Life Scheduler

Jared Bennett

Table of Contents

Introduction

Life Scheduler is an Android App designed to help anyone manage their everyday schedule by taking in *everything* they want to do and generating a schedule that allows them to do *everything* in at a *realistic*, *comfortable* pace. As an app, your schedule is readily available, easily editable, and user friendly. In simple terms, this app aims to help you answer the question "What should I do next?" as best as it can.

The first part of figuring out any schedule is determining exactly what, in detail, must be done to do what you want to do. Afterwards, you need to figure out how to fit everything you must do into your day to day schedule. This application's main goal is to help you solve the latter and help you keep track of the former.

Target Audience

This app can be used by anyone, though the more organized you are the more accurate the generated schedules will be. It is particularly useful for people who have complicated, ever-changing schedules that are hard for them to manage, such as someone with many responsibilities (work, school, family, etc.). This is *not* useful for people who solely have set schedules that they cannot change themselves.

Research

There are many different mobile and desktop scheduling applications available, however I could not find one that covered *everything* you do every day. Most applications are either designed to schedule a certain activity group, such as work tasks or sports events, or just one event for multiple people. No applications were found that could generate your schedule in the same way, or that were intended to decide when to do things based on a set of arbitrary goals. That being said, here are a few related applications that were found to be useful.

Google Calendar (App, Website)

This wonderful application and others like it are used to manage a schedule, but do not create one for you. Google Calendar can generate a few specific schedules with their 'Goals' but they are not as complex as Life Scheduler goals and do not work the same way. It is perfect for viewing a schedule that is entered by the user, and Life Scheduler is not trying to replace it. In fact, I fully intend to integrate events to and from Life Scheduler in order to make it easier to use.

Humanity (Humanity Mobile App, Humanity Website)

Humanity is a service that allows employers to generate shift schedules for their employees. Humanity Mobile is an app that allows employees to access and keep track of their schedules.

JIRA (Website)

JIRA is a fantastic web app used by software development teams to help organize development. It helps developers plan and organize what must be done and tracks the status of projects. There are many things JIRA can do and no other product out their does it better in my opinion. This is not necessarily a scheduling application, but did give me the idea of breaking apart a schedule into goals and tasks, which is similar to its issue management.

The General Idea

After creating an account and signing in, the user provides general **goals** and specific **tasks** or **sub-goals** required to complete these goals. Then, they ask the app to generate a **schedule** based on the provided information. This generated schedule will outline when to work on each task so that each goal will be satisfied. The more specific and inclusive the goals and tasks are, the more useful the schedule will be. Since schedules can be generated on demand, updating you schedule is as simple as updating your goals and tasks and then re-generating it.

Definitions

Goals

Goals are general things you want to do, such as going to work, going to class, sleeping, or eating. These can be abstract as you want, and can include other sub-goals and tasks.

Tasks

Tasks are specific things that you need to do in order to complete a specific goal. For example, a task for a goal of to pass a college class may be to complete a specific homework assignment. The have different priorities and other constrains applied to them that affect schedule generation, such as estimated duration.

