GitHub Username: NecoHorne

Body Planner

Description

Exercise planner app.

- Java language will be used for development
- Create your own exercise plan from a database of exercises.
- Plan what body parts you are going to hit what day and what exercises, sets and reps you are going to do.
- Body calculator
- Body measurement and tracking to track your progress

Intended User

People who go to the gym, people who want to plan their workouts and track their progress.

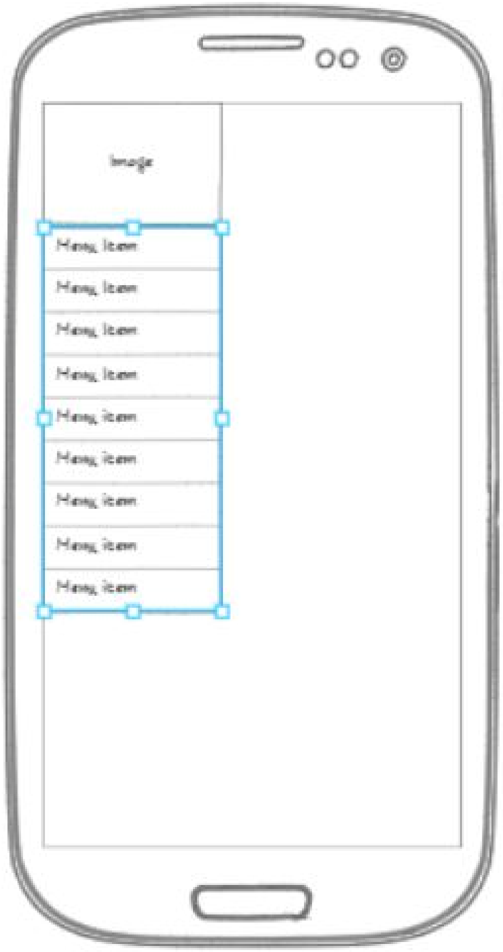
Features

- Plan your workout for each day from a database of exercises.
- Exercise database to provide instructions
- Track your progress weekly or monthly

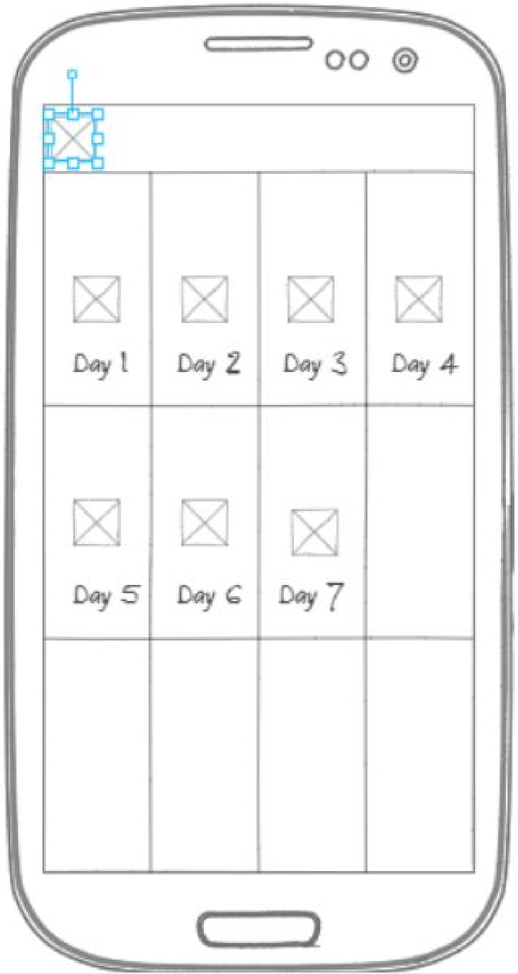
User Interface Mocks

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.

