

SMALL BUSINESS VENDOR DIRECTORY

A web platform connecting local vendors with customers through searchable, moderated listings.

Team 7:

- **Bianca-Gabriela Leoveanu**
- **Berin Venedik**
- **Sevil Nik**
- **Nelli Jemina Tuulikki tarkka**
- **Tun Wellens**



PROBLEM STATEMENT & MOTIVATION

Problem

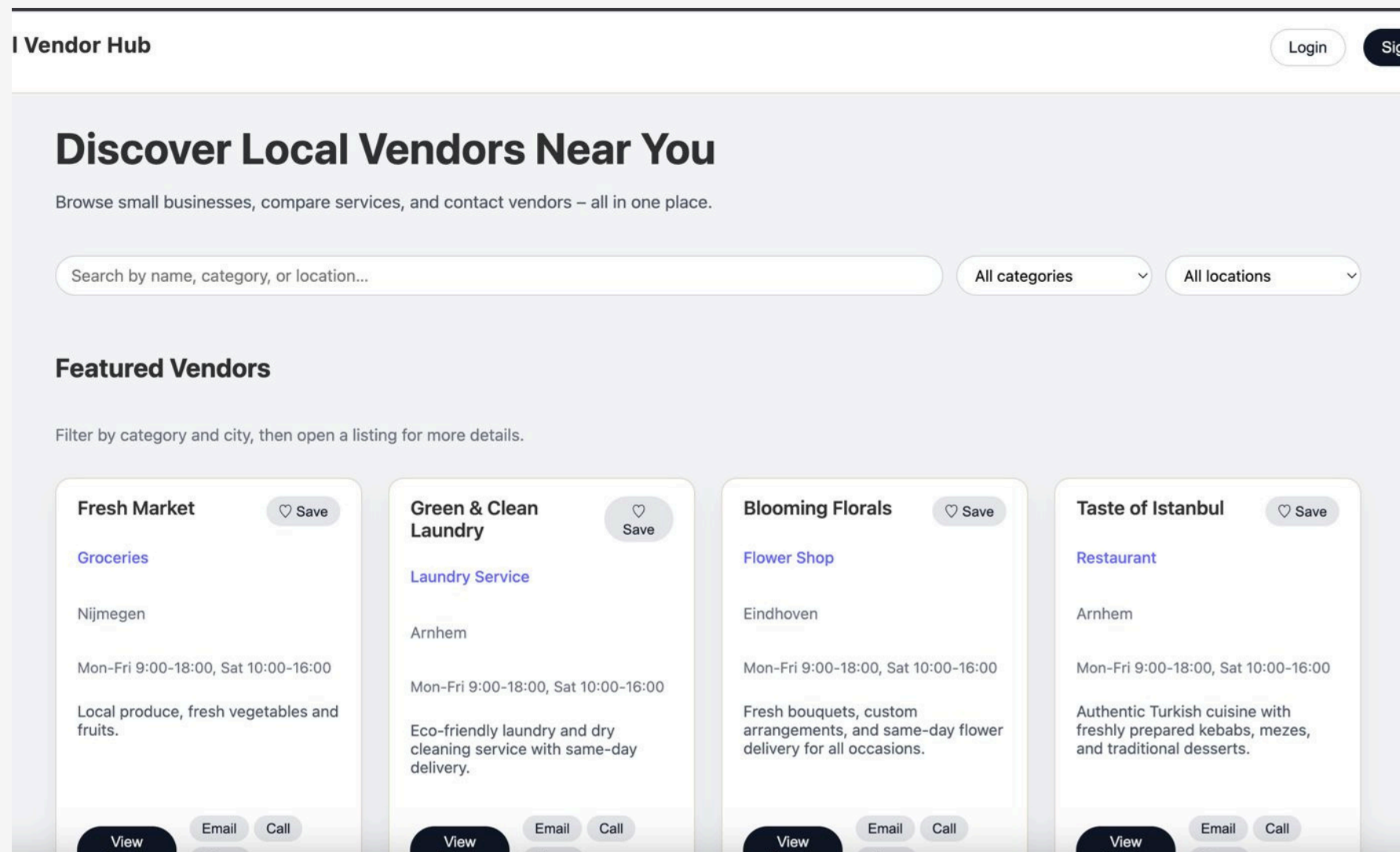
- Local businesses are hard to discover online in a **centralized and trustworthy way**.
- Existing platforms are:
 - Fragmented
 - Poorly moderated
 - Overwhelming for small vendors

Motivation

- Small businesses need visibility
- Customers need **reliable, structured information**
- Cities and communities benefit from accurate local directories



PROJECT GOAL



Conceptual UI mockup of the proposed system

Goal of the system

To build a **web-based directory** that:

- Allows users to **search and browse** local vendors
- Enables vendors to **create and manage listings**
- Ensures quality through **admin moderation**
- Supports **direct communication** between vendors and customers.

