|  |
| --- |
| **Nudgi** |
| Final Project |
| Clifford Ludwig, Timyathus Spikes |
| 4/11/2024 |
| Mr. Buchanan – Server Side Web Programming |
| East Tennessee State University |

A logo with a blue background

Description automatically generated

|  |
| --- |
|  |

Contents

[Phase 1 - Initial Ideas 4](#_Toc163906096)

[GPT Response 4](#_Toc163906097)

[Decision 6](#_Toc163906098)

[Phase 2 – Research and App Outlining 7](#_Toc163906099)

[Results 7](#_Toc163906100)

[Observations 8](#_Toc163906101)

[General Observations 9](#_Toc163906102)

[Phase 3 – App Planning 10](#_Toc163906103)

[Primary Focus 10](#_Toc163906104)

[Challenges 10](#_Toc163906105)

[Diagrams 10](#_Toc163906106)

[Presentation Requirements 11](#_Toc163906107)

[Software and Extensions 11](#_Toc163906108)

[Visual Studio Code 11](#_Toc163906109)

[***.NET MAUI Extension*** 11](#_Toc163906110)

[***GitLens Extension*** 11](#_Toc163906111)

[***C# Extension*** 11](#_Toc163906112)

[***NuGet Package Manager*** 11](#_Toc163906113)

[***Live Share*** 12](#_Toc163906114)

[***Android iOS Emulator*** 12](#_Toc163906115)

[Draw.io 12](#_Toc163906116)

[DataGrip 12](#_Toc163906117)

[Android Studio 12](#_Toc163906118)

[To-Do List 12](#_Toc163906119)

[Meeting Minutes 13](#_Toc163906120)

[Meeting (4/12) 13](#_Toc163906121)

[Signing the Blood Oath 13](#_Toc163906122)

[Meeting (4/13) 13](#_Toc163906123)

[SWOT Analysis 13](#_Toc163906124)

[Shower Thoughts 15](#_Toc163906125)

# Phase 1 - Initial Ideas

For this project I am torn between a couple of Ideas.

The first is an app that helps a user pick the best character to play based on the opposing teams’ characters or even their team. This would be statistically driven based on individuals’ characters. Essentially a "How do I counter this character" app. I have no idea where I would get the data.

The second Idea is a chore tracker and organizer. I have this image in my head of a space that can fit lowers of rectangles.; Each rectangle represents a chore, and the height of the rectangle represents how much time it will take, while the color represents difficulty or maybe category. The space can only hold so many, which allows users to make large to-do lists, and then responsibly pick items from the list to fill the space so that they can be productive on the tasks that fit into the day. I also want the ability to add more chores, map out rooms for home layouts, and I want a logbook for completed chores so users can log how long it takes them to complete, which can go into resizing the rectangle over time. I feel like there are a ton of functions we can do with this, and there aren't that many individualized non corporate daily planning applications out there, plus I wouldn't need a significant data driven API.

The last is a Character sheet builder for tabletop games. There are a ton of PDF fillable sheets out there, and there are also things like DND beyond, but I imagine an application where the user can select which game and which version of the sheet they want, and in the fillable areas we can actually link a database so that users can just drop down and fill in things. We can include spell libraries, spell book builders, and lots of options to make the character sheet and information easily usable, as well as allow users to save sheets. Like and SRD but for many tabletop games and which really easy functionality.

I also do not know where I would get the data for this.

## GPT Response

Choosing a project depends on various factors including your interests, the skills you want to develop, the complexity you're ready to handle, and the resources available to you. Let's break down your ideas based on the project specifications you've provided:

**1. Character Counter App**

**Pros:**

* High appeal to gamers and could fill a niche in competitive gaming.
* Opportunities to learn about data analysis and statistical modeling.
* Potential to integrate with existing game APIs or scrape data from community forums.

**Cons:**

* Finding or generating reliable data might be challenging.
* Requires knowledge of specific games and their metas.

