Cemetery Management System

Samuel Kearns - Project Manager (skearns4)

Brittany Regrut - Quality Assurance / Software Developer (bnr12)

Louis Seefeld - Software Developer (lfs13)

Emily Crabb - Quality Assurance / Documentation (ejc44)

Evan Alexander - Software Developer (ea914)

Due date: 24 NOVEMBER 2015

CS 1530 - FINAL DELIVERABLE

Sprint 5 Accomplishments

During this sprint, we completed two new user stories relating to editing existing entries in our database and printing selected entries from our database into a text file, as is described in the sections on user stories below. We also made some additional functional and cosmetic changes to our user interface. We added searching by section, plot, and grave in addition to searching by name, date of death, and interment number because of how we anticipate our customer using the program. The section, plot, and grave reflect where the person is actually buried in the cemetery, so it seems reasonable the user might want to search by these fields. The other search fields are unrelated to the physical location. We then created additional unit tests in our MainTest.java file for searching using these fields. We also added boxes to our GUI to allow the user to select and deselect entries returned by a search. These entries can then be viewed and edited or printed to a text file. On a related note, we fully implemented the view entry pop-up window, which just displayed “Test” at the end of the previous sprint. It now displays all the known information about an entry stored in the database and allows the user to make changes to the entry. There have also been some minor changes to the locations of buttons and text fields in the GUI.

We met with the customer near the end of this sprint and demonstrated our final product. Both Lonnie, the cemetery manager and the primary intended user for our program, and Rob, the synagogue director and the customer who commissioned our program, were pleased with the final product. They liked how we included real data in the database and requested both documentation and links to the source code after we have finalized the product at the end of this sprint. Additionally, they were happy that one of our team members intends to continue this project next semester as part of her capstone project, as it will give them more opportunities to request additional functionality.

During this sprint, we used the same methods of communication as during our previous sprints. We mainly communicated via Slack, with a few short meetings after class to divide the work and discuss any problems. Unlike in the previous sprint, we also had a longer in-person meeting on the weekend before the program was due to discuss our final product and help write the user’s guide. The user’s guide describes how to install the program and what functionality is available to the user. Writing it required the various team members to each include information about the portions of the program they wrote and tested.

*Tradeoffs*

As this was the final sprint, we were forced to make some tradeoffs to produce a project acceptable to the customer that was delivered on time. Due to limits to the schedule and the costs/resources, we had to also limit our scope and quality. The main functionality we cut was the proposed map interface that would have allowed a user to click on a plot in a cemetery and view information about it. However, we did not have enough time or resources to implement this functionality. The time/schedule was limited because the project had a fixed due date. The resources/costs were limited because we were never provided with a sufficiently high-resolution map of the cemetery by the customer. Another tradeoff we made was not integrating the payment management system with our program as originally planned. We thus limited our scope due to schedule constraints. We also made some tradeoffs related to the quality of our code. Time constraints limited the amount of error-catching for invalid user inputs we implemented. Our program is thus less robust than it could have been given a more flexible schedule. In sum, because our schedule and costs were fixed, our tradeoffs all related to decreasing the scope or quality of our program.

User Stories Completed

Link to the repository:<https://github.com/skearns4/Synagogue-Cemetery>

In this sprint, we completed two new user stories. As listed in the backlog included in our previous deliverable, the completed user stories are:

|  |  |  |
| --- | --- | --- |
| Rank | User Story | Difficulty Estimate |
| 1 | As a user, I want to be able to print search results and individual user entries so that I can easily share the information with others. | 4 |
| 2 | As an administrator, I want to be able to edit existing entries so that I can correct any errors in the information stored in the database. | 4 |

The print functionality was implemented button that allows the user to print information on all selected entries to a text file. The editing functionality was implemented within the previous view entry pop-up window. The user can now edit entries directly in this window and then save the changes.

The user stories we did not complete are:

|  |  |  |
| --- | --- | --- |
| Rank | User Story | Difficulty Estimate |
| 1 | As a user, I want to be able to view a map of the cemetery, including number of open graves per plot so that I can more easily show clients the layout of the cemetery. | 8 |
| 2 | As a user, I want to be able to able to see the status of plots that are in close proximity to a current plot so that I can easily determine whether clients can purchase adjacent plots. | 8 |
| 3 | As a user, I want to be able to click on the map of the cemetery so that I can easily look up information for a specific plot. | 8 |
| 4 | As an administrator, I want to be able to manage payments for the plots so that it is easier to coordinate plot and payment information. | 16 |

Our justification for completing certain user stories and not completing other ones is given in the next section.

