# CSE251 – " Project Proposal

## 1. Project Title

### Smart Reverse Vending Machine with Gamification System

## 2. Project Description

The proposed project aims to design and implement a smart reverse vending machine (RVM) that encourages university students to recycle plastic waste through gamification and rewards. Students scan their unique QR codes, deposit plastic recyclables, and earn points based on weight. The system creates multi-level competition (student vs. student, college vs. college) with leaderboards to drive engagement.

Points earned can be redeemed at the university cafeteria for food and beverages, creating a closed-loop incentive system. The RVM is powered by Raspberry Pi with integrated sensors (QR scanner, weight sensor, material classifier, liquid detector) and operates with offline capability, automatically synchronizing data when connectivity is restored.

**Material Classification:** AI-powered sensors automatically identify whether deposited items are plastic or non-plastic waste. Nonplastic materials are diverted to a separate waste compartment, preventing contamination of recyclable materials.

**Liquid Detection:** Advanced weight and tilt sensors detect liquid content inside plastic bottles. Bottles containing liquid are automatically rejected with clear instructions displayed to users to empty them first, ensuring recycling quality and preventing machine damage.

**The RVM includes a "Guest" button allowing unregistered campus members (doctors, TAs, faculty, staff, visitors) to contribute to recycling efforts without registration. Guest deposits contribute to overall campus sustainability metrics, promoting inclusive environmental responsibility across the entire university community.**

The system consists of a mobile application for students to view their QR codes, track points, check leaderboards, and redeem rewards. Administrators monitor system performance, manage inventory, view analytics through a web dashboard, and manage multiple RVM machines deployed across the university campus. The system automatically alerts administrators when machine capacity reaches 80% for timely maintenance and emptying. Administrators can also manage cafeteria rewards and view detailed reports on transactions, rankings, and environmental impact.

By combining IoT hardware, AI-based material validation, gamification, renewable energy awareness, and smart campus technologies, this project transforms waste management into an engaging, competitive, and eco-friendly campus service available to ALL campus members. **The system architecture follows SOLID principles to ensure maintainability, scalability, and extensibility.**

***"This project demonstrates the integration of IoT, AI validation, gamification, and sustainable practices into a scalable and eco-friendly campus recycling solution, designed with SOLID principles for robust software architecture and inclusive campus-wide participation."***

## 3. Team Members

| ID | Name |
|---|---|
| 23101608 | Yasseen Ahmed El-Sayed |
| 23101594 | Mohamed Mustafa Awad |
| 23101545 | Ahmed Khaled Said |
| 23101599 | Marawan Khaled Ahmed |
| 23101899 | Mohamed Adel Abdulrahman |

## 4. Functional Requirements

***"The following requirements are organized hierarchically with 30 high-level functional requirements, each containing specific low-level implementation details."***

### 1. User Registration  Authentication Management

1.1 The system shall allow students to register using university email, student ID, name, and college affiliation
1.2 The system shall validate university email domains before registration completion
1.3 The system shall authenticate users securely using encrypted credentials
1.4 The system shall provide password recovery functionality via email verification

### 2. QR Code Generation  Management

2.1 The system shall generate unique QR codes for each registered student
2.2 The system shall link QR codes permanently to user accounts
2.3 The system shall allow users to regenerate QR codes if compromised
2.4 The system shall display QR codes in the mobile application for easy scanning

### 3. QR Code Scanning  Validation at RVM

3.1 The RVM machine shall scan QR codes using integrated camera module
3.2 The system shall validate scanned QR codes against the user database
3.3 The system shall reject invalid or expired QR codes with appropriate error messages
3.4 The system shall complete QR validation within 2 seconds

### 4. Material Classification  Validation

4.1 The RVM shall use AI-powered sensors/camera to classify deposited items as plastic or non-plastic
4.2 The system shall achieve minimum 90% accuracy in material classification

4.3 The system shall automatically divert non-plastic waste to a separate waste compartment
4.4 The system shall log classification results for quality analytics

## 5. Liquid Detection  Bottle Validation

5.1 The RVM shall detect liquid content inside plastic bottles using weight differential and tilt sensors
5.2 The system shall reject bottles containing liquid automatically
5.3 The system shall display clear instructions to users to empty bottles before resubmission
5.4 The system shall log all rejected items with rejection reasons

## 6. Weight Measurement  Accuracy

6.1 The system shall measure weight of deposited plastic using calibrated load cell sensors
6.2 The system shall maintain measurement accuracy within ±5g tolerance
6.3 The system shall automatically calibrate sensors weekly
6.4 The system shall detect and report sensor malfunctions to administrators

## 7. Points Calculation System

7.1 The system shall calculate base points as 10 points per kilogram of plastic deposited
7.2 The system shall round weight measurements to nearest 10 grams for point calculation
7.3 The system shall apply minimum deposit threshold of 50 grams to earn points
7.4 The system shall calculate and display points within 3 seconds of deposit

