Jewels Academy Time Management Application

Design Document

Team Number: sddedc22-06
Client: Jewels Academy
Advisors: Rachel Shannon

Team Members/Roles
Adrian Van Der Veer - Team Organization
Olusola Ogunsola - UI Design
Ben Hourigan - File Consistancy/Organization
Christopher Burgos - Client Interaction
Theron Gale - Individual Component Design

Team Email: sddec-06@iastate.edu
Team Website: http://sddec-06.sd.ece.iastate.edu/

Executive Summary

Development Standards & Practices Used

List all standard circuit, hardware, software practices used in this project. List all the Engineering standards that apply to this project that were considered.

- SCRUM / Agile
- ISO/IEC/IEEE 29119-3-2021 standards of software testing
- IEEE 12207-1996 standards for developing and managing software
- IEEE 7002-2022 standards of privacy oriented considerations in software development

Summary of Requirements

List all requirements as bullet points in brief.

- Functional Requirements
    - Access the Dashboard, Tasks List, and Goals pages as well as their corresponding functions
    - Internet Connectivity
    - Cellular Network Connectivity
- Resource Requirements
    - Google Cloud for storage
    - iOS and Android app stores for distribution
    - Finish minimal viable product version by December
- Qualitative Requirements
    - Visually appealing in the Jewels Academy color scheme
    - Speedy transitions from page to page
- Economic/Market Requirements
    - None
- Environmental Requirements
    - None
- UI Requirements
    - Easy to navigate

    - Visually appealing in the Jewels Academy color scheme
    - Compact, but open enough for easy navigation
- Other Requirements
    - Login requests shall be processed within 2 seconds. (Constraint)
    - Page transitions shall occur within 1 second. (Constraint)
    - Form submission shall occur within 5 seconds. (Constraint)
    - Google Cloud integration with a Jewels Academy account
    - Development environment setup

Applicable Courses from Iowa State University Curriculum

List all Iowa State University courses whose contents were applicable to your project.

- Com Sci 227
- Com Sci 228
- Com Sci 309
- Com Sci 363
- S E 319
- S E 329
- S E 422X

New Skills/Knowledge acquired that was not taught in courses

List all new skills/knowledge that your team acquired which was not part of your Iowa State curriculum in order to complete this project.

- Coding in Python
- Coding in React Native
- Cloud Database Management
- Cloud Connector Setup
- Development Environment Setup
- Data Security
- REST API Development
- Javascript to REST connection

Table of Contents

List of figures/tables/symbols/definitions

1 Team

1.1 Team Members

- Adrian Van Der Veer
- Olusola Ogunsola
- Ben Hourigan
- Christopher Burgos
- Theron Gale

1.2 Required Skill Sets for Your Project

- Mobile App Development
- UI/UX Development
- Requirements Engineering
- Database management
- Graphic design

1.3 Skill Sets covered by the Team

- Frontend Development - Theron, Ben, Chris
- Backend Development - Adrian, Ben, Sola, Chris
- UI/UX Development - Theron, Sola
- Database Management - Adrian, Ben

1.4 Project Management Style Adopted by the team

- Agile/SCRUM

1.5 Initial Project Management Roles

- Ben: File Consistency/Organization - Frontend Developer
- Adrian: Team Organization - Backend Developer
- Theron: Individual Component Design - Frontend Developer
- Chris: Client interaction - Full Stack Developer
- Sola: UI design - Full Stack Developer

2 Introduction

2.1 Problem Statement

For many students, school is a stressful time. There are dozens of assignments to keep track of, tests to study for, sports and other extracurriculars to participate in, and much more. This can cause many issues in their day-to-day lives. Our group wants to create an application that will allow students with many responsibilities from school, to work, to social activities to manage their time in a visual and easy-to-use way and to uphold goals to improve personal fulfilment.

2.2 Requirements & Constraints

- Functional Requirements
    - User should be able to:
        - Access the Dashboard, Tasks List, Goals, and Settings pages
            - These are all interconnected
        - Add/Set/Remove tasks
        - Adjust timescale for the tasks list
        - View calendar
        - Adjust timescale for the calendar
        - Filter which tasks, events, etc. appear on the calendar
        - Add/Set/Complete/Remove Goals
    - Login/Logout/Register
    - Adjust account settings
    - Adjust notifications settings
    - Internet Connectivity
    - Cellular Network Connectivity
- Resource Requirements
    - Google Cloud for storage
        - Google Drive for journal entry, feelings chart, and mood chart storage
    - iOS and Android app stores for distribution
    - Finish prototype by early May
- Qualitative Requirements
    - Visually appealing in the Jewels Academy color scheme
    - Speedy transitions from page to page
- Economic/Market Requirements
    - None
- Environmental Requirements
    - None
- UI Requirements
    - Easy to navigate
    - Visually appealing in the Jewels Academy color scheme
    - Compact, but open enough for easy navigation
- Other Requirements
    - Login requests shall be processed within 2 seconds. (Constraint)
    - Page transitions shall occur within 1 second. (Constraint)
    - Form submission shall occur within 5 seconds. (Constraint)

2.3 Engineering Standards

ISO/IEC/IEEE 29119-3-2021 Standards of Software Testing

- Standard specification for testing software. This standard is useful for our project because it's important for us to test our application and ensure its reliability before it is deployed to it's primary users.

IEEE 12207-1996 Standards for Developing and Managing Software

- This standard provides industry best practices for developing and maintaining software. This is important for our project because we'll need to be able to effectively communicate with Jewels Academy on the best way to maintain the application after we're no longer associated with Iowa State.

IEEE 7002-2022 Standards of Privacy Oriented Considerations in Software Development

- This standard describes the requirements for software that include "privacy-oriented considerations regarding products, services, and systems utilizing employee, customer, or other external user's personal data are defined by this standard", and as our application will require the storage of user data for login and user verification this standard applies to the privacy of that data.

2.4 Intended Users and Uses

The primary beneficiary for this project will be students attending the Jewels Academy organization. Students will be able to use the newly created mobile application to manage their daily lives by integrating their daily tasks, academics, and calendar into one app. With the Jewels Academy Time Management mobile app, students will be able to add, set, and edit tasks as well as enabling task notifications. Students can use the app's calendar in order to integrate canvas, display their daily routines, and plan other life events. And another use case for the app is the implementation of a mental health journal so that students will be able to track their mental health status over time.

