Honeycomb: Android App Of Goal and Task Management for Increased Productivity

Arthur Knegtel

Final Degree Work
Bachelor's Degree in
Video Game Design and Development
Universitat Jaume I

July 5, 2020

Supervised by: Diego José Díaz García

Acknowledgments

First of all, I would like to thank my Final Degree Work supervisor, Diego José Díaz García, for his guidance and help in the creation of this project.

I also would like to thank Sergio Barrachina Mir and José Vte. Martí Avilés for their inspiring LaTeX template for writing the Final Degree Work report, which I have used as a starting point in writing this report.

Abstract

Too much time is wasted blindly following the river of life with no clear direction in sight.

This project is made for those who always seek to have more than what they have. This is my first opportunity to create something truly helpful for many people that are struggling to find the motivation to follow though their commitments and achieve greater heights.

This app, Honeycomb, helps put the control of the user's life in its hands by allowing him or her to create a massive action plan for the future and committing to it through constant action and planning.

Honeycomb is developed by me, Arthur Knegtel, and this document represents my Final Degree Work on the Bachelor's Degree in Video Game Design and Development.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

Contents

In this chapter I will clearly reflect what is going to be done during the development of the work. I will state the objectives of the work, comment on the need, idea, etc., that motivates it, and the state from which it was started.

1.1 Work Motivation

The motivation to choose this as the final project comes from me being obsessive about productivity, human capacity, motivation and self-development. I think that you can accomplish anything you really want if you just know what you want, create a plan to achieve it and then work for it.

A pilot flying without a course will just be flying in circles. The need for concrete direction is a reality, and having the capability to develop an app that can help with taking a conscious course in life means it's almost a must in my mind to develop it.

1.2 Objectives

Honeycomb has several objectives that have the end user in mind.

- Create a useful app to increase goal success probabilities, peace of mind and happiness.
- Create an app that differs from those already in the market of self-development.
- Implement gamification techniques to make goal setting a rewarding experience.
- Make a modern and simple user interface, with animations and themes to enhance the user's experience in making a goal list.
- Implement statistics to be able to see progress in every goal.

1.3 Environment and Initial State

Honeycomb is developed be me, Arthur Knegtel, through consistent, daily work for approximately 6 months including every stage of the project, from creative process to testing. Development is the most complicated stage taking more than 4 months to complete.

This project is started from a blank slate with the recommended by Google, Android app architecture[1] and best practices[2] in mind.

1.4 Keywords

- Android
- Mobile Application
- Productivity
- Management
- Goals and tasks

[1] *Google Recommended App Architecture*, https://developer.android.com/jetpack/docs/guide#recommended-app-arch.

[2] *Google Recommended Best Practices*, https://developer.android.com/jetpack/docs/guide#best-practices.

1.5 Related Subjects

The following is a list of related subjects to this final project:

- VJ1208 - Programming II
- VJ1215 - Algorithms and Data Structures
- VJ1224 - Software Engineering
- VJ1229 - Mobile Device Applications
- VJ1235 - Entrepreneurship

1.5.1 VJ1208 - Programming II

Programming II is relevant in this project because of object oriented programming which is taught in Programming II. It's the subject that most resembles programming with Java in Android Studio and certainly relevant to this project.

1.5.2 VJ1215 - Algorithms and Data Structures

Data structures are continually used in programming and to use them effectively requires studying this subject.

Although most of the data structures that are used in Android apps are already implemented, having the knowledge of how they work behind the scenes can certainly improve the efficiency of your code. Crucial in knowing which structure to use in what situation.

1.5.3 VJ1224 - Software Engineering

Software engineering subject included learning about flowcharts and various types of diagrams which can all be used in the early stages of a software project. Specifically useful in the phases of planning and designing of a mobile app.

1.5.4 VJ1229 - Mobile Device Applications

For obvious reasons, this subject is the most related to this project. In this subject we learned many of the techniques and practices that are put to good use here.

1.5.5 VJ1235 - Entrepreneurship

The qualities a person needs to be an entrepreneur are no doubt going to be useful when making all kinds of projects. Single team software projects are no different and therefore the subject of Entrepreneurship is closely related to this work.

1.6 Tools

The following is a list of tools that are used for the development from A (start) to B (finish) of the project.

It can be appreciated that the creation of Honeycomb relies on highly professional software suites such as the Microsoft Office collection or the Adobe Apps Suite. These suites are used in workplaces around the world and using them efficiently to create projects is a good-to-have skill.

1.6.1 Microsoft Word

Word is used to make the required documents, like the technical proposal, the analysis and design document and this final report.

1.6.2 Overleaf: LaTeX Editor

LaTeX is used to format this document as a professional book-like report. Overleaf is a free online editor of the LaTeX language.

1.6.3 Microsoft PowerPoint

In order to keep the professional line of the project, PowerPoint is used to create a high-end presentation of this final work.

1.6.4 Diagrams.net

Diagrams.net is a free online diagram software for making flowcharts, process diagrams, org charts, UML, ER and network diagrams.

In this project it is used to create diagrams such as the Use Case Diagram or the Activity Diagram. These diagrams are especially important as they help define the bounds of the project in the initial uncertainty mess that is starting a project from scratch.

1.6.5 Adobe Photoshop

Adobe Photoshop is a widely know tool to create graphics, edit pictures, and design mock-ups among many other things. In this project it is used to design initial concepts of the app and wireframes along with Adobe XD.

1.6.6 Adobe Illustrator

Illustrator's capability with scalable graphics creation enables it to be used to design logos and vector graphics perfect for implementation in density independent environments such as the Android OS. Creating SVGs and implementing them in Android Studio with XML language gives the perfect end result for all screens independent of size, density or resolution.

1.6.7 Adobe XD

Adobe XD is a vector-based user experience design tool for web apps and mobile apps, developed and published by Adobe Inc. It is available for macOS and Windows, although there are versions for iOS and Android to help preview the result of work directly on mobile devices.

In this project Adobe XD is used to create the final design for the entire app and to prototype it before commencing production. Every graphic created with Photoshop and Illustrator was eventually incorporated into XD to create a full, complete app design to base production on.

1.6.8 Android Studio

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems.

Whilst the aforementioned tools all refer to planning, structuring and designing the app from top to bottom, the tool to actually make it all work is Android Studio. It's the oven where you bake the cake with the ingredients you created previously. It's also the app in which you will pass 90% of the time as an Android developer.

1.6.9 Browser and Google

Obviously a tool that is used for almost everything we do in life is also used for the creation of an Android app. Having Google on your side to develop an app is crucial.

Throughout the development you will need to search for code errors on Stackoverflow, design inspiration on Dribbble, Android design guidelines on Material.io, Android documentation on Developers and many other sites that aid you frequently in overcoming the constant challenges that come with the job.

Chapter 2

Concept & Information

Contents

This chapter presents the concept and overall information of the project as stated at the start of development in previous documents, but updated here with the changes undergone through maturation. We will start with an overview of the idea and continue with the core "modules" that the app is composed of.

Note All the actions a user can do in each of these modules will be presented in chapter 4 when we analyze the app requirements.

