



RevRentals – Final Report

University of Calgary
CPSC 471: Database Management Systems
December 9, 2024

Group #18
Fion Lei – 30134327
Peter Tran – 30160145
Kaylee Xiao – 30173778
Kai Ferrer – 30078143

Table of Contents

Table of Contents	3
Abstract	5
Project Proposal	6
Introduction	6
Problem Definition	6
Proposed Solution	7
Project Design: End Users	8
User	8
Listing an Item	9
Admin	10
Project Design: Extended Entity-Relationship Diagram	11
Diagram	11
Assumptions	11
Changes from Implementation	12
Implementation: Relational Model Diagram	14
Diagram	14
Algorithm	15
Changes from Implementation	16
Implementation: General Application Hierarchical Structures	18
Hierarchical Input Process Output Diagram	18
Changes from Implementation	18
Data Flow Diagrams	19
Implementation: Database Management System	22
User SQL Statements	22
Admin SQL Statements	29
User Manual	31
System Prerequisites	31
Setting Up the Environment	31
Running the Emulator	31
Running the Server	31
Running the App	31
General User Manual	33
How to Login as User	33
How to Register New User	33
How To Complete Profile Details	34
How To Edit Profile Details	34
How To Navigate the Main Menu Screen	35
Gear Widget Screen	35
Lot Widget Screen	36

Motorcycle Filter Menu Screen:	36
Gear Filter Menu Screen:	37
Motorcycle Listing Details Page:	37
Vehicle Maintenance Records Page:	38
Viewing The Gear Details Page:	38
Navigating Garage (Rented Items):	40
Viewing Rental Details	41
Navigating Garage (Listed Items):	41
How to Add a New Motorcycle Listing:	42
How to Add a Maintenance Record:	42
Administration Access	44
How to Log in as Admin:	44
Home Page as Admin:	44
How to View Reservations:	45
How to View Reservation Details:	45
How to View Transaction and Agreement Details:	46
How to View Lot Listings:	46
How to Add a Lot to the Marketplace:	47
How to Edit Existing Lot Details:	47
Appendix A: Original EERD	48
Appendix B: Original RM	49
Appendix C: Original HIPO	50
Appendix D: Original DFD (High Level)	51
Appendix E: Original DFD (Level-0)	52
References:	53
GitHub Frontend Repository & Source Code	53
Github Backend Repository & Source Code	53

Abstract

The current motorcycle rental landscape in Canada presents significant limitations for many motorcycle enthusiasts, owners, and riders which prevents them from fully enjoying and appreciating the sport. Many people find owning a motorcycle impractical since they perceive them as seasonal vehicles and find difficulty in justifying it as storage and other costs must be considered. Even more so, despite the interest, some individuals simply do not wish to invest so much money into buying, maintaining, and overall owning a bike. As a result, **RevRentals** is a mobile application that offers an easy and flexible peer-to-peer motorcycle rental service, backed by a strong database that stores crucial information for users.

RevRentals is a specialized solution in reducing the barrier for Canadian motorcycle enthusiasts and targets two end users, Users and Administrators (Admin). Users can act as either a seller and/or a renter using one profile. They can browse motorized vehicles and gear, and add listings or send rental requests. Admins can manage and oversee agreements and storage lots, which is part of our solution to offer easy services for those who enjoy riding. These actions are possible because we created our app using the Flutter framework for front-end, ensuring a smooth user experience. Moreover, the back-end is built on a Microsoft Azure server, developed using a Django framework, and operational efficiency is implemented using SQL queries.

Project Proposal

Introduction

The thrill of riding and owning a motorcycle comes with commitment—something many people, especially in Canada, find it challenging to justify. Motorcycles require insurance, special gear, essential routine maintenance expenses, and seasonal storage as the Canadian winters make it difficult to use year-round. Many enthusiasts and new riders are often faced with the challenge of deciding whether or not to commit to owning a bike in Canada as the investments are costly and the period of use is limited to the spring and summer months. As a result, **RevRentals** is a mobile application solution that offers an easy and flexible peer-to-peer motorcycle rental service, backed by a strong database that stores crucial information for users. This report will describe the issue of the lack of motorcycle rental services in Canada, our solution, and an in-depth strategy of the design and creation of **RevRentals**.

Problem Definition

The current motorcycle rental landscape in Canada presents significant limitations for many motorcycle enthusiasts, owners, and riders which prevents them from fully enjoying and appreciating the sport. Many people find owning a motorcycle impractical since they perceive them as seasonal vehicles and find difficulty in justifying it as storage and other costs must be considered. Even more so, despite the interest, some individuals simply wish to invest less money into buying, maintaining, and overall owning a bike. Rental providers for this issue, especially in Canada, are scarce and difficult to find as many are skeptical of possible scams and unsafe processes. Riding is also simply not as popular in Canada due to the harsh Canadian winters.

Currently, there are limited rental choices available in Canada; the majority are US-based businesses like Eagle Share and Riders Share. Although these systems provide web-based rentals, they need to address the unique requirements of Canadian riders, such as the absence of mobile-friendly options, and winter storage demands. As a result, these limitations exist and further restrict the options available to owners and renters who wish to best utilize motorcycle usage in Canada.

Proposed Solution

We aim to address this gap through a smartphone app called “**RevRentals**” which functions as a peer-to-peer marketplace where motorcycle owners can rent out their motorcycles and equipment on flexible terms. RevRentals has an interactive dashboard for users to browse, list, and rent motorized vehicles and/or gear, as well as allow owners to search for budget-friendly winter storage lots. A smartphone app will address the need for easy, quick access, on-the-go rental solutions for Canadian riders. The ability to search for winter storage lots will also allow motorized vehicle owners easy storage solutions for their vehicles. Our mission is to provide a convenient, cost-effective, and adaptable rental experience that will allow both owners, enthusiasts and new riders an avenue to experience motorcycle riding without barriers.

Project Design: End Users

User

Our platform is built with the needs of the Canadian motorcycle community in mind, with an emphasis on making the rental experience simple and straightforward to use. Our app caters to end users which include **Users** and **Administrators (Admin)**. To ensure a seamless experience for our target audience, Users are able to list their items for rent, request to rent, or both.

In general, Users can create accounts and log in, browse the marketplace, add or list vehicles and/or gear for rent, or request to rent. All Users start through the login page where three options are available: Log In, Sign Up or Sign In as an Admin. For the purposes of the User experience, we will focus on a User's Log In and Sign Up process. If a User does not have an account, they select the "Sign Up" button on the login page and create an account. They will be asked for a username, email, password and password confirmation. Upon entering this information, they are redirected to a page to enter their personal details such as first name, last name, address, and license number.

Once a User has created an account, they can log in using the "Log In" selection on the homepage. Upon successful login and authentication, Users are directed to the homepage where they can view the marketplace, gear, and garage, access notifications and profile details or log out. The marketplace page is standard for all Users, where they can browse all available motorcycle listings. Users can also browse the gear page which includes all available gear. Depending on the User and their needs, they can also browse through the motorcycle and gear pages by filtering the listings or utilizing the sort function to search for a specific item. All Users are also able to view their profile details which include their username, first name, last name, email, license, and postal code. Users will also have a notifications page, which displays any updates on incoming or outgoing rental requests.

As RevRentals caters to different needs of individuals, Users can have two different intentions for going on the app. One group of Users can look to browse and rent motorized vehicles and/or

gear. From henceforth, this group of Users can be sub-classified as “Renters”. Renters start by browsing the marketplace in search of a motorized vehicle or gear to rent. The Renter may choose to browse the motorcycle listings by filtering the listings through the “Filter” button. They can filter the listings by vehicle type, price, color, model, mileage, insurance, and specific attributes depending on the vehicle type selected. They can also choose to sort the listings depending on price using the “Sort” option. Filtering and sorting can be used together, or independently. Renters may also browse the gear listings by filtering through gear type, brand, size, material, and price and/or sorting by price.

Renting an Item

Once Renters find an item they like, they are able to click and view the item in more detail and put in a rental request. Renters may trigger a rental request by selecting a start and end date in which they would like to rent the specific item. This is directly done on the item’s details page. If the Renter is satisfied with the date they have chosen, they can confirm this by clicking the “Rent” button which allows the system to send the seller a rental request.

This information will get relayed to another User, who has the intention to rent out their items. From here, this group of Users can be sub-classified as a Seller. The Seller can review the Renter’s request, accept the rental request, and the Renter will be notified of an agreement for review. They can also view the Seller’s information. The agreement includes a rental overview, damage compensation details, and an agreement fee. The Renter can choose to agree or disagree with the rental agreement by selecting an option of ‘agree’ or ‘disagree’. If the Renter accepts, the Renter will be sent to a transaction page where they will proceed with payment. After this process, the Renter can see the item that they rented in their garage.

Listing an Item

As mentioned previously, some Users are on our app looking to list motorized vehicles and gear, or store vehicles within a lot. These Users are sub-classified as Sellers, who can list vehicles by navigating to their Garage and adding a listing using the “+” button. A form will prompt the Seller to select options to list a Motorized Vehicle or Gear. Depending on the choice, they are prompted to input details relating to their item. If ‘Motorized Vehicle’ is chosen, vehicle type, specific vehicle attribute (cargo racks for mopeds, dirt bike type for dirtbikes, or engine type for motorcycles), VIN, registration, model/vehicle name, price, and color details will be asked. If

‘Gear’ is chosen, gear type (helmet, pants, gloves, etc.), gear name, brand, size, and price details will be asked.

Once a Seller completes the form, they may press the “Add Listing” button to list the item in the marketplace. Should another user find their item and wish to rent, the Seller who listed the item will be notified of a rental request through the Notifications page, which can be accessed via the notification bell icon on the User homepage. Sellers can review the rental request which will include rental dates and cost, as well as view the Renter’s information. After reviewing, the Seller can approve or deny the request, where the system will automatically relay the decision to the Renter as a notification. From there, the Renter who submitted the rental request will be prompted with the agreement and transaction page, which contains information such as the agreement fee, damage compensation, and rental overview. Now, the rented item is approved and paid for, which will be displayed in the ‘Rented’ tab under the Renter’s garage.

Admin

The Admin is required to log in via the Admin login page, and upon successful login, sends the Admin to the admin home page. On this home page, there are two functionalities: (1) Lots and (2) Reservations. Under the ‘Lots’ navigation, the admin can view the listed storage lots, edit the storage lot, and add a storage lot listing. Admin can click on a specific storage lot, which allows the Admin to update the address or rental price. The Admin adds a new storage lot via the ‘+’ button, which prompts the Admin to enter the address and rental price details. Under the ‘Reservation’ option, the Admin can view all existing reservations, as well as click on a specific reservation to see more details. If the reservation status is paid, this indicates the rental request has been approved by the Seller and paid for by the Renter. When a reservation is paid, it generates a transaction and agreements record. The Admin is then able to view the transaction and agreement details for a specific reservation.

