Lab2 Report

By

Jason Kim (50123816/jaeheunk)

Jason Pettrone (50224194/japettro)

Table of Contents

Project Structure

Lab

- Part1
- Part2
- **Part3**
    - **web**
        - **airline1**(web application for airline1)
            - abi.json
            - app.py(Python3 script to run the web app)
            - templates(folder that contains HTML files used by airline1)
            - static(folder that contains every image file used by airline1)
        - **airline2**(web application for airline2)
            - abi.json
            - app.py(Python3 script to run the web app)
            - templates(folder that contains HTML files used by airline2)
            - static(folder that contains every image file used by airline2)
    - **smart_contract** (Our smart contract code and account details)
        - abi.json
        - Contract Details.txt (What addresses we use on the Ethereum network)
        - **Lab2Part3Contract.sol** (The smart contract we're using)

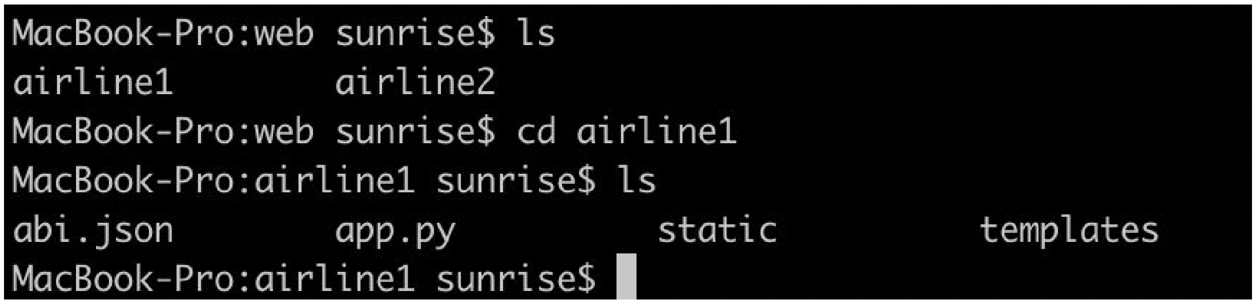
Dependencies

You need following Python3 modules:

- flask(https://pypi.org/project/Flask/)
- flask-pymongo(https://pypi.org/project/Flask-PyMongo/)
- Web3.py(https://pypi.org/project/web3/)
- flask_restful (https://pypi.org/project/Flask-RESTful/)

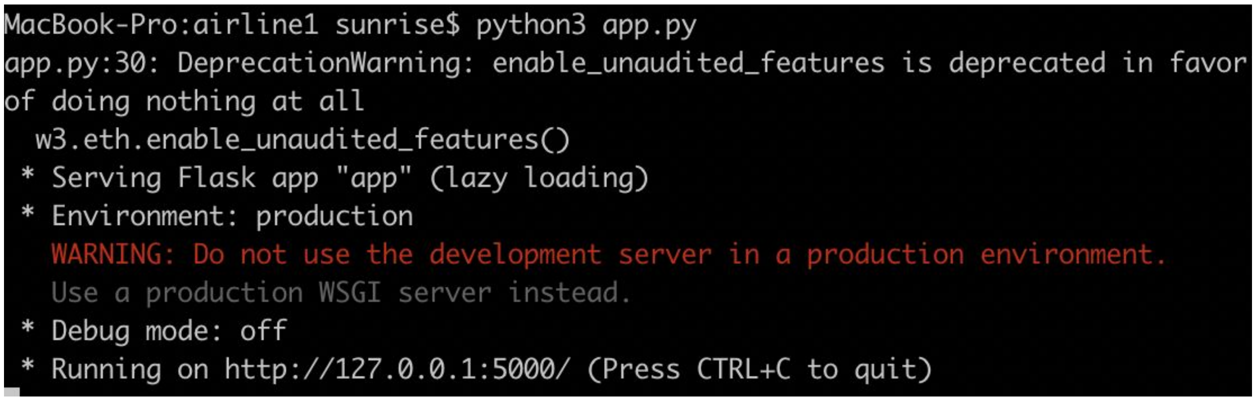
How to Run

1. Go to the directory where app.py for Airline1 is stored.



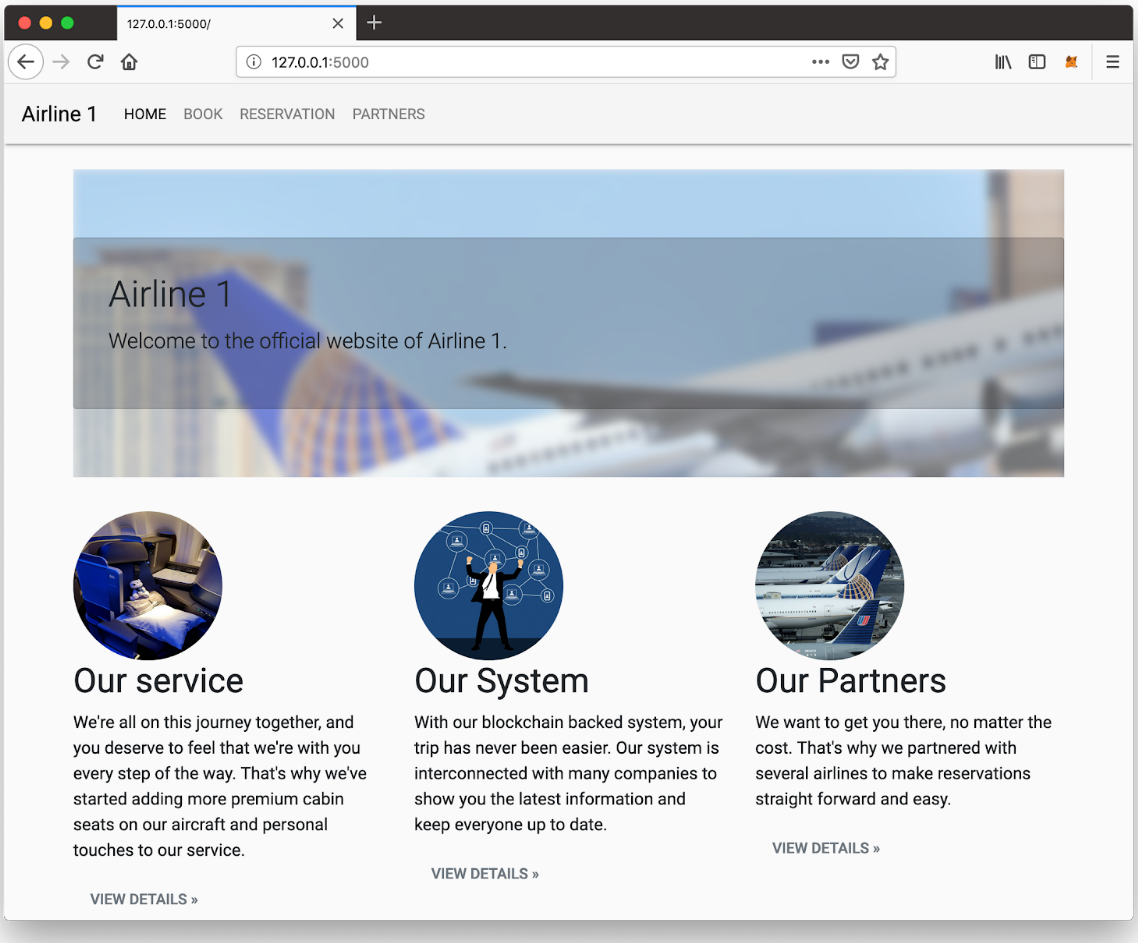
```
MacBook-Pro:web sunrise$ ls
airline1        airline2
MacBook-Pro:web sunrise$ cd airline1
MacBook-Pro:airline1 sunrise$ ls
abi.json        app.py          static          templates
MacBook-Pro:airline1 sunrise$
```

