

SimplyFowl

Connor Astemborski, Cassandra Bailey, Brian Mendoza, Zachary Miles,
Riddhi Patel, Madeline Presnell, Joseph Tagliaferro, Barret Vogtman

December 17, 2019

Senior Project Fall 2019

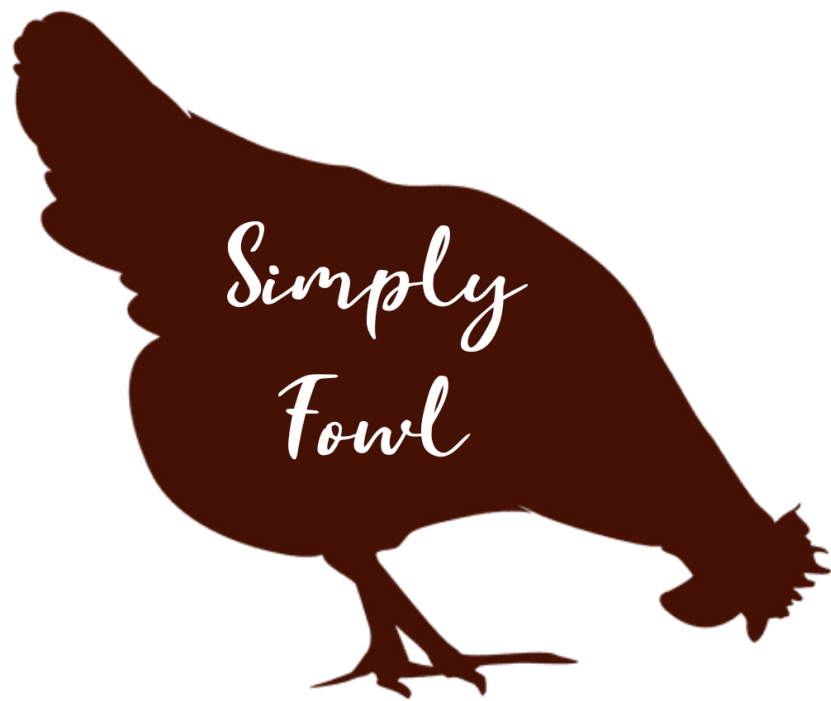


Table of Contents:

High Level Description.....	2
Web Pages.....	3
Flow Diagrams.....	17
Database ER Diagrams.....	24
Technology	
Stack.....	27
Restful Endpoints.....	28

High Level Description

The purpose of this project is to create a streamlined sales system for Raab Enterprises, a company that sells and delivers flocks of chickens. The goal is to make a simple and easy-to-use web-based application for selling chickens, tracking current inventory, handling shipping, and other related tasks.

Each user will be given login credentials with unique permissions that will direct them to a home screen customized for their role. A user may be a Sales Manager, a Flock Manager, a Truck Driver, or an admin (who can see and use all pages). Every homepage has its own unique tabs designed for the use cases of each role, most of which adding and editing information in the chicken sales system.

Web Pages

This system consists of twelve web pages total. Different users will have access to different pages based upon user permissions. Users with admin-level access will be able to see and use any of the pages at any time, with the exception of the Truck Driver page, reserved for drivers' reference only.

Login Page

This will be the first page to be seen by any user when they enter the system. Users will enter their username and password and be redirected accordingly.



Login

Username

Password

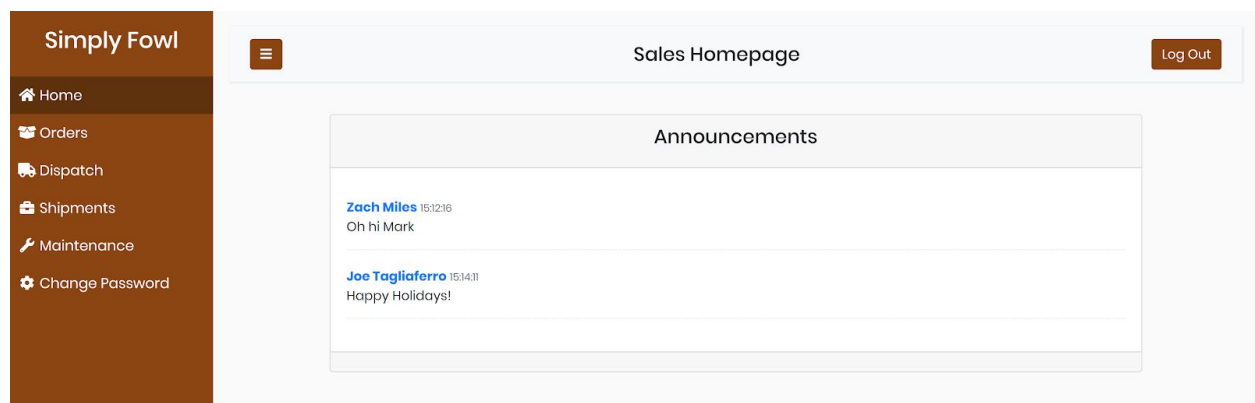
Submit

Sales Manager Homepage

Users who are Sales Managers and users with admin-level access will be sent to this page upon login. It will be seen by these users only.

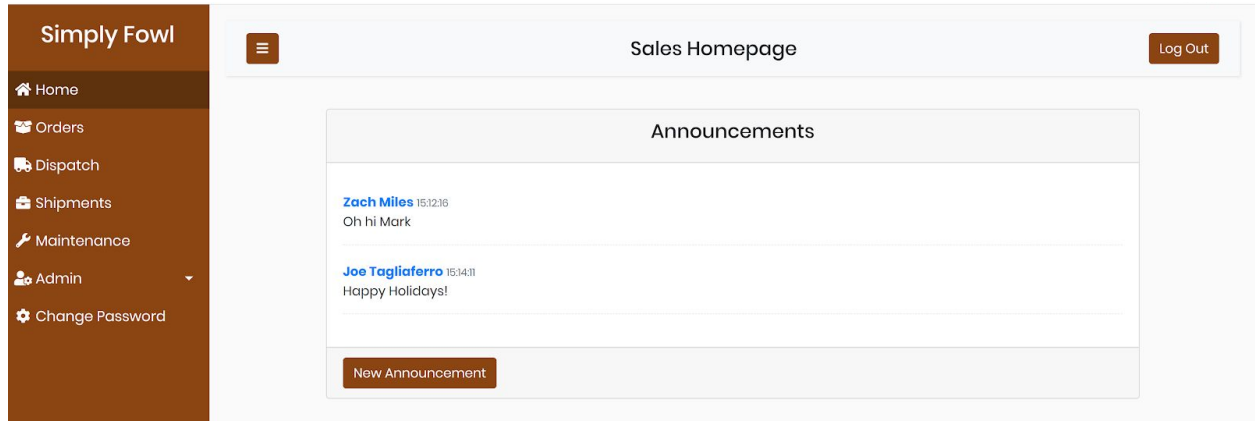
All users with access to this page:

- May click on the name of the Orders, Dispatch, Shipments, Maintenance, or Change Password page to be redirected to it.
- May view important messages written by an admin that were designated for Sales Managers.



Users with admin-level access:

- May click on 'Admin' to view other options available to them, including being redirected to the Account Management or Flock Manager page. Note: This option is available to admins on all webpages.



- May write and post a new message, selecting which type(s) of users it will be sent to.

The screenshot shows a 'New Announcement' modal form. It has a title bar with a close button. The form contains a text area labeled 'Announcement:' with the placeholder text 'Write your message here...'. Below this is a section labeled 'Recipients:' with three checkboxes: 'Sales Managers', 'Flock Managers', and 'Truck Drivers'. At the bottom are two buttons: 'Post Announcement' and 'Clear'.

Order Page

A Sales Manager will have the ability to add an order to the system by inputting its important information and clicking “Create”. Orders that have been created will be able to be viewed and removed. This page can be accessed only by Sales Managers or admins.

