A teamwork management application

Lab2 – Show and edit user profile

Learning objectives

- Designing early UX layers
- Using JepackCompose to build interactive screens
- Using CameraX to take photo
- Starting new Activity to access gallery

Description

From the Lab1 description:
"A teamwork management app is designed to facilitate collaboration, communication, and coordination among team members to enhance users' overall productivity and efficiency.
By using this app, a user may manage her/his participation in one or more teams, get and set information about tasks to be performed by team members, document her/his own progress and achievements, report contributed efforts, and gather feedback from other team members as well as analytic information derived from collected data.

The app will support the following features:
- User Profiles
  - Personal profile management
- Feedback and Performance:
  - Performance metrics and analytics for individual team members
  - Visual representations of team achievements"

During this Lab you have to build an important part of the user experience of the app: the display of the personal profile.
A user's page must display the useful information (starting with the information you chose in Lab1) that you intend to collect. A non-exhaustive list includes: first name, last name, username, profile picture, description, skills, achievements, or personal stats.

Some of this information once added cannot be changed, others can be edited via an edit page while remaining required. Still others may be optional.
Some of this information (e.g. *email* or *phone number*) will need to be checked for syntax at the input form level.
It is therefore necessary to design a **display** screen and an **edit** screen for the information.

The user can choose their profile image from a monogram based on their first and last name, an image from the gallery, or a photo they have just taken.

The statistics shown in this page are different from those of a user within a team.
A person's profile may show relevant KPIs such as the number of teams he or she is a member of, the total number of tasks completed in the various teams, or a system of anonymous reviews and/or ratings of his or her performance in the various teams. This information is useful when assigning a task to a person or when deciding whether to invite them to join a team.

Steps

1. To accept the assignment, use GitHub Classroom as you did in Lab1: https://classroom.github.com/a/sbLBF4Yv
2. Create a new project on Android Studio
    a. *Phone and Tablet → Empty Activity*
3. Create and customize the UserProfileScreen component. Start with the User Information pane component
    a. The purpose of this screen is to show a basic set of data about a user. This should comprise, at least, the following items:
        i. Photo (possibly a default image stored as a drawable resource)
        ii. Full Name (your real name)
        iii. Nickname (your chosen public identity)
        iv. E-mail address
        v. Location of the user (a string for now)
        vi. A short description
        vii. KPI
        viii. Any other extra data items relevant to the application that your

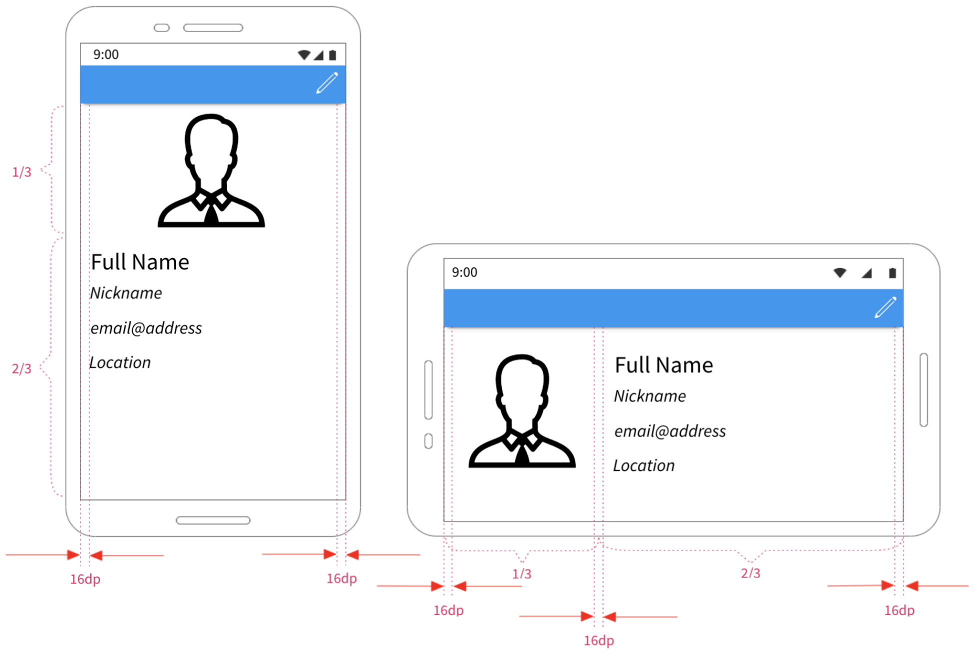
group is conceiving

b. If the user hasn't already chosen a photo, you have to show a monogram with the initials of his first and last name.

c. Each piece of information should be maintained as a property of the view model.

   i. You can set default values in your view model object to test UI.

d. Modify the basic layout as sketched in the following picture



e. The app must be responsive to *different screen sizes* and different *orientations*: check that everything works as expected by choosing different device presets or setting different *Previews*, both in landscape and in portrait mode

f. Commit project. Push it on the remote repository

4. Create an overflow menu in the *TopAppBar* component, with a single item having a pencil icon.

5. Create a second pane component (EditProfile pane) that allows the user to edit his personal data

   a. The new layout will resemble the previous one, with the exception that all Text components will now be editable.

      i. Make sure that the components are filled with the previous values and that the **validation** is performed on saving.

Display useful information to the user if an error occurs during validation.

   ii. Use the background information contained in the document: https://developer.android.com/develop/ui/compose/text/user-input

   iii. The input fields must implement a smart keyboard. You need to customize the displayed keys to the data item to be entered (generic text, e-mail address, numbers, …). Use *KeyboardOptions* for it

b. The ImageComponent will show a *Button* on top of it, labeled with a camera icon

c. If the user clicks on the Button a floating context menu appears, showing the following options:

   i. Select an image from the phone gallery

   ii. Use the camera to take a picture

      1. At this point, simply *log* the user tap for information or display a *toast* for debugging purposes.

d. Use the MVVM architecture to store data and handle their graphical representation, sharing the same model among the screens and edit modes.

e. If the user rotates the device, data entered so long should not be lost

f. If the user clicks the back button while in edit mode, changes must be persisted and navigation to the appropriate screen component will occur

   i. Even in this case, validation of the fields must be ensured.

g. Commit project. Push it to the remote repository.

6. Update the *Edit Profile pane* to allow the user to select a different image. Users can select a profile picture from the gallery or take a new picture.

   a. First option: “Use the camera to take a picture”.
   Carefully read the document
   https://developer.android.com/codelabs/camerax-getting-started
   ~~https://developer.android.com/media/camera/camera-intents~~, and use the contained guidelines to allow the user to launch the default camera application in order to snap a new picture and to return it to the

application

    i. Use the returned bitmap to update the picture shown.

b. Second option: “Select an image from the phone gallery”. Launch a new activity to allow the user to select an image from the gallery. Use coil to display it using its URI.

c. Commit project. Push it on the remote repository

Submission rules

- The work must be submitted by April, 29 23:59
- The design and the code of the user interface will be evaluated.
- The last commit before the deadline will be evaluated. Alternatively, create a release and label it “completed”.