AutoValor

Aplicació per a la compra i venda de vehicles amb intel·ligència artificial

ALUMNE

Víctor Martos Segura

CICLE FORMATIU

CFGS Desenvolupament d'Aplicacions Multiplataforma

CENTRE EDUCATIU

Institut Puig Castellar

TUTOR/A

Luis Elía

ANY

2026

Índex de continguts

Resum i Abstract d'AutoValor

Resum

El present projecte té com a objectiu el disseny i el desenvolupament d'AutoValor, una aplicació multiplataforma —disponible en versió web i mòbil— orientada a facilitar la compra i venda de vehicles d'ocasió entre particulars i professionals del sector. El projecte neix de la necessitat de simplificar i professionalitzar la publicació d'anuncis de vehicles, un àmbit on la qualitat de la informació presentada té un impacte directe en l'èxit de la transacció i en la confiança del comprador.

AutoValor aborda aquesta problemàtica integrant la intel·ligència artificial en el procés de creació d'anuncis. A partir de fotografies del vehicle en sis angles —frontal, posterior, lateral, interior, quadre de comandament i extra— i d'un conjunt de dades bàsiques proporcionades per l'usuari, l'aplicació genera automàticament una proposta de dades tècniques i text descriptiu, que l'usuari sempre revisa i edita abans de publicar. Aquesta validació humana és un principi fonamental del disseny del sistema.

A nivell tècnic, el projecte utilitza **Ionic Vue amb Vue 3, TypeScript, Vite i Capacitor** al frontend, i **Java 21, Spring Boot, Spring Security, JWT, JPA i PostgreSQL** al backend, desplegat sobre l'entorn cloud d'Aiven. La integració d'IA es realitza a través d'un endpoint propi (*/api/ai/vehicle-suggestions*) que processa les imatges i un prompt estructurat per generar la proposta d'anunci.

Abstract

This project aims to design and develop AutoValor, a cross-platform application —available in both web and mobile versions— oriented towards facilitating the buying and selling of second-hand vehicles

between private users and professionals. The project addresses the need to simplify and professionalise vehicle listing publication, a domain where the quality of information directly impacts transaction success and buyer trust.

AutoValor integrates artificial intelligence into the listing creation process. Based on six vehicle photographs —front, rear, side, interior, dashboard and extra— and basic user-provided data, the application automatically generates a proposal of technical details and descriptive text, which the user always reviews and edits before publishing.

Technically, the project uses **Ionic Vue with Vue 3, TypeScript, Vite and Capacitor** on the frontend, and **Java 21, Spring Boot, Spring Security, JWT, JPA and PostgreSQL** on the backend, deployed on Aiven cloud infrastructure. AI integration is handled through a custom endpoint (*/api/ai/vehicle-suggestions*) that processes images and a structured prompt to generate the listing proposal.

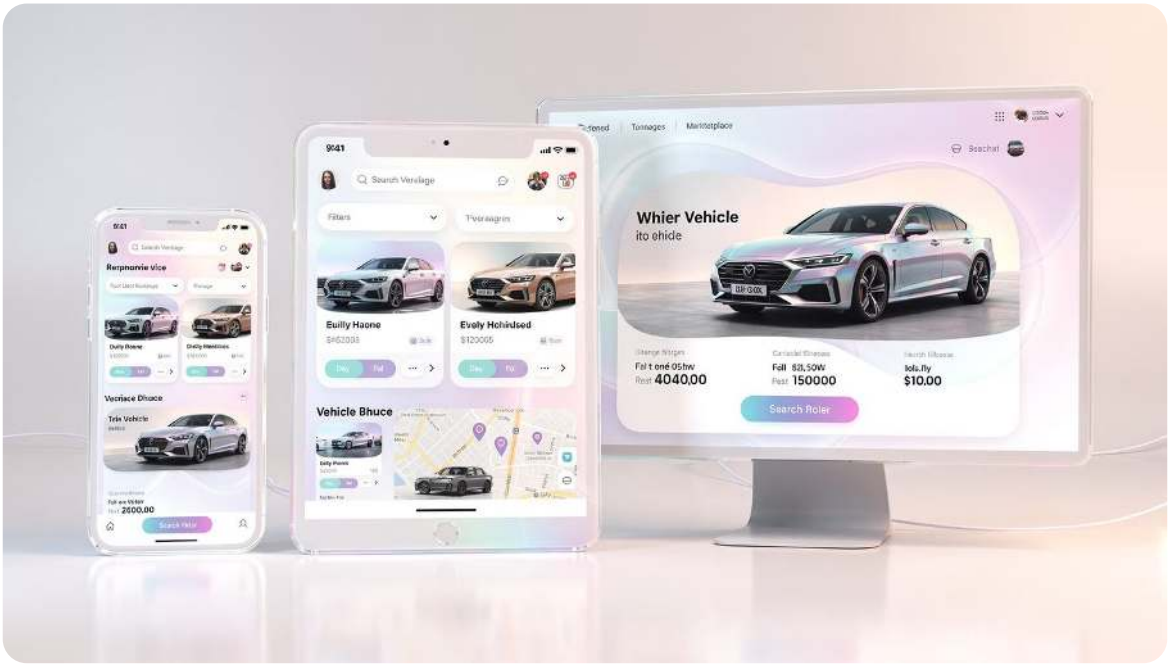
CAPÍTOL 1

1. Presentació del projecte

1.1 Introducció

Descripció del projecte

AutoValor és una aplicació multiplataforma —disponible en versió web i mòbil— orientada a facilitar el procés de compra i venda de vehicles d'ocasió entre particulars i professionals del sector. El projecte neix de la necessitat de simplificar i professionalitzar la publicació d'anuncis de vehicles, un àmbit on la qualitat de la informació presentada té un impacte directe en l'èxit de la transacció.



Imatge conceptual: aplicació multiplataforma disponible en web i mòbil.

Sobre aquest document

El present document constitueix la memòria tècnica del Projecte Final del Cicle Formatiu de Grau Superior en Desenvolupament d'Aplicacions Multiplataforma (DAM). Recull de manera estructurada

totes les fases del cicle de vida del programari: des de la concepció inicial i l'anàlisi de requisits fins al disseny, la implementació i la validació del producte final.

Aquesta memòria té com a objectiu documentar de forma rigorosa les decisions preses al llarg del projecte, les tecnologies emprades i els resultats obtinguts, tot seguint els estàndards formals propis d'un projecte d'enginyeria de programari.

El paper de la intel·ligència artificial

Un dels elements centrals d'AutoValor és la integració de la intel·ligència artificial en el procés de creació d'anuncis. A partir de fotografies del vehicle i d'un conjunt de dades bàsiques introduïdes per l'usuari, l'aplicació és capaç de generar descripcions clares, coherents i atractives de manera automatitzada, reduint l'esforç necessari per part del venedor i millorant la qualitat global de la informació publicada.

Imatge conceptual: mòdul AutoValor AI amb generació de suggeriments de dades.

1.2 Context i justificació

El mercat de vehicles d'ocasió

El mercat de vehicles de segona mà representa un dels sectors de compravenda entre particulars amb major volum de transaccions a nivell nacional. En els darrers anys, la digitalització d'aquest mercat ha propiciat l'aparició de diverses plataformes en línia que actuen com a intermediaris entre venedors i compradors. Portals com Wallapop, Milanuncios, Coches.net o InfoCoche concentren una part significativa d'aquestes operacions, oferint un espai comú on publicar i consultar anuncis de vehicles de segona mà.

Funcionament de les plataformes actuals

Malgrat la consolidació d'aquestes plataformes, el procés de publicació d'un anunci continua recaient íntegrament en el venedor, que ha d'introduir manualment tota la informació rellevant del vehicle: característiques tècniques, estat de conservació, historial de manteniment, preu i fotografies. Aquesta dependència de la iniciativa i el criteri individual del venedor genera una heterogeneïtat notable en la qualitat dels anuncis publicats. És habitual trobar publicacions amb descripcions incompletes, fotografies de baixa qualitat, informació ambigua o directament errònia, la qual cosa dificulta la presa de decisions per part del comprador.

Problemàtica actual del sector

Aquesta manca d'uniformitat en la presentació dels vehicles té conseqüències directes en l'eficiència del procés de compravenda. Els compradors es veuen obligats a invertir temps considerable en la comparació d'anuncis, en la sol·licitud d'informació addicional i en la verificació de dades que haurien d'estar disponibles des del primer moment. Paral·lelament, la desconfiança generada per anuncis poc rigorosos o incomplets és un dels principals factors que allarga els processos de negociació i redueix la taxa de tancament de transaccions.

Oportunitat de millora

En aquest context, s'identifica una oportunitat clara de millora en la fase de creació i presentació dels anuncis. L'automatització parcial d'aquest procés, mitjançant l'ús de tecnologies d'intel·ligència artificial, podria contribuir a elevar el nivell de qualitat i coherència de la informació publicada, beneficiant tant els venedors —que reduirien l'esforç necessari— com els compradors —que disposarien d'informació més completa i fiable per prendre les seves decisions.

Imatge conceptual: el sector de compravenda de vehicles d'ocasió.

1.3 Justificació

Les plataformes actuals de compravenda de vehicles d'ocasió ofereixen un espai de publicació, però no aporten cap mecanisme que garanteixi la qualitat ni la coherència dels anuncis generats. La responsabilitat de redactar una descripció adequada, seleccionar les fotografies pertinents i estructurar la informació de manera comprensible recau exclusivament en el venedor, sense cap guia ni suport automatitzat. Aquesta absència d'assistència en el procés de creació és una de les causes principals de la baixa qualitat mitjana dels anuncis publicats i

constitueix una limitació estructural que cap de les solucions existents ha abordat de manera satisfactòria.

En aquest escenari, el desenvolupament d'AutoValor es justifica per la necessitat de cobrir un buit funcional concret: proporcionar a l'usuari una eina que l'acompanyi en el procés de publicació i que, a partir d'una entrada mínima de dades, sigui capaç de generar un anunci complet, ben estructurat i de qualitat. La incorporació de la intel·ligència artificial en aquest procés no respon a una tendència tecnològica, sinó a una necessitat real: automatitzar una tasca repetitiva i subjectiva —la redacció de descripcions— per obtenir resultats més uniformes i fiables que els que produeix la intervenció manual no assistida.

Des del punt de vista tècnic, el projecte representa una oportunitat per integrar i aplicar de manera conjunta diverses competències adquirides al llarg del cicle formatiu: el disseny i desenvolupament d'aplicacions multiplataforma, la connexió amb serveis externs mitjançant APIs, la gestió de bases de dades i la implementació d'interfícies d'usuari funcionals i accessibles. La integració d'un model de llenguatge per a la generació de text afegeix una capa de complexitat tècnica que enriqueix el projecte i el situa en un context tecnològic actual i rellevant.

1.4 Objectius

Objectiu general

Dissenyar i desenvolupar una aplicació multiplataforma —web i mòbil — que millori el procés de compra i venda de vehicles d'ocasió, facilitant la creació d'anuncis de qualitat mitjançant la integració de la intel·ligència artificial i oferint als usuaris un entorn de cerca, comunicació i transacció fiable i accessible.

Objectius específics

1. Implementar un sistema de registre i autenticació d'usuaris que garanteixi la seguretat i la gestió correcta de les identitats dins de la plataforma.
2. Desenvolupar un mòdul de creació d'anuncis de vehicles que permeti introduir les dades i les fotografies del vehicle de manera guiada i estructurada.
3. Integrar un sistema d'assistència basat en intel·ligència artificial capaç de generar automàticament el contingut descriptiu dels anuncis a partir de les dades i imatges aportades per l'usuari.
4. Implementar un motor de cerca amb filtres avançats que permeti als compradors localitzar vehicles segons criteris específics com la marca, el model, el preu o el quilometratge.
5. Incorporar un sistema de missatgeria interna que faciliti la comunicació directa entre venedors i compradors dins de la mateixa plataforma.
6. Desenvolupar un sistema de valoració d'usuaris que contribueixi a generar confiança i transparència en les transaccions. *(Futura millora)*
7. Desplegar una versió funcional de l'aplicació en un entorn de núvol, garantint la seva disponibilitat i accessibilitat des de qualsevol dispositiu.

1.5 Abast del MVP

El producte lliurat és un MVP (Minimum Viable Product) funcional que cobreix les funcionalitats essencials del sistema. L'abast s'ha definit per garantir una experiència d'usuari coherent i un conjunt de funcionalitats real i demostrable:

ÀMBIT	INCLÒS AL MVP	FUTURA MILLORA
Autenticació	Registre, login, JWT, rols USER/ADMIN	Login social (Google/ Apple)
Anuncis	CRUD complet, imatges via Cloudinary (pujada + eliminació), cerca, filtres, paginació	Vídeos dels vehicles

ÀMBIT	INCLÒS AL MVP	FUTURA MILLORA
IA	Sugeriment de dades a partir de fotos, revisió editable	Detecció automàtica de matrícula
Comunicació	Contacte amb venedor, leads associats a anunci i xat en temps real amb WebSocket	
Valoracions	—	Sistema de puntuació i ressenyes
Perfil	Gestió de perfil, wishlist, MyListings	Historial de transaccions
Admin	Dashboard, gestió d'usuaris i anuncis	Mètriques avançades

CAPÍTOL 2

2. Estratègia i planificació

Imatge conceptual: gestió del projecte amb metodologia iterativa.

2.1 Estratègia de desenvolupament i viabilitat

Enfocament general del desenvolupament

El projecte AutoValor s'ha concebut com un desenvolupament individual de caràcter iteratiu i incremental. Aquesta estratègia implica que l'aplicació no s'ha construït de manera lineal —és a dir, completant cada mòdul de forma definitiva abans de passar al següent —, sinó que s'ha anat evolucionant progressivament, afegint funcionalitats de manera gradual i refinant les existents a mesura que avançava el projecte. Aquest enfocament permet detectar problemes de disseny o d'implementació en fases primerenques, reduint el risc d'errors estructurals difícils de corregir en etapes avançades del desenvolupament.

