Term Project: *ChocAn*

Test Plan Document

Written by Al Khatab Al Rashdi, Eleanore Clark, Michael Fulton, Ivan Lopez, Erin Rasmussen, and Tristan Smith

Table of Contents

1. Introduction

This document describes our team's strategy and methods for ensuring we deliver a product that fulfills the capabilities required by Chocoholics Anonymous (ChocAn) for the Chocoholics Data Processing Software (CDPS). This will be focused on the methods and strategies for ensuring that the product is delivered in an acceptable state.

This document is broken up into the following sections; Testing Plan, Unit Testing, Smoke Testing, and System Testing. When it comes to the Unit Testing this will be more of a general strategy for getting this done than the specific individual tests. The section on Smoke Testing will cover what kinds of tests will need to be performed to make sure the system is generally working. The System Testing section will focus on specific tests to make sure that the integrated system functions as expected, and meets the requirements specified in the requirements document.

1.1 Purpose and Scope

The use for this document will be the creation of an agreed testing strategy, and method to ensure the development team and the stakeholders are satisfied that we have methods in place for ensuring that the product will be correct. This document is limited to the methods with which the system will be tested, and will not cover the design, or specifications of the system.

The scope of the testing will cover the entire project, but each section of the document focuses on a different portion. Unit testing will cover the entirety of our code base, but will be limited to just that. The smoke testing will cover generally to make sure the system is working. The system testing will make sure the comprehensive system works as expected when all components are integrated. Thus everything from individual code methods, to the final product will be covered by testing.

1.2 Target Audience

The target audience for this document is stakeholders of the software, and, most likely, their development team. However, anyone in need of CDPS design documentation can benefit from this document, especially in conjunction with the requirements, and design documents for the same software. In addition to these, our development team will be using this document as our guide while development is in progress.

1.3 Terms and Definitions

The following are the terms and definitions of frequently used vocabulary in the following document.

- ChocAn:
    - Shorthand for Chocoholics Anonymous.
- CDPS:
    - Chocoholics Data Processing Software; the software requested by ChocAn.
- Constraint:
    - Something that limits the degree of freedom available in providing a solution.
- Dependency:
    - A type of relationship between activities that requires them to be performed in a particular order.
- EFT:
    - Electronic Funds Transfer.
- Manager:
    - An employee of ChocAn who works with the CDPS.
- Member:
    - A person seeking access to medical services through ChocAn.
- Provider:

- ○ A medical provider working with ChocAn to provide services to members.

- Software:
  - ○ A computer program and its related documentation.
- Stakeholder:
  - ○ A user of the software, including members, providers, and managers.
- Unit Test:
  - ○ Test code that checks to make sure paths through methods execute in expected methods.
- Smoke Test:
  - ○ A test which is performed during deployment as a general health check to make sure the system is functioning.
- System Test:
  - ○ More in-depth tests to ensure that the functionality of the program is correct when it's complete, and integrated into its usable form.

2 Test Plan Description

The purpose of this section is to outline the scope of testing, testing schedule, and release criteria of the CDPS. Scope of testing includes the scope of the test plan and describes the areas that will or will not be tested in the CDPS. Also, it describes the type of tests that will be performed during the testing process of CDPS. Testing schedule provides an overview of the schedule of the testing such as dates and times to perform testing. The release criteria section includes the maximum fault tolerance and the requirements that should be met before delivering.

2.1 Scope of Testing

This section is a high level overview of unit testing, smoke testing, system testing and all their related test cases, the testing schedule and the criteria that should be met before the system release. Each of these sections in the test plan provide test cases that cover the areas that will be tested in the CDPS to make sure that it works as it is supposed to be. The unit tests are checks on individual functions and object classes. The smoke test checks the basic functions in the CDPS such as making sure the input, and the output in the main menu is accessible to the user. The test will make sure the functionality of the CDPS is working as expected. It would make sure that data from the provider and the member is stored properly in the data structure. The system test is a more detailed test that focuses on component interactions. It would test the system as a whole. It would check the CDPS`s elements that are compatible, react correctly, and move the right data at the right time through the system.

2.2 Testing Schedule

- The Unit testing will be performed during the coding process to make sure the implementation is working as expected.
- The smoke testing will be performed in stages after the implementation and before the system. There will be multiple smoke testings before the final report. The duration of the testing depends on the testing cases.

- The system testing would be performed after all the system implementations are completed and smoke testing. The duration of system testing would depend on the testing cases. The testing must be completed before the final delivery and final report which is due on March 12th, 2021.

2.3 Release Criteria

The criteria that should be met before releasing the system is :

- All the system requirements should be met and it is good enough to be used by the system customer.
- It should perform and function as expected from the customer.
- Operations in the system must work and can be used.
- The system should be dependable.

Implementation would not be a big concern on this stage. The system requirements should be checked from the requirements documents to make sure that each requirement has been met before the release. The design on the other hand would not be a big concern if the requirements are delivered.

3. Unit Testing

The following section outlines the unit tests for the CDPS being developed for ChocAn. This section is aimed at the development team at ChocAn or anyone interested in the testing process of the CDPS. Furthermore, this section aims to outline the different considerations made in the design of the CDPS to insure proper functionality of functional and nonfunctional requirements as outlined in both the Requirements Document and Design Document for the CDPS.

