

GlucoCheck

Submitted by:

Jeremiah Thomas, Computer Programming Student (Scrum Master)

Stuart Watson, Data Science Student (Product Owner)

Anjali Sharma, Data Science Student (Developer)

Elmehdi Mnajli, Data Science Student (Developer)

Kai McGregor, Computer Programming Student (Developer)

Instructor:

Susan Rizzo

Industry Mentor[s]:

Mike Fitzpatrick, IHS Markit Software Developer

William Calhoun, IHS Markit Executive Director

April 24, 2021

Submitted in partial fulfillment of the requirements for CSC 289 Capstone project

Programming & Information Sciences Department

Information Technology Division

Wake Technical Community College, Raleigh, NC 27709

1. Introduction

In our spring 2021 CSC 289 capstone class at Wake Tech, we were given a selection of various projects in order to test our ability to work as a team, showcase our knowledge, and implement industry best practices in the form of agile development. Some projects seemed to suit our prior experiences, but it was only when one of our group members opened up about a health condition—diabetes—that we rallied around a single idea: a glucose tracking app. We had access to both an SRS and HLD and walked through the documentation multiple times. But in the end, we translated many of the concepts into something that was more attuned to the data scientists in the group and created our own user stories, ones based on our needs just as much as the customer's. The GlucoCheck app was born.

1. Project Plan

All project plan documents are copied without change and reflect our initial assumptions and strategies.

* 1. Project Description

Team 6 will be creating GlucoCheck, a program to track glucose and insulin levels. This program will offer a secure, web-based, DB-backed UI for users to create an account, log in, and input health data. Users will be able to visualize this data, as well as receive different dietary recommendations based upon it.

* 1. Project Background

The biggest challenge will be providing a secure, easy-to-use interface for customers to be able to interpret their health data. While there is no shortage of tools to measure glucose, few of those tools provide data visualization or personalized dietary recommendations that allow the customer to go from numbers to knowledge. This will be a vital learning experience for the team because customers very rarely want data, or software, or even logins: they want concrete answers, simplifications, and convenience. By applying programming techniques to a real-world problem in the healthcare industry, members will be able to understand how to make use of their technical knowledge in a supporting role for a useful, profitable business.

* 1. Project Goals

• Enable my customers to sign up and login securely during 1st Sprint.

• Enable my customers to input and review their health data during 1st Sprint.

• Enable my customers to visualize and understand their health data during 2nd Sprint.

• Enable my customers to use the app on any web browser during 2nd Sprint.

• Enable my customers to receive custom dietary recommendations during 3rd Sprint.

• Enable my customers to access instructions and help documentation during 3rd Sprint.

• Enable my customers to export health data during 3rd Sprint.

2.3.1. Resources Required

• Customer UI testing

• Sample health data

2.3.2. Project Constraints

• Basic medical knowledge

• Data visualization experience

• Customer safety and regulations

• USDA food data server rate limits

* 1. Product Vision Statement

# GlucoCheck

**For** diabetics and their caregivers

**Who** need help monitoring, tracking, and managing blood sugar levels and stopping dietary glucose spikes,

**The** program GlucoCheck

**Is a** simple, easy-to-use, and clear web-based personal logbook

**That** allows even non-professionals to log in from anywhere, input insulin and glucose levels, visualize the changes over time, get reminders, and receive dietary recommendations.

**Unlike** competitors—such as Omnipod and Medtronic, which focus heavily on bolus doses and medical intervention to manage diabetes—

**Our product** also offers lifestyle changes and USDA/FDA-approved meal ideas for controlling blood sugar spikes with a more holistic approach that treats the whole person.

1. Project Execution
   1. Methods

Our team had only 1 semester to create the application. Based on the group's experiences—centered around Python 3 and SQLite—we decided to develop in Django, a Python framework a couple of us had used before and which integrates seamlessly with an SQLite database. Coordinating code and management with Git and GitHub and meeting at least twice a week on MS Teams, our scrum master and product owner put in a heavy amount of work at the beginning of the project. The development team's work, in turn, steadily increased during the sprints as we used a pull-style "board" where developers could choose the tasks most suited to their skills. Over the course of 3 sprints, our app grew into a more complete and competitive product.

