TABLE OF CONTENTS

Project Description 2
Testing 3
Readme 4
Diagrams 8
Revisions and Challenges 12

Cole Rogers
David Stefek
Ian Johnson
GROUP 16

LOCKBOX

Our final project is a program that takes a user's files and encrypts them for safekeeping. Upon opening the program, the user is greeted with a login screen. If this is the first time the user has run the program, then they can navigate to the create account window. An account will consist of a username and a password, three security questions and answers for recovery in the event that the user's password is lost, as well as a home directory for the destination of serialized and unserialized fields. If the user has forgotten their password, they are also able to navigate, from the sign-in screen, to a "forgot password" section to access their account via security questions.

The main menu offers options such as locking or unlocking a file, searching for a file, resetting the account password and managing an account. In order to lock a file, a user is prompted to select a file from his computer through a pop-up window. The selected file

will be encrypted using an algorithm in the Jasypt library. To unlock a file, a user must simply select the file to be unlocked. Under the “manage account” menu option, the user will be able to change password, after re-entering their current password, or access security preferences. In security preferences the user is able to change any security question, security answer, or their designated home directory.

TESTING

The test cases and exceptions are covered in the following video:

<https://youtu.be/8S6BKSQJdaY>

README

The following is a list of instructions for the end user. It covers getting the program up and running and goes over the various functions.

FOR NEW USERS

1. To create an account, select “Create Account” from the menu at the bottom of the screen
2. Enter a username and password, then click “Create”
3. On the following screen, you will be prompted to enter three security questions and answers. These will be required of you in the event you forget your password and need to reset it.
4. Type your questions into the three appropriate text boxes along with their respective answers
5. Choose a file directory to store the files you would like to keep safe in LockBox and click “Submit”

FOR EXISTING USERS

IF YOU KNOW YOUR PASSWORD

1. Click “Sign In” from the menu at the bottom of the screen
2. Enter your username and password, then click “Sign in”

IF YOU FORGOT YOUR PASSWORD

1. Select “Forgot Password” from the menu across the bottom of the screen
2. Enter your username, then click “Submit”
3. On the next screen will be a set of prompts regarding the security questions you chose when creating your account
4. Type the questions in the three appropriate text boxes along with their respective answers and click “Submit”
5. Enter a new password, then click “Submit”

Upon logging in, you will be faced with the welcome screen in the main menu. At the top will be a menu bar with the following options:

Under “File,” you have the option to lock a file or unlock a file. Under “Manage Account,” you have the option to reset your password, change the file directory or change your security questions. Outlined below are the steps for each of the aforementioned options.

TO LOCK A FILE

1. Click “File” from the main menu bar
2. Select “Lock File” from the drop-down menu
3. In the pop-up window, select the file from your computer to be locked using the file browser and click “Open”
4. The file will be locked. Its serialized version will be created in your home directory.

TO UNLOCK A FILE

1. Click “File” from the main menu bar
2. Select “Unlock File” from the drop-down menu
3. In the pop-up window, select a file to be unlocked
4. Enter your password, then click “Submit”
5. The file will be unlocked

TO RESET YOUR PASSWORD

1. Select “Forgot Password” from the menu across the bottom of the screen
2. Enter your username, then click “Submit”
3. On the next screen, you will be asked to answer the three security questions you chose when creating your account
4. Enter the answers and click “Submit”
5. Enter a new password and click “Submit”

