INFORME DE ANÁLISIS

Grupo 1-C1.020 | https://github.com/PDJ6975/Acme-ANS-D01-25.3.0

Nombre	Correo Corporativo
Antonio Rodríguez Calderón	antrodcal@alum.us.es
Adrián Ramírez Gil	adrramgil@alum.us.es
Jianwu Hu	jiahu3@alum.us.es
Pablo Castrillón Mora	pabcasmor1@alum.us.es
Pablo Olivencia Moreno	pabolimor@alum.us.es

19 DE FEBRERO DE 2025

ADRIÁN RAMÍREZ | STUDENT #5

Tabla de Contenido

1. Resumen Ejecutivo 2

2. Tabla de Revisión 2

3. Introducción 2

4. Registro de Análisis. 3

4.1 Relacionados con el entregable 1. 3

4.1.1 Requisito 2: 3

4.2 Relacionados con el entregable 2. 4

4.2.1 Requisito 26: 4

4.2.2 Requisito 3: 5

4.3 Relacionados con el entregable 3. 6

4.3.1 Relación entre entidades MaintenanceRecord y Task. 6

4.4 Relacionados con el entregable 4. 6

4.4.1 Moment en MaintenanceRecord. 6

4.4.2 Moment en MaintenanceRecord. 7

5. Conclusión 8

6. Bibliografía 8

1. Resumen Ejecutivo

Este informe detalla los análisis realizados sobre ciertos requisitos individuales del proyecto, abordando las dudas que surgieron durante su desarrollo, las posibles soluciones evaluadas y las decisiones finales adoptadas. A través de este informe, se busca reflejar el proceso llevado a cabo para garantizar el cumplimiento adecuado de los requisitos, minimizando ambigüedades y asegurando la coherencia con la metodología de trabajo establecida.

2. Tabla de Revisión

Versión	Fecha	Descripción de los cambios
1.0	19/02/2025	Creación inicial del documento
2.0	09/03/2025	Añadir los datos relevantes al segundo entregable.
3.0	16/04/2025	Datos relevantes al tercer entregable.
4.0	26/05/2025	Datos cuarto entregable.

3. Introducción

Durante el desarrollo del proyecto, surgieron dudas sobre la implementación y documentación de ciertos requisitos. Este informe expone dichas inquietudes,

analiza las alternativas consideradas y justifica la solución adoptada. Se sigue una estructura clara:

- Presentación del requisito.

- Evaluación de opciones.

- Justificación de la decisión, basada en criterios técnicos y recomendaciones del profesor.

4. Registro de Análisis.

En este apartado vienen expuestos aquellos requisitos que me han propiciado dudas en el desarrollo del proyecto, las distintas posibilidades consideradas y la decisión tomada en cada caso.

4.1 Relacionados con el entregable 1.

4.1.1Requisito 2: "Provide a link to your planning dashboard in GitHub to review the tasks, their current status, and your schedule"

Cuando leí el requisito me surgió la duda de donde debería de ir el enlace del dashboard en nuestro proyecto a lo que me propuse varias opciones:

- **Opción 1: Colocarlo en el propio proyecto como un nuevo apartado con el enlace al dashboard.**

  Pensé en añadir el enlace del dashboard a la propia página de Acme-ANS como los enlaces de Anonymous como si de otro apartado de enlaces a repositorios se tratase, pero no sabía si esa opción sería acertada en este tipo de proyecto.

- **Opción 2: Añadir el enlace en el fichero "README" del proyecto.**

  Otra opción que se me paso por la mente fue la de incluir en el README los enlaces de los dashboard de cada usuario. No me convenció del todo puesto que no lo veía buena opción que esta información estuviese en este archivo.

- **Opción 3: Poner el enlace en la entrega de la Enseñanza Virtual.**
  Incluirlo en la entrega de la Enseñanza Virtual. No me convencía esta opción puesto que no venía especificada en ningún documento por lo que no termine de verla una opción viable.

En una práctica de laboratorio se le comentó al profesor cual sería la opción correcta para poder entregar el enlace. Su respuesta nos confirmó de varias posibilidades de entrega.

