Employee Scheduling Application

Group: Eric Bloom, Kyle Miller, Carl Baumbach, and Lincoln Powell

Current Trends and Projects in Computer Science

CMSC 495 6381 (2165)

Instructor: Clarence Huff

July 10, 2016

Table of Contents

Overview

This paper is to serve as our group's documentation discussing various topics according to the Software Development Life Cycle (SDLC) of our chosen project, presented in detail in the Project Plan, p. 2. Topics discussed will flow from our planning, design, and testing phases throughout this course, presenting commentary on our approach, progress, and deviations to set deliverables. Our chosen project must adhere to the final project description outlined in the CMSC 495 Teaching Guide; this project must use a high-level programming language to solve a specific problem or issue in computer science. Moreover, this project must satisfy several critical learning goals and challenge students with creating a complex program in the areas such as software design and architecture, systems and application security, mobile applications, database design and implementation, concurrent programming, signal processing, or algorithm performance optimization.

Group Member Roles

Eric Bloom: Project manager
Kyle Miller: Back-end programmer (database creation and data manipulation)
Carl Baumbach: Front-end programmer (GUI and output)
Lincoln Powell: Security programmer; testing/debugging/documentation

Individual Contributions

Eric Bloom: Offered support role when assistance was requested. Helped Carl with GUI design. Completed Progress Assessment starting in Week 3 to update instructor on each group member's status against week's Deliverable Milestones.

Kyle Miller: Designed back-end classes: AddressInfo, Alert, AlertListener, DailyAvailability, DataBase, PhoneInfo, PhoneNumber, PhoneType, Schedule, Scheduler, ShiftAvailability, WeekSetup, and WeeklyAvailability. Updated User class as needs required with fields and methods for timesheet generation and storing data pertinent to each user.

Carl Baumbach: Proposed Employee Scheduling Application idea to group. Designed RowHeaders class for database GUI. Updated EmployeeFrame and ManagerFrame to facilitate look and feel of JTable.

Lincoln Powell: Designed LoginDialog (along with EmployeeFrame and ManagerFrame, although bare; these classes were later greatly expanded by Carl on the front-end), User, Employee, Manager, and DataManager classes. Created SerialTest to build initial database for testing and then later expanded functionality in CreateUsers to allow team to build databases for testing purposes and to determine how the program reacts to various amount of users. Handled all documentation deliverables for assignment (along with Eric's help, edit, and final approval).

Project Plan

New Health Central Hospital currently has a dilemma tracking employees' hours worked and spends too much time determining who is supposed to work on a given shift, taking away essential time of managers and supervisors from other necessary duties. The hospital used to track time sheets using pencil and paper methods but have now turned towards advancing their procedures through technology. The hospital seeks an employee scheduling application that generates a schedule and prevents conflicts of over- and under-staffing. The hospital wants the application to monitor and alert a user, preferably a manager, when:

- an employee works over 10 hours per shift
- an employee works over 40 hours per work week (Sunday through Saturday)
- only one head surgeon per shift for 3 shifts per day (Day, Swing, and Mid shifts)

Employees can login to the application to:

- view their individual week timesheet
- review a collective week timesheet listing all employees

Managers (administrators) will have the unique abilities to:

- edit the schedule
- reset employee passwords to the system
- create, modify, and delete user accounts

Stakeholder Intent for Project

This project is supposed to allow managers at a departmental or company level to track and generate an employee schedule upon manager input of availability to work, given that employee's shift hour preferences, and prevents conflicts of over- and under-staffing.

New Health Central Hospital will be the client for this project. The hospital board of directors have challenged our group of programmers to tailor this project to alert managers when employees have worked over 10 hours per shift, over 40 hours per work week, and when over- or under-staffing head surgeons per shift (Day, Swing, and Mid shifts). Moreover, the hospital wants the program to require login authentication features.

Deliverable Milestones for Project

Week 2: Start coding of GUI, primarily login authentication features. Discussion 2 and Project Plan is due this week. Begin Test Plan and Peer Review.

Lincoln will work on the login authentication features while Kyle and Carl test functionality and plan implementation of front/back-end design for database. Eric will supervise all tasks.

Week 3: Discussion 3 and GUI finished by end of week. Start of functionality coding since most/some of the GUI should be finished by now. Test Plan and Peer Review should be finished and turned in; begin Analysis and Design.

Lincoln will continue to work on password reset and challenge question functionality. Kyle and Carl will continue front/back-end programming. Program code syncing will be completed to begin testing incompatibilities of variable naming and etc. Eric will supervise all tasks and report to instructor on each member's progress/performance.

Week 4: Login should be fully coded by now. Skeleton of data input should be done or reviews should be started. Discussion 4 and Analysis and Design are due; Phase 1 source and Peer Review Phase 1 are started.

Kyle and Carl will work on skeleton of data input. Lincoln will continue testing security features. Eric will supervise all tasks and report to instructor on each member's progress/performance.