## 8. Bonus Points  Incentive System

8.1 The system shall award 20% bonus for first deposit of the day
8.2 The system shall award 50% bonus for bulk deposits exceeding 5kg
8.3 The system shall award 30% bonus for maintaining weekly deposit streak (7 consecutive days)
8.4 The system shall clearly display applicable bonuses during transaction confirmation

## 9. User Points Balance Management

9.1 The system shall maintain separate tracking for total earned points, redeemed points, and available points
9.2 The system shall update point balances in real-time after each transaction
9.3 The system shall provide point balance query functionality via mobile app
9.4 The system shall prevent negative balance scenarios with validation checks

## 10. Transaction Recording  History

10.1 The system shall record all transactions including user ID, timestamp, weight, points earned, and machine location
10.2 The system shall store complete transaction history for minimum 2 months
10.3 The system shall provide transaction history access through mobile application
10.4 The system shall allow users to export their transaction history in CSV format

## 11. Real-Time Feedback Display

11.1 The RVM shall display transaction details (weight, points earned) within 3 seconds of deposit
11.2 The system shall show animated visual feedback for successful deposits
11.3 The system shall display current user ranking and points balance after transaction
11.4 The system shall provide audio confirmation for successful transactions

## 12. Student Leaderboard System

12.1 The system shall maintain real-time leaderboards ranking students within their college
12.2 The system shall update leaderboard positions immediately after each transaction
12.3 The system shall display top 100 students in each college leaderboard
12.4 The system shall show user's current rank and points difference from next rank

## 13. College Competition System

13.1 The system shall aggregate points by college affiliation
13.2 The system shall maintain university-wide college leaderboard
13.3 The system shall calculate college rankings based on total points from all students
13.4 The system shall display college competition standings in mobile app and RVM displays

## 14. Time-Based Leaderboard Filtering

14.1 The system shall provide leaderboard views filtered by daily, weekly, monthly, and all-time periods
14.2 The system shall reset daily and weekly leaderboards automatically at midnight
14.3 The system shall archive historical leaderboard data for reporting purposes
14.4 The system shall allow users to switch between different time period views seamlessly

## 15. Badge  Achievement System

15.1 The system shall highlight top 10 students with special badges (Gold, Silver, Bronze)
15.2 The system shall award "Top 3 College" badges to leading colleges
15.3 The system shall display badges prominently in user profiles and leaderboards
15.4 The system shall send notifications when users earn new badges

## 16. Cafeteria Integration System

16.1 The system shall integrate with university cafeteria POS system via secure API
16.2 The system shall synchronize available rewards with cafeteria menu items
16.3 The system shall support real-time point redemption at cafeteria checkout
16.4 The system shall handle cafeteria system outages gracefully with queue mechanism

## 17. Points Redemption Processing

17.1 The system shall validate user's available points before processing redemption
17.2 The system shall process point deduction atomically to prevent double-spending
17.3 The system shall generate unique confirmation codes for each redemption
17.4 The system shall complete redemption transactions within 5 seconds

## 18. Redemption History Management

18.1 The system shall record all redemption transactions with item details and points spent
18.2 The system shall display redemption history in mobile application
18.3 The system shall send email receipts for all redemption transactions
18.4 The system shall retain redemption history for minimum 2 months

## 19. Offline Mode Operation

19.1 The RVM shall continue operating when internet connectivity is unavailable
19.2 The system shall store transactions locally during offline periods
19.3 The system shall cache essential data (user QR codes, point balances) for offline validation
19.4 The system shall display offline status indicator on RVM screen

## 20. Automatic Data Synchronization

20.1 The system shall automatically detect when internet connectivity is restored
20.2 The system shall synchronize offline transactions with central server in chronological order
20.3 The system shall implement retry mechanism for failed synchronization attempts
20.4 The system shall notify administrators of synchronization status

## 21. Conflict Resolution System

21.1 The system shall detect and prevent duplicate transaction submissions
21.2 The system shall resolve timestamp conflicts using server-side validation
21.3 The system shall reconcile point balances after synchronization completion
21.4 The system shall log all conflict resolution actions for auditing

## 22. Guest Mode System

22.1 The RVM shall display prominent "Guest" button on main interface
22.2 The system shall allow unregistered users (faculty, staff, visitors) to deposit recyclables without QR scanning
22.3 The system shall apply same validation rules (material classification, liquid detection) to guest deposits
22.4 The system shall record guest transactions with timestamp, weight, and machine location marked as "guest"
22.5 The system shall display thank you message showing environmental impact contribution
22.6 The system shall not award points for guest deposits

## 23. Multi-Machine Management System

23.1 The system shall support management of multiple RVM machines across campus
23.2 The system shall assign unique identifiers to each RVM machine
23.3 The system shall centralize data from all machines in unified database
23.4 The system shall allow remote configuration updates to all machines

