Ana Macavei, Emma Yakley, Mahad Ghani, Pooja Sounder Rajan, Melissa Fielman

Term Project: *ChocAn*

Design Document

Table of Contents

1. Introduction

The Design Document for the ChocAn software outlines the design plan within any design constraints that shall then be implemented in the ChocAn software development. This document shall describe the design considerations, the system overview and architecture, and list the detailed design for the reader.

1.1 Purpose and Scope

The purpose of this document is to aid in the design and maintenance of the ChocAn Software.This document also communicates the technical details of the project to stakeholders such as developers, testers, project managers, and customers. The document serves as a blueprint for the implementation of the system and provides a common reference point for all members of the development team.The document shall define the functional and non-functional requirements of the system, including user needs, technical constraints, and performance requirements. The document shall also describe the overall architecture of the system, including the components and their relationships, data flows, and interfaces. Along with those details, the document shall provide detailed descriptions of the system's components, including the data structures, algorithms, and implementation details.

1.2 Target Audience

The target audience of this design document includes all stakeholders involved in the development of ChocAn software.

1.3 Terms and Definitions

- **ChocAn -** Chocoholics Anonymous
- **PHI** - Patient Health Information
- **HIPAA** - Health Insurance Portability and Accountability Act
- **LLL** - Linear Linked List

2. Design Considerations

This section outlines the possible constraints that may affect the functionality of the software. These constraints may also affect the overall design of the software. This section also outlines the methodology that will be used in designing the software.

2.1 Constraints and Dependencies

What constraints, either functional or non-functional were you required to adhere to in designing and implementing your system?

Constraints on the ChocAn software design include the following:

- Timeline
- Terminal Required Fields and Functions
- Information Storage Adhering to HIPAA Compliance
- Software Response Time (5 seconds)

The terminal constraints are hardware related. This terminal shall show the necessary fields when appropriate, and will allow the providers to input and update data. The storage of patient information must adhere to HIPPA for protection of PHI. Outside of this, the only other constraint on file storage shall be that the software can access each type of file required, such as the member profiles, provider profiles, service directory, and weekly reports for billing purposes.

The software shall have a response time of five (5) seconds. This constraint shall be met based on the algorithm chosen for the software.

The final constraint is the timeline, which shall be to deliver the software to the client by Mar 17, 2023.

2.2 Methodology

The team shall be using Agile software methodology for the development of ChocAn software. The Agile method focuses on incremental development which allows room for developmental growth. Project requirements can easily be changed and are handled with little impact on timelines.

The ChocAn software shall utilize object oriented programming for containing member and provider information within a class hierarchy. The provider class shall contain the provider's private information such as address, validity of provider ID (if it's existent initially), ID number, and name and number of service given. The member class shall manage the member's name, ID number, and date, time, ID, and name of service received. A linear linked list with an associated node class shall manage each service given by the provider, and contain access to the provider data, service information, as well as the member data.

3. System Overview

The ChocAn system shall be accessed by two types of users: managers and providers. Managers shall be able to edit information for members and providers, along with the provider directory of services. Managers can generate reports, but reports shall also automatically generate each week for members, providers, managers, and third parties for billing purposes.

Both managers and providers shall have access to terminals. Each provider shall have their own terminal. When the terminal is turned on, the system shall access all directories including the services and provider directory.

When a provider signs in before taking in a member for their appointment, the terminal shall ask for the provider number, which shall then be entered.

When a member checks-in for their appointment, the provider shall ask for the member's ID card, which will be swiped through the terminal's card reader and validated. The terminal shall look through the member's directory file and data to see if any fees are owed, which will lead to a message telling the member how much is owed to ChocAn. If fees are owed by the member, he or she will be unable to access services at any provider until payment is received by ChocAn.

After the member receives their service, the member card will be swiped through the terminal's card reader and the terminal will request the provider for details regarding the service. This information shall be recorded in the files for members and providers.

This services information shall be stored in the ChocAn database and shall be used to generate reports for managers, members, and providers, along with using the profiles of the members and providers. Reports shall be generated automatically each week, but can be requested at any additional time.

4. System Architecture

The ChocAn system architecture is shown in the following figure, Figure 4.0. There are mainly two types of users that will use the ChocAn Software via a terminal, managers and providers. Members are a client of the ChocAn software, but they will not explicitly be accessing the ChocAn software through a terminal.

Data of members shall be encrypted to protect PHI. Services rendered shall be written to files for providers and members. Every week, the ChocAn Data Center shall generate reports for various stakeholders based upon the services rendered.

