CAT 2

UML Analysis

Alejandro Pérez Bueno

Apr 25, 2025

Table of Contents

Self-Responsibility Declaration

I certify that I have completed CAT2 entirely individually and only with the help deemed appropriate by the teaching staff of this subject according to the FAQs about plagiarism. I understand that unoriginal work and/or the use of generative AI will mean that the submitted activity will not be corrected, and a grade of D will automatically be assigned.

Question 1

Store Class Diagram

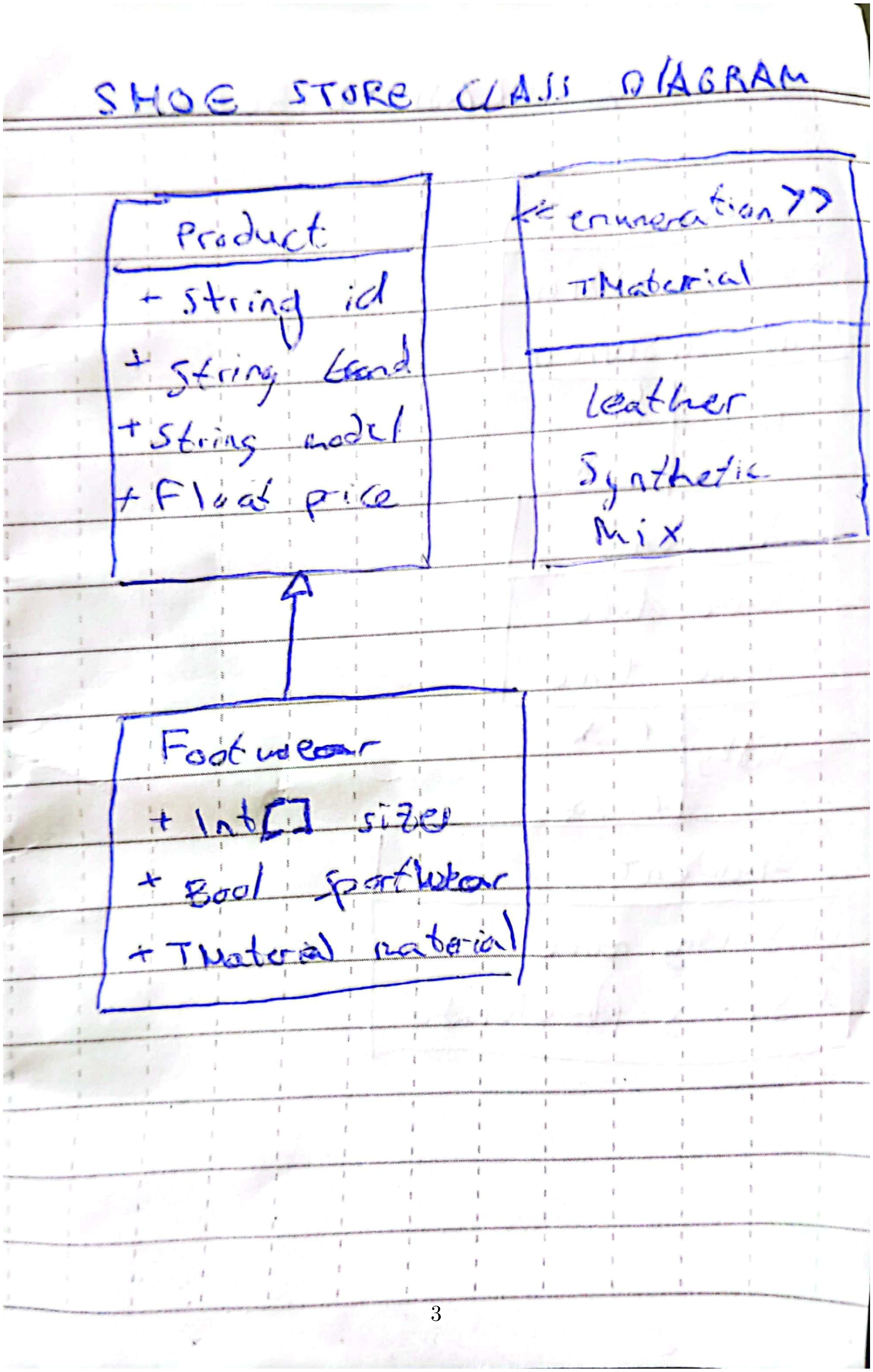

Figure 1: Class Diagram

Academy Class Diagram

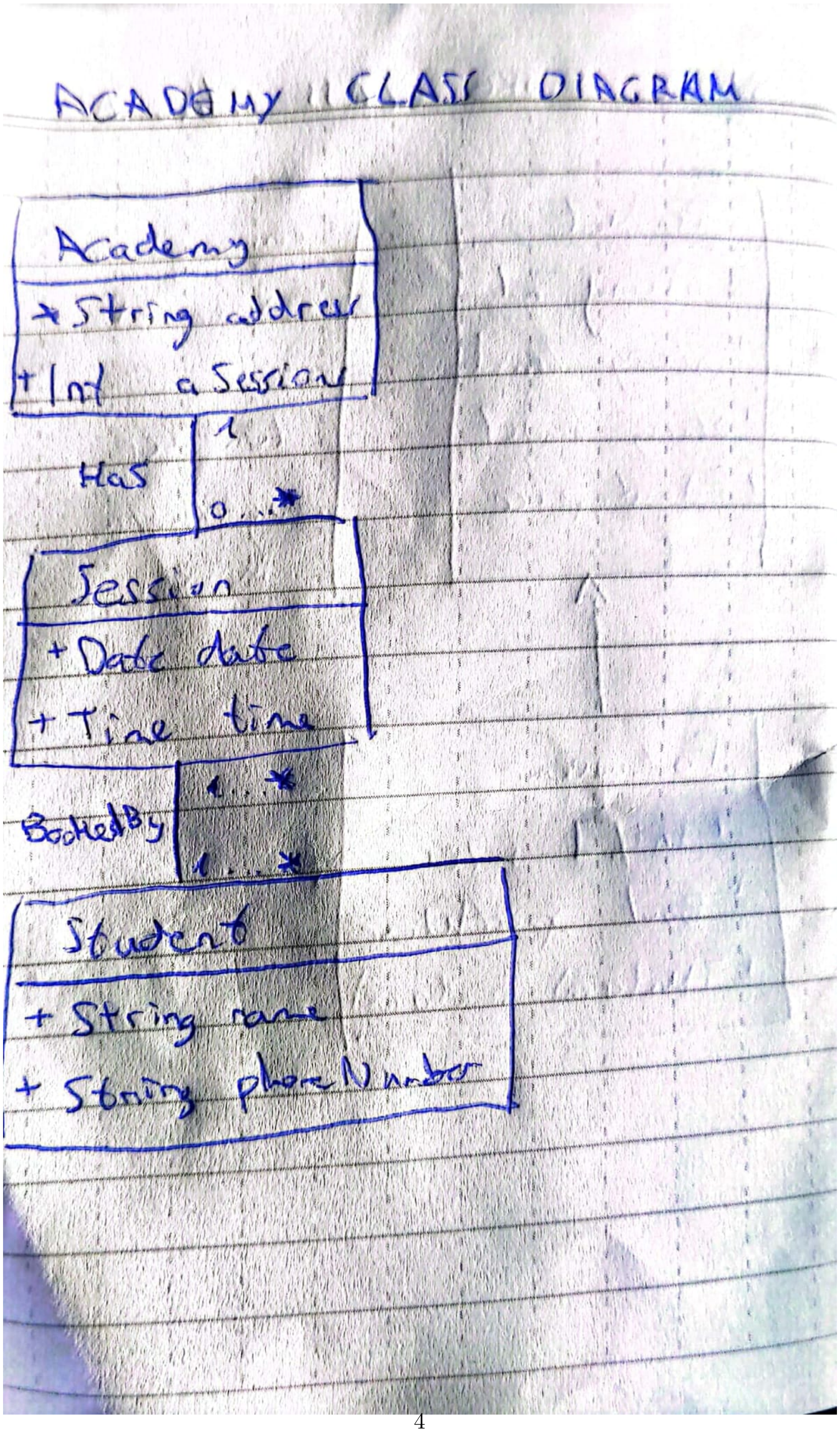

Figure 2: Class Diagram

Activity Diagram

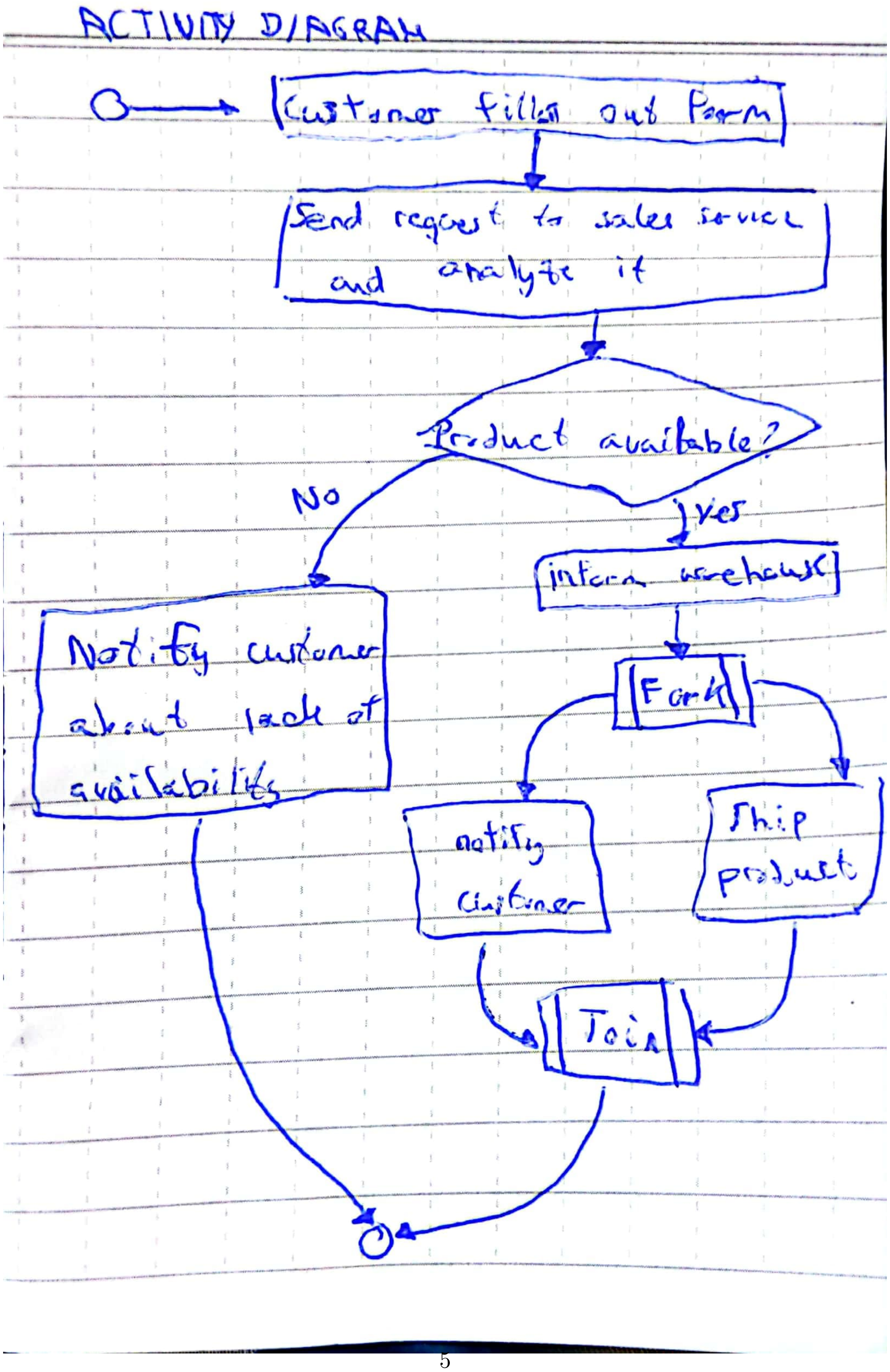