Week 5: A data file should be finished for full testing. Discussion 5, Phase 1 source and Peer Review are due; Phase 2 source is started.

Lincoln, Kyle, and Carl will work on testing and debugging. Eric will supervise all tasks and report to instructor on each member's progress/performance.

Week 6: Debugging and testing. Discussion 6 and Phase 2 source are due; Phase 3 is started.

Lincoln, Kyle, and Carl will work on testing and debugging. Eric will supervise all tasks and report to instructor on each member's progress/performance.

Week 7: Coding should be finished and project should execute without errors. Debugging and testing will continue and be finalized at the end of the week. Discussion 7 and Phase 3 are due; Peer Review – Final is started.

Lincoln, Kyle, and Carl will work on testing and debugging. Eric will supervise all tasks and report to instructor on each member's progress/performance.

Week 8: Discussion 8 and Final paper due at the end of the week. Peer Review final is due including source files, data files, and documentation.

Requirements Specification

Software Requirements:

- Microsoft Windows XP or greater, Linux, or Apple OSX
- Java JRE 1.8

Hardware Requirements:

- 30 KB available hard drive space
- Server capability recommended

Development Software:

- Java JDK 1.8.0_65
- Eclipse and NetBeans IDE 8.1

User's Guide

Logging into the Employee Scheduling Application

Welcome to your new timesheet automation experience! To begin using the Employee Scheduling Application, first login using your proper credentials (NOTE: Username is not case-sensitive, meaning myuser01 is the same as Myuser01, MyUser01, and MYUSER01; all other deviations to how the username is typed are acceptable).

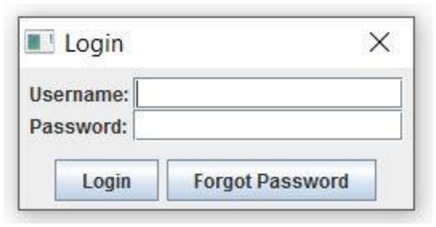
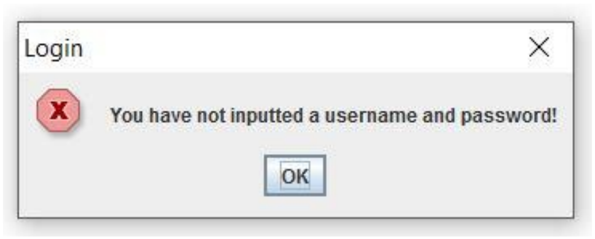


Figure 1. Login dialog window.

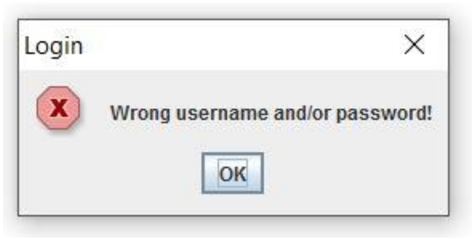
If both Username and Password fields are blank and you attempt to press Login, an error will display:


Login

You have not inputted a username and password!

Figure 2. Error dialog when user attempts to login without inputting a username and password.

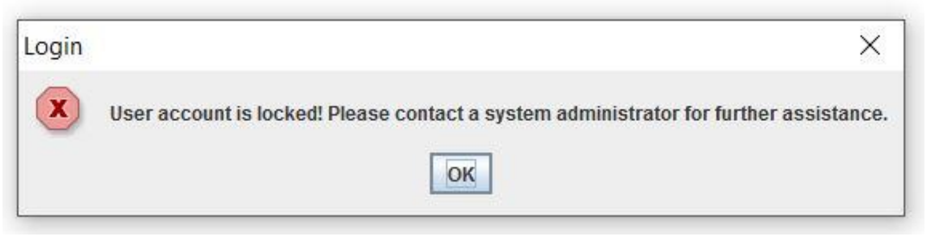
If the username and/or password are incorrect, an error will display upon pressing Login:


Login

Wrong username and/or password!

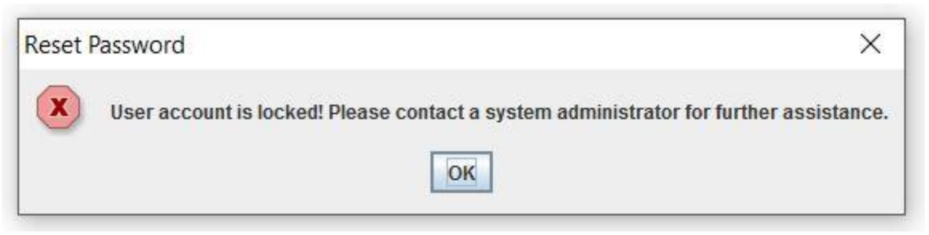
Figure 3. Error dialog when user's inputted credentials are incorrect.

If you attempt incorrect authentication attempts three times, your account will be locked and will require an administrator to unlock your account. Note, if your account is locked, you will not be able to utilize the Forgot Password feature!


Login

User account is locked! Please contact a system administrator for further assistance.

Figure 4. Error dialog when user presses Login after locking account.

