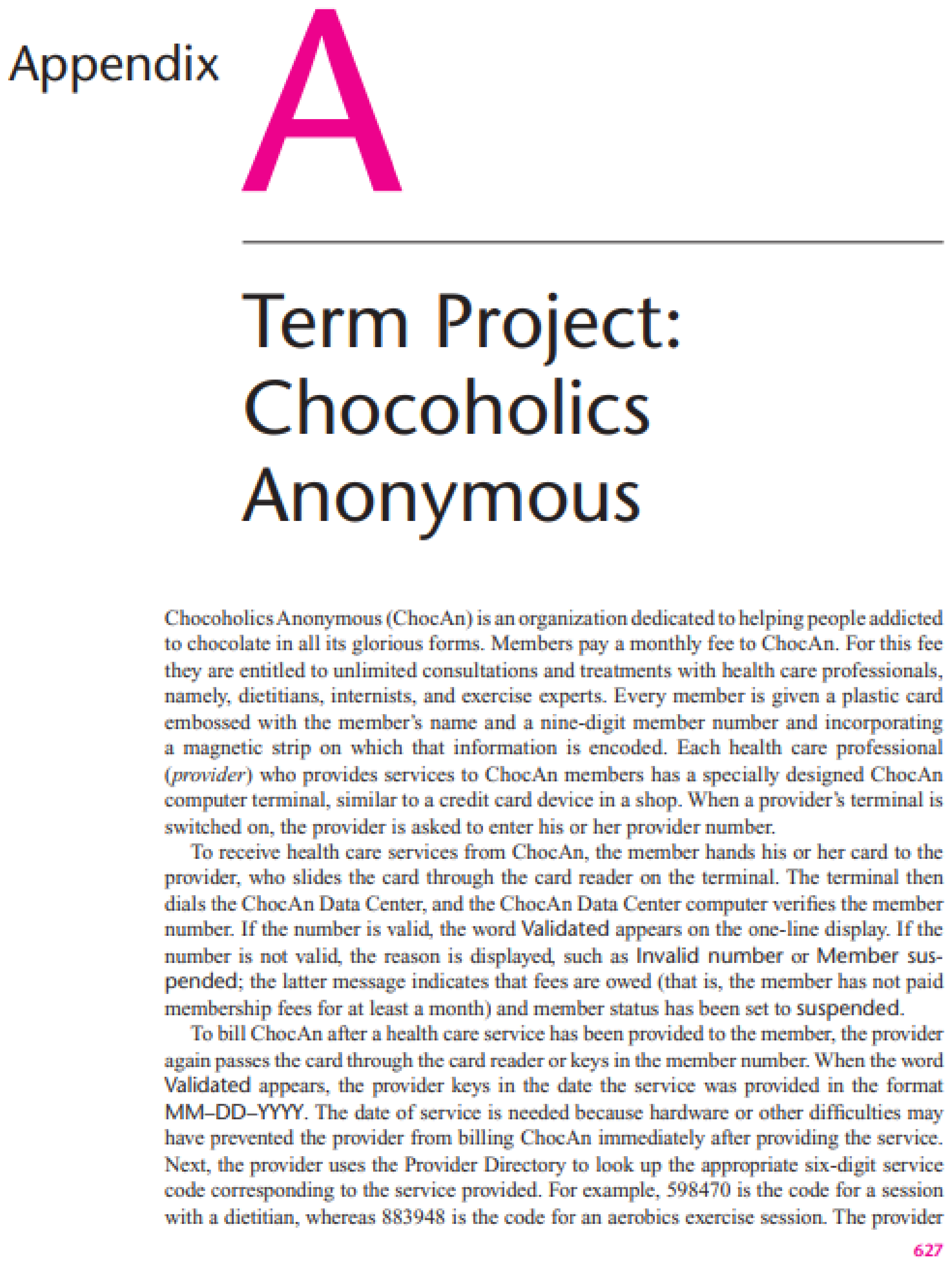
Capstone Project for Software Engineering (COSC-4360)

Repository: https://github.com/ljclarke914/Senior-Project

<u>Description</u>

The following project was assigned to a group consisting of myself and 3 other classmates. It was coded in Java using NetBeans, and we used MySQL Workbench to create our database. We used GitHub to collaborate on the project.

<u>Project Overview</u>



Appendix A

Term Project: Chocoholics Anonymous

Chocoholics Anonymous (ChocAn) is an organization dedicated to helping people addicted to chocolate in all its glorious forms. Members pay a monthly fee to ChocAn. For this fee they are entitled to unlimited consultations and treatments with health care professionals, namely, dietitians, internists, and exercise experts. Every member is given a plastic card embossed with the member's name and a nine-digit member number and incorporating a magnetic strip on which that information is encoded. Each health care professional (*provider*) who provides services to ChocAn members has a specially designed ChocAn computer terminal, similar to a credit card device in a shop. When a provider's terminal is switched on, the provider is asked to enter his or her provider number.

