DESIGN DOCUMENT:
Chocoholics Anonymous

CIS 375

Last Updated: 8/06/18

Version 1.0

Team "2"007 Britney Spears:

Joshua Attard, Veeram Hirekhan, Hassan Mehdi,

Maram Mohammed, Allison Ramasami, Srinivas Simhan

1. Introduction

1.1. Introduction

- This is the Design Document (DD) for Team “2”007 Britney Spears.
- This project is being done for Software Engineering I (CIS 375), and is being undertaken by Joshua Attard, Veeram Hirekhan, Hassan Mehdi, Maram Mohammed, Allison Ramasami, and Srinivas Simhan.
- The project involves automating Chocoholics Anonymous data processing and storage to speed up their current manual processes.

1.2. Problem Statement

- We are creating a database to store member, transaction, and provider information, so that the database can be automatically managed and updated. The information can be accessed at any time by the ChocAn Data Center, the Provider Directory, the member/provider terminals at the ChocAn facility, and Acme Accounting Services.

1.3. Goals and Objectives

Overall Goal:

- Automate the various processes at ChocAn that are currently being done manually.
- Centralize all member and provider information in a database, located at a ChocAn data center. The data will be accessible remotely through ChocAn terminals available to ChocAn employees as well as all providers.
- Database will then be used to automatically generate reports for members, providers, and managers every week.

Objectives:

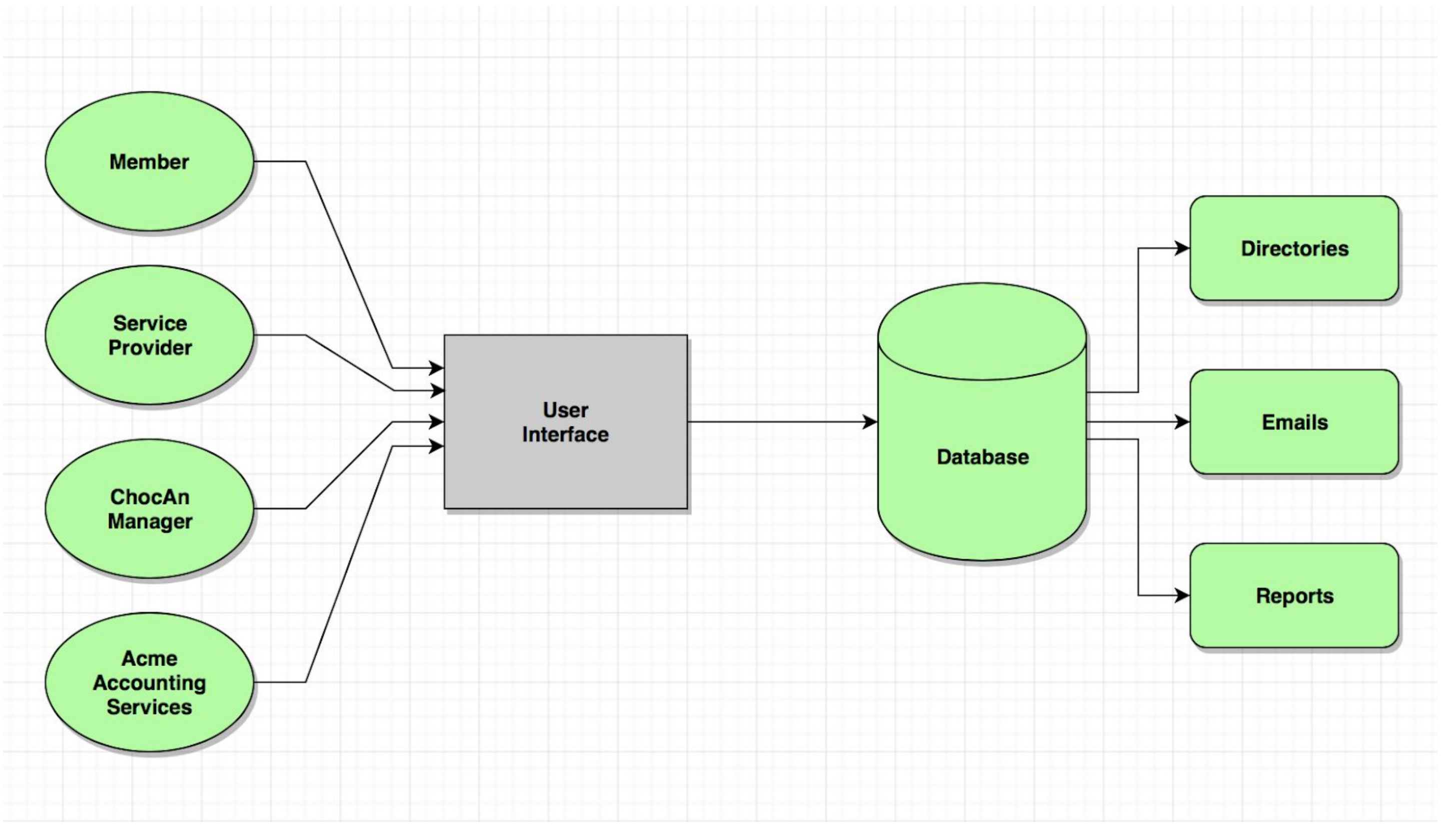
- Create a database to store member and provider data
- Create software to process membership payments and determine membership status
- Create front-end for providers to enter service information into and obtain provider directory
- Create front-end for operators to modify member/provider data
- Create software to generate reports for members, providers, and managers
- Design ChocAn terminals
- Create software that allows terminal to communicate with data center

1.4. Purpose of Design Document

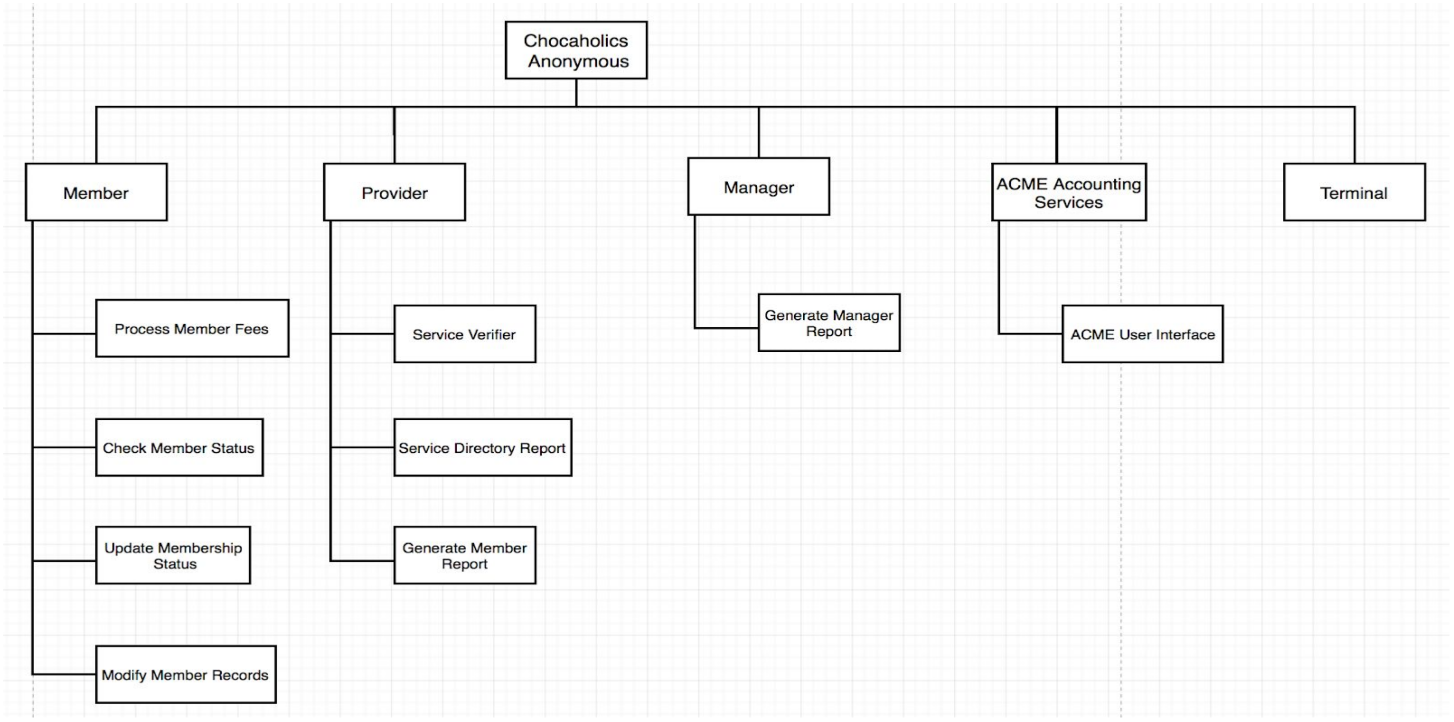
- The purpose of the Design Document is to describe all data, architectural, interface, and component-level design for the software. A design specification provides explicit information about the requirements for a product and how the product is to be put together.

