# Databases II
# Workshop No. 1 — Project Definition and Database Modeling

**Team Members:**
Santiago Sanchez Moya – 20211020032
Juan Daniel Rodriguez Hurtado – 20211020152

Computer Engineering Program
School of Engineering
Universidad Distrital Francisco José de Caldas
**Professor:** Eng. Carlos Andrés Sierra, M.Sc.

Full-time Adjunct Professor

Semester 2025-III

# Contents

# 1. Introduction

This workshop marks the starting point of the course project for *Databases II*. The main objective is to define the business, user, and data context of a real-world, data-intensive application. This document compiles the project baseline, requirements, user stories, and the initial database architecture.

# 2. Business Model Canvas

The chosen application baseline is the global market for tourist accommodations and stays.

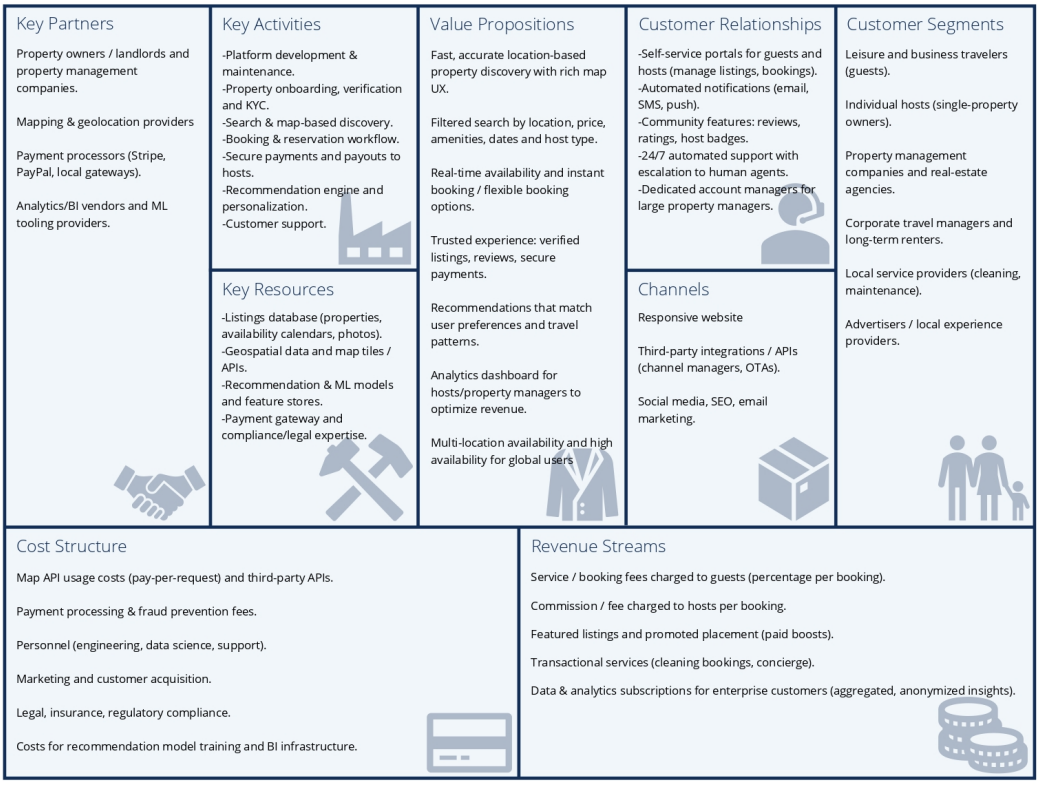


Figure 1: Business Model Canvas of the project baseline.

# 3. Requirements Documentation

## 3.1. Functional Requirements (FR)

FR1. **User Registration & Authentication**

FR1.1 Users must be able to register as guest or host.

FR1.2 The system must allow login/logout with secure password handling.

FR1.3 Hosts must be able to verify their properties (upload info, photos).

FR2. **Property Management (Hosts)**

FR2.1 Hosts must be able to create, update, and delete property listings.

FR2.2 Hosts must manage availability calendars for their properties.

FR2.3 Hosts must be able to view bookings and revenue history.

FR3. **Property Discovery (Guests)**

FR3.1 Guests must be able to search properties on a map by location.

