

Florida International University
School of Computing and Information Sciences

Software Engineering Focus

Final Deliverable

Project Title: Enabling Wearable Technology on Patients with High
Body Mass Index

Team Members: Gustavo Cordido, Alexander Pastoriza, Jose Bello, Robert Rodriguez, Idiel Guerra

Product Owner(s): Dr. Jessica Ramella, Andres Rodriguez

Mentor(s): Andres Rodriguez

Instructor: Masoud Sadjadi

The MIT License (MIT)
Copyright (c) 2016 Florida International University

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Abstract

This document presents the information necessary to gain a good understanding of the Enabling Wearable Technology on Patients with High Body Mass Index project. The Medical Photonics Laboratory at Florida International University, in a collaboration with PATHS-UP center of the National Science Foundation is currently developing wearable technology for the monitoring and treatment of obesity. The project described in this document aims to satisfy the software requirements for this device to function, as well as allowing researchers to conduct early testing and recordings with ease. The results of this project will be key in calibrating and adapting new wearable technology for the obese population.

Table of Contents

INTRODUCTION	5
CURRENT SYSTEM	5
PURPOSE OF NEW SYSTEM	5
USER STORIES	6
IMPLEMENTED USER STORIES	6
PENDING USER STORIES	15
PROJECT PLAN.....	16
HARDWARE AND SOFTWARE RESOURCES.....	16
SPRINTS PLAN	17
SYSTEM DESIGN	21
ARCHITECTURAL PATTERNS	21
SYSTEM AND SUBSYSTEM DECOMPOSITION	23
DEPLOYMENT DIAGRAM	24
DESIGN PATTERNS.....	24
SYSTEM VALIDATION.....	25
GLOSSARY	26
APPENDIX	27
APPENDIX A - UML DIAGRAMS.....	27
APPENDIX B - USER INTERFACE DESIGN.....	38
APPENDIX C - SPRINT REVIEW REPORTS.....	42
APPENDIX D - USER MANUALS, INSTALLATION/MAINTENANCE DOCUMENT, SHORTCOMINGS/WISHLIST DOCUMENT AND OTHER DOCUMENTS	50
REFERENCES	53

INTRODUCTION

Obesity remains an epidemic in the United States of America. With a 42.4% prevalence in adults (Center for Disease Control and Prevention, 2020), it has overtaken many other illnesses and been directly linked with numerous conditions. Conditions such as cardiovascular disease, strokes, type 2 diabetes and certain types of cancer have been related to obesity, and are the leading causes of preventable, premature death.

Proper means of monitoring and treating obesity are still in development, with wearable technology surging as a possible solution.

Current System

Though common in the U.S market and prevalent in the American population, wearable devices are not currently calibrated to account for the skin changes that obesity causes, nor do they process the changes in light absorption and scattering in an obese patient's skin. Furthermore, these are rarely adapted for obese patients and its dimensions are fixed with average sizes in mind.

Purpose of New System

Given the lack of optical properties for obese skin in literature, our solution proposes a wearable device, alongside of an analytical environment consisting of hardware and software components. The wearable is to have calibrated optical sensors that properly capture data on obese skin in the form of spectra. The device is controlled via a software platform, consisting of a Raspberry Pi program to control and manipulate the device and cloud infrastructure for database storage and manipulation. Future work will add a web platform to the system, to provide data accessibility to both patients and researchers.

USER STORIES

The following section provides the detailed user stories that were implemented in this iteration of the Enabling Wearable Technology on Patients with high BMI project. These user stories served as the basis for the implementation of the project's features. This section also shows the user stories that are to be considered for future development.

Implemented User Stories

User Story #1 Define needed tools and resources

Assigned to: Gustavo Cordido

As a developer part of the project team, I want to know which tools and resources will be needed to first approach this project, such that I may begin the planning phase.

Acceptance Criteria:

- Discuss previous work with product owners
- Understand the scope of the project as well as its architectural needs
- Define architectural patterns to be used

User Story #2 Research on current status of the system and literature

Assigned to: All team members

As a developer, I wish to understand the history of the project and current similar implementations, such that I may further understand its scope and propose appropriate solutions.

Acceptance Criteria:

- Research past solutions in the same area, if any.
- Research the BMI topic at hand, as well as the Obesity epidemic in the U.S.

User Story #3 Research cloud platform alternatives

Assigned to: All team members

As a developer, I wish to have a list of cloud platforms and their advantages, such that I can discuss our team's choice as needed for the project.

Acceptance Criteria:

- Research and list the capabilities of available cloud platform services.
- Define costs and budget to propose for PO.

User Story #4 Research on similar devices and microcontrollers

Assigned to: Alexander Pastoriza, Idiel Guerra

As a developer, I want to properly understand the inner workings and software requirements of similar devices to the one this project entails, as well as the state-of-the-art microcontroller technology such that I may implement solutions appropriately.

Acceptance Criteria:

- Consult literature of similar devices.
- List possible microcontroller solutions, its pros and cons.

User Story #5 Research Azure database management options

Assigned to: Jose Bello, Gustavo Cordido

As a developer, I want to know which database platforms are available in Azure, such that I can help my team decide the appropriate one for the project.

Acceptance Criteria:

- List possible Azure database solutions.
- Test Azure connectivity and determine feasibility of use

User Story #6 Database Mapping

Assigned to: Jose Bello

As a researcher, I want to have a way to store, access and filter my recordings, such that I may access a specific patient's data using an individual encrypted ID, body part of the recording, and date.

Acceptance Criteria:

- Using an ER diagram, map the different tables and establish relations between them.

User Story #7 Define table keys

Assigned to: Jose Bello, Robert Rodriguez

As a developer, I want to define each database table's keys appropriately, such that I can ensure an efficient and optimal querying process.

Acceptance Criteria:

- Optimize queries and test possible queries using the defined keys.
- Fix any errors that may come up from these queries.

User Story #8 Define type of data storage

Assigned to: Gustavo Cordido, Robert Rodriguez

As a researcher, I want my recordings to be saved in some type of tabular data file, such that is easily accessible and readable by further revisions of the software for analysis.

Acceptance Criteria:

- Research and list pros and cons of different tabular data types.
- Discuss and pick a specific type with the group and PO.

User Story #9 Familiarize with Raspberry Pi

Assigned to: Alexander Pastoriza

As a developer, I wish to fully understand how to interact with the Raspberry Pi and its requirements, such that I am able to implement it as part of the device and run the software solution in it.

Acceptance Criteria:

- Conduct research on the Pi's documentation websites and test its functionalities.

User Story #10 Basic Interface Design

Assigned to: Idiel Guerra, Robert Rodriguez

As a researcher, I want the software solution to have a simple User Interface that allows my team to control the device via buttons, such that device management can be done with ease.

Acceptance Criteria:

- Define necessary libraries for solution's interface design.
- Design a basic interface and verify that it's to the POs liking.

User Story #11 Implementing database in PostgreSQL

Assigned to: Jose Bello

As a developer, I want to be able to test my database design locally and in accordance to our Azure deployment selection (PostgreSQL), such that I may test the database system with various queries and pinpoint errors or efficiency issues.

Acceptance Criteria:

- Build the database using PostgreSQL software.
- Test its usability through random querying.

User Story #12 Add researcher table to the Database

Assigned to: Jose Bello

As a researcher, I wish to be able to log in with my personal credentials into the software system, such that the sessions I conduct are linked to my profile and are easily identifiable by me and other researchers.

Acceptance Criteria:

- Researcher data table built into the database.
- Test errors in querying after addition of table.
- Optimize querying in accordance to the newly added keys.

User Story #13 Set up version control environment

Assigned to: Gustavo Cordido

As a developer, I want to be able to keep track of updates to the code base in the form of a repository, so that any changes done by me or my team mates are accounted for before merging to the main platform.

Acceptance Criteria:

- Set up GitHub team repository.
- Initialize repository with a README file.
- Test connectivity from development environment to repository.

User Story #14 Azure account connectivity

Assigned to: Gustavo Cordido, Robert Rodriguez

As a researcher, I want the software platform to be linked to my research group's Azure subscription, such that we may all collaborate with the study from the cloud.

Acceptance Criteria:

- Install and implement Azure connectivity libraries into the code base.
- Test connectivity with server pings via the command line.

User Story #15 Raspbian OS installation and testing

Assigned to: Alexander Pastoriza, Idiel Guerra

As a researcher, I want to have an Operative System running the device's Raspberry Pi component such that I can easily run the software platform appropriately.

Acceptance Criteria:

- Verify that Raspbian OS is fully installed and running on the Pi

User Story #16 Determine security protocols and encryption of Database IDs

Assigned to: Idiel Guerra, Jose Bello

As a researcher, I want to access my account safely and securely, such that only I am able to conduct sessions under it and no patient information is at risk of leakage.

Acceptance Criteria:

- Research security protocols available through Azure.
- Test Azure Key Vault option for encryption of keys.

User Story #17 Establish Raspberry Pi-Spectrometer Connection

Assigned to: Alexander Pastoriza

As a developer, I wish to have the Pi and Spectrometer components connected, such that my solutions may control and execute the spectrometer functionalities when ran from the Pi.

Acceptance Criteria:

- Verify that both the Pi and Spectrometer are connected via Wi-Fi pings.

User Story #18 Test internal connectivity

Assigned to: Alexander Pastoriza

As a researcher, I want to send signals from the Pi to the Spectrometer, and vice versa such that the connection between both components is evidenced.

Acceptance Criteria:

- Install and implement spectrometer libraries into the code base.
- Verify that the implementation is working by sending commands such as “Start”, “Stop”, “Average” to the spectrometer.

User Story #19 Capacitate Product Owners in the use of version control repositories

Assigned to: Gustavo Cordido

As a researcher, I want to manage the code base stored in the established repository, so that I may develop the solution further and apply new features in future revisions.

Acceptance Criteria:

- Prepare a GitHub command presentation, demo and sheet for the POs to use.
- Schedule a meeting to showcase the prepared resources.
- Verify understanding by overseeing POs creating and managing a demo repository.

User Story #20 Adjust buttons to meet research's demand

Assigned to: Robert Rodriguez

As a researcher, I want to have access to a *Start, Stop, Averages, Number of Spectra Returned* buttons to operate the device, such that the spectrometer can be fully controlled via the interface

Acceptance Criteria:

- Verify that buttons are sending the appropriate signals to the spectrometer.
- Verify that functions are connected appropriately to their designated buttons.

User Story #21 Add create account/log in page

Assigned to: Robert Rodriguez

As a researcher, I want to be able to create an account to log in to the system, such that my recordings can be tied to my user.

