National Aeronautics and Space Administration

Development of Modeling & Simulation Capability to Analyze Supply Chain

Jarrod Pearman
Manjiri Parchure
Nabeel Ahmed
Tom Zurales
Victor Cabrera

Credit: NASA Image and Video Library

Michael Day, Code A
Summer 2021, **NASA Ames Research Center**

Executive Summary

By **end of Summer 2021,** we committed to expanding on our MVP, from the Spring, which simulates and models the supply chain of sUAS (drones under 55 pounds).

What were your group's largest success?

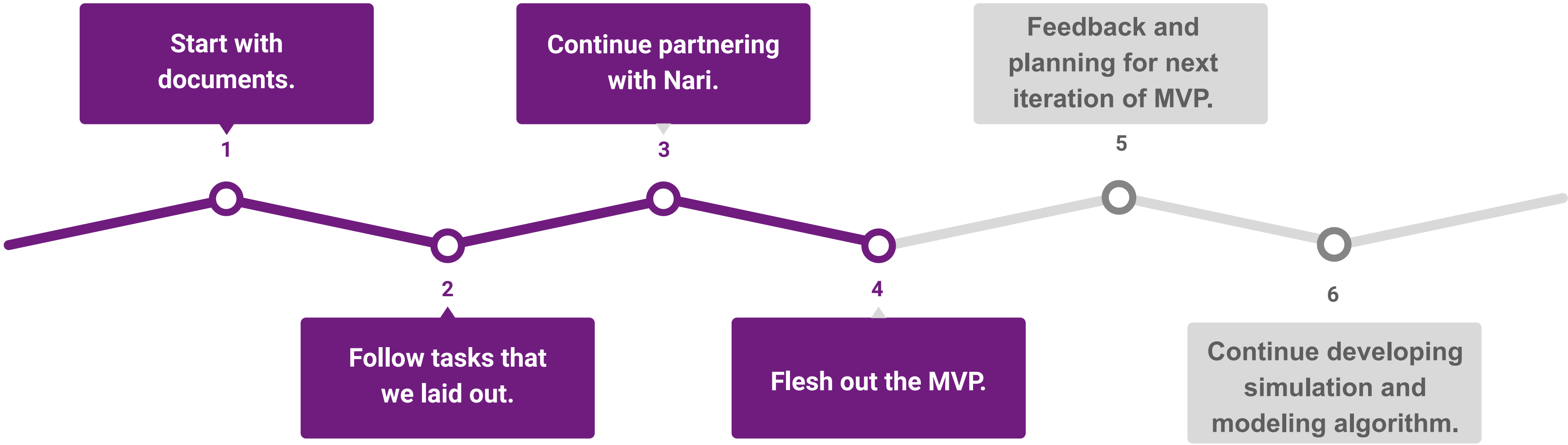
Improved Backend

Database Taxonomy & Tools

API

UI / UX

How will the next round of interns continue your work and where are we?



1 Start with documents.

2 Follow tasks that we laid out.

3 Continue partnering with Nari.

4 Flesh out the MVP.

5 Feedback and planning for next iteration of MVP.

6 Continue developing simulation and modeling algorithm.

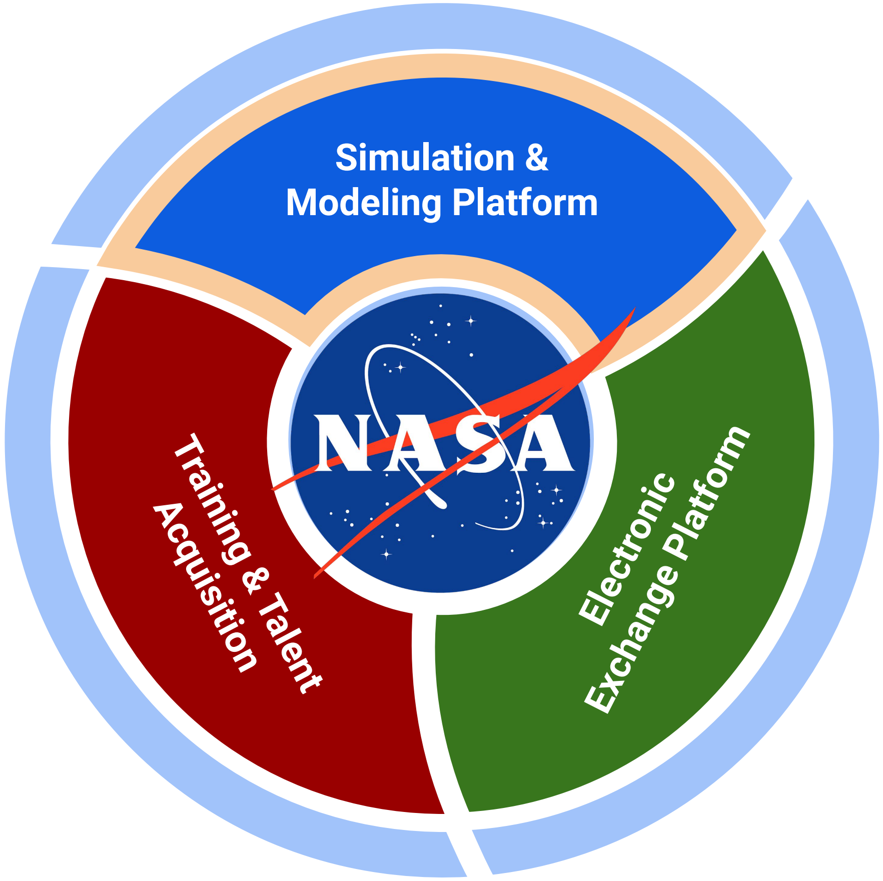
Main Problem

Current Aerospace industry in America has a big opportunity in improving existing supply chain.

AGILITY PRIME

Issues related to:

- Limited sources
- **Raw material shortages**
- **Geopolitical risks**
- **Long-lead times**
- Unauthenticated parts
- **Lack of inventory**
- **Large inventory needs**
- Inefficient coordination
- Access to talent and skills
- Cyber security


Simulation & Modeling Platform
NASA
Training & Talent Acquisition
Electronic Exchange Platform

NARI
NASA AERONAUTICS
RESEARCH INSTITUTE

Project Management Findings/Data

01 Structure

- Collaborative Github Project

02 Foundation

- Created project documents.
- Assigned roles and areas of responsibility based off previous experience.

03 Leadership

- Met regularly with mentors for clarification and suggestions
- Distributed responsibility between teams

Software Development Findings/Data

- **Software Challenges and learning opportunities:**
  - Learning, understanding, and implementing the Google Maps API.
  - Learning frontend and backend development.
  - Learning different technologies and programming languages needed for development.
- **Methodologies Used:**
  - Planning and Prototyping
  - Substituting Data allowing for testing proof of concept.
  - Ensured backend structures were written correctly to ensure correct organization and storing of data.

