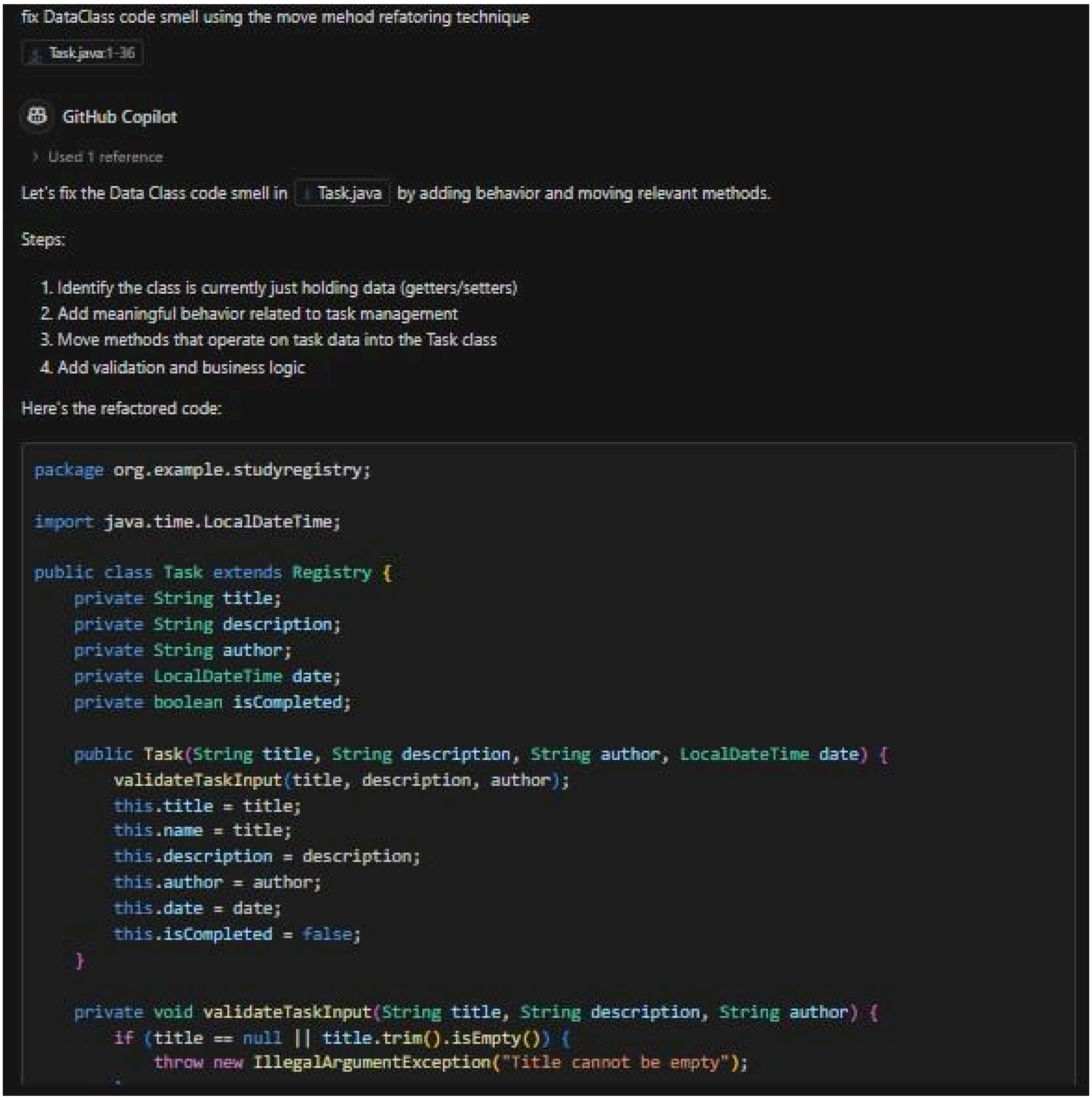


fix DataClass code smell using the move mehod refatoring technique

