# Requirements – Student #1

Please, fill in the following form, make sure that you have ticked the requirements that you consider fulfilled, save this document, **and attach it in its original format (.docx)** to every deliverable. Regarding your ID, please keep only four random digits and mask the others using an asterisk. **Please, note that this document must be edited with the desktop version of Word since the web version does not properly support forms.** Attaching this document entails that you are the authors of the work delivered, you have not cheated in any way, and you have read and understood the information delivered regarding the subject, with a special emphasis on the methodological guidelines and how your work is going to be graded. Make sure that your project works well with the latest version of the development framework.

|  |
| --- |
| **Group:** C2.036 |
| **Repository:** https://github.com/C1-036/Acme-ANS-C2 |
| **Student #2**  **ID Number:** 5\*9\*\*\*5\*M  **UVUS:**  alesevbar  **Name:**  Sevillano Barea, Alejandro  **Roles:**  Developer, Tester, Analyst |
| **Date:** Seville July 04, 2025 |

# MANDATORY Deliverable D01: introduction

## Information requirements

Intentionally blank.

## Functional requirements

1. Modify the anonymous menu so that it shows an option that takes the browser to the home page of your favourite web site. The title must read as follows: “〈id-number〉: 〈surname〉, 〈name〉”, where “〈id-number〉” denotes your DNI, NIE, or passport number, “〈surname〉” denotes your surname/s, and “〈name〉” denotes your name/s.

X

## Non-functional requirements

Intentionally blank.

## Testing requirements

Intentionally blank.

## Managerial requirements

1. Provide a link to your planning dashboard in GitHub to review the tasks, their current status, and your schedule.

X

# MANDATORY Deliverable D02: data models

## Information requirements

1. **Airline managers** are the people responsible for managing flights. The system must handle the following information about **managers**: an **identifier number** (unique, pattern "^[A-Z]{2-3}\d{6}$", where the first two or three letters correspond to their initials), **years of experience** in the airline, his or her **date of birth** and an optional link to a **picture** that must be stored somewhere else.

X ¿Por qué no se ha modelado en el atributo las restricciones de formato para el atributo identifierNumber?

A screenshot of a computer

AI-generated content may be incorrect.

He visto que se hace en la validación de la entidad AirlineManager (AirlineManagerValidator) pero se podría haber puesto como una constraint @ValidIdentifier en vez de @Valid.

En la clase AirlineManager **sí he modelado ya las restricciones de formato a nivel de atributo**

**@ValidIdentifierNumberFormat** es una anotación de campo que implementa internamente:

* 1. **@NotBlank**, para evitar nulos o cadenas vacías.
  2. **@Pattern(regexp = "^[A-Z]{2,3}\\d{6}$")**, para forzar 2–3 letras mayúsculas seguidas de 6 dígitos.
  3. Un message() por defecto que carga la clave i18n acme.validation.identifier.message que muestra el mensaje: “El identificador debe tener entre 2 y 3 letras mayúsculas seguidas de 6 dígitos.”

De este modo **toda la validación de formato** (longitud, caracteres y patrón) queda **en el propio atributo**.

**Reglas de negocio (entidad)**  
A nivel de clase, con @ValidAirlineManager y su AirlineManagerValidator, solo compruebo:

* **Prefijo**: que las dos primeras letras sean las iniciales del nombre y apellido. Si falla, se usa acme.validation.airline-manager.identifier.prefix.message que muestra el siguiente mensaje: “El identificador debe comenzar con tus iniciales.”
* **Unicidad**: que no exista otro AirlineManager con el mismo valor (o, si existe, que sea el mismo registro en actualización). Si falla, se usa acme.validation.airline-manager.identifier.unique.message que muestra el siguiente mensaje: “Ese identificador ya está en uso.”

Así, todo lo que depende **solo del formato** está en el atributo, y lo que requiere el **contexto completo** del objeto (prefijo dinámico y consulta a BD) queda en el validador de entidad.

Así cumplo la indicación de colocar las **restricciones de sintaxis** (regex, no-null) en el atributo, y dejo la **validación de negocio** a nivel de entidad.