3 Project Plan

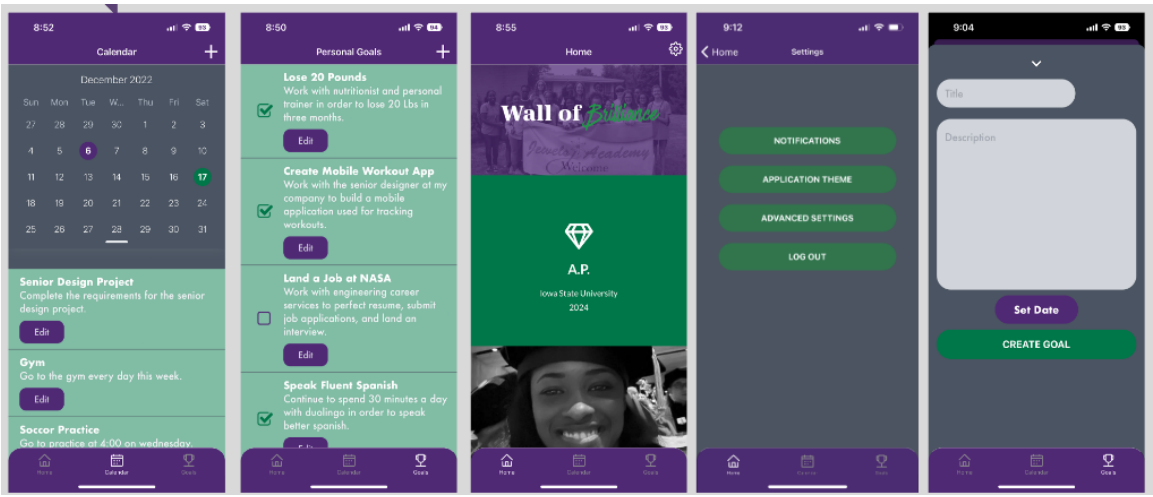
3.1 Project Management/Tracking Procedures

Our group decided to use the Agile development management style, because it is the most adaptable to a relatively complex software task, it allows the group to keep updating continually so when there are things the group doesn't like we can easily change them as very little is dependant on the new content each sprint compared to a waterfall approach where replacement of unwanted parts is much more difficult.

Our group intends to use a mixture of Trello, Git issues, Git, for development and planning and discord for communication among the team as well as webex and email for communication to those outside the team

3.2 Task Decomposition

Our group intended to create several pages with various internal functionalities within them, we will have a dashboard page which will be the landing page when a user logs in to the application, a calendar which will show a monthly or weekly overview of tasks and events with a task list which will show a breakdown of tasks by day, and an account settings page. Each page will have various sub-features broken down below. An example of the project is also provided below:



- Final Design

1.1 Dashboard

1.1.1 General Information
1.1.2 Reminders

1.2 Calendar

1.2.1 Calendar
    1.2.1.1 Change Timescale (Daily, Weekly, Monthly)
    1.2.1.2 Filter Results

1.3 Task List / To-Do List

1.3.1 Add Task
1.3.2 Set Task
1.3.3 Remove Task
1.3.4 To-Do List

1.4 Goals List

1.4.1 Add Goal

1.4.2 Update Goal
1.4.3 Remove/Complete Goal
1.4.4 Goal List

1.5 Account Settings

1.5.1 User Profile
1.5.2 Login / Register New User
    1.5.2.1 Logout
    1.5.2.2 Recover Username / Email / Password
1.5.3 Change Display Theme (Dark, Light, etc.)
1.5.4 Change Notification Settings

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

MileStones:

- Dashboard Page Complete
- Settings Page Complete
- Calendar Page Section Complete
- To-Do Page Section Complete
- Goals Page Complete
- UI complete
- Prototype Complete

We will use the metrics of, response time which we want to keep to a minimum, loading speed which we want to be as fast as possible, and power draw which we want as low as possible for a mobile application.

From the beginning of the design phase the current time there have been several iterations of the project design mostly throught the first semester in 491 when the team decided to add on mental health capabilities and google cloud, but also later in 492. In 492 the visual design of the application changed drastically to the far more modern and refined UI that you see an example of above, but the functionality also changed, the dashboard was streamlined down to al more singular page rather than the overwhelmingly complex page it was initially designed as, as well as replacing the Mental Health page with a page for user long tem or repeating goals. There was also a period during 491 where additional features were discussed for the application including subscriptions, and teired versions, however due to time and resource constraints those were left as far back burner stretch goals along with some of the other additional functionality like weather and a web crawler for inspirational quotes.

3.5 Risks And Risk Management/Mitigation

Our group doesn't have much risk regarding hardware or outside factors, our risks are mostly internal regarding communication and internal software bugs. Our group can mitigate this by having multiple lines of communication including discord, email, and in person meetings as well as using a CI/CD software with useful testing to ensure the software is reliable and relatively bug free. Each push to our collective git will have to be integration tested to lower code conflicts, and each sprint our group will hold a planning meeting that will include time for us to share any faults our members have found so they can be resolved.

Risks	Category	Probability	Impact	RMP
Login Security	Technical	45%	2	
New Group	Operational	80%	2	Try and work synergistically with each other to minimize any potential conflicts that arise
Communication Issues	Schedule	35%	2	
GPS/Mapping Issues	Technical	30%	3	
Page Access Speed	Technical	40%	3	
Team Knowledge	Operational	60%	2	Keep each other up to date on what has been learned, as many of us have little to no experience with some of the project's technologies.
Program Task Exporting	Technical	45%	2	
App will not Reach Expectations in Time	Schedule	33%	1	
Unstable Project Scope	Schedule / Operational	40%	1	

Impact Values:

1 - Catastrophic
2 - Critical
3 - Marginal
4 - Negligible

3.6 Personnel Effort Requirements

Given the teams gereral unfamiliarity with the proposed software and development environment, including such factors as new technology in the case of cloud hosting, a new IDE in the case of Intellij Idea, and new lagngages in the case of Python and React Native, the development time of this application may exceed the average of normal application development time for an application of this scope. Our group took into account the unfamliarity of this project, and came up with the following estimates of how long on average it would take the group members to complete each of the necessary tasks based on their previous experience with software development, and estimates of time to become comfortable with the unfamiliar design components. Below is an estimate of the projected work hours for the development of this application.

Task Name (Frontend)	Estimated Work hours
4.1 Dashboard	10
4.1.1 General Information	5
4.1.2 Reminders	5
4.2 Navigation between pages	20
4.3 Calendar Component	30
4.3.1 Calendar	20
4.3.1.1 Change Timescale (Daily, Weekly, Monthly)	10
4.4 Task List / To-Do List Component	55
4.4.2 Task List	5
4.4.3 Add Task	20
4.4.4 Edit Task	20
4.4.5 Remove Task	10
4.5 Goal List	25
4.5.1 Goal List	5
4.5.2 Add Goal	5
4.5.3 Edit Goal	10

4.5.4 Remove Goal	5
4.6 Account Settings	25
4.6.1 Login / Register New User	10
4.6.1.1 Logout	5
4.6.2 Change Display Theme (Dark, Light, etc.)	5
4.6.3 Change Notification Settings	5
Total Frontend	165

Task Name (Backend)	Estimated Work hours
4.1 Setup	20
4.1.1 Database Setup	5
4.1.2 Flask Setup	15
4.2 User Table	33
4.2.1 Setup	5
4.2.2 Create User / Register User	3
4.2.3 Login	3
4.2.4 Update Password	3
4.2.5 Delete User	1
4.2.6 Testing	3
4.2.7 Debugging / Error Fixes	15
4.3 Task List / To-Do List Component	20
4.3.2 Setup	2
4.3.3 Add Task	2
4.3.4 Edit Task	2
4.3.5 Remove Task	1

4.3.6 Clear Overdue tasks	4
4.3.7 Testing	4
4.3.8 Debugging / Error Fixes	5
4.4 Goal List	10
4.4.1 Setup	1
4.4.2 Add Goal	1
4.4.3 Edit Goal	2
4.4.4 Remove Goal	1
4.4.5 Testing	2
4.4.6 Debugging / Error Fixes	3
4.5 Account Settings	6
4.5.1 User Profile	2
4.5.3 Change Display Theme (Dark, Light, etc.)	1
4.5.4 Change Notification Settings	1
4.5.5 Testing	1
4.5.6 Debugging / Error Fixes	1
4.6 Integration	50
4.6.1 Cloud Integration	20
4.6.2 Backend-Frontend Connection	30
Total Backend	139

3.7 Other Resource Requirements

Identify the other resources aside from financial (such as parts and materials) required to complete the project.