2. Run app.py using Python3



```
MacBook-Pro:airline1 sunrise$ python3 app.py
app.py:30: DeprecationWarning: enable_unaudited_features is deprecated in favor
of doing nothing at all
  w3.eth.enable_unaudited_features()
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

3. Now the Airline1 site should be accessible at 127.0.0.1:5000



Airline 1

Welcome to the official website of Airline 1.

Our service

We're all on this journey together, and you deserve to feel that we're with you every step of the way. That's why we've started adding more premium cabin seats on our aircraft and personal touches to our service.

VIEW DETAILS »

Our System

With our blockchain backed system, your trip has never been easier. Our system is interconnected with many companies to show you the latest information and keep everyone up to date.

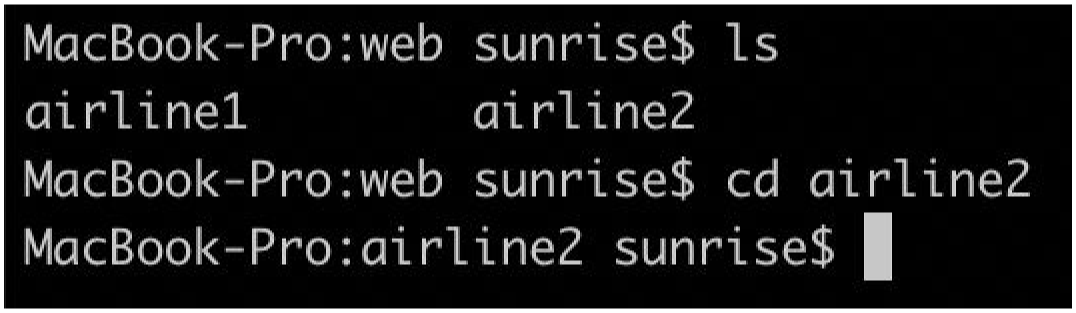
VIEW DETAILS »

Our Partners

We want to get you there, no matter the cost. That's why we partnered with several airlines to make reservations straight forward and easy.

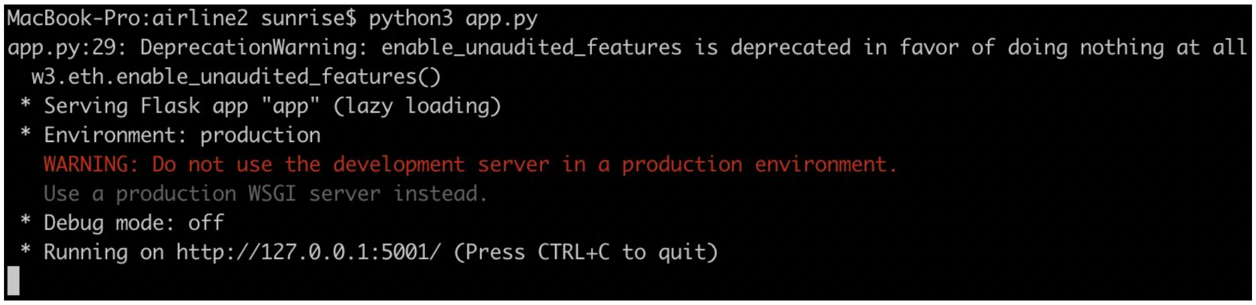
VIEW DETAILS »

4. Go to the directory where app.py for Airline2 is stored.



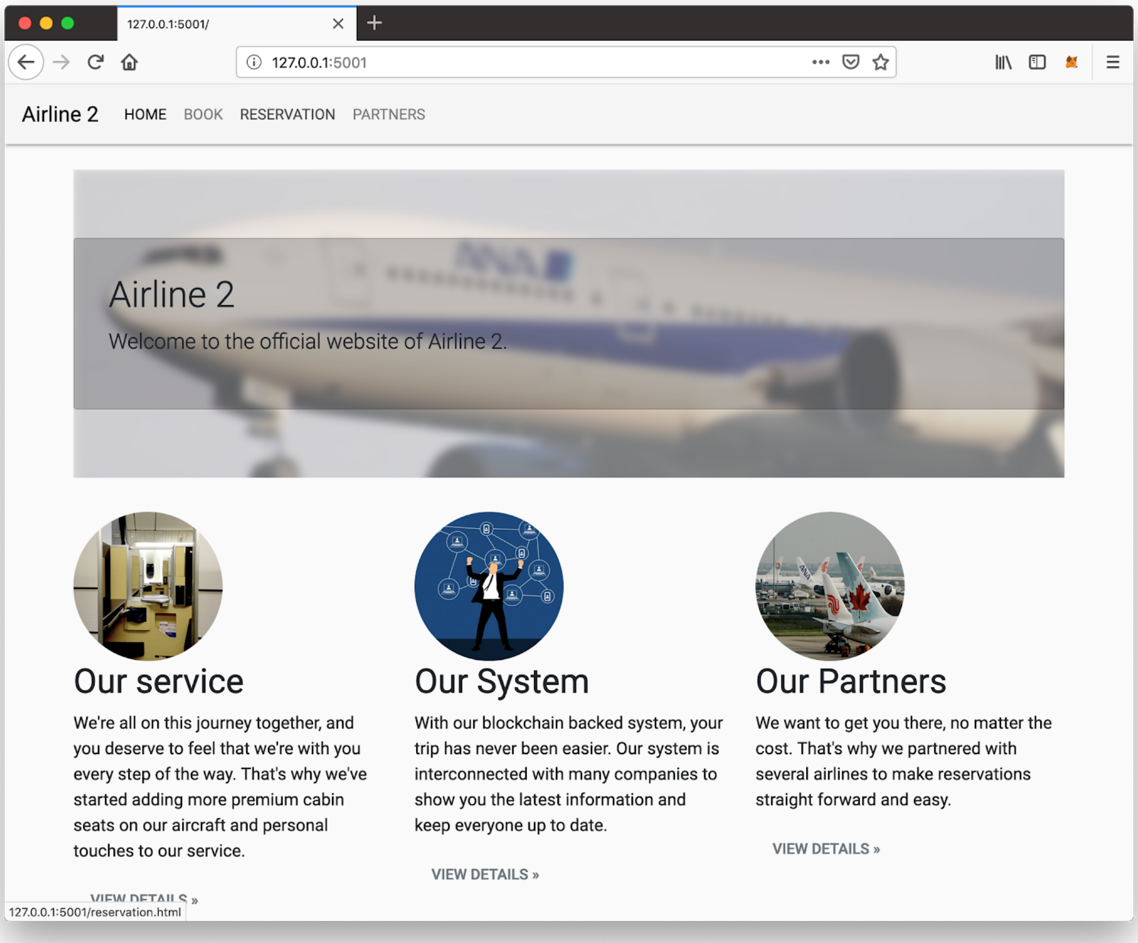
```
MacBook-Pro:web sunrise$ ls
airline1        airline2
MacBook-Pro:web sunrise$ cd airline2
MacBook-Pro:airline2 sunrise$ 
```

5. Run app.py using Python3



