How to Use this Template

1. Create a new document, and copy and paste the text from this template into your new document [Select All → Copy → Paste into new document]
2. Name your document file: "**Capstone_Stage1**"
3. Replace the text in green

GitHub Username: Ammar-JT

Skill Tracker

Description

Skill Tracker helps you track your time spending on skills. You can add skills like Programming, Play Piano, Read Book, etc. And then record your daily activities by hours & minutes.

Intended User

Self-Learners, Freelancers, Readers, Artists, etc.

Features

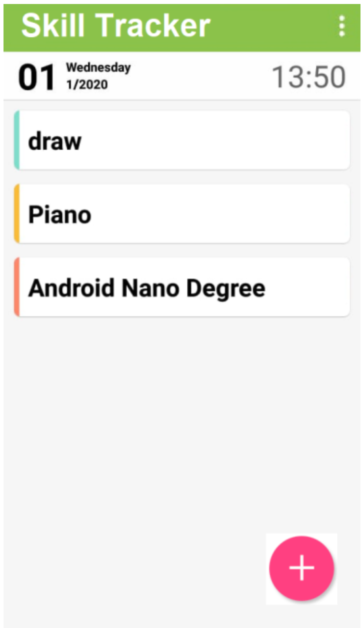
List the main features of your app. For example:

- Add new skill
- A timer for a skill
- A calendar shows you your progress for each day
- A Main page with list of all your skills

User Interface Mocks

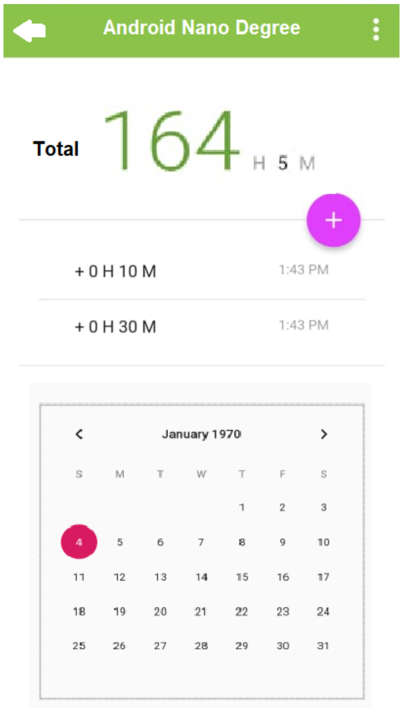
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1



This is the Main Screen, it contains all the skills you added, it also has a FAB to add a new skill.

Screen 2



S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

This is the single skill screen, it has:

1- Total number of hours and minutes for this skill.
2- A button to add time (stop watch or add times directly)
3- A Recycler view to show you all your records for this day.
4- A calendar so you can know the day and the progress of that day using a colored circle (the darker the color the more time spent).

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

Key Considerations

How will your app handle data persistence?

I will use room to deal with SQLite database. Also I will handle it using AsyncTask in the background thread. There will be 2 tables, one for Skills and one for times (will be stored in minutes), and I will use the appropriate schema to create the DB and to build the relations between tables.

Describe any edge or corner cases in the UX.

If user hit the back button, it will return to the Main Screen, which contains all the skills.

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

Butter Knife: to make variable initialize and creation easier.
Recycler view: for list of skills in main activity, and for time records in the single skill activity.
Admob: to get money from the free version.

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

I will use Room to handle the database, and Admob.

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 an empty project using API 29 as default SDK and API 15 as minimum SDK
- Configure libraries and add dependencies to gradle.build
- Make a free and paid variant for the app

Task 2: Implement UI for Each Activity and Fragment

List the subtasks. For example:

- Build UI for MainActivity
- Build UI for singleSkillActivity

Task 3: Create Main Activity

- FAB to add new skill
- Recycler View to list all the skills
- A clickable Recycler view Items.
- An intent to take the user to the singleSkill activity after clicking on the RV item.

Task 4: Create add new skill fragment

After user click at add new skill button:

- Create a fragment layout to fill it with the views.
- Edit text view to add the name of the skill
- Edit text view for the Initial hours user has already done before adding the skill to the app
- Add Skill button

Task 5: Create Single Skill Activity

- A calendar view so you can know the day and the progress of that day using a colored circle (the darker the color the more time spent).
- A FAB to start a Chronometer view to use it as stop watch.
- A Recycler view that shows you all the records for today.
- Total number of hours (calculated by concatenate all the records of this skill)

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**"