Technical Exercise

# Summary

**This exercise is deliberately open ended, you can spend as little or as long on the exercise as you want. There is no obligation to spend more than an evening on this exercise but we encourage you to take as much time as you feel you need.**

We’d like you to build a mini API that could power a very simple dating app.

The functionality is split into 3 parts and an optional bonus. Each part will involve writing some PHP and building a small MySQL database.

You don’t need to build an interface.

Your API endpoints should be available locally. E.g <http://localhost/user/create>

If you have any questions, please do not hesitate to ask.

# Tools to use

API Backend:

PHP (7.x) (Consider using Slim - avoid Laravel)

JSON for request and response payloads.

Database:

MySQL/MariaDB

Source Control:

Git

*Optional - Make use of Docker*

# What to send us

Please **email** us the following:

1. **README.md**
   1. Tell us how to setup & run your API.
   2. Include details that set you apart. Feel free to show off.
2. **Solution\_Your\_Name.zip**
   1. A ZIP folder containing your solution (code, schema etc).
   2. Be sure to include the .git repository.

## Part 1 - The Basics

i) Write an endpoint to create a random user at **/user/create**

It should generate and store a new user.

It should return these fields: *id, email, password, name, gender, age.*

ii) Write an endpoint to fetch profiles of potential matches at **/profiles**

You should specify your user id.

It should return other profiles that are potential matches for this user.

iii) Write an endpoint to respond to a profile at **/swipe**

You should specify your user id + a profile id + preference (YES or NO).

It should store and return if there was a match (both users swipe YES).

iv) Extend **/profiles** to exclude profiles you have swiped.

## Part 2 - Authentication

i) Write an endpoint to authenticate a user at **/login**

You should specify email + password.

It should return a token if successful.

(Please write your own logic - don’t just use a framework)

ii) Extend **/profiles** and **/swipe** to be authenticated by a login token.

## Part 3 - Filtering

i) Extend **/profiles** to filter results by age and or gender.

ii) Extend **/profiles** to sort profiles by distance from the authenticated user.

You will need to add location to the user model.

iii) Extend **/profiles** to sort profiles by attractiveness.

You will need to come up with a ranking based on swipe statistics.

## \*\* Bonus \*\* - Profile Photos

i) Write an endpoint **/user/gallery** to add photos to your profile.

Specify a login token + photo(s).

Format and save the uploads as 100px \* 100px JPEG files.

Update the **/profiles** endpoint to include the users photos.