BUILD A SIMPLE CONTACTS APP (WITH OTP SMS SENDING FUNCTIONALITY)

Overview

For this project, you will use a web / Android app that can send an OTP (via SMS) to a list of contacts, one at a time.

NOTE: Kindly read the specs as "tab" for Android and a "menu" for web. Similarly, there will be other terms that are borrowed from Android (like "activity" aka "page") and if you're doing web, read it as the equivalent for that.

Requirements

1. The main activity should have two tabs (swipe to change tab ANDROID ONLY)
2. First Tab
    a) List of contacts
        - Inputs: JSON list of contacts (you can create your own fake contacts - you can make the fake data by using a static JSON)
        - Outputs: Name of the Contact (First + Last) should show up in each row of the list.
        - Upon selecting a contact in the list, the contact info page for that contact will be displayed (in a new activity / page).
    b) Contact info page - Contact Details with button.
        - Display contact data: name, phone number.
        - Button saying "Send Message" which on click opens next screen.
    c) New Message Screen (Compose).
        - Components: Text Field, Send button.
        - Text Field needs to have text along the following lines: "Hi. Your OTP is: 123456". 123456 needs to be a random six-digit number.
        - Upon hitting send, the contents of the text field will be sent to the selected recipient as an SMS using online services (and not the phone itself natively for Android). For this you can a free trial of any of the services like Twilio, Nexmo etc. If you do use these, then add the number +919810153260 to the accounts that can receive an SMS.
3. Second Tab.
    a) List of messages sent
        - In descending order of date-time
        - Inputs: list of messages already sent (you can choose the best way to do this)
        - Outputs: Each list item will be the Name of the contact, the time of the SMS and the OTP sent in the SMS.
4. User experience
    a) You should Handle Variable Screen widths and resolutions (ANDROID) and be responsive (WEB).
    b) You should try to handle Errors from the server (like the SMS one you choose to use).
    c) You should consider client side input validation.
    d) You should focus on general front end design and make it as professional as

possible. It should work, be easy to navigate and use and free of clutter.

Improvements

Feel Free to add and suggest improvements to the application.

Advice

You are free to use whichever libraries or example/sample code from the web that you see fit (but just reference the source).

Evaluation

The ideal candidate will be able to demonstrate prowess in the following areas

- Functionality
  - Does the app work?
  - Does it handle error conditions?
- Front End Design
  - Is the product easy to navigate?
  - Is the design clean and free of clutter?
  - Does the application look and feel professional?
- Code Design
  - Do you understand basic engineering principles and patterns that allow a code base to grow over time without incurring a large amount of technical debt?
- Code Cleanliness
  - Can anyone jump in and understand the code you have written?
  - Think about how to make the code reusable and run efficiently
  - Comment frequently!
- Testability
  - How easy is it to unit test important parts of the application?
- Mobile Domain (FOR ANDROID ONLY)
  - Did you capitalize on tools and technology already provided by the Android Platform?
- Misc.
  - Did you make use of other helpful 3rd party tools?
  - Did you take into account the limitations of the technologies and platform the application would run on?

Deliverables

1. Full source code (.zip format)
2. APK file (FOR ANDROID)
3. A live website (FOR WEB) for testing purposes
4. A simple doc to describe your product, libs used / any good practices, decisions made (This doc is important for us to understand your code base)