2. System Design

2.1. Architecture Diagram



3. Decomposition Diagram



4. Detailed Design

4.1. getPaymentsForMember

Class	Interface	Functionality	Side effects
Process Member Fees	Inputs: memberinfo, currMember Outputs: the payment is now registered	Retrieves the payment/transac-tion info between a ChocAn member and ACME	

4.2. checkMemberStatus

Class	Interface	Functionality	Side effects
Check Member Status	Inputs: memberNumber Outputs: currStatus	During member log in, returns the current status of the member according to the ChocAn database	

4.3. getMemberList

Class	Interface	Functionality	Side effects
ACME User Interface	Inputs: currScreen, membersChecke d Outputs: list of ChocAn members	Searches for the list of ChocAn members and displays it in the ACME user interface	

4.4. markNewMemberStatus

Class	Interface	Functionality	Side effects
ACME User Interface	Inputs: currScreen, membersChecked, newStatus, memberInfo Outputs: New ChocAn members are marked accordingly	Searches for new members in ACME and marks them invalid, valid, or suspended based on their payment info	

4.5. markMemberChecked

Class	Interface	Functionality	Side effects
ACME User Interface	Inputs: currScreen, membersChecked Outputs: marks the member as checked	After the transaction info is checked for an unchecked member, the member becomes checked in the ACME User Interface	Searches for unchecked member

4.6. updateMemberStatuses

Class	Interface	Functionality	Side effects
ACME User Interface	Inputs: currScreen, memberChecklist Outputs: Confirmation message saying "Member status updated"	Through the ACME user interface, the process of updating the member statuses in the ChocAn database is started	Starts updateMemberStatus function

4.7. updateMemberStatus

Class	Interface	Functionality	Side effects
Update Membership Status	Inputs: status Outputs: status	Compares the membership status in the ChocAn database to the one in ACME and updates accordingly	

4.8. getMemberInfo

Class	Interface	Functionality	Side effects
Modify Member Records	Inputs: currMemberInfo Outputs: Member information is retrieved such as name, number, address, email and status	Retrieves the previous information of ChocAn member whose information needs to be updated	

4.9. changeMemberInfo

Class	Interface	Functionality	Side effects
Modify Member Records	Inputs: currMemberInfo, newMemberInfo Outputs: stores the newMemberInfo as currMemberInfo	The ChocAn employee changes the member information and the new information is stored in the ChocAn database as the member's current information	Member information in the ChocAn database is changed.

4.10. addMember

Class	Interface	Functionality	Side effects
Modify Member Records	Inputs: string memberName, int memberNumber, string street, string city, string state, int zipCode, string email Outputs: new member created	Adds a new member to the system	A member will be created in the database with the input as its currMemberInfo The member starts off with an 'Good Standing' status

4.11. deleteMember

Class	Interface	Functionality	Side effects
Modify Member Records	Inputs: int memberNumber Outputs: member deleted	Deletes a member from the system	A member will be removed from the database

4.12. memberLogin

Class	Interface	Functionality	Side effects
Terminal	Inputs: int memberNumber Outputs: member is now logged into terminal	Logs member into terminal using their member card	

4.13. providerLogin

Class	Interface	Functionality	Side effects
Terminal	Inputs: int memberNumber, Int providerNumber Outputs: provider is now logged into terminal	Logs provider into terminal using a member card and a providerNumber	

4.14. addProvidedService

Class	Interface	Functionality	Side effects
Service Verifier	Inputs: int memberNumber, int ProviderNumber, int serviceCode, time_t dateProvided, time_t dateReceived, string comment Outputs: new service created in Service class	A service is added to the provided service class	A service is added to the provided service class in the database

4.15. requestProviderDirectory

Class	Interface	Functionality	Side effects
Terminal	Inputs: string email Outputs: creates excel file of all services and emails it to the logged in users email	Sends an email to the providers email	Provider receives provider directory

4.16. modifyRecords

Class	Interface	Functionality	Side effects
Terminal	Inputs: bool isMember, int Member/Provider Number, String Name, string street, string city, string state, int zipCode, string email Outputs:	If isMember = true, takes memberNumber and changes information of that members records. If isMember = false, takes providerNumber and changes information of that providers records	A member or provider’s data is changed in the database

4.17. sendReports

Class	Interface	Functionality	Side effects
Terminal	Input: Member, provider, and manager reports (excel files) Output: Emails with report attachments	For each member, provider, and manager that needs a report, generate the report and send it to their email address.	Emails are sent to various members, providers, and managers.

4.18. createProviderDirectory

Class	Interface	Functionality	Side effects
Service Directory Report	Inputs: int serviceCode, string name, float fee Outputs: Provider directory email	Creates a service report based on all the ChocAn services compiled that week	Provider directory email is sent to provider.

4.19. generateMemberReport

Class	Interface	Functionality	Side effects
Generate Member Report	Inputs: int memberNumber, stirng name, string street, string city, string state, int zipCode, and list of provided services (2D array of width 6) Outputs: Formatted member report	Produce a properly formatted report for a given member by looking up their information and the services they received in the database.	

4.20. sendMemberReport

Class	Interface	Functionality	Side effects
Generate Member Report	Inputs: Member report (excel file), string email Outputs: Email to member containing member report as attachment.	Given a member name, send a report to that member's email address via email attachment.	Member report email is sent to member.

4.21. generateManagerReport

Class	Interface	Functionality	Side effects
Generate Manager Report	Inputs: All services billed to ChocAn for this week (2D array of width 6) Output: Formatted manager report	Produces a manager report by taking all services and for each provider, tallying up the total number of services they provided and the total fee that must be paid to them, and displaying that along with totals for number of providers, overall fee, overall number of services.	

4.22. sendManagerReport

Class	Interface	Functionality	Side effects
Generate Manager Report	Inputs: Manager report (excel file) and string email Outputs: Email to manager containing manager report as attachment.	Given a manager name, send a report to that manager's email address via email attachment.	Manager report email is sent to manager.

4.23. generateProviderReport

Class	Interface	Functionality	Side effects
Generate Provider Report	Inputs: currProvider, providerServices Outputs: formatted provider report	Produces a provider report by taking the input and generating it in a excel file formatted properly	

4.24. sendProviderReport

Class	Interface	Functionality	Side effects
Generate Provider Report	Inputs: Provider report Outputs: Provider Report Email	Sends the provider's report via email in a excel file	Sending the email.

