# Project Plan

## Glucose Genie

Francisco Cruz-Urbanc, Krisi Hristova, Carson Ford, Thomas Capro, Jared Jackson

CI 493 – Senior Project

Filippos Vokolos

April 6, 2025

Version 4

This Project Plan aims to define the tasks needed to be fulfilled for the Glucose Genie project, identify who will be responsible for each task, and provide a date by which each task will be completed.

## Project Background:

Diabetes management, particularly for low-income and Spanish-speaking patients, requires careful monitoring of nutrient intake, especially carbohydrates. Many patients lack access to tools needed to simplify the process of meal planning and carb counting. In addition, balancing nutrition with financial constraints when grocery shopping is a significant challenge.

The Glucose Genie application will assist medical professionals in tackling this issue by providing users access to a tailored database of diabetes friendly recipes where carbohydrates and other important macro and micronutrients are already calculated to allow users an intuitive way to track and create weekly meal plans. A detailed definition of what is considered diabetes friendly recipes is mentioned in *Section 3 – Features to be implemented* of the [Customer Requirements Document.](#)

This mobile iOS application also allows users to create personalized weekly meal plans and generate grocery lists as recipes are added. Users can also edit their grocery lists by adding or removing items they are not interested in purchasing.

Overall, the Glucose Genie application is targeted to aid patients in tracking important nutrients, creating weekly meal plans through diabetic friendly recipes and providing user-friendly ways to purchase meal ingredients to help individuals sustain a diabetic friendly diet. A more extensive list of features of the Glucose Genie application is included in the **Customer Requirements Document:** *Section 3-Features to be implemented* found on the Project's GitHub Repository [here](#).

## Statement of Work:

The statement of work defines a list of the products of work that will be produced during the project and the people who will perform them.

- **User Manual:** Thomas
  - Manual to include an installation guide, step-by-step instructions on navigation within the app, troubleshooting procedures, and FAQ.

- **Customer Requirements:** Everyone (Carson, Thomas, Jared, Francisco, Krisi)
  - Document of customer requirements outlining the 1-*Project Background*, 2-*Users*, 3-*Features to be implemented*, and 4-*Features not to be implemented* as communicated by the stakeholders to ensure a clear goal of the product and reduce miscommunication.

- **Software Requirements Specification**: Everyone (Carson, Thomas, Jared, Francisco, Krisi)
  - Outline of the technical features, functionalities and requirements of the app to define how features will be implemented, what software and technologies will be used and define expected performance of the product.

- **System Architecture Diagram:** Carson
  - Visual representation of the app's structure and overall system as well as the relationships between all components. This is important so we can keep track of all elements the app uses and make it easier for developers to understand how the system works.

- **API Documentation**: Carson
  - For each API we choose to use we will have documentation outlining why we are using it, how it is implemented, request and response formats, and authentication methods. This will help other developers understand how the API is used.

- **Code:** Everyone (Carson, Thomas, Jared, Francisco, Krisi)
  - Codebase including backend and frontend for the iOS application.

- **Test Plans:** Francisco
  - This will outline the strategy and resources for testing our app. It will detail the types of tests done, list the tests case by case, and their expected results. This is needed as we want our app to be defect free.

- **Defect Reports:** Jared & Francisco
  - This will keep a record of any bugs found. Each bug should have a description, steps to reproduce, severity level, and when/if it was fixed. They should each be labeled correspondingly to the test plan.

- **Privacy Policy:** Thomas & Krisi
  - We must ensure that we only store information about the quantities of nutrients consumed, as determined by the planner or recipes. These quantities should be measured using the standard nutrient recommendations for diabetics. A clear policy will be provided to users, outlining how we adhere to these guidelines. This approach is essential to maintain user trust in our app and services while demonstrating compliance with applicable regulations.

- **Logo:** Krisi
  - A logo approved by the stakeholders for the product representing Glucose Genie.

## Resource List:

A list of all resources that will be needed for the product and their availability.

