Universidad de las Ciencias Informáticas

Facultad 1

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Sistema para la planificación de la Guardia Obrera Estudiantil de la Facultad 1

Autor:

Hector Alexey Garcés Rey

Tutores:

Prof. Aux. MSc. Geidis Sánchez Michel

Prof. Asist.Ing. Ernesto Soto Gómez

La Habana, 5 diciembre de 2022

Declaración de autoría

Declaro ser el único autor del trabajo de diploma *"*Sistema para la planificación de la Guardia Obrera Estudiantil de la Facultad 1*"*, concedo a la Universidad de las Ciencias Informáticas la autorización a hacer uso del mismo en su beneficio.

Para que conste firman el presente documento a los 5 días del mes de diciembre del año 2022.

Hector Alexey Garcés Rey

_____

Firma del autor

Prof. Aux. MSc. Geidis Sánchez Michel

_____

Firma del tutor

Prof. Asist.Ing. Ernesto Soto Gómez

_____

Firma del tutor

Pensamiento

“Tu trabajo va a suponer gran parte de tu vida, y la única forma de estar totalmente satisfecho es haciendo algo que creas que es un gran trabajo. Y la única forma de hacer un gran trabajo es hacer lo que amas”.

Steve Jobs

Agradecimientos

A mi abuela Elcida, porque solo Dios sabe el esfuerzo que he hecho para hacer de su sueño de verme graduado una realidad.

A mis padres, por su esfuerzo constante y apoyo incondicional.

A mi hermana, por guiarme en estos largos años de estudio.

A mis tías Sandra y Katia, por
ser mis segundas madres y actuar en consecuencia de ello.

A mi abuela
Teresa, por su amor y rezos en los momentos difíciles.

A mi novia, por su apoyo y resistencia a la distancia.

A mi tutora, por su ayuda y guía para hacer de este trabajo la obra que es hoy.

A mis familiares y amigos, por su constante preocupación y apoyo en la realización de este trabajo.

Resumen

En la Universidad de Ciencias Informáticas, específicamente en el Vicedecanato de Economía y Administración de la Facultad 1 se planifica y controla el proceso de guardia obrera estudiantil. Como parte del compromiso contraído y el deber revolucionario de salvaguardar los medios y recursos que convergen en el campus universitario, estudiantes y profesores se encuentran inmersos en el proceso de guardia. Actualmente, el proceso de gestión de la guardia obrera estudiantil resulta tedioso y se cometen continuamente errores humanos: duplicado de datos, falta de integración entre la planificación en la residencia y el docente, débil control en la asistencia diaria a la guardia y no se garantiza la disponibilidad de las planificaciones. En el presente documento se describe la investigación realizada en el proceso de planificación de la guardia obrera estudiantil con el objetivo de lograr una mayor informatización del proceso y evitar la duplicidad de la información en la planificación. Se desarrolló un sistema web multiplataforma capaz de unificar los datos y garantizar la integridad y consistencia de los mismos. Para el desarrollo del Sistema de Gestión de la Guardia Obrera Estudiantil en la Facultad 1 se usó como framework de desarrollo Django, lenguaje principal de programación Python y gestor de bases de datos SQLite.

Palabras Claves: control, guardia, planificación.

Abstract:

At the University of Informatics Sciences, specifically in the Vice-Deanship of Economics and Administration of Faculty 1, the student workers' guard process is planned and controlled. As part of the commitment made and the revolutionary duty to safeguard the means and resources that converge on the university campus, students and professors are immersed in the guard process. Currently, the management process of the student worker guard is tedious and human errors are continually made: duplication of data, lack of integration between the planning in the residence and the teacher, weak control in the daily attendance to the guard and the availability of schedules is not guaranteed. This paper describes the research carried out in the planning process of the student worker on-call with the aim of achieving greater computerization of the process and avoiding duplication of information in the planning. A multiplatform web system capable of unifying data and guaranteeing data integrity and consistency was developed. For the development of the Student Worker Guard Management System in Faculty 1, Django was used as the development framework, Python main programming language and SQLite database manager.

Keywords: control, guard, planning.

Índice de Contenidos

Índice de Tablas

Índice de Ilustraciones

Introducción

Uno de los grandes retos que tiene planteada la Universidad actual es lograr una mayor conexión entre la formación y el desarrollo profesional del alumnado (González, 2009). La enseñanza superior se encuentra en una constante transformación y ha experimentado cambios en todo el mundo. Según García “dichos cambios tienen que ver con sus dimensiones sustantivas, esto es, con lo académico, cuyo nivel de calidad se trata de preservar o mejorar. En países como Estados Unidos, Gran Bretaña, Francia y España, la gestión está hoy experimentando profundos cambios en cuanto a la forma de autoridad o gobierno, los mecanismos de financiamiento y las modalidades que asumen en cada caso la evaluación de la calidad. Aunque sobresalen por su importancia, estas no son desde luego las únicas dimensiones de la gestión, habrá que tener presente que, a nivel institucional, importa también la gestión de los asuntos académicos, de los sistemas de información, del personal, de la investigación, entre otros” (García, 2008).

Los procesos de cambio que se generan en la sociedad influyen en la educación superior, la revolución científico-tecnológica actual exige una mayor relación universidad-sociedad, nuevos retos socio-económicos, nuevas necesidades, pueden marcar el ritmo de adaptación de las instituciones universitarias.

Todos los universitarios en Cuba tendrán un vínculo con los escenarios productivos y de servicios con el objetivo de fortalecer la responsabilidad laboral, la cultura del trabajo y así lograr transformaciones en los propios estudiantes y la comunidad universitaria según (rebelde ,2009). La vinculación de las universidades cubanas a la práctica social, es una experiencia generalizada en Cuba. En este ámbito el sistema académico, laboral e investigativo interactúan contribuyendo a una mejor formación de los egresados de la Educación Superior, la Universidad de Ciencias Informáticas (UCI), no está exenta de estos vínculos.

La Universidad de las Ciencias Informáticas está estructurada por áreas de dirección y administración, departamentos docentes y no docentes en cada una de sus facultades, laboratorios docentes y de producción, una amplia residencia estudiantil con capacidad activa actual de más de 4 mil estudiantes, todo con un alto grado de informatización y recursos. Los estudiantes y trabajadores de la UCI comprometidos con el proceso revolucionario tienen como compromiso inexorable, el deber de cuidar y preservar los recursos de la universidad y personales. Es por eso que resulta vital garantizar la seguridad en el perímetro del centro, prestando profunda atención a la Guardia Obrera Estudiantil (GOE). En la

UCI se realiza con responsabilidad la GOE, como parte del deber y la necesidad de incorporar a estudiantes y trabajadores al cuidado y protección de los medios en el área de la universidad. La gestión y organización que incluye horarios, lugar donde se realiza la guardia, responsables de turnos, objetivos a cuidar, corresponden a los Vicedecanos de Economía y Administración en las facultades.

En la Facultad 1 en la UCI el proceso de planificación se encuentra dividido en dos subprocesos o bloques de planificación que tributan a áreas diferentes, el primer bloque destinado al docente de la facultad donde realizan guardia los profesores y estudiantes y un segundo bloque encargado del área residencial. El área de administración específicamente la Vicedecana de Economía y Administración es la encargada de organizar y generar los reportes de asistencia, así como generar la información necesaria para que el personal conozca detalladamente los eventos de guardia. Es importante conocer el potencial humano disponible y las características de cada uno (externo, interno, situaciones independientes) a la hora de confeccionar el plan de guardia para garantizar el éxito del evento.

Actualmente la planificación del proceso de GEO resulta tedioso y difícil de realizar. La distribución se realiza de forma manual y en formato Excel, por lo que esta es casi imposible de lograr, lo que se traduce en un empleo excesivo de tiempo y la posibilidad de errores humanos, ejemplo de este puede citarse uno de los más comunes, el cual consiste en que un estudiante tiene planificada dos veces en un mismo mes la guardia. No existe una forma segura de almacenar un historial de guardias ni de realizar un control de versiones de las planificaciones, otra dificultad a enfrentar es la falta de integración en los bloques de planificación.

Problema Investigación: ¿Cómo mejorar la gestión de la planificación y control de la Guardia Obrera Estudiantil en la Facultad 1 en la Universidad de Ciencias Informáticas?

El **objeto de estudio** de la presente investigación va dirigido al proceso de gestión de la guardia obrera estudiantil, y el **campo de acción**: gestión de la planificación y control de la guardia obrera estudiantil en la Facultad 1 en la Universidad de las Ciencias Informáticas.

Objetivo general:

Desarrollar un sistema para mejorar la gestión de la planificación y control de la Guardia Obrera Estudiantil en la Facultad 1 de la Universidad de Ciencias Informáticas.

Objetivos específicos:

- Definir el marco teórico de la investigación mediante el análisis de los referentes teóricos que sirvan de base para el desarrollo de la solución.

- Definir las herramientas y tecnologías a utilizar para el desarrollo del sistema, para mejorar la gestión de la planificación y control de la Guardia Obrera Estudiantil en la Facultad 1 en la Universidad de Ciencias Informáticas.
- Diseñar un sistema para mejorar la gestión de la planificación y control de la Guardia Obrera Estudiantil en la Facultad 1 en la Universidad de Ciencias Informáticas.
- Implementar un sistema para mejorar la gestión de la planificación y control de la Guardia Obrera Estudiantil en la Facultad 1 en la Universidad de Ciencias Informáticas.
- Validar que el sistema diseñado mejore la gestión de la planificación y control de la Guardia Obrera Estudiantil en la Facultad 1 en la Universidad de Ciencias Informáticas.

Preguntas Científicas:

- ¿Cuáles son los referentes teóricos relacionados con el uso de sistemas informáticos empleados en el desarrollo de sistemas de gestión de guardias?
- ¿Cuál es el estado actual de las herramientas informáticas estudiadas para el desarrollo del Sistema de Gestión de la Guardia Obrera Estudiantil en la Facultad 1?
- ¿Qué elementos deben tenerse en cuenta para llevar a cabo el análisis y diseño del Sistema de Gestión de la Guardia Obrera Estudiantil en la Facultad 1?
- ¿Cómo materializar, en términos de componentes y código fuente, los elementos definidos para el Sistema de Gestión de la Guardia Obrera Estudiantil en la Facultad 1?
- ¿Qué resultados se obtendrán al validar, a través de una estrategia de pruebas de software, el Sistema de Gestión de la Guardia Obrera Estudiantil en la Facultad 1?

Para dar respuesta a las preguntas científicas se definen como **tareas de investigación**:

1. Identificación de los procesos de control y planificación de la guardia obrera y estudiantil en la Universidad de Ciencias Informática.
2. Análisis de sistemas homólogos para el control y planificación de la guardia obrera y estudiantil.
3. Caracterización de las principales tendencias y tecnologías para el desarrollo de aplicaciones Web.
4. Captura de requisitos funcionales y no funcionales para el Sistema de Guardia Obrera Estudiantil en la Facultad 1.
5. Definición de la arquitectura de software para el Sistema de Guardia Obrera Estudiantil en la Facultad 1.

6. Propuesta de estándares de codificación software para el Sistema de Guardia Obrera Estudiantil en la Facultad 1.
7. Diseño de los casos de pruebas para validar el Sistema de Guardia Obrera Estudiantil en la Facultad 1.

Tomando como referente lo planteado por Rolando Alfredo Hernández en su libro “El paradigma cuantitativo de la Investigación Científica”(Hernández, 2008), se definen los siguientes métodos científicos:

Métodos Teóricos

Analítico-Sintético: para analizar la información obtenida y determinar las causas del problema existente en la planificación de la GOE, se revisaron libros de guardia y la documentación existente del sistema que se encuentra actualmente en uso.

Modelación: para modelar a partir del lenguaje UML las fases del diseño del sistema y mejorar la comprensión de la solución a implementar.

Métodos Empíricos:

Entrevista: constituyó una técnica de recopilación de información. Permitió definir claramente la situación problemática.

La observación: se observó detalladamente el proceso de planificación y control de la GOE, con el fin de determinar insuficiencias y posibles vías de solución a la situación problemática planteada.

Este documento está estructurado en 3 capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos.

En el capítulo 1: Fundamentación teórica de la investigación sobre la planificación de la GOE, se realiza una investigación para esclarecer los conceptos más importantes sobre la gestión de la GOE. Se efectuó un estudio sobre la GOE en Cuba, para entender las aplicaciones y procesos que lo integran. Además, se analizaron la metodología a utilizar y se exponen las características más generales respecto a las tecnologías a utilizar en el presente trabajo.

En el Capítulo 2: Propuesta de solución para la planificación de la GOE, se exponen las características de la aplicación, incluyendo los requisitos funcionales y no funcionales, patrones de diseño y arquitectura utilizados; además de los correspondientes artefactos que requiere la metodología de desarrollo utilizada.

En el Capítulo 3: Validación de la propuesta solución, se exponen los estándares de codificación, los cuales muestran las pautas que se deben seguir para una correcta implementación de la solución. Además, se presentan los casos de pruebas a utilizar en la validación del sistema y se analizan los resultados de las pruebas que permiten evaluar la calidad de la propuesta de solución.

Capítulo 1. Fundamentación teórica de la investigación sobre la planificación de la GOE

Introducción al capítulo.

En el presente capítulo se exponen los fundamentos teóricos que sustentan la presente investigación. Se describe de manera general el proceso de planificación y control de la GOE en las universidades y de forma particular en la Facultad 1 de la UCI. Se exponen las tecnologías a utilizar en el desarrollo del sistema y la metodología que guiará el proceso de software.

1.1 Principales conceptos a manejar

Las Indicaciones para la organización del servicio de seguridad y protección, circulado en marzo del 2022 por la dirección de defensa y seguridad de la UCI, remarcan la responsabilidad de la **GOE** en la custodia de los edificios docentes, la residencia y los edificios administrativos (Heredia, 2022).

La **GOE** es el proceso llevado a cabo en el Ministerio de Educación Superior (MES) mediante el cual los estudiantes y trabajadores de los centros de estudio de manera planificada velan por el cuidado y protección de los recursos humanos y materiales en sus respectivas instituciones. Las indicaciones mencionadas con anterioridad delegan al decanato de cada facultad de la universidad la máxima responsabilidad en la planificación de la GOE.

La **planificación** se puede definir como un proceso bien meditado y con una ejecución metódica y estructurada, con el fin de obtener un objetivo determinado, la planificación en un sentido un poco más amplio, podría tener más de un objetivo, de forma que una misma planificación organizada podría dar, mediante la ejecución de varias tareas iguales, o complementarias, una serie de objetivos. Cuanto mayor sea el grado de planificación, más fácil será obtener los máximos objetivos con el menor esfuerzo (La planificación, 2018).

La planificación es la estructuración de una serie de acciones que se llevan a cabo para cumplir determinados objetivos (Planificación – Economipedia, 2022). El investigador determina como idea esencial de los conceptos antes descritos que: la planificación es el conjunto de acciones bien estructuradas con el fin de obtener el máximo de objetivos con el menor esfuerzo posible.

De alguna manera se utiliza la planeación en la vida cotidiana, en sus diferentes expresiones: económicas, de programas gubernamentales, objetivos empresariales y procesos universitarios, entre otros. Dentro de estos procesos universitarios se encuentra la **guardia obrera estudiantil**.

El **control del proceso** de GOE se debe hacer diariamente y es responsabilidad de los oficiales de guardia realizar esta labor con la rigurosidad requerida, refiere el aspecto cuatro de las indicaciones mencionadas con anterioridad con respecto al servicio de protección y seguridad de la UCI (Heredia ,2022).

Fernández refiere que “la planificación y el control son inversamente proporcionales, es decir, donde hay una buena planificación el control del proceso resulta menos costoso y sencillo” (Fernández, 2018). Estos conceptos aportan un énfasis en el proceso de planificación con el fin de facilitar el control.

1.2 Flujo Actual del Proceso de GOE en la Facultad 1.

La Ilustración 1, modela los eventos fundamentales en el proceso de planificación de la GOE en la Facultad 1 en el momento de la realización de la investigación, comenzando por el levantamiento del potencial humano que se encuentra apto para la guardia y finalizando en el análisis de restricciones en el momento de ubicar a cada ente en su turno correspondiente.

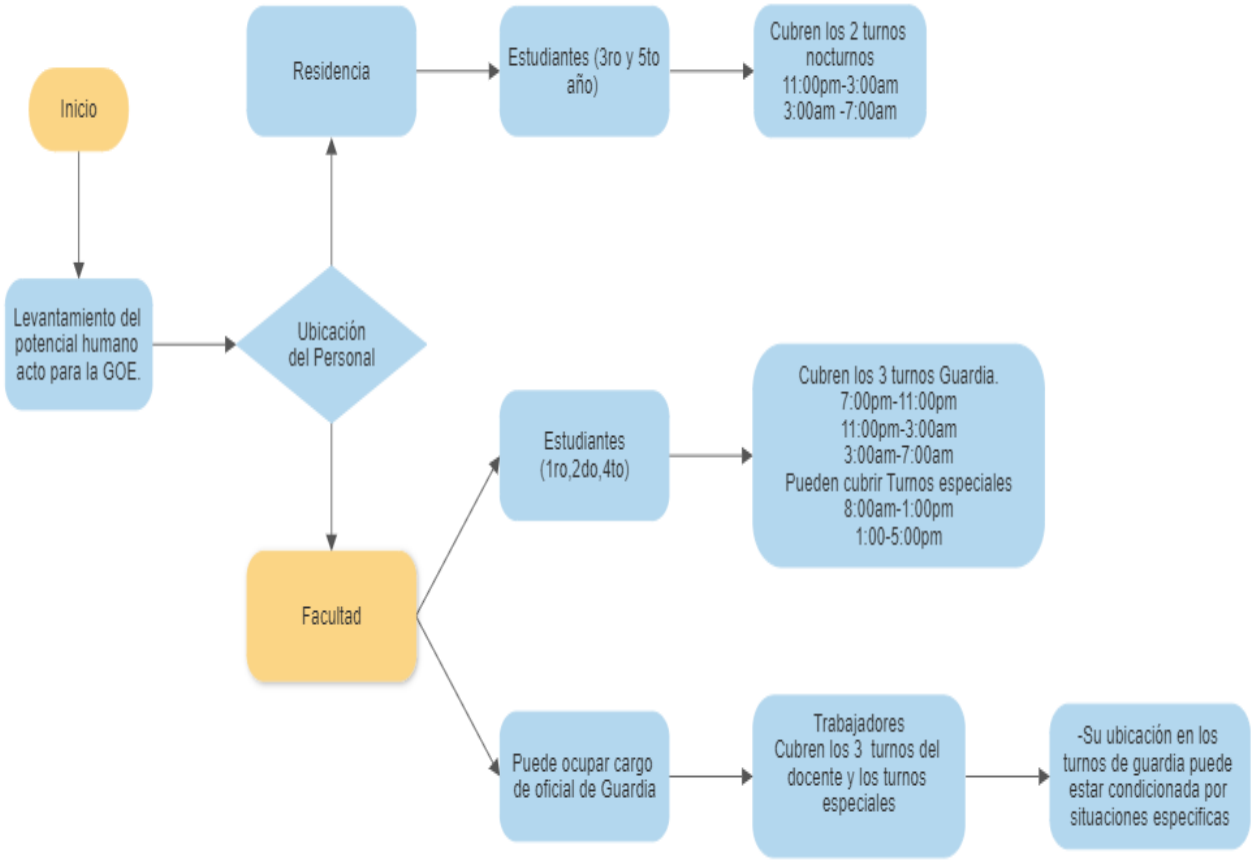


Ilustración 1 Flujo actual del proceso de GOE

1.3 Sistemas informáticos existentes vinculados al campo de acción.

Como parte de la investigación realizada se realizó un estudio de soluciones que resuelven problemas similares al problema de investigación planteado.

Se excluyeron de la búsqueda:

-Sistemas obsoletos.

-Sistemas elaborados en idiomas fuera del español y el inglés.

-Sistemas de los cuales no se pueda obtener información oficial.

La investigación realizada arrojó los siguientes sistemas homólogos:

Ámbito Internacionales:

Factorial: programa de gestión de turnos y cuadrante de horas de trabajo que organiza con eficacia la jornada de los empleados. Dentro de sus principales funcionalidades se encuentran (factorialhr ,2022):

-Asignar turnos de trabajo a los trabajadores.

-Advertir turnos ya asignados a un empleado o en un día.

-Gestionar las ausencias laborales.

aTurnos: es un software que mejora la gestión del personal. Ofrece un conjunto de herramientas que aumentan la productividad de recursos humanos y ayuda a las empresas a configurar equipos de trabajo, gestionar los permisos de acceso necesarios y analizar los datos generados. Dentro de sus principales funcionalidades se encuentran (aTurnos, 2022):

-Manejar restricciones laborales.

