Flashcard Application - Maria and her Minions

# Group Members

|  |  |
| --- | --- |
| Name | Admission Number |
| Maria Muwale | 192414 |
| Nathan Achar | 189206 |
| Githinji Mugambi | 189596 |
| Allan Waithaka | 191604 |
| Faith Muthoni | 178509 |

# What exactly does our project do?

Our project implements a flashcard study tool that mimics how learners use revision cards — flipping through them, going back, tagging useful ones, and dynamically updating the collection. We used four key data structures — linked list, queue, stack, and hash map — to model this behavior. MongoDB was used to store flashcards in the cloud.

# Which structures did we use and why?

|  |  |  |
| --- | --- | --- |
| Data Structure | Usage | Why We Used It |
| Linked List | Flashcard storage in memory | Dynamic and flexible add/edit/delete |
| Queue | Review flow (FIFO order) | Mimics normal flashcard flow |
| Stack | Flip-back or 'undo' feature | Follows LIFO for card history |
| Hash Map | Bookmarks & tag-based lookup | Allows instant access to key cards |

# Which technologies were used to achieve our project?

- Python 3  
- MongoDB Atlas (NoSQL cloud database)  
- pymongo (MongoDB Python driver)  
- Git & GitHub (collaboration & version control)  
- Tkinter GUI

# Which group member handled which module?

* Maria Muwale – Queue (Review Flow)
* Created FlashcardQueue class using doubly-linked Nodes
* Implemented enqueue, dequeue, peek, and display methods
* Faith Muthoni – Stack (Flip History)
* Created FlashcardStack class with push, pop, and peek
* Integrated stack updates into the review loop for backtracking
* Tracked previously reviewed flashcards for LIFO-based undo
* Githinji Mugambi – Hash Map (Bookmarks)
* Designed dictionary-based tag/bookmark system for cards
* Linked cards to topics or 'starred' labels
* Enabled instant retrieval of bookmarked cards
* Enabled the ability to review history
* Nathan Achar – Linked List (Card Storage) + MongoDB
* Built the LinkedList and Node classes for dynamic card management
* Enabled addition, editing, and deletion of flashcards
* Served as the backbone for queue and stack modules
* Set up MongoDB database for storage of data in a cloud platform
* Allan Waithaka – Interface + Integration
* Built the CLI menu to allow user interaction
* Integrated all modules: database, queue, stack, bookmarks
* Tested end-to-end functionality and handled edge cases

# Time and Space Complexity Summary

|  |  |  |  |
| --- | --- | --- | --- |
| Data Structure | Operation | Time Complexity | Space Complexity |
| Queue | Enqueue | O(1) | O(1) |
| Dequeue | O(1) | O(1) |
| Peek | O(1) | O(1) |
| Show All | O(n) | O(1) |
| Stack | Push | O(1) | O(1) |
| Pop | O(1) | O(1) |
| Peek | O(1) | O(1) |
| Linked List | Add Card | O(1) | O(1) |
| Edit/Delete/Search | O(n) | O(1) |
| Display All | O(n) | O(1) |
| Hash Map | Add Bookmark | O(1) | O(1) |
| Remove Bookmark | O(1) | O(1) |
| Lookup Bookmark | O(1) | O(1) |
| Get All Bookmarks | O(k) | O(k) |

# Screenshots

📌 Insert screenshots below once ready:

1. Adding a flashcard

2. Viewing the flashcard

3. Viewing the categories/bookmarks

4. Review history