**Change request log: #3**

**1.Team:** Specify the team members.

* Nazeera Siddiqui (Scripter)
* John Scalley (Code)

**2.Change Request**

Provide the id and description of the change request.

* Change Request # 3 (Merge module)

**3.Concept Location**

Use the table below to describe each step you follow when performing concept location for this change request. In your description, include the following information when appropriate:

* IDE Features used (e.g., searching tool, dependency navigator, debugging, etc.)
* Queries used when searching
* System executions and input to the system
* Interactions with the system (e.g., pages visited)
* Classes visited
* The first class found to be changed (this is when concept location ends)

When there is a major decision/step in the process, include its rationale, i.e., why that decision/step was taken.

Make sure you time yourselves when going through this process and provide the total time spent below.

The following is an example of a concept location process for the change request "Color student schedule":

|  |  |  |
| --- | --- | --- |
| **Step #** | **Description** | **Rationale** |
| **1** | *We located the MergeModule.java file* | *Since this change request involves the merge feature, we wanted to start with the MergeModule.java file* |
| **2** | *We opened the BaseTaskExecutionModule.java* | *MergeModule class extends the BaseTaskExecutionModule class* |
| **3** | *We opened Module.java* | *Because BaseTaskExecutionModule class implements Module* |
| **4** | *We returned to BaseTaskExecutionModule.java* | *We returned to this class because it contains an errorTracker in the function initModuleSettingsPanel(). We determined that this is where the notification is displayed for the user regarding the invalid parameters.* |
| **5** | *We opened TaskExecutionRequestEvent.java* | *We noticed that one of the imports in the BaseTaskExecutionModule.java file.* |
| **6** | *We ran the system* | *We could not locate the appropriate files, so we wanted to view the log to see what information it could provide regarding the errors.* |
| **7** | *We opened TaskExecutionController.java* | *The log mentioned this file.* |
| **8** | *We searched for DefaultTaskExecutionService.java in the pdfsam directory* | *This file was mentioned in the log as well.* |
| **9** | *We looked for the code for sejda* | *We determined that the changes would need to be made for the sejda sdk as pdfsam relies upon this code.* |

**Time spent (in minutes):** 184

**4.Impact Analysis**

Use the table below to describe each step you follow when performing impact analysis for this change request. Include as many details as possible, including why classes are visited or why they are discarded from the estimated impact set.

Do not take the impact analysis of your changes lightly. Remember that any small change in the code could lead to large changes in the behavior of the system. Follow the impact analysis process covered in the class. Describe in detail how you followed this process in the change request log. Provide details on how and why you finished the impact analysis process.

|  |  |  |
| --- | --- | --- |
| **Step #** | **Description** | **Rationale** |
| **1** | *We considered the reason a user may want to have the ability to have intersecting pages.* | *To understand the implementation of the change request. If we allow for intersecting pages, the user may not expect this. We could possibly display a popup instead of throwing an invalid parameters exception.* |
| **2** | *We inspected the class DefaultTaskExecutionService.* | *We realized this class was necessary to understand to figure out where to make the change.* |
| **3** | *We eliminated the classes we had examined from pdfsam as they did not determine the constraints.* | *The constraints are determined in sejda sdk* |

**Time spent (in minutes):** 42

**6.Actualization**

Use the table below to describe each step you followed when changing the code. Include as many details as possible, including why classes/methods were modified, added, removed, renamed, etc.

|  |  |  |
| --- | --- | --- |
| **Step #** | **Description** | **Rationale** |

**Time spent (in minutes):**

**8.Validation**

Use the table below to describe any validation activity (e.g., testing, code inspections, etc.) you performed for this change request. Include the description of each test case, the result (pass/fail) and its rationale.

|  |  |  |
| --- | --- | --- |
| **Step #** | **Description** | **Rationale** |

**Time spent (in minutes):**

**9.Timing**

Summarize the time spent on each phase.

|  |  |
| --- | --- |
| **Phase Name** | **Time (in minutes)** |
| Concept location | 184 |
| Impact Analysis | 42 |
| Prefactoring |  |
| Actualization |  |
| Postfactoring |  |
| Verification |  |
| **Total** | **226** |

**10.Reverse engineering**

Create a UML sequence diagram (or more if needed) corresponding to the main object interactions affected by your change.

Create a partial UML class diagram of the classes visited while navigating through the code. Include the associations between classes (e.g., inheritance, aggregations, compositions, etc.), as well as the important fields and methods of each class that you learn about. The diagram may have disconnected components. Use the UML tool of your preference. When a significant fact about a class or method is learned, indicate it via annotations on the diagram. For each change request, start with the diagram produced in the previous change request. **For the first, you will start from scratch.**

**11.Conclusions**

For this change, the concept location took longer than we expected. We did not initially consider that the necessary files were not in the pdfsam directory. Hence, only concept location and impact analysis steps were completed as we spent a lot of time trying to locate files within the pdfsam directory.  Testing was not performed because we were not able to complete the change because it was located in a different sdk.



