

Final Report

COMP_129 - Software Engineering - Spring 2020

Immunization Compliance Application (ICA)

Team Members: Angela Ayala - Product Owner
Iris Huang - Data Analyst
Colton Remmert - Interface Designer
Jason Van Bladel - Programming Lead/Scrum Master

Submission Date: 5/1/2020

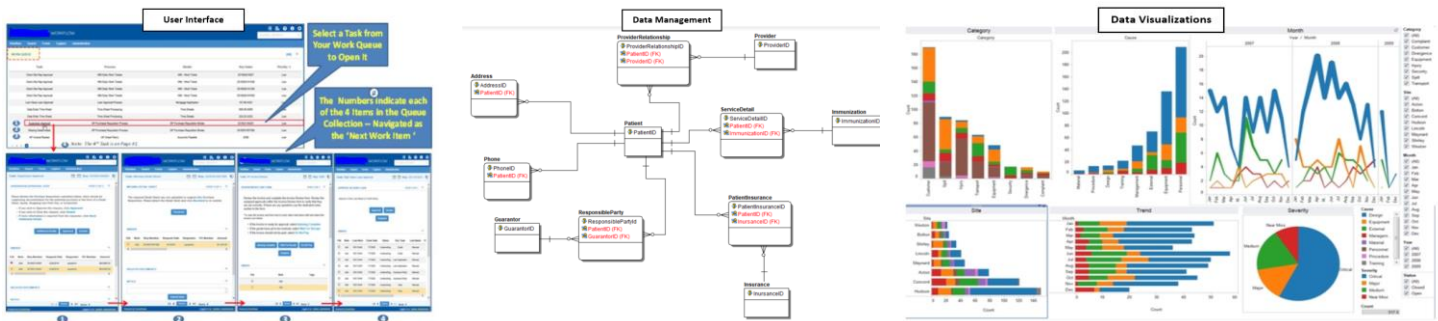
Table of Contents

| | |
|--|---------|
| 1. Summary of Changes..... | 5 |
| 2. Customer Statement of Requirements..... | 6 |
| 3. Glossary of Terms | 7 - 8 |
| 4. Functional Requirements Specification | 9 - 16 |
| a. Use Cases..... | 9 - 14 |
| b. Class Diagram..... | 15 - 16 |
| 5. Nonfunctional Requirements | 17 - 18 |
| 6. System Architecture and System Design..... | 19 - 21 |
| 7. Algorithms and Data Structures..... | 22 - 23 |
| 8. User Interface Design and Implementation..... | 24 - 27 |
| 9. History of Work and Current Status of Implementation..... | 28 - 30 |
| 10. Conclusions and Future Work..... | 31 - 33 |
| 11. References..... | 34 |

Customer Statement of Requirements

Flu vaccination is an important preventative care measure that can reduce flu-related medical care and decrease chances of getting sick with flu-related conditions during flu season. The purpose of our project is to create a tool that supports in the identification and management of patients in need of their annual flu vaccine. Our application will accurately identify patients in need of their flu vaccine, support initial outreach efforts and provide analytical support to the care management team regarding effort-outcomes. With the ever-growing demand of technological resources in the healthcare industry; having a tool that supports recommended practices and allows care teams to effectively manage patients one can create a positive impact in how medical care is used and delivered.

Our application design is inspired by the existing processes taken in the care management realm to identify and manage populations in need of different medical attention. Currently, there exists clinical decision support systems integrated into electronic medical record technology that aid a wide range of roles within the clinical team with identifying patient-specific treatment needs around recommended care guidelines. Our approach is to create a highly intuitive tool strictly focused on identifying patients-in-need to support outreach and compliance that is accessible to any role in the healthcare system including those that are not directly responsible for providing medical attention.



Functional Requirements Specifications

Report 1 Use Case - Implemented

Update patient record from work queue

Click on ICA Icon to log-in.

Enter UserName and Password.

Click “Ok”.

On the home screen, go to the patient list section.

Select line of interest containing patient name, age, phone number, due date.

(After outreach team has called patient - External system)

Double click in patient name.

In the “Patient Detail” screen, go to “Outreach Detail” Tab.

Enter Service Date, Service Outcome, Outreach method.

Click “Ok”.

Report 1 and 2 Use Case - Implemented

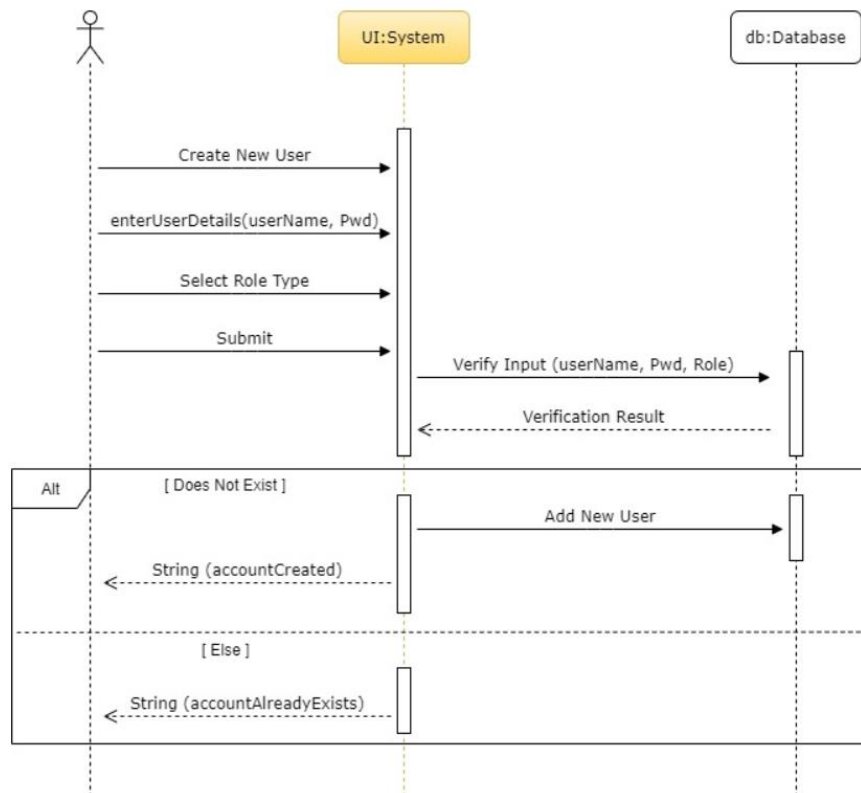
Fully-Dressed Use Cases [1]