## 24. Points Expiration Policy

24.1 The system shall expire unused points after 15 days of account inactivity
24.2 The system shall send reminder notifications  7 days before points expiration
24.3 The system shall send final warning notification 3 days before expiration
24.4 The system shall execute expiration process automatically at midnight on expiration date

24.5 The system shall log all expired points transactions for auditing

## 25. Analytics Generation (Administrator)

25.1 The system shall generate detailed performance reports (transaction volumes, engagement metrics)
25.2 The system shall generate student ranking reports by college and time period
25.3 The system shall generate college competition standing reports
25.4 The system shall generate environmental impact reports (total weight, CO2 savings)
25.5 The system shall generate guest usage analytics reports
25.6 The system shall generate rejection analytics for quality monitoring
25.7 The system shall export reports in PDF, Excel, and CSV formats
25.8 The system shall schedule automated report generation and email delivery

## 26. Rewards Management (Administrator)

26.1 The system shall allow admins to create new cafeteria reward items with point costs
26.2 The system shall allow admins to update existing reward details (name, description, points)
26.3 The system shall allow admins to delete or discontinue reward items
26.4 The system shall allow admins to set reward availability status (active/inactive)
26.5 The system shall allow admins to view redemption statistics for each reward
26.6 The system shall allow admins to manage reward categories and groupings
26.7 The system shall allow admins to set seasonal or promotional reward offerings

## 27. Machine Management (Administrator)

27.1 The system shall allow admins to monitor real-time status of all RVM machines (online/offline/maintenance)
27.2 The system shall display machine locations on an interactive campus map
27.3 The system shall allow admins to check individual machine capacity levels (bin + waste)
27.4 The system shall allow admins to view machine health metrics (sensor status, error logs)
27.5 The system shall allow admins to remotely enable/disable machines for maintenance
27.6 The system shall allow admins to configure machine settings (point rates, bonus multipliers)
27.7 The system shall allow admins to manage machine deployment locations
27.8 The system shall allow admins to track machine maintenance history and schedules
27.9 The system shall allow admins to view machine usage statistics (deposits/day, peak hours)

## 28. System Monitoring Alerts (Administrator)

28.1 The system shall provide admins with real-time system status dashboard

**28.2** Capacity Monitoring & Alerting

28.2.1 The system shall send admins capacity alerts (80%, 95%, 100% full)
28.2.2 The RVM shall monitor bin capacity using ultrasonic distance sensors

28.3 The system shall allow admins to monitor synchronization status of offline machines
28.4 The system shall display system error logs and warnings to admins
28.5 The system shall allow admins to track overall system performance metrics
28.6 The system shall allow admins to monitor database connectivity and health
28.7 The system shall allow admins to view active user sessions and concurrent usage

## 29. User Management (Administrator)

29.1 The system shall allow admins to view registered user accounts and profiles
29.2 The system shall allow admins to verify student registrations manually if needed
29.3 The system shall allow admins to deactivate or suspend user accounts for policy violations
29.4 The system shall allow admins to reset user passwords upon request
29.5 The system shall allow admins to view user transaction histories
29.6 The system shall allow admins to handle user disputes and support tickets

# 5. Initial Identified Subsystems (SOLID-Compliant Design)

***"Each subsystem is designed following SOLID principles: S=Single Responsibility, O=Open/Closed, L=Liskov Substitution, I=Interface Segregation, D=Dependency Inversion"***

| Subsystem Name | Classes | Subsystem Function | Subsystem Interface |
|---|---|---|---|

| | | | |
|---|---|---|---|
| **1. Authentication User Management** | UserAccount, AuthenticationHandler, SessionManager, PasswordManager | S 1.1 Manage user registration (registration responsibility only).<br>S 1.2 Authenticate existing users (authentication responsibility only).<br>S 1.3 Handle user sessions (session management only).<br>S 1.4 Manage password security and recovery (password handling only). | I **IUserAuth:** void registerUser(UserData) boolean loginUser(Credentials)<br><br>I **ISessionManager:** Session createSession(UserID) boolean validateSession(SessionID)<br><br>I **IPasswordManager:** void resetPassword(UserID):void |
| **2. QR Code Management** | QRCodeGenerator, QRCodeValidator, QRScanner | S 2.1 Generate unique QR codes for each user (QR generation only).<br>S 2.2 Validate QR codes at RVM machine (validation responsibility).<br>S 2.3 Scan QR codes using camera module (scanning only). | I **IQRCodeService:** String generateQRCode(UserID) boolean validateQRCode(String)<br><br>I **IQRScanner:** String scanQRCode() |
| **3. Material Classification Validation** | MaterialClassifier, AIModel, ItemValidator, WasteSorter | S 3.1 Classify deposited materials as plastic or nonplastic using AI (classification only).<br>S 3.2 Validate items before acceptance based on material type (validation only).<br>S 3.3 Sort and divert nonplastic waste to separate compartment (sorting only).<br>O 3.4 Extensible for new material types without modifying existing code. | I **IMaterialClassifier:** MaterialType classifyMaterial(Item) boolean isPlastic(Item)<br><br>IO **IItemValidator:** ValidationResult validate(Item)<br>*(Extensible for new validation rules)*<br><br>I **IWasteSorter:** void sortToWaste(Item) void sortToRecyclable(Item) |