-Planificar diario o anualmente.

-Planificar turnos y manejo de restricciones de forma automática.

-Planificar el control de presencia.

Opsgenie: software de gestión de turnos y cuadrantes de horas de trabajo de acceso online. Su planificador de turnos permite introducir necesidades, particularidades y reglas de la empresa, como el calendario laboral, libranzas o demandas del personal. Dentro de sus principales funcionalidades se encuentran (Pgplanning, 2022)**:**

-Manejar variables para las asignaciones de turnos al personal.

-Controlar asistencia a los turnos asignados.

-Autogestionar planificación.

En el ámbito internacional los sistemas analizados tienen como característica que no son libres de pago por lo que va en contra de los principios de soberanía tecnológica definidos por la UCI.

Ámbito Nacional:

Sistema de planificación de la GOE en la Universidad de Holguín: sistema web desarrollado en 2010, como consecuencia de una tesis de culminación de estudios en la universidad de Holguín. Cuenta con módulos destinados a la gestión de la información de la GEO. Desarrollada en ZK Framework de la cual no se hace referencia a la versión de uso de esta herramienta en la documentación, cuenta con 23 requisitos funcionales y su desarrollo se encuentra guiado por la metodología RUP. Dentro de sus principales funcionalidades se encuentran (Almaguer, 2010):

-Planificar la GEO.

-Registrar el cumplimiento de guardia.

-Calcular % de asistencia.

Al hacer uso de ZK Framework, dentro de sus requisitos no funcionales de software se encuentra el uso de un navegador específico que cuente con un navegador capaz de interpretar código JavaScript.

Módulo Guardia Obrera Estudiantil versión 2.0 para el Sistema de Administración y Economía de la Facultad 3: sistema web desarrollado en 2016 como resultado de una tesis de culminación de estudio en la facultad 3 de la UCI. Dentro de sus principales características se encuentran(Morales, 2016) :

-Permitir generar la planificación de la guardia.

-Configurar las postas y los turnos.

-Realizar la planificación teniendo en cuenta criterios como: no repetir los turnos y planificar primero a los evaluados de M del mes anterior.

Para el desarrollo del mismo se utilizó como lenguaje de programación PHP y JQuery, como metodología XP, como IDE NetBeans y como Gestor de Base de Datos PostgreSQL.

Este sistema se centra solamente en el proceso de planificación de la guardia estudiantil y no permite realizar permutas de guardia.

Sistema para la gestión de la guardia obrera en la Facultad 1 en la Universidad de las Ciencias Informáticas: el sistema está desarrollado en Symfony 2.3 mediante PHP y MYSQL como sistema de administración de base de datos. Dentro de sus principales funcionalidades se encuentran(González, 2019):

-Gestionar la planificación de la GOE.

-Controlar la asistencia.

-Crear reportes de planificación.

Para un mejor análisis de los sistemas homólogos se procedió a la realización de una tabla comparativa (Tabla 1) teniendo en cuenta aspectos importantes para determinar si es posible hacer uso de alguno de los sistemas analizados como posible solución a la situación problemática. Para la confección de la tabla comparativa se hace uso de aspectos como licencia, que refiere al cobro por los servicios prestados por los sistemas, clasificándose en privada, para los sistemas en los cuales es necesario pagar servicios adquiridos, y pública para sistemas libres de costo y acceso. Se hace uso del aspecto dominio de la aplicación, en el cual se analiza específicamente si es un sistema web, es decir que puede utilizarse accediendo a un servidor web a través de internet o de una intranet. Código abierto, refiere a la disponibilidad de forma libre y sin costo adicional del código de programación, otorgando permisos para modificar su código fuente. El manejo de variables en las restricciones se traduce en la facilidad de manipular restricciones en las asignaciones del personal a los turnos en general, siendo necesaria la funcionalidad de crear nuevas restricciones para poder adaptar la solución de software a la situación problemática que se enfrenta en la investigación.

Tabla 1 Tabla comparativa de sistemas homólogos.

Sistemas	Licencia	Dominio de aplicación (web)	Código abierto	Manejo de Variables en las restricciones
Factorial	Privada	Si	No	Si
aTurnos	Privada	Si	No	Si
Opsgenie	Privada	Si	No	No
Sistema de planificación de la GOE en la Universidad de Holguín.	Pública	Si	Si	No

Módulo Guardia Obrera Estudiantil versión 2.0 para el Sistema de Administración y Economía de la Facultad 3	Pública	Si	Si	No
Sistema para la gestión de la guardia obrera en la Facultad 1 de la Universidad de las Ciencias Informáticas.	Pública	Si	Si	No

El estudio realizado permite determinar que existen varios sistemas de gestión de planificación de turnos y GOE. Los sistemas presentan similitudes en el proceso de gestión y control a partir de que se organizan estos procesos de forma similar en sus aspectos más generales.

Los sistemas internacionales investigados, en su totalidad no son libres de pago y no dan la posibilidad de acceso a su código fuente. En el ámbito nacional, pertenecen a la categoría de software libre, pero en sus procesos de planificación no contemplan las posibles restricciones con las que una persona puede contar en el momento de determinar su asignación a un turno de guardia (no realizar guardia el día antes de exámenes, duplicidad en la asignación de turno de guardia al personal, entre otros).

Teniendo en cuenta los resultados arrojados por la tabla comparativa y la entrevista realizada al cliente (anexo 1) se propone como propuesta de solución el desarrollo de un Sistema para la Guardia Obrera Estudiantil de la Facultad 1.

1.4 Análisis de metodologías de desarrollo de software.

Una metodología es una colección estructurada de procedimientos que ayudan a los desarrolladores de software en sus proyectos, ofreciendo una guía para la toma de decisiones, así como para planificarlo, gestionarlo, controlarlo y evaluarlo. La elección de la metodología a emplear es clave durante el desarrollo de un software por sus implicaciones en lo referente a efectividad, eficiencia y desempeño del

producto, costo y el tiempo de desarrollo, métodos de control de calidad y de pruebas, los cuales deben ajustarse a las particularidades de cada metodología(Velásquez et al., 2019). En la actualidad se pueden diferenciar dos grandes grupos de metodologías de desarrollo de software: las ágiles y las tradicionales (Santander, 2022).A continuación, la Ilustración 2 refleja una comparación entre las metodologías ágiles y tradicionales(Tinoco, 2017).

Metodologías ágiles	Metodologías tradicionales
Se basan en heurísticas provenientes de prácticas de producción de código	Se basan en normas provenientes de estándares seguidos por el entorno de desarrollo
Preparados para cambios durante el proyecto	Cierta resistencia a los cambios
Impuestas internamente por el equipo	Impuestas externamente
Proceso menos controlado, con pocos principios	Proceso muy controlado, numerosas normas
Contrato flexible e incluso inexistente	Contrato prefijado
El cliente es parte del desarrollo	Cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños (<10)	Grupos grandes
Pocos artefactos	Más artefactos
Menor énfasis en la arquitectura del software	La arquitectura del software es esencial

Ilustración 2 Comparación de metodologías.

A continuación, se describen las características de las principales metodologías ágiles:

Metodología Scrum

Es la metodología ágil más utilizada, diseñada para lograr la colaboración eficaz de los diferentes equipos relacionados con el proyecto. Se realizan entregas iterativas del producto al cliente para que lo pruebe y realice observaciones. Deben efectuarse reuniones diarias de máximo 15 minutos con el equipo de trabajo para coordinar el proyecto debidamente, y asignar roles claros a los miembros del equipo desarrollador(Velásquez et al. ,2019).

Programación extrema (XP)

La metodología se basa en cinco valores: simplicidad, comunicación, respeto y coraje para alcanzar retroalimentación rápida, simplicidad, cambio(Velásquez et al. 2019).

Proceso unificado abierto (OpenUP)

Esta metodología se caracteriza por el desarrollo incremental, uso de casos de uso y escenarios, manejo de riesgos, y diseño basado en la arquitectura (Velásquez et al. ,2019).

Proceso unificado ágil (AUP)

El AUP aplica técnicas ágiles incluyendo desarrollo dirigido por pruebas (TDD por sus siglas en inglés), modelado ágil, gestión ágil de cambios, y refactorización de base de datos para mejorar la productividad. El ciclo de vida del AUP tiene cuatro fases: concepción, donde se define el alcance y las arquitecturas para el software, elaboración en donde el equipo de desarrollo comprende los requisitos y valida la arquitectura del sistema, construcción en la cual el sistema es desarrollado y finalmente se despliega (Velásquez et al. ,2019).

La Universidad de Ciencias Informáticas (UCI) ha desarrollado una variación de la metodología AUP para guiar el proceso de desarrollo de software de la institución.

De las 4 fases que propone AUP (Inicio, Elaboración, Construcción, Transición), AUP-UCI mantiene la fase de Inicio, pero modifica el objetivo de la misma, se unifican las restantes 3 fases de AUP en una sola, la fase de Ejecución y se agrega la fase de Cierre de la siguiente manera (UCI, 2018):

Inicio: Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización del cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.

Ejecución: En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto.

Cierre: En esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

Teniendo en cuenta las características del equipo de desarrollo y del sistema a implementar, haciendo uso de la tabla comparativa (Ilustración 2) y la información que ofrece se decide el uso de una metodología ágil y adoptar dentro de estas la variante AUP-UCI en su escenario número 4, esta decisión esta propiciada por las características siguientes:

-Se cuenta con un equipo pequeño de desarrollo.

-El sistema a desarrollar es propenso a cambios constantemente.

-El cliente acompaña al equipo de desarrollo en la toma de decisiones y está en constante intercambio con los desarrolladores.

-Se cuenta con poco tiempo para el desarrollo por lo que se necesita generar la menor cantidad de artefactos posibles.

1.5 Herramienta CASE y lenguaje UML.

CASE: Son un conjunto de herramientas y métodos asociados que proporcionan asistencia automatizada en el proceso de desarrollo del software a lo largo de su ciclo de vida(Chicaiza, 2022).

Principales objetivos de su uso(Chicaiza, 2022):

- Mejorar la productividad del software
- Aumentar la calidad del software.
- Reducir el tiempo y costo de desarrollo y mantenimiento de los sistemas informáticos.
- Mejorar la planificación de un proyecto.
- Aumentar la biblioteca de conocimiento informático de una empresa ayudando a la búsqueda de soluciones para los requisitos.
- Automatizar el desarrollo del software, la documentación, la generación de código, las pruebas de errores y la gestión del proyecto.
- Ayuda a la reutilización del software, portabilidad y estandarización de la documentación.
- Gestión global en todas las fases de desarrollo de software con una misma herramienta.
- Facilitar el uso de las distintas metodologías propias de la ingeniería del software.

Con el objetivo de acelerar el proceso de desarrollo y mejorar la calidad del sistema a implementar se propone el uso de la siguiente herramienta CASE:

Visual Paradigm V.16.3: Es una herramienta CASE: Ingeniería de Software Asistida por Computación. La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación (Visual Paradigm,2022).

Unified Modeling Language (UML) V.2.5: El lenguaje de modelado unificado (UML) es un estándar para la representación visual de objetos, estados y procesos dentro de un sistema. Por un lado, el lenguaje de modelado puede servir de modelo para un proyecto y garantizar así una arquitectura de información estructurada; por el otro, ayuda a los desarrolladores a presentar la descripción del sistema

de una manera que sea comprensible para quienes están fuera del campo. UML se utiliza principalmente en el desarrollo de software orientado a objetos(Ionos, 2018).

Los principales beneficios de UML son(UML, 2017):

- Mejores tiempos totales de desarrollo (de 50 % o más).
- Modelar sistemas (y no sólo de software) utilizando conceptos orientados a objetos.
- Establecer conceptos y artefactos ejecutables.
- Encaminar el desarrollo del escalamiento en sistemas complejos de misión crítica.
- Crear un lenguaje de modelado utilizado tanto por humanos como por máquinas.
- Mejor soporte a la planeación y al control de proyectos.
- Alta reutilización y minimización de costos.

Como lenguaje de modelado se decide el empleo de UML mediante la herramienta Visual Paradigm, sustentado en la necesidad de diseñar el sistema a implementar y contar con una representación gráfica de lo que posteriormente se va a implementar.

1.6 Propuesta de herramientas y lenguajes de desarrollo.

Cuando se habla de una aplicación web, se hace referencia a una aplicación que no necesariamente está instalada en la computadora del cliente, sino que se encuentra en otro equipo denominado servidor. El servidor puede ser local , así como también podría llegar a encontrarse en un lugar muy diferente y a una distancia muy lejana al cliente , es decir en otro lugar del mundo(Pacheco, 2016).

Las aplicaciones web para internet e intranet presentan una serie de ventajas y beneficios con respecto al software de escritorio, con lo cual se logra aprovechar y acoplar los recursos de una empresa de una forma mucho más práctica que el software tradicional (InternetYA, 2016).

Beneficios:

• El trabajo a distancia se realiza con mayor facilidad.

• Para trabajar en la aplicación web se necesita una computadora con un buen navegador web y conexión a internet.

.• Con una aplicación web se tiene total disponibilidad en cuanto a hora y lugar, podrá trabajar en ella en cualquier momento y en cualquier lugar del mundo siempre que se tenga conexión a internet. Ventajas:

- Las aplicaciones web tienen un camino mucho más sencillo para la compatibilidad multiplataforma.

- Actualización. Las aplicaciones basadas en web están siempre actualizadas con el último lanzamiento.

- Inmediatez de acceso. Las aplicaciones basadas en web no necesitan ser descargadas, instaladas y configuradas. Se pueden acceder de manera online para trabajar sin importar cuál es su configuración o hardware.

La solución a desarrollar se basará en este tipo de sistemas, los principales motivos por los que se procede a esta elección es la compatibilidad a múltiples sistemas operativos y la experiencia del equipo de desarrollo en la implementación de estos sistemas.

1.6.1 Selección del framework de desarrollo.

Los frameworks son un conjunto de archivos y directorios que se utilizan en el desarrollo de aplicaciones. Basado en un lenguaje de programación, los frameworks pueden entenderse como herramientas muy útiles que facilitan el trabajo de desarrollo de aplicaciones(Centro Europeo de Posgrados, 2021).

Ventajas de su uso en el desarrollo de aplicaciones(Agency, 2021):

- No es necesario crear una estructura desde cero.
- Evita escribir código repetitivo como puede ser accesos a las bases de datos, validaciones de formularios, entre otros.
- Permite acortar los plazos de entrega.
- Garantiza editar el trabajo de una manera más sencilla en el largo plazo, ya que dispone de una base estandarizada.
- Facilita localizar utilidades, librerías, y así adaptarlas al framework.

Aunque el uso de esta herramienta no es obligatorio en el desarrollo de aplicaciones, los aportes que este puede ofrecer son incuestionables. En la actualidad las aplicaciones web en su mayoría están respaldadas por esta clase de herramientas.

Las condiciones del sistema a desarrollar en la presente investigación ameritan el uso de un framework de desarrollo, la decisión se basa principalmente en el ahorro de tiempo de implementación, la amplia documentación existente sobre las principales herramientas de esta categoría, la experiencia acumulada por el equipo de desarrollo en el uso de estas herramientas y el manejo fácil y sencillo de la base de datos en las aplicaciones respaldadas por frameworks.

Tomando como base el conocimiento que el equipo de desarrollo posee sobre Django, y con la intención de reducir tiempo y esfuerzo en el estudio de un nuevo framework, se procede a la elección de este como herramienta base para el desarrollo de la aplicación web a implementar en la presenta investigación.

Django es un marco web de alto nivel que fomenta un desarrollo rápido, 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 ,.2022).

Django V.4.1.3 se puede resumir que tiene muchas facilidades en las cuales las ventajas principales son(Aguilar Ordonez, 2016):

- Es rápido de desarrollar.
- Es modular.
- Tiene muy bajo acoplamiento.
- Genera automáticamente un panel de administrador.
- Sus bibliotecas hacen gran parte del trabajo.
- Soporta varias bases de datos.

1.6.2 Selección de Bootstrap V.5

Es un framework de desarrollo web gratuito y de código abierto. Está diseñado para facilitar el proceso de desarrollo de los sitios web responsivos y orientados a los dispositivos móviles, proporcionando una colección de sintaxis para diseños de plantillas(Deyimar, 2020).

Ventajas en el uso de Bootstrap V.5 (Colmena, 2020):

- Permite acceder de manera gratuita, a una gran comunidad de desarrolladores de código abierto que contribuyen con Bootstrap.
- Abundante diversidad de complementos y componentes.
- Diseño web consistente, con coherencia entre las páginas web y las aplicaciones.
- Las imágenes vienen con su propio código para adaptarlas automáticamente según el tamaño de la pantalla.
- Es compatible con la mayoría de los navegadores.

Se tuvo en cuenta para su selección: el ahorro considerable de tiempo que este provoca, la diversidad en sus componentes, la experiencia del equipo de desarrollo en el uso de esta herramienta y las bondades que ofrece en la implementación visual de los sistemas web.

1.6.3 Selección de lenguajes de programación.

Python V.3.10.4: es un lenguaje de programación potente y fácil de aprender. Tiene estructuras de datos de alto nivel eficientes y un simple pero efectivo sistema de programación orientado a objetos. La elegante sintaxis de Python y su tipado dinámico, junto a su naturaleza interpretada lo convierten en un lenguaje ideal para scripting y desarrollo rápido de aplicaciones en muchas áreas, para la mayoría de plataformas(docsPython ,2022).

Principales ventajas que provee el uso de Python V.3.10.4 (KeepCoding, 2021):

- Es un lenguaje de alto nivel, por lo que resulta más fácil su comprensión.
- Se le considera un lenguaje de paradigmas múltiples, que admite programación estructurada, funcional y orientada a objetos.
- Amplia colección de bibliotecas y frameworks.
- Es compatible con todos los sistemas operativos.
- Es un lenguaje de programación desarrollado bajo la licencia de código abierto aprobada por OSI, todos pueden usarlo y distribuirlo libremente.
- La sencillez de la sintaxis de Python permite escribir programas totalmente funcionales en pocas líneas de código, por lo que su curva de aprendizaje es muy baja.

La selección de Python como lenguaje de programación está condicionada por la elección de Django como framework de desarrollo y su basamento en este lenguaje, sumado a la experiencia acumulada por el equipo de desarrollo en el mencionado lenguaje.

CSS V.3: Las siglas CSS (Cascading Style Sheets) significan «Hojas de estilo en cascada» y parten de un concepto de aplicar estilos (colores, formas, márgenes) a uno o varios documentos (generalmente documentos HTML) de forma automática y masiva. Se le denomina estilos en cascada porque se lee, procesa y aplica el código desde arriba hacia abajo(Manz, 2020).

Ventajas del uso de CSS para generar estilos(Manz, 2020) :

- Si se necesita hacer modificaciones visuales, se hacen en un sólo lugar y se aplica a todo el sitio.

- Se reduce la duplicación de estilos en diferentes lugares. Es más fácil de organizar y hacer cambios.
- La información a transmitir es considerablemente menor (las páginas se descargan más rápido).

Los sistemas web actuales, generalmente usan estilos CSS, este lenguaje goza de gran popularidad entre los maquetadores y diseñadores de software, por lo que se decide hacer uso de sus beneficios en el desarrollo de la solución a implementar.

HTMLV.5: HTML es un lenguaje de marcado que permite la estructuración de información y contenido en un documento o sitio web. El marcado se ejecuta a través de etiquetas que cumplen diferentes funciones en la estructuración del documento. Por ejemplo, hay etiquetas para la organización del contenido, etiquetas para incrustar contenido, etiquetas para crear tablas(KeepCoding, 2021).

Ventajas del uso de HTML (Gutiérrez, 2018):

- Es un lenguaje sencillo, fácil de aprender, leer e interpretar.
- Existen numerosas aplicaciones y editores de páginas web (WYSIWYG) que generan el código automáticamente, por lo que no es necesario ser un experto informático para hacer páginas basadas en HTML.
- Su sencillez hace que pueda diseñarse y desplegarse un sitio web en muy poco tiempo.
- Es el lenguaje más extendido, todos los navegadores lo admiten.
- Código visible e interpretable por los buscadores.

El uso de HTML como lenguaje de maquetado en el desarrollo de la solución tributa al estándar del W3C el cual establece la compatibilidad de los sitios web con los principales navegadores y define a HTML como un lenguaje de reconocimiento universal.

JavaScript V.ES12: JavaScript es un lenguaje de programación que los desarrolladores utilizan para hacer páginas web interactivas. Desde actualizar fuentes de redes sociales a mostrar animaciones y mapas interactivos, las funciones de JavaScript pueden mejorar la experiencia del usuario de un sitio web. Como lenguaje de scripting del lado del servidor, se trata de una de las principales tecnologías de la World Wide Web(AWS, 2022).