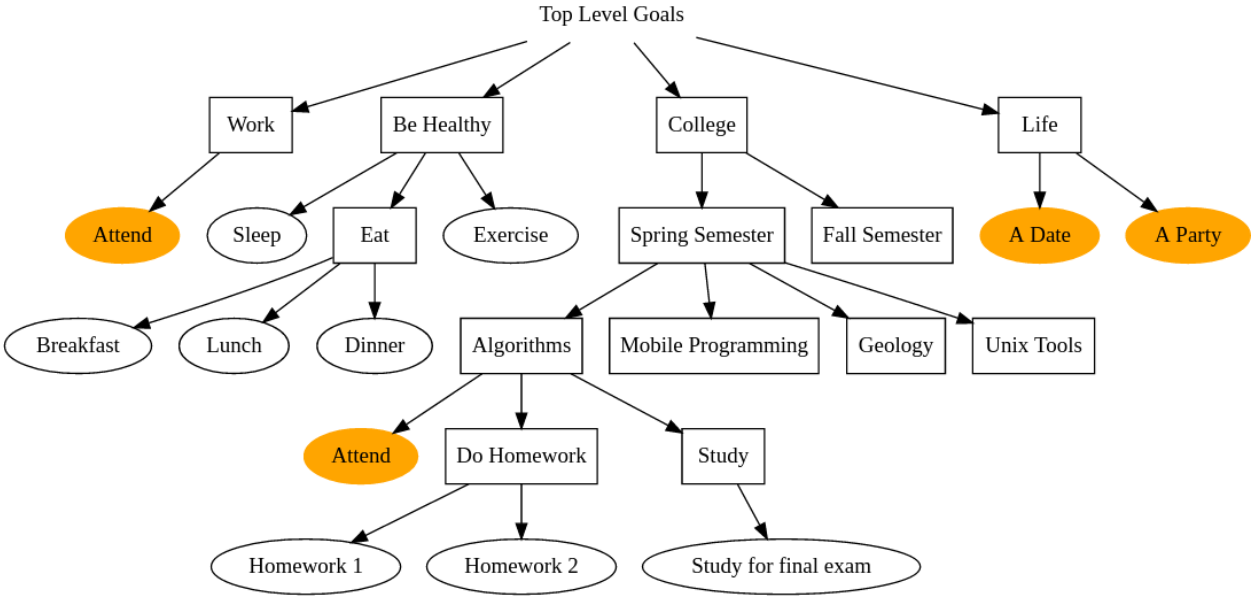
Schedules

Schedules are collections of Time Allotments, which are

A Realistic Example

Say you have a both a job and attend college. As a good employee and student, you want to be on time for work, attend all your classes, and get your all homework done. Both things are very demanding on you, but you also want to have a life and relax while still being able to get enough sleep every night. In addition, you do not always know that far in advance what homework you will have when or what other things you must do. A few things you need to do are beyond your control, such as your work schedule and when your classes occur. However, the rest of your day is up to you to figure out, which is where Life Scheduler becomes useful. By breaking everything apart into goals and tasks, choosing what to do next is as simple as generating a schedule and picking from your options.

Below is a representation of some of the goals and tasks you would submit to the program that would be used in schedule generation. Goals are squares, tasks are ovals, and fixed tasks/events are orange.

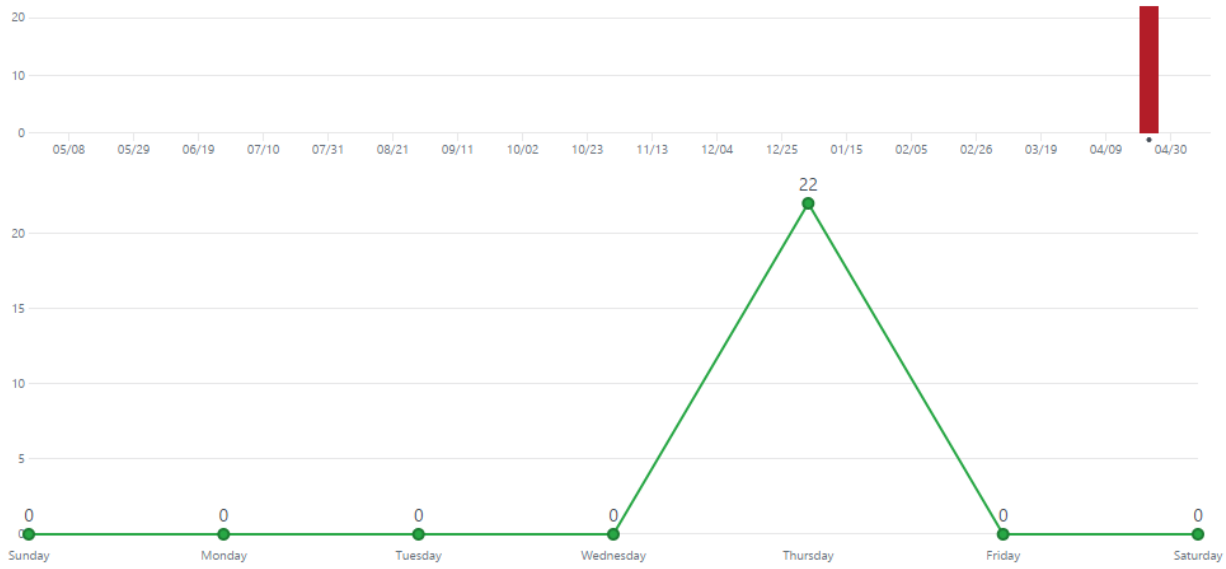


Development Timeline

During most of the week leading up to the presentation I was working on figuring out how the app should work and doing research. Most of the actual development was done the day of and day before the presentation date. I would have spent more time on this but I had other classes' projects to do.