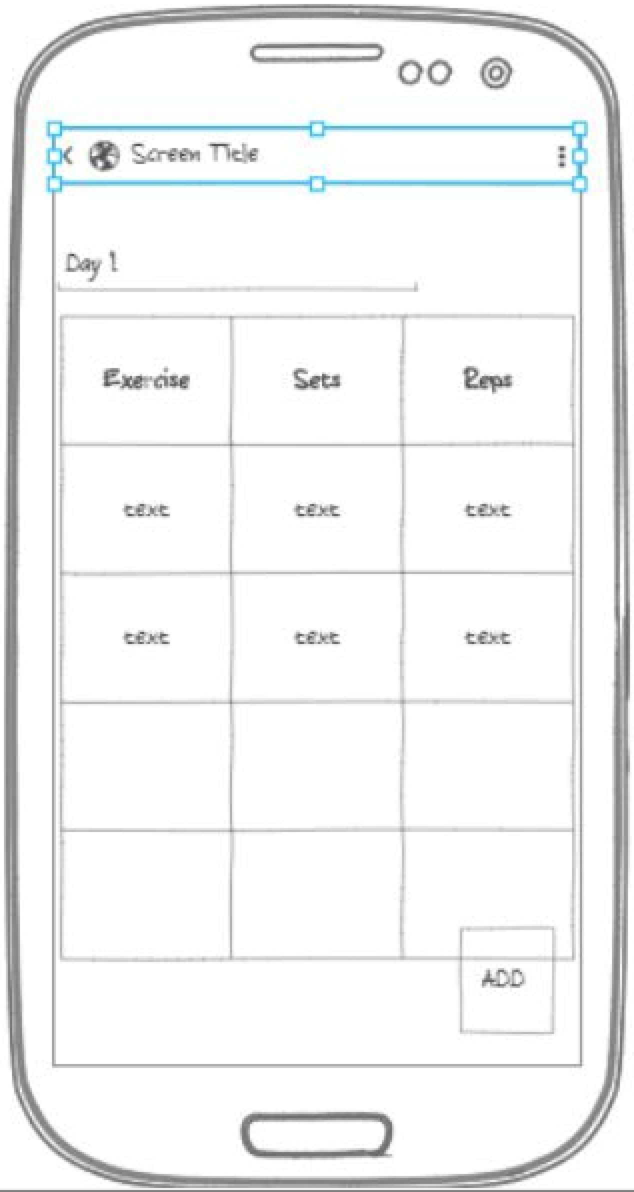
Screen 1



Screen 2

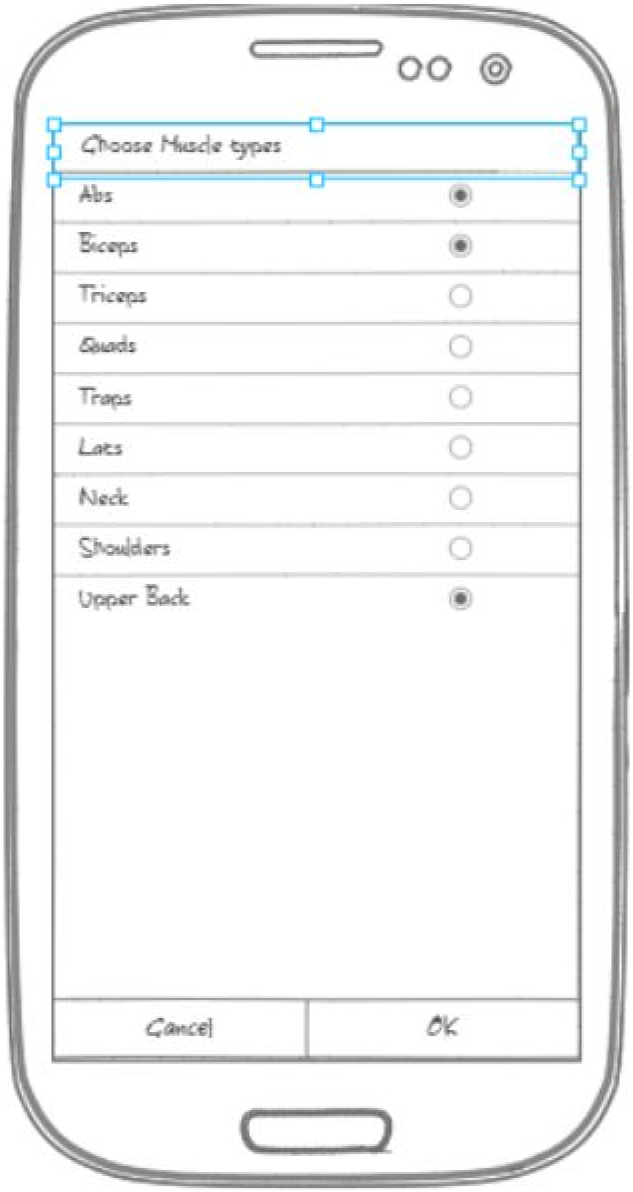


Screen 3



Exercise	Sets	Reps
text	text	text
text	text	text

Screen 4



Choose Muscle types

Abs ●

Biceps ●

Triceps ○

Quads ○

Traps ○

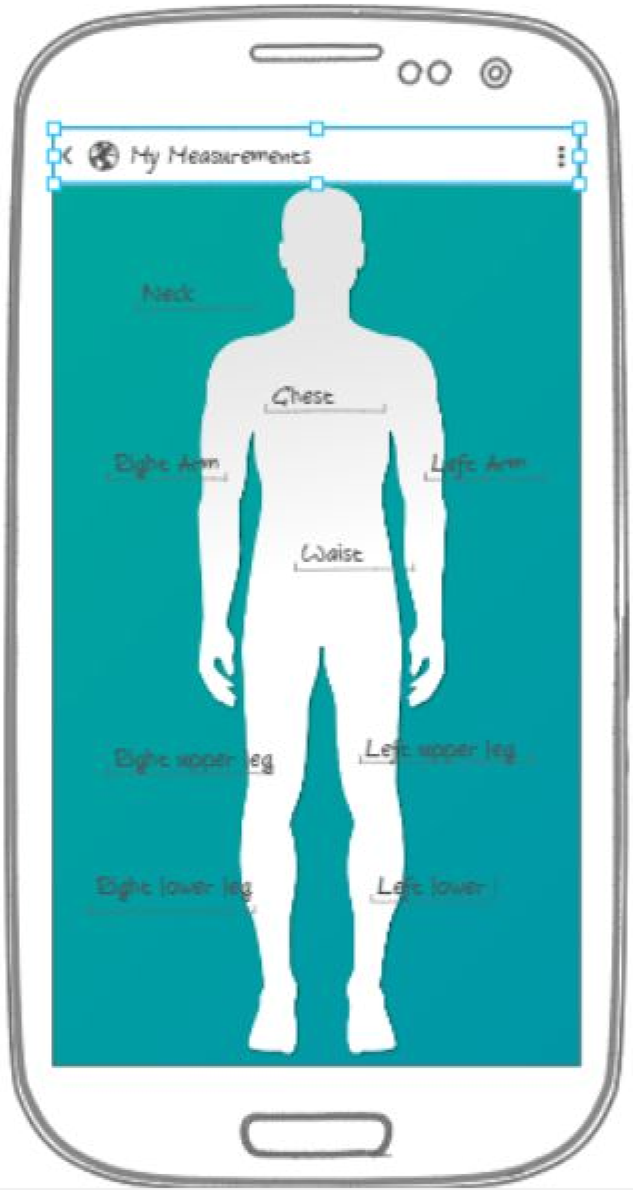
Lats ○

Neck ○

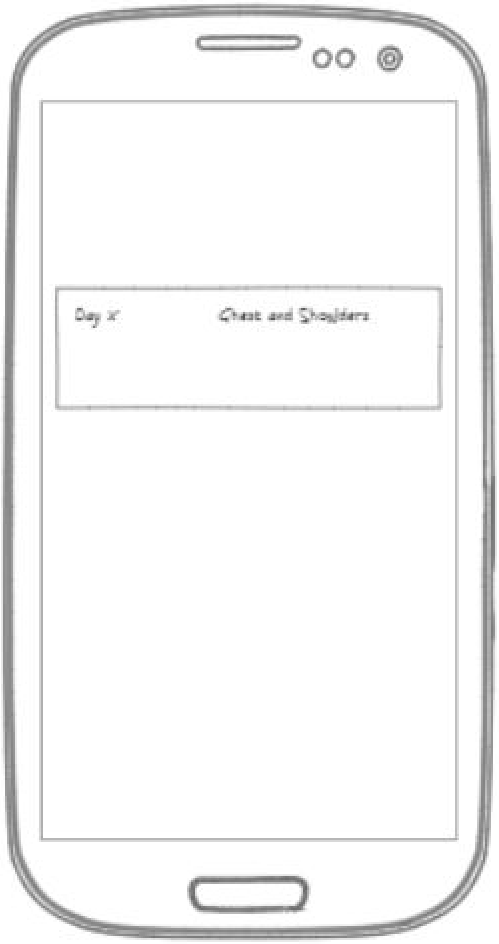
Shoulders ○

Upper Back ●

Screen 5


My Measurements
Neck
Chest
Right Arm
Left Arm
Waist
Right upper leg
Left upper leg
Right lower leg
Left lower

Widget



Key Considerations

How will your app handle data persistence?

- Room DB on the device itself for weekly exercise planning and body measurements.
- SQLite database of exercises to be shipped with app.

Describe any edge or corner cases in the UX.

- Maybe utilise some calendar view for the exercise planning.

- Navigation drawer for the main navigation of the app.

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

- Picasso to handle picture loading. Latest version
- Room to handle on device data latest version
- SQLite to handle app DB. latest version
- Jsoup to parse html crawler/scraped data in the creation of main exercise database. Latest version

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

- Firebase Crashlytics with Fabric for crash and error reporting. Latest version
- Firebase Analytics. Latest version
- Firebase Authentication. Latest version
- Possible usage of Firebase Cloud Storage for image assets in order not to have app too large with image assets. Latest version

Next Steps: Required Tasks

Task 1: Project Setup

