**Generative AI Prompts**

This document should be stored in your team’s GitHub repository in the *docs* folder. All generative AI prompts must be listed here in chronological order. Utilize the template, copying to make more. The *AI Engine* field is the source you used, for example: Copilot, ChatGPT, *etc.*

| Date10/17/2024 |  | Team Member(s) | Joren LeGoullon |
| --- | --- | --- | --- |
|  |  | AI Engine | ChatGPT |

Prompt: Give me a new run-down of everything we did and how it interacts

| Date | 10/26/24 | Team Member(s) | Joren LeGoullon |
| --- | --- | --- | --- |
|  |  | AI Engine | ChatGPT |

Prompt: can we get some Junit Tests for the UserManager Class? Do you need to see the class again?

| Date | 10/26/24 | Team Member(s) | Joren LeGoullon |
| --- | --- | --- | --- |
|  |  | AI Engine | chatgpt |

Prompt: how does the logic for everything look?

| Date |  | Team Member(s) |  |
| --- | --- | --- | --- |
|  |  | AI Engine |  |

Prompt:

| Date |  | Team Member(s) | Leslie Hunt |
| --- | --- | --- | --- |
|  |  | AI Engine | Chatgpt |

Prompt:

give me a snipper of code that logs out and goes to login when the sign out button when USerHomePage is pressed (actionevent?)To add functionality for logging out and navigating back to the Login page when the logoutButton is pressed, you can implement the following code snippet:

### Updated setUpEventListeners Method in UserHomePage:

java

Copy code

private void setUpEventListeners() {

// Other action listeners...

logoutButton.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

// Dispose of the current UserHomePage frame

dispose();

// Redirect to the Login page

SwingUtilities.invokeLater(() -> new Login());

}

});

}

### Explanation:

1. **Dispose of Current Frame**: The dispose() method closes the current UserHomePage frame.
2. **Redirect to Login**: A new instance of the Login class is created using SwingUtilities.invokeLater to ensure thread safety in the Swing UI.

### Integration Steps:

* Ensure the logoutButton is initialized and part of the UI in UserHomePage.
* Add this action listener to handle the logout action.
* Confirm that the Login class is correctly imported and functional to display the login screen.

This will enable users to log out and return to the login screen when they click the "Sign Out" button.

| Date |  | Team Member(s) | Leslie Hunt |
| --- | --- | --- | --- |
|  |  | AI Engine | Chatgpt |

Prompt: I need a timebox for my gui to show the current time

###### **ChatGPT said:**

ChatGPT

To create a GUI that shows the current time and updates it dynamically (timeboxing), you can use a javax.swing.Timer to refresh the displayed time at regular intervals. Below is an example:

### Example Code

java

Copy code

import javax.swing.\*;

import java.awt.\*;

import java.text.SimpleDateFormat;

import java.util.Date;

public class TimeBoxExample {

public static void main(String[] args) {

// Create the JFrame

JFrame frame = new JFrame("Timebox Example");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setSize(300, 150);

frame.setLayout(new BorderLayout());

// Create a JLabel to display the time

JLabel timeLabel = new JLabel("", SwingConstants.CENTER);

timeLabel.setFont(new Font("Arial", Font.BOLD, 24));

frame.add(timeLabel, BorderLayout.CENTER);

// Create a Timer to update the time every second

Timer timer = new Timer(1000, e -> {

// Get the current time and format it

String currentTime = new SimpleDateFormat("HH:mm:ss").format(new Date());

timeLabel.setText(currentTime);

});

// Start the Timer

timer.start();

// Display the JFrame

frame.setVisible(true);

}

}

### How It Works:

1. **JLabel**:
   * Used to display the current time in the center of the GUI.
2. **SimpleDateFormat**:
   * Formats the current time into a readable format (HH:mm:ss).
3. **javax.swing.Timer**:
   * Triggers an action every 1000 milliseconds (1 second) to refresh the time.
4. **Dynamic Update**:
   * The Timer updates the JLabel text with the current time every second.
5. **Font and Alignment**:
   * The Font and SwingConstants.CENTER ensure the time looks clear and centered in the window.