**Implementation Ideas:**

* Data Source: You might need to create your own dataset based on game replays, forums, or community inputs. Alternatively, explore if any game APIs provide the necessary data.
* Frontend: A simple web or mobile app where users can select characters and see recommended counters.
* Deployment: Can be deployed on a cloud platform like AWS or Heroku.
* Version Control: Regular commits to GitHub with clear documentation of your data collection and analysis methods.

**2. Chore Tracker and Organizer**

**Pros:**

* High utility and potential for personal and family use.
* Visual representation of chores is innovative and user-friendly.
* Relatively easier to implement as it relies on user-generated data.

**Cons:**

* Ensuring a good user experience may be challenging, especially with the visual space constraints.
* Could require extensive front-end work to make the interface intuitive.

**Implementation Ideas:**

* Data Source: The data would be user-generated, including chore names, estimated times, and difficulty or category.
* Frontend: A web app with a drag-and-drop interface for organizing chores.
* Deployment: Could be deployed locally for personal use or on a platform like AWS for wider access.
* Version Control: Use GitHub for tracking changes, especially iterations on the user interface based on user feedback.

**3. Character Sheet Builder for Tabletop Games**

**Pros:**

* Appeals to a broad audience of tabletop gamers.
* High potential for customization and expansion.
* Can fill a gap between PDF sheets and more complex systems like D&D Beyond.

**Cons:**

* Data acquisition for different games and versions could be complex.
* Might require extensive backend development for database management.

**Implementation Ideas:**

* Data Source: This could be the most challenging part. You may need to manually compile data or find a community-driven database. Some games may have open APIs or allow access to their databases.
* Frontend: A web application where users can select games, versions, and fill in their character sheets.
* Deployment: Cloud platforms would be ideal for accessibility and ease of use.
* Version Control: GitHub would be essential for managing the complexities of data and the variety of games and rules.

**Recommendation**

Given the constraints and the opportunities to learn, **the Chore Tracker and Organizer** seems like a compelling project. It offers a unique value proposition with the visual organization of chores and appears to have a relatively straightforward path to achieving the project specifications. It also presents an opportunity to focus on user experience and front-end development, with a backend that's manageable but still offers plenty of learning opportunities, especially around database management and API development. You can start simple and scale up with features like chore history analytics or personalized recommendations based on past activity.

For sourcing data on the other projects, you might look into APIs provided by game developers, community forums, or data scraping (with respect to legal considerations), but these paths do present significant challenges. The Chore Tracker, by contrast, relies on user-generated content, simplifying the data acquisition aspect.

### Decision

We will be going with the Chore Tracker App. Name has yet to be determined.

# Phase 2 – Research and App Outlining

In our research we'll be looking for applications or services related to Chore Trackers or day planners. For each application we will be looking at the following items for comparison points:

1. **Operating System(s)**: Identify the platforms (Android, iOS, Windows, etc.) the service is available on.
2. **Price**: Detail all pricing levels, including free versions with their limitations and costs for full feature access or subscriptions.
3. **Ratings and Reviews**: Provide the average rating and the number of reviews that contribute to this average.
4. **Features**: Give an overview of the application's offerings.
5. **Customization Options**: Note the extent to which the app allows users to personalize it according to their needs.
6. **Integration Capabilities**: Mention if the app can integrate with other services and to what extent.
7. **User Interface and Experience (UI/UX)** (as an observation based on available information): Any comments on the app’s design, aesthetic, and ease of navigation might be particularly relevant to your interest in a smooth and comfortable UI.

## Results

**1. OurHome**

* **OS**: Android, iOS
* **Price**: Free
* **Ratings and Reviews**: Highly rated but specific ratings not provided. Appreciated for family organization features.
* **Features**: Task list, scheduling tools, assignable tasks with due dates, points and rewards system, grocery list, and calendar sync across devices.
* **Customization Options**: Assign tasks, set due dates, customize control levels.
* **Integration Capabilities**: Calendar sync.
* **UI/UX**: Designed for family use with easy identification through animal avatars​ ([BestApp.com](https://www.bestapp.com/best-household-chore-apps/))​.