- Google Cloud integration with a Jewels Academy account

- Development environment setup

4 Design

4.1 Design Context

4.1.1 Broader Context

Our project seeks to serve female students from challenging economic areas with a tool to make organizing the busy lives and schedules of students easier and giving them methods to assess and better manage their mental health as well as tools to keep track of upcoming activities.

Area	Description	Examples
Public health, safety, and welfare	How does your project affect the general well-being of various stakeholder groups? These groups may be direct users or may be indirectly affected (e.g., solution is implemented in their communities)	• Possible improved test scores by encouraging more effective study skills using cognitive psychology research • Better understanding and management of stress to improve health and quality of life • Improves time management to reduce undue stress and ensure important activities are done in the correct time frame.
Global, cultural, and social	How well does your project reflect the values, practices, and aims of the cultural groups it affects? Groups may include but are not limited to specific communities, nations, professions, workplaces, and ethnic cultures.	• Development or operation of the solution would violate a profession's code of ethics, implementation of the solution would require an undesired change in community practices
Environmental	What environmental impact might your project have? This can include indirect effects, such as deforestation or unsustainable practices related to materials manufacture or procurement.	• Our group is using Google Cloud which is carbon-neutral, and dedicated to being even more environmentally friendly.
Economic	What economic impact might your project have? This can include the financial viability of your product within your team or company, cost to consumers, or broader economic effects	• This project at its inception should be free to download so as not to cause monetary strain on users who come from challenging economic circumstances

	on communities, markets, nations, and other groups.	• This application should consistently cost a small enough amount for usage that it does not unduly pressure the economic resources of the user base

4.1.2 User Needs

As our users are only students and no other groups, there are only 2 user needs:

- Users need a way to easily manage activities including school, classwork, extra-curricular activities, and work because students especially from low income circumstances have a considerable amount of things to do each day, and without a way to schedule all of that, some activities or assignments may be missed or fall to the wayside.
- Users need a way to manage their stress levels and improve their overall mental health, because stress and poor mental health are incredibly detrimental to overall health, and efficiency and quality of work and schooling.

4.1.3 Prior Work/Solutions

There are some already existing products for scheduling, timekeeping, and time management including the various major calendar applications like google or outlook. APIs from these products can be incorporated into our app with the help of the future group who will pick up where we left off.

Product Name	Product Features	Integration
Google Calendar	• Variable Timeframes • Schedule Goals (scheduled for the user with preferences) • Schedule reminders • Schedule events in advance • Schedule a Task to be completed • Show to do list with tasks, holidays, reminders, and events • Light/Dark Theme • Simple Design • Easy to use	Has API Integration for • Javascript • Java • .NET • Node.js • PHP • Python • GO • Android • IOS
Trello	• Task lists (To do, doing, done) • Simple UI • Search feature	API Integration: • cURL • Node.js

	• Notifications • Workspaces/ Team workspaces • Upload attachments • Scan documents • Comments • Similar to GIT issues, however, there's no code integration. The app's purpose is strictly for task management and organization	• Java • Python • PHP
MyState	• Dashboard • Weather • Schedule • Busses • Map • Events • Easy to use • Simple UI	API Integration: Getting in contact with MyState department
OutLook	• Calendar • Meeting reminders • Schedule recurring events • Create groups to collaborate with • Communication with microsoft teams for online meetings • Application for taking notes • Email communication • Create to-do lists	Supported languages for API integration: • Node.js • PHP • Python • Ruby • Objective-C • Java • C# (.NET) • cURL
Canvas	• Schedule Goals (both scheduled and not scheduled by user) • Schedule reminders/events • To-Do list view (mobile only) • Calendar and day-to-day views • Course Separation • Each course has its own page & modules • Grouping System • Inbox System • File Upload/Download System (Canvas Studio) • Personalized Accounts System • Grades System • Clean and Mostly Simple Design • Mostly easy to use	Has its own API to integrate Shown integration with: • Node.js • Javascript • Python

4.1.4 Technical Complexity

The technical complexity of the project is sufficient because of the up-to-date frameworks being used and modern technological standards being implemented. All of the technologies being used in this project are industry standard and meet all of the complexity guidelines currently being implemented in all modern applications.

Relevant list of frameworks and technologies:

a. Intellij IDE
    i. Modern up to date IDR used for backend development
b. Visual Studio Code
    i. Modern IDE used for frontend development
c. Figma
    i. Used for front end design and showcasing page navigation
d. React Native
    i. JavaScript Framework used by many modern applications to build stunning UI and easy to navigate apps
e. Python
    i. Used for backend design and API calling
f. Google cloud
    i. Used by the backend for hosting our application. Cloud services are what many applications are moving to and away from physical servers.

