

AcadeMice

Página web de videocursos



Hecho por: Verónica Sánchez Justicia

Índice

[Base de datos](#)

[Diagrama entidad relación](#)

[Usuarios](#)

[Analítica](#)

[Imágenes del uso de Google Analytics:](#)

[Inicio:](#)

[Configuración:](#)

[Instalación:](#)

[Ejemplo aplicado de instalación:](#)

[Funnel](#)

[Público objetivo](#)

[Descripción del público objetivo](#)

[Horas y tareas](#)

[Recuento de horas](#)

[Seguimiento de las tareas](#)

[GIT](#)

[Casos de uso e Historias de Usuario](#)

[Historias de Usuario](#)

[Diagrama de casos de uso](#)

[Monetización](#)

[Mockup](#)

[Notas](#)

[Page pg_welcome](#)

[Page pg_help](#)

[Page pg_recovery_pw_petition](#)

[Page pg_recovery_pw_petition](#)

[Page pg_my_account](#)

[Page page_payment_history](#)

[Page pg_deleted_account](#)

[Page pg_account_config](#)

[Page pg_course_list - home](#)

[Form form_login](#)

[Form form_signup](#)

[Form form_contact](#)

[Form form_account_delete](#)

[Form form_update_account](#)

[Form form_change_pw](#)

[Form form_update_img_account](#)

[Form form_complaint](#)

[Form form_block_user](#)

[Estilo Gráfico \(ADD\)](#)

[Logo](#)

Colores

--clr-white:#fff;

--clr-black: #000000;

--clr-grey: #474747;

--clr-yellow:#FFCB70;

--clr-pink:#C751C0;

--clr-blue:#4158D0;

Usos de los colores

Navegación

Botones

Formularios

Vista detalle de un curso

Vista maestro de los cursos

Calendario de Builds

Quality Assurance

Pirámide de testing invertida:

Cuando se han hecho los test

Accesibilidad

Presupuesto

Tecnologías

Base de datos

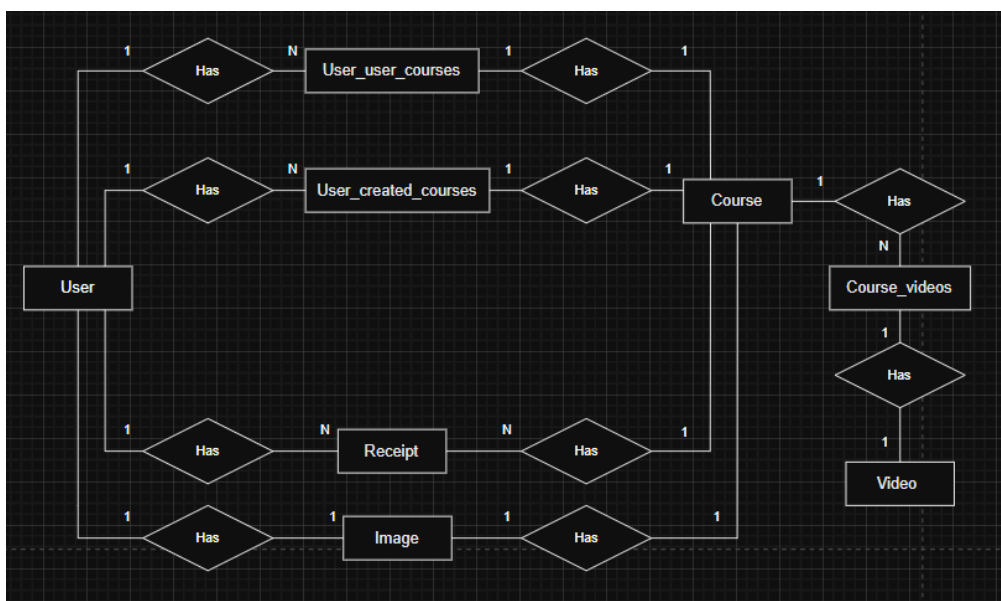
La base de datos está hecha en MySQL. Contamos con las siguientes tablas y las siguientes columnas:

- Course
 - **Atributos:** coin, is_holded, is_public, price, punctuation, creation_date, deletion_date, **id**, **img_id**, last_update, description, name
- Course_videos
 - **Atributos:** course_id, videos_id
- Image
 - **Atributos:** id, name, img
- Receipt
 - **Atributos:** is_already_paid, price, **course_id**, date, **id**, **alumn_username**,
- User
 - **Atributos:** is_professor, **avatar_id**, authorities, email, iban, password, titular, **username**, lang
- User_created_courses
 - **Atributos:** created_courses_id, user_username
- User_user_courses
 - **Atributos:** user_courses_id, user_username
- Video
 - **Atributos:** id, title

Los IDs de cada tabla están marcados en negrita

Diagrama entidad relación

Los atributos de las tablas no se han especificado en el diagrama para que se pueda entender con más facilidad. Este diagrama entidad relación está normalizado.



Usuarios

Tanto la web como la base de datos soportan múltiples conexiones simultáneas.

Contamos con dos tipos de usuario: Genérico y de Profesor.

Usuario Genérico:

Todos los usuarios se generan como un usuario genérico. Tiene permisos para comprar y consumir cursos.

Usuario de Profesor:

Un usuario de Profesor es como un usuario genérico, pero tiene permisos para crear cursos. Cuando un usuario cambia de genérico a profesor debe introducir sus datos bancarios para poder recibir los pagos de sus cursos.

Desde un dispositivo solo se puede tener una cuenta abierta simultáneamente. Esto es debido a que solo se genera una Cookie de inicio de sesión y se va sobrescribiendo.

Hay un usuario especial que se llama “helpAcademice”, este es un usuario de profesor dedicado a gestionar la página de Ayuda. Los cursos subidos por esta cuenta son visibles por todos los usuarios a través de la página de Ayuda.