Commit Timeline for App (https://github.com/jrdbnntt/cop4656_life_scheduler_app)

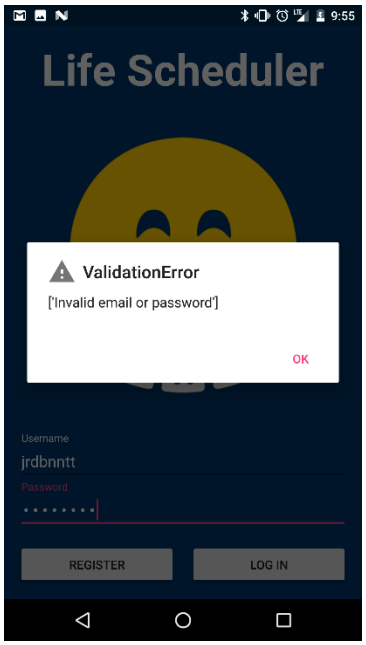
Commit Timeline for Server (https://github.com/jrdbnntt/cop4656_life_scheduler_server)



Implemented Features

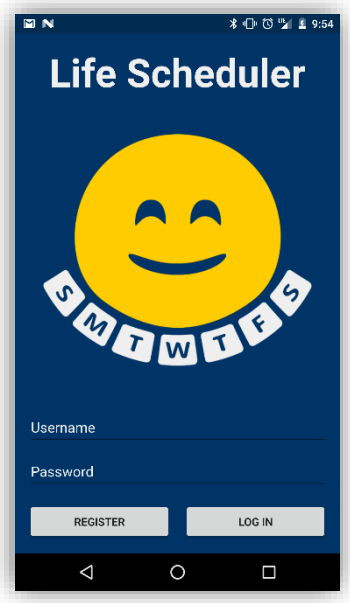
Backend Server & Database

The main logic of the application is performed on a Django Server using a PostgreSQL database. App installations communicate with this server by means of an API via GET/POST HTTPS requests. There is an Android API to abstract out the HTTPS requests made with Volley into a series actions with request and response Gson objects. Cookies are stored in the SharedPreferences in order to maintain user sessions, allowing accounts to work.



To make things simple, request errors can be easily displayed in a dialog using a function of the GsonVolleyApi. This is used app-wide for most things a user submits, as the server handles errors and makes appropriate error messages.

User Accounts: Login (SplashLoginActivity)

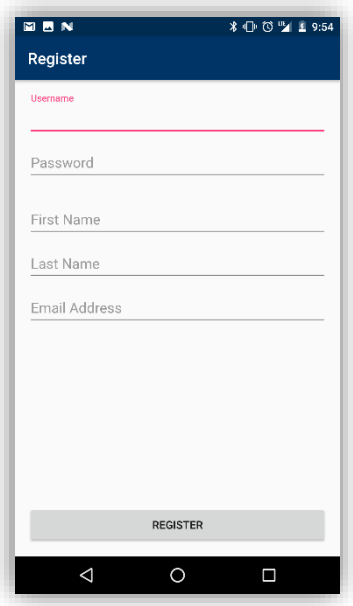


The first screen of the app is this SplashLoginActivity. It is here that the user can enter in the credentials and login, or click register to go to the user registration page. If a user is already logged in, this activity is simply skipped and they are redirected to the MainActivity.

Widgets:

- TextView for the name of the app
- ImageView for the logo
- TextInputLayout for the two text inputs (I like them)
- “Register” Button to go to the RegisterActivity
- “Log In” Button submit the form and go to MainActivity on success

User Accounts: Registration (RegisterActivity)

