Project Parameters	
Android Studio Version	3.4.2
Compile SDK Version	28
Min SDK Version	15
Gradle Version	4.10.1
Libraries Used	AppCompat 28.0.0 Design 28.0.0 ButterKnife 10.2.0 ButterKnife Compiler 10.2.0 Picasso 2.71828 Junit 4.12 Espresso core 3.0.2

GitHub Username: beboTak

BeFitness

Description

This app help user of all fitness levels and provides a tool that will help generate workouts based on selected input such as muscle group, equipment available, desired duration, etc. Data is pulled from the Workout Manager API https://wger.de/en/software/api.

In addition, since exercise is a regular activity that requires consistency to progress, users can save their workouts to return to them during future sessions.

With this app, users will also be able to find nearby gyms to streamline the starting process instead of having to worry about the details.
With a clean, intuitive user interface, users can spend less time figuring out what exercises to do and focus on the routines themselves.

Intended User

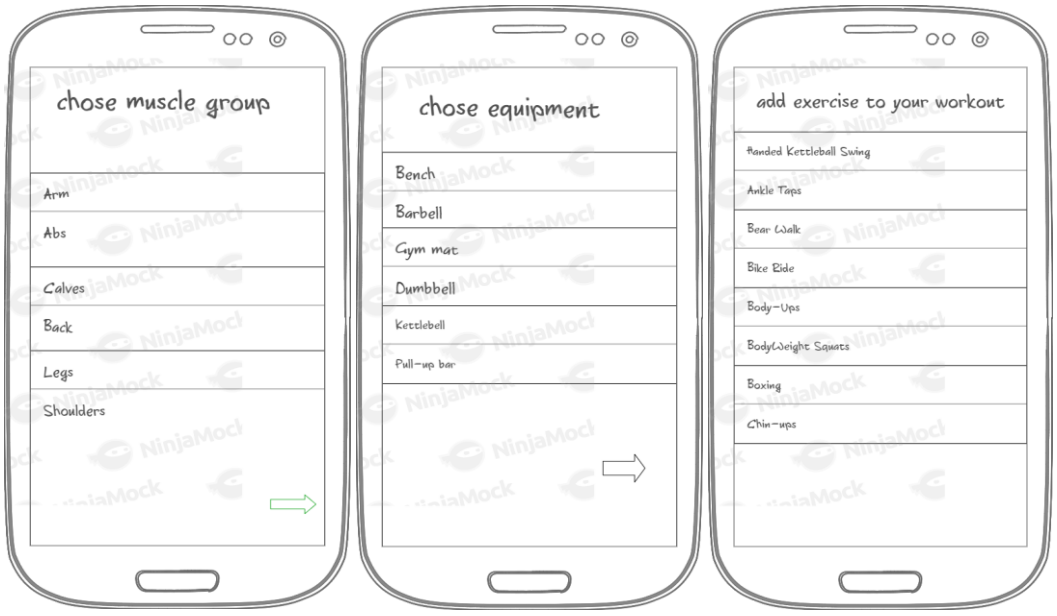
Target users will include people of any fitness level who wish to take a serious approach to their exercise regimes.

Features

● Accepts user-selected parameters to generate workouts
(user chose muscle want to generate workout on it from first screen and click go to second screen to show equipment and select equipment and go to third screen that show exercise)
● Saves favorite workout plans
(in third screen you can save your workout)
● Find nearby gyms based on user-input location

User Interface Mocks

Generate a workout



.

This fragment will follow a simple flow that allows users to select input muscle groups and equipment to generate available exercises that they can select and add to their custom workout.