Credit: NASA Image and Video Library

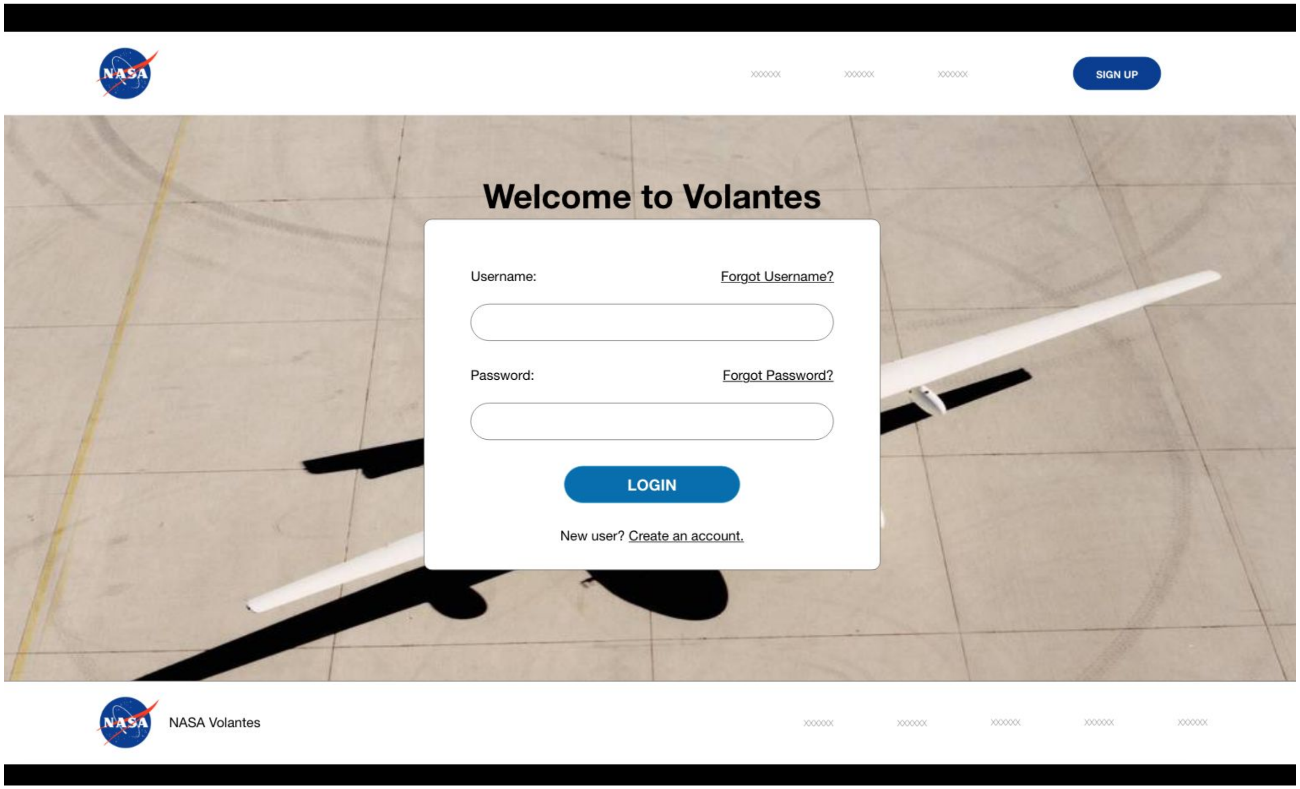
Credit: NASA Image and Video Library

Front End Core Technologies

- **HTML**
  - **HyperText Markup Language** is the set of markup symbols or codes inserted into a file intended for display on the Internet.The markup tells web browsers how to display a web pages' words and images.
- **React.js**
  - **React** is a free and open-source front end **JavaScript** library for building user interfaces or UI components.
- **CSS**
  - **Cascading Style Sheets** is a style sheet language used for describing the presentation of a document written in a markup language such as **HTML**.

UI Prototype

- **Why prototyping?**
  - Avoid expensive reworks
  - Accelerate development
  - Avoid failures
  - Demonstrate functionality



Welcome to Volantes

Username: Forgot Username?

Password: Forgot Password?

New user? Create an account.

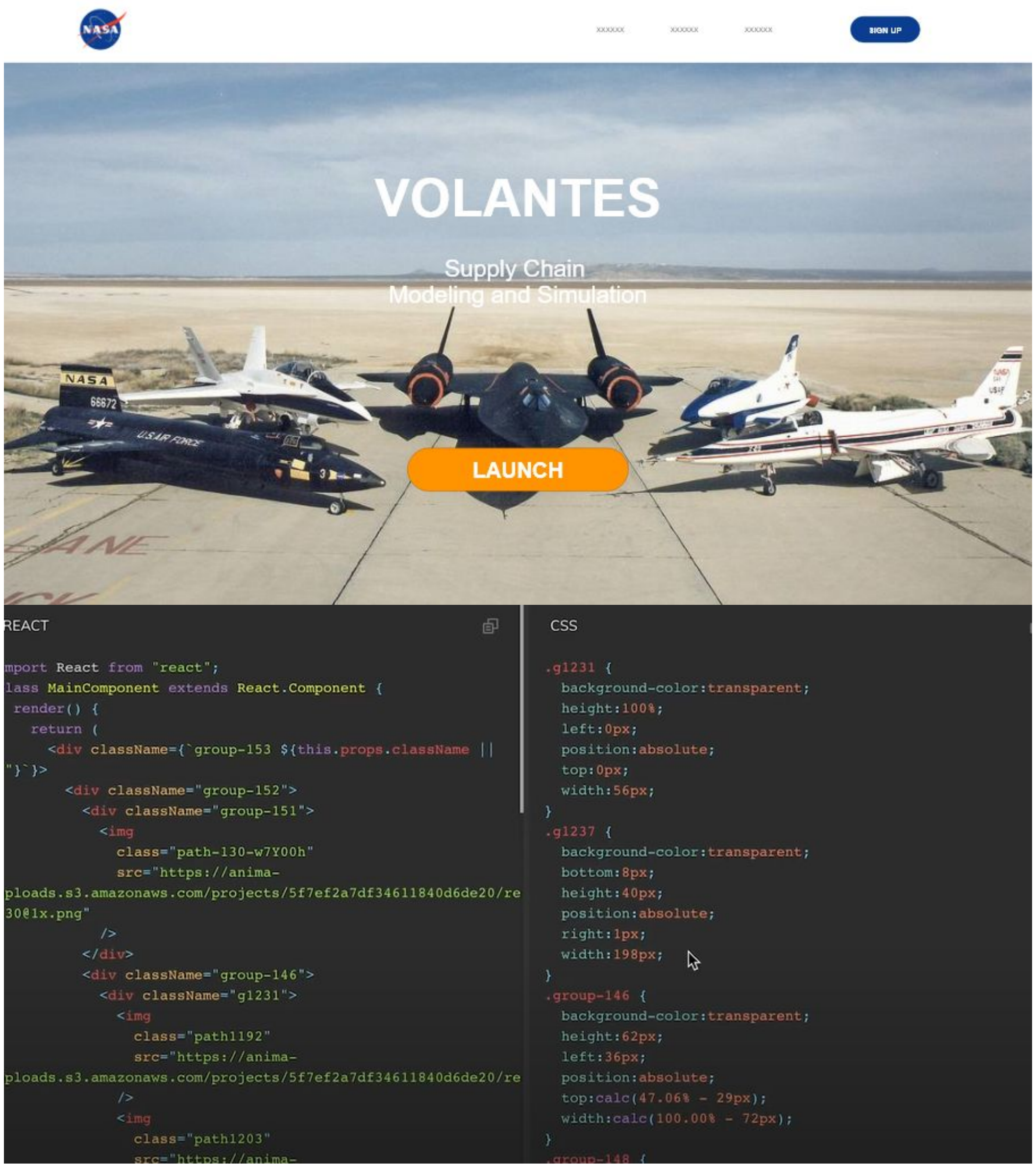
UI Prototype (cont.)

- Use of Adobe XD as main prototyping software.
- Turning static designs and wireframes into interactive prototypes.
- Adobe XD tools allow the team to extract HTML and CSS code to further develop the website.

Credit: NASA Image and Video Library

UI Prototype (cont.)

- Used the Anima software to convert the prototype into code.
- It created high-fidelity prototypes and helped convert pages into working HTML and CSS code automatically.
- The purpose of using react was for it being simple, fast, and scalable.

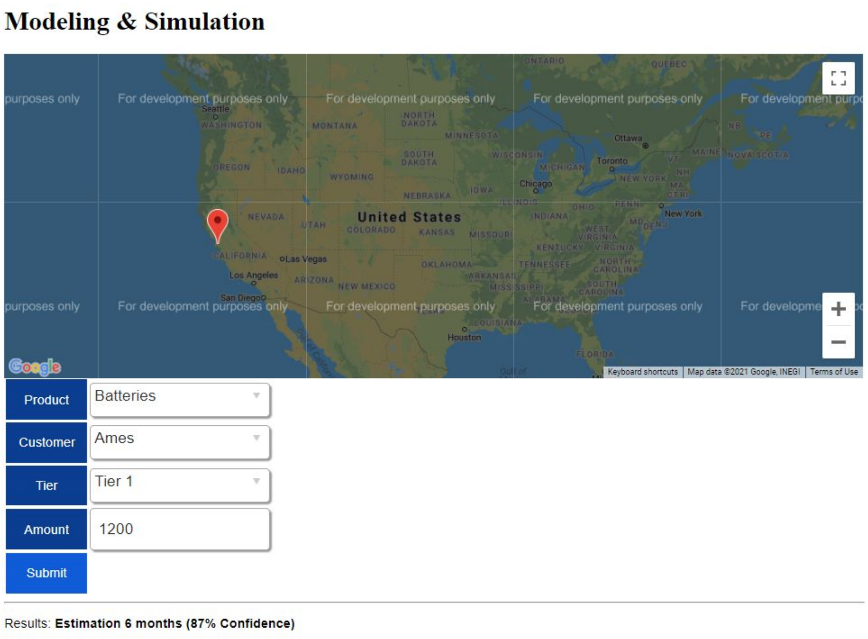


```
mport React from "react";
lass MainComponent extends React.Component {
 render() {
   return (
     <div className={`group-153 ${this.props.className ||
""}`}>
       <div className="group-152">
         <div className="group-151">
           <img
             class="path-130-w7Y00h"
             src="https://anima-
ploads.s3.amazonaws.com/projects/5f7ef2a7df34611840d6de20/re
30@1x.png"
           />
         </div>
         <div className="group-146">
           <div className="g1231">
             <img
               class="path1192"
               src="https://anima-
ploads.s3.amazonaws.com/projects/5f7ef2a7df34611840d6de20/re
             />
             <img
               class="path1203"
               src="https://anima-
```

```
.g1231 {
  background-color:transparent;
  height:100%;
  left:0px;
  position:absolute;
  top:0px;
  width:56px;
}
.g1237 {
  background-color:transparent;
  bottom:8px;
  height:40px;
  position:absolute;
  right:1px;
  width:198px;
}
.group-146 {
  background-color:transparent;
  height:62px;
  left:36px;
  position:absolute;
  top:calc(47.06% - 29px);
  width:calc(100.00% - 72px);
}
.group-148 {
```

Modeling and Simulation Platform

- Uses HTML, CSS, React.js, and JavaScript.
- Platform determines nearest Manufacturers/OEMs and then displays that information on a Map for each part needed using our database API and Google Maps API.
- Simulation will make use of Distance Matrix API to determine length and duration of travel for each part.
- Simulation will eventually assess: Security, Health, Speed, Cost, Geography, Capacity, and Sustainability to determine if the supply chain for any given part / simulated order is robust and gapless.