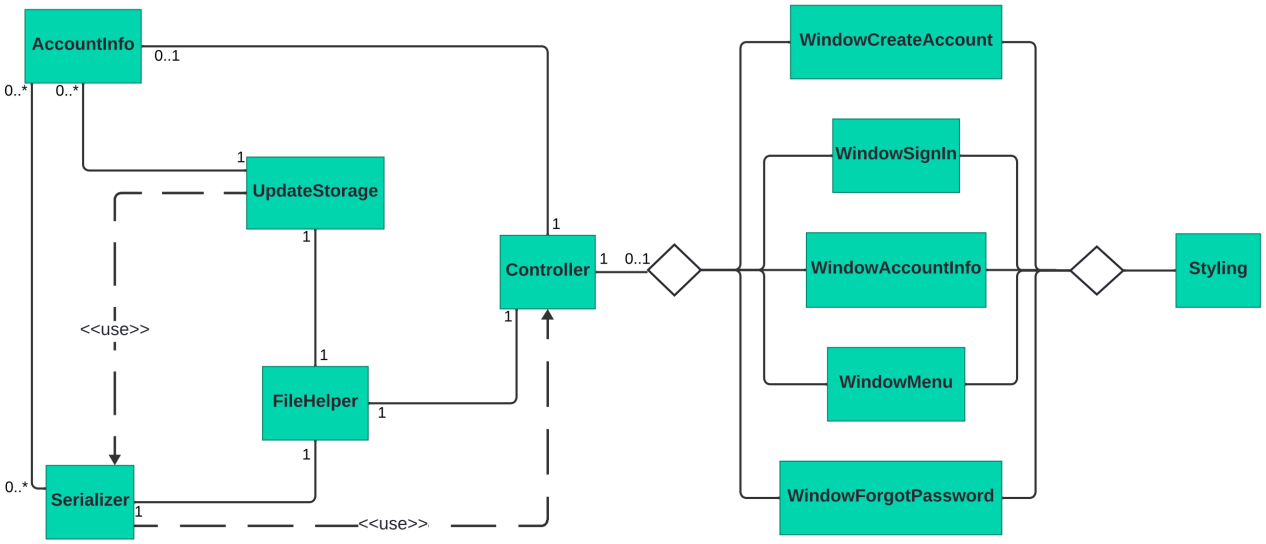
TO CHANGE YOUR SECURITY QUESTIONS

1. Click “Manage Account” from the main menu bar
2. Select “Security Preferences” from the drop-down menu
3. Type the new questions in the three appropriate text boxes along with their respective answers and click “Submit”