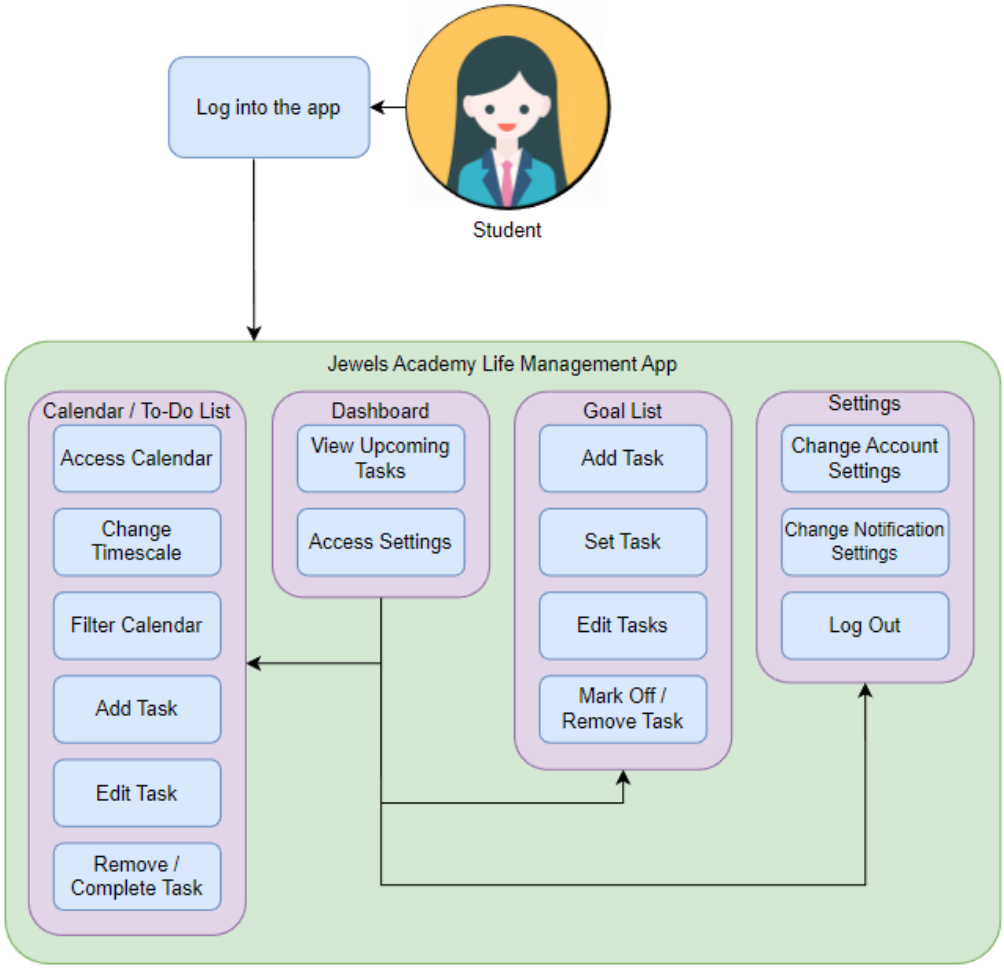
4.2 Design Exploration

4.2.1 Design Decisions

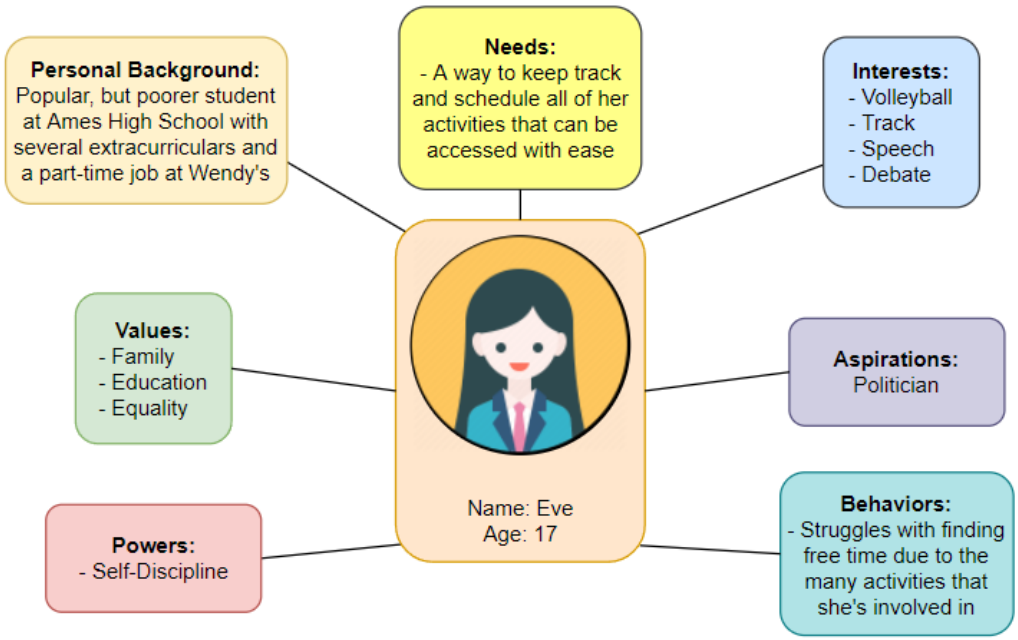
- Our team decided to use a google cloud backend SQL server proposing a SaaS solution to the given project for data storage and encryption.
- Our group has decided to use react native as the development language, Gitlab for version control, and Gitlab CI/CD for integration and testing of proposed changes to the code-base.
- Our group has also used python running on google cloud's app engine to run the functionality of the backend for the application.

4.2.2 Ideation

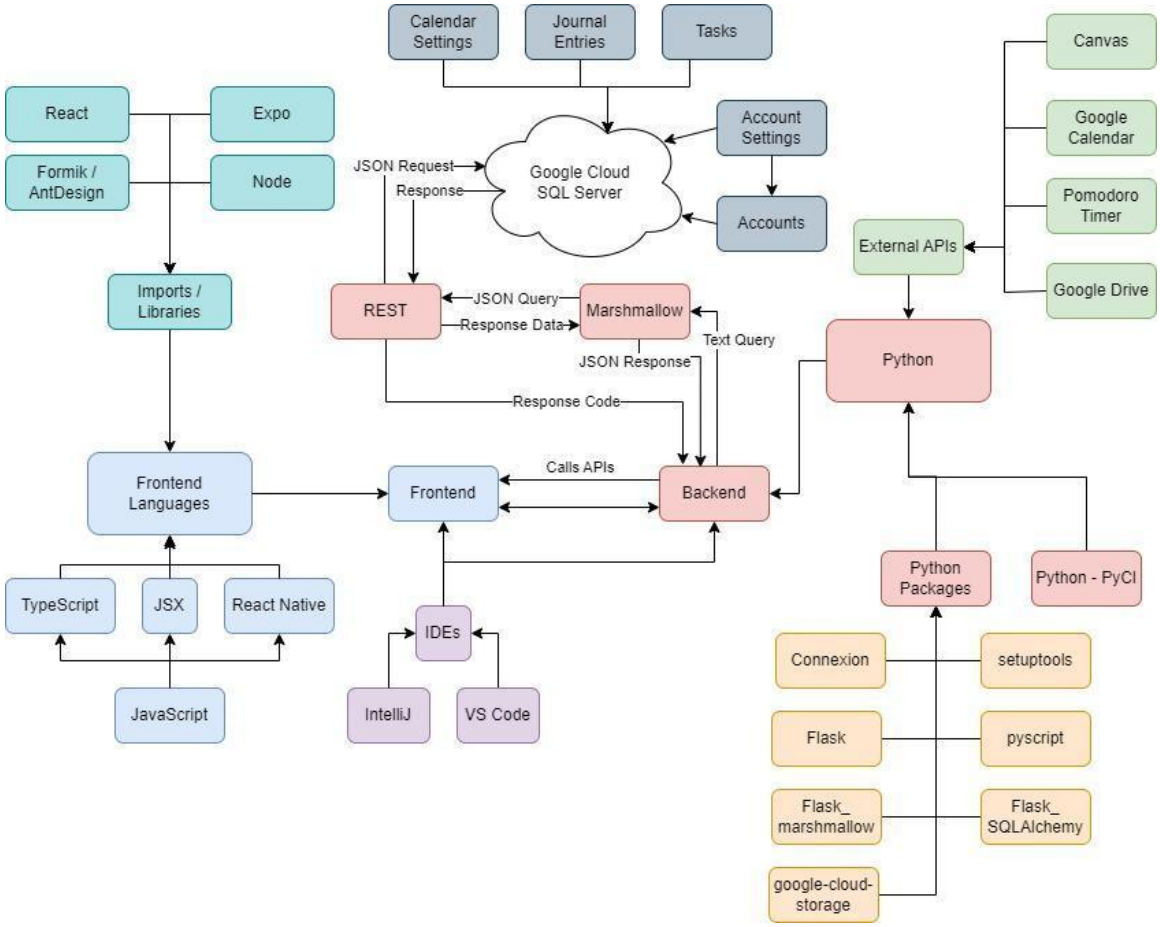
Our group used mainly used user profiles and use case diagrams to brainstorm ideas for the app.