2.1 Overview

The focus of the app is, as has been stated previously, goal management and planning with milestones and tasks.

The app targets specifically grand, long term goals that can be subdivided into milestones which can be further subdivided into tasks to create manageable chunks of work that, once completed, generate visible progress towards the desired goal.

Using the divide and conquer strategy to break down goals and allowing the user to place their focus on a small piece without losing the general view has incredible

potential of increasing productivity and goal completion rates. Therefore this app, if used appropriately, can be the catalyst to huge achievement.

2.2 Core

The core module is what I define as the bone marrow of Honeycomb. Core includes the personal goal list or goal dashboard, goal creation and goal viewing. It is the largest module and contains most of the logic of the app.

2.2.1 Goals

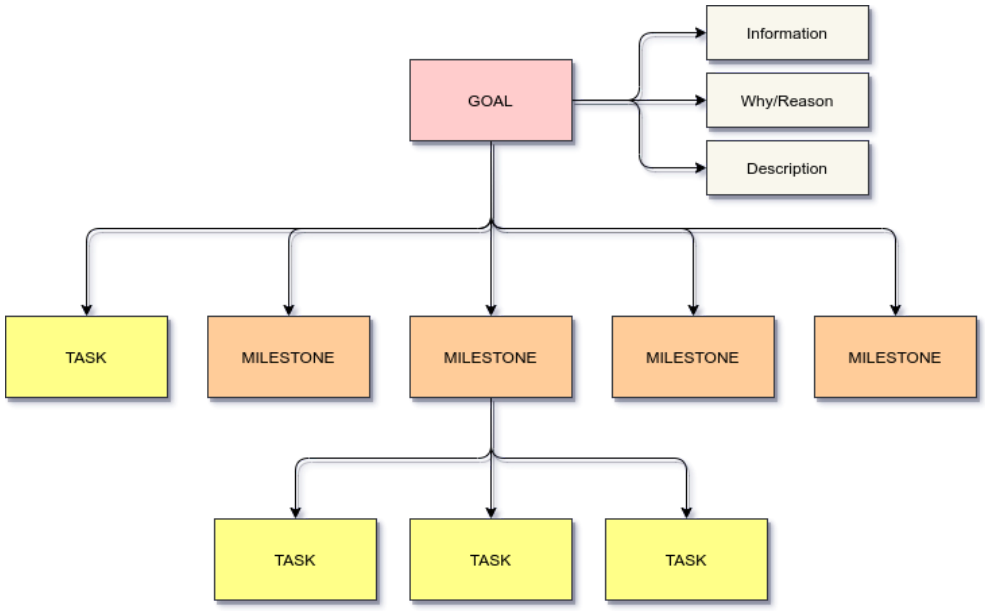


Figure 2.1: Honeycomb Goal Structure

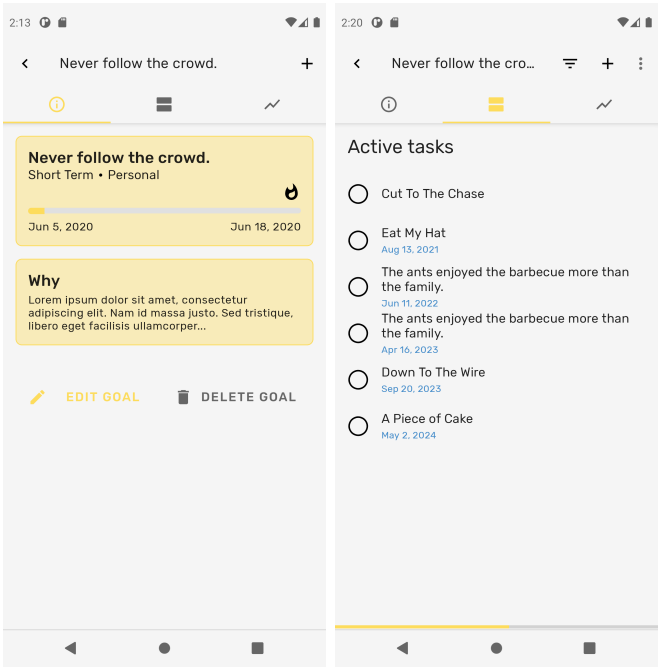
Figure 2.1 represents the goal concept. A goal is an objective to arrive at and complete successfully due to constant work towards its completion. A goal has information important to the user and is made of tasks and milestones bundled into it.

Every goal needs a desired completion date, preferably realistic to set the user's mind focused in its achievement before this time. This date also serves the purpose as being the maximum end date for milestones and tasks that the goal contains.

Figure 2.2 shows the representation of a goal in Honeycomb. It's composed of different tabs that provide all the necessary information and interactivity to the user.

In the first tab it provides information to answer important questions like what, when and why. This section is expandable to include every piece of information the user wants to add, but the previously mentioned are compulsory to include.

The second tab includes information of the how.



(a) Goal Information Tab (b) Goal Action Plan Tab

Figure 2.2: Honeycomb Goal

2.2.2 Milestones

Milestones have double possible purpose, they can act as a type of sub-goal, serve the purpose of sprints or both types can be used simultaneously... The decision lies in the user's hands.

In Agile product development, a sprint is a set period of time during which specific work has to be completed and made ready for review. Sprints are rarely applied to goal management and can be beneficial.

As the definition states, a sprint is a fixed recurring period of time in which you have to complete predefined work. If you apply this to goals, you can set milestones for completion in a week, two weeks or any amount of time you want, define a set of tasks inside them and strive to complete them in the specified time. Then simply repeat the process as many times as you have to and enjoy the rewards as progress is gradually revealed.

All this, and more, is possible in Honeycomb. In figure 2.3, you can appreciate the implementation of the details discussed previously.

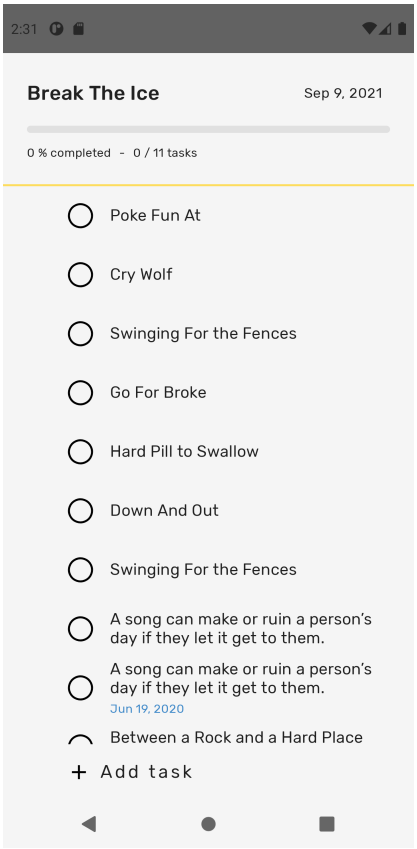


Figure 2.3: Honeycomb Expanded Milestone

2.2.3 Tasks

Tasks are more straightforward, there are no hidden angles, you define a task and add it to a goal directly or to a milestone and then when the time comes you complete it.

