Instructions

The purpose of this assignment is to see how would you develop code. Do it like it was to be on production. Follow your own coding conventions but make it coherent through all the code you develop. **Write clean, maintainable, quality code using the all components you need**.

Assignment

We have a mobile app and we keep track of all the devices our users use. We do so because we can offer some advantages to those who use it in many devices. To model this we consider devices can be: Active, Inactive or Lost. **This is our first approach for day one, but maybe we add some more states in the future**.

When a user uses a device for the first time, it's a new device which is considered Active. If a device has not been used for more than a week, it becomes Inactive. If it's not used for another week, then it becomes Lost. If an Inactive or Lost devices is used again, it becomes Active again too.

When a device becomes either Inactive or Lost, an email is sent to the user so he becomes aware of that fact. An email is also sent to the user when a new device is registered for the first time.

We have a service (Device Tracking) where we only store the list of devices and its last activity. The advantages for the users are managed in another service (User Devices), so there it's where we've modeled this workflow. This service consumes the information from the Device Tracking service periodically, to update each device's state.

We've got pretty much all the code developed but we lack one part: the workflow management.

Our process retrieves the current data from the Device Tracking service obtaining a list of devices with an identifier and the last activity timestamp. Then we iterate that list and, try to match each device to an existing one in the User Devices service and updating it accordingly.

This is a snippet of the User Devices service code:

```
...
public void updateDevices() {
      for (User user : getUsers()) {
            for (TrackedDevice trackedDevice :
            trackingManager.getDevices(user)) {

                  Device device = getDevice(user, trackedDevice.getId());
```

```
                  Device modifiedDevice = deviceWorkflowManager.manage(device,
            trackedDevice):

                  store(modifiedDevice);
            }
      }
}
...
```

We just need you to code a class implementing the `DeviceWorkflowManager` interface to manage the device's states:

```
public interface DeviceWorkflowManager {
      Device manage(Device currentDevice, TrackedDevice trackedDevice);
}
```

A `Device` will have, at least, the user identifier, the device identifier and the status. A `TrackedDevice` will have, at least, the user identifier, the device identifier and the last activity timestamp.

Provide the code for the implementation of the `DeviceWorkflowManager` and all the other components you may consider necessary. **All the code must compile**, so you'll need, at least, to also provide the implementation for the device related classes (Device and TrackedDevice).