Jason Eide, Asif Zaman, Ethan Morris, Jose Hernandez

CSCE 3513 Software Engineering Spring 2015

Abstract

This document contains the report for the third iteration of a Personal Financial Management System

Project Deliverable

Iteration #3 Report

This document gives an overview of the third iteration of the Personal Financial Management System. In this report, we first discuss the delivered features/user stories (specified by the customer). The next section goes over the methodology used in building this software system, specifically principles of Extreme Programming (XP) that were utilized. Finally we outline the cases used in testing each of the user stories.

**User Stories**

For this iteration, the customer requested the main focus should be implementing the Debt Payoff Calculator. The corresponding user story for this feature from the Requirements Specification document was simply titled “Debt Payoff.” The implementation for the Debt Payoff Calculator involved designing a fairly complex algorithm based on the “avalanche” method of tackling financial debt. In the avalanche method, a person’s debts are paid off in order of highest interest rate first. For an example of how the avalanche method works, please refer to the chart below for sample debts.

Table 1: Example Values for Debt Payoff

|  |  |  |  |
| --- | --- | --- | --- |
| **Debt Name** | **Amount** | **Interest Rate** | **Minimum Monthly Payment** |
| Mortgage | $150,000 | 4.00% | $1100 |
| Student Loan | $30,000 | 7.50% | $400 |
| Credit Card | $10,000 | 15.00% | $300 |

From the sample values in the chart, the user would have a minimum monthly payment of $1800 (all three debts combined). If the user choose to pay more than the minimum payment, for example an extra $200 (making it an even $2000), the additional amount would first be applied to the credit card (since it has the highest interest rate), until that particular debt was paid off. The user would then have two remaining debts (the mortgage and student loan) for a combined minimum payment of $1500. However, they would continue to pay $2000, with the extra $500 going toward the student loan until that debt was paid off. Likewise, once the student loan was paid off, $2000 would go toward the $1100 mortgage until that was finally paid off as well.

In coding this feature, some of the intricate details included adjusting for a dynamic number of debts input by the user as well as properly calculating each debt concurrently rather than sequentially. Because there is overlap in paying down debts, it was important to check for paid off amounts and new highest interest rates for each monthly payment cycle. To help the user easily visualize their debt payoff projections, we implemented a linear graph that shows the user exactly how long it will take them to pay off each individual debt using the avalanche method.

Additionally, for this iteration we choose to implement a previously unmentioned user story, a “User Management” tab. This tab allows the user to change their password as well as to delete their account. The database is updated accordingly to what the user chooses to do.

For this iteration, our group spent a total of about 37 person-hours (Jason: 8 hours, Jose: 14 hours, Ethan: 5 hours, Asif: 10 hours). Our initial estimate for this iteration was 40 total person-hours. We estimated that 20 person-hours would be spent on the actual algorithm for the debt payoff calculator, with 10 hours each for implementing the GUI interface for the calculator and the user management tab. While we came in just under our total estimate, our estimates for person-hours on the individual sections were a bit off however. Jason was able to finish the GUI interface in 6 hours and Ethan was able to implement the User Management tab in 5 hours. Figuring out the algorithm for the debt payoff calculator took up the bulk of the remaining 26 person-hours.

**Implementation of Extreme Programming Practices**

For this software system, we are employing Extreme Programming (XP) for development. This includes principles such as incremental planning, small releases, simple design, pair programming, collective ownership, sustainable pace, and on-site customer interaction. The majority of the XP practices in this iteration are the same as the previous two iterations, as our team is employing XP for the entire development cycle.

1. **Incremental Planning** – Our requirements were created through the use of user stories, which were reviewed by the customer. The customer determined which stories should become development “tasks”.
2. **Small Releases** - The Expense/Income Manager is the main component and also provides the minimal useful set of functionality to the customer. Future releases will add functionality to the overall system. This release primarily implemented the “Debt Payoff” user story.
3. **Simple Design** – Each individual component only executes the functionality for the user story that it solves.
4. **Pair Programming** – Approximately 9 of the 37 total person-hours on this iteration were through pair programming. Jose and Asif worked as a pair for 6 hours and Jason and Asif also worked as a pair for 2 hours, all on the debt payoff algorithm. Ethan and Jason worked together as a pair for 1 hour on the User Management tab. For part of these pair programming sessions, all four group members worked at the same table (still divided into two pairs, but able to communicate directly with one another).
5. **Collective Ownership** – Decisions concerning individual components or the overall project were made together as a group to solidify collective ownership. Finally, all code is available to all members of the group.
6. **Sustainable Pace** – Group members put in hours reflective of their overall workload which helped combat loss of code quality and production. Long programming sessions ended if productivity loss became noticeable.
7. **On-site Customer** – Early during this iteration cycle, the customer met directly with the entire team to discuss which system requirements should be implemented. Additionally, the customer is available for consultation at any time through email as well as multiple days per week in person (for this particular iteration, additional consultation was not necessary).