FR3.2 Guests must filter results by price range, amenities, and availability.

FR3.3 Guests must be able to view property details, reviews, and host profile.

FR4. **Booking & Reservations**

FR4.1 Guests must be able to book available properties for selected dates.

FR4.2 The system must prevent double bookings for the same property/date.

FR4.3 Both guests and hosts must receive booking confirmations.

FR5. **Payments**

FR5.1 Guests must be able to pay for bookings using a payment gateway.

FR5.2 Hosts must receive payouts after bookings are completed.

FR5.3 The system must keep a transaction history.

FR6. **Reviews & Ratings**

FR6.1 Guests must be able to leave reviews and ratings after a stay.

FR6.2 Hosts must be able to respond to reviews.

FR6.3 Reviews must be linked to verified bookings only.

FR7. **Admin Functions**

FR7.1 Admins must manage users (ban, verify, reset).

FR7.2 Admins must manage reported properties/reviews.

FR7.3 Admins must access analytics on platform usage.

### 3.2. Non-Functional Requirements (NFR)

NFR1. **Performance**

NFR1.1 The system must return search results within 2 seconds for 95% of queries.

NFR1.2 Booking transactions must be processed in real-time (<3 seconds).

NFR2. **Scalability**

NFR2.1 The system must support at least 1,000 concurrent users without degradation.

NFR2.2 The database must handle growth up to 1 million property records.

NFR3. **Availability**

NFR3.1 The system must be available 99.5% of the time during the semester.

NFR3.2 Maintenance windows must be scheduled and notified in advance.

NFR4. **Security**

NFR4.1 Passwords must be stored securely (hashed + salted).

NFR4.2 All communications must use HTTPS/TLS.

NFR4.3 The system must comply with basic KYC checks for hosts.

NFR5. **Data Integrity & Consistency**

NFR5.1 Booking data must follow ACID transactions to avoid double-booking.

NFR5.2 Payment records must be immutable (append-only).

NFR6. **Usability**

NFR6.1 The UI must be responsive (desktop & mobile web).

NFR6.2 Guests must be able to complete a booking in <5 clicks from search.

NFR7. **Maintainability**

NFR7.1 The codebase must follow modular architecture.

NFR7.2 Documentation must be provided for database schema and APIs.

## 4. User Stories

The following tables present the main user stories for different roles: Guest, Host, and Admin.

| **Title** | Guest Registration and Login |
|---|---|
| **Priority** | |
| **Estimate** | |
| **User Story** | As a Guest, I want to register and log in securely so that I can access the platform's features. |
| **Acceptance Criteria** | Given the user is on the registration page. When they provide valid credentials and submit, then the system must create an account and allow secure login. |

Table 1: Guest Registration and Login

| | |
|---|---|
| **Title** | Property Discovery with Filters |
| **Priority** | |
| **Estimate** | |
| **User Story** | As a Guest, I want to search for properties using location and filters so that I can find the most suitable accommodation. |
| **Acceptance Criteria** | Given the user is on the search page. When they apply filters (price, amenities, dates), then the system must display matching properties on the map. |

Table 2: Property Discovery with Filters

| | |
|---|---|
| **Title** | Booking and Confirmation |
| **Priority** | |
| **Estimate** | |
| **User Story** | As a Guest, I want to book a property for selected dates so that I can secure my stay. |
| **Acceptance Criteria** | Given the user selects available dates for a property. When they confirm the booking, then the system must process the booking and send confirmation to both Guest and Host. |

Table 3: Booking and Confirmation

| | |
|---|---|
| **Title** | Guest Payments |
| **Priority** | |
| **Estimate** | |
| **User Story** | As a Guest, I want to pay for my booking through a secure payment gateway so that I can guarantee my reservation. |
| **Acceptance Criteria** | Given the user has selected a property and confirmed dates. When they provide payment details and submit, then the system must process the payment securely and confirm the booking. |

Table 4: Guest Payments

| | |
|---|---|
| **Title** | Guest Reviews |
| **Priority** | |
| **Estimate** | |
| **User Story** | As a Guest, I want to leave a review and rating after my stay so that I can help other users make better decisions. |
| **Acceptance Criteria** | Given the user has completed a verified booking. When they submit a review with rating, then the system must link the review to the booking and display it on the property page. |