Modeling & Simulation

Product	Batteries
Customer	Ames
Tier	Tier 1
Amount	1200

Results: **Estimation 6 months (87% Confidence)**

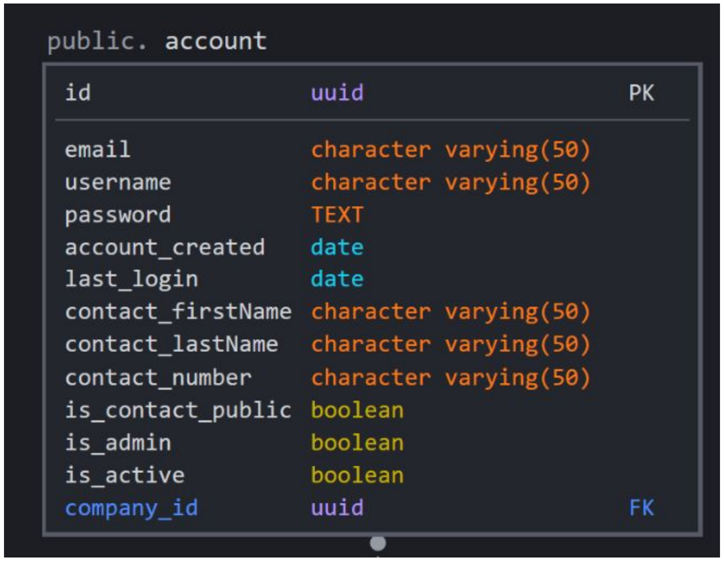
Backend Core Technologies

- **Postgres (PostgreSQL)**
  - An open source object-relational database system that uses and extends the SQL language along with many useful features that aid in building a reliable database.
- **Node.js**
  - An open source development platform used to execute JavaScript on the server side.
- **Express**
  - Javascript library which gives us the ability handle different HTTP requests and routes between the frontend and backend of the application.
- **Sequelize**
  - JavaScript library that is used to connect to and manage the database with support for many different types of databases.

Database Design

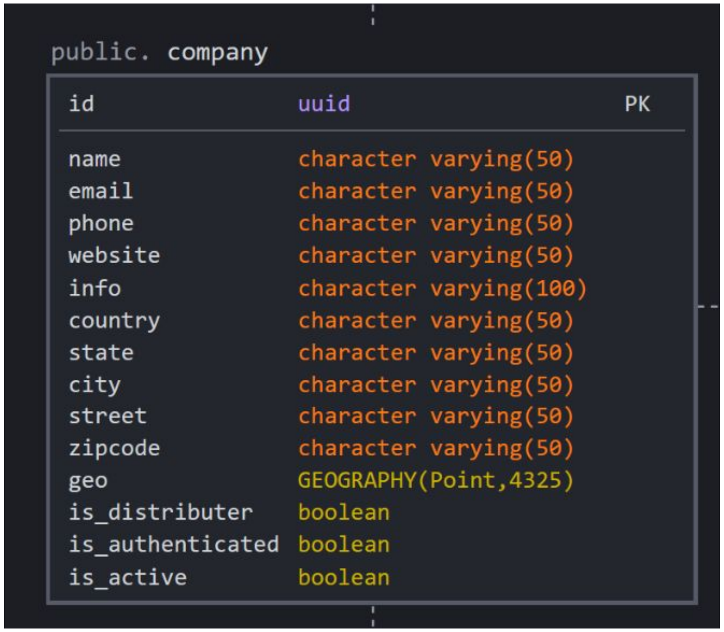
Postgres:

- **Account**
  - Stores created user accounts, contains personal information, login information, and can be linked to a company within the system.



```
public. account
id                 uuid                    PK
email              character varying(50)
username           character varying(50)
password           TEXT
account_created    date
last_login         date
contact_firstName  character varying(50)
contact_lastName   character varying(50)
contact_number     character varying(50)
is_contact_public  boolean
is_admin           boolean
is_active          boolean
company_id         uuid                    FK
```

- **Company**
  - Contains information for multiple companies such as name, contact info, and company location, including coordinates.

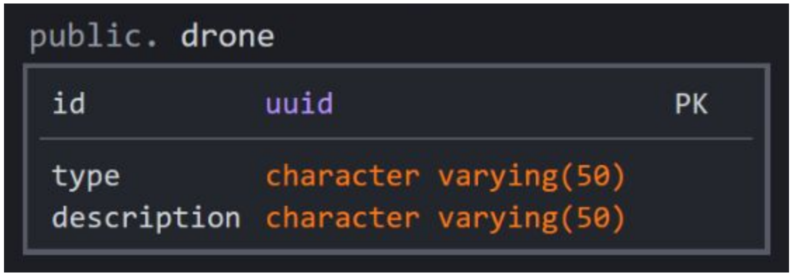


```
public. company
id                uuid                     PK
name              character varying(50)
email             character varying(50)
phone             character varying(50)
website           character varying(50)
info              character varying(100)
country           character varying(50)
state             character varying(50)
city              character varying(50)
street            character varying(50)
zipcode           character varying(50)
geo               GEOGRAPHY(Point,4325)
is_distributer    boolean
is_authenticated  boolean
is_active         boolean
```

Database Design

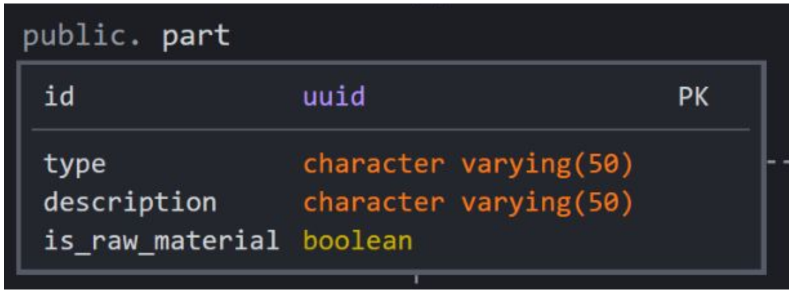
Postgres:

- **Drone**
  - Contains information for different types of drones and provided brief descriptions for each.



```
public. drone
id           uuid                        PK
type         character varying(50)
description  character varying(50)
```

- **Part**
  - Contains multiple different types of parts that are used across the different types of drones, with no specifics as to brand and price.

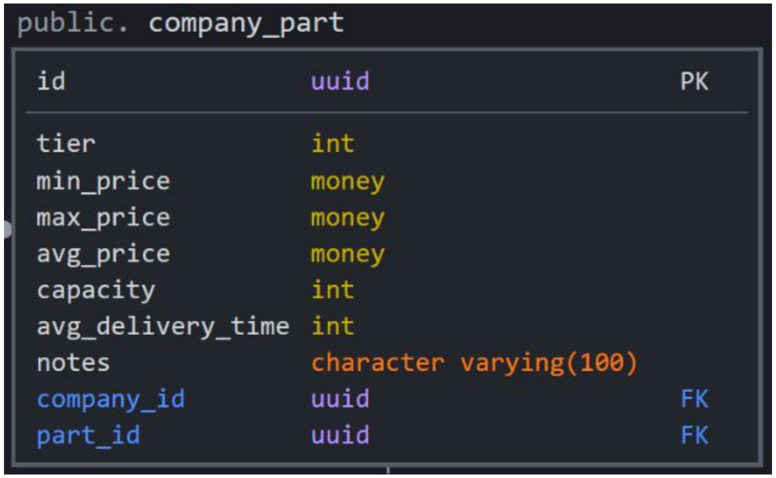


```
public. part
id               uuid                        PK
type             character varying(50)
description      character varying(50)
is_raw_material  boolean
```

Database Design

Postgres:

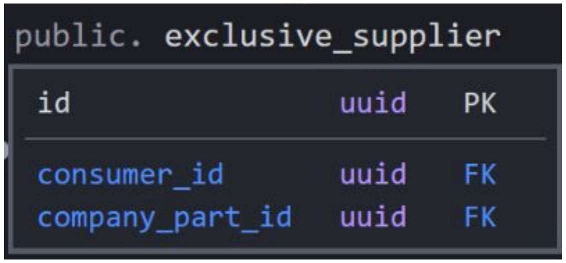
- **Company Part**
  - Table links between companies and parts to represent the types of parts that companies make along with information about their specific version of the part such as price and capacity. Provides a ‘list’ of the potential parts that a customer can select from.



public. company_part

id	uuid	PK
tier	int	
min_price	money	
max_price	money	
avg_price	money	
capacity	int	
avg_delivery_time	int	
notes	character varying(100)	
company_id	uuid	FK
part_id	uuid	FK

- **Exclusive Suppliers**
  - Table show relations involving exclusive consumer and supplier relationships. Meaning, for example, if Company A only buys a part exclusively from Company B. Multiple companies may produce the specified part but Company A only purchases from Company B.