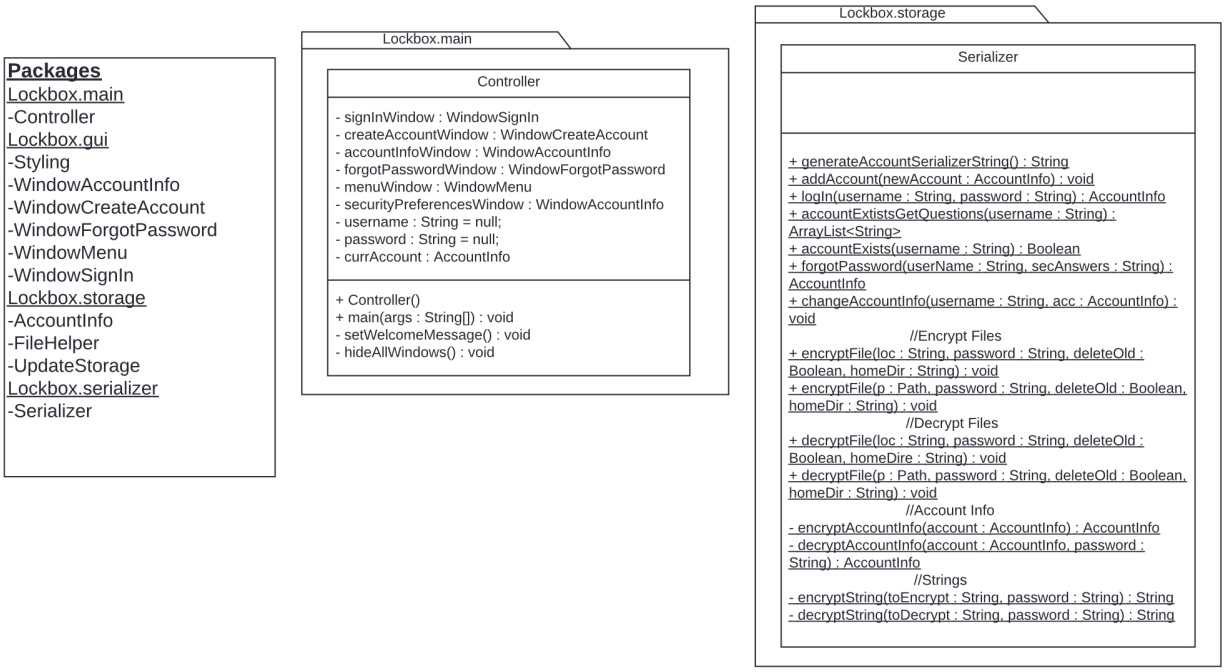
TO CHANGE THE FILE DIRECTORY

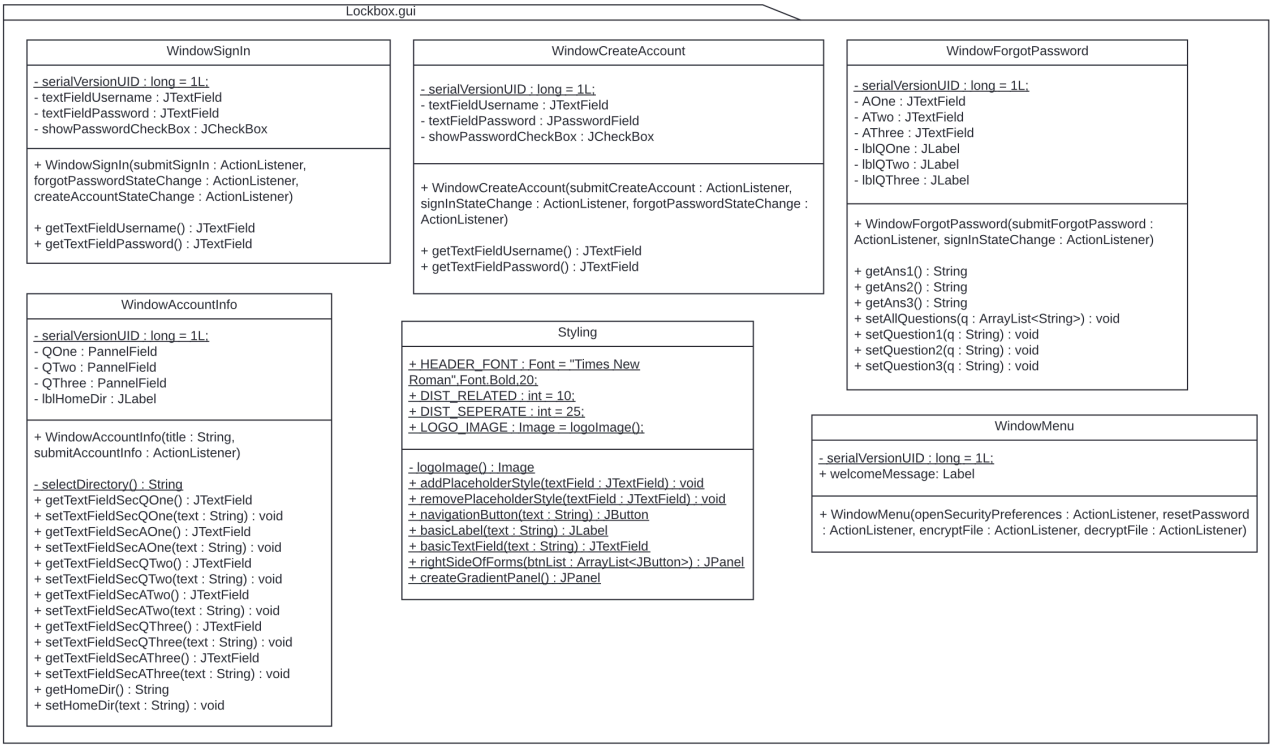
1. Click “Manage Account” from the main menu bar
2. Select “Security Preferences” from the drop-down menu
3. Click “Search” next to the shown file path and select a new directory for your files
4. Click “Submit”