CREATE USER ACCOUNT (fully-dressed version)

| | |
|------------------------------|--|
| Related Requirements: | The system allows adding new authorized users at runtime, update roles or inactivate existing accounts. |
| Initiating Actor: | System administrator |
| Actor’s Goal: | System administrator wants to add a new user |
| Participating Actors: | System administrator, Manager, Lead User, User |
| Preconditions: | System administrator must exist New user information must exist |
| Postconditions: | Generated data is stored into the database |
| Flow of Events | → 1. Admin selects application icon available in desktop ← 2. System executes main program files → 3. Admin enters username and password ← 4. System authenticates user account 5. In the Admin menu, System displays option of activities available to Admin (including “Add User” and “Edit User”) |

| | |
|-------------------------------|---|
| | <p>→ 6. Admin selects the “Add User” option</p> <p>← 7. System screen prompts Admin to add: User’s Last Name, First Name, UserName, Role, Active Date, Initial Password and defaults account active value field to “Y”</p> <p>→ 8. Admin populates fields with requested information and clicks “Save”</p> <p>9. System checks that the UserName does not already exist</p> <p>← 10. System returns a confirmation screen displaying new user information</p> <p>→ 11. Admin reviews information and accepts new user account</p> <p>12. System adds the new user record to the database.</p> |
| Alternative Scenarios: | <ol style="list-style-type: none"> 1. If the Admin’s Username and password is not valid the system will return a “Invalid Username or Password” message. 2. If the New User’s Username already exists the system will return a “Username already in use. Please select a different username” message. 3. If Admin selects to “Update” user account, the system will replace screen with “Edit User” screen. |

a. System Sequence Diagram [2] [here](#)

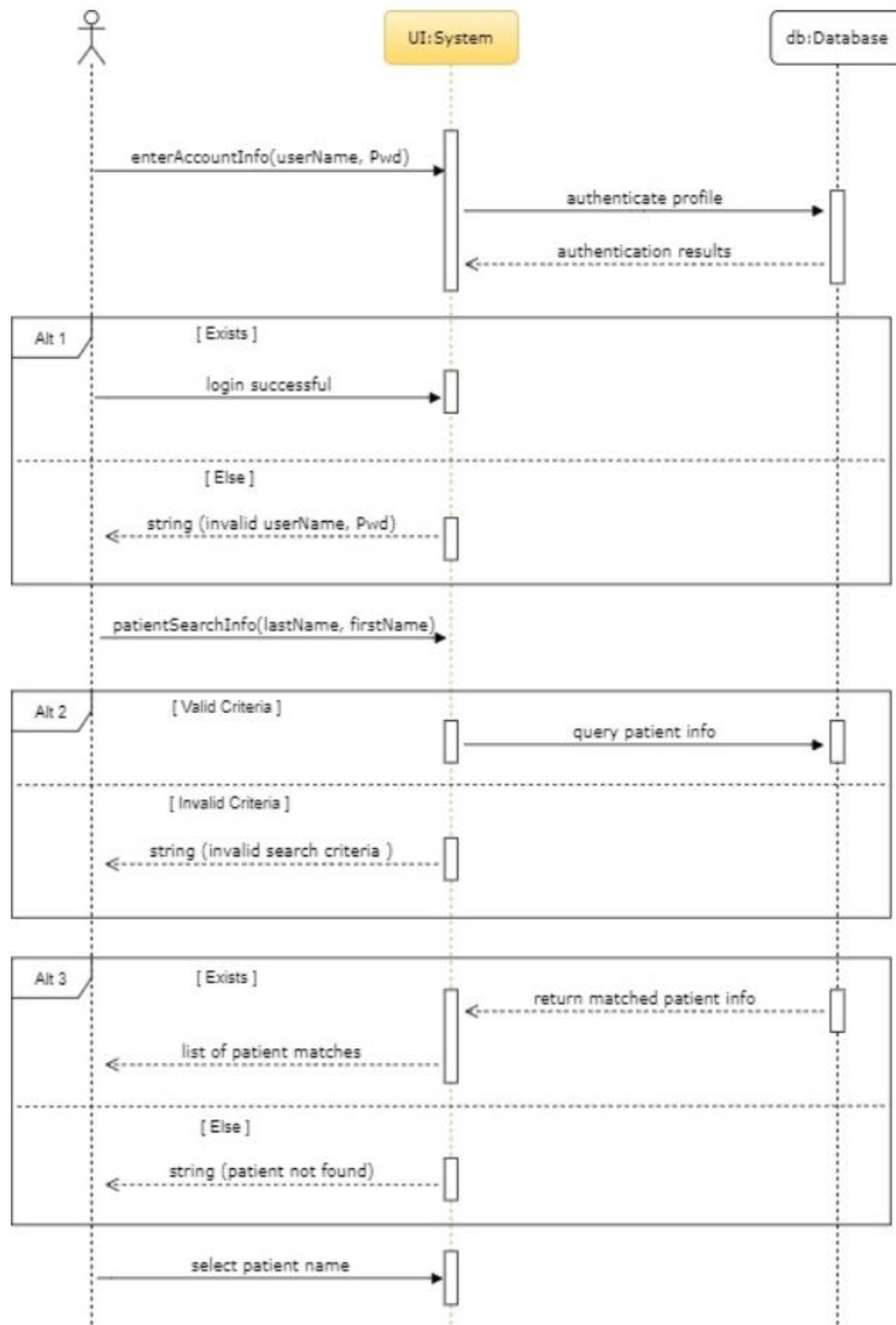


Report 1 and 2 Use Case - Implemented

SEARCH PATIENT RECORD (fully-dressed version)

| | |
|-------------------------------|---|
| Related Requirements: | The system allows querying the database for the retrieval of existing patient information. |
| Initiating Actor: | Patient Service Representative (PSR) |
| Actor's Goal: | Find specific patient record from a list of patients records |
| Participating Actors: | Patient Service Representative |
| Preconditions: | User profile must exist in system User must enter correct user authentication information User must have permissions to search patients in system User must enter the correct search combination |
| Postconditions: | System provides a list of results based on search criteria |
| Flow of Events | <ul style="list-style-type: none">→ 1. User selects application icon available in desktop← 2. System executes main program files→ 3. User enters username and password← 4. System authenticates user account5. In the home screen, user enters valid search criteria: {patient's last name and first name, patient date of birth, patient medical record number, etc.}→ 6. User clicks "Search"7. System queries database for entered patient information8. Database returns list of records to system← 9. System returns list of patient names, date of birth and medical record number→ 10. User selects patient name for list of patient names and clicks "OK"← 11. System opens the patient's demographics screen for the selected patient name |
| Alternative Scenarios: | <ul style="list-style-type: none">1. If the user's username and password is not valid the system will return a "Invalid Username or Password" message.2. If the user enters an invalid search combination the system will return an "Invalid search criteria. Use one of the following search combinations: First Name and Last Name or Date of Birth or Medical Record Number."3. If patient information is not found the system will return a "Patient not found." |

b. System Sequence Diagram [here](#)

