Criterion B

Design

Contents

[Use Cases 3](#_Toc159961110)

[Flowchart Overview 4](#_Toc159961111)

[Prototype Design 5](#_Toc159961112)

[Final Design 7](#_Toc159961113)

[Data Types Used 9](#_Toc159961114)

[Unified Modeling Language (UML) Diagrams 10](#_Toc159961115)

[Hierarchical Chart 13](#_Toc159961116)

[Connection Chart 14](#_Toc159961117)

[Data Flow 15](#_Toc159961118)

[Flowcharts 17](#_Toc159961119)

[Testing Plan 22](#_Toc159961120)

# Use Cases

A diagram of a person's data flow

Description automatically generated

Figure 1 All use cases which the client should be able to do.

# Flowchart Overview

A diagram of a flowchart

Description automatically generated

Flowchart 1 Main dashboard.

# Prototype Design

Designs created after the initial meeting with the client[[1]](#footnote-1).

A screenshot of a computer screen

Description automatically generated

Figure 3 Login, signup, and user dashboard.

A screenshot of a computer

Description automatically generated

Figure 4 Company window.

A screenshot of a computer

Description automatically generated

Figure 5 Company merging.

# Final Design

Designs created after second meeting with the client[[2]](#footnote-2).

A screenshot of a computer

Description automatically generated

Figure 8 Login, signup, and user dashboard.

A screenshot of a computer

Description automatically generated

Figure 9 Company dashboard.

A screenshot of a computer

Description automatically generated

Figure 10 Merging interface.

# Data Types Used

Table 1 Data types used.

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Variable** | **Reasoning** |
| String | username | * Strings allowed for usernames to have any characters. |
| password | * More choice of characters means more secure password. |
| File | file | * File object from java.io.File library[[3]](#footnote-3). * File object is used by various libraries. * Company or user file can be loaded as File object. |
| User | currentUser | * Current user represented as variable of User type.   + Descriptive variable.   + Serves purpose of constructing user as an object.   + User object has useful methods I created. * Current user object has data which needs to be accessed. |
| Company | head | * Used in CompanyList class as beginning of linked list. |
| current | * Current company in linked list. * Useful when looping through linked list. * Allows for checking if there is a next company with methods. |

# Unified Modeling Language (UML) Diagrams

Models of each proposed class.

|  |  |
| --- | --- |
| **Authentication** | |
| -  -  - | usersFile: String  currentUser: User  debug: boolean |
| +  -  -  -  -  +  - | Authentication()  signUp(username: String, password: String): User  logIn(username: String, password: String): User  writeString(file: RandomAccessFile, string: String): void  readString(file: RandomAccessFile, string: String): void  getUser(): User  toggleDebug(toggle: boolean): void |

UML Diagram 1 Authentication class.

|  |  |
| --- | --- |
| **User** | |
| -  -  -  -  - | username: String  isAdmin: boolean  file: File  companyList: CompanyList  debug: boolean |
| +  +  +  +  +  - | User(username: String, isAdmin: Boolean, filePath: String)  User()  getUsername(): String  getIsAdmin(): boolean  getCompanyList(): CompanyList  toggleDebug(toggle: boolean) |

UML Diagram 2 User class.

|  |  |
| --- | --- |
| **CompanyList** | |
| -  -  +  + | head: Company  file: File  debug: boolean  companyListLoaded: boolean |
| +  +  +  +  +  +  +  + | CompanyList(file: File)  add(company: Company)  save()  toArray(): Company[]  length(): int  isEmpty(): boolean  exists(fileName: String): boolean.  getHead(): Company |

UML Diagram 3 CompanyList class

|  |  |
| --- | --- |
| **Company** | |
| -  -  -  -  -  -  -  +  + | next: Company  name: String  description: String  country: String  tickerSymbol: String  revenues: ArrayList<Statistic>  costs: ArrayList<Statistic>  debug: boolean  companyLoaded: boolean |
| +  +  +  + | Company(filePath: String)  Company()  getNext(): Company  setNext(next: Company) |

UML Diagram 4 Company class

|  |  |
| --- | --- |
| **Statistic** | |
| --  - | name: String  filePath: String  data: ArrayList<Data> |
| +  +  +  +  +  + + | Statistic(name: String, filePath: String)  Statistic(name: String, data: ArrayList<Data>)  getName(): String  readData()  getData(): ArrayList<data>  toString(): String  extrapolateData(monthsToExtrapolate: int): ArrayList<Data> |

UML Diagram 5 Statistic class

|  |  |
| --- | --- |
| **Data** | |
| -  -  - | year: int  month: int  value: int |
| +  +  +  + + | Data(year: int, month: int, value: int)  getYear(): int  getMonth(): int  getValue(): int  toString(): String |

UML Diagram 6 Data class

# Hierarchical Chart

Illustration of how proposed windows and interfaces could be linked together.