To receive health care services from ChocAn, the member hands his or her card to the provider, who slides the card through the card reader on the terminal. The terminal then dials the ChocAn Data Center, and the ChocAn Data Center computer verifies the member number. If the number is valid, the word Validated appears on the one-line display. If the number is not valid, the reason is displayed, such as Invalid number or Member suspended; the latter message indicates that fees are owed (that is, the member has not paid membership fees for at least a month) and member status has been set to suspended.

To bill ChocAn after a health care service has been provided to the member, the provider again passes the card through the card reader or keys in the member number. When the word Validated appears, the provider keys in the date the service was provided in the format MM–DD–YYYY. The date of service is needed because hardware or other difficulties may have prevented the provider from billing ChocAn immediately after providing the service. Next, the provider uses the Provider Directory to look up the appropriate six-digit service code corresponding to the service provided. For example, 598470 is the code for a session with a dietitian, whereas 883948 is the code for an aerobics exercise session. The provider

627

then keys in the service code. To check that the service code has been correctly looked up and keyed in, the software product then displays the name of the service corresponding to the code (up to 20 characters) and asks the provider to verify that this is indeed the service that was provided. If the provider has entered a nonexistent code, an error message is printed. The provider also can enter comments about the service provided.

The software product now writes a record to disk that includes the following fields:

Current date and time (MM–DD–YYYY HH:MM:SS).
Date service was provided (MM–DD–YYYY).
Provider number (9 digits).
Member number (9 digits).
Service code (6 digits).
Comments (100 characters) (optional).

The software product next looks up the fee to be paid for that service and displays it on the provider's terminal. For verification purposes, the provider has a form on which to enter the current date and time, the date the service was provided, member name and number, service code, and fee to be paid. At the end of the week, the provider totals the fees to verify the amount to be paid to that provider by ChocAn for that week.

At any time, a provider can request the software product for a Provider Directory, an alphabetically ordered list of service names and corresponding service codes and fees. The Provider Directory is sent to the provider as an e-mail attachment.

At midnight on Friday, the main accounting procedure is run at the ChocAn Data Center. It reads the week's file of services provided and prints a number of reports. Each report also can be run individually at the request of a ChocAn manager at any time during the week.

Each member who has consulted a ChocAn provider during that week receives a list of services provided to that member, sorted in order of service date. The report, which is also sent as an e-mail attachment, includes:

Member name (25 characters).
Member number (9 digits).
Member street address (25 characters).
Member city (14 characters).
Member state (2 letters).
Member ZIP code (5 digits).
For each service provided, the following details are required:
  Date of service (MM–DD–YYYY).
  Provider name (25 characters).
  Service name (20 characters).

Each provider who has billed ChocAn during that week receives a report, sent as an e-mail attachment, containing the list of services he or she provided to ChocAn members. To simplify the task of verification, 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. That is, the fields of the report include:

Provider name (25 characters).
Provider number (9 digits).
Provider street address (25 characters).
Provider city (14 characters).
Provider state (2 letters).
Provider ZIP code (5 digits).
For each service provided, the following details are required:
- Date of service (MM–DD–YYYY).
- Date and time data were received by the computer (MM–DD–YYYY HH:MM:SS).
- Member name (25 characters).
- Member number (9 digits).
- Service code (6 digits).
- Fee to be paid (up to $999.99).

Total number of consultations with members (3 digits).
Total fee for week (up to $99,999.99).

A record consisting of electronic funds transfer (EFT) data is then written to a disk; banking computers will later ensure that each provider's bank account is credited with the appropriate amount.