**2. Chore Pad**

* **OS**: iOS only
* **Price**: Free
* **Ratings and Reviews**: Highly rated but specific ratings not provided. Noted for its simplicity and engagement for kids.
* **Features**: Colorful chore charts, customizable themes, sound effects, animations, trophies, and rewards.
* **Customization Options**: Customizable themes and chore charts.
* **Integration Capabilities**: Not mentioned.
* **UI/UX**: Simple, straightforward, engaging for younger users​ ([BestApp.com](https://www.bestapp.com/best-household-chore-apps/))​.

**3. Homey**

* **OS**: Android, iOS
* **Price**: Core features free, $4.99/month for premium features including bank account transfers.
* **Ratings and Reviews**: Highly rated but specific ratings not provided. Stands out for its allowance and money management features.
* **Features**: Chore to reward linkage, bank transfers for allowances, in-app messaging, tasks grouped by room.
* **Customization Options**: Chore assignments, room groupings.
* **Integration Capabilities**: Not mentioned.
* **UI/UX**: In-app messaging but criticized for its high monthly fee​ ([BestApp.com](https://www.bestapp.com/best-household-chore-apps/))​.

**4. Tody**

* **OS**: Android, iOS
* **Price**: Not specified, focused on smart assistance for chore management.
* **Ratings and Reviews**: Highly rated for its smart assistance in managing chores.
* **Features**: Task hierarchies based on frequency and time investment, role designation for family members.
* **Customization Options**: Chore compartmentalization, role assignments.
* **Integration Capabilities**: Not mentioned.
* **UI/UX**: Clean interface, aimed at decluttering the user's mind​ ([www.top10.com](https://www.top10.com/household-chore-apps))​.

**5. Today**

* **OS**: iOS, Android
* **Price**: Free
* **Ratings and Reviews**: Average score of 9.7 from reviewers, praised for its time tracking and task assignment.
* **Features**: Assigning tasks a time, recurring task functionality.
* **Customization Options**: Time-based task assignment.
* **Integration Capabilities**: Not specified.
* **UI/UX**: Highlighted for its smart repeating task approach and clean interface​ ([TheTodayApp](https://thetoday.app/blog/chore-apps-the-best-5-house-chore-apps-reviewed/" \t "_blank))​.

For your project, focusing on UI/UX and customization options seems essential, as these are areas where users have strong preferences. Additionally, considering the operating system reach could enhance accessibility, and offering unique features like time-based task assignments or family-oriented customization could set your app apart. Integration capabilities were less frequently mentioned but could be a bonus feature, especially for users looking to connect their chore tracking with other apps or services they use.

## Observations

The results preceding this section were returned in response to the prompt given to GPT. A physical verification and analysis of each app was then performed to validate details listed and form articulate observations.

* **~~OurHome~~**
  + I found a site, but the AppStore links were broken.
* **Chore Pad**
  + Upfront Cost of $4.99, only for iPad, and geared towards Kids.
* **Homey**
  + Combo App, one of those that wants to do everything, budgeting, groceries, messaging, calendars, etc. It is too much. The iOS version seems to have far better reviews than the Android version, showing a clear skew in support.
* **Tody**
  + I have personal experience with this one, and out of all of the Chore apps I have used I enjoyed the UI for this one the most, but the paywall is a challenge for family units, and it really just feels like I am setting up a checklist for repetitive tasks than managing an overall list.
* **Today**
  + Very New, barely any downloads. It seems to be focused on time management for individual tasks, utilizing visual timers as incentives. Not bad, but also not really addressing the main issue.

### General Observations

One of the applications had broken links, and after a few minutes of attempting to locate it, I found it prudent to just remove it from the list, reducing our list to 4.

A trend I noticed when studying these applications is that a majority of them tend to be geared towards family units, and aim to combine a bunch of functionalities into a singular app. This approach is common because I believe most developers are attempting to reduce the amount of navigation a user needs to do in order to stay organized with multiple agendas.   
  
I have personally used a couple of the ones on the list, and what I find is that the initial setup of the application takes forever and is satisfying, but then I just sort of…never use it. There are multiple reasons for this, (Paywalls, laggy, ads, too much set-up, messy UI, etc.), but I believe the MAIN reason is that they really don’t do anything more than make a checklist pretty.

