MOBILE TEST APP SPECIFICATION

The task

Develop a native app for Android or iOS that:

- allows the user to sign in to a web service using their login and password
- processes the response containing Base64-encoded picture
- shows the picture in the UI

The app should contain:

- screen with input fields for login and password
- button to start the download
- place for displaying the image

Submit:

- Android: source code + APK file
- iOS: source code + IPA file (if you have access to Apple dev account or .xcarchive)

The name of the app (displayed in the launcher) should be your surname.

Service description

- Username & password are in small caps (example: `user`).
- You'll receive username & password from the recruiter.
- The service is available at https://mobility.cleverlance.com/download/bootcamp/image.php
- Username is sent in the body of the message as POST parameter with key `username`.
- Password must be hashed with SHA-1 and sent in a header field named `authorization`.
- The response contains Base64-encoded picture.

Request example:

```
POST /download/bootcamp/image.php HTTP/1.1
Authorization: 27d941d0a9d7be4441961c164847186841da68c6
Content-Type: application/x-www-form-urlencoded
Host: mobility.cleverlance.com
-----------------------------------------------------
username=user
```

Technical requirements

Use of any open source third-party library is allowed except for image-loading libraries like Picasso, Glide or Coil on Android or SDWebImage, AlamofireImage or Kingfisher on iOS. If in doubt, ask us.
We don't accept Android applications written in Java or iOS applications written in Objective-C.
Please do not make source code of your solution, binary files or this document publicly available (e.g., do not host your source code in public GitHub repository).

What are we looking for in your solution?

We want you to demonstrate the ability to deliver:

Internal quality

Although the total size of the project will be only a few hundred LOC, we would love to see:

- consistent formatting of the source code
- clean, robust code without smells and no cyclic dependencies
- consistent abstractions and logical overall structure
- code organized in meaningful layers
- low coupling and high cohesion
- descriptive and intention-revealing names of packages, classes, methods etc.
- small functions that do one thing
- truly object-oriented design with proper encapsulation, sticking to DRY and SOLID principles, without procedural anti-patterns
- usage of modern APIs, libraries and techniques
- use of techniques like design patterns, dependency injection, correct asynchronous work, design by contract and especially unit (or even functional or integration) tests

External quality

How the app behaves on the outside:

- the app should be fully functional, with every state, user input, boundary condition etc. taken care of (although the app is indeed very small, treat it as a part of big production-ready project)
- the app should correctly handle screen orientation changes, device resources and permissions, incoming calls, network connection issues, being pushed to the background and other platform intricacies and should recover from these gracefully
- lowest API level is not defined–use what you think is reasonable these days
- bonus points if the app interacts with the user in an informative and helpful way
- bonus points for nice looks–use a clean, simple yet effective layout and design

We are looking forward to your solution, good luck!