Student Trade

Information System for Selling and Trading items among Students

(Project for Systems III)

Hristijan Chupetreski

Definition of problem:

It's no secret that students aren't the wealthiest creatures walking the planet — that's just a fact. They often move for the sake of their future: switching faculties, relocating to college, or even moving abroad. With all this change comes a wave of challenges that build up over time. Relocating to a new apartment or dorm means dealing with a ton of unknowns:

- Is it far from campus?
- Is it furnished?
- Will I be living alone or with roommates?
- Is the kitchen usable?
- Is there enough space and storage?
- And more importantly: How do I afford what I need to live decently?

The typical student budget doesn't leave much room for expensive solutions. New environments can feel like a challenge, new languages add a barrier to it, and studies multiply the stress daily. Everything combined, It's overwhelming — and I've felt it myself. On top of that, students are often forced to rely on social media or online marketplaces that are flooded with scams, bots, and overpriced listings. Dishonesty and digital noise make it hard to know what's real and sometimes, in a rush, quick decisions are made that can cost time, money, or both. Hence, you become the victim of the system. So, what if we could change that? The system? What if there was a solution that was simpler, safer, more student-friendly — a platform made specifically for people in that situation? One that helps us buy, sell, and trade what we need, in a community we can trust? That's the solution I want to build — from one student to another.

Functional requirements:

1. **User Registration & Authentication**
    - Students can sign up and log in using a verified student email (e.g. a university domain like @university.edu).
    - Secure login System (with password reset options).
2. **Profile Management**
    - Users can create and manage their personal profiles, including photographs, name, university, location, and list of posted items.
    - Users will be able to view other users' profiles to see general information as well as ratings and trades/purchases they've made.
3. **Item Listing System**
    - Users can create listings by uploading photos, setting item name, price and a short description.
    - Option for picking whether the item is for sale only or available for trade.
4. **In-app Messaging**
    - Secure chat feature between users to negotiate, ask questions, or arrange pickups.
    - Chat only opens once an interest is shown for an item of another user.
    - Chat closes by choice of both parties whether they want to close it or not.
5. **User Ratings & Reviews**
    - After an interaction/trade, users can rate each other and leave short reviews to build community trust.
6. **Admin/Moderator Dashboard**
    - Admins can monitor reported content, suspend suspicious accounts, and manage categories.
7. **Notifications & Alerts**
    - Real-time notifications for messages when users show interest in some item.

Non-functional requirements:

1. **Performance requirements:**
    - The system should support at least 500 simultaneous users performing actions without any delays and we expect to handle around more than 1000 active daily users in future version and upgrades.
    - The system should allow a maximum of 1 second for a normal load.
2. **Information requirements:**
    - Listing and messaging with no lagging and fast delivery and response time.
    - In case verified and accepted, the system works with university email ensuring safety and eligibility in every account.
3. **Budget and timetable – economy requirements:**
    - System uses open-source framework and free cloud infrastructure
    - Optimises storage for images and listing by reducing quality and size
    - System aims to be free and accessible for students
4. **Control requirements:**
    - Verification must be done for account to be created
    - All posted content must follow community standards. Harmful, offensive, or misleading content can be reported by users and will be reviewed and removed by moderators (predefined users) if it violates the rules
5. **Service and system distribution:**
    - The system is expected to maintain 24/7 uptime, ensuring continuous availability.
    - Constant health monitoring and upgrades after deployment
    - The system runs on React.js for frontend development and Node.js with Express for backend routing, server-side logic and APIs.
    - The system also runs with Firebase Firestore for cloud, storage, media handling and secure logins.

Feasibility Study

In order to correctly and safely present our study on why this project could benefit the students, we will deduct our feasibility study in 4 states. The technical, operational, economic and organizational feasibility. This System provides a scope of solutions to the many situations students struggle with and our main focus will be covering everything they might not be able to cover for themselves.

- **Technical Feasibility**
  Selected technologies such as React and Node.js will be used upon creating this project to maintain scalability, modern structure and will allow future low maintenance. Along with these two, Firebase will play a huge role in storage and authentication for our system to reduce complexity and offer stability. The system's core operations and services—including search, user verification, and the reporting system—are designed to be manageable and maintainable by a small development team.
- **Operational Feasibility**
  From the start we require authenticated and valid registration. A university-issued email will be the only accepted email for creating a personal account. Hence, we offer a safe environment and a well-known one. Students will be surrounded by other students in which they trust. Upon this the system will also run a safe and non-harmful content and listings of only acceptable items. In a case where students do not comply with these terms, other users will have the power to report such situations that violate the code of conduct.
- **Economic Feasibility**
  The deployment and maintenance of the system will be done using free-tier or low-cost cloud services with the focus of reducing and cutting any costs for students. In the event of significant growth and increased user activity, monetization may be introduced to support a larger team and more advanced services.
- **Organizational Feasibility**
  By bringing this system into the student's everyday life, we would allow a safer, better and easier start for the students. Their relocation and the switch they experience in their life would start with the system we offer and an environment where students feel comfortable being surrounded by their peers. Offering stress-free adaptation, lowering waste for our environment and popularizing the culture of sharing and giving.