Ventajas en el uso de JavaScript:

- Tiende a ser muy rápido porque a menudo se ejecuta inmediatamente en el navegador.

- A diferencia de PHP u otros lenguajes scripting, JavaScript puede ser usado en cualquier página web.
- Puede ser usado en diferentes tipos de aplicaciones gracias al soporte en otros lenguajes como Pearl y PHP.
- Es client-side, entonces esto reduce la demanda de servidores en general, y las aplicaciones sencillas puede que no necesiten el servidor del todo.
- Puede ser usado para crear características como arrastrar y soltar, y componentes tales como las diapositivas, lo cual mejora enormemente la interfaz de usuario y la experiencia del sitio.

Al usar librerías como JQuery para los efectos y eventos de la aplicación web a implementar sumado a los disimiles beneficios con los que cuenta JavaScript, se propone su uso como lenguaje de script del lado del cliente.

1.6.4 Entorno de desarrollo

Visual Studio Code V.1.67: Editor de código fuente independiente que se ejecuta en Windows, macOS y Linux. La elección principal para desarrolladores web y JavaScript, con extensiones para admitir casi cualquier lenguaje de programación (Visual Studio,2022).

VS Code tiene una gran variedad de características útiles para agilizar el trabajo, que lo hacen el editor preferido por muchos para trabajar los proyectos.

Principales ventajas y características de su uso(OpenWebinars, 2022):

- Multiplataforma: está disponible para Windows, GNU/Linux y macOS.
- IntelliSense: esta característica está relacionada con la edición de código, autocompletado y resaltado de sintaxis, lo que permite ser más ágil a la hora de escribir código.
- Depuración: incluye la función de depuración que ayuda a detectar errores en el código.
- Es capaz de detectar pequeños errores de forma automática antes de ejecutar el código o la depuración como tal.
- Uso del control de versiones: Visual Studio Code tiene compatibilidad con Git, por lo que puedes revisar diferencias o lo que se conoce como git diff, organizar archivos, realizar commits desde el editor, y hacer push y pull.

- Extensiones: permiten personalizar y agregar funcionalidad adicional de forma modular y aislada, para programar en diferentes lenguajes, agregar nuevos temas al editor, y conectar con otros servicios.

La selección de un editor de texto para el desarrollo de un sistema web viene más enfocado a la comodidad del equipo de desarrollo, se selecciona VS Code por las buenas experiencias acumuladas en este entorno por los desarrolladores de la solución a implementar, su buena compatibilidad con Git, su cúmulo de extensiones en beneficio de la comodidad de los programadores y por ser un editor de código abierto y libre de pagos por sus funcionalidades.

1.7 Sistema gestor de base de datos

Un sistema gestor de base de datos (SGBD) o Database Management System (DBMS) es un conjunto de programas invisibles para el usuario final con el que se administra y gestiona la información que incluye una base de datos. Los gestores de datos o gestores de base de datos permiten administrar todo acceso a la base de datos, pues tienen el objetivo de servir de interfaz entre esta, el usuario y las aplicaciones. Entre sus funciones se encuentran la de permitir a los usuarios de negocio almacenar la información, modificar datos y acceder a los activos de conocimiento de la organización. Asimismo, el gestor de base de datos también se ocupa de realizar consultas y hacer análisis para generar informes. Además, los sistemas de gestión de base de datos pueden entenderse como una colección de datos interrelacionados, estructurados y organizados en el ecosistema formado por dicho conjunto de programas que acceden a ellos y facilitan su gestión(Intelequia, 2021).

Un SGBD permite definir los datos, además de manipularlos, aplicar medidas de seguridad e integridad y recuperarlos o restaurarlos después de producirse algún tipo de fallo. Algunas de las funciones principales de los gestores de bases de datos son las siguientes (Darias, 2021):

- Contribuyen a la creación de bases de datos más eficaces y consistentes.
- Determinan las estructuras de almacenamiento del sistema.
- Facilitan las búsquedas de datos de cualquier tipo y procedencia a los usuarios de negocio.
- Ayudan a mantener la integridad de los activos informacionales de la empresa.
- Introducen cambios en la información, si es requerido.
- Simplifican los procesos de consulta.
- Controlan los movimientos que se observan en la base de datos.

En la solución a desarrollar se manejan datos, los cuales se necesitan guardar para futuras consultas o edición de la información registrada, por lo que se hace necesario el uso de un sistema gestor de bases de datos, se procede por parte del equipo de desarrollo a la selección del mencionado sistema.

SQLite V.3.40.0: es una biblioteca en lenguaje C que implementa un motor de base de datos SQL pequeño, rápido, autónomo, de alta confiabilidad y con todas las funciones. SQLite es el motor de base de datos más utilizado en el mundo. SQLite está integrado en todos los teléfonos móviles y en la mayoría de las computadoras y viene incluido dentro de innumerables otras aplicaciones que la gente usa todos los días(SQLite, 2022).

Principales características de SQLite V.3.40.0:

- Multisistema. La biblioteca SQLite se incluye en los sistemas operativos más utilizados, como Windows, Linux, Android y los sistemas de Apple (iOS y macOS).
- Código abierto. El uso de SQLite no requiere del pago de una licencia.
- Multilenguaje. Dispone de diferentes API que le permiten trabajar con lenguajes de programación como C++, Python o PHP, entre otros.
- Soporta múltiples tablas, índices y vistas.
- Sencillez. SQLite dispone de una API que es muy simple, por lo que su uso es muy fácil y no requiere de grandes conocimientos técnicos.
- Autonomía. No tiene dependencias externas.

Como gestor de bases de datos se selecciona SQLite V.3.40.0, la elección está sustentada en la integración de Django con este gestor. Django como farmework seleccionado para la implementación del sistema cuenta con todas las conexiones por defecto para la gestión de datos con SQLite, durante la implementación de los modelos de datos no se hará uso de consultas complejas, por lo que se descarta el uso de gestores más pesados y complejos como PostgreSQL o Oracle.

1.8 Herramientas de prueba

Las herramientas de prueba de software automatizadas ayudan a ejecutar pruebas funcionales y de regresión en la aplicación. Estas herramientas deben producir resultados consistentes con los datos entrantes proporcionados(Khatri, 2020).

JMeter V.2.3.1: es un software de código abierto, una aplicación Java 100 % pura diseñada para cargar, probar el comportamiento funcional y medir el rendimiento. Originalmente fue diseñado para probar

aplicaciones web, pero desde entonces se ha expandido a otras funciones de prueba (Jmater, 2022).

Veneficios del uso de JMeter(Andalucía, 2019):

- Ofrece parámetros para realizar pruebas de rendimiento, carga y regresión con facilidad y eficiencia.
- Existe excelente documentación disponible que facilita la curva de aprendizaje de la herramienta
- Posee la capacidad de soporte para múltiples hilos.
- Ofrece la posibilidad de representar gráficas de rendimiento.

Burp Suite V.2022.9: es una plataforma integrada para la realización de las pruebas de seguridad de las aplicaciones web. Sus herramientas funcionan perfectamente juntas para apoyar todo el proceso de pruebas, de identificación y análisis de la superficie de ataque de una solicitud inicial, a través de la búsqueda y explotación de vulnerabilidades de seguridad(Tarlogic, 2021).

Principales funcionalidades(Tarlogic, 2021):

• Proxy de interceptación tráfico de navegación (HTTP y HTTPs).
• Módulo de descubrimiento e indexación de contenido (Spider).
• Escáner de aplicaciones Web.
• Análisis de sesiones.
• Soporte de plugins (extensiones).

Teniendo en cuenta la experiencia del equipo de desarrollo en el uso de estas herramientas de prueba y la planificación de las posibles pruebas de software a realizar se procede a la elección de JMeter como herramienta para pruebas de carga y estrés y Burp Suite para detectar vulnerabilidades de seguridad tanto en el sistema como en la base de datos a implementar.

Conclusiones parciales del capítulo.

La elaboración de los referentes teóricos-metodológicos relacionados con el proceso de GOE permitió sentar las bases de la investigación realizada. El análisis de los sistemas homólogos al proceso de GOE permitió concluir la necesidad de elaborar un sistema para la GOE en la Facultad 1.

Al identificar como solución la creación de un sistema, se evidenció la necesidad de definir las herramientas y tecnologías. Se seleccionó como metodología para guiar el desarrollo AUP-UCI, como framework de desarrollo Django V.4.1.3, como entorno de desarrollo VS Code V.1.67 y lenguajes de

programación: Python V.3.10.4, HTML V.5, CSS V.3 y JavaScript V.ES12. Como herramienta CASE para realizar el modelado de del software se identificó Visual Paradigm V.16.3, haciendo uso de UML V.2.5 como lenguaje de modelado. Se seleccionó como gestor de bases de datos SQLite 3.40.0, como herramientas de pruebas JMeter V.2.3.1 para las pruebas de tipo carga y estrés, sumada a Burp Suite V.2022.1 para detectar errores de seguridad en el sistema y la base de datos a implementar, sentando las bases tecnológicas para la propuesta de solución.

Capítulo 2. Propuesta de solución para la planificación de la GOE

Introducción al capítulo

Como parte del ciclo de vida del software a desarrollar, en el presente capítulo se procede a la modelación de la solución y la generación de artefactos como parte de la metodología de software escogida, se representa el mapa conceptual con los principales conceptos del negocio, se procede a realizar el levantamiento de requisitos a partir de la propuesta de solución descrita, se plantea la arquitectura, modelo de datos y los patrones de diseño usados en el desarrollo del sistema.

2.1 Mapa conceptual

Un mapa conceptual es un diagrama que ayuda a entender y/o explicar un tema específico al realizar conexiones visuales entre elementos que conforman dicho tema(Canvas, 2022).La ilustración 3 expone el mapa conceptual del proceso de GOE en la Facultad 1.

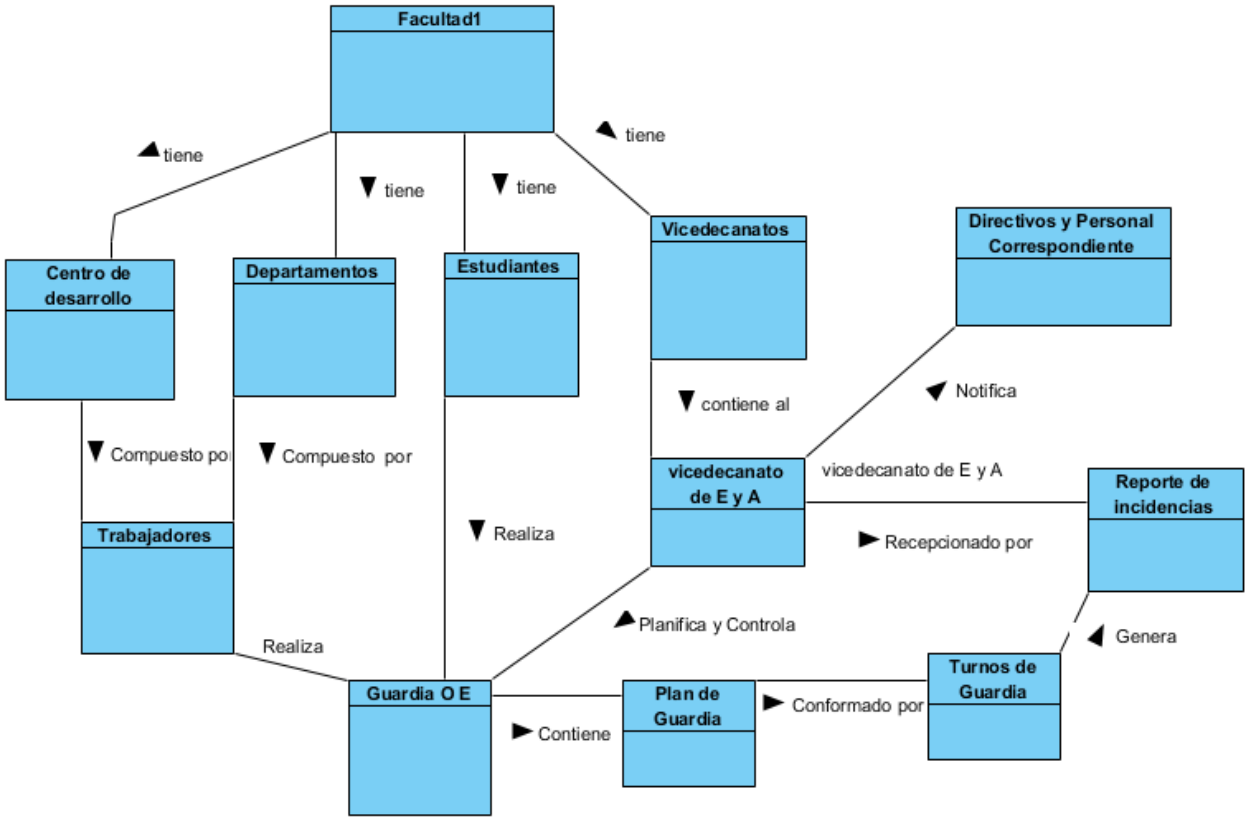


Ilustración 3 Mapa Conceptual del proceso de GOE

Los conceptos representados se describen a continuación:

- Facultad 1: lugar al que pertenece el cliente, está compuesto por centros de desarrollo, departamentos y vicedecanatos.
- Estudiantes: forma parte de la facultad y pertenecen al potencial humano (PH) apto para la guardia.
- Trabajadores: forma parte de los centros de desarrollo y los departamentos, pertenecen al PH apto para la guardia.
- Vicedecano de Economía y Administración: es el cliente del negocio, pertenece a los vicedecanatos y forma parte del PH apto para la guardia.
- Guardia Obrera Estudiantil: Es planificado y controlado por el vicedecano de Economía y Administración, es el componente principal del negocio.
- Plan de Guardia: recoge la planificación de la GOE.
- Turnos de Guardia: horarios en los que se realiza la GOE, forma parte del plan de guardia.
- Reportes de Incidencia: recoge las incidencias identificadas en los turnos de guardia y los ausentes a la guardia.
- Directivos y personal correspondiente: personas que por su responsabilidad en la facultad hacen uso de la información contenida en los reportes de incidencia.

2.2 Propuesta de solución.

Se propone el desarrollo de un sistema capaz de dar solución a las principales problemáticas afrontadas en la elaboración de la panificación de la GOE. El sistema propuesto será capaz de almacenar disímiles datos que son de interés y análisis en el momento de otorgar un turno de guardia a una persona, dígase: última guardia realizada, ubicación residencial de la persona, pruebas a realizar en el mes, además de los datos comunes de registro: nombre, apellidos, año académico (en el caso de los estudiantes).

El sistema debe ser capaz de generar un horario de forma asistida, haciendo uso de las restricciones de cada persona en el momento de otorgar un turno, una vez realizada la planificación se podrá editar, así como intercambiar personas entre turnos, dicha planificaciones debe poder exportarse a formato Exel para su posterior envío al personal. El administrador podrá realizar consultas a un historial de guardia en caso de que surja la necesidad, así como se podrá introducir los incumplidores y enviar esta información al personal pertinente. La experiencia de interfaz de usuario debe ser amigable y predecible.

El análisis detallado de la propuesta da paso al posterior levantamiento de requisitos.

2.3 Ingeniería de Requisitos

Desde la perspectiva del proceso del software, la ingeniería de requisitos es una de las acciones importantes de la ingeniería de software que comienza durante la actividad de comunicación y continúa en la de modelado. Debe adaptarse a las necesidades del proceso, del proyecto, del producto y de las personas que hacen el trabajo (Pressman, 2010).

2.3.1 Requisitos funcionales

Los requisitos funcionales son declaraciones de las funcionalidades que debe cumplir el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares (Pressman, 2010). La tabla número tres refleja los requisitos funcionales definidos para el sistema a desarrollar.

Tabla 2 Requisitos funcionales del sistema.

# RF	Nombre	Descripción	Complejidad	Prioridad
1	Autenticar usuario	El sistema debe permitir el acceso a sus funcionalidades solo al personal que cuente con credenciales para su uso (Usuario y Contraseña).	Baja	Baja
2	Adicionar potencial humano (PH)	Funcionalidad que adiciona al sistema el personal apto para realizar guardia (estudiantes, trabajadores).	Alta	Alta
3	Eliminar PH	El sistema debe ser capaz de eliminar en caso de ser requerido miembros de la GOE.	Baja	Alta
4	Listar PH	El sistema debe listar los miembros de la GOE.	Baja	Alta
5	Editar PH	El sistema debe editar la información de los miembros de la GOE.	Baja	Alta

6	Crear turnos de guardia	El sistema debe crear turnos de guardia regido a las restricciones de asignación de los miembros de la GOE.	Alta	Alta
7	Asignar roles al PH	El sistema debe ser capaz de asignar roles a los trabajadores.	Alta	Alta
8	Añadir incumplidores de la guardia.	El sistema debe registrar los ausentes a la GOE.	Media	Media
9	Modificar incumplidores de guardia	El sistema debe modificar información de los miembros incumplidores de la GOE.	Media	Media
10	Eliminar incumplidores de guardia	El sistema debe eliminar miembros incumplidores de la GOE en caso de ser necesaria esta acción.	Media	Media
11	Notificar incumplimiento de la guardia al usuario implicado	El sistema debe realizar reportes de incumplimientos y enviarlo al personal correspondiente.	Alta	Media
13	Crear reportes de guardia.	El sistema debe crear reportes de guardia, donde se registren incidencias en los turnos de guardia y el personal ausente.	Media	Media
14	Permutar guardia	El sistema debe intercambiar turnos de guardia.	Alta	Media
15	Consultar historial de guardia.	El sistema debe guardar histórico de guardia para su futuro acceso.	Alta	Media
16	Editar reportes de guardia.	El sistema debe ser capaz de editar la información de los turnos de guardia.	Media	Media

17	Crear plan de guardia	El sistema debe reunir todos los planes de guardia del mes y agruparlos en un plan de guardia mensual.	Media	Media
18	Crear restricciones	El sistema debe registrar las restricciones de asignación de guardia.	Alto	Alto
19	Añadir correo	El sistema debe ser capaz de añadir los correos de los destinatarios de los reportes de guardia.	Baja	Baja
20	Salir de la sesión de trabajo	El sistema debe contar con la funcionalidad de cerra la sesión de trabajo.	Baja	Baja

2.3.2 Requisitos no funcionales

Los requisitos no funcionales son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo, estándares, entre otros (Sommerville, 2005).

Apariencia

- El color predominante en el diseño debe ser blanco y azul en tonos suaves y claros, para compatibilidad con los colores característicos del logotipo de la universidad.

-Debe tener visible el logotipo de la Facultad 1 en todas las interfaces.

Usabilidad

- Debe haber una facilidad de uso por usuarios de cualquier nivel, presentando las funcionalidades visibles en todo momento que deben facilitar la navegación.

- El diseño debe ser adaptable a diferentes resoluciones de pantalla.

- Toda funcionalidad del sistema de gestión tecnológica debe responder al usuario en menos de 5 segundos.

- La información almacenada debe ser confiable en cuanto a su veracidad e integridad desde su recopilación a través de la validación de los formularios

Seguridad:

-El usuario debe autenticarse para acceder al sistema.

-Se debe hacer una copia de seguridad de la base de datos de la aplicación.

Portabilidad:

El sistema es multiplataforma gracias a las bondades de usar Django como framework de desarrollo.

Software:

-Se debe contar con acceso a internet en el navegador para una mejor experiencia visual puesto que los paquetes de diseños serán de MDBootstrap.

- Django V.4.1.3 -Python 3.10.4

- Se necesita instalado en la estación de trabajo un sistema operativo: Linux, Windows 7 o superior

Hardware:

El servidor debe tener como mínimo un procesador Intel Core, con 2 GB de memoria RAM que son los requisitos mínimos para el despliegue de aplicaciones en Django.

La PC cliente debe contar como mínimo con 2 GB de RAM.

2.4 Historias de Usuario

Dentro de los artefactos que genera el escenario cuatro de la metodología AUP-UCI se encuentran las historias de usuario (HU). Este artefacto forma parte del enfoque ágil y describe las características y necesidades de un software desde la perspectiva de un usuario, ayudando a alinear expectativas y evitar errores críticos en el futuro. Una historia de usuario puede considerarse como una preparación para establecer los requisitos del software(Vergara, 2021).

En esta sección se hace alusión a las HU de los requisitos RF 2, RF 6 y RF 18 (ver tabla de la 3 a la 5), las cuales hacen referencia a las funcionalidades que se consideran esenciales por su alto grado de complejidad y prioridad en el desarrollo de la solución.