They are the lowest level components of the basic structure and simply serve the purpose a normal task has.

If a task is part of a milestone and that milestone gets completed, then all the tasks inside it are marked completed as well.

2.3 Statistics

Each class/object in Honeycomb goals saves certain information in the database in order to provide it later, processed and wrapped in a nice interface in the **goal statistic view**.

This implies that each goal has a statistics view unique to it that contains information about the tasks, milestones, dates, etc. in the form of pleasantly presented values and graphs.

Statistics include, but are not limited to, the following:

- Amount of milestones completed.
- Amount of tasks completed.
- Dates of creation and completion of milestones and tasks.
- Project start and end.

This information is presented to the user for nothing other than visual confirmation of progress. Every time a task or milestone is completed, statistics will update and eventually look better, aiding in motivating the user to follow through and keep pushing until all is done.

2.4 Recommendations

Another one of the added modules is **recommendations**. This section's main purpose is to assist the user in making better goals granting better options at success.

Information presented in this module will be based on articles, videos and books written by experts in the field and will be updated regularly. Following these recommendations will also grant the benefit of using Honeycomb at its maximum potential and the way it's designed to be used.

Based on the previous statement, recommendations can be thought of as a *supporting* "tutorial" section. *Supporting* because Honeycomb's core module already implements a tutorial of its usage. This section merely adds information for the user to use the app more effectively.

For example, for goals to be effective they should follow the S.M.A.R.T. system. This system means that the goal must be:

- **S**pecific *(simple, sensible, significant)*
- **M**easurable *(meaningful, motivating)*
- **A**chievable *(agreed, attainable)*
- **R**elevant *(reasonable, realistic and resourced, results-based)*
- **T**ime bound *(time-based, time-limited)*

Honeycomb is designed to make all goals follow the S.M.A.R.T. system by default, but presenting this idea consciously in this module to the user can only provide benefits.

2.5 Gamification

Being the final degree work of video game design and development it seems necessary to include a gamification section or module in the app.

Although included, it is inherently simple. The user has a small secondary activity in the menu where he/she can see his/her gamification level and amount of points of the current level.

There are certain levels a user can achieve for earning GP *(Gamification Points)*. GP are received or removed following the actions inside the app the user takes.

Actions like creating, completing, deleting and restructuring goals, milestones and tasks will earn the user GP which count towards progress of a level. In short any interaction that actually requires the user to think, will earn him/her progress in this section.

However if a user does not take any action inside the app for a set period of time, i.e. has it abandoned, the user will gradually start to lose GP and consequently maybe even start to lose levels.

Gamification, along with the modules of **statistics** and **recommendations** all encourage the user to increase the interaction with Honeycomb.

2.6 Feedback

Feedback is extremely valuable in software development, especially in mobile apps. Feedback provides a clear and easy way for the user to communicate directly from within the app to the developer.

This is useful with many aspects, such as communicating errors/bugs or even communicating appreciation for the work done.

For these reasons alone I find it compulsory to include a feedback section.

CHAPTER 3

Planning and resources evaluation

Contents

3.1 Planning

In this section, detailed time planning of the work, including most of its tasks and subtasks, the dependencies between them and the possible dependencies with other jobs or external events, are shown. This information can be found graphically in Figure 3.1.

This work can be subdivided in 4 big phases which are **conceptualization, design, development and testing**. Every one of these phases focuses on some area of the project development.

Obviously, the tasks below aren't all of the tasks I completed, but they are the biggest and most important ones. This is valid for any of the 4 phases. Every phase requires more work than it actually shows, which, at times, can mean hours of research and problem solving in some tasks, but nevertheless its a good overall approximation.

3.1.1 Conceptualization

Conceptualization is the process of specifying what we mean when we use particular terms. During this phase the main focus of the work I was doing was forming the idea and establishing the bounds of the project I wanted to create.

Conceptual Phase	**38 h.**
Goal Research	20 h.
Characteristic Definition	12 h.
Use case diagram	4 h.
Activity diagram	2 h.

Table 3.1: Conceptual phase planning

3.1.2 Design

The design phase includes everything that means getting the newly forged idea that was molded in the conceptualization phase and actually putting and image to all of it. The design phase end result is a prototype of the app.

Design Phase	**40 h.**
Writing of document of analysis and design	4 h.
Wireframing	10 h.
UI & UX Design	26 h.
UML Diagram	6 h.

Table 3.2: Design phase planning

3.1.3 Development

The development phase is the biggest part of this project. Conceptualizing and designing just creates an idea and gives it a face, but to actually make it real you have to develop it and provide it functionality.

As I've set at the start of this chapter, many, many tasks are contained inside the following ones, but stating them all would make this chapter too long and tedious.

Development Phase	**230 h.**
Creation of the project	2 h.
Creation of Android Room Database	4 h.
Development of Fake Data Generator	6 h.
Set up all necessary design files	4 h.
Create and set up Dark theme	4 h.
Create UI w/ XML	***184*** h.
Dashboard layout	4 h.
Settings layout	2 h.
About layout	4 h.
Feedback layout	2 h.
Menu layout	4 h.
Add Goal Activity	***16*** h.
Add Goal Fragment layout 1	4 h.

Development Phase	**230 h.**
Add Goal Fragment layout 2	6 h.
Add Goal Fragment layout 3	6 h.
View Goal Activity	6 h.
Statistics goal layout	10 h.
Edit Goal Activity	6 h.
Resources layout	4 h.
Gamification layout	6 h.
Development of Dashboard	10 h.
Development of Menu	2 h.
Development of Settings Activity	2 h.
Development of Feedback Activity	2 h.
Development of About Activity	4 h.
Development of Create Activity	***24*** h.
Development logic fragment 1	6 h.
Development logic fragment 2	10 h.
Development logic fragment 3	8 h.
Development of Edit Activity	4 h.
Development of View Activity	***94*** h.
Development of information fragment	6 h.
Development of Tasks and Milestones in View	42 h.
Development of statistic view	26 h.
Development of tutorial views	6 h.
Development of Resources Activity	18 h.
Gamification Activity	16 h.

Table 3.3: Development phase planning

3.1.4 Testing

Users interact with apps on a variety of levels, from pressing a button to downloading information onto their device. Accordingly, we developers must test a variety of use cases and interactions as we iteratively develop our apps. Making sure everything works as intended even as we add features.

Testing Phase	**30 h.**
Writing Unit Tests	10 h.
Writing Integration Tests	15 h.
Implementing Robolectric & Using Firebase Testing	5 h.

Table 3.4: Testing phase planning

3.2 Resource Evaluation

The most significant factor in estimating the total cost of this project is time spent. The total amount of hours, as approximated in the previous section exceeds 338 hours.

Conceptual Phase	38 h.
Design Phase	40 h.
Development Phase	230 h.
Testing Phase	30 h.
Total	**338 h.**

Table 3.5: Table with total project time cost

Apart from time spent, costs are minimal, there's only a need of a beefy PC to reduce compile times and development time, a PC of this kind will cost somewhere between 1500 and 2000€. However, time spent is a factor to take into account. Let's say that a programmer has an hourly salary of 7€, then the total salary of this programmer for the amount of time spent on this project would be 2366€.