Logical Design

Table 1: Matrix user role/functions

	User	Administrator
Create an account	Yes	Yes
Delete an account	No	Yes
Log in	Yes	Yes
Edit Profile	Yes	Yes
Post Listing Item	Yes	Yes
Start a chat with user	Yes	Yes
Delete chat	No	Yes
Report Listing	Yes	Yes
Edit/Delete Listing Item	Yes	Yes
Leave a review	Yes	Yes
Edit/Delete Review	Yes	Yes
View Reports	No	Yes
Send Messages	Yes	Yes
Remove Messages	No	Yes

Figure Model I: Matrix User Role/Functions

Table 2 : Data Dictionary

Entity	Description	Attribute	Type	Description of att.
User	The user using the system	id	int	ID of user
		fullname	varchar(255)	Full name of user
		bio	text	Description of user
		email	varchar(255)	Login credentials of users account
		password	varchar(255)	
		profile_picture	varchar(255)	Personal photo of user
Review	Short ratings for users	id	int	Number ID of review
		owner_id	int	ID of user who gets the review
		reviewer_id	int	ID of user who gives the review
		description	text	Description for the review of the user
		date	date	Date of review made
Listing Item		id	int	Number ID of item

	The products users list for sell or trade	owner_id	int	Number ID of user
		date	date	Date of item posted
		title	varchar(255)	Name of item
		description	text	Short description for the item
		price	float	Price of a Listed Item
Report	Reported problem or violation of code with content	id	int	Number ID of report
		owner_id	int	Number ID of owner of reported item
		type	varchar(255)	Type of violation
		date	date	Date of report submitted
		item_id	int	Number ID of reported item
Chat	Communication place between two users	id	int	Number ID of chat
		sender_id	int	Number ID of first user in chat
		receiver_id	int	Number ID of other user in chat
		date	date	Date of chat established
Messages	Text or image content sent and received by users	id	int	Number ID of message
		chat_id	int	Number ID of chat
		sender_id	int	Number ID of user sending the message
		receiver_id	int	Number ID of user receiving the message
		sent_at	datetime	Date of message sent
		content	text	Content of message

Figure Model II: Data Dictionary

Entity Relational Diagram (EDR)