### Output:

* A small GUI window will display the current time in the format HH:mm:ss.
* The time updates dynamically every second.

This is a simple and effective way to timebox the current time in a GUI. Let me know if you want to add more features, such as a stop/start button or a different time format!

| Date | 11/20/24 | Team Member(s) | Leslie Hunt |
| --- | --- | --- | --- |
|  |  | AI Engine | Chatgpt |

Prompt: is there a way to change the theme other than the gui looking like an early 2000s web app?

Given:

public static void main(String[] args) {

try {

// Set the Nimbus look and feel

for (UIManager.LookAndFeelInfo info : UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (Exception e) {

System.out.println("Failed to set look and feel: " + e.getMessage());

}

SwingUtilities.invokeLater(UserHomePage::new);

}

| Date | 12/7/24 | Team Member(s) | Leslie Dawn |
| --- | --- | --- | --- |
|  |  | AI Engine | chatgpt |

Prompt:

whats an addListSelectionListener

* **ChatGPT said:**

ChatGPT

An addListSelectionListener is a method used in Java's JList component to register an event listener that listens for changes in the selection of list items. This is a part of the javax.swing.event.ListSelectionListener interface, which is used to handle list selection events.

* **Purpose of addListSelectionListener**

The purpose of using an addListSelectionListener is to perform some action when the user selects (or deselects) an item in the JList. For example, when a user clicks on an item in a list, you might want to display more details about that item in a separate part of your UI, such as a JTextArea.

* **Example of addListSelectionListener**

In your case, this is how you might use it:

java

Copy code

auctionsList.addListSelectionListener(e -> {

if (!e.getValueIsAdjusting()) { // This prevents the listener from being triggered multiple times while the selection is still changing

Item selectedItem = auctionsList.getSelectedValue(); // Get the currently selected item

if (selectedItem != null) {

// Populate the JTextArea with the selected item's details

txaItemInfo.setText(String.format("Item Name: %s\n" +

"Description: %s\n" +

"Start Price: $%.2f\n" +

"Buy It Now Price: $%.2f\n" +

"Category: %s\n" +

"Auction Active: %b\n",

selectedItem.getItemName(), selectedItem.getDescription(),

selectedItem.getStartPrice(), selectedItem.getBuyItNowPrice(),

selectedItem.getItemType(), selectedItem.isAuction()));

}

}

});

* **Key Points:**
* **addListSelectionListener**: This method is used to add a listener to a JList. It listens for selection changes, and when the user selects an item, the listener's valueChanged method is called.
* **Event Object (ListSelectionEvent e)**: The event (e) passed to the listener contains information about the change in selection. You can use e.getValueIsAdjusting() to determine if the selection is still in the process of changing (for example, if the user is dragging to select multiple items).
* **Selected Value**: You can use getSelectedValue() on the JList to retrieve the selected item (in your case, it seems to be an Item object).
* **Why it's Used:**

In your case, when a user selects an item from the auctionsList, the addListSelectionListener is used to update the JTextArea (txaItemInfo) with details about the selected item. This could be the item's name, description, starting price, and other relevant details.

| Date | 12/7/24 | Team Member(s) | Leslie Dawn |
| --- | --- | --- | --- |
|  |  | AI Engine | chatgpt |

Prompt:

How should I sort my active auctions by soonest to end?

private void populateActiveAuctions() { // Get the list of active auctions (items) List<Item> activeAuctionsList = ItemManager.getInstance().populateDefaultActiveAuctions(); // Create a new JList with the list of items auctionsList = new JList<>(new DefaultListModel<Item>()); // Populate the JList with items from the activeAuctionsList DefaultListModel<Item> listModel = (DefaultListModel<Item>) auctionsList.getModel(); for (Item item : activeAuctionsList) { listModel.addElement(item); } // Optionally, add the auctionsList to a JScrollPane if needed JScrollPane scrollPane = new JScrollPane(auctionsList); activeAuctions.setViewportView(scrollPane); }

* **ChatGPT said:**

ChatGPT

To sort the active auctions by the soonest ending date while keeping your current method structure intact, we can sort the activeAuctionsList **before** populating the JList. This avoids modifying the logic inside the populateActiveAuctions method itself, making the approach more modular.