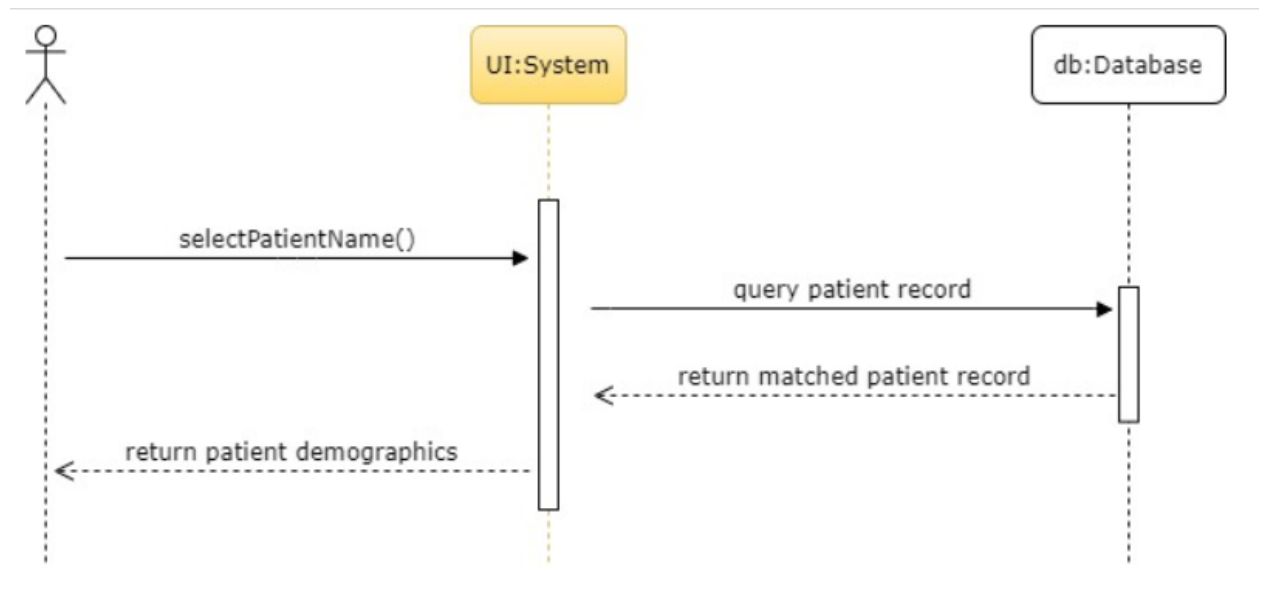


Report 1 and 2 Use Case - Implemented

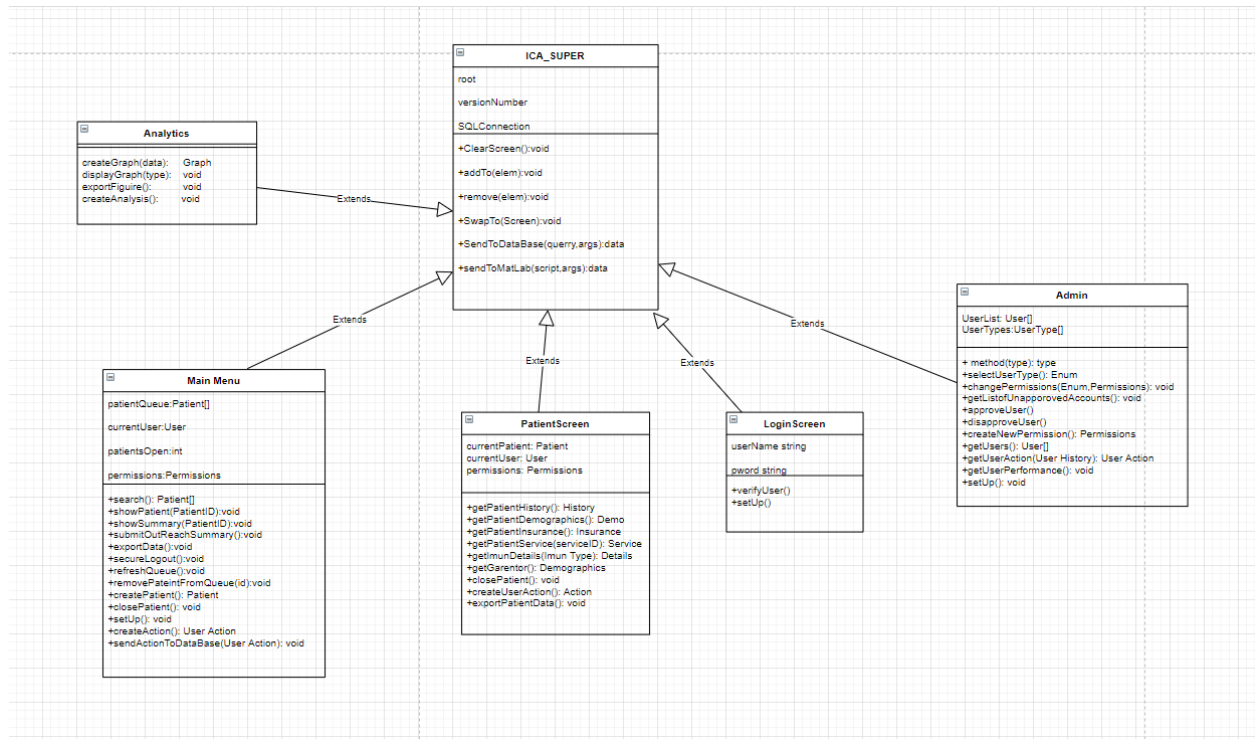
ACCESS PATIENT INFORMATION (fully-dressed version)

| | |
|-------------------------------|---|
| Related Requirements: | The system allows querying the database for the retrieval of detailed patient information. |
| Initiating Actor: | Patient Service Representative (PSR) |
| Actor's Goal: | Display detailed patient information { demographics, guarantor, insurance, service history and outreach details } |
| Participating Actors: | Patient Service Representative |
| Preconditions: | User profile must exist in system User must enter correct user authentication information User must have permissions to access patient detailed information Patient record must exist in database Patient detailed information must exist in database |
| Postconditions: | System displays detailed patient information { demographics, service history, outreach details } |
| Flow of Events | → 1. User selects patient name and clicks "Expand or OK" 2. System queries database to search selected patient record 3. Database returns field values to system ← 4. System displays patient demographic information in new window { Patient Name, Age, Date of Birth and Contact Information } |
| Alternative Scenarios: | <ol style="list-style-type: none"> 1. If patient information is not found the system will return blank in the corresponding field. 2. If the user clicks on the "Service History" tab the system will replace the current window display with the service history tab { Service Date, Immunization Name, Service Outcome }. 3. If the user clicks on the "Outreach Details" tab the system will replace the current window display with outreach details tab { Outreach Date, Outreach agent, Outreach Method, Outreach Outcome }. |