4.25. verifyProvidedService

Class	Interface	Functionality	Side effects
Generate Provider Report	Inputs: Service code Outputs: verifiedService	Check's to make sure the service information is correct, and return a status saying the service is verified	

4.26. addProvidedService

Class	Interface	Functionality	Side effects
Service Verifier	Inputs: currService Outputs: Updated with new provided service	Adding a service to the provided services database.	Updating the database with the updated information.

4.27. getProviderInfo

Class	Interface	Functionality	Side effects
Modify Provider Records	Inputs: Provider Number Outputs: Provider Info	Being able to find a provider based on the search criteria and that the provider is there.	

4.28. changeProviderInfo

Class	Interface	Functionality	Side effects
Modify Provider Records	Inputs: currProviderInfo Outputs: Updates database and saves new info on the UI.	Select one provider info and being able to edit the provider name, provider number, service code etc.	Have to update the database with the new information.

4.29. addProvider

Class	Interface	Functionality	Side effects
Modify Provider Records	Inputs: newProviderInfo Outputs: new provider	Adds a new provider to the provider database.	Adds a new provider to the database.

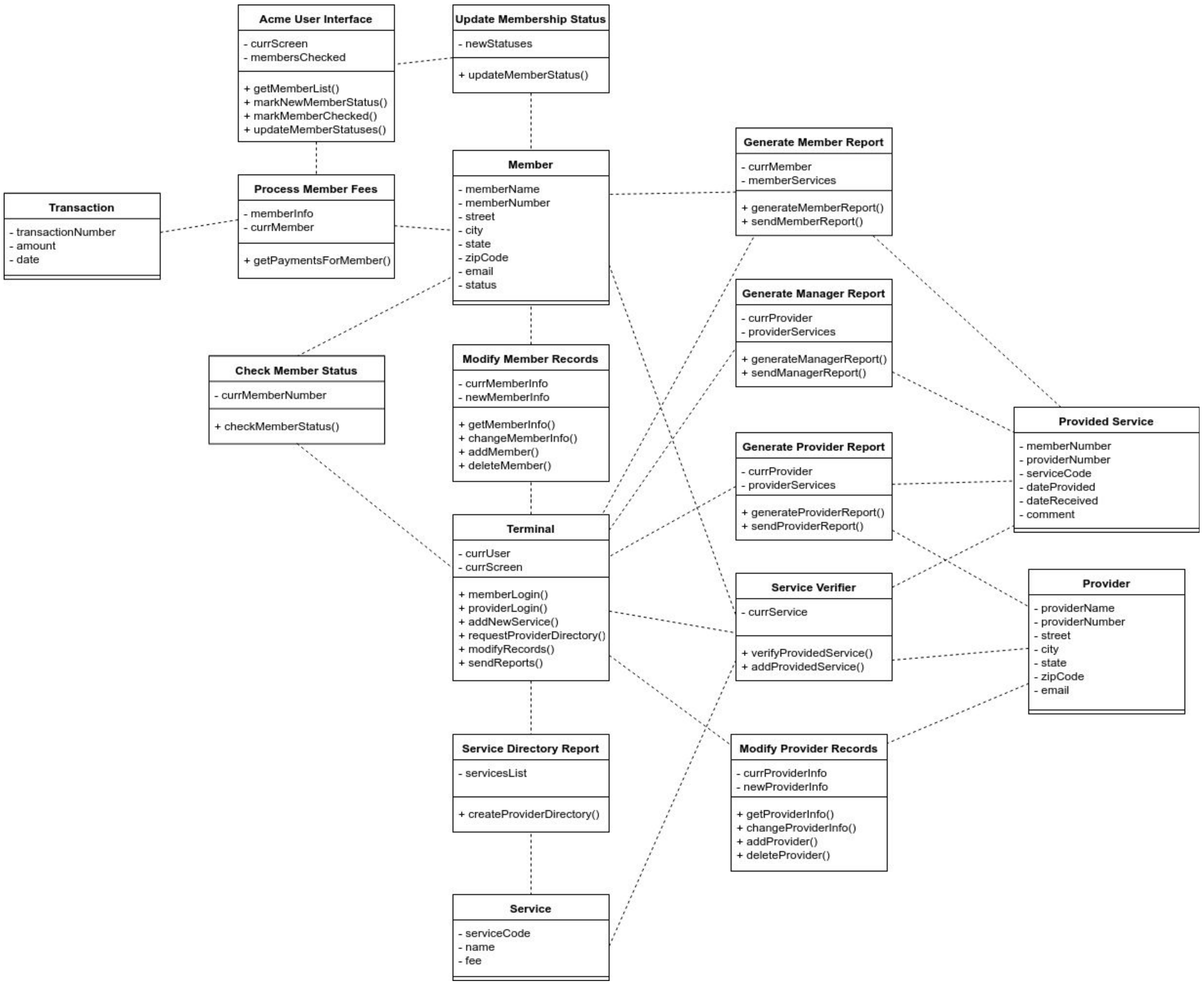
4.30. deleteProvider

Class	Interface	Functionality	Side effects
Modify Provider Records	Inputs: currProviderInfo Outputs: no provider	Ability to delete a existing provider on the provider database.	Update the provider database with one less provider.

5. Data Design

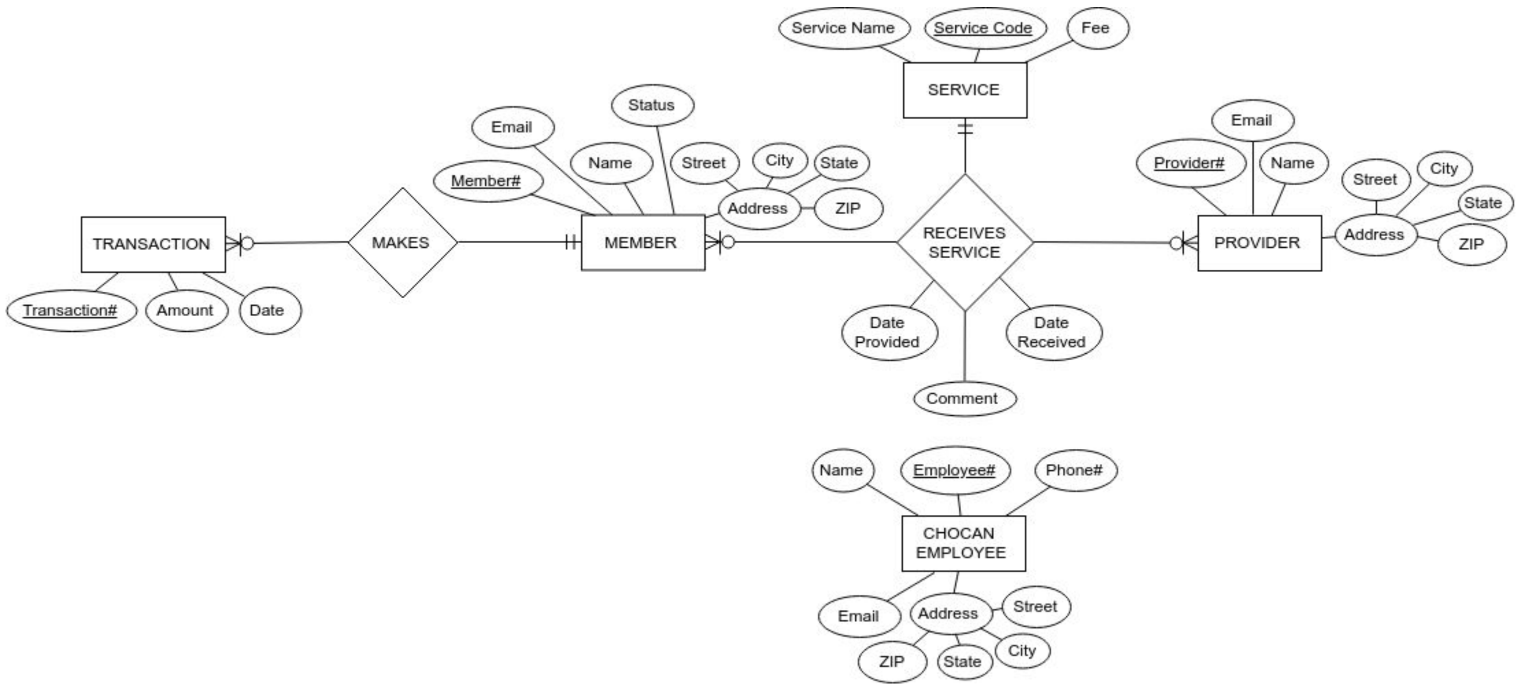
5.1. Class Diagram

A full class diagram is provided here.



