SOEN 342 Project

Section II

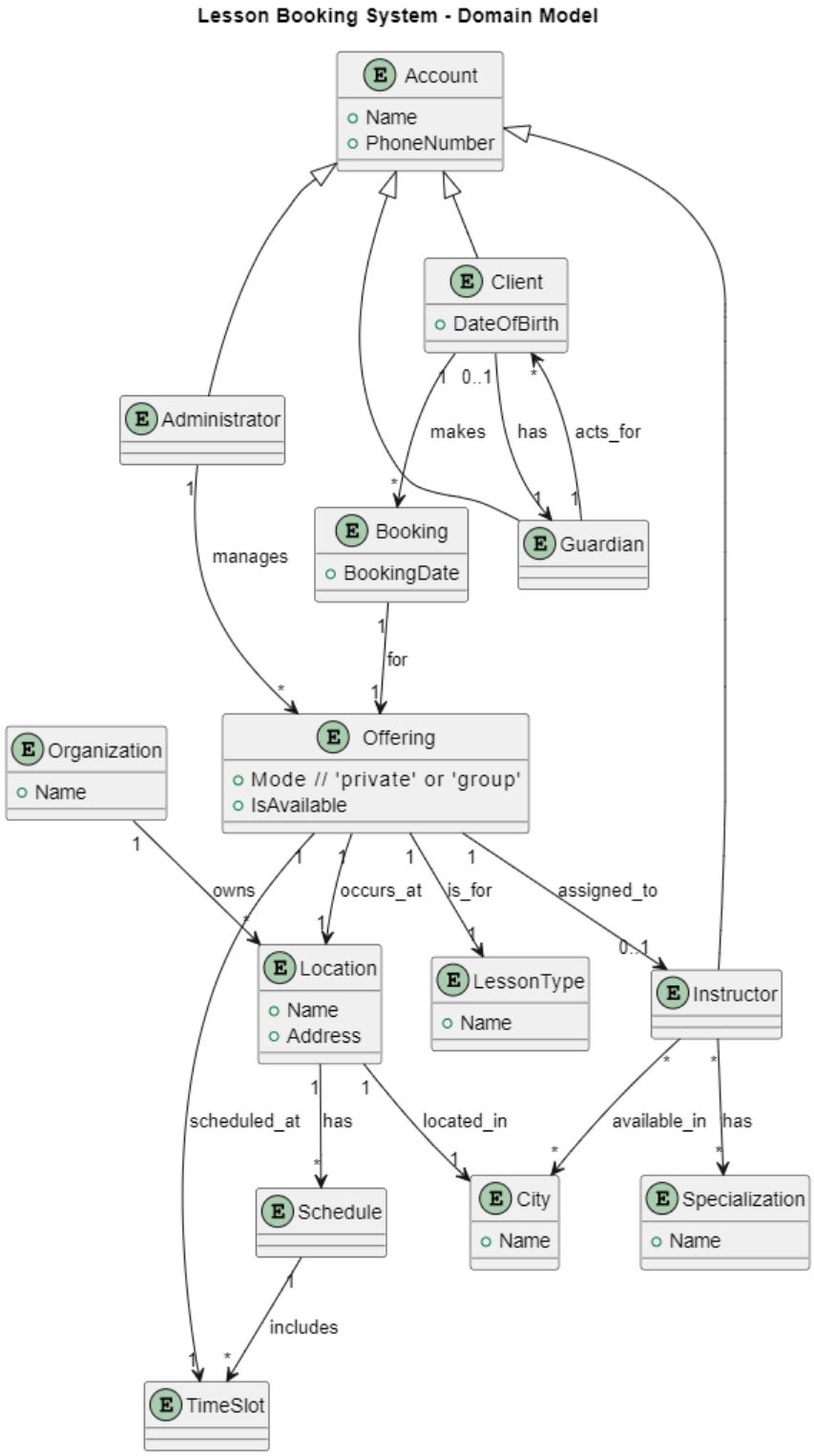
Franco Dominguez (40256199)
Ashkan Forghani (40176561)
Team #33

Iteration #2

Use Case 1:

Process Offerings

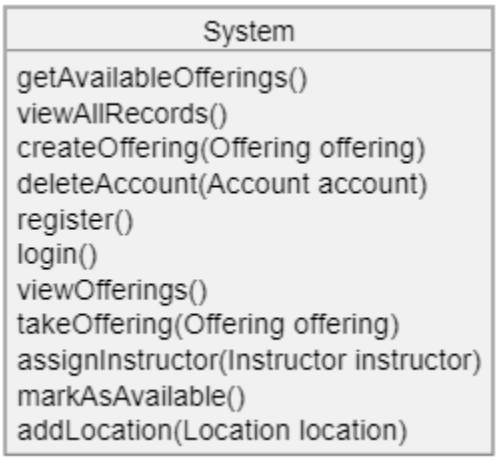
Domain Model and Package Diagram



UML Interaction Diagrams

System Sequence Diagrams

System Operations



NOTE: System operations regarding the processing of bookings will be added in during iteration #3

Operation Contracts

Contract CO1: getAvailableOfferings
Operation: getAvailableOfferings()
Cross References: Use Case: Get available offerings
Preconditions:

- The client is registered in the system.
- There are active lessons and schedules available in the system.
- The system has assigned instructors to the lessons for public viewing.

Postconditions:

- A list of available offerings (lessons with schedules and assigned instructors) is retrieved.
- Only offerings with available time slots are included.
- Offerings that are fully booked are excluded or marked as non-available.

Contract CO2: viewAllRecords
Operation: viewAllRecords()
Cross References: Use Case: Request to view all records
Preconditions:

- The operation is performed by the system administrator.
- The system has records of clients, instructors, lessons, and bookings.

Postconditions:

- All records (clients, instructors, lessons, schedules, bookings) are retrieved and displayed to the administrator.
- No modifications are made to the records during this operation.
- The retrieved records remain consistent and up-to-date.

Contract CO3: createOffering
Operation: createOffering(offering: instance of an Offering object)
Cross References: Use Case: Creating an offering
Preconditions:

- A valid lesson and schedule exist in the system.
- The operation is performed by an administrator.

Postconditions:

- A new offering is created and linked to the appropriate lesson and schedule.
- The offering is made available for clients to book.

Contract CO4: deleteAccount
Operation: deleteAccount(account: an instance of an Account object)
Cross References: Use Case: Deleting an account
Preconditions:

- The operation is performed by an administrator.
- The account to be deleted exists in the system.

Postconditions:

- The specified account is removed from the system.
- All associated bookings or records tied to the account are deleted or flagged as inactive.

Contract CO5: register
Operation: register()
Cross References: Use Case: Registering a new account
Preconditions:

- The user is not already registered in the system.

Postconditions:

- A new client or instructor account is created.
- The account is saved and can now perform operations like booking or assigning lessons.

Contract CO6: login
Operation: login()
Cross References: Use Case: Logging into an account
Preconditions:

- The user must have a registered account.
- Valid login credentials (username and password) are provided.

Postconditions:

- The user is granted access to their account.
- Session information is stored to track the user's login.

Contract CO7: viewOfferings
Operation: viewOfferings()
Cross References: Use Case: Check to see all available offerings
Preconditions:

- The user must be a registered client or instructor.

Postconditions:

- A list of all available offerings (lessons, schedules) is retrieved and displayed to the user.

Contract CO8: takeOffering
Operation: takeOffering(offering: instance of an offering object)
Cross References: Use Case:
Preconditions:

- The offering must be available.
- The user must be a registered client or instructor.

Postconditions:

- The user is successfully booked into the offering.
- The offering's availability status is updated accordingly.

Contract CO9: assignInstructor
Operation: assignInstructor(instructor: instance of an Instructor object)
Cross References: Use Case:
Preconditions:

- The instructor must be registered and available for the specified time slots.

Postconditions:

- The instructor is assigned to the offering.
- The offering is marked as available for clients to book.

Contract CO10: markAsAvailable
Operation: markAsAvailable()
Cross References: Use Case: Marking an offering as Available
Preconditions:

- The offering or time slot exists in the system.
- The offering or time slot is currently marked as unavailable.
- The operation is performed by an administrator or the assigned instructor.

Postconditions:

- The offering or time slot is marked as available.
- Clients can now see the offering in the available offerings list and book it if desired.

Contract CO11: addLoaction
Operation: addLoaction(location: an instance of a Location object)
Cross References: Use Case: Adding a new location
Preconditions:

- The operation is performed by an administrator.
- The location details (address, rooms, available amenities) are provided.

Postconditions:

- A new location is added to the system.
- The location is available for adding schedules, lessons, and offerings.
- The location is visible in the system’s available locations list.

UML Class Diagram