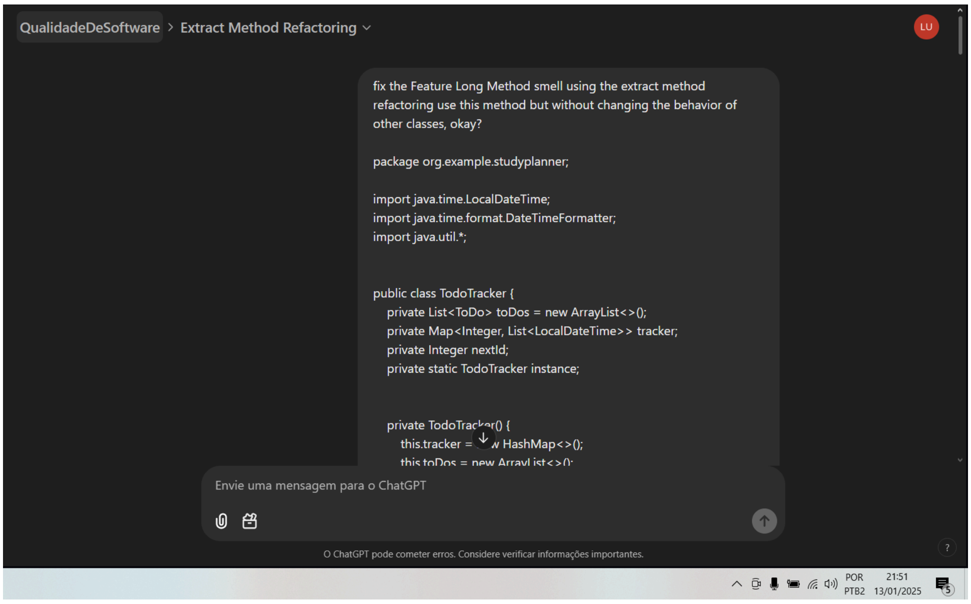
# LongMethod


QualidadeDeSoftware > Extract Method Refactoring
LU
Envie uma mensagem para o ChatGPT
O ChatGPT pode cometer erros. Considere verificar informações importantes.
POR PTB2
21:51 13/01/2025

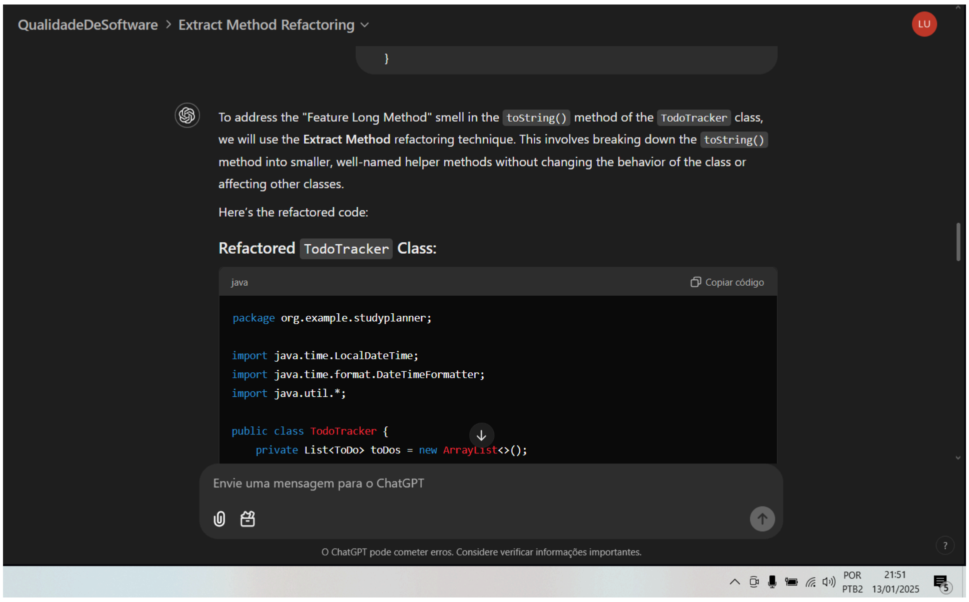
fix the Feature Long Method smell using the extract method refactoring use this method but without changing the behavior of other classes, okay?

```
package org.example.studyplanner;

import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.*;


public class TodoTracker {
    private List<ToDo> toDos = new ArrayList<>();
    private Map<Integer, List<LocalDateTime>> tracker;
    private Integer nextId;
    private static TodoTracker instance;


    private TodoTracker() {
        this.tracker = w HashMap<>();
        this.toDos = new ArrayList<>();
```