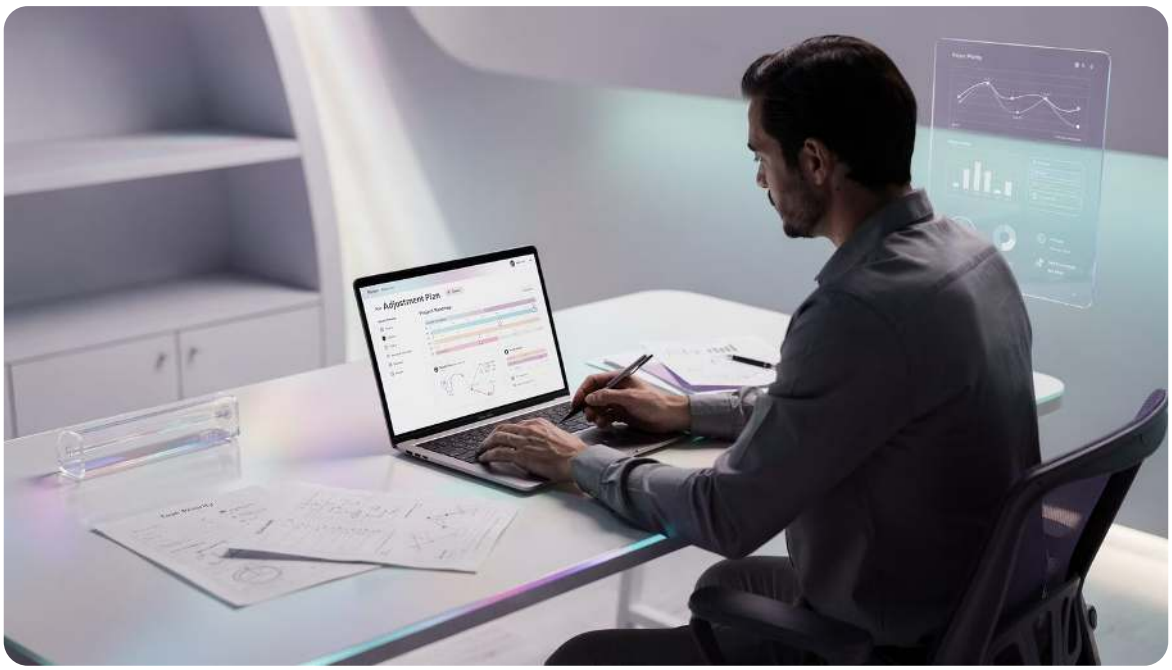
Viabilitat tècnica

Des del punt de vista tècnic, el projecte és viable gràcies a la disponibilitat de tecnologies madures, ben documentades i àmpliament utilitzades en l'àmbit del desenvolupament d'aplicacions multiplataforma. Les eines i els frameworks seleccionats —tant per al frontend com per al backend— formen part del currículum del CFGS DAM o han estat treballats de manera autònoma durant el cicle, la qual cosa garanteix un nivell de familiaritat suficient per abordar el desenvolupament amb garanties. A més, la integració de la intel·ligència artificial es realitza mitjançant APIs externes ja existents, eliminant la necessitat de desenvolupar models propis i reduint significativament la complexitat tècnica associada a aquesta funcionalitat.

Viabilitat econòmica

El projecte presenta una viabilitat econòmica elevada, atès que es basa exclusivament en tecnologies de codi obert i serveis cloud amb plans gratuïts o de baix cost. Totes les eines de desenvolupament —IntelliJ IDEA, Visual Studio Code, Postman, Git— estan disponibles de manera gratuïta per a estudiants i projectes no comercials. Quant als serveis cloud, s'utilitzen els plans gratuïts oferts per Aiven (PostgreSQL), GitHub (repositoris i CI) i Cloudinary (emmagatzematge d'imatges). En conjunt, el cost econòmic directe del projecte és nul o mínim, la qual cosa el fa perfectament assolible en l'àmbit d'un projecte final de cicle.

Viabilitat temporal



La durada prevista per al desenvolupament del projecte és d'aproximadament sis mesos, distribuïts en fases. El pla de treball s'ha estructurat de manera que les funcionalitats de major valor afegit —el mòdul d'IA, la publicació d'anuncis i la cerca— es desenvolupen en primer lloc, deixant per a fases posteriors les funcionalitats complementàries com el sistema de valoració o les notificacions push. Aquesta priorització garanteix que, en cas que el temps disponible sigui inferior a l'estimat, el producte lliurat continuï sent funcional i demostrable.

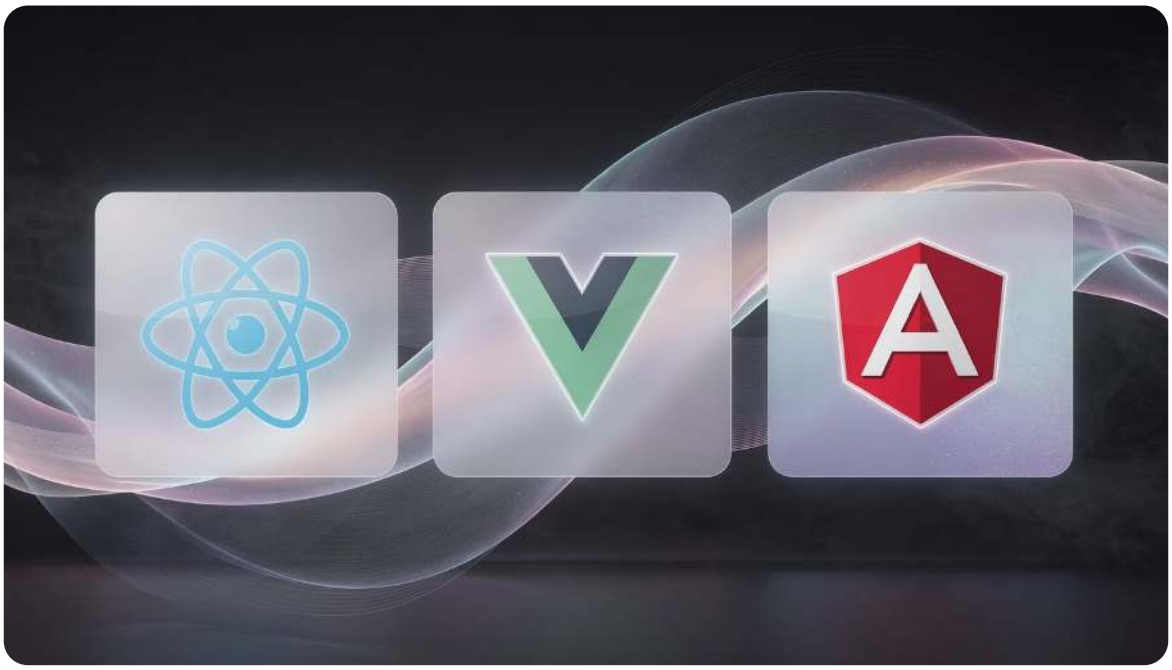
2.2 Evolució tecnològica del projecte

> *La proposta inicial d'AutoValor contemplava un stack diferent al finalment implementat. Durant el procés de validació tècnica, es van analitzar diverses alternatives i es va prendre la decisió d'adoptar un stack més alineat amb les competències DAM i els requisits reals del projecte. Aquest canvi no és un error, sinó una decisió tècnica raonada i ben documentada.*

Stack inicial proposat

En la fase d'anàlisi inicial, la proposta tecnològica contemplava:

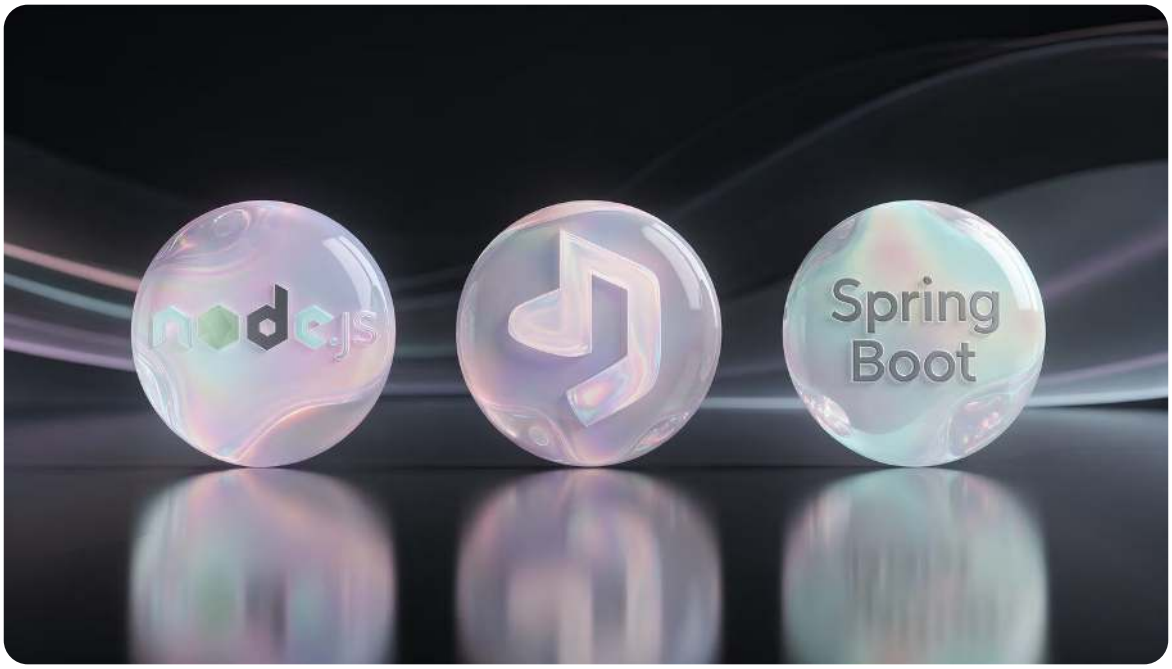
- **Frontend:** React amb Tailwind CSS
- **Backend:** Node.js amb Express
- **IA:** Python + FastAPI amb scikit-learn



Imatge conceptual: frameworks frontend avaluats durant l'anàlisi.

Motius del canvi tecnològic

Després d'una fase d'anàlisi i prototipatge, es van identificar diverses limitacions en l'stack inicial que van motivar el canvi:



Imatge conceptual: alternatives backend avaluades.

- **React vs Ionic Vue:** Ionic Vue ofereix una experiència mobile-first nativa, una barra de navegació inferior integrada i la capacitat d'empaquetar l'aplicació com a app Android/iOS mitjançant Capacitor, sense necessitat de frameworks addicionals. React, sense una capa mòbil específica, requeriria React Native per obtenir resultats comparables, afegint complexitat.
- **Node.js vs Spring Boot:** Spring Boot ofereix una arquitectura per capes molt més estructura i segura (Spring Security, JWT natiu, JPA, validació declarativa), millor alineada amb el currículum DAM i amb els estàndards professionals del backend Java. Node.js requeriria configurar manualment la majoria d'aquestes capes.
- **FastAPI + Python vs endpoint integrat:** La IA es va decidir integrar com un endpoint directament al backend Spring Boot, consumint una API d'IA externa (OpenAI). Això elimina la necessitat de mantenir un microservei separat en Python i simplifica enormement el desplegament i l'arquitectura del sistema.

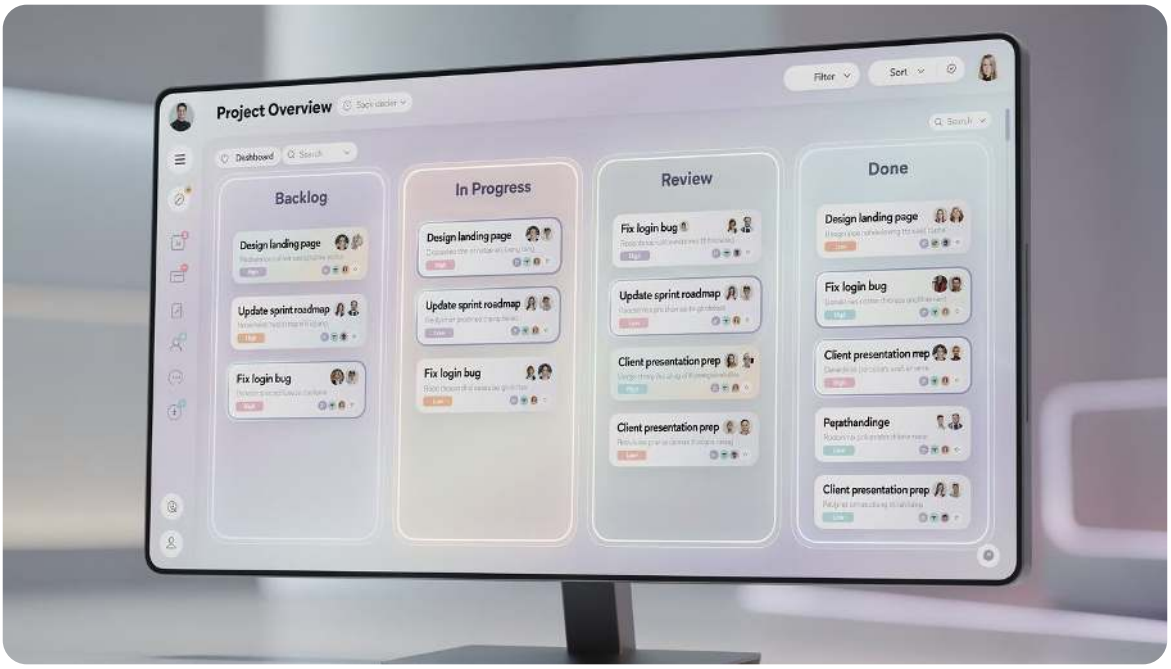
Stack definitiu adoptat

CAPA	PROPOSTA INICIAL	STACK FINAL	MOTIU DEL CANVI
Frontend	React + Tailwind	Ionic Vue + Vue 3 + TypeScript	Mobile-first, Capacitor, experiència nativa
Backend	Node.js + Express	Java 21 + Spring Boot	Seguretat, arquitectura per capes, alineació DAM
Base de dades	PostgreSQL	PostgreSQL (Aiven cloud)	Mantenida, cloud deployment afegit
IA	Python + FastAPI + scikit-learn	Endpoint integrat + OpenAI API	Elimina microservei, simplifica desplegament
Imatges	Cloudinary (implementat)	Cloudinary	CDN professional; pujada, llistat i eliminació d'imatges implementades

2.3 Metodologia de treball

Elecció de la metodologia

Per al desenvolupament d'AutoValor s'ha adoptat una metodologia àgil basada en els principis de Kanban adaptat a projecte individual. A diferència de Scrum —que requereix equips i cerimònies fixes—, Kanban permet una gestió contínua del flux de treball, visualitzant l'estat de cadascuna de les tasques en tot moment i permetent reordenar les prioritats de manera flexible en funció de les necessitats emergents del projecte.



