Software Engineering II - Workshop 2

Students:

Camilo Londono Moreno

Manuel Felipe Espinosa Espanol

Juan Esteban Ruiz Guasca

Camilo Trujillo Garzon

Cristian Camilo Garcia Palacios

Lecturer:

Carlos Andres Sierra Virguez

Sunday 9 November

Universidad Nacional de Colombia

Facultad de Ingeniería

Departamento de Ingeniería de Sistemas e Industrial

2025

CRC Cards

Account	
Responsibility - Knows if the authentications credentials are valid (email, password). - Knows their state (inactive, active, banned).	**Collaboration** - User - Authenticator - Admin

User	
Responsibility - Knows the public information (name, avatar, biography) - Has one personalized card	**Collaboration** - Account - Labels - ContactCard

Administrator	
Responsibility - Delete inappropriate components or profiles - Change the state of an user (ban, activate)	**Collaboration** - Account - Dashboard

Contact List	
Responsibility - Has a user related URL. - Shows the user the cards of his related users.	**Collaboration** - User - Card

Card	
Responsibility	**Collaboration**

- Has it’s own URL - User-modifiable	- User - Labels

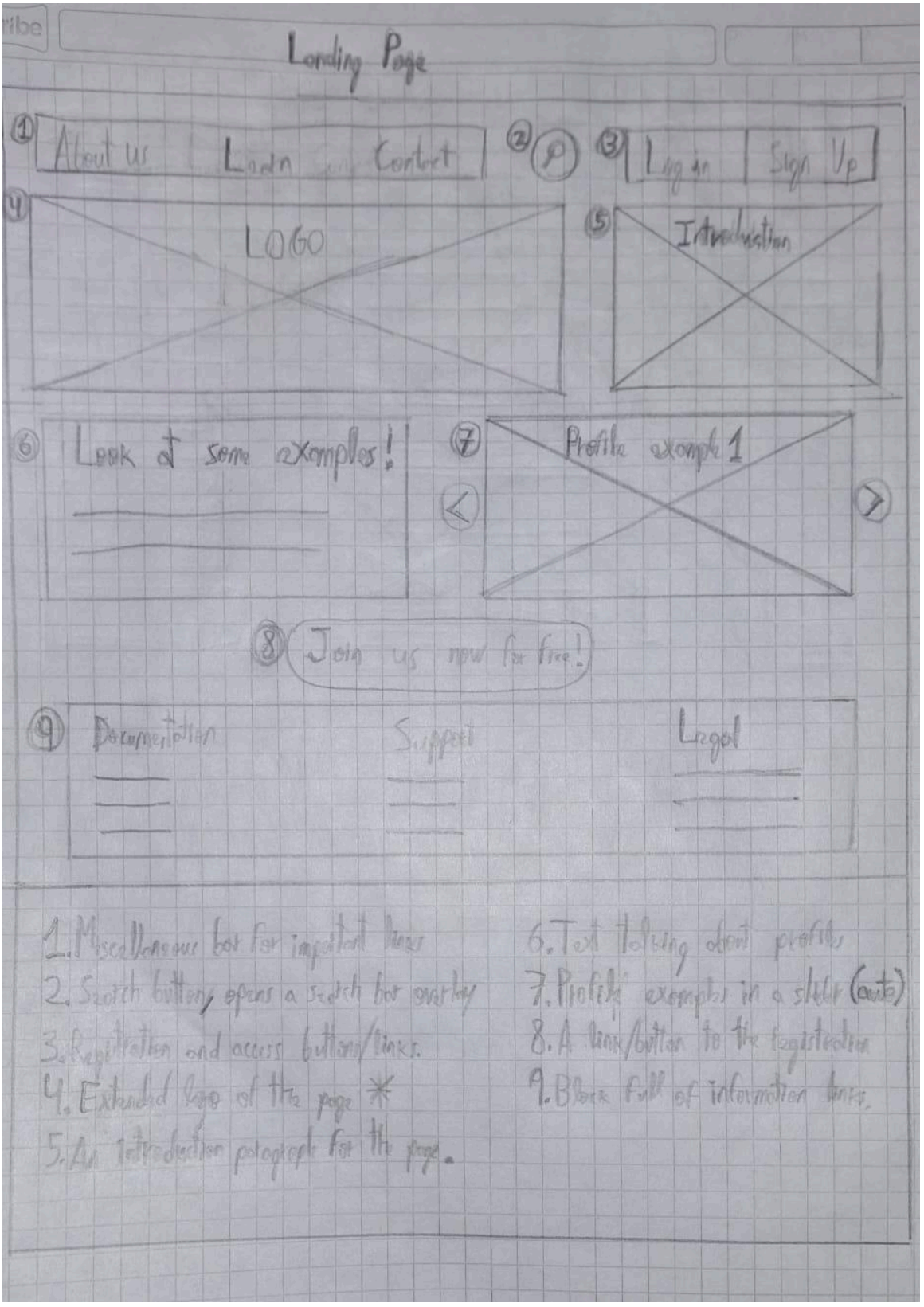
Labels	
Responsibility - Knows its own name - Shows information named by the user about his contacts cards?	**Collaboration** - User - Card