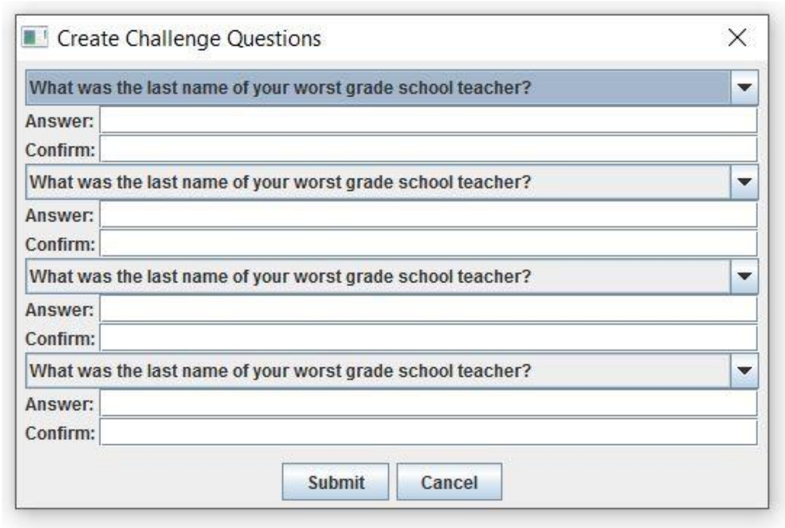

Reset Password

User account is locked! Please contact a system administrator for further assistance.

Figure 5. Error dialog when user presses Forgot Password after locking account.

Upon successful login, if you are a new user, you will be prompted to create four challenge questions. Completion of this task is required in order to proceed in the application! If you decide to press Cancel or close the application, you will still be required to complete this step. In order to complete setting up your challenge successfully, ensure:

1. you are not selecting duplicate questions
2. your answers are at least 5 characters long
3. your answers for each selected question match


Create Challenge Questions

What was the last name of your worst grade school teacher?

Answer:

Confirm:

What was the last name of your worst grade school teacher?

Answer:

Confirm:

What was the last name of your worst grade school teacher?

Answer:

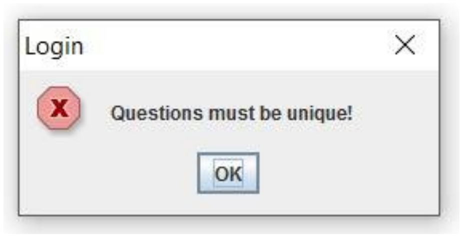
Confirm:

What was the last name of your worst grade school teacher?

Answer:

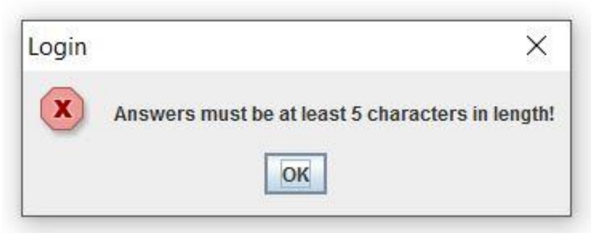
Confirm:

Figure 6. Challenge question creation dialog window.


Login

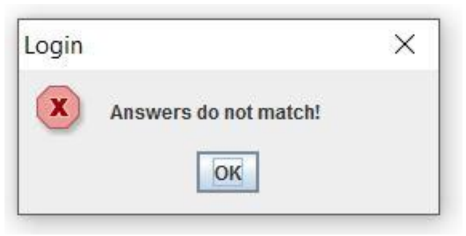
Questions must be unique!

Figure 7. Error dialog when user choose duplicate question choices.


Login

Answers must be at least 5 characters in length!

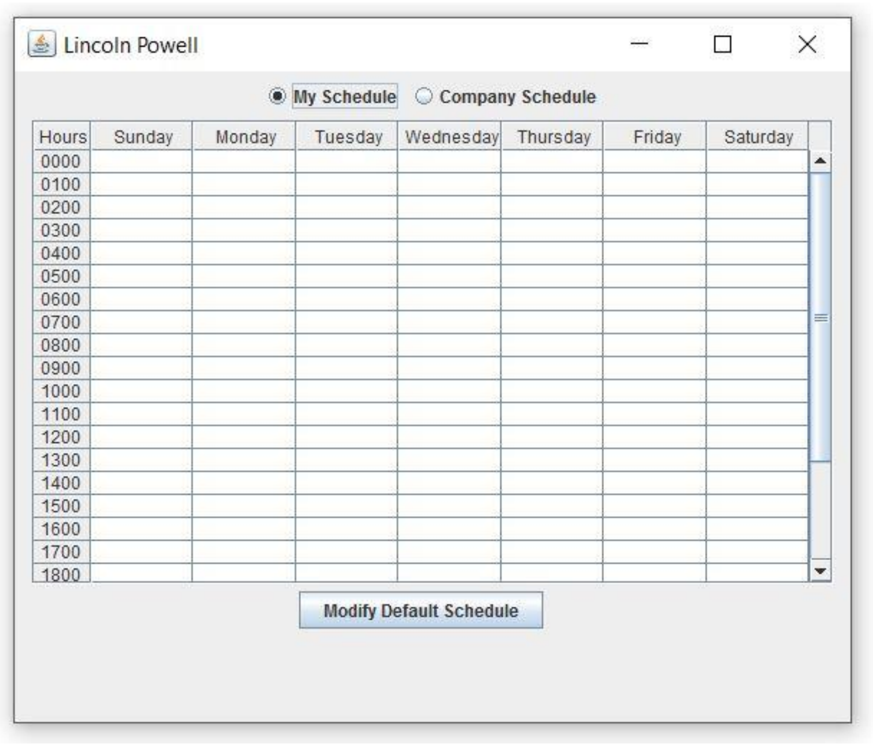
Figure 8. Error dialog when user inputs an answer that is less than 5 characters long.


