GitHub Username: RaulitoGC

Technology Store

Description

Technology use in schools is increasing every year, dominated by the introduction of tablets and netbooks to the classroom. It is not intended to replace teacher in the classroom; instead, most educational tech is designed to increase efficiency and improve the effectiveness of traditional teaching methods. In order to get these technology for you. Always wanted to shop everything related with technology in one place, at one time and free shipping as well? Well. We have got just the right app for you.

Intended User

This app is for persons who want to buy something related with technology for improve their skills in the classroom.

Features

- Java language will be using for development
- App keeps all strings in a strings.xml file and enables RTL layout switching on all layouts.

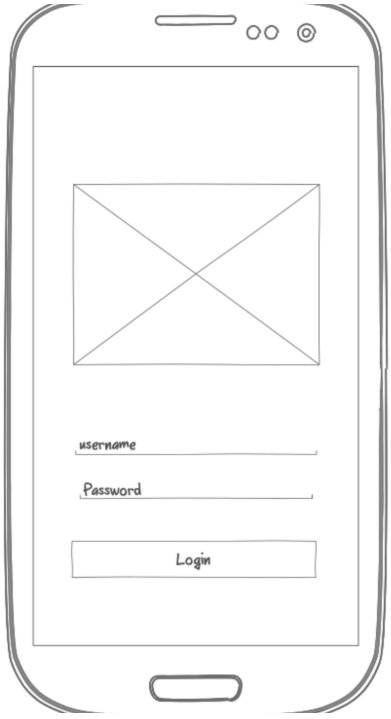
- Support Accessibility
    - Add content description in all the views.
    - Handle tab navigation in the log in screen
    - Show the input method when activity starts
- Material Design metrics
- Show technology products
- Show product per category
- Show product detail
- Notifications arrives with the best seller product
- Widget of the last buy
- Analytics
- Admob
- Share product
- Login

Library	Version
Android Studio	3.3
Gradle version	3.3.2
Android build tools version	28.0.3
Android target sdk version	28
Android compile sdk version	28
Android constraint layout	1.1.0
Android support version	28.0.0
Glide version	4.8.0
Butterknife version	8.8.1
Retrofit version	2.5.0
Dagger version	2.17
Timber version	4.7.1
Room version	1.1.1
Lifecycle version	1.1.1
debugDb version	1.0.4
Gson Converter	2.5.0
Espresso version	3.0.2

Library	Version
Test runner version	1.02
Test rules version	1.0.2

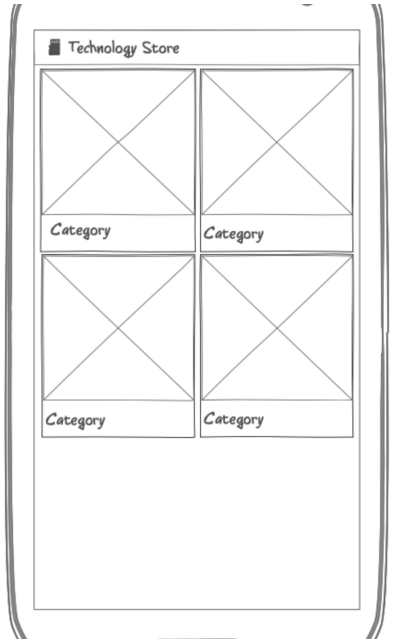
User Interface Mocks

Screen 1



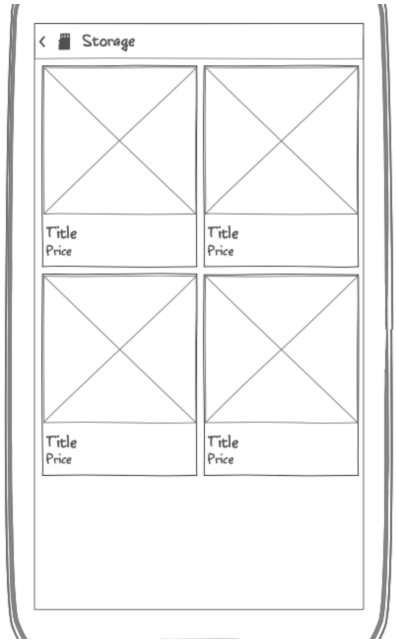
Log in with username and password

Screen 2



This view shows all the categories available from API.

Screen 3

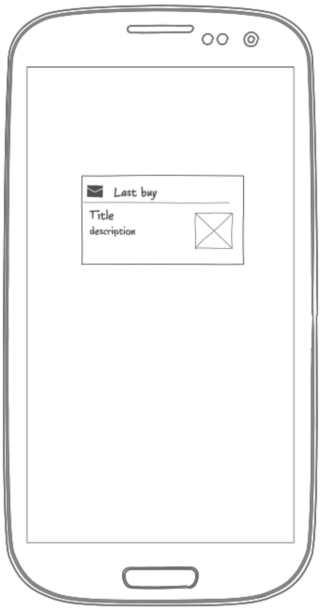


This view shows the name of product and the price, the title is the same as the category chosen before.

Screen 4

This view shows all the detail and the buy button. You can chose the amount as well.

Screen 5



The widget shows the last buy from the app.

Key Considerations

How will your app handle data persistence?

The app will handle data persistence with Room from architecture components.

Describe any edge or corner cases in the UX.

Create custom components and backgrounds for the detail view. Add transition from the cards to the detail

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

- Glide for loading images from url
- Dagger for dependency injection
- Architecture components (Livedata, view model , Room)
- Retrofit for API calls
- Timber for show logs
- Butterknife for binding views

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

I will use Apiary mock end points for network request.

Next Steps: Required Tasks

Task 1: Project Setup

- Configure libraries
- Update the last version per library
- Create the packages per layer
- Configure the applicationId
- Manage the dependencies
- Create keystore for release
- Create signing configuration for release

Task 2: Create de Dependencies

Since we are using Dagger as Dependency Injection

- Create all the libraries dependencies
- Create the Scope activity
- Create dependencies for ViewModel and Data Live

Task 3: Create the folders in each layer

- Presentation layer
    - Create folder for log in view with activity and fragment
    - Create folder for List of products with activity and fragment
    - Create folder for Detail view with activity and fragment
- Domain Layer
    - Create all the entities
    - Create all the uses cases
- Data layer
    - Create the Repository pattern
    - Create the data source (disk and network)
    - Add the end points

Task 4: Implement UI for Each Activity and Fragment

- Create styles
- Create Strings for messages
- Create layouts
- Create classes (Activity and fragments)
- Match layouts with classes (Activity and fragment)
- Create material design transition
- Create the widget of the last buy : ***App implements Service(RemoteViewsService) for developing Widget and the app uses an IntentService to update it.***
- Create notification after the user log in. The application will simulate a notification with mock product data.
- Live data and view model will notify to the view (fragment/ activity) for render information after the use case was executed.

Task 5: Implement end to end flow

After create all the views, we should be create the uses cases and end points for the views.
Every action in the applications is consider as use case. Create those and connect with the endpoints that provide the information. After the endpoint send de data, the application should be map it and then pass al the information to the view. Finally the view render all this information.