Seguiment del projecte amb tauler de tasques: Backlog, In Progress, Review i Done.

Organització del treball per fases

El treball s'ha organitzat en tres grans fases seqüencials, cadascuna amb objectius clars i lliurables identificats:

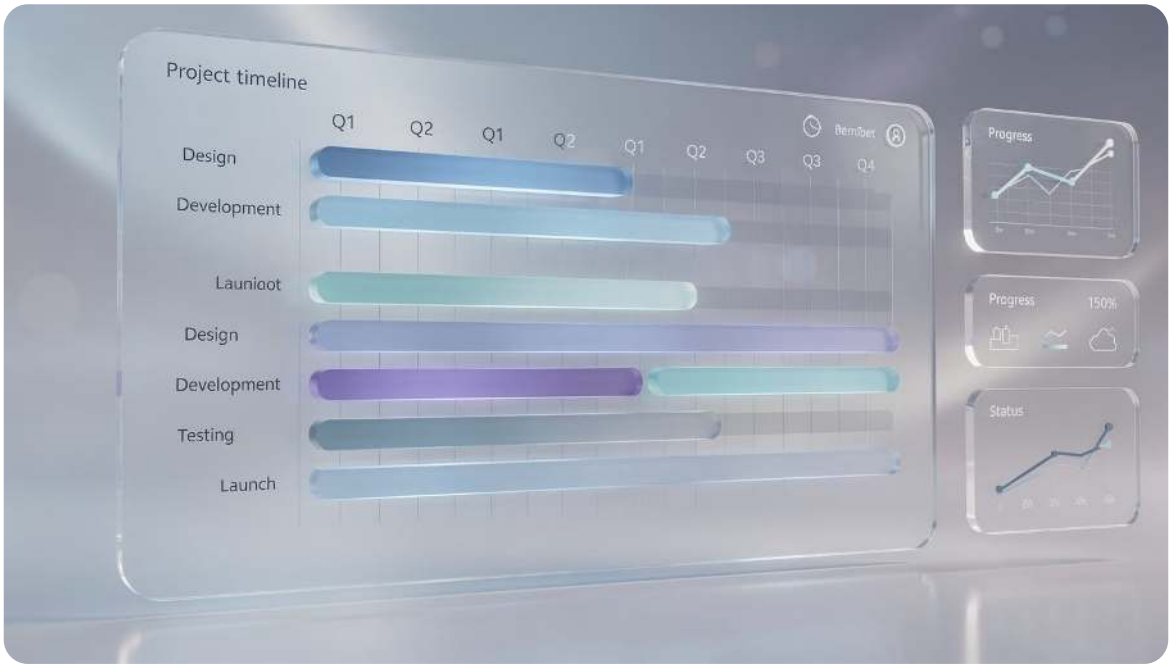
- **Fase 1 – Anàlisi i disseny (setmanes 1–4):** Definició dels requisits, arquitectura del sistema, disseny de la base de dades, wireframes i prototips de les pantalles principals.
- **Fase 2 – Desenvolupament (setmanes 5–18):** Implementació incremental del backend i el frontend. S'ha seguit l'ordre: autenticació → CRUD d'anuncis → imatges → cerca → IA → comunicació → administració.
- **Fase 3 – Proves, ajustos i documentació (setmanes 19–24):** Proves d'integració, correccions, optimitzacions de rendiment i redacció de la memòria tècnica i el README.

Gestió de tasques i control de versions

Les tasques s'han gestionat mitjançant un tauler Kanban (GitHub Projects), on cada tarjeta representava una funcionalitat o una correcció. El control de versions s'ha realitzat amb Git i GitHub, amb

dues branques principals: *master* (backend) i *main* (frontend). El control de versions s'ha realitzat amb Git i GitHub, mantenint dos repositoris separats i registrant l'evolució del projecte mitjançant commits freqüents.

2.4 Planificació temporal



Imatge conceptual: distribució temporal de les fases del projecte.

Estructura de fases del projecte

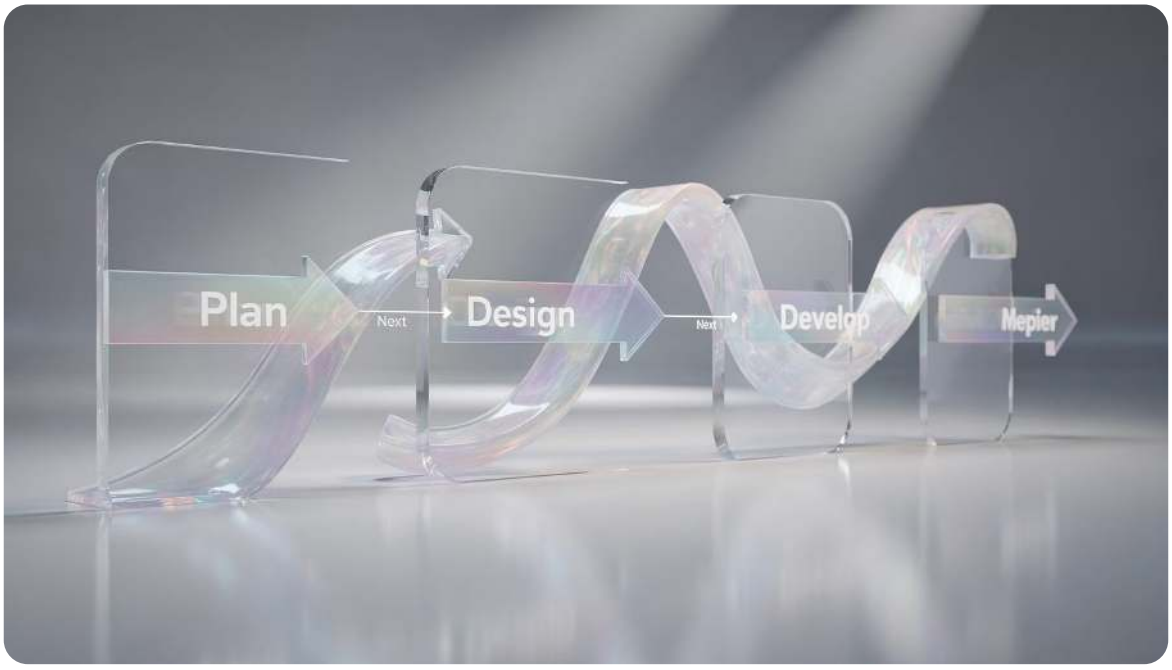
FASE	SETMANES	TASQUES PRINCIPALS	LLIURABLE
Anàlisi	1–3	Requisits, casos d'ús, arquitectura	Document d'anàlisi
Disseny	4–6	Model de dades, wireframes, API design	Prototips i esquemes
Backend core	7–11	Auth, CRUD, imatges, seguretat	API funcional documentada
Frontend core	9–15	Home, Sell, Search, Wishlist, Profile	App web funcional

FASE	SETMANES	TASQUES PRINCIPALS	LLIURABLE
IA i comunicació	14–18	Vehicle AI, leads, missatgeria	Flux IA complet
Admin i ajustos	17–20	Dashboard admin, correccions	Panel d'administració
Proves i docs	20–24	Testing, memòria, README	Lliurables finals

Descripció de les fases

La planificació s'ha dissenyat tenint en compte les dependències entre les diferents parts del sistema. El backend s'ha iniciat lleugerament abans que el frontend per garantir que l'API estigui disponible quan el frontend la necessiti. El mòdul d'IA s'ha deixat per a una fase més avançada, un cop el flux de publicació d'anuncis era estable, per poder integrar-lo sense afectar les funcionalitats existents.

Planificació temporal i distribució d'esforços



Imatge conceptual: flux de treball del projecte.

Gestió del temps i seguiment

Al llarg del projecte s'han realitzat revisions setmanals de l'estat de les tasques per assegurar que el progrés estava alineat amb la planificació. Les desviacions detectades s'han abordat mitjançant la reordenació de prioritats i, en alguns casos, el trasllat de funcionalitats menys crítiques a la llista de millores futures, per garantir que el lliurable principal fos funcional i de qualitat.

2.5 Riscos i mesures correctives

RISC IDENTIFICAT	PROBABILITAT	IMPACTE	MESURA CORRECTIVA
Canvi de tecnologia a mig projecte	Alta	Alt	Validar l'stack en fase d'anàlisi; documentar l'evolució tecnològica
Integració complexa del mòdul d'IA	Mitjana	Alt	Dissenyar el flux com a editable i no bloquejant; la IA és un assistent, no un automatisme
Limitacions de l'API d'OpenAI (cost, latència)	Mitjana	Mitjà	Cache de respostes; endpoint dissenyat per ser fàcilment substituïble
Problemes de CORS entre frontend i backend	Alta	Baix	Configuració explícita de CORS al backend; variable VITE_API_BASE_URL al frontend
Gestió d'imatges i espai d'emmagatzematge	Baixa	Mitjà	Integració amb Cloudinary; validació de mida i format al backend
Falta de temps per a funcionalitats avançades	Mitjana	Baix	MVP ben definit; funcionalitats secundàries documentades com a millores futures

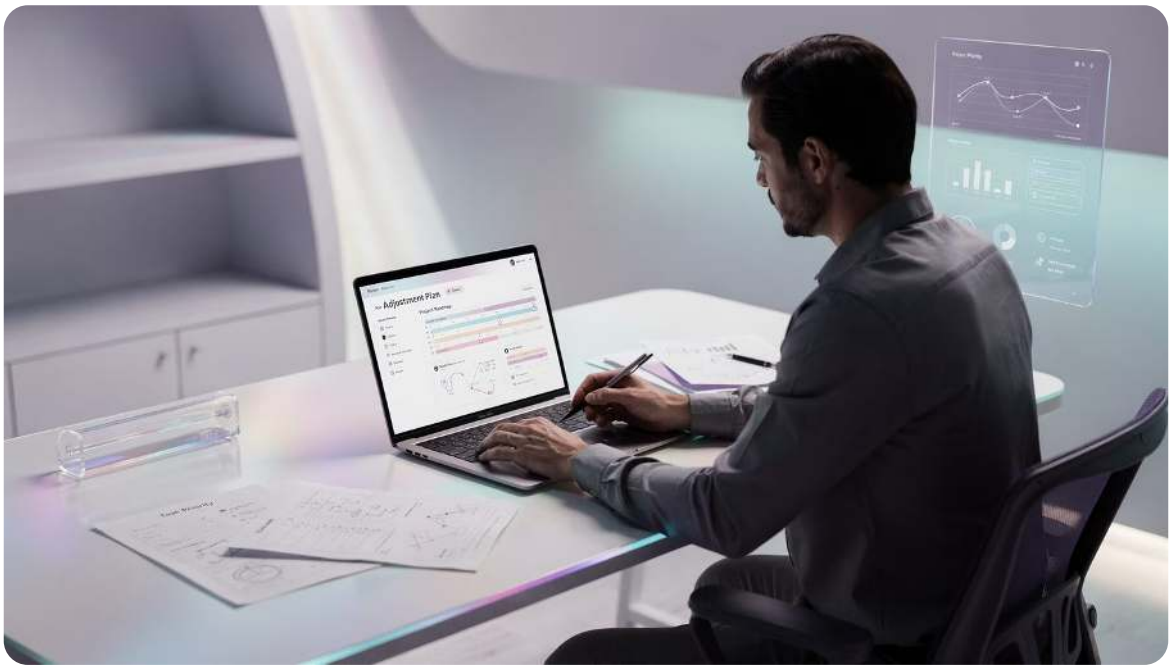
CAPÍTOL 3

3. Anàlisi

3.1 Casos d'ús

Introducció als casos d'ús

Els casos d'ús descriuen les interaccions entre els actors del sistema i les funcionalitats que aquest ofereix. AutoValor contempla tres actors principals: el **Visitant** (usuari no autenticat), l'**Usuari registrat** (amb rol USER) i l'**Administrador** (amb rol ADMIN).



Imatge conceptual: actors del sistema.