Here's how you can integrate sorting without disrupting the existing method structure:

* **Add a Sorting Step**

Update the populateActiveAuctions method to include sorting of activeAuctionsList **before** populating the JList:

java

Copy code

private void populateActiveAuctions() {

// Get the list of active auctions (items)

List<Item> activeAuctionsList = ItemManager.getInstance().populateDefaultActiveAuctions();

// Sort the active auctions by endTime (soonest to end first)

activeAuctionsList.sort((item1, item2) -> Long.compare(item1.getEndTime(), item2.getEndTime()));

// Create a new JList with the list of items

auctionsList = new JList<>(new DefaultListModel<Item>());

// Populate the JList with items from the activeAuctionsList

DefaultListModel<Item> listModel = (DefaultListModel<Item>) auctionsList.getModel();

for (Item item : activeAuctionsList) {

listModel.addElement(item); // Add each sorted item to the model

}

// Optionally, add the auctionsList to a JScrollPane if needed

JScrollPane scrollPane = new JScrollPane(auctionsList);

activeAuctions.setViewportView(scrollPane);

}

* **Key Changes**

1. **Sorting the activeAuctionsList**:

Before populating the JList, sort the list with:  
 java  
 Copy code  
 activeAuctionsList.sort((item1, item2) -> Long.compare(item1.getEndTime(), item2.getEndTime()));

* This compares the endTime of two items and sorts them in ascending order.

2. **Retain Existing Logic**:

* + The rest of the method remains unchanged, keeping the logic for populating the JList intact.

* **How It Works**

1. **Retrieve Auctions**:
   * Fetch the list of active auctions from ItemManager.
2. **Sort Auctions**:
   * Sort the activeAuctionsList based on the endTime field, so the soonest auction ends first.
3. **Populate JList**:
   * Use the existing DefaultListModel logic to populate the JList with the sorted items.
4. **Attach to JScrollPane**:
   * Attach the auctionsList to the JScrollPane (activeAuctions) as before.

* **Example Output**

If activeAuctionsList has the following endTime values (in milliseconds):

* Auction A: endTime = 1700000000000
* Auction B: endTime = 1700000001000
* Auction C: endTime = 1700000000500

After sorting:

* Auction A
* Auction C
* Auction B

The JList will display these auctions in the same order.

* **Testing**

1. Verify that the auctionsList displays items in the correct order based on endTime.
2. Ensure no NullPointerException occurs.

Let me know if this approach works or if additional adjustments are needed!

| Date | 12-7 thru 12/11 | Team Member(s) |  |
| --- | --- | --- | --- |
|  |  | AI Engine | Leslie Hunt |

Prompt:

for my system test, why am i getting this? Username already exists. java.lang.NullPointerException: Cannot invoke "ebay.ItemManager.getActiveAuctions()" because "this.itemManager" is null at ebay.User.getActiveAuctions(User.java:169) at Testing.SystemTest1.testUserStory2\_condition1(SystemTest1.java:332) at java.base/jdk.internal.reflect.DirectMethodHandleAccessor.invoke(DirectMethodHandleAccessor.java:103) at java.base/java.lang.reflect.Method.invoke(Method.java:580) at org.junit.platform.commons.util.ReflectionUtils.invokeMethod(ReflectionUtils.java:725) at org.junit.jupiter.engine.execution.MethodInvocation.proceed(MethodInvocation.java:60) at org.junit.jupiter.engine.execution.InvocationInterceptorChain$ValidatingInvocation.proceed(InvocationInterceptorChain.java:131) at org.junit.jupiter.engine.extension.TimeoutExtension.intercept(TimeoutExtension.java:149) at org.junit.jupiter.engine.extension.TimeoutExtension.interceptTestableMethod(TimeoutExtension.java:140) at org.junit.jupiter.engine.extension.TimeoutExtension.interceptTestMethod(TimeoutExtension.java:84) at org.junit.jupiter.engine.execution.ExecutableInvoker$ReflectiveInterceptorCall.lambda$ofVoidMethod$0(ExecutableInvoker.java:115) at org.junit.jupiter.engine.execution.ExecutableInvoker.lambda$invoke$0(ExecutableInvoker.java:105) at org.junit.jupiter.engine.execution.InvocationInterceptorChain$InterceptedInvocation.proceed(InvocationInterceptorChain.java:106) at org.junit.jupiter.engine.execution.InvocationInterceptorChain.proceed(InvocationInterceptorChain.java:64) at org.junit.jupiter.engine.execution.InvocationInterceptorChain.chainAndInvoke(InvocationInterceptorChain.java:45) at org.junit.jupiter.engine.execution.InvocationInterceptorChain.invoke(InvocationInterceptorChain.java:37) at org.junit.jupiter.engine.execution.ExecutableInvoker.invoke(ExecutableInvoker.java:104) at org.junit.jupiter.engine.execution.ExecutableInvoker.invoke(ExecutableInvoker.java:98) at org.junit.jupiter.engine.descriptor.TestMethodTestDescriptor.lambda$invokeTestMethod$7(TestMethodTestDescriptor.java:214) at org.junit.platform.engine.support.hierarchical.ThrowableCollector.execute(ThrowableCollector.java:73) at org.junit.jupiter.engine.descriptor.TestMethodTestDescriptor.invokeTestMethod(TestMethodTestDescriptor.java:210) at org.junit.jupiter.engine.descriptor.TestMethodTestDescriptor.execute(TestMethodTestDescriptor.java:135) at org.junit.jupiter.engine.descriptor.TestMethodTestDescriptor.execute(TestMethodTestDescriptor.java:66) at org.junit.platform.engine.support.hierarchical.NodeTestTask.lambda$executeRecursively$6(NodeTestTask.java:151) at org.junit.platform.engine.support.hierarchical.ThrowableCollector.execute(ThrowableCollector.java:73) at org.junit.platform.engine.support.hierarchical.NodeTestTask.lambda$executeRecursively$8(NodeTestTask.java:141) at org.junit.platform.engine.support.hierarchical.Node.around(Node.java:137) at org.junit.platform.engine.support.hierarchical.NodeTestTask.lambda$executeRecursively$9(NodeTestTask.java:139) at org.junit.platform.engine.support.hierarchical.ThrowableCollector.execute(ThrowableCollector.java:73) at org.junit.platform.engine.support.hierarchical.NodeTestTask.executeRecursively(NodeTestTask.java:138) at org.junit.platform.engine.support.hierarchical.NodeTestTask.execute(NodeTestTask.java:95) at java.base/java.util.ArrayList.forEach(ArrayList.java:1597) at org.junit.platform.engine.support.hierarchical.SameThreadHierarchicalTestExecutorService.invokeAll(SameThreadHierarchicalTestExecutorService.java:41) at org.junit.platform.engine.support.hierarchical.NodeTestTask.lambda$executeRecursively$6(NodeTestTask.java:155) at