- **Recipe API:** We will begin with using the Enterprise Basic Subscription, and if the application becomes production ready (and we begin to receive a larger amount of API requests because of this), then we can upgrade to the Enterprise Core Subscription. As the subscription plans can change (there used to be a fully free subscription with Edamam), we will maintain a back-up recipe API, Spoonacular, that includes a free subscription in the case that costs become an issue.
  - **Edamam Recipe API:**
    - **Enterprise Basic Subscription** includes recipe data (image, ingredients, title), meal type and cuisine filters, diet/health/allergy filters, measure and quantity for ingredients, recipe caching (recipe id and name), shopping aisle for ingredients, full nutritional details, and up to 100 results/call.
      - Monthly fee: $9, 30-day free trial
      - Included API calls: 10,000/month, 10/minute
      - Commercial use: NO

- **Enterprise Core Subscription** includes recipe data (image, ingredients, title), meal type and cuisine filters, diet/health/allergy filters, measure and quantity for ingredients, **r**ecipe caching (recipe id and name, protein, net carbs, total fat, and Kcal), shopping aisle for ingredients, full nutritional details, and up to 1000 results/call.
  - Monthly fee: $69
  - Included API calls: 100,000/month, 100/minute
  - Commercial use: YES
- **Spoonacular API**: provides nutritional data/analysis, price and cost breakdown, health information, ingredient substitutions, recipe management, and meal planning.
  - **Free Subscription**
    - Monthly fee: FREE
    - Included points (each request costs a certain number of points): 150 points/day then no more calls, 1 request/second
  - **Cook Subscription**
    - Monthly fee: $29
    - Included points (each request costs a certain number of points): 1,500 points/day then $0.005 dollars/point, 5 requests/second
  - **Culinarian Subscription**
    - Monthly fee: $79
    - Included points (each request costs a certain number of points): 4,500 points/day then $0.004 dollars/point, 10 requests/second
- **Map API:** We have not made an official decision on which Map API to use yet. A decision will be made during the development phase when we will take a deeper dive into each of the following services to find out which ones provide the exact information that we will need for the required features.
  - **Google Maps Platform:**
    - Can provide necessary information for nearby grocery stores, pricing details, routes, and more if needed.
    - Pay-as-you-go model: $200/month + additional costs based on number of requests for specific services within Google Maps Platform
  - **Mapbox:**
    - Can provide directions, virtual map, and search for points of interest (ours being grocery stores).
    - Pay-as-you-go model: FREE + additional costs if we exceed number of requests or active users for specific services within Mapbox. Relevant services include:
      - **Directions API:** FREE under 100,000 monthly requests

- **Maps SDKs for Mobile:** FREE under 25,000 monthly active users
- **Search Box API – Requests:** FREE under 50,000 monthly requests
- **Temporary Geocoding API:** FREE under 100,000 monthly requests
    - **Amazon Location Services:**
        - Can provide the following location features:
            - Map tiles retrieved (Vector or Raster) (500 thousand per month)
            - Address suggestion requests (10 thousand per month)
            - Addresses geocoded (10 thousand per month)
            - Positions reverse-geocoded (10 thousand per month)
            - Number of origins (1)
            - Number of destinations (5)
            - Matrix route calls (2 thousand per month)
        - This will be covered by Drexel.
- **Secure database system:** The database system we will be using is Amazon DynamoDB. It is a serverless, NoSQL database that will help provide high performance to users. The database will serve to store shopping/ingredients lists, weekly meal plans, and nutrient counts.

## Assumptions:

Below is a list of assumptions that the stakeholders, the project team and users have made.

**Stakeholder Assumptions**
- App will provide accurate recipes and user data.
- App will be in a functional prototype state by June 2025.
- Team members will communicate any questions they have via the text group chat.
- Team members will keep stakeholders up to date on the project schedule, delays, and other roadblocks.
- App will comply with healthcare and privacy regulations (HIPPA).

**Project Team Assumptions**
- Members will be able to quickly communicate information with stakeholders via text group chat.
- Third-party services used in the app (ex APIs) require an internet connection.
- Third-party services we use to communicate with each other and stakeholders will remain online and accessible (SMS, Discord, Microsoft Online, Email).

