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: ducdungdam

App: Dart Friends!

Description

Dart Friends! is an App which lets you challenge your friends in one of the most popular bar games! Supporting varieties of game modes you can play from tournament mode to fun games, alone or with your friends. Track your scores and see who is the Hawkeye among your Friends - Happy hitting Bullseye!

Intended User

This app is intended to everyone who has a dartboard and wants to count a game with his friends or track is statistics. So the user group are between teenagers and mid-aged users.

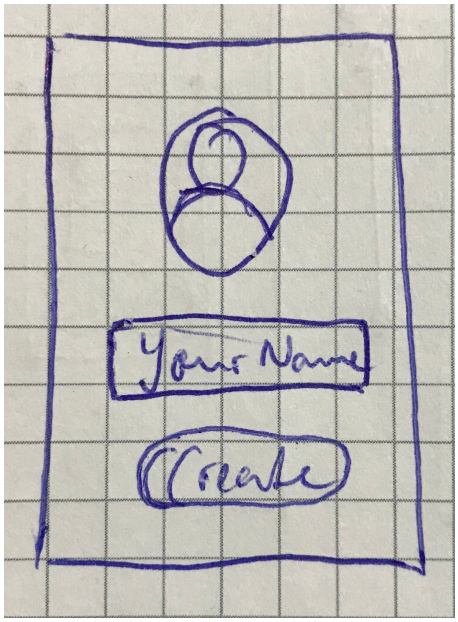
Features

- Play two game modes (301 and 501)
- play alone or with friends
- Track player statistics (e.g. score average, games won)
- Add and delete new players (also add Image by gallery or take new one)
- Game mode description
- See achievements
- Paid and free app
- Available in light and dark theme, English and german

User Interface Mocks

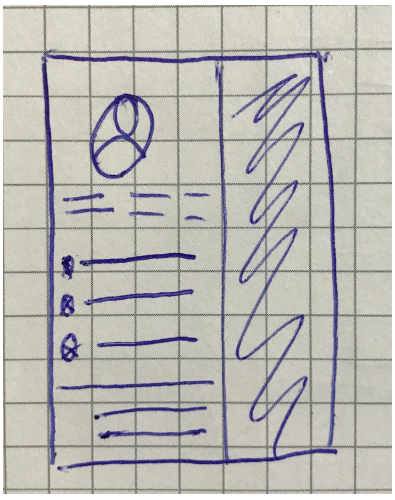
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Welcome Screen / Create Player



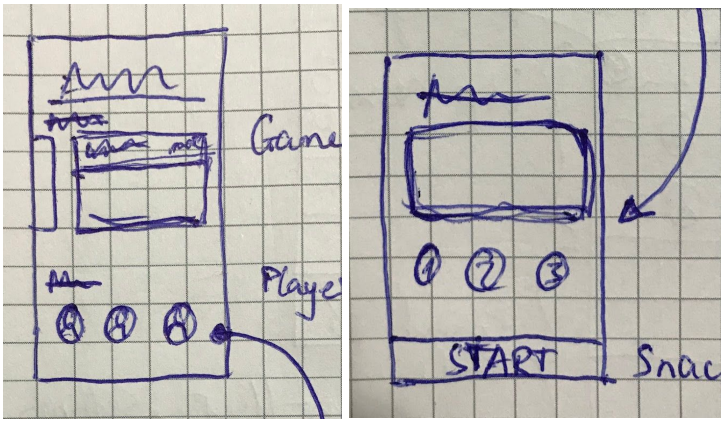
This is the screen a user see, when he first time launch the app. Since this app needs a user, its asking to create one. This screen is similar to the create player screen. Its Providing an ImageView, which a user can tap on it and take a picture with the cam or select one from the gallery. The EditText ist for typing in the player name. The Create Button for wrapping up the process.