Conclusión: decidimos de forma grupal que una única persona añadiese el enlace al dashboard en el README del proyecto.

4.2 Relacionados con el entregable 2.

4.2.1Requisito 26:

*"The system is required to have a notice board to advertise **courses** for technicians. A web service must be used to populate this entity with information about courses. Thus, the exact data to store depends on the chosen service, and it is the students' responsibility to define them accordingly. It is also the students' responsibility to find the appropriate service; no implicit or explicit liabilities shall be covered by the University of Seville or their individual affiliates if the students hire pay-per-use services! The students are strongly advised to ensure that the service they choose is free of charge."*

Al leer el requisito me surgieron varias dudas sobre los posibles atributos de este requisito. Para afrontarlo me propuse varias alternativas:

- **Opción 1: crearme una propia página web donde tomar los datos a partir de mis propios endpoints.**

  Pensé en crearme mis propios datos puesto que al ser unos cursos tan específicos pocos sitios que ofrezcan una api de pago iba a poder encontrar.

- **Opción 2: usar otro sistema que me ofreciese datos del estilo BeautifulSoup.**

  Sabiendo de la dificultad de encontrar una API de ese estilo de forma gratuita, usando BeautifulSoup conseguiría datos de cursos y así podrías usarlos para conseguir los atributos.

- **Opción 3: buscar una api que ofreciese cursos.**

Buscar APIS que ofreciesen cursos con temática de aviación para poder tomar los datos.

Tras pensarlo mucho decidí buscar Apis que ofreciesen este servicio. Tras muchísima búsqueda encontré EDX que ofrecía una api de donde sacar cursos relacionado con técnicos de avión y mantenimiento. Tras probarla en Postman y usando la CMD observe que las respuestas que ofrecían estaban siempre vacías por falta de permisos que se conseguían pagando.

Descarté esta opción de forma automática y empecé de nuevo la búsqueda encontrando así la Api definitiva. Udemy ofrecía lo mismo que EDX, pero de forma gratuita y tras ponerme en contacto con la empresa y darme las claves necesarias. Conseguí tener un api de donde sacar los datos correspondientes

Conclusión: decidí buscar una api que me proveyese los datos que necesitaba.

4.2.2Requisito 3:

*"The **technicians** care of aircraft maintenance by conducting regular inspections, performing repairs, and carrying out other maintenance tasks. The system must store the following data about them: a **license number** (unique, pattern "^[A-Z]{2-3}\d{6}$"), a **phone number** (pattern "^\+?\d{6,15}$"), their **specialisation** (up to 50 characters), whether they have passed their **annual health test** or not, and their **years of experience**. Optionally, the system may store their **certifications** (up to 255 characters)."*

Cuando leí el requisito me surgió la duda de si añadirlo como entidad o como un realms por lo que me propuse varias opciones para resolverla:

- **Opción 1: mantenerlo como una entidad.**

  De esta forma se gestionarían los datos de una mejor forma.

- **Opción 2: mantenerlo como un real**

  Así cada técnico tendría su UserAccount y gestionaría privilegios y acceso a recursos de una mejor forma.

Tras hablarlo con el profesor en teoría, decidí mantenerlo como un realm para una mejor gestión de permisos y recursos.

Conclusión: mantuve technician como un realm.

4.3 Relacionados con el entregable 3.

4.3.1 Relación entre entidades MaintenanceRecord y Task.

Pensé que sería una relación directa como cualquier relación ManyToOne. Pero me surgió uno de los errores más curioso hasta el momento. Al principio no detectaba bien la relación entre MaintenanceRecord y Task. Por mucho que intentaba en las llamadas a repositorio tomar los datos de la otra entidad, nunca los encontraba. Tras realizar algunas modificaciones conseguí que todo funcionase "correctamente". El frontend y backend respondían bien, pero se torcía cuando se intentaba crear una task dentro de un maintenanceRecord, el backend devolvía un post 200 de que todo se había creado, pero al buscar la nueva task creada esta nunca aparecía por mucho que el backend devolviese un 200. Ante este problema vi una opción clara para solucionarlo:

- **Opción 1: añadir una entidad intermedia.**

  Con esto conseguiría que todo funcionase correctamente.