QualidadeDeSoftware > Extract Method Refactoring
LU
Envie uma mensagem para o ChatGPT
O ChatGPT pode cometer erros. Considere verificar informações importantes.
POR PTB2
21:51 13/01/2025

```
    }
```

To address the "Feature Long Method" smell in the `toString()` method of the `TodoTracker` class, we will use the **Extract Method** refactoring technique. This involves breaking down the `toString()` method into smaller, well-named helper methods without changing the behavior of the class or affecting other classes.

Here's the refactored code:

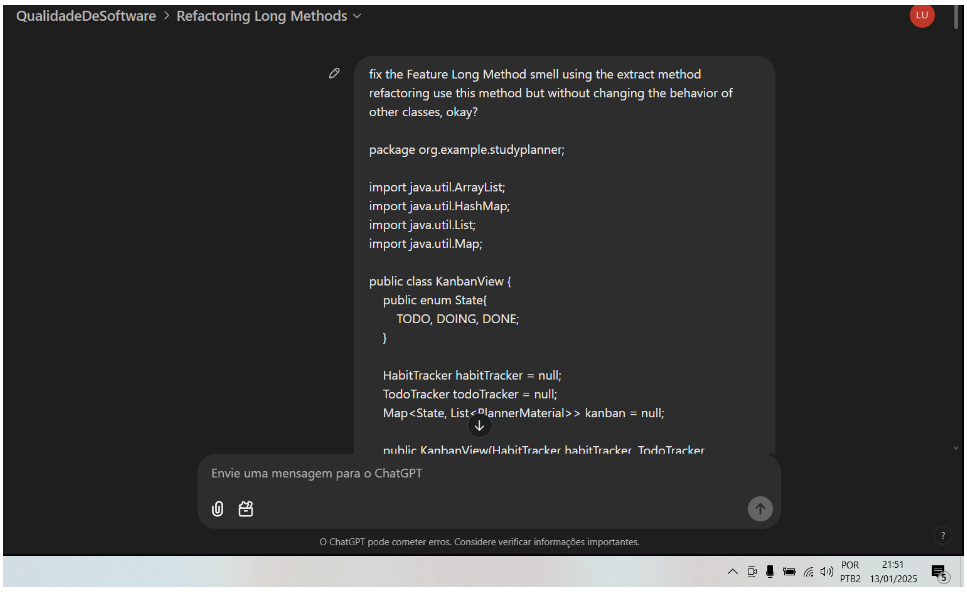
### Refactored `TodoTracker` Class:

```java
package org.example.studyplanner;

import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.*;

public class TodoTracker {
    private List<ToDo> toDos = new ArrayList<>();
```

# Long Method


QualidadeDeSoftware > Refactoring Long Methods
LU
Envie uma mensagem para o ChatGPT
O ChatGPT pode cometer erros. Considere verificar informações importantes.
POR
PTB2
21:51
13/01/2025

fix the Feature Long Method smell using the extract method refactoring use this method but without changing the behavior of other classes, okay?