During sprint 1, our team finished laying the groundwork for the project in terms of communication and collaboration tools. It was also during this sprint that we were able to deal with our most crippling constraints: basic medical knowledge, regulations, and information on food data API access. Our scrum master was able to fill out a series of legal release forms for the food data, but ultimately the server was not set up for the sort of access our team needed. We were able to tackle regulatory issues with a little research and some generic, HIPAA-tested disclaimers. Finally, our lead developer Kai provided textual summaries and Q&A for the team regarding diabetes, allowing us all a window into the condition and giving us sufficient basic knowledge to move forward. All of this was enough to get our app's skeleton built.

By the start of sprint 2, our team had completed the shell of the website and sign-up process but still needed to address the final constraint: visualization. Since no one had much experience with either Python or JS-based graphing libraries, we were able to solicit feedback from our mentors to help us arrive at a solution: client-side JS—in the form of Highcharts—on top of our server-side Python. It was also during this sprint that we started to run up against more personal challenges, delving further into unknown algorithms and complex chains of logic. Though the work was difficult, we finished the sprint with both the ability to input health data and meaningfully visualize it.

For sprint 3, we chose to focus on the next-most-important features that would give our product a competitive advantage: user-facing definitions, legally-safe health advice, and a public API for uploading health data either in bulk or from a device. These user stories brought us back into obscure territory, but we continued to learn new libraries, more complex algorithms, and even how to make better test data. By the end, we we able to implement all 3 of these new features.

Due to time constraints and the difficulty of the work involved, every sprint after the first saw a major reduction in user stories—and story points—completed. For example, during sprint 2, our team had hoped to do extended browser testing. Even though this user story had only 2 story points allotted, we ended up postponing the story in order to focus on features that would add more direct value to the customer; this was based on advice from our mentors. Though we did not reattempt the story in sprint 3, our evolving definition of done obviated the need for much of the testing as a separate story.

* 1. Project Management

The form of project management we used during application development was scrum. Though all of our team members had some familiarity with the framework, none of us had used it in a pure form. But because of the structure of the class and all the amazing stories we'd heard about scrum, we adopted it as closely as we were able to.

For our artifacts, such as sprint backlogs, we stuck exclusively to plaintext hosted on GitHub. This allowed everyone to easily update, read, and upload all changes without any specialized tools and encouraged transparency among team members who were otherwise only able to meet with each other virtually. Extra support for the artifacts, as well as the increment codebase, was provided via our Slack workspace, where we were able to post questions and answers at any time of day or night; this was especially important for our team because of scheduling issues.

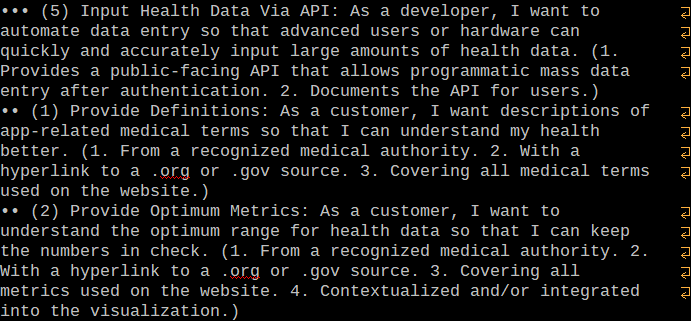
The scrum ceremonies were more difficult to carry out online and involved using the videoconferencing features of Microsoft Teams. This tool was not able to completely replicate the environment of an in-person daily scrum, but the regularity of our twice-weekly "standups" did help the team stay focused on the goal and continue to develop interpersonal connections. The larger ceremonies, such as sprint reviews and retrospectives, were also conducted on Teams, and we were able to make extensive use of screen sharing—both for presentation purposes and for live group editing of scrum artifacts and project documentation.

In accordance with our understanding of scrum, we did not mandate any specific processes in addition to scrum. However, the scrum master did provide a wealth of optional formats and processes for everything from filenames to metadata to docstrings. All of these tools allowed the team's code and documentation to better integrate. We also used a plugin for Slack called Poker Planner, which was invaluable for complexity estimation and sprint planning.

