ATM SIMULATOR APPLICATION

Project Overview

This project is an **ATM Simulator Application** developed in Python. It supports:

- Card validation
- PIN authentication
- Balance inquiry
- Cash withdrawal
- Transaction history
- Proper exception handling

The objective of this report is to explain how this project can be managed using **Kanban** and **Scrum** methodologies.

KANBAN Methodology

What is Kanban?

Kanban is a visual workflow management method that helps track work using simple stages such as:

- **To Do**
- **In Progress**
- **Done**

Work items move from left to right as they are developed and completed.

Kanban Board for ATM Simulator

1. TO DO (Planned Tasks)

These are features that are planned but not yet started:

- Design ATM flow (Card → PIN → Menu)

- Create custom exception classes (ATMError, InsufficientFundsError, etc.)
- Create user data structure (users dictionary)
- Implement card validation function
- Implement PIN validation function
- Design ATM menu options
- Plan transaction history feature

---

2. IN PROGRESS (Currently Being Developed)

These are tasks that are under development:

- Implement check_balance() function
- Implement withdraw_cash() function
- Implement transaction_history() function
- Build menu loop using while True
- Add exception handling for wrong PIN and insufficient balance
- Connect all functions in main program flow

---

3. DONE (Completed Tasks)

These are finished and working features:

- Card validation is working
- PIN authentication is working
- Balance inquiry is working
- Cash withdrawal updates balance correctly
- Transaction history is displayed correctly
- Exception handling works properly
- Full ATM menu flow is working

How Kanban Helps This Project

- Clear visibility of work status
- Easy to track progress
- Simple to manage and update tasks
- Continuous delivery of features

SCRUM Methodology

What is Scrum?

Scrum is an **iterative and incremental** development framework that works in **Sprints** (usually 1–2 weeks). It uses roles, meetings, and reviews to manage development.

Scrum Roles in This Project

1. Product Owner

- Decides what features are needed in the ATM system
- Example requirements:
  - Card validation
  - PIN authentication
  - Withdraw cash
  - Show transaction history

2. Scrum Master

- Ensures Scrum process is followed
- Removes obstacles for the developer
- Makes sure daily meetings and sprint activities happen properly

3. Developer

- Writes the Python code
- Implements features
- Fixes bugs
- Tests the application

Scrum Activities Applied to ATM Project

1. Sprint Planning

In this meeting, we decide what to build in the sprint.

Example Sprint Plan (Sprint 1):

- Implement card validation
- Implement PIN validation

- Create main menu structure

Sprint 2:

- Implement balance inquiry
- Implement cash withdrawal
- Add transaction history
- Add exception handling

---

2. Daily Scrum (Daily Standup)

A short daily meeting where we answer:

- What did I do yesterday? (e.g., implemented withdraw function)
- What will I do today? (e.g., implement transaction history)
- Any problems? (e.g., error handling issue)

---

3. Sprint Review

At the end of the sprint, we **demonstrate the working ATM system**.

Example:

- Show login with card and PIN
- Show balance check
- Show withdrawal and updated balance

Feedback is collected and new features are suggested.

---

4. Sprint Retrospective

This is a **reflection meeting**.

We discuss:

- What went well? (e.g., menu system works smoothly)
- What can be improved? (e.g., better input validation)
- What will we improve in the next sprint? (e.g., add deposit feature)

How Scrum Helps This Project

- Work is divided into small manageable parts
- Regular feedback improves quality
- Problems are found early
- Step-by-step improvement of the ATM system

Comparison Table

Feature	Scrum	Kanban
Planning	Features planned in sprints	Features picked continuously
Delivery	After each sprint	Continuous after task completion
Testing	At end of sprint or during sprint	Immediately after task completion
Adding Features	Added in next sprint	Can be added anytime
Example in Program	Card validation implemented in Sprint 1	Card validation task pulled from board and done

---

Problems & Solutions

Problems of Kanban:

- **No Fixed Deadline**: Kanban does not use time-boxed sprints, making it difficult to predict delivery timelines.
- **No Clear Roles:** Kanban does not define specific roles like Product Owner or Scrum Master, which can cause unclear responsibilities within the team.
- **Lack of Long-Term Planning:** Kanban focuses on current tasks and may ignore future planning.
- **Less Suitable for New Projects:** Kanban works better for ongoing work than brand-new projects.

Solutions for Kanban Problems:

- **No Fixed Deadline:** Introduce internal target dates or service-level expectations (SLEs) to track delivery time.
- **No Clear Roles:** Define team responsibilities informally by assigning ownership for tasks and decision-making.
- **Lack of Long-Term Planning:** Conduct periodic planning or review meetings to align with long-term goals.
- **Less Suitable for New Projects:**Start with Scrum for initial development and transition to Kanban for maintenance.

Problems of Scrum:

- **Fixed Sprint Duration:** Changes during a sprint are difficult to accommodate once the sprint starts.
- **High Dependency on Team Discipline:** Scrum requires experienced and disciplined team members to follow ceremonies and rules effectively.
- **More Meetings:** Daily stand-ups, sprint planning, review, and retrospective can increase meeting overhead.
- **Role Dependency:** Scrum heavily depends on defined roles like Product Owner and Scrum Master; absence or inefficiency of these roles can impact progress.

Solutions for Scrum Problems:

- **Fixed Sprint Duration:** Allow minor changes through backlog refinement and plan urgent items for the next sprint.
- **High Dependency on Team Discipline:** Provide proper Scrum training and enforce clear Scrum practices.
- **More Meetings:** Keep meetings time-boxed and focused to reduce overhead.
- **Role Dependency:** Clearly define responsibilities and ensure backup ownership for key roles.

Conclusion

- **Kanban** is useful for continuous tracking of tasks using To Do, In Progress, Done.
- **Scrum** is useful for structured development using Sprints, Reviews, and Retrospectives.
- Both methodologies can be effectively used to manage the development of this **ATM Simulator Application**.