public. exclusive_supplier

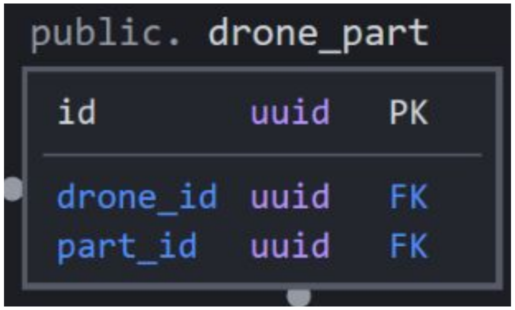
id	uuid	PK
consumer_id	uuid	FK
company_part_id	uuid	FK

Database Design

Postgres:

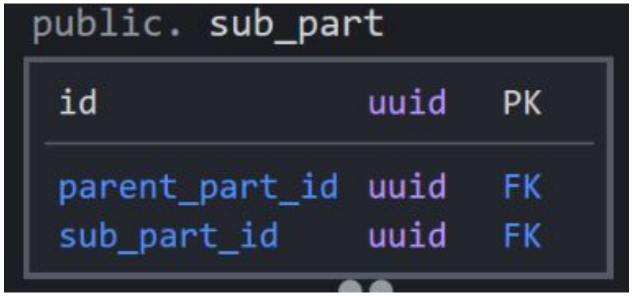
- **Drone Part**
    - Links the drones and parts tables. Provides the ability to list all parts needed for a given drone or list all drones that use a given part.

- **Sub Part**
    - A separate table to manage all ‘sub parts’ that are used to make a part component from the parts table. Helps track the chain of parts needed to make an end product. For a given part in the parts table we can find a list of sub parts that make up this given part.



public. drone_part

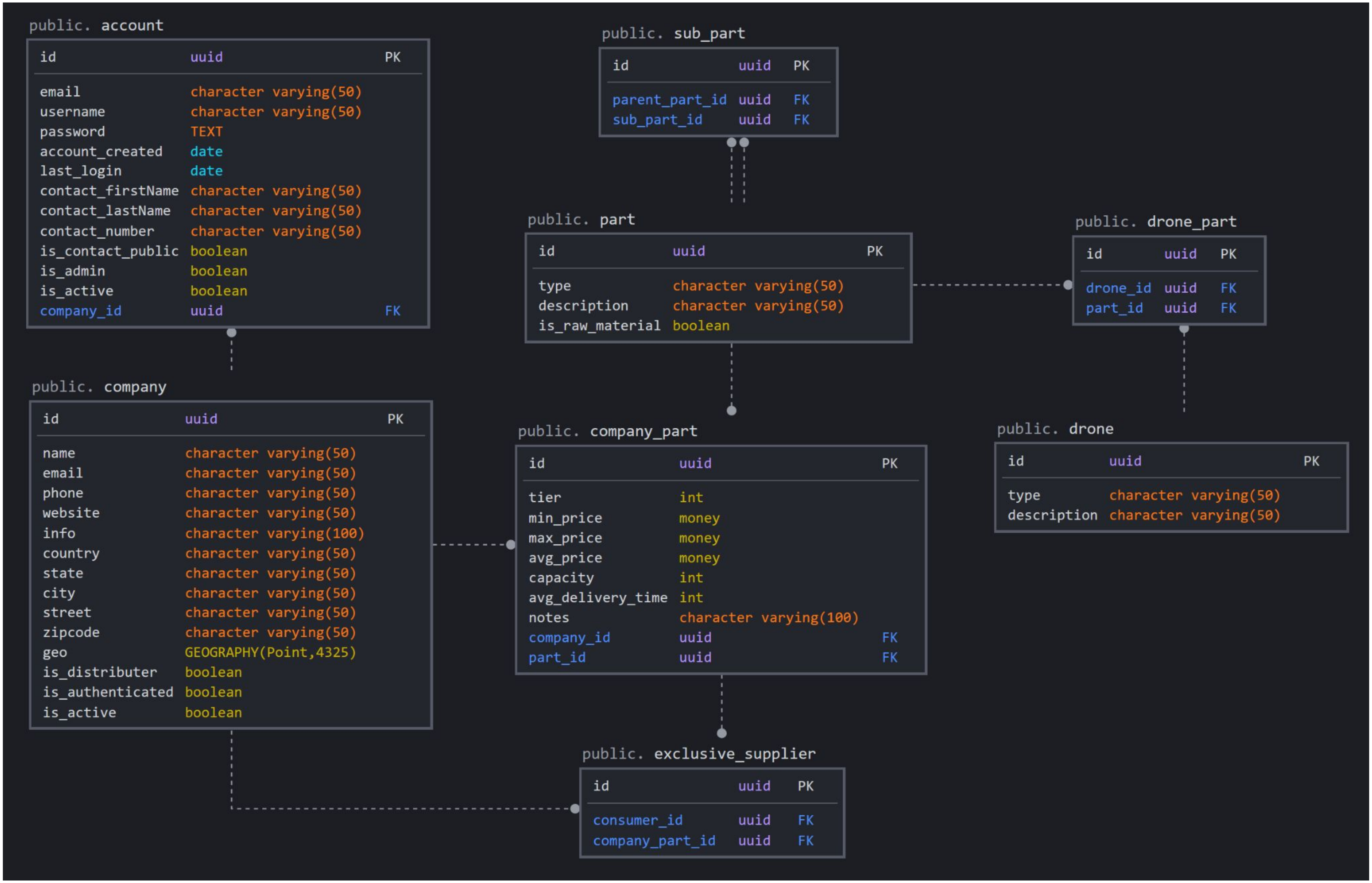
id	uuid	PK
drone_id	uuid	FK
part_id	uuid	FK



public. sub_part

id	uuid	PK
parent_part_id	uuid	FK
sub_part_id	uuid	FK

Database Diagram



public. account

id	uuid	PK
email	character varying(50)	
username	character varying(50)	
password	TEXT	
account_created	date	
last_login	date	
contact_firstName	character varying(50)	
contact_lastName	character varying(50)	
contact_number	character varying(50)	
is_contact_public	boolean	
is_admin	boolean	
is_active	boolean	
company_id	uuid	FK

public. sub_part

id	uuid	PK
parent_part_id	uuid	FK
sub_part_id	uuid	FK

public. part

id	uuid	PK
type	character varying(50)	
description	character varying(50)	
is_raw_material	boolean	

public. drone_part

id	uuid	PK
drone_id	uuid	FK
part_id	uuid	FK

public. company

id	uuid	PK
name	character varying(50)	
email	character varying(50)	
phone	character varying(50)	
website	character varying(50)	
info	character varying(100)	
country	character varying(50)	
state	character varying(50)	
city	character varying(50)	
street	character varying(50)	
zipcode	character varying(50)	
geo	GEOGRAPHY(Point,4325)	
is_distributer	boolean	
is_authenticated	boolean	
is_active	boolean	

public. company_part

id	uuid	PK
tier	int	
min_price	money	
max_price	money	
avg_price	money	
capacity	int	
avg_delivery_time	int	
notes	character varying(100)	
company_id	uuid	FK
part_id	uuid	FK

public. drone

id	uuid	PK
type	character varying(50)	
description	character varying(50)	