Orders

Log Out

Create New Order

U Do It

mm/dd/yyyy

Number of Coops

Webster - yellow pullet

+

Create

Order Search

Search by Store or Date

Store	# Coops	Bird	Date	Remove
Vandelay Industries	25	yellow pullet	12-11-19	Remove #138
Vandelay Industries	75	red fowl	12-11-19	Remove #140
Vandelay Industries	50	yellow pullet	12-11-19	Remove #139
U Do It	10	silky	12-10-19	Remove #137
U Do It	15	red fowl	12-10-19	Remove #136

Dispatch Page

On this page the Sales Manager can create dispatches and delivery routes. A delivery consists of all of the orders to a specific store for tomorrow's date. First the user must pick a truck and truck driver. The user can then set up a route with as many orders as they wish, as long as they are within the capacity of the chosen truck. Existing delivery routes can be edited, have their order altered, or be deleted. This page can be accessed only by Sales Managers or admins.

Create New Dispatch

Truck

14 - Manual

▼

Driver

Danes, Luke

▼

Create

Create Delivery Route

Coops Remaining: 50

Current Route:

Total Coops : 50 - Chicks Chicks Chicks - Wilmington, DE

x

Total Coops : 100 - Chicken Bell - Smyrna, DE

x

Submit

Clear

Unassigned Deliveries

Search

Total Coops : 175 - Shoprite - Devon, PA

Delivery Details



Ordered By:

Store Name : Chicken Bell

Address : 293 Yorkshire Ave Smyrna, DE

Date : 12/12/2019

Order Details

Bird Type	Coops
guinea hen	30
red fowl	30
turkey - tom	20
silky	20

Add

Close

Dispatch Details



Truck:

14

Truck Driver:

Luke Danes

Dispatch Details

Delivery Route

Chicks Chicks Chicks - Total Coops : 50

Chicken Bell - Total Coops : 100

Edit

Delete

Close

Shipment Page

A Sales Manager will have the ability to view a list of invoices (created upon the creation of orders). Details for each can be viewed and the invoices can be sent directly to a printer. Also on this page, the Sales Manager can update the weights and number of coops for each flock of chickens, after they are weighed on delivery day. This page can be accessed only by Sales Managers or admins.

Shipments

Log Out

Incoming Weights

Delivery Date:

Webster - yellow pullet

Weight #

Weight #

of Coop

Trailer #

Submit

Current Invoices

Invoice #	Customer	Date	View
33	Birdie Bird	12-10-19	View Details
40	U Do It	12-10-19	View Details
41	Vandelay Industries	12-11-19	View Details
42	McDonalds	12-11-19	View Details

Shipment Details

✕

Invoice ID:

40

Customer:

U Do It

Date:

12-10-19

Shipment Details

Order Number	# Coops	Chicken Type
136	15	red fowl
137	10	silky

Print Invoice

Close

Accounts Management Page

Only admins have access to this page to modify the users of this system. New users can be added and given roles, while existing users are displayed in a list at the bottom. The users in this list can have their information edited and/or role changed and they may be removed from the system entirely.

Accounts Management

Existing Users

Admin

View All

Select User

Billy Bob - Admin

Maddy Presnell - Admin

Connor Astemborski - Admin

Zach Miles - Admin

Barret Vogtman - Admin

Joe Tagliaferro - Admin

Riddhi Patel - Admin

Brian Mendoza - Admin

Cassie Bailey - Admin

Add New User

Edit User

Delete User

Edit User

First Name:

Last Name:

Zach

Miles

Username

Role

zach

Admin

Status

Active

Save Changes

Maintenance Page

A Sales Manager can add a customers, trucks, and truck drivers to the system by inputting their important information. The current list of customers can be viewed and marked active or deactive. The current list of trucks can be viewed and edited, and the user can mark trucks as being out of service. This page can be accessed only by Sales Managers or admins.

Customer

Add New Customer

Customer Name

Address

City State Zip Code

Phone Number

Truck

Add New Truck

Truck Number

VIN License Plate

Maximum Coops Transmission

Manage Trucks

Active Trucks

Inactive Trucks

Truck 1




Truck 2

Deactivate

Reactivate

Current Trucks

Search by Truck VIN, or License Plate #

Truck #	VIN	License Plate	Max Coops	Status
1	12341	12441	225	
2	90001	87654	250	
3	67854	12987	175	
4	28405	12345	200	
5	20573	29123	200	

Driver

Add New Driver

Driver

Harry Potter

Date of Birth

mm/dd/yyyy

License Type

CDL

Date of License Expiration

mm/dd/yyyy

Date of Medical Expiration

mm/dd/yyyy

License number

123456789

License State

State

Transmission

Automatic

Phone Number

555 555 5555

Add New Driver

Cancel

Flock Manager Pages

Users who are Flock Managers will be sent to a homepage upon login with important messages from the admin that were designated for Flock Managers. They can then select to manage flocks or farms. Both of these actions have their own webpage. New flocks and farms can be added to the system by inputting the important information and clicking “Add.” Flocks and farms already in the system may be viewed and edited. This page can be accessed only by Flock Managers or admins.

Simply Fowl

Home

Manage Flocks

Manage Farms

Admin

Change Password

Manage Flocks

Log Out

Available Flocks

Penn Farm

Show All Flocks

Flock Information:

Farm Name :	Penn Farm
Building # :	2
Building Floor :	lower
Bird Type :	white rooster
Delivery Date :	2019-12-18
Hatchlings :	0

Update

Remove

Add New Flock

Farm

Select Farm

Building

Select Building

Delivery Date

mm/dd/yyyy

No. of Hatchlings

of Hatchlings

Bird Type

Select Bird Type

Add New Flock

Simply Fowl

Home

Manage Flocks

Manage Farms

Admin

Change Password

Manage Farms

Log Out

Available Farms

Prestige Farms World Wide

Show All Farms

Farm Information:

Farm Name	Prestige Farms World Wide
Street Address	255 Anchor Road
City	Somerset

Update

Remove

Add New Farm

Farm Name

Enter farm name

Street Address

Enter address


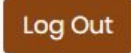
City

Enter city

Add New Farm

Truck Drivers Page

Users who are truck drivers will be sent to this page upon login. On this page they can view any delivery routes that have been assigned to them that day, and any messages sent for Truck Drivers by an admin. This page can be accessed by Truck Drivers only.

 Truck Driver 

Announcements

Joe Tagliaferro 2019-12-17 00:37:29
Have a great clucking day, everyone!

Joe Tagliaferro 2019-12-17 01:10:44
Truck drivers, roads are very icy. be careful!

Maddy Presnell 2019-12-17 15:52:18
Happy Holidays!