Tabla 3 HU número 2 Adicionar PH

Historia de Usuario	
Número: 2	**Nombre del requisito:** Agregar trabajadores PH
Programador: Hector Garcés Rey	**Iteración asignada:**1
Prioridad: Alta	**Tiempo estimado:** 94horas
Riesgo en desarrollo: Alto	**Tiempo Real:** 96 horas
Descripción: siguiendo el levantamiento del potencial apto para la guardia, entregado por los jefes de áreas se deben añadir, listar, eliminar, editar el potencial humano activo. Esta funcionalidad debe generar una lista.	
Observaciones: esta funcionalidad es la base para el buen funcionamiento el sistema.	
Estereotipo:	

Ilustración 4 Funcionalidad Agregar Trabajadores.

Tabla 4 HU número 6 Crear Turnos de Guardia

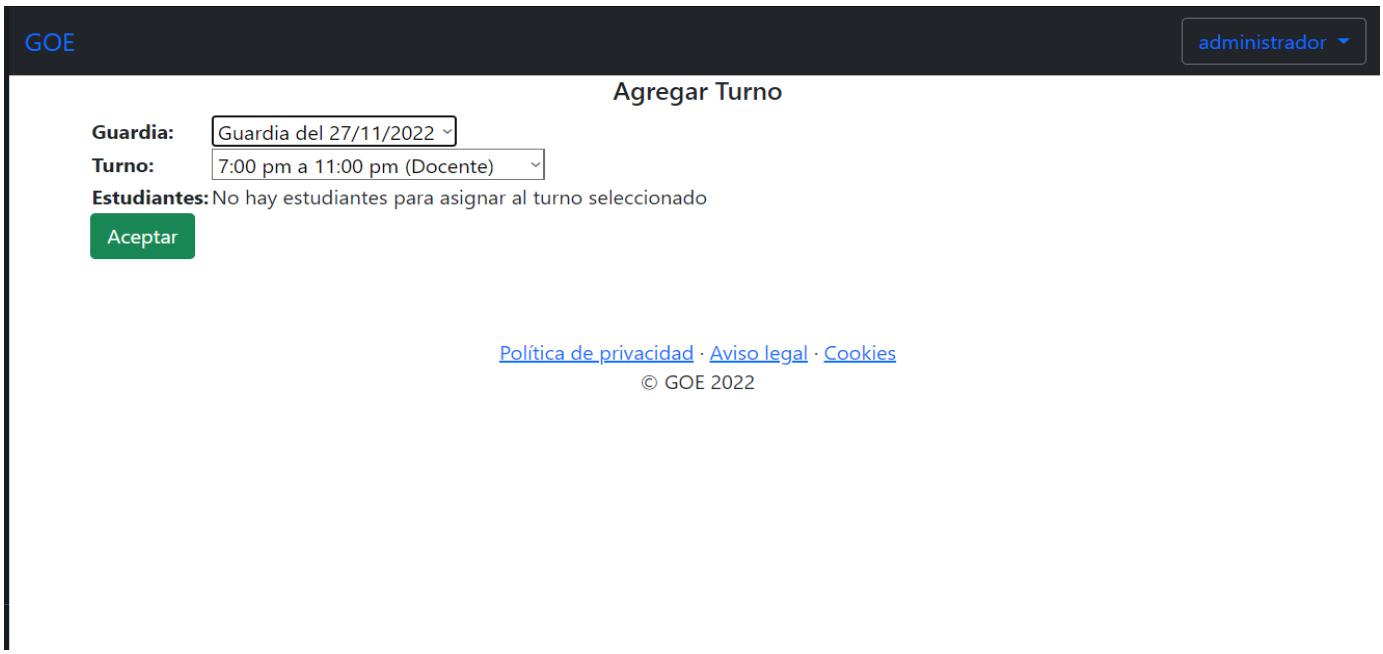
Historia de Usuario	
Número: 6	**Nombre del requisito:** Crear Turnos de Guardia
Programador: Hector Garcés Rey	**Iteración asignada:3**
Prioridad: Alta	**Tiempo estimado:** 120 horas
Riesgo en desarrollo: Alta	**Tiempo Real:** 120 horas
Descripción: el sistema debe crear turnos de guardia regido a las restricciones de asignación de los miembros de la GOE.	
Observaciones: esta funcionalidad es la esencia del sistema	
Prototipo: 	

Ilustración 5 Prototipo funcionalidad crear turnos de guardia

Tabla 5 HU número 18 Crear restricciones.

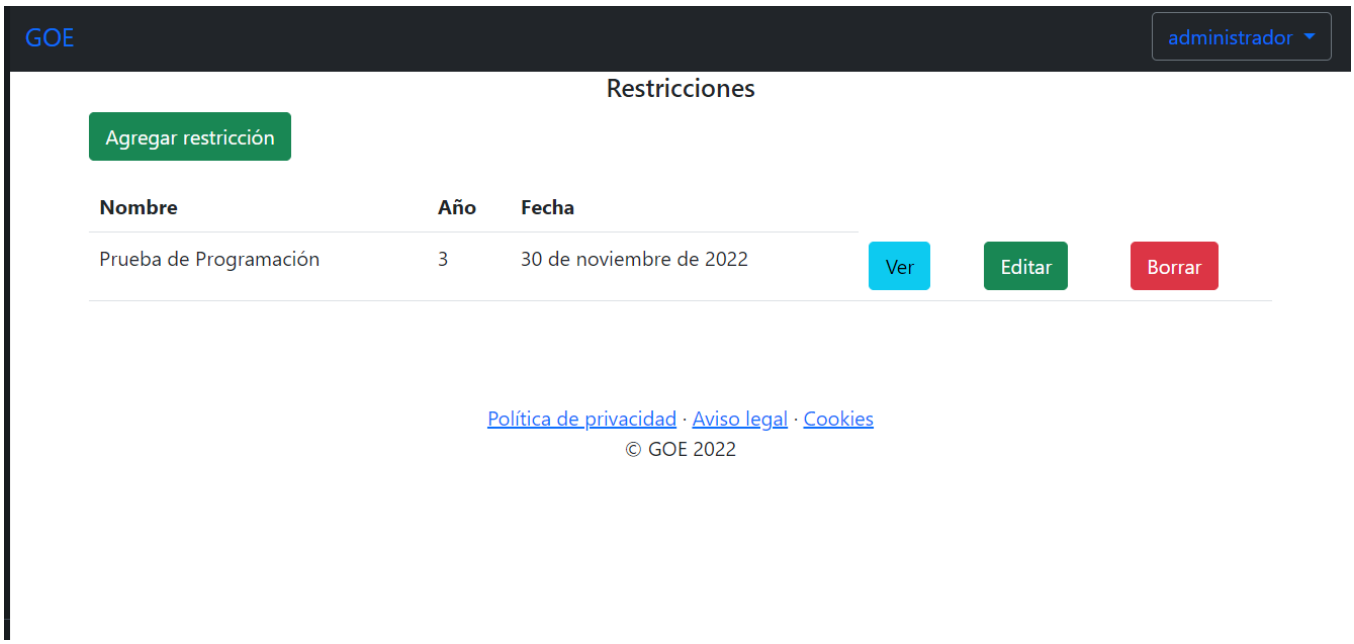
Historia de Usuario	
Número: 18	**Nombre del requisito:** Agregar restricciones.
Programador: Hector Garcés Rey	**Iteración asignada:** 2
Prioridad: Alta	**Tiempo estimado:** 120 horas
Riesgo en desarrollo: Alto	**Tiempo Real:** 120 horas
Descripción: esta funcionalidad debe permitir añadir al sistema las restricciones de asignación de personal a la guardia.	
Observaciones: se deben manejar todas las restricciones requeridas por el cliente.	
Prototipo: 	

Ilustración 6 Prototipo de funcionalidad Agregar Restricciones

2.5 Diseño de la solución.

Un diagrama de clases de diseño(DCD) muestra la especificación para las clases de software de una aplicación(Unad, 2019). A continuación, la ilustración 7 representan el diagrama de clase de diseño con estereotipos web correspondientes al RF 2.

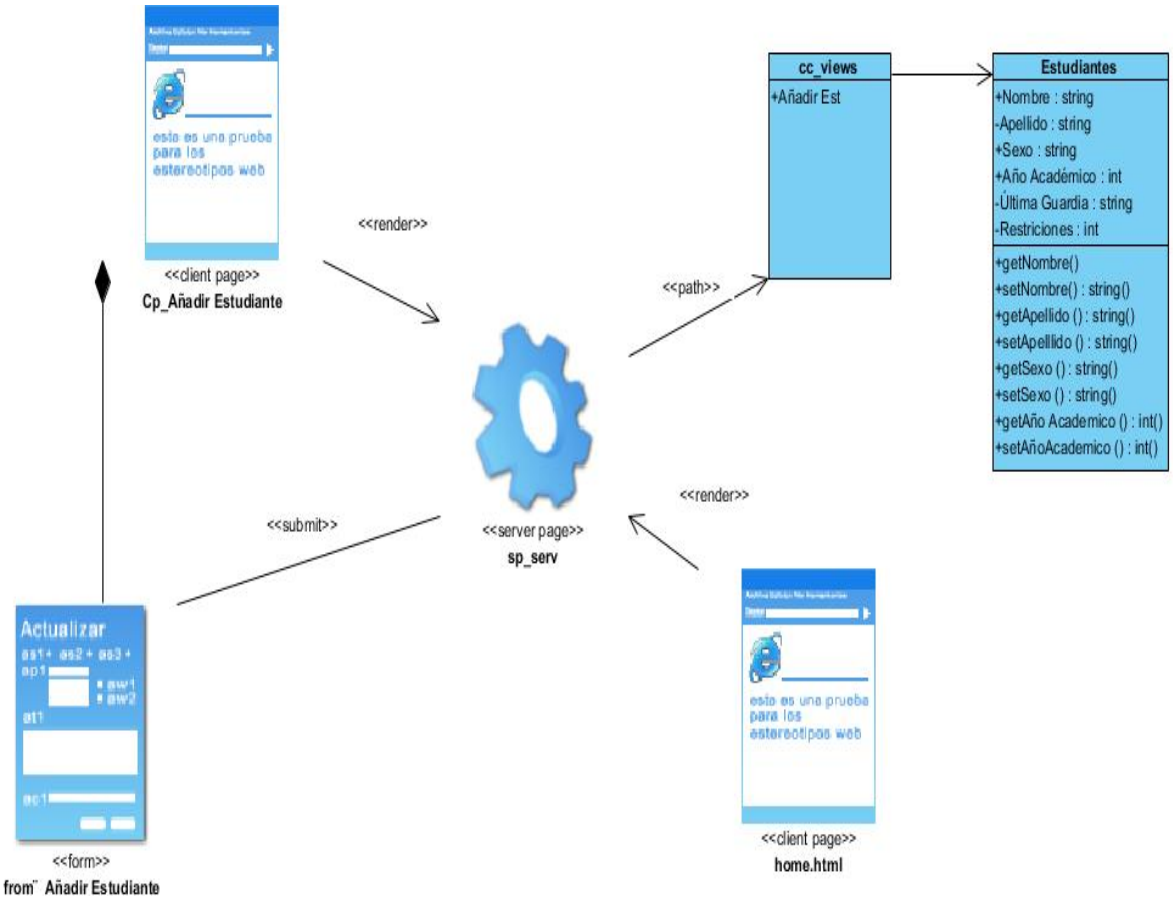


Ilustración 7 DCD con estereotipos web correspondiente al RF 2

En este caso desde la client page home se puede acceder a la CP_Añadir estudiante mediante el server page sp_serv, al solicitar esta funcionalidad se genera el formulario Añadir Estudiante haciendo uso de la clase Estudiantes, los datos y las solicitudes son manejadas a través de la clase controladora cc_view.

Esta misma lógica siguen el resto de las clases y funcionalidades implementadas en el sistema, a continuación, la ilustración 8 y 9 reflejan el DCD con estereotipados webs correspondientes a los RF 6 y 18.

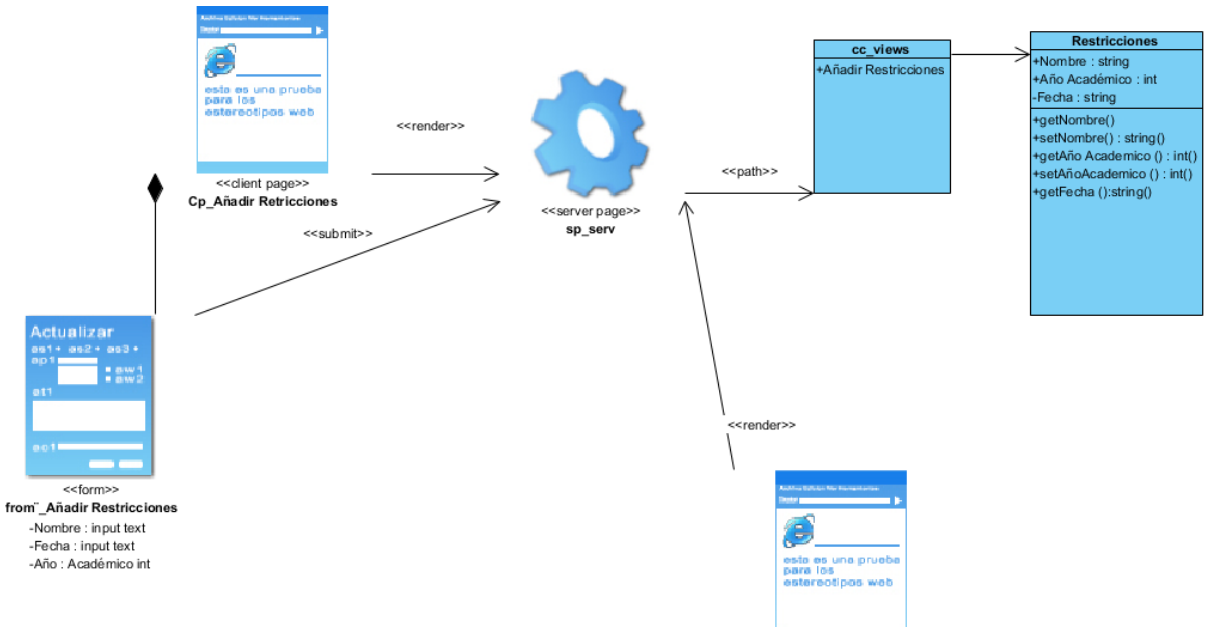


Ilustración 8 DCD correspondiente al RF 18

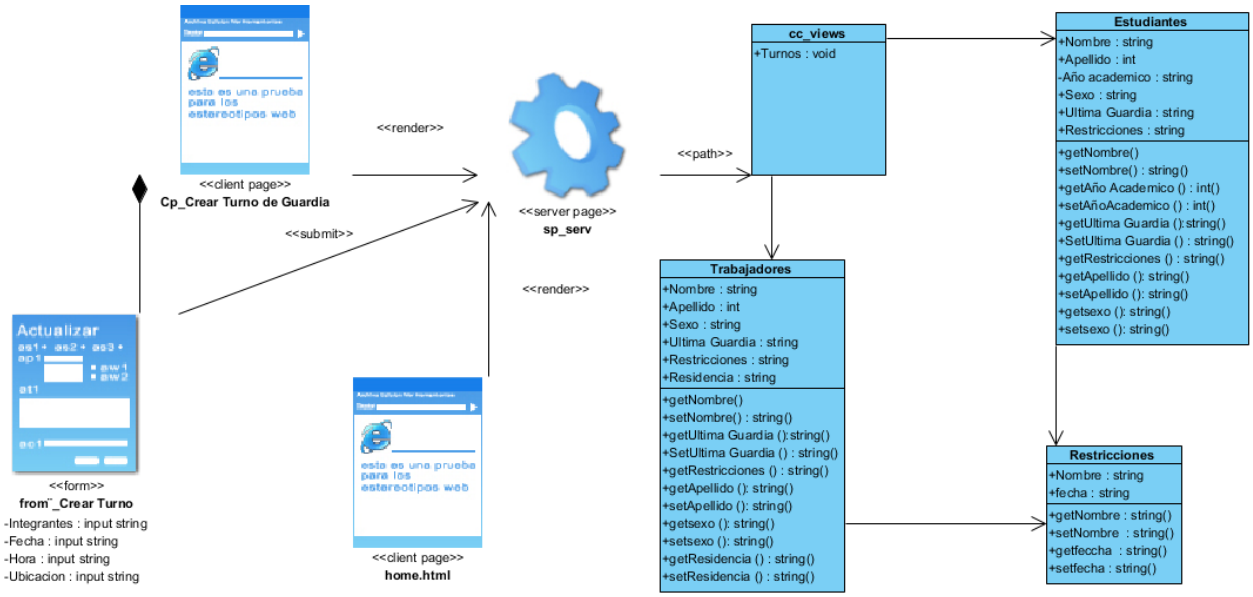


Ilustración 9 DCD correspondiente al RF 6

Patrones:

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptado para resolver un problema de diseño general en un contexto particular. Cada patrón describe un problema

que ocurre una y otra vez en el entorno, y describe la esencia de la solución a ese problema, (Pressman, 2010).

Básicamente los patrones de diseño son la base para solucionar problemas comunes en el desarrollo de software y también en el diseño de interfaces. Pero no todas las soluciones a problemas de diseño o desarrollo son patrones, para que una solución pueda ser considerada un patrón de diseño debe cumplir con ciertas características, como por ejemplo que esta solución pueda resolver problemas semejantes en otras situaciones y que ya tenga un historial de problemas resueltos, es decir, compruebe su efectividad (Patterns, 2019).

Patrones Generales de Software para la Asignación de Responsabilidades (GRASP por sus siglas en inglés). Los patrones de diseño GRASP, describen los principios fundamentales de la asignación de responsabilidades a objetos (Pressman, 2010). A continuación, se evidencian los patrones de diseños utilizados en la propuesta de solución.

Experto: cada clase tiene la responsabilidad de utilizar la información en la cual es experta y domina para realizar la labor para la que fue concebida. Tal es el caso de las clases del modelo que son las encargadas de toda la lógica del acceso a los datos.

No tiene sentido que una clase se deba escribir a sí misma en base de datos o formatearse para presentarse en una plantilla HTML, por el hecho de poseer los datos. Estos son elementos estructurales distintos y deben considerarse desde una perspectiva distinta.

La solución implementada hace uso de este patrón, se reconoce la clase Restricciones (ver Ilustración 9) como la clase experta en la información de restricciones, puesto que esta cuenta con la responsabilidad de manejar todos los eventos que pueden condicionar la ubicación de un trabajador o estudiante en un día determinado.

Creador: identifica quién debe ser el responsable de la creación de nuevos objetos o clases, donde la nueva clase deberá ser creada por la clase que tiene toda la información necesaria para realizar la acción, que usa directamente las instancias creadas del objeto, almacena o maneja varias instancias de clase y contiene o agrega la clase. Una clase B solo puede crear un objeto perteneciente a la clase A solamente cuando (Larman, 2016) :

- B contiene a A.
- B es una agregación (o composición) de A.
- B almacena a A.

- B tiene los datos de inicialización de A (datos que requiere su constructor).

Es evidente la existencia de clases de creación, capaz de engendrar una nueva instancia de clases en caso de que la situación lo amerite. Un ejemplo de esto se evidencia en la clase Estudiantes (ver ilustración 7), a la cual se le solicitan los parámetros de la función constructora en caso de la creación de un objeto de la mencionada clase creadora.

Alta cohesión: cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto-identificable (Larman, 2016).

La Ilustración 10 muestra la clase List_Estudiante la cual realiza una función única en el sistema, de esta forma se encuentran estructuradas el resto de las clases del sistema cumpliendo así con el concepto descrito por Larman para estos patrones. El uso de la clase login con solo nombrar su identificador es otro ejemplo demostrable del uso correcto de patrones en el sistema implementado.

```
1  class List_Estudiante(LoginRequiredMixin,ListView):
2      login_url='login'
3      redirect_field_name='redirect_to'
4      model=Estudiante
5      template_name='estudiante/estudiante_list.html'
6
```

Ilustración 10 Ejemplo de patrón alta Cohesión

Bajo acoplamiento: el acoplamiento se refiere a qué tan relacionadas o dependientes están dos clases o módulos entre sí. El alto acoplamiento en un sistema hace que sea difícil de mantener, ya que un cambio en una clase también tendrá un impacto en otras clases. Esto podría provocar que un cambio fluya a través de un sistema como una mancha de aceite, a veces incluso requiriendo una revisión completa para implementarlo por completo (Chávez, 2019).