public. exclusive_supplier

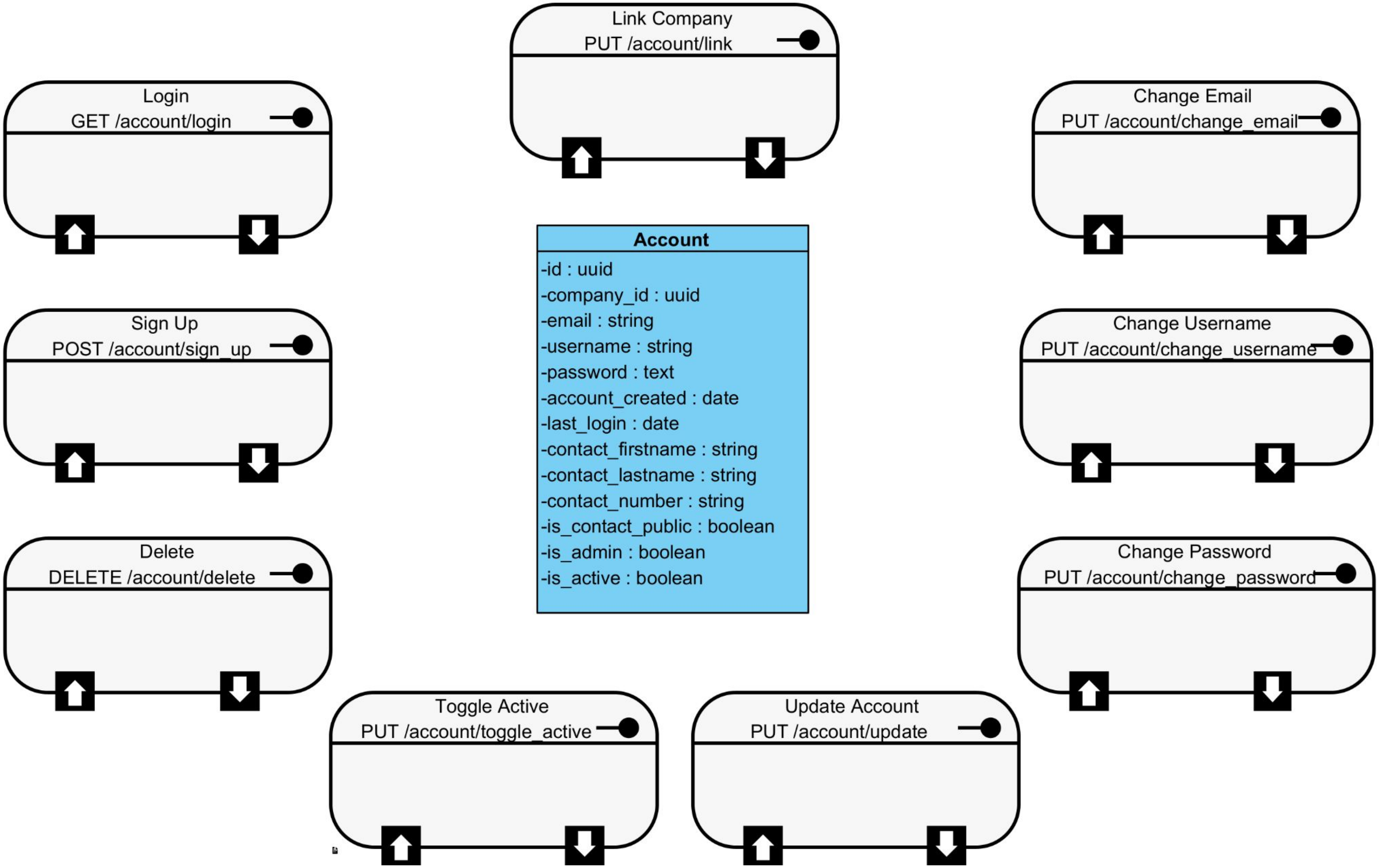
id	uuid	PK
consumer_id	uuid	FK
company_part_id	uuid	FK

API Design

Made with Node.js, Express, Sequelize, + more

- With Node.js allowing the use of JavaScript along with other packages, we utilized Express to create multiple functions that can be executed through HTML request that utilize Sequelize to interact with the database and return with the query results to be used on the frontend.
- Sequelize let us define a model for each table in the database, this model works as a verification of variable name and type for each query.
- The API is modular, divided into 8 smaller APIs that correspond to each table in the database. This allows for clear division of API responsibility and makes the overall API easier to understand and implement. The main backend server file connects each of these APIs allowing each module to work alongside one another and the separation be invisible to the rest of the software stack.

Account API



Link Company
PUT /account/link

Login
GET /account/login

Change Email
PUT /account/change_email

Account

-id : uuid
-company_id : uuid
-email : string
-username : string
-password : text
-account_created : date
-last_login : date
-contact_firstname : string
-contact_lastname : string
-contact_number : string
-is_contact_public : boolean
-is_admin : boolean
-is_active : boolean

Sign Up
POST /account/sign_up

Change Username
PUT /account/change_username

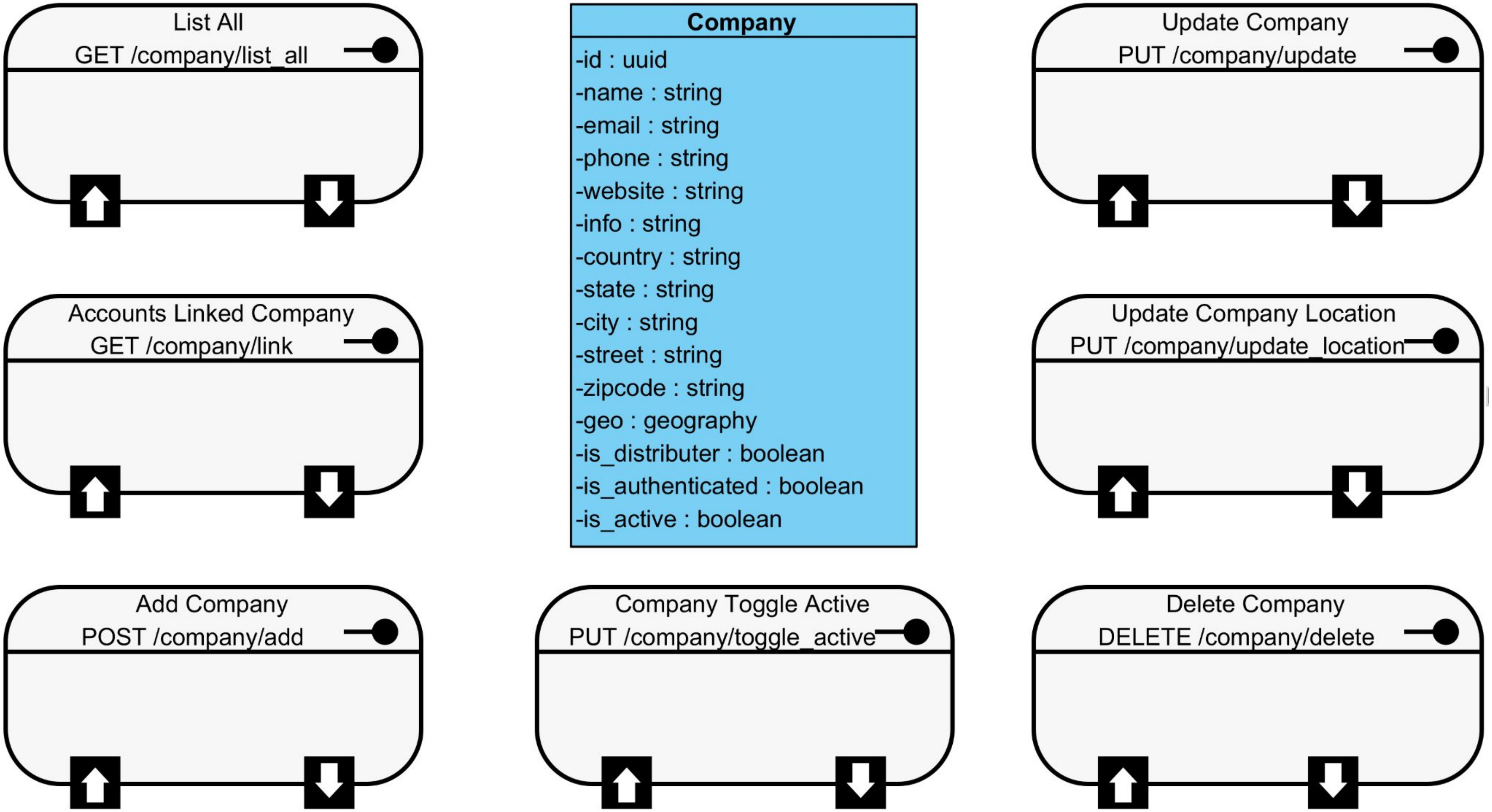
Delete
DELETE /account/delete

Change Password
PUT /account/change_password

Toggle Active
PUT /account/toggle_active

Update Account
PUT /account/update

Company API



List All
GET /company/list_all

Accounts Linked Company
GET /company/link

Add Company
POST /company/add

Company

-id : uuid
-name : string
-email : string
-phone : string
-website : string
-info : string
-country : string
-state : string
-city : string
-street : string
-zipcode : string
-geo : geography
-is_distributer : boolean
-is_authenticated : boolean
-is_active : boolean

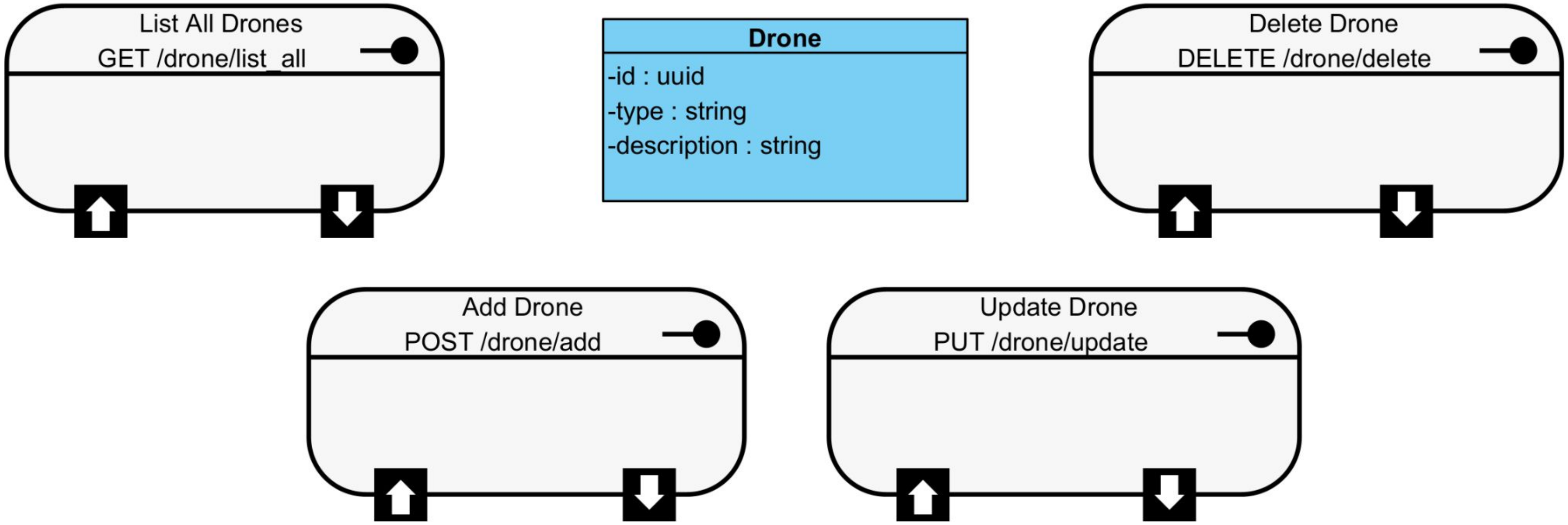
Company Toggle Active
PUT /company/toggle_active