A screenshot of a computer screen

Description automatically generated

Figure 11 Hierarchal chart.

# Connection Chart

A representation of the proposed class relationships.

A diagram with black text

Description automatically generated

Figure 12 Connection chart.

# Data Flow

A diagram of a login

Description automatically generated

Data Flow 1 Process of user log in.

A diagram of a company

Description automatically generated

Data Flow 2 Process of merging companies.

A diagram of a company

Description automatically generated

Data Flow 3 Process of user adding a company.

A diagram of a company data processing

Description automatically generated

Data Flow 4 Process of creating company valuation.

# Flowcharts

A diagram of a flowchart

Description automatically generated

Flowchart 2 Initial screen.

A diagram of a flowchart

Description automatically generated

Flowchart 3 User dashboard.

A screenshot of a diagram

Description automatically generated

Flowchart 4 Company addition and removal.

A diagram of a company

Description automatically generated

Flowchart 5 Company dashboard flowchart.

A diagram of a flowchart

Description automatically generated

Flowchart 6 Company statistics interface.

# Testing Plan

Table 2 Testing plan outlining tests that will need to be performed on the program to ensure it meets success criteria.

|  |  |  |  |
| --- | --- | --- | --- |
| **Success Criterion Tested** | **Description of Test** | **Method of Test** | **Expected Outcome** |
| 1 | Check if company can be added to user file. | * Open the file before adding. * Check contents of file. * Run program and open dashboard. * Press “Add Company” button. * Close program and check contents of user file. | * User file should have new company file name added. |
| Test if companies can be deleted from user file. | * Open file before adding it, check contents. * Run program and open dashboard. * Open company that has been added. * Click “Delete” button. * Close program. * Open user file. | * User file should not have name of removed company file. |
| 2 | Check if all company data is stored on file. | * Open company file. * Check data. * Open program. * Perform action that would change data. | * Contents of company file altered. |
| 3 | Check if user inputs are valid. | * Input abnormal values into input forms in program. | * Program should output data is invalid. * You should be prompted to enter data again. * Or kicked out of input process. |
| Check if existing username check is performed. | * Open signup interface. * Input existing username. | * Program should output that username already exists. |
| Check if company file added by user is CSV file. | * Open main dashboard. * Go to add company. * Try to add a file of other file type. | * Program should output that the file is not CSV. * File should be rejected. |
| 4 | Check if interface is easy-to-use. | * Give a program to someone unfamiliar with the program. * Command them to perform simple task. | * They should be able to perform it without needing help. * Completion in a reasonable amount of time. |
| 5 | Check if documentation is embedded. | * Check if “i” button exists. * Check if it exists on all interfaces. * Click the button. | * Information should pop up. * Information for almost every page should exist. |
| Check if documentation is easily accessible. | * Check if the “i” button appears on every interface. * Open all main program interfaces. * Check top left corner for button. | * “i” button should be in top left corner. * True for all main interfaces. |
| * Give program to user unfamiliar. * Ask them to open documentation. | * They should intuitively click “i” button to get documentation. |
| 7 | Check if company data is retrievable from opening file. | * Open company file. | * Company files should be readable by the user. * User should be able to edit file as well. |
| Check if company data is retrievable by program from company file. | * Open program dashboard. * Add company. | * Company should be added. * Name should appear properly. * Name indicates data was loaded. |
| * Open company dashboard. * Open statistics. | * If statistics are listed, it means data was loaded. |
| 7 | Check if statistics can be graphed. | * Open company panel. * Click on “Statistics” button. * Click on a statistic button. | * Popup with a graph of the data for that statistic should popup. |
| 8 | Check if user can edit basic company details. | * Open company dashboard. * Input new name, description, and country. * Click the button that updates the details. | * Open company file. * Company file should have new values. |
| 9 | Test if user can use program to estimate value of company. | * Calculate company value beforehand. * Open company dashboard. * Press the “Calculate Value” button. * Input number of years to predict for (enter 0). | * A popup should appear with estimated value. * The company value that you calculated earlier should somewhat align. * Or estimate should be reasonable. |
| 10 | Test if program can project company value. | * Open company dashboard. * Press the “Calculate Value” button. * Input random number of years. * Repeat. | * Every time different value should be predicted. |
| 11 | Test if merging works. | * Open company dashboard. * Click “Merge” button. * Select two companies to be merged. * Perform merge. | * New CSV file for merged company should be created. * New file should have combined statistics data. |

1. Please see Appendix A: 1st Meeting with the Client. [↑](#footnote-ref-1)
2. Please see Appendix B: 2nd Interview with the Client. [↑](#footnote-ref-2)
3. See Libraries section of this document. [↑](#footnote-ref-3)