National University
of computer and emerging sciences

Project Phase #1

Course:

Software Construction and Development

Topic:

Arabic Text Editor

Submitted by:

Roll No.	Name
22F-3706	Huzaifa Asif
22F-3708	Syed Muhammad Taqi
22F-3722	Muhammad Abdullah

Submitted to:

Dr. Affan Ruaf

Contents

User Stories:

1. Develop a Text Editor application in JAVA for only text files (.txt).
2. Import Markdown Renderer in JAVA.
3. The software will initially be a desktop application.
4. It will further be integrated into a web-based interface based on future requirements.
5. The System will have two modes:
   - Read Mode
   - Edit Mode: Perform all basic CRUD operations
6. The System will have Arabic, Urdu, and Farsi as their primary language.
7. The System will use the Encoding UTF-8 case-insensitive version.
8. The System will have all basic formatting from Markdown
9. The System will have basic pre-defined colors (rainbow) by using Markdown.
10. All the raw text files will be saved in a database (relational database) using MariaDB.
11. After opening the System will import all the files from the database only once through multi-threading in the form of a list.
12. All the changes will occur in files saved in the database and the raw file will remain unchanged.
13. After editing the files can be exported from the database onto the computer using Markdown.
14. The System will show a list of all recent files opened by the owner in the database.
15. The files will be sorted alphabetically by default but can also be sorted based on the last modified date and the created date.
16. The user can search the file in the database and text in the file itself.
17. The user can find and replace any desired text/word.
18. The System will autosave the text file through multi-threading.
19. The System will be able to transliterate to Roman English.
20. The System will auto-suggest/complete words to the user after typing up to 3 characters.
21. All the files opened in the System will be paginated and not scrollable.
22. The user will be able to zoom in or out the text (based on font size).
23. The System will have all the basic shortcuts to perform functions like copy, cut, paste, etc.

24. The owner of the file will be able to read as well as edit the text whereas the rest of the users will only be able to view/read.
25. The owner will be determined through the database user.
26. The System will perform data analytics and statistics operations mainly TFIDF (Term Frequency-Inverse Document Frequency) on the entire database.
27. The database will be normalized up to 3NF.
28. The UI of the System should be simple and elegant with neutral colors.
29. The database will prevent duplication of files through Hash calculation using MD5, and Sha1.
30. The hash will be calculated based on the content of the file when it is first imported into the database.
31. The original hash calculated will not change even after editing the file.
32. The front end of the System application will be developed in JAVA.

Functional Requirements:

Functional Requirements	**Description**
FR1	The system allows user to Read and Edit files.
FR2	The user shall perform basic CRUD operations on the file in Edit Mode.
FR3	System saves raw text files in a database.
FR4	The user shall import files into the database.
FR5	The user shall export files from the database to the computer.
FR6	The system shall display recent files opened by the user.
FR7	The user can sort files alphabetically by default.
FR8	The user can search files in the database and text within files.
FR9	The user shall find and replace text.
FR10	The system shall autosave text files.
FR11	The system shall perform transliteration to Roman English for words
FR12	The system shall auto-complete words after typing/searching up to three characters.
FR13	The user shall have a user profile (based on the database).
FR14	The system shall have paginated opened files
FR15	The system shall perform basic shortcuts for copying, cutting, and pasting.
FR16	The System shall provide TF-IDF analysis on the database.
FR17	The system shall have a home page
	Description: • User shall create a new document. • User shall open an existing file • User shall import/export files • User shall open recently opened files • System shall display all files in DB • System shall display recently opened files alphabetically according to modified date, and created date.
FR18	The System will prevent file duplication using hash calculation.

