Design Pattern Descriptions

1. Singleton Pattern - Session Management
Ensures only one SessionManager instance exists throughout the application. Manages user authentication, auto-lock timer, and clipboard auto-clear. All components access the same session state.

2. Builder Pattern - Password Generation
Separates password construction from its representation. PasswordDirector orchestrates the building process, while PasswordBuilder constructs passwords step-by-step with configurable options (length, uppercase, numbers, symbols).

3. Observer Pattern - Security Notifications
SecurityMonitor (Subject) notifies registered observers when security issues are detected. ToastObserver subscribes and displays warnings for weak passwords and expiring cards/documents.

4. Proxy Pattern - Data Masking
SensitiveDataProxy controls access to sensitive data stored in RealSensitiveData. Returns masked values (•••••) by default, reveals real values only when toggled. Uses MaskingRules for type-specific masking.

5. Chain of Responsibility Pattern - Password Recovery
Chains three SecurityQuestionHandler objects together. Each handler verifies one security question answer. If correct, passes to the next handler. All three must pass for successful recovery.

6. Mediator Pattern - UI Communication
UIMediator centralizes communication between UI components (AuthColleague, VaultColleague, ToastColleague). Components send events through the mediator instead of communicating directly, reducing coupling.

7. Database Schema - IndexedDB Storage
Local browser database with two object stores. users stores accounts with hashed passwords, security questions, and settings (keyed by email). vault stores all sensitive items (logins, cards, identities, notes) linked to users via userEmail index. Supports CRUD operations for persistent local storage.

UI IMAGES AND DATABASE CLASS DIAGRAM UML IMAGES ARE INSIDE OF ‘ui images’ and ‘Patterns UML 476’ FOLDERS IN THE REPOSITORY