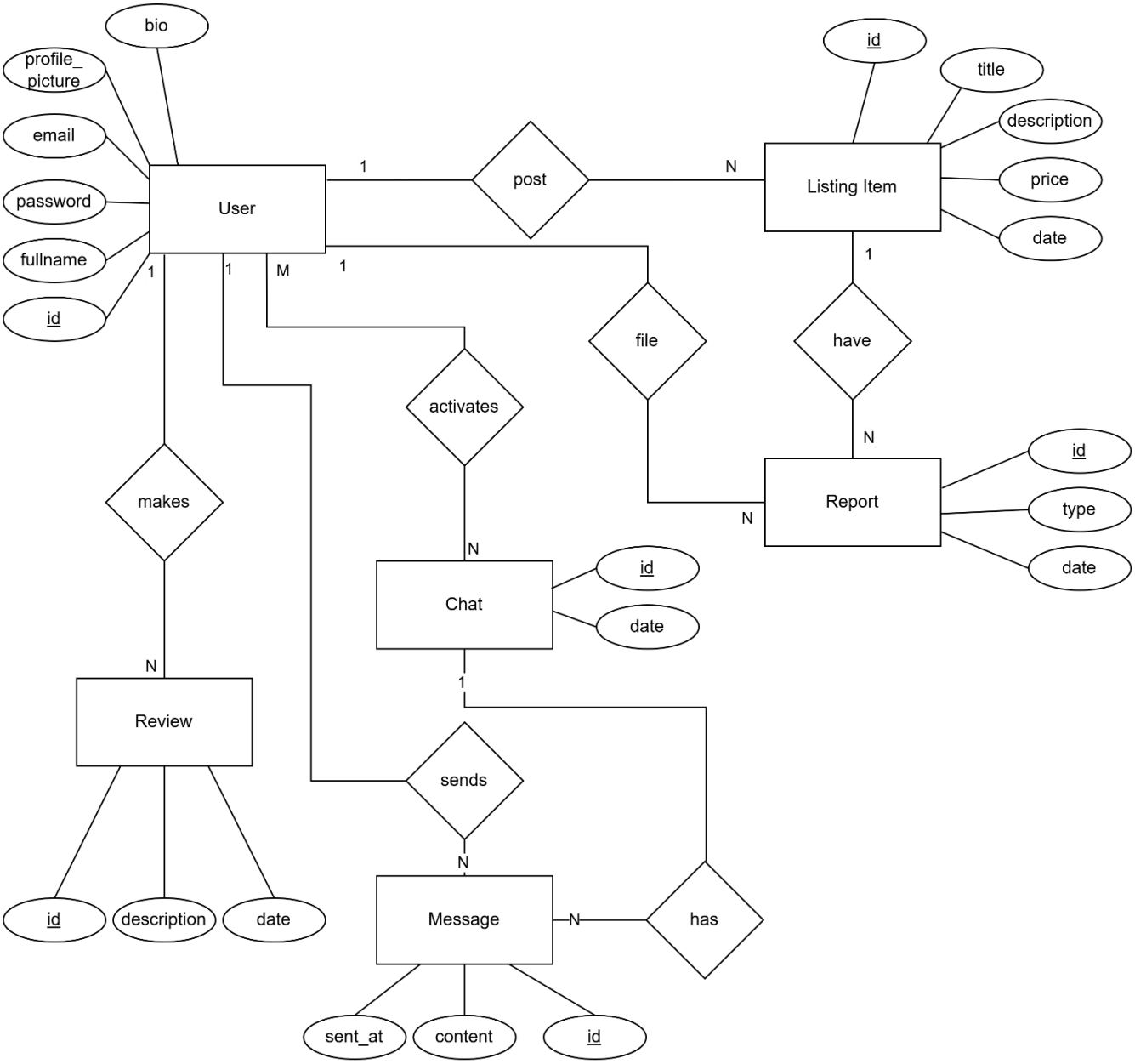

Figure Model III: Entity Relational Diagram

Relational Model

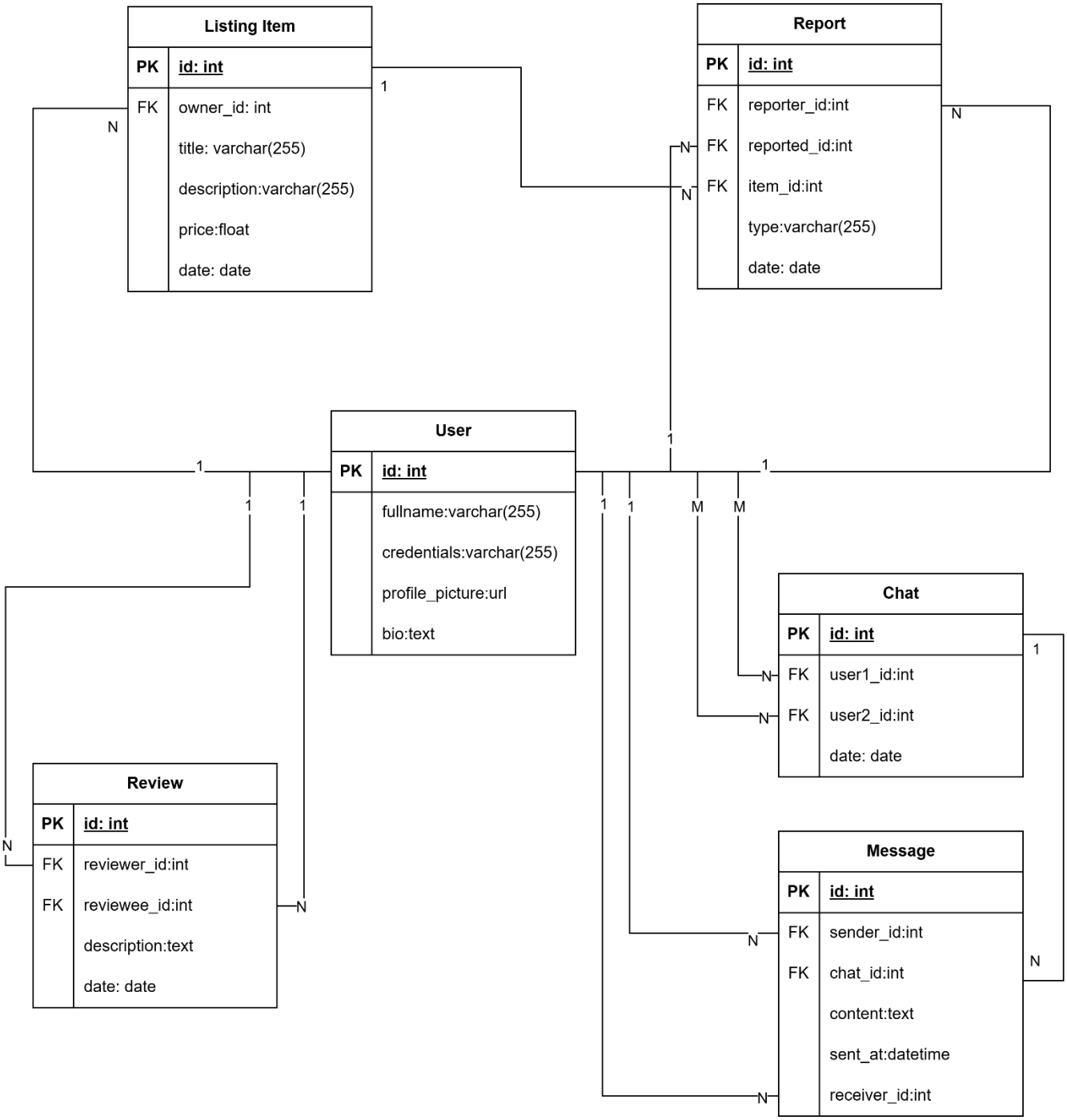

Figure Model IV: Relational Model