Login

Answers do not match!

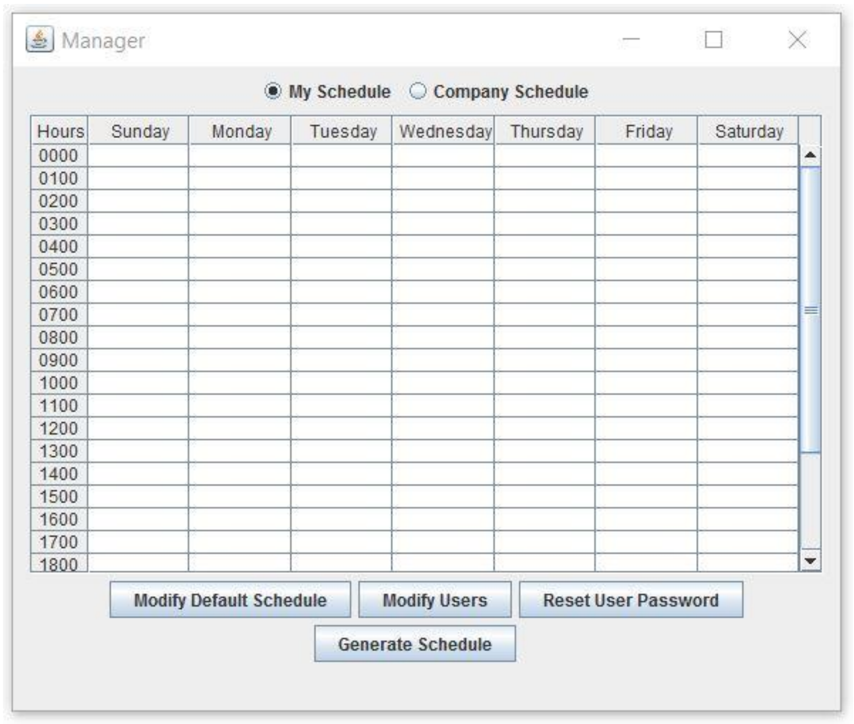
Figure 9. Error dialog when user inputs non-matching answers for a chosen question.

Upon successful challenge question creation, you will finally be displayed a welcome message and then the applicable window depending on your administrative rights.



Hours	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
0000							
0100							
0200							
0300							
0400							
0500							
0600							
0700							
0800							
0900							
1000							
1100							
1200							
1300							
1400							
1500							
1600							
1700							
1800							

Figure 10. Employee window view.

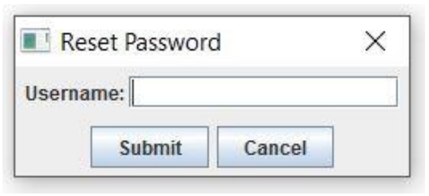


Hours	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
0000							
0100							
0200							
0300							
0400							
0500							
0600							
0700							
0800							
0900							
1000							
1100							
1200							
1300							
1400							
1500							
1600							
1700							
1800							

Figure 11. Manager window view.

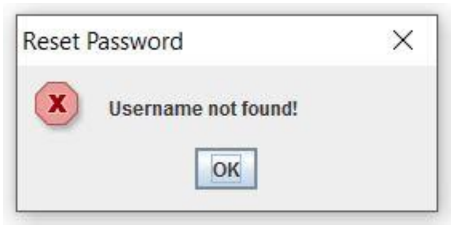
Using the Forgot Password feature

If you wish to use the Forgot Password feature, you will first be displayed a dialog window to authenticate your username. If the inputted username does not exist, you will be displayed an error dialog.