Acceptance Criteria:

- Add a new interface page to the start of the program.
- Verify that there is a Sign Up and Log In button.
- Verify that there is Username and Password fields

User Story #22 Create authentication between sign up/log in and database

Assigned to: Jose Bello

As a researcher, I want my account to be stored as persistent data in the database when I create it, such that I can be authenticated through this connection when logging in.

Acceptance Criteria:

- Verify information is inserted correctly into the database's researcher table from the sign up page fields.
- Verify that the database is appropriately queried when logging in.

User Story #23 Implement secure authentication for Log In

Assigned to: Idiel Guerra

As a researcher, I want my account information to be accessed securely such that my information is not at risk from an outsider attack.

Acceptance Criteria:

- Implement password hashing and decoding from the database.

- Determine if Azure's security protocols are enough for safe authentication.

User Story #24 Implement graphical interface into the recordings page

Assigned to: Alexander Pastoriza, Jose Bello

As a researcher, I want the spectrometer readings to be graphically depicted in the interface while I conduct a recording, so that I may spot errors in the device and ensure that recordings are happening correctly.

Acceptance Criteria:

- Merge existing Seabreeze spectrometer graphics into the software solution.
- Verify that the graphical depiction is showcasing accurate data by light testing on the spectrometer, using multi-wavelength LED light sources.

User Story #25 Migrate the solution to an MVC architectural pattern

Assigned to: Gustavo Cordido

As a developer, I wish to have the codebase organized following the chosen architectural patterns, so that accessing/fixing specific services is easily done and issues are quickly identified.

Acceptance Criteria:

- Depict the architectural patterns for the separation to be conducted.
- Separate functionalities into its appropriate functions and files.
- Verify the integrity of the software solution before pushing to the repository

User Story #26 Adjust graphical representation to depict accurate values

Assigned to: Robert Rodriguez

As a researcher, I want the graphing of my readings to start at 0, rather than the current 1500 value; such that the graphed readings can be seen appropriately.

Acceptance Criteria:

- Adjust the y-axis of the graph to the appropriate scale.

User Story #27 Implement averages function

Assigned to: Alexander Pastoriza, Gustavo Cordido, Jose Bello

As a researcher, I want to take advantage of the averages function from the spectrometer, such that the returned recordings are averaged by a set amount.

Acceptance Criteria:

- Determine whether it's possible to wrap the original function in Python.
- Define its process and draft a similar algorithm if necessary.
- Verify that the chosen solution, whether wrapping or manually implementing provide the desired results

User Story #28 Implement data writing function

Assigned to: Alexander Pastoriza, Jose Bello

As a researcher, I want the averaged recordings returned by the spectrometer to be automatically stored in a tabular file, so that I can easily process it in future revisions.

Acceptance Criteria:

- Import the csv library to controller file.
- Convert the incoming data to csv format using the csv library functions.
- Verify that the data is correct and uncorrupted after the writing is done.

User Story #29 Azure storage account connectivity

Assigned to: Gustavo Cordido, Jose Bello

As a researcher, I want the raw data files created by the program to be stored in Azure, such that it is backed up in the cloud and it is easily accessible from any computer.

Acceptance Criteria:

- Define necessary requirements for storage account connectivity.
- Test connectivity via file upload to one of the storage account's containers.

User Story #30 Upload data to Azure

Assigned to: Gustavo Cordido

As a researcher, I want the .csv files produced by the program to be directly uploaded to our Azure storage, such that me and my team can have access to it for future processing.

Acceptance Criteria:

- Verify the connectivity with storage account from within the codebase.
- Verify that files are uploaded correctly, both formatting and identifiers must be accurate.

User Story #31 Implement Hashing for Password Authentication

Assigned to: Idiel Guerra

As a researcher, I want to ensure that my password is secured when I use it to log in, so that my account's information cannot be easily breached.

Acceptance Criteria:

- Implement a method that uses hashing to encrypt the passwords
- Verify that the passwords are hashed correctly by running tests on the implementation

User Story #32 Documentation Report

Assigned to: All team members

As a researcher, I want the current software solution to be documented in detail, so that any further revisions to the codebase and new developments are compared to the previous development cycle

Acceptance Criteria:

- Reformat project plan, sprint reviews and user stories.
- Detail the architecture used in system design.
- Create appropriate diagrams

User Story #33 User Manual

Assigned to: Alexander Pastoriza

As a researcher, I wish to have access to a user manual for the software, so that I may refer back to it in case of need and share with fellow researchers in the future

Acceptance Criteria:

- Verify that the user manual covers every aspect of the software

User Story #34 Demo Videos

Assigned to: Jose Bello

As a researcher, I wish to have access to a series of videos that demonstrates the usage and functionalities of the project, so that I may refer back to it to understand it.

Acceptance Criteria:

- Record introductory video.
- Record demonstration of interface usage.
- Record demonstration of Azure usage.

User Story #35 Pathable Set Up

Assigned to: Gustavo Cordido

As a senior student, I want my team's project to have a dedicated Pathable website, such that we can showcase our project and work during Senior Showcase.

Acceptance Criteria:

- Verify that project description is correct.
- Verify that the team members' names are correct.
- Verify every team member has set their Pathable account appropriately.

Pending User Stories**User Story #36 Data analysis and processing adaptation**

Assigned to: Future developers

As a researcher, I want to run analysis and data processing models on the extracted data through Azure, such that no downloading is needed and all the data can be processed entirely in the cloud.

User Story #37 Deploy data website

Assigned to: Future developers

As a researcher, I wish to access the processed data through a web application, such that I can visualize the resulting data.

User Story #38 Adapt web application for patients

Assigned to: Future developers

As a user, I wish to access my recordings in the form of graphical data through the website, so that I may consult and have access to any of my analyzed values.

PROJECT PLAN

This section describes the planning that went into the realization of this project. This project incorporated the agile development techniques and as such required the sprints to be planned. These sprint planning are detailed in the section. This section also describes the components, both software and hardware, chosen for this project.

Hardware and Software Resources

The following is a list of all hardware and software resources that were used in this project:

Hardware:

- Raspberry Pi 3b+
- Ocean Insight STS-VIS Spectrometer
- Multi-wavelength LED light sources

Software:

- **Programming Language:** Python 3
- **OS:** Windows 10, Mac OS Catalina, Raspbian
- **Environment:** VSCode, PyCharm
- **Package Manager:** Python's pip3
- **Version Control:** GitHub
- **Deployment:** Azure
- **Database:** PostgreSQL on Azure
- **Database Manager:** Azure Database for PostgreSQL, pgAdmin4
- **Scrum Organization:** Microsoft Teams, Trello

Sprints Plan

Sprint 1

Start time: 7:00PM

End Time: 8:00PM

ID	Story	Assigned To	Points
1	Define needed tools and resources	Gustavo Cordido	5
2	Research on current status of the system and literature	All team members	8
3	Research cloud platform alternatives	All team members	3
4	Research on similar devices and microcontrollers	Alexander Pastoriza, Idiel Guerra	5

Velocity: 21

Sprint 2

Start time: 7:00PM

End Time: 8:00PM

ID	Story	Assigned To	Points
5	Research Azure database management options	Jose Bello, Gustavo Cordido	5
6	Database Mapping	Jose Bello	8
7	Define Table Keys	Jose Bello, Robert Rodriguez	8
8	Define type of data storage	Gustavo Cordido, Robert Rodriguez	5
9	Familiarize with Raspberry Pi	Alexander Pastoriza	13
10	Basic interface design	Idiel Guerra, Robert Rodriguez	8

Velocity: 47

Sprint 3

Start time: 6:30PM

End Time: 7:30PM

ID	Story	Assigned To	Points
11	Implementing database in PostgreSQL	Jose Bello	5
12	Add researcher table to the database	Jose Bello	5
13	Set up version control environment	Gustavo Cordido	5
14	Azure storage account connectivity	Gustavo Cordido, Robert Rodriguez	8
15	Raspbian OS installation and testing	Alexander Pastoriza, Idiel Guerra	13

Velocity: 36***Sprint 4***

Start time: 7:30PM

End Time: 8:30PM

ID	Story	Assigned To	Points
16	Determine security protocols and encryption of database IDs	Jose Bello, Idiel Guerra	13
17	Establish Raspberry Pi-Spectrometer Connection	Alexander Pastoriza	8
18	Test internal connectivity	Alexander Pastoriza	8
19	Capacitate Product Owners in the use of repository	Gustavo Cordido	8
20	Adjust buttons to meet the research demand	Robert Rodriguez	8

Velocity: 45***Sprint 5***

Start time: 7:30PM

End Time: 8:30PM

ID	Story	Assigned To	Points
----	-------	-------------	--------

21	Add create account/log in page	Robert Rodriguez	8
22	Create authentication between sign up/log in and database	Jose Bello	13
23	Implement secure authentication for Log In	Idiel Guerra	8
24	Implement graphical interface into the recordings page	Alexander Pastoriza	13
25	Migrate the solution to an MVC architectural pattern	Gustavo Cordido	13

Velocity: 55

Sprint 6

Start time: 7:30PM

End Time: 8:30PM

ID	Story	Assigned To	Points
26	Adjust graphical representation to depict accurate values	Robert Rodriguez	8
27	Implement averages function	Jose Bello, Gustavo Cordido, Alexander Pastoriza	13
28	Implement data writing function	Jose Bello, Alexander Pastoriza	8
29	Azure storage account connectivity	Gustavo Cordido, Jose Bello	8
30	Upload data to Azure	Gustavo Cordido	8
31	Implement hashing for password authentication	Idiel Guerra	8

Velocity: 53

Sprint 7

Start time: 7:30PM

End Time: 8:30PM

ID	Story	Assigned To	Points
32	Documentation Report	All team members	13

33	User Manual	Alexander Pastoriza	8
34	Demo Videos	Jose Bello	8
35	Pathable set up	Gustavo Cordido	8

Velocity: 37

SYSTEM DESIGN

This section contains information on the design decisions that went into this project. The architecture patterns are outlined and explained. The entire system is shown in a package diagram and the subsystems are explained. Finally, the design patterns used in the project are discussed.

Architectural Patterns

Our project implements a layered architectural pattern, which four different, recognizable layers. It consists of a low-level hardware layer, consisting of a Raspberry Pi connected to an Ocean Insights spectrometer via Pi-USB; a front-end layer consisting of a simple user interface; a back-end controller that communicates and manipulates the spectrometer, its recordings and functionalities, as well as create the data files and upload them to the data layer; which is composed by a relational database system. Figure 1 depicts the architecture and the communication between layers.