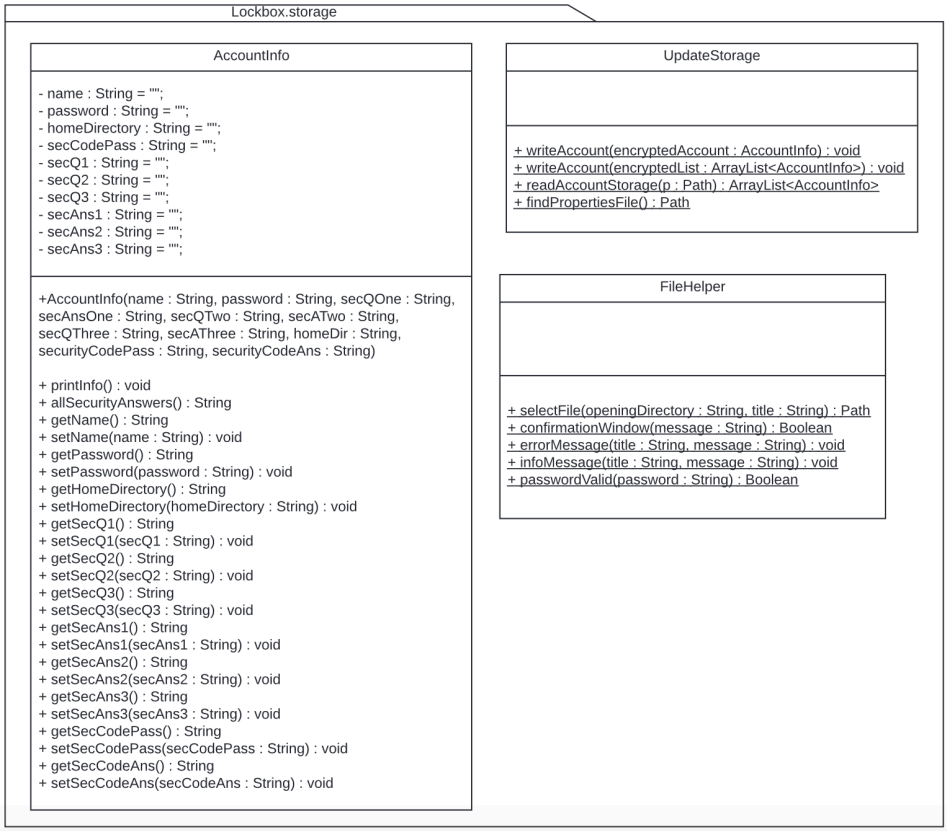
DIAGRAMS

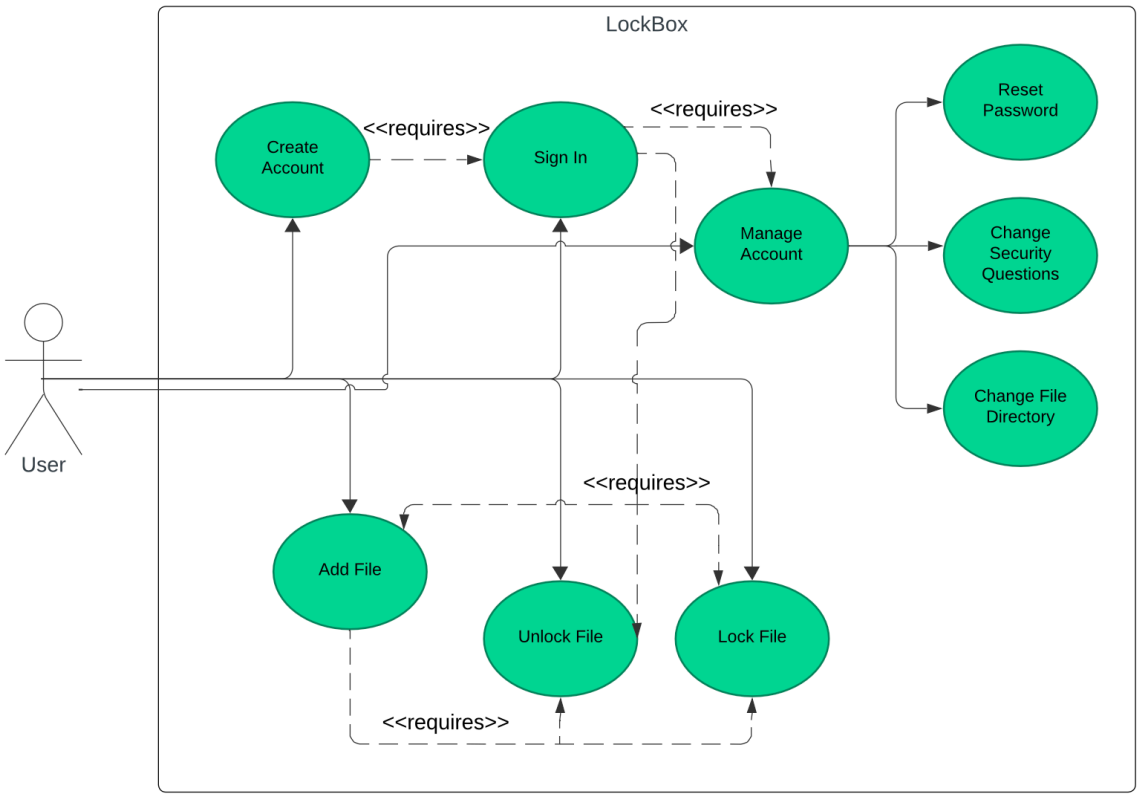


Class Diagram

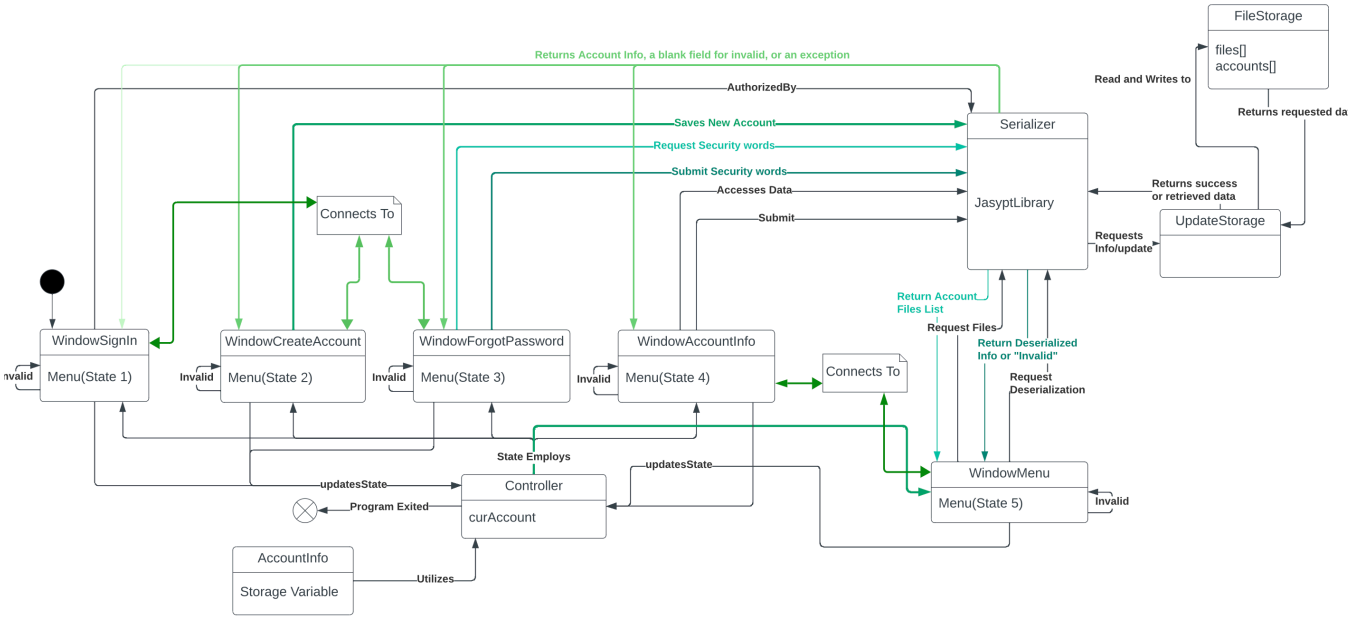



Lockbox.gui
WindowSignIn
- serialVersionUID : long = 1L;
- textFieldUsername : JTextField
- textFieldPassword : JTextField
- showPasswordCheckBox : JCheckBox
+ WindowSignIn(submitSignIn : ActionListener, forgotPasswordStateChange : ActionListener, createAccountStateChange : ActionListener)
+ getTextFieldUsername() : JTextField
+ getTextFieldPassword() : JTextField
WindowCreateAccount
- serialVersionUID : long = 1L;
- textFieldUsername : JTextField
- textFieldPassword : JPasswordField
- showPasswordCheckBox : JCheckBox
+ WindowCreateAccount(submitCreateAccount : ActionListener, signInStateChange : ActionListener, forgotPasswordStateChange : ActionListener)
+ getTextFieldUsername() : JTextField
+ getTextFieldPassword() : JTextField
WindowForgotPassword
- serialVersionUID : long = 1L;
- AOne : JTextField