Reset Password

Username:

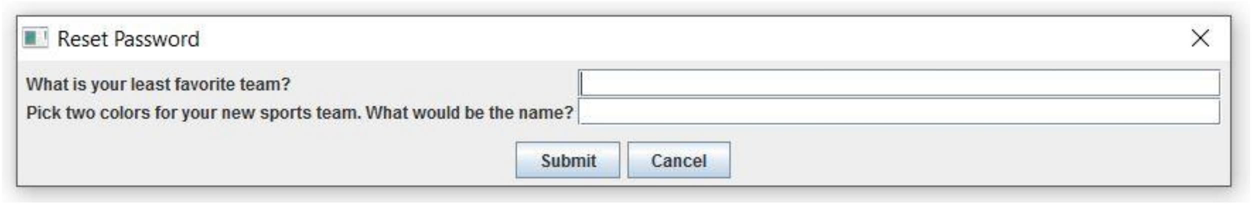
Figure 12. Username verification dialog upon user presses Forgot Password on login dialog window.


Reset Password

Username not found!

Figure 13. Error dialog when inputted username does not exist.

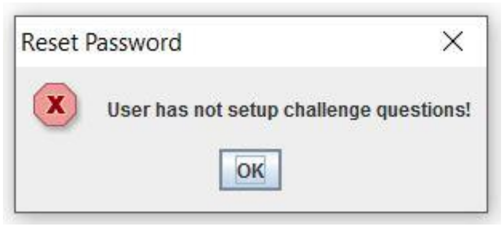
Upon successful verification, you will be displayed two out of the four challenge questions created. If you did not setup challenge questions, you will be displayed an error message. If one or more inputted answers are incorrect, you will be displayed an error message.


Reset Password

What is your least favorite team?

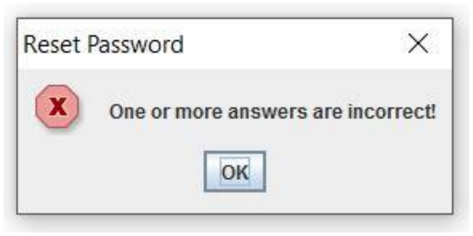
Pick two colors for your new sports team. What would be the name?

Figure 14. Dialog consisting of two out of the four challenge questions created for second-layer authentication.


Reset Password

User has not setup challenge questions!

Figure 15. Error dialog when user has not setup challenge questions and attempts to use Forgot Password feature.


Reset Password

One or more answers are incorrect!

Figure 16. Error dialog when one or more answers to selected challenge questions are inputted by user and submitted.

Upon successful dual-layer authentication, you will be allowed to change your password. Ensure your password is at least five characters long and that your input matches both the Password and Confirm fields; if either is not satisfied, the applicable error message will be displayed.

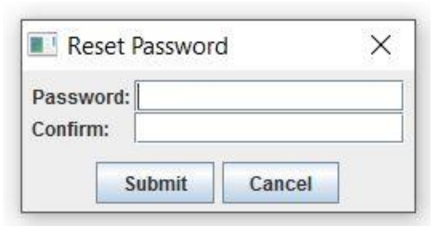
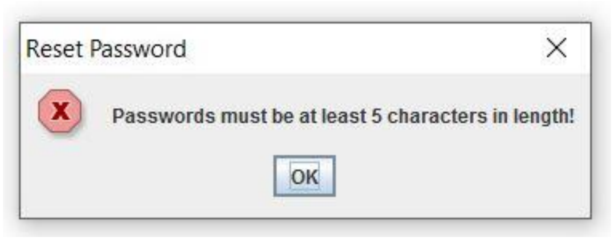
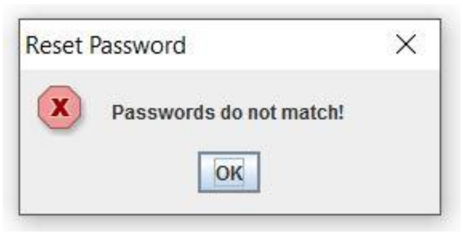


Figure 17. Reset password dialog window.


Reset Password

Passwords must be at least 5 characters in length!

Figure 18. Error dialog when inputted password is less than five characters long.


Reset Password

Passwords do not match!

Figure 19. Error dialog when inputted Password and Confirm fields do not match.

Employee View Features

To be expanded on future iterations of this document.

Manager View Features

To be expanded on future iterations of this document.

Test Plan and Results

Test Case	**Description**	**Pass/Fail?**
1	Does Login dialog process user’s valid credentials, proceeding the user to the next appropriate GUI element?	**Pass**
2	Does Login dialog catch incorrect credentials?	**Pass**
3	Does the Reset Password feature prompt 2 out of the 4 created challenge questions and allow a user to change their password upon successful validation?	**Pass**
4	Does the Reset Password option prevent new users from using the feature until challenge questions are created?	**Pass**
5	Does the challenge question creation dialog work, validating input and saving entries to the user’s record?	**Pass**
6	Does the program display the appropriate frame given user’s administrative role?	**Pass**
7	Does the program securely and safely serialize and deserialized database data file?	**Pass**
8	Does the program lock users when attempting to input incorrect passwords upon the third authentication attempt?	**Pass**
9	Does the program prevent locked users from using the Forgot Password feature?	**Pass**
10	Does the proper frame display for employees and managers?	**Pass**
11	Can employees/managers view their timesheets?	**Fail**
12	Can employees/managers modify default schedule?	**Fail**