Analítica

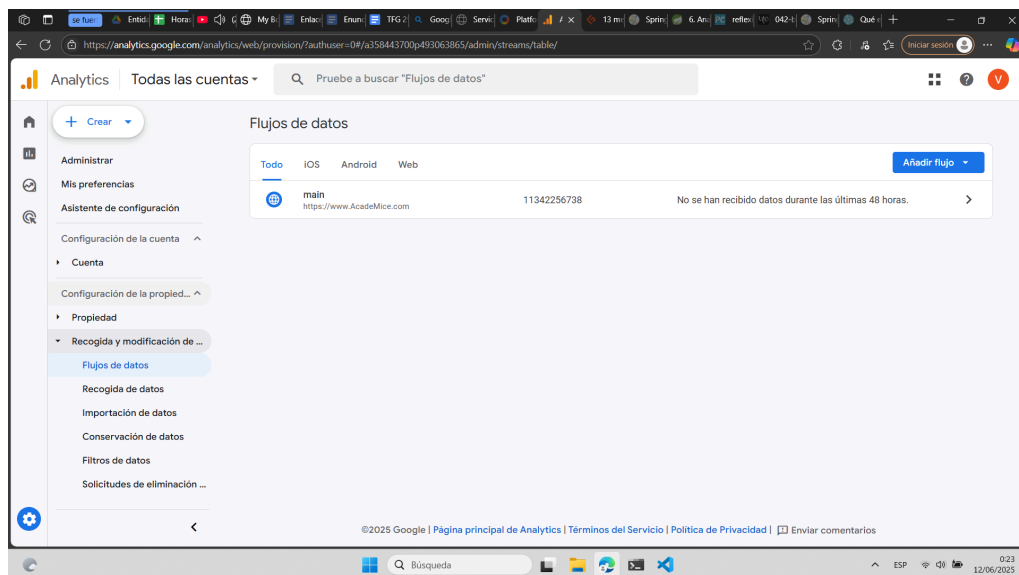
La plataforma utilizada para hacer analíticas de consumo es Google Analytics. Las librerías para hacer la analítica son las ofrecidas por Google Analytics.

He buscado librerías específicas para Spring, pero lo más cercano que he encontrado es que puedo hacer eventos a mano y registrarlos en la base de datos con una nueva tabla. Por suerte encontré que google Analytics te hace ese registro de forma automática.

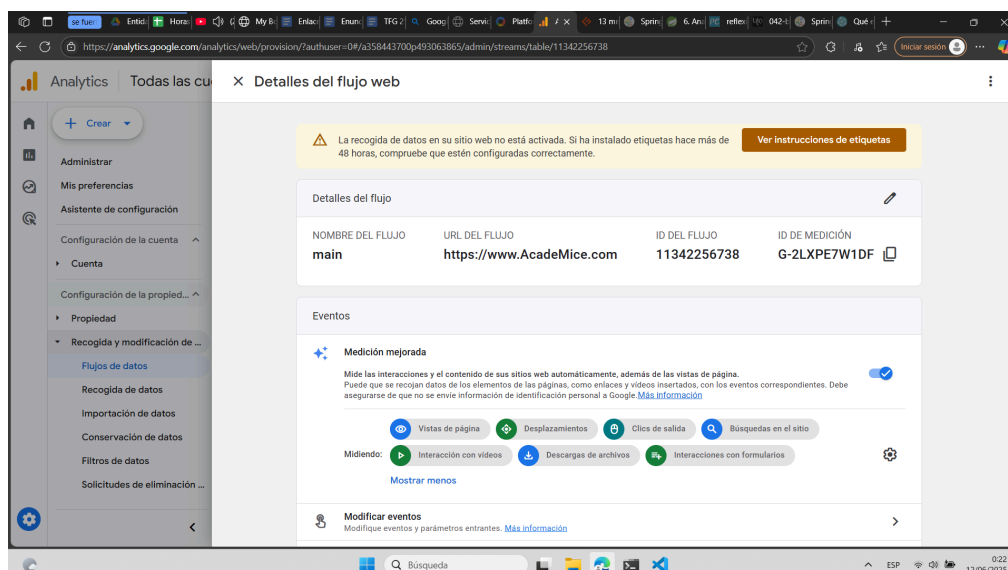
Imágenes del uso de Google Analytics:

Aquí dejo algunas imágenes del uso de Google Analytics:

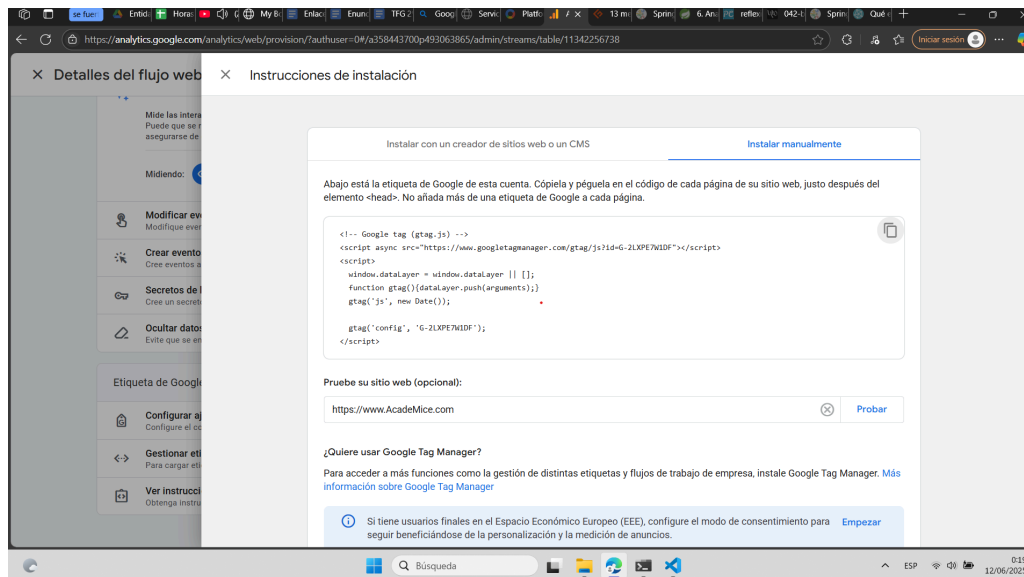
Inicio:



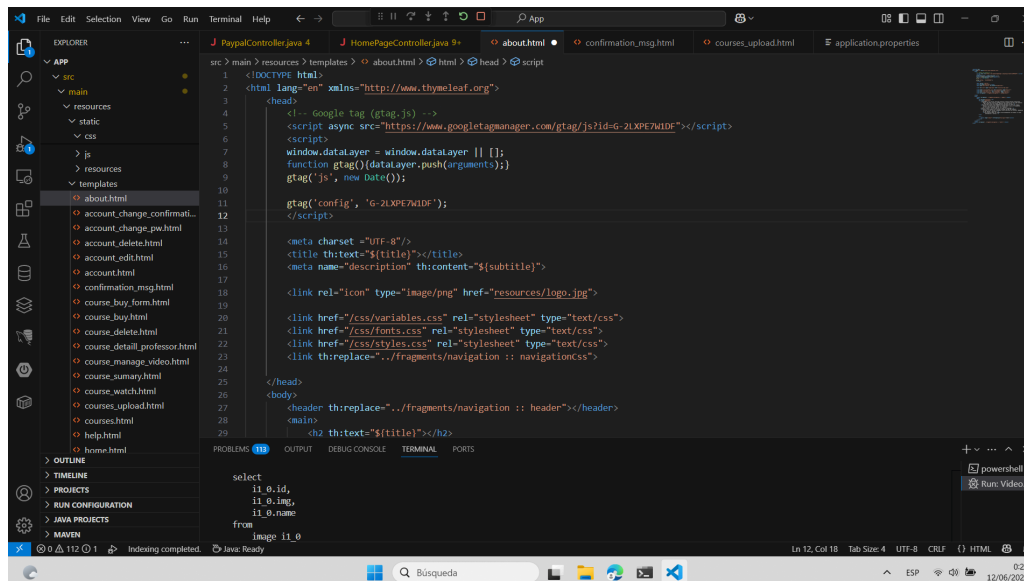
Configuración:



Instalación:



Ejemplo aplicado de instalación:

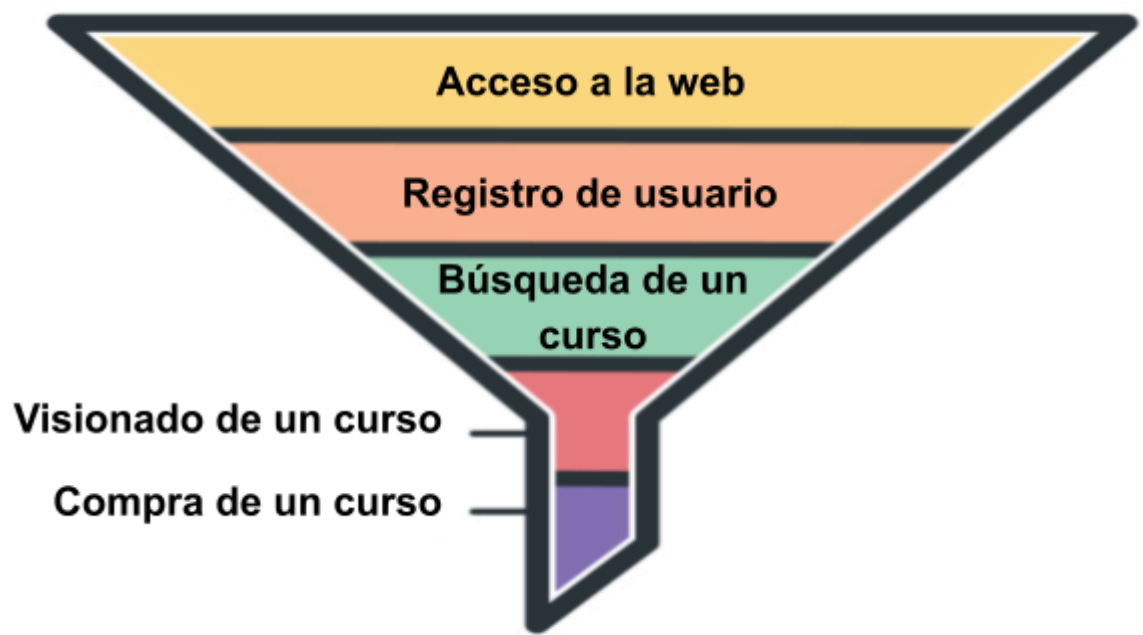


Funnel

Las etapas del Funnel interesantes a tener en cuenta para mi aplicación son las siguientes:

- Acceso a la web
- Registro de usuario
- Búsqueda de un curso
- Visionado de un curso
- Compra de un curso

La estimación de como quedaria el Funnel es la siguiente:



Público objetivo

La Aplicación está enfocada para un público muy diverso. Uno de los objetivos durante la creación de la interfaz de usuario ha sido hacerla tan sencilla que la pueda utilizar cualquier persona desde cualquier dispositivo.