# Phase 3 – App Planning

## Primary Focus

See, the problem I have, and what I want to address, is the never-ending To-Do list. These apps don’t really do anything to help reduce the size of the list or tackle it in chunks. It doesn’t assist with time management or planning; it just gives the user a prettier version of their list. These don’t work for me, and from what I understand they aren’t great long term for most people.

What I want is a program or an app that assists me in the planning portion of my list. I want an application that is designed to reduce the size of my To-Do list while also maintaining a realistic approach to daily chores, prioritization, time management, and scaled planning. I also want the application to be adaptive with information logging through interaction.

My approach for this project will be to try and design something geared towards people with low focus regulation like those with ADHD. If we can find a way to create an application that keeps the long-term attention of that demographic, it will inherently also be useful for those with higher degrees of focus that also want to reduce internal planning.

## Challenges

We need to establish a workflow that intentionally targets and eliminates specific obstacles for ease of use and long-term engagement.

**Obstacles**

* Set-Up
  + The set-up needs to be easy, and not very involved. At the end of the day, we are prompting the user to input the things they need to do and then the program helps the user find the time to do those things. A, meet B.
* Interaction
  + Interactions need to be quick, responsive, visually appealing, free of clutter, and highly intuitive.
    - Types of Interactions
      * Adding a task
        + Button on Lock screen
        + Task Properties

Name

Frequency: ?

None

Every N M’s (and) M

N

Number

Other

M

Days

Week(s)

Month(s)

Specificy day

Difficulty

*Why have this?*

Priority Level: None

None

Low

Medium

High

Critical

Due Date: ?

Time: ?

Estimated Time Needed: 10 min

Dependencies: ?

Category: ?

Subject: ?

Work

School

Chores

Fun

Hobbies

Locations: ?

Home

Rooms?

Notifications: ?

Default as “on” for 5 minutes before end of task

On

Sound:?

Off

N minutes nefore M

N=Numbers

M=Minutes

* + - * + *The UI for adding a task needs to be as simple as possible, easier than adding an event to the Calendar. There should be as many values as possible set to default, and it should be Extremely easy to add a task from any screen or page. There should be a button on every page that allows a user to click and add a new task to the overall task list. Think of phone over PC. People think of things they need to do all day, and most people have a never-ending task list. The main challenge in capturing these tasks is availability. We are not always at a computer or near a pen and paper. We need to be able to effortlessly add on a task at a whim with as few restrictions or obstacles as possible.*
      * Completing a Task
        + See Flowchart
      * Deleting a Task
      * Editing a Task
      * Choosing a task
      * Create Group
        + Grouping tasks/Nesting Tasks

Grouping takes the number of tasks and their assigned Time needed and groups the tasks, presenting the average time as the time needed. This allows users to group tasks into rooms, like multiple tasks involved in cleaning a room, while still being able to target specific tasks.