```
MacBook-Pro:airline2 sunrise$ python3 app.py
app.py:29: DeprecationWarning: enable_unaudited_features is deprecated in favor of doing nothing at all
  w3.eth.enable_unaudited_features()
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5001/ (Press CTRL+C to quit)
```

6. Now the Airline2 site should be accessible at 127.0.0.1:5001



Airline 2

Welcome to the official website of Airline 2.

Our service

We're all on this journey together, and you deserve to feel that we're with you every step of the way. That's why we've started adding more premium cabin seats on our aircraft and personal touches to our service.

VIEW DETAILS »

Our System

With our blockchain backed system, your trip has never been easier. Our system is interconnected with many companies to show you the latest information and keep everyone up to date.

VIEW DETAILS »

Our Partners

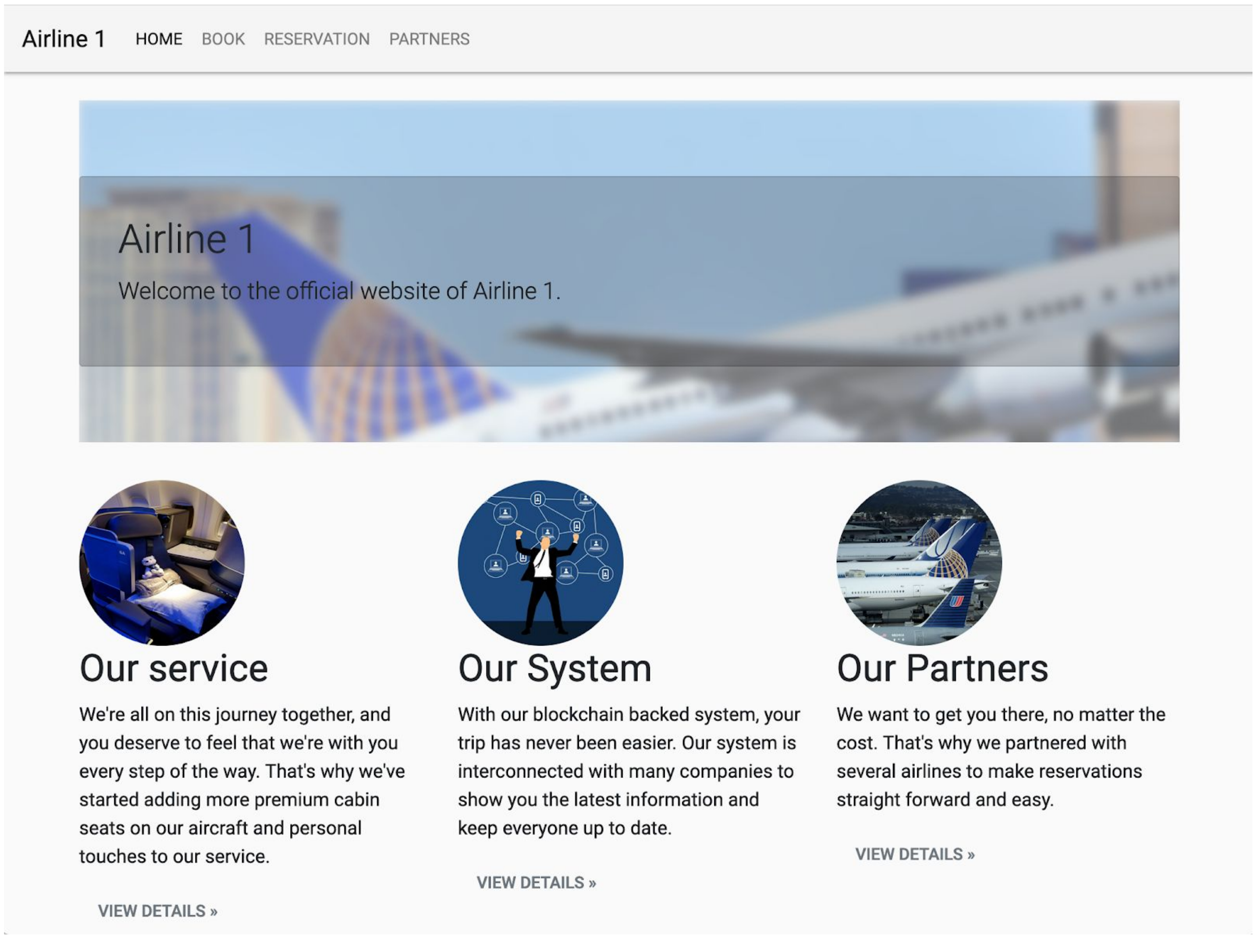
We want to get you there, no matter the cost. That's why we partnered with several airlines to make reservations straight forward and easy.

VIEW DETAILS »

127.0.0.1:5001/reservation.html

How to Test

1. Once both webapps are running, go to airline1's webpage.



Airline 1

Welcome to the official website of Airline 1.

Our service

We're all on this journey together, and you deserve to feel that we're with you every step of the way. That's why we've started adding more premium cabin seats on our aircraft and personal touches to our service.

VIEW DETAILS »

Our System