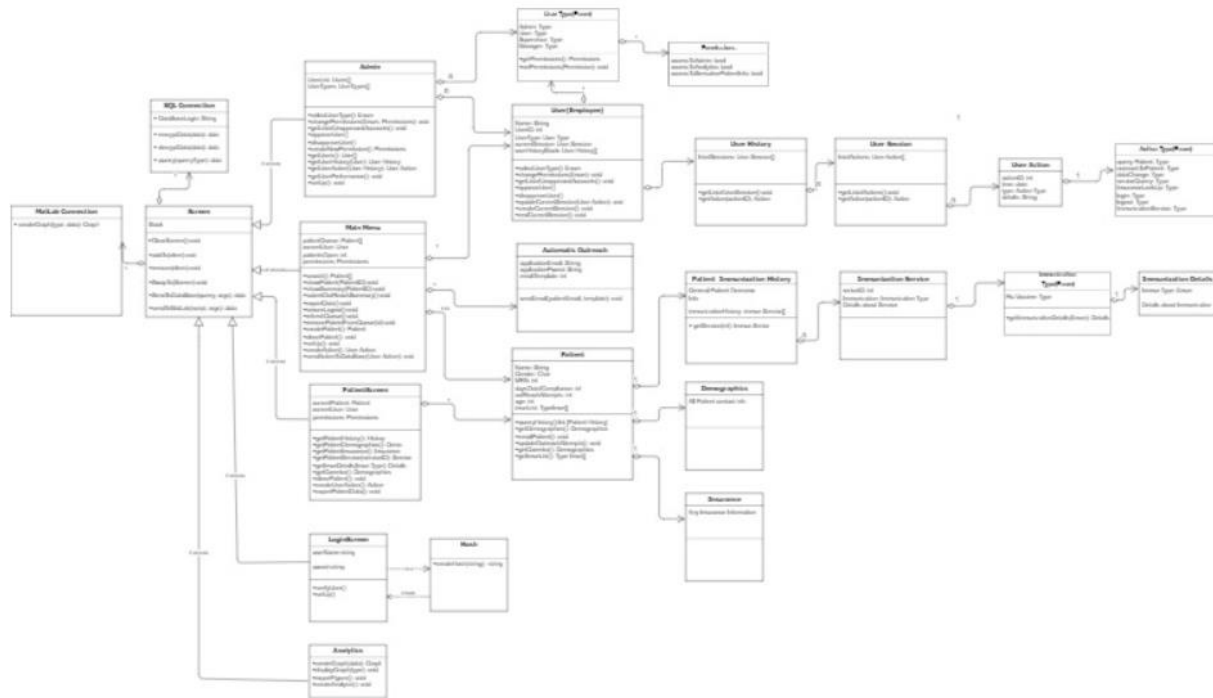
c. System Sequence Diagram



Class Diagram



Complete UML diagram



Nonfunctional Requirements

Functionality

- Identify patients that are not in compliance with their flu vaccine.
- Initiate outreach efforts to patients in need of their flu vaccine.
- Access patient information that includes contact information, medical coverage and immunization history to effectively and conveniently reach out to a patient.
- Update patient information as soon as it becomes available.
- Protect patient information confidentiality.
- Track outcomes of outreaching efforts.

Usability

- Fuzzy search is implemented for the search function of this program.
- Users can arrange work queues based on their work preference.
- Home screen design includes quick access sections.
- Mouse and keyboard functionality is implemented.
- Exporting of files available in commonly used formats .csv, .txt

Reliability

- Query scripts based on SSL certificates to create secure connections to prevent SQL injections.
- Only administrator roles are allowed to create and delete accounts.
- Login error is displayed when username and password does not match.
- Three attempts of failure login, the account would be locked, reset password required.
- Limited number of charts open per user.
- Expanded information on patients is limited to five pop out windows.
- Pop up messages are implemented to warn users of incorrect usage of functionality.
- Users are assigned permissions based on their role. Higher roles have more permissions to access more functions in ICA.
- Text checker when sending an email will prevent users from sending nothing to patients.
- Data changes are stored in user history.

Performance

- Performance is dependent on the speed of the query to the database.
- Search criteria and filtering may also affect performance depending on the size of the work queue.
- Users are able to work concurrently. Multiple users querying the database may affect performance when doing certain functions.
- Minimum disk space to hold the executable file of the program.

Supportability

- The system can be extended to support a wider variety of immunizations for patients.
- For now, ICA is mainly targeting immunizations, but if progress goes well, we could add more health issues.

Implementation

- This software is a desktop application.
- It could run on Windows, Mac OS, and Linux environments.

Interfaces

- Login page, main page with work queue, search, report function, and patient information page is implemented in the existing program.
- Certain screens are accessed as an extension from the main screen. These screens are limited to only five pop outs.
- Patient screen has multiple extended functionalities in a separate panel on screen, different categories extend different features that are accessible.

Operation

- Managers are responsible for configuring work load balancing.
- Information Technology is responsible for troubleshooting, software maintenance and updates.

Packaging

- The information technology department installs the system.
- There will be one installation per device.

Legal

- This software piece is open-source for the current stage.
- Since we are utilizing randomized patient data, we are compliant with HIPAA laws.

System Architecture and System Design

Architectural Styles[3]

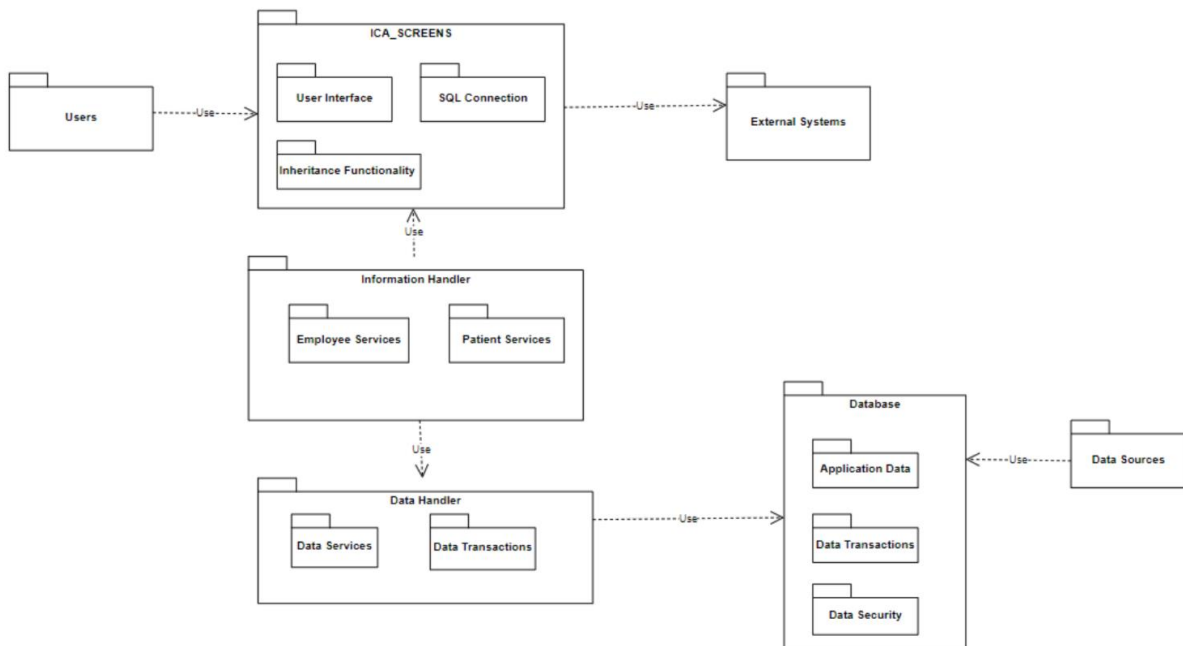
Data-Centric Repository, Object-Oriented Architecture

Data-Centric Repository that is used by various components of the program. The repository is composed of a relational database system that follows a database schema in which data is stored in different tables inside the database that is accessible to different parts of our program. This type of architecture supports data-integrity and it also allows processes to be independently executed.

