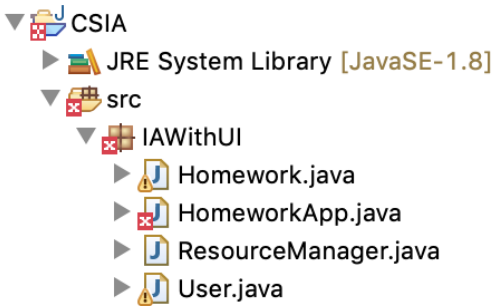


Criterion C: Development

All classes



File of the program:



HomeworkApp.jar

530 KB

Serialization

Java.io.Serializable

The most essential function of the application, storing assignments and user information, is made possible by using Java.io.Serializable. It converts objects into characters sequences, which can be stored into .txt files. (Baeldung)

```
public class ResourceManager {  
    public static void save(Serializable data, String filename) throws Exception  
    {  
        try (ObjectOutputStream oos = new ObjectOutputStream(Files.newOutputStream(Paths.get(filename))))  
        {  
            oos.writeObject(data);  
        }  
    }  
  
    public static Object load(String filename) throws Exception  
    {  
        try (ObjectInputStream ois = new ObjectInputStream(Files.newInputStream(Paths.get(filename))))  
        {  
            return ois.readObject();  
        }  
    }  
}
```

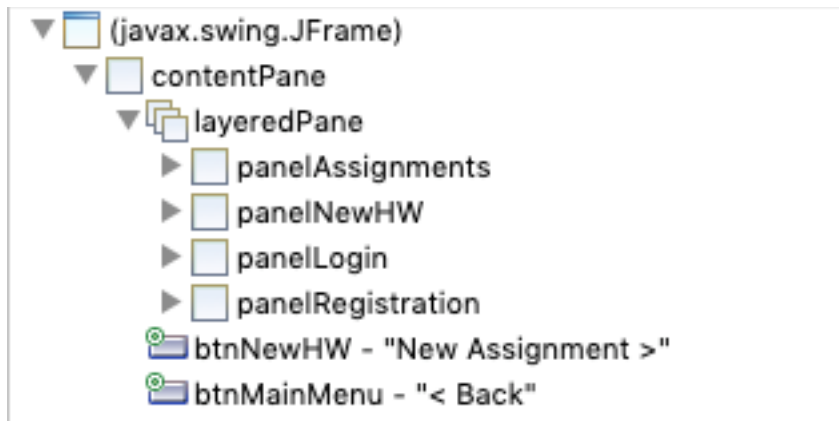
Shown above, `save(Serializable data, String filename)` allows the program to store objects data into file `filename`. `ObjectInputStream` and `ObjectOutputStream` are used to read and write data in binary form. (Baeldung)

`Load(String filename)` enables the program to read the file `filename`.

Try blocks are used during many occasions throughout the program other than the case shown above. It contains commands that may cause exceptions to happen. If an error occurs inside a try block, the system will run the commands in the catch block, which is usually done to report the error.

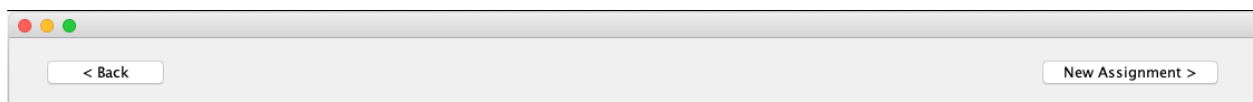
WindowBuilder

WindowBuilder is implemented to achieve a functional GUI.



The entire program is consisted of a JFrame and 4 layers of JPanels, along with 2 buttons to switch over between panelAssignments and panelAssignments.

The different panels allow me to fluently switch over from one window to another or covering up the main window using panelLogin or panelRegistration when logging in or registering.



The buttons above are placed to switch over the panels. Button “New Assignment >” switches the window from panelAssignments to panelNewHW, while button “< Back” does the opposite.

To actualize the switching process between panels, the method `switchPanels(JPanel panel)` is written.

```

public void switchPanels(JPanel panel)
{
    layeredPane.removeAll();
    layeredPane.add(panel);
    layeredPane.repaint();
    layeredPane.revalidate();
}

```

layerPane.removeAll() removes all panels that are currently displayed

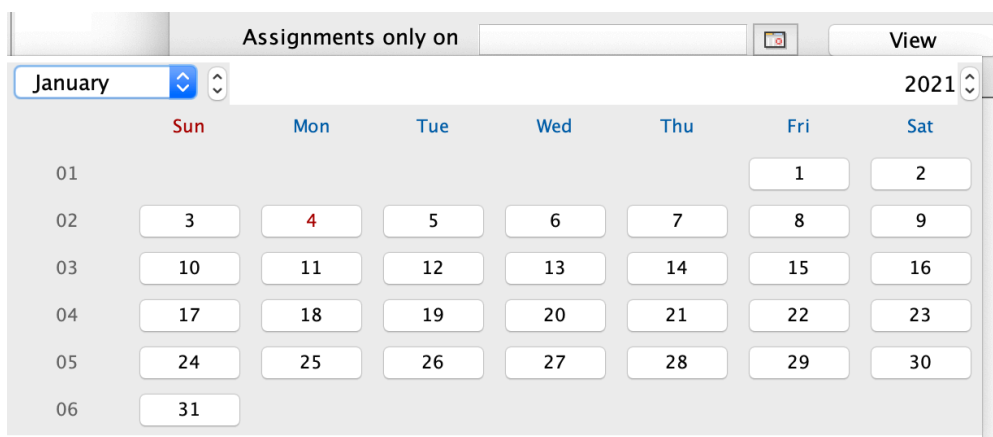
layerPane.add(panel) displays JPanel panel

layerPane.repaint() repaints the panel

layerPane.validate() validates the panel

JCalendar (JCalendar)

Each homework stored on the computer has its specific due date. To help user visualizing the process of entering assignments due dates, a graphical calendar JCalendar is utilized to achieve that. Below shows an image of a DayChooser implemented in the program.