Initial Database setup

- Get Data from the web.
- Create Parsing program to parse through scraped web data.
- Save all parsed data to a SQLite database to be added to android app.

Create android studio project for the app.

- Configure libraries and dependencies
- Write models and utilities classes to get data from database
- Write Calculator class that would handle body calculations.

Task 2: Implement UI for Each Activity and Fragment

List the subtasks. For example:

- Build UI for MainActivity that contains navigation drawer and possibly a calendar type view.
- Build UI for Detail day view.

- Build UI for Detail exercise view.
- Build UI for Add exercise view
- Build UI for adding body measurements
- Build UI for tracking body transformation

Task 3: Create Image Assets

- Create image assets and vectors
- Fine tune app style and colors

Task 4: Create Image Assets

- Create Widget

Task 5: Testing

- Add Firebase Crashlytics to app.
- Test App UI and functionality to ensure everything is working

Task 6: Deployment

- Create a signed APK
- Deploy to google play
- Check user feedback and improve app

App design specification demonstrates implementing all features required for Project 7: Capstone, Stage 2 - Build.

- Gradle:3.2.0
- pull or send data to/from a web service or API only once, or on a per request basis (such as a search application), app uses an IntentService to do so
- App keeps all strings in a strings.xml file and enables RTL layout switching on all layouts.
- app includes support for accessibility.
- Java language will be used for development