* + - * Moving tasks in display
      * Duplicating a Task
        + UI Focus here
* Display
  + The main display should run like an agenda with different layouts depending on user preferences.
    - Immediate (next task, next hour, etc.).
      * This task should be a ribbon display, the (Current) tasking, and should be visible anywhere and easily accessible.
    - Today
    - N Days
    - N Weeks
  + *Free Time*
    - *In settings, cannot reduce free time below 2 hours, warning message, (In order to support your mental health, we require this setting to provide a minimum of 2 hours per day doing what you WANT*
    - *Min standard each day built in by default 2 hours (*
      * *Can turn off in options with mental health warning*
        + *“Taking time for yourself is important!” Sacrifice 5 minutes*
      * *Track statistics on sacrificing free time*
        + *Which tasks you sacrificed time on*
        + *How much time is sacrifice*
    - *Integrating a structured approach to budgeting free time for mental health can indeed draw parallels from financial budgeting strategies. Just as setting aside a certain amount of money for leisure activities can prevent impulsive spending and contribute to financial health, allocating specific periods of free time can promote better mental health and life balance. Here's how you might structure this concept in your app:*
    - ***Budgeting Free Time in Your Day Planner App***
      * ***Define Free Time****:* 
        + *Clearly define what constitutes "free time." This could be any activity that is not work-related, including hobbies, relaxation, socializing, or simply doing nothing.*
      * ***Assess Time Availability****:*
        + *Users should start by assessing how much total time they have in a day, subtracting hours spent on obligatory activities like work, commuting, eating, and sleeping.*
      * ***Set Free Time Goals****:* 
        + *Based on personal needs and lifestyle, users can set goals for daily or weekly free time. For instance, someone might aim for at least one hour of free time each day, or perhaps more on weekends.*
      * ***Flexible Allocation****:* 
        + *Just as with financial budgeting, where some months might require more spending in certain categories, some days might need more free time due to stress or other factors. The app can allow for flexible allocation with suggestions to increase free time during more stressful periods.*
      * ***Monitor and Adjust****:* 
        + *The app can provide tools to track how free time is spent and alert users when their free time falls below their set goals. It can offer suggestions to balance the week or month if some days are particularly busy.*
      * ***Reflective Feedback****:* 
        + *Just as financial apps might provide reports or insights into spending habits, your app can give feedback on how well users are meeting their free time goals and the impact on their reported mood and stress levels.*
      * ***Integration with Physical Activities****:* 
        + *Link free time with activities that are known to reduce stress, such as exercise, meditation, or hobbies that involve physical movement. This can be part of the budgeting, where certain activities are prioritized based on the user's preferences and effectiveness in reducing stress.*
    - ***Benefits of Budgeting Free Time***
      * ***Mental Health Maintenance****:* 
        + *Regularly scheduled free time can help prevent burnout and reduce stress.*
      * ***Improved Productivity****:* 
        + *Paradoxically, having dedicated non-work time can improve productivity during work hours.*
      * ***Enhanced Creativity****:* 
        + *Free time allows the mind to wander, which is crucial for creativity and problem solving.*
      * ***Better Physical Health****:* 
        + *Engaging in leisure activities can promote physical health, especially if those activities include some form of exercise.*
    - *By treating free time as a crucial resource to be managed — akin to money — users can make more conscious decisions about how they spend their time, ultimately leading to a healthier, more balanced life. This approach in your app could be a unique selling point, particularly appealing to users who thrive on structured schedules but still need help managing their leisure time effectively.*
  + Display should be interactive, allowing users to drag and drop tasks easily, as well as group them through drag and drop.
    - There are a lot of Drag and drop functions that really suck, to avoid this I want to add some limitations on how ours works. First, each item that can be grabbed will have a symbol, like a dot, on the task container. You click on hold the dot, that allows the user to move the task. Next, I want to create a “Quick Group” Function. This will be indicated by another symbol on each task that you can drag and drop another task on top of and it will group them.
* Notifications
  + “You have 5 minutes before this task ends” alert
    - Buzz
    - Sound:?
    - Options
      * Acknowledge
      * Sacrifice Free Time
        + (Theory behind this is to explicitly label the consequence of procrastination as a realistic forethought in peoples mind. Failing to plan means less free time when you want it )
  + End Of day Notification
    - Default time, or user selected
      * Half an hour before bedtime
    - User opens app, can click somewhere to mark a task as complete, some pop-up questions can appear
* Database
* Charting and Reporting
  + A radar chart, also known as a spider or web chart, is a great way to visualize multi-dimensional data in a way that's easy to understand at a glance, making it particularly useful for tracking time management and productivity trends. Here are some potential values you could use on a radar chart to help users visualize and manage their time more effectively:
    - **Task Completion Rate**:
      * This could represent the percentage of tasks completed versus those planned each day or week, giving users a quick visual representation of their productivity.
    - **Time Spent on Tasks**:
      * Reflects the actual amount of time spent on tasks versus the estimated time. This can help users see if they are overestimating or underestimating the time tasks take, which is crucial for better planning.
    - **Prioritization Accuracy**:
      * Measures how often high-priority tasks are completed on schedule compared to lower-priority tasks. This helps users understand if they are effectively prioritizing their workload.
    - **Interruptions Managed**:
      * Tracks the number of interruptions (phone calls, emails, etc.) handled during focused work sessions. A lower number could indicate better focus and fewer distractions.
    - **Time Allocation Consistency**:
      * Compares the time allocated to various categories of tasks (work, personal, rest, etc.) against planned distributions. This helps ensure a balanced approach to daily activities.
    - **Deadlines Met**:
      * Indicates the percentage of deadlines met on time, which is critical for assessing reliability and scheduling accuracy.
    - **Efficiency Over Time**:
      * This can be a measure of how much work is accomplished in a given period and can be tracked over days, weeks, or months to show changes in efficiency.
  + To implement this in your application using Xamarin or MAUI:
    - **Collect data** continuously through the app's task management features.
    - **Analyze** the data to compute the values needed for the radar chart.
    - **Visualize** this data using a charting library that supports radar charts, like Syncfusion, Microcharts, or OxyPlot, which are compatible with Xamarin and MAUI.
  + By presenting this data in a radar chart, users can visually assess their performance across multiple dimensions of time management, allowing them to identify areas for improvement more readily.

