CAPSTONE STAGE 1

GitHub Username: Elorri

Tie Us

Description

The app change your “dead” contact list into an alive one by classifying your contacts iinto lists with priorities. It basically answer one question “Who should I contact today ?” and make sure all the contacts you decide to follow will see you or hear from you before they turn unsatisfied. To achieve this, the app encourages you to always think and plan the next appropriate action to take.
Aside from that, the app encourages you to evaluate each interaction you had and associate the contact with a satisfaction face. This is then used to evaluate all your relationships and resume them under one single general info : the forecast (sunny, slightly cloudy, cloudy, rainy).
So, it basically answer another simple question “How are my relationships going ?”

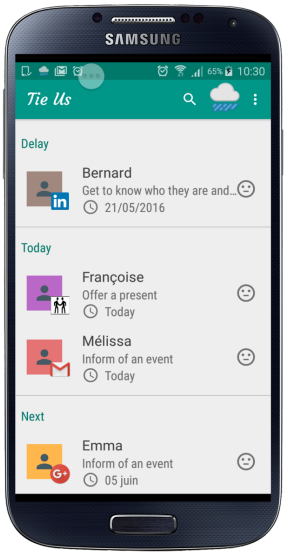
Intended User

People who want to maintain existing relationships with friends, family, or business contacts.
People who want to increase their network and keep track of each action done.

Features

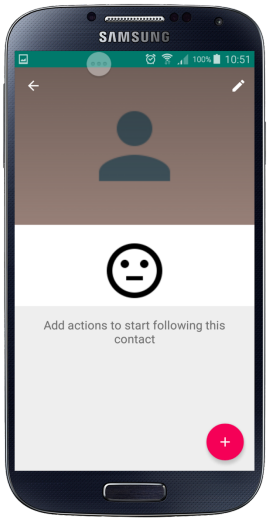
- Show list of contacts ordered with priorities
    - People that should have already be contacted
    - People to contact today
    - People to contact in the future
- Saves the degree of satisfaction of each contact
- Plan future actions
- Allows choosing most appropriate social network
- Displays reminders messages
- Display guidance messages

User Interface Mocks



Main screen

So whenever you feel like networking, open this app, and see what actions you have to perform today.
For instance today, I have to offer a present to Françoise at a local event and I have to go on gmail to inform Mélissa of an event that might interest her.

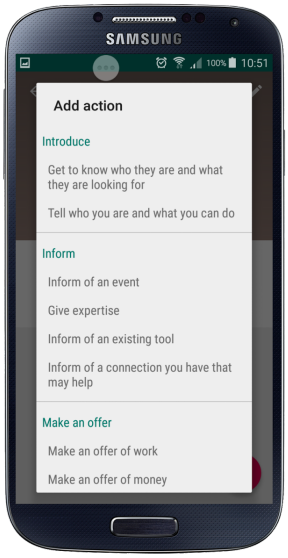


Detail screen

Detail screen is composed of 3 parts and a floating action button.

- The contact picture or default avatar picture (when no picture found) with communication with user default contact management app to allow editing.
- The contact satisfaction represented by a face and that the app encourage to update after each new interaction.
- The contacts next actions planned that gives us a quick overview of the places we have to go to interact with the contact.

- A floating action button that will open a serie of 3 screens and end up adding a new action on the contact profile.



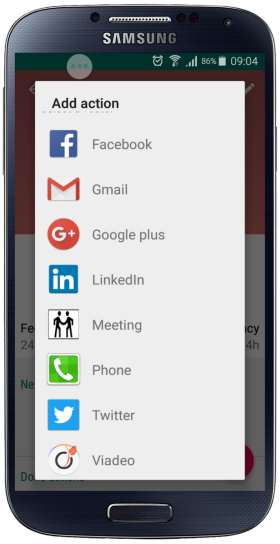
Select action

The list of actions is defined in the app and can't be customised.
It is organise in 4 parts :

- Introduce
- Inform
- Make an offer
- Give feedback

Each of these parts contains a list of relevant actions like :

- Get to know what the contact is looking for (in Introduce)
- Give your expertise (in Inform)
- Make an offer of work (in Make an offer)
- Thanks (in Give feedback)

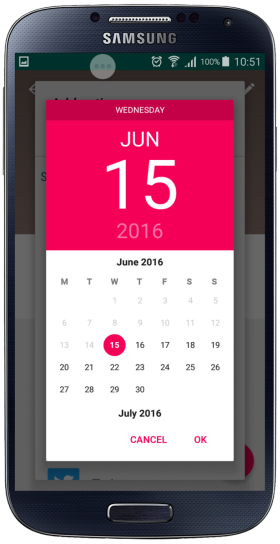


Select vector of communication

Vectors of communications are :

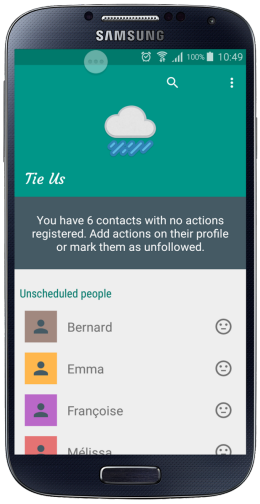
- Social networks installed on the user phone.
- Small text messages
- Phone
- Real life meetings

Once selected they appear both in Detail page and Main page, and gives the user a quick overview of "What are the people I need to contact on facebook today ?" for example.



