I uploaded a project onto GitHub using the command line.

Pointed it towards my project

C:\Users\Tim\Desktop\College\DT354-4\RefactoringExercise

git init,

git add . ,

git commit -m "Initial commit",

git remote add origin https://github.com/humphriest/PaddyRefactoringExercise.git,

git push origin master.

That created and uploaded the project onto github.

EmployeeSummaryDialog.java

I made the class package private because it is good coding practise to encapsulate methods and classes by default for debugging. I did the same, by making private, with three variables , allEmployees, back and DecimalFormatReader the EmployeeSummaryDialog Constructor and made summaryPane() private as it is only accessed inside this class. I have a screenshot of the code before and after declaring the class, methods and variables private. ESDBeforePrivate & ESDAfterPrivate.

I changed this for loop that is inside the summaryPane() method to be a foreach loop as it is better practise and reduces the opportunity for errors to arise by getting rid of the index variable.

for (int i = 0; i < headerName.length; i++) {  
 header.addElement(headerName[i]);  
 }

for (String aHeaderName : headerName) {  
 header.addElement(aHeaderName);  
}

AddRecordDialog.java

Same as in EmployeeSummaryDialog. Encapsulated class methods and variables to be private, have screenshots of before and after of some of them ARDBefore & ARDAfterPrivate. Same for addRecord() and checkInput() and setToWhite().

-correction:

I had to change AddRecordDialog from private to no modifier as it is used in EmployeeDetails

Employee.java

I made all the methods inside the Employee Class with no modifier. The methods are used externally so if I made them private there would be errors when handling the Employee. Making them as they're called 'package-private' means they can only be accessed within this package which is exactly what we need. There is still a level of encapsulation. Visible changes of code in EBeforePrivate & EAfterPrivate.

RandomAccessEmployeeRecord.java

Same as before, changed the class and methods to have no modifier. Methods used inside the class are private. Variable SIZE is also now not modified. Visible changes in RAERBeforePrivate & RAERAfterPrivate.

RandomFile.java

There is duplicate code in this file for closing a file whether it's read or write. I renamed the method to be closeFile() and everywhere where there was the duplicate code I called that method. I had to change the method calls in EmployeeDetails. I showed this in RFBeforeCloseFile & RFAfterClose.

I also changed the methods to be private if they're used inside the class and no modifier if theyre used outside the class inside the package.

In the method changeRecords() he was declaring a new variable currentRecordStart and oldDetails equal to the parameters which is unnecessary and bad practise. It is quicker and more direct to use the variable name given as a parameter instead. You can use these parameters in the functions and the code is now cleaner.

Some methods need to be changed to have no modifier and he has empty catch blocks so I just added a system.out.println to display the error.

I have some of the corrections I made in the pictures RFOne, RFOneA, RFOneB

SearchByIdDialog.java

Changed the classname and methods to have no modifier. Changed variables to be of private as they are only used inside the class. Screenshots called SIDBefore, SIDAfter

SearchBySurnameDialog.java

Changed the classname and methods to have no modifier. Changed variables to be of private as they are only used inside the class. Screenshots called SBSBefore, SBSAfter

EmployeeDetails.java

First fix is the name of the variable FileNameExtensionFilter datfilter changed to datFilter. I changed generatedFileName to private.

I changed a lot of methods to have no modifier. In class getAllEmployees() I changed the element passed into the empDetails.addElement(...) to be just the variable rather than unnecessarily boxing it as its variable type. Evidence in EDGAEBefore & EDGAEAfter.

In the method setEnabled, he has an unnecessary if statement setting search to be either true or false when he can set search equal to the opposite of booleanValue.

EDBB & EDBA

In saveFileAs there was an empty catch block that I filled with

System.*out*.println("Error"+e.getMessage() );

Screenshot called EDErrB EDErrA

In getFileName there was a redundant variable generatedFileName that you could just return fileName.toString() instead.

Screenshot called EDGFNB & EDGFNA