With our blockchain backed system, your trip has never been easier. Our system is interconnected with many companies to show you the latest information and keep everyone up to date.

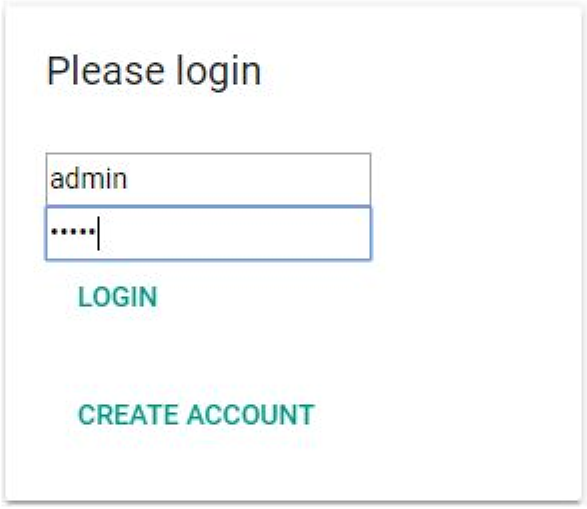
VIEW DETAILS »

Our Partners

We want to get you there, no matter the cost. That's why we partnered with several airlines to make reservations straight forward and easy.

VIEW DETAILS »

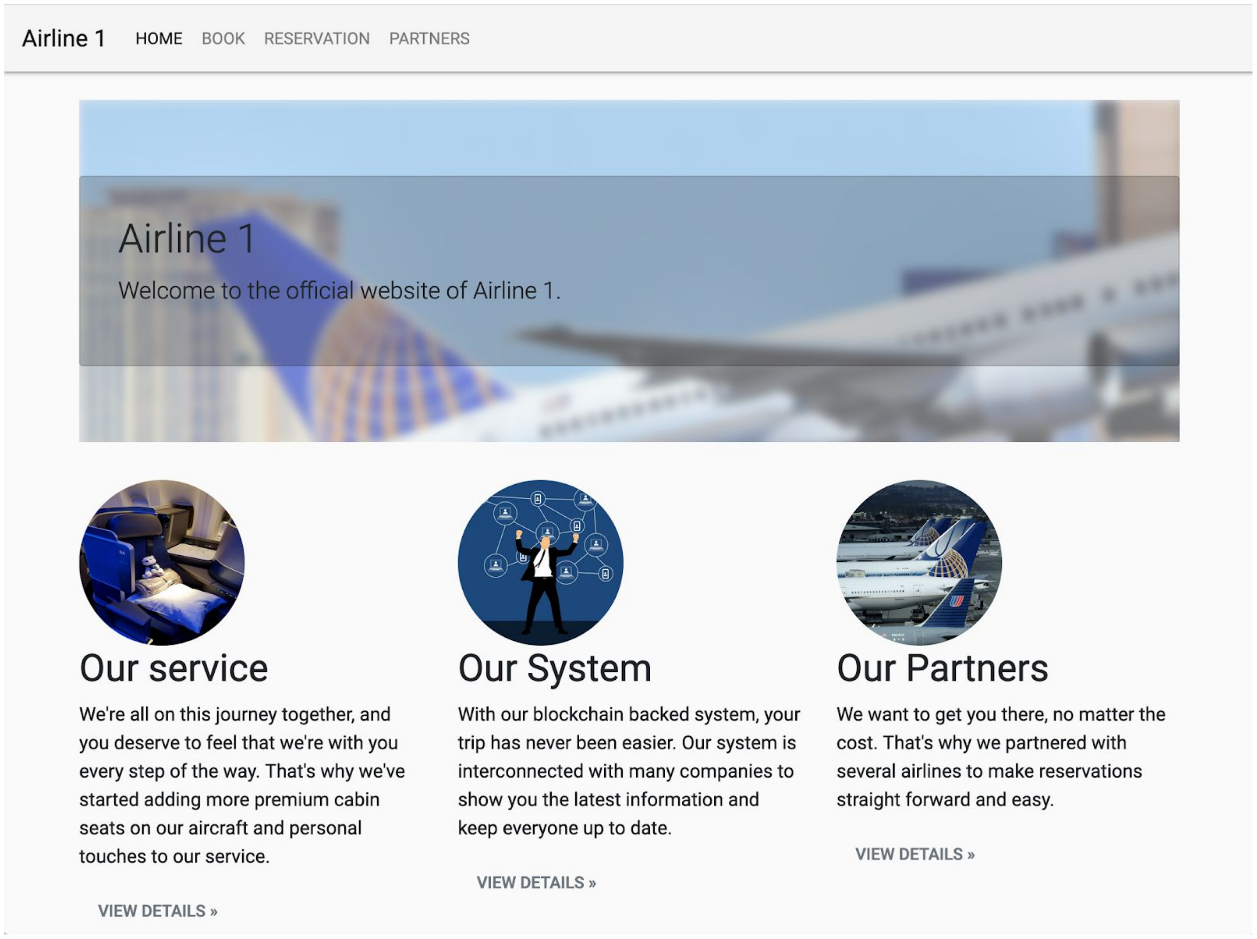
2. Click on the 'Reservation' button near the top of the screen.
3. A login screen should appear, type in 'admin' for the username and 'admin' for the password.


Please login

admin

•••••

4. Click the 'login' button. It should take you back to the home airline screen



Airline 1

Welcome to the official website of Airline 1.

Our service

We're all on this journey together, and you deserve to feel that we're with you every step of the way. That's why we've started adding more premium cabin seats on our aircraft and personal touches to our service.

VIEW DETAILS »

Our System

With our blockchain backed system, your trip has never been easier. Our system is interconnected with many companies to show you the latest information and keep everyone up to date.

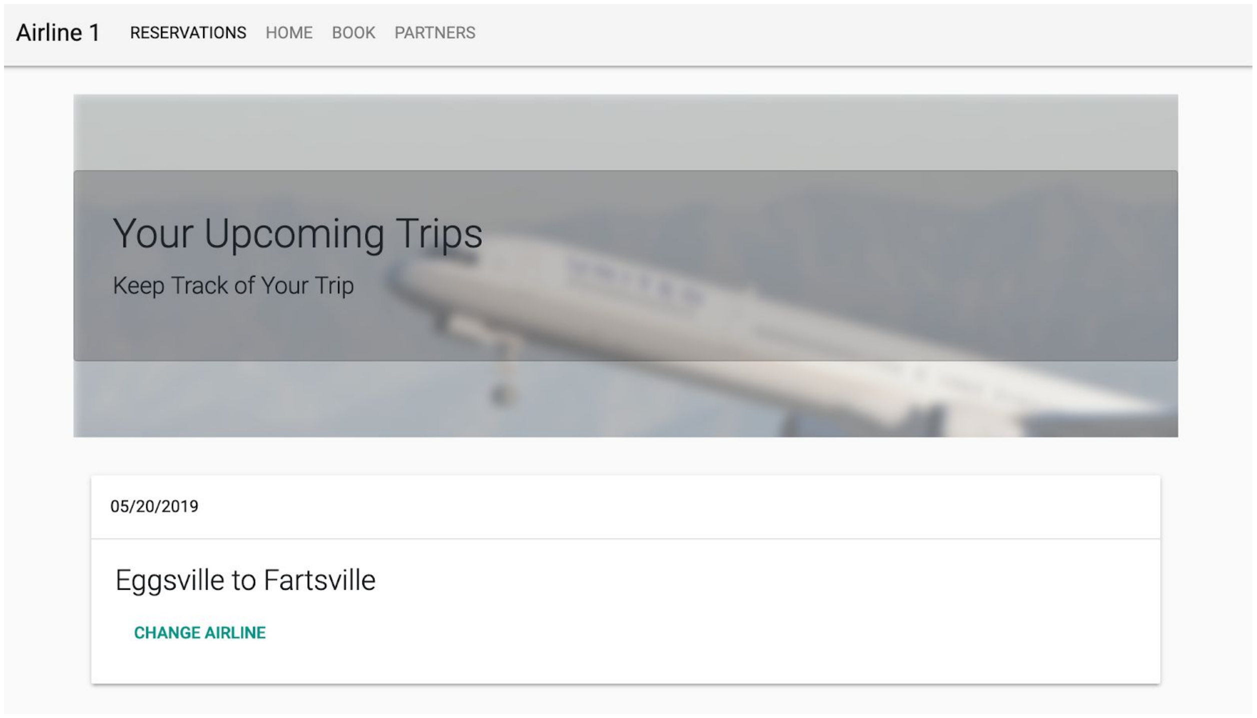
VIEW DETAILS »

Our Partners

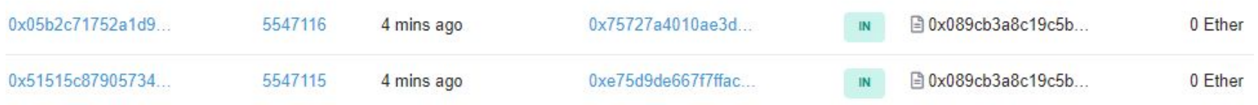
We want to get you there, no matter the cost. That's why we partnered with several airlines to make reservations straight forward and easy.

VIEW DETAILS »

5. Click the Reservation button again on the top



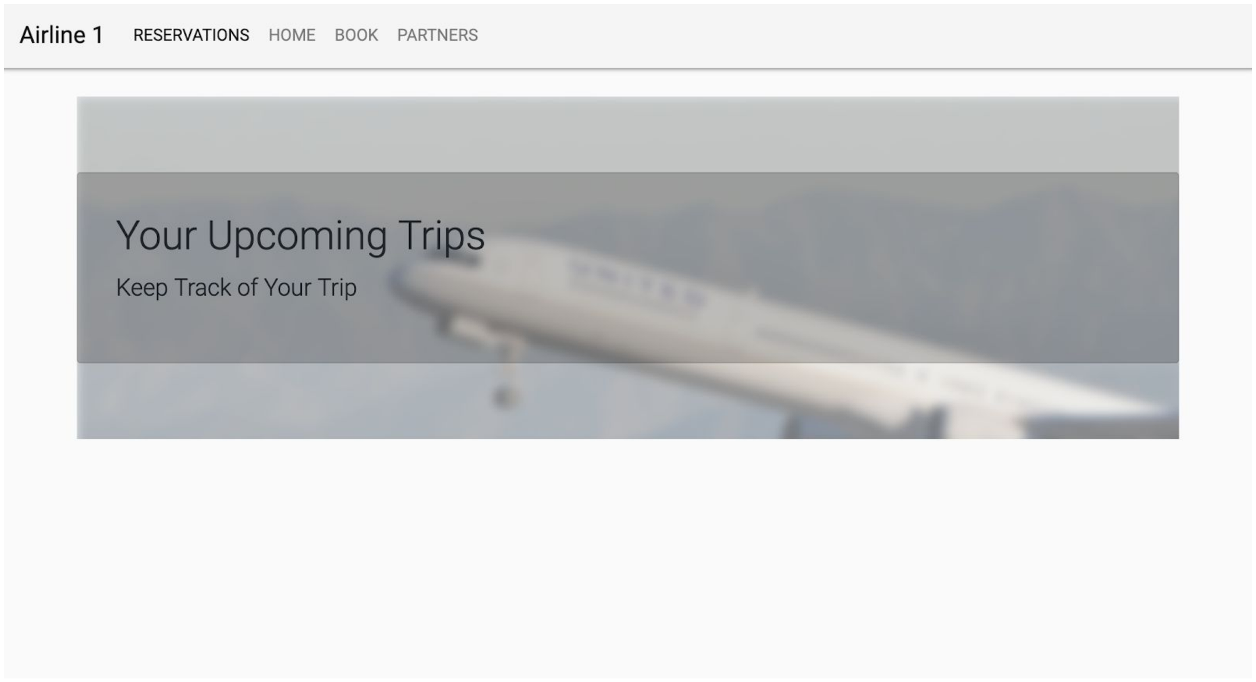