1. A **flight** is a scheduled journey made by airlines to transport passengers between two locations. The system must store the following data about them: a **tag** that highlights some feature of the flight such as "the fastest", "the cheapest" (up to 50 characters), an **indication** on whether it requires self-transfer or not, a **cost**, an optional **description** (up to 255 characters). It also stores information that comes from its **legs**, namely: a **scheduled departure** and a **scheduled arrival** that depends on the first scheduled departure moment of the first leg and the scheduled arrival moment of the last leg, the origin and destination **cities** that comes from the city of the airports to which first and last leg refers to, and finally, the **number of layovers**.

X

1. A flight aggregates several **legs**. A leg represents an individual segment of a flight, typically corresponding to layovers or connections. The system must store the following data for each leg: a unique **flight number** (composed of the airline's IATA code followed by four digits, unique), a **scheduled departure** and a **scheduled arrival**, a **duration** in hours, a **status** ("ON TIME", "DELAYED", "CANCELLED", "LANDED"). Additionally, each leg must track the **departure** and **arrival** **airports,** as well as the **aircraft** that will be deployed for the journey.

X lo mismo ocurre en este requisito que en el requisito 3 con flightNumber.

**Por qué no he podido dejar la validación del flightNumber (o del identifierNumber) como un simple @Constraint de campo**

1. **Necesito el “contexto” de la entidad entera**.
   * El patrón de validación no es estático: debe incorporar el código IATA de la aerolínea asociada al AirlineManager.
   * Esa información solo vive en la instancia completa (leg.getFlight().getAirlinemanager().getAirline().getIataCode()), no en el valor puro del String del campo.
2. **Restricción técnica de Bean-Validation**
   * Una constraint de **campo** implementa ConstraintValidator<MiAnotación, String>, por tanto el validador solo recibe el String y no tiene acceso directo al objeto contenedor.
   * Para leer el IATA dinámico tendría que hacer “trucos” avanzados (unwrap de HibernateConstraintValidatorContext, extraer el root-bean, etc.), lo cual complica mucho el código y rompe la coherencia.
3. **Solución coherente: validación a nivel de entidad**
   * Marcamos la anotación con @Target(TYPE) y nuestro ConstraintValidator<ValidLeg, Leg> (o ConstraintValidator<ValidIdentifierAirlineManager, AirlineManager>).
   * En el método isValid(Leg leg, …) ya tenemos la entidad completa, por lo que podemos:
     1. Leer el IATA correspondiente.
     2. Construir el regex dinámico ("^" + iata + "\\d{4}$").
     3. Comprobar unicidad y solapamientos, duración y orden, todo en un único validador de clase.
4. **Beneficios de esta aproximación**
   * **Claridad**: todo el código de negocio y formato dinámico vive en un solo lugar.
   * **Simplicidad**: no hay “hacks” para extraer el bean raíz ni mezclar lógicas de campo con lógicas de entidad.
   * **Único punto de entrada y salida**: en isValid(...) validamos formato, IATA, unicidad y reglas de horario, y retornamos al final !hasErrors().

Por estas razones he decidido **mantener la validación** de formato dinámico y de reglas de negocio en el **validator de entidad** en lugar de intentar trasladarlo como una simple constraint de campo.

## Functional requirements

Intentionally blank.

## Non-functional requirements

Intentionally blank.

## Testing requirements

1. Produce assorted sample data to test your application informally. The data must include two **manager** accounts with credentials “**manager1**/**manager1**” and “**manager2**/**manager2**”. Create an additional manager account with credentials “**manager3/manager3”** that represents a manager with no associated data, except for his or her profile.

X El manager3 no ha sido creado con el realm de Manager porque no aparece nada en el menú:

A screenshot of a computer

AI-generated content may be incorrect.

* Añadí en src/main/resources/data/sample1/airline-manager.csv esta línea para **manager3**:

airline-manager3,user-account-airline-manager-03,AA789101,18,2002/05/20 00:00,http://www.lorem-ipsum.org,airline-03

El CSV se importó y ya aparece el usuario manager3 con el rol de Manager.