org.junit.platform.engine.support.hierarchical.ThrowableCollector.execute(ThrowableCollector.java:73) at org.junit.platform.engine.support.hierarchical.NodeTestTask.lambda$executeRecursively$8(NodeTestTask.java:141) at org.junit.platform.engine.support.hierarchical.Node.around(Node.java:137) at org.junit.platform.engine.support.hierarchical.NodeTestTask.lambda$executeRecursively$9(NodeTestTask.java:139) at org.junit.platform.engine.support.hierarchical.ThrowableCollector.execute(ThrowableCollector.java:73) at org.junit.platform.engine.support.hierarchical.NodeTestTask.executeRecursively(NodeTestTask.java:138) at org.junit.platform.engine.support.hierarchical.NodeTestTask.execute(NodeTestTask.java:95) at java.base/java.util.ArrayList.forEach(ArrayList.java:1597) at org.junit.platform.engine.support.hierarchical.SameThreadHierarchicalTestExecutorService.invokeAll(SameThreadHierarchicalTestExecutorService.java:41) at org.junit.platform.engine.support.hierarchical.NodeTestTask.lambda$executeRecursively$6(NodeTestTask.java:155) at org.junit.platform.engine.support.hierarchical.ThrowableCollector.execute(ThrowableCollector.java:73) at org.junit.platform.engine.support.hierarchical.NodeTestTask.lambda$executeRecursively$8(NodeTestTask.java:141) at org.junit.platform.engine.support.hierarchical.Node.around(Node.java:137) at org.junit.platform.engine.support.hierarchical.NodeTestTask.lambda$executeRecursively$9(NodeTestTask.java:139) at org.junit.platform.engine.support.hierarchical.ThrowableCollector.execute(ThrowableCollector.java:73) at org.junit.platform.engine.support.hierarchical.NodeTestTask.executeRecursively(NodeTestTask.java:138) at org.junit.platform.engine.support.hierarchical.NodeTestTask.execute(NodeTestTask.java:95) at org.junit.platform.engine.support.hierarchical.SameThreadHierarchicalTestExecutorService.submit(SameThreadHierarchicalTestExecutorService.java:35) at org.junit.platform.engine.support.hierarchical.HierarchicalTestExecutor.execute(HierarchicalTestExecutor.java:57) at org.junit.platform.engine.support.hierarchical.HierarchicalTestEngine.execute(HierarchicalTestEngine.java:54) at org.junit.platform.launcher.core.EngineExecutionOrchestrator.execute(EngineExecutionOrchestrator.java:107) at org.junit.platform.launcher.core.EngineExecutionOrchestrator.execute(EngineExecutionOrchestrator.java:88) at org.junit.platform.launcher.core.EngineExecutionOrchestrator.lambda$execute$0(EngineExecutionOrchestrator.java:54) at org.junit.platform.launcher.core.EngineExecutionOrchestrator.withInterceptedStreams(EngineExecutionOrchestrator.java:67) at org.junit.platform.launcher.core.EngineExecutionOrchestrator.execute(EngineExecutionOrchestrator.java:52) at org.junit.platform.launcher.core.DefaultLauncher.execute(DefaultLauncher.java:114) at org.junit.platform.launcher.core.DefaultLauncher.execute(DefaultLauncher.java:86) at org.junit.platform.launcher.core.DefaultLauncherSession$DelegatingLauncher.execute(DefaultLauncherSession.java:86) at org.junit.platform.launcher.core.SessionPerRequestLauncher.execute(SessionPerRequestLauncher.java:53) at com.intellij.junit5.JUnit5IdeaTestRunner.startRunnerWithArgs(JUnit5IdeaTestRunner.java:57) at com.intellij.rt.junit.IdeaTestRunner$Repeater$1.execute(IdeaTestRunner.java:38) at com.intellij.rt.execution.junit.TestsRepeater.repeat(TestsRepeater.java:11) at com.intellij.rt.junit.IdeaTestRunner$Repeater.startRunnerWithArgs(IdeaTestRunner.java:35) at com.intellij.rt.junit.JUnitStarter.prepareStreamsAndStart(JUnitStarter.java:235) at com.intellij.rt.junit.JUnitStarter.main(JUnitStarter.java:54)

