OpenExpo Europe 2022*. (s. f.). Recuperado 9 de junio de 2022, de <https://openexpoeurope.com/es/chamilo-lms-el-e-learning-etico/>
6. *Ciclo de vida de un proyecto XP*. (s. f.). Recuperado 9 de junio de 2022, de <http://oness.sourceforge.net/proyecto/html/ch05s02.html>
7. *¿Cómo funciona Udemy? Preguntas más frecuentes*. (s. f.). Udeemy. Recuperado 9 de junio de 2022, de <https://support.udemy.com/hc/es/articles/229232187--C%C3%B3mo-funciona-Udemy-Preguntas-m%C3%A1s-frecuentes>
8. *Contenedores de Docker | ¿Qué es Docker?* | AWS. (s. f.). Amazon Web Services, Inc. Recuperado 9 de junio de 2022, de <https://aws.amazon.com/es/docker/>
9. *Definición de repositorio—Definicion.de*. (s. f.). Definición.de. Recuperado 9 de junio de 2022, de <https://definicion.de/repositorio/>
10. *DigitalChalk: Precios, funciones y opiniones* | *GetApp España 2022*. (s. f.). Recuperado 9 de junio de 2022, de <https://www.getapp.es/software/90530/digitalchalk>
11. *DigitalChalk—Opiniones de usuarios—Pros y Contras*. (s. f.). Recuperado 9 de junio de 2022, de <https://opinion-soft.com/digitalchalk/>
12. Digité. (2022). *Pruebas De Aceptación: El Qué Y Porqué + Los Tipos Que Hay Que Conocer*. <https://www.digite.com/es/agile/pruebas-de-aceptacion/>
13. *Django overview* | *Django*. (s. f.). Recuperado 9 de junio de 2022, de <https://www.djangoproject.com/start/overview/>
14. *Documentation for Visual Studio Code*. (s. f.). Recuperado 9 de junio de 2022, de <https://code.visualstudio.com/docs>
15. *Empaquetado de la aplicación: Qué es y por qué es importante*. (s. f.). Recuperado 31 de octubre de 2022, de <https://cpl.thalesgroup.com/es/software-monetization/application-packaging>
16. *Entornos Virtuales de Enseñanza y Aprendizaje (evea). Nivel 1* | *Citep*. (s. f.). Recuperado 9 de junio de 2022, de <http://citep.rec.uba.ar/entornos-virtuales-de-ensenanza-y-aprendizaje-evea-nivel-1-2/>
17. *Estándares de codificación—Programador clic*. (s. f.). Recuperado 6 de septiembre de 2022, de <https://programmerclick.com/article/2139976824/>
18. *Iel ciclo de vida de XP. - Búsqueda de Google*. (s. f.). Recuperado 9 de junio de 2022, de https://www.google.com/search?q=Iel+ciclo+de+vida+de+XP.&tbm=isch&ved=2ahUKEwivlcad4qH4AhWqh-AKHROPDxwQ2-cCegQIABAA&oq=Iel+ciclo+de+vida+de+XP.&gs_lcp=CgNpbWcQA1CoB1ioB2-CsC2gAcAB4AIABswKIAYcEkgeFmI0xLjGYAQCgAQQgAQQnd3Mtd2l6LWltZ8ABAQ&scclient=im g&ei=AKOiYq-VC6qPggeTnr7gAQ&bih=615&biw=1366&client=firefox-b-d#imgcr=SfeTMN5bE9o-WRM
19. KeepCoding, R. (2022, junio 3). *¿Qué son las pruebas de caja negra?* | *KeepCoding Tech School*. <https://keepcoding.io/blog/que-son-las-pruebas-de-caja-negra/>