Functional Decomposition Data Model

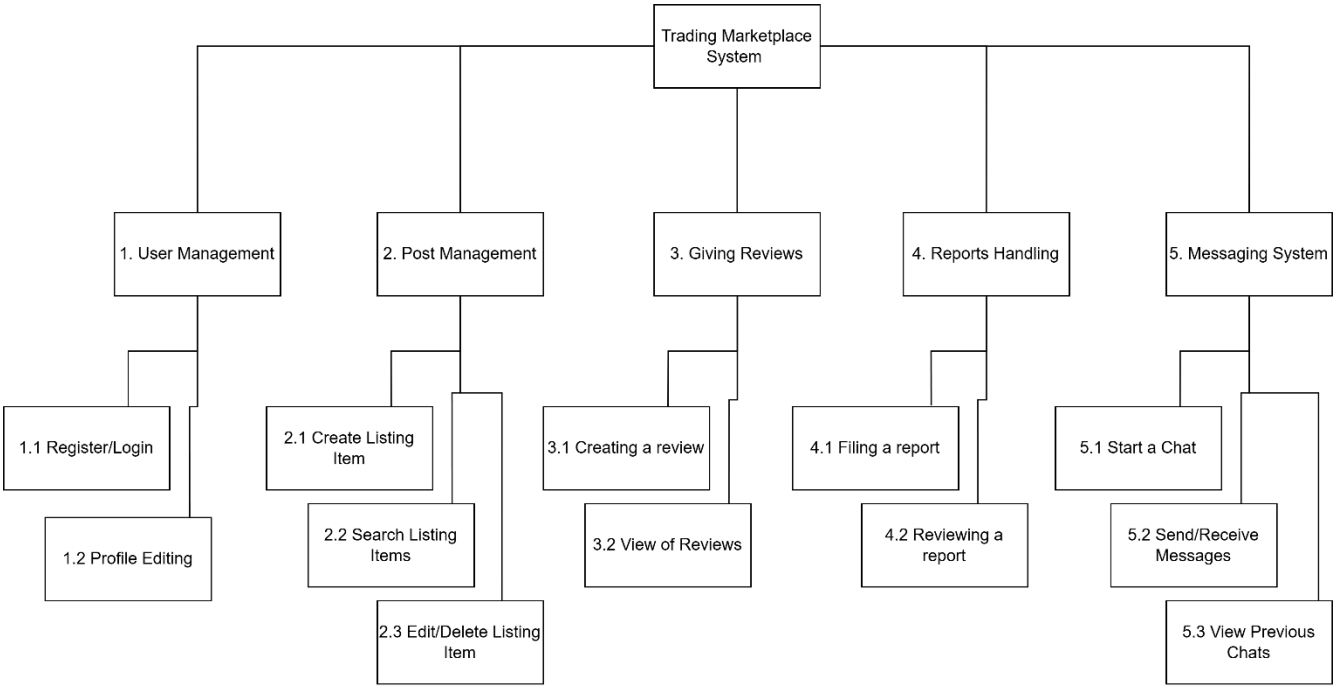

Figure Model V: Functional Decomposition Data Model

Physical Design Phase

Physical Data Model

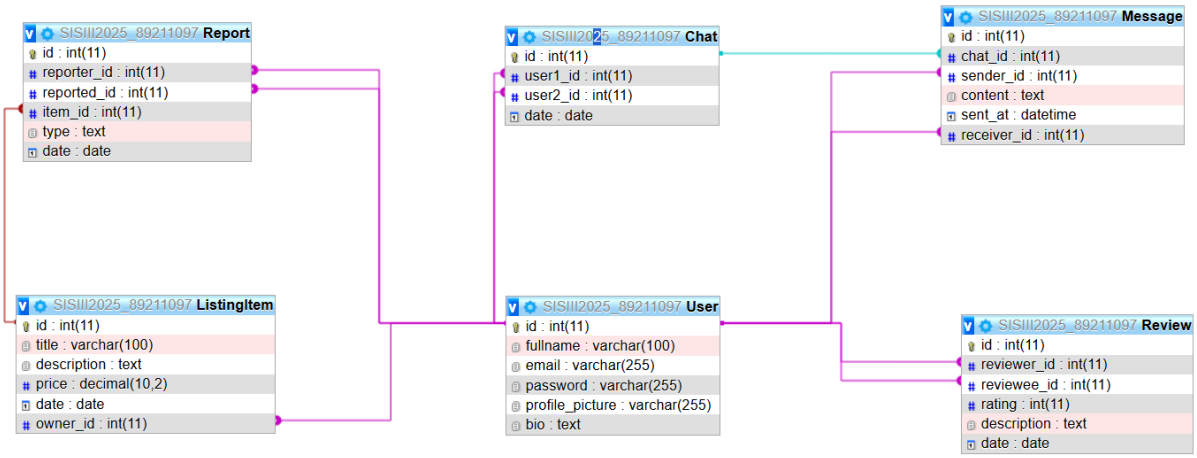

