Notes on Phone Exercise

# Design Notes

Some thoughts on the design decisions, in no particular order:

* Names are not unique, you can have multiple users with the same name. So the interface should rely on userId rather than name. You could add a find-by-name feature if that was important.
* Emails could be unique. But we are not enforcing that yet. I think it is just an annotation on the class, and then all the error handling.
* Phone numbers should be unique on phones, and phones should have one phone number. Though some phones can have multiple sims, and those sims can move. We could model sims in a full implementation
* Preferred phone number might not be one of your phones. It might be your landline
* Don’t return passwords, this is a security issue. I have just returned the Entity to the user. There are suggestions that there should be another layer of model that is the objects that are returned, to increase separation. I haven’t implemented this.
* I am leaving out authentication for now. That is something that will need to be configured. It’s a solved problem I will just need to follow the steps
* Phones could move between users. Though that is an interesting question, do you want to move the phone, and keep any history that phone might have, or just add it as a new phone, which might be cleaner for privacy reasons.
* Update a user. One way is to Put a JSON User object. But this means the update function needs to change everything all the time. This can lead to accidentally changing things you didn’t meant to. I have gone with using request parameters to allow for changing just the setting/s that are needed.

# Outstanding work

I have not added any security layer. Adding Basic authentication is a solved problem with Spring Boot, but I have not yet gone in and learned how. I don’t think that following that adds a lot of value as an exercise.

# Packaging

This builds with maven clean package.

Runs as java -jar target\callers-0.0.1-SNAPSHOT.jar

# API

GET /User get Users

POST /User add User

GET /User/{userId} get specific User

DELETE /User/{userId} deletes a specific user

PUT /User/{userId} updates a specific user with Request Parameters preferredPhoneNumber, name, email

GET /User/{userId}/Phone gets a users Phones

POST /User/{userId}/Phone adds a new Phone to a user

DELETE /User/{userId}/Phone/{phoneId} deletes a specific Phone from a user

# Code coverage:

