Software Requirements Specification (SRS)

Campus Sustainability & Resource Sharing Platform

Version 1.0 — Prepared by ■dal Özer

Revision History

Version 1.0 — 2025-11-09 — Initial detailed SRS prepared by ■dal Özer

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) documents the functional and non-functional requirements for the Campus Sustainability & Resource Sharing Platform. It provides a complete and verifiable description of system behavior and constraints to guide design, implementation, testing, and deployment. This SRS covers the full product for an initial campus-wide release.

1.2 Document Conventions

Requirement identifiers use the form REQ-... Priority levels are H (High), M (Medium), L (Low). Text in italics indicates examples. 'TBD' indicates information to be determined.

1.3 Intended Audience and Reading Suggestions

Primary readers: developers, testers, UI/UX designers, project manager, deployment engineers, university stakeholders and testers. Read Section 2 for a high-level overview, Section 4 for functional requirements, and Section 5 for non-functional requirements. Appendices contain glossary and analysis models.

1.4 Product Scope

The Platform is a responsive web application (desktop + mobile) that combines: (1) resource-sharing (post, search, request items), (2) event management for sustainability activities, (3) QR-enabled recycling check-ins that award eco-points, and (4) an eco-point gamification system with leaderboards and partner rewards. Target launch: one university campus with potential multi-campus scaling.

1.5 References

UML preliminary document – ■dal Özer (UML ÖN ÇALI■MA). SRS IEEE template (Wiegers).

2. Overall Description

2.1 Product Perspective

The product is a new, self-contained web application intended to integrate with the university's existing authentication and email systems. It may later integrate with campus card systems and external reward partners' APIs. The platform exposes RESTful APIs for mobile client or third-party integration.

2.2 Product Functions

- User account management (register, login, profile, roles)
- Item posting, searching, requesting, lending, donating and review workflow
- Event creation, registration, attendance tracking and admin approval
- QR-based recycling check-in, validation, and eco-point awarding
- Eco-point ledger, personal dashboard, and leaderboard views
- Notifications (in-app, email) and admin moderation tools
- Partner organization management and reward redemption

2.3 User Classes and Characteristics

Student (end-user): Typical user with web/smartphone access; non-technical. Uses search, posting, check-in, and event participation features. Club Leader: student with verified club privileges to create/manage events. Admin: university staff with content moderation, analytics and system settings access. Partner Organization: external entity offering sponsorships or rewards and viewing campaign analytics (limited scope).

2.4 Operating Environment

Host: Cloud-hosted web server (Linux), PostgreSQL or MySQL database. Browser support: latest two versions of Chrome, Edge, Safari, Firefox. Mobile: responsive web or PWA on Android/iOS. Email (SMTP) and optional SMS gateway for OTP.

2.5 Design and Implementation Constraints

H: Must integrate with campus SSO (e.g., OAuth2/SAML) if available. M: Use of university-approved cloud provider. L: Use standard frameworks (React/Vue for front-end; Node/Django/Flask for backend).

2.6 User Documentation

User manual (web), quick-start tutorials, FAQ, and admin guide. In-platform contextual help and tooltips.

2.7 Assumptions and Dependencies

Assumes stable campus authentication service, availability of QR code generation/printing for bins, and initial partner agreements for rewards. Dependencies: SMTP, image storage (S3 or equivalent), push notification service (optional).

3. External Interface Requirements

3.1 User Interfaces

Responsive web UI with the following primary screens: Login/Register, Dashboard (personal eco-summary), Item Feed (search & filters), Item Detail & Request, Create Item form, Club Event Management, Event Detail & RSVP, QR Check-In scanner view (mobile), Leaderboard and Admin Panel. Error messages must be clear and actionable; forms must validate client-side and server-side. Accessibility: meet WCAG 2.1 AA where practical.

3.2 Hardware Interfaces

No custom hardware required. QR codes will be printed and attached to recycling bins; smartphones will act as scanners using camera APIs. Optional integration with campus card readers may be defined later.

3.3 Software Interfaces

Authentication: OAuth2/SAML (campus SSO) or email/password fallback. Database: PostgreSQL/MySQL. Storage: object store for images. External APIs: optional partner APIs for rewards, email (SMTP), and SMS gateway.

3.4 Communications Interfaces

All communications over HTTPS (TLS 1.2+). RESTful JSON APIs for client-server interaction. Webhooks for partner integrations. Rate limiting and API keys for third-party access.

4. System Features

4.1 Account Management (Register, Login, Profile, Roles)

Priority: H

Manage users, authentication, authorization, role assignment and profile settings.

Stimulus/Response Sequences:

- User submits registration form
- System validates and creates account
- User verifies email
- User logs in and is redirected to dashboard

Functional Requirements:

REQ-1.1: Users can register using university email or SSO. (H)
REQ-1.2: Email verification required for email/password accounts. (H)
REQ-1.3: Password reset via email OTP. (M)
REQ-1.4: Role-based access: student, club_leader, admin, partner. (H)

Alternative / Variant Requirements:

ALT-1.1: Allow sign-up via Google/Apple when campus SSO is not available. (M)
ALT-1.2: Support account linking for students with multiple campus identities. (L)

4.2 Item Sharing (Post, Search, Request, Lend, Donate)

Priority: H

Core marketplace for sharing physical goods between students with moderation.

