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

GitHub Username: hndykstra

Pomodoro timer

Description

Reference: https://en.wikipedia.org/wiki/Pomodoro_Technique

A convenient, portable method to track and time pomodoros, especially during group meetings to allow appropriate short and longer breaks.

This is conceived as release one of a two release cycle.

Intended User

The intended users are any business or technical users who typically have heads-down meetings or tasks over extended periods of time. The application will allow users to structure their work or meetings so as to maintain mental alertness and physical sense of well being, as well as to track timing and accomplishments.

The primary targeted platform is mobile phone as the intent is to be convenient and portable, even during breaks.

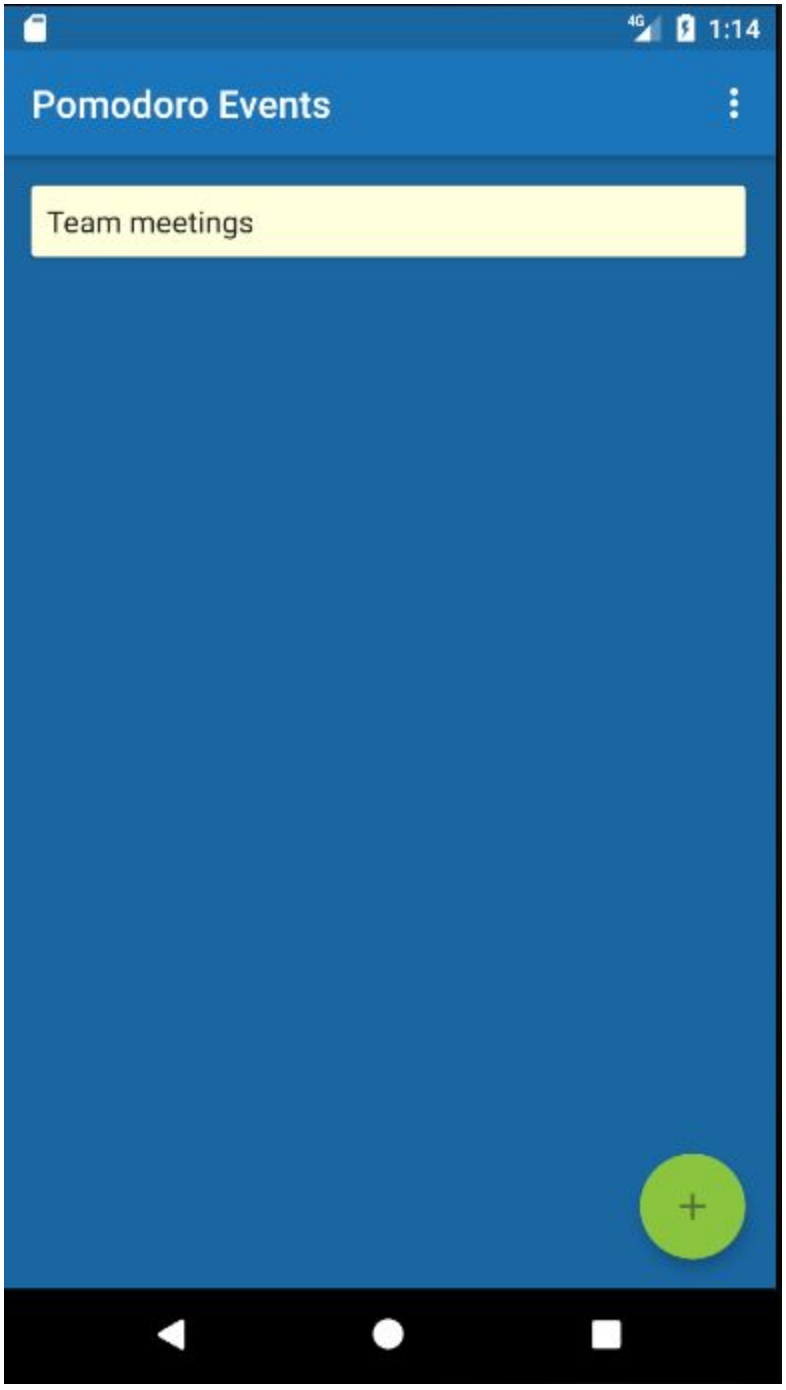
Intended users are expected to be fairly tech savvy so support for older Android versions is not considered a priority.