## Diagrams

The following highlighted section is a template for what diagrams we may make for this project. We will be replacing this information as we work. No UML Has been created yet.

UML diagram for a multipage Android application using .NET MAUI with SQLite as the database, we can focus on key aspects such as the app structure, data flow, and interactions between different components. Here's an example of what this UML diagram might include:

**1. Components:**

* **Pages/Views**: Each page represents a different view in the application, such as the Home Page, Details Page, Settings Page, etc.
* **ViewModels**: These manage the data and logic for the views but do not include any UI code. They often contain commands and data bindings.
* **Models**: Data structures representing the entities in the application, like User, Product, etc.
* **Database Service**: A class or set of classes handling all database interactions using SQLite.

**2. Diagram Sections:**

* **Use Case Diagram**: Shows the interactions between users and different parts of the system, highlighting features and user capabilities.
* **Class Diagram**: Displays classes, their attributes, methods, and relationships. This will include ViewModel classes, Model classes, and Services.
* **Activity Diagram**: Describes the flow of activities in the application, such as user logging in, viewing details, editing data, etc.
* **Sequence Diagram**: Visualizes how objects interact with each other over time for specific operations, like adding or fetching data.

**3. Example Class Diagram Details:**

* **HomeViewModel**: Contains commands to navigate to other pages and properties bound to UI components.
* **UserService**: Methods for CRUD operations on User objects, interacting with the SQLite database.
* **ProductViewModel**: Methods and properties related to product interactions.
* **SQLiteDataService**: Generic service handling all SQLite operations.

**4. Example Sequence Diagram for Adding a Product:**

* **User triggers action on UI**: User clicks 'Add' button on the product page.
* **ViewModel executes command**: ProductViewModel processes add command.
* **DataService updates SQLite DB**: SQLiteDataService executes SQL commands to insert data into the database.
* **Callback to ViewModel**: DataService informs ViewModel that operation was successful.
* **UI Update**: ViewModel updates UI with new product list or confirmation message.

This structure ensures that the application is maintainable and scalable, and using UML diagrams like these helps visualize system components and interactions effectively. Would you like a visual representation of any specific part of this architecture?

# Presentation Requirements

Who, what, why, which platform, which tools, live demonstration of SOME KIND.

## Presentation Outline

* Title
* About the App
  + Reason/Purpose
    - Specifically targeting ADD/ADHD and other focus based disorders.
  + Supporting Research
    - One in Six College Freshmen Has ADHD
    - 58 percent of students with ADHD have at least one comorbid disorder.

## Software and Extensions

### Visual Studio Code

### *.NET MAUI Extension*

This extension is designed to support developers using .NET Multi-platform App UI (.NET MAUI), which is a framework for creating native mobile and desktop apps with a single codebase. For Android application development, it helps by allowing developers to use C# and XAML to build Android apps alongside other platforms, which can simplify the development process and enhance code reusability.