| | | | |
|---|---|---|---|
| **4. Liquid Detection Rejection** | LiquidDetector, TiltSensor, RejectionHandler | S 4.1 Detect liquid content in bottles using sensors (detection only).<br>S 4.2 Reject bottles containing liquid automatically (rejection handling).<br>S 4.3 Display clear instructions to users to empty bottles (user feedback).<br>S 4.4 Log all rejected items with rejection reasons (logging only). | I **ILiquidDetector:**<br>boolean hasLiquid(Item)<br>float getLiquidLevel(Item)<br><br>I **IRejectionHandler:**<br>void rejectItem(Item, String reason)<br>void logRejection(RejectionData) |
| **5. Weight Measurement System** | WeightSensor, SensorCalibrator, MeasurementValidator | S 5.1 Measure weight of deposited plastic (weight measurement only).<br>S 5.2 Calibrate sensors automatically (calibration responsibility).<br>S 5.3 Validate measurement accuracy (validation only). | I **IWeightSensor:**<br>float measureWeight()<br>void calibrate()<br><br>I **IMeasurementValidator:**<br>boolean validateMeasurement(float) |
| **6. Points Calculation Bonus Engine** | PointsCalculator, IBonusStrategy, FirstDepositBonus, BulkDepositBonus, WeeklyStreakBonus | S 6.1 Calculate base points from weight (calculation responsibility only).<br>SOL 6.2 Apply bonus multipliers using strategy pattern (extensible for new bonus types, strategies are substitutable).<br>O 6.3 Add new bonus types without modifying existing calculation logic. | I **IPointsCalculator:**<br>int calculatePoints(float weight)<br><br>IOL **IBonusStrategy:**<br>int applyBonus(int basePoints, Context)<br>*(Implementations: FirstDepositBonus, BulkDepositBonus, WeeklyStreakBonus - substitutable)* |
| **7. Transaction Processing Recording** | TransactionHandler, TransactionRecorder, TransactionValidator, DatabaseUpdater, **GuestTransactionProcessor** | S 7.1 Create and record transaction details (transaction creation).<br>S 7.2 Validate transaction data (validation only).<br>S 7.3 Update central database in real-time (database update task).<br>S 7.4 Process guest | I **ITransactionCreator:**<br>Transaction createTransaction<br>(Data)<br><br>I **ITransactionRecorder:**<br>void recordTransaction(Transaction)<br><br>I **IGuestTransactionProcessor:**<br>Transaction processGuestDeposit (float |

| | | | |
|---|---|---|---|
| | | deposits without authentication (guest transaction handling). | weight)<br>String generateThankYouMessage(float weight) |
| **8. Points Balance Account Management** | BalanceManager, ExpirationHandler, AccountTracker | S 8.1 Update user and college point balances (balance management only).<br>S 8.2 Track total, redeemed, and available points (tracking responsibility).<br>S 8.3 Expire points after 6 months of inactivity (expiration handling). | I **IBalanceManager:**<br>void updateBalance(UserID, int)<br>int getAvailablePoints(UserID)<br><br>I **IPointsExpiration:**<br>void expirePoints(UserID) |
| **9. Leaderboard Ranking System** | LeaderboardManager, RankingEngine, CollegeAggregator, ITimeFilter, DailyFilter, WeeklyFilter, MonthlyFilter, AllTimeFilter, BadgeSystem | S 9.1 Maintain student rankings within colleges (ranking responsibility).<br>S 9.2 Aggregate collegelevel point totals (aggregation task only).<br>S 9.3 Generate universitywide college rankings (college ranking task).<br>SOL 9.4 Filter leaderboards by time period using strategy pattern (extensible and substitutable filters).<br>S 9.5 Award badges to top contributors (badge assignment only).<br>**Note:** Guest contributions do not appear in leaderboards but contribute to campus totals. | I **IRankingEngine:**<br>void updateRankings(UserID)<br><br>I **ILeaderboardQuery:**<br>List getStudentLeaderboard(CollegeID)<br>List getCollegeLeaderboard()<br><br>IOL **ITimeFilter:**<br>List filterByPeriod(LeaderboardData)<br>*(Implementations: DailyFilter, WeeklyFilter, MonthlyFilter, AllTimeFilter - substitutable)*<br><br>I **IBadgeAssigner:**<br>void assignBadges(UserID) |