## Managerial requirements

1. Provide a link to your planning dashboard in GitHub to review the tasks, their current status, and your schedule.

X

# MANDATORY Deliverable D03: implementing features

## Information requirements

Intentionally blank.

## Functional requirements

1. Operations by **managers** on their **flights**:

* List the flights that they have created and show their details.
* Create, update, or delete their flights. Flights can be updated or deleted as long as they have not been published. For a flight to be published, it must have at least one leg, and all its legs must have been published.

X

1. Operations by **managers** on their **legs**:

* List the legs in their flights ordered by their moments (no other sorting criteria is allowed).
* Show the details of their legs.
* Create and publish a leg.
* Update or delete a leg as long as it is not published.

He intentado crear un flight siendo un manager, he puesto los datos siguientes y el sistema ha lanzado una excepción de autorización:

A screenshot of a computer

AI-generated content may be incorrect.

A close-up of a text

AI-generated content may be incorrect.

Con datos correctos tampoco lo permite:

A screenshot of a computer

AI-generated content may be incorrect.

Aquí paramos la evaluación

----------------------------------------------------Comentario-Student1----------------------------------------------------

*He subido el vídeo a la plataforma USE para que se entienda mejor todo y también voy a dejar un comentario detallando mi interpretación de la corrección. He interpretado que quería decir crear un Leg en vez de un Flight, porque en la imagen se muestra la información de un Leg. No obstante, me gustaría exponer de manera formal las incidencias detectadas:*

**Patricia Jiménez**: en efecto, me refería a leg en vez de a flight, disculpa por la confusión.

***1. Imagen con datos incorrectos***

***- FlightNumber y airline****: Se está probando el número de vuelo con un código IATA desconocido para mí, pues no aparece en mi fichero CSV de airline. En mi proyecto cada manager está asociado a una aerolínea: el manager 1 a códigos IATA que comienzan por DGX y el manager 2 a los que comienzan por VZX, seguidos de cuatro dígitos.*

***- Aeropuertos y aeronave****: Sucede lo mismo con los campos Departure Airport, Arrival Airport y Aircraft: no existen esos valores en mis CSV, por lo que no puedo validarlos.*

*-* ***Fechas de programación de legs****: Asumo que en Scheduled Departure y Scheduled Arrival se introdujeron fechas en el pasado para provocar el error de validación “La salida programada no puede ser en el pasado”, el cual sí contempla mi implementación.*

***- Origen de los datos****: En mi proyecto no existe un servicio (createService) que permita crear instancias de Airport o Aircraft; por ello, supongo que los valores “Seville” “Brisbane” y “Aircraft-test-1” fueron insertados directamente en el CSV por usted.*

***- Acceso no autorizado****: Dado que esos registros no aparecen en mis CSV, el sistema los trata como si un usuario los hubiese inyectado manualmente (por ejemplo, a través de las herramientas de desarrollo), desencadenando un bloqueo por posible intento de intrusión.*

**Patricia Jiménez**: así es, he creado nuevos datos y he realizado todas las operaciones con el manager1/manager1. Tenga en cuenta que el manager3 no estaba disponible como manager porque no se ha creado correctamente y esto debe de resolverse para la siguiente convocatoria. Si se introducen los datos que indiqué, el sistema no devuelve ningún error de validación, devuelve un acceso no autorizado. Si no se ha indicado correctamente el IATA code de la airline correspondiente, debería de informarse con un error de validación para hacérselo saber al usuario que está manejando el sistema. Como indicaba en este informe, he introducido datos incorrectos y ha saltado un error de autorización, pero no estaba haciendo ningún tipo de hacking, por lo que deberían de mostrarse los errores de validación y no un acceso denegado.