Project Design: Extended Entity-Relationship Diagram

Diagram

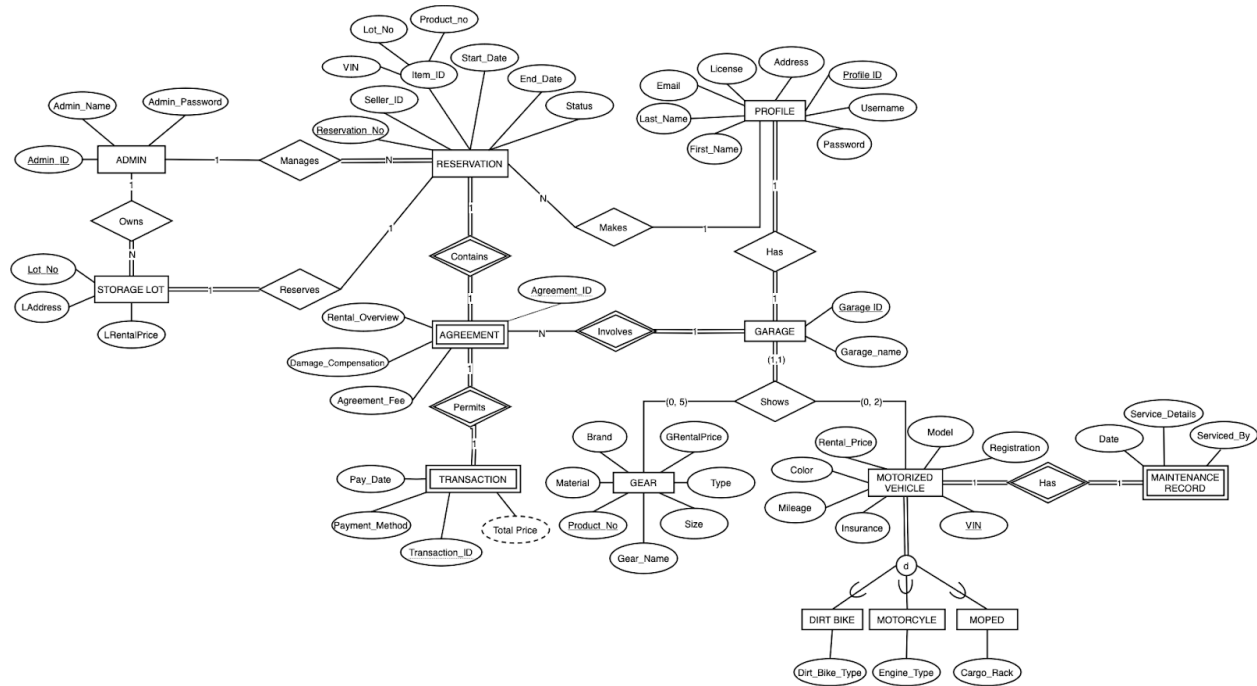


Figure 1. RevRentals Extended Entity-Relationship Diagram

Assumptions

Our assumptions for our Extended Entity-Relationship Diagram (EERD) are as follows:

- ADDRESS is the user's postal code.
- Every profile will have a GARAGE, it's a virtual garage that contains the user's rented or listed items from the marketplace.
- One PROFILE can only have a maximum of 5 pieces of GEAR and a maximum of 2 bikes (MOTORIZED VEHICLE, hereafter, VEHICLE) in their GARAGE.
- AGREEMENT and TRANSACTION do not exist without RESERVATION.
- The ADMIN is the company, not an individual person.
- A MOTORIZED VEHICLE has strictly one MAINTENANCE RECORD that contains all service history records.
- Users can store any rented VEHICLE and/or GEAR in their reserved STORAGE LOT.
- PROFILE can only reserve 1 STORAGE LOT.
- ADMIN oversees and has access to all RESERVATION details.

- A TRANSACTION can only be done after an AGREEMENT between two USERS (a seller and a renter) is signed.
- Damage compensation is determined by the ADMIN.
- Product # attribute of GEAR is generated by the system and is unique to our database.
- Total_Price is calculated using the database entries and shown in the front-end user's homepage.

Changes from Implementation

We have made some changes from our original EERD model (Appendix A). Some changes from implementation are:

1. We omitted the SELLER/RENTER subclasses and made a single PROFILE entity so that a user can rent and/or list items without having to make a separate account. The purpose of having two subclasses was to distinguish the two actions of renting and listing, however for ease of users and user experience, we decided to combine this into one PROFILE entity. Separating SELLER_ID and RENTER_ID also posed some complications when implementing the rental and listing process, so omitting the subclasses was the most beneficial move for both the system and the user.
2. We removed the Favorite_Listing attribute. During the implementation of the PROFILE and marketplace, we realized we needed a separate table to store a user's favorite listings and this was not feasible due to the time constraints of the project. This feature was also not a core feature to our solution, therefore we chose to omit it.
3. For better communication throughout the rental process, we added the Status attribute to RESERVATION which indicates a status of a user's rental request, (hereafter, reservation). This allows the system to send notifications between the users for the rental process. The status options include: Pending Approval, Approved, Rejected, and Paid.
4. Although we removed the SELLER/RENTER entities, we added an explicit Seller_ID attribute to RESERVATION. In order to identify which user was renting their vehicle or gear, we needed to incorporate the items' profile identification to send notifications.
5. For a secure database design, we added Admin_Password to the ADMIN entity. We needed an admin password to authenticate the admin and protect our database.
6. We included Gear_Name attribute to GEAR entity, as we needed a name for the gear to place on listings for users.

7. We included the Model attribute to MOTORIZED_VEHICLE entity which says the model of a listed vehicle. This was also crucial as we needed a name for the vehicle to place on listings for users.
8. We added the LRentalPrice attribute to the STORAGE_LOT entity. We were missing this attribute before, meaning lot rentals had no price. Adding this attribute allows users to choose and rent a lot with more details to support their decision.
9. We added a foreign key of Item_ID that is composed of VIN, Product_No and Lot_No to RESERVATION.
10. We changed the name of the attribute of DIRTBIKE to Dirt_Bike_Type instead of Terrain_Type for clarity.
11. We changed the Color attribute of MOTORIZED_VEHICLE to be single attribute, instead of a multi-valued attribute. Implementing the interface to take multiple attributes from the user on the front-end was not feasible due to the time constraints of the project.

Implementation: Relational Model Diagram

Diagram

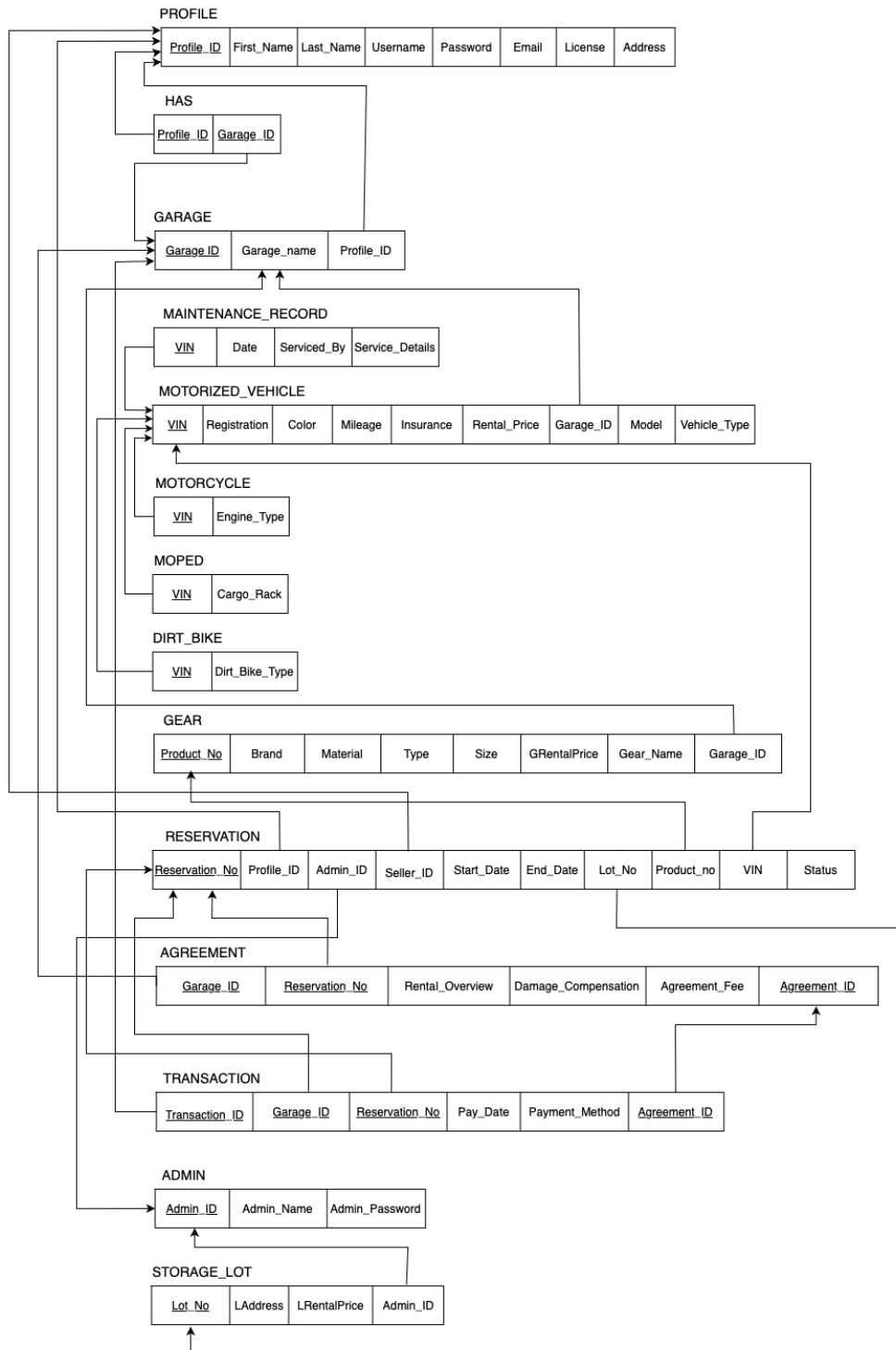


Figure 2: RevRentals Relational-Model Diagram

Algorithm

Our Relational-Model (RM) diagram is above, which is translated from our EERD. The algorithm and steps we took are as follows:

1. Map regular entity types
 - a. We created relations that includes all the simple attributes of regular entities: PROFILE, GARAGE, MAINTENANCE_RECORD, MOTORIZED_VEHICLE, MOTORCYCLE, MOPED, DIRT_BIKE, GEAR, RESERVATION, ADMIN, and STORAGE_LOT.
2. Map weak entity types
 - a. Then, we added each weak entity type, MAINTENANCE RECORD, AGREEMENT and TRANSACTION and their simple attributes.
 - b. We also included the primary key, VIN to MAINTENANCE_RECORD as a foreign key and Garage_ID and Reservation_No to AGREEMENT as foreign keys. Likewise, we included the primary keys, Agreement_ID, Garage_ID, and Reservation_No to TRANSACTION as foreign keys. Including foreign keys for weak entities AGREEMENT and TRANSACTION is crucial for mapping weak entity types from our EERD into our RM.
3. Map binary 1:1 relationship types
 - a. We mapped the following 1:1 relationship types:
 - i. RESERVATION to STORAGE LOT
 - ii. RESERVATION to AGREEMENT
 - iii. AGREEMENT to TRANSACTION
 - iv. PROFILE to GARAGE
 - v. MOTORIZED VEHICLE to MAINTENANCE RECORD.
 - b. To do so, we chose one of the relations to include a foreign key of the primary key of another entity. For example, RESERVATION includes the primary key of STORAGE_LOT, Lot_No, as a foreign key.
4. Map binary 1:N relationship types
 - a. We mapped the following 1:N relationship types:
 - i. PROFILE to RESERVATION
 - ii. GARAGE to AGREEMENT
 - iii. GARAGE to MOTORIZED VEHICLE

- iv. GARAGE to GEAR
- b. To achieve this, we added the primary key of the entity participating in the relationship as a foreign key. For example, RESERVATION includes the primary key of PROFILE (Profile_ID) as a foreign key because one PROFILE can make n RESERVATIONS. Likewise, AGREEMENT includes the primary key of GARAGE (Garage_ID).