Object-Oriented Architecture which supports our program's division of responsibilities into classes that are then supported by functions or methods. This allows us to easily extend new functionality, test and maintain the program without affecting it all in its entirety.

Identifying Subsystems

Draw and describe **UML package diagram** of subsystems in your system



Mapping Subsystems to Hardware

Does your system need to run on multiple computers?

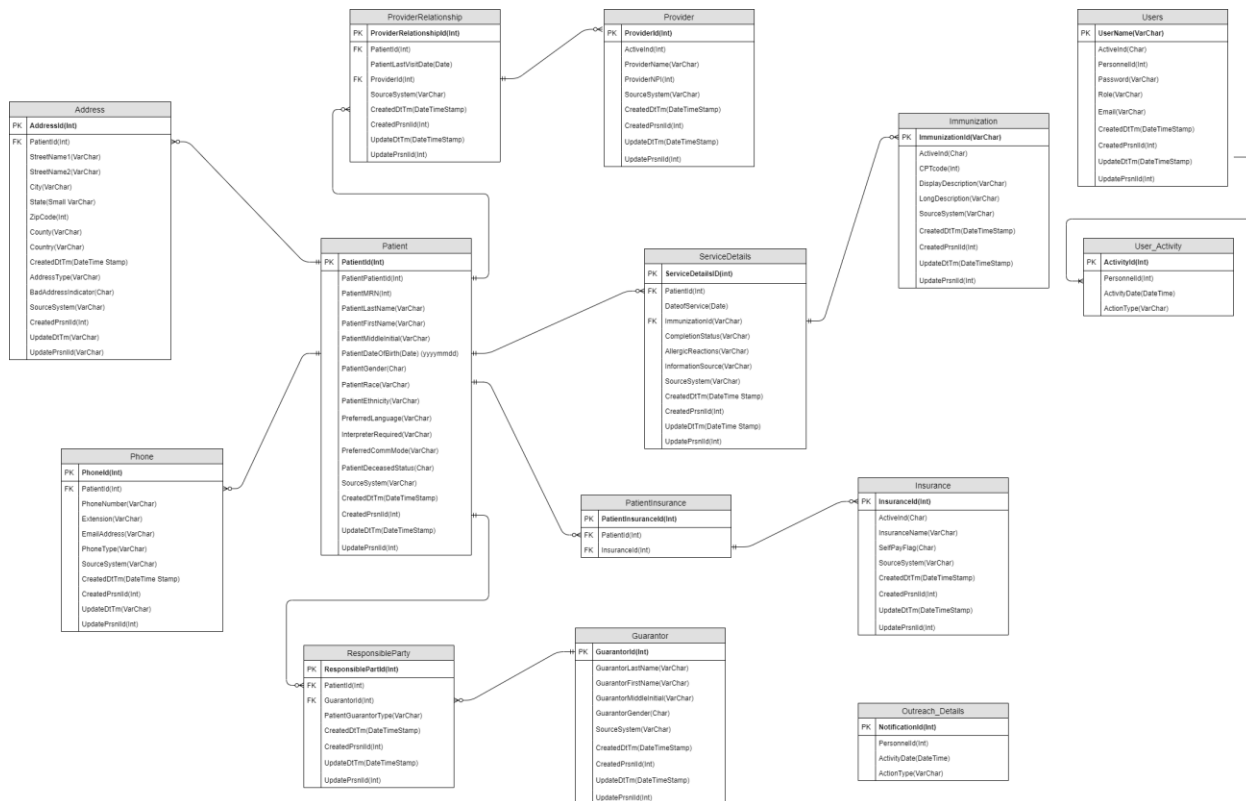
The system is able to be run by a single client or multiple clients, there is a database server connection that is required.

Persistent Data Storage

Does your system need to save data that need to outlive a single execution of the system?

Yes, user, patients, demographics, guarantor, insurance, contact, address, history, last service date

The storage management strategy is through a relational database.



Link to database schema [here](#).

Network Protocol

We use Simple Mail Transfer Protocol (SMTP) to send electronic mail transmissions, we opted to use it because it is supported by Python, simple to implement and it is the most reliable for sending email to customers and patients.

The Simple Mail Transfer Protocol is also the most cost-effective way to send a large number of emails to a large group of people, hence for our application, it is easy to integrate and send patients updates about their immunization status [4].

The application uses an ODBC (Open Database Connectivity) driver to connect to the

Amazon RDS database Instance. This driver uses multiple network protocols including NetBEUI, TCP/IP, DECnet, and SPX/IPX and are specific to each network. ODBC driver is not just one it uses multiple based on the system.

Global Control Flow

- Execution order:

The application is event-driven listening for the users inputs. The actions of the user can be executed in different orders to execute the same outcome because the program is set up to listen to a series of triggers to execute various events.

- Time dependency:

Yes, our application implements a timer to show the users the time within the application. The timer also refreshes the connection to the database on a time set by an admin in the system. This is used to prevent the connection from being lost over a long session.

- Concurrency: Does your system use multiple threads?

No, our application does not utilize multiple threads.

Hardware Requirements

| SYSTEM REQUIREMENTS |
|---|
| MINIMUM |
| OS: Windows 7, 8, or 10 (32-bit or higher) |
| Processor: Core 2 Duo 1.8 Ghz (or equivalent AMD) |
| Memory: 4GM RAM |
| Graphics: 1024 x 768 resolution with 32-bit colour depth |
| Storage: 500 MB available space |

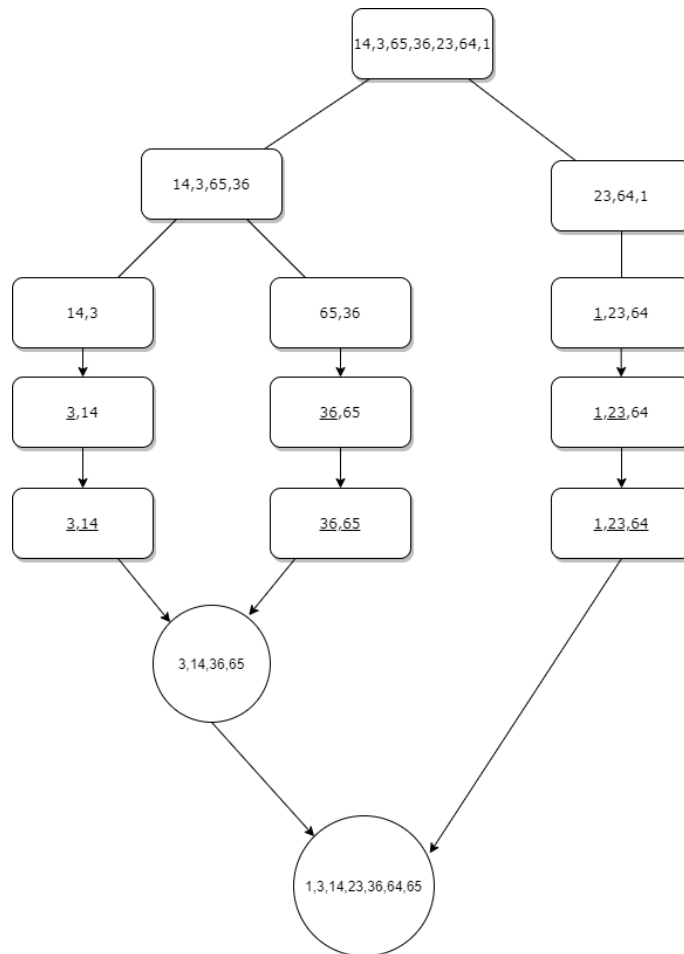
Algorithms and Data Structures

Algorithms

The python list built-in sort algorithm, Timsort, is being used to provide users the ability to sort patient lists to their preference. Timsort is a hybrid algorithm based on merge sort and insertion sort. The runtime complexity of the Timsort algorithm is $O(n \log (n))$ for worst cases and $O(n)$ for best case, which is one of the fastest sorting algorithms available.