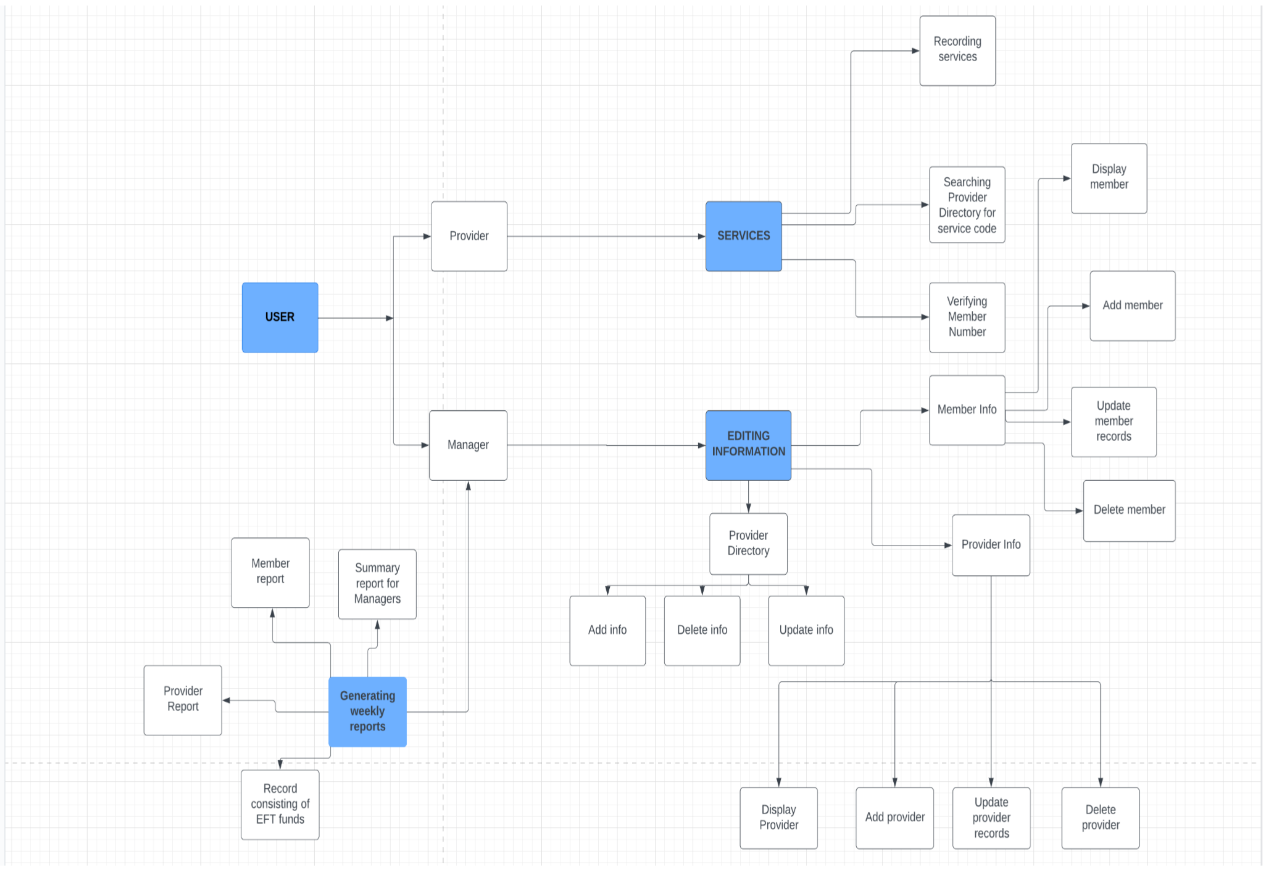

Figure 4.0 High level view of the system architecture

4.1 Data Structure

A linear linked list shall be utilized to gather member appointment information from a provider and write members data and appointment information out to file for the provider login session.

4.2 Information Storage

The information for ChocAn shall be stored in a series of directories and files, as seen in Figure 4.2.0, 4.2.1, 4.2.2 and 4.2.3. Each member and provider utilizing ChocAn shall have a directory to store their member profiles and services utilized or rendered.

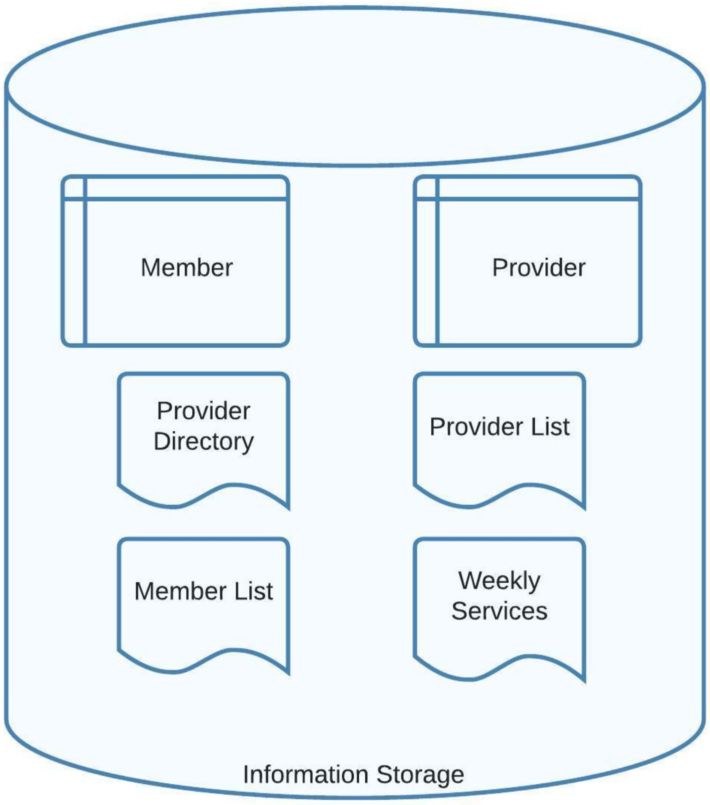

Figure 4.2.0 Information Storage Structure

The member and provider information shall be written to files within their respective directories. The Provider Directory of services shall be a file in the main directory, as well as the weekly services file. Information in the subdirectories of members and providers shall be used to generate their weekly reports.