3.1 Testing Framework

The strategy used for the CDPS is a test driven development model where testing is the foremost important step of the development process. By implementing tests before and during development, we can strive towards proper core functionality of the CDPS. To assist in the unit testing process, our development team will use a variety of tools that enable collaboration and testing. First we will be using the unit testing framework JUnit 5 the latest version of JUnit. This framework was chosen because of our former commitment to write the CDPS in Java and our development team's use of either IntelliJ or VS Code, which JUnit supports.

3.3 Version Control

Another tool used by our development team to assist in the unit testing process is version control via git and GitHub. All contributions to the code base will be via git and GitHub allowing for review prior to merging to the main branch. This allows tests to be performed prior to inclusion in the code base.

3.4 Testing Scope

The scope of our unit tests will be limited to the CDPS, so items not developed by our development team will not be included in our tests. That said, all items developed internally will be tested in some capacity including but not limited to I/O, management of data, etc.. We shall have at least 70% code coverage of program units.

4. Smoke Testing

In line with the concept of smoke testing, the most basic functions of the CDPS should be verified before more detailed tests are run. Our smoke testers will base their strategy on the following test cases:

4.1 Case 1: The CDPS boots successfully

- The CDPS must not crash on startup.

Testers will verify that the CDPS generally boots successfully.

4.2 Case 2: The main menu is accessible and usable

- The main menu must be visible on startup.
- The menu must be navigable via user input.

Passing these tests does not require individual menu options to work correctly. Testers will verify that the main menu is displayed correctly and takes input correctly.

4.3 Case 3: The manager terminal is accessible and usable

- The manager terminal must be accessible via user input from the main menu.
- The manager terminal must be navigable via user input.

Passing these tests does not require individual manager terminal options to work correctly. Testers will verify two things: that the manager terminal loads successfully upon selecting the associated option from the main menu; and that the manager terminal menu options can be selected from the manager terminal.

4.4 Case 4: The provider terminal is accessible and usable

- The provider terminal must be accessible via user input from the main menu.
- The provider terminal must be navigable via user input.

Passing these tests does not require individual provider terminal options to work correctly. Testers will verify two things: that the provider terminal loads successfully

upon selecting the associated option from the main menu; and that the provider terminal menu options can be selected from the provider terminal.

4.5 Case 5: The ChocAnList class descendants hold data

The descendants of the ChocAnList class must be capable of holding their respective data and correctly loading it from external files.

- ServiceRecord class instances must be able to load and hold the following fields: creationDateTime, LocalDateTime, serviceDateTime, providerNumber, providerName, memberNumber, memberName, serviceCode, and comments.
- Provider class instances must be able to load and hold the following fields: providerNumber, providerName, streetAddress, city, state, and zipCode.
- Member class instances must be able to load and hold the following fields: memberNum, memberName, streetAddress, city, state, zipCode, membershipStatus, and feesDue.
- Service class instances must be able to load and hold the following fields: code, name, and price.

Testers will verify that data is loaded correctly from external files into each of the above classes.

5. System Testing

The purpose of system testing is to ensure the CDPS System components properly interact with one another. These components must be compatible, interact correctly, and transfer the correct data at the right time across their interfaces.

5.1 Test Case 1: Checking Account Credentials

Both the providers and the managers must be able to access permissions based on their account credentials. The CDPS System needs to ensure that credentials are processed and accesses the group requesting it. Upon receiving a request from the terminal, the access permissions must match the data attempting to be accessed.

- If permission is granted, allow access
- If permissions is not granted, do not allow access and display an error message

5.2 Test Case 2: Accessing Provider Directory

The providers shall have access to a provider directory, from their terminal, that allows them to view service codes. The provider directory should supply a listing of six digit service code numbers and a corresponding service name and fee to the provider email. The provider directory must:

1. Allow the provider to input billing information for a service report.
2. Prompt the provider to enter a six digit service code number for the provided service.
3. Allow to be requested and sent via email and displayed alphabetically to the provider with names of services and corresponding service code numbers.
4. The provider should then be able to enter a number from the directory.
    a. If successful, a fee is displayed to the terminal from the service code selected from the provider directory for confirmation.
    b. If unsuccessful, an error message is displayed.

5.3 Test Case 3: Providing Reporting Services

The CDPS must correctly log completed services and store this data in a file at the ChocAn Data Center, which will be tied to both a provider and a member. The provider must be able to bill ChocAn for services by entering information about the services provided and have the data stored.

- A provider provides a member with a service and needs to input information about the service for billing.
- The provider logs into their terminal and verifies that the member has an active membership with ChocAn using the member's name and member number.
- Date of the service will be keyed in using the following format: MM-DD-YYYY.
- Provider finds the correct service code from the provider directory.
    - If the service code is incorrect, an error message will appear.
- The provider enters any comments about the service that was provided .
- The above information is written to disk.
- A fee for the service is displayed to the provider's terminal.
- The fee is then entered into a form with the current date, time, date the service was provided, member name, number, and service code.

5.4 Test Case 4: Creating Manager Reports

Managers will have the capability to manually generate reports through their terminal throughout the week in addition to the weekly reports provided by the CDPS. The reports received by the manager must include member, provider, EFT, and weekly reports.

1. The manager must be able to request one/all of the following reports:
    a. Member Report
    b. Provider Report
    c. EFT Report
    d. Weekly Report (Automatically sent at the end of the week)
2. A report generated from data kept on file in the database should be able to be returned to the manager.