The biggest challenge for this iteration was designing the debt payoff algorithm. Jose and Asif worked primarily on the algorithm (with Jose dedicating his entire 14 person-hours for this iteration and Asif 8 of his 10). Figuring out how to make the calculation work within the context of our system required three mini-iterations on just the design alone. The first iteration was simply to figure out the calculations for one single debt using hardcoded sample data. The second iteration was to figure out how to use data input by the user for one single debt. Finally the third (and final) iteration implemented user input for a variable number of debts.

**Testing**

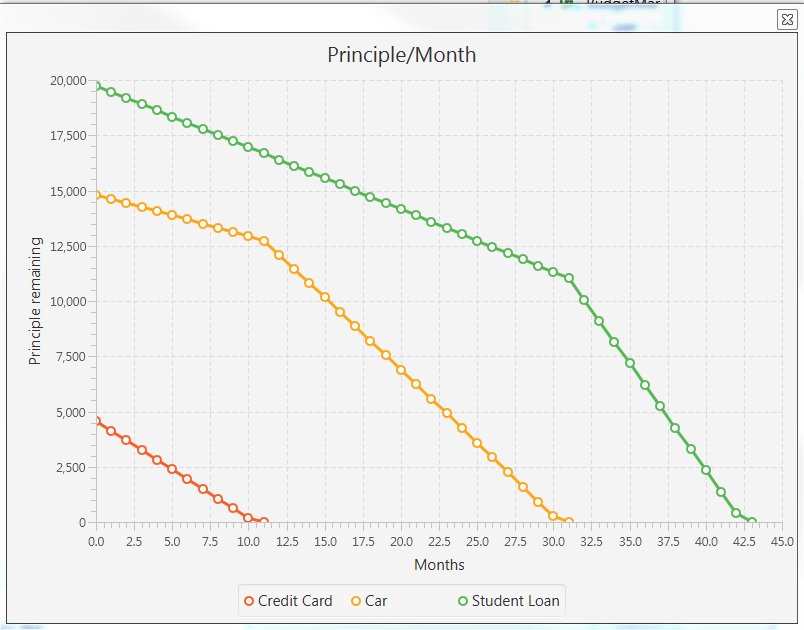
Testing for the third iteration included testing user input on the Debt Tracker page, verifying the payoff chart after clicking the calculation button on the debt tracker page, making/verifying password changes on the user management page, and finally verifying that the delete account feature works on the user management page.

Table 1: Test Cases for Debt Tracker Chart

|  |  |  |  |
| --- | --- | --- | --- |
| **Loan Name** | **Current Balance** | **Minimum Payment** | **Interest Rate** |
| Credit Card | $5,000.00 | $450.00 | 4.500% |
| Car | $15,000.00 | $225.00 | 3.125% |
| Student Loan | $20,000.00 | $300.00 | 1.500% |

When using the test values in the chart above, the avalanche method will apply an extra payment toward the credit card debt first, then the car payment, and then finally the student loan (this is based on interest rate from highest to lowest). Once a debt is paid, its minimum payment amount is considered an extra payment going forward to the remaining debts. To observe this in action, if viewing the principle amounts in a line chart over time, each debt should decrease at a steady slope until a debt with the highest interest rate is paid off. At that point, the remaining debt with the highest interest rate will receive an extra monthly payment applied to it, so the principle should decrease at a faster slope, and so on until all the debts are paid. This can be observed in the results (found on the following page) for the test values given above, using the minimum monthly payment total of $975.00 as the payment pool.

Chart 1: Debt Payoff Calculator Results



As can be observed in the chart, once the credit card is paid off around month 11, then the extra payment is applied toward the car loan, which causes that to decrease at a faster amount while the student loan continues to decrease at a steady amount. Once the car loan is paid off, the extra amount is applied to the student loan, which then decreases at a faster amount.

To test the user management page, we first created a user account “delete@email.com“ with the password “asdf”. On the user management page, the password is then changed to “fdsa”. Upon trying to login to the account with the password “asdf” an error dialog box pops up informing the user that the username/password is incorrect. We then try again with the new password “fdsa” and we are successfully able to login. This sequence can be seen in the figures below:

Figure 1: Change Password Screen

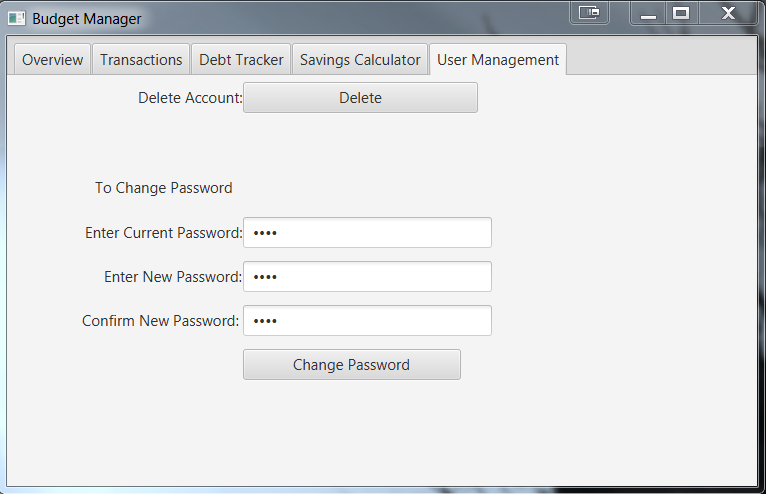
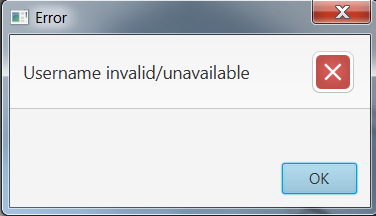