When the program starts running, it attempts to read the file “user.txt”, which is done to determine whether the user has registered before or not.

```

File fileUser = new File("user.txt");

if (ResourceManager.fileExists(fileUser) == false)
{
    switchPanels(panelRegistration);
    btnNewHW.setEnabled(false);
    btnMainMenu.setEnabled(false);
}

if (ResourceManager.fileExists(fileUser) == true)
{
    switchPanels(panelLogin);
    btnNewHW.setEnabled(false);
    btnMainMenu.setEnabled(false);
}

```

fileExists(File file) method:

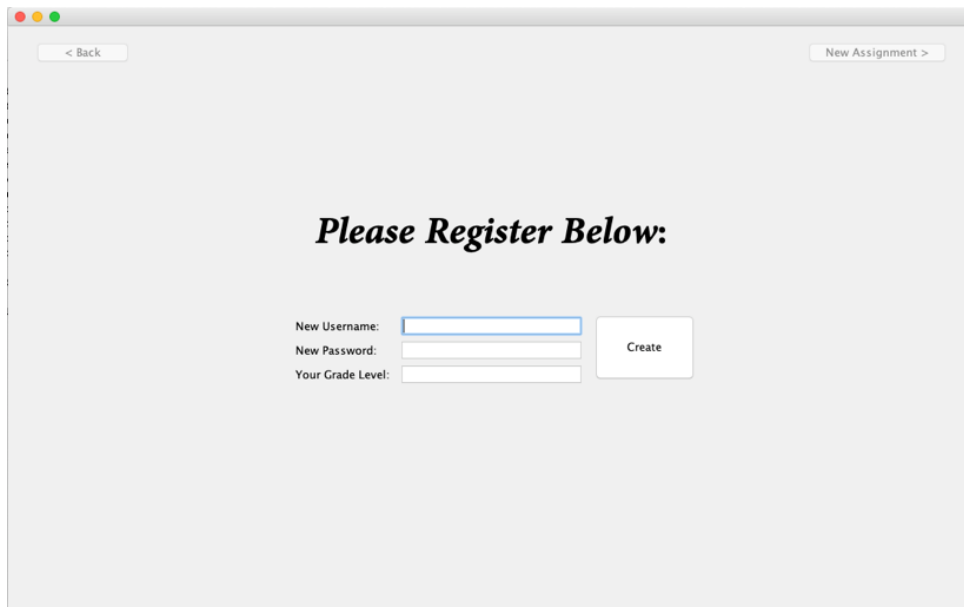
```
public static boolean fileExists(File file)
{
    if (file.exists())
    {
        return true;
    }
    else
    {
        return false;
    }
}
```

If the user has registered before, then the panel is switched to panelLogin, allowing user to input his/her username and password for logging in.

On the other hand, if the user has not registered before, then the program switches the panel to panelRegistration.

In the mean time, as both if statements indicate the login and registration process, since they switch either to panelLogin or panelRegistration, I decided to turn the panel switching buttons transparent by using .setEnabled(Boolean) method. This prevents user to skip the login/registration process.

Registration



In the process shown above, a user account is going to be registered and stored in a .txt file.

User is asked to enter his/her username, password and grade level. When user presses the button “Create”, the three variables are stored in a file titled “user.txt”, and the panel is changed to the main panel showing all assignments, granting user access to the rest of the program.

```

JButton btnCreateUser = new JButton("Create");
btnCreateUser.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        String username = textUsername.getText();
        char[] pass = passwordFieldRegister.getPassword();
        String password = new String(pass);
        int gradeLevel = Integer.parseInt(textGradeLevel.getText());

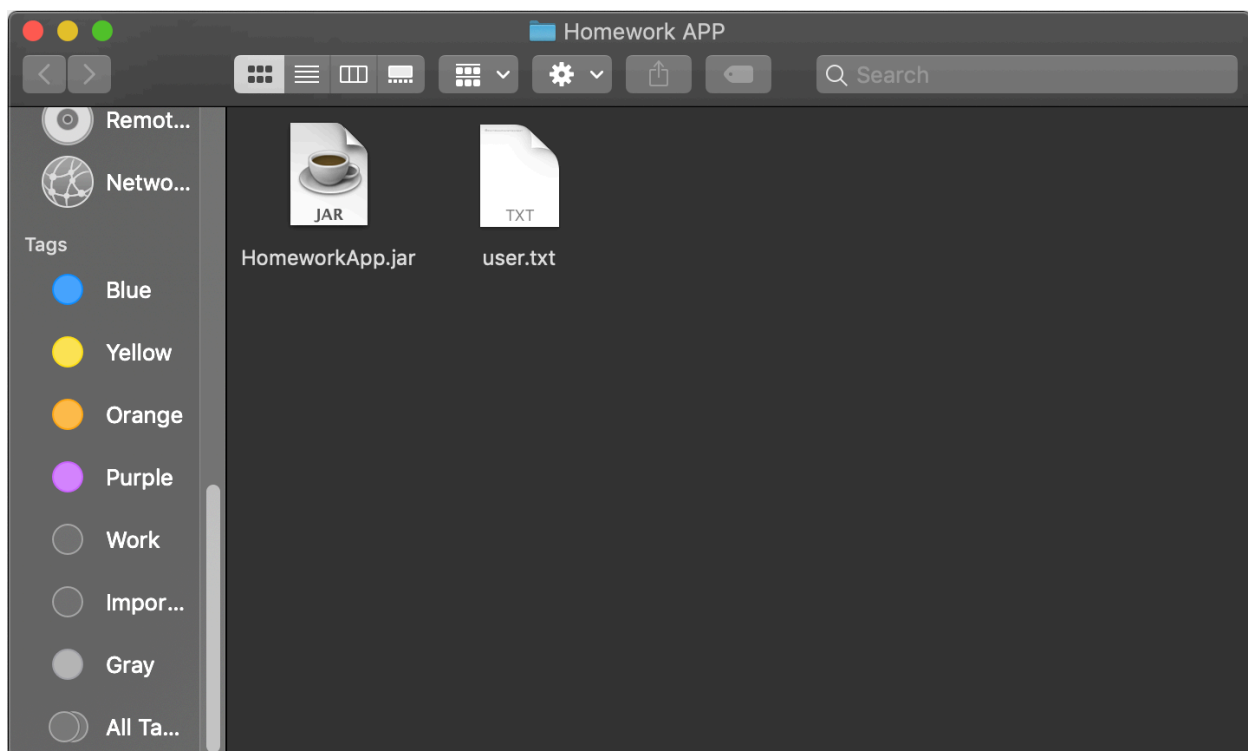
        User user = new User(username, password, gradeLevel);

        try {
            ResourceManager.save(user, "user.txt");
            JOptionPane.showMessageDialog(null, "Successful! Account Created!");
        } catch (Exception e1) {
            System.out.println("Couldn't save: " + e1.getMessage());
        }

        switchPanels(panelAssignments);
        btnNewHW.setEnabled(true);
        btnMainMenu.setEnabled(true);
    }
});

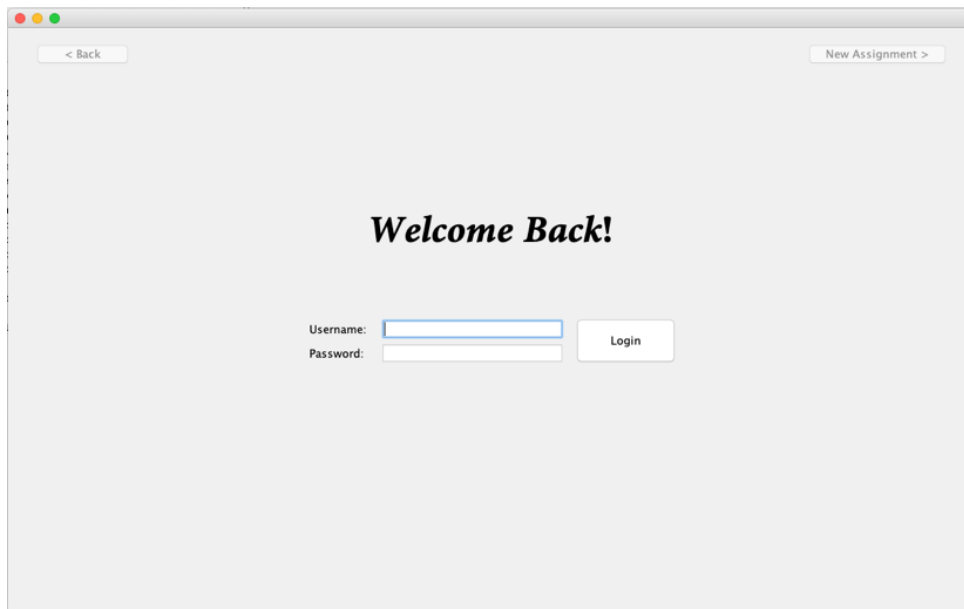
```