Figure Model VI: Physical Database Model

Object-Oriented Analysis

UML Class Diagram

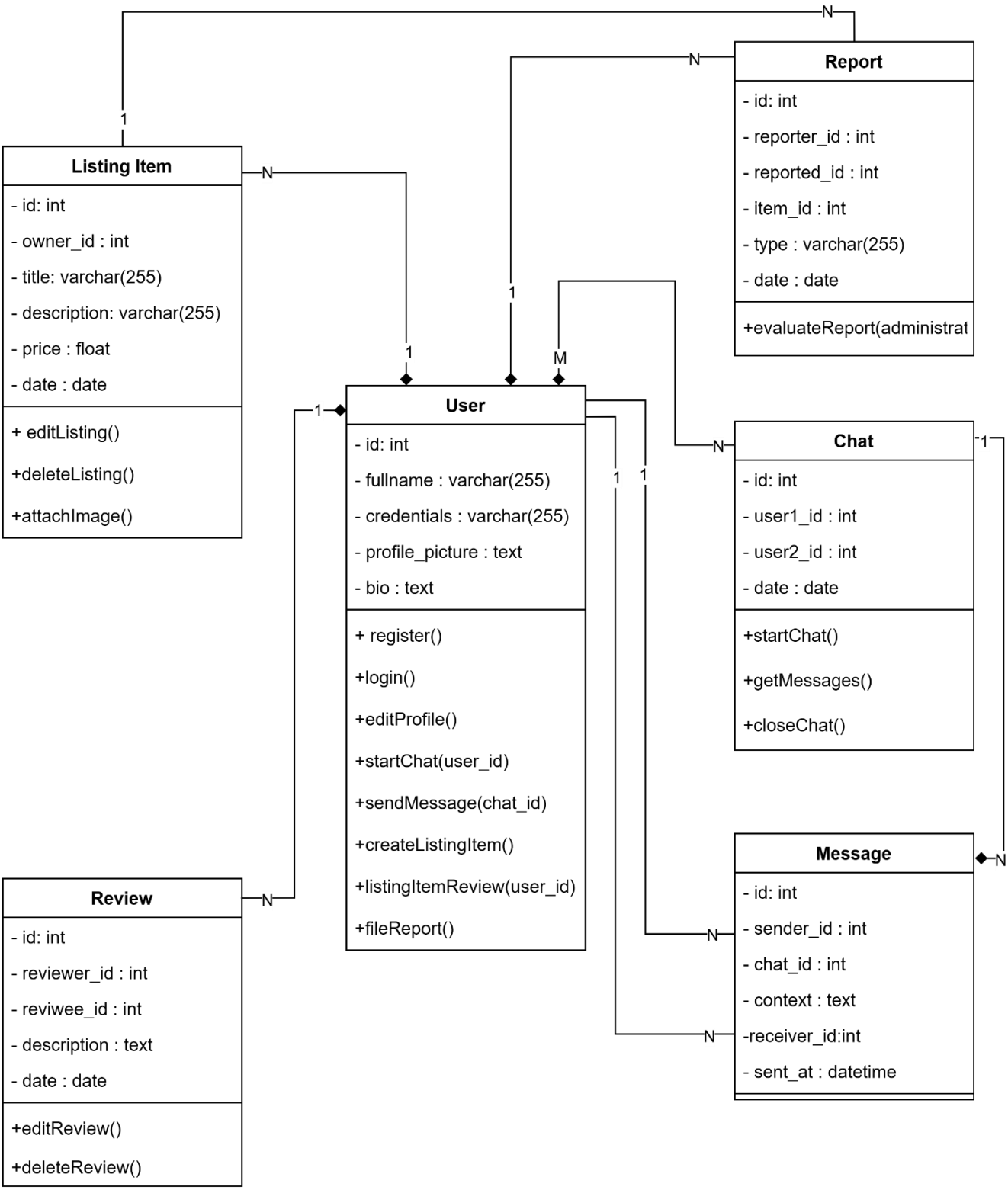

Figure Model VII: UML Class Diagram

UML Sequence Diagrams