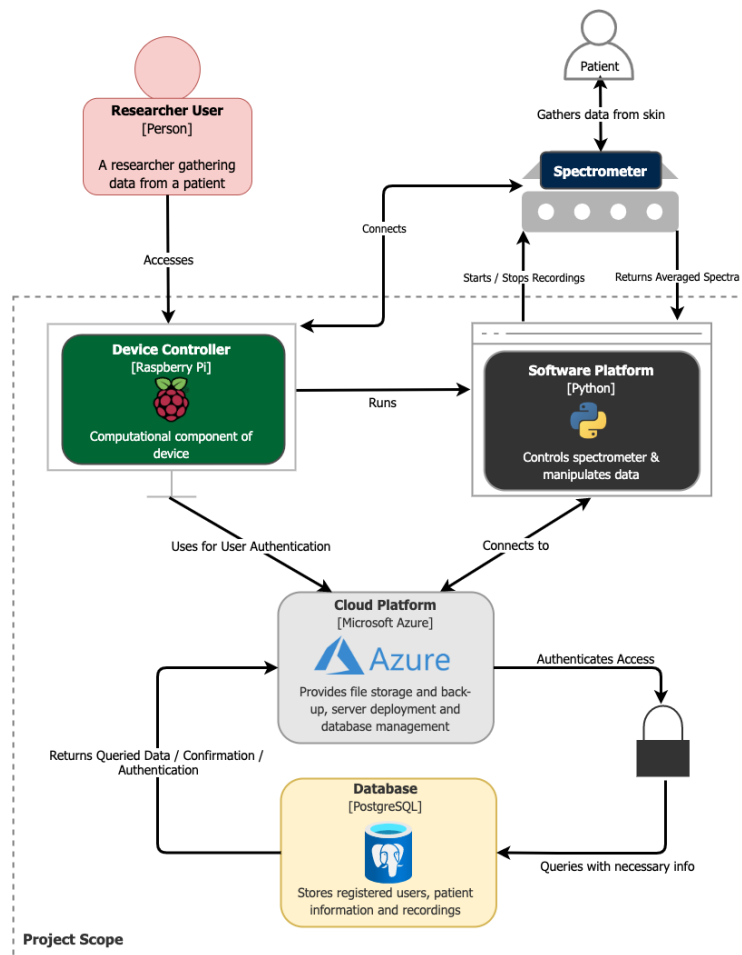


Figure 1. Architecture Design. *Designed by Gustavo Cordido*

The software platform was developed using an MVC architecture with Python. The user is presented with views in the form of a simple *tkinter* interface, which provides the necessary fields for patient info and parameter values required to start recording session. Models are used to fetch and send data to and from a PostgreSQL database. A main controller bridges the extracted data with the views and allows access to services from both ends.

System and Subsystem Decomposition

Our main systems are identified by our Database, Azure and Python. These systems communicate with each other and have subsystems of their own. As depicted in Figure 2, the Python system contains a Controller and a Log In/Sign Up subsystems. Each of these subsystems manage subsystems of their own, such as the spectrometer subsystem under controller. Dependencies are depicted using dotted arrows, such as the data creation subsystem depending on the patient info and spectrometer readings subsystems.

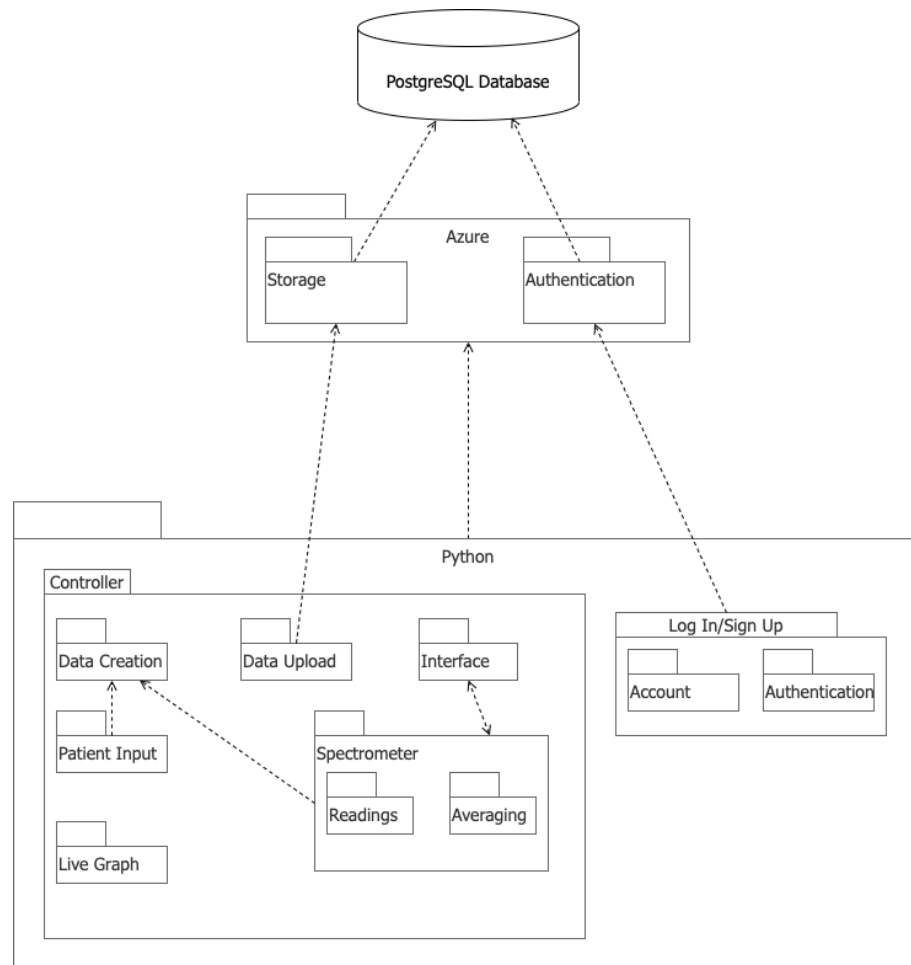
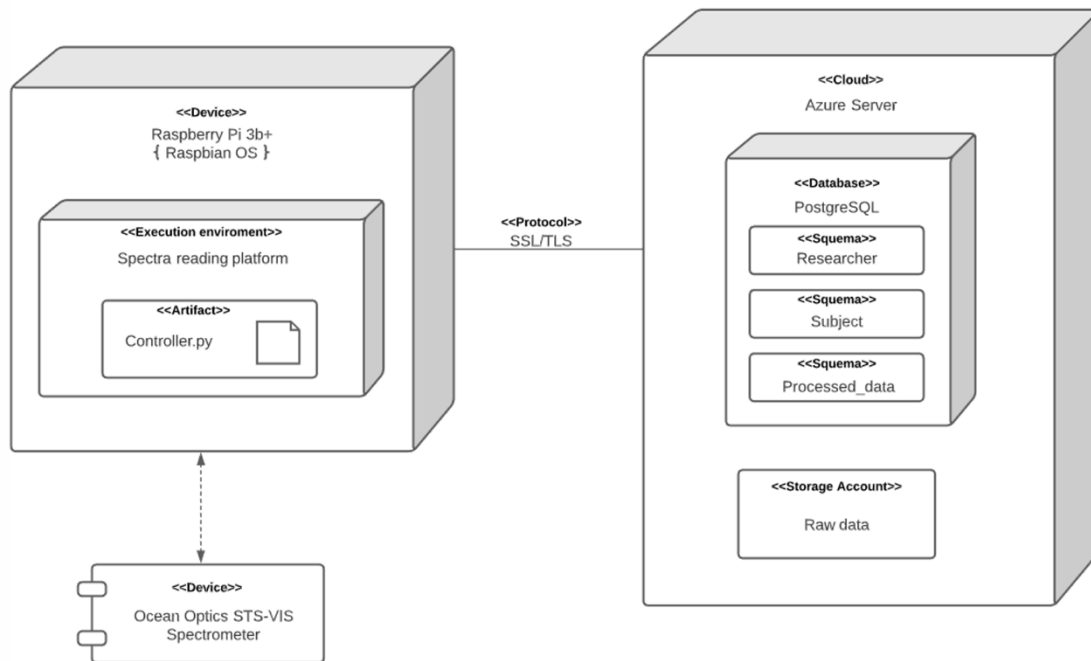


Figure 2. Systems and Subsystems Diagrams. *Designed by Gustavo Cordido*

Deployment Diagram



Designed by Jose Bello

Design Patterns

The main design patterns used by our software are Façade, Mediator and Composite. The Façade pattern is observed in the data recording class of the project, which handles spectrometer readings, manages number of spectra returned, number of averages to compute as well as recording duration; which makes the class a representation of an entire subsystem. Moreover, our solution includes a file that contains multiple methods used to expand the functionality of other components, following a Mediator design. This enhances the communication between classes and components, as the file is accessible project-wide. Lastly, we assembled our entire system using Composite pattern design. All classes must go through a controller class which creates all the necessary objects for successful operation.

SYSTEM VALIDATION

Our system conducted different types of testing through its development. Connectivity testing between the device's components was originally tested via simple pings between the Pi and spectrometer and were later moved into full commands. The software platform testing was conducted locally, divided in three stages of testing.

Firstly, we tested account creation and verification through a local implementation of the PostgreSQL database. The team tested that the accounts were being accurately created by querying the necessary details from the database. We moved on to testing the spectrometer functionalities and commands, through the use of an emulator. All team members conducted test recordings utilizing the emulator, and logged their results for comparison with expected results.

The Azure deployment was tested via demo data upload using individual member subscription, and then tested again using the Product Owner's subscription.

We validated the system under the following criteria:

- Researchers can create an account.
- Researchers can log in to their account.
- Researchers can input patient data.
- Patient data is stored appropriately into the database.
- Spectrometer is initiated correctly, and its live recordings graphed appropriately.
- Raw data files are created according to the input parameters.
- Raw data files are uploaded to Azure.