13	Can managers modify users?	**Fail**
14	Can managers reset user passwords?	**Fail**
15	Can managers generate schedules?	**Fail**

Design and Alternate Designs

In order to complete our project given our analysis on how to perform the necessary functions desired by our stakeholders, our group decided to split our workload into three separate levels: (1) security authentication level, (2) front-end GUI level (end user experience), and (3) back-end database level. Using the attached Master UML to facilitate an introductory primer into our design and flow, the program starts off using the LoginDialog.java, then goes into the EmployeeFrame.java or ManagerFrame.java based if the user has administrative rights (e.g. isAdmin = “true”). From the start of the program, on the backend, DataManager.java deserializes the userdata.txt, if the file exists, then decrypts the information for use. If the program is closed via the close button, the data is encrypted then serialized into userdata.txt. The encryption algorithm chosen, due to time constraints from New Health Central Hospital, was the XOR encryption algorithm; limitations and methodology has been explored in our analysis. The rest of the classes pertain the database table creation, user elements, alerts, and GUI-related frames and buttons. In this section, we will discuss our present design more thoroughly with research on our methodology and possible alternate designs available.

In completing the deliverables for New Health Central Hospital, our group had to determine the functional and technical requirements for the employee scheduling application. The hospital wishes to employ security, usability, and affordability with advancing their time tracking using technological means. That means that the application should be able to function fully without errors on their systems and database elements should persist beyond the scope (lifetime) of the application. Additionally, software tools should be low-cost or free; our group based the application using Java while using freeware Java Development Kit tools such as Eclipse and NetBeans. The hospital will only be required to have computer systems that have a 1.8 (or greater) Java Runtime Environment regardless of operating system or other variables. Finally, to keep the project manageable in terms of costs, strict adherence to Eric’s Deliverable Milestones, p. 2 – 3, will be followed by all team members.

First, our group, after discussions, settled for Java’s serialization to preserve database object data. According to JusForTechies (2011), “In general, serialization is used when we want the object to exist beyond the lifetime of the JVM […] To persist data for future use […] To send data to a remote computer using such client/server technologies.” Moreover, serialization helps our efforts in fine tuning the code when developing and testing the application, as refactoring does not impact the process. Gaddis (2013, p. 741) highlights during the process when an object is serialized, “[the object] is converted into a series of bytes that contain the object’s data […] the resulting set of bytes can be saved to a file for later retrieval.” The only issues present is that serialization is not secure; to circumvent this limitation, encryption using a basic XOR algorithm was used to obscure the data. This algorithm is easy to implement but is also simple to break if the key is revealed or is not secure. Given the time constraints put on us by New Health Central Hospital, this algorithm will be explored and improved over time to add more complex security methods (in hypothetical future releases) versus basic concepts used from an academic standpoint. Data input and output will be based on the applicable field held in the GUI and the user’s administrative rights.

Next, simplicity has to be implemented in the front-end, end-user experience to allow a low cost of ownership for the hospital in terms of training and change management. Functionality of the GUI elements should be easy to understand; therefore, Carl ensured that the frames would have low-end functionality to avoid hidden shortcuts that a common user would not know unless training was implemented. The same methodology was also followed in the back-end database level of the application.

Keeping in sync with the rest of the application, we initially had an idea to employ Java SQL capabilities but resorted to keeping everything in Java to prevent confusion, as this is an area that has not been explored by each team member of the group. Using tables, accessor/mutator methods, and other basic/intermediate Java concepts, we created a database that holds employee data and manipulates data accordingly.

Finally, the application should be usable at any time regardless of user volume. Testing will have to be conducted on running instances of the application and how data manipulation impacts the userdata.txt file, which holds all object data for the application's use. The application should follow a path of user authentication to applicable timesheets based on the user's administrative rights. If the user is a manager, administrator functions should be present along with the ability to manipulate subordinate data.

Alternative designs that were introduced during the planning and design phases of our project mirrored our current design with minor deviations; however, utilized a web-based application instead of an application launched outside a browser. This design scheme would allow the same functionality of the current iteration of the program; however, would allow our stakeholders the ability to have a more convenient, modern, and portable application. Our stakeholders would be able to make edits and changes to the timesheet on-the-go, regardless if the user is in the office or outside the workspace.

Development History

Week 2: Plans and Specifications completed, encompassing Requirements and System Specification. Project intent from our stakeholder was specified to the instructor along with applicable scenarios the program will handle. Finally, our Deliverable Milestones were created and approved by Eric.

Lincoln submitted to group initial framework for the program, creating the DataManager, LoginDialog, EmployeeFrame, ManagerFrame, User, Employee, and Manager classes. Framework is to be expanded by Kyle and Carl within Week 3.

Week 3: Test Plan completed and Peer Review Test submitted to instructor.