The following figures shall show the directory structure, as each folder is entered.

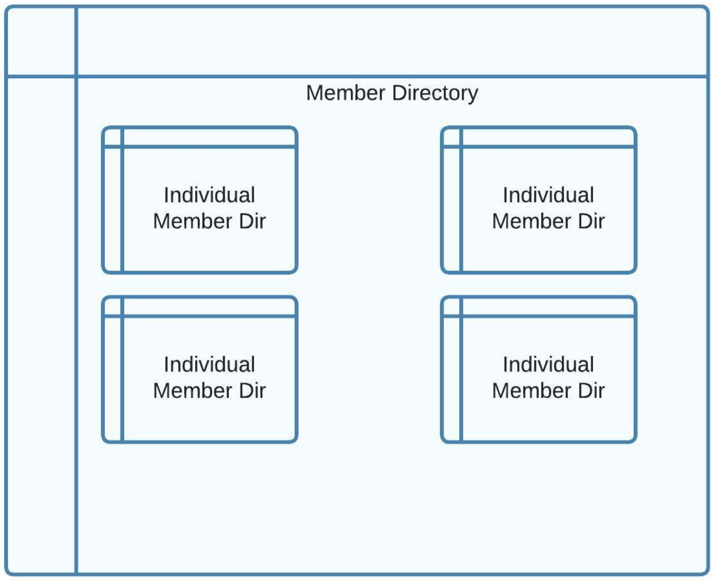

Figure 4.2.1 Member Directory

Figure 4.2.1 shows the member directory structure with individual members each having their own respective directory. The provider Directory shall be the same for the providers.

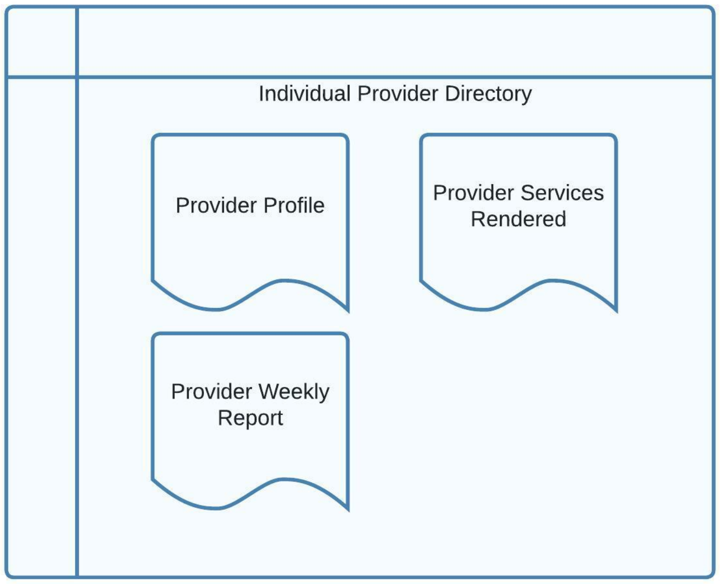

Figure 4.2.2 An Individual Provider Directory

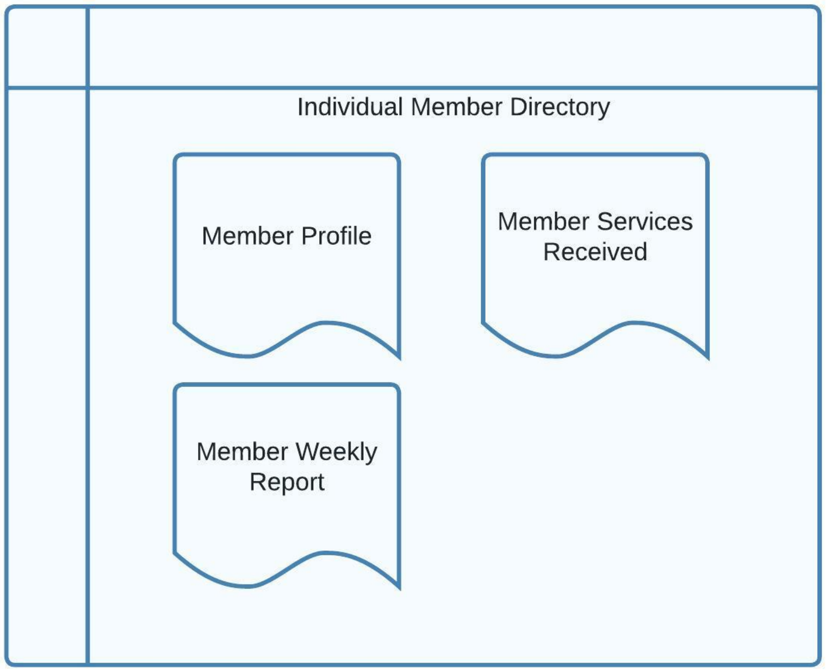

Figure 4.2.3 An Individual Member Directory

4.3 Class Objects

The internal architecture of the software shall make use of object-oriented design principles. The clients of ChocAn (members and providers) shall be represented as an object of a separate class, such as the “Member” or “Provider” as seen in the UML diagrams 4.3.0 and 4.3.1.