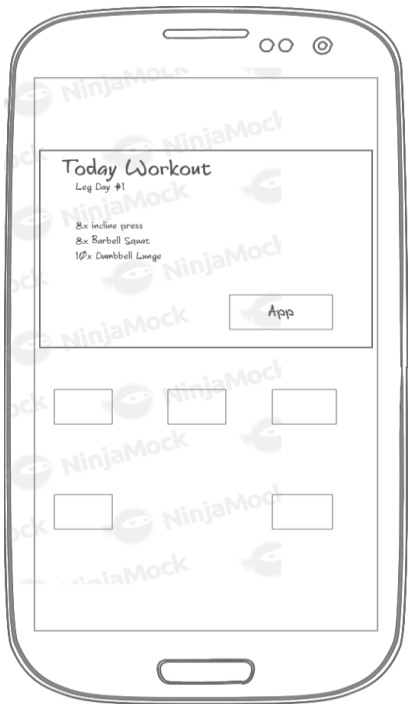
View Saved Workout

When users have created their own custom workouts, they can view the list of saved workouts and return to them in future sessions

View Saved Workout

When users have created their own custom workouts, they can view the list of saved workouts and return to them in future sessions.
And find near gym.

Widget View :Last view Workout



Users can add the last viewed workout to their lock screen via a widget. From this widget, they can view more of the workout in their app by clicking a button to open it.

Key Considerations

How will your app handle data persistence?

Data persistence will involve connecting to a Room database This will be used to save stored user workouts.

Describe any edge or corner cases in the UX.

Navigating between each “Activity” (implemented as Fragments for this app) will be handled with a Bottom Navigation Bar.
When a user generates a workout, they will be able to follow the flow until the final screen, which will allow them to add any exercises they wish to their workout. At this screen, the submit button will take them back to the beginning of the generate fragment.

For the other two fragments (finding a gym and viewing saved workouts), they will only be one screen fragments, although the detail view for saved workouts will be able to return back to the
parent via the up button.

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

The app will use ButterKnife to focus more on logic code than having to bind each view

manually. It will also use Picasso to easily load images from the workout API into their respective ViewHolders

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

Maps, Location, and Admob will be used in this app. Maps will provide the location information of all the nearby gyms based on the user's current Location, or from a given input location. Admob will be used throughout the application, sparingly, to provide users with ads.

Next Steps: Required Tasks

Task 1: Project Setup

- Create project in Android Studio
  - Add a Bottom Navigation Activity as the MainActivity
- Application will be written entirely in Java
- Configure libraries
  - Add ButterKnife dependency for easy view binding
  - Add Picasso dependency for image loading capabilities
- Create strings.xml file to store all string values used throughout the app
- Enable RTL layout switching in overall app structure
- Add Internet and Google Play Services permissions to manifest.

Task 2: Implement UI for Each Activity and Fragment

- Create Fragments for each feature of the app
  - Generate a workout
  - View saved workouts
  - Find a nearby gym
- Connect each Fragment to the MainActivity using SupportFragmentManager
- Add AppBar to Activity and corresponding navigation features
- Add individual RecyclerView to each screen of the Generate Workout fragment
- Add RecyclerView to Saved Workouts Fragment
- Add map placeholder to Find Gym Fragment

Task 3: Your Next Task

- Implement corresponding adapter and layout manager for each RecyclerView
- Create network utilities to help build URLs and make network requests
- Extend AsyncTask class to fetch data from Workout Manager API

Task 4: Your Next Task

- Implement Room Database (make model ,Entities Dao ,database class,etc)
  - Connect to Room database
  - Implement database management functions to interact with saved workouts
- Implement Loader class to add any data from database into views

Task 5: Your Next Task

- Implement Admob into placeholder views
- Implement Location feature for getting user's current location
- Implement Maps feature into map placeholder
  - Find nearby gyms based on user's input location from Location feature

Task 6: Create test cases for UI and logic

• Add UI tests to test the edge cases of the UX
o Test views that may be affected by Admob
• Add unit tests for various application logic components
o Test to see if exercise data is being retrieved from the API
o Test to see if nearby gyms are generated correctly
o Test to see if saved workouts are correctly saved and retrieved from SQLite database