- Members will inform the team if they encounter any roadblocks that might potentially delay the project.
- App will meet the stakeholder's priority requirements by the end of the project.
- Data used in the app from APIs will be accessible and up to date.
- Any bugs or inconsistencies will be addressed during the testing phase of the design project.

**<u>User Assumptions</u>**
- App will be easy for non-tech-savvy to navigate.
- App will save their account information (login credentials, meal schedules, grocery lists).
- App will function properly if they have a stable network connection (back-end technologies will be online and accessible).
- App will inform them if an error occurs (ex. Click add recipe to meal plan, error occurs causing recipe to not be added to meal plan, app lets user know that the recipe was not added to meal plan).
- App will be responsive (ex. Click remove recipe from meal plan, recipe will be removed from meal plan changing the icon to a blank slot).

## Project Schedule:

The project schedule created is in the format of a GANTT chart in Excel and a link to the most updated version of the schedule is provided.
The most recent version of the GANTT chart project schedule can be viewed **here.**

Photos of project schedule as of 4/06/2025:

# Glucose Genie (Group 2)

**Diabetic Meal Planner**                                          Legend:

**Thomas Capro, Francisco Cruz-Urbanc, Carson Ford, Krisi Hristova, Jared Jackson**

Project start date:              9/27/2024

Scrolling increment:            5

| Milestone description | | | Assigned to | | Progress | | Start | Day | End |
|---|---|---|---|---|---|---|---|---|---|
| Project background | | On Track | Krisi | | 100% | | 9/30/2024 | 15 | 10/14/2024 |
| Statement of Work | | On Track | Carson | | 100% | | 9/30/2024 | 15 | 10/14/2024 |
| Resource List | | On Track | Francisco | | 100% | | 9/30/2024 | 13 | 10/13/2024 |
| Research Recipe APIs | | On Track | Francisco | | 100% | | 10/7/2024 | 6 | 10/13/2024 |
| Research Map APIs | | On Track | Francisco | | 100% | | 10/7/2024 | 6 | 10/13/2024 |
| Research database systems | | On Track | Francisco | | 100% | | 10/7/2024 | 6 | 10/13/2024 |
| Assumptions | | On Track | Jared | | 100% | | 9/30/2024 | 14 | 10/14/2024 |
| Project Schedule | | On Track | Everyone | | 100% | | 9/27/2024 | 17 | 11/6/2024 |
| Importances | | On Track | Thomas | | 100% | | 9/30/2024 | 14 | 10/14/2024 |
| Stakeholder Meeting 1 + Notes | | On Track | Everyone | | 100% | | 10/2/2024 | 2 | 10/4/2024 |
| Research iOS Development on Mobile | | On Track | Thomas | | 100% | | 10/7/2024 | 43 | 11/18/2024 |
| Review Document and Submit | | On Track | Everyone | | 100% | | 10/13/2024 | 4 | 10/16/2024 |

## Customer Requirements

| | | | | | | |
|---|---|---|---|---|---|---|
| Project background | On Track | Carson | 100% | 10/14/2024 | 14 | 10/28/2024 |
| Users | On Track | Thomas | 100% | 10/14/2024 | 14 | 10/28/2024 |
| Features to Implement | On Track | Krisi & Francisco | 100% | 10/14/2024 | 14 | 10/28/2024 |
| Features *not* to implement | On Track | Jared | 100% | 10/14/2024 | 14 | 10/28/2024 |
| Intellectual Rights | On Track | Krisi | 100% | 10/21/2024 | 28 | 11/17/2024 |

## Project Logistics

| | | | | | | |
|---|---|---|---|---|---|---|
| Create Project Github Repo | Low Importance | Francisco | 100% | 10/21/2024 | 7 | 10/28/2024 |
| Calculate AWS usage | High Importance | Krisi & Francisco | 100% | 10/21/2024 | 7 | 10/28/2024 |
| Indiviuals create accounts with pfp | High Importance | Everyone | 100% | 10/21/2024 | 7 | 10/28/2024 |
| Create WebPage for senior project | High Importance | Krisi | 100% | 10/21/2024 | 7 | 10/28/2024 |
| End of Term Presentation - Fall | High Importance | Everyone | 100% | 11/25/2024 | 7 | 12/2/2024 |
| Create AWS Accounts | High Importance | Everyone | 100% | 12/25/2024 | 7 | 12/9/2024 |