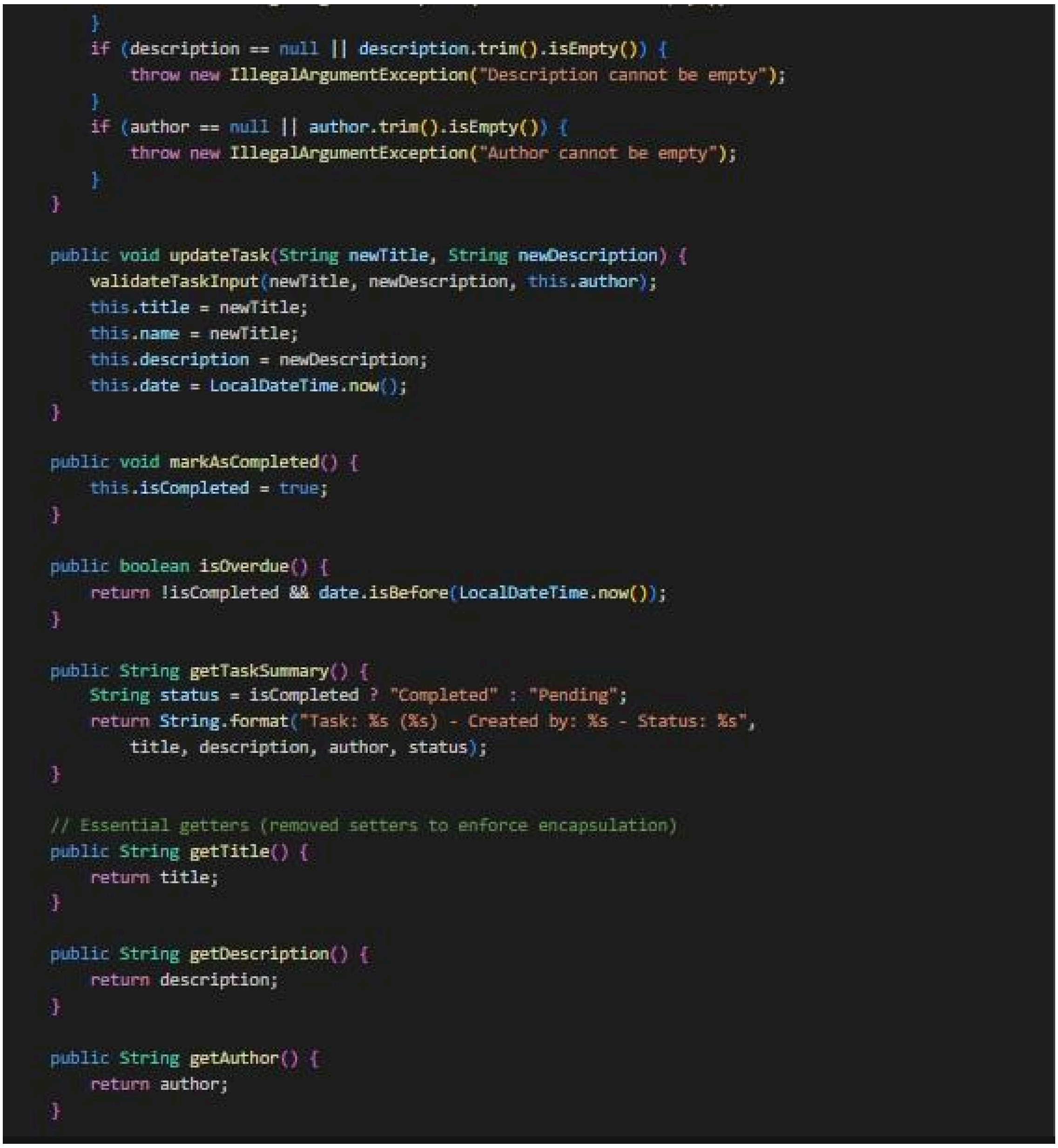
Let's fix the Data Class code smell in `Task.java` by adding behavior and moving relevant methods.

Steps:

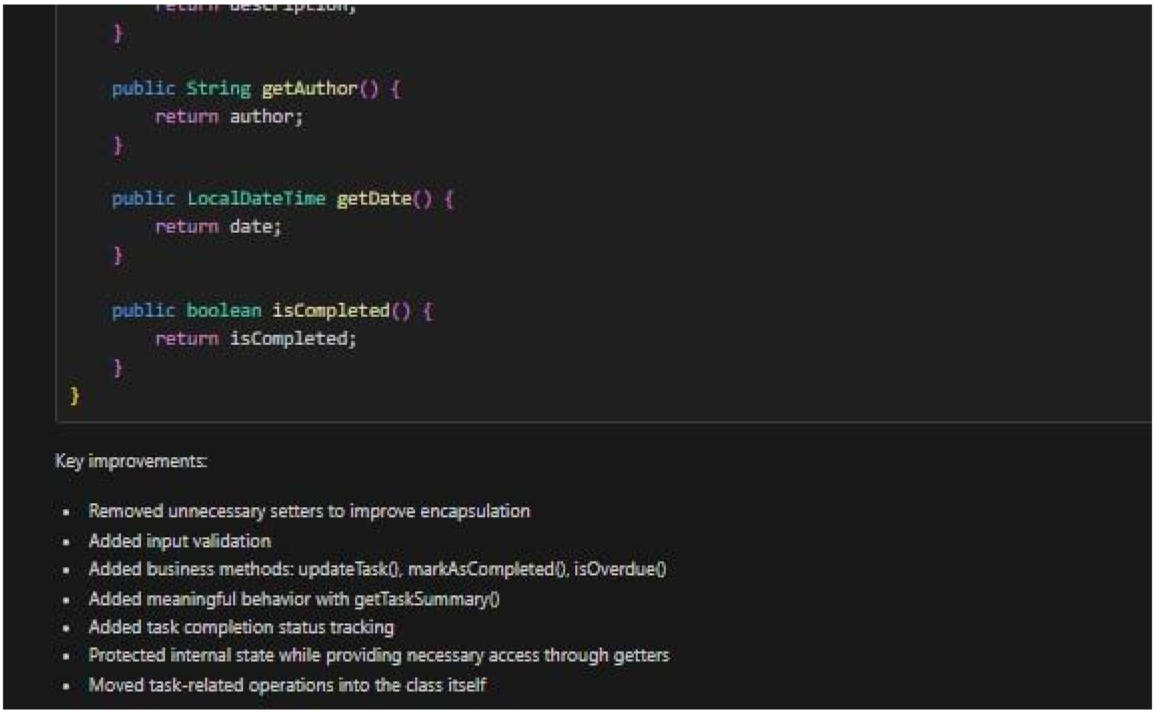
1. Identify the class is currently just holding data (getters/setters)
2. Add meaningful behavior related to task management
3. Move methods that operate on task data into the Task class
4. Add validation and business logic

Here's the refactored code:

```
package org.example.studyregistry;

import java.time.LocalDateTime;

public class Task extends Registry {
    private String title;
    private String description;
    private String author;
    private LocalDateTime date;
    private boolean isCompleted;

    public Task(String title, String description, String author, LocalDateTime date) {
        validateTaskInput(title, description, author);
        this.title = title;
        this.name = title;
        this.description = description;
        this.author = author;
        this.date = date;
        this.isCompleted = false;
    }

    private void validateTaskInput(String title, String description, String author) {
        if (title == null || title.trim().isEmpty()) {
            throw new IllegalArgumentException("Title cannot be empty");
```



```
        }
        if (description == null || description.trim().isEmpty()) {
            throw new IllegalArgumentException("Description cannot be empty");
        }
        if (author == null || author.trim().isEmpty()) {
            throw new IllegalArgumentException("Author cannot be empty");
        }
    }

    public void updateTask(String newTitle, String newDescription) {
        validateTaskInput(newTitle, newDescription, this.author);
        this.title = newTitle;
        this.name = newTitle;
        this.description = newDescription;
        this.date = LocalDateTime.now();
    }

    public void markAsCompleted() {
        this.isCompleted = true;
    }

    public boolean isOverdue() {
        return !isCompleted && date.isBefore(LocalDateTime.now());
    }

    public String getTaskSummary() {
        String status = isCompleted ? "Completed" : "Pending";
        return String.format("Task: %s (%s) - Created by: %s - Status: %s",
            title, description, author, status);
    }

    // Essential getters (removed setters to enforce encapsulation)
    public String getTitle() {
        return title;
    }

    public String getDescription() {
        return description;
    }

    public String getAuthor() {
        return author;
    }
```



```
        return description;
    }

    public String getAuthor() {
        return author;
    }

    public LocalDateTime getDate() {
        return date;
    }

    public boolean isCompleted() {
        return isCompleted;
    }
}
```

Key improvements:

- Removed unnecessary setters to improve encapsulation
- Added input validation
- Added business methods: updateTask(), markAsCompleted(), isOverdue()
- Added meaningful behavior with getTaskSummary()
- Added task completion status tracking
- Protected internal state while providing necessary access through getters
- Moved task-related operations into the class itself