Name: Samuel Damas  
Student No: ST10496103  
MODULE NAME: PROGRAMMING 1A   
MODULE CODE: PROG5121  
ASSESSMENT: POE PART 2 - FORMATIVE 2  
LECTURER: Anelisa  
DATE OF SUBMISSION: 13 October 2025

|  |  |  |
| --- | --- | --- |
|  |  |  |

BIT0701

PROG5121

POE PART 2

**Part 1**

**GITHUB REPOSITORY LINK:** <https://github.com/TaSam13/st10496103_Prog5121_POE>

**PROJECT OVERVIEW:**

This Java application implements a comprehensive **QuickChat Messaging System** with user registration, authentication, and message management. The system features robust validation, JSON-based data persistence, and a graphical user interface using Swing dialogs. Building upon the Part 1 foundation, this extension adds complete messaging functionality with professional pop-up interactions.  
  
KEY FEATURES IMPLEMENTED:  
- **Part 1 Features:**

* User registration with comprehensive input validation
* Username validation (underscore requirement and ≤5 characters)
* Password complexity validation (≥8 chars, capital letter, number, special character)
* South African cellular number validation with international formatting (+27 followed by 9 digits)
* User authentication and login functionality
* Complete unit test suite with 19 test cases

**Part 2**

**Part 2 Extended Features:**

* Messaging System: Send, store, and discard messages with status tracking
* Message Validation: Recipient number validation and message length checks (≤300 characters)
* Message Hash Generation: Automatic generation in format [first two ID numbers]:[message number]:[FIRSTWORDLASTWORD]
* JSON Data Persistence: All messages saved to messages.json file using Jackson library
* Popup User Interface: All user interactions through Swing JOptionPane dialogs
* Message Storage: Robust error handling for file operations

Extended Unit Tests: Additional test suites for messaging functionality (MessageTest, MessageStorageTest)

**TECHNICAL SPECIFICATIONS:**

* Programming Language: Java 11
* Development Environment: IntelliJ IDEA
* Testing Framework: JUnit 4.13.2
* JSON Processing: Jackson Library 2.17.1
* User Interface: Swing JOptionPane for popup dialogs
* Version Control: Git with GitHub repository
* Build Tool: Maven

SYSTEM ARCHITECTURE:

The application follows object-oriented principles with four main classes:

* Main: Handles user interface flow and popup interactions
* Login: Manages user registration, validation, and authentication
* Message: Represents message entities with validation and hash generation
* MessageStorage: Handles JSON serialization and file operations
* LoginTest: Comprehensive unit test suite
* MessageTest: Unit tests for message validation and hash generation (10 test cases)
* MessageStorageTest: Unit tests for JSON persistence (2 test cases)

**TESTING IMPLEMENTATION:**

* Comprehensive Test Coverage: 31 total test cases across all functionality
* Message Validation Tests: Verify 250-character limit and proper error messages
* Phone Number Validation: Tests for correct South African format (+27 followed by 9 digits)
* Message Hash Verification: Ensures correct format 00:0:HITONIGHT per requirements
* JSON Storage Tests: Validate file creation and data persistence

**IMPLEMENTATION HIGHLIGHTS:**

* Message Hash Algorithm: Extracts first two digits from Message ID, combines with message number and first/last words in uppercase
* Input Validation: Multi-layer validation for usernames, passwords, phone numbers, and message content
* Error Handling: Graceful handling of file I/O exceptions and invalid user inputs
* Data Persistence: JSON-based storage maintains message history between sessions
* Popup Integration: All user interactions through modal dialogs for professional user experience

**CHALLENGES AND SOLUTIONS:**

* JSON Serialization: Overcame Jackson library constraints by implementing proper constructors and annotations
* Popup Integration: Successfully converted console-based interactions to modal dialogs for better user experience
* Message Hash Format: Implemented precise string manipulation to meet specification requirements
* Test Environment: Configured Java 11 compatibility for successful test execution

**REFERENCES:**  
  
1. **OpenAI.** (2025). ChatGPT (Sep 18 version) [Large language model].  
[https://chat.gpt.com](https://chat.gpt.com/)

* + Used for development of regular expression pattern for South African phone number validation
  + Specific conversation: <https://chatgpt.com/share/68cda86d-9eb0-8010-a03d-4ed6d53b6eb2>

1. **JUnit Team.** (2020). JUnit 4.13.2 Testing Framework.  
   Retrieved from <https://junit.org/junit4/>
   * Unit testing framework implementation
2. **Oracle Corporation.** (2025). Java Programming Language Documentation.  
   Retrieved from <https://docs.oracle.com/javase/8/docs/api/>
   * Java language features and API implementation
3. **FasterXML Jackson.** (2024). Jackson JSON Processor.  
   Retrieved from <https://github.com/FasterXML/jackson>
   * Used for JSON serialization/deserialization of Message objects
4. **Oracle Corporation.** (2025). Java Swing Documentation.  
   Retrieved from <https://docs.oracle.com/javase/8/docs/api/javax/swing/JOptionPane.html>
   * Used for popup dialog implementations
5. **GitHub, Inc.** (2025). GitHub Repository Hosting.  
   Retrieved from <https://github.com/>
   * Version control and code repository management