| | | | |
|---|---|---|---|
| **10. Redemption Cafeteria Integration** | RedemptionHandler, ICafeteriaService, CafeteriaAPIAdapter, ConfirmationGenerator, PointsValidator, PointsDeductor, RewardManager | SD 10.1 Interface with cafeteria system (depends on ICafeteriaService abstraction, not concrete API).<br>S 10.2 Validate user's available points for redemption (validation only).<br>S 10.3 Process point deduction atomically (deduction responsibility).<br>S 10.4 Generate confirmation codes for cafeteria (code generation task).<br>S 10.5 Record redemption transaction history (recording task only).<br>S 10.6 Allow admins to create, update, delete rewards (reward CRUD operations).<br>**Note:** Only registered users can redeem points. | ID **ICafeteriaService:**<br>boolean requestRedemption(RedemptionData)<br>*(Implementation: CafeteriaAPIAdapter)*<br><br>I **IPointsValidator:**<br>boolean validatePoints(UserID, int)<br><br>I **IPointsDeductor:**<br>void deductPoints(UserID, int)<br><br>I **IConfirmationGenerator:**<br>String generateConfirmation()<br><br>I **IRedemptionRecorder:**<br>void recordRedemption(Transaction)<br><br>I **IRewardManager:**<br>void manageRewards(RewardData) |
| **11. Guest Mode Management** | **GuestSessionHandler**, **GuestTransactionProcessor**, **GuestAnalytics** | S 11.1 Display prominent "Guest" button on RVM interface (UI responsibility).<br>S 11.2 Handle anonymous guest sessions (guest management only).<br>S 11.3 Process guest deposits without QR scanning (guest transaction handling).<br>S 11.4 Record guest transactions with timestamp and machine location (recording only).<br>S 11.5 Display thank you message showing environmental impact (feedback generation).<br>S 11.6 Aggregate guest contributions separately ( analytics tracking). | I **IGuestSessionManager:**<br>GuestSession createGuestSession()<br>void endGuestSession(SessionID)<br><br>I **IGuestTransactionProcessor:**<br>Transaction processGuestDeposit(float weight)<br>String generateThankYouMessage(float weight)<br><br>I **IGuestAnalytics:**<br>GuestData aggregateGuestData()<br>float getGuestContribution() |

| | | | |
|---|---|---|---|
| **12. Data Synchronization Offline Mode** | StorageProvider, LocalStorageProvider, CloudStorageProvider, SyncManager, DatabaseUpdater, ConflictResolver | S 12.1 Continue RVM operation when internet unavailable (offline operation).<br>SL 12.2 Store transactions locally or in cloud (storage abstraction with substitutability).<br>S 12.3 Synchronize offline data with server automatically (sync-only responsibility).<br>S 12.4 Implement conflict resolution mechanisms (conflict handling only).<br>S 12.5 Update central database when online (database update task). | IL **IStorageProvider:**<br>void store(Transaction)<br>List retrieve()<br>*(Implementations: LocalStorageProvider, CloudStorageProvider - substitutable)*<br>I **ISyncManager:**<br>SyncResult syncOfflineTransactions():SyncResult boolean isOnline()<br><br>I **IConflictResolver:**<br>void resolveConflict(Transaction)<br><br>I **IDatabaseUpdater:**<br>void updateDatabase(Transaction) |
| **13. Admin Analytics Reporting** | ReportGenerator, AnalyticsEngine, ExportHandler, EmailScheduler, ChartBuilder, DataAggregator | S 13.1 Generate system performance reports (transaction volumes, user engagement metrics) (report generation responsibility only).<br>S 13.2 Generate student ranking reports by college and time period (ranking analytics responsibility only).<br>S 13.3 Generate college competition standings reports (competition analytics responsibility only).<br>S 13.4 Generate environmental impact reports (total weight recycled, CO2 savings, sustainability metrics) (environmental analytics responsibility only).<br>S 13.5 Generate guest usage analytics and contribution reports (guest analytics responsibility only).<br>S 13.6 Generate rejection analytics for quality monitoring (rejection tracking responsibility only).<br>O 13.7 Export reports in multiple formats without modifying existing code (export extensibility).<br>D 13.8 Schedule automated report generation using abstracted email service (scheduling abstraction). | I **ReportGenerator:**<br>Report generatePerformanceReport (DateRange)<br>Report generateRankingReport (CollegeID, Period)<br>Report generateEnvironmentalReport (DateRange)<br>I **AnalyticsEngine:**<br>Analytics aggregateTransactionData(DateRange)<br>Analytics calculateEngagementMetrics (UserID)<br>I **ExportHandler:**<br>File exportToPDF(Report)<br>File exportToExcel(Report)<br>File exportToCSV(Report)<br>I **EmailScheduler:**<br>void scheduleReport (ReportType, Schedule)<br>void sendReportEmail(Report, Recipients) |