Update Company
PUT /company/update

Update Company Location
PUT /company/update_location

Delete Company
DELETE /company/delete

Drone API



List All Drones
GET /drone/list_all

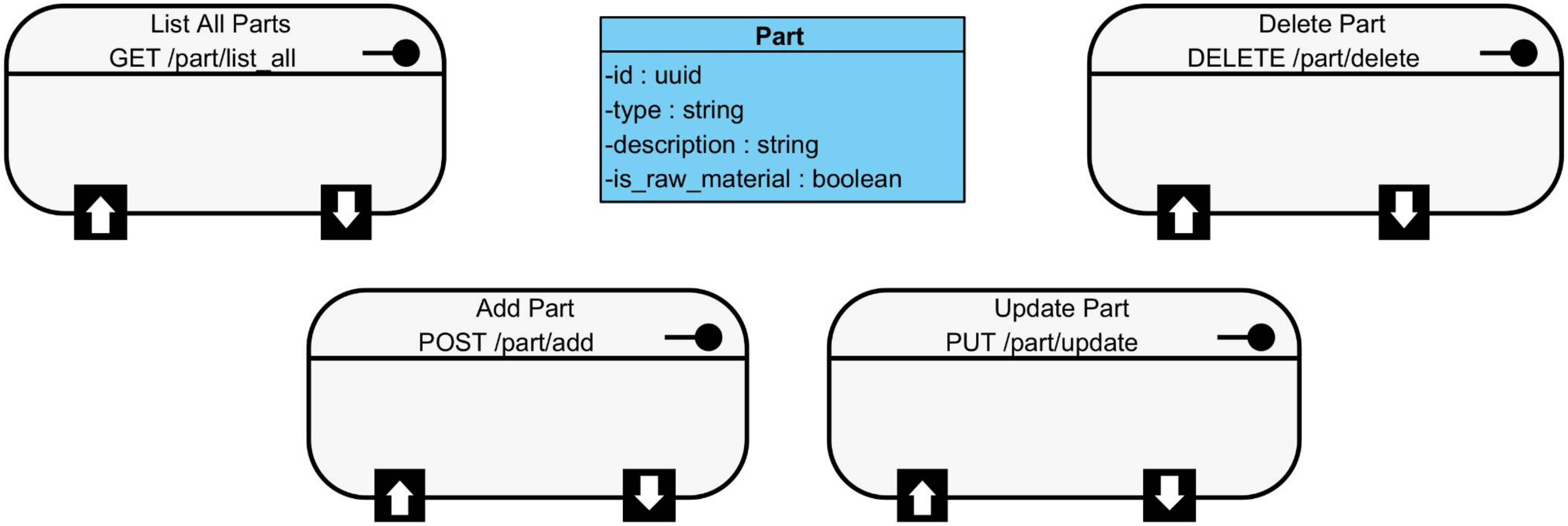
Drone
-id : uuid
-type : string
-description : string

Delete Drone
DELETE /drone/delete

Add Drone
POST /drone/add

Update Drone
PUT /drone/update

Part API



List All Parts
GET /part/list_all

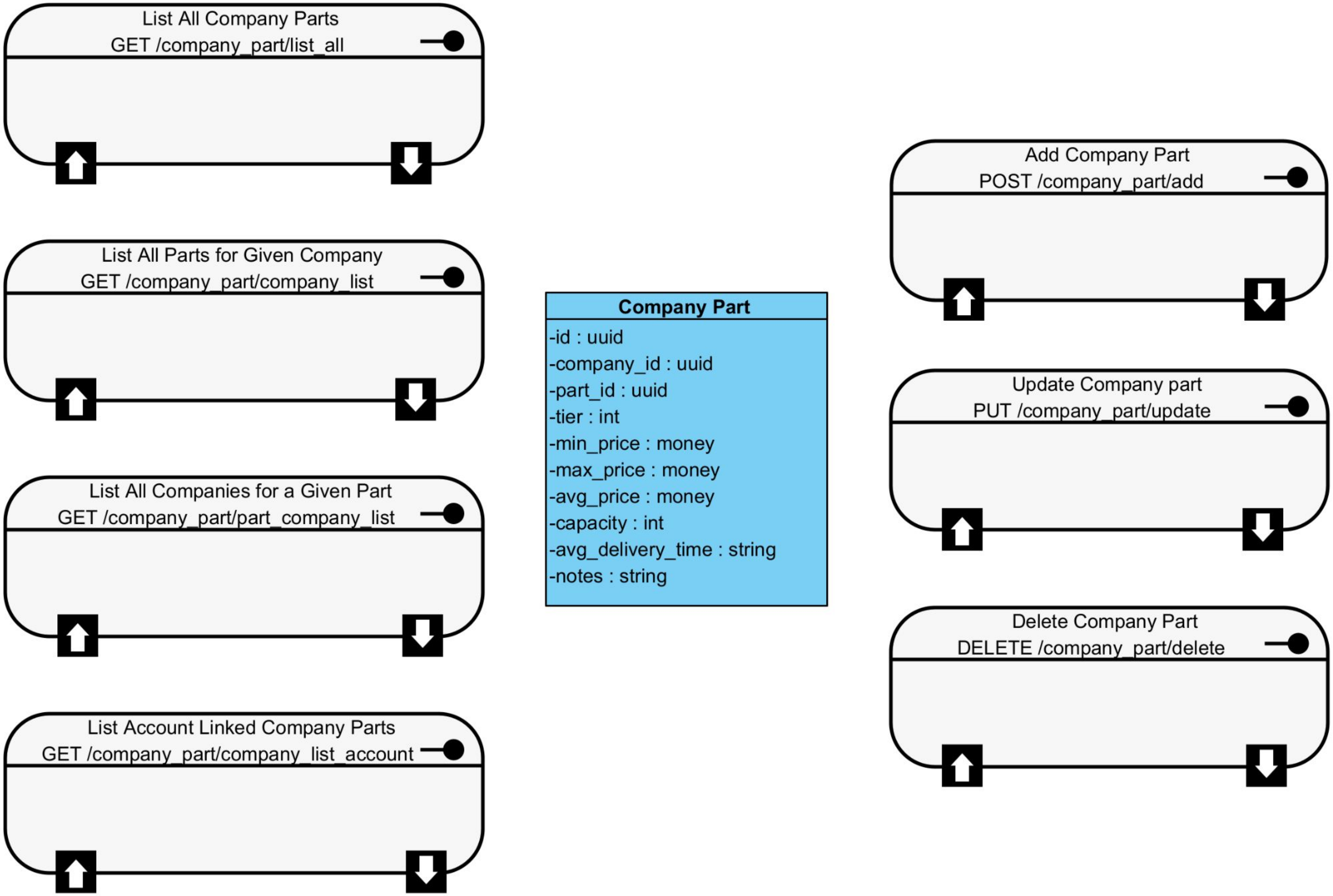
Part
-id : uuid
-type : string
-description : string
-is_raw_material : boolean