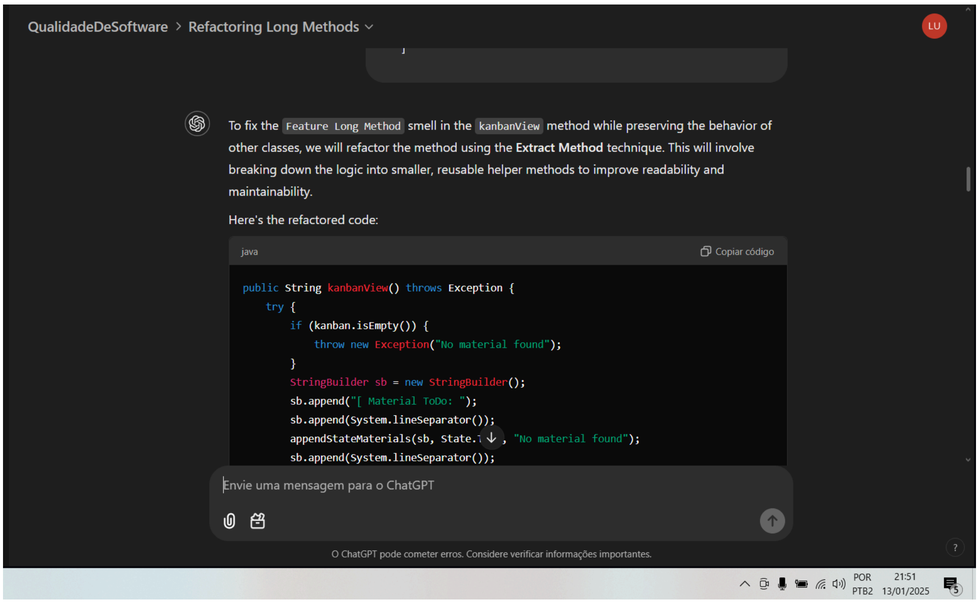
```
package org.example.studyplanner;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class KanbanView {
    public enum State{
        TODO, DOING, DONE;
    }

    HabitTracker habitTracker = null;
    TodoTracker todoTracker = null;
    Map<State, List<PlannerMaterial>> kanban = null;

    public KanbanView(HabitTracker habitTracker, TodoTracker
```

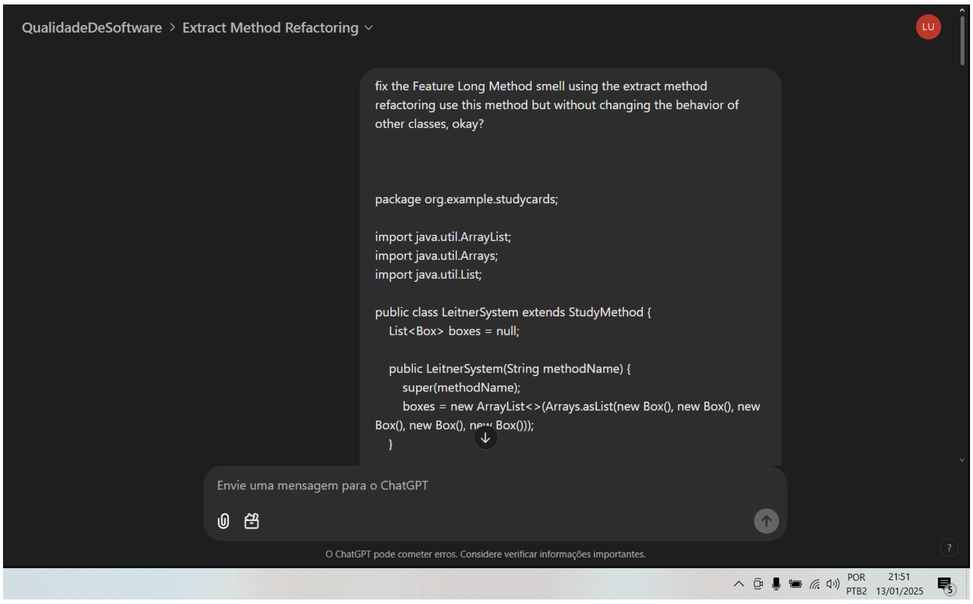

QualidadeDeSoftware > Refactoring Long Methods
LU
java
Copiar código
Envie uma mensagem para o ChatGPT
O ChatGPT pode cometer erros. Considere verificar informações importantes.
POR
PTB2
21:51
13/01/2025

To fix the `Feature Long Method` smell in the `kanbanView` method while preserving the behavior of other classes, we will refactor the method using the **Extract Method** technique. This will involve breaking down the logic into smaller, reusable helper methods to improve readability and maintainability.