| | | | |
|---|---|---|---|
| **14. Admin Management** | RewardManager, MachineController, SystemMonitor, UserAdministrator, AlertManager, ConfigurationManager | S 14.1 Manage cafeteria reward catalog (create, update, delete rewards) (reward CRUD responsibility only).<br>S 14.2 Configure reward availability and point costs (reward configuration responsibility only).<br>S 14.3 Monitor real-time status of all RVM machines (online/offline/maintenance) (machine monitoring responsibility only).<br>S 14.4 Control machine operations remotely (enable/disable, configure settings) (machine control responsibility only).<br>S 14.5 View machine locations on interactive campus map (location display responsibility only).<br>S 14.6 Track machine capacity, health metrics, and usage statistics (machine analytics responsibility only).<br>S 14.7 Display real-time system health dashboard (system monitoring responsibility only).<br>S 14.8 Send capacity and maintenance alerts to administrators (alert management responsibility only).<br>S 14.9 Monitor synchronization status and database health (sync monitoring responsibility only).<br>S 14.10 View and manage user accounts (view, verify, suspend) (user administration responsibility only).<br>S 14.11 Handle password reset requests from users (password management responsibility only).<br>D 14.12 Interface with external systems via abstracted service interfaces (dependency abstraction). | **RewardManager:**<br>void createReward(RewardData)<br>void updateReward(RewardID, RewardData)<br>void deleteReward(RewardID)<br>List getActiveRewards()<br>RewardStats getRedemptionStats(RewardID)<br>**MachineController:**<br>MachineStatus getMachineStatus(MachineID)<br>void toggleMachine(MachineID, boolean)<br>void updateMachineConfig(MachineID, Config)<br>List getAllMachines()<br>**SystemMonitor:**<br>Health getSystemHealth()<br>List getActiveAlerts()<br>Status getSyncStatus()<br>**UserAdministrator:**<br>User getUserProfile(UserID)<br>void suspendUser(UserID, String reason)<br>void resetUserPassword (UserID)<br>List getUserHistory(UserID) |

## 6. Traceability Matrix

***"This traceability matrix links each functional requirement (from Section 4) to the subsystem(s) responsible for its implementation."***

| Req. # | Requirement Description (Summary) | Auth User | QR Code | Material | Liquid | Weight | Points | Balance | Leaderboard | Redemption | Guest | Sync | Analytics | Administrator |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | User Registration & Authentication Management | ✓ | | | | | | | | | | | | |
| 2 | QR Code Generation & Management | ✓ | ✓ | | | | | | | | | | | |
| 3 | QR Code Scanning & Validation at RVM | ✓ | ✓ | | | | | | | | | | | |
| 4 | Material Classification & Validation | | | ✓ | | | | | | | | | | |
| 5 | Liquid Detection & Bottle Validation | | | | ✓ | | | | | | | | | |
| 6 | Weight Measurement & Accuracy | | | | | ✓ | | | | | | | | |
| 7 | Points Calculation System | | | | | ✓ | ✓ | | | | | | | |
| 8 | Bonus Points & Incentive System | | | | | | ✓ | | | | | | | |
| 9 | User Points Balance Management | | | | | | | ✓ | | | | | | |
| 10 | Transaction Recording & History | | | | | | | | | | | ✓ | | |
| 11 | Real-Time Feedback Display | | | | | ✓ | ✓ | ✓ | ✓ | | | | | |
| 12 | Student Leaderboard System | | | | | | | | ✓ | | | | | |

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **13** | College Competition System | | | | | | | | ✓ | | | | | |
| **14** | Time-Based Leaderboard Filtering | | | | | | | | ✓ | | | | | |
| **15** | Badge & Achievement System | | | | | | | | ✓ | | | | | |
| **16** | Cafeteria Integration System | | | | | | | | | ✓ | | | | |
| **17** | Points Redemption Processing | | | | | | | ✓ | | ✓ | | | | |
| **18** | Redemption History Management | | | | | | | | | ✓ | | | | |
| **19** | Offline Mode Operation | | | | | | | | | | | ✓ | | |
| **20** | Automatic Data Synchronization | | | | | | | | | | | ✓ | | |
| **21** | Conflict Resolution System | | | | | | | ✓ | | | | ✓ | | |
| **22** | Guest Mode System | | | | | | | | | | ✓ | | | |
| **23** | Multi-Machine Management System | | | | | | | | | | | ✓ | | ✓ |
| **24** | Capacity Monitoring & Alerting | | | | | ✓ | | | | | | | | ✓ |
| **25** | Points Expiration Policy | ✓ | | | | | | ✓ | | | | | | |

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **26** | Analytics Generation | | | | | | | | | | ✓ | | ✓ | |
| **27** | Rewards Management | | | | | | | | | ✓ | | | | ✓ |
| **28** | Machine Management | | | | | | | | | | | | | ✓ |
| **29** | System Monitoring & Alerts | | | | | | | | | | | ✓ | | ✓ |
| **30** | User Management | ✓ | | | | | | | | | | | | ✓ |