Actors del sistema

ACTOR	DESCRIPCIÓ	ACCÉS
Visitant	Usuari no autenticat que navega per la plataforma	Consulta pública d'anuncis, cerca i detall de vehicle

ACTOR	DESCRIPCIÓ	ACCÉS
Usuari (USER)	Usuari registrat i autenticat	Totes les funcionalitats: publicar, wishlíst, missatgeria, perfil
Administrador (ADMIN)	Usuari amb privilegis de gestió de la plataforma	Dashboard admin, gestió d'usuaris, anuncis i estadístiques

Casos d'ús principals

ID	CAS D'ÚS	ACTOR	DESCRIPCIÓ
CU-01	Registrar-se	Visitant	L'usuari crea un compte amb email i contrasenya
CU-02	Iniciar sessió	Visitant	L'usuari s'autentica i obté un token JWT
CU-03	Cercar vehicles	Visitant / Usuari	Cerca per marca, model, preu, quilometratge i filtres avançats
CU-04	Veure detall de vehicle	Visitant / Usuari	Consulta informació completa i imatges d'un anunci
CU-05	Publicar anunci amb IA	Usuari	Pujar fotos → IA genera proposta → usuari edita → publica
CU-06	Gestionar anuncis propis	Usuari	Editar, eliminar i canviar estat (disponible/venut/reservat)
CU-07	Afegir a la wishlist	Usuari	Guardar vehicles d'interès per consultar-los posteriorment
CU-08	Contactar amb el venedor	Usuari	Enviar un missatge/lead associat a un anunci concret
CU-09	Gestionar perfil	Usuari	Editar dades personals, veure MyListings i historial
CU-10	Administrar la plataforma	Administrador	Accedir al dashboard, gestionar usuaris i anuncis

3.2 Requisits funcionals

Mòdul d'autenticació i usuaris

ID	REQUISIT	PRIORITAT
RF-01	El sistema ha de permetre el registre de nous usuaris amb email i contrasenya.	Alta
RF-02	El sistema ha d'autenticar els usuaris i retornar un token JWT vàlid.	Alta
RF-03	El sistema ha de gestionar dos rols: USER i ADMIN, amb accés diferenciat.	Alta
RF-04	Els visitants no autenticats han de poder consultar anuncis públics sense registrar-se.	Alta
RF-05	Els usuaris han de poder gestionar el seu perfil (nom, email, avatar).	Mitjana

Mòdul d'anuncis de vehicles

ID	REQUISIT	PRIORITAT
RF-06	El sistema ha de permetre crear anuncis amb dades tècniques: marca, model, any, km, preu, combustible, transmissió, carrosseria i color.	Alta
RF-07	Cada anunci ha de suportar la pujada de múltiples imatges associades.	Alta
RF-08	El sistema ha d'oferir cerca i filtratge per marca, model, preu, km, any i disponibilitat.	Alta
RF-09	El sistema ha de suportar paginació en el llistat d'anuncis.	Mitjana
RF-10	Els usuaris han de poder editar i eliminar els seus propis anuncis.	Alta
RF-11	Els anuncis han de tenir un estat (disponible, reservat, venut) que l'usuari pugui canviar.	Mitjana

Mòdul d'intel·ligència artificial

ID	REQUISIT	PRIORITAT
RF-12	El sistema ha de permetre pujar fins a 6 fotografies del vehicle per al flux de IA.	Alta
RF-13	El backend ha d'enviar les imatges i un prompt estructurat a l'API d'IA i retornar una proposta de dades.	Alta
RF-14	La resposta de la IA ha d'incloure: title, description, brand, model, year, km, fuelType, transmission, bodyType, color, confidence i warnings.	Alta
RF-15	L'usuari ha de poder revisar i editar tots els camps abans de publicar l'anunci.	Alta

Mòduls complementaris

ID	REQUISIT	PRIORITAT
RF-16	Els usuaris han de poder afegir i eliminar vehicles de la seva wishlist.	Alta
RF-17	El sistema ha de permetre enviar un missatge/lead associat a un anunci concret.	Alta
RF-18	L'administrador ha de tenir accés a un dashboard amb estadístiques bàsiques del sistema.	Mitjana
RF-19	L'administrador ha de poder gestionar usuaris i anuncis des del panell d'administració.	Mitjana
RF-20	L'API ha d'estar documentada i accessible via Swagger UI.	Alta

3.3 Requisits no funcionals

ID	REQUISIT	CATEGORIA
RNF-01	L'aplicació ha de ser responsive i funcional en dispositius mòbils (mobile-first).	Usabilitat
RNF-02	Les contrasenyes s'han d'emmagatzemar xifrades amb bcrypt.	Seguretat
RNF-03		Seguretat

ID	REQUISIT	CATEGORIA
	Totes les rutes protegides han de validar el token JWT en cada petició.	
RNF-04	Les variables sensibles (secrets, claus API) no s'han de versionar al repositori.	Seguretat
RNF-05	El backend ha d'estar dockeritzat per facilitar el desplegament.	Operacions
RNF-06	L'API ha de seguir les convencions REST i retornar codis HTTP adequats.	Interoperabilitat
RNF-07	El temps de resposta de les operacions principals ha de ser inferior a 2 segons.	Rendiment
RNF-08	El sistema ha de gestionar errors de forma consistent, retornant missatges descriptius.	Fiabilitat
RNF-09	El codi ha de seguir les convencions de cada tecnologia i estar ben estructurat per facilitar el manteniment.	Mantenibilitat

3.4 Anàlisi d'alternatives tecnològiques

Imatge conceptual: alternatives de base de dades analitzades.

Frontend: Ionic Vue vs alternatives

TECNOLOGIA	AVANTATGES	INCONVENIENTS	DECISIÓ
Ionic Vue	Mobile-first, Capacitor, components natius, ecosistema Vue 3	Menys popular que React	✓ **Seleccionat**
React Native	Gran comunitat, rendiment natiu	Corba d'aprenentatge, no comparteix codi web	Descartat
Angular	Framework complet, TypeScript natiu	Verbós, complex per a un sol desenvolupador	Descartat
Flutter	Rendiment excel·lent, cross-platform	Dart —fora del currículum DAM—, ecosistema diferent	Descartat

Backend: Spring Boot vs alternatives

TECNOLOGIA	AVANTATGES	INCONVENIENTS	DECISIÓ
Spring Boot (Java)	Seguretat robusta, arquitectura per capes, JPA, Swagger natiu, alineació DAM	Més verbós que Node.js	✓ **Seleccionat**
Node.js + Express	Ràpid de configurar, JavaScript full-stack	Seguretat manual, menys estructura	Descartat
Django (Python)	Bateries incloses, ORM potent	Fora del currículum DAM	Descartat
NestJS	Estructura similar a Angular, TypeScript	Menys madur, ecosistema més petit	Descartat

Base de dades: PostgreSQL vs alternatives

TECNOLOGIA	AVANTATGES	INCONVENIENTS	DECISIÓ
PostgreSQL	Relacional, ACID, cloud deployment (Aiven), JPA compatible	Requereix esquema definit	✓ **Seleccionat**
MySQL	Molt estesa, fàcil de configurar	Menys funcionalitats avançades que PostgreSQL	Descartat

TECNOLOGIA	AVANTATGES	INCONVENIENTS	DECISIÓ
MongoDB	Flexible, sense esquema	Menys adequada per a dades relacionals (vehicles, usuaris, anuncis)	Descartat

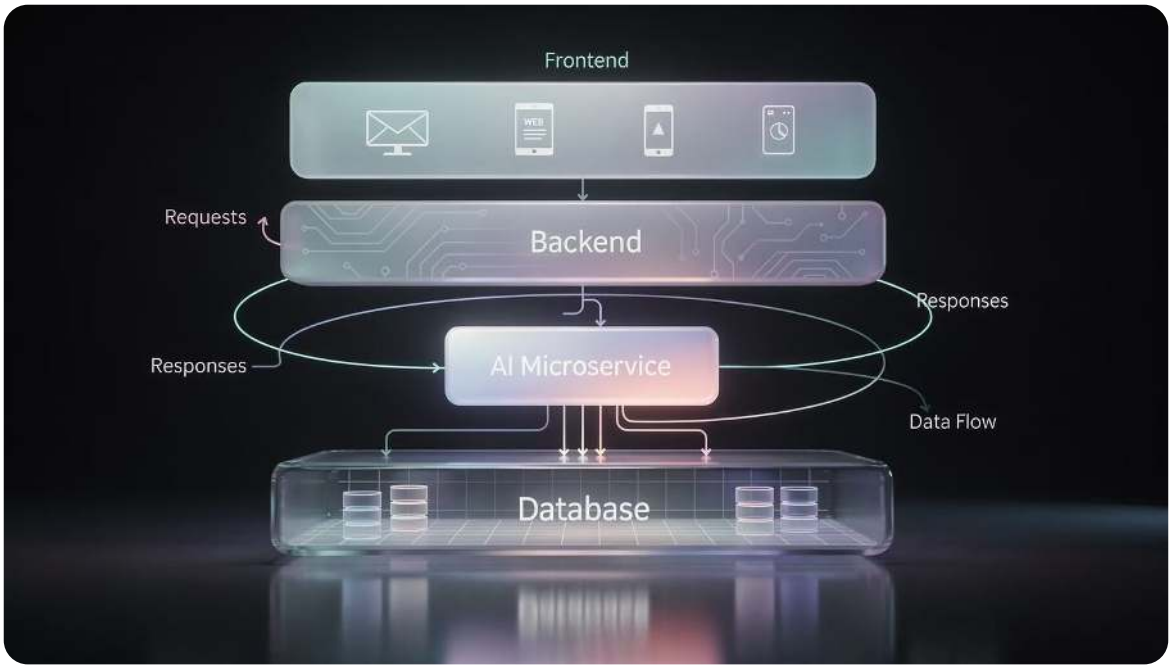
CAPÍTOL 4

4. Disseny

4.1 Arquitectura del sistema

Vista general i model arquitectònic

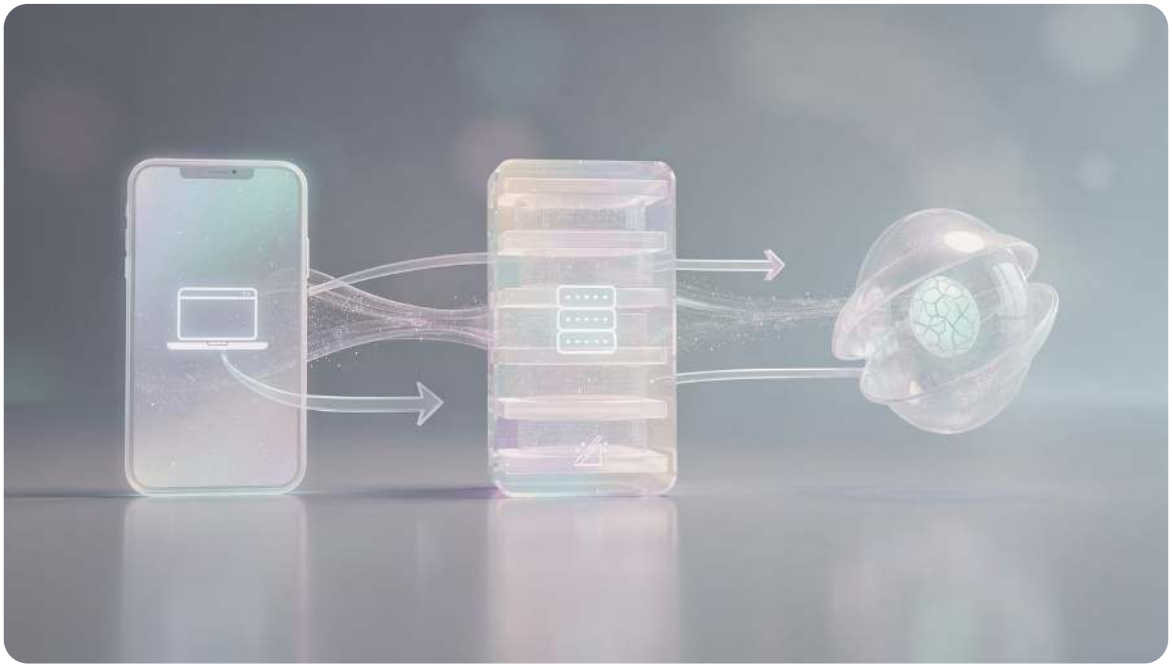
AutoValor segueix una arquitectura **client-servidor multicapa**, en la qual el frontend i el backend estan completament desacoblats i es comuniquen exclusivament a través d'una API REST protegida amb JWT. Aquesta separació facilita el manteniment independent de cadascuna de les capes, permet el desplegament en entorns separats i fa que el sistema sigui escalable de forma horitzontal.



Imatge conceptual: arquitectura multicapa client-servidor d'AutoValor.

Diagrama d'arquitectura

Descripció de les capes



Imatge conceptual: flux de comunicació entre les capes del sistema.