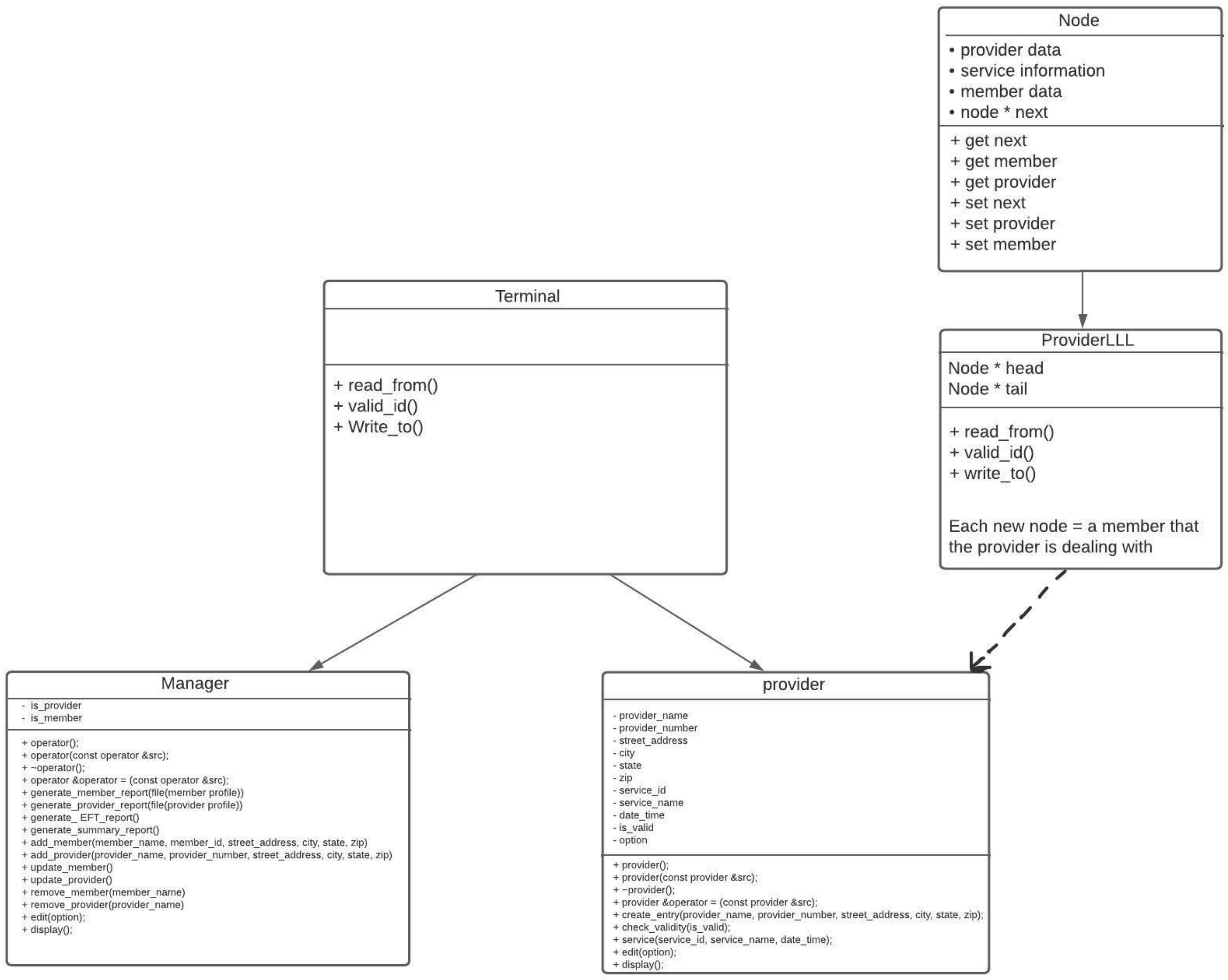
Node
• provider data
• service information
• member data
• node * next
+ get next
+ get member
+ get provider
+ set next
+ set provider
+ set member
Terminal
+ read_from()
+ valid_id()
+ Write_to()
ProviderLLL
Node * head
Node * tail
+ read_from()
+ valid_id()
+ write_to()
Each new node = a member that the provider is dealing with
Manager
- is_provider
- is_member
+ operator();
+ operator(const operator &src);
+ ~operator();
+ operator &operator = (const operator &src);
+ generate_member_report(file(member profile))
+ generate_provider_report(file(provider profile))
+ generate_ EFT_report()
+ generate_summary_report()
+ add_member(member_name, member_id, street_address, city, state, zip)
+ add_provider(provider_name, provider_number, street_address, city, state, zip)
+ update_member()
+ update_provider()
+ remove_member(member_name)
+ remove_provider(provider_name)
+ edit(option);
+ display();
provider
- provider_name
- provider_number
- street_address
- city
- state
- zip
- service_id
- service_name
- date_time
- is_valid
- option
+ provider();
+ provider(const provider &src);
+ ~provider();
+ provider &operator = (const provider &src);
+ create_entry(provider_name, provider_number, street_address, city, state, zip);
+ check_validity(is_valid);
+ service(service_id, service_name, date_time);
+ edit(option);
+ display();