Meanwhile, a user.txt file is generated in the same directory as where the program is located that stores the username and password and is shown below.



After switching to panelAssignment using “switchPanels(panelAssignments)”, both buttons, “btnNewHW” and “btnMainMenu”, are set visible to grant user access to the panel of creating new assignment and returning back to the main menu. In the meantime, a message reveals on the screen notifying user that the account is successfully created.

Login



< Back

New Assignment >

Welcome Back!

Username:

Password:

Login

panelLogin is similar to panelRegistration, requiring user input for username and password. Upon pressing the Login button “btnLogin” on the right-hand side, the program reads the entered username and password and compares them with username and password saved in “user.txt”.

When both match each other, the program changes panel to panelAssignment, giving user access to rest of the program.

If one or none of the entered user info match the saved password and username, the program displays a message showing that he/she had inputted the wrong password or username.

```

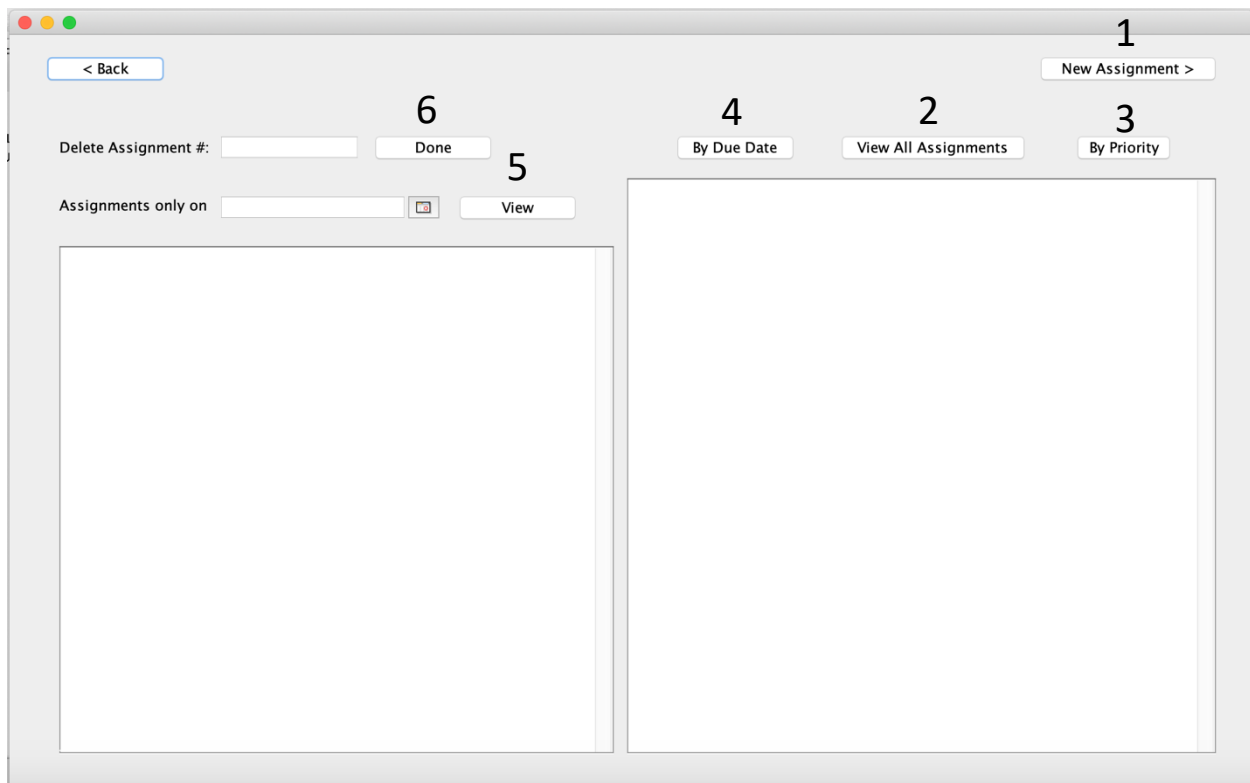
JButton btnLogin = new JButton("Login");
btnLogin.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        String username = textUsernameLogin.getText();
        char[] pass = passwordFieldLogin.getPassword();
        String password = new String(pass);

        try {
            User user1 = (User) ResourceManager.load("user.txt");

            if (username.equals(user1.getUserName()))
            {
                if (password.equals(user1.getPassword()))
                {
                    JOptionPane.showMessageDialog(null, "Welcome back! " + user1.getUserName());
                    switchPanels(panelAssignments);
                    btnNewHW.setEnabled(true);
                    btnMainMenu.setEnabled(true);
                }
                else
                {
                    JOptionPane.showMessageDialog(null, "Wrong password/username! Please try again! ");
                }
            }
            else
            {
                JOptionPane.showMessageDialog(null, "Wrong password/username! Please try again! ");
            }
        } catch (IOException e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
        } catch (Exception e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
        }
    }
});