- ATwo : JTextField
- AThree : JTextField
- lblQOne : JLabel
- lblQTwo : JLabel
- lblQThree : JLabel
+ WindowForgotPassword(submitForgotPassword : ActionListener, signInStateChange : ActionListener)
+ getAns1() : String
+ getAns2() : String
+ getAns3() : String
+ setAllQuestions(q : ArrayList<String>) : void
+ setQuestion1(q : String) : void
+ setQuestion2(q : String) : void
+ setQuestion3(q : String) : void
WindowAccountInfo
- serialVersionUID : long = 1L;
- QOne : PannelField
- QTwo : PannelField
- QThree : PannelField
- lblHomeDir : JLabel
+ WindowAccountInfo(title : String, submitAccountInfo : ActionListener)
- selectDirectory() : String
+ getTextFieldSecQOne() : JTextField
+ setTextFieldSecQOne(text : String) : void
+ getTextFieldSecAOne() : JTextField
+ setTextFieldSecAOne(text : String) : void
+ getTextFieldSecQTwo() : JTextField
+ setTextFieldSecQTwo(text : String) : void
+ getTextFieldSecATwo() : JTextField
+ setTextFieldSecATwo(text : String) : void
+ getTextFieldSecQThree() : JTextField
+ setTextFieldSecQThree(text : String) : void
+ getTextFieldSecAThree() : JTextField
+ setTextFieldSecAThree(text : String) : void
+ getHomeDir() : String
+ setHomeDir(text : String) : void
Styling
+ HEADER_FONT : Font = "Times New Roman",Font.Bold,20;
+ DIST_RELATED : int = 10;
+ DIST_SEPERATE : int = 25;
+ LOGO_IMAGE : Image = logoImage();
- logoImage() : Image
+ addPlaceholderStyle(textField : JTextField) : void
+ removePlaceholderStyle(textField : JTextField) : void
+ navigationButton(text : String) : JButton
+ basicLabel(text : String) : JLabel
+ basicTextField(text : String) : JTextField
+ rightSideOfForms(btnList : ArrayList<JButton>) : JPanel
+ createGradientPanel() : JPanel
WindowMenu
- serialVersionUID : long = 1L;
+ welcomeMessage: Label
+ WindowMenu(openSecurityPreferences : ActionListener, resetPassword : ActionListener, encryptFile : ActionListener, decryptFile : ActionListener)


