2.1: Record or Tasks

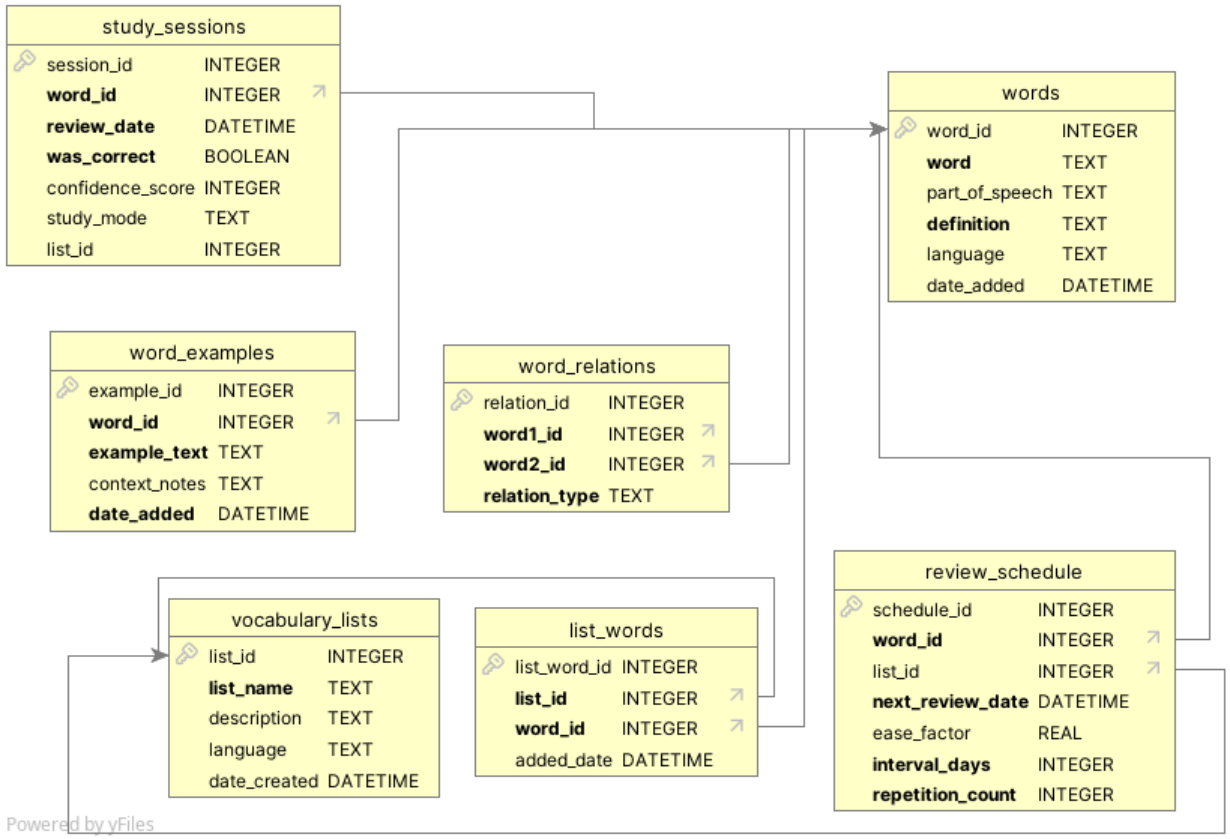
Task #	Date	Phase	Description	Duration
1	Nov 21	Planning	First interview to establish problem and gather requirements from client	10 minutes
2	Nov 29	Planning	Finalized success criteria according to the first interview with client.	30 minutes
3	Nov 30	Development	Set up git, github, Qt Creator, and core project structure for program	30 minutes
4	Dec 4	Development	Added sqlite3 as a library to the project. Ran into several errors due to existing library conflicts, linkage errors, and syntax	3 hours
5	Dec 7	Development	Finished database class constructor. Ran into some SQL syntax errors, but the added error handling made it trivials	1 hour
6	Dec 8	Design	Designed UI for the different windows in my application. Then implemented it into code without any actual back-end functionality	1 hour
7	Dec 8	Development	Began implementing the ability to add vocabulary lists. Ran into several issues with pointers and being able to hand over a member variable of a class	2 hours
8	Dec 9	Development	Finished implementing adding vocabulary lists. Ran into more issues with class variables and pointers. Resolved by making the database variable a pointer	1 hour
9	Dec 10	Development	Began adding the ability to add words to a list. Also began adding the supermemo algorithm.	1 hour
10	Dec 14	Design Planning	Added more structure to the project directory. Made sure everything uploaded to github correctly	1 hour