Changes from Implementation

As mentioned in the previous section regarding change in our EERD, our group effectively applied these changes to our RM as well, with notable changes being:

1. Removal of SELLER and RENTER entities, effectively removing the SELLER and RENTER database.
2. Adding and/or changing the following attributes:
 - a. Adding a Seller_ID to RESERVATION to track which user is renting out their items. Seller_ID is directly associated with the general Profile_ID.
 - b. Adding Status to RESERVATION so that users can receive communications regarding the rental process.
 - c. Adding Admin_Password to ADMIN for secure admin authentication.
 - d. Adding Gear_Name to GEAR and Model to MOTORIZED VEHICLES to allow the front-end to display names of items to users in the marketplace.
 - e. Adding LRentalPrice to STORAGE LOT so that users can receive more information on which lots they would like to rent.
 - f. Changing the name of DIRTBIKE's Terrain_Type attribute to Dirt_Bike_Type for clarity.
 - g. Removing Favorite_Listing for users as this would have required a new database to store multiple listings that users save.

In addition to these changes, we have the following assumptions:

- ADDRESS is the user's postal code.
- AGREEMENT has a partial key of Agreement_ID as it is a weak entity, and TRANSACTION also relies on AGREEMENT.

- In the RESERVATION table, VIN, Lot_No, and Product_No attributes can be null to ensure that only one of these fields are populated per reservation, corresponding to the specific type of item being reserved (vehicle, lot, or gear).

Implementation: General Application Hierarchical Structures

Hierarchical Input Process Output Diagram

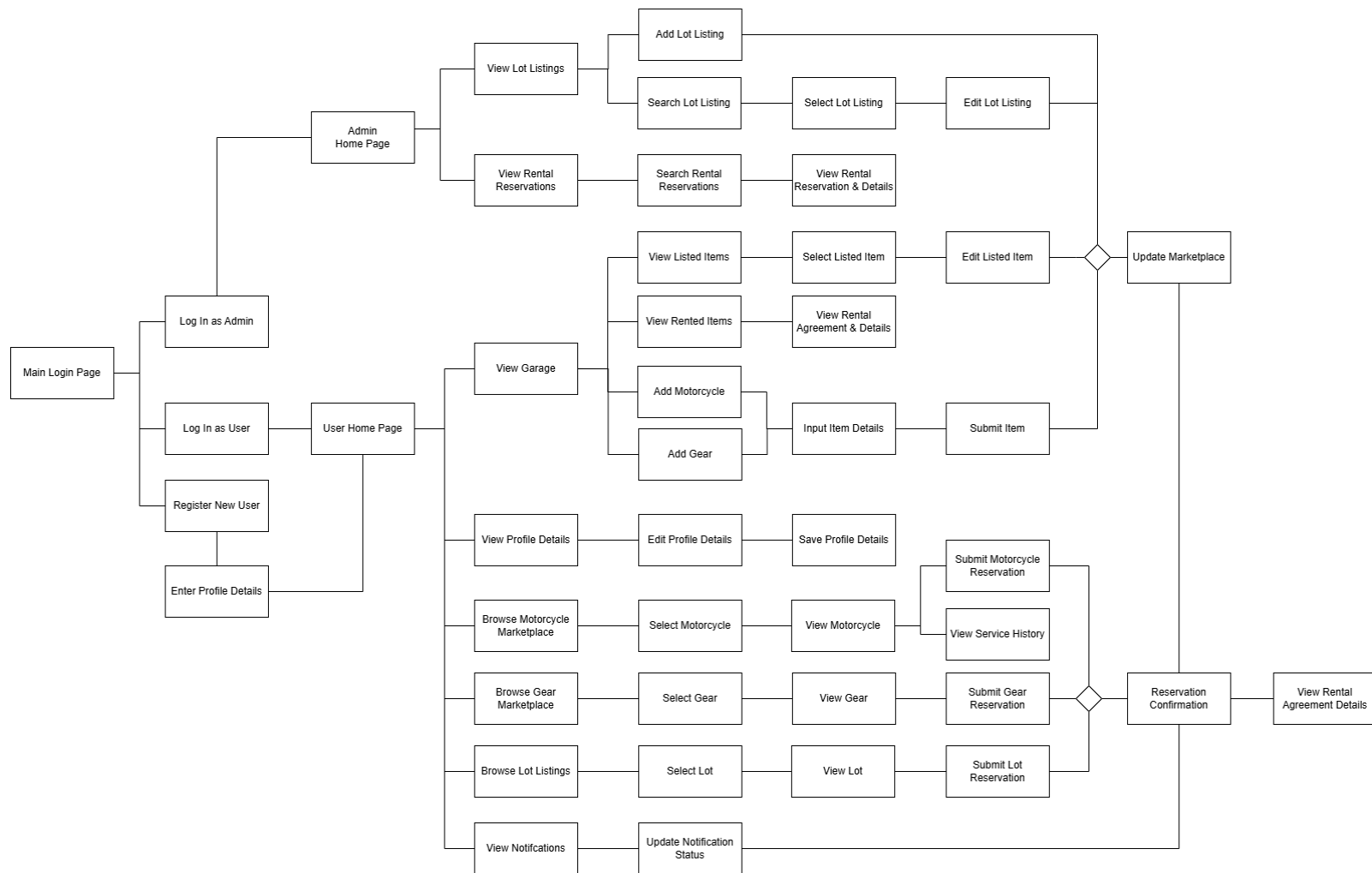


Figure 3. RevRentals Hierarchical Input Process Output Diagram

Changes from Implementation

Our Hierarchical Input Process Output (HIPO) Diagram involved only minor changes. Firstly, our original process for users who wish to add a listing involved being able to edit and remove a listed item. Due to the time constraints of the project, we omitted the removal of a listed item. In future developments of our app, we hope to implement this feature. Secondly, our admin process originally included the admin being able to view a rental reservation, search reservations, select a reservation and then update the rental request (reservation) status. We chose to remove the admin's ability to update the reservation status as we leave it up to the system to notify the user

once a rental request has been approved by another user (seller). Having the admin update the status was an extra step and in hindsight, was irrelevant to the rental process as it is mainly an interaction between two users.

Data Flow Diagrams

Context Diagram (High-level)

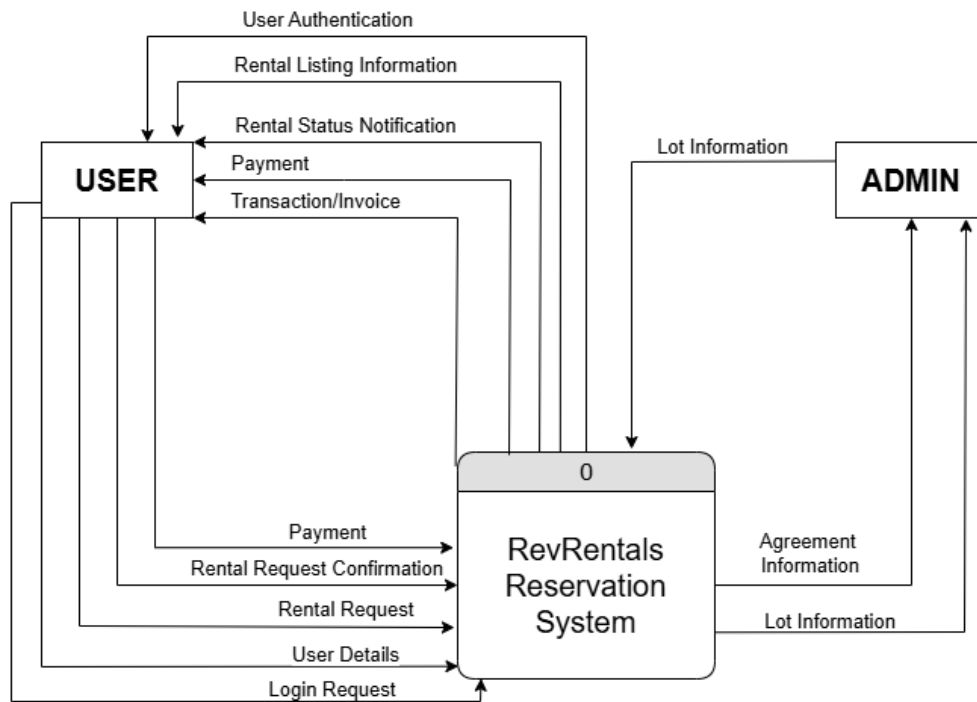


Figure 4. RevRentals High Level Data-Flow Diagram

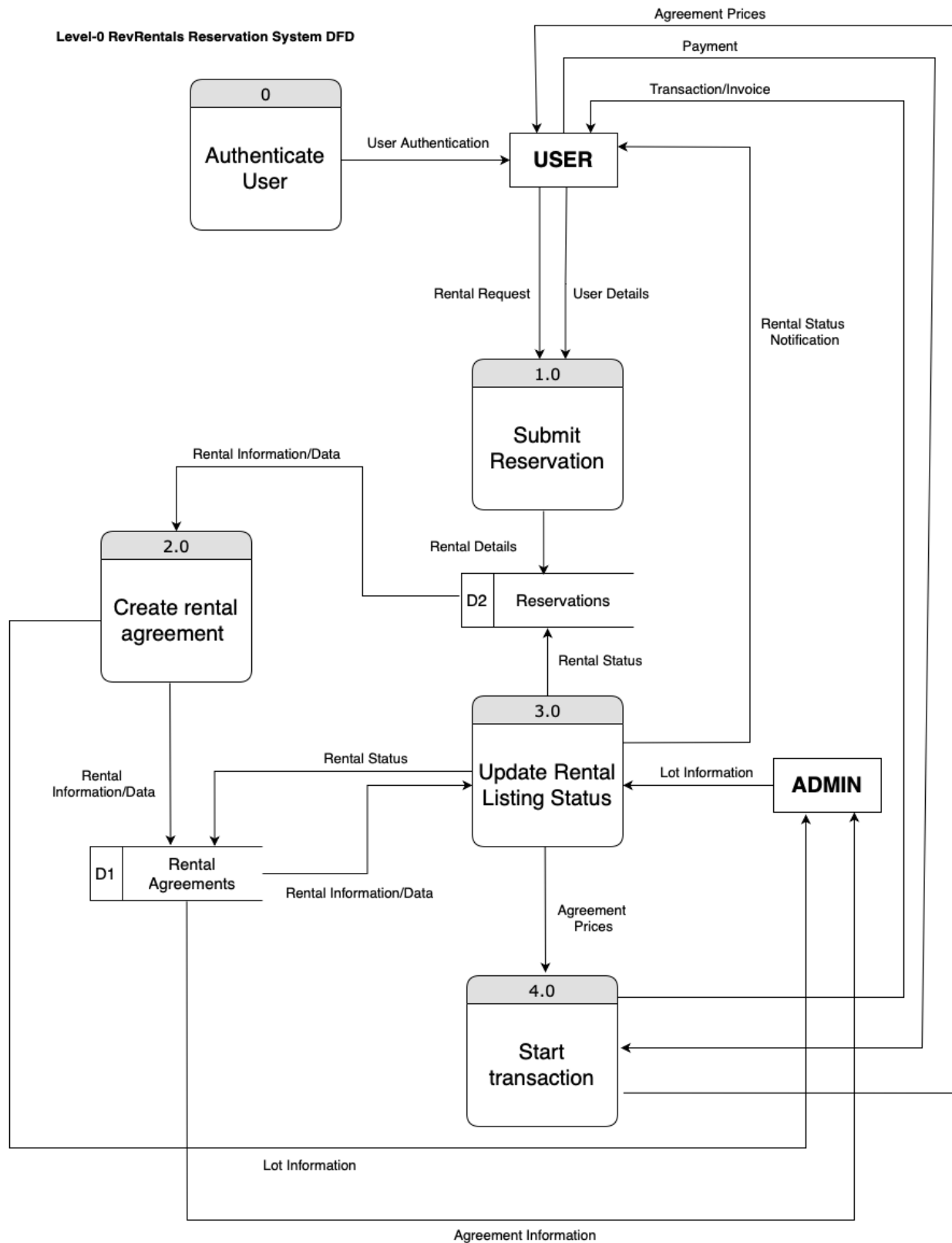


Figure 5. RevRentals Level-0 Data-Flow Diagram

Changes from Implementation

Our Data-Flow Diagrams (DFD) had minor changes from the original (Appendix D and Appendix E). Figure 4 shows our updated High Level DFD. Since the Admin is no longer responsible for managing and approving rental requests, we adjusted the diagram to reflect how the system receives data when the Admin manages lot listings, and how the Admin end user receives agreements data as well. Figure 5 shows our updated Level-0 DFD. Likewise, since the Admin is no longer responsible for managing and improving rental requests, we also adjusted this diagram so that the Admin only receives lot and agreement information, as well as sends lot listing information to the system.

