GitHub Username: lilggamegenius

Person Generator

Description

This app, Written solely in the java programming language, will enable developers to create fake person info easier and enable them to work on their projects without needing to spend lots of time thinking of fake names.

Intended User

This is for developers who need to generate info for a random person.

Features

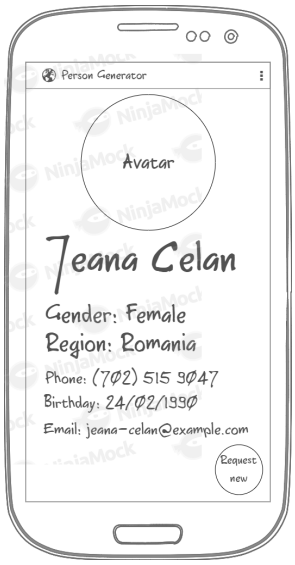
Generate fake info with:

- Age
- Region
- Gender

User Interface Mocks

These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1

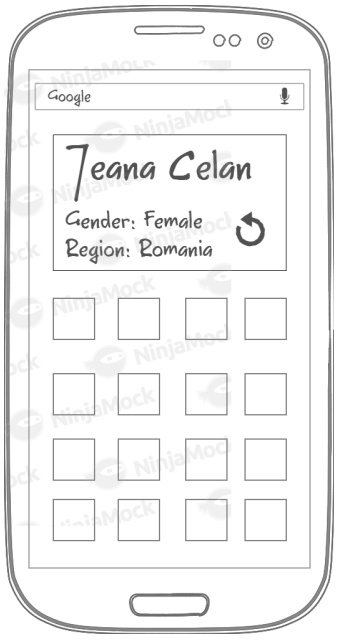


The main results screen that gives the result

Screen 2

Settings screen to allow filtering results

Screen 3



Widget for generating info on the home screen

Add as many screens as you need to portray your app's UI flow.

Key Considerations

How will your app handle data persistence?

The app will handle data persistence using Firebase Realtime Database

Describe any edge or corner cases in the UX.

If the user presses the back button while the Drawer menu is open, the drawer will close and generate a new person. If there is no internet connection, the application will show a toast notification or snackbar. If the api fails to respond, the application will respond by telling the user that it received faulty data and show an event in firebase

Describe any libraries you'll be using and share your reasoning for including them.

Volley: 1.1.1	Allow easily calling and managing API use
Picasso: 2.5.2	Easily load and manage downloading and using online images
Jackson: 2.9.7	For use in deserializing the json response from API

Describe how you will implement Google Play Services or other external services.

FireBase Analytics: Collect info on how often the app is used and what settings are used most often
FireBase Crashlytics: Collect crash info to help diagnose issues

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup

Create and test classes that will connect to the uinames API. Implement AsyncTask to fetch data. Add all UI strings to strings.xml.

Task 2: Implement UI for Each Activity and Fragment

List the subtasks. For example:

- Build UI for MainActivity
- Build UI for Settings
- Build UI for bulk generation
- Update UI to have Content Descriptions for accessibility
- Enable RTL layout switching on all layouts

Task 3: Handle Errors

Setup checks to prevent network errors, and handle API errors

Task 4: Create Widget

Create UI for widget and add functionality

Add as many tasks as you need to complete your app.

Submission Instructions

- After you've completed all the sections, download this document as a PDF [File → Download as PDF]
    - Make sure the PDF is named "**Capstone_Stage1.pdf**"
- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
- Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"