* 1. Artifacts

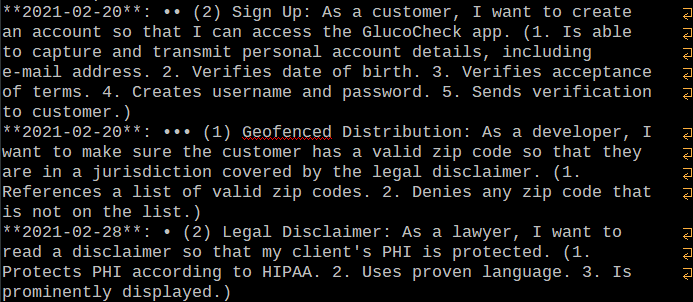
While a complete presentation of all our team's artifacts—to say nothing of their various states—is not possible, we've provided a brief sampling to highlight just how simple and effective our plaintext formats are for implementing everything from a backlog to a retrospective.

* + 1. Product/Sprint Backlog (example)



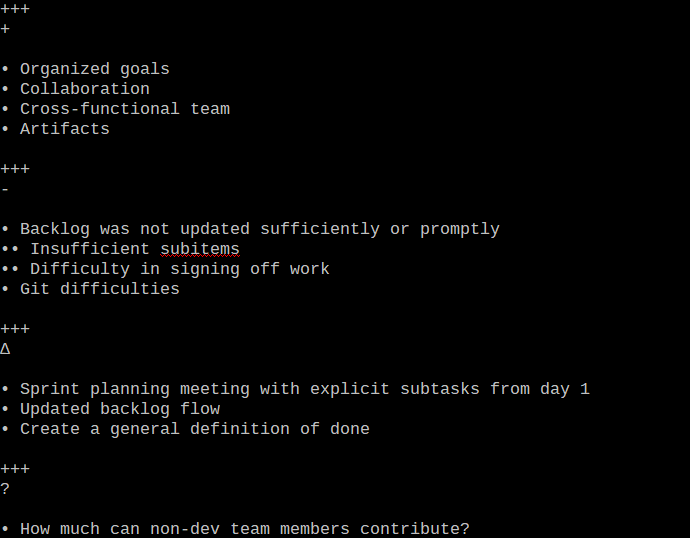
\*\*Figure 3.3.1.a\*\*: An image of the plaintext product backlog, showing user story precedence (bullets), story points (parenthetical number), title, body, and acceptance criteria (also parenthesized).

* + 1. Sprint Status Report (example)



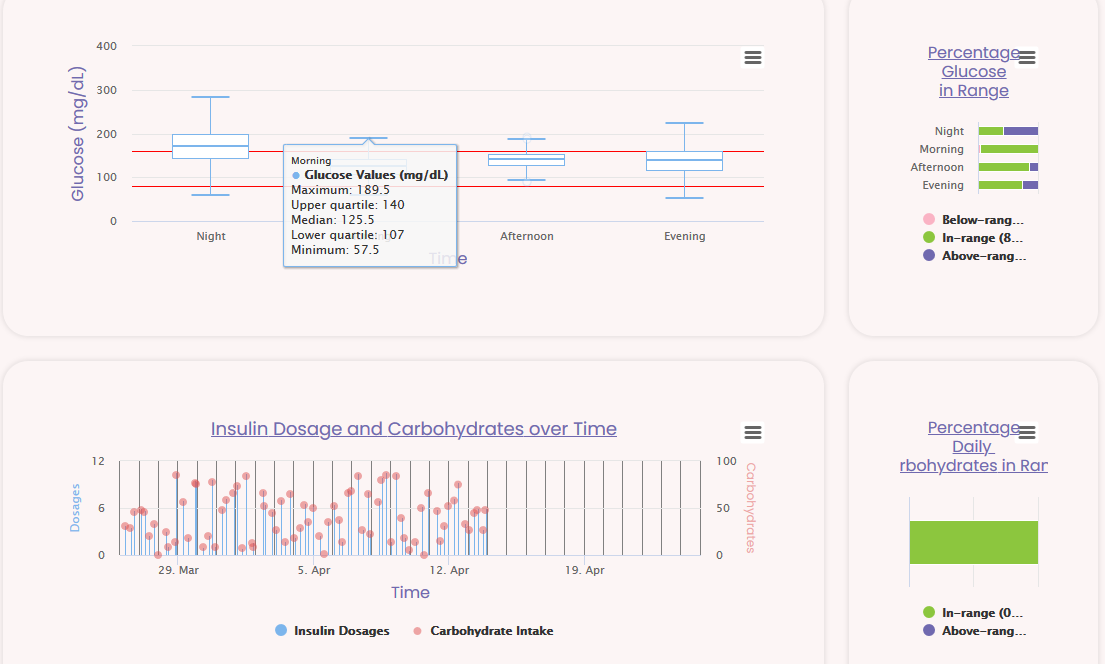
\*\*Figure 3.3.2.a\*\*: An image of a plaintext sprint backlog, derived directly from the product backlog to ensure single source of truth as far as scrum allows; note the added description lists denoting the completion date of the stories.