Navigation Drawer



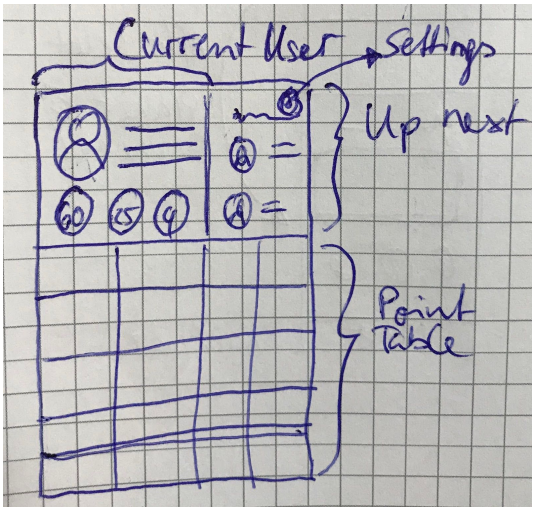
This Screen shows a custom navigation drawer. The intention of the Navigation Drawer is, to let the user easily navigate through the app. The Navigation Drawer providing interesting information about the user, like his avg. 3 hit score or his won games. It is also scrollable, whereas the user image will behave like a coordinator layout and get smaller.
The Navigation Drawer will be divided into two groups (main and sub). Tapping on the user image will direct him to the player statistics

Game Selection Screen



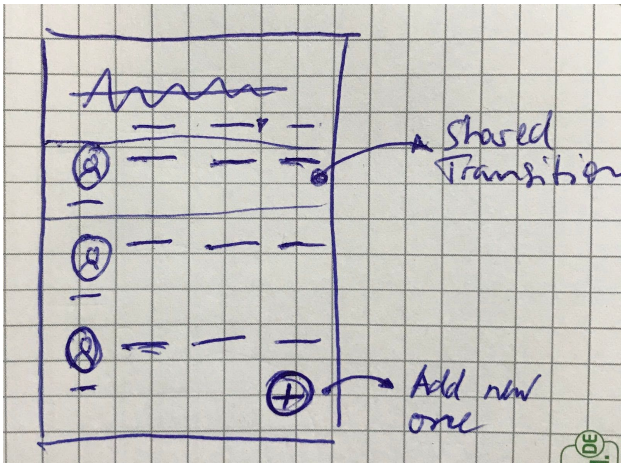
This Screen allows a user to select a game mode. By swiping horizontally, the game mode card which is showing, will be the selected game mode. The card contents the game mode title and a short description about the mode. There will be a ‘read more’ button, for a brief description. Scrolling down, the user can select all existing user. By tapping a player in order, he will also defines the order of players for the game. If at least one player is selected, a SnackBar will slide in, which indicates to start the game.

Game Screen



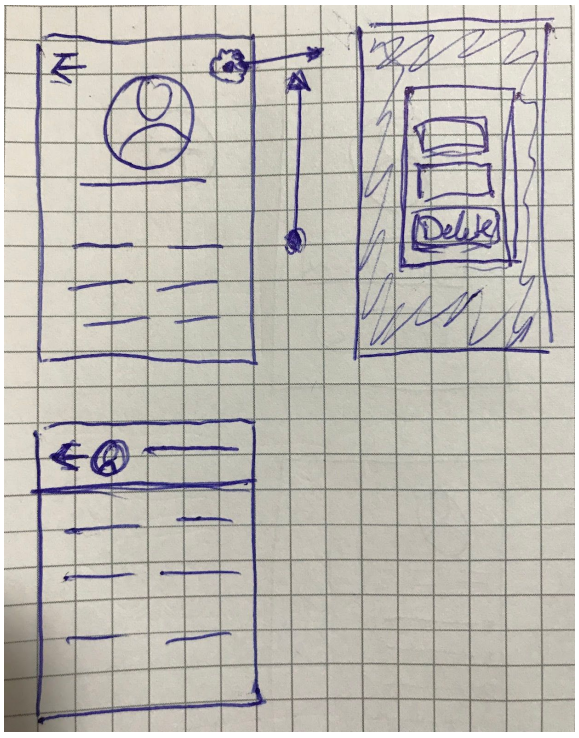
The Game Screen is structured in two sections. The upper one shows the current player and his 3 thrown scores, combined with his current score. On its right there are players who are up next. There is a settings icon, to allows the user to exit the current game. The user can also tap on the single score to edit it. The bottom section is the point table, where the user can tap the score, which a player made (including combinations like double + point or triple + point). If all 3 score are set, a FAB will pop up to get to the next user.