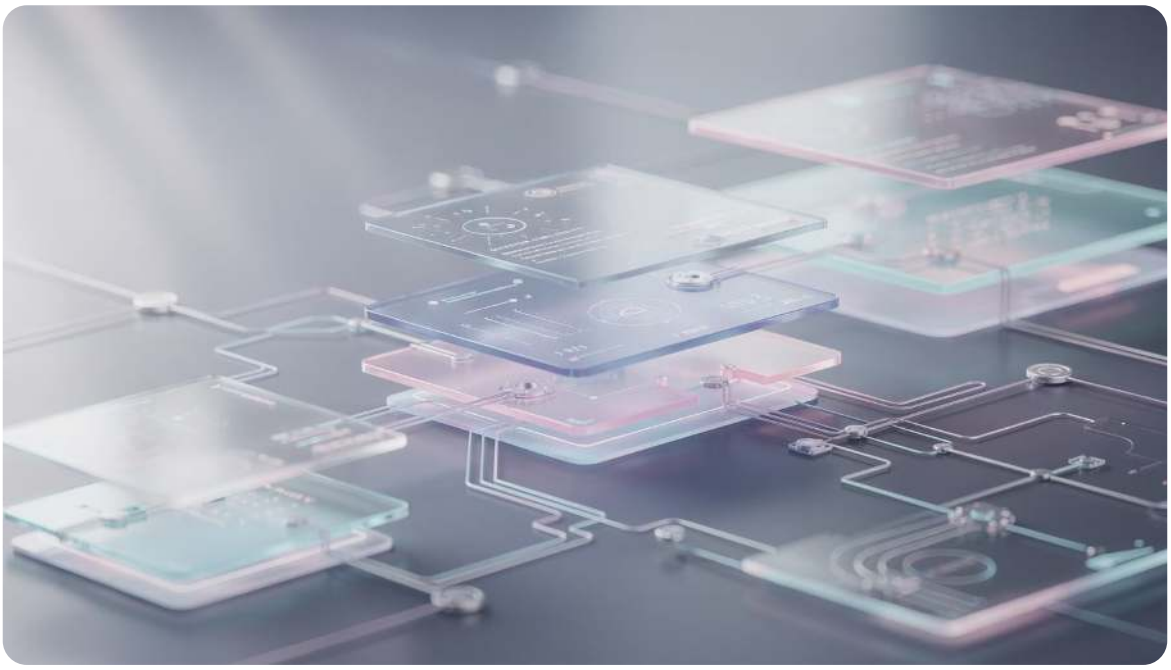
CAPA	TECNOLOGIES	RESPONSABILITAT
Frontend	Ionic Vue, Vue 3, TypeScript, Vite, Capacitor, Pinia	Interfície d'usuari, navegació, gestió d'estat, consum de l'API

CAPA	TECNOLOGIES	RESPONSABILITAT
API REST	Spring Boot, Spring MVC, Spring Security, JWT	Lògica de negoci, autenticació, validació, exposició d'endpoints
Persistència	PostgreSQL (Aiven), JPA/ Hibernate	Emmagatzematge persistent de dades relacionals
Imatges	Cloudinary / sistema de uploads	Emmagatzematge i servei d'imatges dels vehicles
IA	OpenAI API, endpoint /api/ai/ vehicle-suggestions	Anàlisi de fotografies i generació de proposta de dades
Documentació	Swagger UI / OpenAPI 3	Documentació interactiva de l'API REST

4.2 Model de dades

Entitats principals

El model de dades d'AutoValor s'estructura al voltant de les entitats principals del domini de negoci: usuaris, vehicles, imatges, favorits i missatges de contacte. Les relacions entre entitats s'han definit seguint els principis de normalització relacional per garantir la integritat de les dades.



Imatge conceptual: model de dades relacional del sistema.

ENTITAT	CAMPS PRINCIPALS	RELACIONS
User	id, email, password (hash), name, role, createdAt, updatedAt	1:N amb Car (com a venedor), N:M amb Car (com a wishlist)
Car / Listing	id, title, description, brand, model, year, km, price, fuelType, transmission, bodyType, color, status, userId, createdAt	N:1 amb User, 1:N amb CarImage, N:M amb User (wishlist), 1:N amb ContactMessage
CarImage	id, carId, imageUrl, publicId (Cloudinary), order, createdAt	N:1 amb Car
Favorite	id, userId, carId, createdAt	N:1 amb User, N:1 amb Car
ContactMessage	id, carId, senderId, receiverId, message, createdAt, read	N:1 amb Car, N:1 amb User (sender), N:1 amb User (receiver)

Diagrama ER simplificat

4.3 Disseny d'interfície i pantalles principals

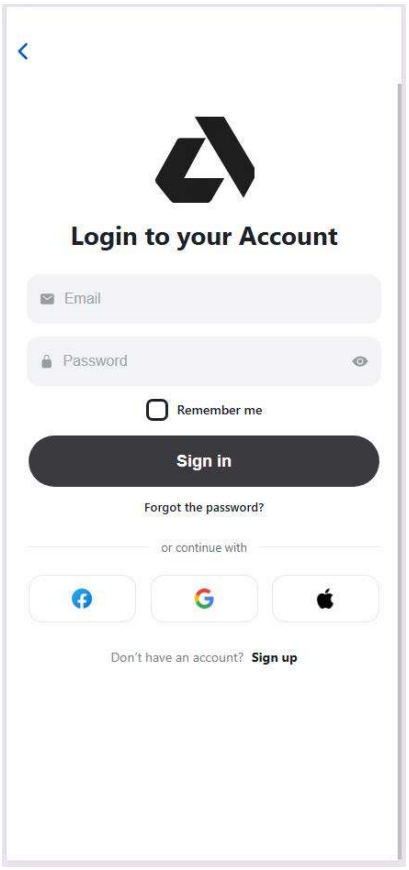
Principis de disseny

La interfície d'AutoValor s'ha dissenyat seguint els principis de **mobile-first** i de coherència visual. L'enfocament mobile-first implica que totes les decisions de disseny es prenen primer per a pantalles petites (360–414 px d'amplada) i s'adapten progressivament a pantalles més grans, mai a l'inrevés. Aquesta prioritat garanteix que l'experiència principal —la compra i la venda de vehicles— sigui fluida i natural en un dispositiu mòbil.

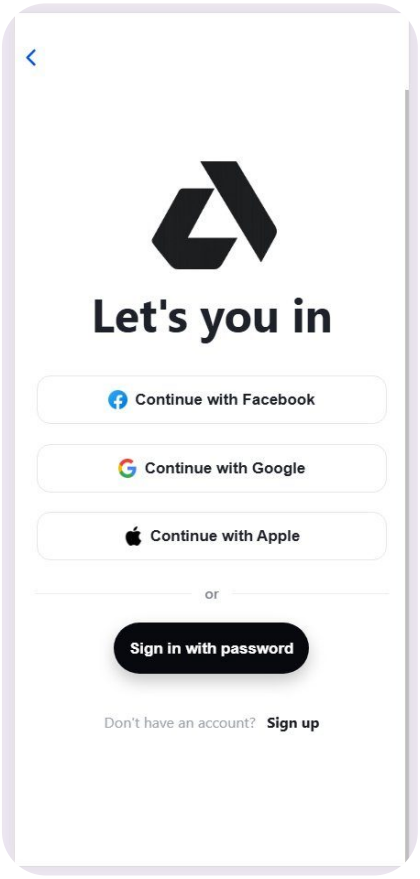
PRINCIPI	APLICACIÓ A AUTOVALOR
Mobile-first	Disseny pensat per a pantalles tàctils; tots els elements interactius superen els 44 px d'alçada mínima per facilitar la interacció amb el dit.
Targetes arrodonides	Els anuncis es presenten en targetes amb cantonades arrodonides, sombra subtil i espai blanc generos per facilitar la lectura i la separació visual de continguts.
Navegació inferior	La barra de navegació inferior fixa conté les cinc seccions principals: **Home, Orders, Sell, Messages i Profile**. Segueix el patró estàndard de les apps mòbils natives i permet canviar de secció amb un sol toc.
Color d'accentuació	S'utilitza el morat/lavanda com a color d'accentuació principal. Apareix als botons primaris, als indicadors d'estats actius i als elements destacats, creant una identitat visual recognoscible i consistent.
Jerarquia visual	Cada pantalla aplica una jerarquia tipogràfica clara: títol gran → subtítol → cos de text → metadades. El pes visual guia l'ull de l'usuari cap a la informació més rellevant (preu, marca, model) sense requerir esforç de cerca.
Disseny tàctil	Els botons d'acció principal (Contact, Make offer, Sell) ocupen tota l'amplada disponible o una zona de toc àmplia per evitar errors en pantalla tàctil.
Consistència entre pantalles	S'utilitzen els mateixos components d'Ionic (IonCard, IonButton, IonItem, IonLabel) a totes les pantalles, garantint una experiència visual homogènia i reduint la corba d'aprenentatge de l'usuari.
Accessibilitat bàsica	Contrast de color adequat entre text i fons; labels descriptives als camps de formulari; icones acompanyades de text descriptiu a la barra inferior.

Imatge conceptual: interfície de llistat de vehicles.

Pantalla de Login i Registre



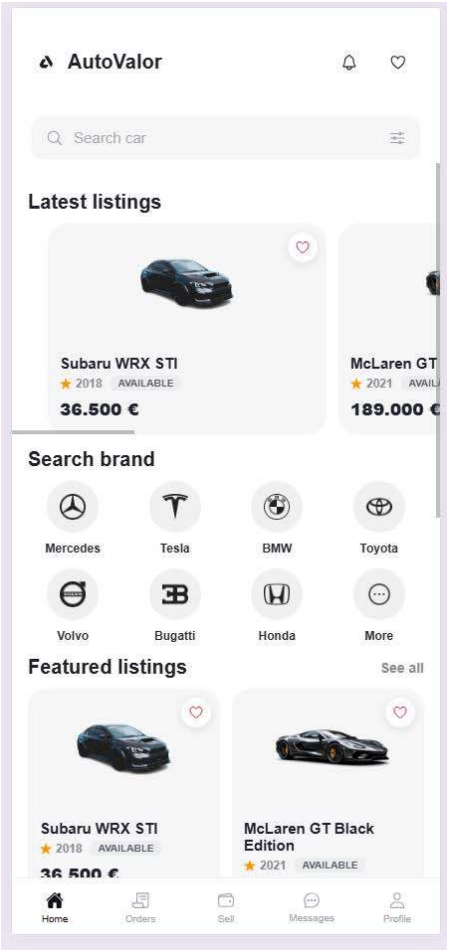
Captura real de l'aplicació — Login d'AutoValor



Captura real de l'aplicació — Pantalla Let's you in d'AutoValor

La pantalla de login permet a l'usuari autenticar-se amb el seu email i contrasenya. Inclou l'opció "Remember me" i accés ràpid a la pantalla de registre. La pantalla *Let's you in* ofereix el registre com a nou usuari. Ambdues pantalles presenten el logotip d'AutoValor centrat, un disseny net i camps de formulari amb icones descriptives. El backend valida les credencials, genera el token JWT i el retorna al frontend, que l'emmagatzema i el fa servir en totes les peticions posteriors.

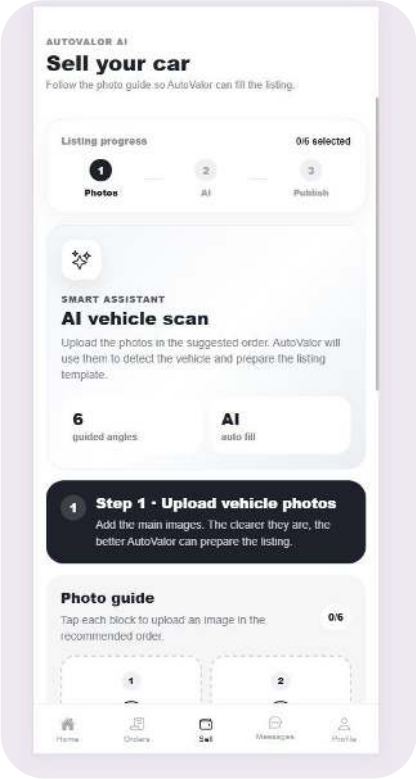
Pantalla Home



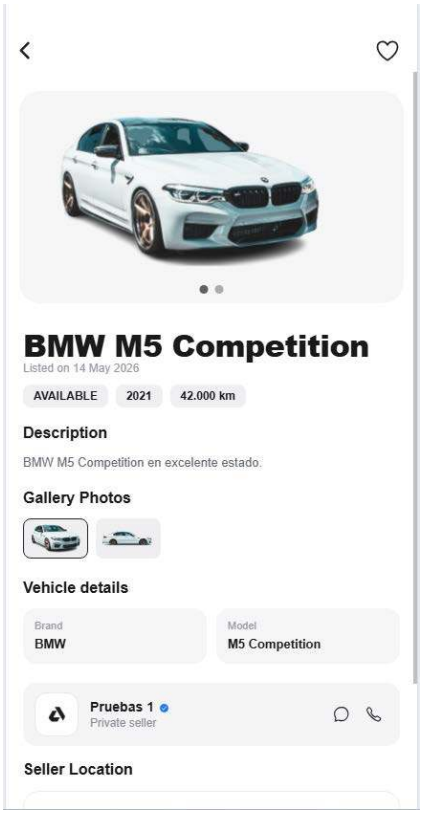
Home — llistat principal d'anuncis

La pantalla Home mostra tres seccions diferenciades: *Latest listings* (anuncis més recents), *Search brand* (cerca ràpida per marca amb logos de Mercedes, Tesla, BMW, Toyota, Volvo, Bugatti, Honda i altres) i *Featured listings* (anuncis destacats). La barra superior inclou el logotip, notificacions i accés ràpid a la wishlist.

Pantalla Sell — Flux Photos → AI → Publish



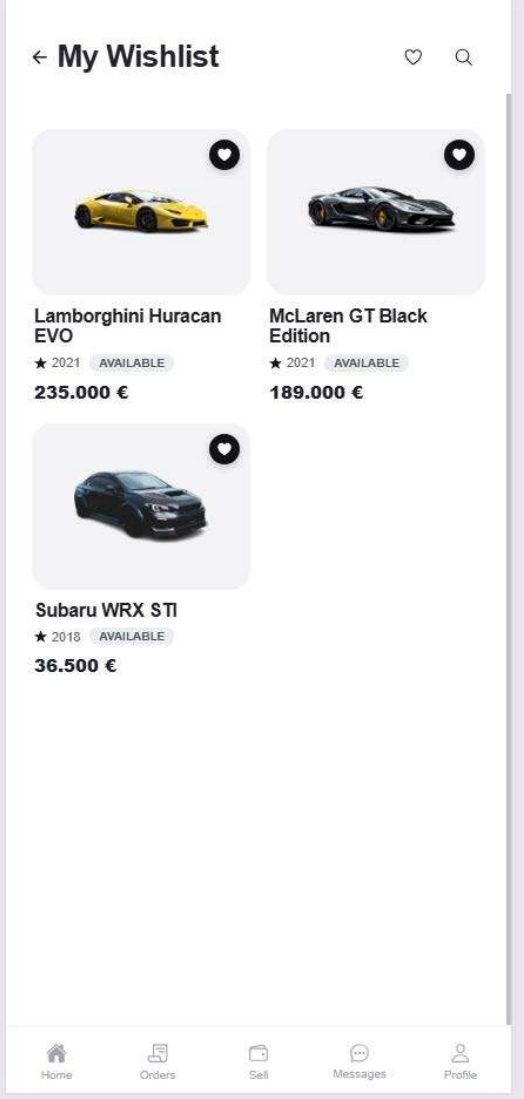
Captura real de l'aplicació — Pantalla Sell d'AutoValor



Captura real de l'aplicació — *Anunci creat*

La pantalla Sell guia l'usuari a través d'un flux en tres passos: **1. Photos** (pujada de fins a 6 fotografies en ordre suggerit), **2. AI** (anàlisi de les imatges i generació de la proposta) i **3. Publish** (revisió, edició i publicació). La guia fotogràfica indica els sis angles recomanats per a una millor detecció per part de la IA.