## Project Plan

| | | | | | | |
|---|---|---|---|---|---|---|
| Submit baselined Project Plan | Med Importance | Everyone | 100% | 10/28/2024 | 7 | 11/6/2024 |
| Create Project Plan Cover Page | Med Importance | Krisi | 100% | 10/21/2024 | 10 | 11/6/2024 |

## Software Requirements Specification

| | Status | Owner | Progress | Start Date | Days | End Date |
|---|---|---|---|---|---|---|
| Introduction | On Track | Thomas | 100% | 11/4/2024 | 23 | 11/27/2024 |
| Description | On Track | Krisi | 100% | 11/4/2024 | 23 | 11/27/2024 |
| Nonfunctional Requirements | On Track | Francisco | 100% | 11/4/2024 | 23 | 11/27/2024 |
| Functional Requirements | On Track | Carson | 100% | 11/4/2024 | 23 | 11/27/2024 |
| External Interface Requirments | On Track | Jared | 100% | 11/4/2024 | 23 | 11/27/2024 |
| Appendix/Glossary | On Track | Everyone | 100% | 11/4/2024 | 23 | 11/27/2024 |

### Design

| | Status | Owner | Progress | Start Date | Days | End Date |
|---|---|---|---|---|---|---|
| Logo | On Track | Krisi | 100% | 10/28/2024 | 29 | 11/29/2024 |
| Create home page designs | Low Importance | Jared | 100% | 1/6/2025 | 21 | 1/27/2025 |
| SoftwareDesignDoc Submission | High Importance | Krisi | 100% | 1/20/2025 | 16 | 2/5/2025 |
| SDD Intro | Med Importance | Francisco | 100% | 1/20/2025 | 16 | 2/5/2025 |
| SDD Design Overview | Med Importance | Krisi | 100% | 1/20/2025 | 16 | 2/5/2025 |
| System Architecture Diagram | Med Importance | Carson | 100% | 1/20/2025 | 23 | 2/5/2025 |
| System Operation/Sequence Diagram | Med Importance | Krisi | 100% | 1/13/2025 | 23 | 2/5/2025 |
| SDD Requirements Tracability | Med Importance | Thomas | 100% | 1/20/2025 | 16 | 2/5/2025 |
| SDD User Interface (UML) | High Importance | Francisco | 100% | 1/20/2025 | 16 | 2/5/2025 |
| SDD Data Model Diagram | High Importance | Jared | 100% | 1/20/2025 | 16 | 2/5/2025 |
| SDD Storage | High Importance | Thomas | 100% | 1/20/2025 | 16 | 2/5/2025 |
| SDD Doc Formatting | On Track | Krisi | 100% | 1/20/2025 | 16 | 2/5/2025 |