* **ChatGPT said:**

ChatGPT

The issue stems from two main points:

* **1. "Username already exists"**
* This indicates that the registerUser() method in the UserController class is being called with a username that is already present in the userDatabase.
* This is likely because:
  + The userDatabase was not cleared between tests.
  + Or, the same username was used multiple times during the test setup.
* **Solution**:
  + Before starting a test, clear or reinitialize the userDatabase to ensure no residual data remains. Add a method like this to UserController:

java

Copy code

public void clearUserDatabase() {

userDatabase.clear();

saveUserDatabase();

}

* + Call this method at the beginning of your test setup.
* **2. itemManager is null**
* The NullPointerException indicates that the ItemManager instance (itemManager) is not properly initialized in your test or User class.
* The itemManager is likely expected to be a singleton (ItemManager.getInstance()), but in your test or User constructor, it might not be initialized.
* **Root Cause**:
  + The User class is trying to call getActiveAuctions() on a null itemManager instance, as seen in this snippet:

java

Copy code

this.itemManager.getActiveAuctions();

* **Solution**:
  + Ensure that the ItemManager is properly initialized before calling methods on it.
  + For example, modify your User class to ensure itemManager is initialized:

java

Copy code

private ItemManager itemManager = ItemManager.getInstance();

* + Alternatively, ensure it is set correctly in your test setup:

java

Copy code

ItemManager itemManager = ItemManager.getInstance();

* **Steps to Fix**

1. **Initialize itemManager in Test Setup**:

java

Copy code

@BeforeEach

public void setUp() {

ItemManager.getInstance().clearItems(); // Clear any residual data

}

1. **Reinitialize or Clear userDatabase**:

java

Copy code

@BeforeEach

public void resetUserDatabase() {

UserController userController = new UserController();

userController.clearUserDatabase();

}

1. **Verify itemManager in the User Class**:
   * Ensure that the itemManager is initialized or injected properly into the User class.
2. **Debug Test Logic**:
   * Check where the User instance is created in your test (SystemTest1).
   * Ensure itemManager is set up correctly before invoking getActiveAuctions().
3. **Add Logging**:
   * Add debug logs in your User and ItemManager classes to confirm when itemManager is null.

| Date | 12/06/24 | Team Member(s) | Mashayla |
| --- | --- | --- | --- |
|  |  | AI Engine | Copilot |

Prompt: I would like user to be able to select the bid

| Date | 12/09/24 | Team Member(s) | Mashayla |
| --- | --- | --- | --- |
|  |  | AI Engine | Copilot |

Prompt: How can I make bid text box inedible

| Date | 12/09/24 | Team Member(s) | Mashayla |
| --- | --- | --- | --- |
|  |  | AI Engine | Copilot |

Prompt: How can I Improve bid functionality

| Date | 12/10/24 | Team Member(s) | Mashayla |
| --- | --- | --- | --- |
|  |  | AI Engine | Copilot |

Prompt: Do a quick test case for Seller and Buyer Report

@Test

public void testShowSellerReport() {

// Arrange

User currentUser = userController.getCurrentUser();

Item soldItem1 = new Item("Sold Item 1", "Description 1", 50.0, "http://example.com/image1.jpg", true, "Electronics", 50.0, Instant.now().plusSeconds(8400));

Item soldItem2 = new Item("Sold Item 2", "Description 2", 75.0, "http://example.com/image2.jpg", true, "Fashion", 75.0, Instant.now().plusSeconds(8400));

currentUser.addSoldItem(soldItem1);

currentUser.addSoldItem(soldItem2);

// Act

userHomePage.showSellerReport(currentUser);

// Assert

String expectedReport = "Seller's Report for " + currentUser.getUsername() + "\nSold Item 1\nSold Item 2\n";

assertEquals(expectedReport, userHomePage.getReportsText().getText());

}

@Test

public void testShowBuyerReport() {

// Arrange

User currentUser = userController.getCurrentUser();

Item boughtItem1 = new Item("Bought Item 1", "Description 1", 50.0, "http://example.com/image1.jpg", true, "Electronics", 50.0, Instant.now().plusSeconds(8400));

Item boughtItem2 = new Item("Bought Item 2", "Description 2", 75.0, "http://example.com/image2.jpg", true, "Fashion", 75.0, Instant.now().plusSeconds(8400));

currentUser.addBoughtItem(boughtItem1);

currentUser.addBoughtItem(boughtItem2);

// Act

userHomePage.showBuyerReport(currentUser);

// Assert

String expectedReport = "Buyer's Report for " + currentUser.getUsername() + "\nBought Item 1\nBought Item 2\n";

assertEquals(expectedReport, userHomePage.getReportsText().getText());

}

| Date | 12/11/24 | Team Member(s) | Mashayla |
| --- | --- | --- | --- |
|  |  | AI Engine | Chatgpt |

Prompt: Help me make a description for my test case.

| Date | 12/11/24 | Team Member(s) |  |
| --- | --- | --- | --- |
|  |  | AI Engine |  |

Prompt: Rearrange the System test in numerical order that I had them labeled