Deliveries

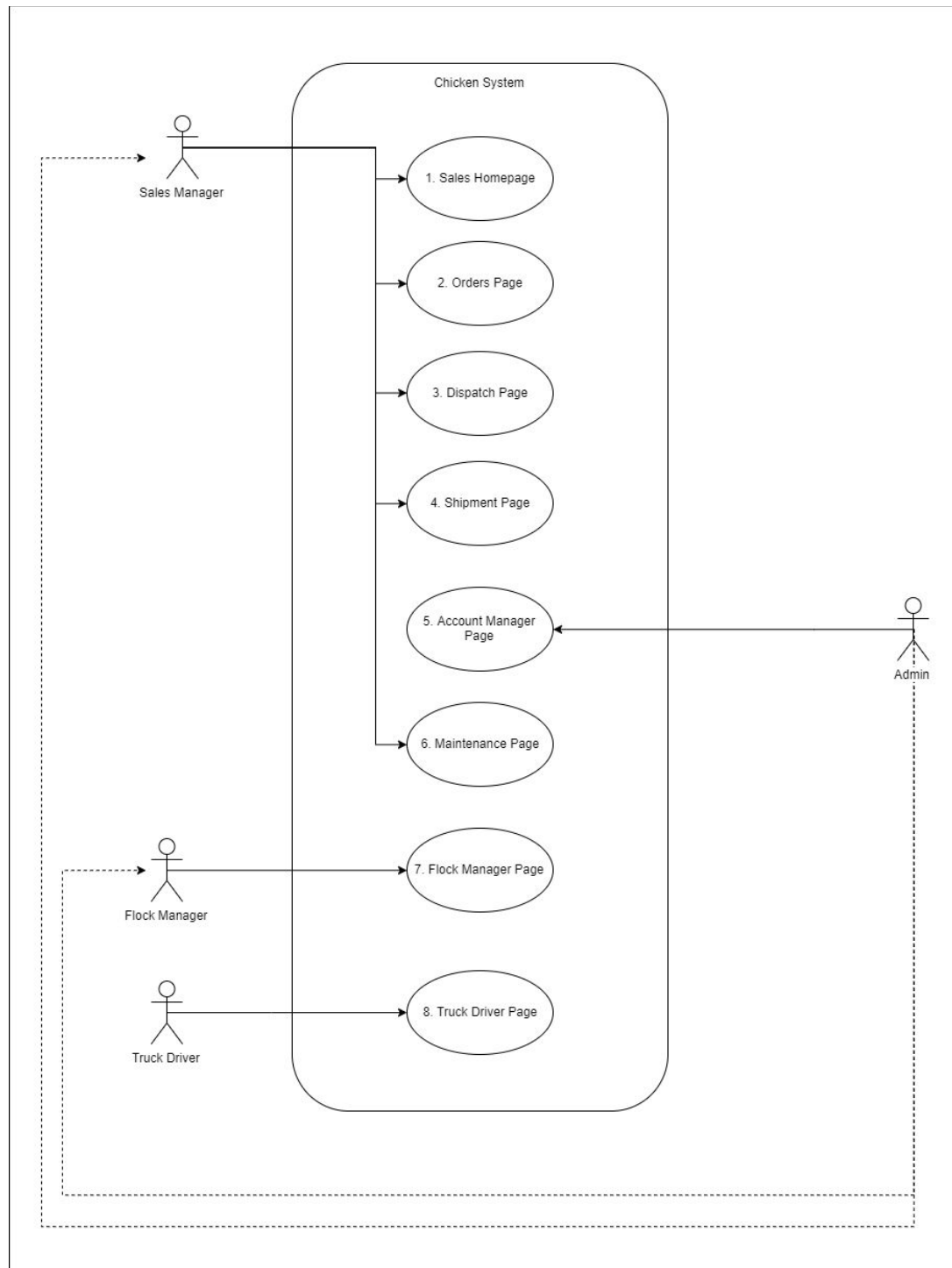
Delivery Number - 160
Delivery Number - 161
Delivery Number - 162
Delivery Number - 163

Change Password Page

A user may access this page in one of two ways. If a user logs in and their username is the same as their password, this signifies that it is the user's first time ever logging in and they will be redirected to this page to change their password. A user may also access this page from any screen within the website if they are logged in.