Table 5: Guest Reviews

| | |
|---|---|
| **Title** | Host Property Management |
| **Priority** | |
| **Estimate** | |
| **User Story** | As a Host, I want to create, update and delete property listings so that I can manage what I offer on the platform. |
| **Acceptance Criteria** | Given the Host is logged in. When they add property details with photos, then the system must validate, save and display the listing in search results. |

Table 6: Host Property Management

| | |
|---|---|
| **Title** | Host Availability Calendar |
| **Priority** | |
| **Estimate** | |
| **User Story** | As a Host, I want to manage an availability calendar for my properties so that Guests cannot book unavailable dates. |
| **Acceptance Criteria** | Given the Host has created a property. When they update the availability calendar, then Guests must only see available dates when booking. |

Table 7: Host Availability Calendar

| | |
|---|---|
| **Title** | Host Revenue and Booking History |
| **Priority** | |
| **Estimate** | |
| **User Story** | As a Host, I want to view my bookings and revenue history so that I can track my earnings and occupancy. |
| **Acceptance Criteria** | Given the Host is logged in. When they access the booking history page, then the system must display all past bookings and payouts. |

Table 8: Host Revenue and Booking History

| | |
|---|---|
| **Title** | Host Respond to Reviews |
| **Priority** | |
| **Estimate** | |
| **User Story** | As a Host, I want to respond to reviews so that I can engage with guests and maintain my reputation. |
| **Acceptance Criteria** | Given a Guest has left a review. When the Host writes a response and submits, then the response must appear below the Guest's review. |

Table 9: Host Respond to Reviews

| | |
|---|---|
| **Title** | Admin User Management |
| **Priority** | |
| **Estimate** | |
| **User Story** | As an Admin, I want to ban, verify, and reset user accounts so that I can ensure platform security and compliance. |
| **Acceptance Criteria** | Given the Admin is logged into the dashboard. When they select a user account and choose an action (ban, verify, reset), then the system must apply the changes immediately. |

Table 10: Admin User Management

| | |
|---|---|
| **Title** | Admin Moderation of Properties and Reviews |
| **Priority** | |
| **Estimate** | |
| **User Story** | As an Admin, I want to manage reported properties and reviews so that I can maintain quality and trust in the platform. |
| **Acceptance Criteria** | Given a property or review has been reported. When the Admin reviews the report, then they can approve, remove, or take further action on the content. |

Table 11: Admin Moderation of Properties and Reviews

| | |
|---|---|
| **Title** | Admin Analytics Dashboard |
| **Priority** | |
| **Estimate** | |
| **User Story** | As an Admin, I want to access analytics about platform usage so that I can evaluate performance and make strategic decisions. |
| **Acceptance Criteria** | Given the Admin is logged into the dashboard. When they open the analytics section, then the system must display key metrics (users, bookings, revenue, reports). |