GLOSSARY

BMI – Body Mass Index

Emulator – Program that enables one system to behave like another

MVC – Model-View-Controller

PO – Product Owner

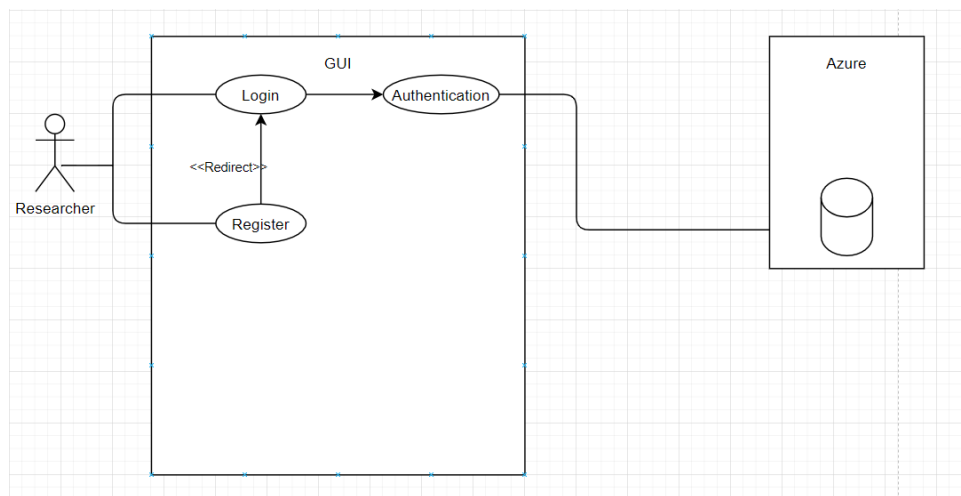
Repository – Remote location where data is stored