Implementation: Database Management System

User SQL Statements

Insert New Profile

```
INSERT INTO profile (First_Name, Last_Name, Email, License,
Username, Password, Address)
VALUES ('FirstName', 'LastName', 'Email', 'License', 'Username',
'Password', 'Address');
```

Log In User

```
SELECT Profile_ID, First_Name, Last_Name, Email, License,
Username, Password, Address FROM profile
WHERE Username = 'enteredUsername' OR Email = 'enteredEmail';
```

Check If Username Exists

```
SELECT COUNT(*) FROM profile
WHERE Username = 'enteredUsername';
```

Check If Email Exists

```
SELECT COUNT(*) FROM profile
WHERE Email = 'enteredEmail';
```

Update Profile Details

```
UPDATE profile
SET First_Name = 'UpdatedFirstName', Last_Name =
'UpdatedLastName', License = 123456-789, Address =
'UpdatedAddress'
WHERE Profile_ID = ProfileID;
```

View Single User

```
SELECT Profile_ID, First_Name, Last_Name, Email, License,
Username, Password, Address
FROM profile
```

```
WHERE Username = 'enteredUsername' OR Email = 'enteredEmail';
```

View Profile Details

```
SELECT Profile_ID, Username, First_Name, Last_Name, Email,  
License, Address  
FROM profile  
WHERE Profile_ID = ProfileID;
```

Marketplace SQL Statements

View All Motorized Vehicles

```
SELECT * FROM motorized_vehicle;
```

View all gear items

```
SELECT * FROM gear;
```

View all storage lots

```
SELECT *  
FROM storage_lot;
```

Insert a reservation

```
INSERT INTO reservation(Profile_ID, Admin_ID, Product_no, VIN,  
Lot_No, Start_Date, End_Date, Status, Seller_ID)  
VALUES (ProfileID, AdminID, ProductNo, VIN, LotNo, StartDate,  
EndDate, 'status', SellerID);
```

View a reservation by reservation number

```
SELECT r.Reservation_No, r.Start_Date, r.End_Date, p.First_Name,  
p.Last_Name, r.VIN, r.Product_No, r.Lot_No, r.Status  
FROM reservation r  
JOIN profile p ON r.Profile_ID = p.Profile_ID  
WHERE r.Reservation_No = ReservationNo;
```

Update a reservation

```
UPDATE reservation
SET Status = 'Paid'
WHERE Reservation_No = ReservationNo;
```

Insert an agreement

```
INSERT INTO agreement (Agreement_ID, Reservation_No, Garage_ID,
Rental_Overview, Damage_Compensation, Agreement_Fee) VALUES
(AgreementID, ReservationNo, GarageID, 'RentalOverview',
Damage_Compensation, AgreementFee);
```

View an agreement by reservation number

```
SELECT Agreement_ID, Garage_ID, Rental_Overview,
Damage_Compensation, Agreement_Fee
FROM agreement
WHERE Reservation_No = ReservationNo;
```

Insert a transaction

```
INSERT INTO transaction (Transaction_ID, Agreement_ID,
Garage_ID, Reservation_No, Pay_Date, Payment_Method)
VALUES (TransactionID, AgreementID, GarageID, ReservationNo,
'NOW()', PaymentMethod);
```

View a transaction by reservation number

```
SELECT t.Transaction_ID, t.Agreement_ID, t.Garage_ID, t.Pay_Date,
t.Payment_Method, a.Agreement_Fee
FROM transaction t
JOIN agreement a ON t.Agreement_ID = a.Agreement_ID
WHERE t.Reservation_No = ReservationNo;
```

View filtered motorized vehicles

```
SELECT DISTINCT MV.VIN, MV.Garage_ID, MV.Registration,
MV.Rental_Price, MV.Color, MV.Mileage, MV.Insurance, MV.Model,
```

```

MV.Vehicle_Type, M.Engine_Type, Mo.Cargo_Rack,
DB.Dirt_Bike_Type,MR.SERVICE_DETAILS
FROM motorized_vehicle AS MV
LEFT JOIN motorcycle AS M ON MV.VIN = M.VIN
LEFT JOIN moped AS Mo ON MV.VIN = Mo.VIN
LEFT JOIN dirtbike AS DB ON MV.VIN = DB.VIN
LEFT JOIN maintenance_record AS MR ON MV.VIN = MR.VIN
WHERE MV.Mileage <= 10000 AND "MV.Rental_Price <= 100 AND
M.Engine_Type = "Any" AND Mo.Cargo_Rack= "Any" AND
DB.Dirt_Bike_Type= "Any" AND MV.Insurance= "Any" AND
MV.Vehicle_Type = "Any" AND MV.Color = "Any";

```

View filtered gear items

```

SELECT * FROM GEAR
WHERE BRAND = 'ANY' AND MATERIAL = 'ANY' AND GEAR_TYPE = 'ALL'
AND SIZE = 'ANY' AND GRENTALPRICE <= 100;

```

Garage SQL Statements

View listed garage items

```

SELECT mv.VIN, mv.Registration, mv.Rental_Price, mv.Color,
mv.Mileage, mv.Insurance, mv.Model, mv.Vehicle_Type
FROM motorized_vehicle AS mv
WHERE mv.Garage_ID = GarageId;

```

```

SELECT g.Product_No, g.Brand, g.Material, g.Type, g.Size,
g.GRentalPrice, g.Gear_Name
FROM gear AS g
WHERE g.Garage_ID = GarageId;

```

View rented garage items

```

SELECT r.Reservation_No, r.Start_Date, r.End_Date, r.Status,
t.Pay_Date AS Transaction_Date, t.Payment_Method,
a.Rental_Overview, a.Damage_Compensation, a.Agreement_Fee,

```



```

COALESCE(mv.Model, g.Gear_Name, sl.LAddress) AS Item_Name,
mv.VIN, g.Product_No, sl.Lot_No AS Storage_Lot_No
FROM reservation r
LEFT JOIN agreement a ON r.Reservation_No = a.Reservation_No
LEFT JOIN transaction t ON a.Agreement_ID = t.Agreement_ID
LEFT JOIN motorized_vehicle mv ON r.VIN = mv.VIN
LEFT JOIN gear g ON r.Product_No = g.Product_No
LEFT JOIN storage_lot sl ON r.Lot_No = sl.Lot_No
WHERE r.Profile_ID = ProfileID AND r.Status = 'Paid';

```

Insert a motorized vehicle

```

INSERT INTO motorized_vehicle (VIN, Garage_ID, Registration,
Rental_Price, Color, Mileage, Insurance, Model, Vehicle_Type)
VALUES ('VIN', GarageID, 'Registration', RentalPrice, 'color',
Mileage, 'Insurance', 'Model', 'VehicleType');

```

Insert maintenance records

```

INSERT INTO maintenance_record(VIN, DATE, SERVICED_BY,
SERVICE_DETAILS)
VALUES (VIN, date, ServiceBy, ServiceInformation);

```

Insert a gear item

```

INSERT INTO gear(Garage_ID, Brand, Material, Type, Size,
GRentalPrice, Gear_Name)
VALUES (GarageID, 'Brand', 'Material', 'Type', 'Size',
RentalPrice, 'Name');

```

Update motorized vehicle rental price

```

UPDATE motorized_vehicle
SET rental_price = RentalPrice WHERE vin = VIN;

```

Update gear rental price

```

UPDATE gear

```

```
SET grentalprice = RentalPrice
```

```
WHERE product_no = ProductNo;
```

View seller notification

```
SELECT r.Reservation_No, r.Start_Date, r.End_Date, r.Status,  
p.First_Name AS renter_first_name, p.Last_Name AS  
renter_last_name,  
COALESCE(mv.Model, g.Gear_Name, s.LAddress) AS item_name  
FROM reservation r  
JOIN profile p ON r.Profile_ID = p.Profile_ID  
LEFT JOIN motorized_vehicle mv ON r.VIN = mv.VIN  
LEFT JOIN gear g ON r.Product_No = g.Product_No  
LEFT JOIN storage_lot s ON r.Lot_No = s.Lot_No  
WHERE r.Seller_ID = SellerID AND r.Status = 'Pending Approval';
```

View buyer notification

```
SELECT r.Reservation_No, r.Start_Date, r.End_Date, r.Status,  
COALESCE(mv.Model, g.Gear_Name, sl.LAddress) AS item_name,  
s.First_Name AS seller_first_name, s.Last_Name AS  
seller_last_name  
FROM reservation r  
LEFT JOIN motorized_vehicle mv ON r.VIN = mv.VIN  
LEFT JOIN gear g ON r.Product_No = g.Product_No  
LEFT JOIN storage_lot sl ON r.Lot_No = sl.Lot_No  
JOIN profile s ON r.Seller_ID = s.Profile_ID  
WHERE r.Profile_ID = %s AND r.Status IN ('Approved', 'Rejected');
```

Check notifications

```
SELECT COUNT(*)  
FROM reservation r  
WHERE r.Profile_ID = ProfileID AND r.Status IN ('Approved',  
'Rejected');
```

```
SELECT COUNT(*)  
FROM reservation r  
WHERE r.Seller_ID = SellerID AND r.Status = 'Pending Approval';
```

Update reservation request

```
UPDATE reservation  
SET Status = 'Approved'  
WHERE Reservation_No = ReservationNo;
```

```
UPDATE reservation  
SET Status = 'Rejected'  
WHERE Reservation_No = ReservationNo;
```

```
UPDATE reservation  
SET Status = 'Paid'  
WHERE Reservation_No = ReservationNo;
```

Reject a reservation request

```
DELETE FROM reservation  
WHERE Reservation_No = ReservationNo;
```

Admin SQL Statements

View Admin

```
SELECT Admin_ID, Admin_Name, Admin_Password
FROM admin
WHERE Admin_Name = 'AdminName';
```

