How to Use this Template

1. Make a copy [File → Make a copy...]
2. Rename this file: “**Capstone_Stage1**”
3. Replace the text in green

Submission Instructions

1. After you’ve completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it “**Capstone Project**”
3. Add this document to your repo. Make sure it’s named “**Capstone_Stage1.pdf**”

GitHub Username: fabricio-suarte

Task Manager

Description

This app helps users to manage a kind of “To do” list, making possible to register tasks that should be accomplished. It is possible to set a due date, an alarm, a location and chose if it is a priority task or not. Easily navigate through your task list and mark they as done, edit, etc.

Intended User

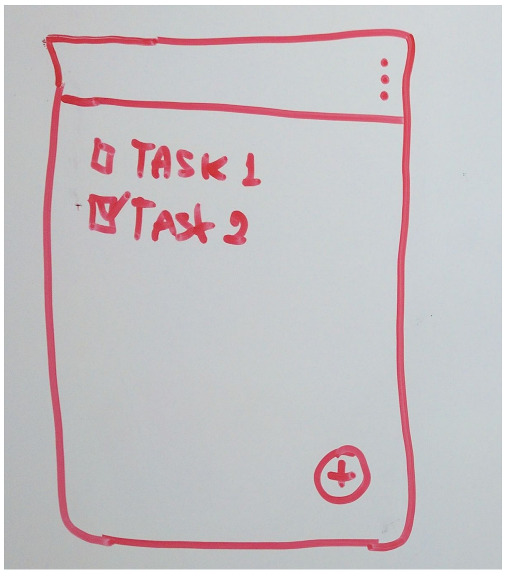
Any people that need or want a simple and efficient way to organize, schedule and accomplish tasks.

Features

- Saves registered tasks
- Allows alarm setting (a notification is sent) for tasks
- Allows location setting for tasks
- Allows tasks to be marked as “done”
- Automatically remove “done” tasks periodically
- Allows some settings (tasks list order and periodicity of execution for the auto removing done tasks job)
- App provides a widget that shows tasks having “today” as due date.

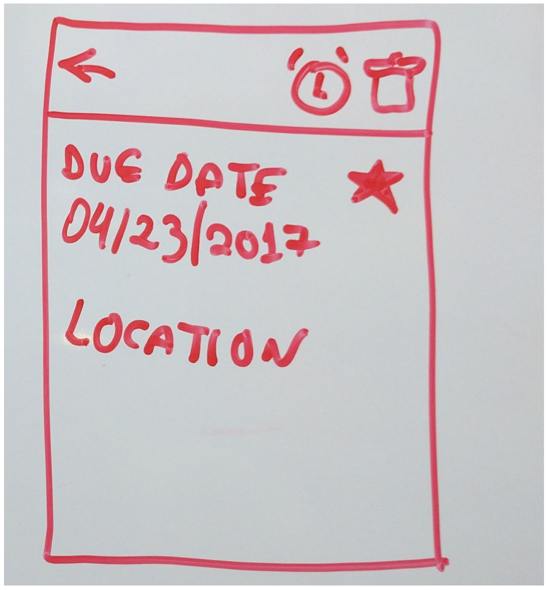
User Interface Mocks

Screen 1: Main Activity



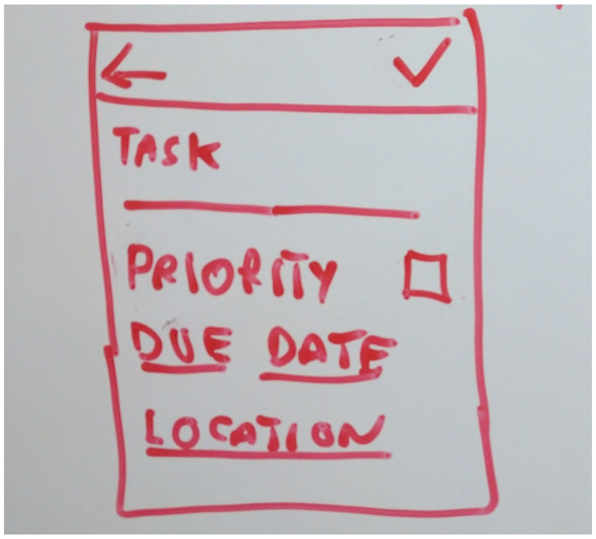
The main activity, containing a list of registered tasks, a FAB to add new tasks and the a taskbar options menu item. It is possible to mark / unmark tasks as “done”. Clicking on a list item (task) leads to the details task activity.