Select date

Last step that needs to be done to add action is to fill the “due date”.
This step achieved, the app will be able sort all contacts actions and display them in relevant order on the Main page.



Guidance messages

The app is also made up with a list of messages suggesting the user to take action. They shows up on Main screen when appropriate.
Here is a few examples :

- Make sure all your contacts are scheduled:”You have 6 contacts with no actions registered. Add actions on their profile or mark them as unfollowed.”
- Reminder to update your contact satisfaction face : “Are those 3 contacts satisfied by your actions ?”
- Reminder to contact people before they turn unsatisfied : “You have 3 contacts that may turn unsatisfied if you don't interact with them. Take action, it's still time !”. Note : After a delay (you choose) satisfied people will become neutral or unsatisfied automatically if no action taken.

Key Considerations

Data persistence

The app will connect to the android contact ContentProvider existing on the user phone, and store contact_id and relevant data in a dedicated app ContentProvider call TieUsContentProvider. This provider will store additional infos like planned actions and contact satisfaction.
A synchronisation will happen every 3 days via a SyncAdapter to avoid battery damage.
If users added a new contact and don't want to wait the next synchronisation to happen, they can perform a manual synchro using the top left action menu.

UX Description

There will be 2 screens on phones and portrait tablets. And only one screen on tablets used in landscape orientation.

For phone and tablet used in portrait mode :

- Going from the Main screen to the Detail screen is achieved by clicking on a contact.
- Going from the Detail screen to the Main screen is achieved by pressing the back button or the up arrow.

For tablet used in landscape mode :

- Main and Detail screens are displayed side by side.

In addition to those 2 screens, there will be 1 Dialog that will display lists one after another in order to achieve the "add action" feature. The selection of an element in the Dialog list make the Dialog reload with the next relevant list and so on until the date is selected and the user is automatically brought back to the Detail screen (updated with the new action).

Libraries I'll be using.

- Glide to handle the loading and caching of images, and also to handle retrieving images from a given uri. This avoid a lot of time consuming work and and will help me focus on developing app specific features, which is more important. Glide is under is mostly under Apache 2 licence, meaning that I can use it and redistribute it as long as I include a copy of their licence.

- MaterialDateTimePicker. I've chosen this one for the beauty of its design as well as its simplicity and the fact that it was following the material design guidelines. Also it is an Apache 2 licence as well.

Planning

Step 1: Project Setup

In Android Studio

- Android Studio > File > New Project >
- Application name : TieUs
- Company Domain : com.elorri.android.tieus
- Min sdk (leave as it is) : 16
- Select Phone and Tablets > Empty Activity > Finish

On git

- Commit : initial commit

Step 2: Implement UI and Model for master/detail view for a phone

- Add tables contact, action, events
- Add a ContentProvider (that read from those tables)
- Build UI for MainActivity
- Build UI for MainFragment using MainAdapter (and connect to the ContentProvider)
- Build UI for DetailActivity
- Build UI for DetailFragment using DetailAdapter (and connect to the ContentProvider)
- Add syncAdapter to sync user contact list and fill contact table
- Add Test DelayedPeopleQuery in TestQueries.
- Show 'Delayed contacts' on top of Main screen.
- Add Test TodayPeopleQuery in TestQueries.
- Show 'Today contacts' in the middle of Main screen.
- Add Test TodayDonePeopleQuery in TestQueries.
- Show 'Today done contacts' in the middle of Main screen below 'Today contacts' .
- Add Test NextPeopleQuery in TestQueries.
- Show 'Next contacts' at the bottom of Main screen.

