Ethan Fleming, Long Vo, Tommy Ngo, Alex Staley, Trent Wilson, Carl Bunt

The ChocAn Data Center

Test Plan Document

1 Introduction

In this document we (the developers) provide a detailed description of how we plan to test the ChocAn Data Center to ensure it meets the criteria specified in the requirements document, while minimizing bugs. In the pages that follow we describe in detail the procedures we plan to use with respect to unit, smoke and system testing.

1.1 Purpose and Scope

This document will lay out a detailed test plan for the system comprising the ChocAn Data Center. The tests described here show that the system described in the design document conforms to the specifications described in the requirements document.

The scope of this document, and the testing described herein, is congruent with the scope of the documents just mentioned. We do not test functionality associated with the ChocAn Simulator, communications and user interfaces such as the manager and provider terminals, hardware specifications, or any operations of ChocAn or associated entities.

Within the system comprising the ChocAn Data Center, our tests aim to ensure specified, predictable and deterministic behavior, to the greatest extent practical.

1.2 Target Audience

This document is written primarily for those who will be implementing the tests of the system it describes--namely, the developers and the dedicated (though nonexistent) team of quality assurance engineers assigned to implement testing. In writing this document we also have in mind the future contractors, engineers and data scientists who have a vested interest in the processes used to test the ChocAn Data Center.

1.3 Terms and Definitions

1.3.0 ChocAn Simulator:

The full software suite which the end-user will experience and interact with, and to which the ChocAn Data Center provides its functional services. This is not part of the scope of the software to be described by this document.

1.3.1 ChocAn Data Center:

The software application underlying the ChocAn Simulator. The testing of the ChocAn Data Center is the subject of this document.

1.3.2 Member:

1.3.2.0 member: A subscriber of ChocAn who pays the organization's required monthly fee and obtains access to services from providers recognized by ChocAn.

1.3.2.1 Member: A data abstraction representing a ChocAn member.

1.3.3 Provider:

1.3.3.0 provider: A health care professional who provides services (ref: sec 1.3.8) for ChocAn members. These providers can be dieticians, internists, exercise specialists, addiction treatment specialists or other care entities recognized by ChocAn.

1.3.3.1 Provider: A data abstraction representing a ChocAn provider.

1.3.4 Manager:

1.3.4.0 manager: An individual responsible for ChocAn operations. They can generate member, provider and summary reports at any time through the manager terminal. In interactive mode, they can also add, remove, and edit the records associated with members, providers, services and other managers.

1.3.4.1 Manager: A data abstraction representing a ChocAn manager.

1.3.5 Operator:

A data abstraction that forms the base of the derived classes ‘Manager’ and ‘User’.

1.3.6 User:

A data abstraction that forms the base of the derived classes ‘Provider’ and ‘Member’. Not to be confused with a user of the ChocAn Simulator.

1.3.7 Terminal:

The interface used by an end-user to interact with the ChocAn Simulator. Terminals are not within the scope of this design document.

1.3.7.0 Provider terminal: The terminal to be used only by providers.

1.3.7.1 Manager terminal: The terminal to be used only by managers.

1.3.8 I.D. Number:

A 9-digit identification number associated with a ChocAn-affiliated entity.

1.3.8.0 Member number: The I.D. number associated with a member.

1.3.8.1 Provider number: The I.D. number associated with a provider.

1.3.9 I.D. Number Status:

The status associated with verifying an I.D. number.

1.3.9.0 Validated: A validated I.D. number can be used to access an account or log a service.

1.3.9.1 Invalid: An incorrectly formulated or unused I.D. number is considered invalid.

1.3.9.2 Suspended: A suspended I.D. number is associated with a member who has been suspended (e.g. by having dues outstanding with ChocAn).

1.3.10 Service:

1.3.10.0 service: A consultation or treatment performed for a ChocAn member by a recognized provider that can be billed to ChocAn.

1.3.10.1 Service: A data abstraction representing a service.

1.3.10.2 Service code: The 6-digit code ChocAn associates with a predefined service.

1.3.10.3 Service list: A list of services provided that week. Service lists are data fields of both Member and Provider classes.

1.3.11 Provider Directory:

A list of service codes and their associated service names and fees, provided in a file format. The provider directory does not include information associated with ChocAn providers.

1.3.12 Interactive Mode:

An operating mode of the ChocAn Data Center that offers a manager additional functionality to modify service data and providers’ and members’ records.

1.3.13 Member Report:

A summary of the services obtained by a member over the past week.

1.3.14 Provider Report:

A summary of the services rendered members by a certain provider over the past week.

1.3.15 Summary Report:

A compendium of the providers and associated total fees to be paid by ChocAn over the past week.

2 Test Plan Description

In this section we describe what we will and will not test. We also outline a schedule for the described testing. In addition we describe the acceptance criteria and maximum fault tolerance of our system.

2.1 Scope of Testing

The scope of the tests we describe here is as close as we can make it to the scope of the system itself. We will test functionality associated with the user and system requirements specified in the requirements document--those that deal with the data pertaining to ChocAn members, providers and services which the ChocAn Data Center is designed to manage. Functionality associated with the ChocAn Simulator, communications and user interfaces such as the manager and provider terminals, hardware specifications, and any operations of ChocAn or associated entities outside the Data Center will not be tested.

2.2 Testing Schedule

Schedule will be based on scale, and testing will proceed from the fine-grained divisions of the system all the way up to the system as a whole. We will begin by unit testing the simplest operations--those that can be defined in several lines of code or less. These typically represent acquisition, deletion or simple editing of a single piece of data.

As we gain confidence in the correctness of each unit we test, we will integrate them into the system described in the design document. This process will be accompanied by component integration testing, which will focus on the interaction between objects in the system at various levels.

During the final phase, system testing, we will ensure that the functionality of the system as a whole conforms to the specifications laid out in the requirements document. This final phase begins with smoke testing (section 4) and concludes with the implementation of the system tests described in section 5.

The testing schedule begins presently and ends on December 6, 2019.

2.3 Release Criteria

2.3.1 Manager Terminal Criteria:

Operating the system from the manager terminal, the user should be able to perform all interactive mode functionality described in the requirements and design documents. Namely, they should be able to add, remove, fetch, edit and display Member, Provider and Service objects without causing any unintended side effects in the database. Specifically with fetch and display functions, data should not be altered at all.

2.3.2 Provider Terminal Criteria:

Operating the system from the provider terminal, the user should be able to request the provider directory, look up members by their I.D. number, and log services. The services logged should be immediately included in the relevant reports, if generated by a user from the manager terminal.

2.3.3 Report Criteria:

Operating the system from the manager terminal, the user should be able to request member reports, provider reports and summary reports. The reports requested should be up to date with the most recent services logged by providers via the provider terminal. Generation of reports should in no way alter the data involved.

3 Unit Testing

In this section we describe the functionality to be tested within the unit testing program. Like the system itself, the unit tests will be written in C++. We organize them in a way that parallels the organization of the system itself, with separate testing suites implemented for each major data abstraction in the system.

3.1 Testing Strategy

When every function in the system has passed its unit tests, the code in the “develop” branch will be pushed to the “alpha” branch on GitHub. All code in the “alpha” branch will then undergo integration and smoke testing, and if passing be pushed to the “beta” branch to undergo system testing. Any new changes to be made will be pushed to the “develop” branch and unit-tested again before they are pushed to the “alpha” branch. It would then be retested for integration before being pushed to the “beta” branch. Once a complete working version has been made, it is pushed to the “master” branch. For new changes to be made, it must go through each branch’s stage of testing before making a change to the master. In this way we hope to ensure correctness and reliability at each level of the system.

3.2 The ID Struct

This section describes test criteria associated with the identification information common to Members, Providers and Managers.

3.2.1 Name:

The name of the member, provider or manager should contain no more than 25 characters.

3.2.2 I.D. Number:

3.2.2.1 Manager I.D. number: The manager I.D. number should be exactly nine digits, and should begin with a ‘1’.

3.2.2.2 Provider I.D. number: The provider I.D. number should be exactly nine digits, and should begin with a ‘2’.

3.2.2.3 Member I.D. number: The member I.D. number should be exactly nine digits, and should be greater than 299999999.

3.2.3 Street Address:

The street address should contain no more than 25 characters.

3.2.4 City:

The city should contain no more than 14 characters.

3.2.5 State:

The state should comprise a two-letter, standard United States state code.

3.2.6 Zip Code:

The zip code should be a five-digit, valid United States zip code.

3.3 Manager Terminal

The manager terminal is where the bulk of the functionality is accessed. Unit tests in this section involve the reports to be generated weekly and on-demand by the manager, and also the data management functionality accessible in interactive mode.