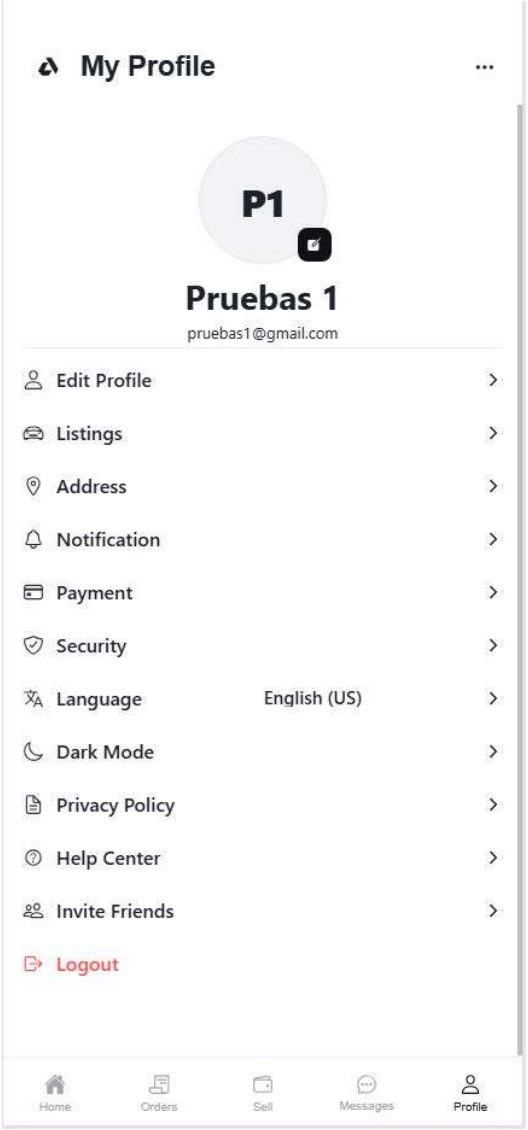
Pantalla Wishlist



Captura real de l'aplicació — Wishlist d'AutoValor

La pantalla Wishlist mostra tots els vehicles que l'usuari ha marcat com a favorits. Cada targeta inclou la imatge principal, el nom, l'any, l'estat i el preu del vehicle. L'usuari pot eliminar un vehicle de la wishlist prement la icona de cor.

Pantalla de Perfil



Captura real de l'aplicació — Perfil d'AutoValor

La pantalla de perfil centralitza tota la gestió del compte de l'usuari: informació personal, accés als anuncis propis (Listings), adreça, notificacions, pagament, seguretat, idioma i mode fosc. L'opció de Logout tanca la sessió i elimina el token JWT del dispositiu.

Pantalla Home

Captura real de l'aplicació — Detall de vehicle a AutoValor

La pantalla de detall mostra la galeria d'imatges del vehicle (amb carrusel), les dades tècniques estructurades (marca, model, any, km, combustible, transmissió), la descripció, el preu i la informació del venedor. Inclou botons de "Make offer" i "Contact" per iniciar la comunicació amb el venedor.

CAPÍTOL 5

5. Desenvolupament i implementació

5.1 Estructura dels repositoris

El projecte es divideix en dos repositoris independents a GitHub, cadascun corresponent a una de les capes principals del sistema:

REPOSITORI	URL	TECNOLOGIA PRINCIPAL
Backend	github.com/VictorMartosSegura/autovalor	Java 21 + Spring Boot
Frontend	github.com/VictorMartosSegura/autovalor_frontend	Ionic Vue + TypeScript

Estructura del backend (Spring Boot)

```
src/main/java/com/autovalor/ ├── config/ # CORS, Swagger, beans de configuració ├── controller/ # Endpoints REST (AuthController, CarController...) ├── service/ # Lògica de negoci (AuthService, CarService...) ├── repository/ # Interfaces JPA (UserRepository, CarRepository...) ├── entity/ # Entitats JPA (User, Car, CarImage, Favorite...) ├── dto/ # DTOs de petició i resposta ├── security/ # JWT filter, UserDetailsService, SecurityConfig └── exception/ # GlobalExceptionHandler, errors personalitzats
```

Estructura del frontend (Ionic Vue)

```
src/ ├── views/ # Pàgines principals (HomePage, SellPage, ProfilePage...) ├── components/ # Components reutilitzables (CarCard, FilterModal...) ├── services/ # apiClient, listingService, vehicleAiService, authService... ├── stores/ # Pinia stores (authStore, wishlistStore, preferencesStore) ├── router/ # Ionic Vue Router amb guards (requiresAuth, requiresAdmin) ├── assets/ #
```

```
Imatges, icones i recursos estàtics └── types/ # Definicions
TypeScript (Car, User, ApiResponse...)
```

5.2 Backend: Spring Boot

Arquitectura per capes

El backend segueix una arquitectura estricta per capes: els **controllers** reben les peticions HTTP i delegen la lògica als **services**, que al seu torn accedeixen a les dades a través dels **repositories** de JPA. Els **DTOs** s'utilitzen per a la transferència de dades entre capes, evitant exposar les entitats directament a l'API.

Imatge conceptual: panel d'administració amb estadístiques.

Mòduls del backend

MÒDUL	ENDPOINTS PRINCIPALS	DESCRIPCIÓ
Auth	POST /api/auth/register, POST /api/auth/login	Registre d'usuari, login i generació de JWT
Users		Gestió del perfil d'usuari

MÒDUL	ENDPOINTS PRINCIPALS	DESCRIPCIÓ
	GET/PUT /api/users/me, GET /api/users/{id}	
Cars	GET/POST /api/cars, GET/PUT/DELETE /api/cars/{id}	CRUD complet d'anuncis de vehicles
Images	POST /api/cars/{id}/images GET /api/cars/{id}/images DELETE /api/cars/{id}/images/{imageId}	Pujada, llistat i eliminació d'imatges per anunci via Cloudinary
Favorites	GET/POST/DELETE /api/favorites	Gestió de la wishlist de l'usuari
Contact	POST /api/contact, GET /api/contact-messages	Enviament i recepció de leads/missatges
AI	POST /api/ai/vehicle-suggestions	Anàlisi d'imatges i generació de proposta de dades
Admin	GET /api/admin/stats GET /api/admin/users PATCH /api/admin/users/{id}/role GET /api/admin/listings PATCH /api/admin/listings/{id}/status DELETE /api/admin/listings/{id} GET /api/admin/contact-messages	Estadístiques i gestió administrativa (sol ADMIN)
Health	GET /api/health	Verificació de l'estat del servei

Gestió d'imatges amb Cloudinary

Les imatges dels anuncis s'emmagatzemen a Cloudinary. Quan l'usuari puja una fotografia, el backend valida el fitxer (format JPG, PNG o WEBP), comprova que no se superi el límit màxim d'imatges per anunci i envia el contingut a Cloudinary. El servei retorna una URL segura i un *public_id* (identificador únic del fitxer al núvol), ambdós associats a l'anunci a la base de dades. D'aquesta manera, PostgreSQL només conserva les metadades de la imatge, mentre que el fitxer binari es distribueix des d'un servei especialitzat al núvol.

La integració de Cloudinary aporta avantatges significatius respecte a l'emmagatzematge en local o en la pròpia base de dades: les imatges es distribueixen via CDN (Content Delivery Network), s'apliquen

transformacions automàtiques (redimensionat, compressió, conversió de format), i l'eliminació d'una imatge és possible en qualsevol moment identificant-la pel seu *public_id*.

OPERACIÓ	DETALL
Pujada	Validació de format (JPG, PNG, WEBP) + enviament a Cloudinary + emmagatzematge de URL i public_id
Llistat	Consulta de les imatges associades a un anunci a PostgreSQL, amb les URLs de Cloudinary
Eliminació	Crida a l'API de Cloudinary amb el public_id + eliminació del registre a la BD
Límit	Nombre màxim d'imatges per anunci configurable; el backend retorna error si se supera

Validació i gestió d'errors

El backend implementa un sistema de validació declarativa mitjançant les anotacions de Bean Validation (`@NotNull`, `@Email`, `@Size`, etc.) als DTOs de petició. Els errors de validació i les excepcions del domini es gestionen de forma centralitzada a través del **GlobalExceptionHandler**, que captura les excepcions i les transforma en respostes JSON estructurades amb el codi HTTP adequat.

5.3 Frontend: Ionic Vue

Routing i guards de navegació

El routing s'implementa amb Ionic Vue Router, que combina les capacitats de Vue Router amb les animacions i transicions natives d'Ionic. S'han definit dos guards de navegació:

- **requiresAuth:** verifica que l'usuari tingui un token JWT vàlid emmagatzemat. Si no el té, redirigeix a la pantalla de login.
- **requiresAdmin:** verifica, a més, que el rol de l'usuari sigui ADMIN. Si no ho és, redirigeix a la pantalla principal.

Gestió d'estat amb Pinia

L'estat global de l'aplicació es gestiona amb **Pinia**, el store oficial de Vue 3. S'han definit tres stores principals:

- **authStore:** emmagatzema el token JWT, les dades de l'usuari i l'estat d'autenticació. Proveeix mètodes de login, logout i refresc del perfil.
- **wishlistStore:** gestiona la llista de vehicles favorits de l'usuari, sincronitzant-se amb el backend a cada modificació.
- **preferencesStore:** emmagatzema les preferències locals de l'usuari (idioma, mode fosc, filtres de cerca recents).

Client API centralitzat

Totes les peticions al backend es realitzen a través d'un **apiClient** centralitzat basat en la **API Fetch** nativa del navegador, configurat a partir de la variable d'entorn `VITE_API_BASE_URL`. No s'utilitza Axios ni cap llibreria externa de peticions HTTP. El client:

- Construeix les URLs combinant `VITE_API_BASE_URL` amb la ruta de l'endpoint.
- Afegeix automàticament la capçalera `Authorization: Bearer <token>` quan l'usuari disposa de sessió activa.
- Parseja les respostes com a JSON o text segons el `Content-Type` retornat.
- Centralitza els errors mitjançant la classe `ApiClientError`, que captura els codis HTTP d'error i en propaga el missatge al component corresponent.

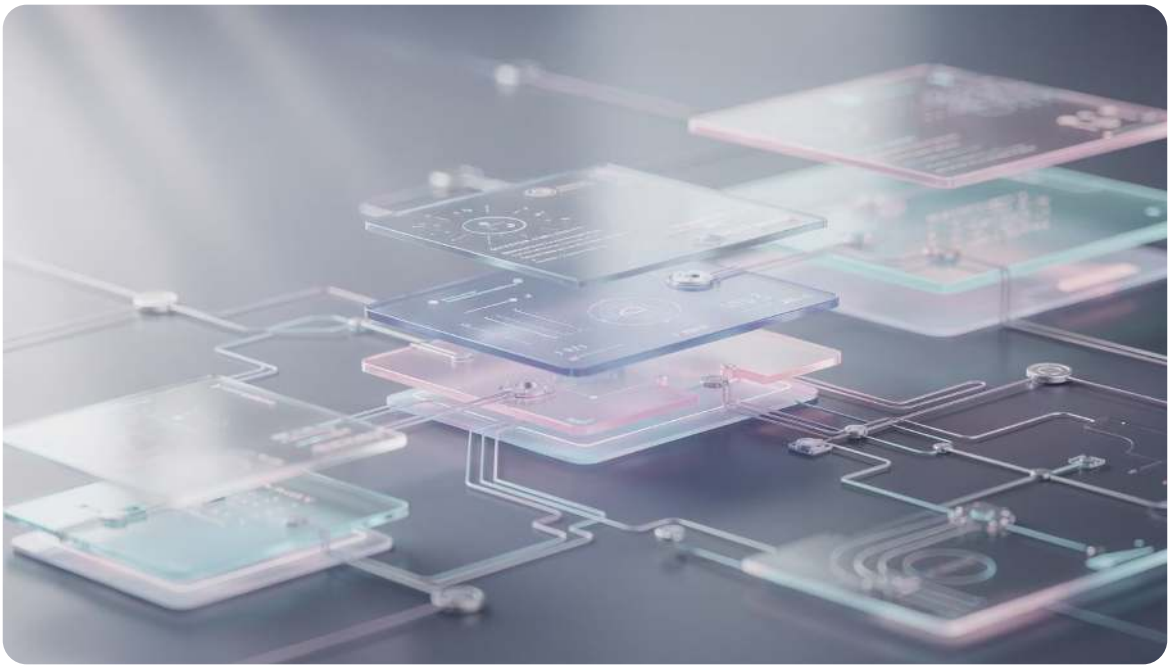
Serveis TypeScript

SERVEI	RESPONSABILITAT
authService	Login, registre, renovació de token, logout
listingService	CRUD d'anuncis, cerca, filtres i paginació
imageService	Pujada i eliminació d'imatges per anunci
vehicleAiService	Enviar imatges al backend i rebre la proposta de la IA
favoriteService	Afegir/eliminar de wishlist, obtenir llista de favorits
contactService	Enviament i recepció de missatges/leads
adminService	Estadístiques, gestió d'usuaris i anuncis (sol ADMIN)

5.4 Seguretat

Model de seguretat

La seguretat d'AutoValor es basa en un model d'autenticació stateless basat en **JSON Web Tokens (JWT)**, gestionat per Spring Security al backend i verificat pels guards de navegació al frontend.