```



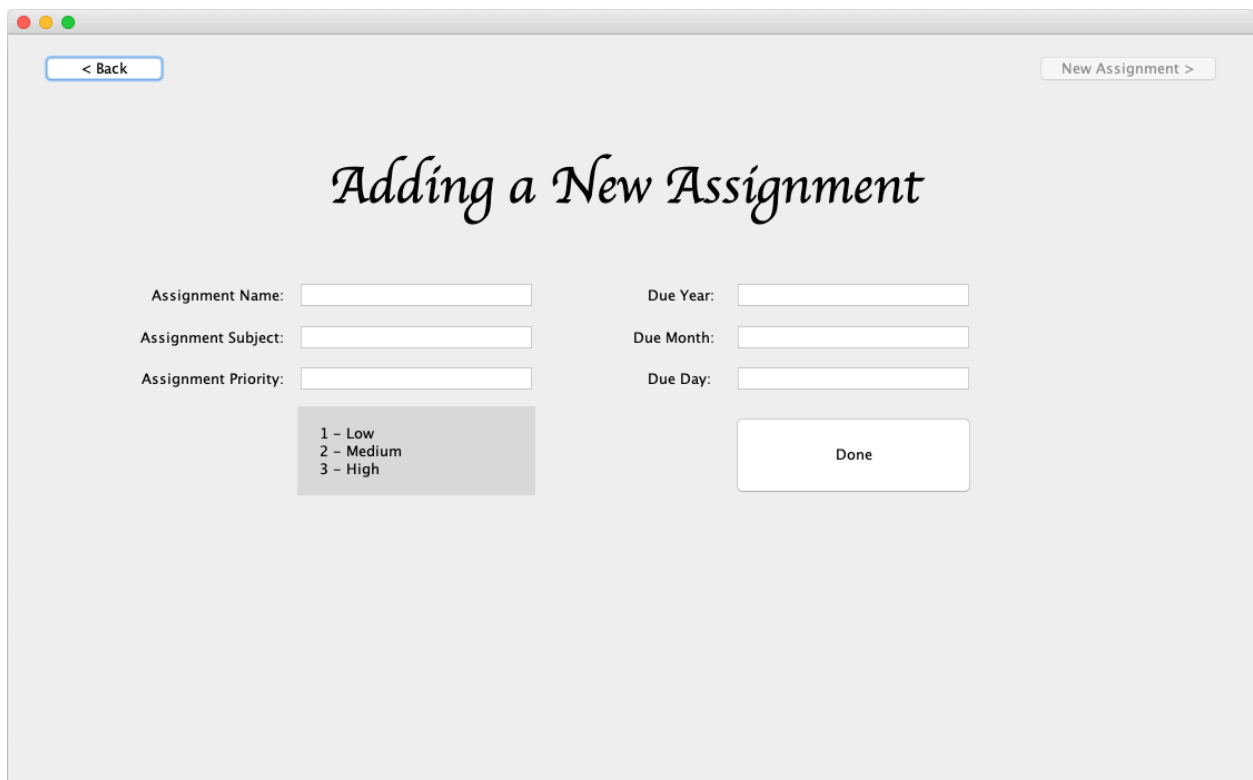
The main panel consists 5 operations:

1. Add new assignment
2. View all assignments (by assignment number)
3. View all assignments by priority
4. View all assignments by due dates
5. View all assignments on the selected date
6. Delete assignment

Operation 2 and 3 display the texts on the right textPanel, while operation 4 shows the texts on the left textPanel.

1. Add new assignment

The panelNewHW is created for the purpose of adding new assignments. 4 variables are entered separately: Assignment name, subject, priority and due date (broken down into year, month and day).



The screenshot shows a Java Swing window with a light gray background. At the top left, there is a '< Back' button. At the top right, there is a 'New Assignment >' button. In the center, the title 'Adding a New Assignment' is displayed in a large, italicized, black serif font. Below the title, there are six text input fields arranged in two columns. The left column contains 'Assignment Name:', 'Assignment Subject:', and 'Assignment Priority:'. The right column contains 'Due Year:', 'Due Month:', and 'Due Day:'. Each label is followed by a white rectangular text box. Below the 'Assignment Priority:' label, there is a gray rectangular button with the text '1 - Low', '2 - Medium', and '3 - High' stacked vertically. To the right of this button, there is a white rectangular button with the text 'Done'.

Upon pressing the button “Done” after inputting, the program reads the current number of existing assignment files, as each assignment is stored as an individual .txt file and is titled with: HW1, HW2 and so on.

Let's say there is an assignment file HW6.txt stored in the computer but there is no HW7.txt, then the program writes all the assignment data into the newly created file HW7.txt and stores them into HW7.txt.