3.3.1 Generating a Member Report:

3.3.1.0 Archive: Upon generation, the member report should be written to a file. Only one member report for each member should be stored at a time.

3.3.1.1 Member ID: The information stored in the member's ID struct (see section 3.2) should appear clear and correct in the member report.

3.3.1.2 Service date: For each service provided to the member that week, the date the service was provided should be clearly and correctly displayed in the member report.

3.3.1.3 Provider name: For each service provided to the member that week, the name of the service provider should be clearly and correctly displayed in the member report.

3.3.1.4 Service name: For each service provided to the member that week, the name of the service should be clearly and correctly displayed in the member report.

Note that the member report should NOT include any provider I.D. numbers, service codes, or any fees.

3.3.2 Generating a Provider Report:

3.3.2.0 Archive: Upon generation, the provider report should be written to a file. Only one provider report for each provider should be stored at a time.

3.3.2.1 Provider ID: The information stored in the provider's ID struct (see section 3.2) should appear clear and correct in the provider report.

3.3.2.2 Service date: For each service provided that week, the date the service was provided should be clearly and correctly displayed in the provider report.

3.3.2.3 Log date: For each service provided that week, the date the service was logged should be clearly and correctly displayed in the provider report.

3.3.2.3 Member name: For each service provided that week, the name of the member who received the service should be clearly and correctly displayed in the provider report.

3.3.2.4 Member I.D. number: For each service provided that week, the I.D. number of the member who received the service should be clearly and correctly displayed in the provider report.

3.3.2.5 Service code: For each service provided that week, the associated service code should be clearly and correctly displayed in the provider report.

3.3.2.6 Fee: For each service provided that week, the associated fee should be clearly and correctly displayed in the provider report.

3.3.3 Generating a Summary Report:

3.3.3.0 Archive: Upon generation, the summary report should be written to a file.

3.3.3.1 Provider name: For each provider who provided at least one service that week, the provider's name should be clearly and correctly displayed in the summary report.

3.3.3.2 Provider I.D. number: For each provider who provided at least one service that week, the provider's I.D. number should be clearly and correctly displayed in the summary report.

3.3.3.3 Number of services: For each provider who provided at least one service that week, the number of services that provider logged should be clearly and correctly displayed in the summary report.

3.3.3.4 Service fees owed: For each provider who provided at least one service that week, the total fee owed by ChocAn to that provider should be clearly and correctly displayed in the summary report.

3.3.3.5 Number of providers: The number of providers who logged at least one service that week should be clearly and correctly displayed in the summary report.

3.3.3.6 Total services: The total number of services logged by ChocAn service providers that week should be clearly and correctly displayed in the summary report.

3.3.3.7 Total fees owed: The sum of all fees owed to providers by ChocAn for the week's services should be clearly and correctly displayed in the summary report.

3.3.3.8 EFT file: When a summary report is generated, an EFT file should be created for each provider included in the report that contains their name and I.D. number as well as the dollar amounts to be transferred to them.

3.3.4 Interactive Mode:

3.3.4.1 Add a User: When a user is added, all the information should be input correctly or prompt an error.

3.3.4.2 Update a User: When a manager wants to update a user, an existing ID should be input in order to display that user's current information. The program should then prompt for input on an update on the user's information. The program should display the updated information and prompt for verification before making changes.

3.3.4.3 Remove a User: When a manager wants to remove a user, an existing ID must be input first. Once a matching profile is found, the program should display the information and prompt for verification before removing the user's profile.

3.3.4.4 Add a Service: When adding a service, it should be stored under the member's profile. It should also store the provider's ID who provided the service. If all the information is not within the requirement criteria the system should return an error.

3.3.4.5 Update a Service: When a manager wants to update a service, an existing service code should be input in order to display that service's information. The program should then prompt the user for input on an update on the service's information. The program should display the updated information and prompt for verification before making changes.

3.3.4.6 Remove a Service: When a manager wants to remove a service, an existing service ID should be input first. Once a matching ID is found, the program should display the information and prompt for verification before removing the service.

3.4 Provider Terminal

The provider terminal is the point of access for ChocAn service providers to interact with the Data Center. This section describes the testing of the functionality available therein.

3.4.1 Requesting a Provider Directory

When a provider requests a provider directory, it should be displayed immediately (in the ChocAn Simulator it will be sent via email upon request, but this functionality is not implemented here). The names of all services recognized by ChocAn, and their associated service codes, should be displayed clearly and correctly.