* + 1. Sprint Retrospective (example)

\*\*Figure 3.3.3.a\*\*: An image of a plaintext sprint retrospective, showing 4 chapters covering the 4 basic points: sprint positives, negatives, planned changes, and questions.

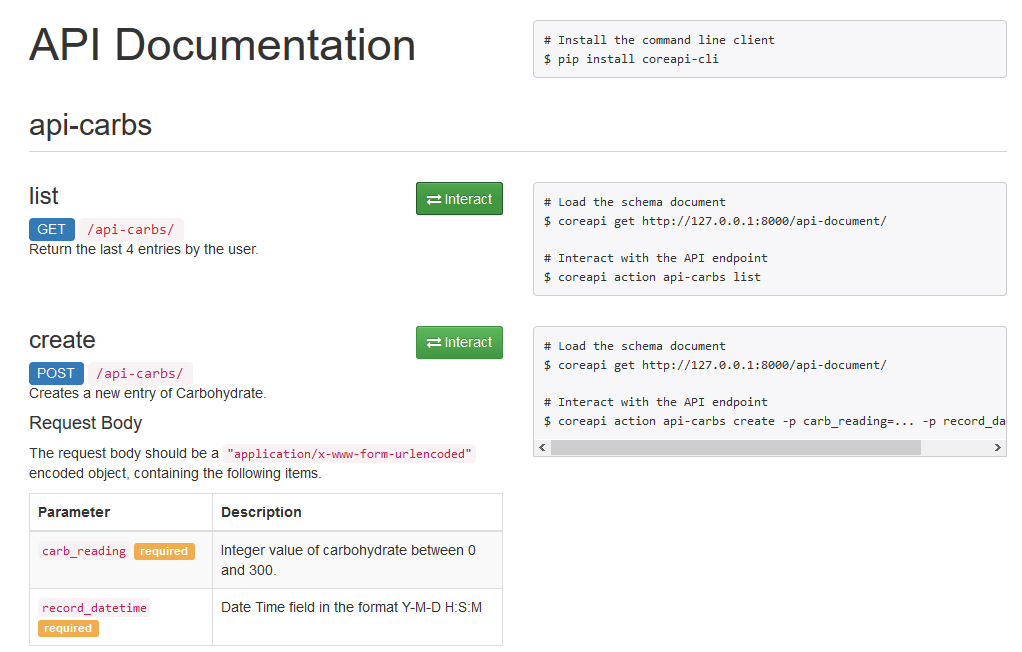
* 1. Results

By the end of 3 sprints, we fell far short of some of the goals we'd hoped to achieve. Some of the difficulties were brought on by forces outside of our control, such as the copyright and legal issues surrounding us implementing any robust form of meal planning. Other challenges were brought on by the scrum framework itself, which—despite its claims—created some unbreakable silos in our group and prevented valuable work from being done because of strict, role-based boundaries. Further roadblocks were truly our own: we lacked "real life" working experience with most of the tools, libraries (yet see our eventual visualization results, below), and even algorithms we needed to implement in order to create our app.