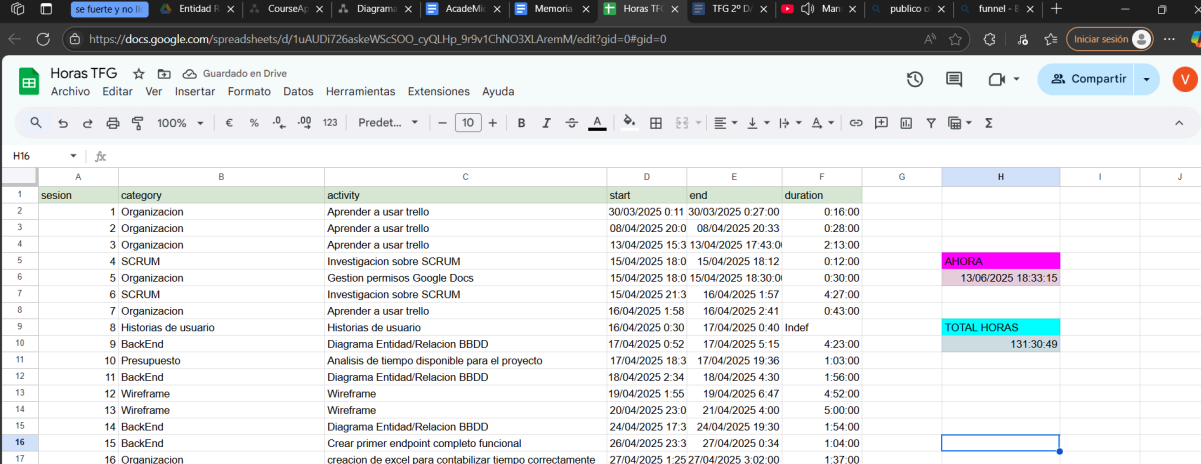
Descripción del público objetivo

- Edad: Entre 18 y 35 años
- Género: Cualquiera
- Nivel Socioeconómico: Personas que llegan a fin de mes sin demasiada dificultad pero no tienen demasiados ahorros
- Motivaciones de uso:
 - Quieren buscar un trabajo mejor o que se les valore mejor en su empresa
 - Les apasiona algún campo
- Valores:
 - Les gusta dar lo mejor de sí mismos
 - Saben valorar económicamente el trabajo de los demás
 - Disfrutan de estudiar

Horas y tareas

El recuento de horas se ha hecho con Google sheets y el seguimiento de las tareas se ha hecho con trello.

Recuento de horas

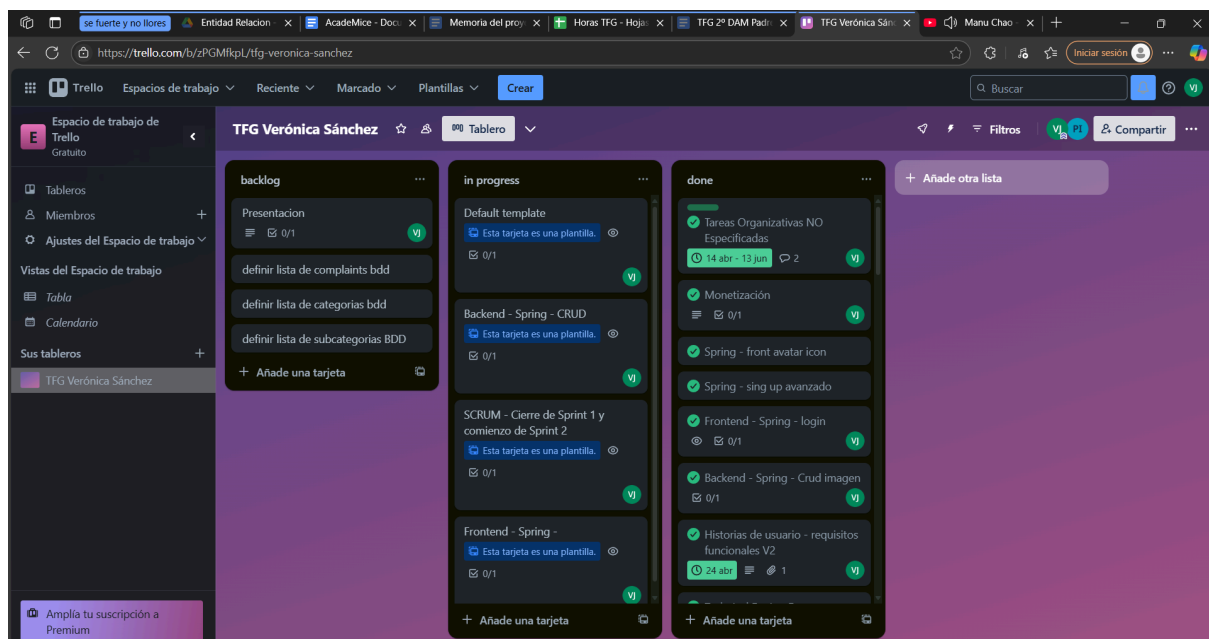


sesion	category	activity	start	end	duration
1	Organization	Aprender a usar trello	30/03/2025 0:11	30/03/2025 0:27:00	0:16:00
2	Organization	Aprender a usar trello	08/04/2025 20:0	08/04/2025 20:33	0:28:00
3	Organization	Aprender a usar trello	13/04/2025 15:3	13/04/2025 17:43:0	2:13:00
4	SCRUM	Investigacion sobre SCRUM	15/04/2025 18:0	15/04/2025 18:12	0:12:00
5	Organization	Gestion permisos Google Docs	15/04/2025 18:0	15/04/2025 18:30:0	0:30:00
6	SCRUM	Investigacion sobre SCRUM	15/04/2025 21:3	16/04/2025 1:57	4:27:00
7	Organization	Aprender a usar trello	16/04/2025 1:58	16/04/2025 2:41	0:43:00
8	Historias de usuario	Historias de usuario	16/04/2025 0:30	17/04/2025 0:40	Indef
9	BackEnd	Diagrama Entidad/Relacion BBDD	17/04/2025 0:52	17/04/2025 5:15	4:23:00
10	Presupuesto	Analisis de tiempo disponible para el proyecto	17/04/2025 18:3	17/04/2025 19:36	1:03:00
11	BackEnd	Diagrama Entidad/Relacion BBDD	18/04/2025 2:34	18/04/2025 4:30	1:56:00
12	Wireframe	Wireframe	19/04/2025 1:55	19/04/2025 6:47	4:52:00
13	Wireframe	Wireframe	20/04/2025 23:0	21/04/2025 4:00	5:00:00
14	BackEnd	Diagrama Entidad/Relacion BBDD	24/04/2025 17:3	24/04/2025 19:30	1:54:00
15	BackEnd	Crear primer endpoint completo funcional	26/04/2025 23:3	27/04/2025 0:34	1:04:00
16	Organization	creacion de excel para contabilizar tiempo correctamente	27/04/2025 1:25	27/04/2025 3:02:00	1:37:00

Se puede acceder al documento en este enlace:

- [Horas TFG - Hojas de cálculo de Google](https://docs.google.com/spreadsheets/d/1uAUDI726askeWScSOO_cyQLHp_9r9v1ChNO3XLArEM/edit?gid=0#gid=0)

Seguimiento de las tareas



Se puede acceder al tablero de trello a través del siguiente enlace:

- <https://trello.com/invite/b/67e87e2cebe067fb3a638f73/ATT1b2802d7be5cd42e4455ea5081d36bfd269576082/tfg-veronica-sanchez>

GIT

Se ha utilizado GitHub para hacer control de versiones del proyecto. Además se ha usado la librería Git Flow para hacer el control de las ramas. Se han usado los siguientes tipos de rama:

- **Main** -> contiene la versión final
- **Develop**-> contiene la versión en desarrollo. Cuando se termina una version se hace merge en la rama Main
- **Feature**-> contiene la funcionalidad en desarrollo. Se genera una nueva rama feature para cada funcionalidad en desarrollo. Cuando se termina una funcionalidad, se testea todo el código y se hace merge en la rama Develop

El repositorio se puede encontrar en el siguiente enlace:

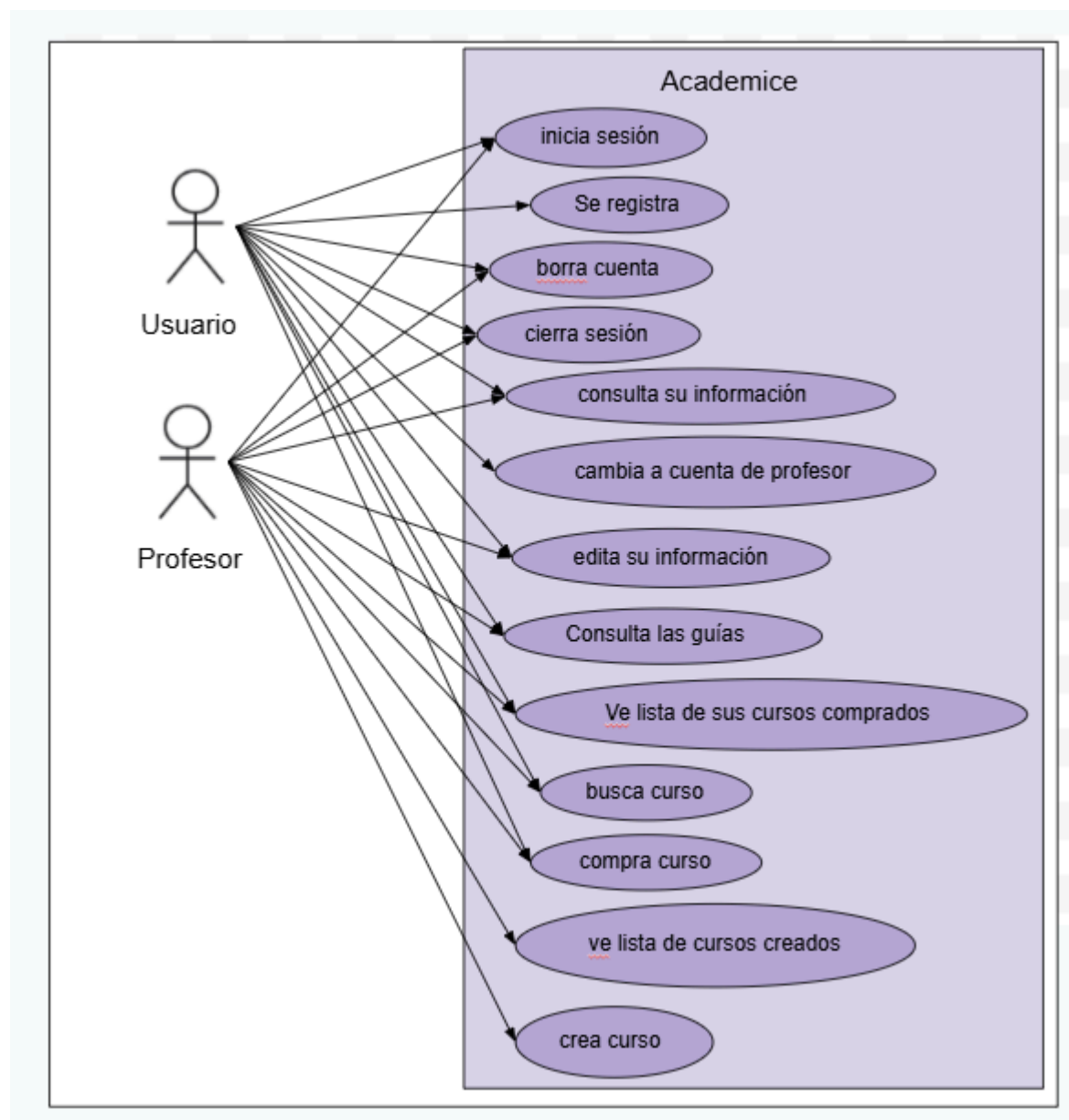
- [VeronicaSj/VideoCoursePlataform: This is my Video Course Plataform.](#)

Casos de uso e Historias de Usuario

Historias de Usuario

- La página web debe tener una pagina de bienvenida
- La página web debe poder registrar usuarios
- La página web debe tener un control de acceso,un Log In
- La pagina debe ofrecer información general sobre la pagina
- una vez se inicia sesion se deben acceder facilmente a tres funcionalidades básicas:
 - Página con contenidos interesantes para el usuario y búsqueda de contenidos interesante
 - Pagina con informacion de la cuenta
 - Página de ayuda
- Se deben poder buscar cursos a través de la categoría o el titulo
- Se deben poder comprar cursos
- Se deben poder vender cursos
- Se debe poder actualizar el e-mail, la foto del usuario y la contraseña
- Se debe poder borrar un usuario
- Se debe poder cerrar sesión
- Se debe poder cambiar de usuario normal a usuario de profesor
- Los usuarios normales pueden consumir los cursos
- Los usuarios de profesor pueden subir cursos y consumir cursos
- Los usuarios de profesor deben introducir los datos necesarios para recibir beneficios económicos de vender los cursos
- La página de ayuda debe tener el formato de un curso común
- La aplicación debe ser multilenguaje

Diagrama de casos de uso



Monetización

La aplicación se rentabiliza a través de la venta de cursos a través de la propia aplicación. Cuando un usuario compra un curso realiza un pago directo a través de paypal. Esos pagos generan un recibo con el que se le pasará una transferencia a los profesores a final de mes.