Authenticator	
Responsibility - Connects to the DB for login - Connects to the DB for registering to the app - Creates the users	**Collaboration** - Account

Mockups

Those were made as hand-drawn sketches of low-fidelity wireframes.

First the Landing Page. The purpose of this page is to show any user the basic functionalities of Weaver and how to become a user of the page (and show support links).

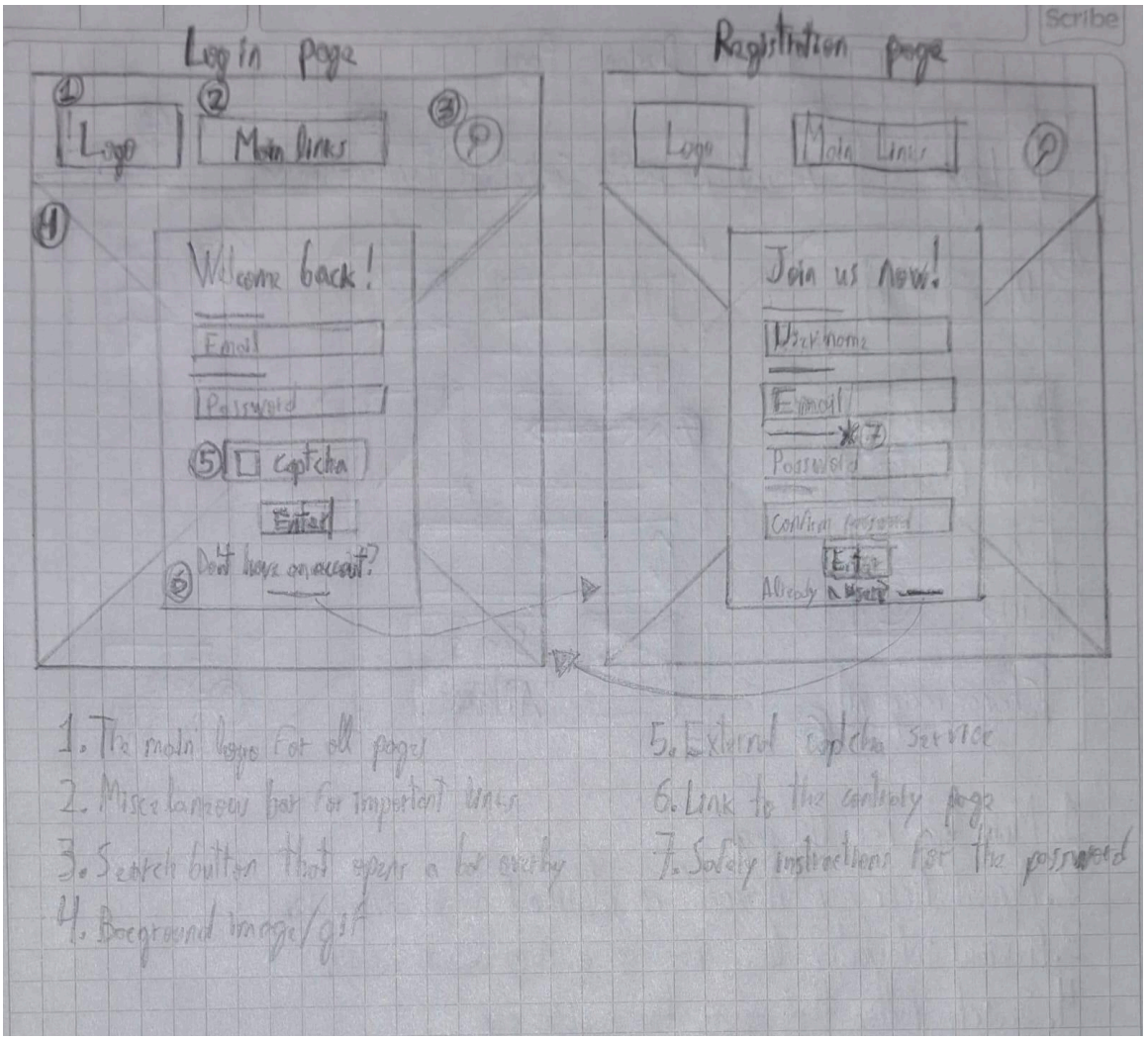


Legend of the Landing page mockup:

1. Miscellaneous bar for important links.
2. Search button, opens a search bar overlay.

3. Registration and access buttons.
4. Extended logo of the page.
5. An introductory paragraph for the page.
6. Text showing off the profiles.
7. Profile examples in a slider (automatic).
8. A button that redirects the user to the registration page.
9. Block full of information (Documentation, support and legal) links.

Second and third pages, Log in and Registration pages. The idea is to show the user how to access his account if he already has one and let him create one with ease in case the user doesn't have one.



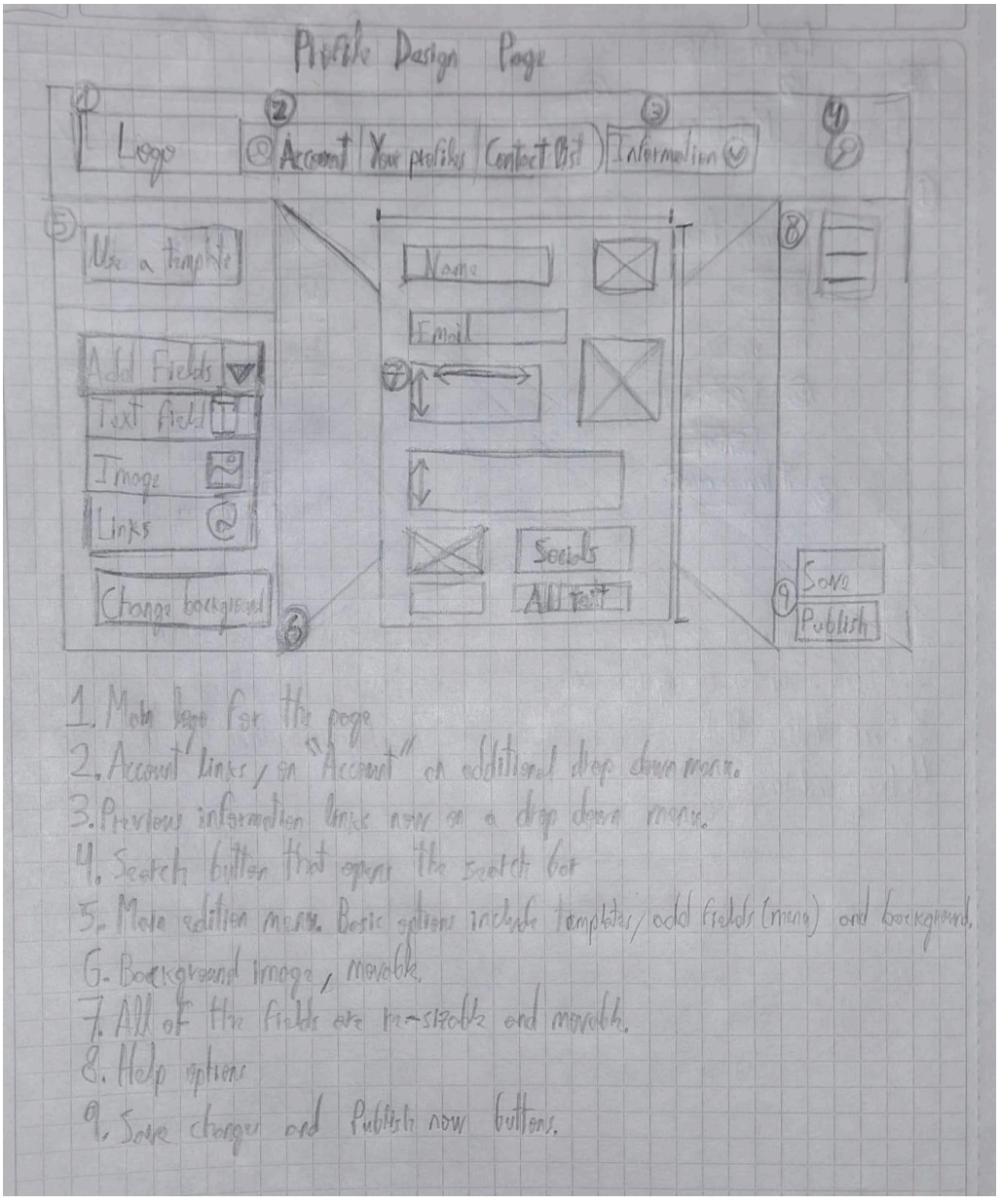
Legend of the Log In and Registration pages mockups:

1. The main logo for all of the pages (except for the landing one).
2. Miscellaneous bar for important links.
3. Search button, opens a search bar overlay.
4. Background image/GIF.

5. External captcha service
6. Link to the contrary page (from log in to sign up and viceversa).
7. Safety instructions for the password creation.

The fourth page, the Profile Design page. A customizable page for the creation of a user profile. The purpose of this page is to let the user design and publish his very own card. The user can check the help options and learn how to create a profile (card) page according to his wishes. We give him a button to add multiple types of fields, another with a few templates and one to manage the background.

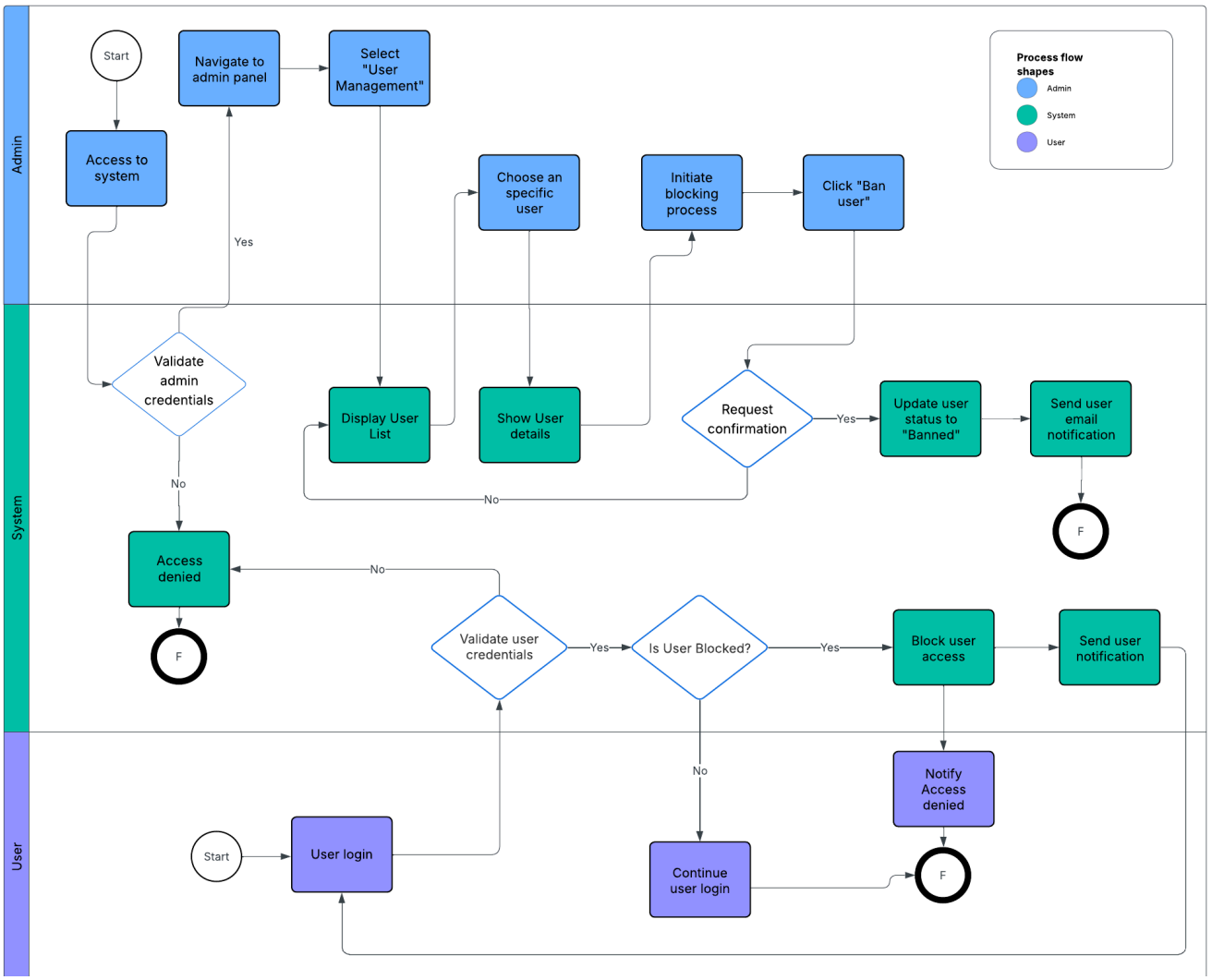
After the user finishes he can save the changes on the design or publish the card for anyone to see on the platform.