APPENDIX

Appendix A - UML Diagrams

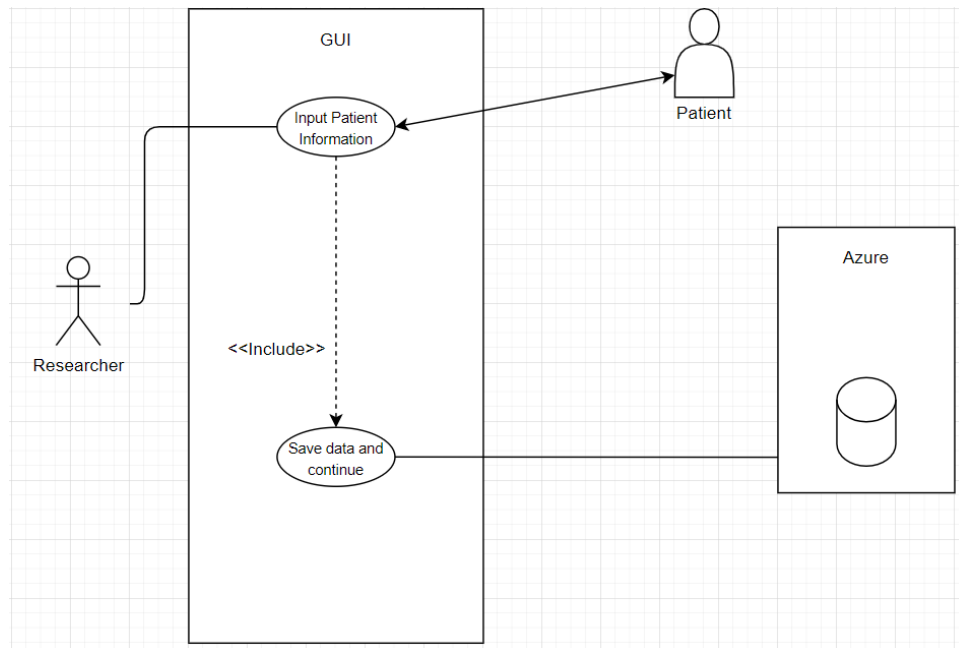
Use Case Diagrams

General Authentication



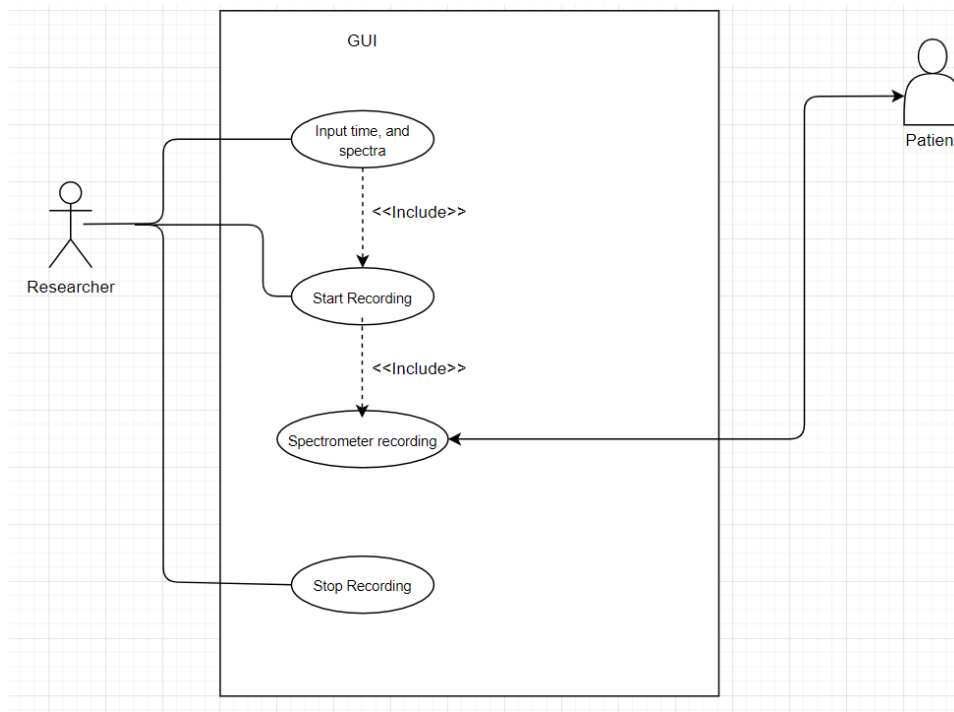
Designed by Idiel Guerra

Data Gathering



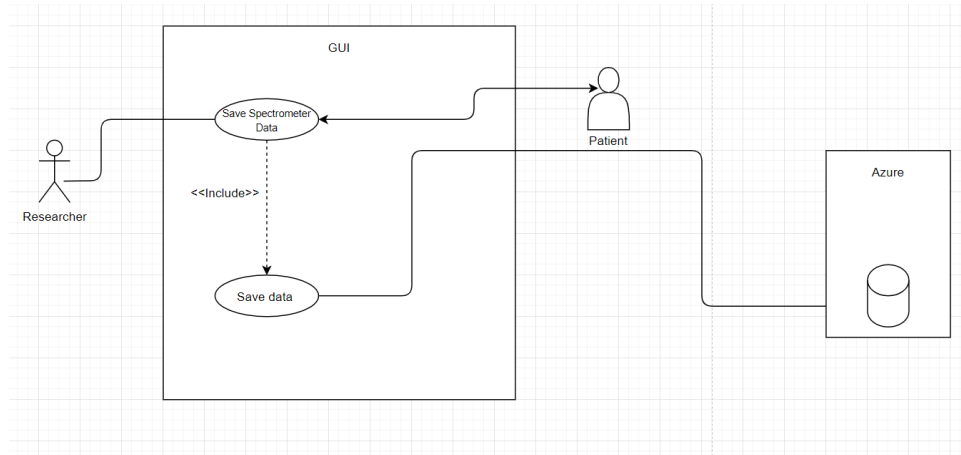
Designed by Robert Rodriguez

Data Recording

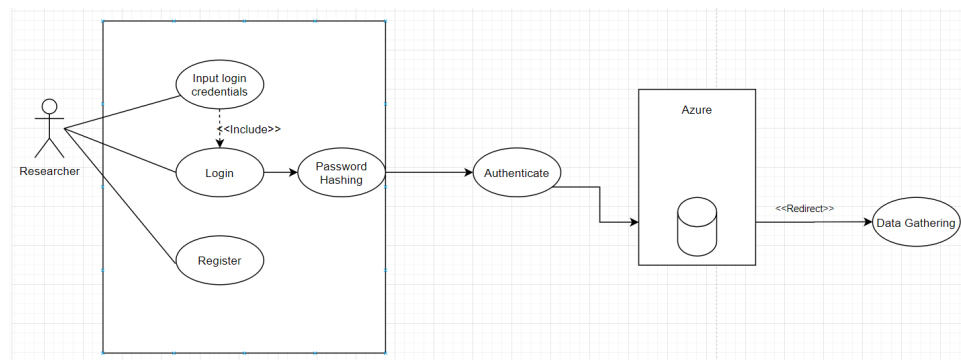


Designed by Alexander Pastoriza

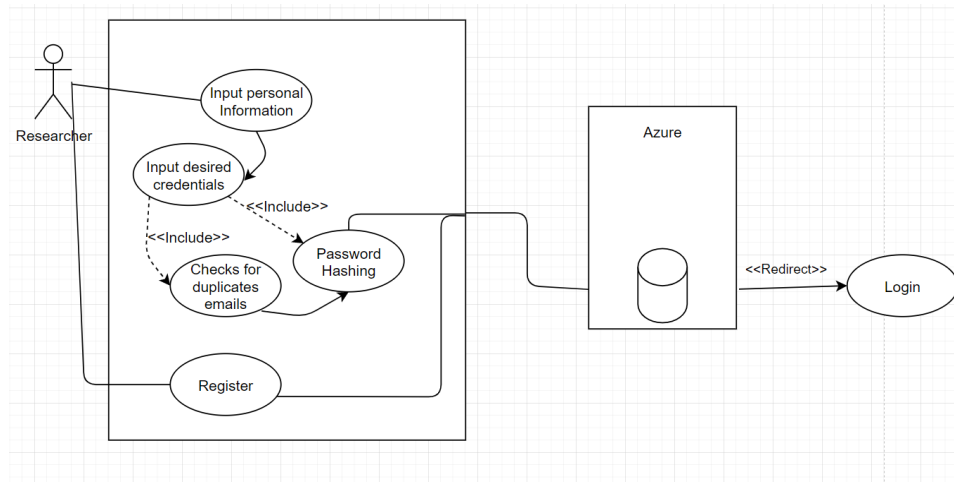
Data Saving

*Designed by Jose Bello*

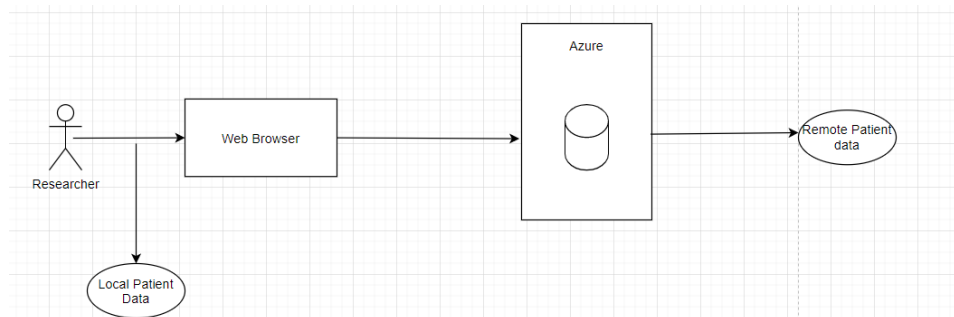
Login

*Designed by Jose Bello*

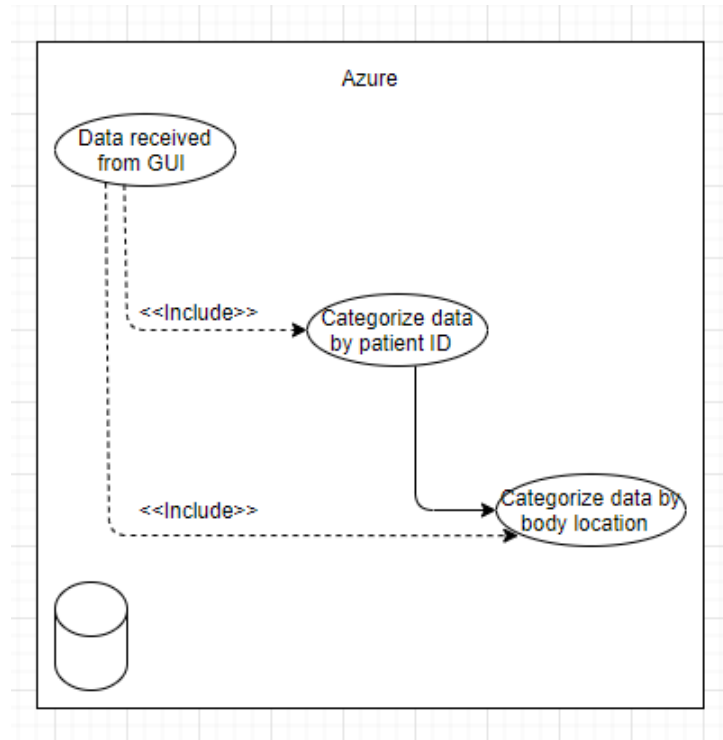
Sign Up

*Designed by Gustavo Cordido*

Fetch Patient Data

*Designed by Robert Rodriguez*

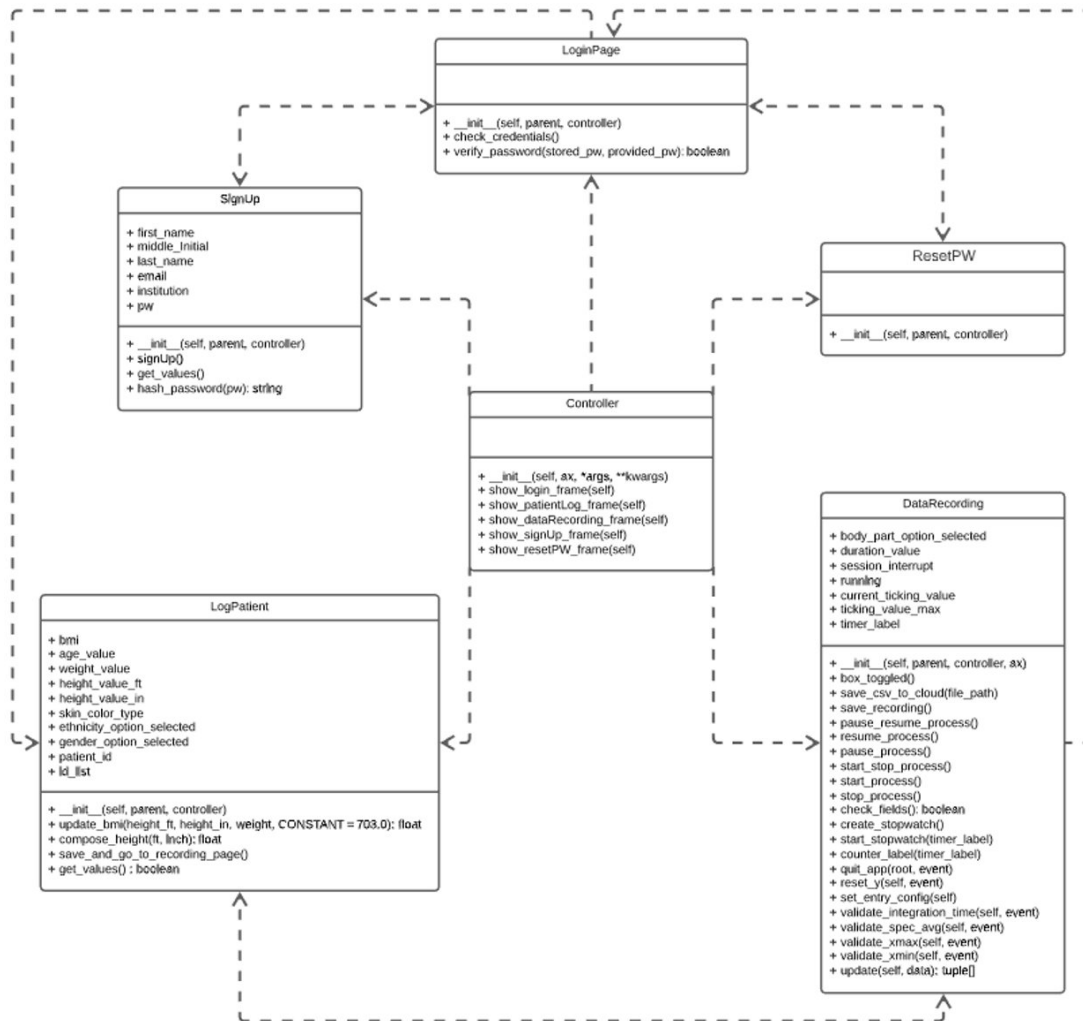
Categorize Data



Designed by Idiel Guerra

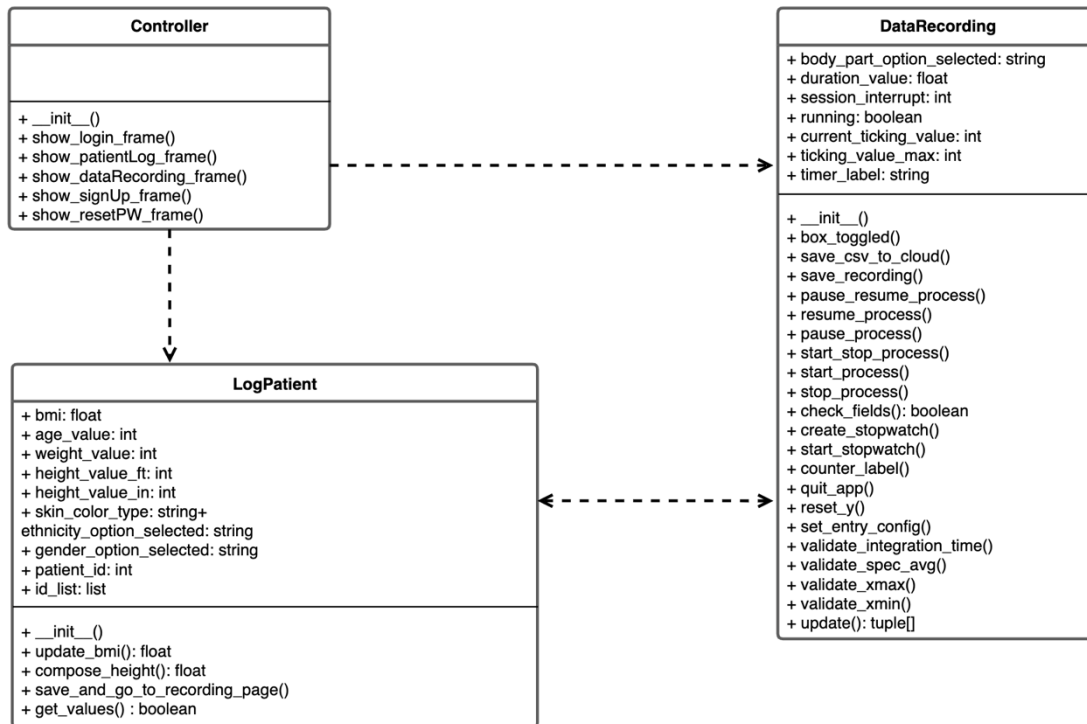
Class Diagrams

Project Class Diagram



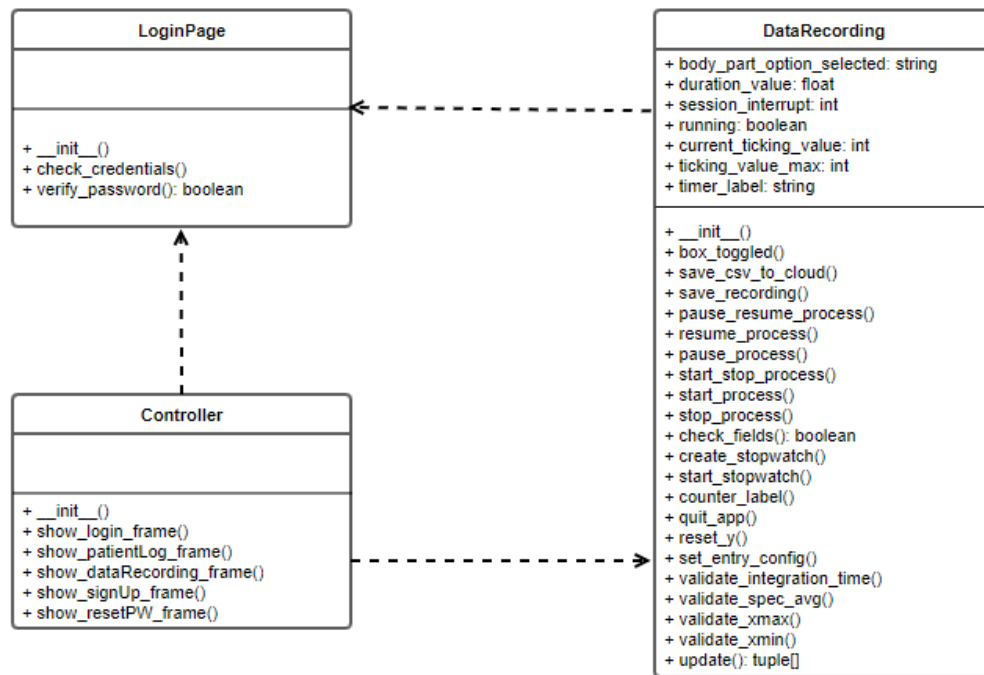
Designed by Alexander Pastoriza & Gustavo Cordido

