**SDK Bank System - Project Report**

*Developed with Python & Streamlit*

**1. Introduction**

A web-based banking application simulating real-world operations:

* **Core Features**: Account management, transactions, bill payments, currency conversion.
* **Tech Stack**: Python, Streamlit (UI), file-based transaction logging.
* **Audience**: Educational demonstration (can be extended for production).

**2. Features Overview**

**2.1 Account Management**

| **Function** | **Description** |
| --- | --- |
| **User Registration** | Username, numeric password, phone, security Q&A. Prevents duplicates. |
| **Login** | Supports username or phone number + password. |
| **Password Reset** | Security question verification required. |

**2.2 Financial Operations**

| **Function** | **Description** |
| --- | --- |
| **Deposit/Withdraw** | Validates amount; updates balance. |
| **Click Transfer** | Peer-to-peer transfers via phone numbers. Logs sender/receiver transactions. |

**2.3 EFAWATEER (Bill Payments)**

| **Bill Type** | **Calculation Logic** |
| --- | --- |
| Electricity | (Total Device Power × Usage Hours × 30 Days) / 3.3 → JOD |
| Water | Weekly Usage (m³) × 4 → JOD |
| Telecom | (Call Minutes × 0.5) + (Data GB × 1.5) → JOD |

**2.4 Additional Features**

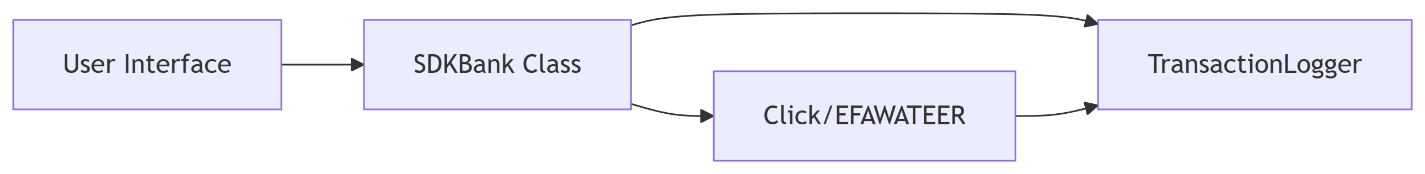
* **Currency Converter**: JOD to 20+ currencies (hardcoded rates).
* **Transaction History**: Logs all actions with timestamps.

**3. Technical Architecture**

**3.1 Module Breakdown**

| **File** | **Purpose** |
| --- | --- |
| sdk\_bank.py | Core logic (accounts, balances, passwords). |
| transaction\_logger.py | Logs transactions to transactions.txt; retrieves user-specific history. |
| clicker.py | Handles fund transfers between users. |
| efawateer.py | Bill payment workflows (electricity, water, telecom). |
| currency\_converter.py | Currency conversion using predefined rates. |
| web\_page.py | Streamlit UI (forms, navigation, session management). |

**3.2 Data Flow**



**4. Security & Validation**

**4.1 Implemented Measures**

* **Session Management**: st.session\_state tracks logged-in users.
* **Input Validation**:
  + Numeric passwords/phone numbers.
  + Meter numbers (e.g., 10 digits for electricity).

**4.2 Identified Risks**

| **Risk** | **Severity** | **Suggested Fix** |
| --- | --- | --- |
| Plaintext passwords | High | Use hashlib for SHA-256 hashing. |
| eval() in transaction logs | Critical | Replace with ast.literal\_eval(). |
| No account lockout | Medium | Add failed-attempts counter. |

**5. Limitations & Future Work**

**5.1 Current Limitations**

* **Data Loss**: In-memory storage (no database).
* **Error Handling**: Generic messages (e.g., "Invalid credentials").
* **Redundancy**: Duplicate withdraw method in SDKBank.

**5.2 Recommended Enhancements**

1. **Database Integration**: SQLite/PostgreSQL for persistent data.
2. **Password Hashing**:

python

import hashlib

hashed\_pw = hashlib.sha256(password.encode()).hexdigest()

1. **Unit Tests**: Pytest for core functions (e.g., transfers, withdrawals).

**6. Conclusion**

**Strengths**: Modular design, comprehensive features, clear UI.  
**Areas for Improvement**: Security, persistence, testing.

**Final Verdict**: Excellent learning project; with refinements, could transition to production.

**Appendix**:

* **Sample Screenshots**: (Add UI images if available)
* **Code Snippets**: (Highlight critical sections)

**To create this in Word**:

1. Open **Microsoft Word**.
2. Paste this content.
3. Apply styles:
   * **Headings**: Use Heading 1, Heading 2.
   * **Tables**: Insert → Table (for feature summaries).
   * **Bullets**: Home → Bullet List.
4. Export as .docx.