View Admin ID

```
SELECT Admin_ID
FROM admin
LIMIT 1;
```

View All Reservations

```
SELECT *
FROM Reservation;
```

View Reservation Details by Reservation Number

```
SELECT r.Reservation_No, r.Start_Date, r.End_Date, p.First_Name,
p.Last_Name, r.VIN, r.Product_No, r.Lot_No, r.Status FROM
reservation r
JOIN profile p ON r.Profile_ID = p.Profile_ID
WHERE r.Reservation_No = ReservationNo;
```

View Transaction by Reservation Number

```
SELECT t.Transaction_ID, t.Agreement_ID, t.Garage_ID, t.Pay_Date,
t.Payment_Method, a.Agreement_Fee
FROM transaction t
JOIN agreement a ON t.Agreement_ID = a.Agreement_ID
WHERE t.Reservation_No = ReservationNo;
```

View Agreement by Reservation Number

```
SELECT Agreement_ID, Garage_ID, Rental_Overview,
Damage_Compensation, Agreement_Fee
```

```
FROM agreement  
WHERE Reservation_No = ReservationNo;
```

Insert a Storage Lot

```
INSERT INTO Storage_lot(Admin_ID, LAddress, LRentalPrice)  
VALUES (AdminID, Address, RentalPrice);
```

Edit a Storage Lot

```
UPDATE Storage_Lot  
SET Laddress = 'Address', LRentalPrice = RentalPrice  
WHERE Lot_No = LotNo;
```

User Manual

System Prerequisites

1. To run our mobile app on your system, you need to download the [Flutter SDK](#) on your operating system.
2. Install [Android Studio](#) to run an emulator.
 - a. Open the SDK Manager → ‘SDK Tools’ tab then select the following options and press ‘OK’:
 - Android SDK Command Line Tools
 - Android SDK Platform Tools
 - Android Emulator
 - b. Open Virtual Device Manager and click on ‘Create Virtual Device Manager’, and select the type of Android emulator you want. We suggest ‘Medium Phone’.
3. Install [Visual Studio Code](#) to run our mobile app with the Android Studio emulator.

Setting Up the Environment

1. After installing the system prerequisites, open Visual Studio code and install Flutter and Dart in the Extensions to configure the Flutter SDK path.
2. Once the extensions are enabled and installed, copy our [Front-end Repository](#) and [Back-end Repository](#) into separate Visual Studio Code windows.

Running the Emulator

3. In the front-end window, create a connection to the emulator from the bottom right of the screen, select the Flutter Device option, and click on the emulator that was just created.

Running the Server

4. In the back-end window, follow the instructions in the README.md file.

Running the App

5. Navigate back to the front-end window, open the terminal, and enter the following commands:

```
flutter pub get
flutter run --release
```

General User Manual



Welcome to RevRentals

Create an account or log in to access the app.

Log In Sign Up

Email or Username

Password

Log In

[Sign in as Admin](#)

How to Login as User

To Log in as a general User to our application, you must already have an existing account on our platform. Start by first entering an existing email or username in the “Email or Username” field, and then enter the associated Password to that account in the “Password” field. Once the fields are correctly filled out, press the “Log In” button at the bottom to confirm. If the login was successful (meaning the account exists and all checks pass), then you will have access to our app.



Welcome to RevRentals

Create an account or log in to access the app

Log In Sign Up

Username

Email

Password

Confirm Password

Sign Up

[Sign in as Admin](#)

How to Register New User

To Register a New User into our database, the fields MUST be filled out correctly following our detailed REGEX for each field. Username field can only contain letters and numbers, and must be between 6-16 characters. Email must be a valid email following REGEX requirements (EX: example@email.com). Password must be a minimum of 5 characters, and must include at least one letter and one number. Confirm Password MUST match the field entered in “Password”. Users will be properly prompted if the issue of problem arises prior to adding users to the database. Assuming the user has entered the fields as required, press the “Sign Up” button and you will be properly registered into our system.



Complete Profile Details

First name must contain only letters.

Last name must contain only letters.

Enter your Alberta License Number in format: xxxxxx-xxx

Enter your Canadian postal code in format: A1A 1A1, A1A1A1, or A1A-1A1

Save Details

How To Complete Profile Details

If the user is prompted to complete their profile details on their account, Then they MUST fill out the following form. The fields must follow the REGEX rules in order to successfully finish profile details. First name and Last name must only contain letters, License number must match a 9-digit number of the format “xxxxxx-xxx” (exactly like the number shown on your alberta license card). The Postal code must be a valid Canadian Postal Code in the format of X1X 1X1 or X1X-1X1 or X1X1X1. If these fields are correctly filled out as suggested by the regex, the user can click the “Save Details” button to complete their profile information. The user will be redirected to the marketplace page once complete.

8:39

← Profile

RevRentals
First name must contain only letters.

RevRentals
Last name must contain only letters.

revrentals@gmail.com

revrentals

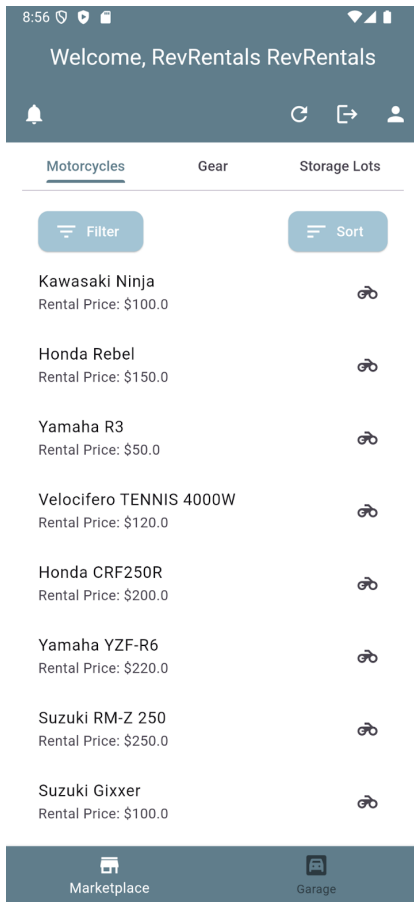
123456-789
Enter your Alberta License Number in format: xxxxxx-xxx

A1A 1A1
Enter your Canadian postal code in format: A1A 1A1 or A1A-1A1

Update Profile

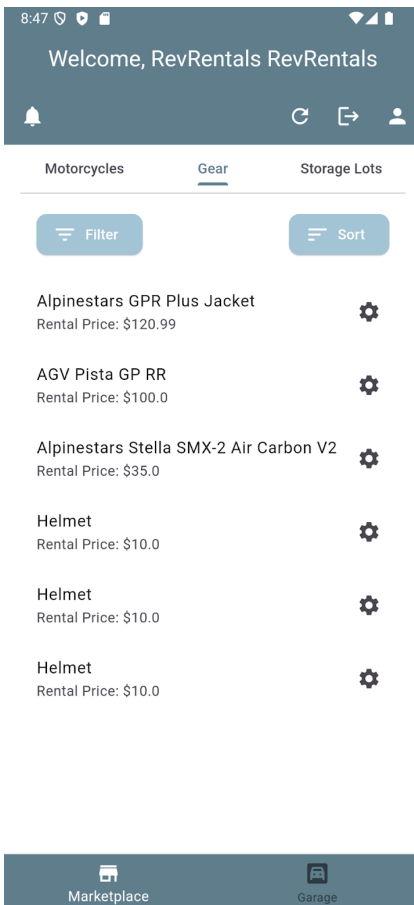
How To Edit Profile Details

If a user desires to change their profile information, we allow that option to. The user can change any information they like, whether it's all information or only 1 while Still following the correct REGEX. If the user wants to Change their first name, they must enter a valid First name and then click the “Update Profile” button. This will successfully update profile details assuming the user has entered the fields correctly. If a user wants to change multiple aspects of their account, they can enter any new information as available. Although we lock the changing of Username and Email. Any errors will be gracefully handled depending on user input.



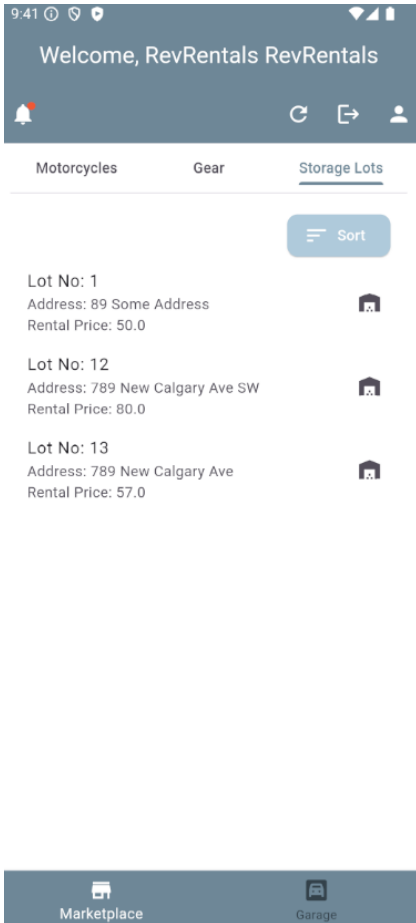
How To Navigate the Main Menu Screen

In our main menu screen, you will be able to access all of our services, the user will automatically be loaded into the motorcycle marketplace that lists all current motorcycles on the market. To view available gear, click the gear widget, to view available storage lots, click the storage lot widget. To view your garage, click the garage widget at the bottom. To view notifications, click the bell icon at the top left. To refresh the page, click the refresh button. To logout, click the logout symbol next to the refresh button. To view profile details, click the “person” icon at the top right. The user will be able to filter, and sort listings, they will also be able to access the listing details by directly clicking the item.



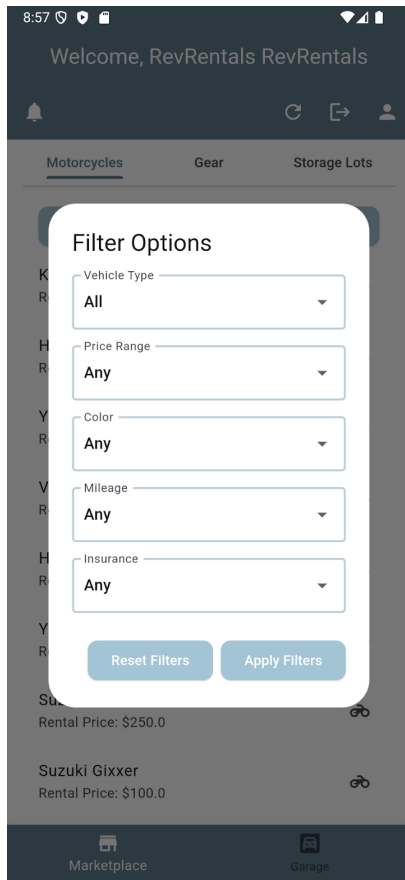
Gear Widget Screen

If the user has accessed the gear widget screen, you will be able to see the listed gears on our marketplace. If the user wants to filter certain items in the marketplace, they can click the filter button or they can click the sort button to sort by price. To view listing details, the user can directly click on the item to see details and rent the listed item. While on this page, the user will still have access to our other services such as notification, garage, etc.