The algorithm first divides the array into subsets known as Run, then sorts each individual block of data using insertion sort due to the property that insertion sort runs better with smaller arrays, lastly it merges the subarrays using merge sort. If the size of the list is smaller than Run, the list will be sorted using insertion sort only.[5]

The following diagram shows an example of how Timsort works. The rounded rectangles illustrate how the list changes during insertion sort, and the circles demonstrate how merge sort works.



Data Structures


Our application contains a lot of data that is being displayed to outreach specialists in order to make sure patients are getting the immunizations they need. The two main data structures we used for our application are list and dictionaries. Since we are concerned about a lot of data, the efficiency of the program is the main focus when we were choosing the right data structure to use. Lists are used to hold data retrieved data from the database and populate on the screens. The advantage of using lists is its usability. It is easy to create and use. It only takes $O(1)$ for direct indexing and $O(n)$ for sequentially access. Dictionaries are also used because they can contain a lot of information that has quick look up time, making them extremely time efficient. For most cases, dictionaries are $O(1)$ case complexity. Our program is going to deal with a large amount of data that query from the database therefore array/list and dictionaries are going to improve the performance of our program. The table below provides list and dictionary run time complexity for common operations on average cases.

| | access | search | insertion | deletion |
|---------------------------|--------|--------|-----------|----------|
| array/list | $O(1)$ | $O(n)$ | $O(n)$ | $O(n)$ |
| Hash table/ dictionary | N/A | $O(1)$ | $O(1)$ | $O(1)$ |

User Interface Design and Implementation

Screen Implementations/Modifications

Login Screen: The login screen was implemented just like the mock ups. The login screen contains only widgets that either display text, image, text boxes for user input, and buttons that contain functionality. By clicking on a button, a specific scenario will occur based on the information provided on the screen such as denied access or ability to request a new account creation. We did however add a few things to the screen such as the version number for the screen along with the “add account” button which allows a user to request access from the system admin to use ICA.



The screenshot displays the login interface for the Immunization Compliance Application. At the top, the application's name is written in a large, blue, sans-serif font, with a stylized logo to its right. Below the title, there are two input fields: one for the username, which contains the text 'AUser', and one for the password, which is masked with asterisks. To the left of each field is its respective label. Below the password field, there are three buttons: a 'Login!' button, a 'Cancel' button, and an 'Add Account' button. The 'Add Account' button is positioned below the other two. At the bottom left of the screen, the text 'Version 2.1.2' is displayed.

Immunization
Compliance
Application

Username: AUser

Password: *****

Login! Cancel

Add Account

Version 2.1.2

Main Menu: From the initial mock up, the main menu has changed very slightly. It still contains all the functionality we were looking for, however the search field buttons have a more improved design. The interface contains a menu frame that contains access to different functionalities from the main menu contained in tabs such as; File, Help, Analytics and Admin controls. The three fields that contain the

queue, search field, and patient summary are all separate frames that can be expanded or shrunk based on user preference. The search frame contains a drop down menu that contains multiple different options for the user to search the patient queue on. Each one is then followed by a check button that, when checked, will be utilized on the queue to filter it based on the selected options by the user. The patient queue frame contains a list of buttons that contain basic information on a patient. These buttons will then pop out additional information on the patient to the patient summary frame. If a user wants to display more information on a patient, there is a button to do so in the patient summary frame that will display a pop out window for the patient screen.

The screenshot displays the 'Immunization Compliance Application (Version 3.0 FINAL)' interface. It features a horizontal menu bar with 'File', 'Help', 'Analytics', and 'Admin' tabs. The 'Admin' tab is active, showing the user 'Admin' at '05:31 PM'. The main content area is divided into three sections: a search and filter panel on the left, a patient queue table in the center, and a patient summary panel on the right.

Search and Filter Panel: Includes a 'Search:' text input field. Below it are checkboxes for 'First Name', 'Last Name', 'Date of Birth', 'MRN', and 'Gender'. A 'Default Work Queue' button and a blue 'Search' button are present. The 'Queue Filters' section includes checkboxes for 'First Name', 'Last Name', 'Days Overdue', 'Sex', 'Immunization', and 'Age', each followed by a dropdown menu. At the bottom are 'Default' and 'Filter' buttons.

Patient Queue Table: A table with columns 'First Name', 'Last Name', 'Due Date', and 'Days Overdue'. The row for 'Larae Andrade' is highlighted in green.

| First Name | Last Name | Due Date | Days Overdue |
|------------|------------|------------|--------------|
| Raymond | Cantu | 01/06/2018 | 845 |
| Robena | Lin | 01/30/2018 | 821 |
| Bethany | Valenzuela | 02/22/2018 | 798 |
| Anisa | Holt | 02/24/2018 | 796 |
| Micheline | Graves | 03/03/2018 | 789 |
| Larae | Andrade | 03/13/2018 | 779 |
| Ellsworth | Mcdowell | 03/27/2018 | 765 |
| Karena | Joseph | 04/24/2018 | 737 |
| Muriel | Graham | 05/09/2018 | 722 |
| Alvina | Haley | 05/16/2018 | 715 |
| Jarred | Frost | 06/08/2018 | 692 |
| Darwin | Barber | 06/20/2018 | 680 |
| Elliott | Burton | 08/23/2018 | 616 |
| Ling | Nicholson | 08/24/2018 | 615 |
| Forrest | Joseph | 09/01/2018 | 607 |
| Ivan | Jordan | 09/05/2018 | 603 |
| Dominick | Norris | 09/06/2018 | 602 |
| Rosalba | Reyes | 09/13/2018 | 595 |

Patient Summary Panel: Contains tabs for 'Summary', 'Medical History', and 'Contact'. The 'Summary' tab is active, displaying patient information for 'Larae Andrade': 'First Name: Larae', 'Last Name: Andrade', 'Date of Birth: 1935-11-23', 'Age: 85', 'Sex: Female', and 'Race: Native Hawaiian or Other Pacific Islander'. At the bottom are 'Outreach' and 'Expand Patient' buttons.