Delete Part
DELETE /part/delete

Add Part
POST /part/add

Update Part
PUT /part/update

Company Part API



List All Company Parts
GET /company_part/list_all

List All Parts for Given Company
GET /company_part/company_list

List All Companies for a Given Part
GET /company_part/part_company_list

List Account Linked Company Parts
GET /company_part/company_list_account

Company Part

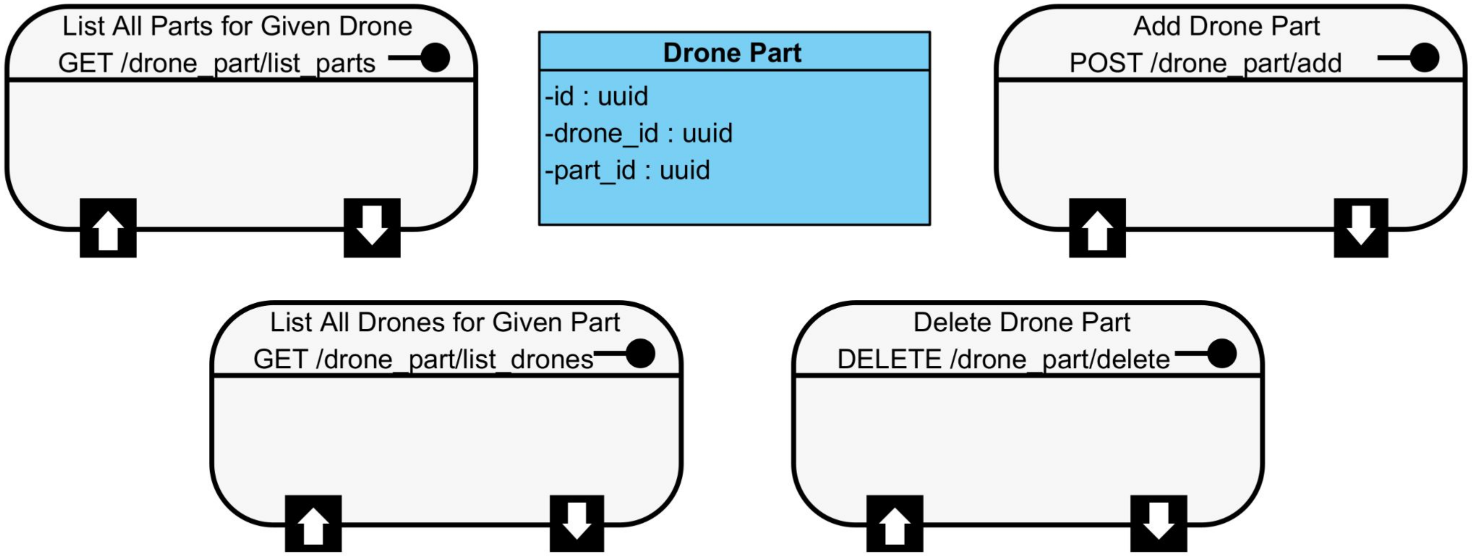
-id : uuid
-company_id : uuid
-part_id : uuid
-tier : int
-min_price : money
-max_price : money
-avg_price : money
-capacity : int
-avg_delivery_time : string
-notes : string

Add Company Part
POST /company_part/add

Update Company part
PUT /company_part/update

Delete Company Part
DELETE /company_part/delete

Drone Part API



List All Parts for Given Drone
GET /drone_part/list_parts

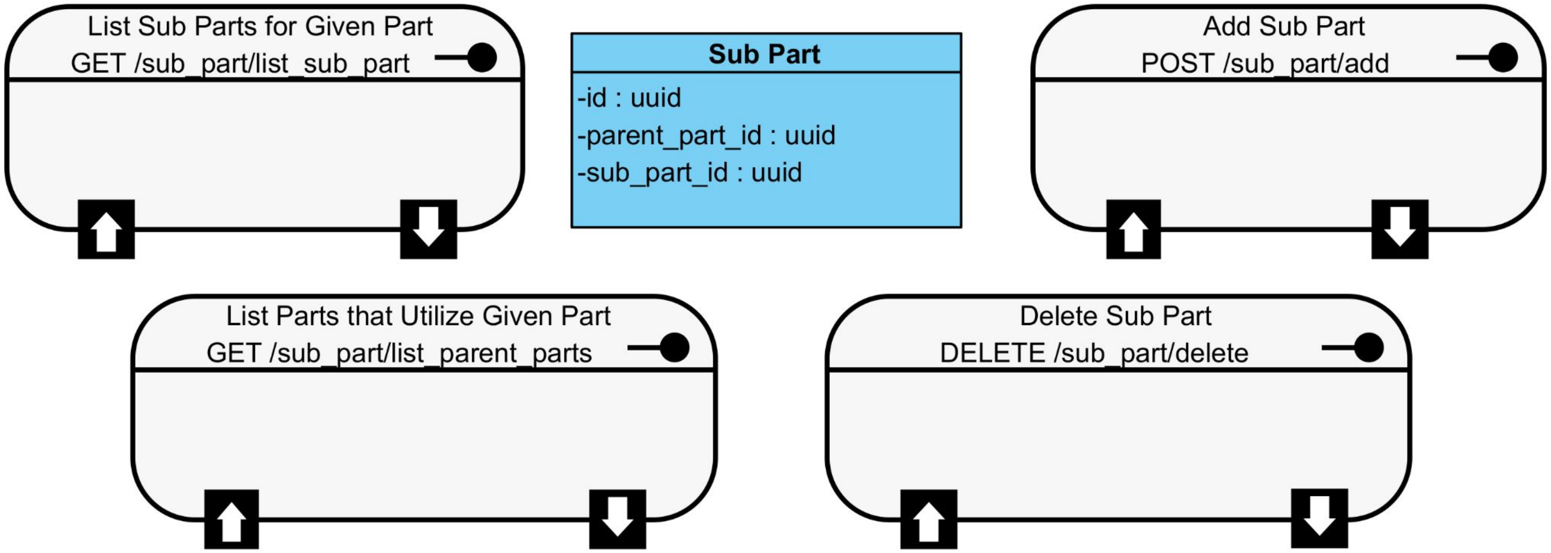
Drone Part
-id : uuid
-drone_id : uuid
-part_id : uuid

Add Drone Part
POST /drone_part/add

List All Drones for Given Part
GET /drone_part/list_drones

Delete Drone Part
DELETE /drone_part/delete

Sub Part API



List Sub Parts for Given Part
GET /sub_part/list_sub_part

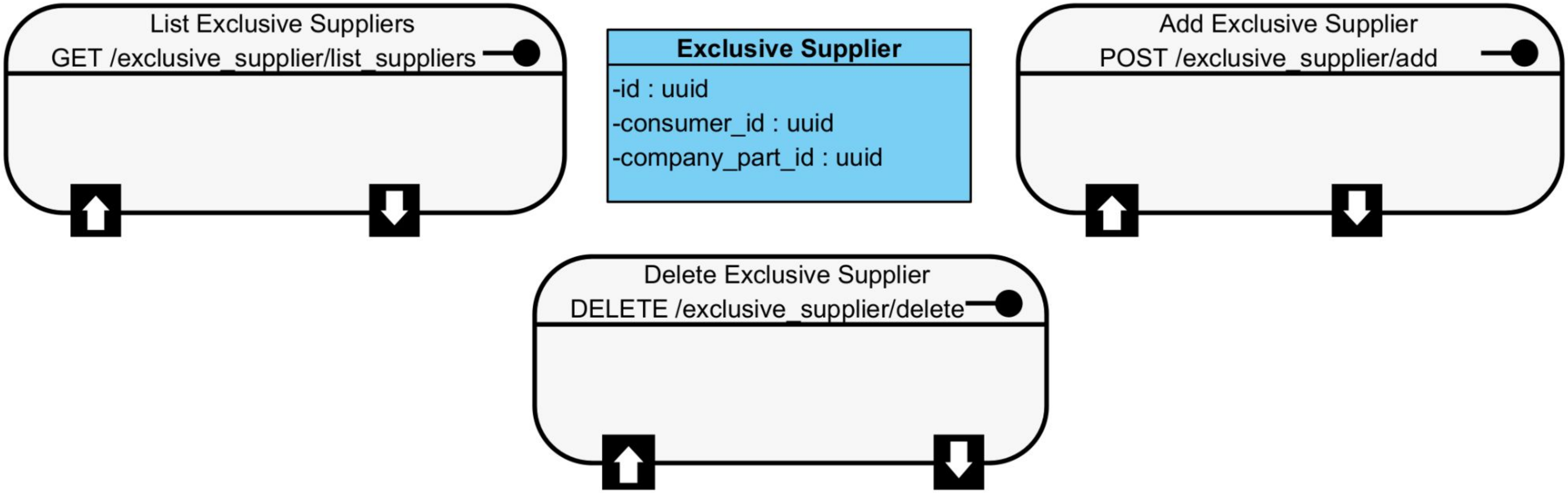
Sub Part
-id : uuid
-parent_part_id : uuid
-sub_part_id : uuid

