Description of task and its objectives.
The project aims to create an ATM simulation software that enables users to perform banking operations including deposit, withdrawal, password modification, and logout. The program is designed to support different account types, including savings, current, and business accounts. The software also keeps a track of transactions if needed in the future. The software's primary objective is to offer users a reliable and secure interface for managing their financial transactions effectively.

How you tend to solve this problem
The solution to this problem involves the isolation of the back-end and GUI logic into separate classes. Each screen in the program has its own dedicated class, resulting in a highly modular and flexible design. This approach not only ensures that the code is organized and easy to maintain, but also makes it easier to add new features or modify existing ones in the future.

A discussion of programming constructs I'll be using
In research for the project, it is determined that the object-oriented paradigm is ideal for modeling the behaviors and interactions amongst various components of the ATM. In addition to providing extensibility for future upgrades (Sharma 2020), it also provides a scalable approach to design. The programming constructs used in this project include:

- OOP concepts such as inheritance, polymorphism, and encapsulation.
- GUI components such as JFrame, JButton, JLabel, and JTextField.
- Event-driven programming with ActionListener (Javatpoint 2022) and ComponentListener interfaces
- Exception handling with try-catch blocks to handle errors and user input validation
- Collection framework with ArrayList to store and manipulate account information
- Control structures such as if-else statements and loops to control the flow of the program

What measures I've taken to detect and prevent errors.
To detect and prevent errors, I have implemented several validators (Baeldung 2022) such as input validator and authenticators to ensure stable functioning of the program. Furthermore, I have written extensive main functions to test individual modules as well as the program as a whole. By taking these measures, I can detect any errors and bugs that may arise during the testing phase and ensure a smooth user experience.

Evidence of Appropriate Testing
Please refer to main(String[] args) of classes present in the code. Every one of them includes test cases to validate functionality of individual modules as well as their coupling.

References

Sharma, N. (2020) ATM‑an object-oriented design, Medium. The Startup. Available at: https://medium.com/swlh/atm-an-object-oriented-design-e3a2435a0830 (Accessed: March 5, 2023).

Baeldung, W. (2022) Check if a string is numeric in Java, Baeldung. Available at: https://www.baeldung.com/java-check-string-number (Accessed: March 5, 2023).

Javatpoint (2022) - Javatpoint www.javatpoint.com. Available at: https://www.javatpoint.com/java-actionlistener (Accessed: March 5, 2023).