Figure 4.3.0 UML Diagram

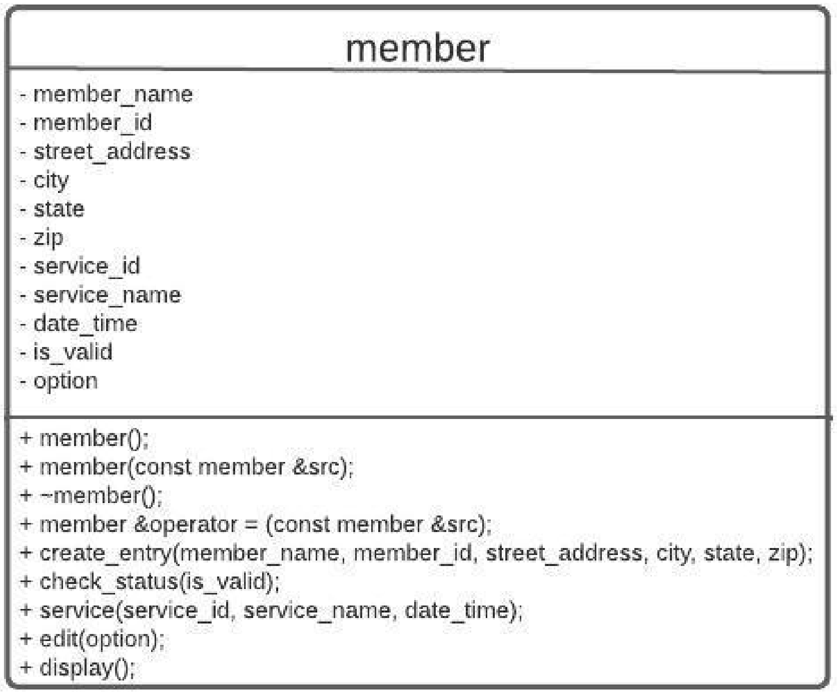
member
- member_name
- member_id
- street_address
- city
- state
- zip
- service_id
- service_name
- date_time
- is_valid
- option
+ member();
+ member(const member &src);
+ ~member();
+ member &operator = (const member &src);
+ create_entry(member_name, member_id, street_address, city, state, zip);
+ check_status(is_valid);
+ service(service_id, service_name, date_time);
+ edit(option);
+ display();

Figure 4.3.1 UML Diagram

These member and provider objects shall store information, such as their name, ID, and address. ChocAn managers shall be able to update these as necessary.

4.4 Client Interface

The client program shall display information based upon the privileges of the user of the software. There are two types of users who shall be able to access different parts of the software: manager and provider. Each of these users have different privileges and abilities.

If a manager is utilizing the software, they shall be able to access the member, provider, and service information to add, delete, edit, and create reports.

The provider interface shall allow the user to search for member and service information, record services rendered, and request reports.

The client program shall access the information stored within the file structure and display the appropriate information based upon the class of the object.

4.5 Reports

Based on the information stored each week in the files, reports shall be generated for managers, providers, members, and third party companies. The file structure mentioned above shall be accessed for each member and provider to create their respective reports.

5. Detailed System Design

The ChocAn software shall be developed using Python 3. As listed in the architecture above, the software shall utilize object oriented programming. The data shall be stored in files and directories for the members, providers, and services. These shall be used to generate reports at any time they are requested, but at least weekly.

5.1 Data Structure: Linear Linked List

While a provider is signed into the terminal, they are able to add services for a member. In order for a provider to not have to sign in each time they are rendering services for multiple clients, a linked list shall be created for the provider login.

As seen in Figure 5.1.0, each node in the Provider Linear Linked List shall hold the provider object, the member object, and service information. This list shall be passed into the function to write out the data to the appropriate files.

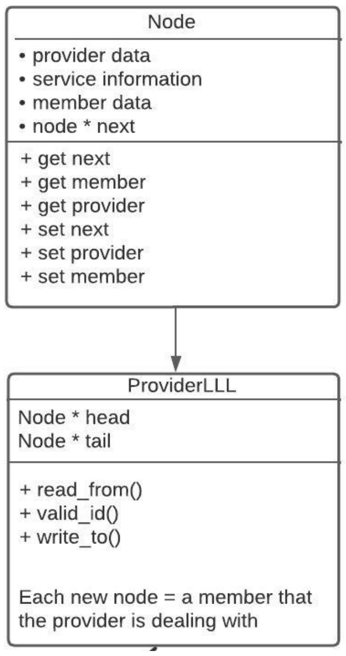

Figure 5.1.0 Classes for the Linear Linked List

5.2 User

There are two main users of the ChocAn software: managers and providers. Each shall have access to a terminal to perform the different functions associated with their administrative privileges.