STAKEHOLDERS AND ACTIONS

Main Stakeholders

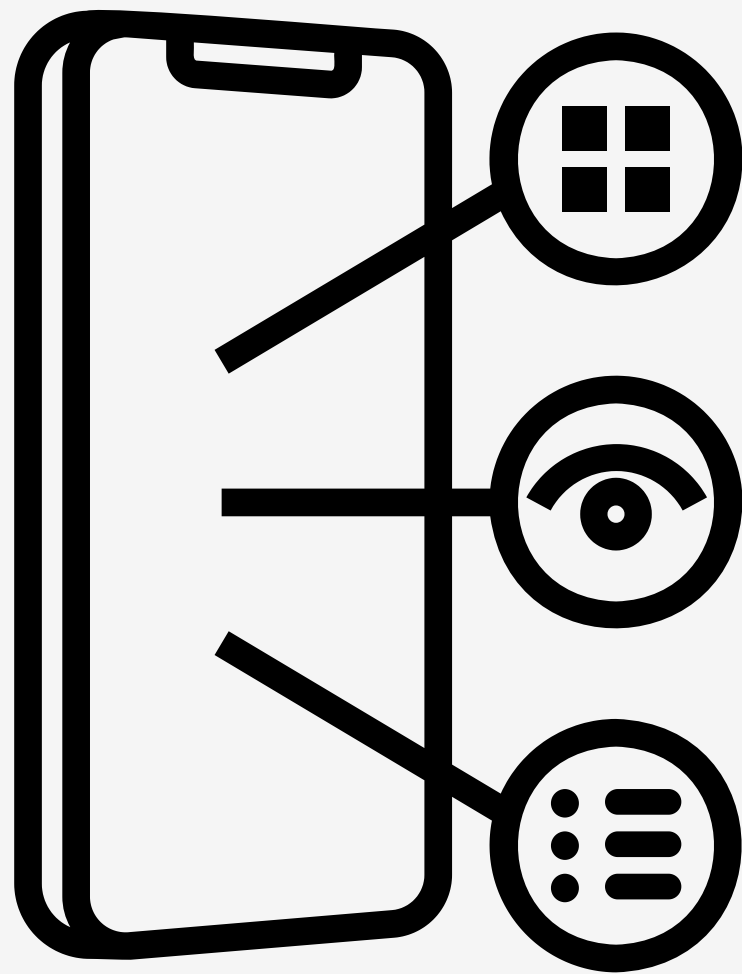
- **Visitor** → browses listings without an account
- **Customer (Registered Users)** → saves favorites and contacts vendors
- **Vendor** → creates and manages business listings.
- **Admin / Moderator** → moderates listings and manages categories

External Actors

- Email service (notifications, verification)
- Authentication system



FUNCTIONAL REQUIREMENTS (MVP)



MVP Functional Requirements

- User registration and login
- Browse and search listings
- Filter listings by category and location
- Vendors can create and edit listings
- Admin approval workflow
- Messaging between users and vendors