Features

- Starts events with a specified work ./ break timing profile.
- Signals when a work or break session is complete.
- Saves information so that adherence and productivity can be correlated.

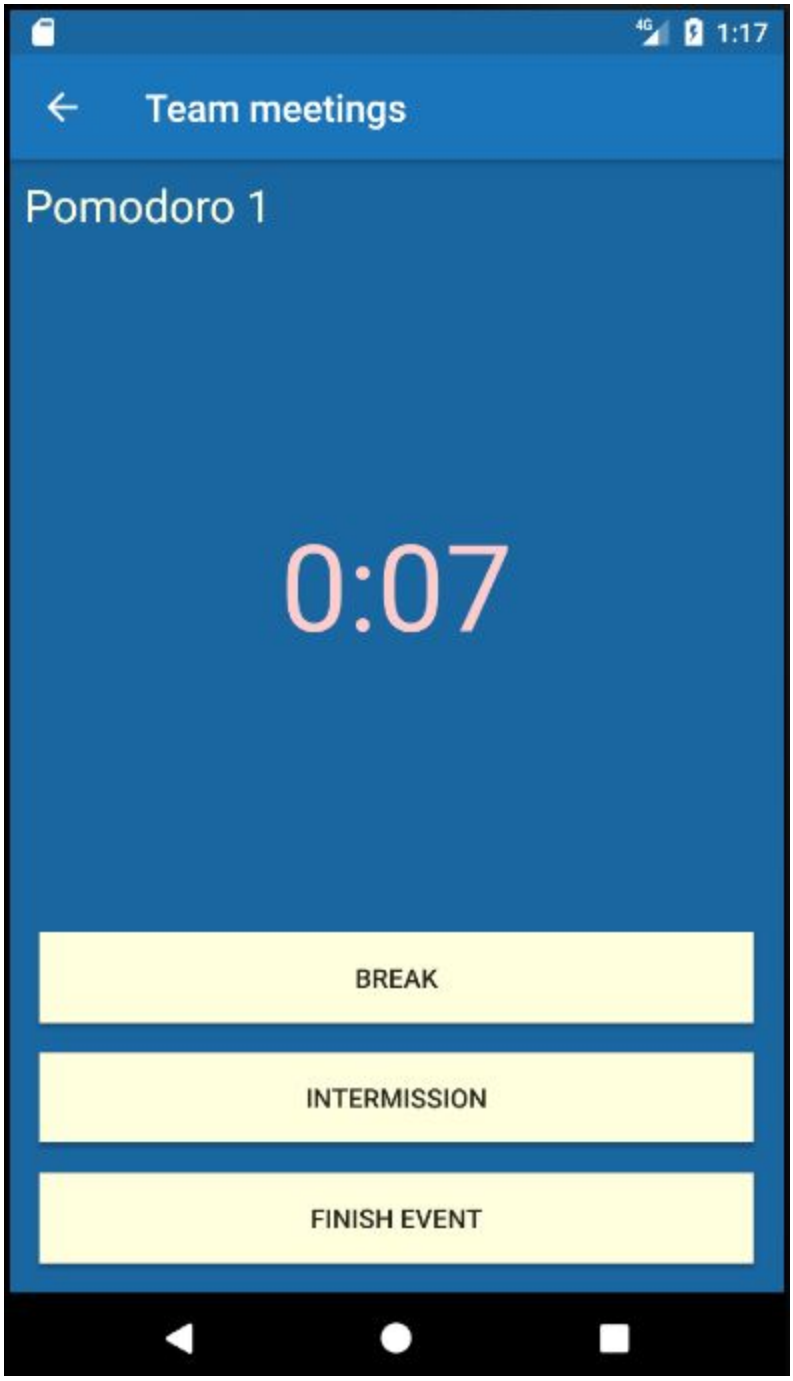
User Interface Mocks

Screen 1



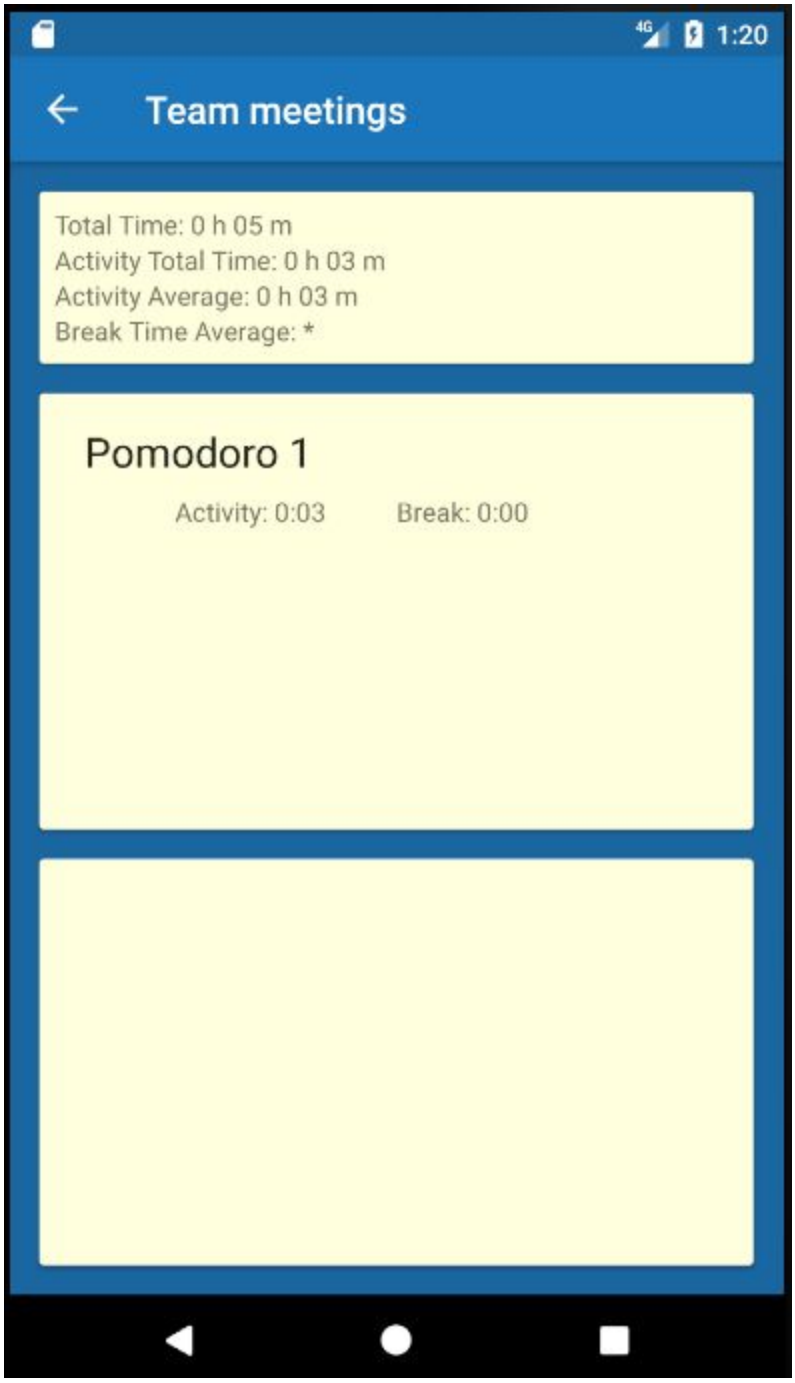
Main activity screen shows a list of current and past events, and allows the user to create a new event, rejoin an existing one, or view statistics of a past event.

Screen 2



In progress screen views the timer for the currently active pomodoro, status (active or on break) and allows simple transition between states (active -> break -> next pomodoro, or active / break -> intermission, or active / break / intermission -> end event). Timer counts up to show elapsed time and optionally sounds a tone when the alloted time is reached.

