[**Software Maintenance and Evolution ‬‏**](https://classroom.google.com/c/NjYyNjE5OTQ2Mjgz)**Project**

**Phase 1**

Names and ID’s:

|  |  |
| --- | --- |
| **Name** | **ID** |
| Ahmed Waleed Mohamed | 20216017 |
| Moaaz Amr Mohamed | 20216099 |
| Omar Salah | 20206133 |
| Mohand aly ahmed | 20216104 |
| Mostafa Ayman | 20206114 |

1- GitHub repository:

https://github.com/AhmedWaleedKalifa/Maintenance-project

2- Description of what the project does:

* General Description:

jMemorize is an open-source Java application that manages your

learning processes by using flashcards and the famous Leitner

system.

It makes memorizing facts not only more efficient but also more fun. It boosts your whole learning experience and features categories, statistics and a visually appealing and intuitive interface.

jMemorize manages your learn progress and features categories,

statistics and a visually appealing and intuitive interface.

It supported learning a foreign language (like english), and also learning historical dates and mathematical formulas.

1) Allot of flash card learning systems feature either aesthetically unpleasing or too unusual graphical user interfaces. jMemorize uses a interface that orientates itself on already matured eMail-clients where organizing information is already well explored. Additionally jMemorize features an aesthetically appealing interface by using icons, visual statistics and appropriate use of layouts.

2) jMemorize allows to organize Cards in a tree category hierarchy where every card of a subcategory also belongs to its father categories. This allows users to organize their cards and quickly change between the card subsets that should be learned right now.

3) jMemorize uses modern object-oriented design that facilitates future developments. Editing flashcards or categories fires events which are independently handled by the GUI, allowing to add new interface elements quickly and keeping the program flow intuitive. Clarity is taken very serious and the code is refactored continuously.

It brings the much anticipated learn settings window and allows you to customize your learning pattern in nearly every way imaginable. You can limit your sessions by time and/or card limits,

group your cards by categories, set custom time plans, learn in reversed sides mode and more.

When starting a new learn session you can choose between only learning new cards, learning expired cards and learning all (unlearned and expired) cards. Also there is additional info shown in the status bar while learning and some minor bugfixes.

* Detailed description:
* jMemorize is an open-source Java application that manages your
* learning processes by using flashcards and the famous Leitner
* system.
* It makes memorizing facts not only more efficient but also more fun. It boosts your whole learning experience and features categories, statistics and a visually appealing and intuitive interface.
* jMemorize manages your learn progress and features categories,
* statistics and a visually appealing and intuitive interface.
* It supported learning a foreign language (like english), and also learning historical dates and mathematical formulas.
* 1) Allot of flash card learning systems feature either aesthetically unpleasing or too unusual graphical user interfaces. jMemorize uses a interface that orientates itself on already matured eMail-clients where organizing information is already well explored. Additionally jMemorize features an aesthetically appealing interface by using icons, visual statistics and appropriate use of layouts.
* 2) jMemorize allows to organize Cards in a tree category hierarchy where every card of a subcategory also belongs to its father categories. This allows users to organize their cards and quickly change between the card subsets that should be learned right now.
* 3) jMemorize uses modern object-oriented design that facilitates future developments. Editing flashcards or categories fires events which are independently handled by the GUI, allowing to add new interface elements quickly and keeping the program flow intuitive. Clarity is taken very serious and the code is refactored continuously.
* It brings the much anticipated learn settings window and allows you to customize your learning pattern in nearly every way imaginable. You can limit your sessions by time and/or card limits,
* group your cards by categories, set custom time plans, learn in reversed sides mode and more.
* When starting a new learn session you can choose between only learning new cards, learning expired cards and learning all (unlearned and expired) cards. Also there is additional info shown in the status bar while learning and some minor bugfixes.

3-Provide a description for the feature/bug fix that you will start working on it, state it’s type:

1. Feature/Bug: Enhancing the management and functionality of the Category class.

Description: The Category class represents a category that can hold multiple decks of cards, with the cards organized into different levels. The class provides methods for managing cards within these decks, including adding and removing cards, moving cards between decks, and resetting card levels. Additionally, the class supports the creation and management of child categories, allowing for hierarchical organization of card sets. The class also provides functionality for observing and handling events related to card and category operations.

Type: Feature Enhancement

1. Feature/Bug: Implementing the CategoryObserver interface to enable observation of category events.

Description: The CategoryObserver interface defines methods for objects that wish to observe category-related events within the application. Observers implementing this interface can receive notifications when events such as card additions, removals, edits, and deck changes occur within a category or its child categories. Additionally, observers can be notified of category-related events such as category additions, edits, and removals.

Type: Interface Implementation

1. Feature/Bug: Implementing the Events interface to define constants for category events.

Description: The Events interface serves as a collection of constant values representing various category-related events within the application. These constants include events such as card addition, removal, movement, deck changes, edits, and expiration. By centralizing these event types in one interface, it promotes consistency and ease of reference throughout the codebase.

Type: Interface Definition

1. Feature/Bug: Improving management of card statistics in the Card class.

Description: This change focuses on enhancing the management of card statistics within the Card class to provide more accurate and insightful data tracking. Currently, the Card class maintains various statistics related to the card's learning progress, such as the number of tests taken, the number of successful repetitions, and the pass ratio. However, there is room for improvement in terms of how these statistics are updated and utilized. The proposed enhancement involves refining the logic for updating statistics to ensure consistency and accuracy across different operations, such as incrementing learned amounts and resetting statistics. Additionally, improvements will be made to calculate and display statistics more effectively, providing users with valuable insights into their learning progress. These enhancements aim to enhance the usability and reliability of the Card class, making it a more valuable tool for tracking and improving learning outcomes.

Type: Feature Enhancement

1. Feature/Bug: Improving image handling in the CardSide class.

Description: This change aims to enhance the handling of images within the CardSide class to provide a more robust and flexible solution for managing image content. Currently, the class supports storing image IDs in a list, but there is room for improvement in terms of functionality and usability. The proposed enhancements include adding methods to add, remove, and manipulate individual images more easily. Additionally, improvements will be made to ensure proper synchronization and notification mechanisms when images are modified, ensuring consistency across the application. These enhancements will enhance the usability and reliability of the CardSide class, making it more effective in managing image content within jMemorize.

Type: Feature Enhancement

4-Register the change request you defined in step 3 to a free ticketing system or software:

https://mohand-aly.atlassian.net/jira/software/projects/KAN/boards/1