1. Workflow between User and System – Logging in and Posting a Listing Item

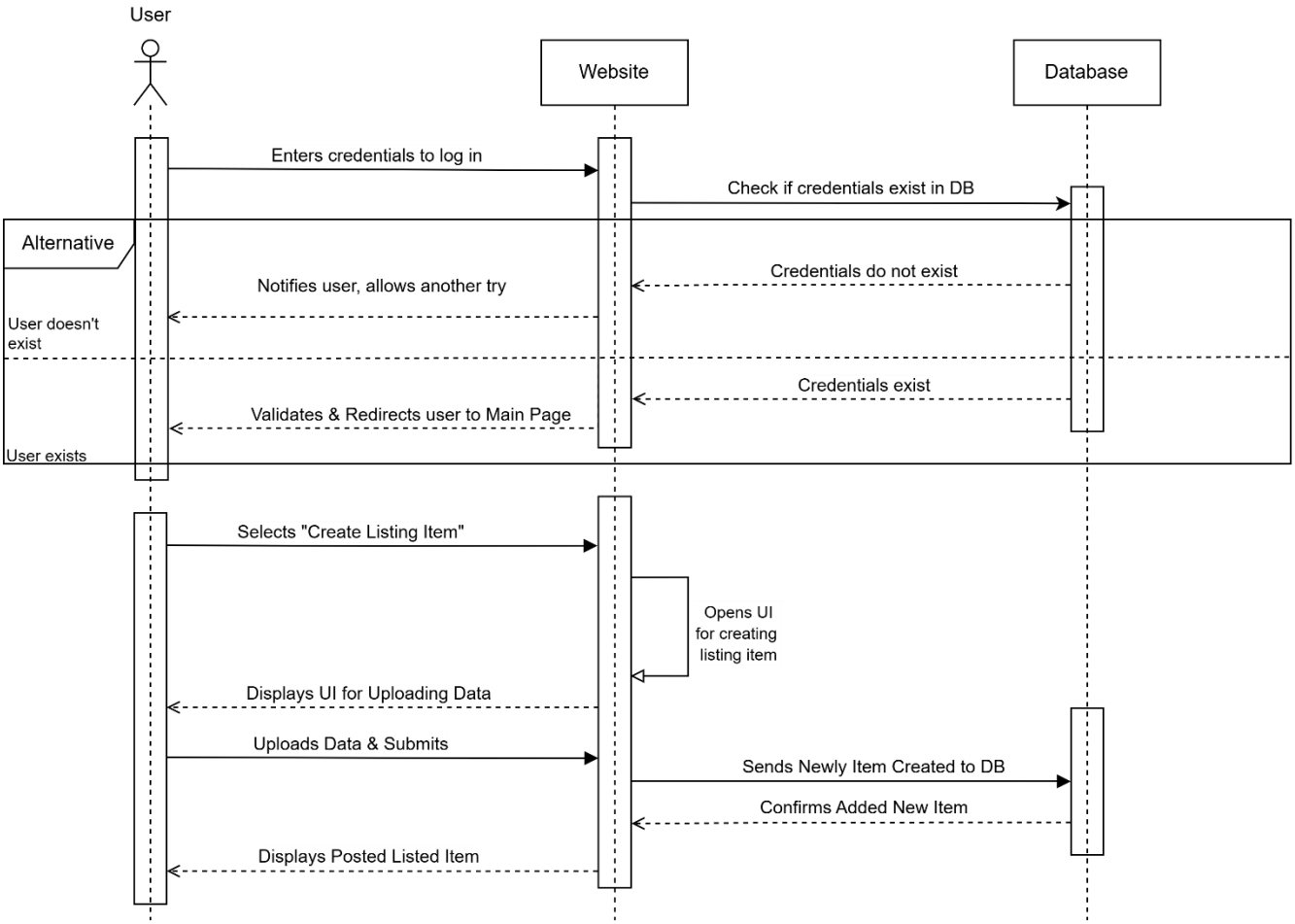

Figure Model VIII: UML Sequnce Diagram 1.0 (User – System Interaction)