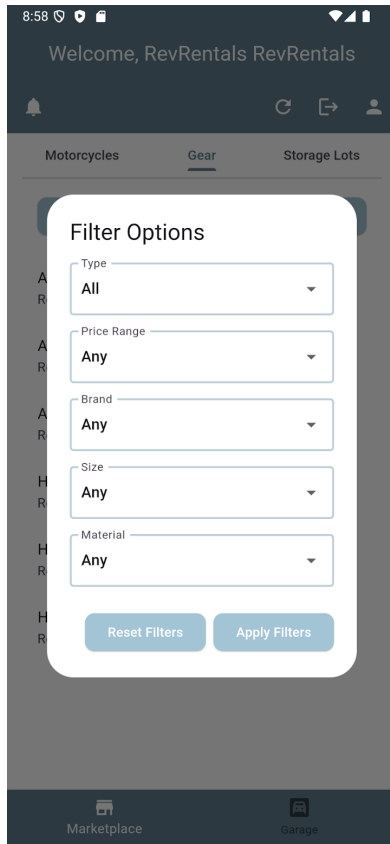
Lot Widget Screen

If the user has accessed the Storage Lots widget screen, you will be able to see listed storage lots by our admin. To view listing details, the user can directly click on the item to see details and rent the listed item. The user can also sort the listings by price by pressing the ‘Sort’ options. While on this page, the user will still have access to our other pages, such as notification, garage, profile details, and the motorcycle and gear marketplace.



Motorcycle Filter Menu Screen:

If the user has accessed the filter option menu, they must select their desired options in the drop down menu. If they have selected their options, they must click the “Apply Filters” button to filter vehicles in the database. If the user desires to return to the original marketplace setting, they can click the “Reset Filters” button to return to the default marketplace settings. The filter screen will also dynamically change depending on the ‘Vehicle Type’ option they have clicked. For example, if ‘Vehicle Type’: ‘Motorcycle’ is selected, a new dropdown will appear that will allow the user to precisely select another filter attribute based on the vehicle.



Gear Filter Menu Screen:

If the user has selected to filter their options in the gear widget screen, the user will be prompted to select the specific fields from a drop down menu. This will include type, price range, brand, size, and material. The user can select one or more options to filter. If the user has selected their options, they can click the “Apply Filters” button in order to confirm their selection and the gear marketplace will automatically update. If the user wants to reset their filters, they can click the “Reset Filters” button to return the gear listings to their default state.



Motorcycle Listing Details Page:

In the vehicle details page, the user will be able to see the motorcycle details such as the vehicle type, color, mileage, and insurance coverage. They will also be able to see the rental price per day. To view the maintenance records of the vehicle, they can click the “view maintenance records” button. To rent this vehicle, the user must first select the start date, then select the end date. They cannot select the other way around, it is enforced that the user must first select the start date. Then they can click the “rent vehicle” button to send a request to the seller for approval.



Date: 2024-11-01

Serviced By: RevRental's Mechanics
Service Details: Oil Change

Date: 2024-11-02

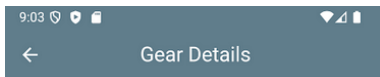
Serviced By: RevRental's Mechanics
Service Details: Fluid Changes

Date: 2024-12-04

Serviced By: RevRental's Mechanics
Service Details: Battery Change

Vehicle Maintenance Records Page:

When the user has clicked the “View Maintenance Records” button they will be able to see all records that were inputted by the seller. These are ordered by no specific date, they are ordered by when they were inputted. It will display the date of maintenance, who it was serviced by, and the specific service details. Otherwise, the user can exit this page by pressing the back button at the top left.



AGV Pista GP RR

Brand: AGV
Material: Plastic
Type: Helmet
Size: M

Rental Price Per Day: \$100.00 CAD

Select Start Date

Select End Date

Rent Gear

Viewing The Gear Details Page:

In the gear details page, the user will be able to see the listings information. This includes the name, the brand, the material, the type, the size, and the rental price. In order to rent this item, they must first select a valid start date before the end date. This is enforced, if the user does not select the start date first, an error message will be prompted to remind the user. If the dates were correctly filled out, the user can send a request to the seller to approve their rental by clicking the “Rent Gear” button.



Viewing the Notification Page:

The notifications page include the following possible message:

- Rental Request for [Vehicle Name]
 - This means another user has requested to rent their item.

It must be approved in order to proceed.

- If the user is prompted for this message, they must either reject the request or approve it.

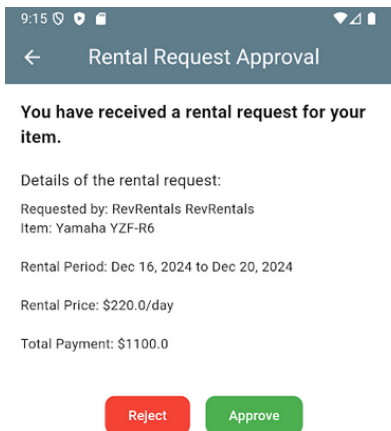
- Rental for [Vehicle Name] (green checkmark)

- This means the seller has approved your request to rent the item and you will be able to proceed with the agreement and transaction.

- Rental for [Vehicle Name] (red checkmark)

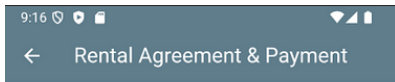
- This means the seller has rejected your request to rent the vehicle. The user must acknowledge the rejection in order to remove the notification.

These objects are clickable and will require further action by clicking on each notification message.



Viewing the Rental Request Page:

If the seller has received a rental request and has clicked the message on the notification screen, they will be taken to this screen where they will be able to see the rental details. This includes the buyer, the rental period, the rental price, and the total payment. The seller will be able to reject or approve the request by either clicking the reject button or the approve button depending on their input. If the reject button is clicked, it will send a notification to the buyer that it was rejected. If the approve button was clicked a notification will be sent to the buyer to finish the agreement and transaction.



Rental Agreement

Item: Yamaha YZF-R6
Sold By: owner ownertest
Rental Period: 2024-12-16 to 2024-12-20
Rental Price: \$220.00 per day
Total Price: \$1100.00

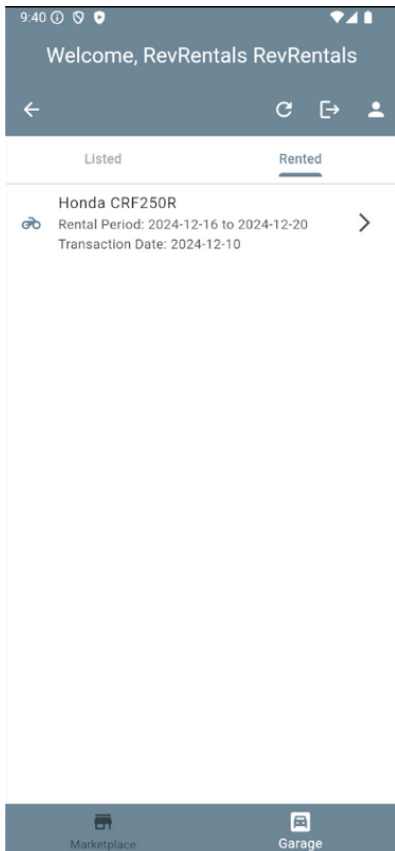
Payment Details

Select a Payment Method

Confirm Payment

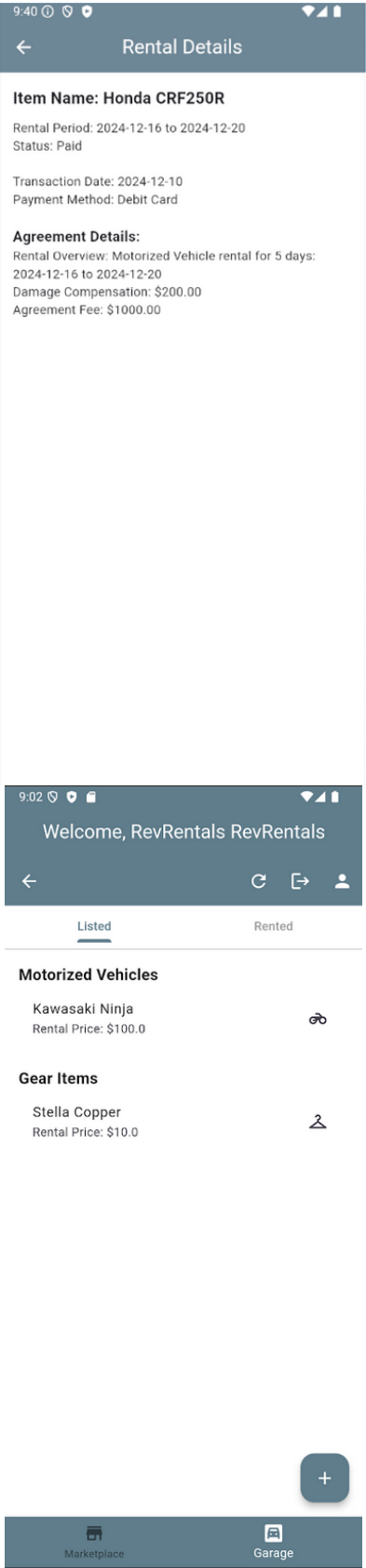
Viewing the Rental Agreement Page

In the Rental Agreement Page, the buyer will be able to finalize their rental. On this page the rental details are given. This includes the item name, who it is sold by, the rental period, the rental price, and the total price. In order to complete the payment process, the buyer must select their payment method, and click the confirm payment button. Once this payment is finished, the user has now officially rented the item. They can see this information in the rented tab of their garage.



Navigating Garage (Rented Items):

If a user has successfully rented out an item, it will appear in the “Rented” tab of the user’s Garage. Both rented items and vehicles will appear in a list. To view the details, the user can press the rented item in the list.

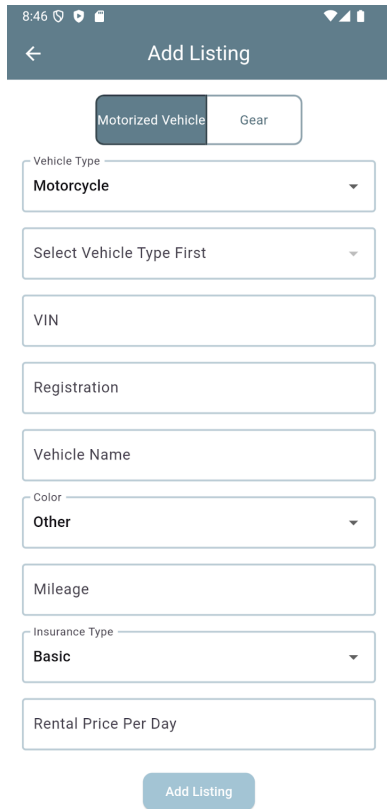


Viewing Rental Details

If a user presses an item in the ‘Rented’ tab of their Garage, they can easily view the specific details of the rented item, as well as the agreement details. The specified item information includes: rental period, status, transaction date, and payment details. The specified agreement details include: a rental overview, damage compensation fee, and an agreement fee.

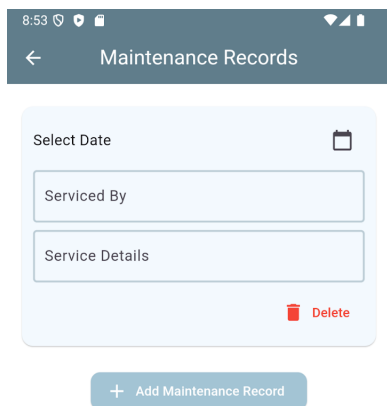
Navigating Garage (Listed Items):

When the user has entered the garage widget page, they will be able to see the vehicle that they own and have listed on their marketplace. This includes both motorized vehicles and gear items. They can directly view these listings by clicking the associated item. To add a new listing, they can click the “+” button at the bottom right. The user is limited to listing 2 motorized vehicles on the marketplace and 5 pieces of gear. Otherwise, the system will deny the user from adding new items. The user will have access to other functionalities like going back to the marketplace, refreshing the page, and etc. If they want to view vehicles they have rented off the marketplace these will be located under the “rented” tab.