5.2. ER Diagram

The build ERD is provided here.



5.3. Data Flow Diagrams

The data flow diagrams are provided here.

Context Diagram

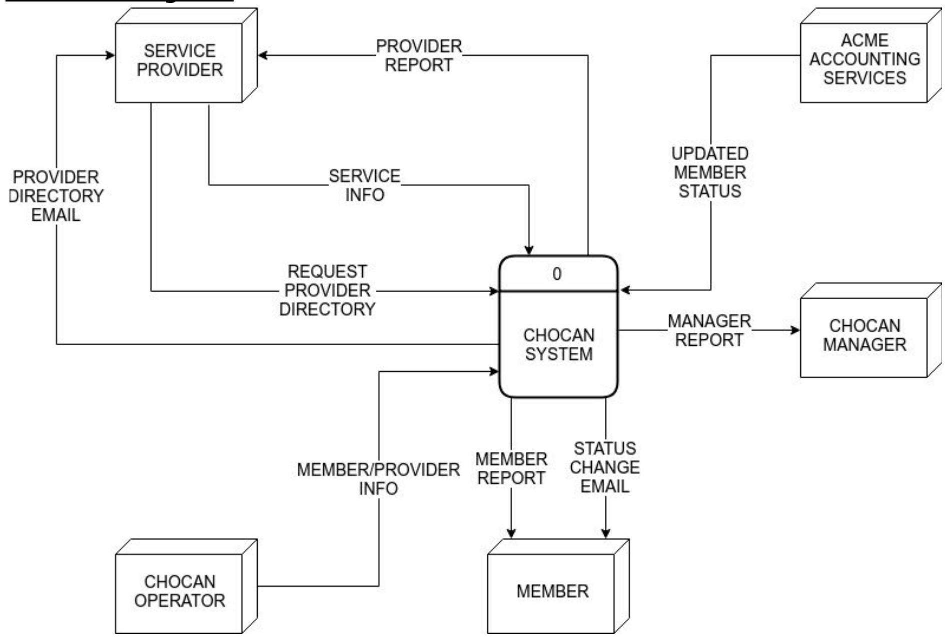


Diagram 0 DFD

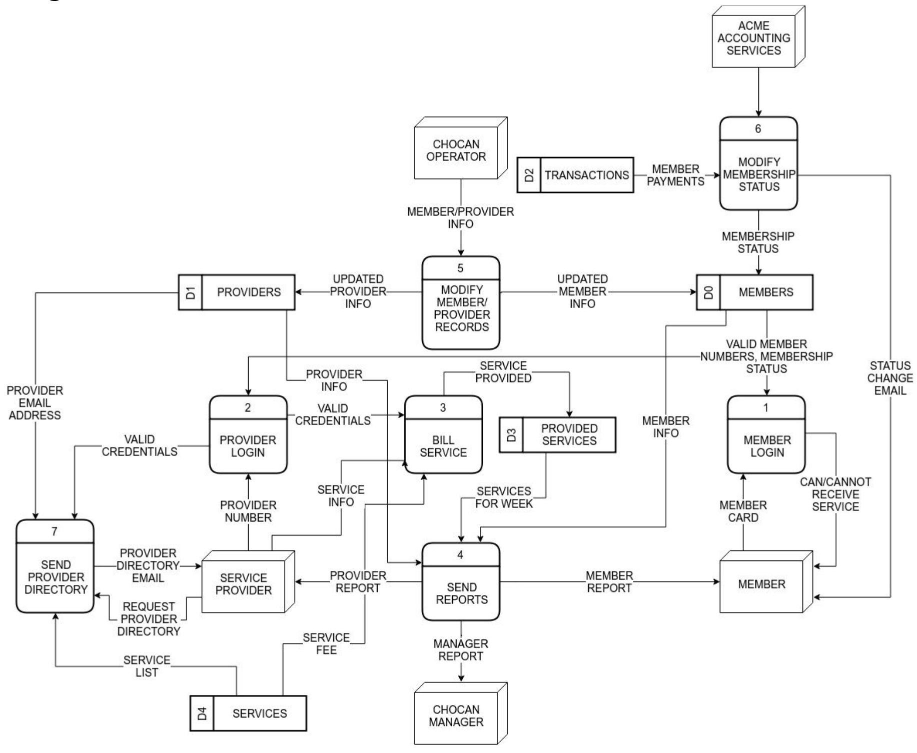


Diagram 1 DFD