5.2.1 Provider

A ChocAn provider shall be a doctor or medical provider utilizing the ChocAn software for their client sessions. When a provider sees a member, they shall verify the member's ID number. The member shall state which service they would like to receive, and the provider shall find the corresponding service code through the provider directory. After performing the service, the personal information of the member will be written to the provider's records, and a record of the service received shall be written to the member's records.

The provider number's length shall be compared to how many characters should be in a default provider member, and if not matching, the terminal shall print out a message stating the provider number has invalid number of characters. Otherwise, if the number of characters is valid, the terminal shall access the master provider directory which contains all provider IDs as well as their associated providers' names. If the provider directory file is nonexistent, the terminal shall end its processes and print out a system error message. However if the terminal opens the provider directory file, all of the lines shall be processed until the given provide number matches one of the provider numbers in the directory, at which point the terminal shall print out a success message and confirm with the provider if the name associated with the number is who they want to sign in as. If not, the terminal shall again repeat the previous process and ask for a valid provider number. If the name matches what the provider wants to sign as, the terminal will adjust its current provider name and current provider number variable to the input provider number and its associated name, respectively.

5.2.2 Manager

Managers of ChocAn are responsible for updating and editing the member's and providers' personal information. A manager shall be able to add and delete members, update information (such as address, phone number, et cetera), and display information. A manager can request reports to be run at any time based on the services for the week for a provider, member, or a general summary.

Managers shall have access to the stored files for each member and provider and the class functions associated with the member and provider classes.

5.3 Editing Information (ChocAn Manager Functionality)

A ChocAn Manager shall be able to edit information for both members and providers utilizing the ChocAn software. The ChocAn Manager's role shall give them access to objects for both members and providers and functions to edit the information in these classes. The classes for the Member and Provider can be seen in the UML diagram in Figures 4.3.0 and 4.3.1.

These classes shall hold the name, ID number, street address, city, state, and zip code for each of the clients. Along with this, services rendered can be captured in these objects. Each member and provider object will be able to store the service ID, service date, time, description, and optional comments. The software shall determine if the object is a member or provider to edit, add, or delete members or providers.

5.3.1 Member Information

The function to create a new member shall be a function within the member class. This function will pass the data members into it from an outside source, most likely the client interface. It will then set all of the corresponding data members equal to these arguments.

The function to edit member information will be a function within the member class. The function shall prompt the manager to input which data member they would like to edit (name, address, et cetera) as a string. It will then prompt the manager to input what they would like to change the data to, and set the data equal to said input. It will continue to ask the manager if they would like to edit more data members until the manager answers no.

Pseudocode:

```
edit()
    while manager wants to continue to edit:
        prompt for data member to edit
        prompt for new value of data member
        data member = new value
```

In order to check if a member has paid their bill, and is therefore a valid member, the manager would use the is_valid() function. This function would check if the is_valid data member is true or false, and return it.

The display function for a single member would print to the screen all of the data members of said member, separated by tabs.

5.3.2 Provider Information

The function to create a new provider shall be a function within the provider class. This function will pass the data members into it from an outside source, most likely the client interface. It will then set all of the corresponding data members equal to these arguments.

The function to edit provider information will be a function within the provider class. The function shall prompt the manager to input which data member they would like to edit (name, address, et cetera) as a string. It will then prompt the manager to input what they would like to change the data to, and set the data equal to said input. It will continue to ask the manager if they would like to edit more data members until the manager answers no.

Pseudocode:

```
edit()
    while manager wants to continue to edit:
        prompt for data member to edit
        prompt for new value of data member
        data member = new value
```

The display function for a single provider would print to the screen all of the data members of said member, separated by tabs.

5.3.3 Provider Directory

The provider directory contains all of the services provided by ChocAn to its members, and the corresponding service codes. The manager shall be able to add, delete, and edit these services. The manager can edit either the name of a service or its service code. The function would iterate and search through the directory, and compare if the service name or code matches what the manager entered. If it does, then the function would overwrite this information to the file.

Pseudocode:

```
edit_service(string service_to_change, string new_service)
    while (not at file end)
        if service name == service_to_change
            service name = new_service
        write to file
```

Additionally, if ChocAn begins to provide a new service, it can be added to the directory. This would be done by prompting the manager for the name and service code of the new service, and writing them into the provider directory file.

5.4 Provider Services

The provider shall have access to a ChocAn terminal to perform necessary functions. These include verifying members, recording services rendered, and searching the Provider Directory. In order to access these services, a provider enters their ID number to begin. After this provider number is verified by the valid_id() function in the provider class, the provider shall then see a menu on the terminal to perform any of the available functions that are open to them, as listed below.