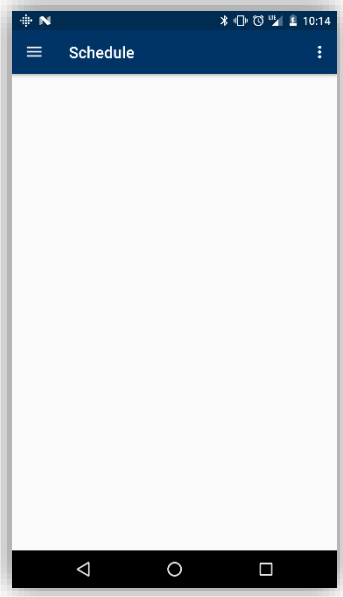


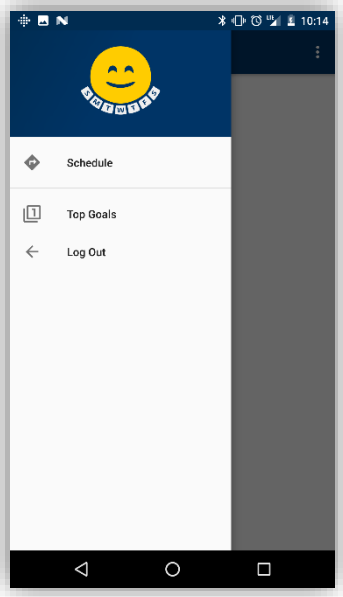
This RegistrationActivity is where a user can register for a new user account with the required information.

Widgets:

- TextInputLayout for all text fields with respective input types (i.e. “Password” field is type password so it is obscured)
- “Register” Button to submit the form and go to MainActivity on success (they are automatically logged in)

Main Navigation Screen & Schedule View (MainActivity)





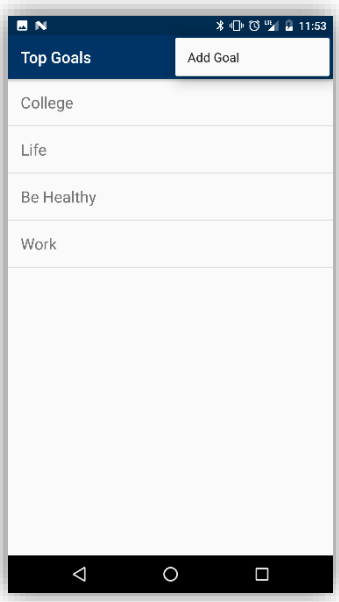
This activity has multiple purposes:

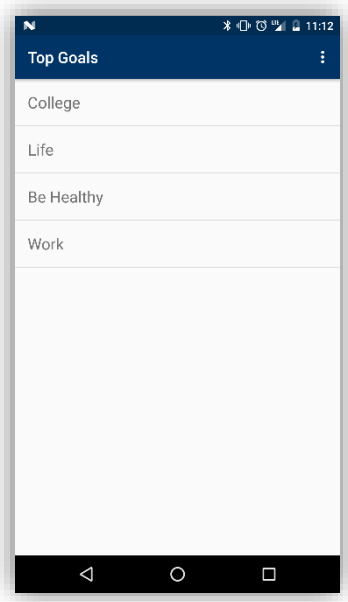
1. Display the current schedule (not implemented yet)
2. Allow the user to navigate to different activities
3. Allow the user to log out

Widgets/Elements:

- NavigationDrawer to contain navigation options:
  - "Schedule" – Just closes the drawer because the schedule is the same activity
  - "Top Goals" – Go to the TopGoalsActivity
  - "Log out" – Logs the user out using the API and then navigates to the SplashLoginActivity

View Top Level Tasks (TopGoalsActivity)



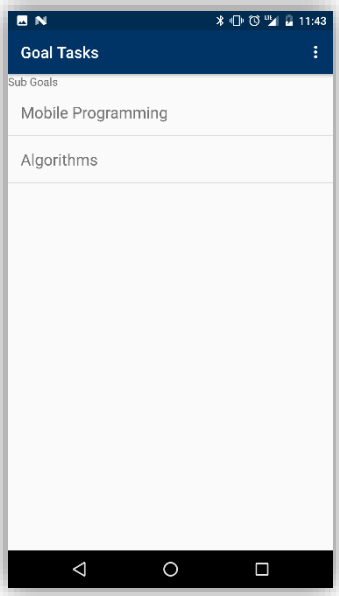


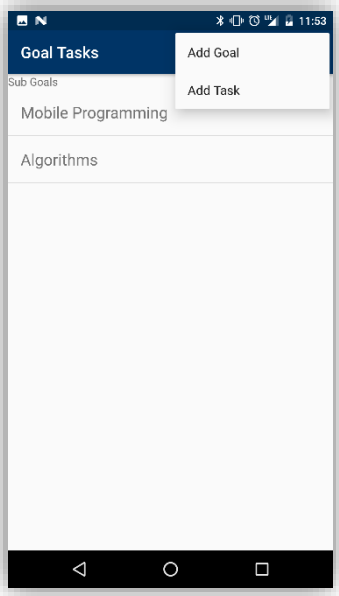
This activity will load the users' top level goals. From here you can click the options menu and create a new goal (Go to CreateGoalActivity with parent id = null) or click on a goal and manage it (Go to GoalsTasksActivity with parent id = that goal's id).