Screen 2: Details Activity



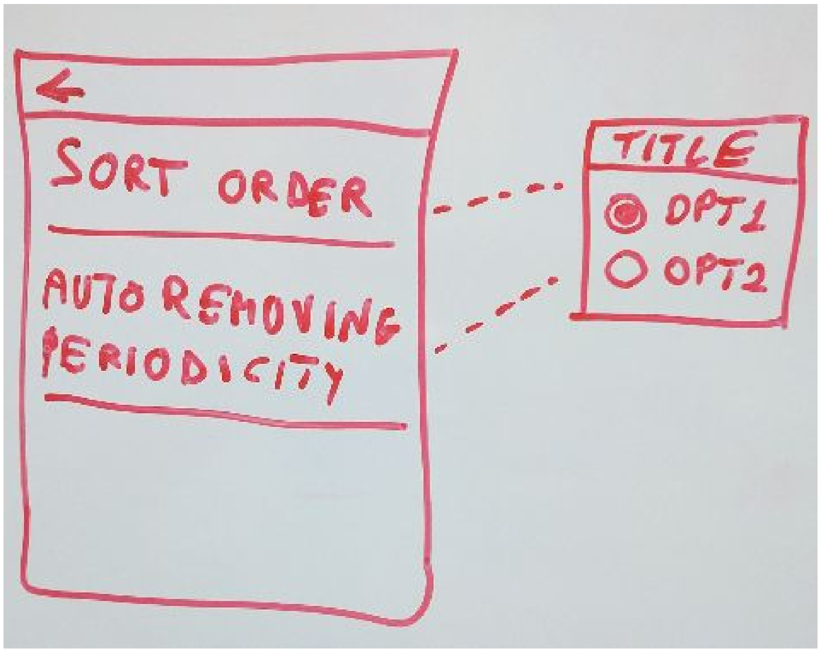
The details activity show information of a given task: a due date (if set), priority task or not (star) and location (if set). It is possible to schedule an alarm notification for a given date (clock icon in the action bar) and delete the task (trash can icon in the action bar).

Screen 3: Add new task Activity



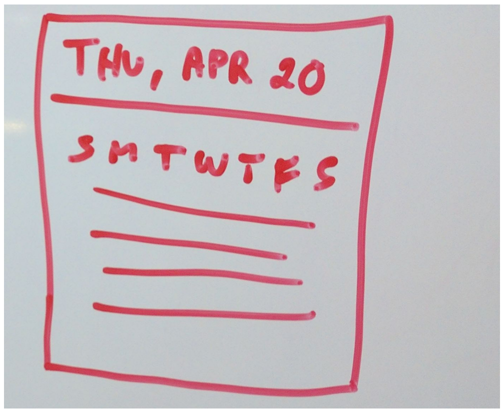
The Add Task Activity: you can provide a description for what is your task, set if it is a priority task, set a due date and a location. Clicking on “Due Date” and “Location” will lead to the “Date picker Dialog” and “Location picker Activity” respectively. Clicking on the “Check” action bar icon, will save the task.

Screen 4: Settings Activity



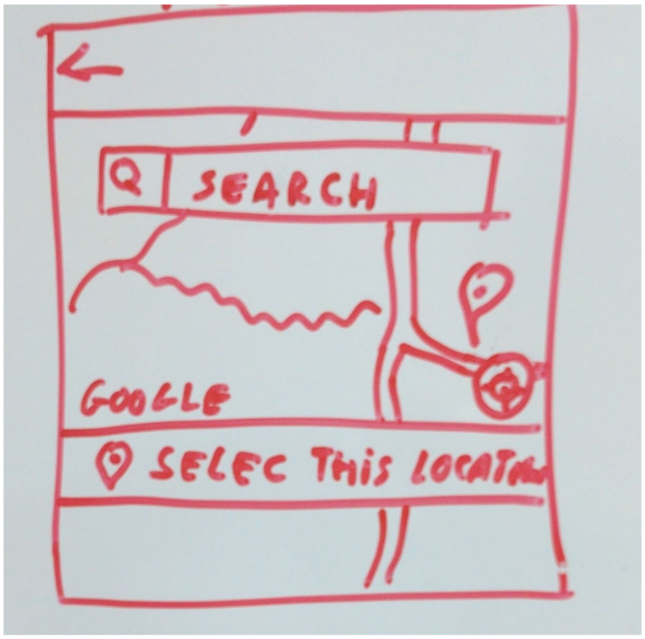
The settings Activity, where it is possible to set the sort order of tasks and the periodicity of the auto removing service job execution.