Explicitly out of MVP

- Reviews and ratings
- Payments
- Advanced analytics

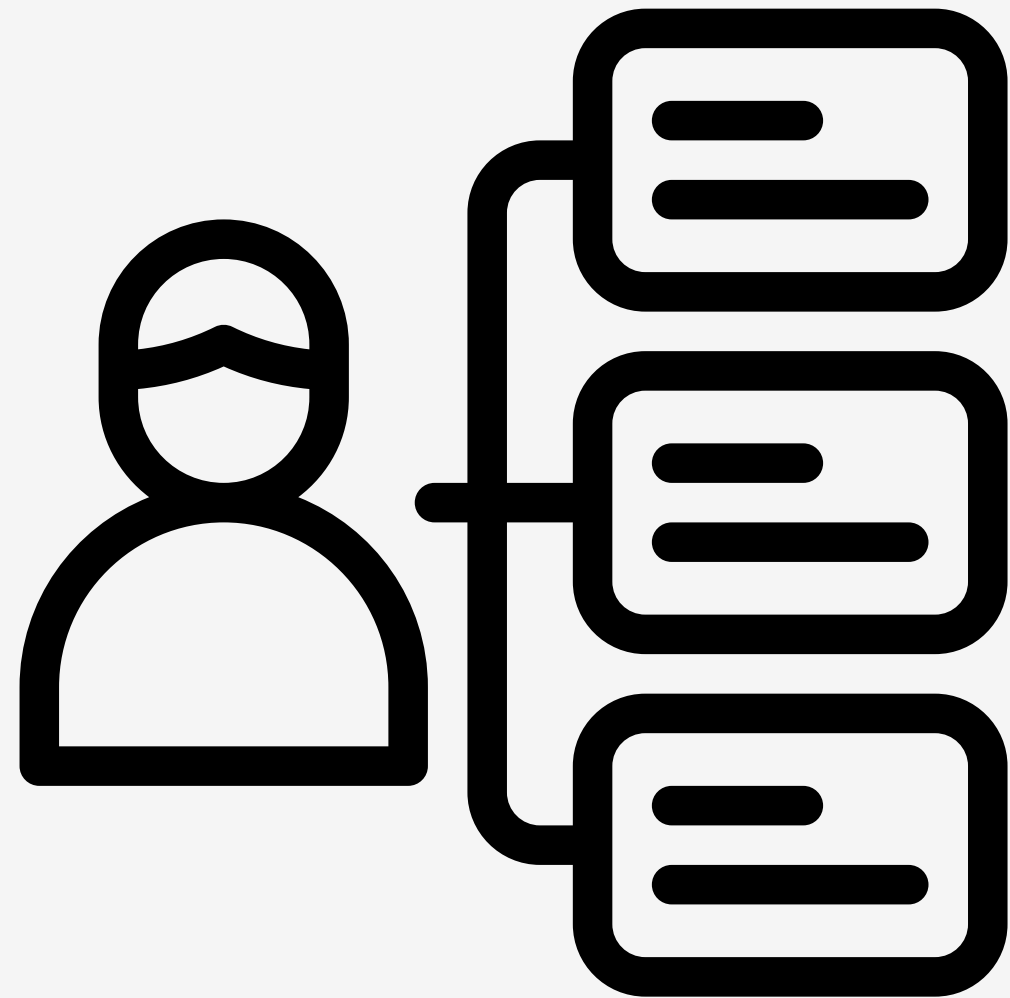
NON-FUNCTIONAL REQUIREMENTS

Quality Requirements

- **Performance:** fast search and page loading
- **Security:** hashed passwords, role-based access
- **Reliability:** stable handling of listings and messages
- **Usability:** clear navigation, responsive UI
- **Accessibility:** usable for user with different needs and Devices