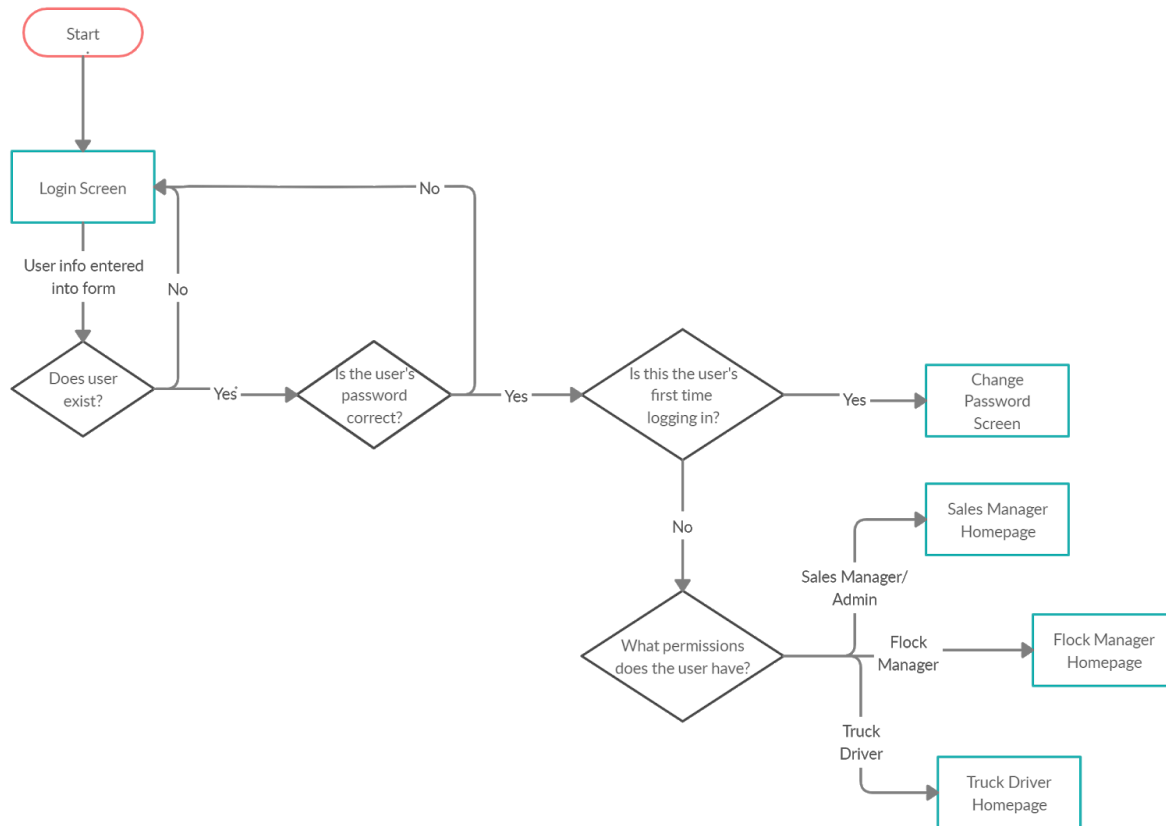
Change Password

Use Case Diagram

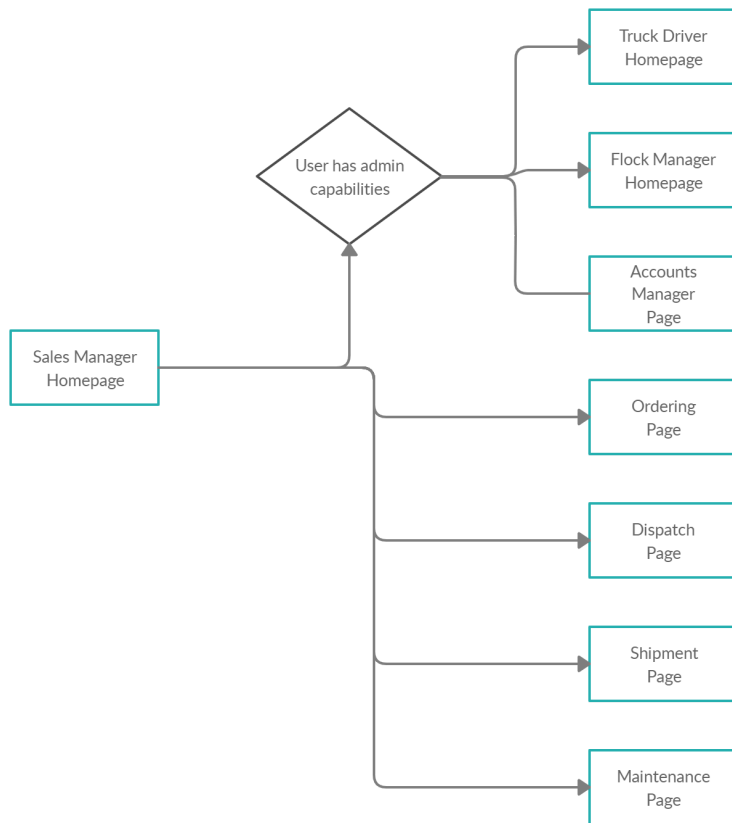


Flow Diagrams

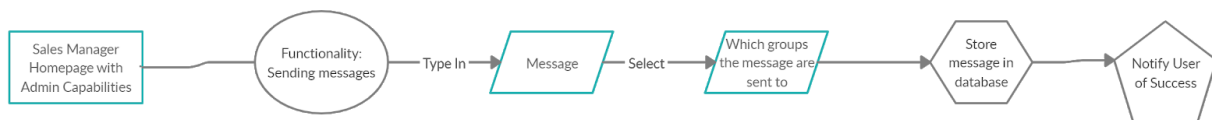
Login Page to Landing Pages



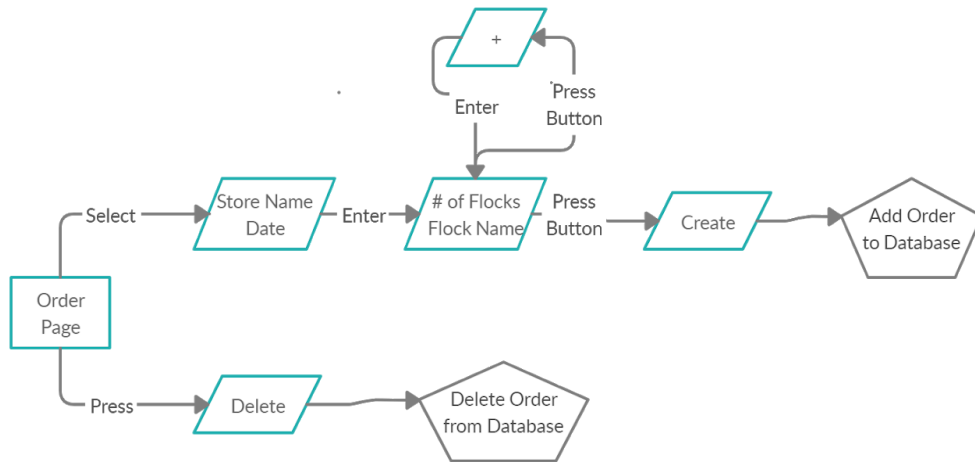
Sales Manager Homepage



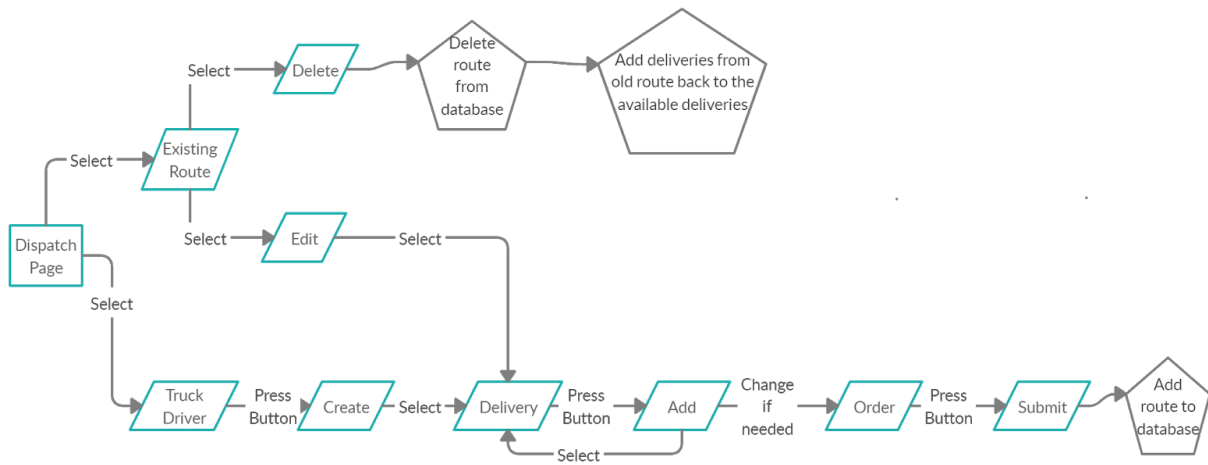
Sales Manager Homepage for Admin Messaging System