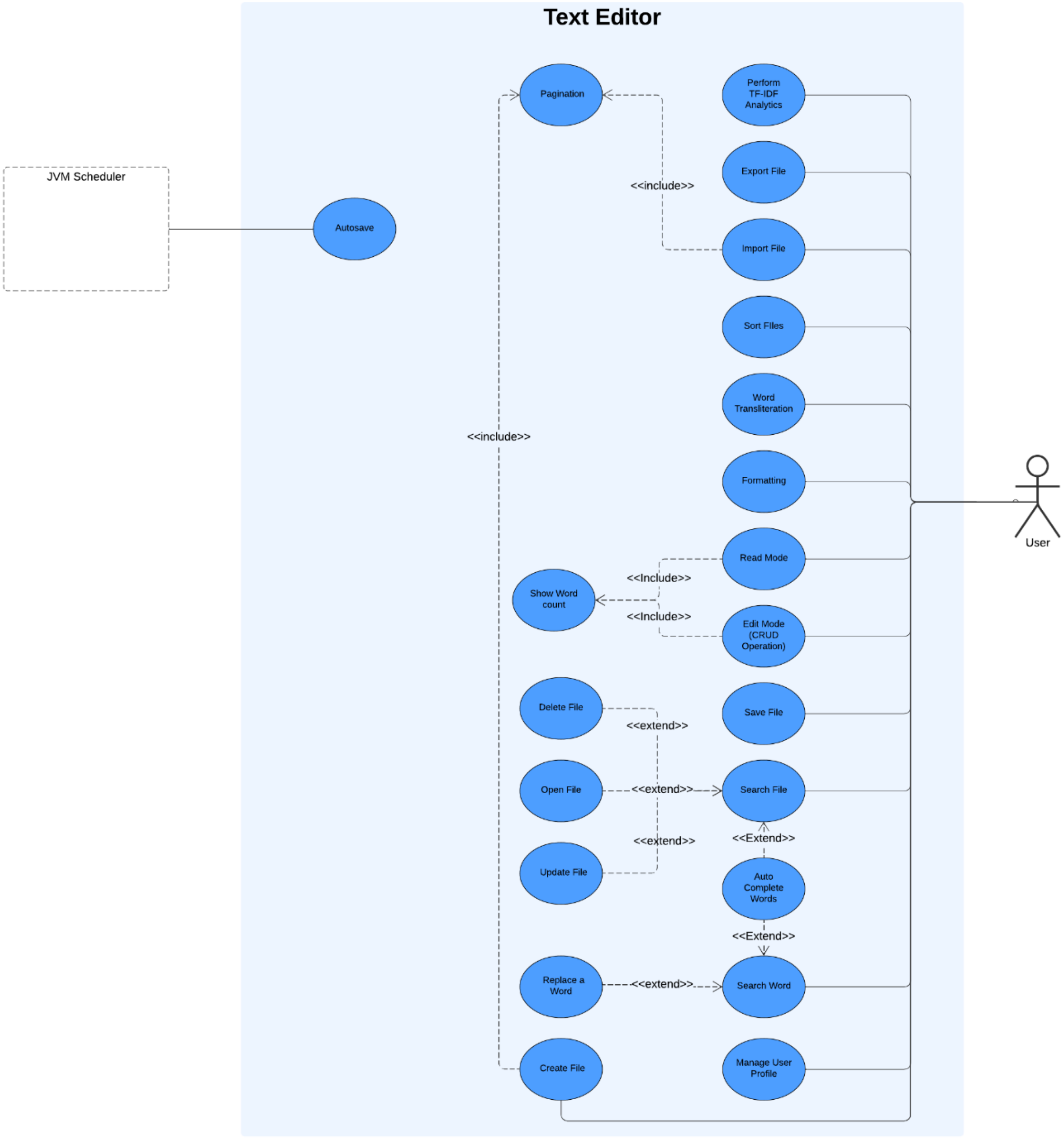
Non-Functional Requirements:

Non-Functional Requirements	Description
NFR1	The software will initially be a desktop application.
NFR2	The software will be integrated into a web-based interface based on future requirements.
NFR3	The System will use multi-threading for file import and autosave.
NFR4	The database will be normalized up to 3NF.
NFR5	The database will prevent duplication of files through a hash calculation using MD5 and SHA1.
NFR6	System will have Multithreading (asynchronous operations).
NFR7	The System calculates hash based on file content when imported.
NFR8	Original Hash must not change.