Si para hacer cambios en un módulo del programa es necesario hacer cambios en otro módulo distinto, existe acoplamiento entre ambos módulos, en el sistema implementado sigue el orden estructural recomendado por la documentación oficial de Django disponible en su sitio oficial , el cual expone la separación de las clases de los modelos de bases de datos y las vistas del sistema, la evidencia más importante del cumplimiento de este patrón se observa en la baja dependencia entre los módulos que

conforman el sistema , la Ilustración 11 muestra las aplicaciones autenticación, base, geo y guardia, todas son estructuralmente separables, cada una de ellas puede funcionar por separado y ser sustraídas del sistema y no variar su funcionamiento.

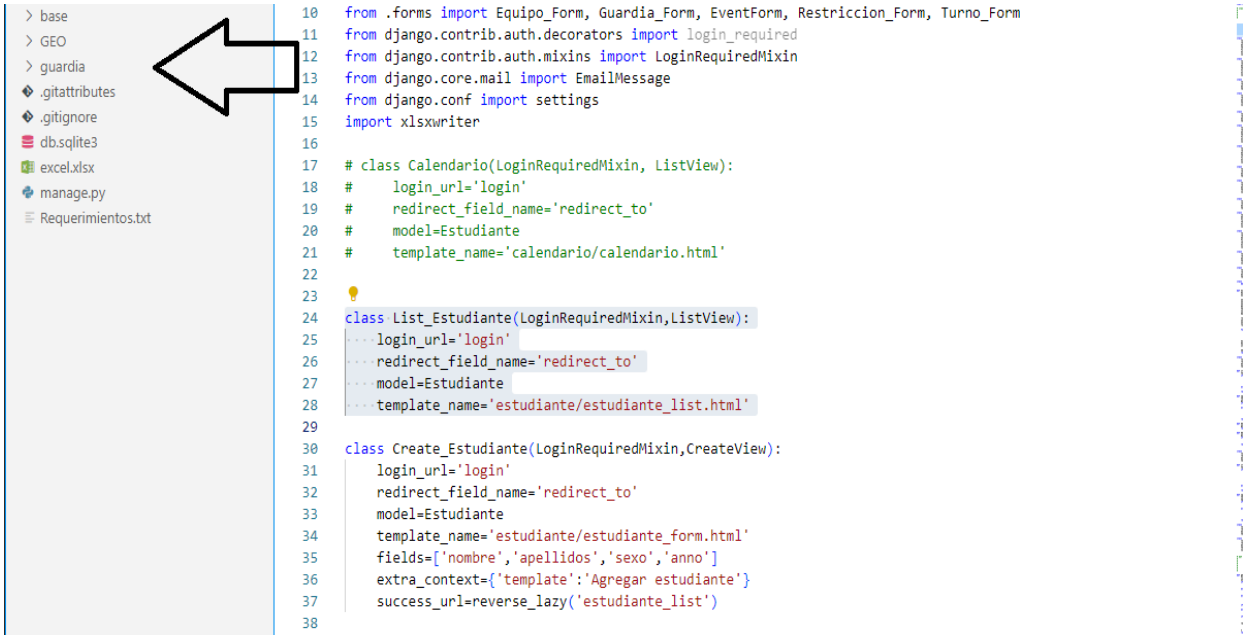


Ilustración 11 Ejemplo de Bajo acoplamiento en el sistema

Controlador: esta responsabilidad se le asigna una interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Define a demás el método de su operación(Larman , 2016).

Dentro de la estructura de las aplicaciones en Django se encuentra la clase views.py la cual controla todas las peticiones provenientes de las plantillas y realiza el manejo de los datos provenientes de la base de datos (ver Ilustración 12).

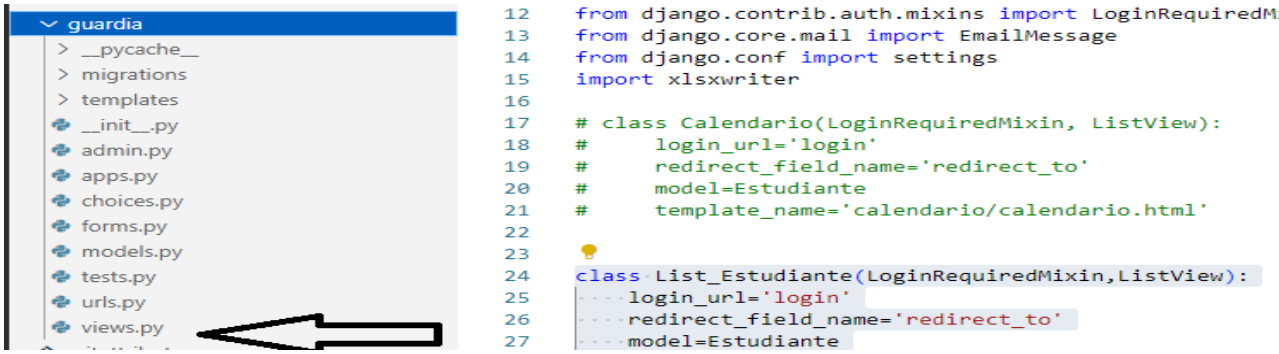


Ilustración 12 Vista general de la clase controladora views.py

2.6 Diagrama de secuencia

El diagrama de secuencia (DS) es un tipo de diagrama de interacción, cuyo objetivo es describir el comportamiento dinámico del sistema de información, haciendo énfasis en la secuencia de los mensajes intercambiados por los objetos (Cillero, 2019). Las ilustraciones 13 representa el diagrama de secuencia correspondiente al RF 2.

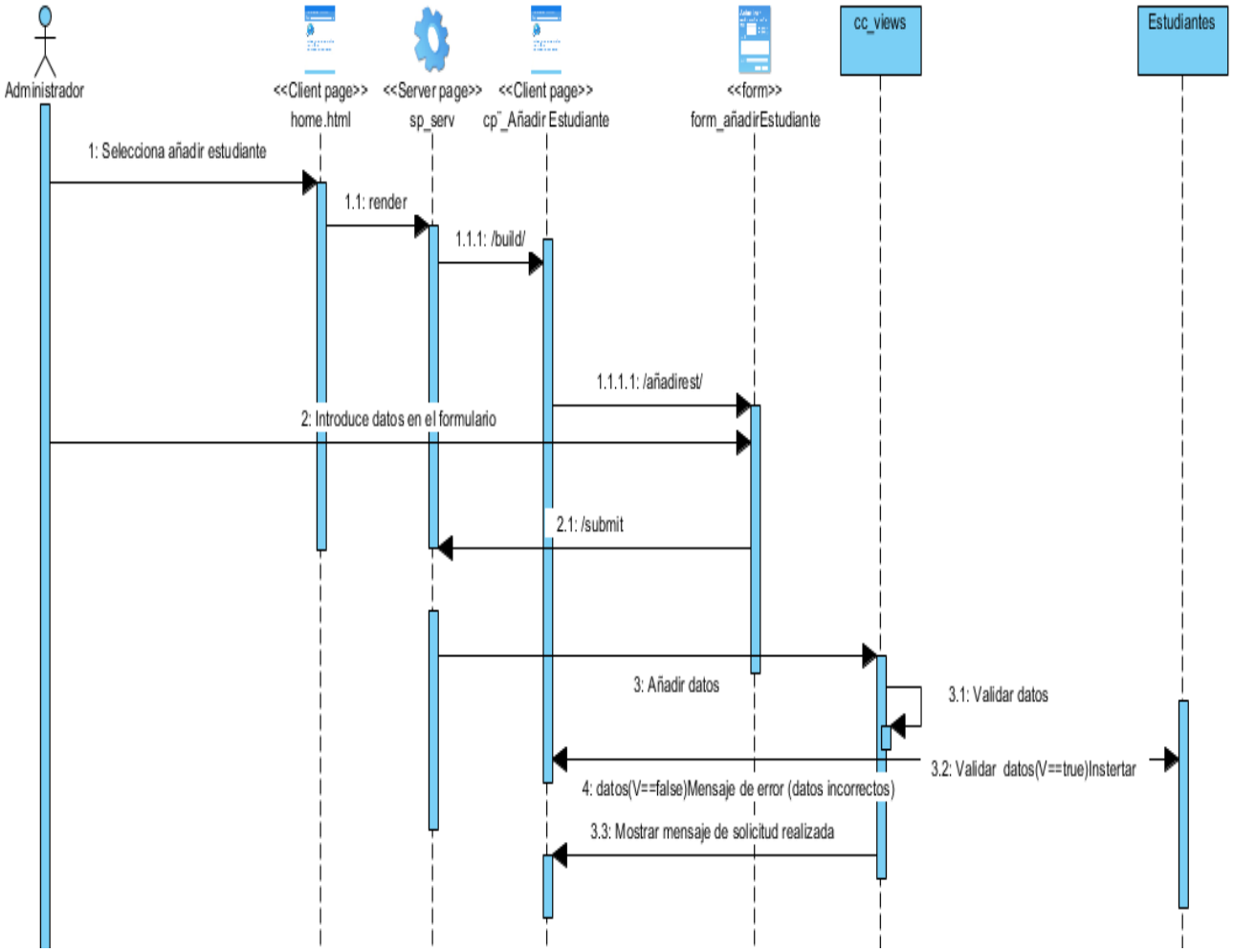


Ilustración 13 DS correspondiente al RF 2

El diagrama de secuencia del RF2 comienza cuando el actor (Administrador) solicita añadir un nuevo estudiante, esta acción se ejecuta al presionar un botón en la client_page nombrada home, la solicitud pasa a través del server_page (página controladora/servidora) y renderiza la client_page Añadir estudiantes en el cual se introducen los datos del formulario añadir estudiante. Una vez rellenados los campos al presionar el botón que ejecuta la acción, los datos pasan a través de la página servidora hacia

la clase controladora views, en esta instancia los datos son validados por las funciones propias de la clase , si los datos son válidos se procede a la creación del objeto estudiante y al envió de un mensaje de confirmación de creación hacia la client_page Añadir Estudiantes, en caso contrario se envía una alerta de datos incorrectos hacia la client¨_page Añadir Estudiantes.

Los procesos a ejecutarse en el sistema poseen homogeneidad con respecto a la secuencia de sus acciones, las ilustraciones 14 y 15 muestran los DS correspondientes a los RF 6 y 18.

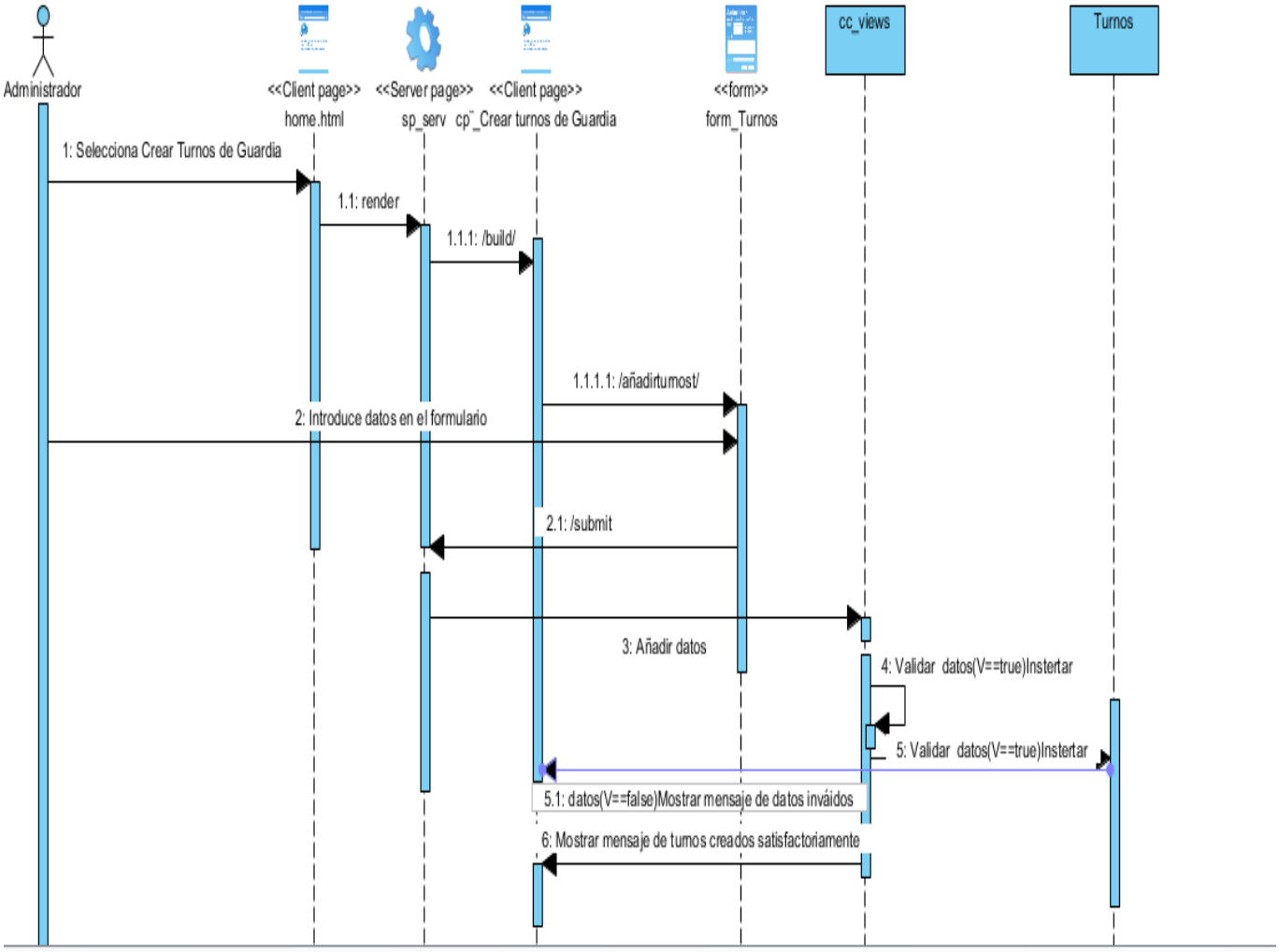


Ilustración 14 DS correspondiente al RF 6

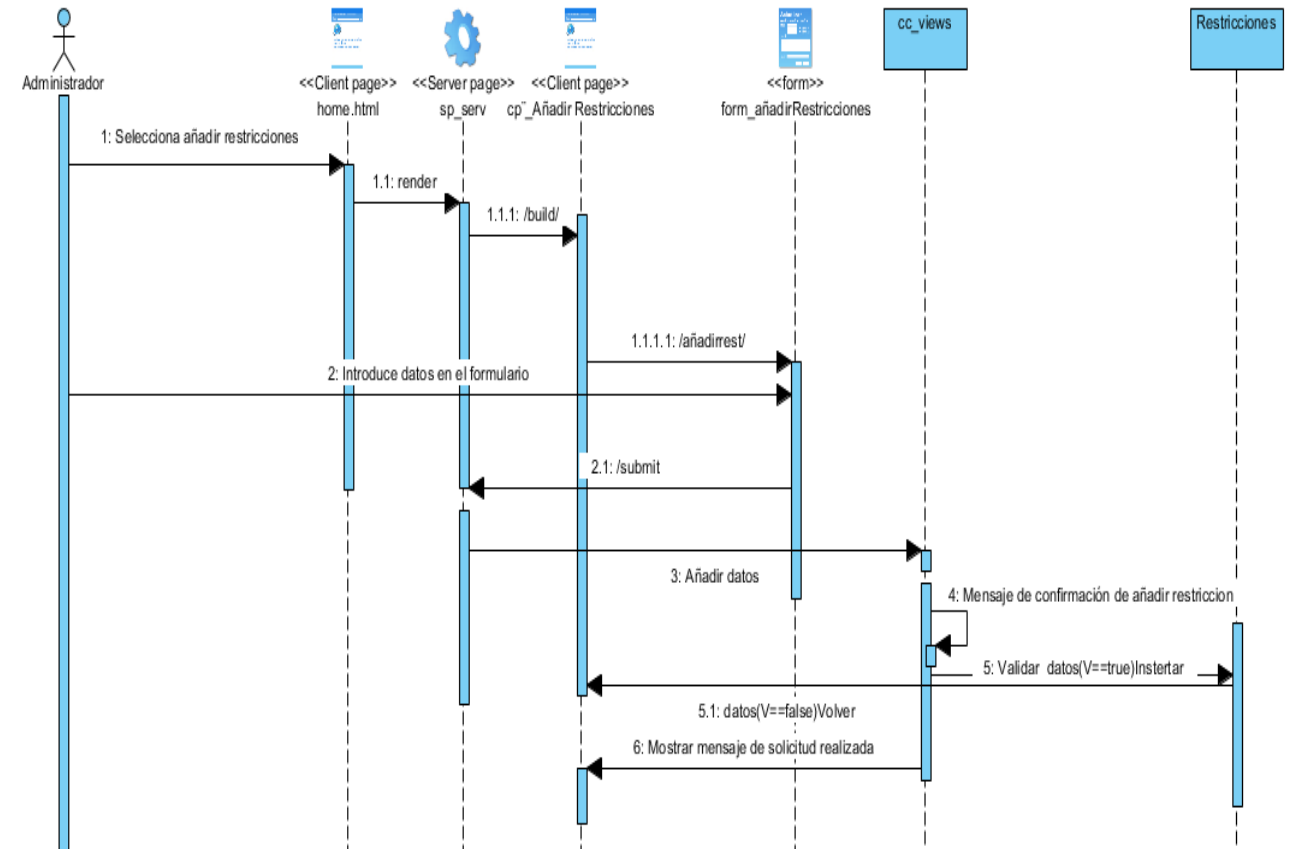


Ilustración 15 DS correspondiente al RF 18

2.7 Arquitectura de Software

La arquitectura de un software se refiere a una planificación basada en modelos, patrones y abstracciones teóricas, a la hora de realizar una pieza de software de cierta complejidad y como paso previo a cualquier implementación. De esta forma se dispone de una guía teórica detallada que permite entender cómo van a encajar cada una de las piezas del producto o servicio (Huet, 2022).

La elección de una arquitectura adecuada recae en los siguientes aspectos (Apiumhub ,2021):

- Crea una base sólida para el proyecto.
- Consigue que la plataforma creada sea escalable.
- Aumenta el rendimiento de la plataforma.
- Reduce considerablemente los costes y evita duplicaciones del código.
- Incrementa la calidad de la plataforma.
- Ayuda en las tareas más complejas.

- Permite hacer la plataforma de manera más rápida.
- Permite una mayor adaptabilidad. Por ejemplo, a la hora de modificar características técnicas del lado del cliente, o implementar motores de reglas. Son tareas más sencillas de adaptar cada una a su debido tiempo, ya que por otro lado el arquitecto de software establece prioridades.
- Ayuda a la gestión de riesgos reduciendo el porcentaje de fracasos.
- Reduce los tiempos de creación y de entrega de los proyectos.
- Da prioridad a los conflictos. Permite comunicación entre las partes y decisiones de diseño previas a su implementación, de forma que sea más fácil un diseño especializado.

Open Webinars la plataforma de *eLearning* tecnológico Argentina define los siguientes patrones de arquitectura de software (OpenWebinars, 2022):

Patrón Cliente-Servidor

El patrón cliente servidor es muy usado sobre todo en el diseño de web y servicios online, y se basa en el concepto de la existencia de un servidor (que proporciona el servicio) y una serie de clientes, que piden al servidor y reciben una respuesta del mismo.

Patrón de capas

En este patrón se subdivide la estructura del programa en un número de capas que representan una subtarea, cada una perteneciendo a un nivel de abstracción diferente. Cada capa está diseñada para proporcionar un servicio a la siguiente capa de mayor nivel.

Patrón Master-Slave

Este patrón consiste en dos grupos: maestro y esclavo. Los esclavos realizan la tarea propuesta por el maestro, computan los resultados y los envían de nuevo a este, quien los presenta, almacena o procesa. Esto se realiza así para tener una parte que autoriza y dirige los cálculos necesarios y otra parte que procesa de manera agnóstica estas decisiones.

Modelo-Vista-Controlador

Este famoso patrón, también conocido como patrón MVC, divide una aplicación interactiva en tres partes diferenciadas:

- Modelo: contiene la funcionalidad central y los datos.
- Vista: muestra la información al usuario, siempre es posible definir una o más vistas para una misma aplicación.

- Controlador: maneja la entrada del usuario. Esto se hace para separar las representaciones internas de la información de las formas en que se presenta y se acepta la información del usuario. De esta manera se desacopla los componentes y permite una reutilización eficiente del código.

MVC supone una alternativa más que fiable a la hora de crear e implementar sistemas en los que se incluyen interfaces de usuario. El uso de esta técnica proporciona ventajas como una mayor fiabilidad y solidez en el software, lo que aumenta su ciclo de vida; reutilización del código; separación de conceptos y una mayor facilidad en el mantenimiento. MVC se está convirtiendo en una de las tendencias más destacadas en el mundo del desarrollo web. Esta propuesta, ya presentada en los años 70, ha retomado su vigencia gracias a la aparición de un gran número de frameworks (estructura que sirve de base para la organización y desarrollo de software) que utilizan el patrón MVC como el modelo a la hora de construir aplicaciones web(API_Market, 2022).