Kyle requested clarity on managers searching by ID, name, or will a list of valid employees that are attached to the manager. Lincoln stated if the program lists all employees attached to that manager, then it should display each employee associated for that manager. Kyle will communicate to Carl on this design choice. Eric directed the team to focus on this design choice going forward and to report to him on testing for Week 4. Decision to have employees only view the timesheet instead of inputting availability to work was made to stay in spirit of the program’s purpose. This decision for managerial input versus automation is still being debated, given the hospital’s need for a technological and automated approach to solve their manpower and payroll issues with tracking employees’ time. Overall, the Test Plan and Program Flowchart encompasses all goals for the employee scheduling application. Lincoln will consolidate all UMLs into one master UML for better oversight of each class, its fields, its methods, and how they are associated with other classes.

Week 4: Project Analysis and Design completed.

Login fully implemented despite missing challenge question and user lockout functionality; delay due to GUI look and free issues arising from FlowLayout layout manager and how elements within the dialog aligned within the dialog window. Back-end design was further experimented by Kyle and Carl submitted modified EmployeeFrame and ManagerFrame for team meeting and Eric’s approval.

Week 5: Phase 1 Source and Peer Review Phase 1 completed.

Carl and Kyle have been working closely on gluing all segments of the employee scheduling application together albeit complications with generating schedules of employees. Kyle has written methods for generating the schedules and the group has been working on the complexity of the program when it comes to doing a hospital-wide schedule. Kyle has also completed work on a scheduler object that will attempt to generate four day blocks of work based on preferences of the employee – an area that we have wavered on for some time in regards to having the application either allow a manager to create a schedule or have the application automate a schedule then allowing a manager to further edit the schedule. Kyle states that if the availability is such that it can’t give three days off in a row, it will adjust accordingly. Lincoln has been continuing with the challenge question look and feel of the dialog; the implementation of the challenge question system works however layout design has caused some minor issues when it comes to the dialog. Lincoln states that using flow layout has worked for the login GUI dialog box but he will have to work with this layout some more to have each piece within the dialog appear professional for the challenge question dialog. Lincoln, at the moment, is avoiding the use of relative positioning, as it was discussed in a previous course that this layout should be avoided if possible. Kyle has requested an exemption for head surgeons, as dividing the week between the 3 individuals would lead to 56 hours per week of work per individual. This request is pending Eric’s approval at the present time although Kyle is continuing at the present course with this exemption.

In accordance to our project milestone for Week 5, in which the application would be ready for full testing after connecting all segments of the program together, our project team is on route to testing and debugging as planned minus minor delays to small segments of the application due to designing the automation of the scheduling system, application look/feel, and team member relocations/familial-related travel. Issues with class inheritance came to light when Kyle wanted to expand on the Employee and Manager class by adding unique fields and methods. Due to how the array list is implemented using the abstract User class, in which Employee and Manager inherits from, adding such unique fields and methods would cause issues when attempting to employ these additions from the User array list. Kyle requested that special methods for managers doing administrative actions that employees would not need should be held in the Manager class, a request that Lincoln agreed despite having disapproval, as the User class was to be used as a blueprint and such methods would not be used because the GUI for employees and managers would offer unique functionality between the two (e.g. employees could not employ password resets as there would be no button on their GUI). Another issue that came to light was how different data types would take to object serialization. Lincoln was unsure if integers, floats, and Booleans would appear to be readable in the serialized file. Lincoln suggested that all fields should be strings, if possible, or, converted into a string when serializing the data. Upon deserialization, another method would take said fields that were converted and return them back to their original data type. Despite the alternative solution, the group wished to stay on course with sticking with strings to reduce the complexity of the serialization/deserialization and encryption code.

Week 6: Phase 2 Source and Peer Review completed.

According to our project milestone for Week 6, last week’s delay has been corrected and our project is now on course with our milestone heading into Week 7. Lincoln’s issue regarding the look and feel of the challenge question creation GUI has been resolved using GridBagLayout instead of the default FlowLayout, which caused issues when trying to construct the components to align appropriately. Kyle’s changes to the User class has been integrated using serialization to correct issues with tracebacks for exemptions thrown when executing the program when User class changes attempted to serialize data that did not implement the Serializable interface. Kyle also added a DataBase class with a simple linear search method to test several of the methods he created in the source. Alerts and searching will still demand more substantial test data set; Lincoln's idea is to create a java class that builds a user database based off of randomization – Lincoln will work on this idea for Kyle's use during the early part of Week 7 to allow the group to slowly increase the data set to test how the program handles the load. Kyle will also need to come up with methods for managers to make changes to employee schedules, which may require a hospital-wide scheduling algorithm that will prompt group discussion on this topic, scheduled during Week 7. Error handling was discussed and will be thoroughly analyzed on implementation during Week 7. Role assignment for end users was discussed to check for conflicts with employee scheduling and what is displayed when an employee views his or her schedule.