Player statistics screen



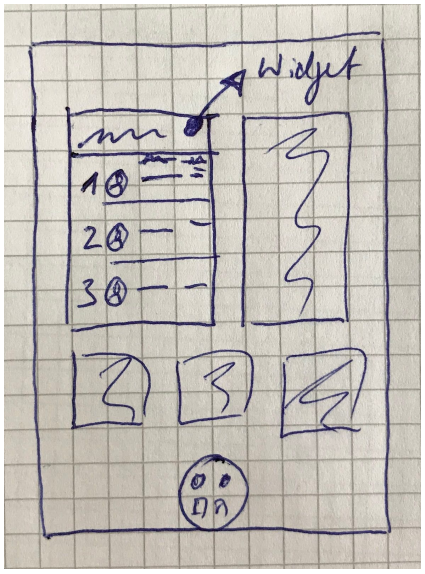
This screen shows a table with all existing players and their statistics, like best rate (most points with less throws in a game mode), won games in game mode, or avg. score of 3 throws. The table can be sorted and the items are clickable, so a detailed screen of the player will be shown. There also will be a shared transition between this two views. A FAB in this screen will provide a core function to add new players which will direct the user to the create player screen.

Player details screen



This screen shows more details of a player. It's also provides a 'setting dialog menu' which allows the user to delete this player or configure him (e.g. change image, change name). It behaves like coordinatorlayout.

Home Screen Widget



Widget shows, depending on its size, a table of a player ranking (sorted by won games). Tapping on a player on the the widget will direct the user to the player details statistics screen. This function will be realized by an IntentService.

Key Considerations

The App will be written solely in the Java Programming Language by using
Android Studio 3.1.3 with Gradle Plugin 3.1.3. The App will have a minSdkVersion of 19 and will target at API 27, since API 28 is still on alpha phase. Therefore all Android related libraries (e.g. support-appcompat, cardview, design) will use stable versions of 27.1.1.

How will your app handle data persistence?

The data of players (user, statistics) and game modes (description, game logic) are handled in a Content Provider. All data will stored and managed by Room (1.1.1).
Following Table with Entities will be created:

1. game_mode
    a. game_mode_id
    b. short_description
    c. description
    d. image_path

2. player
    a. player_id (Main User is always player_id == 0)
    b. name
    c. image_path
    d. total_won
    e. total_score
    f. score_average

All strings are kept in a strings.xml file and app enables RTL layout switching on all layouts. All strings will be provided in two languages, where english will be the default one. For the alternative language, german, all depended files will be handled in the folder with its specific qualifier (*-b+de).

For accessibility, all images are provided with a content description.

Describe any edge or corner cases in the UX.

While in a ongoing game, the user will return to the game screen after the user hits the home button and returns to the app.

If the user hits the back button while in a game, it will ask him whether to end the game and returns to the game selection screen.

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

Picasso (2.5.2): Easy way to load images and handle placeholders
Timber (4.7.1): Easy way for logging
Arch.Lifecycle (1.1.1): Easy way to separate Data and View code for viewmodel, as well as handling asynchronous tasks.

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

- Google Mobile Ads (15.0.1): Ads are as popup when the game starts and when it ends
- Google Play Games Services (15.0.1): Adds achievements (first time hit bullseye, first time 3x Triple 20, first time reached 1000 points in total)

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

- Configure libraries
- Enable DataBinding
- Configure Flavours for paid and free version
- Setup Project Folder structure and files (modifier related files, language related files)

Task 2: Implement UI for Each Activity and Fragment

- Using ConstraintLayout, DataBinding, Implement UI interactions
- Implement Custom navigation drawer
- Build UI for MainActivity (Paid and Free)
- Build UI for GameSelectionFragment
- Build UI for GameModeDescripitonFragment
- Build UI for PlayerStatisticsFragment
- Build UI for WelcomeActivity / PlayerCreation
- Build UI for GameActivity (Paid and Free)
- Build UI for SettingsFragment
- Build UI for LegalFragment

Task 3: Implement Core Functions

- Using ViewModel for processing Data
- Implement Database (Room)
- Implement ContentProvider
- Implement Add and Delete Player Function
- Implement Statistics handling
- Implement Google Services AdMob for ads
- Implement Google Play Services for Game Achievements

Task 4: Implement game logic and functions

- Implement Game logic for game mode (301, 501)
- Implement start, win, exit and resume handling
- Implement score tracking

Task 5: Implement App Widget

- Implement UI for App Widget
- Implement logic and data handling for App Widget
- Implement IntentService to direct user to the player detail screen, of the player he has clicked on in the widget

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**"