## 7. Initial GUI Sketches

### Mobile Application:

- **Home Screen:** Displays large QR code for scanning, current points balance with animated counter, quick stats (deposits today/this week), and user's current rank badge.
- **QR Scan Page (RVM Display):** Shows instructions for depositing plastic, real-time weight display during deposit, and transaction confirmation with points earned animation.
- **Leaderboard Page:** Three tabs (My College, All Colleges, My Rank), filterable by time period (daily/weekly/monthly/all-time), displays top contributors with avatars and point totals.
- **Transaction History:** Scrollable list of all deposits showing date, time, weight, points earned, and machine location with search functionality.
- **Redemption Page:** Browse available cafeteria items with point costs, confirmation dialog before redemption, displays confirmation code after successful redemption.
- **Profile Page:** Displays student information, college badge, total points earned/redeemed, environmental impact stats (kg recycled, CO2 saved equivalent).

### RVM Machine Interface:

- **Welcome Screen:** Two prominent options: "I'm a Registered Student" (QR code icon) and "Guest Mode" (universal recycling icon). Shows machine status, capacity indicator, and motivational message.

- **Registered Student Flow:**

- **QR Scan Screen:** Camera viewfinder with “Scan your QR code” message.
- **Item Validation:** "Analyzing item..." message while MaterialClassifier and LiquidDetector work.
- **Rejection Feedback:** If rejected: Clear message ("Please empty bottle" or "Non-plastic item detected - diverted to waste") with visual icons.
- **Weight Measurement:** Real-time weight display with animated progress bar (if item accepted).
- **Transaction Confirmation:** Shows weight, points earned (with bonuses), new balance, current rank.

❖ **Guest Mode Flow:**

- **Welcome Message:** "Thank you for contributing to campus sustainability!"
- **Instructions:** Simple visual guide showing how to deposit plastic.
- **Item Validation:** Same validation as registered users (material classification + liquid detection).
- **Rejection Feedback:** Same clear rejection messages as registered flow.
- **Weight Display:** Real-time weight measurement with animated progress bar (if item accepted).
- **Thank You Screen:** Shows weight contributed, equivalent environmental impact (e.g., "5 water bottles recycled = 0.15kg CO2 saved"), and total campus contribution today. Option to "Learn More About Our Program" with QR code to download student app.

## Mobile Application & RVM Machine Interface Prototype :

❖ https://www.figma.com/make/m60CsUQZHsDM8dk78AJd3x/3-prototype-of-RVM-machine--app-webite-rvm-machine-interface-?node-id=0-1&p=f&t=37yP96t0RkJJdfuX-0&fullscreen=1

## Admin Dashboard :

❖ **Overview Page:** Shows total active machines, machine capacity status with alerts for 80% full machines, system status indicators, and real-time alerts. Guest contribution widget showing daily guest deposits vs registered user deposits.

❖ **Analytics Page:** Displays charts for recyclables collected over time, points distribution by college, machine usage statistics, and college competition standings. Guest analytics section with guest usage trends, peak guest usage times, and guest-to-registered user ratio. Rejection analytics showing rejection rates, common rejection reasons, and trends over time.

❖ **Machine Control Page:** Map view of all RVM locations across university, individual machine status (online/offline/maintenance/80% capacity), bin fill levels (recyclable + waste compartments), and remote enable/disable controls.

❖ **Reports Page:** Generate and download detailed reports for total transactions, student rankings, college performance, total weight recycled, and environmental impact metrics. Guest contribution report showing guest usage patterns. Rejection analysis report for user education campaigns.

❖ **Rewards Management Page:** Create, edit, and delete cafeteria reward items with point costs, view redemption statistics, and manage reward availability.

❖ **Quality Monitoring:** Dashboard showing validation system performance, rejection reasons breakdown, contamination prevention metrics, and user education opportunities.

❖ https://www.figma.com/make/VAET9CvjBQE7JnsU8CbvnJ/Smart-Reverse-Vending-Machine-UI?node-id=0-1&p=f&t=tkZDjZb4hInEaLah-0

## 8. Impact Analysis: Enhanced Features

### Benefits of Guest Mode:

- **Increased Participation:** Potentially 3-5x more recyclables collected by including entire campus community.
- **Inclusive Sustainability:** Faculty, staff, and visitors actively participate in environmental goals.
- **Marketing & Awareness:** Guest mode serves as a gateway to full program registration.
- **Better Metrics:** True campus-wide environmental impact measurement.
- **Social Proof:** Higher total contribution numbers motivate registered users.
- **Operational Efficiency:** Better capacity utilization of RVM machines.

