MIAMI

CSC 431

Cane Wallet

System Architecture Specification (SAS)

Team 11

Bender Al-Awwad	<Role>
Preston Gmernicki	<Role>
Blaise Driscoll	<Role>

Version History

Version	Date	Author(s)	Change Comments

Table of Contents

1. System Analysis

1.1. System Overview

Cane Wallet is a digital ID solution for university students, allowing them to add their Cane Card to Apple Wallet. It enables payment with Dining Dollars, access to campus buildings, and resource checkout using NFC technology. The architecture emphasizes modularity and Apple Wallet API integration, designed exclusively for iOS platforms.

1.2. Actor Identification

Student: Adds Cane Card, makes payments, accesses buildings, checks out resources.
University System: Authenticates and manages student records.
Apple Wallet: Stores digital Cane Card and interacts via NFC.
NFC Terminal: Accepts inputs from Apple Wallet and grants access or completes transactions.
Campus Facilities: Backend systems for dining, access control, and library/resource tracking.

1.3. Design Rationale

1.4. Architectural Style

3-Tier Architecture:
Presentation Tier: iOS front-end app + Apple Wallet interface
Logic Tier: University backend authentication and control systems
Data Tier: Student and transaction data storage systems

1.5. Design Pattern(s)

Model-View-Controller (MVC): Keeps front-end interactions cleanly separated from business logic.
Model:
Student: Stores user info like name, ID, device link status. Contains methods like authenticate() and makeTransaction().
CaneCard: Holds data about the digital card (balance, access permissions). Handles updateBalance() and verifyAccess().
Transaction: Manages payment records and logs via processTransaction() and logTransaction().
AccessControl: Handles logic related to determining whether a student can access a specific building.
AuthenticationManager: Handles security-related logic like issuing and verifying tokens.

View:This is the interface the user interacts with — typically a front-end app or Apple Wallet screen
The Apple Wallet interface that shows the Cane Card, balance, and prompts for NFC interaction.
Any in-app views showing login forms, wallet linking confirmation, or transaction status.

Controller:The Controller receives input from the user (e.g., tap phone to NFC reader), processes it by interacting with the Model, and then updates the View.

Responsibilities in the Controller:

Receives user actions (e.g., tap to pay).
Calls methods like Student.makeTransaction() or AccessControl.validateAccess(...).
Updates the UI with feedback (success/failure, updated balance, error messages).

Singleton Pattern: For consistent state handling (e.g., current session or wallet info).