Afterwards, the program resets the input boxes by replacing the user inputted texts by "".

```

JButton btnAddHWDone = new JButton("Done");
btnAddHWDone.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        String assignmentName = textName.getText();
        String subject = textSubject.getText();
        int priority = Integer.parseInt(textPriority.getText());
        int dueYear = Integer.parseInt(textDueYear.getText());
        int dueMonth = Integer.parseInt(textDueMonth.getText());
        int dueDay = Integer.parseInt(textDueDay.getText());

        Homework HW = new Homework(assignmentName, subject, dueYear, dueMonth, dueDay, priority);

        try {
            ResourceManager.save(HW, "HW" + Homework.getHWNu(1) + ".txt");
        } catch (Exception e1) {
            JOptionPane.showMessageDialog(null, "Couldn't save assignments into " + e1.getMessage());
        }

        textName.setText("");
        textSubject.setText("");
        textPriority.setText("");
        textDueYear.setText("");
        textDueMonth.setText("");
        textDueDay.setText("");

        textAreaAssignments.setText(tempText);
    }
});

```

getHWNu(int HWNum) is a recursive method that returns the current numbers of assignment files which are stored inside the computer. It attempts to read assignment files in the order of HW1.txt, HW2.txt, HW3.txt and etc. Upon finding an assignment i, for example, that does not exist, that number i would be the total number of assignments.

```

public static int getHWNu(int HWNum)
{
    File file = new File("HW" + HWNum + ".txt");

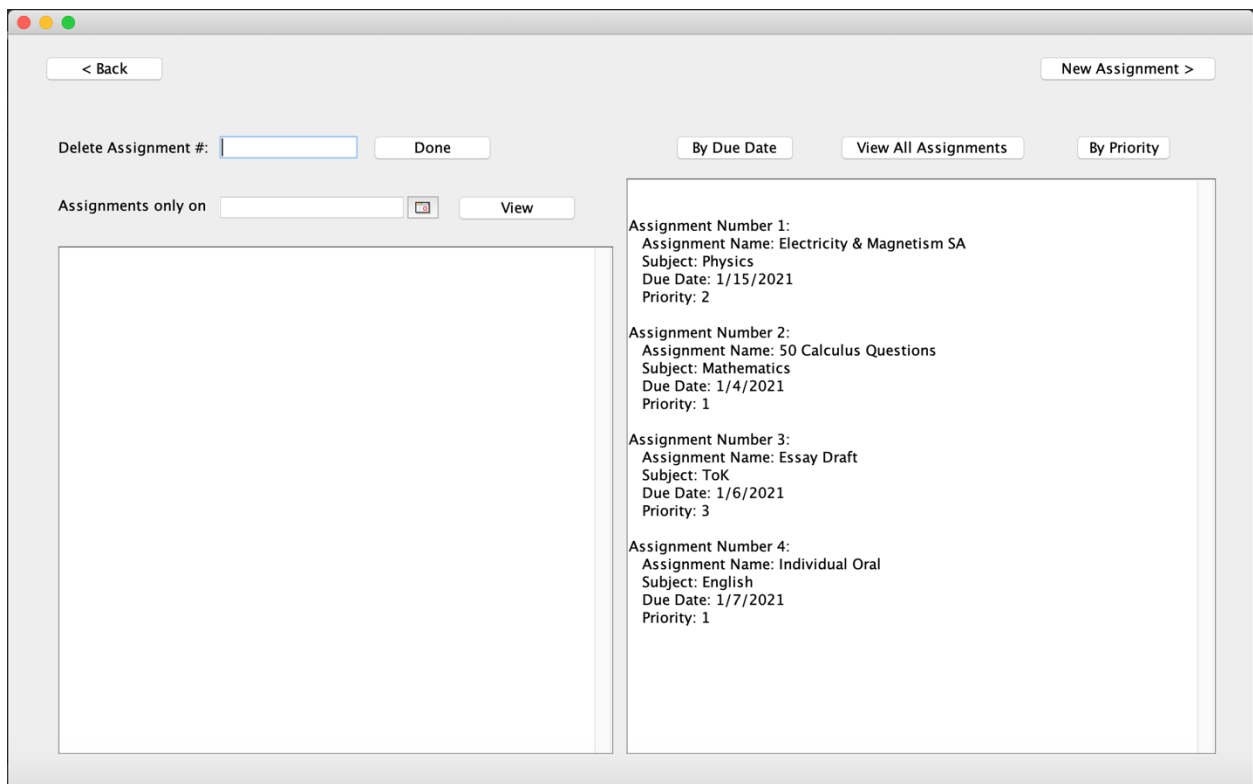
    if (ResourceManager.fileExists(file) == true)
    {
        HWNum++;
        return (getHWNu(HWNum));
    }
    else
    {
        return HWNum--;
    }
}

```

The output of total assignment number is used to the naming of newly created assignment file.

It is repeatedly used throughout the program to help printing all assignments and locating/deleting a specific assignment.

2. View all assignments



The screenshot shows a graphical user interface for managing assignments. At the top, there are two buttons: "< Back" on the left and "New Assignment >" on the right. Below these, on the left, is a "Delete Assignment #:" label followed by a text input field and a "Done" button. To the right of this is a sorting section with three buttons: "By Due Date", "View All Assignments" (which is highlighted), and "By Priority". Below the sorting buttons is a large rectangular area divided into two columns. The left column is labeled "Assignments only on" followed by a dropdown menu and a "View" button. The right column contains a list of four assignments, each with its number, name, subject, due date, and priority.

Assignment Number	Assignment Name	Subject	Due Date	Priority
1	Electricity & Magnetism SA	Physics	1/15/2021	2
2	50 Calculus Questions	Mathematics	1/4/2021	1
3	Essay Draft	ToK	1/6/2021	3
4	Individual Oral	English	1/7/2021	1