Figure 3: Activity Diagram

Question 2

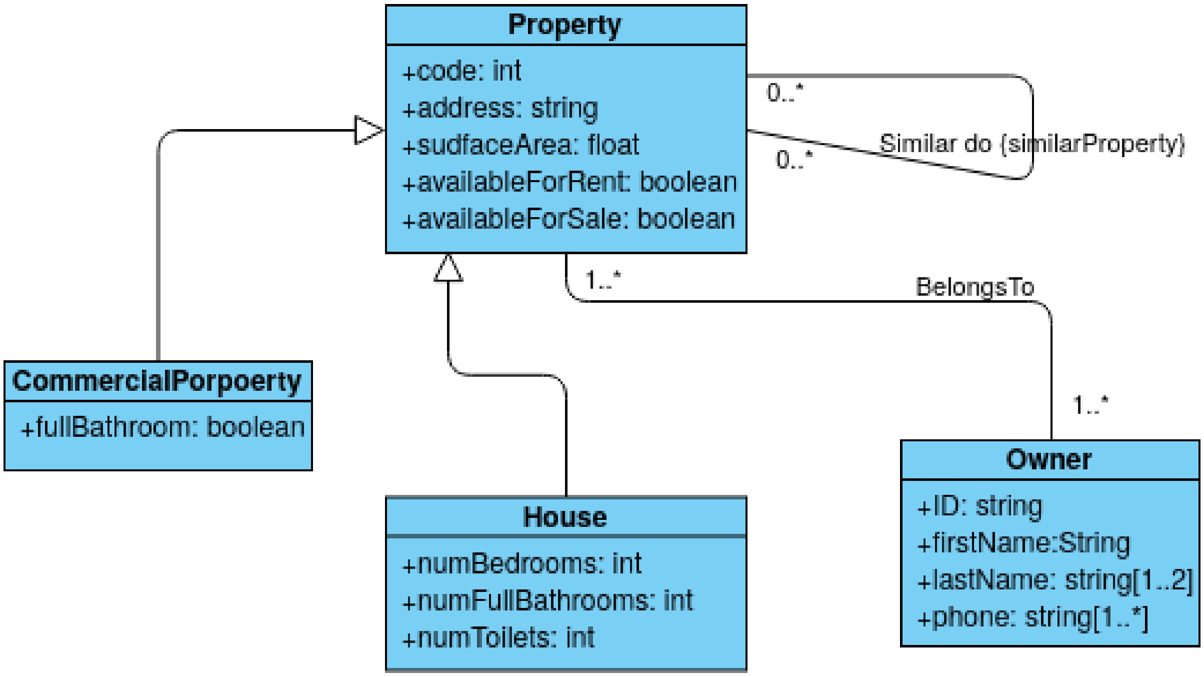

Figure 4: Class Diagram

Question 3

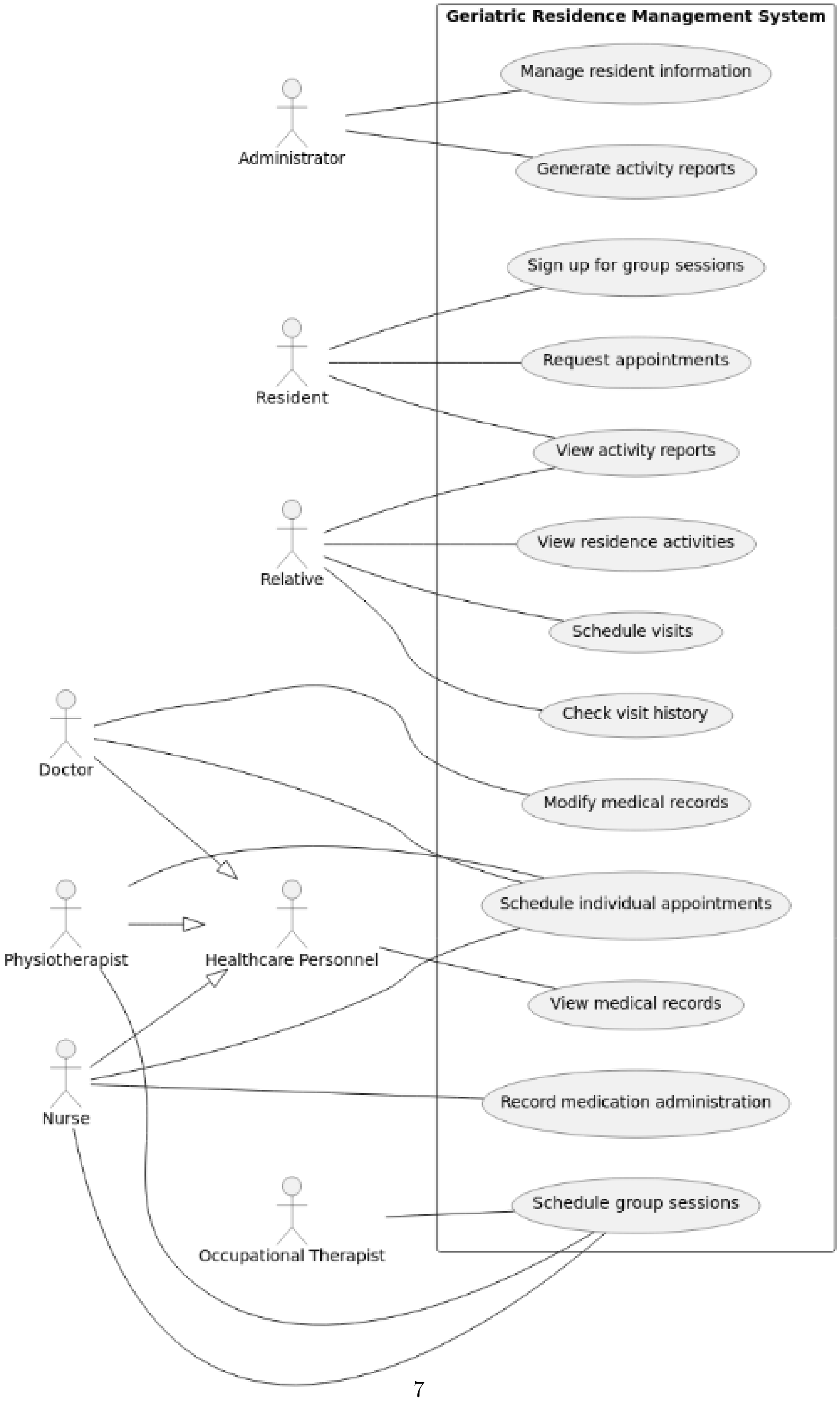

Figure 5: Use Case Diagram

Question 4

Section A

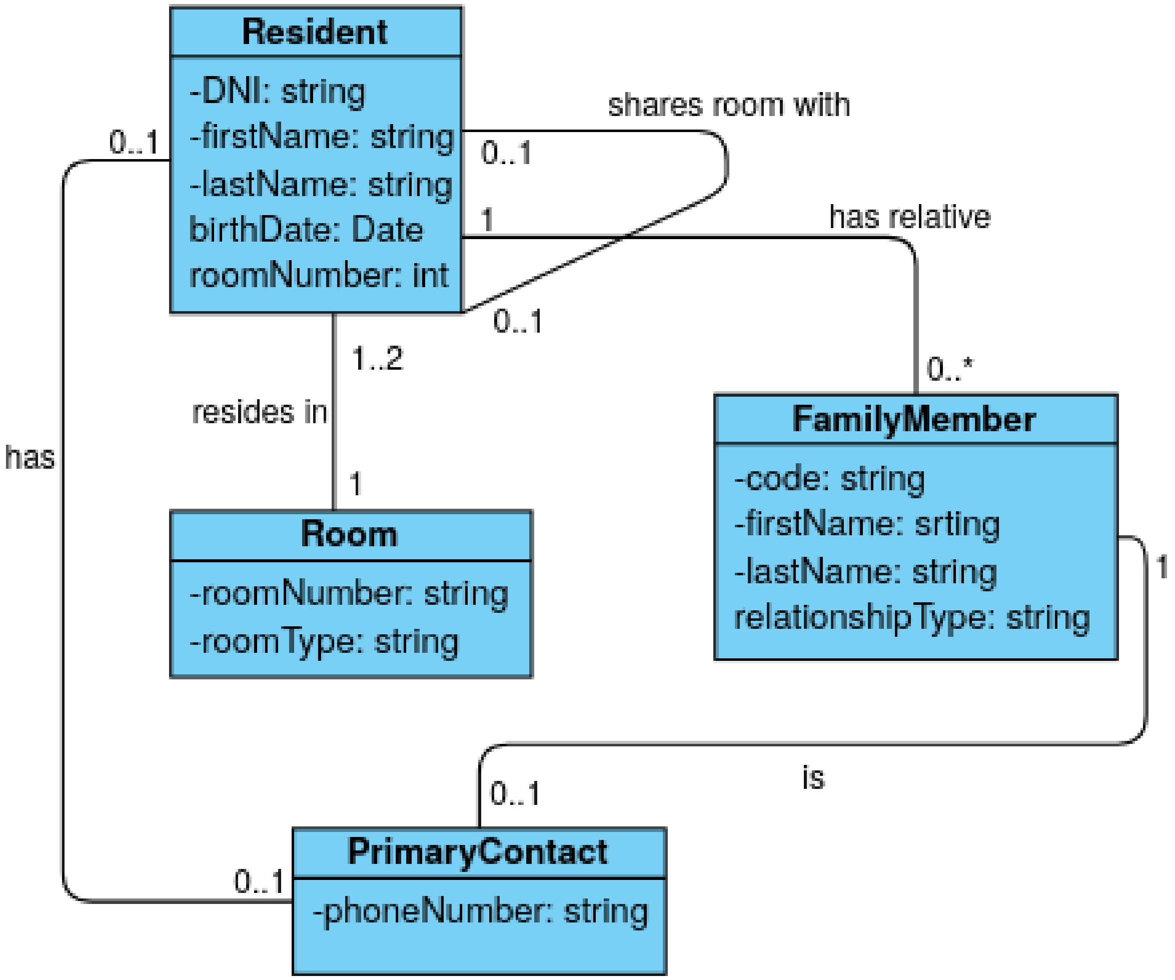

Figure 6: Class Diagram

Key Restrictions:

1. Each Resident is uniquely identified by their DNI.
2. Each FamilyMember is uniquely identified by their code.
3. A Resident can have zero or more FamilyMembers.
4. A FamilyMember can be related to multiple Residents.
5. A FamilyMember can be a PrimaryContact for some Residents but not for others.
6. A Room can have one or two Residents (single or double rooms).

Integrity Constraints:

1. If a Room has two Residents, they must share a room with each other.
2. A FamilyMember must have a relationshipType from the set {partner, son/daughter, brother/sister, nephew/niece, grandson/granddaughter}.
3. A PrimaryContact must be a FamilyMember of the corresponding Resident.
4. Each Resident can have at most one PrimaryContact.