- Above is our use case diagram to show what all needs to be included in our app



- Above is an example user persona for an ideal user for our app



- Above is a visual map for the design of the application, including technologies, languages, and APIs

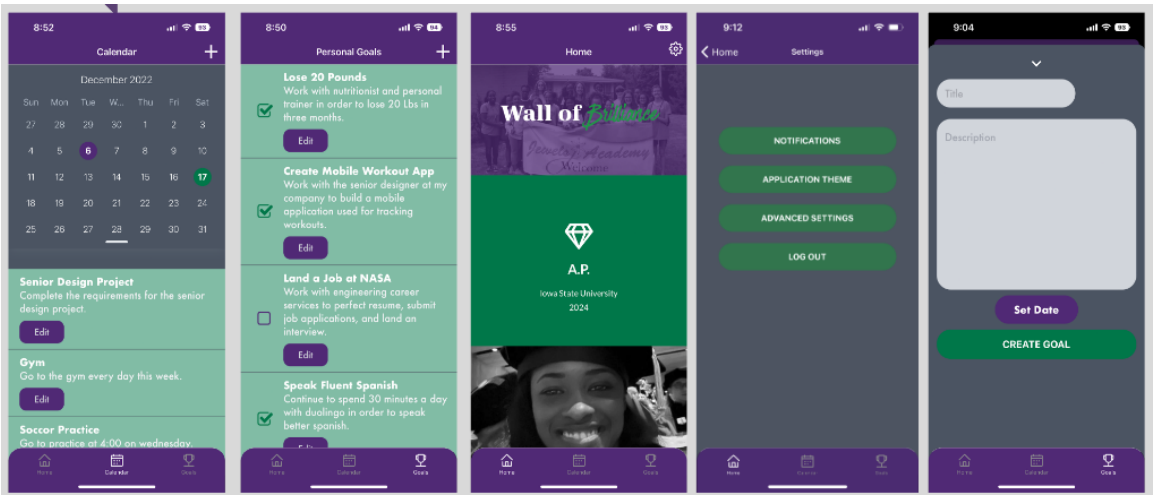
4.2.3 Decision-Making and Trade-Off

Our group went through several iterations of design descriptions within the team, including our faculty advisor and client, for the refinement of ideas into a true framework of the final project. We designed multiple iterations of prototypes using the current ideas of the project and refined down the things that we felt did not fit well within the scope of the project, and added in other functionalities given user needs, and stakeholder interests in features that would be good additions. After further review, both internally and with input from the client the specifications changed to more accurately reflect the current application with a Flask REST backend, and slimmed down functionality for a more refined product that more accurately appealed to the faculty liaison to Jewels Academy as well as its president.

4.3 Proposed Design

4.3.1 Design Visual and Description

Final UI Design



4.3.2 Functionality

Our current design is intended to operate as an academic life management application. The goal is to help students manage stress from academics through creating tasks/homework reminders as well as setting and tracking personal goals. Our design is aimed to be used so the user has less buttons to push to create their goals and assignments. We want the application to be easy to use, easy to track, and ultimately a pleasurable experience for the user. The current layout navigation and provisional design of the application can be seen above.

4.3.3 Areas of Concern and Development

During the process of development the team found several areas of improvement as well as unforeseen stumbling blocks including data security, changing specifications, funding issues, re-rendering issues, and others that led to considerable changes to the product specification, and the methods used to implement that specification. The team had to switch directions several times, especially the backend who experimented with several solutions to the development issues, in the process of development when there were issues leading to unforeseen delays and shifts in the project specification however even still the group believes that the final project made considerable progress towards meeting the expectations of both the client and the development team.

4.4 Technology Considerations

Technology	Strengths	Weaknesses	Alternative Technologies
IntelliJ Idea	• High functionality	• High Complexity	• Visual studio

IDE	• Able to develop in multiple languages • Albe to run and test many languages • Highly modifiable • Has built in GIT functionality	• Power intensive • CPU intensive	code • Android Studio • Xcode
React Native	• Fast execution for web and mobile applications • Multifunctional • Works well with Python	• Unfamiliar to the group • High learning curve	• Java • C# • Swift • Ionic • Xamarin • JavaScript
Python	• Easy to learn • High functionality • Low complexity • Usable with many APIs including the ones necessary for this project • Usable with React Native and Google Cloud	• Reletively unfamiliar to the group • Low memory management • Very little internal control over the language • Reletively slow in comparison to some other languages	• Java • C# • C++ • C • Ruby • GO • Javascript •
Google Cloud	• Low latency • High uptime • Automatic backup • High functionality • Extremely Adaptable • Automatinc database management • Automatic patching • Automatic updates	• Unfamiliar to the group • External (group has little internal control) • Difficult pricing models • Overwhelming documentation • Difficult to influence with outside applications	• Microsoft Azure • AWS • MySQL • SQL Server • Microsoft Access • ISU Servers
Gitlab	• Easy to use • Linked to ISU • Easy to use visual GIT UI • Automatic record keeping for metrics/contributions • Inclues task list • Includes CI/CD for integration testing	• N/A	• Github • Git console

4.5 Design Analysis

During the development process we ran into several issues that required shifts in the design and methodologies that were required to progress in the project development, we switched up the frameworks for, but using the methodologies of the AGILE process we were able to pivot and continue development with little interruption in the process, in general we did follow the design plan partially including beginning with page connectivity and processing, however the team split in two and concurrently worked on both data storage and the UI/UX and navigation then regrouped to work on connection between the database and the UI/UX functionality. We also attempted to keep in mind the constraints of the application in order to make the most refined version of the application the group was capable of.

4.6 Design Plan

We plan to design starting from the most vital features for the user experience first beginning with the dashboard and login, then moving on to the other main pages including the task list, and goals, ending with the user settings last. From these main pages we can further develop into more complexity and functionality, however the plan is to first have a minimum viable product that can be expanded into a more fully functional application as time progresses and the group after us takes over.

4.7 Security Concerns and Mitigations

It was vital to the team to ensure the security of user data stored in the database especially sensitive information such as user email addresses and passwords, in pursuit of this as well as forward scalability the team decided use Google Cloud as a backend and data storage server because all data stored in a Google ColudSQL database is encrypted using Google's encryption at rest and extends to information sent from the server using Google's encryption in transit. The backend team also decided against the original implementation of the backend relying on the frontend having access to the user email address in order to filter tables for relevant information, instead switching to using a unique identifier for each user, that if found would not give any compromising data not available actively to the front end UI. Compromising user data is at no point returned to the front end where it could be easily accerssed, as all user based communication is performed with standardised HTTP response codes, as well as informative response messages that do not compromise user information.

5 Testing

5.1 Unit Testing

Our units for this project are our UI pages, along with the backend support for each. For each page, we will test the data entry and retrieval using Postman. We will also be making sure each page, overlay, and all of their internal components are smoothly interconnected through manual usage.

5.2 Interface Testing

The interfaces used in the design are the UI pages, this includes the login page, dashboard, task list, calendar, and mental health pages. The user interface will go through manual testing through running the application as well as junit tests that will test for expected behavior and function of the application.