La aplicación se queda con un porcentaje de la venta. Los profesores siempre estarán al tanto de qué porcentaje ganan de cada video.

La aplicación por ahora no tiene anuncios debido a que tener anuncios por un contenido que ya has pagado podría parecerles una falta de respeto. Me gustaría poner anuncios en un futuro, especialmente en los cursos gratuitos, ya que alojar esos datos en un servidor podría provocar pérdidas si los profesores abusan demasiado de ese formato.

Todos los pagos se realizan a través de Paypal.

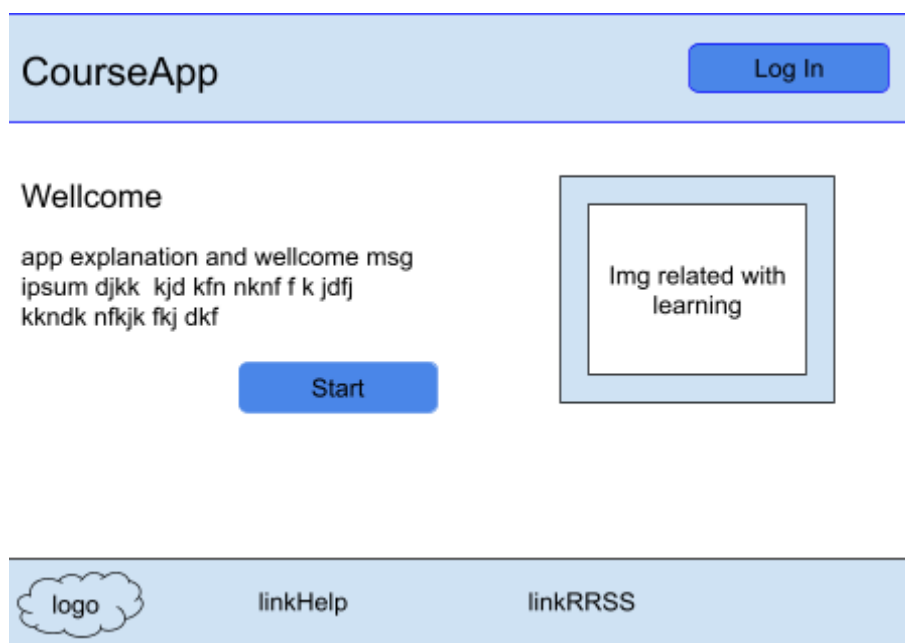
Mockup

Notas






El header y el footer no están definidos en este wireframe. Dependiendo del estado del usuario se mostrarán distintos enlaces en estos apartados.

El wireframe se creó antes de empezar a trabajar en la aplicación, hay gran parte del diseño original que no se ha reflejado en el resultado final.

Page pg_welcome



Page pg_help

Navigator placeholder
How can we help?
<div>Buscar</div>
FAQs
<div>frequent asked question 1?</div>
<div>frequent asked question 1?</div>
<div>frequent asked question 1?</div>
<div>frequent asked question 1?</div>
Do you need more help?
<div>Contact</div>
Footer placeholder

Page pg_recovery_pw_petition

Navigator placeholder
Password Recovery
<div>E_mail</div>
<div>If you have an account, you will receive a mail for your password recovery</div>
<div>Submit</div>
Footer placeholder

Page pg_recovery_pw_petition

Navigator placeholder

Set your new Password

New Password


Repeat Password

Submit

Footer placeholder

Page pg_my_account

Navigator placeholder



update

Name

alumnt info section

update

options

Bought courses

bought courses list section

My courses

account courses list section. Only shown if the account is type professor.

Footer placeholder

Page page_payment_history

Payment history		
course	payment date	credit card number
table info		

Page pg_deleted_account

Navigator placeholder

Deleted account msg

accept

Footer placeholder

Page pg_account_config

Navigator placeholder
<div>Configuration</div> <div>Account</div> <div>update account</div> <div>Pay</div> <div>History</div> <div>Saved Credit Cards</div> <div>Security and pw</div> <div>change password</div> <div>close session on every gadget</div>
Footer placeholder

Page pg_course_list - home

Navigator placeholder
<div><div>search</div><div>filter</div></div> <div>keep watching</div> <div>keep watching section</div> <div>bought courses</div> <div>bought courses section</div> <div>For you</div> <div>For you section</div>
Footer placeholder

Form form_login

CourseApp

Log In

E_mail

Password

Log In

ForgotPw

OR

Continue with google

I want to Sing Up

Form form_singup

CourseApp

Sign up

E_mail

Password

Sign up

OR

Continue with google

I want to Log In

Form form_contact

Contact

Subject

▼

Submit

Cancel

Form form_account_delete

Delete my account

Warning

warning section. user will lose his bought courses

R U sure?

We will send a mail to the introduced mail

Submit

Cancel

Form form_update_account


Update account

Form form_change_pw

Change password

Form form_update_img_account

Update account image



Form form_complaint

what is happening with this video?

• Cat1 ▾

○ subcat1.1

○ subcat1.2

○ subcat1.3

• cat2 ▾

○ subcat2.1

○ subcat2.2

details

Submit

Cancel

Form form_block_user

Are you sure you want to block this user?

Warning

money return warning

Accept

Cancel

Estilo Gráfico (ADD)

En este apartado voy a hacer una recolección de elementos visuales y normas relevantes para mantener una cohesión visual a lo largo de toda la aplicación.