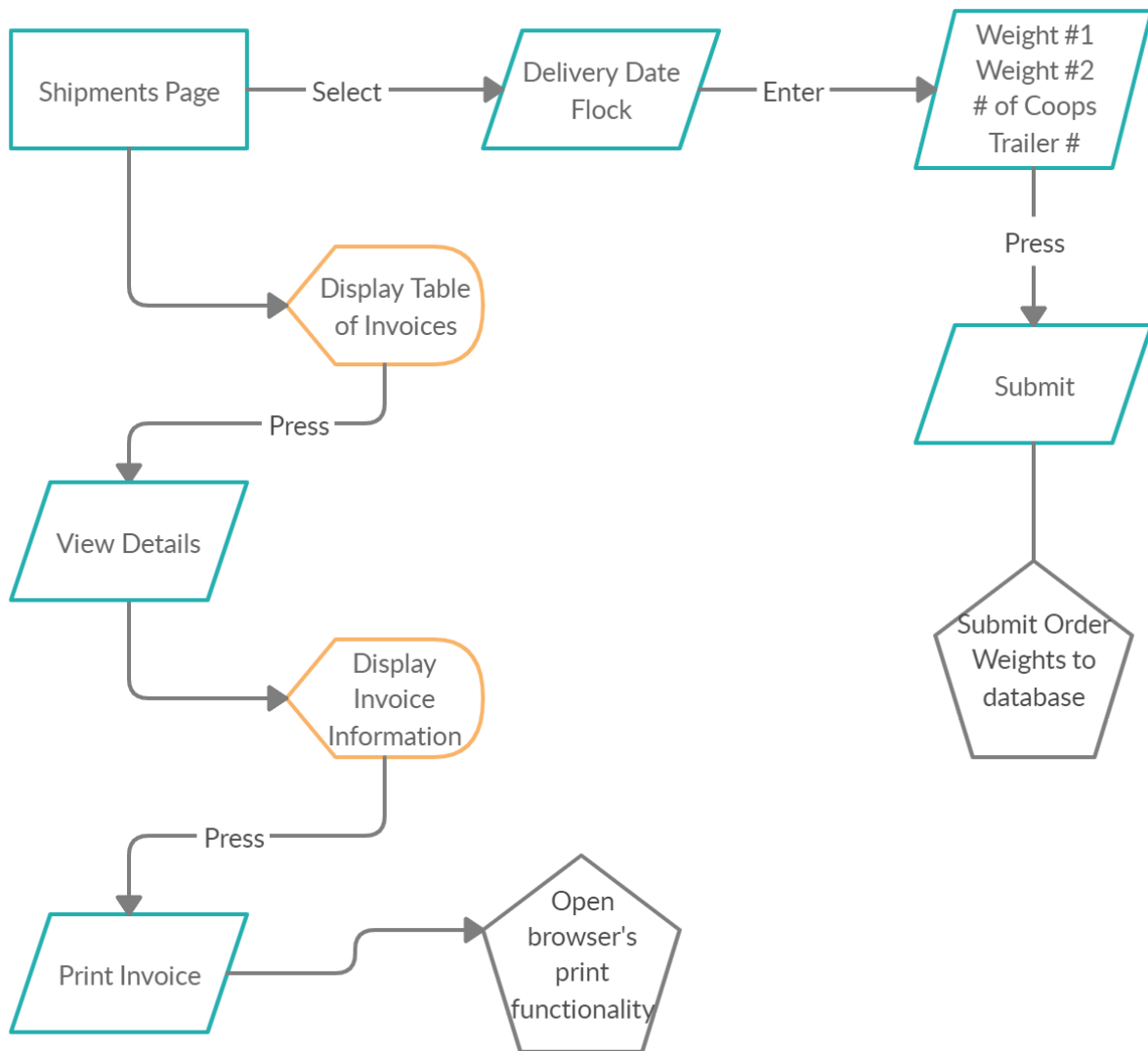
Order Page



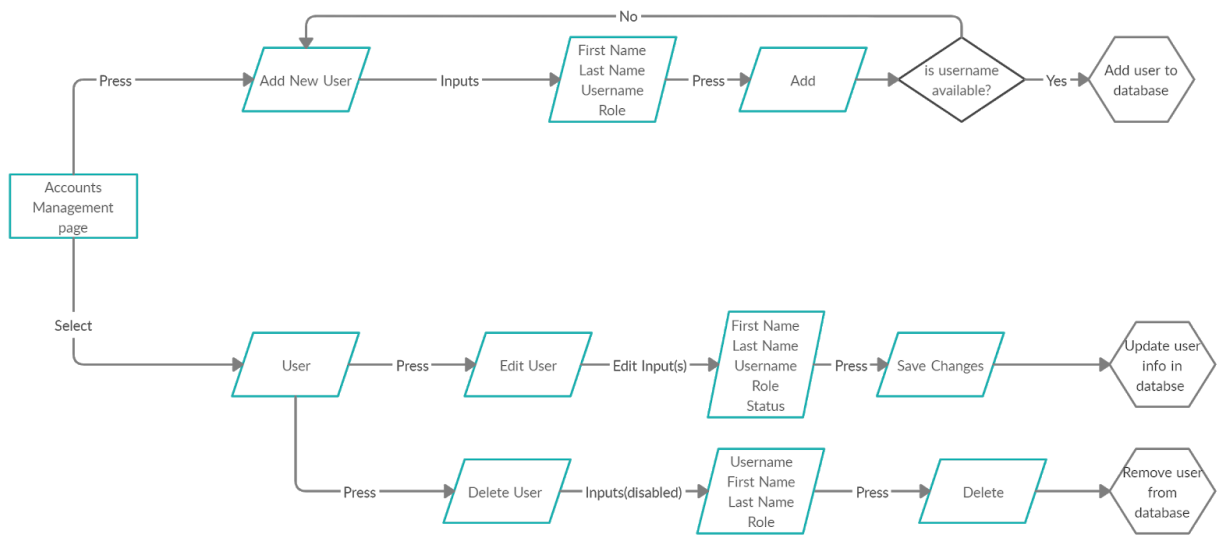
Dispatch Page



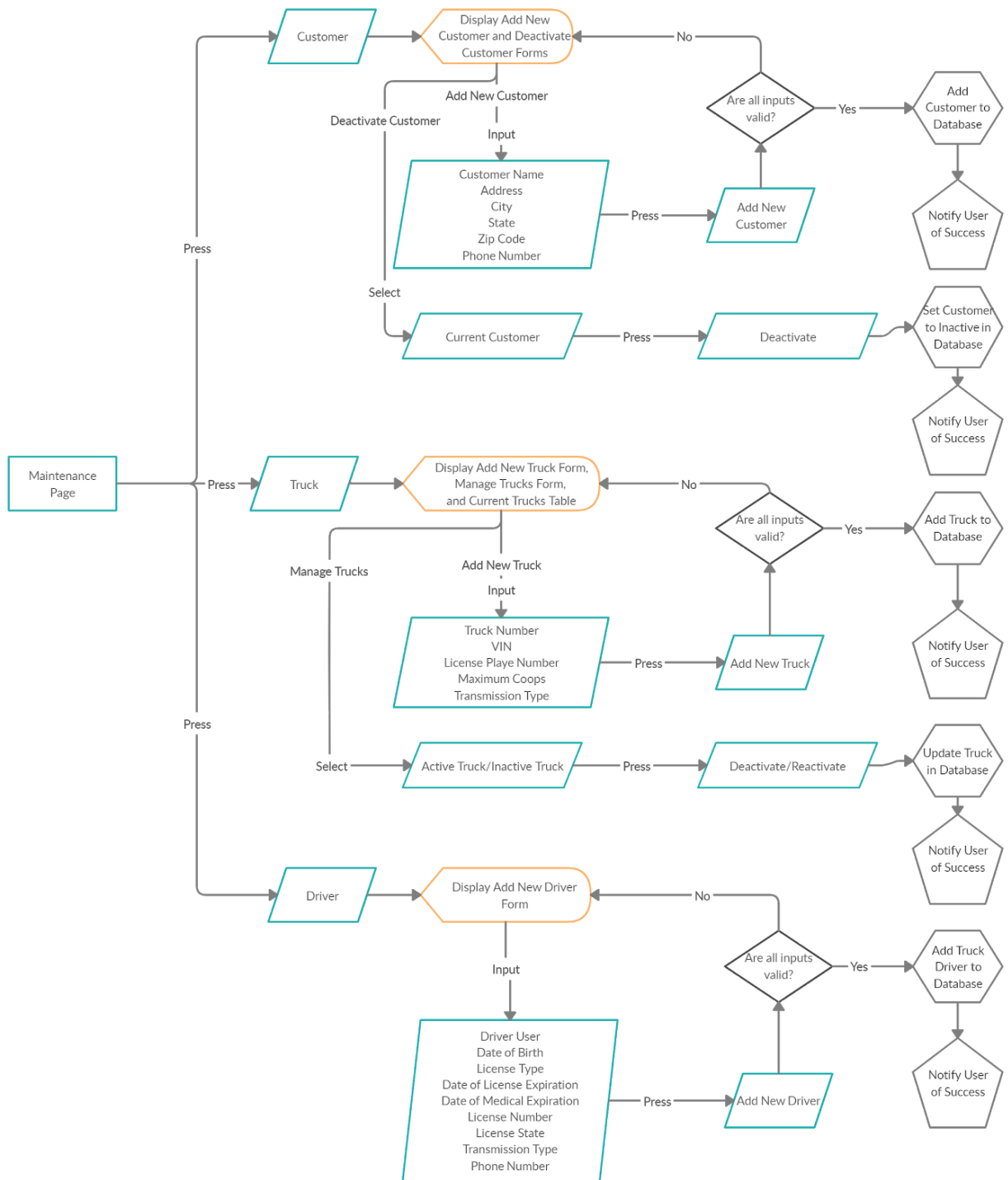
Shipments Page



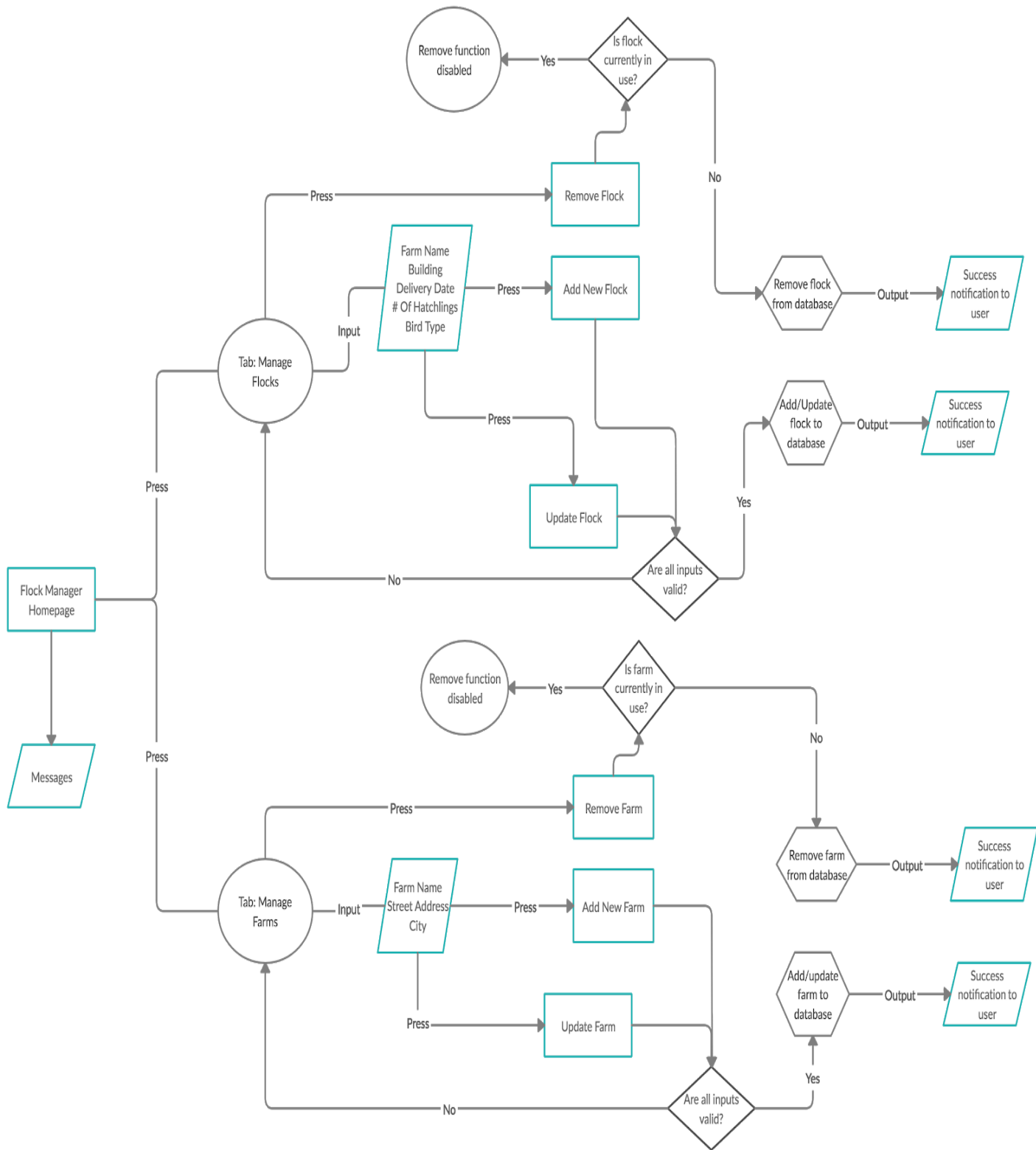
Account Manager Page



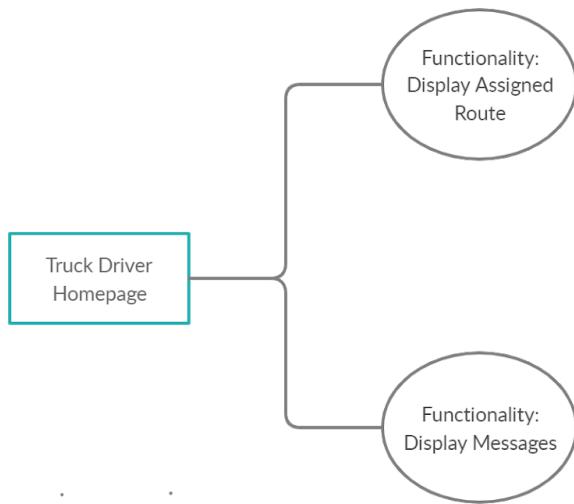
Maintenance Page



Flock Manager Homepage

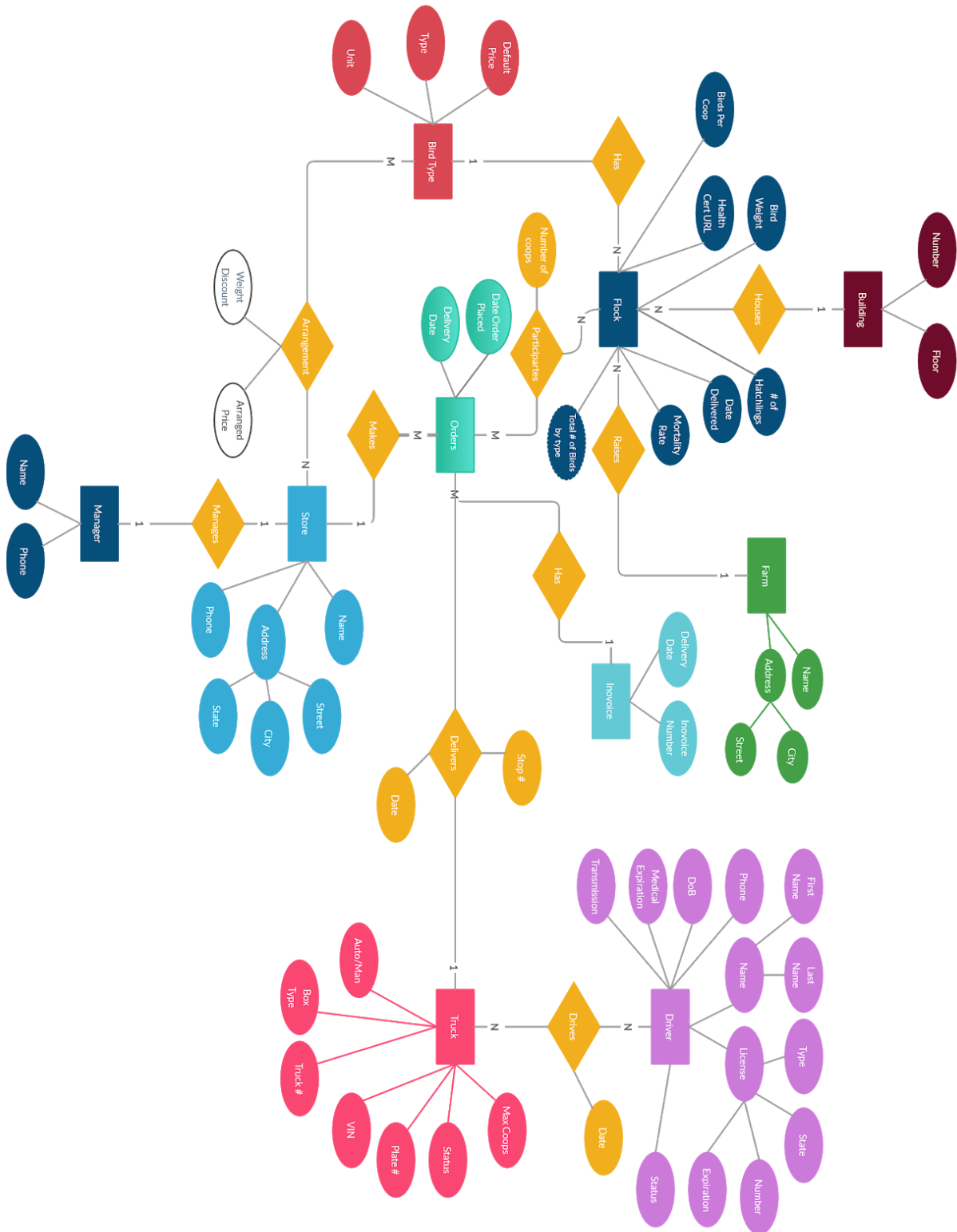


Truck Driver Homepage

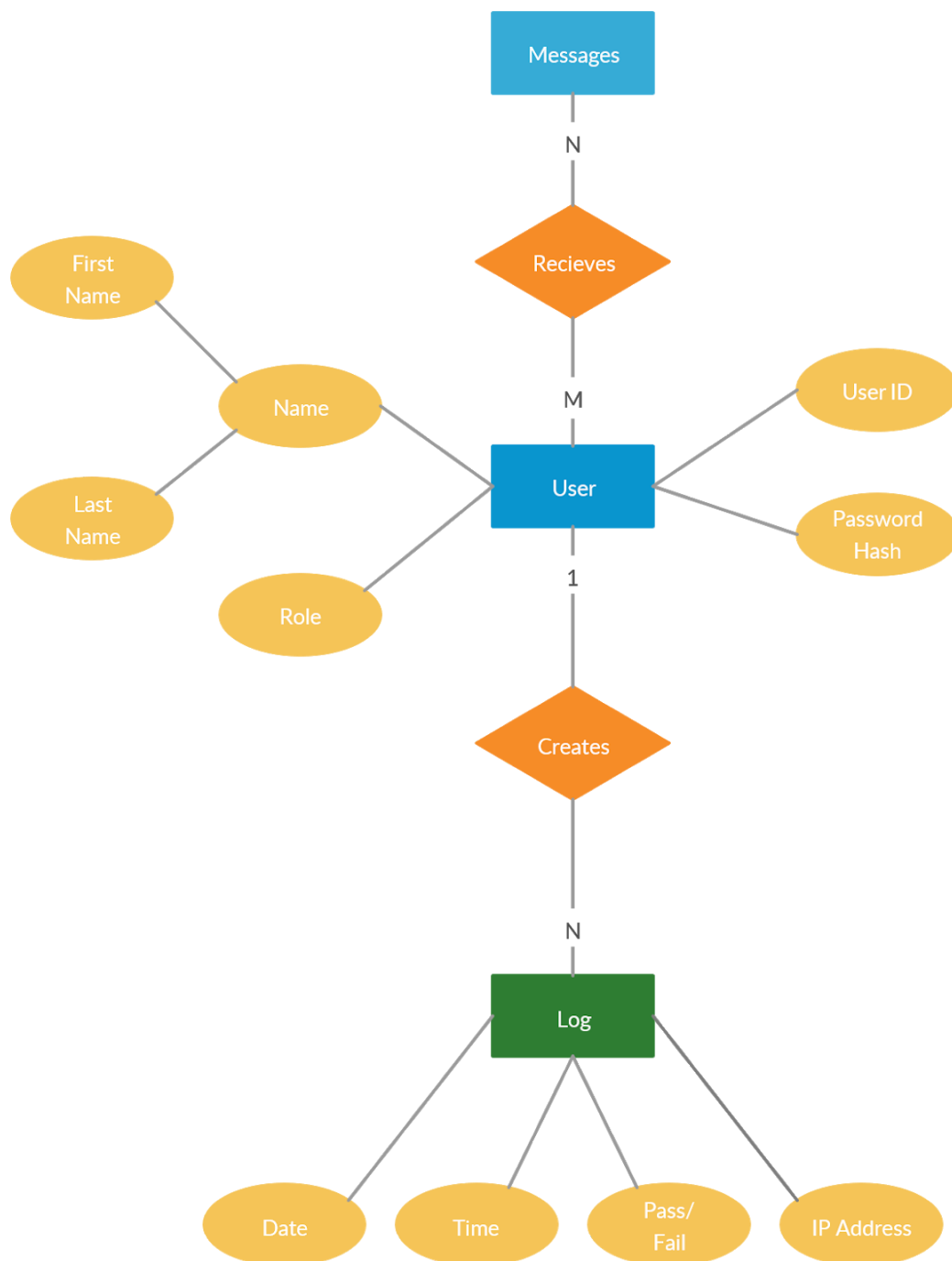


Database ER Diagrams

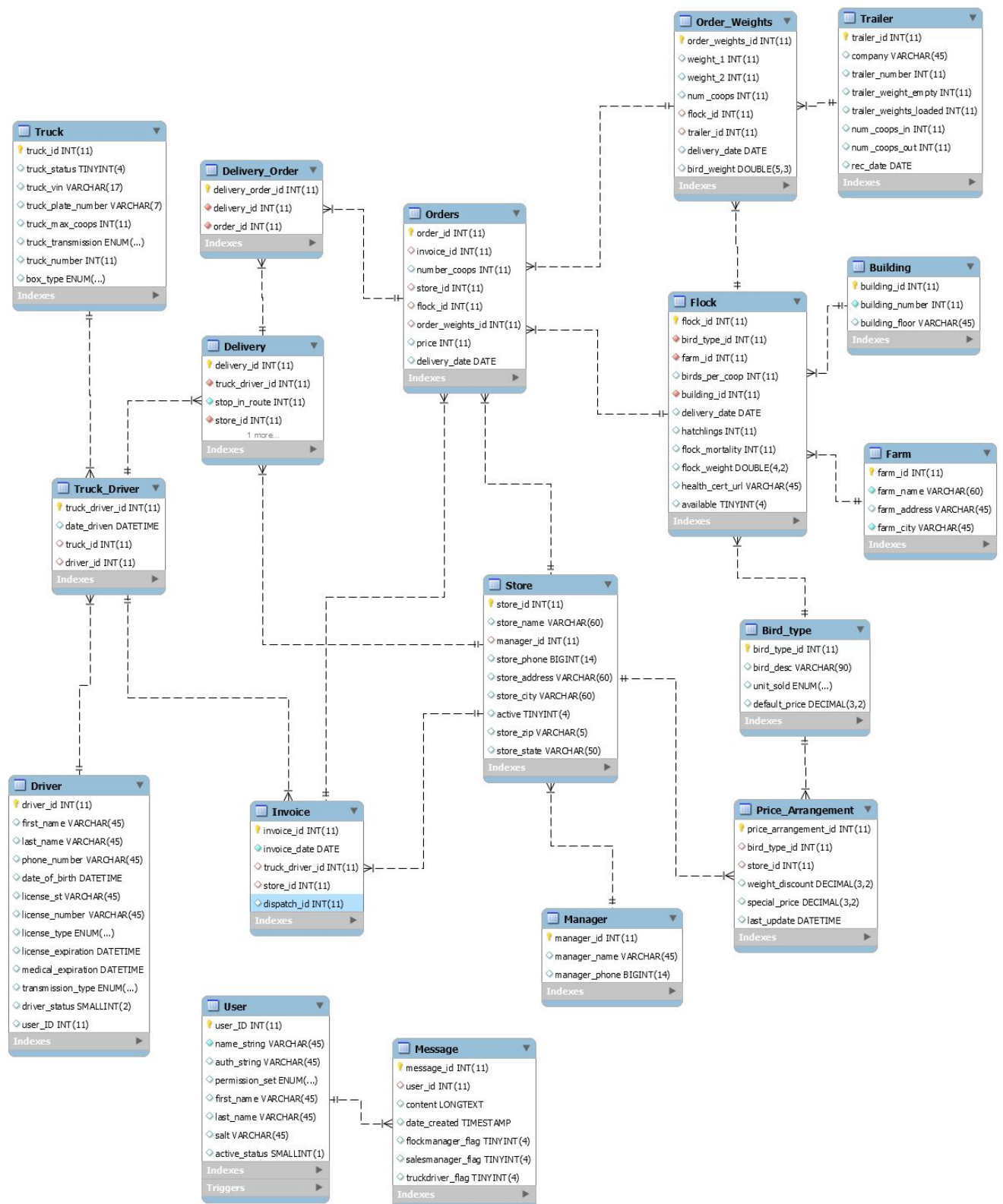
Main Database Design:



Users Design:



Database Physical Model



Technology Stack

Frontend

The frontend for the system is a series of interactive web pages. Each of these pages were developed using Bootstrap 4, one of the most popular HTML, CSS, and JavaScript frameworks for developing responsive mobile projects. Bootstrap 4 was chosen over other frameworks due to its simplicity, approachability, and its compatibility with the other aspects of our project. Since it is one of the most popular frameworks and has been used for some time, there is a wealth of documentation that made developing web pages that also work with mobile devices a much easier task. This was used in conjunction with JavaScript to maintain an asynchronous system and consistent display on the web server. The format and appearance of the pages were designed using HTML and CSS.

Backend

A standard relational database was created using MySQL. Since there are direct relationships between entities we felt that a NoSQL type database, while useable, would perform worse and possibly lead to issues with data integrity and data loss. MySQL was chosen because it is an easy to use and widely adopted database solution.

PHP was used for the endpoints as it can generate dynamic webpage content on the from the server side. It is very flexible and useful to create, delete, read, and modify the data retrieved from the SQL database. Using PHP also make this web application very user friendly as it will allow them to freely interact with the content as needed.