Imatge conceptual: model de seguretat JWT i protecció de rutes.

Mecanismes de seguretat implementats

MECANISME	IMPLEMENTACIÓ	DESCRIPCIÓ
JWT	spring-security-jwt, jjwt	Token signat amb secret configurable, expiració definida
Xifrat de contrasenyes	BCryptPasswordEncoder	Les contrasenyes s'emmagatzemen sempre xifrades amb bcrypt (cost 10)
Rols i autoritzacions	Spring Security @PreAuthorize	Rols USER i ADMIN; rutes protegides per rol a nivell de mètode
Rutes públiques	SecurityConfig.permitAll()	GET /api/cars, GET /api/cars/{id}, POST /api/auth/* accessible sense token
Variables sensibles	.env / application.properties	JWT_SECRET, DB_URL, OPENAI_API_KEY mai versionats al repositori
CORS	Spring Boot CorsConfigurationSource	Orígens permesos configurables via CORS_ALLOWED_ORIGINS
Validació d'entrada	Bean Validation + GlobalExceptionHandler	Tots els DTOs validen camps obligatoris, format i longitud

MECANISME	IMPLEMENTACIÓ	DESCRIPCIÓ
Frontend guards	Ionic Vue Router beforeEach	requiresAuth i requiresAdmin protegeixen les rutes del frontend

5.5 AutoValor AI: integració d'intel·ligència artificial

Funcionament del mòdul d'IA

El mòdul AutoValor AI és el component diferenciador del projecte. Permet als venedors generar automàticament una proposta completa de dades per al seu anunci a partir de les fotografies del vehicle, sense necessitat de conèixer els detalls tècnics del cotxe. El flux s'estructura en tres fases:

Imatge conceptual: flux AutoValor AI de generació de proposta d'anunci.

Flux Photos → AI → Publish

1. **Photos (pujada de fotografies):** L'usuari puja fins a 6 fotografies del vehicle, seguint la guia de la pantalla Sell: *frontal, posterior, lateral, interior, quadre de comandament /*

quilometratge i extra (motor / documentació). Cada imatge s'afegeix a un FormData que s'enviarà al backend.

2. **AI (anàlisi i generació de proposta):** El frontend envia les imatges al backend via `POST /api/ai/vehicle-suggestions`. El backend construeix un prompt estructurat en anglès que descriu el context, les imatges aportades i el format de resposta esperat (JSON estructurat). El prompt s'envia juntament amb les imatges a l'API d'IA (OpenAI). La IA analitza les imatges i retorna una resposta JSON amb la proposta de dades.
3. **Publish (revisió i publicació):** El frontend rep la proposta de la IA i omple el formulari de creació d'anunci amb els valors suggerits. L'usuari pot revisar i editar qualsevol camp abans de confirmar la publicació. La IA mai publica l'anunci de forma automàtica.

Estructura de la resposta de la IA

```
{ "title": "Subaru WRX STI 2018 300CV", "description": "Subaru WRX
STI en molt bon estat...", "brand": "Subaru", "model": "WRX STI",
"year": 2018, "km": 79000, "fuelType": "Gasoline", "transmission":
"Manual", "bodyType": "Sedan", "color": "Black", "confidence": 0.87,
"warnings": ["El quilometratge s'ha estimat a partir del quadre;
confirma'l manualment"] }
```

Principis de disseny del mòdul d'IA

- **La IA és un assistent, no un automatisme:** totes les dades generades per la IA poden ser editades per l'usuari i mai es publiquen sense la seva revisió i confirmació explícita.
- **Transparència en la confiança:** el camp *confidence* indica el nivell de confiança de la IA en la proposta generada. El camp *warnings* adverteix l'usuari dels camps que s'han estimat amb menys certesa i que hauria de verificar manualment.
- **Prompt estructurat:** el backend construeix el prompt de forma programàtica, incloent el context del sistema, les instruccions de format de resposta i les restriccions per evitar al·lucinacions de la IA.

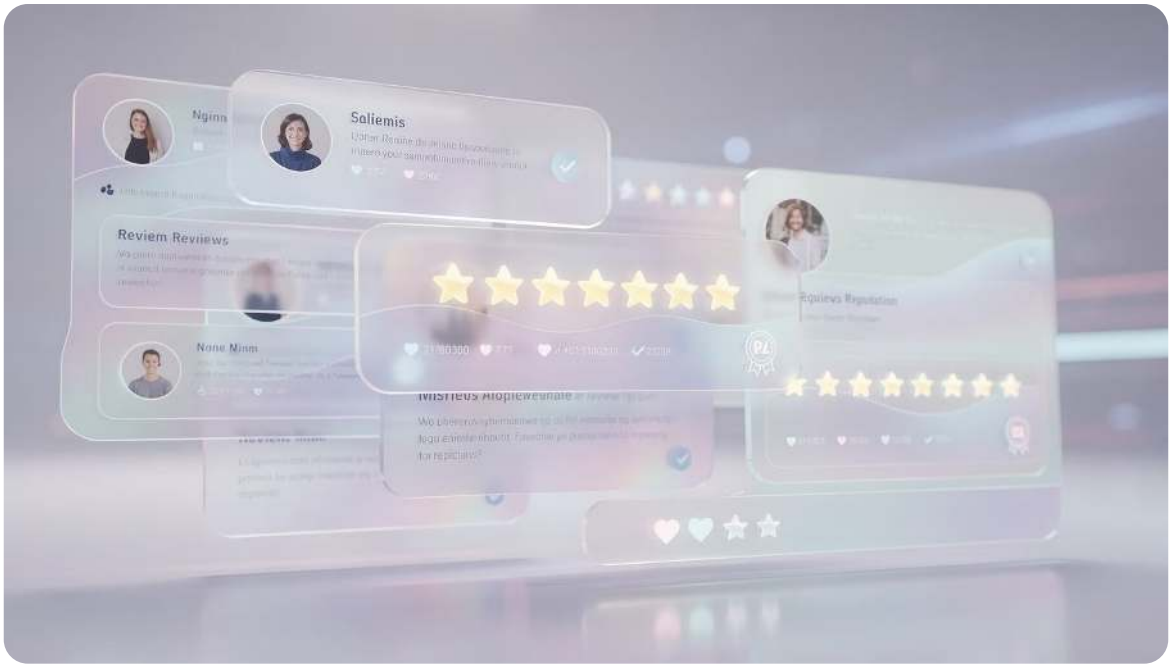
Limitacions del mòdul d'IA

- La precisió de la proposta depèn directament de la qualitat i l'angle de les fotografies aportades.
- Camps com el quilometratge o l'any exacte poden requerir confirmació manual si no són clarament visibles a les imatges.
- El sistema no detecta danys ni l'estat de conservació del vehicle de forma fiable; aquesta valoració és responsabilitat del venedor.
- L'endpoint depèn de la disponibilitat de l'API externa d'IA; si l'API no està disponible, el flux Sell continua funcionant sense la proposta automàtica.

5.6 Proves i verificació

Estratègia de proves

La verificació del sistema s'ha realitzat mitjançant una combinació de proves manuals d'integració, proves d'endpoints a través de Swagger UI i Postman, i proves de build dels dos repositoris. A continuació es detallen els casos de prova principals:



Imatge conceptual: verificació i proves del sistema.

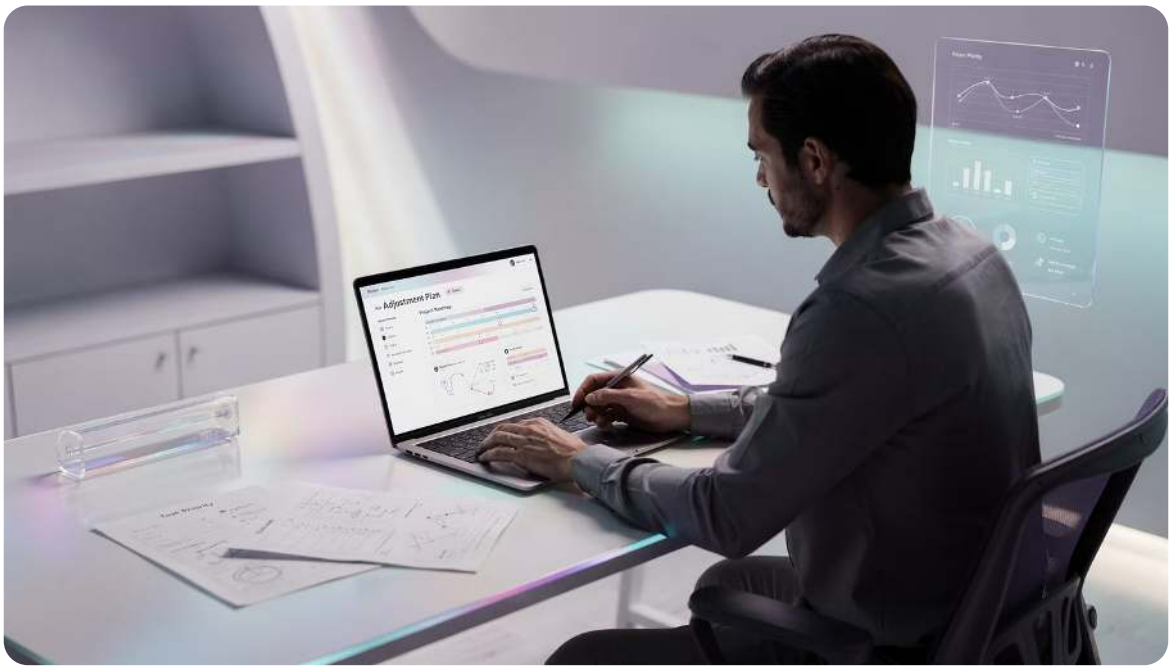
ID	PROVA	PAS PRINCIPAL	RESULTAT ESPERAT	ESTAT
PT-01	Registre d'usuari	POST /api/auth/register amb dades vàlides	201 Created + usuari creat a la BD	✓ OK
PT-02	Login correcte	POST /api/auth/login amb credencials vàlides	200 OK + token JWT vàlid	✓ OK
PT-03	Login incorrecte	POST /api/auth/login amb contrasenya errònia	401 Unauthorized + missatge d'error	✓ OK
PT-04	Crear anunci	POST /api/cars amb token vàlid i dades completes	201 Created + anunci a la BD	✓ OK
PT-05	Crear anunci sense token	POST /api/cars sense Authorization header	401 Unauthorized	✓ OK
PT-06	Pujar imatges	POST /api/cars/{id}/images amb fitxer JPEG	200 OK + URL d'imatge retornada	✓ OK
PT-07	Flux IA complet	POST /api/ai/vehicle-suggestions amb 6 fotos	200 OK + JSON amb proposta de dades	✓ OK
PT-08	Cerca de vehicles	GET /api/cars?brand=Subaru&maxKm=100000	200 OK + llistat filtrat paginat	✓ OK
PT-09	Afegir a wishlist	POST /api/favorites amb carId vàlid	201 Created + vehicle a la wishlist	✓ OK
PT-10	Eliminar de wishlist	DELETE /api/favorites/{id}	200 OK + vehicle eliminat de la wishlist	✓ OK
PT-11	Contactar venedor	POST /api/cars/{listingId}/contact amb missatge vàlid	201 Created + lead registrat	✓ OK
PT-12	Accés admin amb ADMIN	GET /api/admin/stats amb token ADMIN (rol ADMIN)	200 OK + estadístiques del sistema	✓ OK
PT-13	Accés admin	GET /api/admin/stats amb token USER (rol USER)	403 Forbidden	✓ OK

ID	PROVA	PAS PRINCIPAL	RESULTAT ESPERAT	ESTAT
	sense ADMIN			
PT-14	Build backend	./gradlew clean bootJar	Build SUCCESS, .jar generat	✓ OK
PT-15	Build frontend	npm run build	Build completat sense errors	✓ OK
PT-16	Swagger UI accessible	GET /swagger-ui/index.html	Interfície Swagger carregada correctament	✓ OK
PT-17	Health check	GET /api/health	200 OK + {"status": "UP"}	✓ OK

CAPÍTOL 6

6. Conclusions

6.1 Conclusions generals



El projecte AutoValor ha assolit els seus objectius principals, culminant en una aplicació multiplataforma funcional que cobreix el cicle complet de compravenda de vehicles d'ocasió amb assistència d'intel·ligència artificial. La integració de la IA en el flux de publicació d'anuncis és el component diferenciador del projecte i constitueix una proposta de valor real per al mercat de la compravenda de vehicles en línia.

Des d'un punt de vista tècnic, el projecte ha demostrat la viabilitat de construir una aplicació completa —frontend, backend, base de dades, imatges i IA— en un entorn de projecte final de cicle, amb tecnologies professionals i estàndards de qualitat adequats. La separació en dues capes ben definides (frontend Ionic Vue i backend Spring Boot), la gestió de la seguretat amb JWT i rols, la integració amb serveis cloud (Aiven, Cloudinary, OpenAI) i la documentació de l'API amb Swagger

reflecteixen un enfocament madur i professional del desenvolupament de programari.