Task #	Date	Phase	Description	Duration
11	Dec 15	Development	Added functionality for studying words such as displaying lists, getting the words, etc. Ran into issue making a new window as Qt has specifications within the file that need to be met and I forgot about them.	3 hours
12	Dec 16	Development	Added more than one study method. Ran into issue when switching between multiple panels in the main window of the project.	2 hours
13	Dec 16	Development	Ensured the easiness factor was always in the accepted range so that it wouldn't throw an error when attempting to add to the sql database	30 minutes
14	Dec 17	Development	Added the option to view all the words and definitions in a list	1 hour
15	Dec 17	Development	Began adding the ability to use ai for the creation of a list. Ran into issues with using environmental variables for the api key. Also added the ability to delete a list.	3 hours
16	Dec 18	Development	Added two distinct themes for the user to be able to use. Ran into issues with style sheets because of the new syntax. Took some rereading to get it to work again	2 hours
17	Dec 18	Development	Worked on making sure the user can't add multiple of the same word to the same list	1 hour
18	Dec 18	Development	Cleaned up the ui so it wouldn't look so jagged. Again had to look into how to use style sheets and different Qt methods to get everything to work smoothly	1 hour
19	Dec 19	Development	Tried to work more API keys for ai. Could not get google gemini to work	1 hour

Task #	Date	Phase	Description	Duration
			and OpenAI is paid so I left it for the future	
20	Dec 26	Development	Added another study method for the user. Ran into issues displaying whether the user was correct. Figured it out eventually, it an issue with order of the code	2 hours
21	Dec 26	Development	Made sure everything displayed was in the correct order, including the ui. Fixed a bug where the ui was inconsistent when starting the application	1 hour
22	Dec 26	Development	Made sure bug/error handling is done through native Qt methods to make sure they work all the time	30 minutes
23	Dec 31	Design	Worked on UML diagram for all of the classes in my program	1 hour
24	Jan 1	Design	Added the ERD diagram of the sql database after generating it with Database Visualizer	30 minutes

2.2: Data Structures

ERD Diagram



SQLite database tables

vocabulary_lists

Column Name	Data Type	Description
list_id	Integer	Unique identifier for each vocabulary list
list_name	Text	The name of each vocabulary list
description	Text	The description of the list
language	Text	The language studied
date_created	DATETIME	When the list was created

list_words

Column Name	Data Type	Description
list_word_id	Integer	The id of the connection between a word and a list
list_id	Integer	The id of the list the word is connected
word_id	Integer	The id of the word that is connected to the list
added_date	DATETIME	The date when the word was added

word_examples

Column Name	Data Type	Description
example_id	Integer	The id of the example
word_id	Integer	The id of the word that the example is connected to
example_text	Text	The example itself
context_notes	Text	Any notes on the example
date_added	DATETIME	When the example was created

word_relations

Column Name	Data Type	Description
relation_id	Integer	The id of the relation
word1_id	Integer	The first word in the relation
word2_id	Integer	The word with which the initial word is related
relation_type	Text	The relation, eg. synonym

study_sessions

Column Name	Data Type	Description
session_id	Integer	The id of the study session
word_id	Integer	The id of the word studied
review_date	DATETIME	The date when studied
was_correct	Boolean	Whether the user was correct when studying
response_time	Integer	How long it took the user to answer
confidence_score	Integer	The confidence score for the supermemo algorithm
study_mode	Text	The study mode the user chose
list_id	Integer	The list from which the word came from