CORE USE CASES

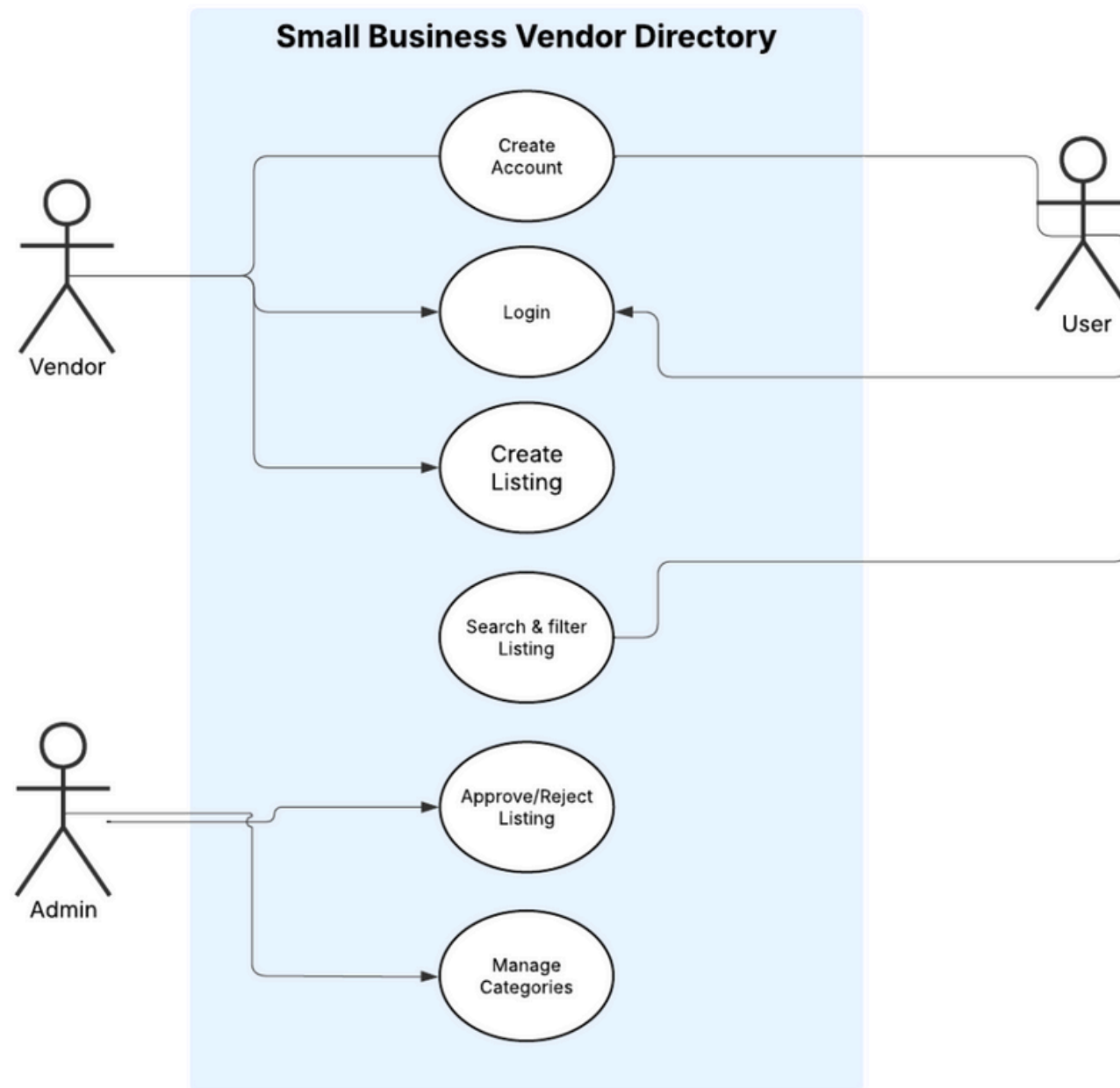


Key Use Cases (MVP)

- Register / Login
- Search and filter listings
- Create vendor listing
- Approve or reject listing (Admin)
- Send and receive messages

These use cases directly implement the MVP functional requirements.

USE CASE DIAGRAM



UML Use Case View

Actors

- Visitor
- Customer
- Vendor
- Admin

Key interactions:

- Account management
- Listing creation and moderation
- Search and browsing
- Messaging between users and vendors

This UML use case diagram captures the **system boundaries and main user interactions.**