Section B

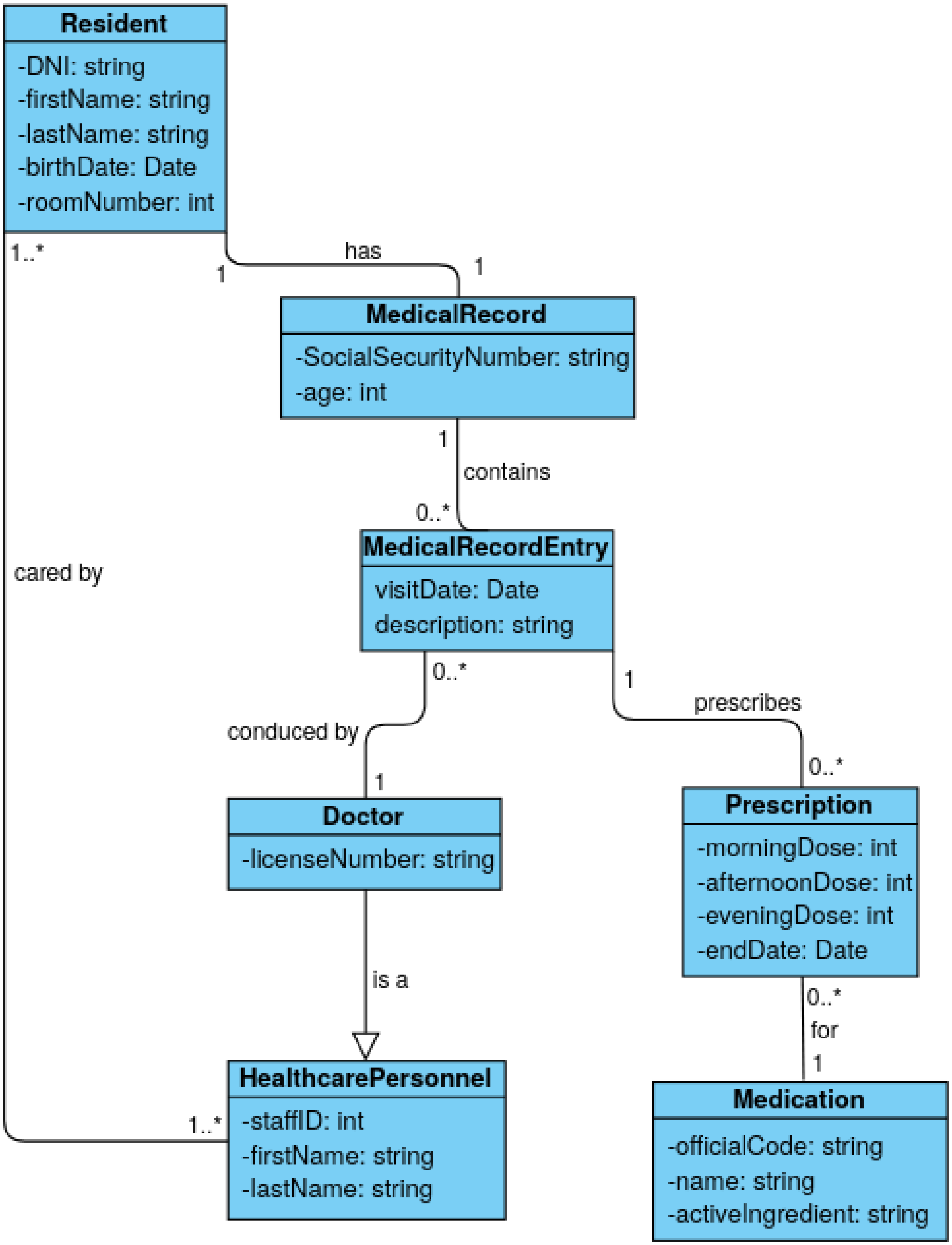

Figure 7: Class Diagram

Key Restrictions:

1. Each MedicalRecord belongs to exactly one Resident.
2. Each MedicalRecordEntry belongs to exactly one MedicalRecord.
3. Each Medication is uniquely identified by its officialCode.
4. Each HealthcarePersonnel is uniquely identified by their staffID.
5. Each Doctor has a unique licenseNumber.

Integrity Constraints:

1. The age in MedicalRecord must be calculated from the Resident's birthDate.
2. A Doctor is a specialized type of HealthcarePersonnel.
3. The endDate of a Prescription is mandatory only for non-chronic medications.
4. All HealthcarePersonnel who care for a Resident can access their MedicalRecord.

Derived Information:

1. The age of a Resident is derived from their date of birth.
2. Whether a medication is chronic or not can be derived from the presence or absence of an endDate in the Prescription.
3. The roomType (single or double) can be derived from the number of Residents assigned to a Room.