Django sigue el patrón MVC, pero en el caso particular de este framework el objeto controlador (C) es manejada por Django y otorga las responsabilidades del usuario a los objetos de vista (V), por lo que el MVC se transforma en MTV *(Model-Templete-View)*, los objetos se redefinen de la siguiente manera(Téllez, 2022):

- M significa "Modelo" (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.
- 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: se puede pensar en esto como un puente entre el modelo y las plantillas.

Al usar Django como framework de desarrollo en la implementación de la solución, se introduce el uso de MTV como modelo arquitectónico a seguir, es real que el uso de Django no condiciona usar MTV por obligación, se pueden realizar modificaciones a la infraestructura creada por el framework y adaptarlo para el uso de otros patrones como Cliente-Servidor y Capas, el equipo de desarrollo decide regirse a su experiencia en el uso de Django y en las características de la solución para seguir el patrón MTV.

La ilustración 16 ejemplifica el uso de este patrón en la solución implementada, dentro del módulo guardia se encuentran los archivos correspondientes a la arquitectura seleccionada, al desplegar los

archivos internos del módulo, se puede observar la existencia de la carpeta templetes, dentro se encuentran todos los archivos HTML correspondientes a las plantillas del modelo, debajo de la carpeta antes mencionada se encuentra el archivo models.py y views.py, el primero contiene toda la lógica de la base de datos, definición de campos y sus características, el segundo contiene las configuraciones de conexión entre las plantillas HTML y los objetos del modelo .

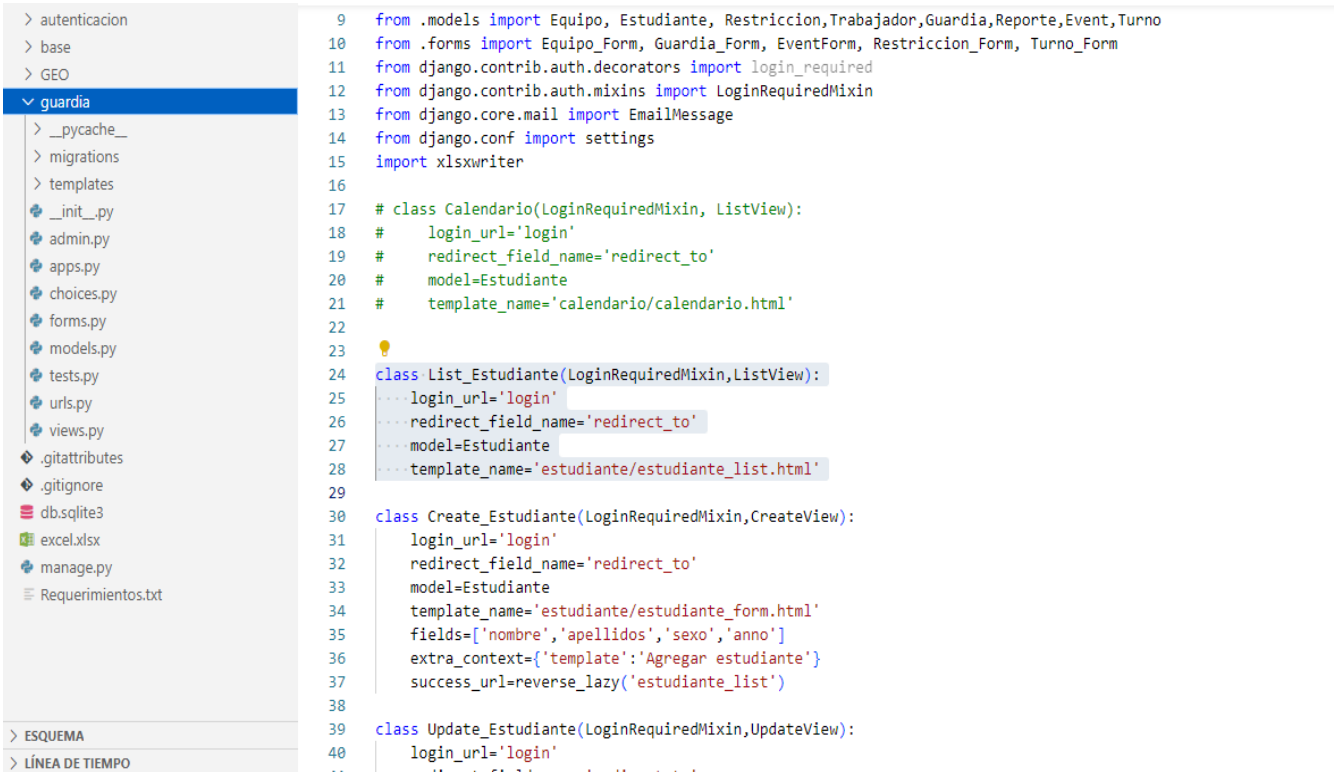


Ilustración 16 Uso del MTV en la solución.

2.8 Modelo de datos:

El modelado de datos es el proceso de diagramación de los flujos de datos. Al crear la estructura de una base de datos nueva o alternativa, el diseñador comienza con un diagrama del flujo de los datos por dentro y fuera de la base de datos. Este diagrama se usa para definir los formatos y estructuras de los datos y las funciones de gestión de la base de datos, a fin de dar un soporte eficiente al flujo de datos. Una vez creada e implementada la base de datos, el modelo de datos es la documentación y justificación de por qué existe la base de datos y cómo se diseñaron los flujos (Reyman, 2021).

La Ilustración 17 muestra la estructura lógica de la base de datos del sistema de GOE de la Facultad 1, incluidas las relaciones y limitaciones que determinan cómo se almacenan los datos y cómo se accede a ellos.

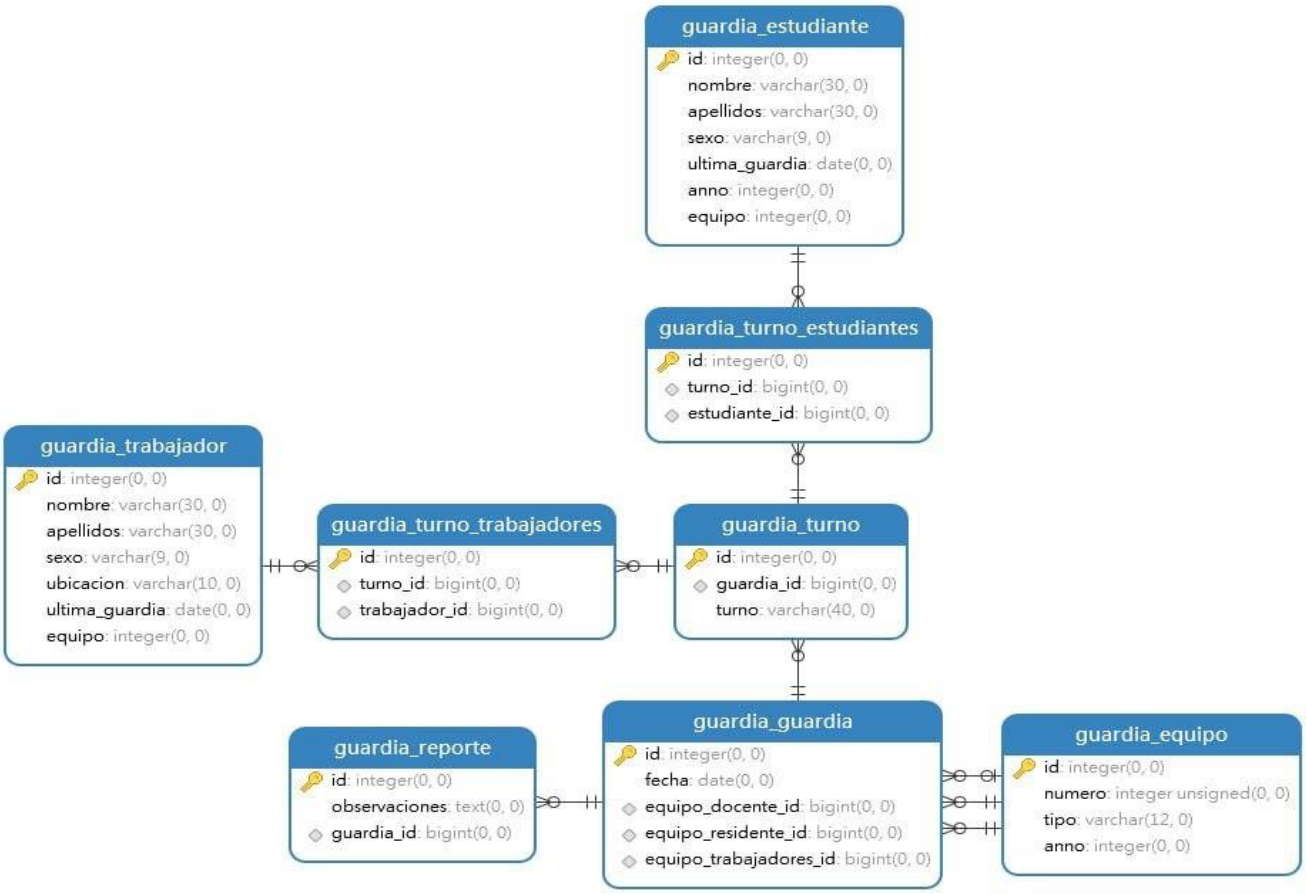


Ilustración 17 Diagrama de base de datos del sistema a implementar.

Conclusiones Parciales del Capítulo.

En el presente capítulo, para mejorar la comprensión de la problemática planteada se diseñó un mapa conceptual que permitió identificar los principales componentes del negocio. Se realizó la descripción de la solución planeada lo que propició el correcto levantamiento de los RF y RNF del sistema a implementar.

Se representan las HU que corresponden a los RF 2, 6 y.18 como parte de los artefactos generados por la metodología seleccionada. Como parte del proceso de modelado de la solución se procedió al diseño

de los diagramas de clases con estereotipados webs correspondiente a los RF 2, 6 y 18, facilitando la comprensión de la relación entre las clases. Se describieron los patrones GRASP utilizados como elementos para la reutilización de código y la especialización de las clases implementadas. Se definió el patrón arquitectónico MTV para guiar el proceso de implementación del sistema, proporcionando al equipo de desarrollo de una guía teórica en la relación de los componentes que integran la solución.

Capítulo 3. Validación de la propuesta de solución

Introducción al capítulo

Como parte del desarrollo de la solución, en este capítulo se representa el diagrama de componentes del sistema. Se analizan los estándares de codificación a emplear en la programación de la solución y se somete la propuesta de solución a un proceso de pruebas como parte del proceso de validación del sistema implementado.

3.1 Diagrama de componentes

Los diagramas de componentes se utilizan para visualizar la organización de los componentes de un sistema y las relaciones de dependencia entre ellos. Proporcionan una visión de alto nivel de los componentes de un sistema (Siriwardhana, 2020). A continuación, la Ilustración 18 muestra el diagrama de componentes que integran el sistema:

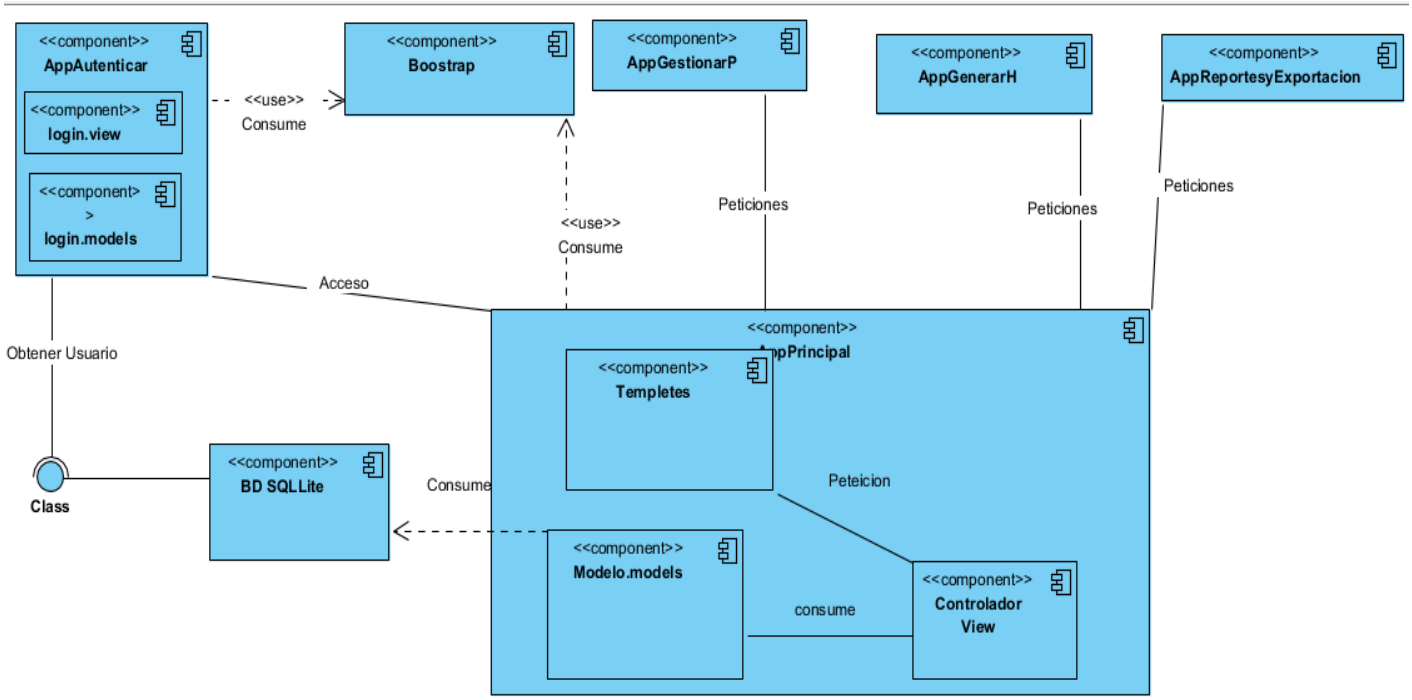


Ilustración 18 Diagrama de Componentes del sistema

3.2 Estándar de codificación

Django se encuentra regido por estándares de codificación descritos en su documentación oficial, el uso de estos va más allá de buenas prácticas de programación, su aplicación es esencial para mantener el uso de patrón de bajo acoplamiento en las aplicaciones que lo componen. A continuación, se evidencian los estándares de codificación que serán utilizados en la propuesta de solución (Crespo, 2021):

Sangría:

-Utilice 4 espacios para la sangría

Grosor de línea:

-Cada línea de código no debe exceder los 80 caracteres en la medida de lo posible (en circunstancias especiales, puede exceder ligeramente los 80, pero la más larga no puede exceder los 120).

Comillas:

-El lenguaje natural usa comillas dobles"...".

-Por ejemplo, mensajes de error; en muchos casos sigue siendo Unicode.

-Las expresiones regulares usan comillas dobles nativas "...".

-La cadena de documentos usa tres comillas dobles"""....."""".

Línea en blanco

-Dos líneas en blanco entre las funciones de nivel de módulo y las definiciones de clase.

-Línea en blanco entre las funciones de los miembros de la clase.

-Puede utilizar varias líneas en blanco para separar varios conjuntos de funciones relacionadas.

-Las líneas en blanco se pueden utilizar en funciones para separar códigos relacionados lógicamente.

Codificación:

-El archivo usa codificación UTF-8

Convenciones de Nombres:

-Usar todos los nombres en minúsculas para paquetes y módulos, intentar no usar guiones bajos.

-Uso del nombre de claseCamelCaseEstilo de nomenclatura, las clases internas pueden comenzar con un guion bajo.

-Las funciones se nombran en minúsculas separadas por guiones bajos.

-Cuando el nombre del parámetro entra en conflicto con las palabras reservadas de Python, puede agregar un guion bajo al final en lugar de usar abreviaturas.

-Las constantes se nombran en mayúsculas separadas por guiones bajos.

Declaración de importación:

-La declaración de importación debe escribirse en líneas separadas.

-La declaración de importación debe usar importación absoluta.

-La declaración de importación debe colocarse al principio del archivo, después de la descripción del módulo y la cadena de documentos, y antes de las variables globales.

-Las declaraciones de importación deben organizarse en orden, separadas por una línea en blanco entre cada grupo.

-Al importar definiciones de clases de otros módulos, puede utilizar la importación relativa.

-Si se produce un conflicto de nombres, puede utilizar el espacio de nombres.

3.3 Pruebas

Las pruebas de software son el proceso en el cual emergen todas las actividades del ciclo de vida, tanto estáticas como dinámicas relacionadas con la planificación, preparación y evaluación de productos de software para determinar que cumplen los requisitos especificados, para desmostar que son aptos para el propósito y para detectar defectos en los sistemas desarrollados(Peño, 2015). A continuación, se describen el conjunto de pruebas realizadas para validar la solución implementada.

3.3.1 Pruebas unitarias

Las pruebas unitarias de software, pueden definirse como un mecanismo de comprobación del funcionamiento de las unidades de menor tamaño de un programa o aplicación en específico. Estas forman parte de la estrategia de metodología ágil del trabajo de desarrollo, donde se busca ofrecer piezas pequeñas de software en funcionamiento en un corto periodo de tiempo, con el objetivo de aumentar la

satisfacción del cliente. Es importante aclarar que este tipo de pruebas son de vital importancia para la detección de errores, ya que, sin este testeo, no podrían identificarse hasta fases más avanzadas del desarrollo, como, por ejemplo, la fase de integración. Esto implica que las pruebas unitarias de software evitan la escalada de errores en el código al identificarlas de manera temprana (KeepCoding, 2022).

Las pruebas unitarias realizadas al Sistema de GOE de la Facultad 1 se ejecutaron a través del marco de pruebas de Python Pytest. El equipo de desarrollo acordó ejecutar este tipo pruebas al culminar la programación de cada módulo, al identificarse tres módulos en el sistema, las pruebas se ejecutaron esta misma cantidad de veces, cabe destacar que todos los errores fueron erradicados en el momento de su identificación obteniendo los resultados siguientes:

Módulo Base: se identificaron 7 clases dentro de este módulo, de las cuales 3 presentaron errores en el tiempo de ejecución de la clase.

Módulo GOE: en este módulo se identificaron 10 clases, de las cuales 2 presentaban errores en el tiempo de ejecución y 3 errores de indentación.

Módulo Guardia: en este módulo se identificaron 15 clase, de las cuales 4 presentaban errores en el tiempo de ejecución, 2 errores de indentación y 2 en la función de validación de correos.

A continuación, se expone el caso de prueba puesto en práctica mediante el marco de trabajo de Python Pytest.

La ilustración 19 representa la configuración de dos casos de prueba con el objetivo de evaluar el funcionamiento de la función de correo, la ilustración 20 muestra la salida obtenida una vez aplicada la prueba en la cual se obtiene dos errores en la función evaluada, y la ilustración 21 expone los resultados de la consola de VS Code una vez corregido los errores en la función.

guardia > tests.py > test_regular_email_validates

```
1   ...
2   def test_valid_email_can_have_plus_sign():
3       assert is_valid_email_address('john.doe+abc@gmail.com')
4
5   def test_valid_email_must_have_a_tld():
6       assert not is_valid_email_address('john.doe@example')
7
8
```

Ilustración 19 Caso de prueba para identificar errores en la función de validación de correos.

tests.py base ● tests.py guardia 9+ ●

guardia > tests.py > ...

```
1   ============================= test session starts ==============================
2   platform darwin , pytest-7.0.1, pluggy-1.0.0
3   rootdir: /Users/GOE/guardia
4   collected 5 items
5
6   test_validator.py ...FF                                                  [100%]
7
8   =================================== FAILURES ===================================
9   ____________________ test_valid_email_can_have_plus_sign ____________________
10
11      def test_valid_email_can_have_plus_sign():
12  >       assert is_valid_email_address('john.doe+abc@gmail.com')
13  E       AssertionError: assert False
14  E        +  where False = is_valid_email_address('john.doe+abc@gmail.com')
15
16  test_validator.py:17: AssertionError
17  ____________________test_valid_email_must_have_a_tld____________________
```

Ilustración 20 Salida por consola una vez aplicada la prueba en Pytest.

uardia > tests.py

```
1   ======================== test session starts =========================
2   platform  pytest-7.0.1, pluggy-1.0.0
3   rootdir: /Users/bascodes/Code/blogworkspace/pytest-example
4   collected 3 items
5
6   test_validator.py ...                                          [100%]
7
8   ========================= 3 passed in 0.01s ==========================
9
10
```

Ilustración 21 Salida por consola una vez corregido los errores

3.3.2 Pruebas funcionales

Las pruebas funcionales implican evaluar y comparar cada función de software con los requisitos funcionales validados con el cliente (Font, 2021).