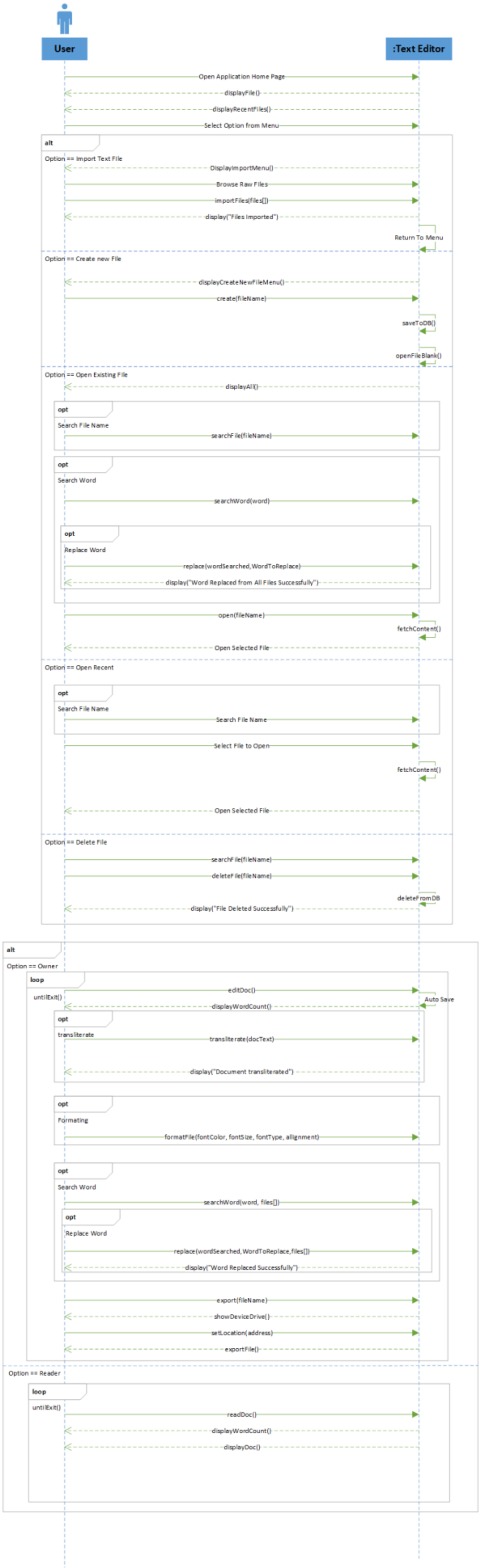
Quality Constraints:

Quality Constraints:	Description
QC1	The System will use UTF-8 encoding in a case-insensitive manner.
QC2	The owner has read and edit permissions.
QC3	The system will support Arabic, Urdu, and Persian languages.
QC4	The System provides pre-defined rainbow colors using Markdown.
QC5	The System will use multi-threading for performance optimization.
QC6	The System will have Maria DB for storage purposes
QC7	The hash will be calculated based on the content of the file when it is first imported into the database.
QC8	The original hash calculated will not change even after editing the file.
QC9	The UI should be simple and elegant with neutral colors.