Question 5

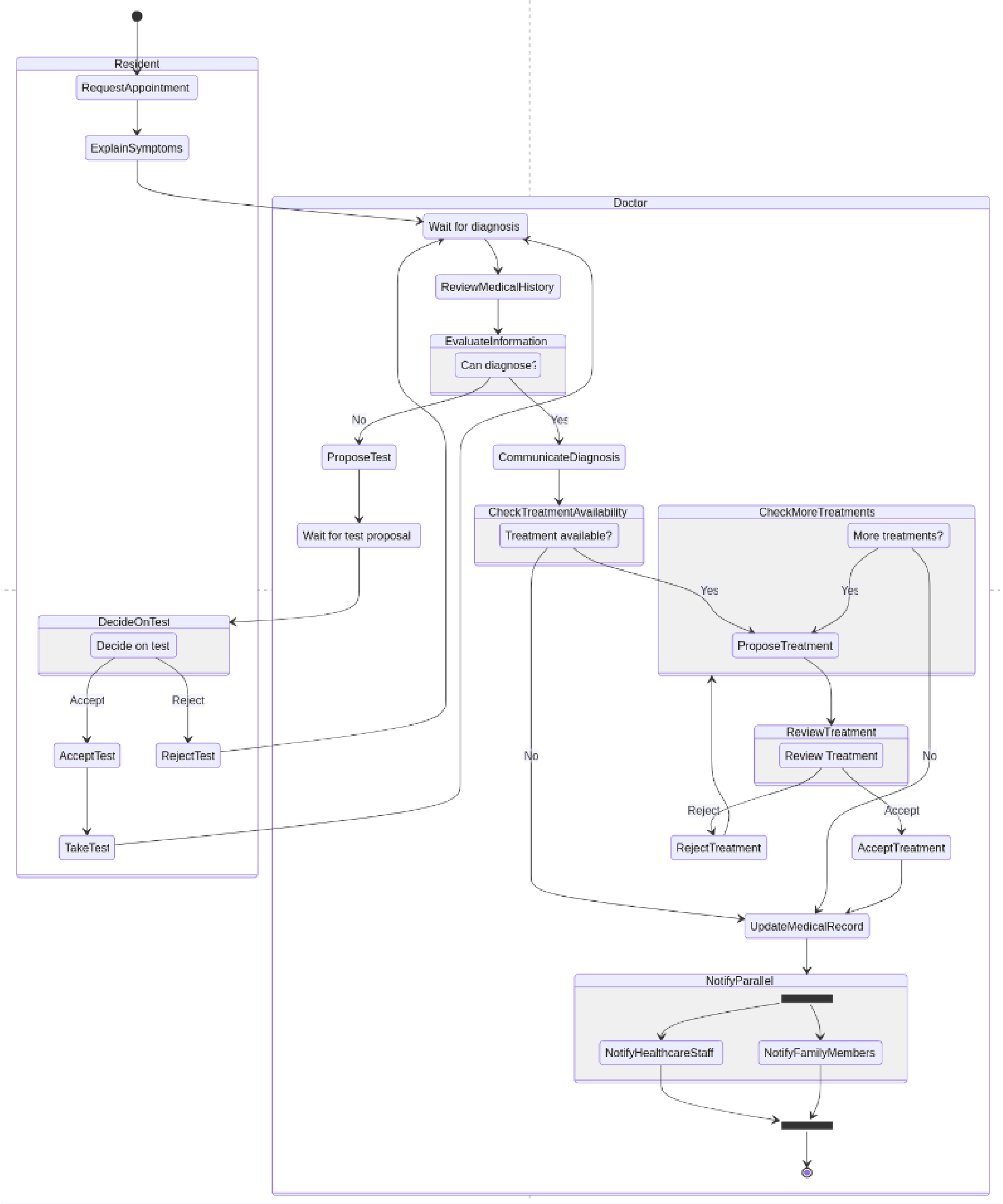

Figure 8: Activity Diagram

Question 6

ℹ Note

I used a mermaid.js editor in this exercise

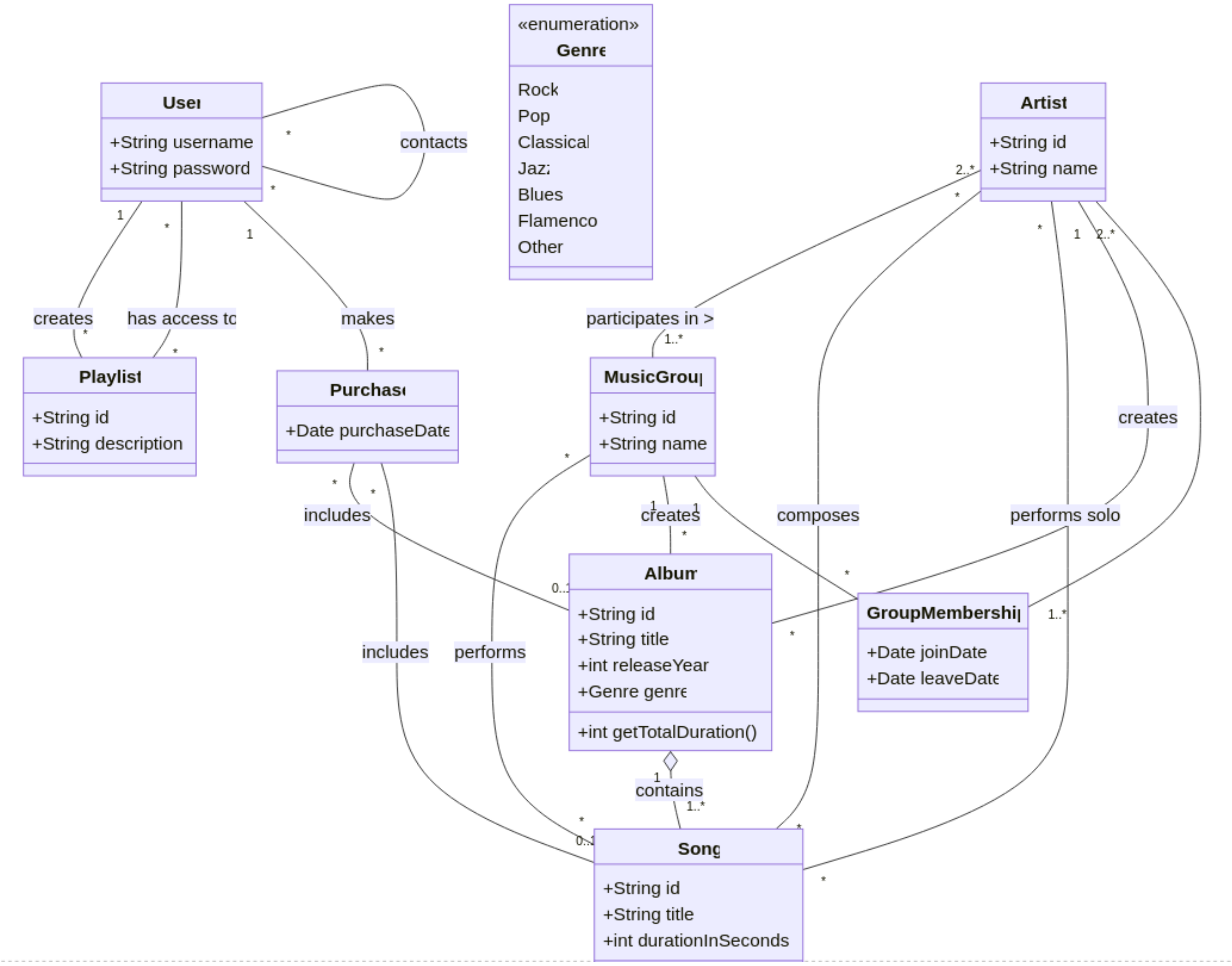

Figure 9: Class Diagram

Key Constraints:

1. Songs are uniquely identified by their alphanumeric ID.
2. Artists and Music Groups are uniquely identified by their alphanumeric ID.
3. Albums are uniquely identified by their alphanumeric ID.
4. Users are uniquely identified by their username.
5. Playlists are uniquely identified by their ID.

Integrity Constraints:

1. **Music Group Constraint**: A music group must have at least two artists as members.
2. **Group Membership Consistency**: For any artist in a group, the join date must be provided. If the leave date exists, it must be after the join date.
3. **Song Creators**: Each song must have at least one composer (artist) and at least one performer (artist or music group).
4. **Album Organization**: Every song must belong to exactly one album. No standalone songs are allowed.
5. **Album Ownership**: An album must belong to either an artist or a music group, but not both.
6. **Genre Limitation**: Album genre must be one of the predefined values: Rock, Pop, Classical, Jazz, Blues, Flamenco, or Other.
7. **Playlist Creator**: Each playlist must have exactly one creator (owner).
8. **Purchase Consistency**: A purchase must include either a song or an album, but not both simultaneously.
9. **Password Security**: User passwords must be stored securely (hashed and salted).

Derived Information:

1. **Album Total Duration**: The total duration of an album is calculated by summing the durations of all songs it contains.
   - Calculation: `sum(song.durationInSeconds)` for all songs in the album.
2. **Playlist Duration**: The total duration of a playlist can be calculated by summing the durations of all songs it contains.
   - Calculation: `sum(song.durationInSeconds)` for all songs in the playlist.
3. **Active Group Members**: Current members of a music group can be derived by filtering group memberships where leave date is null.
   - Calculation: Select all artists from GroupMembership where the leaveDate is null for a specific group.
4. **Artist's Discography**: The complete collection of albums an artist has created (solo or with groups) can be derived.
   - Calculation: Union of albums directly created by the artist and albums created by groups the artist was a member of during the album's release year.
5. **Shared Playlists**: For a user, the list of playlists shared with them can be derived from the access relationships.
   - Calculation: Select all playlists where the user has access but is not the creator.