6. Click change airline, it should take some time as it does a transaction over the Ethereum network. After it's done it should redirect you back to the home screen.
7. To verify the transaction took place, go to the following etherscan link: https://ropsten.etherscan.io/address/0x089cb3a8c19c5b20cabadf0691dad52083fd7aa2
8. You should see two transactions similar to this:



0x05b2c71752a1d9...	5547116	4 mins ago	0x75727a4010ae3d...	IN	0x089cb3a8c19c5b...	0 Ether
0x51515c87905734...	5547115	4 mins ago	0xe75d9de667f7ffac...	IN	0x089cb3a8c19c5b...	0 Ether

9. The earlier transaction is the request from airline1 and the later transaction is the response from airline2.
10. The DBs of each airlines site will update and transfer user information and reservation data. In this case, airline1 will delete the reservation data from their DB and airline2 will add the reservation data to their DB.

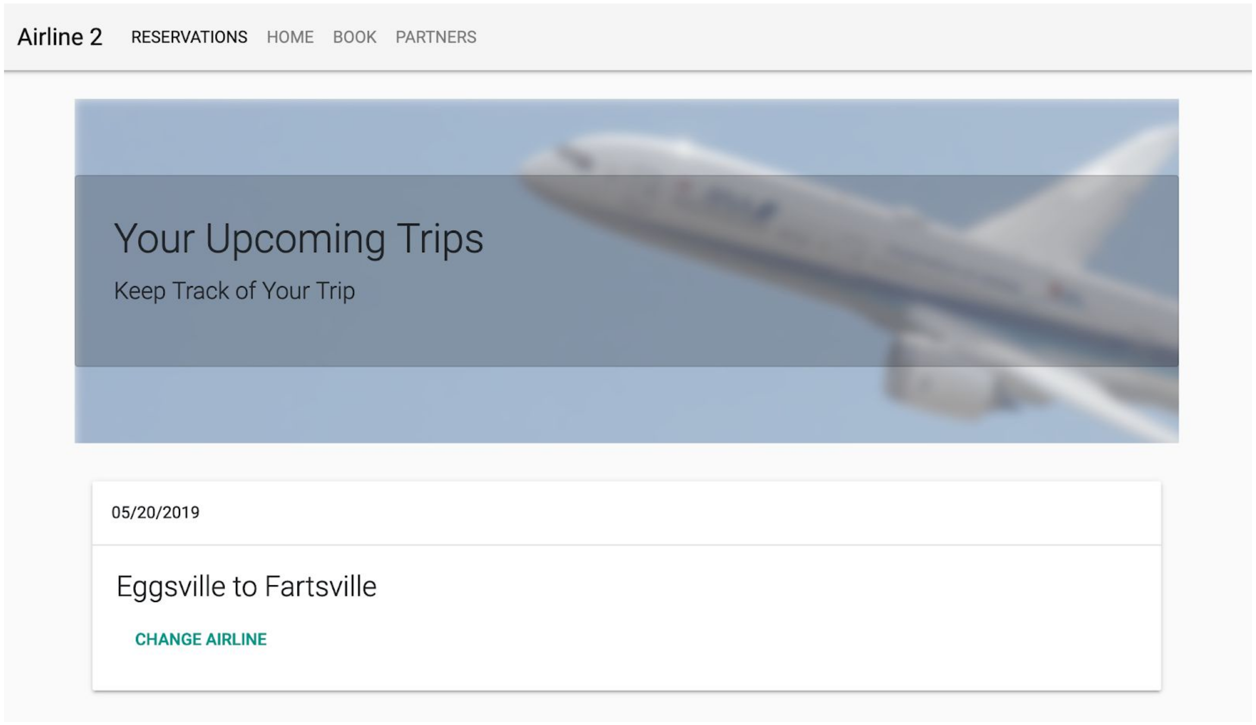
11. Now if you check your reservation on the airline1 website you can see that you have no reservation



Your Upcoming Trips

Keep Track of Your Trip

12. If you go to the airline2 website and login with the same credential(admin/admin) you can check that your reservation has been successfully transferred over.



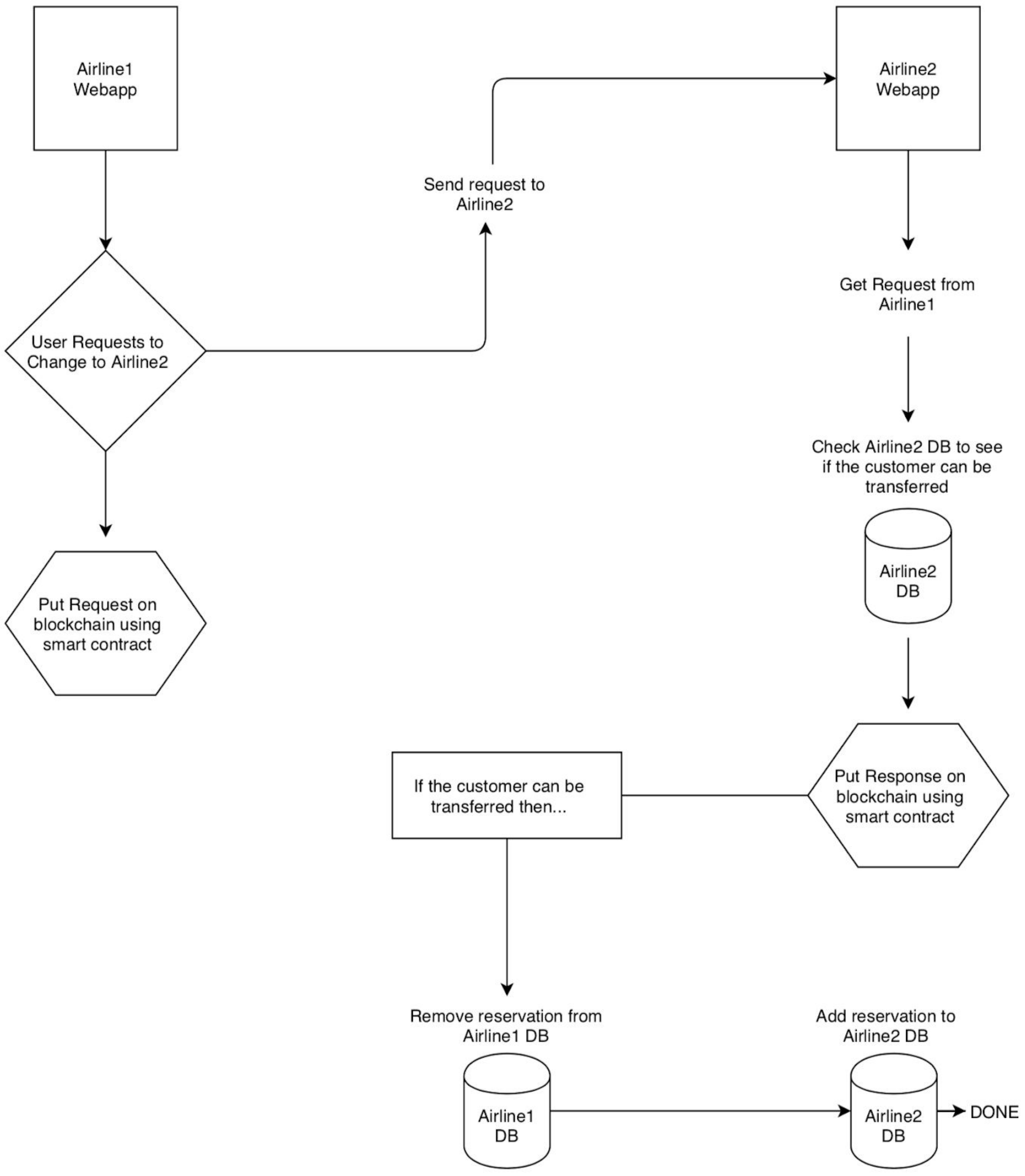
Your Upcoming Trips

Keep Track of Your Trip

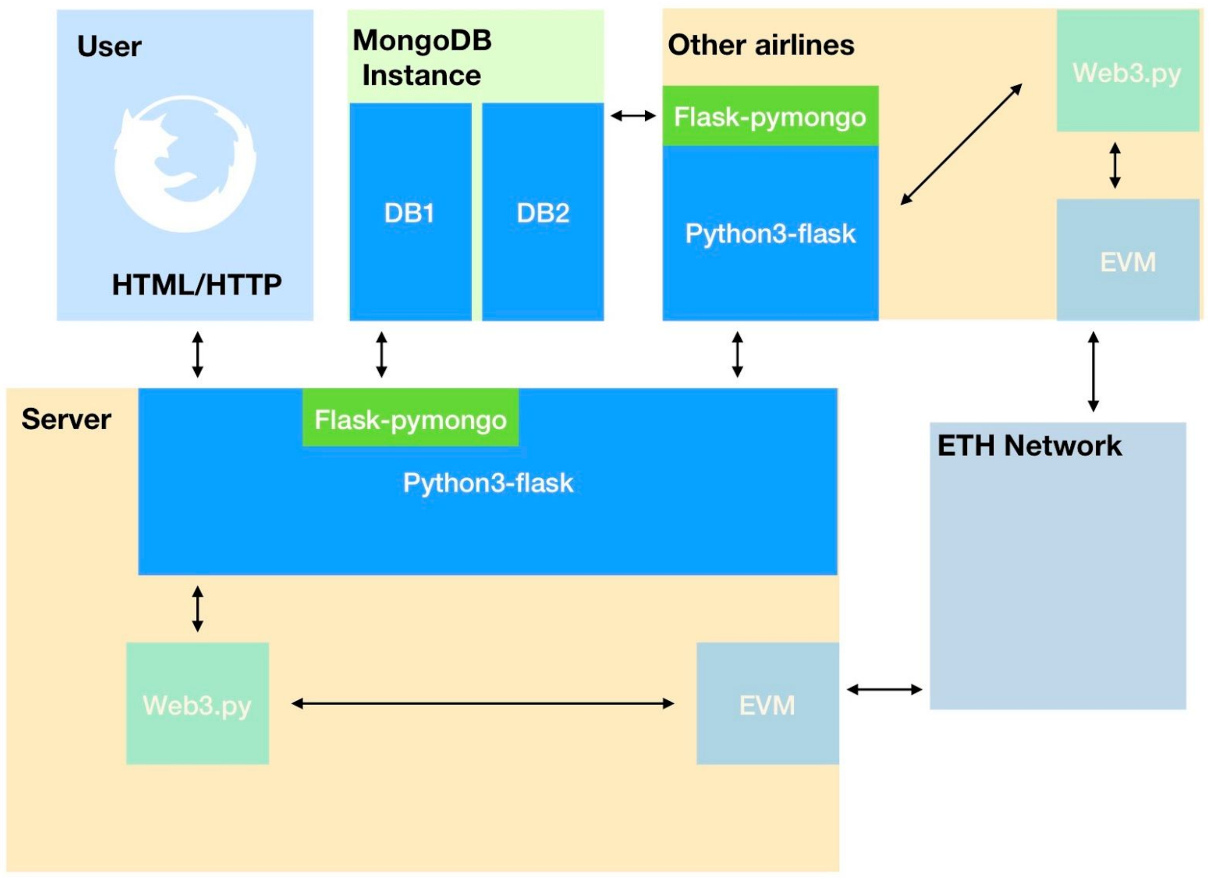
05/20/2019

Eggsville to Fartsville

Operation diagram


Airline1 Webapp
User Requests to Change to Airline2
Put Request on blockchain using smart contract