Log Patient Class Diagram



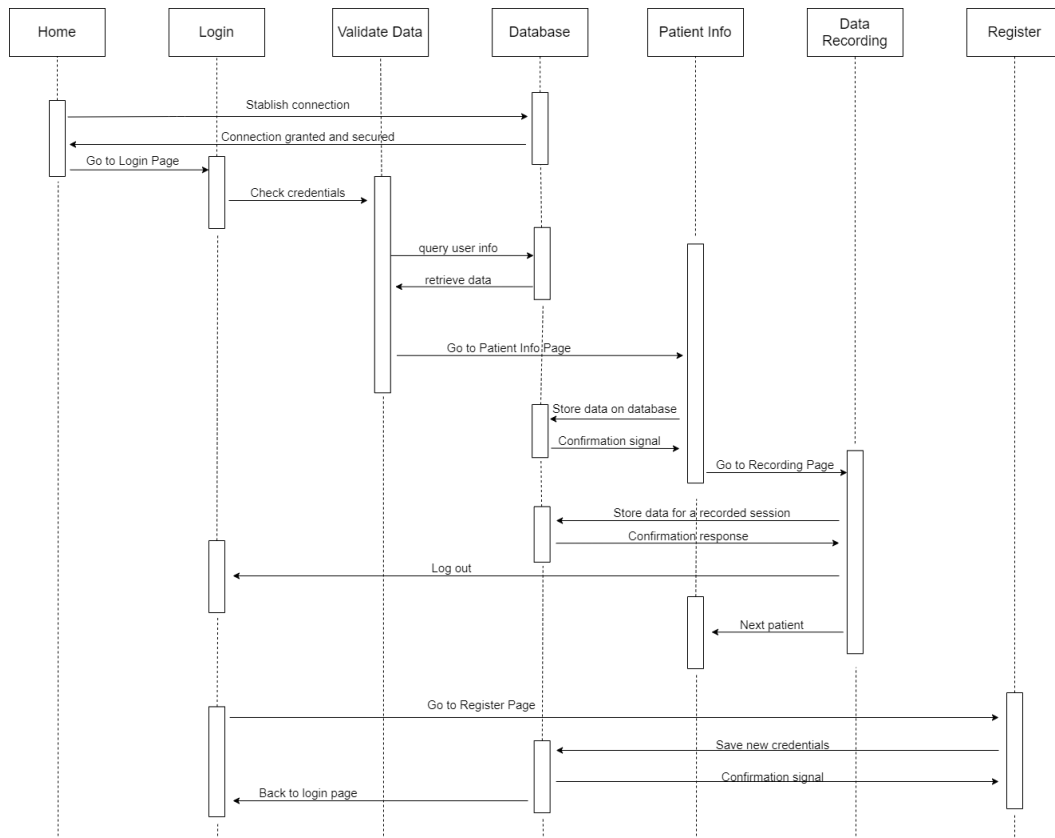
Designed by Robert Rodriguez & Idiel Guerra

Data Recording Page Class Diagram



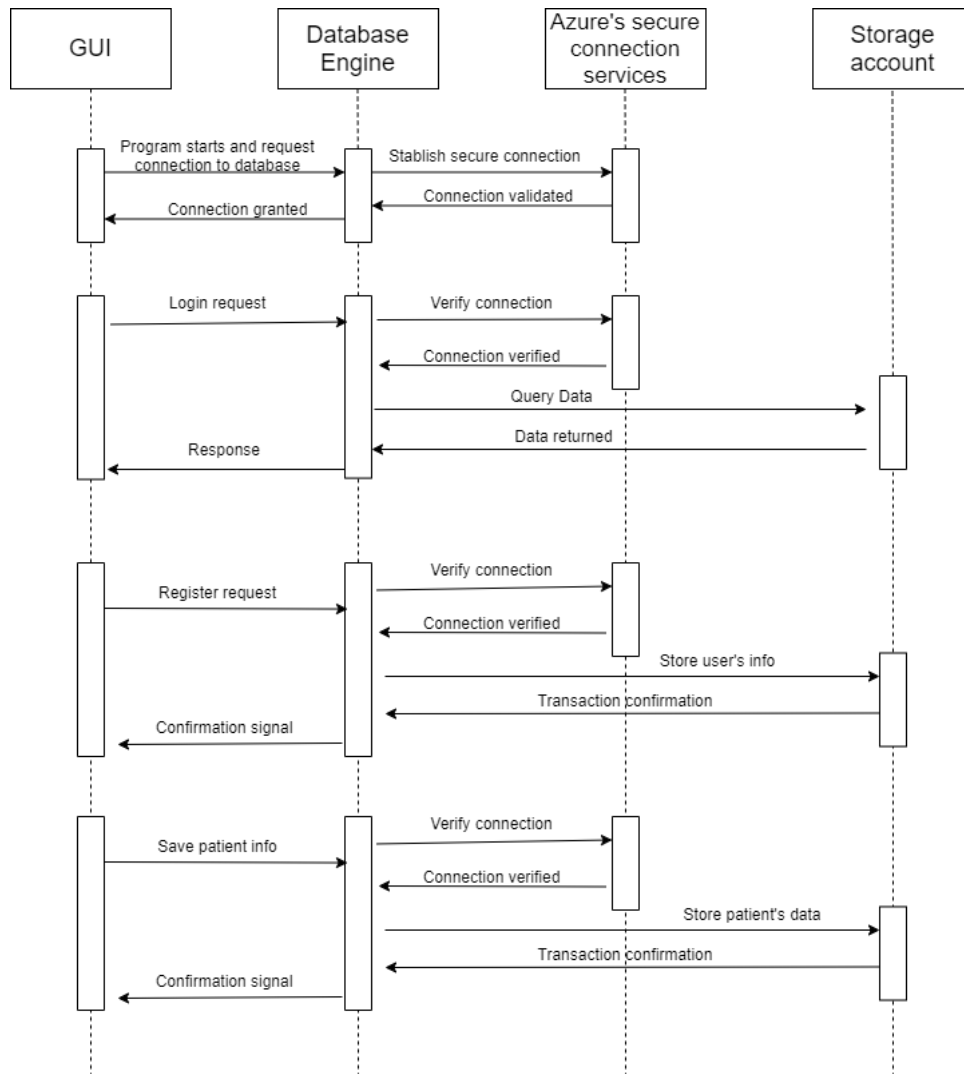
Designed by Jose Bello

General Sequence Diagram



Designed by Robert Rodriguez, Idiel Guerra & Alexander Pastoriza

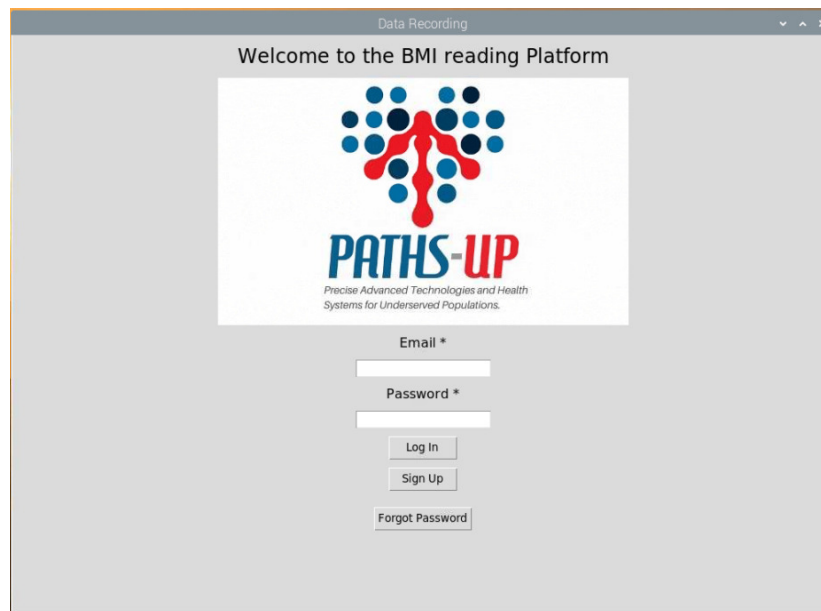
Database Sequence Diagram



Designed by Jose Bello & Gustavo Cordido

Appendix B - User Interface Design

Login Frame



The image shows a login frame for a web application. The window title is "Data Recording". The main heading is "Welcome to the BMI reading Platform". Below the heading is a logo for "PATHS-UP" which consists of a stylized red and blue figure made of dots, with the text "PATHS-UP" in blue and red, and the tagline "Precise Advanced Technologies and Health Systems for Underserved Populations." below it. The login form includes two input fields: "Email *" and "Password *". Below these fields are three buttons: "Log In", "Sign Up", and "Forgot Password".

Data Recording

Welcome to the BMI reading Platform


PATHS-UP
Precise Advanced Technologies and Health
Systems for Underserved Populations.