Here's the refactored code:

```java
public String kanbanView() throws Exception {
    try {
        if (kanban.isEmpty()) {
            throw new Exception("No material found");
        }
        StringBuilder sb = new StringBuilder();
        sb.append("[ Material ToDo: ");
        sb.append(System.lineSeparator());
        appendStateMaterials(sb, State.T , "No material found");
        sb.append(System.lineSeparator());
```

# Long Method


QualidadeDeSoftware > Extract Method Refactoring
LU
Envie uma mensagem para o ChatGPT
O ChatGPT pode cometer erros. Considere verificar informações importantes.
POR
PTB2
21:51
13/01/2025

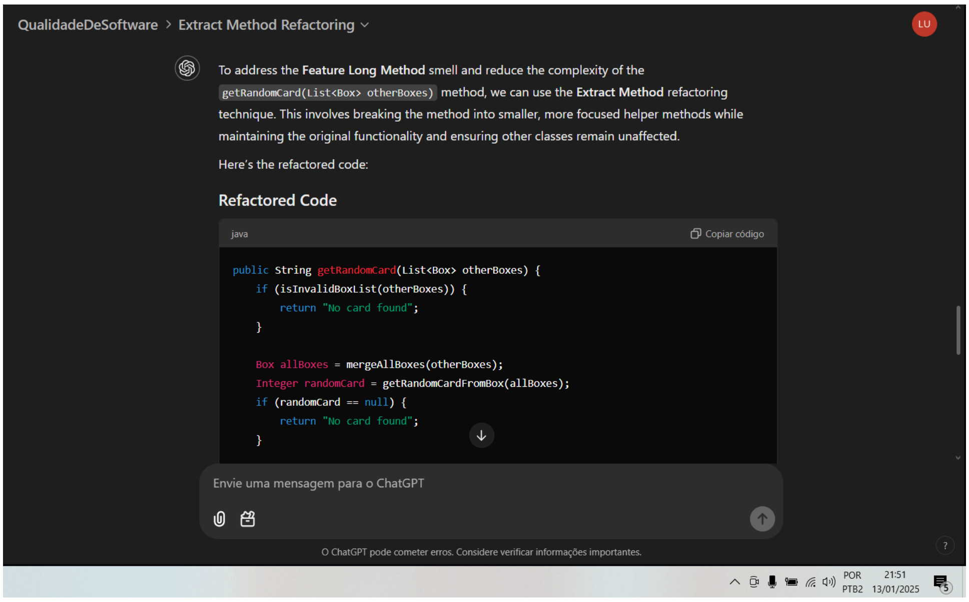
fix the Feature Long Method smell using the extract method refactoring use this method but without changing the behavior of other classes, okay?

```
package org.example.studycards;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class LeitnerSystem extends StudyMethod {
    List<Box> boxes = null;

    public LeitnerSystem(String methodName) {
        super(methodName);
        boxes = new ArrayList<>(Arrays.asList(new Box(), new Box(), new Box(), new Box(), new Box()));
    }
```


QualidadeDeSoftware > Extract Method Refactoring
LU
java
Copiar código
Envie uma mensagem para o ChatGPT
O ChatGPT pode cometer erros. Considere verificar informações importantes.
POR
PTB2
21:51
13/01/2025

To address the **Feature Long Method** smell and reduce the complexity of the `getRandomCard(List<Box> otherBoxes)` method, we can use the **Extract Method** refactoring technique. This involves breaking the method into smaller, more focused helper methods while maintaining the original functionality and ensuring other classes remain unaffected.