2. Workflow between two Users – Interaction with Chat and Giving a Review

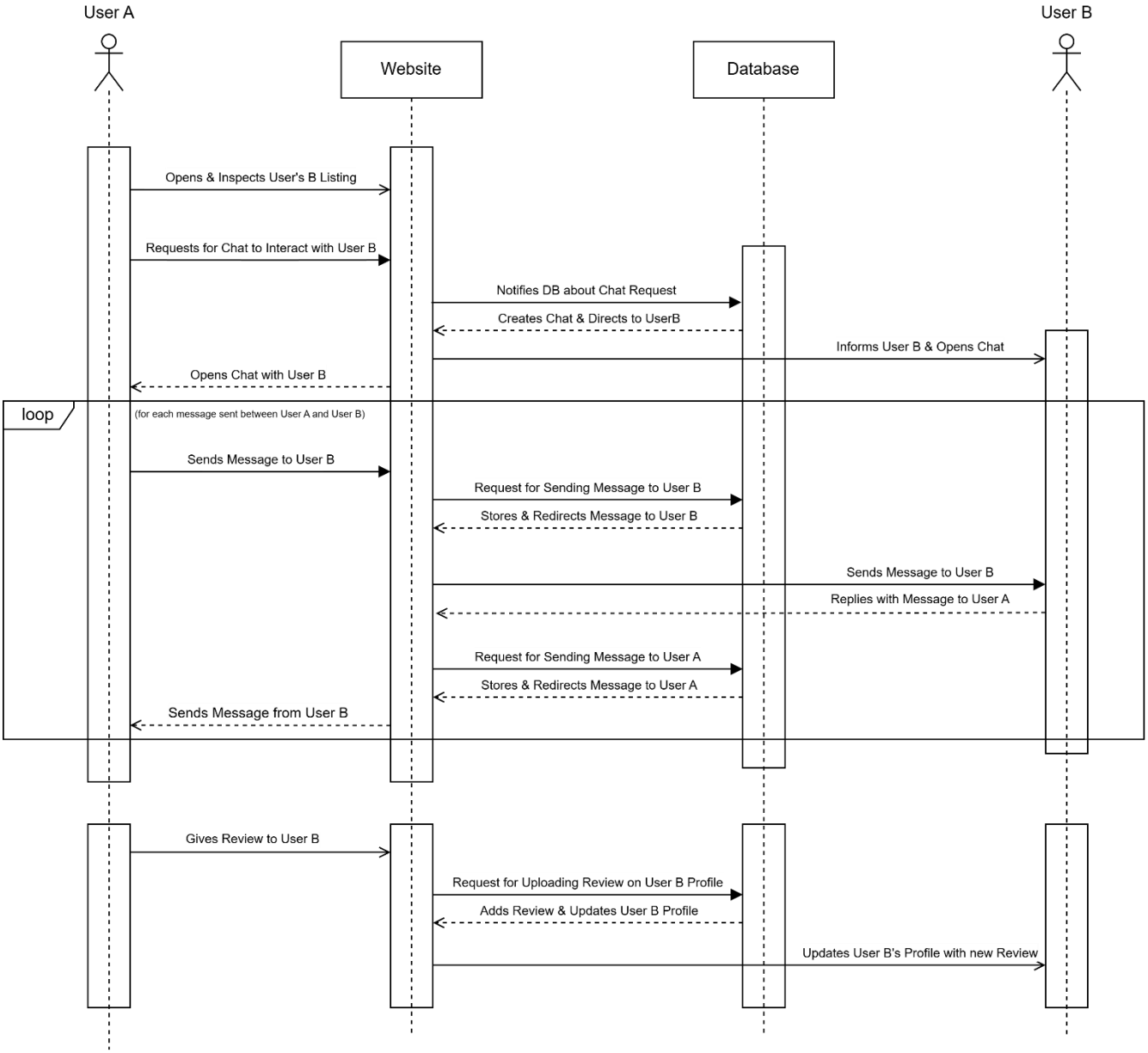

Figure Model IX: UML Sequnce Diagram 2.0 (User – User Interaction)