Send request to Airline2
Airline2 Webapp
Get Request from Airline1
Check Airline2 DB to see if the customer can be transferred
Airline2 DB
Put Response on blockchain using smart contract
If the customer can be transferred then...
Remove reservation from Airline1 DB
Airline1 DB
Add reservation to Airline2 DB
Airline2 DB
DONE

Application diagram



Smart Contract Overview

Contract Variables

<u>**Type**</u>	<u>**Name**</u>
mapping(address => bool)	registeredAirlines
uint	priceToRegister

Modifiers

<u>**Name**</u>	<u>**Function**</u>
contractOwner()	Only the owner of the contract can use these functions
registered()	Only registered airlines who buy-in can call these functions
costs(uint *price*)	How much a function will cost to be called

Functions

<u>**Name**</u>	<u>**Description**</u>
register()	Payable function that unregistered airlines call (and pay some ETH) to get put into the *registeredAirlines* mapping.
unregister()	The contract owner can call this function to remove a registered airline from the *registeredAirlines* list.
changeRegisterPrice(uint *newPrice*)	Changes how much it costs for an airline to call the *register* function.
request(address *toAirline*, string *details*)	Function that is called when an airline customer makes a request to transfer to another airline. Emits the *Request* event.

response(address *fromAirline*, string *details*, bool *successful*)	Function that a responding airline calls when they can take a transferred customer. Calls the *settlePayment* function which is listed below and emits the Response event.