Justification for Choosing These User Stories

We chose to implement the user stories relating to printing entries and editing existing entries because this functionality is highly valued by the user. The program the synagogue currently uses to manage its cemetery information allows both editing and printing. Therefore, implementing this functionality was a step towards our goal of making our program at least as useful as the existing program. In fact, our final program completely meets this goal, as it includes all the functionality of the program currently used by the synagogue. However, our program is also more user-friendly and will be simpler for future programmers to edit. With our program, the user interacts with a simple GUI with buttons and text fields, making it easy to search, edit, and print. It is also written in Java, a common programming language. The previous program only ran on a Microsoft DOS computer, making it both somewhat obsolete and fairly difficult to use. As such, our program is a more user-friendly and maintainable replacement that still encompasses all of the old functionality.

We did not implement the user stories relating to the map user interface or managing payments for several reasons. For one, this functionality is not nearly as critical to the program and is not valued as highly by the user. The current program which has been used for years does not have a map interface, and payments are managed in a completely different system. The user stories relating to the map interface and payment integration were therefore desired as useful features but were not necessary to making the program functional, as the user prioritized more critical features like searching and editing. Additionally, the users never gave us a high-definition map with which to create the map interface. Finally, our difficulty estimates for the map and payment user stories were fairly high (8 or 16). We therefore anticipated that we might need over one full sprint to really get a useful map interface or payment system working. As this was the final sprint, we decided we should not sacrifice any of the more critical functionality (printing and editing) in order to implement a map interface or payment system that might not even be completed.

Customer Feedback

We met with the customers the day before the end of this sprint to show them our completed program. Lonnie, the cemetery manager and the customer with whom we had the most interaction throughout the project, was excited to see that the program was interacting with actual data. He was concerned that our database did not have columns representing the contact address of the deceased family, the rabbi that held the funeral, information about payments, and room for notes, but we assured him that these fields can easily be added to our database by a future software developer.  
 We also met with Rob, the director of the synagogue and the person who initiated this project, for the first time since our initial meeting during the first sprint. He had not seen the software since that first meeting when it was just a walking skeleton, so we were interested in his response. He generally seemed happy with our product. He did bring up the proposed map integration but understood why we were not able to implement it within our fixed schedule.  
 Lonnie, Rob, and we all agreed that our product is essentially an updated version of their old DOS system. This was the realistic goal for the project we agreed on during our first meeting with the customers. Lonnie and Rob were therefore pleased with the product and asked for documentation and a link to the source code, which can easily be provided after we submit the final versions for this last sprint.  
 Additionally, one of our team members, Brittany Regrut, will continue working on this project next semester as part of her capstone project. Rob and Lonnie were excited to hear this, as it will give them the opportunity to request additional functionality and improvements that we did not complete during this term or that become necessary once they start using the software.

Defects and Limits on Functionality

One defect we discovered was that the printing function did not work on Windows computers. The original developer of the print function used a Mac computer. On his computer, the entries were printed to the file when the print function was selected, which was the expected behavior. However, when another team member tried to print on a Windows computer, the observed behavior was that a blank file was created. She then used print line debugging to discover that the problem was with the format of the file’s name. The name of the file to which the entries are printed contains the current time formatted into a string. The previous format used a colon, which is not a valid character to have in a file name on a Windows computer. Therefore, to correct the defect, the colon was replaced with an underscore, creating a valid format for the file name.

Another defect we discovered was that editing an entry did not work if the date deceased entry was null, which is true of most of the entries in our database. The expected behavior when we tried to save changes in the view entries pop-up window was that the changes would be saved to the database without any sort of error message. However, when the date deceased field of the entry was null and no new date was entered, the observed behavior when one tried to save any changes was that the changes were not saved to the database and an error message was outputted. The defect was related to the fact that the SQL statement updating the entry in the database attempted to parse the date deceased entry but could not do so because the entry was empty. Therefore, we corrected the defect by adding an if-else statement that caught this case and executed an alternate SQL statement with a null entry for the date deceased field.