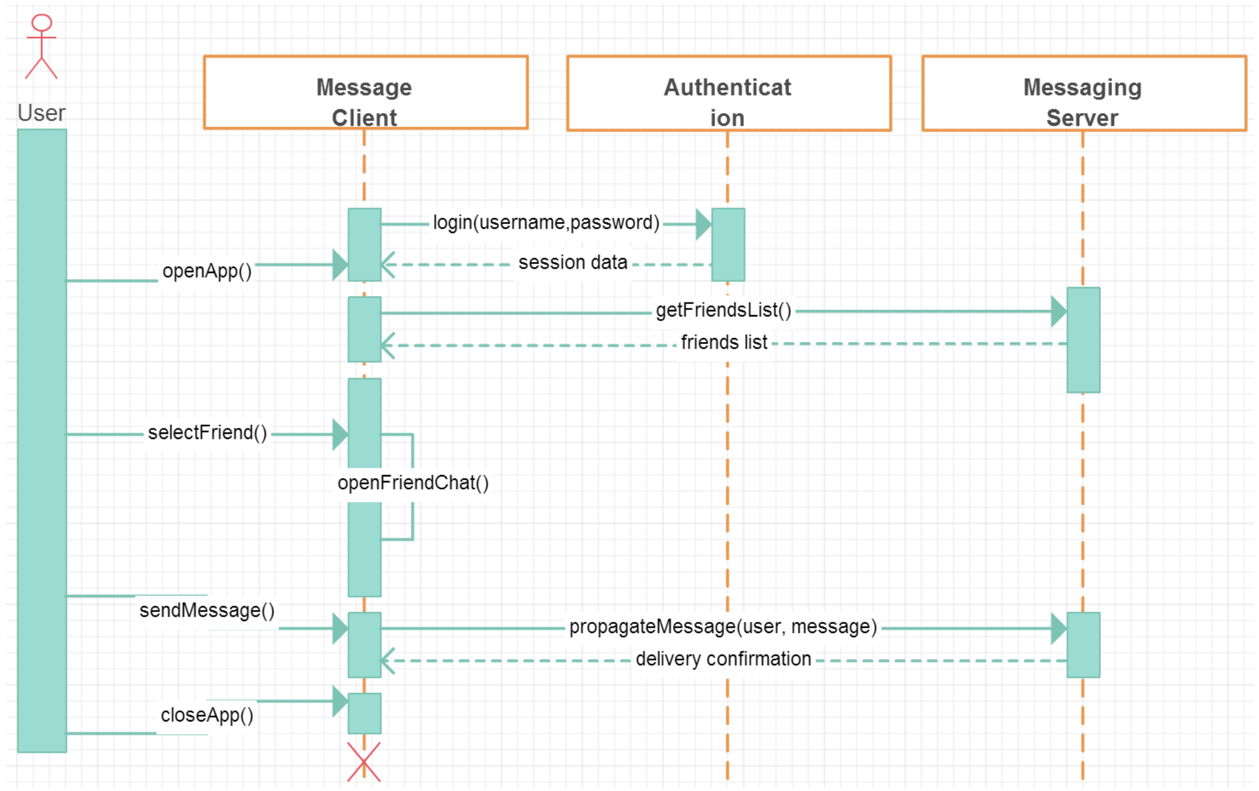
1.6. Framework

Apple Wallet API: Facilitates seamless integration with digital cards.
Swift: Primary language due to iOS exclusivity.
Xcode: IDE for building and testing iOS applications.

These frameworks are designed for Apple products, ensuring best-in-class performance and security, reducing development needs with pre-built, secure functionality that aligns with the system's requirements.