Lockbox.storage
AccountInfo
- name : String = "";
- password : String = "";
- homeDirectory : String = "";
- secCodePass : String = "";
- secQ1 : String = "";
- secQ2 : String = "";
- secQ3 : String = "";
- secAns1 : String = "";
- secAns2 : String = "";
- secAns3 : String = "";
+AccountInfo(name : String, password : String, secQOne : String, secAnsOne : String, secQTwo : String, secATwo : String, secQThree : String, secAThree : String, homeDir : String, securityCodePass : String, securityCodeAns : String)
+ printInfo() : void
+ allSecurityAnswers() : String
+ getName() : String
+ setName(name : String) : void
+ getPassword() : String
+ setPassword(password : String) : void
+ getHomeDirectory() : String
+ setHomeDirectory(homeDirectory : String) : void
+ getSecQ1() : String
+ setSecQ1(secQ1 : String) : void
+ getSecQ2() : String
+ setSecQ2(secQ2 : String) : void
+ getSecQ3() : String
+ setSecQ3(secQ3 : String) : void
+ getSecAns1() : String
+ setSecAns1(secAns1 : String) : void
+ getSecAns2() : String
+ setSecAns2(secAns2 : String) : void
+ getSecAns3() : String
+ setSecAns3(secAns3 : String) : void
+ getSecCodePass() : String
+ setSecCodePass(secCodePass : String) : void
+ getSecCodeAns() : String
+ setSecCodeAns(secCodeAns : String) : void
UpdateStorage
+ writeAccount(encryptedAccount : AccountInfo) : void
+ writeAccount(encryptedList : ArrayList<AccountInfo>) : void
+ readAccountStorage(p : Path) : ArrayList<AccountInfo>
+ findPropertiesFile() : Path
FileHelper
+ selectFile(openingDirectory : String, title : String) : Path
+ confirmationWindow(message : String) : Boolean
+ errorMessage(title : String, message : String) : void
+ infoMessage(title : String, message : String) : void
+ passwordValid(password : String) : Boolean



Build State Model



State Diagram

REVISIONS & CHALLENGES

Fortunately, our project didn't require a whole lot of revision or redirection based on the review we received from our peers in the late November project checkpoint initial submission. One of the revisions we needed to make from our initial submission to the final one was updating our class diagram as it had become outdated as the actual code base we were working with became more streamlined and involved different associations/methods than we had initially anticipated. One piece of advice we took from our peer review was "allowing for users to also encrypt images/other file types". We looked into our jasypt library implementation and were excited to find that it was able to encrypt basically any standard file that a user could hope to encrypt.

Learning about and implementing encryption was one of the foundational parts of our project which is all about protecting user's submitted files through encryption. We ended up using the jasypt library which was one of the most accessible ways to implement encryption as we were aiming to. We were able to troubleshoot our encryption/decryption methods and work around exception cases by implementing try and catch statements within our multiple encryption/decryption methods in the 'Serializer' class.

The most challenging aspect of our implementation of Jasypt's Password-Based-Encryption, a type of encryption that uses a string as a sort of codex, was enabling an option for password recovery. Considering multiple options, we chose the route of generating a security code for the users file encryption. This option separated the codex used for encryption from the account password, meaning it can be reset. However, we also had to consider the security questions used to recover the account. The end result was we created two security questions for the account, one encrypted by the account password, the other encrypted by a compilation of the users security question answers. When the user logs in or submits "forgot password" the encryption codes can be decrypted without fault, the account accessed, without ever changing a users codex for Password-Based-Encryption.