DOMAIN MODEL

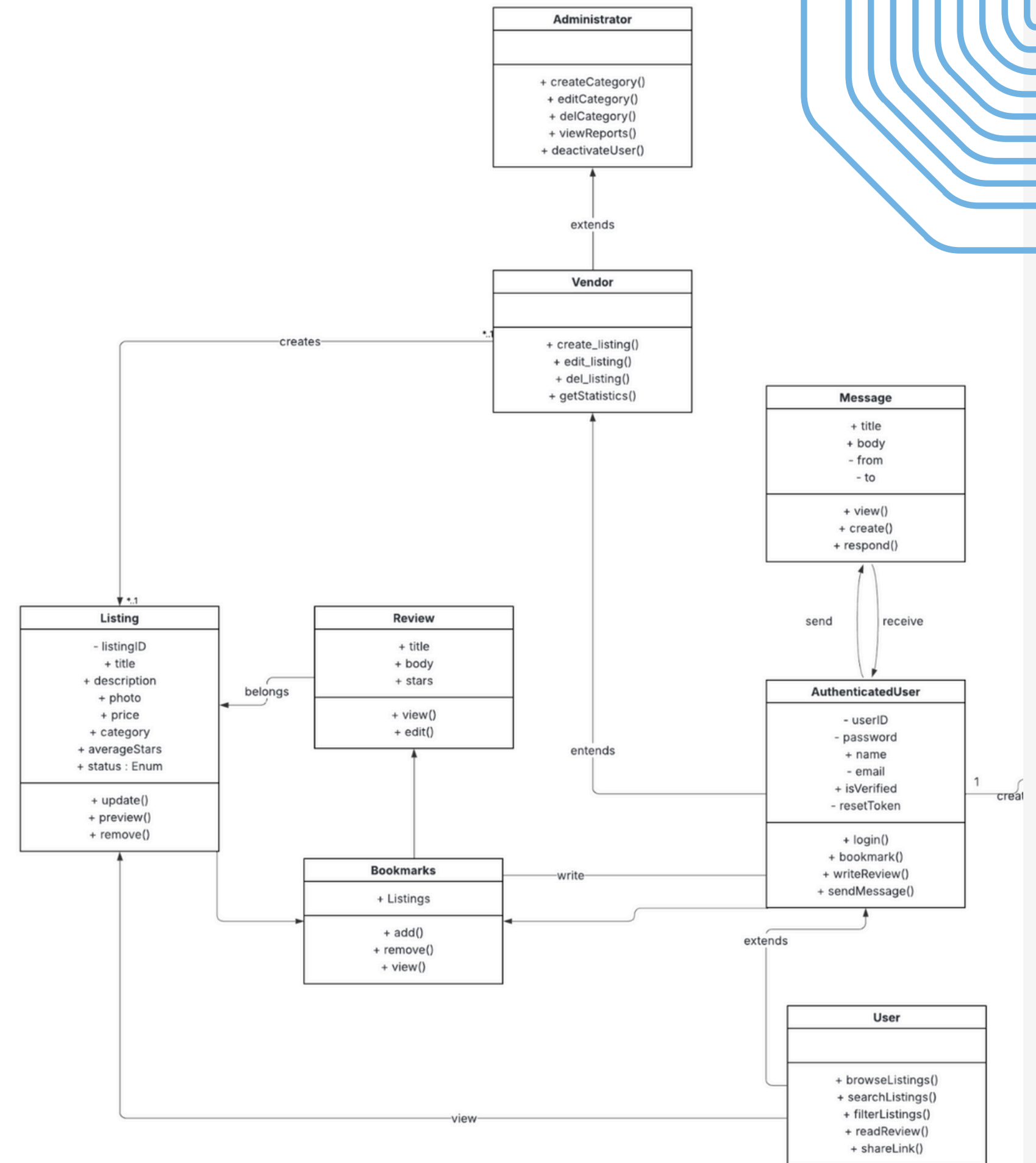
Conceptual Domain Model

Main entities:

- User
- Vendor (inherits from User)
- Listing
- Category
- Message
- Favorites
- Admin Actions

Key Relationships

- Vendor creates multiple listings.
- Listings belong to one or more categories
- Users can favorite listings
- Users and vendors exchange messages
- Admin moderates listings



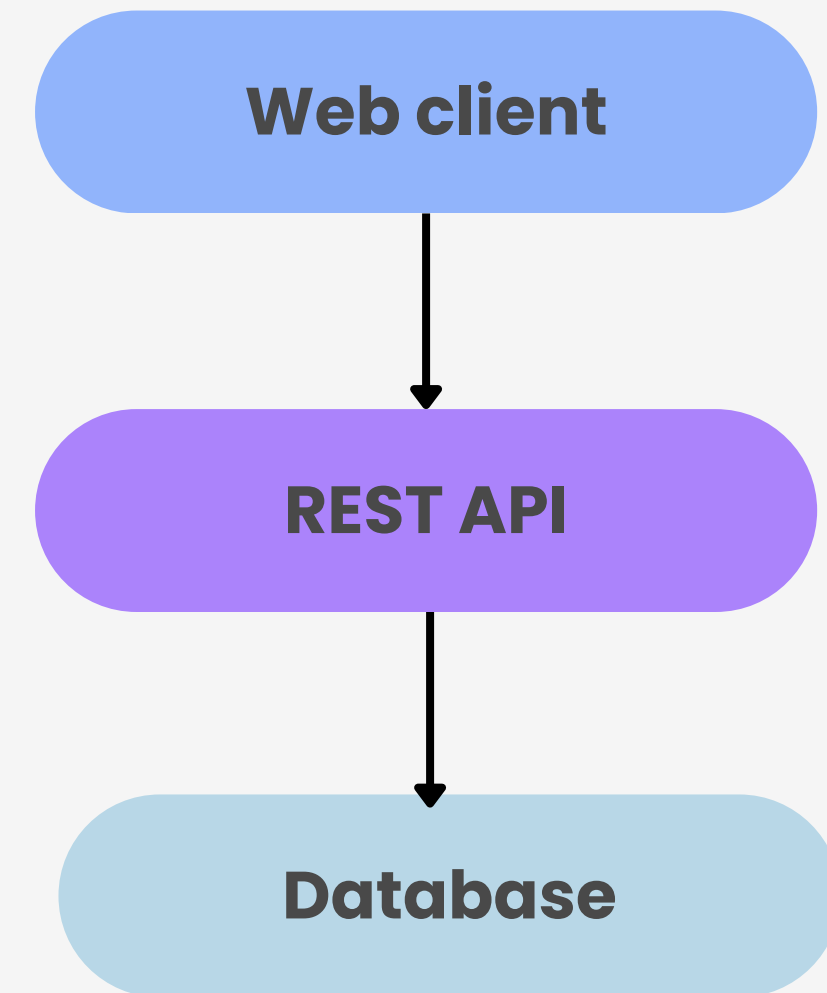
SYSTEM ARCHITECTURE

High-Level Architecture

- **Frontend:** user interface (web client)
- **Backend:** REST API handling business logic
- **Database:** relational data storage