2. Functional Design

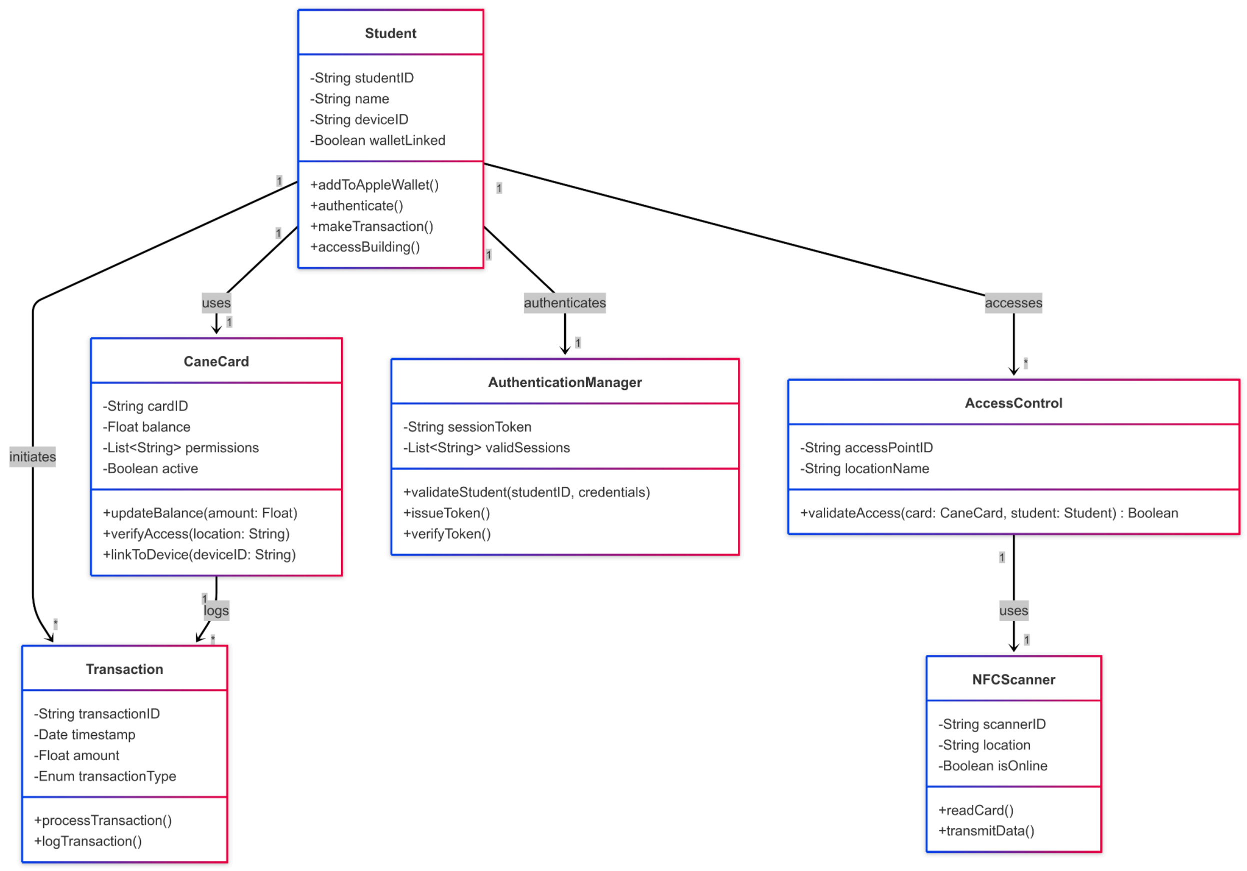
2.1. Sequence Diagram



3. Structural Design

- Class: Student
  - Attributes:
    - studentID: String
    - name: String
    - deviceID: String
    - walletLinked: Boolean
  - Methods:
    - addToAppleWallet()
    - authenticate()
    - makeTransaction()
    - accessBuilding()
- Class: CaneCard
  - Attributes:
    - cardID: String
    - balance: Float
    - permissions: List<String>
    - active: Boolean
  - Methods:
    - updateBalance(amount: Float)
    - verifyAccess(location: String)
    - linkToDevice(deviceID: String)
- Class: Transaction
  - Attributes:
    - transactionID: String
    - timestamp: Date
    - amount: Float

        - transactionType: Enum (Dining, Resource, Other)
    - Methods:
        - processTransaction()
        - logTransaction()
- Class: AccessControl
    - Attributes:
        - accessPointID: String
        - locationName: String
    - Methods:
        - validateAccess(card: CaneCard, student: Student): Boolean
- Class: NFCScanner
    - Attributes:
        - scannerID: String
        - location: String
        - isOnline: Boolean
    - Methods:
        - readCard()
        - transmitData()
- Class: AuthenticationManager
    - Attributes:
        - sessionToken: String
        - validSessions: List<String>
    - Methods:
        - validateStudent(studentID, credentials)
        - issueToken()
        - verifyToken()


Student
-String studentID
-String name
-String deviceID
-Boolean walletLinked
+addToAppleWallet()
+authenticate()
+makeTransaction()
+accessBuilding()
1
1
1
1
uses
1
authenticates
1
accesses
*
initiates
*
CaneCard
-String cardID
-Float balance
-List<String> permissions
-Boolean active
+updateBalance(amount: Float)
+verifyAccess(location: String)
+linkToDevice(deviceID: String)
1
logs
*
AuthenticationManager
-String sessionToken
-List<String> validSessions
+validateStudent(studentID, credentials)
+issueToken()
+verifyToken()
AccessControl
-String accessPointID
-String locationName
+validateAccess(card: CaneCard, student: Student) : Boolean
1
uses
1
Transaction
-String transactionID
-Date timestamp
-Float amount
-Enum transactionType
+processTransaction()
+logTransaction()
NFCScanner
-String scannerID
-String location
-Boolean isOnline
+readCard()
+transmitData()