## Software Implementation

| Task | Importance | Assigned | Progress | Start | Days | End |
|---|---|---|---|---|---|---|
| Setup Development Environment | Low Importance | Everyone | 100% | 1/20/2025 | 21 | 2/10/2025 |
| Learn and get comfortable with Swift | Med Importance | Krisi, Francisco, Carson, Jared | 100% | 2/5/2025 | 12 | 2/17/2025 |
| Create intial project | High Importance | Thomas | 100% | 2/17/2025 | 2 | 2/18/2025 |
| GLUC-001- Create classes outlined in Design Document and upload to github | High Importance | Thomas | 100% | 2/18/2025 | 6 | 3/3/2025 |
| GLUC-002- Create UI for user settings page | Med Importance | Carson | 100% | 2/18/2025 | 14 | 3/3/2025 |
| GLUC-003-Create Individual Recipe Page | High Importance | Thomas | 0% | 4/7/2025 | 49 | 5/25/2025 |
| GLUC-004-Implement Login/Out Functionality | High Importance | Francisco | 100% | 2/18/2025 | 14 | 3/2/2025 |
| GLUC-005-Implement Main Page Back End-AWS | High Importance | Jared | 100% | 2/18/2025 | 20 | 3/7/2025 |
| GLUC-006- Implement User Location | High Importance | Carson | 0% | 4/7/2025 | 7 | 4/14/2025 |
| GLUC-007- Implement Language Settings | High Importance | Francisco | 0% | 4/14/2025 | 22 | 5/5/2025 |
| GLUC-008- Implement Recipe Display UI | Med Importance | Krisi | 100% | 2/27/2025 | 12 | 3/10/2025 |
| GLUC-009- Implement Grocery List Page | High Importance | Jared | 0% | 4/7/2025 | 22 | 4/28/2025 |
| GLUC-011-Implement Main Page UI | Med Importance | Krisi | 100% | 2/18/2025 | 9 | 2/26/2025 |
| GLUC-012-Implement daily nutrient tracker | Med Importance | Carson | 100% | 3/4/2025 | 27 | 3/30/2025 |
| GLUC-013- Implement Recipe Searching | High Importance | Krisi | 15% | 3/31/2025 | 31 | 4/30/2025 |
| GLUC-014- Implement Weekly Meal Plan | High Importance | Jared | 0% | 4/7/2025 | 15 | 4/21/2025 |
| GLUC-015-Implement Grocery Store Page | High Importance | Carson | 0% | 4/7/2025 | 22 | 4/28/2025 |
| GLUC-010- Implement Recipe Saving | Low Importance | Anyone | 0% | 5/25/2025 | 11 | 6/4/2025 |
| GLUC-016- Create custom login UI | Low Importance | Anyone | 0% | 5/25/2025 | 11 | 6/4/2025 |
| GLUC-017-Implement user notifications | Low Importance | Anyone | 0% | 5/25/2025 | 11 | 6/4/2025 |
| GLUC-018-Add logo to application | Low Importance | Krisi | 100% | 3/6/2025 | 5 | 3/10/2025 |
| GLUC-019-Connect Nutrient Tracker to Database | Low Importance | Anyone | 0% | 5/25/2025 | 11 | 6/4/2025 |

## Testing

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Read Over Example Test Plans | Med Importance | Everyone | 0% | 4/7/2025 | 3 | 4/9/2025 |
| Test Plan - Intro | High Importance | Krisi | 0% | 4/9/2025 | 7 | 4/14/2025 |
| Test Plan - Approach to Testing | High Importance | Carson, Jared, Francisco | 0% | 4/15/2025 | 12 | 4/26/2025 |
| Test Plan - Test Cases | High Importance | Everyone | 0% | 4/7/2025 | 20 | 4/26/2025 |
| Test Plan - Risks and Assumptions | High Importance | Thomas | 0% | 4/20/2025 | 7 | 4/26/2025 |
| Test Features According to Test Plan | High Importance | Everyone | 0% | 5/11/2025 | 10 | 6/3/2025 |
| Test User Manual Against App | Low Importance | Everyone | 0% | 5/30/2025 | 5 | 6/3/2025 |

## Extra Documents

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| User Manual | Med Importance | Thomas | 0% | 5/5/2025 | 7 | 5/11/2025 |
| API Documentation | Low Importance | Carson | 0% | 4/14/2025 | 45 | 4/28/2025 |
| Defect Reports | Med Importance | Jared, Francisco | 0% | 5/11/2025 | 14 | 5/25/2025 |
| Privacy Policy | Low Importance | Krisi, Thomas | 0% | 5/26/2025 | 7 | 6/1/2025 |

# Risks:

Below is a list of risks that could potentially derail the project.