Dentro de las pruebas funcionales se pueden encontrar las pruebas de caja negra. En esta variante la funcionalidad del software bajo prueba se estudia sin tener en cuenta la estructura del código interno, los detalles de implementación y el conocimiento de las rutas internas del software (Menéndez, 2022).

La tabla número 6 ejemplifica el diseño de un caso de prueba para el RF Añadir Estudiantes, utilizado para la detección de errores en el software.

Los valores de las variables pueden ser: Validos(V), Inválidos(I) o No Aplica (NA).

Tabla 6 Ejemplo de diseño de casos de prueba de caja negra

Escenario de Prueba (EP)	Descripción	Nombre	Apellido	Año Académico	Sexo	Respuesta del Sistema	Respuesta de Prueba
EP 1	Añadir estudiante	V	V	V	V	Mensaje de Confirmación “El estudiante de ha añadido con éxito”	
	Añadir estudiante	Pedro	Gonzáles	6(I)	M	“El estudiante de ha añadido con éxito”	No muestra el mensaje de año fuera de rango
	Añadir estudiante	Pedro	G onzáles(I)	5	M	Mensaje de error “verificar	

						espacios en datos”	

Resultado de las pruebas funcionales

Al realizar las pruebas funcionales en la primera iteración se identificaron 16 no conformidades en el sistema, de las cuales 8 estaban vinculadas a errores ortográficos las cuales fueron erradicadas al momento, dos correspondían a errores de navegación dentro del sistema, cinco pertenecieron a ausencias en las notificaciones de usuarios y uno se refería al listado de datos. Este último a pesar de ser detectado mediante la técnica de caja negra fue referenciado para su posterior análisis en el código de programación, por reflejar este un posible error de codificación. De las 16 no conformidades detectadas 12 fueron resueltas en la primera iteración quedando pendiente dos de error de navegación y dos perteneciente a la ausencia de alerta de usuario en la funcionalidad eliminar plan de guardia

En la segunda iteración sumado a las cuatro insuficiencias que no se resolvieron en la etapa 1, se detectaron cuatro errores en el funcionamiento del sistema debido a la incorporación de nuevas funcionalidades. De las cuatro nuevas insuficiencias 3 pertenecían a errores ortográficos los cuales se resolvieron al momento y uno a error en la navegación del sistema. Este último se refiere a la incapacidad del sistema de pasar al módulo de reportes de guardia mostrando un error 404 de error en la petición al servidor, el equipo de desarrollo resolvió este error en el momento de ser detectado y en esta misma etapa se corrigieron los errores pendientes de la iteración 1.

En la iteración número 3 se añadieron a prueba nuevas funcionalidades, en las cuales se detectó 1 error perteneciente a la ausencia de alertas por espacio entre nombres, esta deficiencia fue erradicada en el momento de ser detectada.La figura muestra en resumen la evolución a lo largo de las 3 iteraciones de la relación errores detectados -errores erradicados.

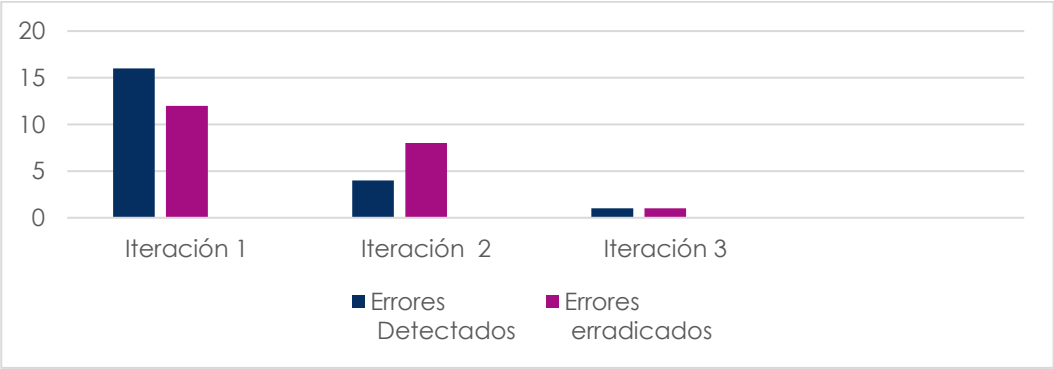


Ilustración 22 Gráfico resumen de las pruebas funcionales

3.3.3 Pruebas de rendimiento.

Las pruebas de carga clasifican dentro de las pruebas de rendimiento de un sistema, es una prueba planificada para realizar un número específico de solicitudes a un sistema, para probar la funcionalidad del sistema bajo niveles específicos de solicitudes simultáneas. Una prueba de carga garantiza que un sistema web pueda controlar un volumen de tráfico esperado y, por lo tanto, a veces se conoce como pruebas de volumen. El objetivo de una prueba de carga es demostrar que un sistema puede controlar el volumen esperado con una degradación del rendimiento mínima a aceptable. Una prueba de estrés es una prueba diseñada para aumentar el número de solicitudes simultáneas en un sistema más allá de un punto donde el rendimiento se degrada, incluso hasta el punto de falla completa (LoadView, 2022).

Haciendo uso de la herramienta JMeter se definen los siguientes parámetros para realizar la prueba de carga:

-Servidor con 2 GB de RAM

-La computadora cliente con 2 GB de RAM

-Sistema operativo Nova.

- Tipo de Red: Ethernet 10/100 Mbps.

-Se varia la carga de peticiones de usuarios en 0 ,5 ,7 ,10 respectivamente para los que se espera un rendimiento del 100% sin caída del servidor.

La Ilustración 15 muestra un fragmento de la tabla de reportes generada por JMeter en la realización de la prueba de carga al sistema.

Summary Report

Name: Summary Report

Comments:

Write results to file / Read from file

Filename | Browse... | Log/Display Only: Errors Successes | Configure

Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB...	Sent KB/sec	Avg. Bytes
1 /	2000	242	9	650	230,94	0,00%	41,2/sec	1343,39	79,81	33354,2
2 /reserve.p...	1000	65	56	141	13,50	0,00%	20,8/sec	216,40	9,64	10658,9
3 /purchase...	1000	65	56	132	10,49	0,00%	20,8/sec	202,78	10,84	9973,0
4 /confirmat...	1000	64	56	133	9,48	0,00%	20,8/sec	182,97	12,16	8995,9
5 /home	1000	133	114	232	17,71	0,00%	20,8/sec	335,88	15,18	16533,3
6 /index.php	1000	63	56	184	10,20	0,00%	20,8/sec	161,82	7,37	7949,0
7 /register	1000	66	57	129	10,89	0,00%	20,8/sec	224,42	7,45	11022,1
TOTAL	8000	117	9	650	138,14	0,00%	165,0/sec	2655,04	141,85	16480,1

Ilustración 23 Fragmento de la tabla de reportes generada por JMeter

Al finalizar la prueba de carga se deben analizar los datos obtenidos. La tabla 7 refleja el resumen de la prueba de carga y los resultados de la mencionada prueba.

Tabla 7 Resumen de la prueba de Carga

Fecha	Número de peticiones	Numero de fallos	Tiempo de respuesta(segundos)
27/11/22	0	0	0
27/11/22	5	0	0.5
27/11/22	7	0	1.2
27/11/22	10	0	1.6

Al evaluar los resultados obtenidos a partir de la aplicación al sistema desarrollado de la prueba de carga se aprecia que el tiempo de respuesta (TR)cumple con lo planteado en los requisitos no funcionales en cuanto a usabilidad, dado que el TR se encuentra por debajo de los 5 segundos establecidos por los requisitos antes mencionados.

3.3.4 Pruebas de seguridad

Es 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 (Zuñiga, 2020).

Para el desarrollo de las pruebas de seguridad correspondientes al sistema desarrollado, se hace uso de la herramienta Burp Suite. Al aplicar la prueba se detectaron 18 alertas, reflejadas en el cuadro resumen de la tabla 8.

Como parte de la estrategia de mitigación de las alertas de seguridad se procedió a incluir los certificados CSFR en todas las plantillas que componen el sistema desarrollado, para evitar el acceso al sistema sin antes haber realizado autenticación se procedió a incluir como parámetro en todas las funciones y clases del sistema el decorador login_required.

Tabla 8 Resumen de las deficiencias encontradas en la prueba de seguridad

Tipos de alerta	Cantidad de alertas

Ausencia de certificado de falsificación de solicitudes (CSFR)	8
Acceso a paginas sin la necesidad de Loguin	10

3.4 Diagrama de Despliegue

El Diagrama de despliegue es un diagrama estructurado que muestra la arquitectura del sistema desde el punto de vista del despliegue (distribución) de los artefactos del software en los destinos de despliegue. Los artefactos representan elementos concretos en el mundo físico que son el resultado de un proceso de desarrollo. Ejemplos de artefactos son archivos ejecutables, bibliotecas, archivos, esquemas de bases de datos, archivos de configuración, entre otros (Sarmiento, 2013). La Ilustración 24 muestra el diagrama de despliegue del sistema GOE de la Facultad 1.

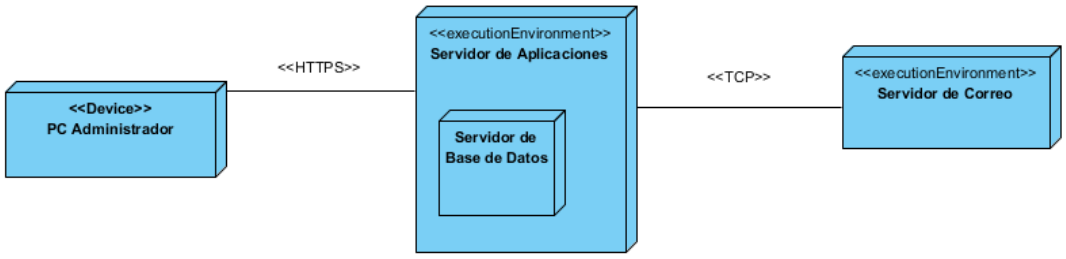


Ilustración 24 Diagrama de despliegue

Descripción de elementos e interfaces de comunicación:

En el diagrama de despliegue el proceso de solicitud al sistema se inicia en el nodo pc_administrador, a través del navegador utilizando el protocolo HTTPS, mediante el cual se emite una petición al servidor de aplicaciones en el cual se encuentra hosteado el sistema. El servidor web establece una conexión con la base de datos, la cual se encuentra integrada al servidor de aplicaciones. En caso de realizar una petición de envío de correo, el sistema mediante protocolo TCP realiza una petición al servidor de correo.

Pc_administrador: representa el ordenador cliente, el cual realiza peticiones al servidor de aplicaciones para el manejo de información, consulta de datos o cualquier acción que se genere dentro del sistema que se encuentra en funcionamiento.

Servidor de Aplicaciones: recibe las peticiones realizadas por el pc_administrador y efectúa las consultas necesarias, al hacer uso de Django, el servidor de base de datos se encuentra integrado al sistema.

Servidor de Correo: recibe peticiones del sistema que se encuentra en el servidor de aplicaciones en caso de hacer uso de la funcionalidad de correo, en este caso se usan correo bajo el dominio uci.cu.

Conclusiones del capítulo

En el presente capítulo, se expuso el diagrama de componentes del sistema lo que propició una mejor comprensión de la estructura arquitectónica de la solución desarrollada. Los estándares de codificación utilizados permitieron establecer un mejor entendimiento del código, facilitando el posterior mantenimiento y modificación del mismo. Las pruebas aplicadas arrojaron errores que fueron corregidos en cada iteración, lo que permitió la obtención de un software que satisface los RF establecidos.

Conclusiones

Con el desarrollo de la presente investigación se dio respuesta a los objetivos propuestos, arribando a las siguientes conclusiones:

- La investigación realizada sobre el estado del arte relacionados con la planificación y el control de la GOE en la Facultad 1, permitió definir los conceptos necesarios para guiar la investigación. Mediante el análisis de sistemas homólogos se evidenciaron las funcionalidades comunes entre sistemas de esta naturaleza y generó la necesidad de desarrollar un nuevo sistema de GOE en la Facultad 1.
- Se selección del ambiente tecnológico para el desarrollo del sistema y la confección de artefactos que propone la metodología seleccionada permitieron la implementación de la propuesta de solución a la problemática planteada.
- La aplicación de pruebas de software permitió identificar y corregir errores cometidos en la implementación del sistema y propició evidenciar que la herramienta desarrollada cumple con los requisitos funcionales determinados por el cliente.

Recomendaciones

Se recomienda el desarrollo de un módulo que permita a los integrantes de la GOE solicitar permuta de guardia sin la necesidad de hacerlo presencial.

Referencias bibliográficas

AGENCY, M.D., 2021. Framework: Qué es y Todas sus Ventajas. *Epitech España* [en línea]. [Consulta: 17 noviembre 2022]. Disponible en: https://www.epitech-it.es/que-es-un-framework/.

AGUILAR ORDONEZ, E.E., 2016. *Ventajas del uso de framework para aplicaciones web.* [en línea]. Thesis. S.l.: s.n. [Consulta: 17 noviembre 2022]. Disponible en: http://repositorio.digital.tuxtla.tecnm.mx/xmlui/handle/123456789/98.

ANDALUCÍA, 2019. JMeter | Marco de Desarrollo de la Junta de Andalucía. [en línea]. [Consulta: 18 noviembre 2022]. Disponible en: https://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/224.

ANDRÉS GUTIÉRREZ, 2018. ¿Cuáles son los pros y los contras de HTML? *Quora* [en línea]. [Consulta: 18 noviembre 2022]. Disponible en: https://es.quora.com/Cuáles-son-los-pros-y-los-contras-de-HTML.

API_MARKET, 2022. MVC, desarrollo fiable de aplicaciones web. *BBVA API_Market* [en línea]. [Consulta: 23 noviembre 2022]. Disponible en: https://www.bbvaapimarket.com/es/mundo-api/mvc-desarrollo-fiable-de-aplicaciones-web/.

APIUMHUB, 2021. Importancia de la arquitectura de software | Apiumhub. [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: https://apiumhub.com/es/tech-blog-barcelona/arquitectura-de-software/.

ATURNOS, 2022. Gestión global de equipos. Simplifica tus turnos de trabajo. [en línea]. [Consulta: 16 noviembre 2022]. Disponible en: https://www.aturnos.com.

AWS, 2022. ¿Qué es JavaScript? - JavaScript explicado - AWS. *Amazon Web Services, Inc.* [en línea]. [Consulta: 18 noviembre 2022]. Disponible en: https://aws.amazon.com/es/what-is/javascript/.

CANVAS, 2022. ¿Qué es un mapa conceptual? (con ejemplos y plantillas) | Canva. *Aprende* [en línea]. [Consulta: 20 noviembre 2022]. Disponible en: https://www.canva.com/es_mx/aprende/que-son-mapas-conceptuales/.

CENTRO EUROPEO DE POSGRADOS, 2021. ¿Por Qué Son Importantes Los Frameworks? | 2022. [en línea]. [Consulta: 17 noviembre 2022]. Disponible en: https://posgradosadistancia.com.ar/por-que-son-importantes-los-frameworks/.

CILLERO, M., 2019. Diagrama de secuencia. *manuel.cillero.es* [en línea]. [Consulta: 29 noviembre 2022]. Disponible en: https://manuel.cillero.es/doc/metodologia/metrica-3/tecnicas/diagrama-de-interaccion/diagrama-de-secuencia/.

CRESPO, G., 2021. Estándares de codificación Python y uso de AutoPEP8 - programador clic. [en línea]. [Consulta: 29 noviembre 2022]. Disponible en: https://programmerclick.com/article/23781149295/.

DARIAS, S., 2021. Gestor de Base de datos: Qué es, Funcionalidades y Ejemplos. [en línea]. [Consulta: 29 noviembre 2022]. Disponible en: https://intelequia.com/blog/post/2949/gestor-de-base-de-datos-qu%C3%A9-es-funcionalidades-y-ejemplos.

DEYIMAR, 2020. ¿Qué es Bootstrap? - Una guía para principiantes. *Tutoriales Hostinger* [en línea]. [Consulta: 17 noviembre 2022]. Disponible en: https://www.hostinger.es/tutoriales/que-es-bootstrap.

DJANGO, 2022. Django. *Django Project* [en línea]. [Consulta: 17 noviembre 2022]. Disponible en: https://www.djangoproject.com/.

DOCSPYTHON, 2022. El tutorial de Python — documentación de Python - 3.11.0. [en línea]. [Consulta: 17 noviembre 2022]. Disponible en: https://docs.python.org/es/3/tutorial/index.html.

FACTORIALHR, 2022. Software para la gestión de turnos - Factorial. [en línea]. [Consulta: 16 noviembre 2022]. Disponible en: https://factorialhr.es/gestion-turnos.

FONT,ALBERTO, 2021. ¿Qué es la prueba funcional? - definición de techopedia - Desarrollo 2022. *Icy Science* [en línea]. [Consulta: 28 noviembre 2022]. Disponible en: https://es.theastrologypage.com/functional-testing.

FRANCO COLMENA, 2020. Qué es Bootstrap y cuáles son sus beneficios en el desarrollo web. [en línea]. [Consulta: 17 noviembre 2022]. Disponible en: https://sebcreativos.es/que-es-bootstrap/.

HERRERA GONZÁLEZ, D. de la C., 2019. *Sistema para la gestión de la guardia obrera en la Facultad 1 de la Universidad de las Ciencias Informáticas* [en línea]. bachelorThesis. S.l.: Universidad de las Ciencias Informáticas. Facultad 1. [Consulta: 17 noviembre 2022]. Disponible en: https://repositorio.uci.cu/jspui/handle/123456789/10009.

INTELEQUIA, 2021. Gestor de Base de datos: Qué es, Funcionalidades y Ejemplos. *Intelequia* [en línea]. [Consulta: 18 noviembre 2022]. Disponible en: https://intelequia.com/blog/post/2949/gestor-de-base-de-datos-qué-es-funcionalidades-y-ejemplos.

INTERNETYA, 2016. Ventajas y beneficios de las aplicaciones Web. *INTERNET YA* [en línea]. [Consulta: 17 noviembre 2022]. Disponible en: https://www.internetya.co/ventajas-y-beneficios-de-las-aplicaciones-web/.

IONOS, 2018. UML, lenguaje de modelado gráfico. *IONOS Digital Guide* [en línea]. [Consulta: 18 noviembre 2022]. Disponible en: https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/uml-lenguaje-unificado-de-modelado-orientado-a-objetos/.

KEEPCODING, R., 2021. Ventajas y Desventajas de Python | KeepCoding Tech School. [en línea]. [Consulta: 17 noviembre 2022]. Disponible en: https://keepcoding.io/blog/ventajas-y-desventajas-de-python/.

KEEPCODING, R., 2022. ¿Qué son las pruebas unitarias de software? [en línea]. [Consulta: 28 noviembre 2022]. Disponible en: https://keepcoding.io/blog/que-son-las-pruebas-unitarias-de-software/.

KHATRI, V., 2020. Herramientas de prueba de software. *Geekflare* [en línea]. [Consulta: 18 noviembre 2022]. Disponible en: https://geekflare.com/es/software-testing-tools/.

LARMAN, 2016. LIBRO: UML y Patrones - 2da Edicion por Craig Larman. [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: https://groups.google.com/g/uptospnfi/c/iZltQJ37xsE?pli=1.

LOADVIEW, 2022. Pruebas de carga vs pruebas de estrés - LoadView. [en línea]. [Consulta: 28 noviembre 2022]. Disponible en: https://www.loadview-testing.com/es/pruebas-de-carga-vs-pruebas-de-estres/.

LUIS GABRIEL CHAVEZ RAMOS, 2019. Cohesión y Acoplamiento relacionado con la rigidez, fragilidad e inmovilidad. [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: https://es.linkedin.com/pulse/cohesi%C3%B3n-y-acoplamiento-relacionado-con-la-rigidez-e-chavez-ramos.

MANZ, 2020. ¿Qué es CSS? - CSS en español. [en línea]. [Consulta: 17 noviembre 2022]. Disponible en: https://lenguajecss.com/css/introduccion/que-es-css/.

MARCO CHICAIZA, 2022. Ing. Marco Chicaiza: Herramientas CASE. *Ing. Marco Chicaiza* [en línea]. [Consulta: 18 noviembre 2022]. Disponible en: http://marcochicaiza72.blogspot.com/p/herramientas-case.html.

MENÉNDEZ , CIRO, 2022. ¿Qué son las pruebas funcionales? Definición de SearchSoftwareQuality (Actualizado 2022) - Krypton Solid. [en línea]. [Consulta: 28 noviembre 2022]. Disponible en: https://kryptonsolid.com/que-son-las-pruebas-funcionales-definicion-de-searchsoftwarequality/.