Legend for the Design Profile Page mockup:

1. Main logo for the page.
2. Account links, the “Account” button has an additional drop down menu.
3. Previous information links now on a drop down menu.
4. Search button that opens the search bar overlay.
5. Main edition button. Basic options include templates, add fields (menu) and change background.
6. Background image, movable.
7. All of the fields are movable and re-sizable.
8. Help options menu.
9. Save Changes and Publish now buttons.

Business Model Processes



Process: User Blocking by Administrator

This process describes the systematic suspension of a user's account by an administrator when the user violates platform rules or terms of service. Ensuring moderated, secure, and appropriate use of the application while maintaining audit trails and user notification.

Role in the Application

1. Platform Moderation & Safety

Maintains a safe environment for all users by removing harmful or rule-breaking users from active participation. This is to prevent spam, harassment, and inappropriate content dissemination.

2. Administrative Control

Let and provide tools for monitoring and controlling user behavior to administrators. And enable quick response to reported issues or violations reported by the community.

3. Compliance & Accountability

Ensures adherence to platform guidelines and creates documented evidence of administrative actions, according transparency and audit requirements

4. User Experience Protection

Preserves quality of service for legitimate users and isolates problematic accounts before they affect others to maintain platform reputation and user trust.

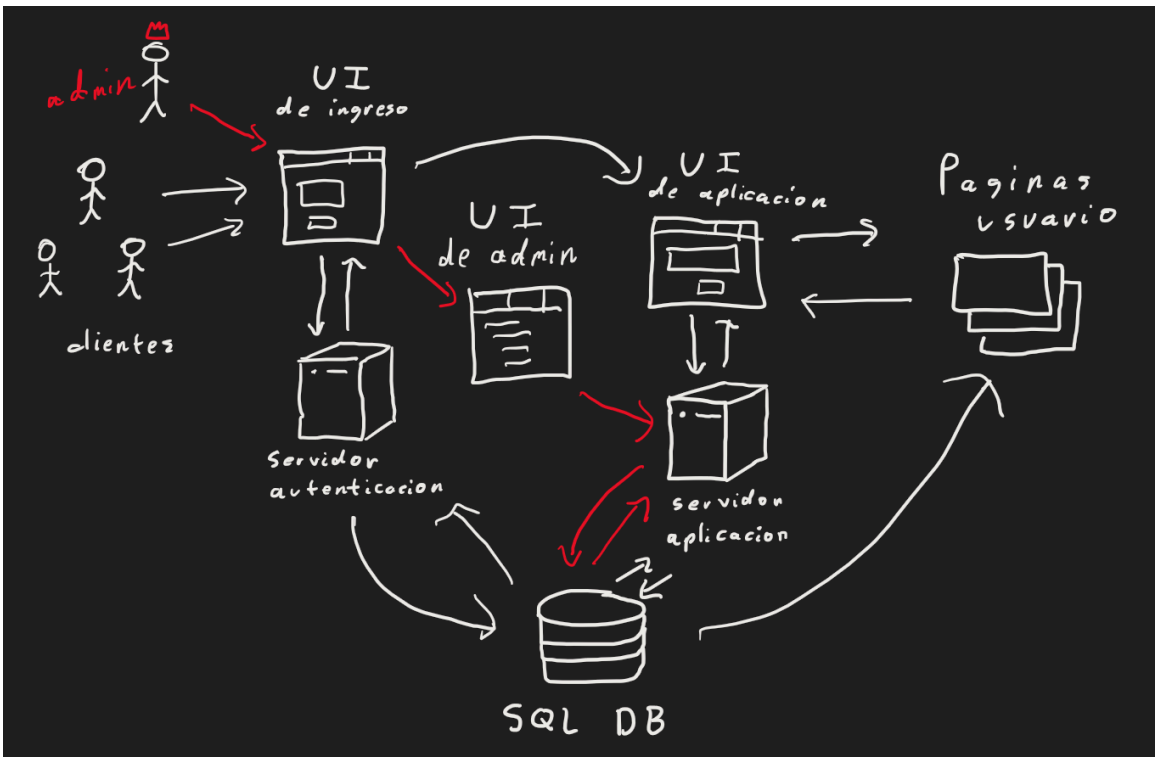
Business Value

For the platform, this user blocking process delivers critical business value by mitigating legal and reputational risks associated with harmful user behavior. Ensures the ongoing maintenance of community standards and quality, and safeguarding the application's integrity. Furthermore, it provides the necessary operational control and oversight mechanisms for administrators to effectively manage the user base and ensure smooth platform functioning.

For the users, the process is fundamental in creating a safe and protected environment for interaction, free from spam, harassment, and other forms of misconduct. The visible and active moderation demonstrated by this system is essential for building and maintaining user trust. It also provides a clear

mechanism for conflict resolution, reassuring users that the platform is committed to upholding a positive and respectful community.

Architecture Diagram



Our diagram illustrates a classic architecture with centralized authentication and two separate interfaces. Clients and administrators first access the login UI. This UI communicates with the authentication server, which validates credentials, issues a session token, and queries/updates the SQL database for users and permissions.

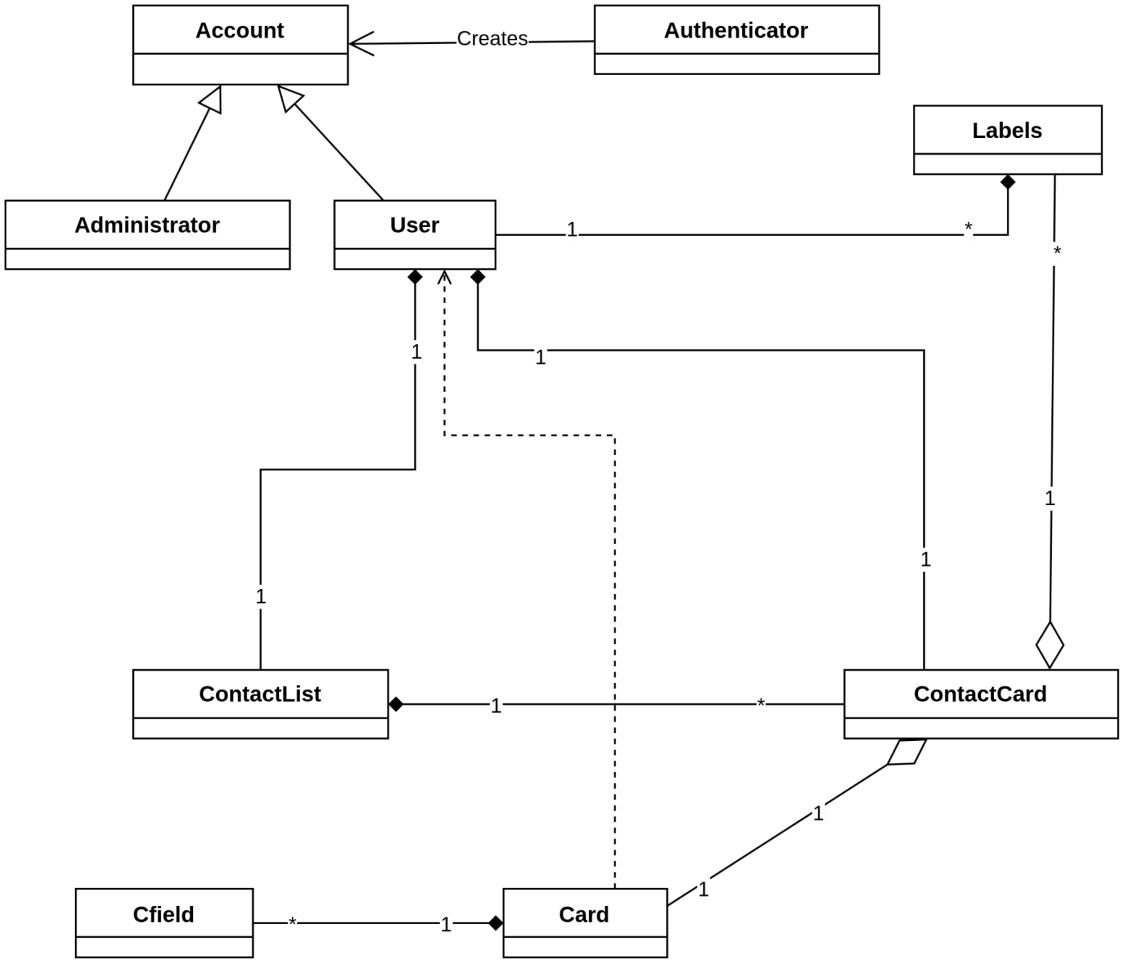
Once granted access, the administrator enters the admin UI, from where they perform management operations. These actions are sent to the application server via secure APIs, and the application server persists the changes to the database (red arrows highlight sensitive administrative operations).