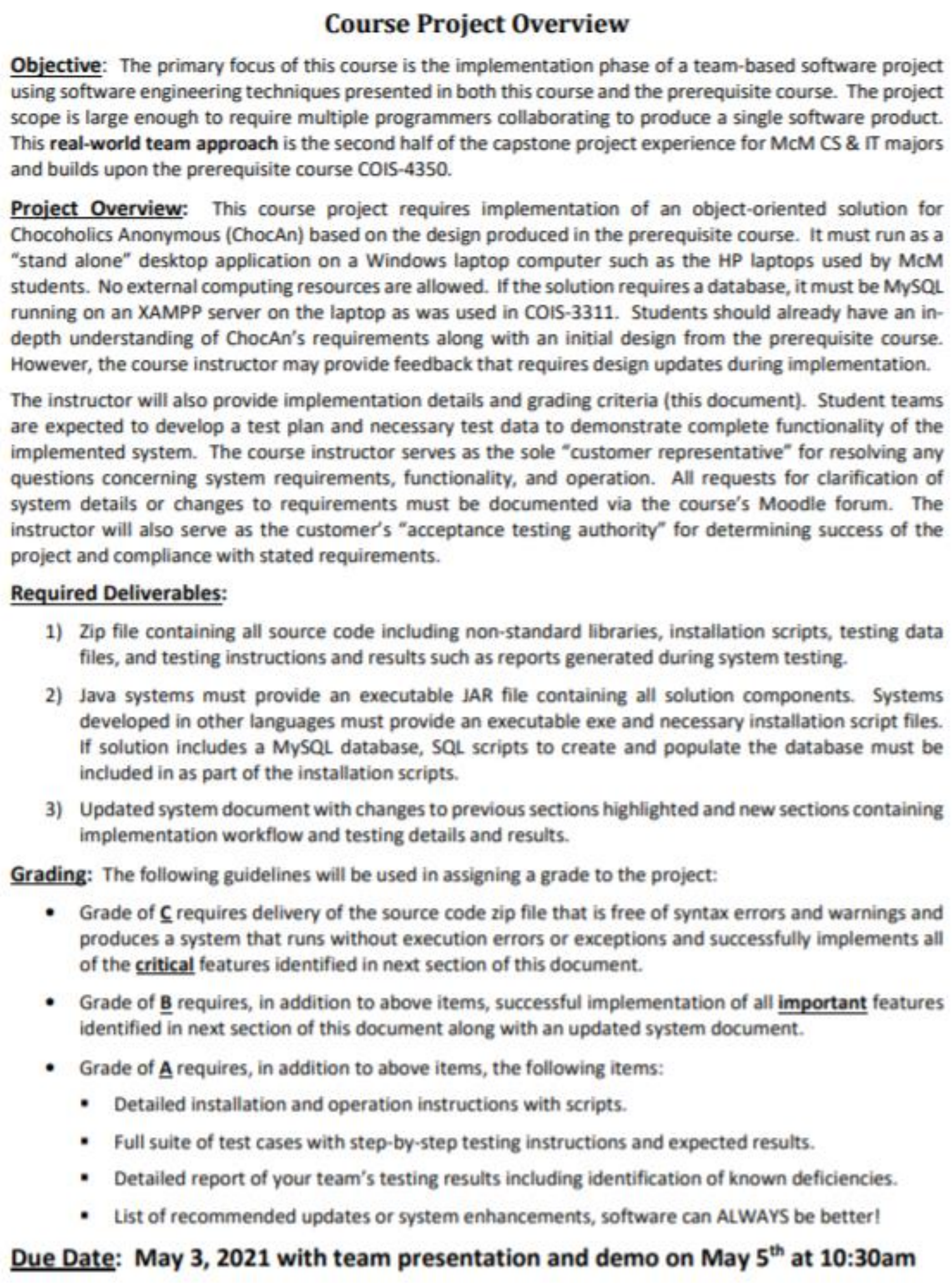
A summary report is given to the manager for accounts payable. The report lists every provider to be paid that week, the number of consultations each had, and his or her total fee for that week. Finally, the total number of providers who provided services, the total number of consultations, and the overall fee total are printed.

During the day, the software at the ChocAn Data Center is run in interactive mode to allow operators to add new members to ChocAn, to delete members who have resigned, and to update member records. Similarly, provider records are added, deleted, and updated.

The processing of payments of ChocAn membership fees has been contracted out to Acme Accounting Services, a third-party organization. Acme is responsible for financial procedures such as recording payments of membership fees, suspending members whose fees are overdue, and reinstating suspended members who have now paid what is owing. The Acme computer updates the relevant ChocAn Data Center computer membership records each evening at 9 P.M.

Your organization has been awarded the contract to write only the ChocAn data processing software; another organization will be responsible for the communications software, for designing the ChocAn provider's terminal, for the software needed by Acme Accounting Services, and for implementing the EFT component. The contract states that, at the acceptance test, the data from a provider's terminal must be simulated by keyboard input and data to be transmitted to a provider's terminal display must appear on the screen. A manager's terminal must be simulated by the same keyboard and screen. 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 provider reports should be handled the same way. The Provider Directory must also be created as a file. None of the files should actually be sent as e-mail attachments. 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.

Project Requirements



Course Project Overview

Objective: The primary focus of this course is the implementation phase of a team-based software project using software engineering techniques presented in both this course and the prerequisite course. The project scope is large enough to require multiple programmers collaborating to produce a single software product. This **real-world team approach** is the second half of the capstone project experience for McM CS & IT majors and builds upon the prerequisite course COIS-4350.