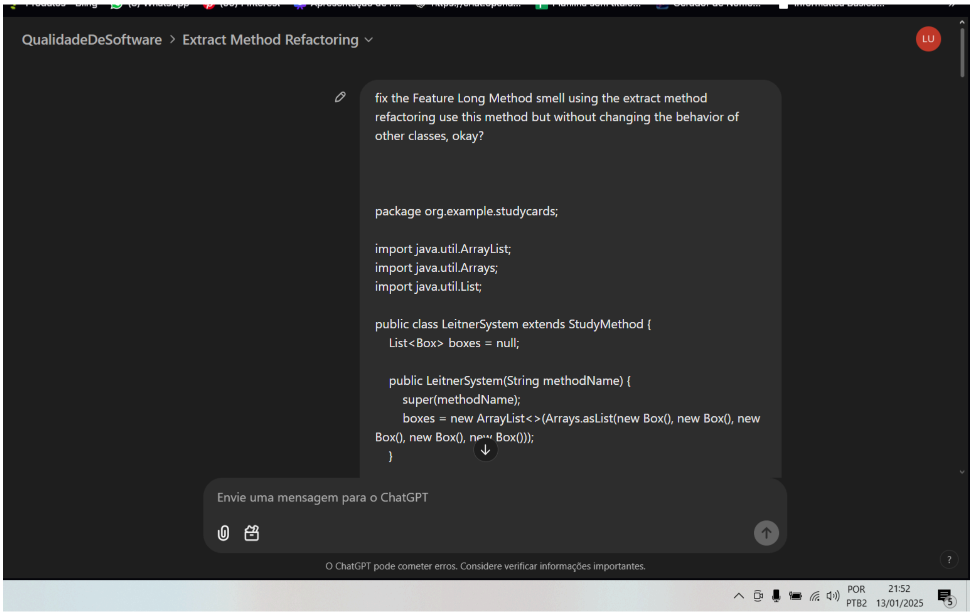
Here's the refactored code:

### Refactored Code

```java
public String getRandomCard(List<Box> otherBoxes) {
    if (isInvalidBoxList(otherBoxes)) {
        return "No card found";
    }

    Box allBoxes = mergeAllBoxes(otherBoxes);
    Integer randomCard = getRandomCardFromBox(allBoxes);
    if (randomCard == null) {
        return "No card found";
    }
```

## LongMethod


QualidadeDeSoftware > Extract Method Refactoring
LU
Envie uma mensagem para o ChatGPT
O ChatGPT pode cometer erros. Considere verificar informações importantes.
POR
PTB2
21:52
13/01/2025