Patient Detail Screen: This screen contains the most changes from the mock up provided. After trying the initial mock up, the screen looked too crunched together with a lot of information clustered together. The horizontal menu bar containing the buttons to perform certain tasks along with an empty extension box always being open looked out of place. To remedy this, the screen's size was increased along with removing these widgets. The patient detail screen is a pop out window from the main screen. The root of this screen's implementation only contains three widgets; the headline widget for the patient's basic information, the notebook widget which contains most if not all of the widgets on this screen, and

the extension widget which contains a small amount of information. With the notebook widget, we can easily hide and display different frames that are appended to the notebook's tab list. The notebook widget displays all the main information that is important to this screen such as the service details, patient demographics, outreach reports, and insurance information. With this is an extension frame that can be utilized to include a variety of different functions for the user to utilize. This can range from, displaying information in the extension to be referenced, displaying a more detailed description of an immunization received, and a variety of methods to contact a patient. The Service Detail tab contains a header along with a list of previous services that the patient has. It ranges from different Influenza vaccines that the patient has gotten, the date it was received, and if it was administered. When clicking on a service button, it will open a new display showing additional information about the service. On the Outreach Page, we have a frame of the contact info to reference again, an outreach form which can also display previous outreach attempts, and an email template where the user can email the patient. Finally we have the Immunization tab which contains all of the immunizations that a patient has received and the dates when they were administered. When clicking on the "learn more" button, a user can get more information from the CDC about that specific Immunization.

| PATIENT: Larae Andrade | | | | GENDER: Female | | AGE: 85 | | DOB: 1935-11-23 | | MRN: 99617 | | | | |
|--|----------------------------------|--|-----|--|----------------------|---------|--|-----------------|-------------|------------|--|--|--|--|
| <div>Demographics</div> <div>Service History</div> <div>Outreach Report</div> <div>Immunizations</div> | | | | | | | | | | | | | | |
| Patient Demographics | | | | | | | | | | | | | | |
| Last Name | Andrade | | | D.O.B | 1935-11-23 | | | | | | | | | |
| First Name | Larae | | | Age | 85 | | | | | | | | | |
| Nickname | Larae | | | Race | Native Hawaiian or O | | | | | | | | | |
| Middle Initial | | | | Ethnicity | Unknown | | | | | | | | | |
| Prefix | | | | Pref. Language | English | | | | | | | | | |
| Sex | Female | | | Deceased Status | No | | | | | | | | | |
| Patient Address Details | | | | | | | | | | | | | | |
| Street 1 | 15 Pulaski Ave | | | City | Stockton | | | Zipcode | 95220 | | | | | |
| Street 2 | | | | State | CA | | | County | San Joaquin | | | | | |
| | | | | | | | | Country | USA | | | | | |
| <div>Contact Information</div> <div>Guarantor Information</div> <div>Insurance</div> | | | | | | | | | | | | | | |
| Home Number | | | | Contact Notes | | | | | | | | | | |
| Mobile Number | | | | 2020-12-02 J.VanBladel >>> Patient asked to have IZ appoint scheduled for next week. Clinic rep will call back with appt time and day. | | | | | | | | | | |
| Work Number | (312) 959-2564 | | Ext | | | | | | | | | | | |
| Email Address | immunizationcompliance@gmail.com | | | | | | | | | | | | | |
| Preferred Mode of Contact | e-mail | | | | | | | | | | | | | |
| Interpreter Required | No | | | Update Contact Notes | | | | | | | | | | |

Ease-of-use interface

We have implemented our interface to be as user friendly as possible. We have considered the visibility of all the widgets that are clickable on the screen by making them larger and contain the text as to what they perform when clicked. We have been consistent with similar applications by referencing their user interface when developing our mock up screens. This way users who have work experience similar to an outreach specialist can easily use our interface. We have also been consistent with our font and text sizes to ensure that everything is readable and legible for users.

Conclusions and Future Work

Angela R Ayala

o Key Accomplishments

- Database Design
- Data Management
- Query Creation
- Documentation
- Project Management
- Leadership

o Describe the technical challenges you encountered in the development of your software product.

Python was a fairly new programming language for me, I was not comfortable with the syntax or the different libraries available to incorporate all of the functionality needed in our application however our team was very strong in a wide variety of skill sets so we were able to all learn and support each other.

o Describe how the software engineering techniques you learned in this course helped you to address those challenges.

Documentation - Allowed us to create and share a vision of what we wanted our project to be and adjust effort based on our progress.

Project Management - The management methodology we approached the project with allowed us to break apart the massive tasks we had into smaller much more manageable sprints that allowed us to complete and show progress week by week.

System Design - Supported us in realizing what best tools and approaches to take to achieve a product that would meet our vision.

o Describe what other knowledge you feel might have helped you with the project development.

I would've liked to have a greater comfort level with my programming ability. I was able to support my team with the user interface during the last weeks of the project but would have liked to have made a more significant contribution to the team in this area.

o Describe in what manner each of other team members support your work.

Our team was blessed with a multitude of talented individuals that complemented each other very nicely.

Colton - He is our user interface design master, he was responsible for the design and implementation of the graphical components of our user interface. As well as made sure that every screen built was in harmony with the data attributes stored in our database, and gave me a lesson or two in Python.

Iris - She is our analytical all-round expert, she supported me with query building. Implemented fuzzy search and the analytics functionality of the application.

Jason - He shone in every area of this project, he is our pantologist. He troubleshooted, implemented email transmissions, user permission management and integrated the database with the user interface.

o Discuss possible directions for the future work on this project.

This project was designed with scalability in mind, possible future work would be incorporating notification systems, a wider schedule of immunizations and integration with a health systems electronic medical record.

Iris Huang

o Summarize (as a bulleted list) your key accomplishments in this project.

- Search and sort functionality
- Query generator for search and sort functionality
- Analytic diagrams model
- Assisted to populate data from the database to our program
- Designed the logo for our program

o Describe the technical challenges you encountered in the development of your software product.

I have little knowledge of GUI libraries in Python, therefore I had a difficult time dealing in integrating the GUI with my code. Specifically, I did not know where to create the connection between the GUI and my code. Moreover, I have little knowledge of the SQL database and query. In the beginning, many tasks seemed impossible for me.

o Describe how the software engineering techniques you learned in this course helped you to address those challenges.

Knowing the waterfall and agile technique, we have incorporated both techniques for our project. Breaking down each task into small pieces helps to focus on one target at a time. Frequent meetings help everyone to stay on the same page and ensure the quality for our project, and I am able to seek help from other team members utilizing that time.

o Describe what other knowledge you feel might have helped you with the project development.

As for me, more knowledge with GUI would help me to move quicker with my development process and have more time to produce aesthetic diagrams. It would be nice if I am more familiar with more python libraries.

o Describe in what manner each of other team members support your work.

Angela - Angela setted up the database that has data for me to work with on my analytics work. She always keeps the team on the right track for the project. With her meeting notes, everyone is able to be on the same page for what needs to be done. I have also received help from her on SQL queries for diagrams.

Colton - Colton established screens with sections for where things should be placed in. With his knowledge of the Tkinter library, I was able to integrate the diagram in the appropriate place. He also helped to have the "about us" page able to display in the program.

Jason - Jason initialized the foundation of our program having all the screens connected. He helped me with the integration for the sort and search functionality to the GUI. He also helped in the debugging phase.

o Discuss possible directions for the future work on this project.

