API Endpoints

All the responses from the API will be in this format:

```
{
    "message": result,
    "data": data
}
```

Where `result` is either `"success"` or `"failure"` depending on the result of the operation. If it failed, there will be no data.
So a response from a successful request might look like this:

```
{
    "message": "success",
    "data": { "user_id": 0 }

}
```

Whereas the response from an unsuccessful request will look like this:

```
{
    "message": "failure"
}
```

Aside from the reasons stated for each endpoint, most requests will also fail if the wrong type of data is sent, or if some data is not sent at all.

/newaccount

Takes user information and creates a new account for them, fails if an account already exists with that username.

Input

```
{
    "name": name,
    "username": username,
    "email": email,
    "password": password
}
```

Output

```
{
    "message": "success",
    "data": {
        "user_id": id
    }
}
```

/login

Checks the username and password against the database and returns the user id. Fails if the username or password is incorrect, or the account does not exist.

Input

```
{
    "username": username,
    "password": password
}
```

Output

```
{
    "message": "success",
    "data": {
        "user_id": id
    }
}
```

/profile

Gets the profile information about a user using their id. The business id is the id of the business the user is associated with, null if they are not associated with a business. Fails if the user id is not in the database.

Input

```
{
    "user_id": user_id
}
```

Output

```
{
    "message": "success",
    "data": {
        "name": name,
        "username": username,
        "email": email,
        "business_id": business_id
    }
}
```

/linkbusiness

Links a user to a business by setting their business id. This means that they are now the owner of this business (you can set multiple people to be the owner of one business this way). Fails if the user id is not in the database.

Input

```
{
    "user_id": user_id,
    "business_id": business_id
}
```

Output

```
{
    "message": "success"
}
```

/newbusiness

Creates a new business and puts it in the database, returns the id of the new business. Fails if t

Input

```
{
    "name": name,
    "email": email,
    "address": address,
    "postcode": postcode,
    "description": description
}
```

Output

```
{
    "message": "success",
    "data": {
        "business_id": id
    }
}
```

/businessprofile

Gets the business profile information using the business id. Fails if the business id is not in the database.

Input

```
{
    "business_id": business_id
}
```

Output

```
{
    "message": "success",
    "data": {
        "name": name,
        "email": email,
        "address": address,
        "postcode": postcode,
        "description": description
    }
}
```

/addreview

Adds a new review into the database. For the values in scores, the true/false ones should be represented by 0 or 1 for false and true respectively, the others should just be the rating.

Input

```
{
    "business_id": business_id,
    "user_id": user_id,
    "text": text,
    "scores": {
        "oneway": oneway,
        "sanitizer": sanitizer,
        "mask_usage": mask_usage,
        "bouncers": bouncers,
        "temperature_checking": temperature_checking,
        "staff_ppe": staff_ppe,
        "social_distancing": social_distancing,
        "ventilation": ventilation
    }
}
```

Output

```
{
    "message": "success",
    "data": {
        "review_id": id
    }
}
```

/getreviews

Gets reviews filtered either by user id or business id. In the request object, only set the value of the one you want to filter by and leave the other one null. If both are set, then it filters by user.

Input

```
{
    "user_id": user_id,
    "business_id": null
}
```

OR

```
{
    "user_id": null,
    "business_id": business_id
}
```

Output

```
{
    "message": "success",
    "data": reviews
}
```

Where `reviews` is an array of review objects.

Review object:

```
{
    "review_id": review_id,
    "business_id": business_id,
    "user_id": user_id,
    "text": text,
    "oneway": oneway,
    "sanitizer": sanitizer,
    "mask_usage": mask_usage,
    "bouncers": bouncers,
    "temperature_checking": temperature_checking,
    "staff_ppe": staff_ppe,
    "social_distancing": social_distancing,
    "ventilation": ventilation
}
```

/namesearch

Searches through the names of all the businesses we have in our database, and returns any which have the search query in them as a subword. The input query should be a string

Input

```
{
    "query": query
}
```

Output

```
{
    "message": "success",
    "data": businesses
}
```

Where `businesses` is an array of business objects.

Business object:

```
{
    "business_id": businessid,
    "name": name,
    "email": email,
    "address": address,
    "postcode": postcode,
    "description": description
}
```

/averagereview

This takes a business id and returns the average of all the reviews it has gotten. This only averages the numerical fields (i.e. everything except the text field). Since the true/false options are represented as 1 or 0, they are also averaged, this means that the average for those ones is the percentage of people who chose true for that option.

Input

```
{
    "business_id": business_id
}
```

Output

```
{
    "message": "success",
    "data": {
        "oneway": oneway,
        "sanitizer": sanitizer,
        "mask_usage": mask_usage,
        "bouncers": bouncers,
        "temperature_checking": temperature_checking,
        "staff_ppe": staff_ppe,
        "social_distancing": social_distancing,
        "ventilation": ventilation
    }
}
```