MIRIAM C. GONZÁLEZ, DAVID LÓPEZ AGUILAR, y PEDRO R. ÁLVAREZ PÉREZ, 2009. La enseñanza universitaria y la formación para el trabajo: Un análisis desde la opinión de los estudiantes. [en línea]. [Consulta: 14 noviembre 2022]. Disponible en: http://ve.scielo.org/scielo.php?script=sci_arttext&pid=S1011-22512009000200002.

OPENWEBINARS, 2022. Qué es Visual Studio Code y qué ventajas ofrece. *OpenWebinars.net* [en línea]. [Consulta: 18 noviembre 2022]. Disponible en: https://openwebinars.net/blog/que-es-visual-studio-code-y-que-ventajas-ofrece/.

PABLO HUET, 2022. Arquitectura de software: Qué es y qué tipos existen | OpenWebinars. [en línea]. [Consulta: 22 noviembre 2022]. Disponible en: https://openwebinars.net/blog/arquitectura-de-software-que-es-y-que-tipos-existen/.

PACHECO MORALES, O.A., 2016. *Módulo Guardia Obrera Estudiantil versión 2.0 para el Sistema de Administración y Economía de la Facultad 3* [en línea]. bachelorThesis. S.l.: Universidad de las Ciencias Informáticas. Facultad 3. [Consulta: 16 noviembre 2022]. Disponible en: https://repositorio.uci.cu/jspui/handle/123456789/7778.

PACHECO, R., 2016. Caracteristicas de una aplicacion web. *Culturación* [en línea]. [Consulta: 17 noviembre 2022]. Disponible en: https://culturacion.com/caracteristicas-de-una-aplicacion-web/.

PGPLANNING, 2022. Programa de cuadrantes de trabajo. *PGPlanning* [en línea]. [Consulta: 16 noviembre 2022]. Disponible en: https://www.pgplanning.es/.

ROLANDO HERNÁNDEZ, 2008. El paradigma cuantitativo de la investigación científica. | Universo Abierto. [en línea]. [Consulta: 26 noviembre 2022]. Disponible en: https://universoabierto.org/2020/10/01/el-paradigma-cuantitativo-de-la-investigacion-cientifica/.

RONDA ALMAGUER, T., 2010. *Sistema para la gestión de información de la guardia obrera y estudiantil en la Universidad de Holguín "Oscar Lucero Moya"* [en línea]. bachelorThesis. S.l.: Universidad de Holguín, Facultad de ingeniería Matemática e Informática, Departamento Ingeniería Informática. [Consulta: 16 noviembre 2022]. Disponible en: http://repositorio.uho.edu.cu/xmlui/handle/uho/6878.

SÁNCHEZ PEÑO, J.M., 2015. Pruebas de Software. Fundamentos y Técnicas. [en línea]. [Consulta: 24 noviembre 2022]. Disponible en: https://oa.upm.es/40012/.

SANTANDER, 2022. Metodologías de desarrollo de software: ¿qué son? [en línea]. [Consulta: 17 noviembre 2022]. Disponible en: https://www.becas-santander.com/es/blog/metodologias-desarrollo-software.html.

SARMIENTO, F., Johana, 2013. Visión General de los Diagramas de Despliegue. *UML* [en línea]. [Consulta: 24 noviembre 2022]. Disponible en: http://umldiagramadespliegue.blogspot.com./.

SERGIO VERGARA, 2021. ¿Qué son las historias de usuario? *Blog ITDO - Agencia de desarrollo Web, APPs y Marketing en Barcelona* [en línea]. [Consulta: 21 noviembre 2022]. Disponible en: https://www.itdo.com/blog/que-son-las-historias-de-usuario/.

SIRIWARDHANA, S., 2020. Tutorial del Diagrama de Componentes | Guía Completa con Ejemplos. *Blog de Creately* [en línea]. [Consulta: 28 noviembre 2022]. Disponible en: https://creately.com/blog/es/diagramas/tutorial-de-diagrama-de-componentes/.

SQLITE, 2022. SQLite Home Page. [en línea]. [Consulta: 18 noviembre 2022]. Disponible en: https://www.sqlite.org/index.html.

TARLOGIC, 2021. Burp Suite Professional. *Tarlogic Security* [en línea]. [Consulta: 18 noviembre 2022]. Disponible en: https://www.tarlogic.com/es/productos/burp-suite-professional/.

TELLEZ, MARCOS, 2022. 5.2. El patrón de diseño MTV (El libro de Django 1.0). [en línea]. [Consulta: 23 noviembre 2022]. Disponible en: https://uniwebsidad.com/libros/django-1-0/capitulo-5/el-patron-de-diseno-mtv.

TINOCO, O., LÓPEZ, P. y BACALLA, J., 2017. Criterios de selección de metodologías de desarrollo de software. *Industrial Data*, vol. 13, pp. 070. DOI 10.15381/idata.v13i2.6191.

UML, 2017. ¿Qué es UML. [en línea]. [Consulta: 18 noviembre 2022]. Disponible en: http://profesores.fi-b.unam.mx/carlos/aydoo/uml.html.

UNAD, 2019. Diagrama de Clases de Diseño | LENGUAJE DE MODELADO UNIFICADO - UML. [en línea]. [Consulta: 28 noviembre 2022]. Disponible en: https://unadzsurlab.com/UML/U2/diagrama_de_clases_de_diseo.html.

VELÁSQUEZ, S.M., MONTOYA, J.D.V., ADASME, M.E.G., ZAPATA, E.J.R., PINO, A.A. y MARÍN, S.L., 2019. Una revisión comparativa de la literatura acerca de metodologías tradicionales y modernas

de desarrollo de software. *Revista CINTEX*, vol. 24, no. 2, pp. 13-23. ISSN 2422-2208. DOI 10.33131/24222208.334.

ZUÑIGA , LEANDRO, 2020. ¿Qué es una prueba de seguridad? Tipos con ejemplo. *Ebooks Online* [en línea]. [Consulta: 28 noviembre 2022]. Disponible en: https://ebooksonline.es/que-es-una-prueba-de-seguridad-tipos-con-ejemplo/.

Anexos

Anexo 1

1-Entrevista:

Realizada a la Vicedecana de Economía y Administración.

Objetivo: recopilar información sobre el proceso de planificación de la GOE en la Facultad 1 de la UCI.

Preguntas:

1. ¿Como está organizada la GOE?
2. ¿Está conforme con el sistema actual?
3. ¿Mencione las principales limitaciones a la hora de realizar la planificación y el control de la GOE?
4. ¿Cuáles serían los aspectos a tener en cuenta a la hora de la planificación?

Anexo 2

Historias de Usuario

Historia de Usuario	
Número: 1	Nombre del requisito: Autenticar Usuario
Programador: Hector Garcés Rey	Iteración asignada: 1
Prioridad: Baja	Tiempo estimado: 5 h
Riesgo en desarrollo: bajo	Tiempo Real: 7 h
Descripción: Como parte de las medidas de seguridad del sistema, un usuario solo podrá acceder con las credenciales asignadas por el administrador.	

Observaciones: En caso de que el cliente decida dar acceso al sistema a un usuario no registrado debe solicitar al administrador credenciales para el nuevo usuario
Prototipo elemental de interfaz gráfica de usuario:

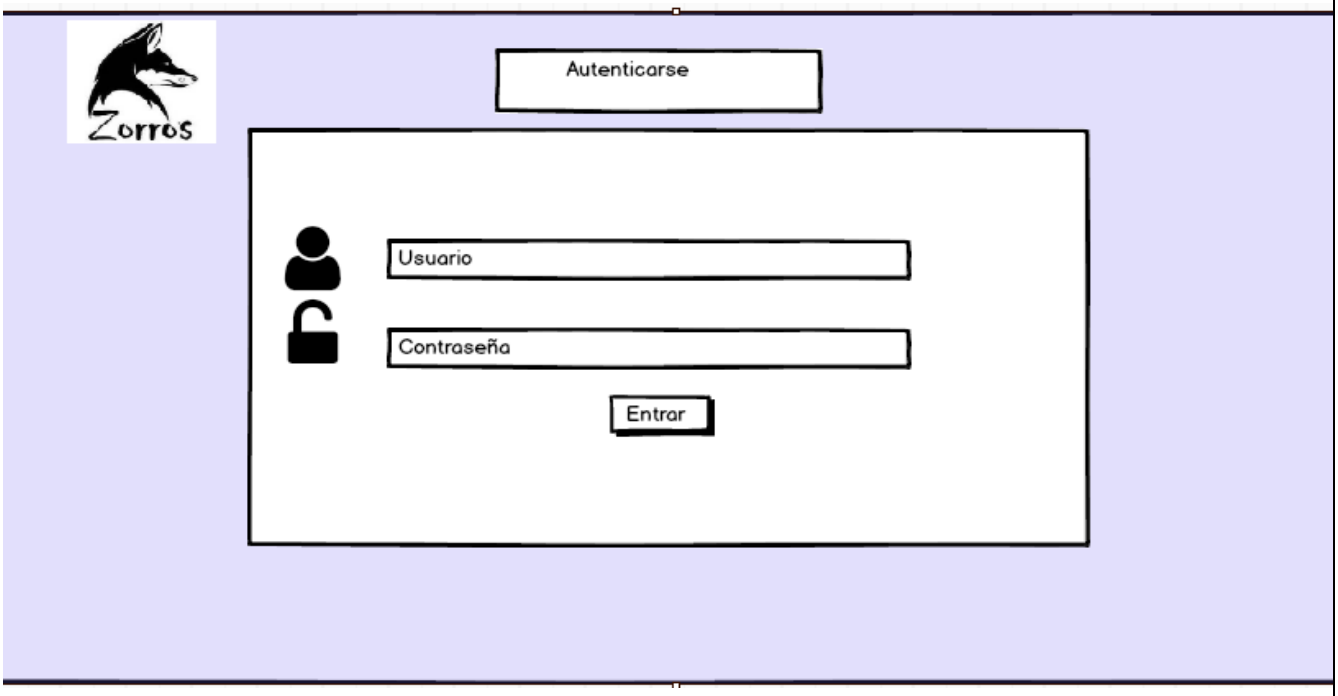


Ilustración 25 Prototipo de interfaz de usuario "Autenticar". Fuente (Elaboración propia)

Historia de Usuario	
Número: 14	**Nombre del requisito**: Permutar guardia
Programador: Hector Garcés Rey	**Iteración asignada**: 5
Prioridad: Media	**Tiempo estimado:** 30h
Riesgo en desarrollo: bajo	**Tiempo Real:** 35 h
Descripción: Dada la concurrencia de 2 permutas previamente acordadas, se podrán intercambiar los turnos de guardia.	
Observaciones: Se deben tener en cuenta las restricciones de asignación a turnos de guardia Prototipo:	

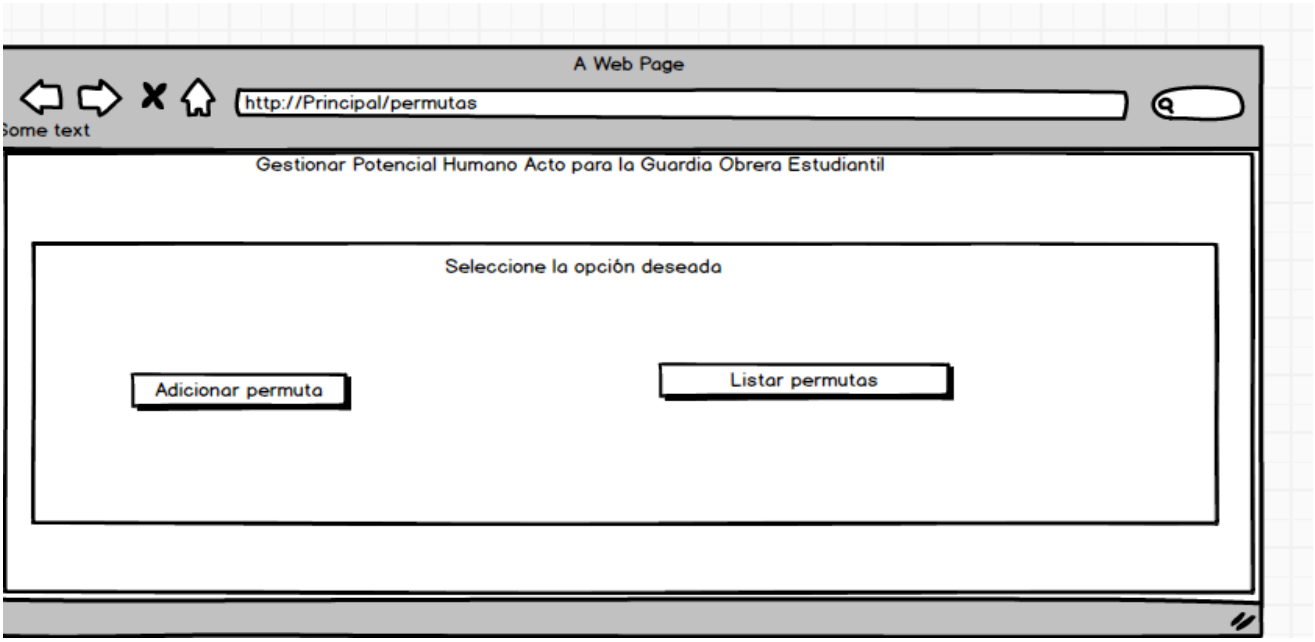


Ilustración 26 Prototipo de RF 14. Permuta de Guardia

Historia de Usuario	
Número: 8	**Nombre del requisito**: Añadir incumplidores de Guardia
Programador: Hector Garcés Rey	**Iteración asignada**: 5
Prioridad: Media	**Tiempo estimado:** 30h
Riesgo en desarrollo: Medio	**Tiempo Real:** 35 h
Descripción: El sistema guardará un registro sobre los incumplidores de la guardia, el usuario debe introducir la información para su posterior gestión.	
Observaciones:	

Historia de Usuario	
Número: 13	**Nombre del requisito**: Crear Reportes de Guardia.
Programador: Hector Garcés Rey	**Iteración asignada**: 2
Prioridad: Media	**Tiempo estimado:** 24h
Riesgo en desarrollo: bajo	**Tiempo Real:** 24h
Descripción: El sistema debe crear reportes de guardia, donde se registren incidencias en los turnos de guardia y el personal ausente	
Observaciones:	

	Historia de Usuario
Número: 12	**Nombre del requisito**: Exportar a Exel cumplimiento de la guardia
Programador: Hector Garcés Rey	**Iteración asignada**: 6
Prioridad: Media	**Tiempo estimado:** 48h
Riesgo en desarrollo: bajo	**Tiempo Real:** 48h
Descripción: El sistema debe brindar la posibilidad de exportar a formato Excel el registro de cumplimiento de la guardia.	
Observaciones: Se podrá añadir al reporte los ausentes a la guardia Prototipo:	

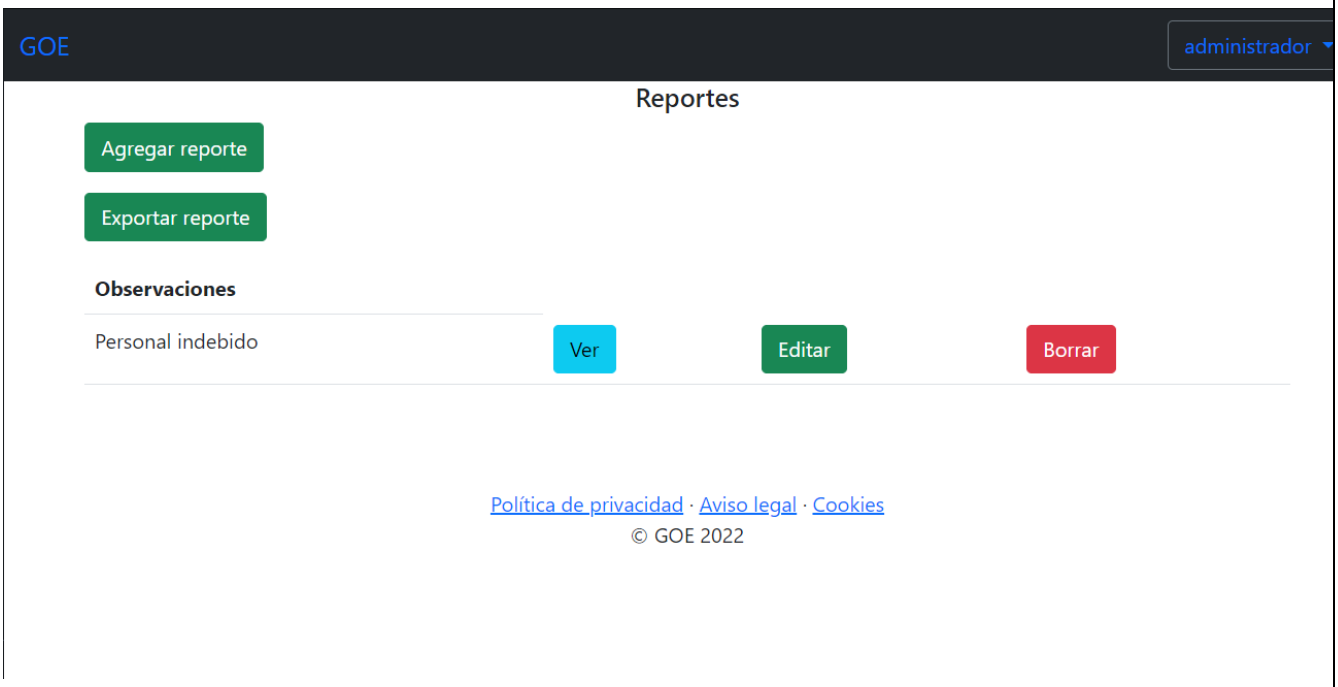


Ilustración 27 Prototipo de interfaz de usuario del RF número 12

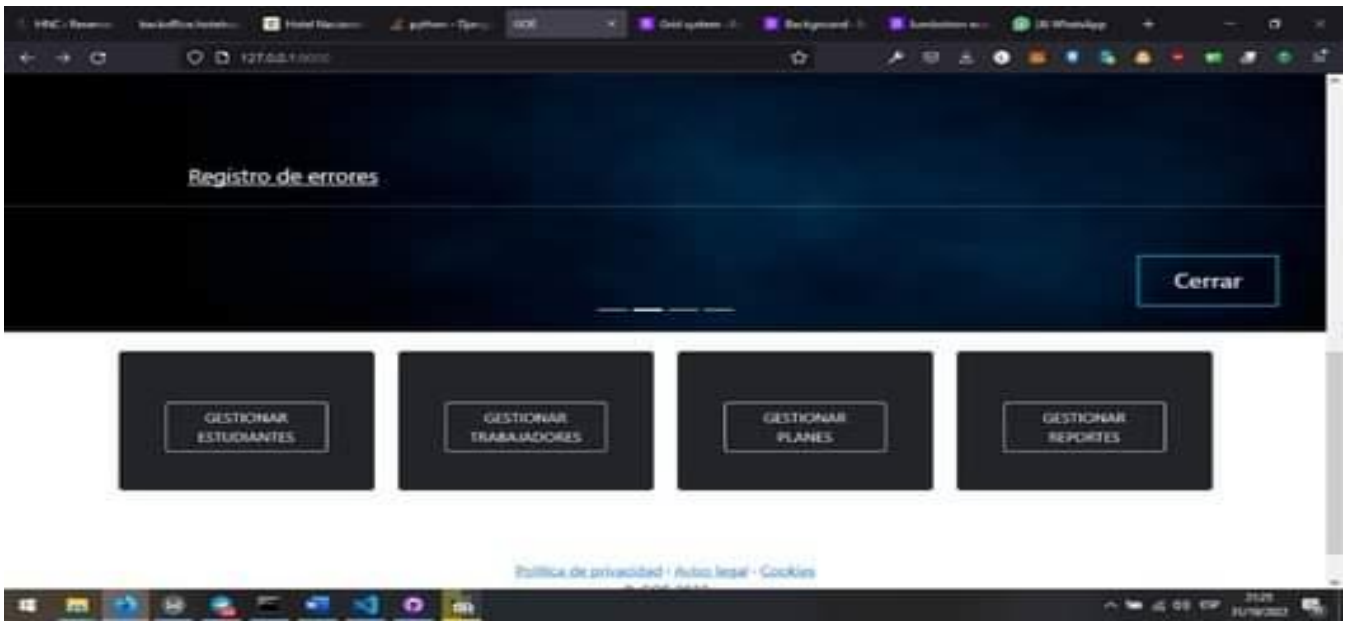


Ilustración 28 Prototipo de interfaz de página principal.