Shown above, once the “View All Assignments” button is pressed, all assignments details are printed out in the order of assignment number and in a specific format using a for loop, which prints out assignment details one by one.

```

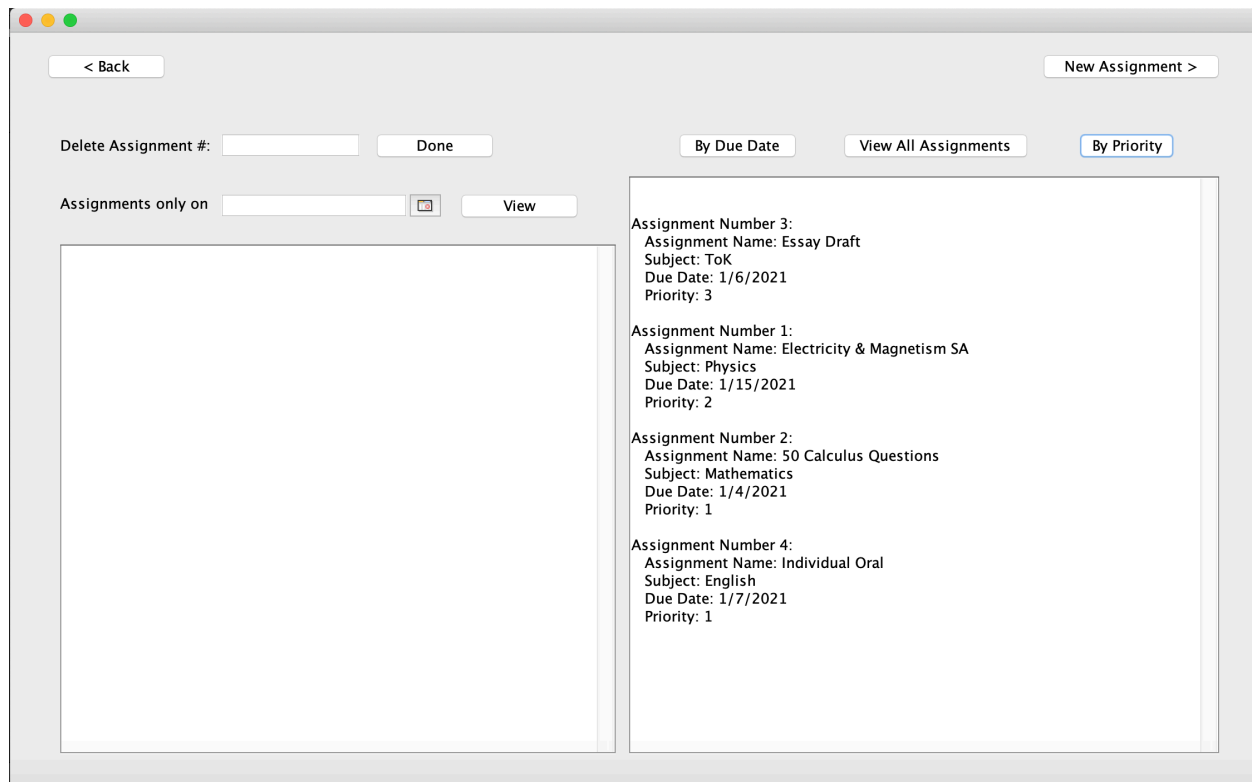
JButton btnListHW = new JButton("View All Assignments");
btnListHW.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        textAreaAssignments.setText("");
        tempText = "";

        for (int i = 1; i < Homework.getHWNum(1); i++)
        {
            try {
                Homework HW = (Homework) ResourceManager.load("HW" + i + ".txt");
                tempText = tempText + "\n\n" + "Assignment Number " + i + ": " +
                    "\n Assignment Name: " + HW.getAssignmentName() +
                    "\n Subject: " + HW.getSubject() +
                    "\n Due Date: " + HW.getDueMonth() + "/" + HW.getDueDay() + "/" + HW.getDueYear() +
                    "\n Priority: " + HW.getPriority();

                textAreaAssignments.setText(tempText);
            } catch (Exception e1) {
                JOptionPane.showMessageDialog(null, "Couldn't load assignments from " + e1.getMessage());
                // TODO Auto-generated catch block
                e1.printStackTrace();
            }
        }
    }
}

```

3. View all assignments by priority



The actualization of displaying assignments by top to lowest priorities is done printing the assignments one by one in order of their priorities.

As the priority ranks from 1 to 3, I first used linear search algorithm to locate the assignments with priorities of 3 and print them out, then followed by the ones with priorities of 2 and 1:

```

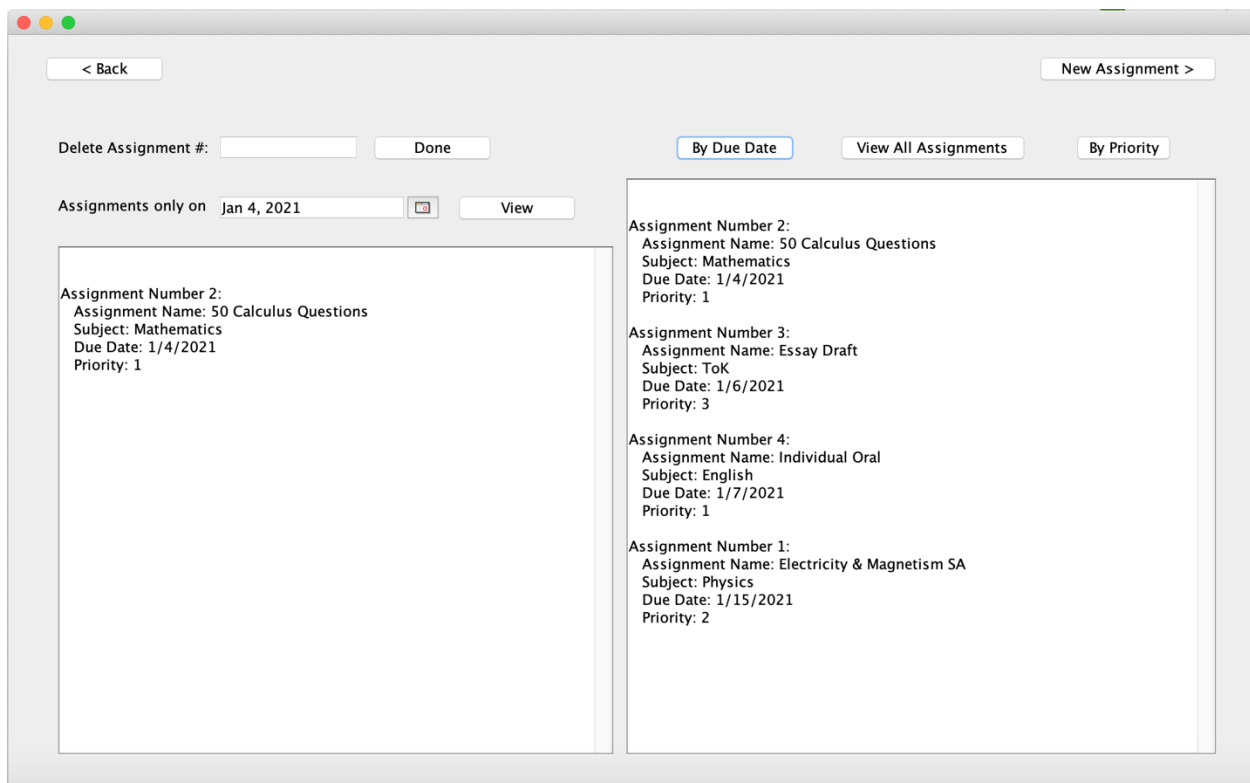
JButton btnListHWByPriority = new JButton("By Priority");
btnListHWByPriority.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        tempText = "";
        textAreaAssignments.setText("");
        //Gets all priorities of assignments
        for (int p = 3; p > 0; p--)
        {
            for (int i = 1; i < Homework.getHWNum(1); i++)
            {
                try {
                    Homework HW = (Homework) ResourceManager.load("HW" + i + ".txt");
                    if (HW.getPriority() == p)
                    {
                        tempText = tempText + "\n\n" + "Assignment Number " + i + ": " +
                            "\n  Assignment Name: " + HW.getAssignmentName() +
                            "\n  Subject: " + HW.getSubject() +
                            "\n  Due Date: " + HW.getDueMonth() + "/" + HW.getDueDay() + "/" + HW.getDueYear() +
                            "\n  Priority: " + HW.getPriority();
                    }
                } catch (Exception e1) {
                    JOptionPane.showMessageDialog(null, "Couldn't load assignments from " + e1.getMessage());
                    // TODO Auto-generated catch block
                    e1.printStackTrace();
                }
            }
        }
    }
}