Architecture Style

- Client – Server
- Stateless REST communication
- Clear separation of concerns



INTERNAL DESIGN - BACKEND

Core features delivered

Authentication System

- JWT-based token authentication
- Password hashing with bcrypt
- Role-based access control (customer, vendor, admin)

Business Logic

- **Listings:** CRUD with search/filter capabilities
- **Categories:** Hierarchical organization system
- **Favorites:** User bookmark functionality
- **Messaging:** Customer-to-vendor communication

Status Workflow Management

- Draft → Pending → Approved → Active
- Vendor submission, admin moderation
- Role-specific permissions enforcement

Development Practices

- Testing: 25/25 tests passing (55.36% coverage)
- Documentation: Complete API docs + contribution guidelines
- CI/CD: Automated testing pipeline via GitHub Actions
- Integration: successful frontend-backend data persistence

DATABASE DESIGN

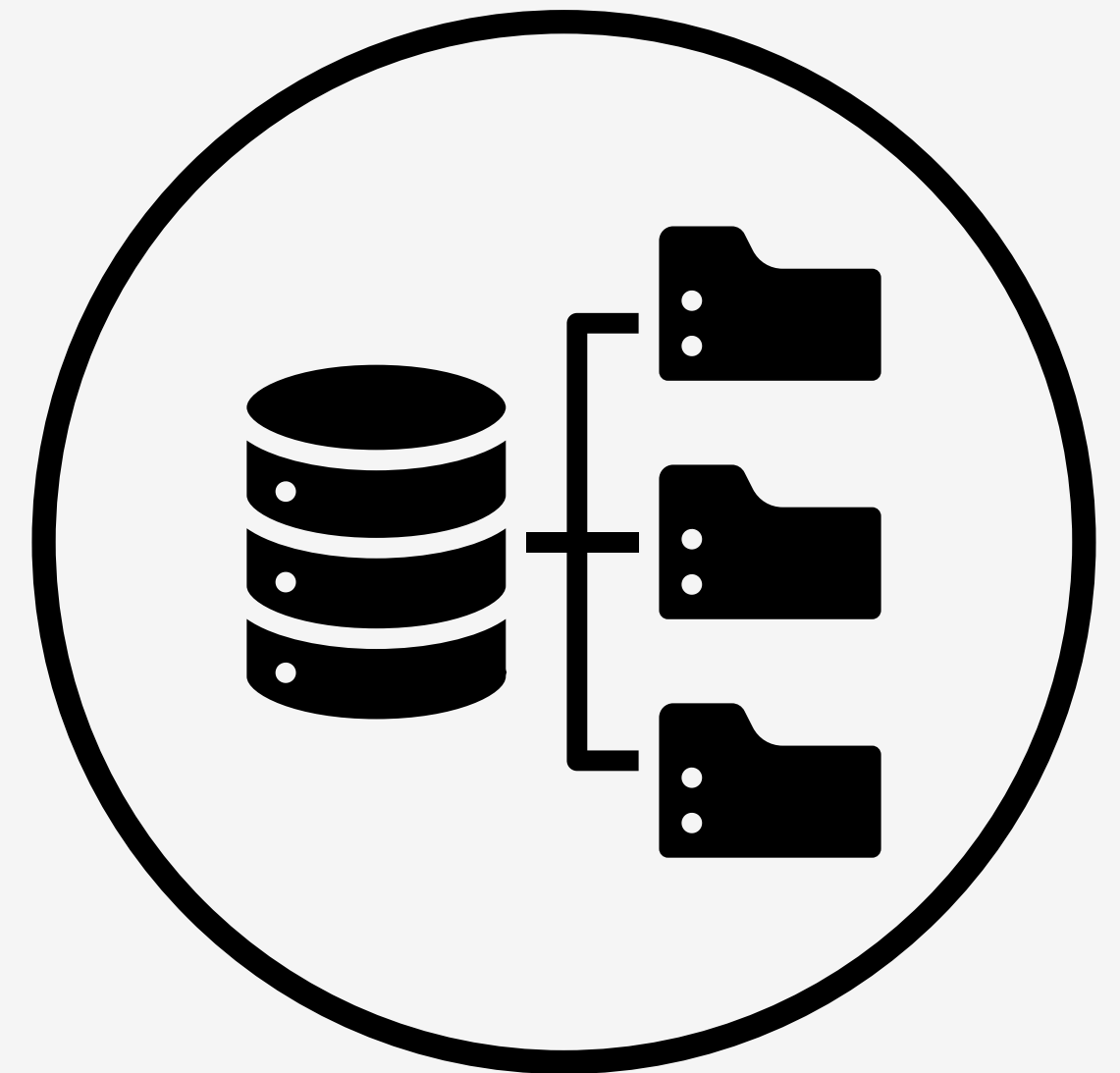
Relational Database Model

Main Tables:

- **Users** — customers, vendors, and administrators (role-based)
- **Listings** — business listings created and managed by vendors
- **Categories** — classification of listings
- **Listing_Categories** — join table for many-to-many relationships
- **Favorites** — user-listing bookmarks
- **Messages** — communication between users and vendors
- **Admin_Actions** — moderation and approval history

Design Principles

- Primary keys for entity identity
- Foreign keys enforcing domain relationships
- Join tables for many-to-many relations
- Referential integrity aligned with the domain model



FRONTEND DESIGN

Main Components:

- Page-based routing for Customer, Vendor, and Admin views
- Reusable components (shared navigation bar)
- TypeScript models for consistent data handling
- Page-level CSS for isolated and maintainable styling

Core UI Features:

- **Landing page** — search and filter vendors
- **Customer pages** — register, login, dashboard, profile, messages, saved vendors
- **Vendor pages** — register, login, dashboard, inbox, listing detail
- **Admin pages** — login and moderation dashboard
- Navigation between all pages

Design Principles

- Clear separation of UI concerns
- Simple, consistent layouts
- User-focused, intuitive interactions

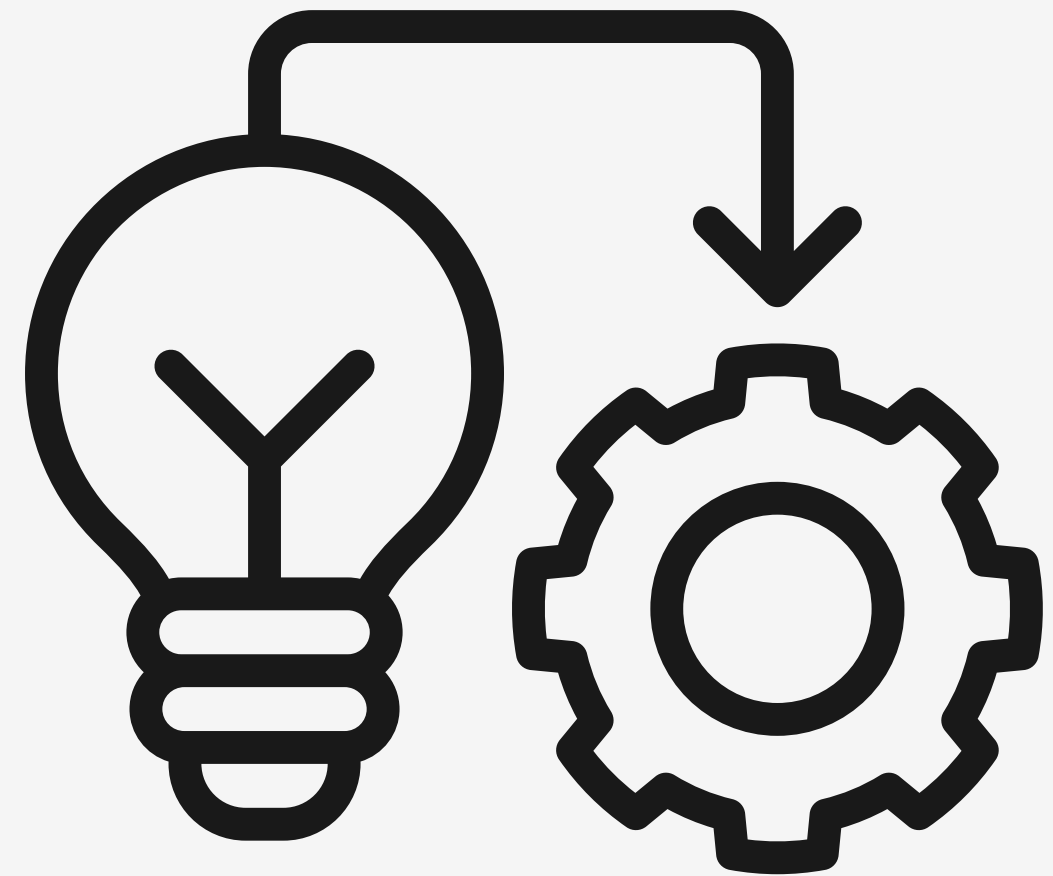


MAPPING DESIGN TO IMPLEMENTATION

From Design to Code

- Use cases → API endpoints
- Domain model → database schema
- Roles → access control logic

This ensures **traceability** from requirements to implementation.