Figure 2: Username Invalid (Wrong Password After Change)



To test the delete account feature, we first seeded the account with data that displays immediately upon logging in. We then use the delete account option on the user management tab, and then try to login to the account that we previously seeded with data. When the system finds that there is no account of that username (since we just deleted it), a new account is created—without the data that we had previously seeded. This sequence can be found in the figures below:

Figure 3: Seeding Account with Sample Data

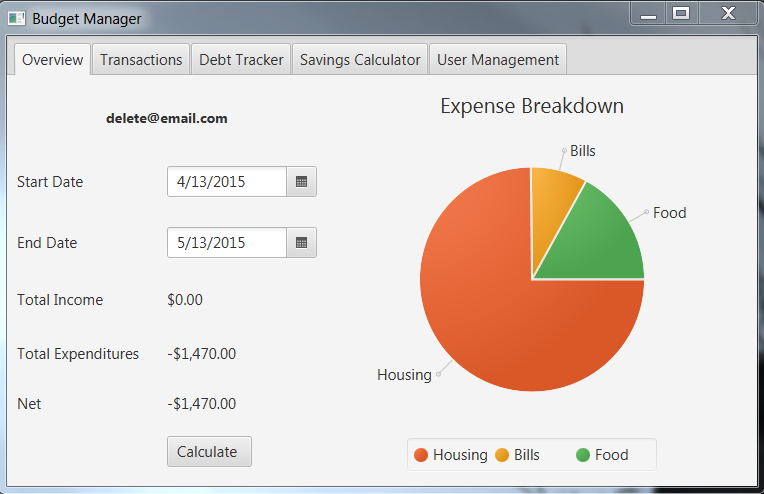


Figure 4: Deleting Account

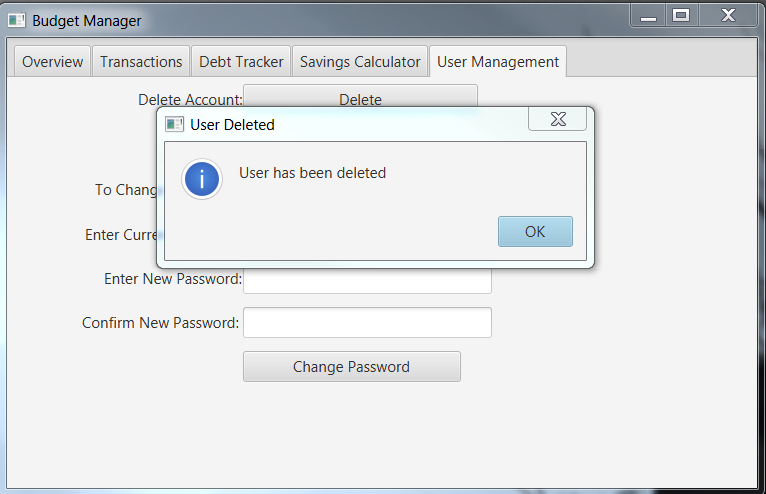
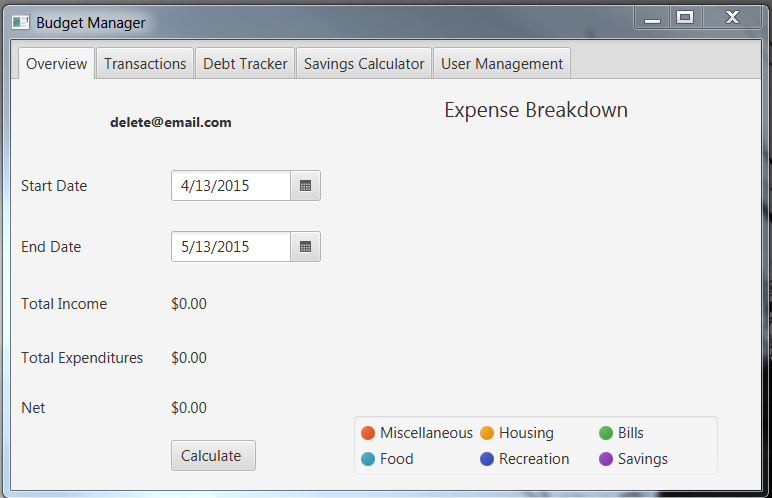


Figure 5: Trying to Access Deleted Account (Creating New Account in the Process)



As you can see from the series of screenshots in this section, all of the testing was successful. The debt payoff calculator worked as expected and the change password and delete account options on the user management page worked as well.