With these assumptions, the total monetary cost of the project would ascend to 3866€.

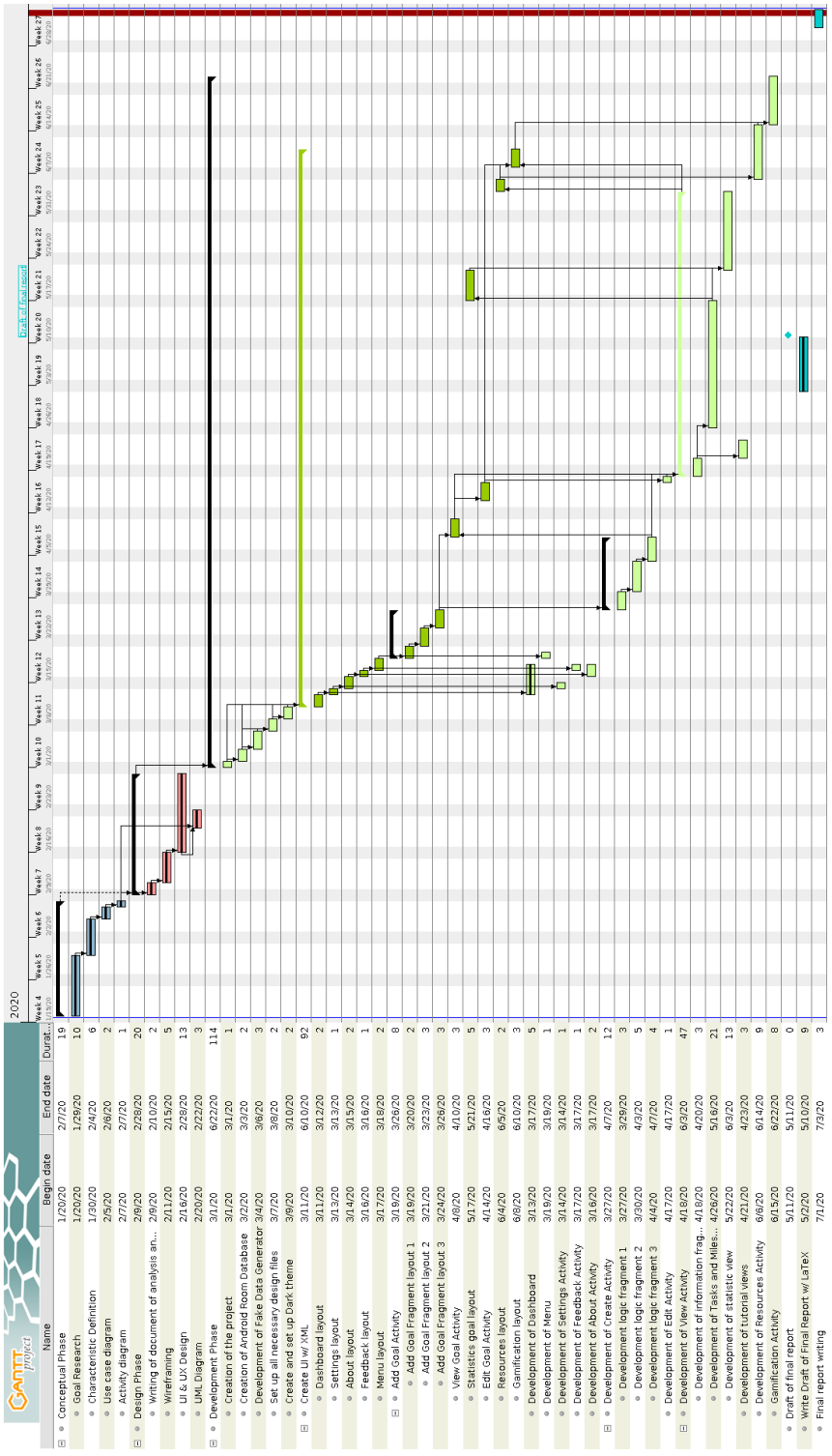


Figure 3.1: Gantt Chart of tasks related with Honeycomb app

Chapter 4

System Analysis and Design

Contents

This chapter presents the system design and architecture of the proposed work, as well as, its interface design.

4.1 System Design

This section presents the logical design of the system to be carried out. This is illustrated by means of the following types of diagrams:

- Case use diagram (See Figure 4.1).

4.2 System Architecture

Honeycomb is made for the Android OS and is available for all devices with Android 7.0 (API level 24) or above through the Google Play Store. This and having a bare minimum of 10 MB of free storage are the only requirements that devices have to meet in order to be able to install and use it.

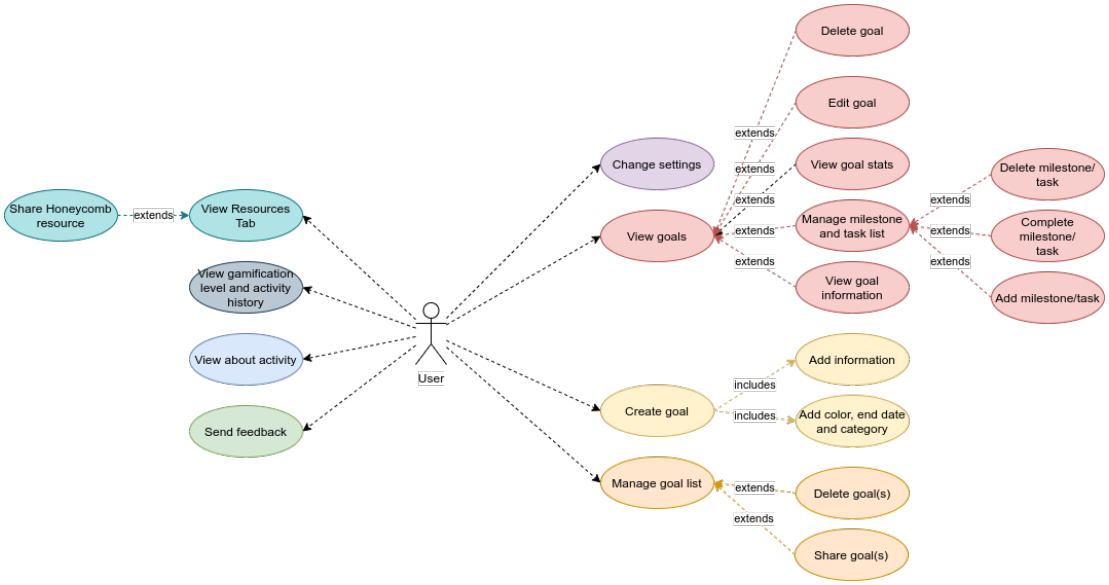


Figure 4.1: Case use diagram of Honeycomb app

4.3 Interface Design

Let's talk about the design, the process and the choices. Every type of design is an iterative process, you most surely never come up with the best result in the first concept or attempt. It's a process of starting with something raw and fine tuning it until satisfied. Honeycomb's design is no different, it's designed completely by me, Arthur Knegtel, through many iterations and changes.