Tras leerlo en el foro, decidí hacerlo con la entidad intermedia la cual, aunque no sea práctica (porque define una relación ManyToOne completamente) es útil ya que solventa el error comentado anteriormente.

Conclusión: añadí la entidad MaintenanceTask para solventar el problema.

4.4 Relacionados con el entregable 4.

En este entregable me he visto un poco enredado por el tiempo puesto que en este cuatrimestre se me ha juntado un poco el tiempo de la vida laboral, con el hecho de sacarme 8 asignaturas por parciales en esta entrega. Por este motivo en esta entrega he tenido unos cuantos más de problemas.

4.4.1Moment en MaintenanceRecord.

Al realizar el testing de maintenance record observe que no sabía muy bien cómo tratar que era el atributo moment de la entidad, por lo que me surgieron muchas dudas sobre este atributo:

- **Opción 1: usar el momento como una variable que indicaba la creación de la del registro de mantenimiento.**

Con esta opción veía que tenía que controlar varias casuísticas de la edición del maintenance record, como la de entrar a actualizar un atributo que fuese un escrito, si era así la fecha debía de mantenerse. Esto no lo veía 100% seguro de que fuese una opción viable, así que la descarté.

Tras hablarlo con un compañero de clases donde tuvo la misma duda y fue resuelta por nuestro profesor, me dio las pautas a seguir para realizar correctamente el requisito y es tratar a momento como la fecha en la que se crea o actualizar el maintenance record donde debe de ponerla automáticamente el sistema y resolver las casuísticas de un read_only.

Conclusión: añadí momento como read_only en la vista para solventar el problema.

4.4.2 Moment en MaintenanceRecord.

Con las tasks me surge la duda de que mis urls como tal están capadas puesto que una manera que he tenido de proteger el sistema es hacer que si metes una url de un método haces un get y para confirmar debes de darle al botón para ejecutar la acción, con esta función controlaba bastantes cosas hasta que llego el actualizar una task de una maintenace record. Mis tasks solo se pueden crear accediendo a las maintenance record y dándole a task, no hay otra forma de hacerlo. Sin embargo, podemos actualizar desde la página de my tasks o desde la página de maintenance record y accediendo a las tasks posibles. La cosa está en que la url estaba capada para los casos donde cambiabas el id de la task. Si no pertenece al technician o esta publicada no se puede actualizar la cuestión viene si cambiaba el maintenance record a mano de la url, que lo que provocaba que pudiese actualizar tasks de donde quisiese:

- **Opción 1: buscar otra forma de actualización de las tasks.**

  No veía muy bien esta opción ya que iba a necesitar de un tiempo que carecía en ese momento por lo que la descarte automaticamente.

- **Opción 2: capar la url para que ignore el id del maintenance record que pongas.**

  Esta opción me gusto bastante ya que por mucho que tocases el parámetro de maintenance record en la url siempre lo iba a ignorar y a la tarea se creaba para la maintenance record a la que estaba asociada en la tabla intermedia.

No lo pensé mucho y me puse con la opción 2 que, tras acabar de implementarla, y comprobar su funcionamiento, vi que todo estaba bajo control.

Conclusión: arregle el problema capando el parámetro de la url y usando la base de datos como referencia además de proteger intentos de cambio en peticiones maliciosas.

5. Conclusión

Este informe ha registrado las dudas que surgieron durante el desarrollo del proyecto, proponiendo varias alternativas y justificando las decisiones tomadas en cada caso. Se ha recurrido a un enfoque estructurado para evaluar las opciones disponibles y determinar la solución más adecuada en cada situación.

El análisis realizado facilita una interpretación precisa de los requisitos y una correcta aplicación de los criterios definidos. De esta manera, mejoramos la toma de decisiones y optimizamos el proceso de desarrollo, garantizando coherencia y alineación con los objetivos del proyecto.

6. Bibliografía

Intencionalmente en blanco.