

Hands-On Activity: Reverse-Engineering a Real-Life App






Objective:

Trainees will explore a real online application, identify the core features and data involved, extract entities, relationships, and attributes, and then create an ERD and relational schema.

Learning Outcomes

- Practice system analysis based on real apps
- Strengthening ERD and schema design skills
- Connect database theory to real-world systems
- Encourage team-based problem solving and communication

Top 5 Suggested Apps for the Activity

#	Application	Link	Why It's a Great Fit
1	Talabat	talabat.com	 E-commerce logic (food delivery), user profiles, orders, restaurants, payment methods – excellent for entity recognition and M:N relationships.
2	Airbnb	airbnb.com	 Complex relationships between users, hosts, properties, bookings, availability – great for scheduling and role-based logic.
3	Booksy	booksy.com	 Service booking platform – clear booking and scheduling flows, service providers, customers, and time slots.
4	Fiverr	fiverr.com	 Freelancing e-commerce with complex user roles, gig listings, transactions, and reviews – strong on entity diversity and business rules.
5	Coursera	coursera.org	 Online learning platform – includes courses, instructors, students, enrollment, certifications – blends e-commerce and education logic.

Each App Helps Them Practice:

- Analyzing business requirements
- Identifying entities and attributes
- Recognizing 1:1, 1:M, and M:N relationships
- Handling real-world constraints (like availability, user roles, time slots, payments)

Real-World Constraints Checklist

Use this while analyzing the app and designing your ERD & schema


User Roles

- Are there different types of users (e.g., customer, provider, admin)?
- Can one person have multiple roles?
- Does each role have different access or behavior?

 **Example:** A user might be both a host and a guest on Airbnb.


Time Slots / Scheduling

- Are there bookings, appointments, or availability?
- Can two users book the same time slot?
- How is availability managed?

 **Example:** A course scheduling system should not allow overlapping sessions for the same classroom or instructor.


Payments & Transactions

- Is there an online payment process?
- Do payments go through multiple states (pending, paid, failed)?
- Are refunds or receipts tracked?

 **Example:** An order might be placed but payment could fail—how does the system handle that?


Capacity or Inventory

- Are there quantity limits (e.g., stock, room capacity, course seats)?
- Can items run out or be fully booked?
- Is overbooking prevented?

 **Example:** A class might have only 20 seats available.


Location & Area Rules

- Are services or products limited by location?
- Can users only interact with certain providers based on their city or address?

 **Example:** Talabat restricts restaurant options based on the user's delivery zone.

Ratings & Reviews

- Can users leave feedback?
- Are reviews tied to specific orders or bookings?
- Can a user rate the same item more than once?

 **Example:** A student should only review a course they've completed.

Usage Tip:

As you design the ERD and relational schema, ask:

- "How does the database enforce this rule?"
- "What would happen if we ignored this constraint?"

Example:

Constraint:

A service provider (e.g., a stylist or host) should not be double-booked for overlapping time slots.

- **How does the database enforce this rule?**

We use a Bookings table with fields like StartTime, EndTime, and ProviderID. Before inserting a new booking, the system checks for time overlap using application logic or a stored procedure. Alternatively, we can define unique time slots and apply a unique constraint on ProviderID + TimeSlotID.

- **What would happen if we ignored this constraint?**

The system would allow double-bookings, resulting in service conflicts, customer frustration, and operational issues for the provider.

Activity Steps

Step 1: Explore the Application

Each team opens the app and navigates through it:

- What actions can a user perform? (register, purchase, review, book, etc.)
- What kind of data is displayed or entered?
- What pages or modules exist? What changes from one page to another?

Deliverable:

A short list (5–7 points) describing the **main features** and the **data** involved in each.

Step 2: Extract Entities & Requirements

Based on the exploration, teams will brainstorm:

- What **entities** seem to exist? (e.g., User, Order, Product, Review...)
- What **attributes** do they have?
- What **relationships** exist between entities?

Deliverable:


A list of potential entities, their main attributes, and the type of relationships (one-to-one, one-to-many, many-to-many).

Step 3: Draw the ERD

Part A – Individual Work

Each team member should individually sketch their own ERD based on their understanding of the app:

- Identify entities, attributes, primary keys.
- Define relationships and cardinalities.

 **Purpose:** This ensures everyone practices their analysis skills before being influenced by group discussion

Part B – Team Discussion & Consolidation

Once everyone has a draft:

- The team comes together to compare their ERDs.
- Discuss differences, add missing ideas, and combine the best elements into a final team version.
- Clarify any uncertain relationships or missing data.


 Goal: Collaboratively agree on one ERD that best represents the app's database design.

They can use:


- Whiteboards or large paper for a group sketch
- Or digital tools (e.g., draw.io, [Lucidchart](https://lucidchart.com)) if preferred

Step 4: Map to Relational Schema

After finalizing the ERD as a team:

 Work as a Group:

- Use your agreed team ERD from Step 3.
- Together, map each entity and relationship to relational tables.
- Decide on:
 - Primary keys
 - Foreign keys
 - How to handle many-to-many relationships using bridge tables

 This step should be fully collaborative—discuss each decision as a group to ensure everyone understands the structure.

Step 5: Presentation

- A quick summary of the app
- Their inferred ERD
- Their relational schema