Project Overview: This course project requires implementation of an object-oriented solution for Chocoholics Anonymous (ChocAn) based on the design produced in the prerequisite course. It must run as a "stand alone" desktop application on a Windows laptop computer such as the HP laptops used by McM students. No external computing resources are allowed. If the solution requires a database, it must be MySQL running on an XAMPP server on the laptop as was used in COIS-3311. Students should already have an in-depth understanding of ChocAn's requirements along with an initial design from the prerequisite course. However, the course instructor may provide feedback that requires design updates during implementation.

The instructor will also provide implementation details and grading criteria (this document). Student teams are expected to develop a test plan and necessary test data to demonstrate complete functionality of the implemented system. The course instructor serves as the sole "customer representative" for resolving any questions concerning system requirements, functionality, and operation. All requests for clarification of system details or changes to requirements must be documented via the course's Moodle forum. The instructor will also serve as the customer's "acceptance testing authority" for determining success of the project and compliance with stated requirements.

Required Deliverables:

1) Zip file containing all source code including non-standard libraries, installation scripts, testing data files, and testing instructions and results such as reports generated during system testing.
2) Java systems must provide an executable JAR file containing all solution components. Systems developed in other languages must provide an executable exe and necessary installation script files. If solution includes a MySQL database, SQL scripts to create and populate the database must be included in as part of the installation scripts.
3) Updated system document with changes to previous sections highlighted and new sections containing implementation workflow and testing details and results.

Grading: The following guidelines will be used in assigning a grade to the project:

- Grade of **C** requires delivery of the source code zip file that is free of syntax errors and warnings and produces a system that runs without execution errors or exceptions and successfully implements all of the **critical** features identified in next section of this document.
- Grade of **B** requires, in addition to above items, successful implementation of all **important** features identified in next section of this document along with an updated system document.
- Grade of **A** requires, in addition to above items, the following items:
  - Detailed installation and operation instructions with scripts.
  - Full suite of test cases with step-by-step testing instructions and expected results.
  - Detailed report of your team's testing results including identification of known deficiencies.
  - List of recommended updates or system enhancements, software can ALWAYS be better!

Due Date: May 3, 2021 with team presentation and demo on May 5th at 10:30am

Course Project Requirements:

- The following items are **CRITICAL** features:
  - All teams must use the GitHub repository to manage the collaboration for this project. The course instructor will create the project and invite students to participate. Basic use of GitHub will be demonstrated in class and additional guidance may be provided via discussion forums.
  - DOS-style console user interface (or GUI in next section).
    - Upon startup, program must display a "splash" screen with the system name, course number and semester, list of implementation team members and a prompt to allow the user to either continue with processing or exit the system.
    - When the user continues with system processing, program must prompt for and obtain the working directory that provides necessary data files for a test run and collects associated output files for test result analysis (must default to the execution directory).
    - Program then displays the main menu screen which must contain an exit option.
    - Subsequent screens would then provide interfaces to selected functions as needed.
  - Program must implement all application features described in the system document produced during the prerequisite course with any changes or clarifications "approved" by the user representative during the current semester (via discussion forum posts).
  - Program must terminate normally without any exceptions when acceptable input data is provided and must display a closing screen message indicating success of the application.
  - Program must terminate normally when <u>unacceptable</u> input data is provided and must display a closing screen message indicating failure of the application (details of issue not required).
  - Program must use programmer-defined objects to store and work with system's data and perform all system processing via programmer-defined methods (forces use of OOP).
- The following items are **IMPORTANT** features:
  - Full graphical user interface (GUI) instead of DOS-style console user interface.
    - Upon startup, program must display a window with your team's system name that is moveable and resizable with standard window controls. It must display system name, course number, and semester as above but the team info gets moved to an "About" screen that is available through a "Help" menu item. The startup screen must also provide buttons and a "File" menu with commands and hot keys to either continue with processing or exit.
    - Upon option to continue processing, GUI must use appropriate file open and file save dialog screens with meaningful titles to obtain necessary directory and file names. Readability or writeability of each must be checked and resolved before continuing.
    - After all directories and files have been selected, display a summary dialog containing names and locations of all selected files and present the user with options to confirm and continue or cancel processing.
    - Upon option to continue, screen must display main system menu with a "status" area to indicate success or failure of previous processing action upon return to main menu. Subsequent screens must be presented as appropriate for selected function. Main menu screen must have an option to exit the system.
    - Screens for displaying reports must be non-editable, scrollable windows. Reports generated as pdf files should open in Adobe Reader windows.
  - Program must perform data validation checks for any input provided by user and react appropriately to prevent subsequent exceptions or failures during processing.