Logo



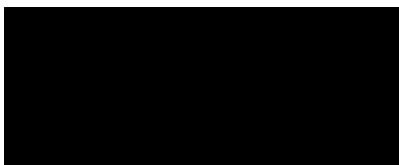
Colores

Los colores utilizados durante toda la aplicación deben ser únicamente los siguientes o la mezcla de ellos en caso de degradados:

--clr-white:#fff;



--clr-black: #000000;



--clr-grey: #474747;



--clr-yellow: #FFCB70;



--clr-pink: #C751C0;



--clr-blue: #4158D0;



Usos de los colores

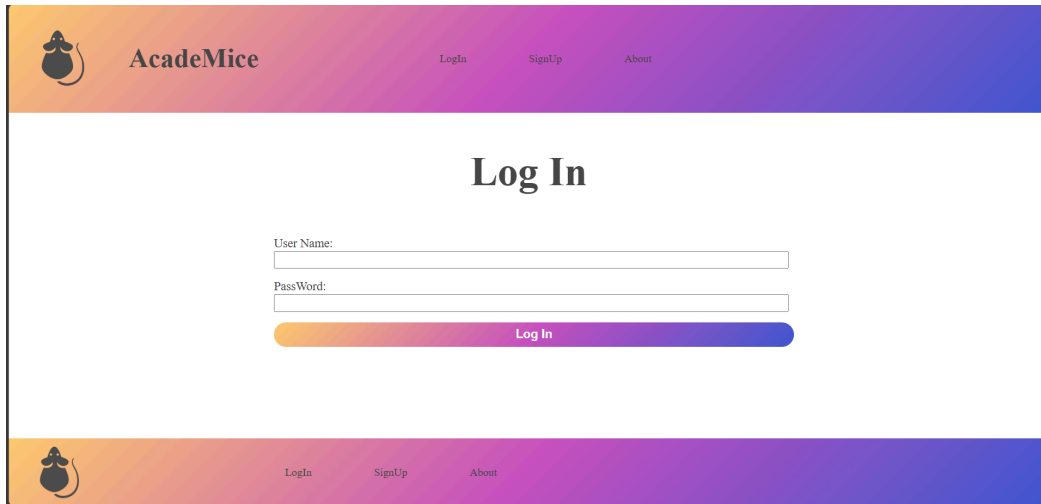
Los colores tendrán los siguientes significados y usos:

- --clr-white: #fff;
 - Fondos
 - Textos sobre fondos oscuros
- --clr-black: #000000;
 - textos que necesiten un contraste especial
- --clr-grey: #474747;
 - textos
 - fondo en la version oscura
- --clr-yellow: #FFCB70;
 - fondos
 - mensajes de advertencia
- --clr-pink: #C751C0;
 - fondos
 - mensajes de error
- --clr-blue: #4158D0;

- fondos
- mensajes de resultado positivo

Navegación

Hay dos barras de navegación, header y footer. Independientemente de los enlaces que contengan, las barras de navegación deben verse así:




The image shows a web page for 'AcadeMice' with a login form. The header and footer are identical, featuring a logo on the left and navigation links ('LogIn', 'SignUp', 'About') on the right. The main content area is titled 'Log In' and contains a form with 'User Name:' and 'PassWord:' labels, input fields, and a 'Log In' button. The button has a gradient from orange to purple.

La imagen del logo contiene un enlace a Index o a Home, dependiendo de si has iniciado sesión

Botones



Formularios



The image shows a web page for 'AcadeMice' with a sign up form. The main content area is titled 'Sign Up' and contains a form with 'User Name:', 'Email:', 'PassWord:', and 'Matching PassWord:' labels, input fields, and a 'Sign Up' button. The button has a gradient from orange to purple. There is also a 'Language:' dropdown menu with options 'English', 'Español', and 'English'.

Upload Image

No se ha seleccionado ningún archivo

Subir Imagen

Siguiente

Vista detalle de un curso

Curso - Resumen



Título:carta curso con imagen

Descripcion: carta curso con imagen subida por el profesor

Precio:0.0

Moneda:EUR

Es publico:true

Lista de videos

Confirmar

Vista maestro de los cursos

Los cursos mostraran siempre una imagen por defecto (la del logo) cuando no se les ha puesto ninguna imagen

preview carta curso



descripcion de preview carta curso lore ipsum preview carta curso

carta curso con imagen



carta curso con imagen subida por el profesor

Calendario de Builds

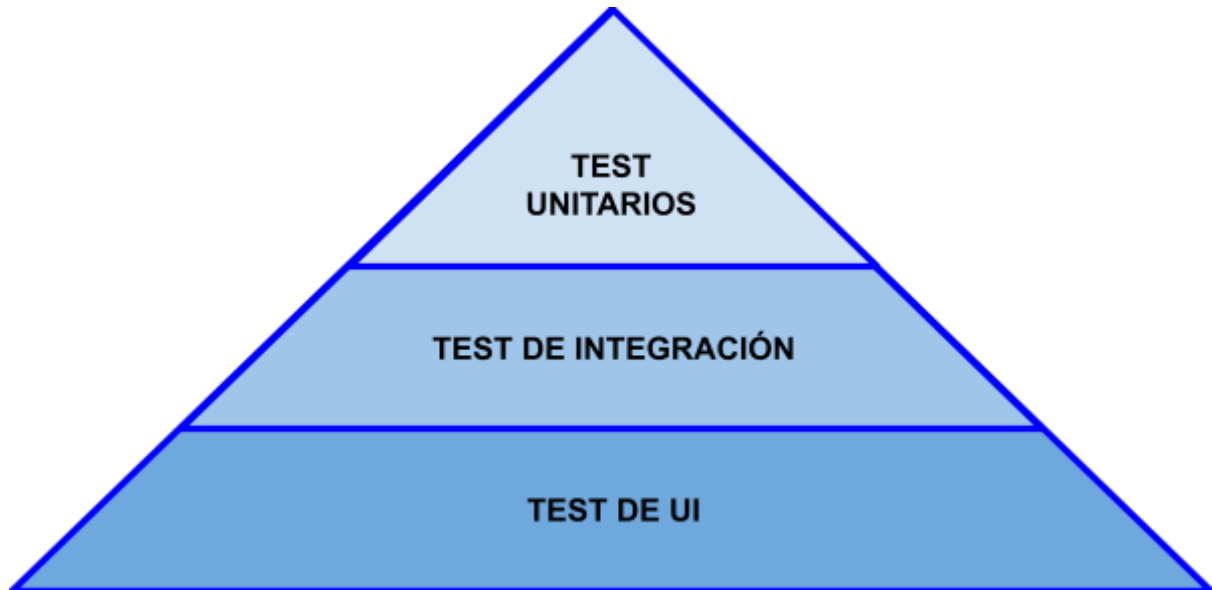
Fecha	Build	Descripción
24/04/2025	Documentacion tecnica basica	<ul style="list-style-type: none"> • Diagrama Entidad/Relación BBDD • Wireframe
27/04/2025	Crear primer endpoint completo funcional	<ul style="list-style-type: none"> • Primer endpoint completo funcional con información de usuario
5/05/2025	CRUD usuarios	<ul style="list-style-type: none"> • Todos los endpoints completos y funcionales para hacerle un crud a Usuario
8/05/2025	CRUD Image	<ul style="list-style-type: none"> • Todos los endpoints completos y funcionales para hacerle un crud a Image
11/05/2025	NavigationBars y Pagina Wellcome	<ul style="list-style-type: none"> • Primera pagina en thymeleaft • Header y footer
12/05/2025	Pagina About	<ul style="list-style-type: none"> • Pagina de informacion
15/05/2025	Seguridad	<ul style="list-style-type: none"> • Sistema de control de acceso: <ul style="list-style-type: none"> ◦ login ◦ signup ◦ logout ◦ permisos
4/06/2025	Relacion videos curso	<ul style="list-style-type: none"> • Crud de videos • Crud de cursos • Todas las paginas necesarias para: <ul style="list-style-type: none"> ◦ crear curso ◦ subir video a curso
6/06/2025	Mostrar videos de profesor	<ul style="list-style-type: none"> • Sistema simplificado para reproducir videos
7/06/2025	Detall videos de profesor	<ul style="list-style-type: none"> • Listado de cursos de profesor y videos de curso
8/06/2025	Reproducir curso en streaming y pagina de help	<ul style="list-style-type: none"> • Sistema completo para reproducir videos streaming • listado de cursos help en la pagina de help
12/06/2025	Implementacion de Google Analitics	<ul style="list-style-type: none"> • Implementacion de Google Analitics • Sistema de cobros y pagos
13/06/2025	Multilenguaje	<ul style="list-style-type: none"> • traducción de todos los textos a castellano de forma automática dependiendo del idioma del usuario