5.3 Integration Testing

The overall most critical path in our software is the communication between the front end application the user will use, and the cloud service that will be handling server duties. Our group will be using postman to test the integration between backend and frontend to simulate data entry, our group will also rigorously test the software through manual usage of the software.

5.4 System Testing

Similarly to our method of integration testing, we'll be using Postman to monitor and keep track of entered data and ensure that data is not lost upon leaving the page it was entered on. Rigorous testing of the flow of the UIs will also be utilized to make sure that all pages have continuity between each other.

5.5 Regression Testing

App flow and the basic functionalities of the app are vital in ensuring that any other functionality doesn't break. Our group will be using gitlab's CI/CD to ensure there are no issues with adding in new code and regression testing in regards to new contributions.

5.6 Acceptance Testing

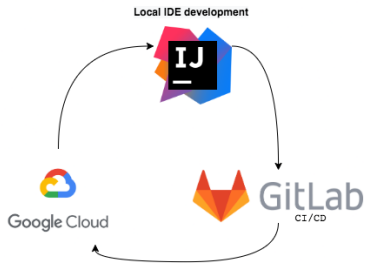
Jewels Academy wishes for a trial of testing the app through some of their students to collect feedback on what does work, what doesn't work, and what other features are or are not needed or utilized.

5.7 Security Testing (if applicable)

Once we set up our server or cloud on the backend we will look for proper strategies into increasing security throughout our application. For example, we are looking to do penetration testing on different within the application, such as wireless, network, firewall, and web application

5.8 Results

In addition to regular development testing, our team will use GitLab's CI/CD in order to ensure that our application works seamlessly on Google Cloud. By using this feature, we'll be able to make sure that each iteration of our development allows our application to work as intended before being deployed.



6 Implementation

The team created the application simultaneously as two split teams one working on UI/UX development while the second group worked on data storage and endpoint development. The team realized the original proposed design left the team with an unsatisfactory model of the final product, so the methods needed to be switched away from the original monolithic code base methodology into a more modular code base and in doing so switched from PyMySQL to Flask using a REST API backend that was more implementable than the previous model which did not allow for reasonable connectivity between the front and back end of the application, in making this pivot however much of the initial progress on the application was lost and the design needed to adapt to the more updated specification. There was an initial idea to use the Django web framework

however after some experimentation the team decided that Django was too multifunctional, allowing for far more than was necessary for this project, and complexity to match, so instead it was decided to implement the Flask framework. The original languages of python and React did not change however there were added frameworks in the form of Formik and YUM on the frontend as well as Flask, Marshmallow, and SQLAlchemy on the frontend as well as the initial frameworks of JSX to make javascript more object oriented, as well as EXPO for testing the UI/UX design.

7 Professionalism

7.1 Areas of Responsibility

Area of responsibility	Definition	NSPE Cannon	SE code of ethics
Work Competence	Perform work of high quality, integrity, timeliness, and professional competence.	Perform services only in areas of their competence; Avoid deceptive acts.	Principle 6: Profession. Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.
Financial Responsibility	Deliver products and services of realizable value and at reasonable costs.	Act for each employer or client as faithful agents or trustees.	Principle 2: Client and employer. Software engineers shall act in a manner that is in the best interests of their client and employer, consistent with the public interest.
Communication Honesty	Report work truthfully, without deception, and are understandable to stakeholders.	Issue public statements only in an objective and truthful manner; Avoid deceptive acts.	Principle 7: Colleagues. Software engineers shall be fair to and supportive of their col-leagues.
Health, Safety, Well-Being	Minimize risks to safety, health, and	Hold paramount the safety, health, and welfare of the public.	Principle 5: Management. Software engineering

	well-being of stakeholders.		managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.
Property Ownership	Respect property, ideas, and information of clients and others.	Act for each employer or client as faithful agents or trustees.	Principle 3: Product. Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.
Sustainability	Protect the environment and natural resources locally and globally.	Adhere to principles of sustainable development to protect the environment for future generations	Principle 8: Self Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.
Social Responsibility	Produce products and services that benefit society and communities.	Conduct themselves honorably, responsibly, ethically, and lawfully so as to enhance the honor, reputation, and usefulness of the profession.	Principle 1: Software engineers shall act consistently with public interest.

7.2 Project Specific Professional Responsibility Areas

- Work Competence:
    - We agree that this applies to our project because our clients expect professional and organized work. Whether that be the program itself, our communication on the project throughout the team. The professional communication with the client is a High priority when it comes to business in the software industry.
- Financial Responsibility:

    - This criteria applies to our project in a professional context because our client will be providing any necessary expenses that are required for us to create their application. In this area, our team is performing high because there has not yet been any monetary transactions between our group and the client.

- Communication Honesty:

    - This area of responsibility is vital for our project in a professional context as cohesive, honest communication is needed in order to streamline and clarify the process in which the client wants us to complete the deliverables for this project. This also allows us to discuss potential and needed features clearly without any tensions or doubts.

- Health, Safety, Well-being:

    - This area of responsibility doesn't apply to our group's project because there aren't any health/safety risks that are associated with developing our application. The performance of our team in this aspect of the project is not applicable because the vast majority of our work and communication will be done using a computer.

- Property Ownership:

    - This is something that applies to our project. As we create the application our team wants to take ownership and output the best possible application we can for our client. This means taking responsibility for our product and continuing professionalism with creating the application. Our team is performing this at a High rate due to the fact that we care about our product and want to help out our client as much as possible with the knowledge we have gained over the course of our years.

- Sustainability:

    - This area of professional responsibility does not correlate with our project as much as the other areas. Our project will be entirely set in a virtual environment with no impact to the outside environment. This area is not applicable to our project.

- Social Responsibility:

    - This professional responsibility area is key to our project's success because we will be creating an application that supports a better life-style for the community we are creating it for. This is a high priority for this project because this area will be one of the main focuses and purposes of this application. Our team is performing high in this criteria.

7.3 Most Applicable Professional Responsibility Area

Property ownership is an important area of responsibility to our project because each of our team members shares the goal of producing a product that is high quality and exceeds the expectations of our stakeholders. We've demonstrated a high level of proficiency in this area by staying continuously engaged with our client, our faculty advisor, and each other. Each member of our team is an aspiring software engineer and therefore we have a mutual understanding that

delivering a high quality product will be in the best interest of developing our educational skill set, and a professional responsibility in the future. Our group has spent a considerable amount of time and effort with primary and secondary research in accordance with this professional responsibility area, that research has allowed our group to get to a much more refined product than if the group were to work without considering that we want the final product to reach its highest quality given our skills.

8 Closing Material

8.1 Discussion

The main result of our project will be a completed mobile application, which will not involve the same type of experimentation as a project in the electrical engineering, or computer engineering fields will as this project is completely software based. The experimentation that our group will be doing is impermant and can easily be iterated upon, as there is no physical construction and software development is extremely malleable. If all the requirements are met we will be left with a high efficiency, visually appealing, and multifunctional application with features ranging from time management to goal setting and engagement with the user in progression towards their personal goals and tasks. This product will be easy to use for students as young as 9 years old using visual description and a quality UI/UX and will integrate with popular services to increase usability including Google services and others.