Screen 3

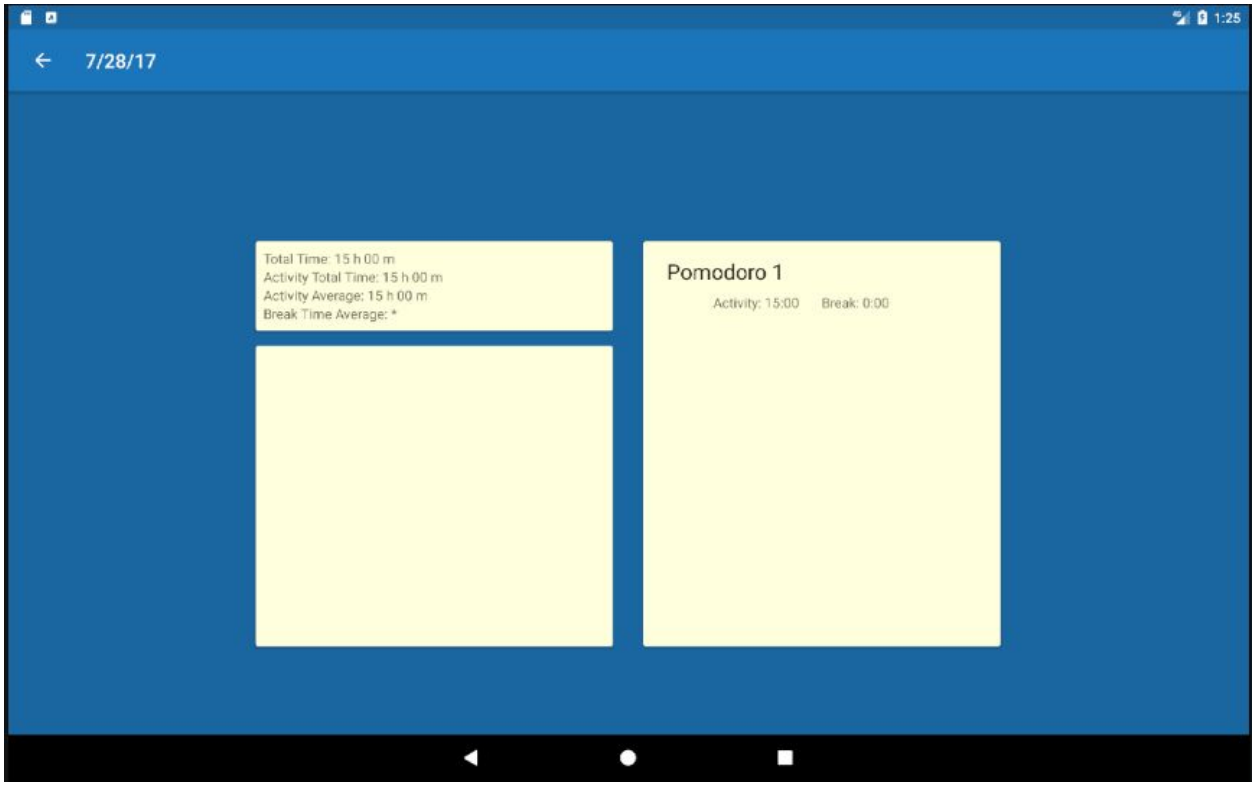


Summary screen shows when selecting a past event from the main activity, or when you finish an activity. It shows basic statistics and details of the activity. In future when shared events are possible, the lower section may show a list of people who joined the meeting.

Alternate layouts

The primary target is mobile phones. Layouts for landscape and portrait modes will be required.

Tablets will be supported but the primary goal of tablet style layout is to keep the UI tight and coherent by preventing elements from spreading out excessively.



Widget

Widget provides a quick view / link to the most recent activity.

Key Considerations

How will your app handle data persistence?

In this phase, data persistence is shared via Firebase database.Real time updates allow remote users to join in-progress events and share state. (For example, team members who are remote will be able to see pomodoro and break timing.

Describe any edge or corner cases in the UX.

Primary edge case occurs if the user navigates away on their phone or the phone goes to sleep. The state must be restored, including any in progress timers or any changes to the event state that have occurred remotely.

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

The only external library in use is ButterKnife for binding..

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

Firebase Auth UI will be used to identify users so they can join teams and events. Firebase database will be used to persist data and share updates..

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 project. Configure gradle build with libraries for recycler view and other required support libraries.

Task 2: Implement UI for Each Activity and Fragment

- Build main activity, recycler view, adapter, handler for FAB.
- Add timer activity to display current pomodoro unit timer with actions to move from work activity to break, continuing until the task / event is ended.
- Add event summary screen.
- Include FirebaseAuth support for login.

Task 3: Implement Firebase

- For this release cycle, implement the data model and queries to FirebaseDatabase.

Task 4: Implement Event Logic

- Logical transitions to enter an event, start timing, move between states - waiting, activity, break, intermission, end of event.

Task 5: Implement Event Summary

Read the final state of the event and display statistics and details.

Task 6: SyncAdapter

The SyncAdapter is required for project specs, so I will implement one to query Firebase whenever the system is online, to force Firebase to cache key data for offline availability. This is pretty much useless as the key data will automatically be cached whenever the user starts the app, but it is required.

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**"