settlePayment(address *toAirline*)	Function that emits the *Payment* event. Signifies a payment between two airlines.

Events

Name	**Description**
Request(address *fromAirline*, address *toAirline*, string *details*)	Emits what airlines are involved in the transfer request and details of the transfer onto the blockchain.
Response(address *toAirline*, address *fromAirline*, string *details*, bool *successful*)	Emits the airlines involved in the response to a request and whether the airline was able to successfully transfer the customer.
Payment(address *toAirline*, address *fromAirline*)	Emits a payment transaction between two airlines on the blockchain.
Register(address *airline*)	Emits an event when *register()* is called.

Database Overview

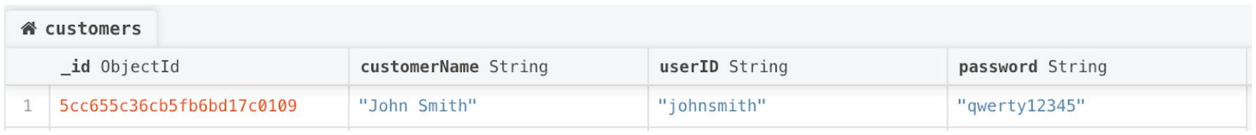
Each airline has their own unique database. Each database has two collections(tables): **customers** and **reservations**.

Customers

Field name(type)	
_id(ObjectId)	Unique ID given to each entry by MongoDB
customerName(String)	Customer's name
userID(String)	Login credential
password(String)	Login credential

When a user signs up, they have to provide **customerName(String)**, **userID(String)**, and **password(String)**. Then, a handler function will determine whether the **userID(String)** given already exists in **customers** or not. If not, then it will be added to the database.

Example



customers

	_id ObjectId	customerName String	userID String	password String
1	5cc655c36cb5fb6bd17c0109	"John Smith"	"johnsmith"	"qwerty12345"