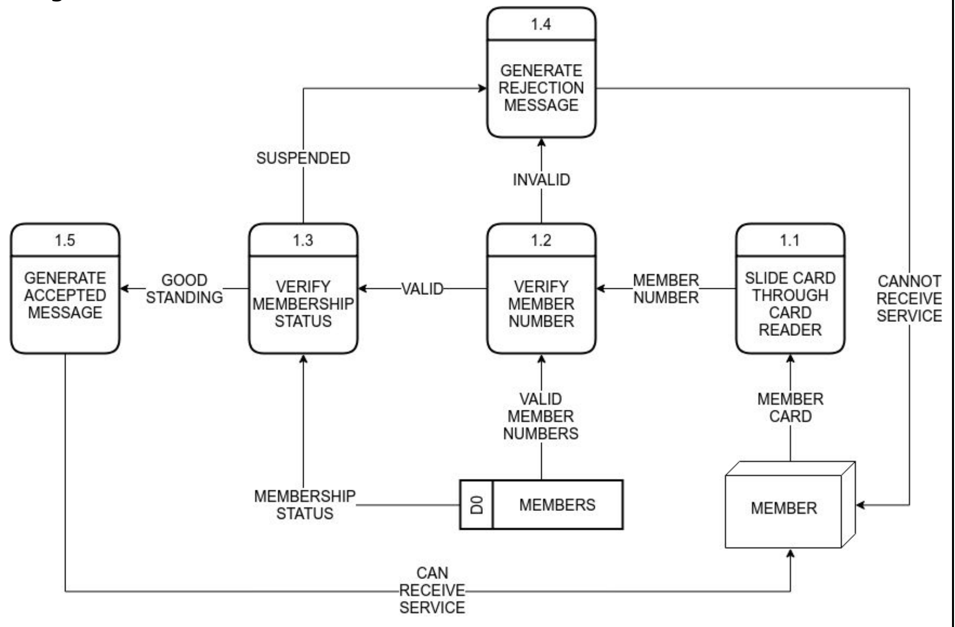


Diagram 2 DFD

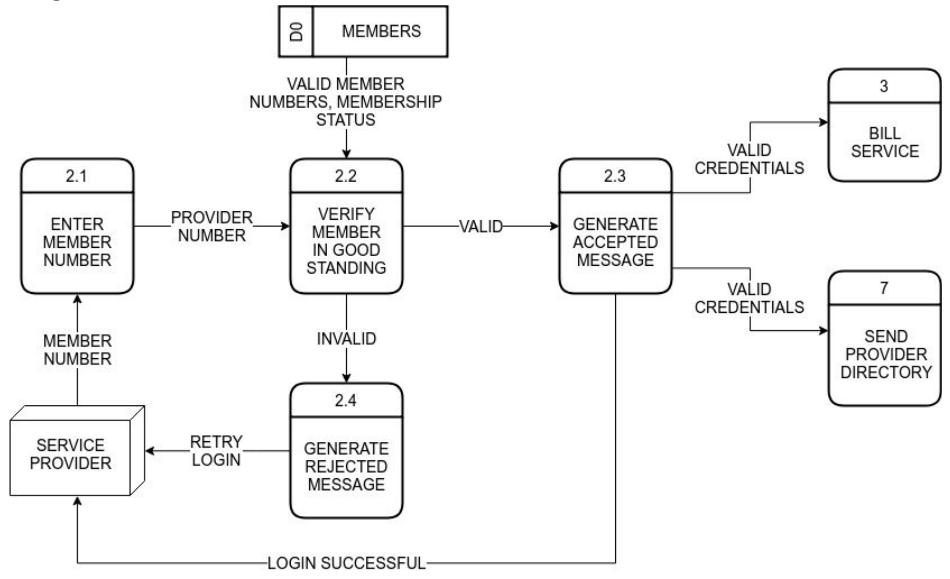


Diagram 3 DFD

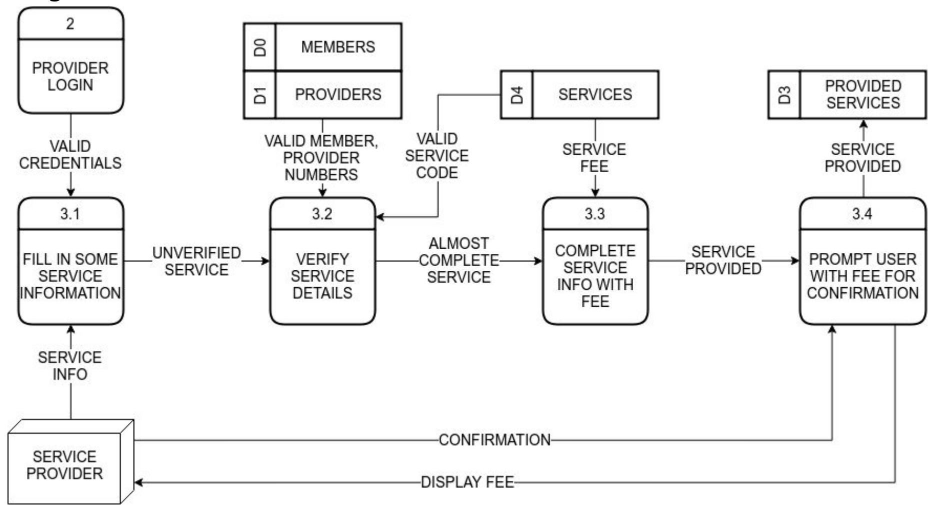


Diagram 4 DFD

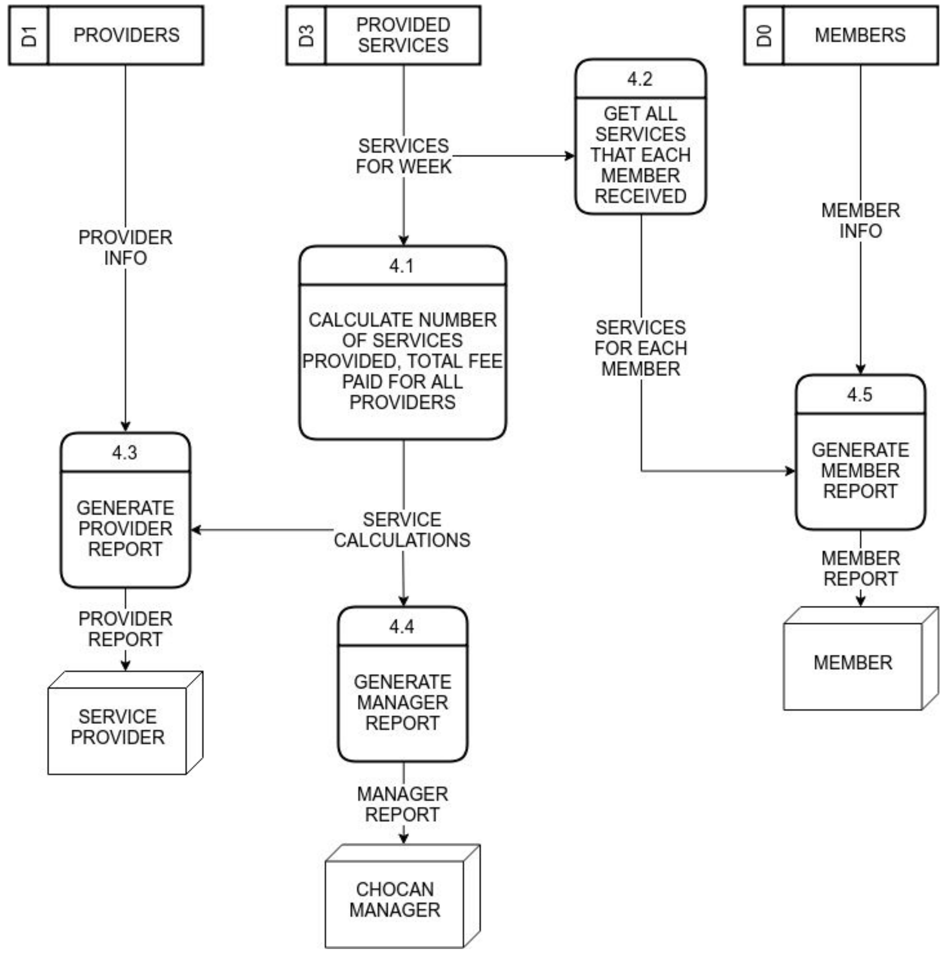


Diagram 5 DFD

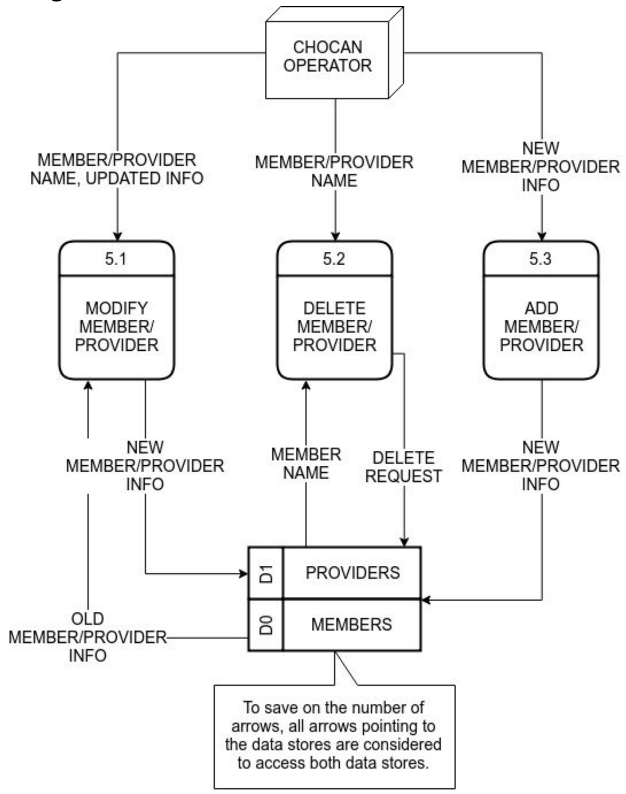


