
A Simple & Clean Flight Tracker & Manager

# Aero Server - Backend

## How do i run this locally?

### Environment

You must include the following in `/server/.env`

```
PORT = PORT (ex: 3000)
DB_HOST = MongoDb
```

### Terminal

Open the terminal and run the following from `/`

```
cd server
npm install
npm run tsc
npm start
```

You can also run

```
cd server
npm install
npm run develop
```

## Airplane Controller

**`createAirplane`**

This function is for the creation of an individual airplane when given the id, seats and the airline name. It will create a new airplane and save it to the database

**`listAirplanes`**

This function is for going through the database and returning all the planes that are currently in it.

## Flight Controller

**`createFlight`**

This function is where the flight is being created. When given all of the necessary details such as departure date and arrival date a new flight is create and saved to the database.

**`listFlights`**

This function simply returns all the flights from the database in descending order based off of the arrival date of the flight.

**`getFlightByID`**

This function is meant to return a specific flight when given a specific id. It simply goes through the database and returns the flight the matched the id.

**`getFlightSearchWithDateRange`**

This function is used to find all the flights that fall between a specific date range. It takes a date range given and then compares the range given to the respective arrival and departure dates of all the flights in the databasea and then only returns the flights that fall in the range.

`listings`

This function is meant for the purchasing of tickets. When given an id it will find the flight corresponding to it and then purchaseOptions are devised and then depending on who is selling the ticket their respective information is returned.

`createRating`

This functino is meant to add a rating to a flight. When given an id it will go through the database to find the respective flight and allow the user to leave a comment and rating on the flight.

`getReturnFlightsByID`

This function is for getting all the return flights from a specific location based off an id. When given an id the function will find the flight from withing the data and then return all the returning flight from the same location of the given flight back to where the flight departed from.

`updateFlightStatus`

This function is used for changing the specific status of an individual flight. It will go through all the flights and when it finds the specific flight that corresponds to the given id it will change the status to what the user determines.

`getFlightByAirline`

This function is meant to simply return all the flights that are coming from a specific airline. It will simply query through the database of flights and return all the flights whose airline name corresponds to the one given to the function.

## Airport Controller

`createAirport`

This function is used for creating airports when given the name and the city. It will create and save a new airport to the database.

`listAirports`

This function is for going through the database and returning all the airports that are currently present.

## Transaction Controller

`createTransaction`

This funcion does quite a few tasks. It will essentially create and save a transaction, purchaseinfo and a ticket. It only returns the ticket though. The function is essentially given all the information necessary for transactions, purchases and tickets and add all the information to the respective data types and saves all of them to the database to be accessed later.

## User Controller

`revenue`

This function is for calculating the revenue of the respective individual whether it be a booking agent or airline staff. This function isnt accessible to a user who is just a customer. It is also shown as a devision over months to show where revenue was made per month.

`frequent`

This function is also not available for any user who is just a customer. When the user is a booking agent it calculates the top customers based off of tickets bought as well as top customers based on the commission from tickets. When the user is a staff member it calculates customers who have flown most frequently on a particular airline.

`tickets`

This functions is for finding all the tickets that belong a specific user. When given an id, the user corresponding to the id is found and all the tickets related to them are returned.

`flights`

This function is for orienting the flights of a specifc user. The function is given an id which is uses to find the respective user and then returns all of their previous flights as well as all of their upcoming ones.

`getUser`

This function is simply for finding and returning the specifc user when given their id. The database will be searched to find the user with the same id and return them.

`login`

This function is for logging in when you have already been register. It simply does the correct checks if the users email and password are in the database and if they are they get logged in otherwise they receive an error.

`signup`

This function is for allowing people to sign up as one of three types of users. They can sign up as a customer, agent or staff. Customers must fill data specific to them just as email, password, name, and address where as agent require different information such as commission and staff also require different information. After all the respective information is given the user is saved to the database with their information which will allow them to login later.

## Destination Controller

`createDestination`

This function is simply for create and saving a new destination. When the information is given for the destination a new destination with the same information is saved and added to the database.

`listDestinations`

This function simply going through the databases and returns all the destinations that are currently in it.

## Flight Routes

### get `/flights/view`

https://projectaero-api.herokuapp.com/#/Flights/get_flights_view

### post `/flights`

https://projectaero-api.herokuapp.com/#/Flights/post_flights

### get `/flights`

https://projectaero-api.herokuapp.com/#/Flights/get_flights

### get `/flights/:id`

https://projectaero-api.herokuapp.com/#/Flights/getFlightById

### patch `/flights/:id/status`

https://projectaero-api.herokuapp.com/#/Flights/updateFlightStatus

### get `/flights/search/:query`

https://projectaero-api.herokuapp.com/#/Flights/searchFlightByQuery

### get `/flights/:id/returns`

https://projectaero-api.herokuapp.com/#/Flights/returnFlightByID

### get `/flights/search/:departure_date/:arrival_date/:depature_airport/:arrival_airport`

https://projectaero-api.herokuapp.com/#/Flights/returnFlightByDateAirport

### get `/flights/airline/:airline`

https://projectaero-api.herokuapp.com/#/Flights/returnFlightByAirline

### get `/flights/:id/listings`

https://projectaero-api.herokuapp.com/#/Flights/listFlightById

### post `/flights/createRating`

https://projectaero-api.herokuapp.com/#/Flights/createRating

### get `/flights/:id/ratings`

https://projectaero-api.herokuapp.com/#/Flights/ratings

get `/flights/:id/:flightID/usersRatings`

https://projectaero-api.herokuapp.com/#/Flights/userRatings

## User Routes

get `/user/login/:email/:password`

https://projectaero-api.herokuapp.com/#/User/get_user_login__email___password_

get `/user/:id`

https://projectaero-api.herokuapp.com/#/User/get_user__id_

post `/user/signup`

https://projectaero-api.herokuapp.com/#/User/post_user_signup

get `/user/:id/spending`

https://projectaero-api.herokuapp.com/#/User/get_user__id__spending

get `/user/:id/revenue`

https://projectaero-api.herokuapp.com/#/User/get_user__id__revenue

get `/user/:id/tickets`

https://projectaero-api.herokuapp.com/#/User/get_user__id__tickets

get `/user/:id/flights`

https://projectaero-api.herokuapp.com/#/User/get_user__id__flights

get `/user/:id/frequent`

https://projectaero-api.herokuapp.com/#/User/frequent

## Airport Routes

post `/airports`

https://projectaero-api.herokuapp.com/#/Airports/post_airports

get `/airports`

https://projectaero-api.herokuapp.com/#/Airports/get_airports

## Destinations

post `/destinations`

https://projectaero-api.herokuapp.com/#/Destinations/post_destinations

get `/destinations`

https://projectaero-api.herokuapp.com/#/Destinations/get_destinations

## Transactions

post `/transactions`

https://projectaero-api.herokuapp.com/#/Transactions/post_transaction

## Models

`all models`

https://projectaero-api.herokuapp.com/#/