### Benefits of Intelligent Validation System:

- **Recycling Quality:** Prevents contamination by separating non-plastic waste automatically.
- **Machine Protection:** Liquid detection prevents machine damage and maintenance issues.
- **User Education:** Clear rejection feedback teaches proper recycling behavior.
- **Operational Efficiency:** Reduces manual sorting and post-processing costs.
- **Data-Driven Improvement:** Rejection analytics identify patterns for targeted education campaigns.
- **Environmental Impact:** Higher quality recyclables increase actual recycling rates vs. landfill diversion.
- **Scalability:** Automated validation allows deployment without constant human supervision.

### SOLID Compliance:

- **SRP:** MaterialClassifier, LiquidDetector, ItemValidator, WasteSorter, GuestSessionHandler, GuestTransactionProcessor each have single, focused responsibilities.
- **OCP:** New validation rules (AluminumDetector), report types (RejectionAnalyticsReport, GuestAnalyticsReport), and bonus strategies can be added without modifying existing code.
- **LSP:** Different storage providers, time filters, bonus strategies, and validators can be substituted seamlessly.
- **ISP:** Separate focused interfaces (IMaterialClassifier, ILiquidDetector, IItemValidator, IGuestSessionManager, IGuestTransactionProcessor) instead of monolithic interfaces.
- **DIP:** High-level modules depend on abstractions (IItemValidator, IStorageProvider, ISensor) not concrete implementations.

### Technical Considerations:

- **Database Schema:** Transaction table includes user_type (enum: 'registered', 'guest'), validation_status ('accepted', 'rejected_liquid', 'rejected_material'), material_type ('plastic', 'glass', 'aluminum', 'other').
- **Hardware Integration:** Camera module for visual classification, weight sensors for liquid detection, dual compartment system (recyclable + waste).
- **Abuse Prevention:** Rate limiting on guest deposits (max 5 per hour per machine); rejection logging to identify malicious behavior.

- **Offline Support:** Guest transactions and validation results cached locally during offline mode.
- **Analytics Separation:** Clear distinction between registered user metrics and guest metrics for accurate gamification.

## 9. Competitive Analysis in Egypt

*In Egypt, several startups and initiatives have emerged to address plastic waste management and recycling through smart technologies. The most relevant competitors to our Reverse Vending Machine (RVM) project are **Recyclobekia**, **Dawarha**, **ZeroPrime Technologies (CanBank)**, **Bekia**, and traditional campus recycling programs.*

***Recyclobekia** operates smart recycling bins and reverse vending machines in select Cairo locations, primarily targeting shopping malls and corporate offices. Their machines provide basic monetary rewards (cash or vouchers) but lack gamification elements and campus-specific features. Recyclobekia has limited presence in educational institutions and does not offer college competition frameworks or integration with campus services.*

***Dawarha** is currently the most prominent company in the Egyptian recycling sector. It manufactures AI-powered Reverse Vending Machines capable of identifying and collecting plastic bottles and cans. Dawarha has partnered with major brands such as Nestlé and Coca-Cola and has deployed multiple machines across Cairo, collecting thousands of bottles monthly. Their focus is on largescale deployment and public environmental awareness in commercial areas, rather than educational institutions. While technologically advanced, Dawarha's machines do not incorporate student-specific features, college competitions, or campus service integration.*

***ZeroPrime Technologies**, through its product **CanBank**, offers smart vending-style recycling machines placed in malls and some universities. The system rewards users with phone credit or digital vouchers when they deposit bottles or cans. While ZeroPrime has notable local visibility and demonstrates effective public engagement through basic incentivization, its deployment scale remains smaller than Dawarha's. CanBank lacks comprehensive gamification features, leaderboard systems, and the closed-loop campus integration our project offers.*

***Bekia** operates differently as a digital platform for waste exchange rather than deploying physical RVMs. Users can exchange recyclable materials for goods or services through scheduled pickups, targeting households and businesses. While Bekia successfully connects waste collectors with users, it does not provide automated, real-time recycling solutions or campus-specific functionality. The platform's reliance on scheduled pickups makes it unsuitable for instant, on-demand campus recycling needs.*

***Traditional campus recycling initiatives** across Egyptian universities (AUC, GUC, Cairo University) rely on manual collection bins and volunteer-led recycling clubs. These programs suffer from inconsistent operation, lack of tracking mechanisms, poor participation rates due to absence of incentives, and no data analytics capabilities. Student engagement remains low, and environmental impact measurement is virtually non-existent.*

***"Overall, while existing competitors have demonstrated market viability for recycling technologies in Egypt, none currently integrate secure QR-based student identification, university-level competition frameworks, campus service integration, guest participation, and comprehensive offline capability as our proposed project does. Our system fills this gap by combining secure RVM hardware with a reward-based mobile application, institutional gamification, and inclusive participation— specifically designed to foster sustainability and environmental awareness within Egyptian university communities."***