```

Loops three times, each for a level of priority (1, 2, 3)

This loops n times, where n is the number of assignments

4. View all assignments by due dates



This method is made possible by comparing the extracted due date of each assignment at once and then matching the assignments with the correct due dates for printing out.

Below shows the creation of an arraylist for organizing the due dates of assignments. Each due date is in the format of Year + Month + Day. For instance, an assignment due on Jan 7th, 2021 will have its due date stored in the arraylist as 20210107. This is done for the ease of comparison.

```
//Gets all due dates of assignments
String day = "0";
String month = "0";

ArrayList<Integer> dueDates = new ArrayList<Integer>();

for (int i = 1; i < Homework.getHWNum(); i++)
{
    try {
        Homework HW = (Homework) ResourceManager.load("HW" + i + ".txt");
        if (String.valueOf(HW.getDueDay()).length() == 1)
        {
            day = "0" + HW.getDueDay();
        }
        else
        {
            day = "" + HW.getDueDay();
        }

        if (String.valueOf(HW.getDueMonth()).length() == 1)
        {
            month = "0" + HW.getDueMonth();
        }
        else
        {
            month = "" + HW.getDueMonth();
        }

        String date = String.valueOf(HW.getDueYear()) + month + day;

        dueDates.add(Integer.valueOf(date));

    } catch (Exception e1) {
        JOptionPane.showMessageDialog(null, "Couldn't load assignments from " + e1.getMessage());
        // TODO Auto-generated catch block
        e1.printStackTrace();
    }
}
```

Adds a "0" in front of the day if day is a 1-digit number (1~9 → 01~09)

Adds a "0" in front of the month if month is a 1-digit number (1~9 → 01~09)

String date is the String form of each assignment's date used to compare

Then I used `Collections.sort(dueDates);` method for sorting assignments' due dates from earliest to latest. It is in the `java.util.Collections` class for organizing arraylist in alphabetical order or sizes.

```

for (int i = 0; i < dueDates.size(); i++)
{
    for (int j = 1; j < Homework.getHWNum(1); j++)
    {
        try {
            Homework HW = (Homework) ResourceManager.load("HW" + j + ".txt");
            String Year = "" + Integer.toString(dueDates.get(i)).charAt(0)
                + Integer.toString(dueDates.get(i)).charAt(1)
                + Integer.toString(dueDates.get(i)).charAt(2)
                + Integer.toString(dueDates.get(i)).charAt(3);
            String Month = "" + Integer.toString(dueDates.get(i)).charAt(4)
                + Integer.toString(dueDates.get(i)).charAt(5);
            String Day = "" + Integer.toString(dueDates.get(i)).charAt(6)
                + Integer.toString(dueDates.get(i)).charAt(7);

            if (Month.charAt(0) == '0')
            {
                Month = Month.substring(1);
            }
            if (Day.charAt(0) == '0')
            {
                Day = Day.substring(1);
            }

            if (Year.equals(Integer.toString(HW.getDueYear())))
            {
                if (Month.equals(Integer.toString(HW.getDueMonth())))
                {
                    if (Day.equals(Integer.toString(HW.getDueDay())))
                    {
                        int assignmentNum = j;
                        tempText = tempText + "\n\n" + "Assignment Number " + assignmentNum + ": " +
                            "\n Assignment Name: " + HW.getAssignmentName() +
                            "\n Subject: " + HW.getSubject() +
                            "\n Due Date: " + HW.getDueMonth() + "/" + HW.getDueDay() + "/" + HW.getDueYear() +
                            "\n Priority: " + HW.getPriority();
                    }
                }
            }
        } catch (Exception e1) {
            JOptionPane.showMessageDialog(null, "Couldn't load assignments from " + e1.getMessage());
            // TODO Auto-generated catch block
            e1.printStackTrace();
        }
    }
}

textAreaAssignments.setText(tempText);
});

```

Converts due date of each assignment to the



5. View assignments on the selected date

< Back

New Assignment >

Delete Assignment #:

Done

By Due Date

View All Assignments

By Priority

Assignments only on

View

January

2021

Sun	Mon	Tue	Wed	Thu	Fri	Sat
01					1	2
02	3	4	5	6	7	8
03	10	11	12	13	14	15
04	17	18	19	20	21	22
05	24	25	26	27	28	29
06	31					

Assignment Number 3:

Assignment Name: Essay Draft

Subject: ToK

Due Date: 1/6/2021

Priority: 3

Assignment Number 1:

Assignment Name: Electricity & Magnetism SA

Subject: Physics

Due Date: 1/15/2021

Priority: 2

Assignment Number 2:

Assignment Name: 50 Calculus Questions

Subject: Mathematics

Due Date: 1/4/2021

Priority: 1

Assignment Number 4:

Assignment Name: Individual Oral

Subject: English

Due Date: 1/7/2021

Priority: 1

< Back

New Assignment >

Delete Assignment #:

Done

By Due Date

View All Assignments

By Priority

Assignments only on

Jan 4, 2021

View

Assignment Number 2:

Assignment Name: 50 Calculus Questions

Subject: Mathematics

Due Date: 1/4/2021

Priority: 1

Assignment Number 3:

Assignment Name: Essay Draft

Subject: ToK

Due Date: 1/6/2021

Priority: 3

Assignment Number 1:

Assignment Name: Electricity & Magnetism SA

Subject: Physics

Due Date: 1/15/2021

Priority: 2

Assignment Number 2:

Assignment Name: 50 Calculus Questions

Subject: Mathematics

Due Date: 1/4/2021

Priority: 1

Assignment Number 4:

Assignment Name: Individual Oral

Subject: English

Due Date: 1/7/2021

Priority: 1

Selected dates are first converted to String in order to compare with assignments due dates.

As the `dateChooser.getDate()` method can return: `Mon Jan 04 09:16:58 CST 2021`, I decided to extract the date info off from it using `.charAt(int)`.

```
String date = String.valueOf(dateChooser.getDate());

String year = "" + date.charAt(24) + date.charAt(25) + date.charAt(26) + date.charAt(27);
String monthInLetters = "" + date.charAt(4) + date.charAt(5) + date.charAt(6);
String day = "" + date.charAt(8) + date.charAt(9);
```

Preprocessing the data into wanted format:

```
if (day.charAt(0) == '0')
{
    day = "" + date.charAt(9);
}

int month = 0;
if (monthInLetters.equals("Jan")){month = 1;}
if (monthInLetters.equals("Feb")){month = 2;}
if (monthInLetters.equals("Mar")){month = 3;}
if (monthInLetters.equals("Apr")){month = 4;}
if (monthInLetters.equals("May")){month = 5;}
if (monthInLetters.equals("Jun")){month = 6;}
if (monthInLetters.equals("Jul")){month = 7;}
if (monthInLetters.equals("Aug")){month = 8;}
if (monthInLetters.equals("Sep")){month = 9;}
if (monthInLetters.equals("Oct")){month = 10;}
if (monthInLetters.equals("Nov")){month = 11;}
if (monthInLetters.equals("Dec")){month = 12;}
```

This converts dates from 01~09 to 1~9 and lettered months to numerical months.

```
for (int i = 1; i < Homework.getHWNum(); i++)
{
    try {
        Homework HW = (Homework) ResourceManager.load("HW" + i + ".txt");
        try {
            if (Integer.toString(HW.getDueYear()).equals(year))
            {
                if (HW.getDueMonth() == month)
                {
                    if (Integer.toString(HW.getDueDay()).equals(day))
                    {
                        tempText = tempText + "\n\n" + "Assignment Number " + i + ": " +
                            "\n    Assignment Name: " + HW.getAssignmentName() +
                            "\n    Subject: " + HW.getSubject() +
                            "\n    Due Date: " + HW.getDueMonth() + "/" + HW.getDueDay() + "/" + HW.getDueYear() +
                            "\n    Priority: " + HW.getPriority();

                        textAreaHWForTheDay.setText(tempText);
                    }
                }
            }
        }
    }
}

catch (Exception e1) {
    // TODO Auto-generated catch block
    e1.printStackTrace();
}

catch (Exception e1) {
    // TODO Auto-generated catch block
    e1.printStackTrace();
}
}
```

Using linear search algorithm by sorting out assignments with the right due year, due month and due day, the program prints them out at the end.

6. Delete assignment

Lastly, assignments can be removed by deleting the assignments files with `.delete()`.

For instance, when the user wants to delete assignment number 1 by inputting integer 1 into the input box, the following steps below display the process implemented to achieve deleting the assignment:

1. Deletes file HW1.txt
2. For each homework file after HW1.txt, rename them by its number - 1.

Renaming the remaining assignment files after the deleted one is to ensure that all assignments have continuous sequence of assignment number.

```
JButton btnDeleteAssignment = new JButton("Done");
btnDeleteAssignment.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        File file = new File("HW" + Homework.getHWNum() + ".txt");

        int deleteHWNum = Integer.parseInt(textDeleteAssignmentNum.getText());
        File fileDelete = new File("HW" + deleteHWNum + ".txt");
        fileDelete.delete();

        for (int i = deleteHWNum + 1; i <= Homework.getHWNum(); i++)
        {
            file = new File("HW" + i + ".txt");
            int s = i - 1;
            File file2 = new File("HW" + s + ".txt");
            file.renameTo(file2);
        }
    }
});
btnDeleteAssignment.setBounds(299, 22, 117, 29);
panelAssignments.add(btnDeleteAssignment);
```

The `delete()` method is made possible by importing `java.io.File`.

Before deleting assignment #1:

< Back New Assignment >

Delete Assignment #: Done View All Assignments By Priority

Assignments only on:

Assignment Number 2:
Assignment Name: 50 Calculas questions
Subject: Math
Due Date: 1/4/2021
Priority: 1

Assignment Number 1:
Assignment Name: Summative Assessment
Subject: Physics
Due Date: 1/15/2021
Priority: 1

Assignment Number 2:
Assignment Name: 50 Calculas questions
Subject: Math
Due Date: 1/4/2021
Priority: 1

Assignment Number 3:
Assignment Name: Essay Draft
Subject: ToK
Due Date: 1/6/2021
Priority: 2

Assignment Number 4:
Assignment Name: Individual Oral
Subject: English
Due Date: 1/7/2021
Priority: 3

After deleting assignment #1:

< Back New Assignment >

Delete Assignment #: Done View All Assignments By Priority

Assignments only on:

Assignment Number 1:
Assignment Name: 50 Calculas questions
Subject: Math
Due Date: 1/4/2021
Priority: 1

Assignment Number 1:
Assignment Name: 50 Calculas questions
Subject: Math
Due Date: 1/4/2021
Priority: 1

Assignment Number 2:
Assignment Name: Essay Draft
Subject: ToK
Due Date: 1/6/2021
Priority: 2

Assignment Number 3:
Assignment Name: Individual Oral
Subject: English
Due Date: 1/7/2021
Priority: 3

Word Count: 1422