UML Use-case Diagram

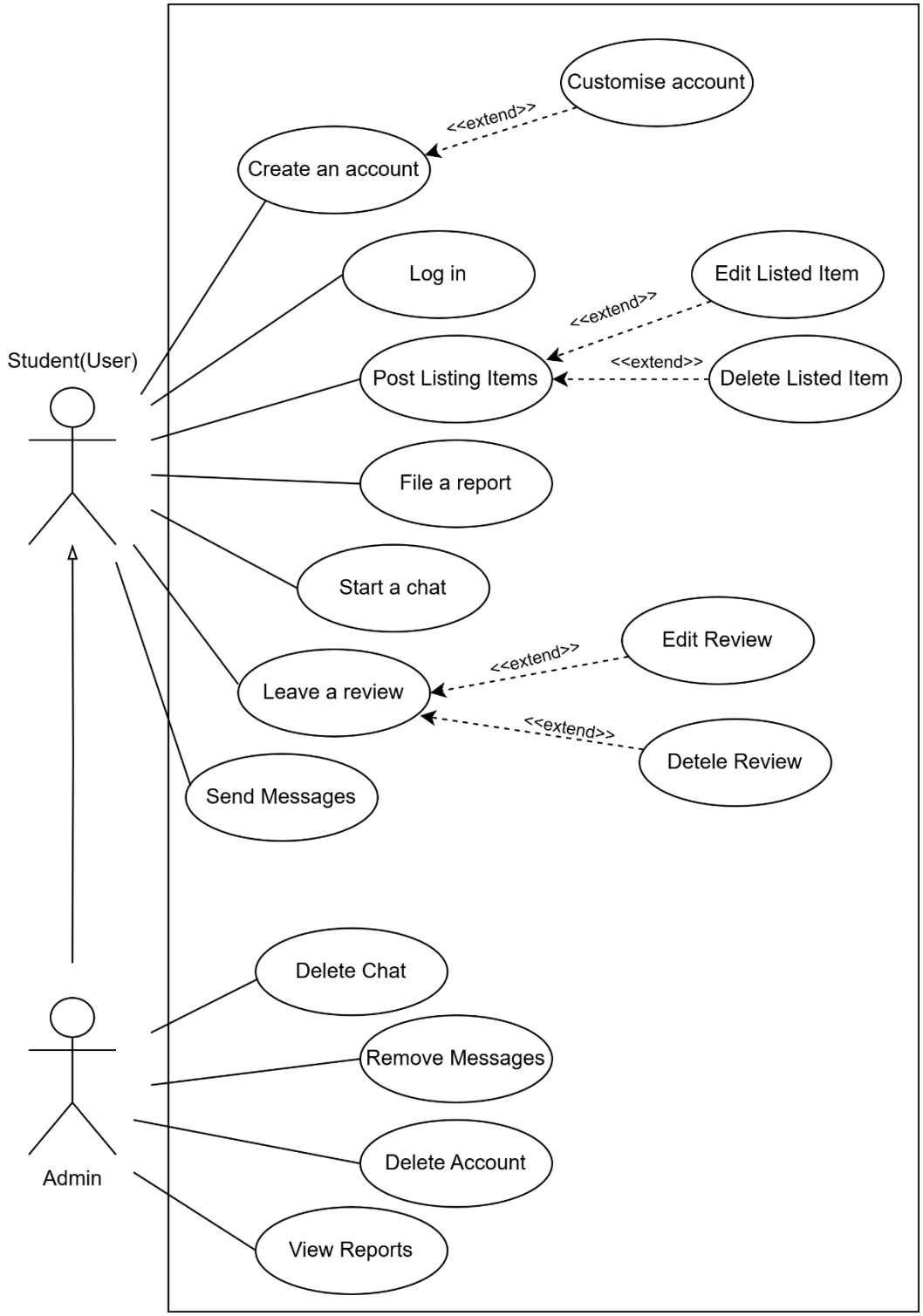

Figure Model X: Use-case Diagram