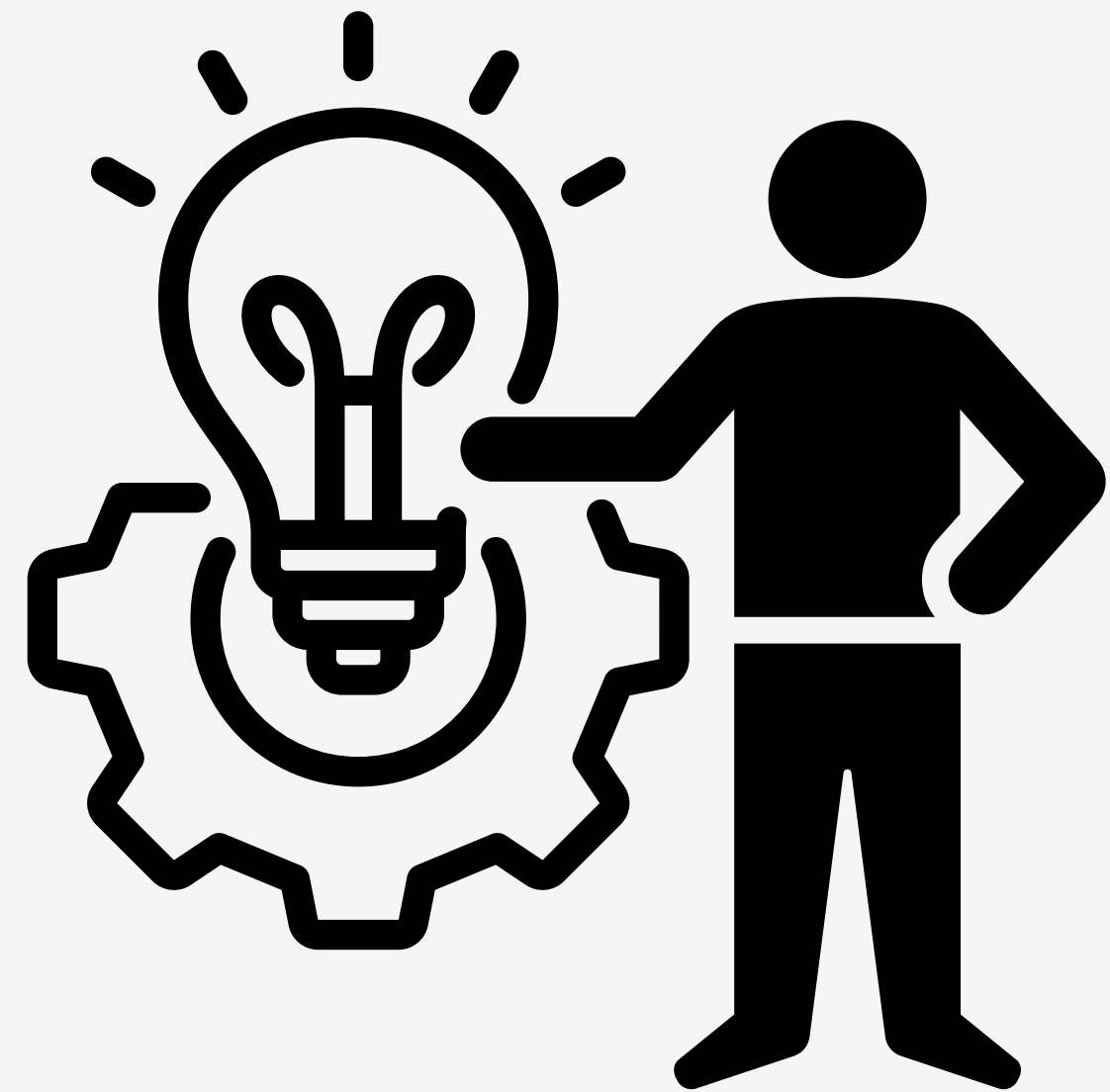
COLLABORATION & ENGINEERING PRACTICES

Team Collaboration

- Shared GitHub repository
- Feature branches
- Pull requests for integration

Software Engineering Practices

- Modular structure
- Clear separation of responsibilities
- Iterative development



OUTLOOK & FUTURE WORK

Possible Extension

- Reviews and ratings
- Vendor analytics dashboards
- Improved search ranking
- Scalability improvements
- Advanced moderation tools

Conclusion

The MVP demonstrates a **complete, working system** aligned with the project requirements.



The image features a light gray background with decorative blue lines in the corners. In the top right and bottom left, there are multiple parallel lines that curve and then extend horizontally or vertically, creating a sense of depth and movement. The text "THANK YOU!" is centered in the middle of the image.

THANK YOU!