Diagram 6 DFD

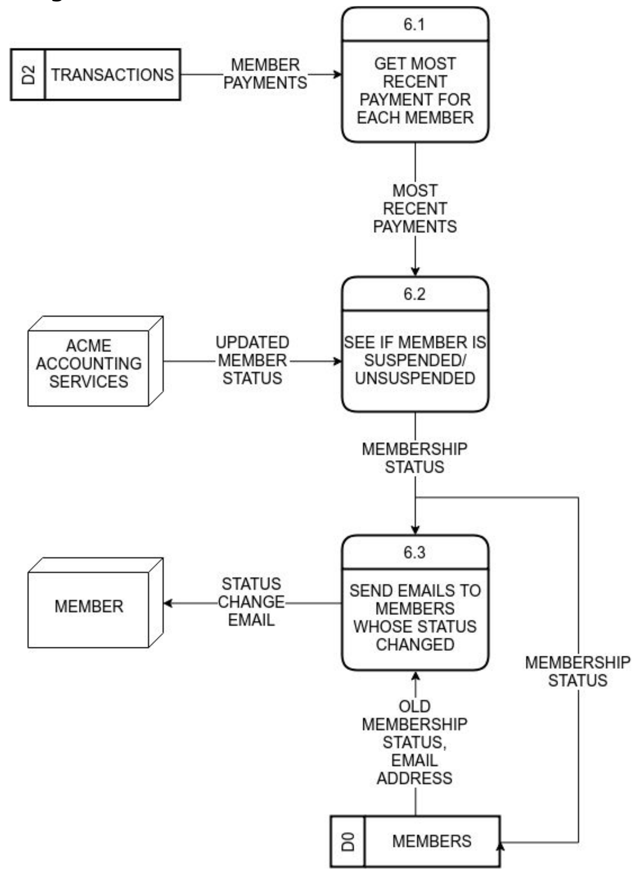
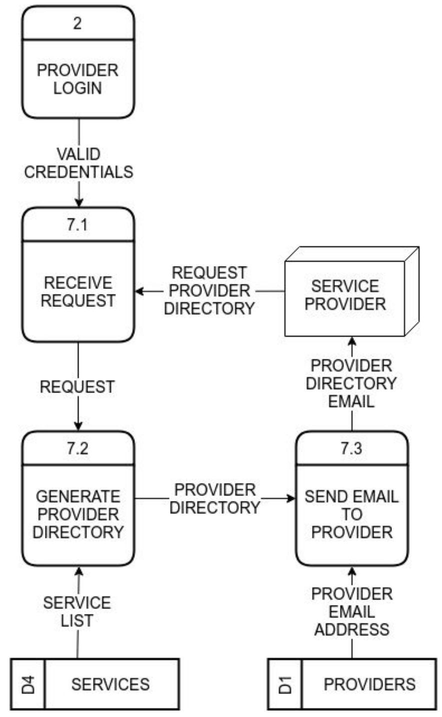
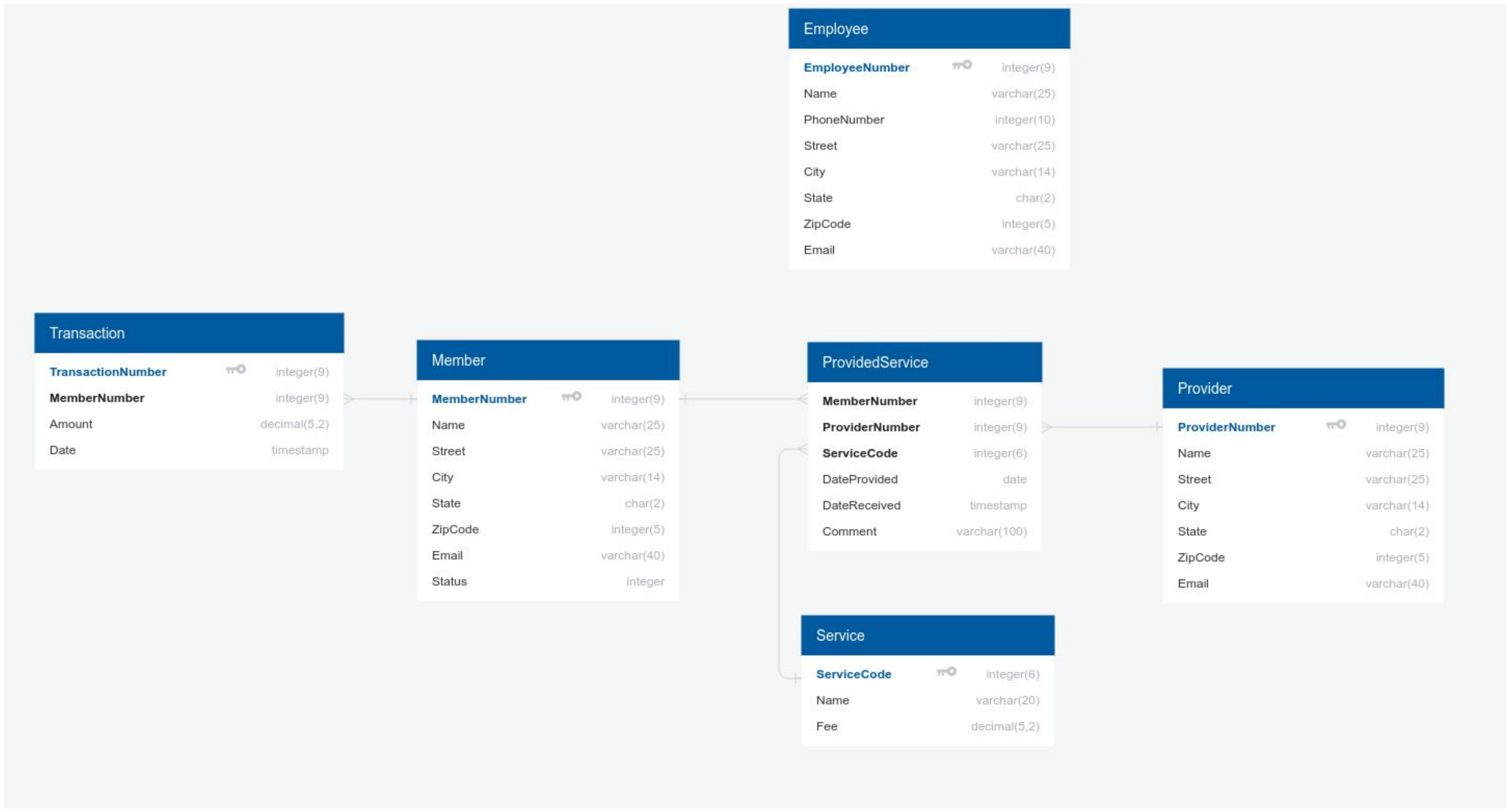


Diagram 7 DFD



5.4. Database Schema

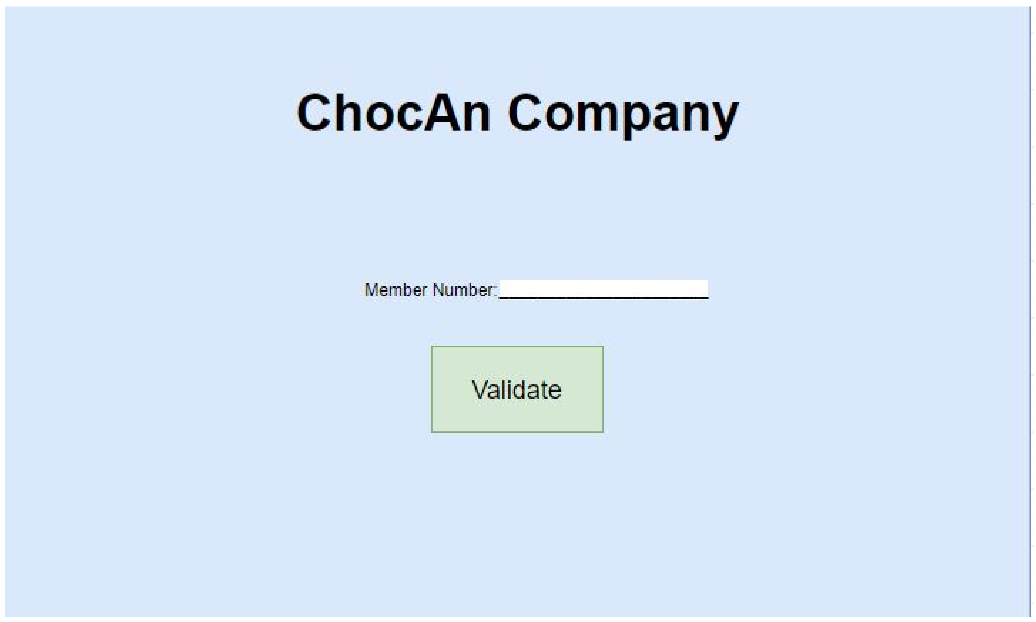
A database schema is provided here.