8.2 Related Products or literature

Our app will be one of many applications to help manage time, however it will be more narrow and specialized being specifically for Jewels Academy and its students. Having a specific base means that our product will in general be more specific in meeting the requirements of the shareholders because it does not need to apply to a vast audience, and with this refined scope the group could put more time and effort into the design and user experience tailored for the client and their students.

8.3 Conclusion

From the development experience through this application we experienced many triumphs as well as tribulations. We ended up with a product with considerable polish and refinement, however in the course of that, with input from our industry client, some of the proposed features were left behind in the ideation stage in pursuit of a refined final product even if the breadth of the project was slimmed. We have a functional frontend and backend tunning on local machines, though there is still some issue with connection and the google cloud, as well as some concerns from the client about the clouds cost effectiveness given the issues with considerable price jumps the group experienced over the summer between semesters of development. On the backend front there is a fully functional local database including Flask REST endpoints for http request routing, on the front end there is great progress on both the implementation of http requests as well as a nearly fully styled and functional UI and navigation layout. The group decided to continue development with both Python and Javascript, however the frameworks used by the group both frontend and backend changed multiple times as integration or development issues were found in the process of the application creation.

8.3 References

[1] "SQLAlchemy - The Database Toolkit for Python", *Sqlalchemy.org*, 2022. [Online]. Available: https://www.sqlalchemy.org/. [Accessed: 23- Apr- 2022].

[2]"Google Cloud documentation | Documentation", *Google Cloud*, 2022. [Online]. Available: https://cloud.google.com/docs/. [Accessed: 23- Apr- 2022].

[3]"GitHub - GoogleCloudPlatform/cloud-sql-python-connector: Cloud SQL connector for Python", *GitHub*, 2022. [Online]. Available: https://github.com/GoogleCloudPlatform/cloud-sql-python-connector#how-to-use-this-connector. [Accessed: 23- Apr- 2022].

[4]"API Reference | Calendar API | Google Developers", *Google Developers*, 2021. [Online]. Available: https://developers.google.com/calendar/api/v3/reference. [Accessed: 23- Apr- 2022].

[5]"Outlook calendar API overview - Microsoft Graph", *Docs.microsoft.com*, 2021. [Online]. Available: https://docs.microsoft.com/en-us/graph/outlook-calendar-concept-overview. [Accessed: 23- Apr- 2022].

[6]"Using the Trello API", *Trello.org*. [Online]. Available: http://www.trello.org/help.html. [Accessed: 23- Apr- 2022].

[7]"Canvas LMS REST API Documentation", *Canvas.instructure.com*, 2022. [Online]. Available: https://canvas.instructure.com/doc/api/index.html. [Accessed: 23- Apr- 2022].

[8]"Welcome to Python.org", *Python.org*, 2001. [Online]. Available: https://www.python.org/doc/. [Accessed: 23- Apr- 2022].

[9]“OpenAPI Specification - Version 3.0.3 | Swagger.” API Documentation & Design Tools for Teams | Swagger, https://swagger.io/specification/. Accessed 6 Dec. 2022.

[10]“Python Venv: How To Create, Activate, Deactivate, And Delete • Python Land Tutorial.” Python Land, https://python.land/virtual-environments/virtualenv. Accessed 6 Dec. 2022.

[11]“Flask-Testing — Flask-Testing 0.3 Documentation.” PyPI Package and Documentation Storage, https://pythonhosted.org/Flask-Testing/. Accessed 6 Dec. 2022.

[12]“Pipreqs · PyPI.” PyPI, https://pypi.org/project/pipreqs/. Accessed 6 Dec. 2022.

[13]“Welcome to Flask — Flask Documentation (2.2.x).” Welcome to Flask — Flask Documentation (2.2.x), https://flask.palletsprojects.com/en/2.2.x/. Accessed 6 Dec. 2022.

9 Appendices

9.1 Appendix 1: Operation Manual

9.1.0 Environment Setup

1. Install your IDE of choice to begin setting up the app (we recommend IntelliJ or Visual Studio Code)
2. Using the terminal, find your way to the project's location, labelled sddec22-06
3. Ensure that Node and npm are installed by running their corresponding version commands: "node -v" and "npm -v"
4. Most, if not all libraries should come preinstalled
    a. If this is not the case, use npm to install these libraries

9.1.1 Expo Setup

1. Navigate to the project's frontend by utilizing the following command: 'cd .\src\JA-Time-Management
2. Install Expo to your project by running the following command: 'Npm install -g expo-cli'
3. Install the Expo Go app from any app store to whatever device you wish to run the app on.
4. Ensure that both the device with Expo Go and the computer running Expo are on the same Wi-Fi connection
5. Run the app by entering 'expo start' into the terminal
6. After a small period of time, a QR code will appear in the terminal. Scan this QR code with the Expo Go app

7. Once the QR code is scanned, the app will begin to boot up on your device.

9.1.2 Running the App

1. Once the app has booted up, you will see the Start screen. To begin as a new user, head to the Register screen by hitting the corresponding button.
2. Once inside the Register screen, input all information to log yourself as a new user. It will then boot you back to the Start screen to log in.
3. Head to the Login screen and enter the same information as you did in the Register screen to be able to proceed further into the app.
4. You will then be sent to the Home screen, which has a few avenues where you can enter. You can do 5 different things here:
    a. Navigate to the Calender/To-Do screen via the bottom tabs
    b. Navigate to the Goals screen via the bottom tabs
    c. Navigate to the Settings screen via the cog icon in the top right corner
    d. As the Home screen is simply the mobile version of the Jewels Academy website, you can navigate through the website as normal

9.1.3 The Calendar/To-Do Screen

1. Navigating to this screen shows you an interactive calendar and list maker combo. To add a task to the list, press the '+' button in the upper right corner.
2. This will navigate you to the Add Task screen, which asks for and requires a title to name the task, a description of the task, and a date in which the task is due. The date is set using a button which leads to an interactive date and time setter.
    a. Days can be selected in the calendar before hitting the '+' button to preset which date is passed into adding a task.
3. Once complete, hit the 'Add Task' button to return to the Calendar/To-Do screen. The new task should show up at the bottom of the to-do list.
4. To edit a task, hit the 'Edit' button corresponding to the task you want to change. You will be brought to a similar screen to the Add Task screen, except all values are prefilled with the data of the task you're editing.
5. Once complete, hit the 'Edit Task' button to return to the Calendar/To-Do screen. The edited task should show up in the same place as the old task.
6. To exit out of the Calendar/To-Do screen, press either the 'Home' or 'Goals' buttons to navigate to their respective screens.