Reservations

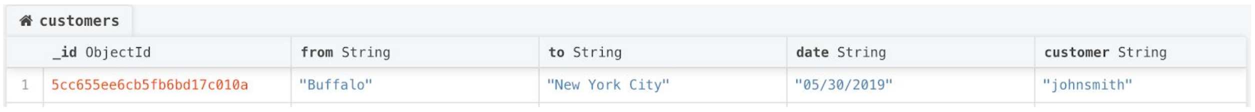
Field name(type)	
_id(ObjectId)	Unique ID given to each entry by MongoDB
from(String)	A name of a city where a flight booked departs from
to(String)	A name of a city where a flight booked heads to
date(String)	A date when a flight booked departs(it is strictly in the MM/DD/YYYY format)
customer(String)	This field saves **userID(String)** from

	customers and indicates who booked a ticket.

Because **MongoDB** is a **NoSQL DB**, we write a custom driver function that looks for a reservation or reservations each user has by comparing **customer(String)** from **reservations** to **userID(String)** from **customers**.

We don't have something like that for **date(String)**, but a function to add a reservation is written in a way so that it would save dates in the MM/DD/YYYY format only. Each area(MM, DD, or YYYY) can't go over the limit(e.g. 13 for month is not possible) for each, as they are limited by HTML.

Example



customers

	_id ObjectId	from String	to String	date String	customer String
1	5cc655ee6cb5fb6bd17c010a	"Buffalo"	"New York City"	"05/30/2019"	"johnsmith"

Warning

If you are a Mac user who is running *Cisco Anyconnect*, it is possible that a component of it, called *Cisco Anyconnect Web Security*, is locking TCP 5001 which is needed to run this program. If that’s the case, you need to uninstall it first. Check **https://blog.felipe-alfaro.com/2014/02/10/cisco-anyconnect-web-security-module-acwebsecagent-in-mac-os-x/**