LATEX EDITOR PROJECT

PHASE 1 - REVERSE ENGINEERING AND QUALITY ASSESSMENT

	Date 9/30/2019

GOAL OF THE PROJECT

The goal of this project is to reengineer a Java application. At a glance, the objective of this project is to develop a simple Latex editor for inexperienced Latex users. Latex is a well known high quality document preparation markup language. It provides a large variety of styles and commands that enable advanced document formatting. Typically, a Latex document is compiled with a tool like MikTex, Lyx, etc. to produce a respective formatted document in pdf, ps, etc. Formatting documents with Latex is like a programming process as it involves the proper usage of Latex commands which are embedded in the document contents. The goal of the Latex editor is to facilitate the usage of Latex commands for the preparation of Latex documents. One of the prominent features that distinguishes the LatexEditor from other similar applications is its multi-strategy version tracking functionalities that enable undo and redo actions.

PHASE 1 TASK LIST

1. **[Skim the documentation]** The legacy application has been developed based on a more detailed requirements specification that is available along with the application source code (LatexEditorRequirementsDefinition-2019.pdf). The design of the application borrows ideas and employs certain design patterns, explained in another document (GuidelinesAndHints-2019-v0.pdf). In a first step, study the documentation to get more information concerning the application's architecture and use cases.

2. **[Do a mock installation]** The application source code is provided as an eclipse project (LatexEditorProject folder). Setup a running version of the project and test it.

3. **[Build confidence]** Read all the source code once and try to understand the legacy architecture, the role/responsibilities of each class, and so on.

4. **[Capture the design]** Specify the application architecture in terms of a UML package diagram. Specify the detailed design in terms of UML class diagrams. Prepare CRC cards that describe the responsibilities and collaborations of each class.

5. **[Identify problems]** Assess the quality of the code. Look for general problems

    i. Detect possibilities of problematic classes with many responsibilities.

    ii. Detect possibilities of problematic classes with very few responsibilities.

iii. Detect possibilities of similar classes/methods with duplicated code.

6. **[Prepare report]** prepare a detailed report based on the given template (Project-Deliverable-Phase1.doc).