Le comento que he vuelto a revisar y seguido los pasos que le he indicado y he obtenido los mismos resultados. Le describo los pasos:

1. Me he logueado como administrador.

2. He creado una aerolínea a la que he llamado Airline Test con código IATA AAT (indico los datos en la siguiente captura):

A screenshot of a computer

Description automatically generated

3. A continuación, he creado una aircraft (adjunto los datos en la siguiente captura):

A blank lined paper with a couple of people standing

Description automatically generated with medium confidence

4. Finalmente he creado un flight y posteriormente he intentado crear una leg para ese flight y he obtenido el error de autorización (fíjese que ahora he usado airports que ya estaban en su BD):

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

¿El problema puede deberse a la nueva airline o la nueva aircraft que he creado? podría ser. Tenga en cuenta que cada vez que he puesto algún dato, ha sido añadiendo entidades a través de la interfaz o modificando algún dato del csv, nada más. En el caso de los aeropuertos de Sevilla y Brisbane, simplemente modifiqué en la BD sus nombres para que fuera más fácil localizarlos.

Revisando su proyecto de nuevo, también he detectado que creando un flight y una leg correctamente (tal y como lo ha hecho Ud. en el vídeo), si luego hago clic en editar y cambio las fechas al pasado y en vez de actualizar, hago clic en publicar, la leg se publica con fechas en el pasado. Esto no debería de estar permitido y por tanto, el requisito tampoco estaría satisfecho por este motivo. Por favor, compruebe cada una de las acciones que puede realizar en el sistema exhaustivamente para no cometer los mismos errores en la siguiente convocatoria.

***2. Imagen con datos correctos***

***- Código IATA, aeropuertos y aeronave****: El código IATA empleado no figura en mi CSV de airline; desconozco si usted lo creó específicamente para la corrección. Lo mismo ocurre con los aeropuertos de origen y destino, que no están presentes en mis datos y por tanto no puedo validarlos en mi proyecto.*

***- Acceso no autorizado****: Dado que dichos registros(Departure Airport y Arrival Airport) no existen en mis CSV, el sistema los interpreta como si un usuario los hubiese inyectado manualmente (por ejemplo, mediante las herramientas de desarrollo del navegador), provocando un bloqueo automático por posible intento de intrusión.  
  
Le agradezco de antemano el tiempo y la atención que dedicará a revisar estas incidencias. Quedo a su disposición para cualquier aclaración o ajuste que considere necesario.  
  
Muchas gracias.*

He comprobado que llevaba razón en lo que decía, pero sí que se puede editar una leg

* **Autorización de creación de “Leg”**  
  En authorise() pasé de usar consultas que sólo aceptaban aeropuertos y aeronaves ya ligados al manager, a comprobaciones genéricas de existencia en la BD:

*boolean validDep = depId == 0 || repository.existsAirportById(depId);*

*boolean validArr = arrId == 0 || repository.existsAirportById(arrId);*

*boolean validPlane = planeId == 0 || repository.existsAircraftById(planeId);*

**Antes** usaba

*repository.findAirportById(depId) != null*

*repository.existsAircraftOfManager(managerId, planeId)*

lo que impedía usar aeropuertos o aviones recién creados por admin.

Adicionalmente, me gustaría puntualizar que, por lógica de negocio la entidad AirlineManager está mapeada con @ManyToOne hacia la entidad Airline. Esto garantiza que cada gestor únicamente pueda operar sobre recursos — vuelos, legs, aeronaves, aeropuertos, etc. — cuyo código IATA coincida con el de la aerolínea a la que está asignado. Dicha restricción se materializa en la capa de validación: si se intenta emplear un IATA distinto al autorizado para ese manager, el sistema arroja un mensaje de validación indicando que no es el IATA de la aerolínea a la cual está asociado ese manager.   
  
Le ruego disculpe si mi explicación anterior no fue lo suficientemente clara y le agradezco sinceramente su atención y ayuda.

## Non-functional requirements

Intentionally blank.

## Testing requirements

Intentionally blank.

## Managerial requirements

1. Provide a link to your planning dashboard in GitHub to review the tasks, their current status, and your schedule.

X

# MANDATORY Deliverable D04: formal testing

## Information requirements

1. Create appropriate indices for your entities, if required.

X

## Functional requirements

Intentionally blank.

## Non-functional requirements

Intentionally blank.