**Risks During Project Development**

- **Scope Creep:**
  - Risk: The project's scope could expand beyond initial plans due to additional stakeholder requests, leading to delays and missed deadlines.
  - Mitigation: Establish clear project requirements and implement a change management process to evaluate new requests before incorporating them.
- **Resource Shortages:**
  - Risk: The project may experience shortages in development resources, such as developers or testers, due to team members leaving or being reassigned.
  - Mitigation: Develop a staffing plan, maintain backup team members, and regularly assess team capacity.
- **Technical Challenges and Unfamiliar Technologies:**
  - Risk: Developers may face unexpected technical difficulties or struggle with unfamiliar tools, delaying progress.
  - Mitigation: Allocate time for research and prototyping and encourage early training on new tools and technologies.
- **Budget Constraints:**

- o Risk: Running out of funds could halt development, especially if unexpected costs arise (e.g., third-party service fees).
- o Mitigation: Create a detailed budget, regularly review expenses, and ensure stakeholders are informed about financial status.

- **Data Source Availability or Changes:**
  - o Risk: Third-party data sources and APIs for recipes or grocery store locations could become unavailable or change access policies, requiring adjustments.
  - o Mitigation: Identify alternative data sources during planning and design the app's architecture to accommodate data source changes.

**Potential Implications after Product Release**

- **User Compliance and Engagement:**
  - o Risk: Users may not engage with the app consistently, reducing its overall utility.
  - o Mitigation: Design an intuitive and user-friendly interface. Include engagement features like reminders and meal planning tools that align with the user's needs, while keeping the interface as simple as possible for the non-tech-savvy target audience.

- **Adoption by Target Audience:**
  - o Risk: The app's main target audience (low-income, non-tech-savvy, Spanish-speaking users) may find it difficult to use.
  - o Mitigation: Ensure the app is simple, with clear language options (English and Spanish) and accessible design features that are easy to navigate, such as large buttons and a straightforward layout.

- **Changing Nutritional Guidelines:**
  - o Risk: Nutritional guidelines may change over time, impacting the relevance of the app's database.
  - o Mitigation: Keep track of changes in general guidelines for diabetes-friendly diets and ensure the app can easily update its database and filter criteria as needed.

- **External Factors (Supply Chain or Economic Shifts):**
  - o Risk: Economic changes or supply chain disruptions could affect the availability or cost of ingredients, making the grocery list less accurate or helpful.
  - o Mitigation: Provide flexibility for users to edit their grocery lists and select alternate items based on availability. Allow for manual substitutions and location-based grocery store pricing updates.

- **Third-Party API or Service Dependency:**
  - o Risk: Reliance on external services (e.g., API for grocery store locations) could lead to disruptions if the API services become unavailable.
  - o Mitigation: Plan for backup data sources and alternative methods to fetch grocery store location information to reduce reliance on any one external provider.

- **User Misinterpretation of Information:**

- o Risk: Users might misinterpret nutritional information or carb counts, leading to incorrect meal planning and potentially harmful dietary decisions.
- o Mitigation: Ensure that all information is presented clearly and with simple language. Provide educational resources or FAQs explaining key concepts like carb counting and portion sizes and include disclaimers advising users to consult healthcare professionals for personalized guidance.
- **Language and Cultural Sensitivity:**
  - o Risk: Misinterpretation or poor translation of the app's content into Spanish (or other languages) may lead to confusion or alienate users.
  - o Mitigation: Either find a professional translator or a trusted translation source and conduct user testing with native Spanish speakers to ensure that language and cultural nuances are correctly handled. Include culturally appropriate recipes and recommendations that resonate with the target audience.
- **Limited Access to Technology:**
  - o Risk: The target audience (low-income users) may not have access to consistent internet, smartphones, or other necessary technology to use the app effectively, reducing adoption or long-term engagement.
  - o Mitigation: Design the app to work offline as much as possible (e.g., nutrient tracking) and minimize data usage. Consider optimizing the app for lower-end devices to make it accessible to users with older technology.
- **User Support and Troubleshooting:**
  - o Risk: Users may encounter issues or bugs while using the app and, if they don't receive adequate support, could abandon it altogether.
  - o Mitigation: Set up a simple, accessible support system (e.g., FAQs, help desk) to address user issues quickly. Offer tutorials and consider creating a feedback loop where users can report bugs or request new features.