Lincoln noticed a bug when a user has not setup challenge questions and attempts to reset their password via Reset Password. The program is supposed to direct the user back to the login GUI, which it does; however, the login GUI reads the input fields for username and password, prompting an error of incorrect username/password, then redirects the user back to the reset password dialog. Lincoln will continue debugging this issue during the week. A more robust test case section will be incorporated in the project paper using data from the project flowchart to communicate testing to stakeholders and project members. Moreover, the UML will need to be updated to communicate changes to classes. Finally, the User Guide will need to be revisited to add portions of the project that are completed with screen shots and guidance on functionality of sections.

Week 7: Phase 3 Peer Review submitted to instructor.

According to our project milestone for Week 7, debugging and testing are continuing as intended and issues with the timesheet functionality on the Employee/Manager frame views are being worked on by Carl, Kyle, and Lincoln. Lincoln corrected the bug associated to when a user has not setup challenge questions and attempts to reset their password via Forgot Password. The program now directs the user back to the login GUI; moving the block of code outside the for loop that iterates through the userList arraylist fixed the issue. Lincoln also finished building a class that creates a database based off of randomization upon a user inputting how large of a data set for testing he or she wishes. Finally, Lincoln updated the User's Guide to include all actions that are or can be performed for logging into the application and using the Forgot Password feature. Kyle completed methods for managers to change schedules and, after a lot of issues, he was able to have objects and methods that will auto-schedule every employee in the database. Kyle also made small changes to the User class. In order to schedule the entire hospital, the changes were necessary to the User class; a manager will now have to come up with a schedule template. The manager will do this in the database by using a method that allows them to add shifts to a weekly schedule. This is represented by a 3D array. No employees are scheduled at this point. The template only communicates that there is a shift to be filled on this day, at this time. Then, the manager can start the auto-schedule routine. This routine takes a look at the needs the manager provided and schedules all available employees to meet the demand. In order for this to work, four different roles for employees were created: orderly, nurse, doctor, and surgeon. Surgeons are set aside and given a set schedule of 8 hours Mon-Fri to meet the confines we set forward. The other employees are processed by their availability into all possible shifts that they can fill. The scheduler then fills each slot designated by the manager until complete. Each employee can only be scheduled to four, ten hour shifts. After that, they are no longer considered for a shift.

As of Week 7 going into Week 8, our project team is behind schedule to incorporate functionality into the Employee/Manager frames. Testing the application in these views are limited to Kyle testing via console; however, Eric has demanded expedited work into getting the frames functional by mid-week 8. Lincoln has an idea on how to flow the reset password feature in the manager frame using a similar approach to the login dialog. Carl will have to incorporate Kyle's work into how the GUIs for the Employee/Manager frames are supposed to function in regard to displaying a user's schedules, viewing the company schedule, modifying default schedule, modifying users, and generating schedules.

Week 8: Peer Review Final has been written and awaiting approval from team.

To be expanded...

Conclusions

This project challenged our group to tackle a complex scenario in the areas of software design and architecture, application security, encryption, serialization, database design and implementation, and algorithm design. Given the task to create an Employee Scheduling Application, our team focused on efficiency, simplicity, and effectiveness towards achieving our stakeholders' intent of the application. In our conclusion, we would like to discuss lessons learned during this eight-week adventure from planning, design, and to testing, design strengths and limitations, and suggestions for future improvement.

Lessons Learned

- Object serialization and serializable interface; serializing objects with members that are not serializable needs to be avoided in the future and needs to be corrected via Serializable interface or transient keyword
- Encryption using XOR algorithm
- GUI creation and layout managers such as GridBagLayout
- JTable implementation
- Modifying GUI element behaviors via window listeners

Design Strengths and Limitations

Strengths

- Focusing on building the program via application instead of web-based offers the program added security from exposure to internet threats
- Object serialization allows for easier IO to data file instead of needing to modify the data file when user data is added, deleted, or modified; restricts small changes to the program to enforce gradual updates which helped the team understand when the User class was being modified
- Easily implemented encryption using XOR algorithm

Limitations

- XOR algorithm, although easily implemented, is not a secure encryption method; if the key is compromised, encryption is lost
- Serialization decreases flexibility to change the User class's implementation once a data set userdata.txt file has been created (e.g. adding, modifying, or deleting fields or methods will cause deserialization to fail due to the objects not being in the same state to be read)
- Due to serialization flexibility issue, this option may not be the best approach for long-term data storage, as if an update to the program occurs, the existing data file will not work unless means where employed to save the state of the file for porting to newer versions

Possible Improvements

- Turning the application into a web-based application offers convenience to the end user and allows added portability to the software despite security risks
- Creating a randomly generated key that is securely stored on each serialization and as long as the text to be encrypted (e.g. when the program closes) offers bolstered security that is much harder to break (this concept is known as stream cipher); however, creating a truly random key results in an one-time pad which offers true security despite difficulties in implementation

References

Gaddis, T. (2013). Starting out with Java (5th ed.). Boston: Pearson.

JusForTechies. (2011). Serialization in Java. Retrieved from http://www.jusfortechies.com/java/core-java/serialization.php.

TutorialsPoint. (2016). Java - Serialization. Retrieved from http://www.tutorialspoint.com/java/java_serialization.htm.