How to Add a New Motorcycle Listing:

If the user presses the ‘+’ item in their Garage, they are prompted to the “Add Listing” page. To add a new vehicle listing, ensure the “Motorized Vehicle” option is selected at the top and follow the fields as prompted. They must first select a vehicle type, with options being “Motorcycle”, “Moped”, and “Dirt Bike”. Once this field is selected, the required fields will change. If they select “Motorcycle”, they must specify the engine type. If they select “Moped”, they must specify the number of cargo racks. If they select “Dirt Bike”, they must specify the dirt bike type. The rest of the information will remain the same. Registration must follow a REGEX as well as VIN. These WILL be enforced. Vehicle names will be left to the freedom of the user, although they cannot enter “null”. All fields must be entered correctly in order to add a listing. Once the fields have been filled, click the “Add Listing” button.



How to Add a Maintenance Record:

Once a motorcycle is added to the marketplace, the user will be able to add a maintenance record to their associated vehicle listing. They must select the date of the service, they must specify who serviced the vehicle and the details of the service, for example “Oil Change”, “Suspension Fix”, etc. We leave these fields to be filled out to the freedom of the user, although “null” will be enforced. The user can add multiple maintenance records, if they have more than 1, they can click the “Add Maintenance Record” button to add an additional field to enter a new one. They can also delete a maintenance record by clicking the red “delete” button. The user can also



8:47

Add Listing

Motorized Vehicle Gear

Gear Type
Helmet

Gear Name

Brand
Any

Material
Any

Gear Size
Any

Rental Price Per Day

Add Listing

How to Add a New Gear Listing:

If the user presses the ‘+’ item in their Garage, they are prompted to the “Add Listing” page. To add a new gear listing, ensure the “Gear” option is selected at the top and follow the fields as prompted. Users must complete the following fields: Gear Type, Gear Name, Brand, Material, Gear Size, and rental price. We Leave the naming to the freedom of the user, although they cannot enter “null”. If the Gear Type is “Helmet” or “Boots”, the material is locked to “Kevlar”. Rental pricing must be convertible to a double, otherwise the user will be prompted for an error. All errors are handled gracefully and shown to the user for information on filling out fields.

Administration Access



How to Log in as Admin:

To log in as an administrator, the admin will have to click the ‘Sign in as Admin’ option in the main login interface. The admin will have to fill in their username and password, which is stored in our database. Once the admin clicks ‘Sign In’, the system sends a request to the database to verify the credentials. Once the admin logged in, they will be able to:

- **Manage Storage Lots:** Add or edit storage lots into the database
- **Manage Reservations, Agreements, and Transactions:** view rental reservation details



Sign in as Admin

Admin Username

Password

Sign In



Home Page as Admin:

After logging in, the admin is navigated to the admin home page. The admin can select two options: 1) “Lots” to view storage lots or 2) “Reservations” to view rental reservations. The admin also has a sign-out function to sign them out and navigate to the main login page.



Search by reservation number...

Reservation No: 22
Profile_ID: 102

Reservation No: 23
Profile_ID: 102

Reservation No: 28
Profile_ID: 102

Reservation No: 39
Profile_ID: 102

Reservation No: 91
Profile_ID: 100

Reservation No: 92
Profile_ID: 115

Reservation No: 93
Profile_ID: 115

Reservation No: 95
Profile_ID: 115

Reservation No: 96
Profile_ID: 115

Reservation No: 100
Profile_ID: 115

How to View Reservations:

To view the list of reservations, click the ‘Reservations’ button in the home page. This will then display all available Reservations where, as an admin, you can search reservations by number. To view more details, you can click on a reservation item and this will display specific reservation details.

Reservation No: 23
Start Date: 2024-12-01
End Date: 2024-12-06
Renter First Name: Peter
Renter Last Name: Tran
Item Name: Suzuki RM-Z 250
Rental Price: 250.0
Duration Days: 6
Total Price: 1500.0
Status: Paid

How to View Reservation Details:

After pressing a specific reservation to view the details, the admin can check details such as reservation number, start date of the reservation, end date, and the renter’s information. To view further information on transaction and agreement details, the admin can press the “View Transaction & Agreement Details” button at the bottom of the screen.

View Transaction & Agreement Details

Transaction Id:	23
Agreement Id:	23
Garage Id:	95
Pay Date:	2024-12-02
Payment Method:	Debit Card
Amount Paid:	1500.0
Agreement Id:	23
Garage Id:	95
Rental Overview:	Motorized Vehicle rental for 6 days: 2024-12-01 to 2024-12-06
Damage Compensation:	200.0
Agreement Fee:	1500.0

How to View Transaction and Agreement Details:

If the admin has chosen to view the Transaction and Agreement details, they will be redirected to a new screen. The Transaction and Agreement page will show information such as transaction ID, agreement ID, the garage ID associated with the transaction, payment details and specific rental details. Rental details such as rental overview, damage compensation fees and agreement fees will be displayed.

Search storage lots by lot number...

- Lot No: 1

Address: 89 Some Address
- Lot No: 12

Address: 789 New Calgary Ave SW
- Lot No: 13

Address: 789 New Calgary Ave

How to View Lot Listings:

If the admin has pressed the “Lots” option in the Admin homepage, they will be directed to the Lot Listings page. This page will display a full list of current lots. For ease, the admin can also search storage lots by lot number and the list will automatically display lots applicable to the search. To edit a lot, the admin can press the specific lot listing they wish to edit.





How to Add a Lot to the Marketplace:

If the admin has selected to add a new lot to the marketplace, they will be redirected to the “Add New Lot” page. The admin must input the new lot address as well as the rental price. Once these details have been added, they can add the lot to the available list of rental lots by pressing the “Add Lot” button.

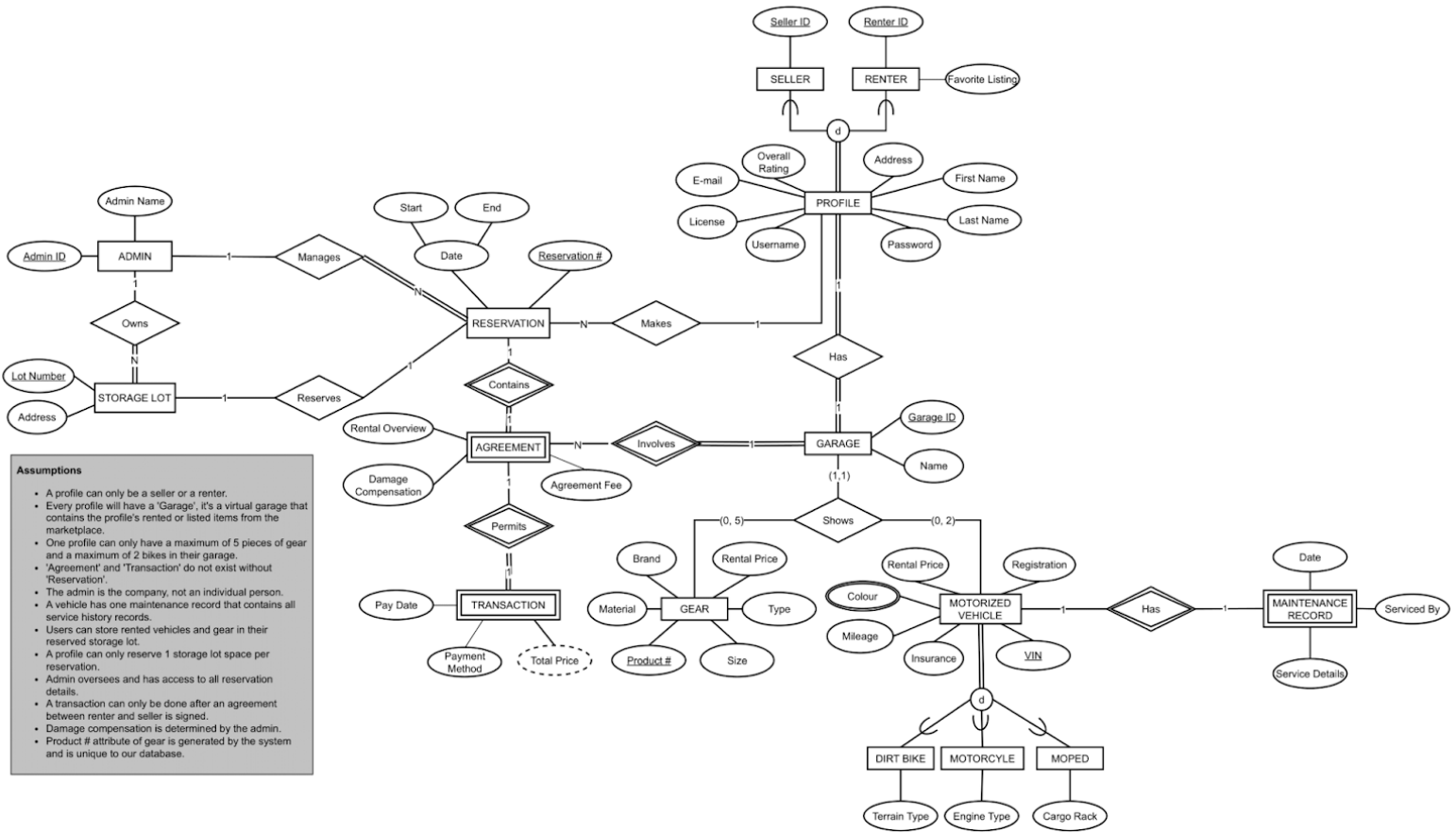
A form for adding a new lot. It has a title 'Lot Address' above a text input field containing the placeholder 'Enter lot address'. Below this is a label 'Price:' followed by another text input field containing the placeholder 'Enter the lot price'. At the bottom center is a blue button with the text 'Add Lot'.