4.3.1 Design Process

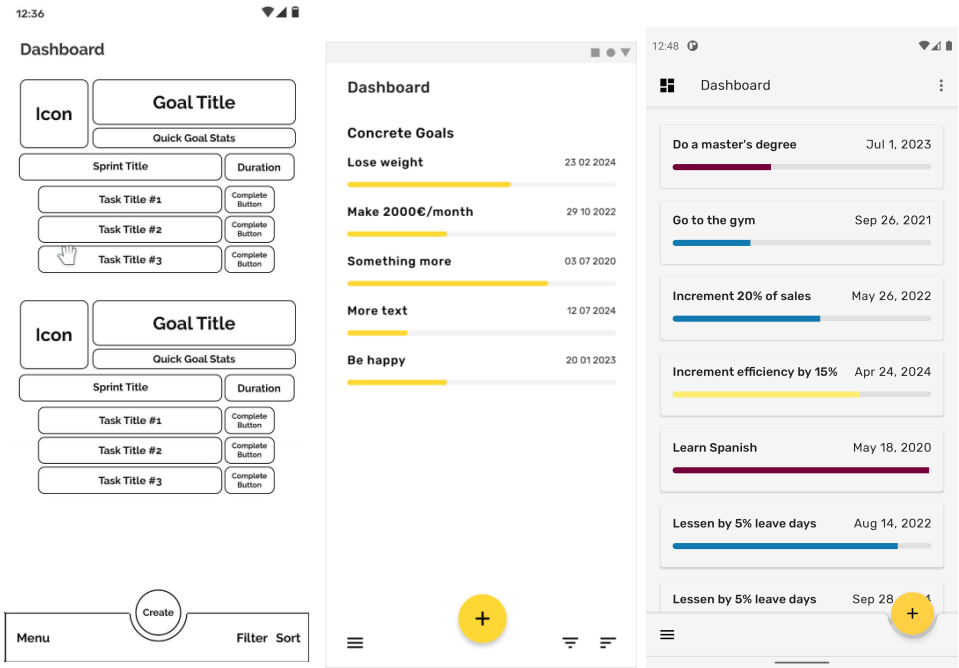
Note For the sake of briefness I will only cover wireframing and prototype design for the app's dashboard, nevertheless I'll cover as much as possible of the final app's working interface.

The first step in designing an app after knowing what you want to create is wireframing. This means placing simple boxes in order to find a structure you're comfortable with. The initial wireframes for Honeycomb's dashboard (main) screen can be seen in Figure 4.2a.

After knowing the general layout you want to go for in most of the screens, the time comes to iterate through multiple prototypes and come up with a fairly "final" UI design.

Figure 4.2b states the result of the final prototype design of Honeycomb app. To see the full Honeycomb prototype design visit this link.

The next step then is to implement the design into the app. This step always comes with subtle differences attached because of the way the implementation works. The implementation of honeycomb's dashboard can be seen in Figure 4.2c.



(a) Dashboard Wireframes (b) Dashboard Design (c) End Dashboard Design

Figure 4.2: Contrast between wireframe, prototype design and final implementation

4.3.2 Design Choices

Guidelines

The design for the app is inspired by Google's Material Design specifications and it's implemented by customizing the Material variants of Android's View Components.

Structure

Early in the development, design choices were made and implemented in one file and from there on every component inherited it's look from that file, using specified widths, spacing, colors, etc. This enables making even a small change in design to propagate everywhere, thus guaranteeing and maintaining consistency across the design.

Maintenance is a large portion of development. As explained in the previous section, "centralizing" the design in a small and manageable group of files is as important as keeping the code clean, simple and organized. Figure 4.3 shows this file subdivision and organization.

The theme.xml file contains all the custom design choices of the components that make up the UI of Honeycomb. The theme.xml file and its variants are the central design configuration files and are complemented by all the other files like colors.xml, shape.xml, dimens.xml, etc.

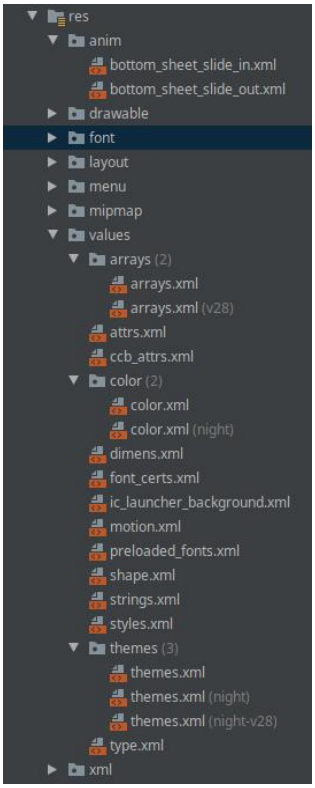


Figure 4.3: Honeycomb Resources File Structure

Theme

Every design choice made along the path of the work aims to get the resulting theme of having a **modern and simple, yet elegant look and feel**.

Colors

The primary color for Honeycomb is yellow. This was a conscious choice made early on in the project since every color has an underlying meaning and psychology.

The color yellow has many different meanings, but mostly they are personal power and fulfilment, abundance, courage and self-confidence. The color Yellow has stood for wisdom and intellect throughout the ages. It aids logic, memory, concentration, will - power and communication. Yellow is full of creative and intellectual energy... Yellow represents happiness, clarity and sunlight.

Given all these meanings it's obvious to choose it for such a project. However, like anything in this world if it's overused, it can have a disturbing effect. That's the reason why every goal you create in the app has a separate user-selected color value.

Light and Dark

Since version 10, Android has embraced Dark Themes. As per the guidelines in material.io:

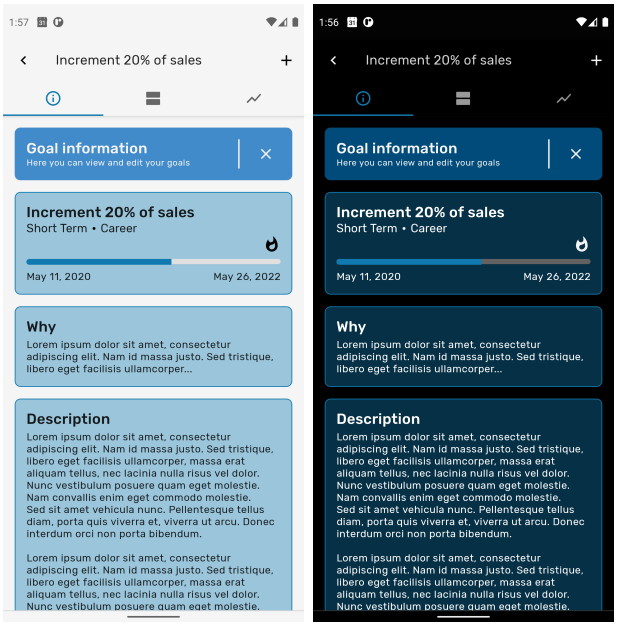
"A dark theme displays dark surfaces across the majority of a UI. It's designed to be a supplemental mode to a default (or light) theme."