20. Leyva, A., Gaubeca, F., & Alfonso, R. (2020). *Textos Especializados de la asignatura Defensa Nacional para los estudiantes de Ingeniería en Bioinformática*.
21. *LiClipse by brainwy*. (s. f.). Recuperado 9 de junio de 2022, de <https://www.liclipse.com/>
22. Mesias Torres, F. (2018). *INGENIERÍA DE REQUERIMIENTOS: MÉTODOS Y TÉCNICAS*. Universidad Autónoma Aguascalientes.
23. *Metodologías ágiles frente a las tradicionales en el proceso de desarrollo de software*. (s. f.). Recuperado 9 de junio de 2022, de <https://1library.co/document/ye3d7v1q-metodologias-agiles-frente-tradicionales-proceso-desarrollo-software.html>
24. Modelos conceptuales para definir las plataformas digitales | IDA Chile. (2016, noviembre 14). *Blog IDA Chile | Estrategia para el éxito de tu negocio*. <https://blog.ida.cl/experiencia-de-usuario/modelos-conceptuales-plataformas-digitales/>
25. Netsparker Web Application Security Scanner * QMA MSS. (2019, marzo 3). *QMA MSS*. <https://www.qma.mx/socios-estrategicos/netsparker/>
26. *Online Education Platform Udemy Claims 10 Million Users | Time*. (s. f.). Recuperado 9 de junio de 2022, de <https://time.com/4215787/udemy-dennis-yang/>
27. Oracle Corporation. (2018). <http://www.mysql.com/why-mysql/>.
28. Oscar Figueredo Reinaldo, E. C. T. (s. f.). *Cuba en datos: Informatización de la sociedad, apuntes más allá de la infraestructura*. <http://www.cubadebate.cu/especiales/2021/05/17/cuba-en-datos-informatizacion-de-la-sociedad-apuntes-mas-alla-de-la-infraestructura-infografia/>
29. Piattini. (s. f.). *Análisis y diseño de aplicaciones informáticas de gestión*.
30. Pressman, Roger S. *Ingeniería de Software - Un enfoque práctico 9ma*. MEXICO : s.n., 2021.
31. *¿Qué es la prueba de aceptación del usuario (UAT): Una guía completa? - Tipos De Pruebas*. (s. f.). Recuperado 2 de noviembre de 2022, de <https://spa.myservername.com/what-is-user-acceptance-testing>
32. *¿Qué es un IDE en programación?* (s. f.). UNIR. Recuperado 9 de junio de 2022, de <https://www.unir-net/ingenieria/revista/ide-programacion/>
33. *¿Qué es un patrón de diseño?* (s. f.). Recuperado 11 de septiembre de 2022, de <https://refactoring.guru/es/design-patterns/what-is-pattern>
34. RAE. (s. f.-a). *Definición de enseñanza militar—Diccionario panhispánico del español jurídico—RAE*. Diccionario panhispánico del español jurídico - Real Academia Española. Recuperado 9 de junio de 2022, de <https://dpej.rae.es/lema/ense%C3%B1anza-militar>
35. RAE. (s. f.-b). *Definición de Universidad—Diccionario panhispánico del español jurídico—RAE*. Diccionario panhispánico del español jurídico - Real Academia Española. Recuperado 9 de junio de 2022, de <https://dpej.rae.es/lema/universidad>
36. Series, A. A. (s. f.). *Agile Software Development ~*.
37. *Significado de Software libre*. (s. f.). Significados. Recuperado 9 de junio de 2022, de <https://www.significados.com/software-libre/>
38. *Sistema informático: Características, funciones, clasificación y etapas de desarrollo*. (s. f.). Recuperado 3 de noviembre de 2022, de <https://www.cesuma.mx/blog/sistema-informatico-caracteristicas-funciones-clasificacion-y-etapas-de-desarrollo.html>
39. *Software Propietario ¿Qué es el Software Propietario?* (2016, noviembre 25). *OK HOSTING | Hospedaje Web, Dominios, Desarrollo de Software, Marketing Online, SEO*. <https://okhosting.com/blog/software-propietario/>
40. Turrado, J. (s. f.). *¿Qué son las pruebas de software*. campusMVP.es. Recuperado 12 de septiembre de 2022, de <https://www.campusmvp.es/recursos/post/que-son-las-pruebas-de-software.aspx>
41. Tutorial de Diagrama de Despliegue | ¿Qué es un Diagrama de Despliegue. (2020, octubre 26). *Blog de Creately*. <https://creately.com/blog/es/diagramas/tutorial-de-diagrama-de-despliegue/>

42. *Visual Paradigm User's Guides*. (s. f.). Recuperado 9 de junio de 2022, de <https://www.visual-paradigm.com/support/documents/>
43. *visualparadigm.com—Visualparadigm Recursos e información*. (s. f.). Recuperado 9 de junio de 2022, de <https://online.visualparadigm.com/es/solutions/diagramming-tool/>

ANEXOS

ANEXO 1. GLOSARIO DE TÉRMINOS

Sistemas Informáticos: Pueden definirse como el propio equipo de cómputo con su sistema y software de aplicación y medios de almacenamiento electrónico, redes de área local y redes de área amplia. (*Sistema informático*, s. f.)

Lenguaje de modelado: Se entiende por lenguaje de modelado cualquier lenguaje artificial que puede ser utilizado para expresar la información, el conocimiento o sistemas en una estructura que está definida por un conjunto coherente de reglas.

Lenguaje de programación: Un Lenguaje de Programación es un conjunto de reglas, notaciones, símbolos y/o caracteres que permiten a un programador poder expresar el procesamiento de datos y sus estructuras en la computadora. Cada lenguaje posee sus propias sintaxis. También se puede decir que un programa es un conjunto de órdenes o instrucciones que resuelven un problema específico basado en un Lenguaje de Programación

MVC: Arquitectura Modelo-Vista-Controlador.

ANEXO 2. ENTREVISTA

Entrevista con el cliente:

- 1- ¿Cómo se gestiona los contenidos de la asignatura Preparación para la Defensa en la Educación Superior a nivel Nacional?
- 2- ¿Cómo se lleva a cabo la información de las principales actividades realizadas por la asignatura Preparación para la Defensa?
- 3- ¿Cuáles considera que sean los contenidos más importantes de la asignatura Preparación para la Defensa que sean importantes y necesarios que deban conocer los estudiantes?
- 4- ¿Cómo debería estar conformado el sistema una vez concluido el desarrollo?
- 5- ¿Cuál sería el alcance que se espera con el sistema?
- 6- ¿Tienen las universidades en Cuba páginas en internet o en redes sociales?
- 7- ¿Qué privilegios podría otorgársele al administrador del sistema?