The provider directory should be written to an external data file.

3.4.2 Checking In a Member

When checking in a member from the provider terminal, the system should accept only valid member I.D. numbers. Invalid I.D. numbers, I.D. numbers of members marked as 'suspended,' provider I.D. numbers and manager I.D. numbers should all return an error.

3.4.3 Logging a Service

When the provider logs a service, all the required information must be entered in the correct format. Once all the information is entered, they will be displayed for verification. Once verified the service will be saved in the database

3.4.3.1 Entering the date: When the provider logs a service, they will be prompted to enter the date the service was provided. If this information is entered incorrectly (i.e. a format other than MM-DD-YYYY) the system should return an error. The system should also return an error should even a correctly-formatted date be invalid (i.e. the date is in the future).

3.4.3.2 Coding the service: When the provider logs a service, they will be prompted to enter the corresponding service code. The system should cross-reference the provider directory and display the name of the appropriate service for confirmation. The system should return an error if an invalid code is entered.

3.4.3.3 Entering comments: When the provider logs a service, they will be prompted to enter any associated comments, up to 100 characters. This field is optional, so this function should only return an error if more than 100 characters are entered.

3.4.3.5 Fee: When the provider successfully logs a service, the associated fee should be displayed.

3.5 Member Unit Tests

3.5.1 Retrieve Information:

Member user will not have their own terminal, but they will have a function to retrieve their service information. Their information will be displayed on the provider terminal when requested. An existing member ID must be entered before the information can be retrieved.

4 Smoke Testing

Smoke tests are a subset of system tests that we run to make sure of general functionality before we run the real system tests. The program will be deployed in a QA environment to verify the stability of the application. Smoke testing will be considered passed if and when the user can:

- Use a manager’s I.D. number to access interactive mode
- Add and remove members and providers from the database in interactive mode
- Log services from the provider terminal
- Generate a member, provider, and summary report
- Show that edits made in interactive mode are reflected in reports generated immediately thereafter

5 System Testing

The system will be tested on a generic console to validate that the system will function as specified in the requirements documents. Each class object’s functionality is tested at this point, and then the system tests the interactions between all of the class objects. Specifically, the system will be run using dummy manager, member, and provider information saved into their respective tables. These derived user class objects are then instantiated to test user interaction, and their status’ are validated with checks to the tables within the database object. Member and provider objects then interact using their interface with the database object in order to produce the appropriate reports which are detailed below.

5.1 Generating the Member Report:

We will test that the member reports are accurate and up-to-date with respect to all services received by the member that week. All the specified information should be correctly displayed, including the member's

- Name
- I.D. number
- Street address
- City
- State
- Zip code

In addition, for each service provided to the member that week the report should display:

- The date the service was provided
- The name of the provider
- The name of the service provided.

Note that the member report should NOT include provider I.D. numbers, service codes, or any fees. Both a manager and provider class should be able to access this functionality through their respective provider and manager class objects. Other object types will be used to check for correct error handling and data protection.

5.2 Generating the Provider Report:

We will test that the provider reports are accurate and up-to-date with respect to all services rendered by the provider that week. Provider objects will be instantiated with a valid dummy provider ID, and then their provider ID will be compared against the database to check if any services were provided for a given week. This system test relies on checking the interface between the provider and database interface. If services were provided within the timeframe, all the specified information should be correctly displayed, including the provider's

- Name
- I.D. number
- Street address
- City
- State
- Zip code

In addition, for each service provided that week the report should display

- The date the service was provided
- The date the service was logged
- The name of the member who received the service
- The I.D. number of the member who received the service
- The service code of the service provided
- The fee associated with the service provided

The provider report should also display the week's

- Total number of services provided to all members
- Total fee owed by ChocAn

5.3 Generating the Summary Report:

We will test that the summary reports are accurate and up-to-date with respect to all services logged by all providers that week. An instantiated Manager object, given a valid dummy manager ID, will test interfacing with the database object to retrieve the report, and all the specified information should be correctly displayed, including the:

- Name of every provider who logged at least one service that week
- Number of services each provider logged
- Fee to be paid to each provider who logged at least one service that week

Also included in a correctly generated summary report are the

- Total number of providers who logged at least one service that week
- Total number of services logged that week
- Total fees owed to all providers for the week

A provider object will not have derived function that implements generating this report, and thus it will not be tested to either have or not have this functionality.