## Testing requirements

1. Produce a test suite for Requirements #8 and #9.

X

## Managerial requirements

1. Provide a link to your planning dashboard in GitHub to review the tasks, their current status, and your schedule.

X

1. Produce a testing report.

X

# SUPPLEMENTARY I Deliverable D01: introduction

## Information requirements

Intentionally blank.

## Functional requirements

Intentionally blank.

## Non-functional requirements

Intentionally blank.

## Testing requirements

Intentionally blank.

## Managerial requirements

Intentionally blank.

# SUPPLEMENTARY I Deliverable D02: data models

## Information requirements

1. The system must handle **manager** **dashboards** with the following **indicators**:

* The ranking the manager achieves based on their years of experience. The more years of experience, the higher the position in the ranking.
* The number of years to retire, assuming that they retire at 65.
* Ratio of on-time and delayed legs.
* The most popular and less popular airports within their flights. An airport is popular as long as it has been an origin or destination for many flights.
* The number of legs of their flights grouped by their status.
* The average, minimum, maximum, and standard deviation of the cost of their flights.

X

## Functional requirements

Intentionally blank.

## Non-functional requirements

Intentionally blank.

## Testing requirements

Intentionally blank.

## Managerial requirements

1. Produce a UML domain model regarding the information requirements.

X

# SUPPLEMENTARY I Deliverable D03: implementing features

## Information requirements

Intentionally blank.

## Functional requirements

1. Operations by **anonymous principals** on **user accounts**:

* Sign up to the system and become a manager.

X

1. Operations by **managers** on **user** **accounts**:

* Update their profiles.

X

1. Operations by **any principals** on **flights**:

* List the flights in the system that are published.
* Show the details of the flights that they can list (including their legs).

X

1. Operations by **managers** on **manager dashboards**:

* Show their manager dashboards.

## Non-functional requirements

Intentionally blank.

## Testing requirements

Intentionally blank.

## Managerial requirements

1. Provide a link to a video in which you informally test Requirements #8 and #9. Videos should not exceed 10 minutes in length and must be stored at the USE's facilities.

# SUPPLEMENTARY I Deliverable D04: formal testing

## Information requirements

Intentionally blank.

## Functional requirements

Intentionally blank.

## Non-functional requirements

Intentionally blank.

## Testing requirements

1. Perform five mutations in your code and report on the results.

## Managerial requirements

1. Produce a lint report.

# SUPPLEMENTARY II Deliverable D01: introduction

## Information requirements

Intentionally blank.

## Functional requirements

Intentionally blank.

## Non-functional requirements

Intentionally blank.

## Testing requirements

Intentionally blank.

## Managerial requirements

1. Produce an analysis report.

X

1. Produce a planning and progress report.

X

# SUPPLEMENTARY II Deliverable D02: data models

## Information requirements

1. The system must track **weather conditions**. A web service must be used to populate this entity with information about weather conditions. 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 contract pay-per-use services!  The students are strongly advised to ensure that the service they choose is free of charge.

## Functional requirements

Intentionally blank.

## Non-functional requirements

Intentionally blank.

## Testing requirements

Intentionally blank.

## Managerial requirements

1. Produce an analysis report.

X

1. Produce a planning and progress report.

X

# SUPPLEMENTARY II Deliverable D03: implementing features

## Information requirements

Intentionally blank.

## Functional requirements

1. Operations by **any principals** on **weather conditions**:

* List the forecast weather conditions and show their details.
* List the flights that took place in the last month under bad weather conditions.

1. Operations by **administrators** on **weather conditions**:

* Populate the database with forecast weather conditions.

## Non-functional requirements

Intentionally blank.

## Testing requirements

Intentionally blank.

## Managerial requirements

1. Produce an analysis report.

1. Produce a planning and progress report.

# SUPPLEMENTARY II Deliverable D04: formal testing

## Information requirements

Intentionally blank.

## Functional requirements

Intentionally blank.

## Non-functional requirements

Intentionally blank.

## Testing requirements

1. Produce as a complete test suite as possible for Requirement #29 ensuring that the API is properly mocked.

## Managerial requirements

1. Produce an analysis report.

1. Produce a planning and progress report.