6.2 Assoliment dels objectius

OBJECTIU	ESTAT	COMENTARI
Registre, login i JWT amb rols USER/ ADMIN	✓ Assolit	Implementat amb Spring Security i JWT
CRUD d'anuncis amb imatges (Cloudinary)	✓ Assolit	Imatges via Cloudinary: pujada, llistat i eliminació implementades
Cerca i filtres d'anuncis	✓ Assolit	Filtres per marca, model, preu, km i any
Flux IA: Photos → AI → Publish	✓ Assolit	Amb proposta editable i confidence + warnings
Wishlist de favorits	✓ Assolit	Sincronitzada amb backend
Contacte amb venedor / leads	✓ Assolit	Missatgeria bàsica implementada
Perfil d'usuari i MyListings	✓ Assolit	Gestió completa del compte
Panel d'administració	✓ Assolit	Stats, usuaris i anuncis gestionables
Desplegament en cloud	○ Assolit parcialment	PostgreSQL en Aiven ✓, Cloudinary implementat ✓, Docker preparat ✓. El desplegament públic complet del backend i frontend es completa com a part de la demo final.
Documentació API (Swagger)	✓ Assolit	Swagger UI disponible i funcional a /swagger-ui/ index.html
Sistema de valoració d'usuaris	○ Futura millora	No inclòs al MVP; documentat com a millora futura
Xat en temps real WebSocket	✓ Assolit	Xat en temps real implementat i funcional amb WebSocket.

6.3 Línies de futur

Imatge conceptual: línies d'evolució futura d'AutoValor.

Les funcionalitats següents **no formen part del MVP lliurat** però s'han identificat com a línies de treball futur amb potencial real per al producte:

- **Sistema de valoració i reputació d'usuaris:** puntuació i ressenyes entre compradors i venedors per incrementar la confiança en les transaccions. *(Futura millora)*

  - Millores del xat en temps real: el xat amb WebSockets ja està implementat i funcional; com a evolució futura es podrien afegir confirmacions de lectura, adjunts i notificacions avançades.

- **Login social:** autenticació via Google, Facebook i Apple. Al MVP, l'autenticació és per email i contrasenya. *(Futura millora)*
- **Notificacions push:** alertes per a Android i iOS via Capacitor quan un vehicle de la wishlist canvia d'estat o s'obté una resposta. *(Futura millora)*
- **Millora del mòdul d'IA:** detecció automàtica de danys, estat de conservació del vehicle i integració de dades de mercat per suggerir un preu just. *(Futura millora)*

- **Historial de preus:** evolució del preu d'un model concret al llarg del temps. *(Futura millora)*
- **Publicació a l'App Store i Google Play:** desplegament de l'app iOS/Android firmada via Capacitor. L'app està preparada tècnicament; la publicació als stores és un pas addicional. *(Futura millora)*
- **Desplegament públic complet:** URL pública del backend i frontend accesible sense configuració local. Al MVP, l'entorn de demo es configura localment o via Docker. *(Pendent de lliurament final)*

7. Glossari

TERME	DEFINICIÓ
API REST	Interfície de programació d'aplicacions que segueix els principis REST (Representational State Transfer), basada en HTTP i que retorna dades en format JSON.
Capacitor	Eina d'Ionic que permet empaquetar aplicacions web en apps natives per a Android i iOS, donant accés a les APIs del dispositiu.
DTO	Data Transfer Object. Objecte que s'utilitza per transferir dades entre les capes d'una aplicació, sense exposar directament les entitats del domini.
FormData	Objecte de JavaScript que permet enviar dades de formulari, incloent fitxers binaris, en peticions HTTP multipart/form-data.
Guard de navegació	Funció en Vue Router que s'executa abans de navegar a una ruta i pot bloquejar la navegació si l'usuari no té els permisos adequats.
JPA	Java Persistence API. Especificació de Java per a la gestió de la persistència de dades en bases de dades relacionals, implementada per Hibernate.
JWT	JSON Web Token. Estàndard obert (RFC 7519) que defineix un format compacte i autocontingut per a la transmissió segura d'informació entre parties com a objecte JSON signat digitalment.
MVP	Minimum Viable Product. Producte mínimament viable que inclou les funcionalitats essencials per a la seva demostració i validació.
Pinia	Llibreria de gestió d'estat per a Vue 3, successora de Vuex, que ofereix una API senzilla i compatible amb TypeScript per a la gestió de l'estat global de l'aplicació.
Prompt	Instrucció o conjunt d'instruccions enviades a un model de IA per guiar la generació de la resposta.
Spring Boot	Framework de Java que simplifica la creació d'aplicacions Spring autosuficients i de producció, amb configuració automàtica i servidor integrat.
Spring Security	Marc de seguretat per a aplicacions Java/Spring que proporciona autenticació, autorització i protecció contra atacs comuns.
Swagger/ OpenAPI	

TERME	DEFINICIÓ
	Estàndard i conjunt d'eines per a la documentació, visualització i proves d'APIs REST, que permet generar una interfície interactiva per explorar els endpoints.
TypeScript	Superconjunt de JavaScript que afegeix tipat estàtic opcional i altres funcionalitats per millorar la mantenibilitat i la detecció d'errors en temps de compilació.
Vite	Eina de construcció i servidor de desenvolupament moderna per a aplicacions web frontend, que ofereix temps de compilació molt ràpids gràcies als mòduls ES nadius.

8. Bibliografia

Documentació oficial de tecnologies

Ionic Framework. (s. f.). *Ionic Vue documentation*. Ionic Framework. Consultat el 16 de maig de 2026, de https://ionicframework.com/docs/vue/overview

Ionic Framework. (s. f.). *Capacitor documentation*. Capacitor by Ionic. Consultat el 16 de maig de 2026, de https://capacitorjs.com/docs

Vue.js. (s. f.). *Vue 3 documentation*. Vue.js. Consultat el 16 de maig de 2026, de https://vuejs.org/guide/introduction

Vite. (s. f.). *Vite documentation*. Vite. Consultat el 16 de maig de 2026, de https://vitejs.dev/guide/

Pinia. (s. f.). *Pinia documentation*. Pinia. Consultat el 16 de maig de 2026, de https://pinia.vuejs.org/introduction.html

Spring. (s. f.). *Spring Boot reference documentation*. Spring. Consultat el 16 de maig de 2026, de https://docs.spring.io/spring-boot/docs/current/reference/html/

Spring. (s. f.). *Spring Security reference documentation*. Spring. Consultat el 16 de maig de 2026, de https://docs.spring.io/spring-security/reference/

PostgreSQL Global Development Group. (s. f.). *PostgreSQL documentation*. PostgreSQL. Consultat el 16 de maig de 2026, de https://www.postgresql.org/docs/

OpenAPI Initiative. (s. f.). *OpenAPI specification*. OpenAPI Initiative. Consultat el 16 de maig de 2026, de https://spec.openapis.org/oas/latest.html

SmartBear. (s. f.). *Swagger UI documentation*. Swagger. Consultat el 16 de maig de 2026, de https://swagger.io/tools/swagger-ui/

Auth0. (s. f.). *Introduction to JSON Web Tokens*. JWT.io. Consultat el 16 de maig de 2026, de https://jwt.io/introduction

Cloudinary. (s. f.). *Cloudinary documentation*. Cloudinary. Consultat el 16 de maig de 2026, de https://cloudinary.com/documentation

OpenAI. (s. f.). *OpenAI API reference*. OpenAI. Consultat el 16 de maig de 2026, de https://platform.openai.com/docs/api-reference

Aiven. (s. f.). *Aiven for PostgreSQL documentation*. Aiven. Consultat el 16 de maig de 2026, de https://aiven.io/docs/products/postgresql

9. Annexos

Annex A — Manual d'instal·lació i posada en marxa

Backend (Spring Boot)

```
# 1. Clonar el repositori git clone https://github.com/
VictorMartosSegura/autovalor.git cd autovalor # 2. Configurar les
variables d'entorn cp .env.example .env # Editar .env amb els valors
correctes: # DB_URL, DB_USERNAME, DB_PASSWORD # JWT_SECRET,
ADMIN_EMAIL, ADMIN_PASSWORD # OPENAI_API_KEY, CORS_ALLOWED_ORIGINS #
CLOUDINARY_URL (o claus individuals) # 3. Executar en mode
desenvolupament ./gradlew bootRun # 4. Executar els tests ./gradlew
clean test # 5. Generar el JAR de producció ./gradlew clean bootJar #
6. Construir la imatge Docker docker build -t autovalor-backend .
docker run -p 8080:8080 --env-file .env autovalor-backend # Swagger
UI disponible a: http://localhost:8080/swagger-ui/index.html
```

Frontend (Ionic Vue)

```
# 1. Clonar el repositori git clone https://github.com/
VictorMartosSegura/autovalor_frontend.git cd autovalor_frontend # 2.
Instal·lar dependències npm install # 3. Configurar la variable
d'entorn cp .env.example .env # Editar .env:
VITE_API_BASE_URL=http://localhost:8080 # 4. Executar en mode
desenvolupament npm run dev # 5. Build de producció npm run build #
6. Previsualitzar la build npm run preview # 7. Executar tests
unitaris npm run test:unit # 8. Executar tests e2e npm run test:e2e
```

Annex B — Variables d'entorn

Backend (.env)

VARIABLE	DESCRIPCIÓ	EXEMPLE
DB_URL	URL de connexió a PostgreSQL	jdbc:postgresql://host:5432/autovalor
DB_USERNAME	Usuari de la base de dades	avnadmin
DB_PASSWORD	Contrasenya de la base de dades	••••••••
JWT_SECRET	Secret per a la signatura de tokens JWT	clau-secreta-llarga-i-aleatoria
ADMIN_EMAIL	Email de l'usuari administrador inicial	admin@autovalor.com
ADMIN_PASSWORD	Contrasenya de l'administrador inicial	••••••••
OPENAI_API_KEY	Clau de l'API d'OpenAI per al mòdul d'IA	sk-••••••••
CORS_ALLOWED_ORIGINS	Orígens permesos per CORS	http://localhost:5173,https://autovalor.app
CLOUDINARY_URL	URL de connexió a Cloudinary	cloudinary://api_key:api_secret@cloud_name

Frontend (.env)

VARIABLE	DESCRIPCIÓ	EXEMPLE
VITE_API_BASE_URL	URL base del backend API	http://localhost:8080 (dev) / https://api.autovalor.app (prod)

Annex C — Endpoints principals de l'API

MÈTODE	ENDPOINT	AUTH	DESCRIPCIÓ
POST	/api/auth/register	Públic	Registre d'un nou usuari
POST	/api/auth/login	Públic	Login i obtenció de token JWT
GET	/api/cars	Públic	Llistat d'anuncis amb filtres i paginació
GET	/api/cars/{id}	Públic	Detall d'un anunci específic
POST	/api/cars	USER	Crear un nou anunci
PUT	/api/cars/{id}	USER (propietari)	Editar un anunci existent
DELETE	/api/cars/{id}	USER (propietari)	Eliminar un anunci
POST	/api/cars/{id}/images	USER	Pujar imatges per a un anunci
GET	/api/cars/{id}/images	Públic	Llistat d'imatges d'un anunci
POST	/api/ai/vehicle-suggestions	USER	Generar proposta de dades a partir de fotos
GET	/api/favorites	USER	Wishlist de l'usuari autenticat
POST	/api/favorites	USER	Afegir un vehicle a la wishlist
DELETE	/api/favorites/{id}	USER	Eliminar un vehicle de la wishlist
POST	/api/cars/{listingId}/contact	USER	Enviar un lead vinculat a un anunci concret
GET	/api/contact-messages	USER	Consultar missatges rebuts de compradors
GET	/api/cars/{listingId}/ contact-messages	USER	Missatges associats a un anunci concret
GET	/api/admin/stats	ADMIN	Estadístiques generals del sistema
GET	/api/admin/users	ADMIN	Llistat de tots els usuaris

MÈTODE	ENDPOINT	AUTH	DESCRIPCIÓ
PATCH	/api/admin/users/{userId}/role	ADMIN	Canviar el rol d'un usuari
GET	/api/admin/listings	ADMIN	Llistat de tots els anuncis
PATCH	/api/admin/listings/{listingId}/status	ADMIN	Canviar l'estat d'un anunci
DELETE	/api/admin/listings/{listingId}	ADMIN	Eliminar un anunci
GET	/api/admin/contact-messages	ADMIN	Consultar tots els missatges de contacte
GET	/api/health	Públic	Health check del backend

Annex D — Repositoris i branques

REPOSITORI	URL	BRANCA PRINCIPAL	TECNOLOGIA
Backend	github.com/VictorMartosSegura/autovalor	master	Java 21 + Spring Boot
Frontend	github.com/VictorMartosSegura/autovalor_frontend	main	Ionic Vue + TypeScript

Annex E — Possibles millores futures

FUNCIONALITAT	DESCRIPCIÓ	PRIORITAT ESTIMADA
Sistema de valoració	Puntuació i ressenyes entre compradors i venedors	Alta
Millores del xat	Xat WebSocket implementat; possibles millores	
Login social	Autenticació via Google, Facebook i Apple	Mitjana
Notificacions push	Alertes per a iOS i Android via Capacitor	Mitjana
	Anàlisi de l'estat de conservació del vehicle	Baixa

FUNCIONALITAT	DESCRIPCIÓ	PRIORITAT ESTIMADA
Detecció de danys per IA		
Historial de preus	Evolució del preu d'un model al llarg del temps	Baixa
Publicació a stores	App Store i Google Play via Capacitor (tècnicament preparada)	Mitjana
Desplegament públic complet	URL pública de backend i frontend; actualmente configurable via Docker i .env	Alta

QR cap a la web

10. Nota sobre l'ús de la intel·ligència artificial

Nota final

Durant l'elaboració d'aquesta memòria he utilitzat eines d'intel·ligència artificial com a suport en diferents fases de la redacció, principalment per estructurar alguns apartats, millorar la fluïdesa del text i generar esborranys inicials que posteriorment he revisat, adaptat i completat.

En tot moment he estat jo qui ha pres les decisions tècniques, qui ha definit el contingut de cada secció i qui ha verificat que el que s'hi explica reflecteix fidelment el treball realitzat en el projecte AutoValor.

La intel·ligència artificial ha actuat com una eina de suport a la redacció i a l'organització d'idees, no com a substitut del meu criteri ni del meu treball propi. Assumeixo la responsabilitat completa sobre el contingut d'aquest document.