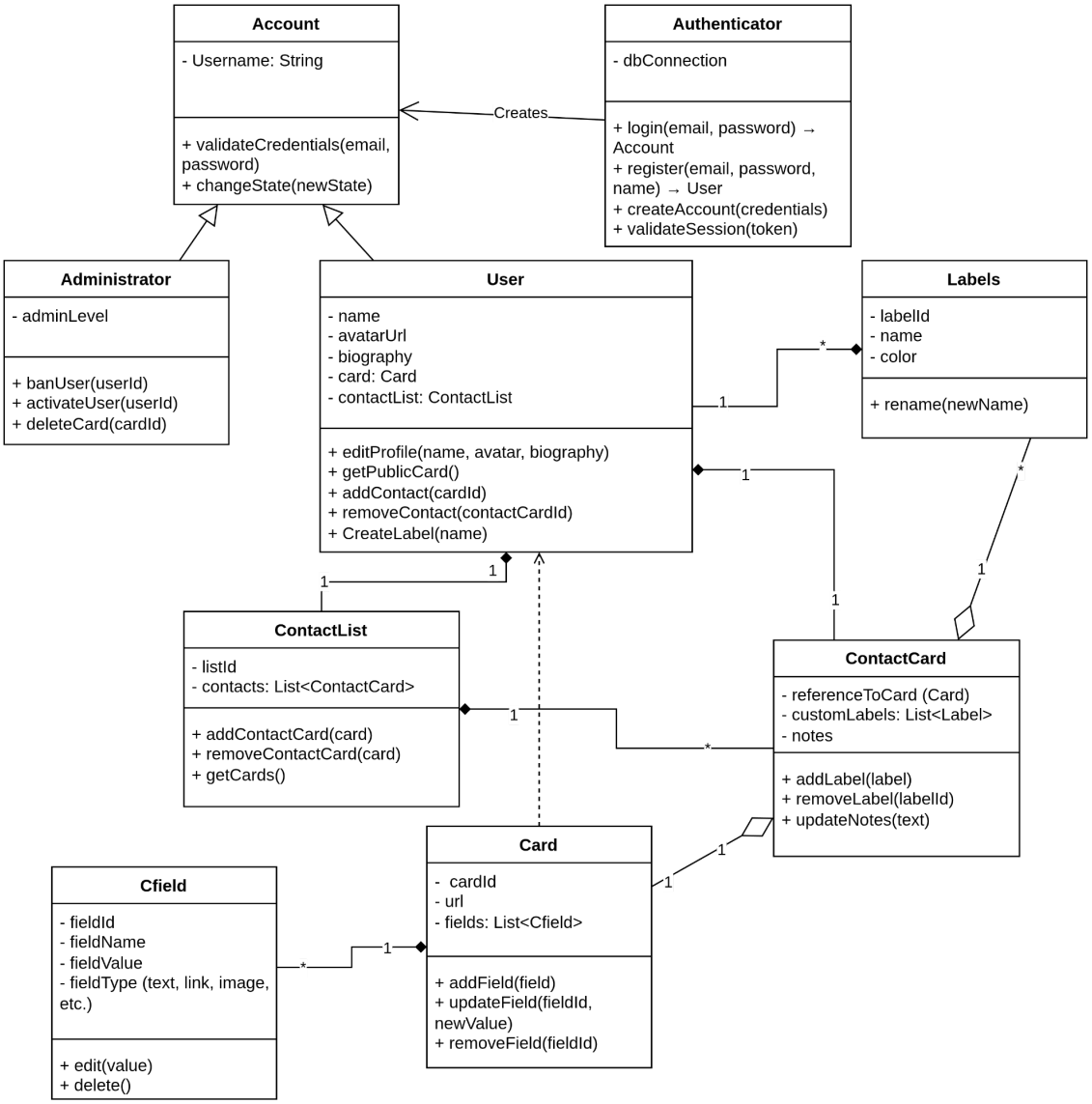
Clients navigate the application UI, which communicates with the application server to execute business logic and build user pages.