Restful Endpoints

GET /bird/read

Returns the list of bird types and bird id

GET /bird/selectBird

Returns corresponding bird id to bird type selected

GET /building/read

Returns list of buildings with the following attributes: building id, building floor, and building number

POST /delivery/deleteRoute/{truck_driver_id}

Deletes the route corresponding the given truck_driver_id

GET /delivery/getDeliveriesForTD/{user_id}

Returns the list of deliveries assigned to a specific truck driver with given user_id

GET /delivery/getDeliveryDetails/{delivery_ID}

Returns the order information for the delivery corresponding to the delivery_id.

GET /delivery/getDeliveryStore/{delivery_ID}

Returns the store information for the delivery.

POST /delivery/getDeliveryList/

Returns the list of tomorrow's deliveries along with the corresponding driver first name, last name, truck number, truck id, driver id and truck driver id.

POST /delivery/getDispatchDeliveries/{truck_driver_id}

Returns the list of deliveries corresponding to the given truck driver id.

POST /delivery/insertNewDeliveryRoute/{truck_id, driver_id, deliveries}

Creates a new route with the truck id, driver id and list of deliveries.

POST /driver/addDriver/{first_name, last_name, phone_number, date_of_birth, license_st, license_number, license_type, license_expiration, transmission_type, user_ID}

Creates a new driver using the user provided information for first name, last name, phone number, date of birth, license state, number, type, and expiration, favored transmission type, as well as the user_ID for the drivers account on the simply fowl web application.

POST driver/getTomorrowsAvailableDrivers/{transmission}

Returns the list of available drivers based on the given truck transmission.

GET /driver/get_drivers

Returns a list of all drivers in order by last name.

GET /driver/readDrivers/{user_id}

Returns users with “Truck Driver” privileges who do not exist in the truck driver table

POST /farm/create

Creates new farm with all required inputs by the user: name, address and city

POST /farm/delete/{farm_id}

Removes farm selected by corresponding farm_id from database

GET /farm/read

Returns list of all farms existing in the database

GET /farm/read_one/{farm_id}

Returns information about farm selected by the user. Information returned includes farm name, address, and city

POST /farm/update/{farm_id}

Updates changes to farm selected by the user. Updates may include changing farm name, address, and/or city

POST /flock/create

Creates a new flock with all required inputs by the user: farm name(using corresponding farm_id), bird type(using corresponding bird_type_id), building (using corresponding building_id), delivery date, and number of hatchlings

POST /flock/delete/{flock_id}

Removes flock selected by user by corresponding flock_id from the database(May only be removed if selected flock is not being used in a current order)

GET /flock/read

Returns a list of all flocks in the database. Attributes to flock returned includes: Farm name, bird type, building (floor and number), delivery date, and number of hatchlings

GET /flock/read_farms/{farm_id}

Returns list of flocks from the database by the farm name corresponding to the farm id selected by the user

GET /flock/read_one/{flock_id}

Returns information about flock selected by the user from the database by the corresponding flock id

POST /flock/update/{flock_id}

Updates information about flock selected by the user in the database. Updates may include: Farm name, bird type, building, delivery date, and/or number of hatchlings

POST /invoice/addInvoice/{invoice_date, store_id}

Creates a new invoice using the invoice date and store id provided by the user

GET /invoice/getInvoices

Returns information about all invoices stored in the database to be used for populating the table on the shipments page

GET /invoice/getNewID

Retrieves the id of the most recently created invoice to be used to add orders to that invoice

GET /invoice/getTomorrowsInvoices

Retrieves information about all the invoices that have a delivery date one day after today's date

POST /message/create

Creates a new message for the flock manager page.

POST /message/create_message/{user_id, content, flockmanager_flag, salesmanager_flag, truckdriver_flag}

Creates a new message using the user specified content and role flags, as well as the user id of the user who created the message.

GET /message/read

Returns information about all messages in the database.

GET /message/readforFM

Returns information about all messages for flock managers.

GET /message/readforSM

Returns information about all messages for sales managers.

GET /message/readforTD

Returns information about all messages for truck drivers.

POST /misc/getTomorrowsDate

Returns tomorrow's date in Eastern Standard Time

POST /order/getById/{invoice_id}

Returns information about all orders on a particular invoice using the invoice id specified by the user to populate the invoice details modal on the shipments page.

GET /order/getOrders

Returns information about all orders set to be delivered on today's date or later.

GET /order/getStores

Returns a list of all customers that can make orders.

POST /order/getTomorrowsDeliveryOrders

Returns the list of tomorrow's orders which have not yet been assigned to a route.

POST /order/getTomorrowsIndividualOrders/{store_id}

Returns tomorrow's individual order details for the corresponding store id.

POST /order/insertOrder/{store_id, flock_id, number_coops, delivery_date, invoice_id}

Creates a new order using the user provided store id, flock id, number of coops, and delivery date as well as the invoice id generated for the current set of orders.

POST /order/removeOrder/{order_id}

Deletes an order using the order id for the order to be deleted.

POST /order/submitWeights

Receives the incoming weights and calculates the individual bird weight and coop weight for use in invoicing.

POST /store/addStore/{store_name,store_address,store_phone,store_city,store_zip,store_state}

Creates a new Store with the given parameters

POST /store/addStore/{store_name, store_address,store_phone, store_city, store_zip, store_state}

Creates a new store using the store name, address, city, zip code, and state, store_phone.

POST /store/removeStore/{store_id}

Sets the store of the user provided id to be not active.

PATCH /store/removeStore/{store_id}

Changes the status of a store from 1 to 0 on the.

GET /store/getStores/

Returns the name and Store Id for all stores in the database that are currently active.

POST /truck/addTruck/{truck_number,truck_vin,truck_max_coops,truck_plate_number, truck_transmission}

Creates a new Truck with the given parameters

PATCH /truck/activateTruck/{truck_id}

Changes the status of a truck from 0 to 1.

PATCH /truck/deactivateTruck/{truck_id}

Changes the status of a truck from 1 to 0.

POST /truck/activateTruck/{truck_id}

Sets the status for the truck of the user specified truck id to active.

POST /truck/addTruck/{truckNumber, truckVIN, truckPlateNumber, truckMaxCoops, truck_transmission}

Creates a new truck using the truck number, truck VIN, truck plate number, max coops, and transmission type that were specified by the user.

POST /truck/deactivateTruck/{truck_id}

Sets the status for the truck of the user specified truck id to not active.

POST /truck/getMaxCoops/{truck_id}

Returns the maximum number of coops for the truck corresponding to the given truck_id

POST /truck/getTomorrowsAvailableTrucks/{transmission}

Returns the list of tomorrow's available trucks corresponding to the driver transmission capability.

POST /truck/getTrans/{truck_id}

Returns the transmission type of the truck corresponding to the given truck id

GET /truck/get_trucks

Returns information about all trucks for the maintenance page.

GET /truck/read

Returns specific information about trucks for the maintenance page.

POST /user/change_password/{username, currPass, newPass}

Updates the password for the user corresponding to the given username. If currPass matches the currently stored password for the user, then the stored password is changed to newPass.

POST /user/create_user/{first_name, last_name, name_string, permission_set}

Creates a new user using the user provided first name, last name, and permissions, and also sets the username and password for the new user both to the username provided by the creator

POST /user/delete_user/{userid}

Deletes an existing user that corresponds to the user id given as an argument.

GET /user/read_user

Returns various information about the users of the simply fowl web application.

GET/user/read_userbyID/{user_ID}

Returns all the information about the user corresponding to the provided user_ID.

GET/user/read_userbystat/{id}

Returns all the active users or inactive users based on the provided id.

POST/user/update_userinfo/{userid, firstName, lastName, permission, userName, activestatus}

Updates the user information for the specified userid.

POST /user/user_login

Returns the user information for the corresponding username and password; this information is returned only if the username exists and the entered password matches the stored password for the given username.

The restful endpoints follow the format REQUEST URL/endpoint/. Inputs to an endpoint are denoted by curly braces. Additional input and parameter entry will be handled