Add Sub Part
POST /sub_part/add

List Parts that Utilize Given Part
GET /sub_part/list_parent_parts

Delete Sub Part
DELETE /sub_part/delete

Exclusive Supplier API



List Exclusive Suppliers

GET /exclusive_supplier/list_suppliers

Exclusive Supplier

-id : uuid

-consumer_id : uuid

-company_part_id : uuid

Add Exclusive Supplier

POST /exclusive_supplier/add

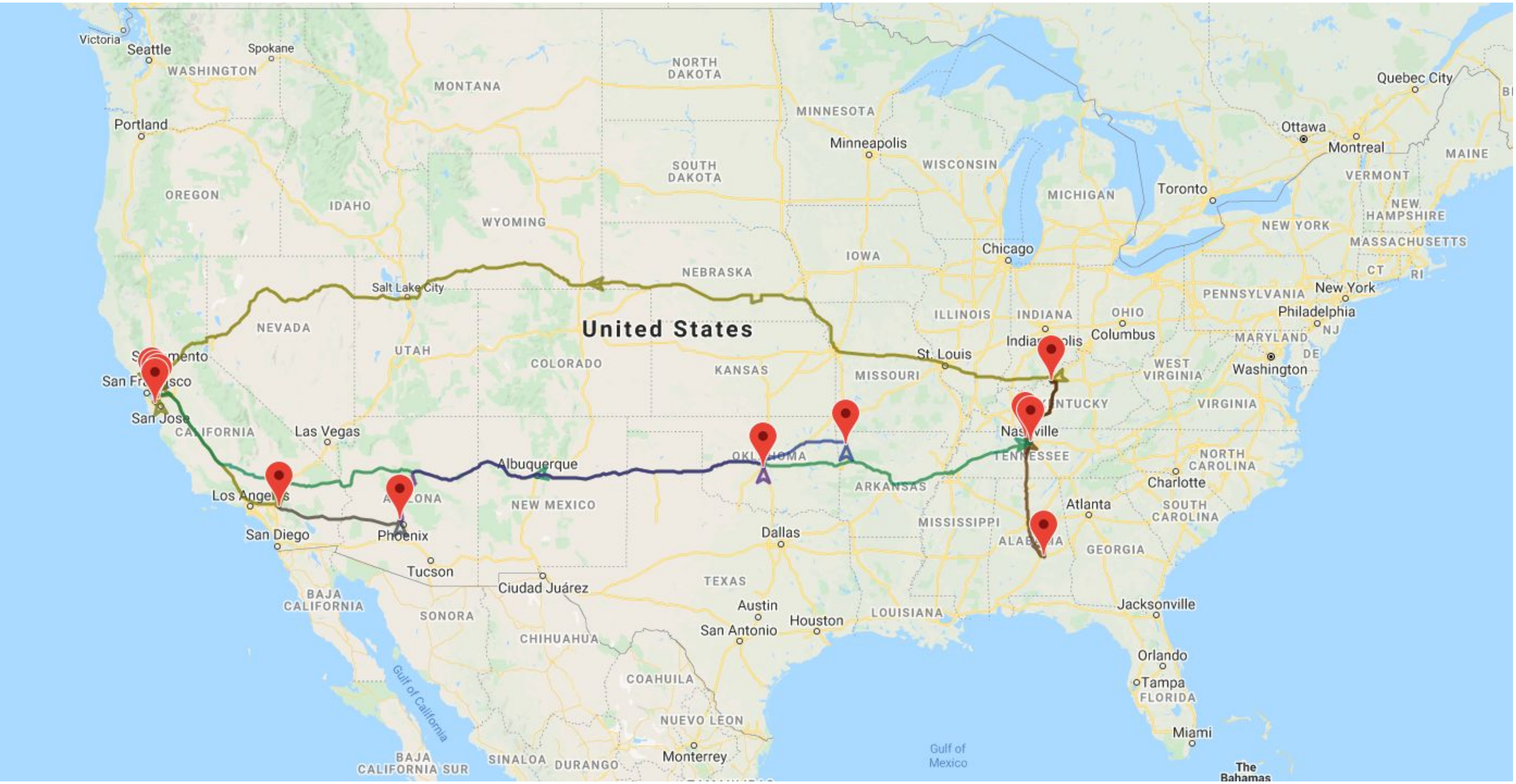
Delete Exclusive Supplier

DELETE /exclusive_supplier/delete

Database Capabilities

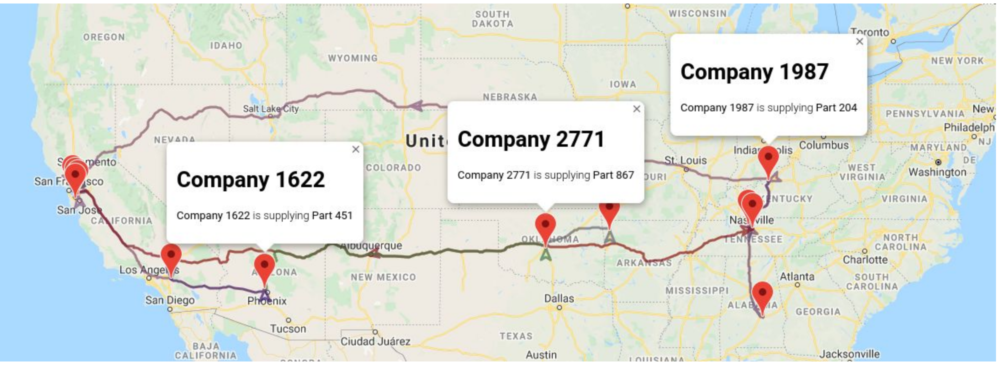
- Storage of geospatial data allows for efficient queries based on distance.
  - Determination of nearest suppliers through geospatial data is significantly faster than using a map routing API.
- Recursive queries allow for full chain of offered products to be returned by location.
  - Fewer requests needed to the backend improve responsiveness especially in large supply chains.
- Companies with existing contracts can select which suppliers they will utilize.
  - Allows the tool to develop more accurate supplier-consumer chains for more accurate time estimations.
- Returned number of suppliers for each part can be modified.

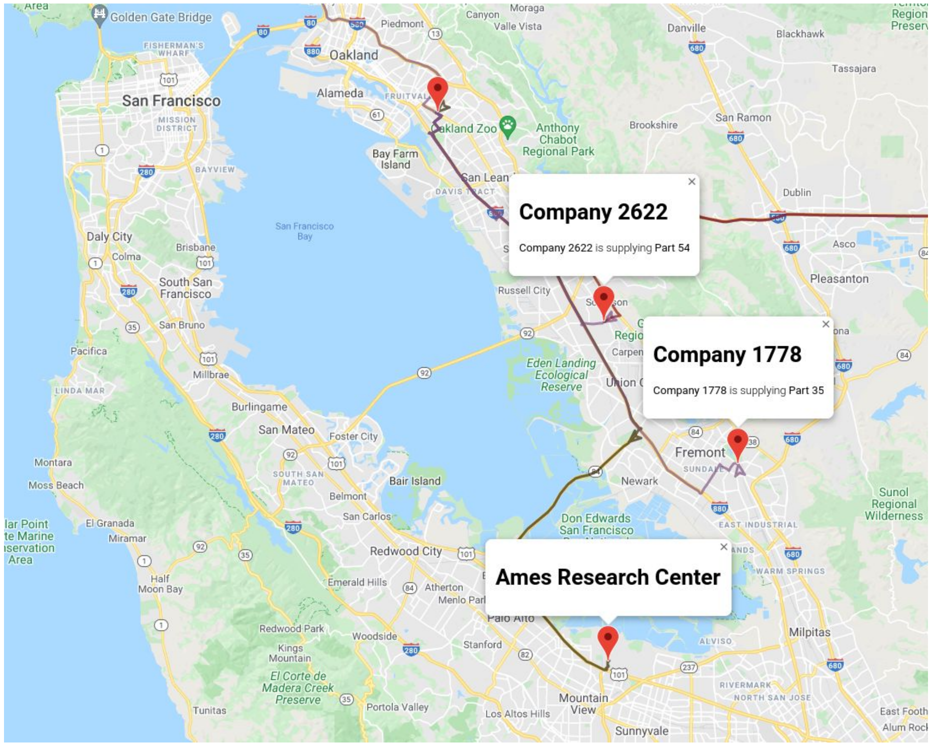
Mapping Capabilities



Mapping Capabilities

- Routing API specifically finds routes safe for shipping trucks.
- Company specific information available at each marker.
- Directionality of routes from supplier to consumer.
- Descriptions of parts available at each point.





Future Mapping Capabilities

- Estimation of risk for each link of the supply chain.
    - Risk could be estimated as a function of the number of companies supplying required parts and distance to these companies.
- Inclusion of current stock and lead times for new products.
    - This will require cooperation with suppliers in order to maintain and update accurate stock information.
- Stock and lead time inclusion in delivery timeframe estimation.

Conclusions & Future Research

Conclusions

- Use of API to interact with the database allows for easier development and allows for more modular code.
- Standardizing technologies across projects allows for easier access to data and widens information flow between projects.
- Prototyping the UI turns static designs and wireframes into interactive prototypes and makes extracting HTML / CSS easier.

Future Research

- Designing a Virtual Exchange Platform
  its fundamental to support NASA and the community's vision of Advanced Air Mobility (AAM).
- More information about Manufacturers and OEMS are needed to create a robust supply chain database with enough entries to determine if all gaps are filled.

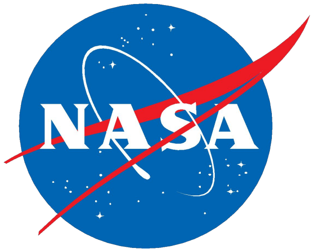
Credit: NASA Image and Video Library

Acknowledgments

Our internship team would like to extend special thanks to **Michael Day**, **Parimal Kopardekar** and **Christine Clark** for their support & mentorship to further the project's initiatives within the growing industry.

The team would also like to especially thank **Walter Harper**, **Raj Pai**, **Michael Roberts**, **Roshan Kalghatgi**, **Jim Williams**, **Rahul Srinivasan** for furthering the project's success from by their extensive insights.

Additionally, another special thanks to **Haley Feck**, and the **National Space Grant Foundation** for their continuous support & funding.


NASA