5.4.1 Verifying Member Number

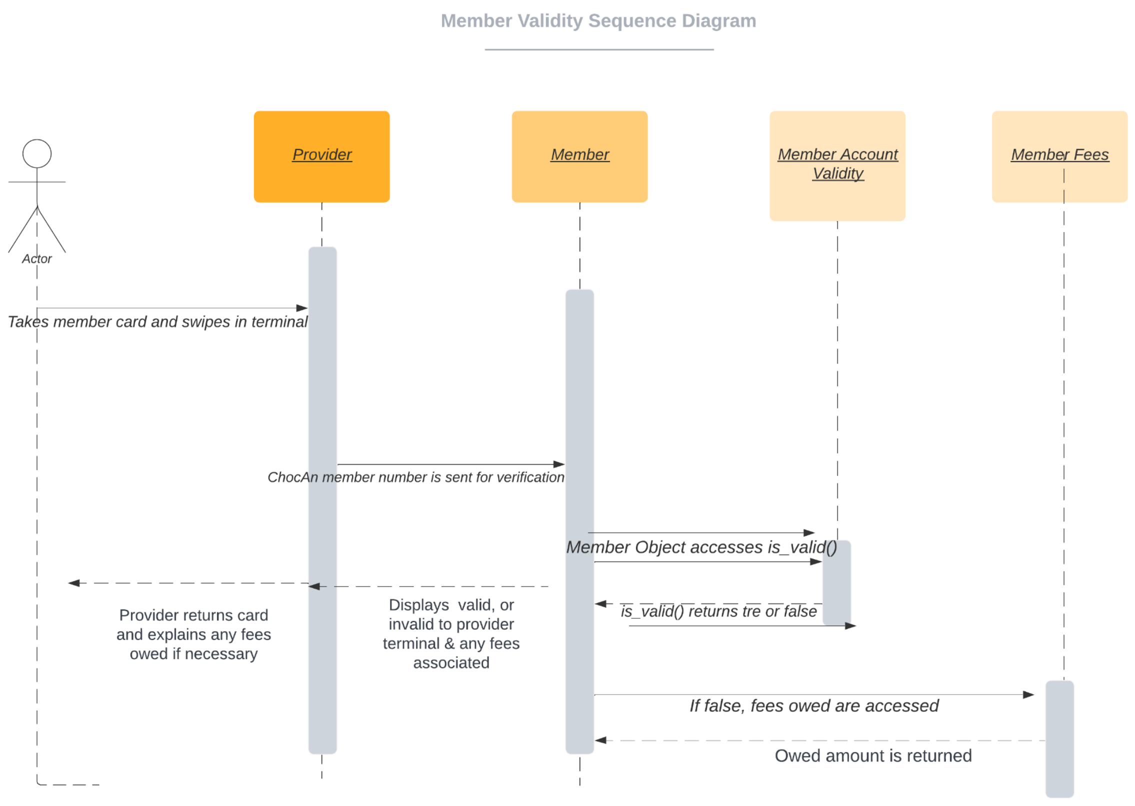
Member Validity Sequence Diagram
Provider
Member
Member Account Validity
Member Fees
Actor
Takes member card and swipes in terminal
ChocAn member number is sent for verification
Member Object accesses is_valid()
is_valid() returns tre or false
Provider returns card and explains any fees owed if necessary
Displays valid, or invalid to provider terminal & any fees associated
If false, fees owed are accessed
Owed amount is returned

Figure 5.4.0 Sequence Diagram for Member ID Validity

When a member checks-in for their appointment, the provider will ask for the member's ID card, which will be swiped through the terminal's card reader. The terminal will scan the ID card's member number and check its validity through the use of functions within the member class. If verified, the terminal will print a validated message. If it is not a valid account, the terminal will print a message prompting the member that their account is invalid. The terminal will then look through the member's directory file and data to see if any fees are owed, which will lead to a print message telling the member how much is owed to ChocAn. If fees are owed by the member, he or she will be unable to receive services until payment is received by ChocAn.

The member ID shall be searched within the ChocAn software. When the ID is located, the member object will call the display function to show the member name.

5.4.2 Service Code Search

The provider shall have a function to display the contents of the Provider Directory. Once that has been displayed, the provider can see the appropriate service number for the appointment the member has with them. The provider shall input this number into the appointment information. The terminal shall display if the provider entered an invalid code. If the code is valid, the terminal shall display the service name.

5.4.3 Entering Appointment Information

The appointment information is entered by the provider after services are rendered. The provider shall enter the information as they are logged into the system.
The provider shall access the member ID calling the member ID validation function. After this is completed and the service is rendered, the provider shall enter this information into the system. As a provider enters information, a linear linked list shall be utilized to keep track of all of the appointments for different members.
The provider linear linked list shall consist of nodes. Each node shall hold a provider object for the provider rendering services. The node shall also hold a member object (for the member receiving services) and the service information. These nodes shall use set and get information and next functions to create the linear linked list.
The list shall then write these appointments to the appropriate directories for the provider and for each member.