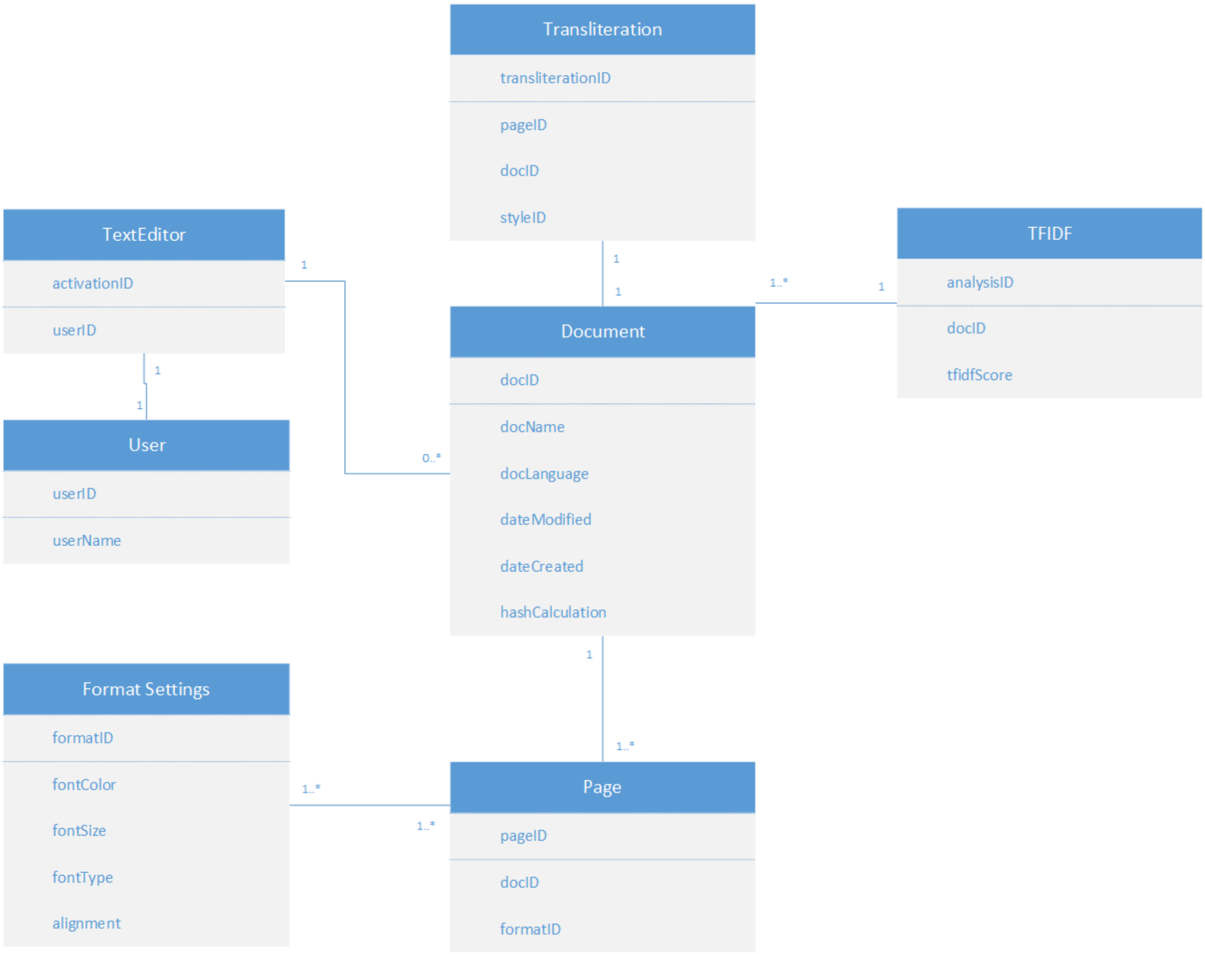
Use Case Diagram:



System Sequence Diagram:


User
:Text Editor
Open Application Home Page
displayFile()
displayRecentFiles()
Select Option from Menu
alt
Option == Import Text File
DisplayImportMenu()
Browse Raw Files
importFiles(files[])
display("Files Imported")
Return To Menu
Option == Create new File
displayCreateNewFileMenu()
create(fileName)
saveToDB()
openFileBlank()
Option == Open Existing File
displayAll()
opt
Search File Name
searchFile(fileName)
opt
Search Word
searchWord(word)
opt
Replace Word
replace(wordSearched,WordToReplace)
display("Word Replaced from All Files Successfully")
open(fileName)
fetchContent()
Open Selected File
Option == Open Recent
opt
Search File Name
Search File Name
Select File to Open
fetchContent()
Open Selected File
Option == Delete File
searchFile(fileName)
deleteFile(fileName)
deleteFromDB
display("File Deleted Successfully")
alt
Option == Owner
loop
untilExit()
editDoc()
Auto Save
displayWordCount()
opt
transliterate
transliterate(docText)
display("Document transliterated")
opt
Formating
formatFile(fontColor, fontSize, fontType, allignment)
opt
Search Word
searchWord(word, files[])
opt
Replace Word
replace(wordSearched,WordToReplace,files[])
display("Word Replaced Successfully")
export(fileName)
showDeviceDrive()
setLocation(address)
exportFile()
Option == Reader
loop
untilExit()
readDoc()
displayWordCount()
displayDoc()

Domain Model:


Transliteration
transliterationID
pageID
docID
styleID
TextEditor
activationID
userID
TFIDF
analysisID
docID
tfidfScore
Document
docID
docName
docLanguage
dateModified
dateCreated
hashCalculation
User
userID
userName
Format Settings
formatID
fontColor
fontSize
fontType
alignment
Page
pageID
docID
formatID
1
1
1
1..*
1
1
0..*
1
1
1
1..*
1..*
1..*