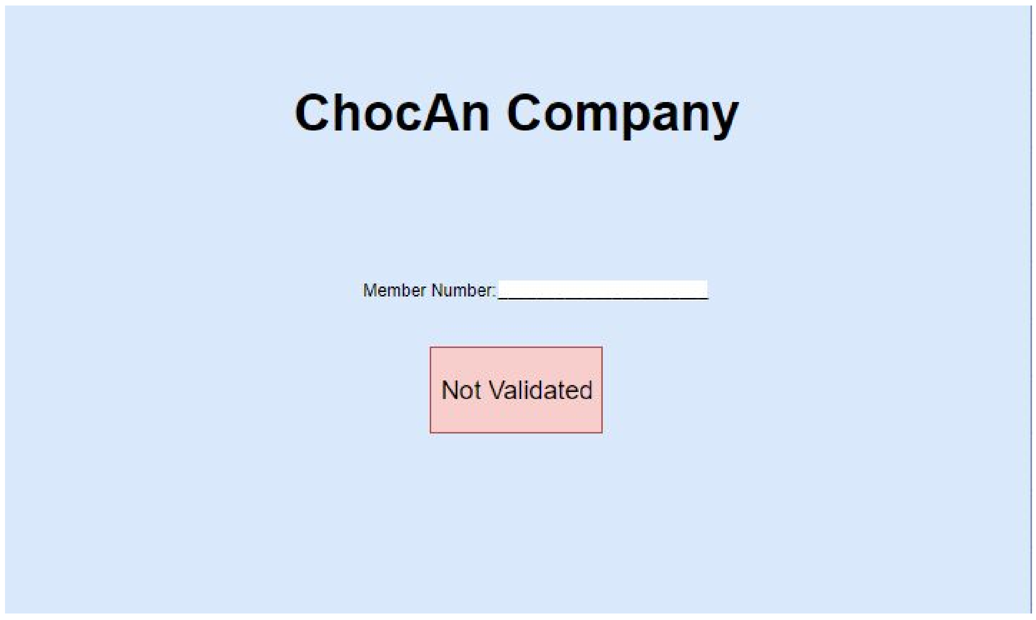


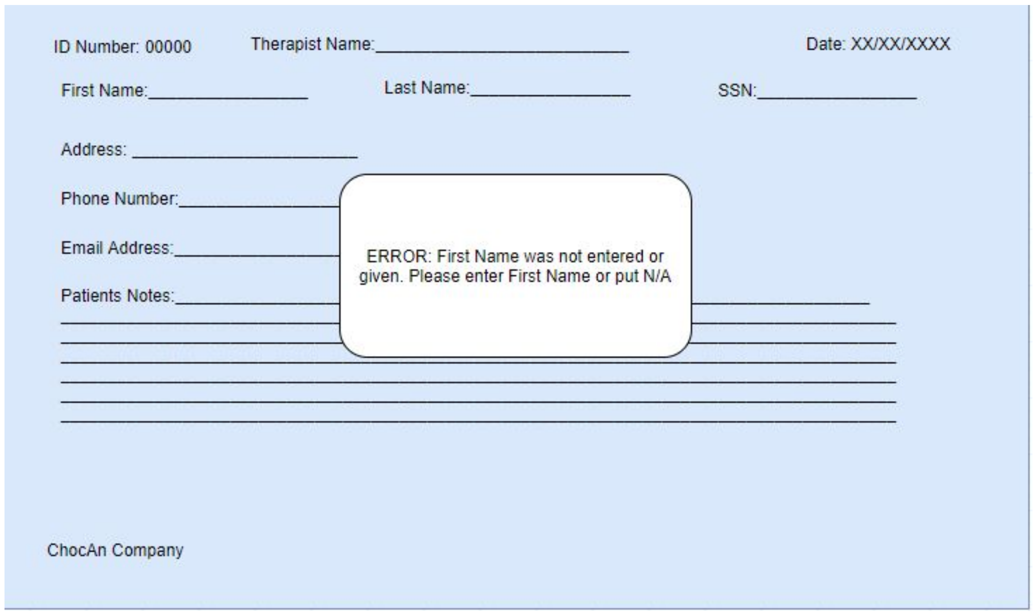
6. User Interface Design

6.1. Software Layouts

User interface layouts for the software are given here.