There are many areas this program can grow. However, if the time is permitted, I would create a new screen that can generate more diagrams and models to perform analytics on the dataset that can help the managers and other users to better understand the data, and be able to utilize those diagrams to make decisions to improve the teams' efficiency.

Colton Remmert

o Summarize (as a bulleted list) your key accomplishments in this project.

- Gui Inheritance functionality
- Implementation of Login Screen
- Assisted implementation of Main Menu
- Implementation of Med_info_screen
- Exporting functionality
- Screen swapping functionality

o Describe the technical challenges you encountered in the development of your software product.

For me the biggest challenge was figuring out how to get SQL to work with the widgets I used for the GUI. So to overcome this I noted lists that were being utilized by the GUI to Angela and Jason to create SQL queries to have data from the database to be displayed.

o Describe how the software engineering techniques you learned in this course helped you to address those challenges.

By making incremental changes and more frequent meetings, my teammates were able to help me to overcome some of the challenges with querying the database. Also by leaving more comments for team members helped my teammates to understand where to modify the code to add the SQL connection..

o Describe what other knowledge you feel might have helped you with the project development.

I feel if I spent a bit more time researching different GUI's it would have been a lot simpler to make the interface better.

o Describe in what manner each of other team members support your work.

Angela - Angela helped me design the interface to be what it currently is today. When I had trouble deciding what would be a better choice how a user might receive the data I would ask her. Angela also created the database which is crucial to the interface. She also helped me create a lot of the notebook tabs that are displayed on the Med_info_screen. She also made the queries that were used.

Iris - Iris helped with a lot of the functionality the screens have. Especially with the main screen where she included the search functions, analytics screen, and helped out with a lot of the querrying. Iris helped me out on the service detail page when we were connecting the GUI to the database.

Jason - Jason helped out with a lot of the GUI and the queries. Jason had a helping hand in almost all of the development that was done in the project. He implemented the Main Menu screen to have the functionality it currently has and he helped out with a lot of the queries that were utilized by the team. He identified areas to fix bugs and helped to improve the quality of the code.

o Discuss possible directions for the future work on this project.

This project has plenty of potential for continued development. We have a scalable foundation that, while currently only supports one type of immunization, could support a much wider variety of immunizations. The first thing we would do in future development would be code refactor then onto more implementation of the functionality on the menu tabs in the main screen. There are a lot of things we could add to this project such as web support, implement the user history, and additional admin features but we would mainly focus on appending functionality to the program.

Jason Van Bladel

o Summarize (as a bulleted list) your key accomplishments in this project.

- Implementation of Main Menu
- Automatic Emailing of Patients
- Secure Login with Permissions
- Secure refreshing database connection
- Building Queries
- Admin Permissions
- UML Diagram

o Describe the technical challenges you encountered in the development of your software product.

I began the project with little to no experience with databases, so I had many technical challenges setting up the database connection. These challenges included writing and updating to the database, maintaining a connection over a span of time (longer than 5 minutes), and creating a SQL code reader. All of these had their own unique challenges I had to overcome to complete the final product.

o Describe how the software engineering techniques you learned in this course helped you to address those challenges.

To approach this project I broke down each major feature into individual problems that I could approach easily and efficiently. The most important aspect of software engineering that I learned was continuous integration with the scrum process. This type of planning and teamwork makes the project easier to approach and focuses on the overarching goal of creating the best product.

o Describe what other knowledge you feel might have helped you with the project development.

The knowledge that would have helped me with project development is knowledge of how you can connect databases with other programs such as Python or Java because a lot of research and experimentation happened to create the secure connection.

o Describe in what manner each of other team members support your work.

Iris: Helped implement the sorting feature into the user interface and expanded the main menu screen to prevent various text from being cut off. She also did a great job helping with the SQL connection by figuring out the correct libraries.

Colton: He did a great job working on the user interface with me by teaching me how to use the tkinter library and implementing features into many of the menus. Colton also did a wonderful job by implementing the email code into his UI and by adding the various templates I had previously made.

Angela: She did a great job working with me to make the queries for the application. Whenever I had an SQL error I went to Angela because she was so knowledgeable on the language and wrote very essential code to make the queries work as they were designed.

o Discuss possible directions for the future work on this project.

Future work includes implementing additional admin features such as notifications, user history and system settings. We could also work on implementing the system to be compliant with HIPAA laws so it could be theoretically used

for real patients. Another way to expand the project would be to include multiple immunizations besides just influenza.

Glossary of Terms

Centers for Disease Control and Prevention: one of the major operating components of the Department of Health and Human Services in the United States, whose mission is to increase health security.

Flu Season: In the United States flu season usually runs between October and May.

HIPAA: Health Insurance Portability and Accountability Act of 1996, law in the United States that provides data privacy and security provisions for safeguarding medical information.

ICA: Immunization Compliance Application, desktop application used for outreaching and managing patient populations in need of flu vaccines.

Immunization Compliance: Immunization compliance is the main focus of the ICA system. This term refers to a patient's status regarding meeting recommendations for the administration of the influenza vaccine as it pertains to the Centers for Disease Control and Prevention's standards.

Influenza: an acute, highly contagious, respiratory disease that causes thousands of deaths annually in the United States.

Influenza Immunization: vaccine used to stimulate the production of antibodies and provide immunity against a mixture of strains of the influenza virus.

Outreach Specialist: Role responsible for reaching out, managing and coordinating patient populations in need of influenza vaccines.

Patient Outreach: The application's or outreach specialist's attempt to inform a patient of their immunization compliance status. These efforts are supported by the application by automating email communications, providing letter templates and patient contact information to the outreach specialists.

Searching Criteria: Methodology in which a user can look for patient information in ICA. This includes: first name, last name, date of birth and medical record number.

Service History: When a patient gets an immunization, it is stored into our database as a service. In ICA each patient will have a list of services that were provided to them. ICA will keep track of: Service Identifier, Immunization Name, Service Date and Outcome among other components.

System Administrator: Role responsible for creating, approving and updating user accounts managing roles, password resets, configuring and running system tasks.

User Account: Access to ICA is user-specific as a security feature to protect patient confidentiality and comply with HIPAA. Every user account created in the application has to be approved by a system administrator.

Work Queue: In the ICA application this is a list of patients that can be sorted in a variety of methods. The general way to format the order of patients is by risk factor. In the work queue an outreach specialist can click on a patient to either outreach to a patient or see access to more detailed information about a specific patient.

References

1. "Use Case." *Wikipedia*, Wikimedia Foundation, 16 Mar. 2020, en.wikipedia.org/wiki/Use_case.
2. "Unified Modeling Language (UML): Sequence Diagrams." *GeeksforGeeks*, 12 Feb. 2018, www.geeksforgeeks.org/unified-modeling-language-uml-sequence-diagrams/.
3. "Software Engineering: Architectural Design." *GeeksforGeeks*, 23 July 2018, www.geeksforgeeks.org/software-engineering-architectural-design/.
4. "What Is Simple Mail Transfer Protocol (SMTP)?" *WhatIsMyIPAddress.com*, whatismyipaddress.com/smtp.
5. "TimSort." *GeeksforGeeks*, 4 Apr. 2020, www.geeksforgeeks.org/timsort/.