9.1.4 The Goals Screen

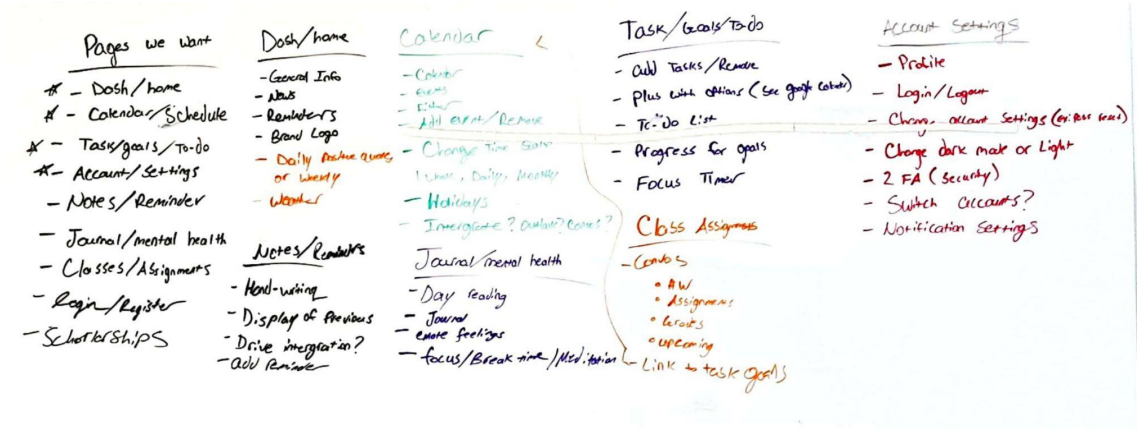
1. Navigating to this screen shows you a list of any and all goals you wish to achieve. To add a goal to the list, press the '+' button in the upper right corner.
2. This will navigate you to the Add Goal screen, which asks for and requires a title to name the goal, a description of the goal, and a date in which the goal is due. The date is set using a button which leads to an interactive date and time setter.
3. Once complete, hit the 'Add Goal' button to return to the Goals screen. The new goal should show up at the bottom of the list.
4. To edit a goal, hit the 'Edit' button corresponding to the goal you want to change. You will be brought to a similar screen to the Add Goal screen, except all values are prefilled with the data of the goal you're editing.
5. To remove or complete a goal, hit the checkmark box next to the corresponding goal you want to remove or complete. This will remove that goal from the list entirely.
6. To exit out of the Goals screen, press either the 'Home' or 'Calendar/To-Do' buttons to navigate to their respective screens.

9.1.5 The Settings Screen

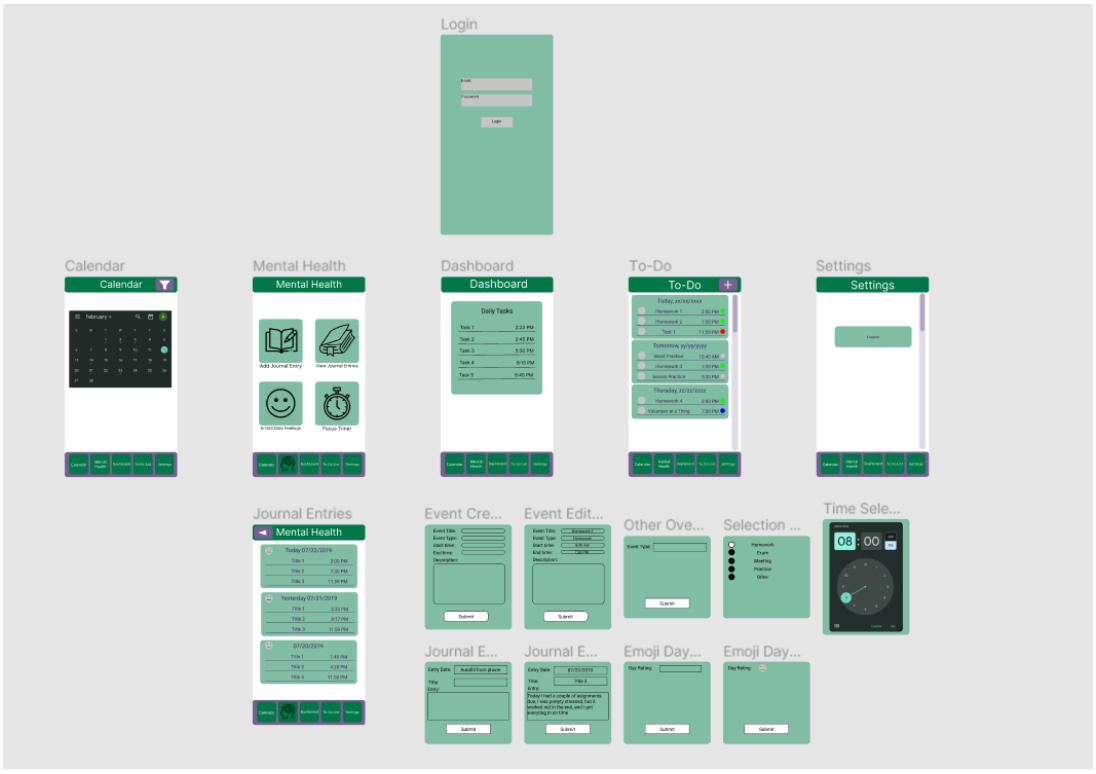
1. Navigating to this screen shows you a list of settings a user can set on their account.
2. The first option, 'Notification Settings' simply contains a switch turning on or off notifications from the app itself.
3. The second option, 'Theme Settings' can change the theme of the app based on hex color codes (i.e. #00FFFF). There is a primary and secondary color utilized across the app.
4. The third option, 'Log Out' will simply log the user out and boot them to the Start screen
5. To return to the Home Screen, hit the 'Back' button in the top left corner.

9.2 Appendix 2: Previous Versions

Over the course of this semester, there have been many changes since the original design. Originally, our design looked like this:



Scanned with CamScanner



However, over many meetings with our client, Jewels Academy, we have worked to whittle down what we did and didn't want. The largest change in design was when we completed a demo for Jewels Academy in early November. The scope was changed such that the To-Do List and Calendar pages are combined, the Mental Health page is removed, and the Goals page was added into our current design. As the Mental Health page at this point was not worked on much, along with the similarity of the To-Do List and Goals pages, this was not as huge a setback as it could have been.

9.3 Appendix 3: Other Considerations

Throughout the implementation process of this app, there were a variety of different lessons learned. The largest factor of learning was simply setting up a brand new project from scrap, finding the pieces that will fit together, and building off of that. We had extreme difficulties with several aspects of each side of the project, resulting in significant delays. The backend suffered two separate rebuilds and several issues with how Google Cloud handled things, and the frontend dealt with a variety of issues, such as rendering, data storage, and navigation. From this, we have learned quite a bit regarding how to be adaptable to the circumstances the group found themselves in as well as the ability to ideate design and implement ever-changing solutions to different issues that come up in the process. Had we had more time to get past these issues and continue work, there would have been more diversification between the To-Do List and the Goals components. Additionally, more features would have been added across the app, such as Federated Identity, 2 factor authentication, task and goal categorization, and of course, less bugs.

Additionally, there were a time or two where we found a library or resource that would have made all of our lives easier had we known about them earlier. The prime example of this would be Firebase. Firebase is a platform for developing apps with a stable backend support, including cloud support, that is directly backed from Google. Had we known about this software, we would have had a much, much easier time developing Jewels Academy's app. This teaches us how to fully research *all* available resources that we can before diving into development.

As time got away from us due to the outstanding issues discussed before, we have been notified that this project will be passed onto another senior design team. Thus, we have been putting a more significant amount of effort documenting just about everything in our code. Learning how to document as you go and staying on top of it is important to reduce the amount of time this takes, and it would have saved us some time had we had full documentation from the get-go.