E-Learning Platform

GROUP NO: C

Team Members:

Muhammad Farzad Ali: 126559

Aishwarya Kole: 126792

Anusha Rallapalli: 125800

Purpose and Overview: The Java application, an e-learning platform, is meticulously crafted to offer users a seamless experience in managing files without the complexities of a graphical user interface (GUI). Its core objective is to provide a robust solution for login procedures, file management, and data security within an educational context.

Application Structure: The application architecture embodies a modular design, with distinct Java classes fulfilling specialized roles. The AppController orchestrates interactions between various components, including DataStoreCsv for data persistence, FileManager for file operations, LoginManager for authentication tasks, and UserAuthenticator for verifying user credentials. The AppCLI acts as the command-line interface, facilitating user interactions.

Functionality Overview:

User Authentication: Leveraging secure authentication protocols, users access the platform by providing their unique credentials. The system validates these credentials against a predefined set of users stored securely within the application.

Password Recovery: In scenarios where users encounter login issues, a robust password recovery mechanism enables them to regain access by providing their username. Additional verification steps may be incorporated, such as email confirmation or security questions, to fortify this process further.

File Upload: Users seamlessly upload files by specifying relevant details such as the file owner, title, category, and path. The application ensures data integrity by storing uploaded files both in a structured CSV table for efficient retrieval and in a designated folder for immediate access.

File Download: Accessing uploaded files is effortless as users simply input the filename of interest. The application promptly retrieves the actual file path, facilitating swift download and uninterrupted learning experiences.

File Deletion: To maintain data hygiene and user privacy, the application empowers users to delete files as needed. File deletion operations are meticulously handled, ensuring removal from both the CSV database and the designated storage folder.

Implementation Details:

Input Handling: User inputs are meticulously managed through the Scanner class, enabling intuitive interactions via command-line prompts. This approach ensures user-friendly engagement while maintaining robustness.

Error Handling: The application boasts a comprehensive error handling mechanism, adept at addressing diverse scenarios such as invalid file paths, non-existent files, and input validation errors. These safeguards enhance user experience and minimize disruption during operation.

Data Management: User credentials and uploaded files are securely stored in memory using advanced data structures like Maps, bolstered by encryption algorithms where applicable. This approach not only ensures data integrity but also fortifies the application against potential security threats.

Potential Areas for Improvement:

Enhanced Error Handling: Continual refinement of error handling mechanisms to encompass rare edge cases and unexpected scenarios, thereby fortifying the application's resilience.

File Deletion Authorization: Implement stringent authorization protocols to restrict file deletion privileges solely to authorized users, safeguarding against unauthorized data manipulation.

User Management Enhancement: Augment user management functionalities to establish robust associations between database entries and respective users, fostering accountability and streamlined data governance.

Limitations:

- The storage of passwords in plaintext poses inherent security risks, necessitating the adoption of industry-standard encryption techniques to safeguard sensitive user information.
- The absence of file deletion restrictions exposes the application to potential data integrity breaches, warranting immediate attention to fortify access controls and bolster security measures.
- Constraints stemming from Java version compatibility hinder the integration of advanced frameworks such as Spring Boot, necessitating alternative approaches to achieve optimal functionality and performance.
- The lack of hyperlink support within the command-line interface poses usability challenges, prompting users to manually copy file addresses for access, thus detracting from the overall user experience.
- The application's inability to handle duplicate files and provide intuitive file indexing diminishes efficiency and user convenience, highlighting areas for refinement in file management capabilities.

Possible Implementations:

Security Enhancement: Integrate state-of-the-art authentication mechanisms, including multi-factor authentication and biometric verification, to fortify the application against evolving security threats.

Streamlined User Experience: Implement intuitive features such as auto-generated unique filenames, file indexing, and hyperlink support within the CLI interface to enhance user engagement and productivity.

Advanced Logging and Monitoring: Deploy robust logging and monitoring solutions to track user activities, detect anomalies, and mitigate security breaches effectively, thereby enhancing system transparency and resilience.

Handover Meeting Resolutions for Requirements Review:

1. **User Registration and Validation:**
    - Ambiguity: It wasn't clear how user registration and validation processes would be handled, especially regarding unique usernames and email IDs.
    - Resolution: In the handover meeting, it was clarified that user registration would entail unique username checks against existing records in the database. This was agreed upon to prevent duplicate users.
2. **Forget Password Functionality:**
    - Incompleteness: The forget password functionality was mentioned but lacked details on how it would be implemented.
    - Resolution: During discussions, it was decided that forget password functionality would involve users providing their username.
3. **Inactivity Account Removal:**
    - Imprecision: The duration of inactivity leading to account removal was not specified.
    - Resolution: It was agreed upon that accounts inactive for more than six months would be automatically removed from the database. This was considered a reasonable timeframe to maintain database hygiene. However this feature required some backend framework to track the user activity which therefore considered not possible in the current implementation and hence ignored while development.