Widgets:

- A ListView containing the names of the top-level goals, with data populated by the API
  - Items contain a TextView with the pulled goal's name.

View A Goal's Sub-Goals and Tasks (GoalsTasksActivity)



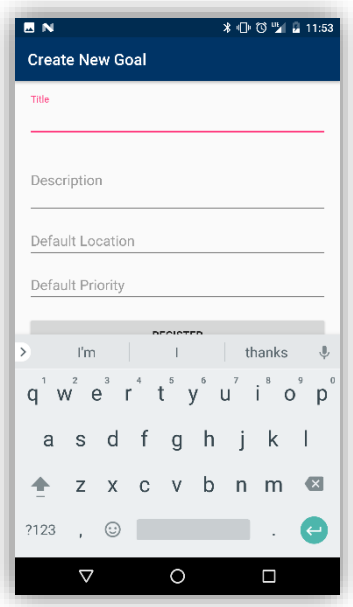


This activity will load a goal's sub-goals and tasks. The goal to display is based on the id received in the intent. From here you can click the options menu and create a new goal (Go to CreateGoalActivity with parent id = current goal's id) or create a new task (CreateTaskActivity with parent id = current goal's id), or click on a goal and manage it (Go to GoalsTasksActivity with parent id = that goal's id). In the future, you would also be able to click on a task and edit it and view/edit the current goal.

Widgets:

- A ListView containing the names of the top-level goals, with data populated by the API
  - Items contain a TextView with the pulled goal's name.
- A ListView (not pictured) containing the names of the top-level goals, with data populated by the API
  - Items contain a TextView with the pulled task's name.

Add a new Goal (sub/top) (CreateGoalActivity)

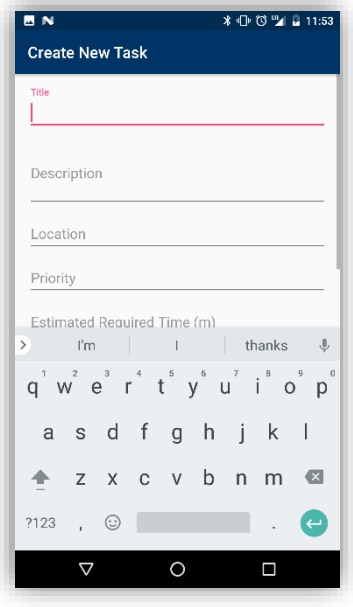


This activity is a form to create a new goal with the provided parent goal id via intent.

Widgets:

- TextInputLayout for all text fields with respective input types
- Button to submit the form with the API. Finishes activity on success, therefore returning to the previous activity.

Add a new Task to a Goal (CreateTaskActivity)



This activity is a form to create a new task with the provided parent goal id via intent.

Widgets:

- TextInputLayout for all text fields with respective input types
- Button to submit the form with the API. Finishes activity on success, therefore returning to the previous activity.

Future Development

There is much to be done with this app. At the moment, most features are outlined but not implemented. I have many features planned for this and fully intend to complete after finals. The main reason it is not complete now is due to time constraints and the fact that this entire project was done by just me. If given more time, time that was not so close to finals, this would be much more developed. I probably made this a bit more complicated than necessary, but I feel like I have done enough work at least for this project as far as this class is concerned. To fully complete this app it would most likely take a month or two, not a week.

- Implement the schedule generation – The algorithm is written on paper only right now and will be complicated to implement and will be performed on the server
- Integrate with Google Calendar (I have done research on how to use this API)
    - Retrieve current calendar events as fixed tasks
    - Manage a custom calendar with the generated schedule
- Schedule generation pages – Will query the API for a schedule
- Edit Goals/Tasks
- More information displayed for the Goal/Task view
- Allowing for a set of arbitrary constraints to be added to a given Task/Goal for schedule generation purposes
- Generated schedule saving + editing

Conclusion

This is (or at least will be) a very useful app. I have not seen anything else available that does the same thing. I am going to continue working on this after finals and polish it into something actually useable. If I find that it works well for myself I will polish it up more and then release it in the play store.