Table 12: Admin Analytics Dashboard

# 5. Initial Database Architecture

## 5.1. High-level Architecture

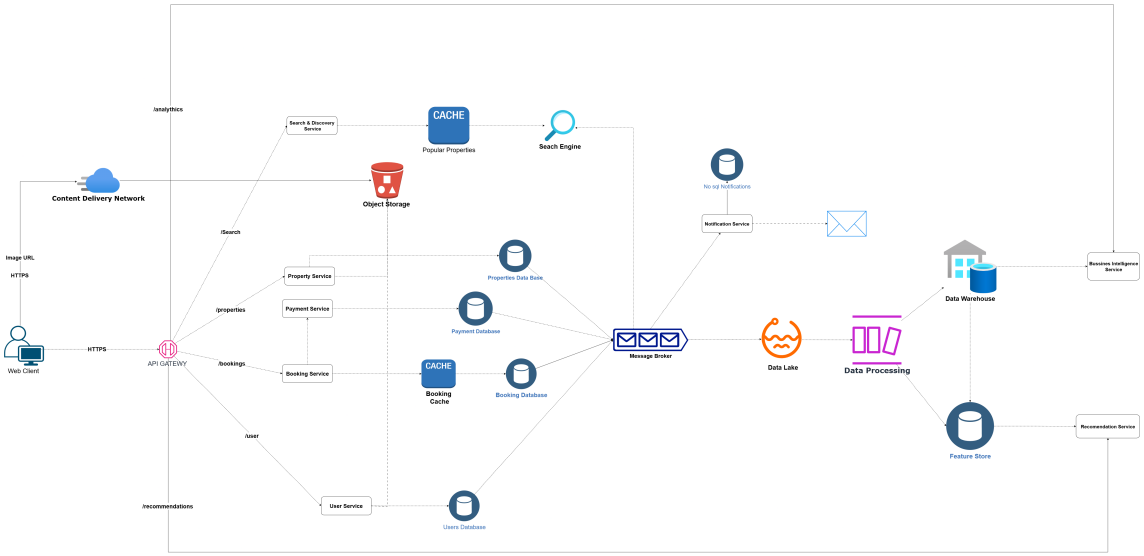


Figure 2: High-level database architecture.

## 5.2. Entity-Relationship Diagram

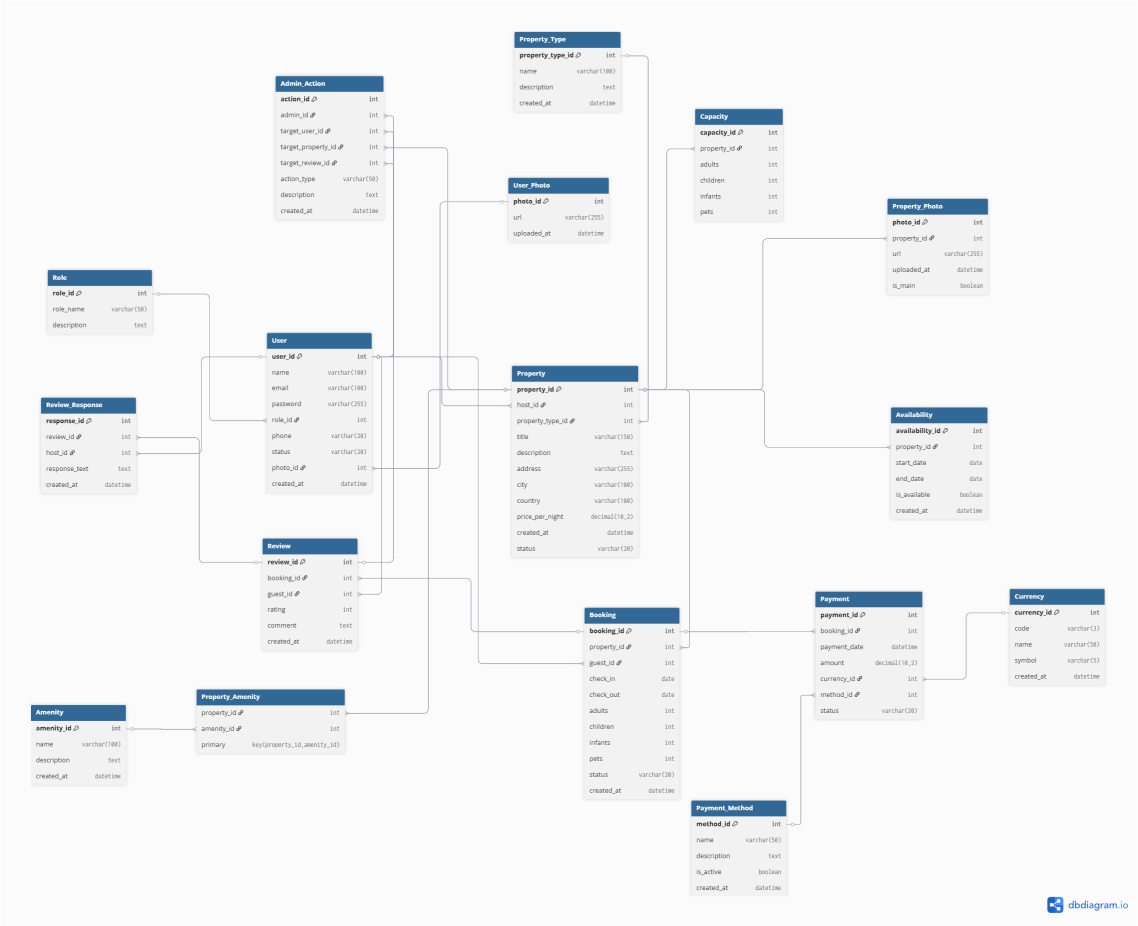


Figure 3: Initial ER diagram of the database.