5.5 Information Storage

ChocAn shall make use of several directories and files which hold the data points, including a member directory, provider directory, services file, as well as master files of members and providers. The member directory shall have subdirectories in it to house the individual member directories labeled by ID number. These directories shall hold the member profile file and a file holding the services received. These two files shall be used to create the weekly member report. This report shall be housed in the member's individual directory as well.

The provider directory shall have subdirectories in it, in which each provider shall have their own dedicated directory. Inside that, all information related to the provider shall be contained.

The services file shall be a text file containing all the services available for members with their names and service numbers. The master files of members and providers shall contain on each line the ID number of each individual, followed by their name.

5.5.1 Member List

The member identification numbers will be stored in a file. Each line in the text file will first include the member ID number followed by the name of the member, and then the validity of the account. To find the member's name and verify the existence of an ID, the file will be looped through until the line matches the number the provider is trying to find, which if it matches, will return a validated message to the program. This list shall be used in the is_valid() function to show if a member's account is up to date.

5.5.2 Provider List

The provider identification numbers will be stored in a file. Each line in the text file will first include the provider ID number and then the name of the provider. To find the provider's name and verify the existence of an ID, the file will be looped through until the line matches the number the provider is trying to log in as, which if it matches, will return a validated message. This list shall be utilized as a member is trying to log in to ensure that they are a valid provider accessing the system. This shall be called upon by the is_valid() function in the terminal class, see Figure 4.3.0.

5.5.3 Member Profile

The member profile will contain the following information:

- Member Name (25 characters)
- ID Number (9 digits)
- Member Street Address (25 characters)
- Member City (14 characters)
- Member State (2 letters)
- Member Zip (5 digits)

5.5.4 Provider Profile

The provider profile will contain the following:

- Provider Name (25 characters)
- ID Number (9 digits)
- Provider Street Address (25 characters)
- Provider City (14 characters)
- Provider State (2 letters)
- Provider Zip (5 digits)

5.5.5 Member Services File

This file shall keep a record of each service received for the week. This file shall be written at the time a member signs in for a service and the provider validates the service to be performed. This service information shall be written to the weekly service file and include the following:

- Provider Number (9 digits)
- Member Number (9 digits)
- Date of Service (MM-DD-YYYY)
- Date and time data were received by the computer (MM-DD-YYYY HH: MM: SS)

- Service Code (6 digits)
- Additional comments

This information shall be separated by a delimiter and each line shall be ended with a new line character.

5.5.6 Provider Services File

This file shall keep a record of each service rendered for the week. This file shall be written at the time a member signs in for a service and the provider validates the service to be performed. This service information shall be written to the weekly service file and include the following:

- Provider Number (9 digits)
- Member Number (9 digits)
- Date of Service (MM-DD-YYYY)
- Date and time data were received by the computer (MM-DD-YYYY HH: MM: SS)
- Service Code (6 digits)
- Fee for Service (up to $999.99)
- Additional comments

This information shall be separated by a delimiter and each line shall be ended with a new line character.

5.5.7 Provider Directory (Services)

The Provider Directory shall store the list of services available that providers could provide to members.
This file shall hold the following:

- The nine (9) digit service ID code
- The name of the service (up to 20 characters)
- The fee associated with the service in the format of $XX,XXX.XX.

The ID, name, and fee shall be separated by a delimiter and each line shall be ended with a new line character.

<u>Pseudocode:</u>

```
While (line != null)
        if our id == 9 digit id:
                note service name and id
                validate
return
```

This file shall be stored in the main directory of the ChocAn information storage.

5.6 Reports

Each week, reports shall be run for the clients of ChocAn. There shall be a weekly member report for each paying member and a weekly provider report for each participating provider. Along with these, a summary report shall be generated for the managers.

The software shall aggregate the weekly services for each provider, having a summary section at the bottom of each report. These summaries shall be used to create the larger overall summary report for the managers.

These reports shall be run by utilizing the read functions to access the individual members and providers' service files. The write function shall take the information found and write to the reports as listed below.

5.6.1 Weekly Member Report

Each member report must be written to its own file; the name of the file should begin with the member name followed by the date of the report. The service file in each member's directory shall be parsed by week based on the dates of service. This information will then be output to the weekly report.

The weekly report shall take the member profile information and create the header of the report. Following this shall be the list of the services utilized during that week.

5.6.2 Weekly Provider Report

The provider reports should be handled in the same way.
The report contains the same information as that entered on the provider's form, in the order that the data were received by the computer. At the end of the report is a summary including the number of consultations with members and the total fee for that week.

The weekly report shall take the provider profile information and create the header of the report. Following this shall be the list of the services rendered during that week. At the bottom of the file there will be the summary section that shows the total number of patients seen and the cost to be billed.

5.6.3 Record of EFT Funds

As for the EFT data, all that is required is that a file be set up containing the provider name, provider number, and the amount to be transferred. This information shall be parsed from the summary report.

5.6.4 Summary Report

A summary report is given to the manager for accounts payable. The report lists every provider to be paid that week, the number of consultations each had, and their total fee for that week. This information is