Dark themes reduce the luminance emitted by device screens, while still meeting minimum color contrast ratios. They help improve visual ergonomics by reducing eye strain, adjusting brightness to current lighting conditions, and facilitating screen use in dark environments – all while conserving battery power. Devices with OLED screens benefit from the ability to turn off black pixels at any time of day.

So in Honeycomb there is a light theme and an accompanying dark theme and can be switched whenever you like. There is also the option to let the system decide which theme to set. If this option is selected the dark theme automatically activates when the system is in "saving battery" mode.

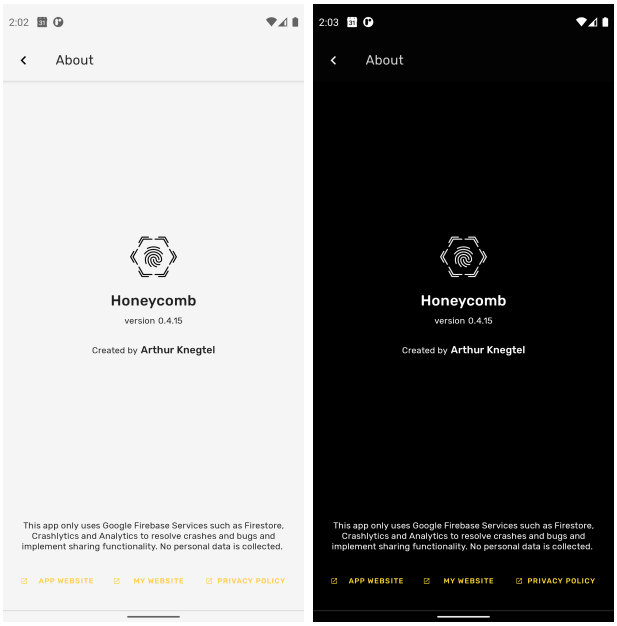
UI Examples

Let's see some examples of the design in both themes to see the contrast.



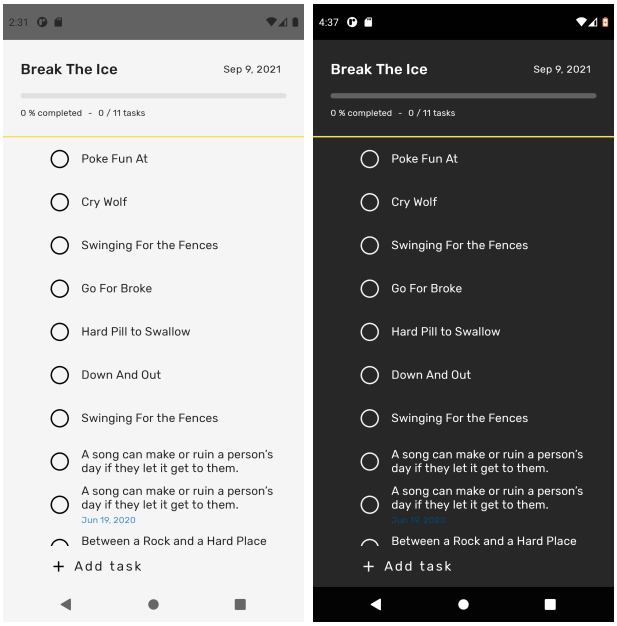
(a) Goal Information Light (b) Goal Information Dark

Figure 4.4: Goal Information in both themes



(a) About Honeycomb Light (b) About Honeycomb Dark

Figure 4.5: About Honeycomb in both themes



(a) Honeycomb Milestone View Light (b) Honeycomb Milestone View Dark

Figure 4.6: Honeycomb Milestone View in both themes

4.4 Logo Design

Figure 4.7: Honeycomb Logo Design

Honeycomb's logotype is designed fully by me. It's a hexagon with a print inside and represents the idea that the goals we have make us who we are, they are a part of ourselves.

The hexagon is actually a reference to the honeycomb structure. The honeycomb structure is made by grouping many hexagons together and it represents the goals that all work together. Every hexagon would in fact be a goal. The id-print is the part that ties the goals (hexagon) to the person (print)

4.5 Color

Every person that studies basic color principles knows that there are meanings associated to colors that vary from culture to culture, known as color psychology. Honeycomb's primary color currently (subject to change) is Material Design's Amber 600 color. As the name suggests it's a combination between yellow and orange and it's the perfect color for what they represent.

According to color psychology, yellow orange's vibrant appearance can uplift, inspire boldness and promote feelings of happiness, excitement and enthusiasm. As it is the color of sunshine, it's likely to be associated with warmth and energy. Its likeness to the color of autumn leaves means it could also be linked with the change of seasons.

Frankly, when it comes to goal setting, which in fact is the action of turning the rudder of your life towards the direction you desire, these feelings are a good thing to have.

Also, it looks good.

Chapter 5

Work Development and Results

Contents

In this chapter I will outline the work developed in these last few months and the results obtained. I will also detail and justify any deviations in the project from the initial planning.

We have to keep in mind that at the time of writing, the application development is not finished yet and everything from the file explorer and structure to the design of the app is subject to change.

5.1 Development Workflow

Before talking about the work that has been done in the app, we must first understand the typical development workflow when creating Android apps.

Writing apps is an iterative process, much like designing them, but where in this case you write some code, run it, debug it and repeat the process until you complete an intended behaviour or characteristic.

After months of development, a pattern becomes apparent. The pattern of app development is repeated every screen, characteristic or behaviour you implement. Firstly you have to create the layout with XML code and once you complete this, you have to create the classes to manage this layout and implement the behaviour you need.

Android Studio includes a variety of tools and intelligence to help you work faster, write quality code, design a UI, and create resources for different device types. These tools and intelligence streamline the process and lessen the boilerplate code needed, which is extremely important when creating big apps.

5.2 Chronological Work Development

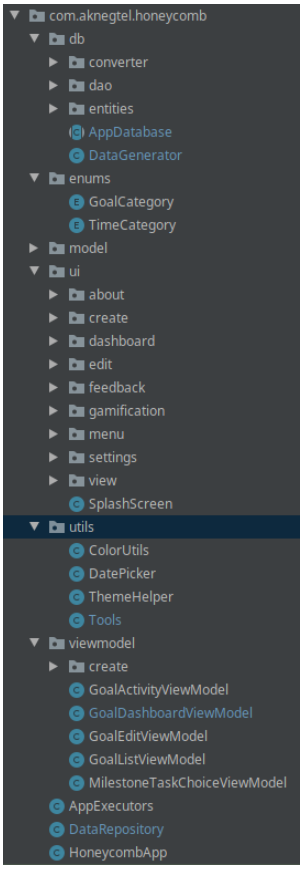


Figure 5.1: Java File Structure

To start off any development, you first have to create the project. Honeycomb was started as black project in Android Studio. After this is done, the next step is to set up the file structure needed in order to implement the recommended app architecture.

The implemented app architecture recommends database files to be sorted into the "db" package, enums in the "enums" package, serializable implementation of classes into the "models" package, any UI characteristic in the "ui" package and the communication between the UI classes and the DB classes are made though classes in the "viewmodel" packages.