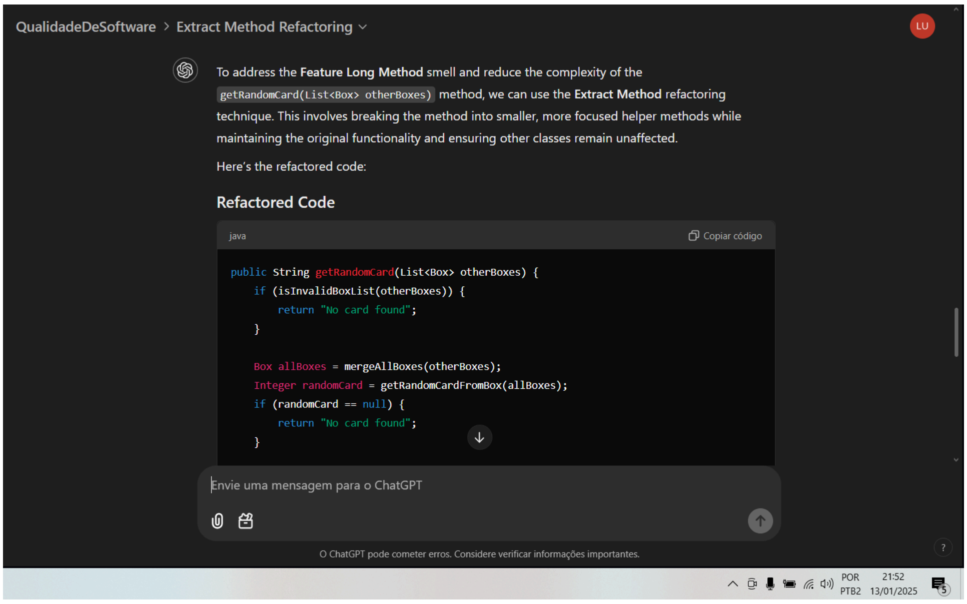
fix the Feature Long Method smell using the extract method refactoring use this method but without changing the behavior of other classes, okay?

```
package org.example.studycards;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class LeitnerSystem extends StudyMethod {
    List<Box> boxes = null;

    public LeitnerSystem(String methodName) {
        super(methodName);
        boxes = new ArrayList<>(Arrays.asList(new Box(), new Box(), new
Box(), new Box(), new Box()));
    }
```


QualidadeDeSoftware > Extract Method Refactoring
LU
java
Copiar código
Envie uma mensagem para o ChatGPT
O ChatGPT pode cometer erros. Considere verificar informações importantes.
POR
PTB2
21:52
13/01/2025

To address the **Feature Long Method** smell and reduce the complexity of the `getRandomCard(List<Box> otherBoxes)` method, we can use the **Extract Method** refactoring technique. This involves breaking the method into smaller, more focused helper methods while maintaining the original functionality and ensuring other classes remain unaffected.

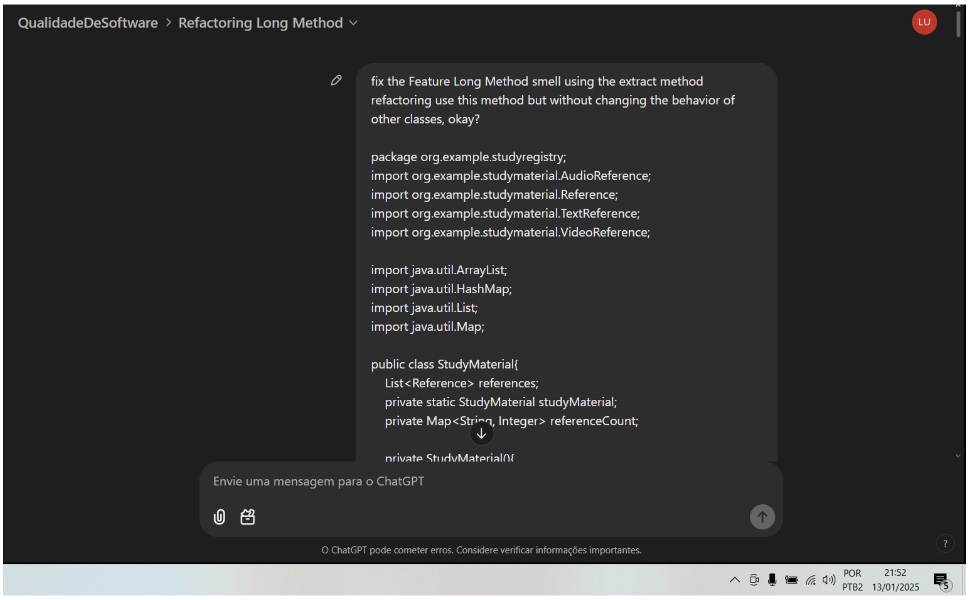
Here's the refactored code:

### Refactored Code

```java
public String getRandomCard(List<Box> otherBoxes) {
    if (isInvalidBoxList(otherBoxes)) {
        return "No card found";
    }

    Box allBoxes = mergeAllBoxes(otherBoxes);
    Integer randomCard = getRandomCardFromBox(allBoxes);
    if (randomCard == null) {
        return "No card found";
    }
```

## Long Method


QualidadeDeSoftware > Refactoring Long Method
LU
Envie uma mensagem para o ChatGPT
O ChatGPT pode cometer erros. Considere verificar informações importantes.
POR
PTB2
21:52
13/01/2025

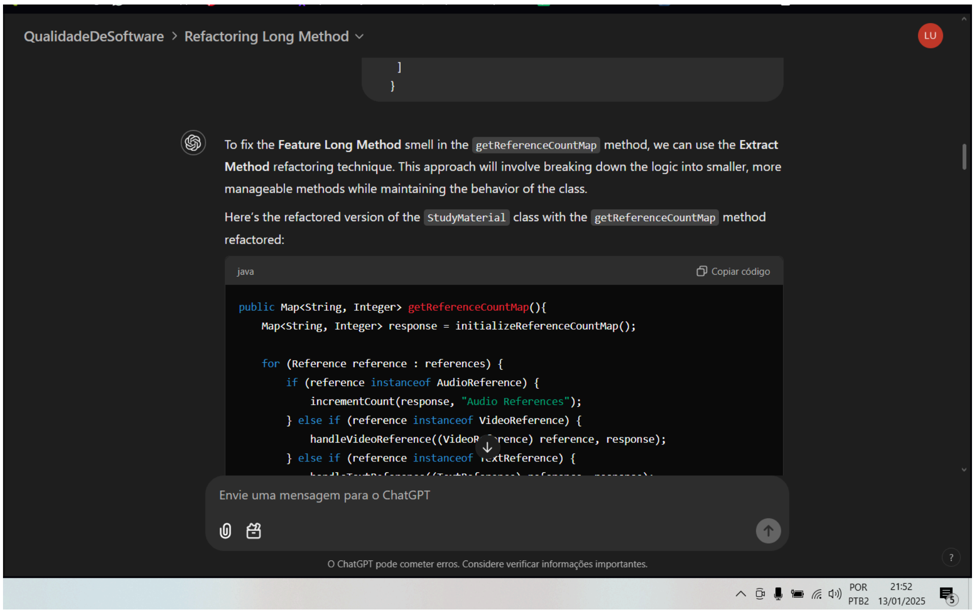
fix the Feature Long Method smell using the extract method refactoring use this method but without changing the behavior of other classes, okay?

```
package org.example.studyregistry;
import org.example.studymaterial.AudioReference;
import org.example.studymaterial.Reference;
import org.example.studymaterial.TextReference;
import org.example.studymaterial.VideoReference;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class StudyMaterial{
    List<Reference> references;
    private static StudyMaterial studyMaterial;
    private Map<String, Integer> referenceCount;

    private StudyMaterial(){
```


QualidadeDeSoftware > Refactoring Long Method
LU
java
Copiar código
Envie uma mensagem para o ChatGPT
O ChatGPT pode cometer erros. Considere verificar informações importantes.
POR
PTB2
21:52
13/01/2025

```
    ]
  }
```

To fix the **Feature Long Method** smell in the `getReferenceCountMap` method, we can use the **Extract Method** refactoring technique. This approach will involve breaking down the logic into smaller, more manageable methods while maintaining the behavior of the class.

Here's the refactored version of the `StudyMaterial` class with the `getReferenceCountMap` method refactored:

```java
public Map<String, Integer> getReferenceCountMap(){
    Map<String, Integer> response = initializeReferenceCountMap();

    for (Reference reference : references) {
        if (reference instanceof AudioReference) {
            incrementCount(response, "Audio References");
        } else if (reference instanceof VideoReference) {
            handleVideoReference((VideoReference) reference, response);
        } else if (reference instanceof TextReference) {
            handleTextReference((TextReference) reference, response);
```