Screen 5: Date picker dialog



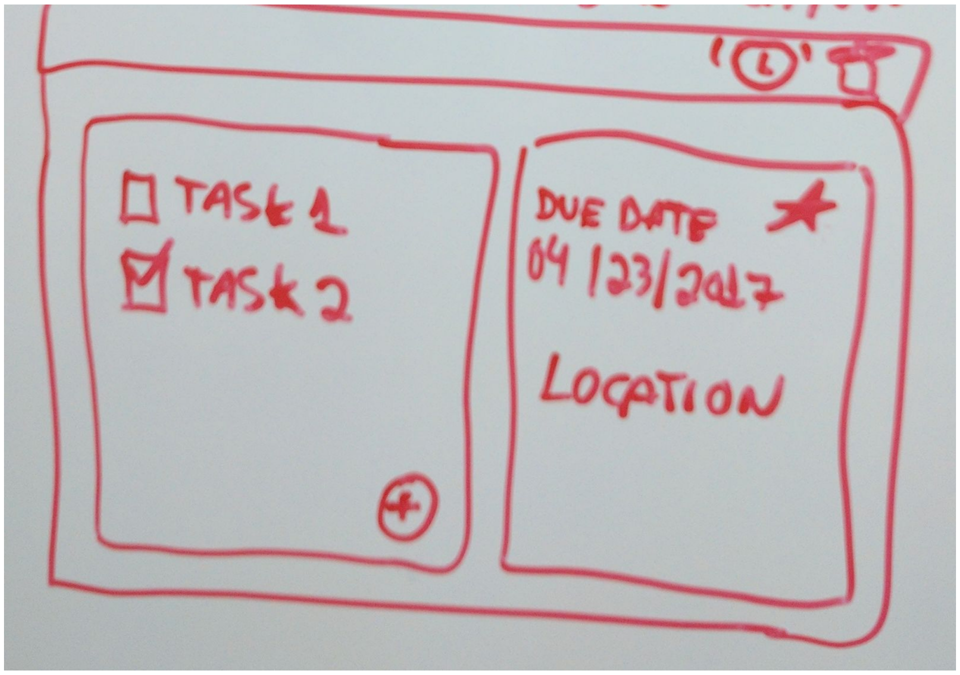
Just a standard Android “Date picker Dialog”, for picking due dates and alarms sets.

Screen 6: Location picker Activity



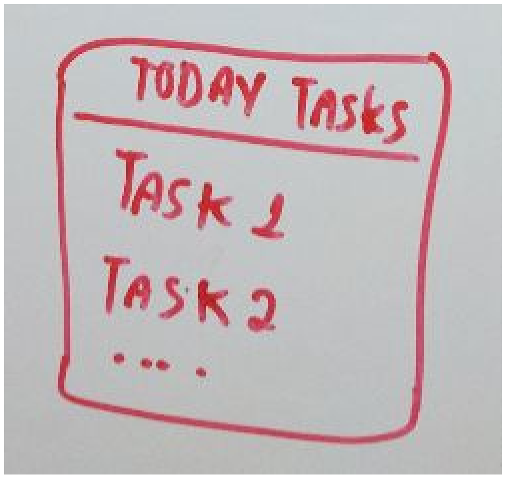
This is going to be an activity using Google Maps for allowing user to search and select a location for associating to a task.

Screen 7: Large Screen / Tablet Layout



A large screens / tablets layout. The main content is shown in the left while detail content is shown in the right.

Screen 8: Due date tasks widget



A widget that shows a list of tasks that have a due date for today. If a task is clicked, the app should be opened at the details activity for the clicked task. If a generic area of the widget is clicked, the app is opened normally.

Key Considerations

How will your app handle data persistence?

The app will build its own content provider and persist data using the SQLite engine.

Describe any corner cases in the UX.

N/A

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

Butter Knife: makes easy the referring to UI components in your Java code and substantially reduces the amount of code you have to write to accomplish this.

Describe how you will implement Google Play Services.

Basically, at least two Google Play services will be used:

1. Firebase Crash, for crashing handling and crash report / analysis.
2. Google Maps, for searching a location and task association.

Next Steps: Required Tasks

Task 1: Project Setup

- Setup a new GIT repository and link to remote Github
- Create a new Android project
- Configure libs, basic gradle configuration
- Setup a release keystore and a key for release compilation

Task 2: Modeling the layout solution

- Think about how UI components can be built in order to promote great reusability.

Task 3: Modeling domain problem

- Create domain classes
- Create helper classes

Task 4: Implement the content provider (persistence)

- Create database helper
- Create content provider

Task 5: Implement UI and Java Code for Each Activity and Fragment

- Build main fragment
- Build main activity
- Build main activity menu
- Build details fragment
- Build details activity
- Build add new task fragment
- Build add new task activity
- Build settings fragment
- Build settings activity
- Build location picker fragment
- Build location picker activity

- Build date picker dialog
- Build tablet layout

Task 6: Implement Google Services

- Implement Firebase Crash service
- Implement Google Maps service

Task 7: Implement application services

- Build cleanup job service (periodically removes tasks marked as “done”)
- Build reminder alarm service

Task 8: Implement application widget

- Build widget layout
- Implement java code

Task 8: Assure accessibility best practices

- Check if UI components have set properly (content description)
- Check if navigation pad works properly

Submission Instructions

1. After you’ve completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it “**Capstone Project**”
3. Add this document to your repo. Make sure it’s named “**Capstone_Stage1.pdf**”