Email *

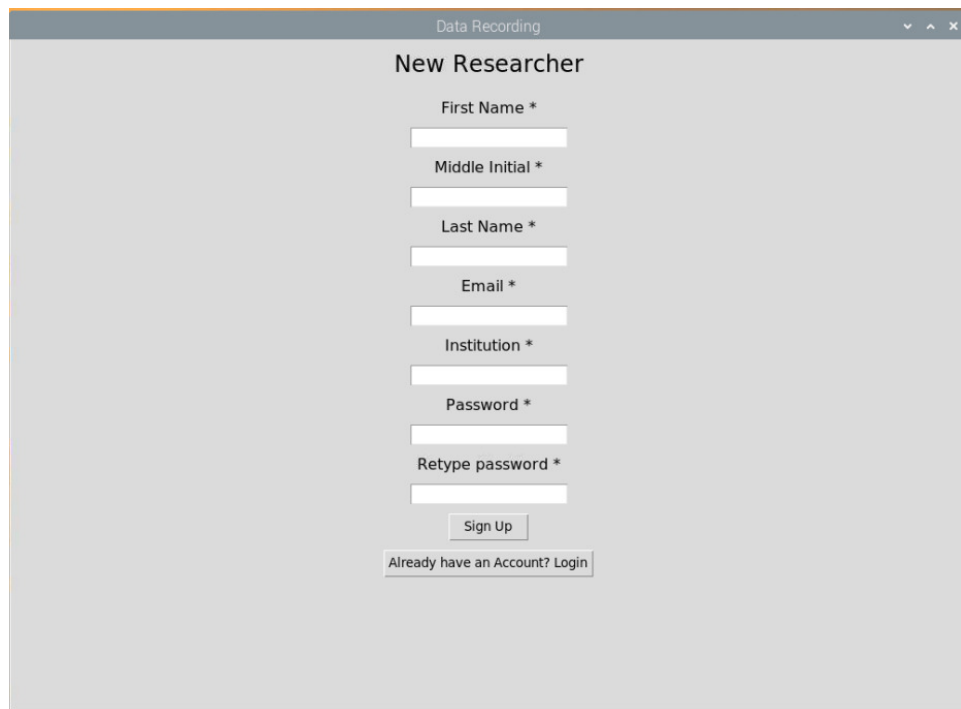
Password *

Log In

Sign Up

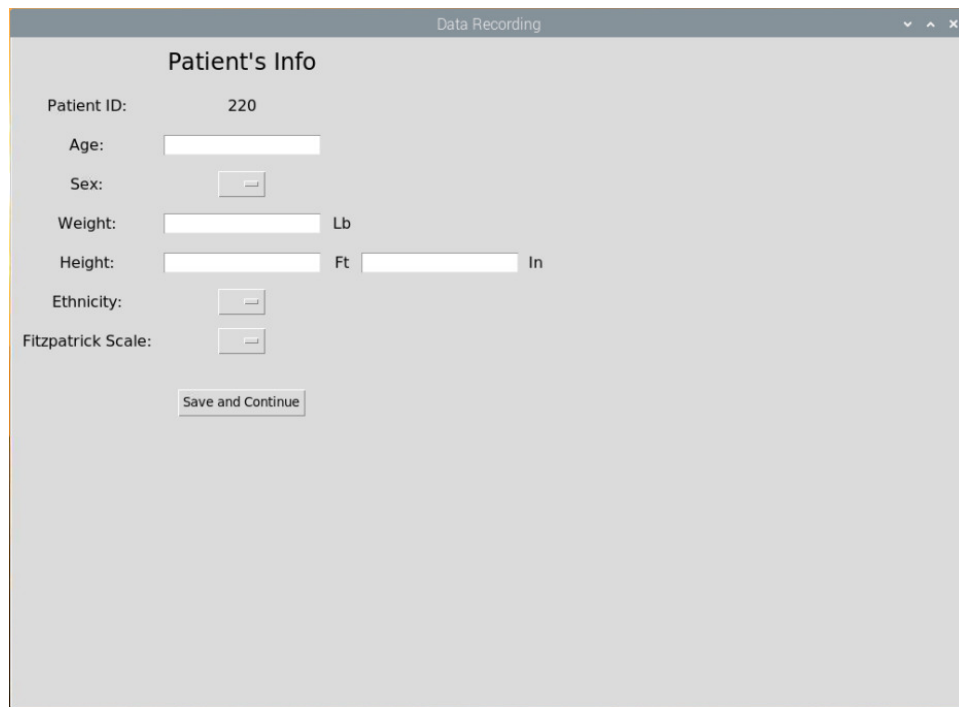
Forgot Password

New Account Creation Frame



The image shows a web application window titled "Data Recording" with a standard window control bar (minimize, maximize, close). The main content area has a light gray background and is titled "New Researcher". It contains a vertical stack of input fields, each with a label and an asterisk indicating it is required: "First Name *", "Middle Initial *", "Last Name *", "Email *", "Institution *", "Password *", and "Retype password *". Each label is positioned above its corresponding white input box. At the bottom of the form is a "Sign Up" button. Below the button is a link that reads "Already have an Account? Login".

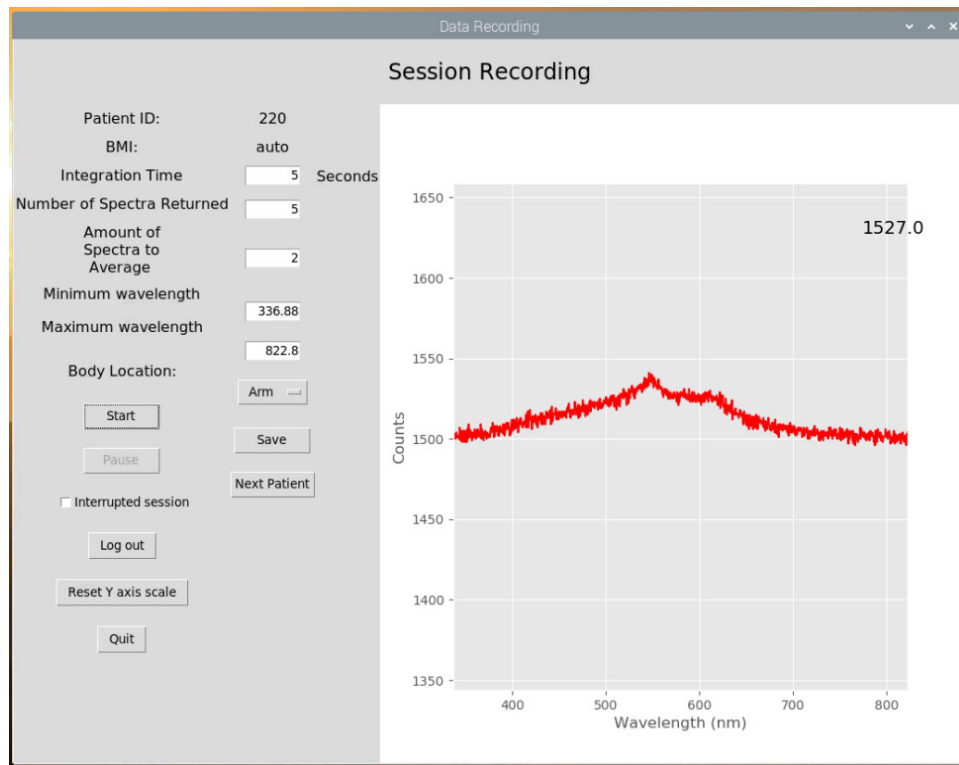
Patient Information Frame



The screenshot shows a software window titled "Data Recording" with a standard Windows-style title bar (minimize, maximize, close buttons). The main content area is titled "Patient's Info". It contains a form with the following fields and controls:

- Patient ID:** A text field containing the value "220".
- Age:** A text input field.
- Sex:** A dropdown menu.
- Weight:** A text input field followed by a unit selector "Lb".
- Height:** A text input field followed by a unit selector "Ft", and another text input field followed by a unit selector "In".
- Ethnicity:** A dropdown menu.
- Fitzpatrick Scale:** A dropdown menu.
- Save and Continue:** A button at the bottom of the form.

Recording Frame



Appendix C - Sprint Review Reports

Sprint 1 Review Meeting Minutes

Attendees: Gustavo Cordido, Jose Bello, Robert Rodriguez, Idiel Guerra, Alexander Pastoriza

Start time: 8:00PM

End time: 9:00PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

User Story #1 Define needed tools and resources

Assigned to: Gustavo Cordido

As a developer part of the project team, I want to know which tools and resources will be needed to first approach this project, such that I may begin the planning phase.

User Story #2 Research on current status of the system and literature

Assigned to: All team members

As a developer, I wish to understand the history of the project and current similar implementations, such that I may further understand its scope and propose appropriate solutions.

User Story #3 Research cloud platform alternatives

Assigned to: All team members

As a developer, I wish to have a list of cloud platforms and their advantages, such that I can discuss our team's choice as needed for the project.

User Story #4 Research on similar devices and microcontrollers

Assigned to: Alexander Pastoriza, Idiel Guerra

As a developer, I want to properly understand the inner workings and software requirements of similar devices to the one this project entails, as well as the state-of-the-art microcontroller technology such that I may implement solutions appropriately.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

N/A.

Sprint 2 Review Meeting Minutes

Attendees: Gustavo Cordido, Jose Bello, Robert Rodriguez, Idiel Guerra, Alexander Pastoriza

Start time: 8:15PM

End time: 9:00PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

User Story #5 Research Azure database management options

Assigned to: Jose Bello, Gustavo Cordido

As a developer, I want to know which database platforms are available in Azure, such that I can help my team decide the appropriate one for the project.

User Story #6 Database Mapping

Assigned to: Jose Bello

As a researcher, I want to have a way to store, access and filter my recordings, such that I may access a specific patient's data using an individual encrypted ID, body part of the recording, and date.

User Story #7 Define table keys

Assigned to: Jose Bello, Robert Rodriguez

As a developer, I want to define each database table's keys appropriately, such that I can ensure an efficient and optimal querying process.

User Story #8 Define type of data storage

Assigned to: Gustavo Cordido, Robert Rodriguez

As a researcher, I want my recordings to be saved in some type of tabular data file, such that is easily accessible and readable by further revisions of the software for analysis.

User Story #9 Familiarize with Raspberry Pi

Assigned to: Alexander Pastoriza

As a developer, I wish to fully understand how to interact with the Raspberry Pi and its requirements, such that I am able to implement it as part of the device and run the software solution in it.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

User Story #10 Basic Interface Design

Assigned to: Idiel Guerra, Robert Rodriguez

As a researcher, I want the software solution to have a simple User Interface that allows my team to control the device via buttons, such that device management can be done with ease.

Sprint 3 Review Meeting Minutes

Attendees: Gustavo Cordido, Jose Bello, Robert Rodriguez, Idiel Guerra, Alexander Pastoriza

Start time: 5:00PM

End time: 6:00PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

User Story #11 Implementing database in PostgreSQL

Assigned to: Jose Bello

As a developer, I want to be able to test my database design locally and in accordance to our Azure deployment selection (PostgreSQL), such that I may test the database system with various queries and pinpoint errors or efficiency issues.

User Story #12 Add researcher table to the Database

Assigned to: Jose Bello

As a researcher, I wish to be able to log in with my personal credentials into the software system, such that the sessions I conduct are linked to my profile and are easily identifiable by me and other researchers.

User Story #13 Set up version control environment

Assigned to: Gustavo Cordido

As a developer, I want to be able to keep track of updates to the code base in the form of a repository, so that any changes done by me or my team mates are accounted for before merging to the main platform.

User Story #14 Azure account connectivity

Assigned to: Gustavo Cordido, Robert Rodriguez

As a researcher, I want the software platform to be linked to my research group's Azure subscription, such that we may all collaborate with the study from the cloud.