Quality Assurance

Para este proyecto vamos a utilizar los conceptos de la pirámide de testing invertida. Ambos conceptos quedan definidos a continuación

Pirámide de testing invertida:

La pirámide de testing es una representación visual de la organización lógica de los distintos tipos de pruebas. La pirámide nos indica que la mayoría de los tests son test de UI, luego una capa de test de servicio y finalmente unos pocos test unitarios



- **Test Unitarios** -> no dependen de recursos externos, es decir, no se conectan a la base de datos, ni a ficheros ni usan recursos api externas, prueban una unidad (es decir, una funcionalidad atómica del código).
 - En mi caso los Test Unitarios se harán con herramientas de test unitarios como JUnit y serán utilizados para hacer Test Driven Development.
- **Test de Servicio, Funcionales o de Integración** -> son test que prueban una funcionalidad. Necesita diferentes módulos de software que están integrados lógicamente y se prueban como un grupo. Estos test pueden acceder a la base de datos y a sistemas que estén bajo nuestro control, pero en el caso que haya sistemas externos que no estén bajo nuestro control utilizaríamos dobles de pruebas (por ejemplo, llamar a la API de Google o Facebook).
 - En mi caso estos test se realizarán de forma manual, de forma incremental, es decir, cada vez que un módulo haya pasado los test unitarios se realizarán test de integración manuales
- **Test de UI** -> verificamos la aplicación desde la interfaz de usuario, ya sea llamando a un endpoint de una API o simulando clicar en botones.
 - En mi caso estos test se realizarán de forma manual haciendo uso de la interfaz gráfica

Cuando se han hecho los test

En un principio me hubiera gustado hacer Test Driven Development, pero al no tener conocimientos de como hacer test para APIS tenía muchos problemas para saber cuando había hecho un test correctamente. Es decir, en Test Driven Development, el test siempre comienza dando fallo, entonces tienes que estar muy seguro de que el test está bien escrito para empezar a desarrollar.

Por este motivo el sistema que he utilizado es la realización de tests de UI incrementales. Mientras he desarrollado una funcionalidad, la he testado continuamente. Cuando terminaba la funcionalidad, antes de hacer commit en git he hecho testeo de control de todas las funcionalidades. Una vez todo funcionaba, podía pasar a otra cosa.

Accesibilidad

La aplicación está completamente en inglés. El usuario tiene la opción de cambiarla a Español a través de la información de su perfil.

Presupuesto

Al comenzar el proyecto hice una estimación del tiempo que podía dedicarle al proyecto. Hice un análisis de mis horarios y mis rutinas. Contabilice 285 horas.

El precio de cada hora trabajada es de 9,375 €.

Este dinero proviene de dividir 1500€/mes (brutos) entre las 160 horas trabajables de cada mes.

Teniendo en cuenta lo anteriormente mencionado, al comienzo del proyecto **calculé que iba a costar 2.671,875 €**

El tiempo que realmente le he dedicado actualmente al proyecto son 136:25:43 horas. Pero calculo que le voy a dedicar 140 horas.

Teniendo en cuenta todo lo anteriormente mencionado, **el proyecto ha costado 1.312,50 €**

Si publico la pagina web, calculo que voy a dedicarle mínimo de 40 horas semanales en el mantenimiento y resolución de incidencias. Y seguramente en los comienzos haya que gastar un maximo de 100 € al mes en servicios como dominio, hosting y otros.

Por lo tanto, **por cada més que el proyecto esté activo, cuesta alrededor de 475€**

Tecnologías

- Base de datos: MySql
- Diagramas entidad Relación: [Draw.io](https://draw.io)
- Documentación: Google Docs
- Repositorio de documentación: Google Drive
- Analítica: Google Analytics
- Backend: Java Spring
- FrontEnd: Java Spring Thymeleaf
- Scrum: Trello
- Recuento de horas: Google Sheets
- Repositorio: Git, GitHub y GitFlow
- Pasarela de pagos: Paypal
- Tests unitarios: JUnit