Step 3: Implement UI and Model for 'Add action' feature for phone

- Add floating action button in DetailFragment that will launch AddActionFragment in a Dialog with the initial/default Uri (when no actions are selected yet =step_zero)

- Add AddActionFragment and AddActionAdapter with launch with the default Uri. Make AddActionFragment reload itself with Uri step_action (when user has selected an action)
- Add select vector feature.
- Add pick a date feature. And make AddActionFragment save data and disappear (when user has selected an action, vector and date) leaving user back on Detail screen.
- Scan user phone to get social network installed and store their package_name and name in the database.
- Display the social network icons on AddActionFragment vectors item, MainFragment items and DetailFragment items.
- Add delete action feature (swipe right)
- Add complete/uncomplete action feature (long press)

Step 4: Building a Total Experience for phone

- Create a custom AvatarView for display on MainFragment and DetailFragment
- Add collapsing toolbar on MainFragment and DetailFragment
- Add a contact search box
- Add a collection widget for displaying people to contact today.
- Add a sharedElementTransition between the Main screen avatar and Detail Screen avatar.
- Add a slideLeft transition when entering the detail screen

Step 5: Handling help and guidance messages

- Add educative messages for delete action feature (swipe right)
- Add educative messages for complete/uncomplete action feature (long press)
- Add Test UnscheduledPeopleQuery in TestQueries
- Add ‘unschedule contact message’ on Main screen (when there is some unschedule contact)
- Add Test PeopleThatNeedsToFillInTimeLimitResponseQuery in TestQueries.
- Add ‘fill_in_time_limit_response_message’ on Main screen (when there is some contacts that needs we fill their answer delay).
- Add Test PeopleThatNeedSatisfactionUpdateQuery in TestQueries.
- Add ‘update_satisfaction_face_message’ on Main screen (when there are contacts that need a satisfaction update).
- Add Test PeopleThatNeedFrequencyQuery in TestQueries.
- Add ‘fill_up_frequency_message’ on Main screen (when there are contacts with a frequency of contacts not setted up yet).
- Add Test AskForFeedbackQuery in TestQueries.
- Add ‘ask_for_feedback_message’ on Main screen (when there are contacts that haven't tell us if they are satisfied).
- Add Test PeopleApprochingFrequencyQuery in TestQueries.

- Add ‘nearby_decreased_mood_message’ on Main screen (when there are contacts that will turn unsatisfied soon).
- Add Test PeopleWhoDecreasedSatisfactionQuery in TestQueries.
- Add ‘decreased_mood_message’ on Main screen (when there are contacts that turned unsatisfied).
- Add ‘unschedule contact message’ on Main screen (when there is some unschedule contact)

Step 6: Add tablet support

- Create layouts
    - layout-land/activity_main.xml
    - layout-land/fragment_detail.xml
    - layout-w600dp-port/activity_main.xml
    - layout-w600dp-port/activity_detail.xm
- Make sure MainAdapter and DetailAdapter assume their jobs on those new layouts.
- Create layouts
    - layout-w700dp-land/activity_main.xml
    - layout-w700dp-land/fragment_detail.xml
- Modify MainActivity.onContactClicked() method to handle the case where there is 2 pane.
- Add gray selector only for layout-w700dp-land by adding a style.
- Make sure recyclerview scroll to the correct position by adding a scrollToPosition method in the onLoadFinished of the MainFragment.

Step 7: Accessibility and Localization

- Add content descriptions for user interface components that do not have visible text
- Make sure any new SimpleDateFormat() is done with the Locale in 2nd parameter
- Make sure all strings used in the app are declared in the res/values/strings.xml file in the default chosen language. In our case english.
- Although, this app is not currently supporting languages with Right to Left display, in case this happen in the future we have to :
    - Add the start-end tags along with left/right tag because our app is min version 16 and that’s needed for all app min android version < 17.
    - Add android:layoutDirection="ltr" in our widget main layout.
- Add "translatable" attribute in all strings in strings.xml.
- Create a values-fr/strings.xml with only strings whose attributes are marked translatable=true and translate them.

Step 8: Google Services

- Add google AdMob
    - Add build variant for free and paid flavor
    - Create an AndroidManifest.xml for the free flavor
    - Add INTERNET and ACCESS_NETWORK_STATE permission in the free manifest.
    - add freeCompile 'com.google.android.gms:play-services-ads:9.0.2' in the app/build.gradle.
- Add google Analytics using the new Firebase platform
    - In Firebase console https://firebase.google.com/
        - Create a firebase project called TieUs with package name com.elorri.android.tieus
        - Add app free with com.elorri.android.tieus.free to the project
        - Add app paid with com.elorri.android.tieus.paid to the project
        - Download google-services.json and add it to app/ directory
        - Build project
    - In Android Studio
        - Add getFirebaseAnalytics() method in TieUs application to get a single instance of FirebaseAnalytics.
        - Log new contacts added, updated or deleted during the last synchronisation in MainActivity onCreate
        - Log ‘start add action event’ in DetailFragment when clicked fab button
        - Log ‘action selected event’ in AddActionFragment
        - Log ‘vector selected event’ in AddActionFragment