User Story #15 Raspbian OS installation and testing

Assigned to: Alexander Pastoriza, Idiel Guerra

As a researcher, I want to have an Operative System running the device's Raspberry Pi component such that I can easily run the software platform appropriately.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

N/A

Sprint 4 Review Meeting Minutes

Attendees: Gustavo Cordido, Jose Bello, Robert Rodriguez, Idiel Guerra, Alexander Pastoriza

Start time: 5:15PM

End time: 6:30PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

User Story #10 Basic Interface Design

Assigned to: Idiel Guerra, Robert Rodriguez

As a researcher, I want the software solution to have a simple User Interface that allows my team to control the device via buttons, such that device management can be done with ease.

User Story #16 Determine security protocols and encryption of Database IDs

Assigned to: Idiel Guerra, Jose Bello

As a researcher, I want to access my account safely and securely, such that only I am able to conduct sessions under it and no patient information is at risk of leakage.

User Story #17 Establish Raspberry Pi-Spectrometer Connection

Assigned to: Alexander Pastoriza

As a developer, I wish to have the Pi and Spectrometer components connected, such that my solutions may control and execute the spectrometer functionalities when ran from the Pi.

User Story #18 Test internal connectivity

Assigned to: Alexander Pastoriza

As a researcher, I want to send signals from the Pi to the Spectrometer, and vice versa such that the connection between both components is evidenced.

User Story #19 Capacitate Product Owners in the use of version control repositories

Assigned to: Gustavo Cordido

As a researcher, I want to manage the code base stored in the established repository, so that I may develop the solution further and apply new features in future revisions.

User Story #20 Adjust buttons to meet research's demand

Assigned to: Robert Rodriguez

As a researcher, I want to have access to a *Start, Stop, Averages, Number of Spectra Returned* buttons to operate the device, such that the spectrometer can be fully controlled via the interface.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

N/A

Sprint 5 Review Meeting Minutes

Attendees: Gustavo Cordido, Jose Bello, Robert Rodriguez, Idiel Guerra, Alexander Pastoriza

Start time: 5:15PM

End time: 6:30PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

User Story #21 Add create account/log in page

Assigned to: Robert Rodriguez

As a researcher, I want to be able to create an account to log in to the system, such that my recordings can be tied to my user.

User Story #22 Create authentication between sign up/log in and database

Assigned to: Jose Bello

As a researcher, I want my account to be stored as persistent data in the database when I create it, such that I can be authenticated through this connection when logging in.

User Story #23 Implement secure authentication for Log In*Assigned to: Idiel Guerra*

As a researcher, I want my account information to be accessed securely such that my information is not at risk from an outsider attack.

User Story #25 Migrate the solution to an MVC architectural pattern*Assigned to: Gustavo Cordido*

As a developer, I wish to have the codebase organized following the chosen architectural patterns, so that accessing/fixing specific services is easily done and issues are quickly identified.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

User Story #24 Implement graphical interface into the recordings page*Assigned to: Alexander Pastoriza*

As a researcher, I want the spectrometer readings to be graphically depicted in the interface while I conduct a recording, so that I may spot errors in the device and ensure that recordings are happening correctly.

Sprint 6 Review Meeting Minutes

Attendees: Gustavo Cordido, Jose Bello, Robert Rodriguez, Idiel Guerra, Alexander Pastoriza

Start time: 7:00PM

End time: 8:30PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

User Story #24 Implement graphical interface into the recordings page*Assigned to: Alexander Pastoriza*

As a researcher, I want the spectrometer readings to be graphically depicted in the interface while I conduct a recording, so that I may spot errors in the device and ensure that recordings are happening correctly.

User Story #26 Adjust graphical representation to depict accurate values

Assigned to: Robert Rodriguez

As a researcher, I want the graphing of my readings to start at 0, rather than the current 1500 value; such that the graphed readings can be seen appropriately.

User Story #27 Implement averages function

Assigned to: Alexander Pastoriza, Gustavo Cordido, Jose Bello

As a researcher, I want to take advantage of the averages function from the spectrometer, such that the returned recordings are averaged by a set amount.

User Story #28 Implement data writing function

Assigned to: Alexander Pastoriza, Jose Bello

As a researcher, I want the averaged recordings returned by the spectrometer to be automatically stored in a tabular file, so that I can easily process it in future revisions.

User Story #29 Azure storage account connectivity

Assigned to: Gustavo Cordido, Jose Bello

As a researcher, I want the raw data files created by the program to be stored in Azure, such that it is backed up in the cloud and it is easily accessible from any computer.

User Story #30 Upload data to Azure

Assigned to: Gustavo Cordido

As a researcher, I want the .csv files produced by the program to be directly uploaded to our Azure storage, such that me and my team can have access to it for future processing.

User Story #31 Implement Hashing for Password Authentication

Assigned to: Idiel Guerra

As a researcher, I want to ensure that my password is secured when I use it to log in, so that my account's information cannot be easily breached.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

N/A

Sprint Review Meeting Minutes

Attendees: Gustavo Cordido, Alexander Pastoriza, Jose Bello, Idiel Guerra, Robert Rodriguez

Start time: 8:00AM

End time: 8:30AM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

User Story #32 Documentation Report

Assigned to: All team members

As a researcher, I want the current software solution to be documented in detail, so that any further revisions to the codebase and new developments are compared to the previous development cycle

User Story #33 User Manual

Assigned to: Alexander Pastoriza

As a researcher, I wish to have access to a user manual for the software, so that I may refer back to it in case of need and share with fellow researchers in the future

User Story #34 Demo Videos

Assigned to: Jose Bello

As a researcher, I wish to have access to a series of videos that demonstrates the usage and functionalities of the project, so that I may refer back to it to understand it.

User Story #35 Pathable Set Up

Assigned to: Gustavo Cordido

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

Appendix D - User Manuals, Installation/Maintenance Document, Shortcomings/Wishlist Document and other documents

User Manuals

Using the STS-VIS spectrometer in tandem with the Raspberry Pi 3 B+ as the core controller unlocked the opportunities for scalability in the future for mobile development into wearable technologies. Raspbian (Full) is the Linux based Operating System that was used to develop our projects user interface to control the various data acquisition functions of the STS-VIS spectrometer.

- Installing Raspbian (Full) comes with all necessary libraries required to operate our software with python-seabreeze at its core. At the Raspberry Pi's current configuration, it is already coded to automatically connect to the spectrometer by USB and begin acquiring relevant data upon completion of the patient's biographical information.
- The opening screen validates login security and new account creation to ensure that only authorized researchers are allowed to access spectrometer functions.
 - Login authorization and new account creation is handled by the Azure Cloud Database which provides extensive capabilities of data management methods and security. Once login handshake is authorized, the user will then be guided through the process of collecting necessary biographical information of the patient before being allowed to collect data.
- Upon completion of the patient's information, the UI display is then focused on a real-time updated graph of the current spectra being analyzed by the spectrometer with the ability to customize the wavelength, averages of spectra collected, intensity in microseconds (capture rate) and length of time in seconds for data to be collected.
- The foundation of this project was to allow our users to have the ability to customize the spectrometer's data acquisition to suit their varying needs and to have a user friendly experience when accessing and manipulating the raw data acquired.
- This is where Microsoft's Azure Cloud database comes in to provide a seamless experience from spectrometer to the cloud where researchers from anywhere in the world can access with proper permissions and utilize the acquired raw data to be processed with their own custom functions. The idea of open source development and custom applications of information gathered remained at the core of every feature we developed.

Installation

1.) Flash Raspberry Pi 3 B+'s SD card with the Raspbian (Full) Operating System

- Raspbian (Complete) comes with all the necessary libraries to operate the spectrometer

2.) Open the terminal once Raspbian has booted (ctrl + alt + T) and run the following code:

- `# Ensure OS architecture is up to date`
- `sudo apt-get update && sudo apt-get upgrade`
- `# Comes standard with the installation of Raspbian (Complete)`
- `sudo apt-get install pip3`
- `pip3 install seabreeze`
- `# Necessary to provide OS development rules`
- `seabreeze_os_setup`
- `sudo apt-get install git`
- `git clone https://github.com/CPRGB/Senior\_Design\_2020.git`

3.) Access the Senior_Design_2020 directory and run the commands

- `cd git Data-recording-UI/`
- `python3 Controller.py`

4.) After sending the last command, the spectrometer's UI should open automatically

5.) If the UI did not open, please refer to the next section concerning troubleshooting

Troubleshooting

1.) Verify that Raspbian (Full) version of the Operating System was installed. If any other version was installed then there are various dependencies that would require manual installation for the software to function. It is highly recommended that if Raspbian (Full) was not installed to begin the installation process from the beginning

2.) Remove the python-Seabreeze folder:

- `sudo rm -r python-seabreeze`

3.) Re-clone pythong-seabreeze:

- `pip3 install seabreeze`

4.) Ensure that the command: "seabreeze_os_setup" ran successfully as this would prevent the software from accessing the spectrometer

- `seabreeze_os_setup`

5.) If none of the above works, the source reference material is available at:

- [Seabreeze Documentation](#)

Maintenance

1.) Update the Raspberry Pi software and libraries weekly by running this in terminal:

- `sudo apt-get update && sudo apt-get upgrade`

2.) Future Senior Design teams will be updating the spectrometer's User Interface that we developed. The location as to where they will post their updates is unknown at this time,

however, the FIU CS Department would have knowledge on the Senior Team that was assigned this project to continue its development.

Shortcomings/Wishlist

Ocean Insight (aka: Ocean Optics) is the company behind the STS-VIS spectrometer's technology and the Seabreeze software that operates the spectrometer's core functions. Ocean Insight discontinued support of their Seabreeze software which became the first major hurdle for our team. Although the company used to provide a Raspberry Pi Software developers kit which came fully equipped to operate the STS-VIS spectrometer out of the box, our solution required a deeper foundational approach.

Shortcomings:

- 'X' button on UI does not allow software to quit at any moment of operation
 - Currently, the only successful way to exit the software is by navigating to the end frame of the UI and clicking the 'quit' button
- 'Forgot Password' is not implemented
- 'Signup' does not have an authentication phase for an admin to approve the new account
- We do not have an active website integrated into the UI

Wishlist:

- Add the ability to exit successfully out of software at any moment of operation
- Implement 'Forgot Password' feature to query relevant username data and email password reset function to user
- Admin authentication for all new user accounts to be approved before automatically being allowed to collect data with the spectrometer
- Host a website to remotely connect to the spectrometer and allow multiple researchers to witness data gathering in real-time

REFERENCES

“Products - Data Briefs - Number 360 - February 2020.” *Centers for Disease Control and Prevention*, Centers for Disease Control and Prevention, 27 Feb. 2020, www.cdc.gov/nchs/products/databriefs/db360.htm.