5.2.1 Local Database

To start off development, after creating the project and file structure, it's necessary to implement the persistence side of the app to be able to test data later and check if everything you develop later works with the database.

To implement the DB, Android Room was used, creating all the necessary entities (tables), converters (for non-primitive SQL-unfriendly data types) necessary because of Room's ORM design, and the DAOs for every entity that contain every SQLite query for reading and writing the persistent data.

5.2.2 Pseudo-random Fake Data

Based on experience gathered in other apps, it's a good idea to create a class that generates fake pseudo-random data to fill up the database at it's creation, with the press of a button, or based off of other events.

"DataGenerator" is that class in Honeycomb, it creates a goal list when the database is created, generating individual goals and filling them up with data selected from a limited pool of names, set in a period of time and so forth... setting everything up randomly. The class also takes care of generating tasks and milestones at the press of buttons.

This is specially useful when testing newly created screens that work based on data.

5.2.3 Tools

There are functions that are needed on multiple occasions to solve the same problem with slight parameter changes that vary from situation to situation, in order to achieve consistency and reduce code duplication, these functions are grouped together and located in the tools package. Examples of code that is placed in package are error/information/-success messages (toasts and snackbars), methods to gather information of the device, methods to set up a behaviour, etc.

5.2.4 App Architecture

Even though we have talked about the file structure representing the app architecture, we haven't actually talked about what it consists. The recommended app architecture is one that uses view models to connect the database and every UI Activity or Fragment. You can see this represented in figure 5.2.

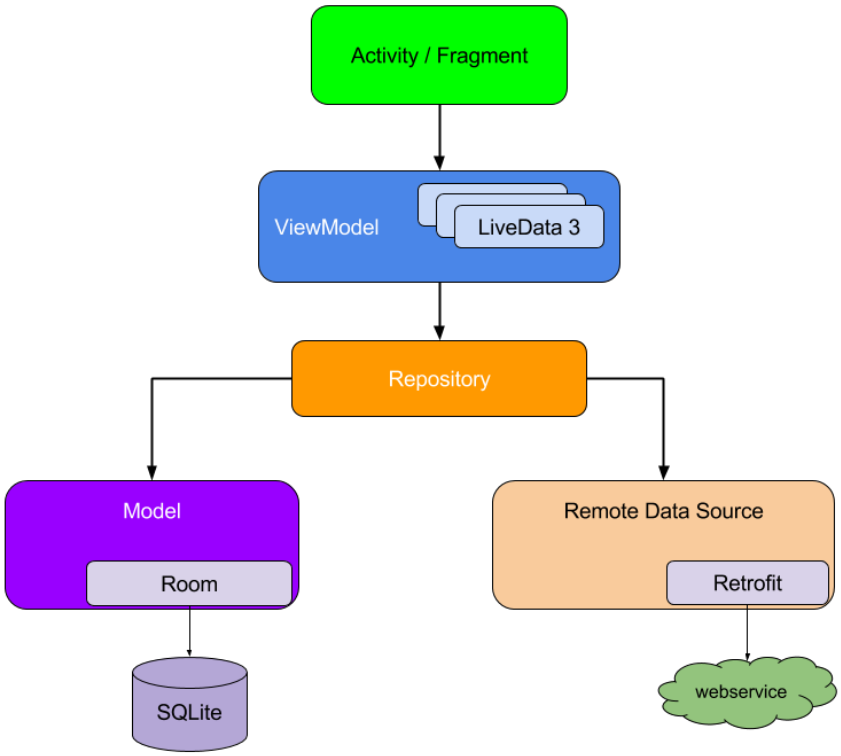


Figure 5.2: Final App Architecture

Notice that each component depends only on the component one level below it. For example, activities and fragments depend only on a view model. The repository is the only class that depends on multiple other classes; in this example, the repository depends on a persistent data model and a remote back-end data source.

This design creates a consistent and pleasant user experience. Regardless of whether the user comes back to the app several minutes after they've last closed it or several days later, they instantly see a user's information that the app persists locally. If this data is stale, the app's repository module starts updating the data in the background.

5.2.5 UI Development

As stated at the beginning of this section, the UI package in the file explorer contains lots of packages on its own, each of these packages implements one UI characteristic. Some activities are simple, while others are more complicated and require more explaining.

About Activity

The "about, feedback, menu and settings" packages are simple examples and contain few classes that complete the behaviour. They simply inflate the XML layout for the UI and set their respective text values.

Create Goal Activity

Creating a goal in Honeycomb is achieved through a three step process that starts off asking information like name, reason and description of the goal. Once this information is provided, an option to advance to the next step is provided, in this second step goal information like category, end date and color are set. In the third phase, the user has the option to provide some milestones in order to avoid starting with a blank slate.

To accomplish this functionality, there is a main activity that controls on which "phase" the user is located on the process and it shows the appropriate fragment.

Once the user saves the goal, they are sent to the main dashboard of the app, where they can see the newly created goal in the list, ready for viewing.

Dashboard Activity

This section is the main activity of the app, it's what the user will see every time they open the app. It simply contains a custom "recycler view" (list) with their goals.

Every "list" in Honeycomb is actually a Recycler View with a custom adapter to show the necessary information every screen and list needs.

Edit Goal Activity

Editing a goal is an action that's available only from within a goal. It takes all the information that's provided when creating the goal and that can be viewed in the information

fragment in the goal view, and it places that in a convenient activity with editable text inputs that allow you to change a goals information values as the name would suggest.

View Goal Activity

A goal view is the most complicated package in the entire app, it could even be reasoned that it contains an entire sub-app of to-do-lists.

The View goal activity is made out of three fragments laid out with a view pager that enables horizontal scroll between fragments.

The information fragment displays the available information of the goal. When optional fields are not included, they are not shown and the layout adapts following this criterion.

The goal dashboard fragment contains a list of all tasks and milestones that pertain to this goal. A user can create, complete, edit or delete tasks and milestones. A user can also open milestones to view associated tasks.

The statistics fragment shows information of the activity that the user has done for the goal, the more it interacts, the more information it shows.

5.3 Results

Although the app is not finished at the time of writing this document, most of the basic work is completed. Following is a list of results, a list of things that have been accomplished in the app. This list includes the initial objectives that were mentioned in Chapter 1 and goes into more detail.

- ***Create a useful app to increase goal success probabilities, peace of mind and happiness***. If put to good use, Honeycomb does exactly that.
- ***Create an app that differs from those already in the market of self-development***. Honeycomb distinguishes itself from the competition by combining different features of competing apps whilst providing a personal touch to these and wrapping them up in a different structure, thus being unique.
- ***Implement gamification techniques to make goal setting a rewarding experience***. The gamification implemented in Honeycomb is certainly simple, but undeniably effective. Having a point system based on actions encourages users to interact more with the app.
- ***Make a modern and simple user interface, with animations and themes to enhance the user's experience in making a goal list***. Honeycomb's final user interface is simple, clean and elegant. It's also available in light and dark theme.
- ***Implement statistics to be able to see progress in every goal***. Honeycomb's goals indeed have a statistic system that keeps track of the actions and progress of said goal.