Class Diagram

The class diagram is implemented trying to have the minimal use of inheritance, for this it connects the main classes using associations

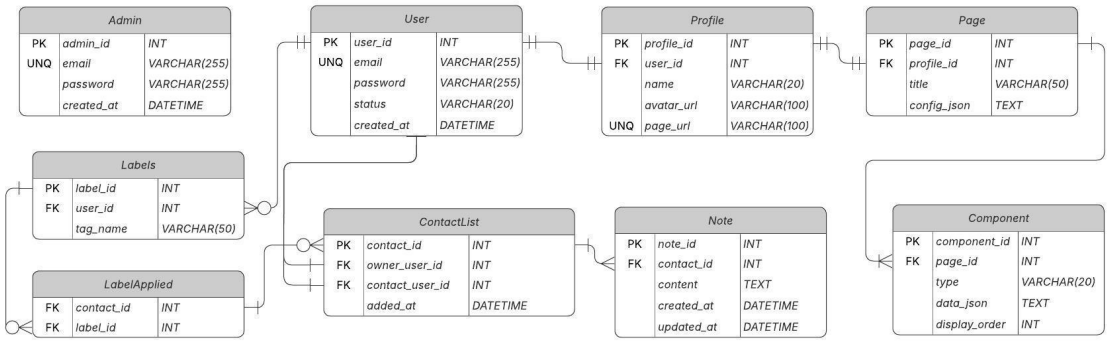
The model works like this: each Account is created through the Authenticator, and it can represent either a regular User or an Administrator, both inheriting the basic account data. Every User owns exactly one ContactList (composition), meaning the list only exists while the user exists. That ContactList contains multiple ContactCard entries (composition), and each entry represents another user's Card stored in the list. ContactCard only holds a reference to the target user's Card, it does not own it. The Card itself owns its Cfield elements (composition), which define the public profile data of that user. Additionally, ContactCard can be linked to Labels, but this is just an association, not ownership. The solid diamonds show ownership and lifecycle dependency: deleting the parent automatically deletes the contained elements.





Relational Database Model

This is the ER diagram of the application. Each user has a profile with a webpage linked to it. The page is made up of different parts, such as links, text, and other content blocks. Users can also save other users as contacts in the ContactList table and add private notes and Labels to them.



The fields config_json and data_json let us store information without changing the database, only the fields. The config_json field stores general settings and how the user's page looks, like the background color or font. The data_json field, in the Component table, stores the content of each component block. Each type of component needs different data, so this field keeps it flexible.

- Example for a link: { "url": "https://google.com", "label": "Website" }
- Example for text: { "text": "Welcome to my page", "alignment": "center" }

Bibliography

[1] M. Galvan, "20 Wireframe Examples for Web Design," *Flux-academy.com*, 2022. https://www.flux-academy.com/blog/20-wireframe-examples-for-web-design

[2] Visual Paradigm, "What is BPMN? [Quick Guide]," *Visual-paradigm.com*, 2010. https://www.visual-paradigm.com/guide/bpmn/what-is-bpmn/

[3] "Business Process Modeling Notation | BPMN Tutorial and Templates," *Lucidchart*, 2025. https://www.lucidchart.com/pages/tutorial/bpmn

[4] "Entity Relationship Diagram (ERD) Tutorial - Part 1," *www.youtube.com*, Oct. 27, 2023. https://www.youtube.com/watch?v=xsg9BDiwiJE

[5] "Entity Relationship Diagram (ERD) Tutorial - Part 2," *www.youtube.com*, Nov. 11, 2023. https://www.youtube.com/watch?v=hktyW5Lp0Vo

[6] "Tutorial de estructura y diseño de bases de datos," *Lucidchart*. https://www.lucidchart.com/pages/es/tutorial-de-estructura-y-diseno-de-bases-de-datos

[7] GeeksforGeeks, “How to Design a Database for Content Management System (CMS),” *GeeksforGeeks*, Mar. 05, 2024. https://www.geeksforgeeks.org/sql/how-to-design-a-database-for-content-management-system-cms/ (accessed Nov. 10, 2025).