words

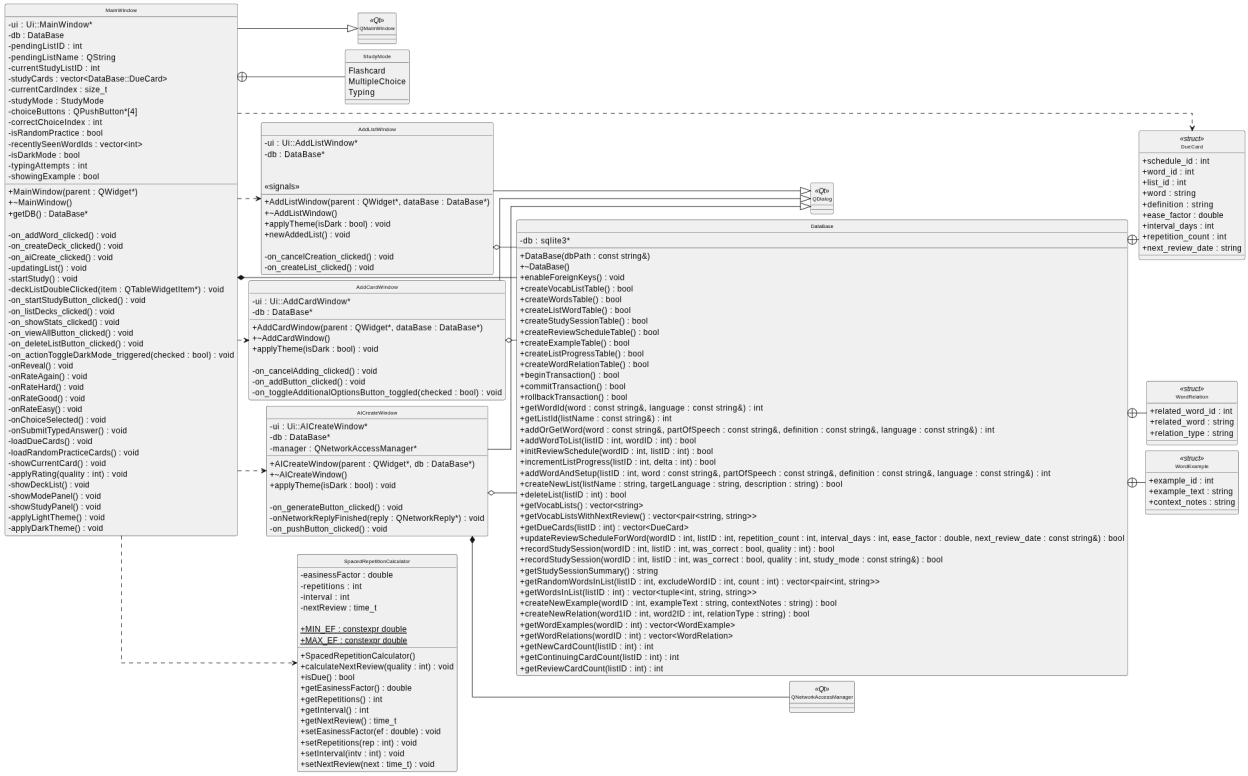
Column Name	Data Type	Description
word_id	Integer	The id of the word
word	Text	The word itself
part_of_speech	Text	The part of speech the word is
definition	Text	The definition of the word
language	Text	The language the word is studied
date_added	DATETIME	The date the word was added

review_schedule

Column Name	Data Type	Description
schedule_id	Integer	The id of the next scheduled time for review
word_id	Integer	The word to be studied
list_id	Integer	The list from which the word comes
next_review_date	DATETIME	The next date for review
ease_factor	Real	The ease factor based on user response, used to calculate next review date
interval_days	Integer	The number of days in between studies
repetition_count	Integer	The number of times a word has been repeated

2.4: Algorithms:

UML Diagram


MainWindow
-ui : Ui::MainWindow*
-db : DataBase
-pendingListID : int
-pendingListName : QString
-currentStudyListID : int
-studyCards : vector<DataBase::DueCard>
-currentCardIndex : size_t
-studyMode : StudyMode
-choiceButtons : QPushButton*[4]
-correctChoiceIndex : int
-isRandomPractice : bool
-recentlySeenWordIds : vector<int>
-isDarkMode : bool
-typingAttempts : int
-showingExample : bool
+MainWindow(parent : QWidget*)
+~MainWindow()
+getDB() : DataBase*
-on_addWord_clicked() : void
-on_createDeck_clicked() : void
-on_aiCreate_clicked() : void
-updatingList() : void
-startStudy() : void
-deckListDoubleClicked(item : QTableWidgetItem*) : void
-on_startStudyButton_clicked() : void
-on_listDecks_clicked() : void
-on_showStats_clicked() : void
-on_viewAllButton_clicked() : void
-on_deleteListButton_clicked() : void
-on_actionToggleDarkMode_triggered(checked : bool) : void
-onReveal() : void
-onRateAgain() : void
-onRateHard() : void
-onRateGood() : void
-onRateEasy() : void
-onChoiceSelected() : void
-onSubmitTypedAnswer() : void
-loadDueCards() : void
-loadRandomPracticeCards() : void
-showCurrentCard() : void
-applyRating(quality : int) : void
-showDeckList() : void
-showModePanel() : void
-showStudyPanel() : void
-applyLightTheme() : void
-applyDarkTheme() : void
«Qt»
QMainWindow
StudyMode
Flashcard
MultipleChoice
Typing
AddListWindow
-ui : Ui::AddListWindow*
-db : DataBase*
«signals»
+AddListWindow(parent : QWidget*, dataBase : DataBase*)
+~AddListWindow()
+applyTheme(isDark : bool) : void
+newAddedList() : void
-on_cancelCreation_clicked() : void
-on_createList_clicked() : void
«Qt»
QDialog
AddCardWindow
-ui : Ui::AddCardWindow*
-db : DataBase*
+AddCardWindow(parent : QWidget*, dataBase : DataBase*)
+~AddCardWindow()
+applyTheme(isDark : bool) : void
-on_cancelAdding_clicked() : void
-on_addButton_clicked() : void
-on_toggleAdditionalOptionsButton_toggled(checked : bool) : void
AICreateWindow
-ui : Ui::AICreateWindow*
-db : DataBase*
-manager : QNetworkAccessManager*
+AICreateWindow(parent : QWidget*, db : DataBase*)
+~AICreateWindow()
+applyTheme(isDark : bool) : void
-on_generateButton_clicked() : void
-onNetworkReplyFinished(reply : QNetworkReply*) : void
-on_pushButton_clicked() : void
SpacedRepetitionCalculator
-easinessFactor : double
-repetitions : int
-interval : int
-nextReview : time_t
+MIN_EF : constexpr double
+MAX_EF : constexpr double
+SpacedRepetitionCalculator()
+calculateNextReview(quality : int) : void
+isDue() : bool
+getEasinessFactor() : double
+getRepetitions() : int
+getInterval() : int
+getNextReview() : time_t
+setEasinessFactor(ef : double) : void
+setRepetitions(rep : int) : void
+setInterval(intv : int) : void
+setNextReview(next : time_t) : void
DataBase
-db : sqlite3*
+DataBase(dbPath : const string&)
+~DataBase()
+enableForeignKeys() : void
+createVocabListTable() : bool
+createWordsTable() : bool
+createListWordTable() : bool
+createStudySessionTable() : bool
+createReviewScheduleTable() : bool
+createExampleTable() : bool
+createListProgressTable() : bool
+createWordRelationTable() : bool
+beginTransaction() : bool
+commitTransaction() : bool
+rollbackTransaction() : bool
+getWordId(word : const string&, language : const string&) : int
+getListId(listName : const string&) : int
+addOrGetWord(word : const string&, partOfSpeech : const string&, definition : const string&, language : const string&) : int
+addWordToList(listID : int, wordID : int) : bool
+initReviewSchedule(wordID : int, listID : int) : bool
+incrementListProgress(listID : int, delta : int) : bool
+addWordAndSetup(listID : int, word : const string&, partOfSpeech : const string&, definition : const string&, language : const string&) : int
+createNewList(listName : string, targetLanguage : string, description : string) : bool
+deleteList(listID : int) : bool
+getVocabLists() : vector<string>
+getVocabListsWithNextReview() : vector<pair<string, string>>
+getDueCards(listID : int) : vector<DueCard>
+updateReviewScheduleForWord(wordID : int, listID : int, repetition_count : int, interval_days : int, ease_factor : double, next_review_date : const string&) : bool
+recordStudySession(wordID : int, listID : int, was_correct : bool, quality : int) : bool
+recordStudySession(wordID : int, listID : int, was_correct : bool, quality : int, study_mode : const string&) : bool
+getStudySessionSummary() : string
+getRandomWordsInList(listID : int, excludeWordID : int, count : int) : vector<pair<int, string>>
+getWordsInList(listID : int) : vector<tuple<int, string, string>>
+createNewExample(wordID : int, exampleText : string, contextNotes : string) : bool
+createNewRelation(wordID : int, word2ID : int, relationType : string) : bool
+getWordExamples(wordID : int) : vector<WordExample>
+getWordRelations(wordID : int) : vector<WordRelation>
+getNewCardCount(listID : int) : int
+getContinuingCardCount(listID : int) : int
+getReviewCardCount(listID : int) : int
«Qt»
QNetworkAccessManager
«struct»
DueCard
+schedule_id : int
+word_id : int
+list_id : int
+word : string
+definition : string
+ease_factor : double
+interval_days : int
+repetition_count : int
+next_review_date : string
«struct»
WordRelation
+related_word_id : int
+related_word : string
+relation_type : string
«struct»
WordExample
+example_id : int
+example_text : string
+context_notes : string