How to Edit Existing Lot Details:

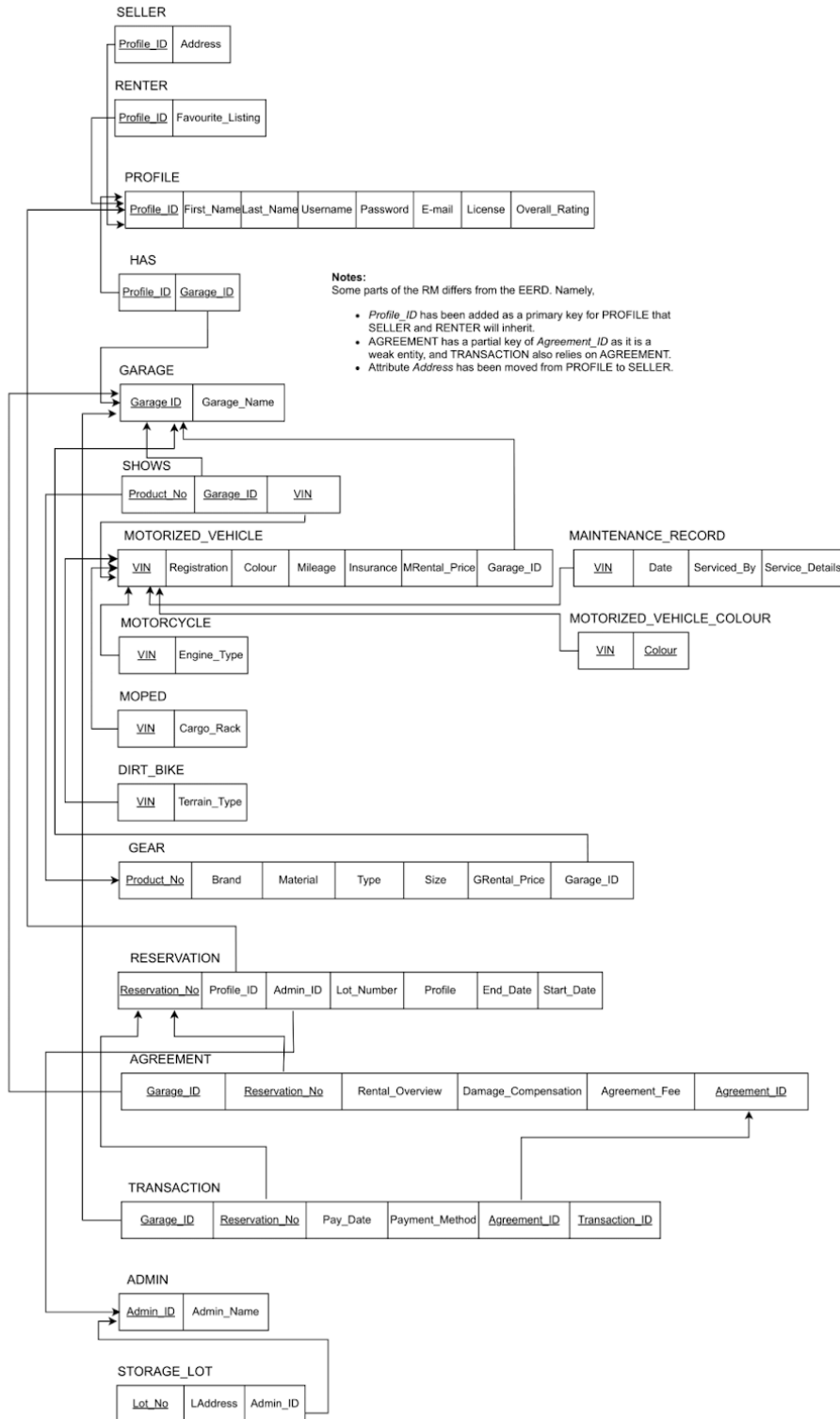
If the admin presses a specific lot from the Lot Listings page, they can edit the details of the lot. The number of the lot is fixed and cannot be edited, however the address and price can be adjusted. The admin can change only the address, only the price, or both. Both fields must have information, and neither can be “null” or empty. Once the new changes have been inputted, the admin can finalize the updated lot information by pressing “Save Changes”.

A form for editing an existing lot. It starts with the text 'Lot No: 1'. Below this is a label 'Address:' followed by a text input field containing the text '89 Some Address'. Underneath is a label 'Price:' followed by a text input field containing the text '50.0'. At the bottom center is a blue button with the text 'Save Changes'.

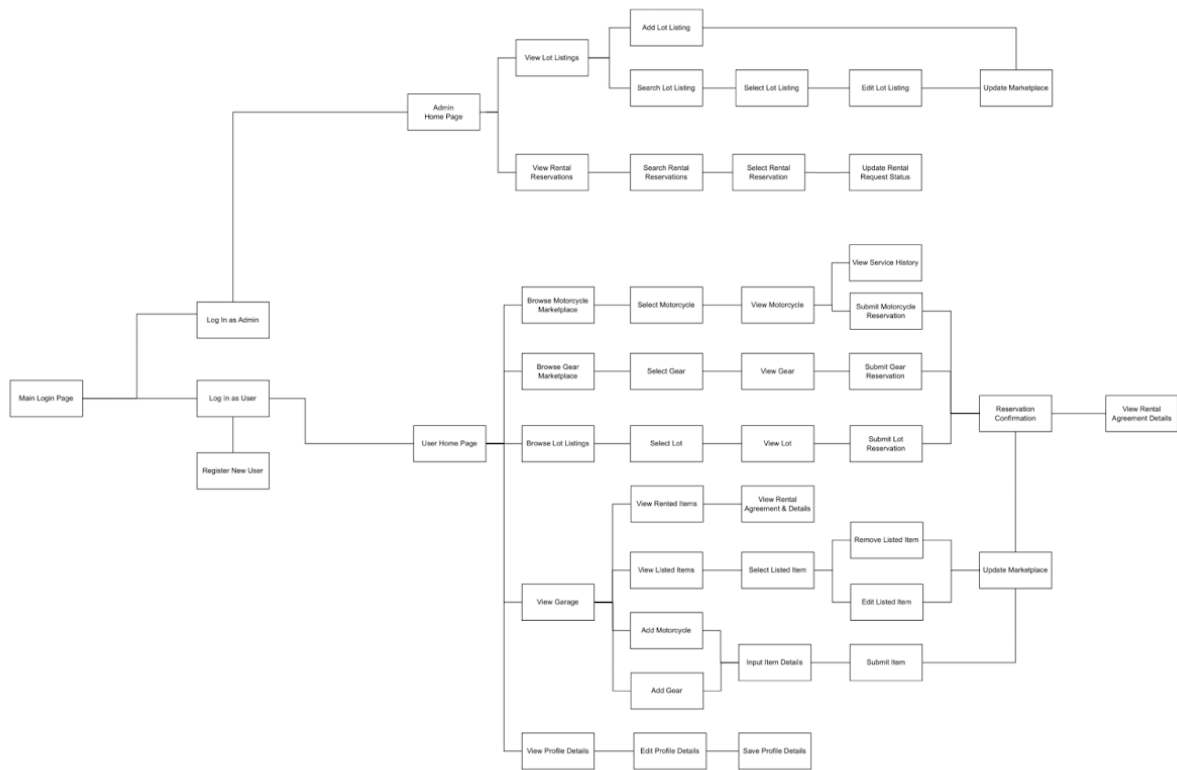
Appendix A: Original EERD



Appendix B: Original RM

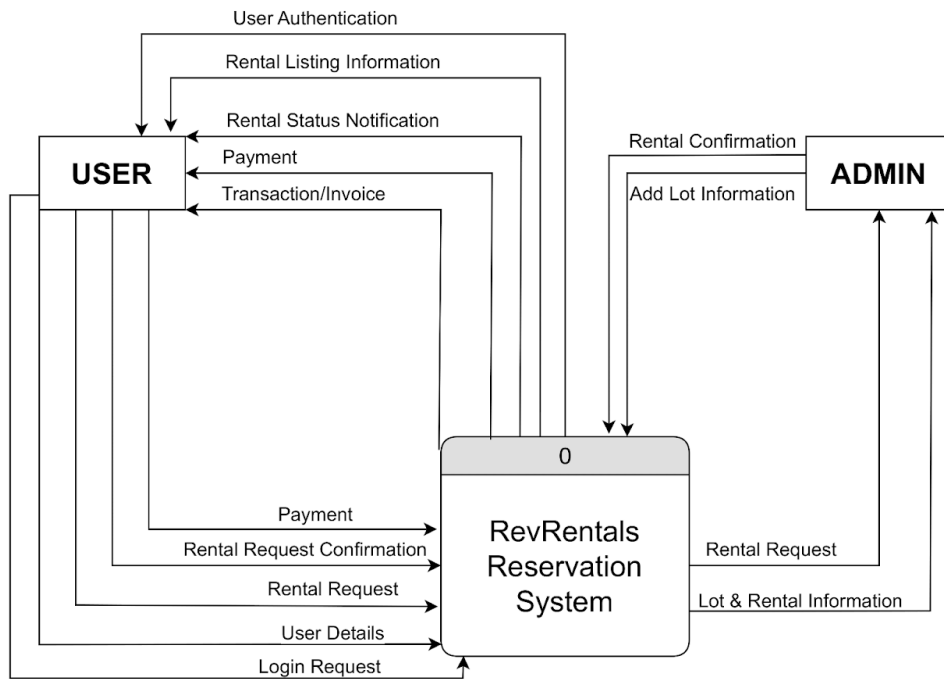


Appendix C: Original HIPO

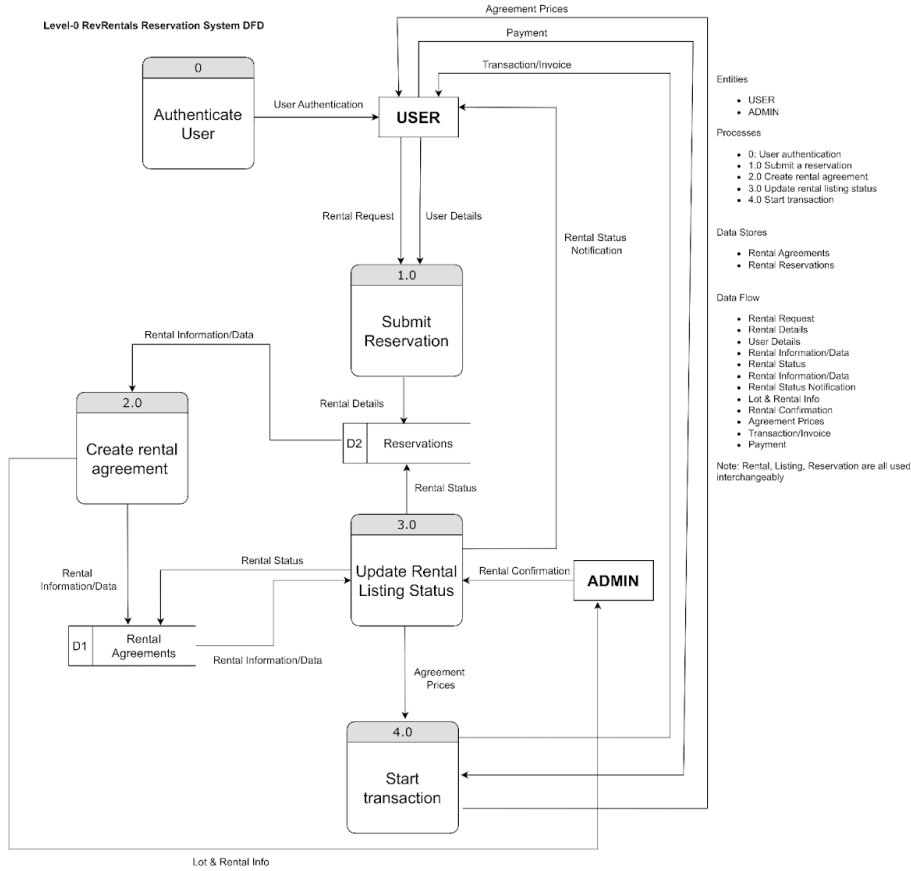


Appendix D: Original DFD (High Level)

Context Diagram (High-level)



Appendix E: Original DFD (Level-0)



References:

[GitHub Frontend Repository & Source Code](#)

[Github Backend Repository & Source Code](#)

EagleShare:

<https://www.eagleshare.com/yamaha-rentals/mt-09/cumming-georgia-united-states/21592>

RiderShare:

<https://www.riders-share.com/listings/las-vegas-nv/dr09-25-2024,09-30-2024/z10>

<https://www.geeksforgeeks.org/how-to-run-a-flutter-app-on-android-emulator/>