5.4 Real-Life Applications

As was stated in the introduction of this document, Honeycomb has multiple use cases and can serve multiple applications depending on the need of the user.

The most obvious application is the intended one: to keep track of goals the user may have, especially long term ones where it's easier to lose focus in the grand scheme of things. Nevertheless it's also useful to keep track of short and/or mid term goals.

In the same line but with a little twist, Honeycomb can be used to track projects. They operate on the same basis, you have an end date, tasks and possibly milestones to keep track of.

This said, you can extend this idea and prove it really is useful to keep track of anything. Honeycomb is basically an app that saves a collection of items and for each item in that collection you have an entire to-do list app inside. It's designed for goals, but the type of collection ultimately relies on the user's choice.

5.5 Launch

Honeycomb will be available on Android phones through the Google Play Store on the 12^{th} of July 2020 for free. Also, on the 6^{th} of July the code of the entire project at the state of said date will be publicly available on GitHub for grading.

The launch/promotion plan for the app is to first of all do extended testing with friends and family on different kinds of devices. It's not the first time I upload an app and it works great on my device and then breaks in all possible ways on other devices.

Once the app is confirmed to work as intended, I will design a few banners/posters that I will share through social media, and mainly through Reddit where there are entire communities that could benefit from this app. As these communities are already full of people interested in productivity, motivation and achievement, making an enticing post to gain a solid base of users can be a good idea.

CHAPTER 6

Conclusions and Future Work

Contents

In this chapter, the conclusions of the work, as well as its future extensions are shown.

6.1 Conclusions

To close off this report I'm going to comment on any notable experiences I had during the development of the project. Any opinions are strictly personal and based on the experience gained with this work added to my previous experiences.

First of all, I'd like to state that its been a tough and long but really enjoyable and rewarding experience, a real roller coaster of emotions.

I already had some experience creating Android apps. I've created dozens of mini-apps in the last years trying things out but only developed one from start to "finish" (abandoned) with a colleague last year for the Android Development subject in this degree. Honeycomb is my first big solo project and having to create a full size project from scratch, on my own, from start to finish for the first time has been very insightful.

I've always thought the development stage of a software project was the hardest part of it, but I've learned that the stages of conception and design really are harder than the development itself. The development is the largest part and definitely the most laborious, but it's pretty straightforward if the previous stages are completed thoroughly.

The stages of conception and design require creativeness, inspiration and "flow" which can sometimes be lacking and can cost precious time...

Another experience I want to express is that the whole process of software development is an iterative process. You don't come up with the idea fully baked with details and all. You start with something promising and keep improving it every time you can until you have something worthwhile.

You never see the initial drafts, the prototypes and the process that's behind every app you download. Frequently as a user you jump on an app that has years of development history and changes to get to where it is and you can fall into the trap that is thinking that it was always like that when it surely wasn't.

So, if I've learned anything it's that anyone with enough time, determination and persistence can make something great.

In order to make the most of this project I've read articles, books and watched countless videos on goals, android programming, design, etc. and yet I feel like there's a whole world of information on these topics I've yet to discover. It's like Albert Einstein said: "The more I learn, the more I realize how much I don't know.".

That said, this work has taken my app development skills to a higher level, forcing me to up my game to solve problems I encountered. I used a new architecture with new structures like LiveData, I also learned about data binding, I learned about animations, themes, and so much more.

During the process I've experienced many emotions, some of which are victory, defeat, happiness, sadness, satisfaction, frustration, doubt, etc. But at the end, after following through, all that is left is the experience, the project and a sense of accomplishment.

6.2 Friends and Family Feedback

As suggested by Diego, my supervisor, I shared a semi-final version with a close circle of people for a couple of days and asked them to provide feedback on the app.

The general feeling surrounding Honeycomb is positive. They all say the design is pleasing, the animations are a nice touch and having the possibility to select a night theme is a life-saver. In terms of the usability they also had no problems, few to none bugs reported and happiness all around.

However, there was a recurring pattern of feedback that mentioned the over-simplicity, the lack of extra features. I was already aware/wary of this, Honeycomb is only the foundation of a much bigger project. A project that could not be completed in 4 months by a one-man team.

6.3 Future work

In my honest but undoubtedly biased opinion, this project has potential to be something, therefore, it will be updated and improved at least until Q1 2021 on a consistent basis.

Depending on the user base and numbers as time passes, it's life will be extended or not. In the coming months I'm planning also on adding certain features useful and big enough to be behind a paywall.

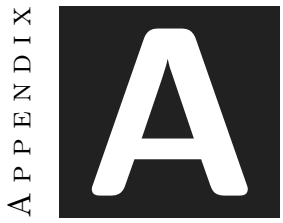
This is a list of things I might want to focus on getting done in the next months:

- Fix bugs.
- Polish the app.
- Add setting to change the main color of the app interface.
- Add more customization.
- Add activity to see a daily list of tasks from all goals or upcoming tasks.
- Add possibility to back up data to cloud.
- Increase functionality of task list. Add searching, for example, or ordering.
- Add ordering and filtering/searching for the main "collection".
- Add goal sharing functionality.
- Making a leader-board for the gamification points.

Although I'm sure as I continue to implement new things and improve what's already in there I will find more things to add and improve (there's basically no end to this) I have to keep in mind the original objectives and keep it clean and simple.

At some point in the future, once it gets messy (which is inevitable, really) I could even make a version 2 of Honeycomb using my experience to make it more solid and future-proof from the start.

The possibilities are endless.



Other Considerations

A.1 Bibliography

I maybe literally visited hundreds of pages to get the necessary information to complete this report and the whole project. Here, the most relevant links are provided.

A.1.1 Books

- Griffiths, Dawn, and David Griffiths. Head First Android Development: a Brain Friendly Guide. Beijing: OReilly Media, 2017.
- Henney, Kevlin. 97 Things Every Programmer Should Know: Collective Wisdom from the Experts. Beijing: OReilly, 2010.
- Marsicano, Kristin, Brian Gardner, Bill Phillips, and Chris Stewart. Android Programming: the Big Nerd Ranch Guide. Atlanta, GA: Big Nerd Ranch, 2019.

A.1.2 Websites

Design

- UpLabs Andorid Design
  `https://uplabs.com/android`
- Coolors, color palette generator
  `https://coolors.co/`
- UnDraw Illustrations
  `https://undraw.co/illustrations`

- Dribbble, Android design inspiration
  https://dribbble.com/search/android
- Guide to app design
  https://99designs.es/blog/web-digital/how-to-design-an-app/

Coding

- Android Documentation
  https://developer.android.com/docs
- Stackoverflow, questions
  https://stackoverflow.com
- Android Dark theme
  https://raywenderlich.com/6488033-android-10-dark-theme-getting-started
- Android ButterKnife Extension
  https://jakewharton.github.io/butterknife/

Goals Information

- Types of goals
  https://developgoodhabits.com/types-of-goals/
- Goals and Sprints
  https://bit.ly/3d5MN2d
- SMART Goals
  https://www.mindtools.com/pages/article/smart-goals.htm
- Objetivos
  https://enciclopediaeconomica.com/objetivos/