|  |
| --- |
| CS 307 – Software Engineering |
| ATM 2.0 |
| Project Requirements Backlog |

|  |
| --- |
| **Team 25**  Anthony Goeckner  Krutarth Rao  Austin Reed  Harold Smith  9-5-2016 |

**Problem Statement**

Currently, automated teller machines (ATMs) are subject to fraud by use of stolen bank cards and information. Our solution, the ATM 2.0, involves the use of three-point biometric and traditional PIN authentication, which will effectively negate this risk by requiring fingerprints and facial recognition in order to dispense money. Such ATM systems are not used commercially in the United States at this time.

**Background Information**

Automated teller machine technology reached a plateau in the 1970s. The machines have continued to use PIN and card authentication to verify the user’s identity. Over time, these systems have grown increasingly vulnerable to security threats, both through software and through physical means.

ATM 2.0 hopes to provide banks a more secure way to dispense cash to their customers. Through 3-point biometric authentication, the bank can be sure that the intended customer is collecting the cash and the customer can be sure that no unauthorized user can withdraw cash from their bank account without their presence.

In recent years, biometric ATM systems have been developed in many foreign countries. However, this increased security has not yet reached American shores. While American banks are currently researching the possibility of adding biometric systems, we believe that we can deliver better value in a faster timeframe and in a more cost-effective manner.

Originally ATMs were created because banks wanted to have tellers 24 hours a day, this was proving to be quite expensive. Keeping this in mind, current ATM machines should provide at least as much security as a human teller. However, a fraudster today only needs a PIN that he/she can acquire through a multitude of attack vectors on the user’s digital data and the card can be acquired through physical theft. Then on, if the thief were to walk up to a teller with a card and PIN, a teller would be able to detect that the person doesn’t match the photo ID on record and prevent any fraudulent withdrawals. The current machines don’t provide this level of security but ATM 2.0 does.

Standard ATMs use only two-factor authentication. The addition of biometric authentication ensures protection of the user’s bank accounts against malicious activity, since the user’s biometric characteristics are unique and cannot be stolen or forged.

**Functional Requirements**

* As a user, I need to withdraw specific amounts of money.
* As a user, I would like an intuitive, touch based interface.
* As a user, I need to deposit money.
* As a user, I need to manage my security preferences, such as enabling/disabling biometric authentication.
* As a user, I need to view my account balance.
* As a user, I should be able to set my PIN.
* As a user, I should be able to enter a backup password received from the bank.
* As a user, I need to transfer funds between accounts.
* As a user, I would like to select my language.
* As a user, I would like a “Support” button in case I run into problems.
* As a user, I would like the ability to print a receipt.
* As a user, I would like the ability to make multiple transactions.
* As a user, I need the ability to log out of my account.
* As a user, my account should log out automatically after a certain period of time.
* As a user, I need the ability to cancel a transaction.
* As a user, I would like the ability to withdraw using a “fast cash” feature.
* As a user, I would like the ability to set my “fast cash” amount.
* As a user, I would like the ability to enter an “emergency PIN”, which will silently contact the police in case of a robbery.
* As a developer, I would like to use Triple DES level encryption on the authentication data collected from the user before being sent to the network layer.
* As a developer, I would like USB ports disabled to prevent unauthorized access.
* As a developer, I would like Windows to be the operating software for interoperability with Azure.
* As a developer, I would like the connection between the Raspberry pi and the server to be over a VPN.
* As a developer, I would like to create an interface for the bank to add/modify customers’ details in the customer database.
* As a developer, I would like to use Azure services to host my customer database, authentication protocol and banking interface.
* As a developer, I would like to use C# and LINQ queries to modify my database.
* As a customer, I would like the machine to be capable of detecting sensor failures.
* As a customer, I would like the system to be capable of functioning as a standard ATM in the case of biometric sensor failure.
* As a customer, I would like to warn the user that the ATM is operating in reduced security mode in case of biometric sensor failure.
* As a customer, I would like each user to be able to choose whether their account will be accessible in reduced security mode.
* As a customer, I would like the ATM 2.0’s backend systems to be compatible with standard ATMs.
* As a customer, I would like my logo to be displayed on the screen.
* As a customer, I would like for my users to be authenticated a second time before allowing them to change their settings.
* As a customer, I would like optional remote terminal access to the machine.
* As a customer, I would like the ATM to enter “sleep” mode while waiting for additional users.
* As a customer, I would like the ATM to display my logo on the screen while in sleep mode.

**Non-Functional Requirements**

**Energy costs**

As a customer, I need the machine and software to be energy efficient to minimize electricity costs because the machines will be running 24hr a day. Also because the ATM2.0 cannot go be allowed to go offline during a power outage, the machine should be able to run on backup power for a sustained period of time.

**Security**

The ATM2.0 will offer high incentives for theft due to the large amounts of cash in the machine, this can be mitigated by a strong and physically secure machine case. Also, the “check balance” features revels sensitive information about the user so the machine needs to make sure the TCP packets are not hijacked during transit from database to machine. Otherwise, the database should also be secure against unauthorized querying from external nodes.

* As a developer, I would like to have code that is sufficiently documented and well formatted.
* As a developer, I would like code that is modular and built to accommodate updates in the near or far future.
* As a developer, I would like a robust and customizable framework for the graphics user interface. Some options might be the wxWidgets or QT graphics libraries.
* As a developer, I would my customer database to be easily scalable to allow for growth and additional ATMs.