|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requirement ID** | **Description** | **Type** | **Subtype** | **Priority** | **Database Query / Implementation** |
| **RQ-01** | The system shall allow users to register with a unique username and password. | Functional | User Management | Must have | SELECT \* FROM Customer WHERE Phone\_No = ?  INSERT INTO Customer (Name, Password, Phone\_No, IsAdministrator) VALUES (?, ?, ?, 0) |
| **RQ-02** | The system shall authenticate users upon login. | Functional | User Management | Must have | SELECT Name, IsAdministrator, User\_ID FROM Customer WHERE LOWER(TRIM(Name))= ? AND Password = ? |
| **RQ-03** | Administrators shall be able to add and update movie details, including title, genre, and duration. | Functional | Movie Management | Must have | INSERT INTO Movie (Title, Genre, Duration, Description, Status) VALUES (?, ?, ?, ?, ?) UPDATE Movie SET Title = ?, Genre = ?, Duration = ?, Description = ?, Status = ? WHERE Movie\_ID = ? |
| **RQ-04** | The system shall allow users to view a list of available movies. | Functional | Movie listing | Must have | SELECT \* FROM Movie WHERE Status = 'Available' |
| **RQ-05** | The system shall display the total number of available seats for a movie | Functional | Booking Management | Should Have | SELECT Available\_Seats FROM Cinema\_Shows WHERE Movie\_ID = ? |
| **RQ-06** | The system shall enable customers to book tickets for selected movies | Functional | Booking Management | Must have | INSERT INTO Booking (User\_ID, Movie\_ID, Seat\_No, Total\_Amount) VALUES (?, ?, ?, ?) |
| **RQ-06** | The system should enable users to make specific seat selection | Functional | Booking Managment | Must have | SELECT CS.Seat\_Matrix, TR.Price\_Per\_Seat ,CS.Movie\_ID              FROM Cinema\_Shows CS              JOIN Theatre\_Rooms TR ON CS.Room\_ID = TR.Room\_ID              WHERE CS.Show\_ID = ?   UPDATE Cinema\_Shows                  SET Seat\_Matrix = ?, Available\_Seats = Available\_Seats - 1                  WHERE Show\_ID = ?  INSERT INTO Booking (User\_ID, Show\_ID, Movie\_ID, Seat\_No, Total\_Amount)                  VALUES (?, ?, ?, ?,?) |
| **RQ-07** | Upon successful booking, the system shall generate a simple confirmation message on the screen. | Functional | Booking Confirmation | Must Have | No direct database query; handled in the application logic. |
| **RQ-08** | Customers shall be able to cancel bookings via a simple form, and the system shall acknowledge the cancellation. | Functional | Booking Management | Could Have | SELECT Show\_ID, Seat\_No FROM Booking WHERE Booking\_ID = ?  SELECT Seat\_Matrix FROM Cinema\_Shows WHERE Show\_ID = ?                  UPDATE Cinema\_Shows                  SET Seat\_Matrix = ?, Available\_Seats = Available\_Seats + 1                  WHERE Show\_ID = ?  DELETE FROM Booking WHERE Booking\_ID = ? |
| **RQ-09** | The system shall display available snacks as a static list. | Functional | Concession Management | Should Have | SELECT \* FROM Snack\_Beverage |
| **RQ-010** | The system shall provide a basic feedback form for users to submit after watching the movie. | Functional | Feedback Management | Could have | No direct database query; handled in the application logic. |
| **RQ-011** | User passwords shall be stored using secure hashing algorithms. | Non-Functional | Security | Must have | Use a secure hashing algorithm (e.g., SHA-256) to hash passwords before storing them in the database. |
| **RQ-012** | The system shall be designed to be simple and intuitive using basic custom tkinter. | Non-Functional | Usability | Must have | Use customtkinter to create a user-friendly and intuitive GUI. |
| **RQ-013** | The speed of the system during any operation connected to both UI and database should not exceed two seconds | Non-  Functional | performance | Must Have |  |