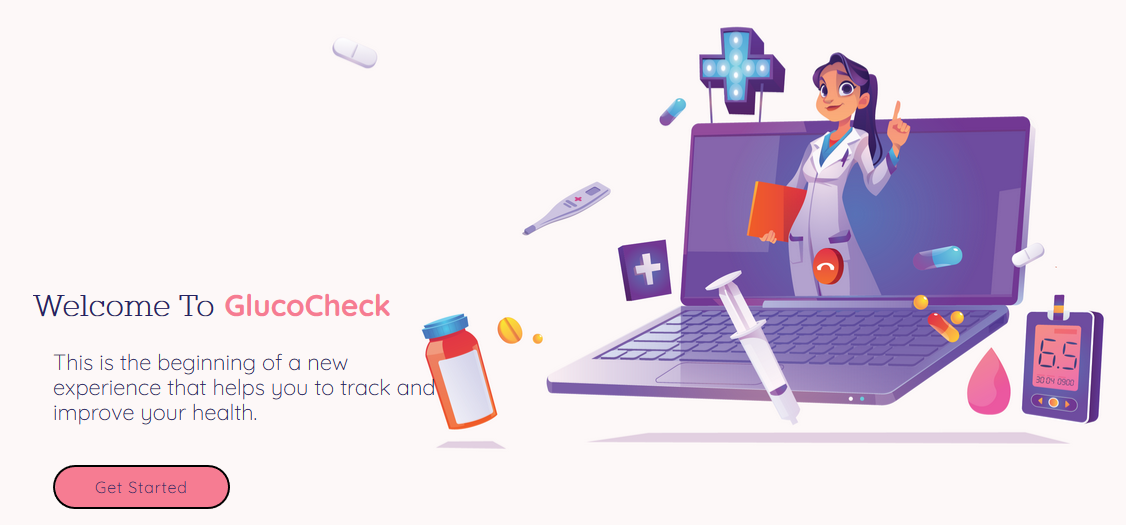
\*\*Figure 3.4.a\*\*: An image of part of the visualization dashboard, showing a whisker plot and scatter plot of glucose readings, carb intake, and insulin dosages, with summary bar graphs.

Yet with the continued support of our mentors, we were able to meet these challenges not as problems but opportunities. We were forced to confront questions such as feature priority, marketability, and usability. We had to rethink our definition of done. And we had to cut some features off completely, like exporting health data—it would not be too difficult to implement in the future, but there was no time to research and/or develop our own data format. All of this made our app—and our team—stronger. This focus gave us the ability to hone in on features like the API (see below).



\*\*Figure 3.4.b\*\*: An image of part of the API documentation, which allows low-level access for programmatic input and output of health data.

By the end of the project, we completed a total of 10 user stories (38 points): 5 in the 1st (15 points), 2 in the 2nd (7 points), and 3 in the 3rd (16 points). The app has working sign-up and login, supports multiple forms of data entry, and supports very nice visualizations. The custom artwork is also stunning (see example, below)—this was not included as part of any acceptance criteria and did divert some resources away form the stories, but it was a good tradeoff because it will help with customer acquisition and retention. Overall, the app met many of our goals and went a long way towards implementing our vision.



\*\*Figure 3.4.c\*\*: An image of the homepage, showcasing the specially-made art and graphic design of our website.

1. Conclusions / Summary

Our GlucoCheck app is a viable Python web app for storing, graphing, and providing context for diabetes health data provided by users. In 3 sprints, we went from scrum novices to veterans, and created a product with real commercial potential. The open API, vendor-neutral approach, and health suggestions give our application real advantages over competitors. Though some of the most distinguishing features are not quite ready, our app could be deployed as-is on a public website in a matter of minutes—fully ready for new customers. And because we have monetization strategies planned out already, our implementation can easily transition from a school project to a money-making enterprise. This shared experience will help our team in résumés, in future employment, and maybe even as a job in and of itself!

1. Reflections

*\****\*Kai\****\**: As a developer on the GlucoCheck capstone project, I was able to familiarize myself with the scrum/agile framework through our 'daily' standups and product demos. Due to the structure our scrum master provided, these meetings were thorough and well-organized. As part of the development team, I worked with the other developers and the product manager in the development of the product. This was my first time using a Git repo that was actually receiving updates from multiple people, and our team didn't have any particular issues with this once we got familiar with Git. I also had the opportunity to familiarize myself with the Django fullstack framework and write and experiment with code for the functionality it provides. This included writing HTML and CSS for webpage displays, views in order to render the webpages, forms for gathering user data, widgets to be used in those forms, and some test cases when there was time. Most of the code for this was relatively simple, just requiring re-familiarizing myself with the language or with the framework, although writing the code for and learning about sending asynchronous requests through JavaScript and Django was more interesting and will likely be more directly applicable to different web-based projects. The analysis code associated with this was also relatively complicated and allowed me to experiment with different methods for representing trends that data was matched to. If we were to move forward with this project, this is one area that I would expand as I am not positive the method I used was the best available.

*\*****\*Stuart\*****\**: This team is amazing, willing to focus on each user story or sprint as a collective. This was new to the team because we had not worked together on a project to identify a particular weakness and strength, amongst ourselves to have an idea on what would work and what would not work. Saying that, we did figure out what worked and what did not work by meeting and collaborating for each user story. We continue to agree to meet and share ideas meaning outside of schedule meetings, that held each other accountable by being present and that allowed us to complete a user story during the week. Ideas and strategies that were implemented to complete a user story, inspired the team. Instead of separating and working alone, by not communicating with each other and not working together, we push each other to meet the expectation, we agreed to overcome the challenges that what set before us. Admiration, respect, and accountable were the key components to have in a team environment. As a team we have great leadership in each team member and direction to keep the team going in the right direction, to not allow any short comings. The team did create a rapport that would allow us to present suggestions, even personal situation that everyone respected if never experience. The way the team continue to ask questions to be align, this was a learning process to identify what we need to do as a team to meet our goal. I am so proud and happy to be a part of something so special to witness, what worked for a team success story.

*\****\*Anjali\****\**: The GlucoCheck capstone project is a team project where multiple students from different elective courses come together and collaborate to achieve a common goal by sharing their fair share of workload on the basis of their skills, strengths, past experience, and knowledge gained during the courses. From this teamwork experience, I learned and developed my team building skills; it also helped me understand the importance of teamwork required for quality task completion and goal achievement. Participating as a developer I was able to learn more about Scrum framework through the videos, scrum meetings and materials provided by the instructor from time to time. This project provided me opportunity to learn and explore web development technologies like Django web framework that includes creating forms for collecting user's data, storing user data in a database, creating a database-driven web application, working on a model template view architecture, and API creation and documentation. I also learned a new way of sharing your work with others through GitHub.

*\****\*Elmehdi\****\**: The capstone project was a good experience that allowed me to have the opportunity to achieve different skill sets. As a developer, I was able to learn more about Django Web Framework and to collaborate with our team members to successfully finish our user stories during the frame time. The process of "Daily Scrum" meetings was a good opportunity to share ideas and get feedback from the team and to split the tasks and user stories for each sprint. We were also available almost every day to help each other, and collaborating to make sure that everything was working as we planned. When creating the user stories, I contributed to the web design by creating new HTML pages, applying CSS styling, adding JavaScript functions, and styling Highcharts graphs. GitHub also was a great tool to collaborate with our team and exchange codes and files, and to discover or review update on the repositories. Slack and MSTeams were other platforms to communicate and get feedback from our mentors which help us during working on the user stories.

*\****\*Jeremiah\****\**: Since other projects I've done at Wake Tech—outside of C++, at least—have gone so well and/or allowed me great flexibility to pursue my own interests, try creative solutions, and exercise strong leadership, this capstone project was very much a shock for me. I do not feel the scrum framework lived up to its hype and would probably not ever use it again, at least not without serious modifications in place, at minimum an increase in non-development-team flexibility along with a corresponding increase in openness, accountability, and standardized processes. As scrum master, I felt incredibly constricted by my role and was always looking for something more I could do. But despite all the limitations of scrum, the development team really worked hard to deliver an app—and it showed: we have something that looks great, contains useful features, and could easily be commercialized. I can only imagine what it would be like to work with the team under a different framework where all our skills were more utilized. Overall, it was definitely a learning experience and I would not hesitate to reach out to the dev team after graduation to continue work on this project or to try something new.

1. Acknowledgments

WTCC, our school, which provided all of us amazing opportunities for growth

IHS Markit, which provided us with patient and knowledgeable mentors

Tandem Diabetes Care, which graciously allowed us to use their tested disclaimers and policies

And the developers and maintainers of all the languages, frameworks, and libraries we used throughout our app