Ethylocl\u00e9 backend web server using Express framework

I used CoffeeScript as programmation language and MarkDown to document our code.

Functionalities

- Manage account profile: sign in, sign up, update and delete
- Manage trip: create trip, get trips, join trip, get trip data
- Import users data from csv and json files
- Export users data to csv file

User request

Sign in

If the email and password are correct, a session cookie is returned to the client

```
url: 195.154.9.74:3000/usr/signin
Paramètres: {email, password}
Retour: { "result": bool, "data": null }
```

Check password

Allows client to check password before allowing client user to reset critical information

```
url: 195.154.9.74:3000/usr/checkpassword
Paramètres: {password}
Retour: { "result": bool, "data": null }
```

Sign up

Client user can sign up with an email and password if the email is not already used

```
url: 195.154.9.74:3000/usr/signup
Paramètres: {email, password}
Retour: { "result": bool, "data": null }
```

Sign out

Just destroying the cookie session

```
url: 195.154.9.74:3000/usr/signout
Paramètres: {}
Retour: { "result": bool, "data": null }
```

Update email

As the email is set as index in user database, it needs special treatments

```
url: 195.154.9.74:3000/usr/updateemail
Paramètres: {email}
Retour: { "result": bool, "data": null }
```

Update user data

This request allows client to update user data as password but not email and id

```
url: 195.154.9.74:3000/usr/update
Paramètres: {"image", "lastname", "firstname", "birthDate", "gender", "weight", "address", "zipCode", "city", "country", "phone", "passw
Retour: { "result": bool, "data": null}
```

Get user data

Client will get user data without password

```
url: 195.154.74:3000/usr/get
Paramètres: {}
Retour: { "result": bool, "data": userObject }
```

Get user data by Id

A user can get data of another user by providing user's id

```
url: 195.154.74:3000/usr/getbyid
Paramètres: { "id" }
Retour: { "result": bool, "data": ["id", "image", "lastname", "firstname", "birthDate", "gender", "phone"] }
```

Delete user

Delete all user data in user database if he hasn't got a trip in progress

```
url: 195.154.9.74:3000/usr/delete
Paramètres: {}
Retour: { "result": bool, "data": null }
```

Trip request

Has trip

Allows client to know if the user has a trip in progress

```
url: 195.154.9.74:3000/trp/hastrip
Paramètres: {}
Retour: { "result": bool, "data": null }
```

Get trips

Client can get all the trips available based on user's search preference

```
url: 195.154.9.74:3000/trp/gettrips
Paramètres: { "latStart", "lonStart", "latEnd", "lonEnd", "dateTime", "numberOfPeople" }
Retour: { "result": bool, "data": [ { "id", "distanceToStart", "distanceToEnd", "dateTime", "numberOfPassenger", "maxPrice" }, ... ] }
```

Join trip

User can join an existing trip by providing its id and the numberOfPeople. If the trip is no longer available, the user will be notified in this way.

```
url: 195.154.9.74:3000/trp/jointrip
Paramètres: { "id", "numberOfPeople" }
Retour: { "result": bool, "data": null }
```

Create trip

If the user doesn't want to join an existing trip, he can create his own one by providing the essential data requested

```
url: 195.154.9.74:3000/trp/createtrip
Paramètres: { "addressStart", "latStart", "lonStart", "addressEnd", "latEnd", "lonEnd", "dateTime", "numberOfPeople" }
Retour: { "result": bool, "data": null }
```

Get trip data

Once a user has joined or created a trip, he can access all the trip data such as passengers id

```
url: 195.154.9.74:3000/trp/gettripdata
Paramètres: {}
Retour: { "result": bool, "data": { "id", "addressStart", "latStart", "lonStart", "addressEnd", "latEnd", "lonEnd", "dateTime", "maxPric
```

Get trip data by id

All users can access a part of the trip data by providing trip id

```
url: 195.154.9.74:3000/trp/gettripdatabyid
Paramètres: { "id" }
Retour: { "result": bool, "data": { "id", "addressStart", "latStart", "lonStart", "addressEnd", "latEnd", "lonEnd", "dateTime", "maxPric
```

LevelDB schema

```
User namespace key: "users:#{id}:#{property}"
Properties: "email", "image", "lastname", "firstname", "birthDate", "gender", "weight", "address", "zipCode", "city", "country", "phone'
birthDate format: 'DD-MM-YYYY'
User namespace index: "usersEmailIndex:#{email}:#{property}"
property: "id"

Trip namespace key: "trips:#{id}:#{property}"
Properties: "addressStart", "latStart", "lonStart", "addressEnd", "latEnd", "lonEnd", "dateTime", "price", "numberOfPassenger", "passeng
dateTime format: 'DD-MM-YYYY H:mm'
Trip namespace index: "tripsPassengerIndex:#{userId}:#{id}:#{property}"
property: "dateTime"

Tripsearch namespace key: "tripsearch:#{userId}:#{distance}:#{tripId}"

Stop namespace key: "stops:#{id}:#{property}"
Properties: "name", "desc", "lat", "lon", "lineType" and "lineName"
Stop namespace index: "stops:#{lineType}:#{id}"
```

Install

Use this command to install locally all the dependencies needed:

```
npm install
```

Use this command to install globally forever module:

```
npm install forever -g
```

Test

Several tests are provided, execute them using the following command:

```
npm test
```

You can test the sign in request from a client using the following command on windows:

```
curl -H "Content-Type: application/json" -X POST http://195.154.9.74:3000/usr/signin -d "{\"email\":\"dorian@ethylocle.com\", \"passworc
```

You can test the sign up request from a client using the following command on windows:

```
curl -H "Content-Type: application/json" -X POST http://195.154.9.74:3000/usr/signup -d "{\"email\":\"dorian@ethylocle.com\", \"passworc
```

Launch server

Execute the following command for launching server:

```
npm start
```

If you want to launch server in production mode, run the following command:

```
forever start ./bin/start.js
```

Manage database

Several scripts are provided for managing database

Show data

```
node ./bin/show user
node ./bin/show trip
```

Just pipe it with grep command to look at a particular entry !

```
node ./bin/show user | grep id:8
```

Delete data

```
node ./bin/delete --type user 0
node ./bin/delete --type trip 0
```

Import data

```
node ./bin/import --format csv --type users "user sample.csv"
node ./bin/import --format json --type users "user sample.json"
node ./bin/import --format csv --type stops "ratp_stops_with_routes.csv"
```

Export data

```
node ./bin/export --format csv users.csv
```