### *GitLens Extension*

GitLens is an extension that supercharges the built-in capabilities of Git in Visual Studio Code. It helps developers better understand code history through Git annotations, including who changed what and why. In Android development, this can be particularly useful for tracking changes across a team, reviewing past code decisions, and managing versions of an app as it evolves.

### *C# Extension*

The C# extension for Visual Studio Code provides rich support for the C# language, including features like IntelliSense (auto-completions), debugging, and more. This is essential for Android application development when using frameworks like Xamarin or .NET MAUI, as it allows developers to write and debug their C# code directly within Visual Studio Code.

### *NuGet Package Manager*

This extension allows developers to manage NuGet packages directly from within Visual Studio Code. NuGet is a package manager for .NET, making it easy to include and manage third-party libraries in your projects. For Android developers using .NET-based frameworks, this extension simplifies adding and updating libraries that enhance functionality or facilitate various aspects of app development.

### *Live Share*

As previously described, the Live Share extension by Microsoft enables real-time collaborative coding. This is highly beneficial in team settings where multiple developers may need to work together on the same codebase, including Android application projects, facilitating instant sharing and editing of code across different devices and platforms.

### *Android iOS Emulator*

This extension provides integration with Android and iOS emulators directly in Visual Studio Code, allowing developers to test and debug their mobile applications without leaving the editor. It simplifies the workflow by enabling quick testing across multiple device emulations, which is crucial for ensuring that Android apps function correctly across different devices and configurations.

### Draw.io

Will be used for Diagrams like UML

### DataGrip

Will be used to visualize our SQLite DB. We will be building an API that interacts with the Database we are building, which will act as our primary API.

### Android Studio

Responsible for our emulator so we can live test the Application.

## To-Do List

* Diagrams
  + UML
  + FlowCharts
  + ERD for DB
* ~~Setup VSCode File~~
* ~~Name our App~~
* Start Working on Presentation
  + Create outline, gather supporting documentation and research to support our ideas, or maybe to counter them.
* Link GitHub Repo

# Meeting Minutes

## Meeting (4/12)

### Signing the Blood Oath

We talked about the layout of what the application would look like from the functionality aspect of things. We talked about the idea of creating and setting limits for the users like hours for sleeping, hours for free time, etc and holding the user fully accountable for any changes in these. For example, the user has a default limit of 2 hours of free time and if they are wanting to incorporate less free time then they must fully commit to doing so and “suffer the consequences” that come with that. Another way we thought to hold the user accountable for sacrificing free time is pushing everything on their schedule down if they choose to sacrifice free time because this will make them less likely to do this on a consistent basis. Overall, this day was planning out the complete functionality of our application including things we liked, didn’t like, and wanted to change from other planning applications. Halfway through, we had an epiphany that our application will stretch further than just the general use case but will also solve the problems of procrastination, sleep deprivation, and lack of organization.

## Meeting (4/13)

### SWOT Analysis(Group, not individual)

* Strengths
  + Coding
  + Planning
  + Group work gives us an advantage
  + Presenting
  + Brainstorming
  + Time Management
  + Research
* Weaknesses
  + Time Blindness
  + Procrastination
  + Overloaded Schedule
* Opportunities
  + Mobile application should set us apart
* Threats
  + New Coding Languages
  + New unfamiliar frameworks
  + Possible bugs or other difficulties
  + Schedule conflicts with other courses

Today, we started by figuring out our strengths and weaknesses so we could become more aware of how we were going to divvy out our tasks. Next, we focused mainly on installing the programs and applications that we would be using for our application, including Android Studio which would allow us to get live updates on the functionality and progress that we make on our code. Once we could install the different programs we would be using, we could make a phone that would emulate the code we were composing. This day was heavy on the installation and setup aspect of things which is good because we are gearing up to start coding and composing our application.

We scheduled time to begin coding before the presentation. The current problem we are both running into is just time availability. The planning is going well, and there are many instances of great cohesion on the team, now it is just time to begin individual portions of tasks, and revisit occasionally.

# Shower Thoughts

A place to document ideas formed outside of working hours to explore later.

Automatically re-prioritize tasks with Due dates based on the relative distance to the due date.

Method – If task.dueDate == True, then AssignPriorityLevel();