### User Management

- **User**: Represents a system user (guest, host, or admin). Contains personal details, status, and one profile photo.
- **User_Photo**: Stores the user's profile image (one-to-one with User).
- **Role**: Defines the role of the user (guest, host, admin). One role can be assigned to many users.
- **Relationships**:
  - User–Role: Many-to-One
  - User–User_Photo: One-to-One

### Property Management

- **Property**: Listing created by a host. Includes title, description, location, price, and status.
- **Property_Type**: Classifies properties (Apartment, House, Cabin, Villa, etc.).

- **Property_Photo**: Stores photos of a property (One-to-Many with Property).
- **Capacity**: Defines the maximum allowed adults, children, infants, and pets (One-to-One with Property).
- **Availability**: Defines date ranges where a property is available or blocked (One-to-Many with Property).
- **Amenity**: Represents features (WiFi, Kitchen, Parking, etc.).
- **Property_Amenity**: Join table to model the many-to-many relationship between Property and Amenity.
- **Relationships**:
  - Property–User: Many-to-One (host creates many properties).
  - Property–Property_Type: Many-to-One.
  - Property–Property_Photo: One-to-Many.
  - Property–Capacity: One-to-One.
  - Property–Availability: One-to-Many.
  - Property–Amenity: Many-to-Many (via Property_Amenity).

**Booking and Payments**

- **Booking**: Reservation made by a guest for a property. Includes check-in, check-out, and number of guests.
- **Payment**: Stores payment details (amount, method, currency, status).
- **Payment_Method**: Defines available methods (Credit Card, PayPal, etc.).
- **Currency**: Stores supported currencies (USD, COP, EUR).
- **Relationships**:
  - Booking–Property: Many-to-One.
  - Booking–User: Many-to-One (guest makes bookings).
  - Booking–Payment: One-to-One.
  - Payment–Currency: Many-to-One.
  - Payment–Payment_Method: Many-to-One.

**Reviews and Responses**

- **Review**: Guest feedback after a booking (rating and comment).
- **Review_Response**: Host's reply to a review (One-to-One with Review).
- **Relationships**:
  - Review–Booking: Many-to-One.

- Review–User: Many-to-One (guest leaves reviews).
- Review_Response–Review: One-to-One.
- Review_Response–User: Many-to-One (host responds).

**Administration**

- **Admin_Action**: Represents administrative actions (ban user, verify property, delete review).
- **Relationships**:
  - Admin_Action–User: Many-to-One (admin performing action).
  - Admin_Action–User (target): Optional link to affected user.
  - Admin_Action–Property (target): Optional link to affected property.
  - Admin_Action–Review (target): Optional link to affected review.

## 6. References

## References


[1] IBM. (2025, July 25). *What is a data architecture?*. IBM Think. Disponible en: `https://www.ibm.com/think/topics/data-architecture`

[2] Corporate Finance Institute. (s. f.). *Business Model Canvas Examples*. Disponible en: `https://corporatefinanceinstitute.com/resources/management/business-model-canvas-examples/`

[3] Autor(es). (Año). *Designing Machine Learning Systems: An Iterative Process for Production-Ready Applications*. Editorial/Editorial académica. *[Completar con autor y año cuando estén disponibles].*

[4] Autor desconocido. (s. f.). *[Figura sobre arquitectura de datos].* Substack. Disponible en: `https://substackcdn.com/image/fetch/f_auto,q_auto:good,fl_progressive:steep/https%3A%2F%2Fsubstack-post-media.s3.amazonaws.com%2Fpublic%2Fimages%2Feb15f23f-a99f-406b-add4-eaffe0d1633d_2332x2696.png`

[5] YouTube. (2025, September 27). *Hotel Reservation (AirBnb, Booking.com) - System Design Interview Question* [Video]. YouTube. Disponible en: `https://www.youtube.com/watch?v=m67Mjbx6DMY`