ID Number: 00000 Therapist Name:_________________________ Date: XX/XX/XXXX

First Name:________________ Last Name:________________ SSN:________________

Address: _______________________

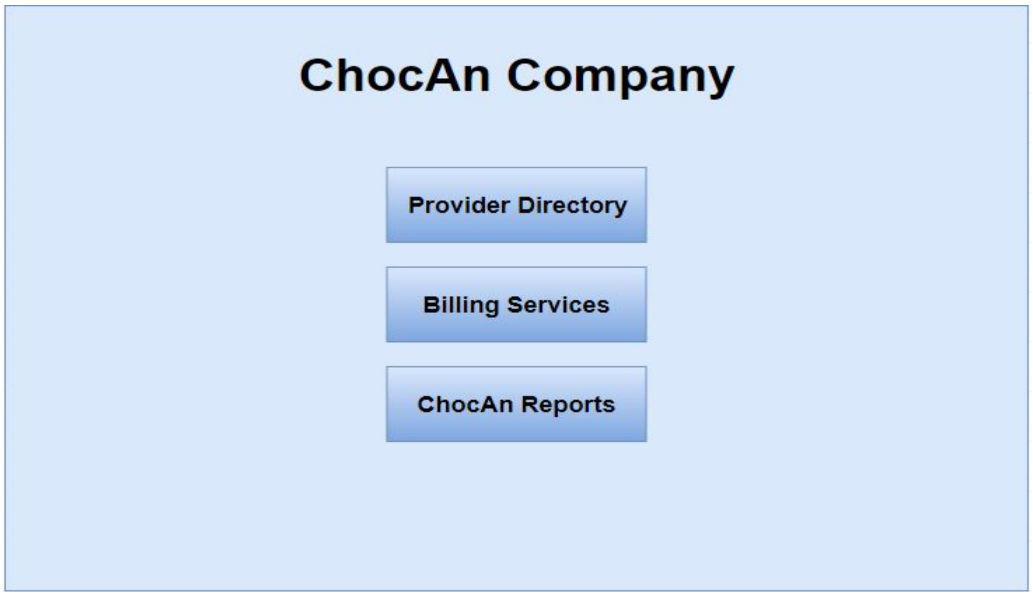
Phone Number:________________

Email Address:________________

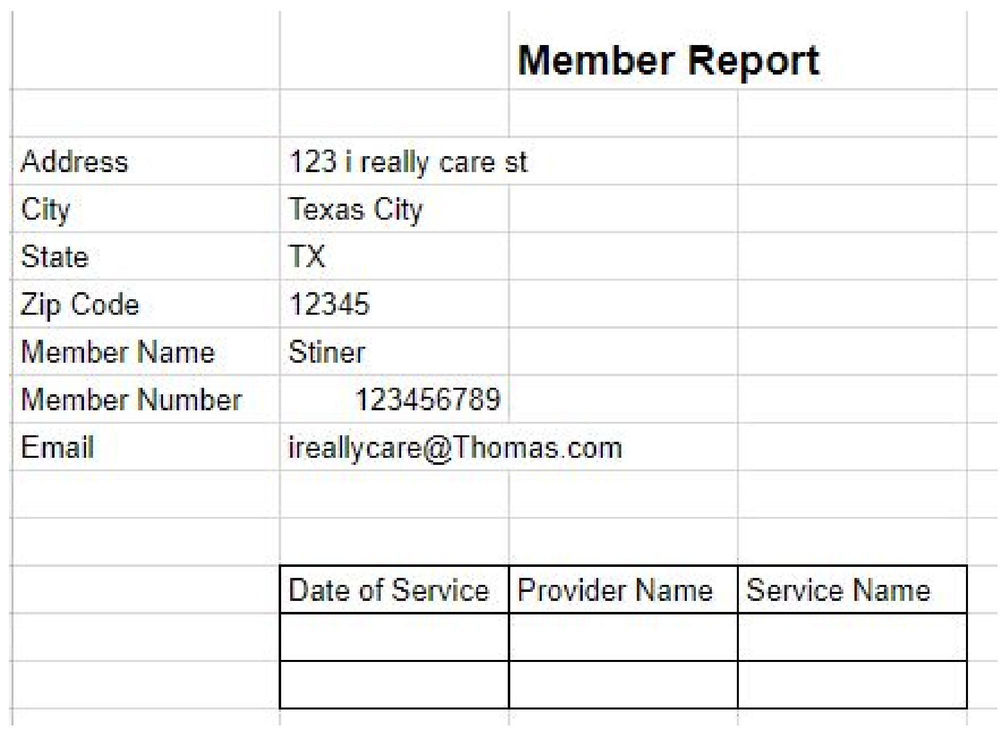
Patients Notes:________________

ERROR: First Name was not entered or given. Please enter First Name or put N/A

ChocAn Company



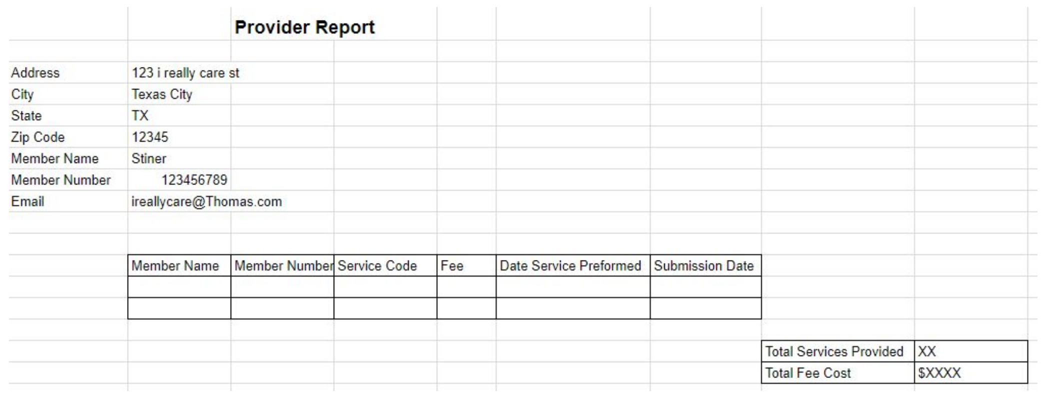
6.2. Report Layouts



Member Report

Address	123 i really care st
City	Texas City
State	TX
Zip Code	12345
Member Name	Stiner
Member Number	123456789
Email	ireallycare@Thomas.com

Date of Service	Provider Name	Service Name

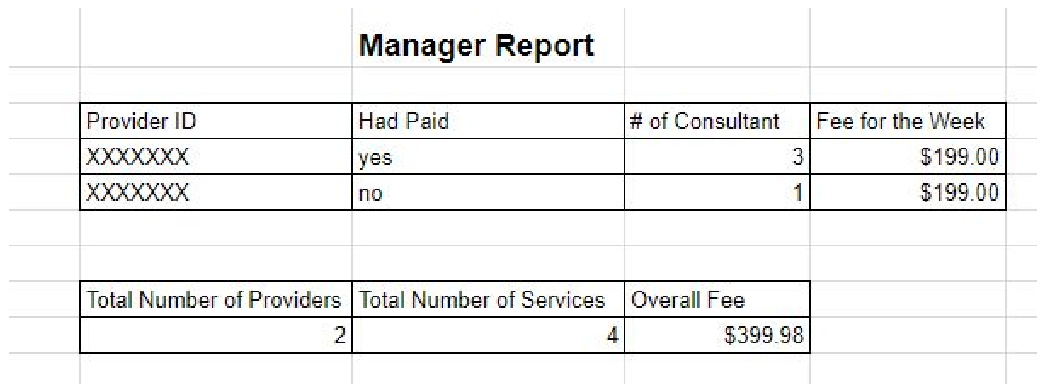


Provider Report

Address	123 i really care st
City	Texas City
State	TX
Zip Code	12345
Member Name	Stiner
Member Number	123456789
Email	ireallycare@Thomas.com

Member Name	Member Number	Service Code	Fee	Date Service Preformed	Submission Date

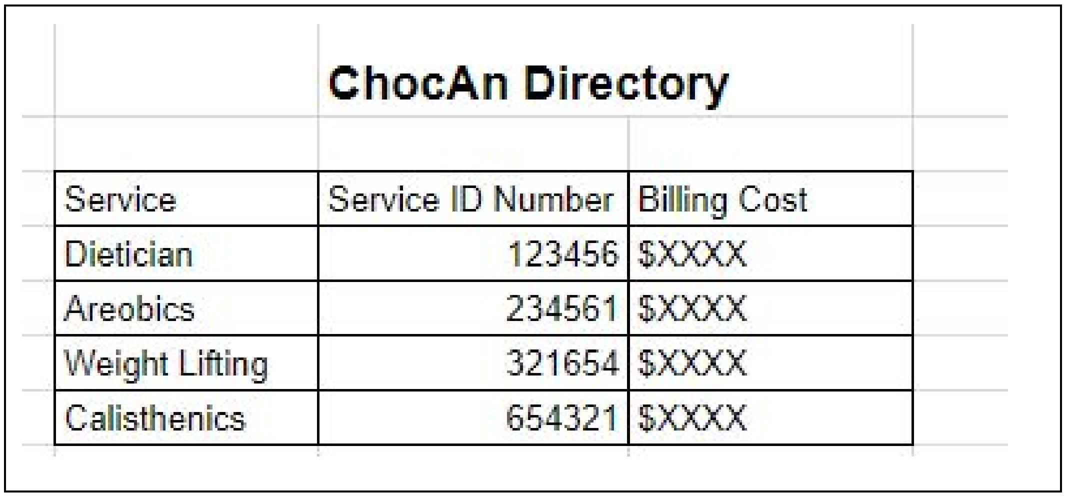
Total Services Provided	XX
Total Fee Cost	$XXXX



Manager Report

Provider ID	Had Paid	# of Consultant	Fee for the Week
XXXXXXX	yes	3	$199.00
XXXXXXX	no	1	$199.00

Total Number of Providers	Total Number of Services	Overall Fee
2	4	$399.98



ChocAn Directory

Service	Service ID Number	Billing Cost
Dietician	123456	$XXXX
Areobics	234561	$XXXX
Weight Lifting	321654	$XXXX
Calisthenics	654321	$XXXX