Stimulus/Response Sequences:

- User creates item post with photos and description
- Admin reviews and approves
- Others search & request item
- Owner accepts request and arranges handoff

Functional Requirements:

REQ-2.1: Users can create item posts with images, category, condition, location, and tags. (H)
REQ-2.2: Posts are set to 'pending' until admin approval; owner notified upon status change. (H)
REQ-2.3: Search supports filters: category, distance (campus areas), condition, availability, and sort by newest or eco-points impact. (M)
REQ-2.4: Messaging: requester and owner can communicate via in-app messaging; messages stored and rate-limited. (M)
REQ-2.5: Owners can mark item as lent, donated, available or removed. (H)

Alternative / Variant Requirements:

ALT-2.1: Allow peer-review and reputation for users to reduce admin backlog. (M)
ALT-2.2: Automatic categorization suggestions via image recognition (optional). (L)

4.3 Event Management (Create, Approve, RSVP, Attendance)

Priority: M

Club leaders can create sustainability events; admins moderate; attendees register.
Stimulus/Response Sequences:
• Club leader submits event
• Admin approves
• Students RSVP
• Event organisers record attendance or system verifies via QR at event
Functional Requirements:
REQ-3.1: Event creation requires title, description, date/time, location, capacity, and required resources. (H)
REQ-3.2: Admin approval required before public listing. (H)
REQ-3.3: RSVP system with optional waitlist when capacity reached. (M)
REQ-3.4: Attendance tracked via check-in QR codes or admin manual marking. (M)
REQ-3.5: Event participants earn eco-points per event policy. (M)
Alternative / Variant Requirements:
ALT-3.1: Recurring events support (weekly recycling drives). (M)
ALT-3.2: Allow external collaboration requests from partner organizations (M)

4.4 QR Recycling Check-In & Eco-Points Accounting
Priority: H
Students scan QR codes at recycling bins to log activity and earn eco-points; system validates scans and prevents fraud.
Stimulus/Response Sequences:
• Student scans QR at bin
• System validates QR, checks recent scans, awards points
• Leaderboards updated
Functional Requirements:
REQ-4.1: QR codes are unique per bin and signed to prevent tampering. (H)
REQ-4.2: Points awarded according to bin type and current campaigns; points stored in immutable ledger. (H)
REQ-4.3: Rate-limiting: prevent awarding points more than once per 24 hours per bin per user unless campaign allows. (H)
REQ-4.4: Admin can flag suspicious accounts for review. (H)
Alternative / Variant Requirements:
ALT-4.1: Allow manual check-in by attendees for special events when QR unavailable. (M)
ALT-4.2: Temporary boosted points for campaign periods. (M)

4.5 Eco Leaderboard, Badges and Partner Rewards
Priority: M
Display rankings, allow badge achievements and reward redemption with partners.
Stimulus/Response Sequences:
• System recalculates ranking nightly
• Users view ranks, redeem rewards
• Partners confirm redemptions
Functional Requirements:
REQ-5.1: Leaderboard shows top users, top clubs, and weekly/monthly filters. (M)
REQ-5.2: Badge system awards achievements for milestones (e.g., 50 scans, 10 donations). (M)
REQ-5.3: Reward redemption requires partner confirmation and may require in-person verification. (M)
Alternative / Variant Requirements:
ALT-5.1: Allow private leaderboard among clubs or groups. (L)

4.6 Admin Console & Analytics
Priority: H
Admin tools for moderation, content approval, user management, and analytics dashboards for campaign performance.
Stimulus/Response Sequences:
• Admin reviews pending items/events
• Admin views analytics dashboards
• Admin configures campaigns
Functional Requirements:
REQ-6.1: Admin can approve/reject posts and events; reason logged. (H)
REQ-6.2: Analytics: daily active users (DAU), scans/day, items posted, requests fulfilled, redemption rates. (M)
REQ-6.3: Admin can configure campaign rules and point values. (M)
Alternative / Variant Requirements:
ALT-6.1: Role-based admin tiers (content moderator vs campus admin). (M)

5. Other Nonfunctional Requirements

5.1 Performance Requirements

REQ-PERF-1: Typical pages (dashboard, item feed) should load within 2.5 seconds on campus network. REQ-PERF-2: System should support 5,000 concurrent users for campus peak times with horizontal scaling. REQ-PERF-3: API responses (95th percentile) under 300ms.

5.2 Safety Requirements

The system must not store sensitive personal data beyond necessary profile fields. Any PII storage follows university GDPR-equivalent policies. There are no life-critical controls.

5.3 Security Requirements

REQ-SEC-1: All data in transit must be encrypted (TLS 1.2+). REQ-SEC-2: Passwords hashed with bcrypt or stronger. REQ-SEC-3: Role-based access control enforced server-side. REQ-SEC-4: Audit logs for moderation actions retained for 180 days. REQ-SEC-5: Protection against common web attacks (XSS, CSRF, SQL injection).

5.4 Software Quality Attributes

Usability: Onboarding flow less than 3 steps. Reliability: 99.9% uptime SLA target. Maintainability: modular architecture and documented APIs. Portability: containerized deployment (Docker).

5.5 Business Rules

Only verified student emails can register as students. Club leaders require manual verification by admins. Eco-points cannot be exchanged for cash. Admin decisions are final for content moderation.

6. Other Requirements

6.1 Internationalization

Support for Turkish and English in UI (initial). Date/time formats localized to user's locale.

6.2 Data Retention

User data retention policy: inactive accounts may be deleted after 2 years; images retained for 1 year after post removal unless legally required otherwise.

6.3 Accessibility

Meet WCAG 2.1 AA where feasible; keyboard navigation and screen reader compatibility for core flows.

7. Fully Dressed Use Cases (selected examples)

UC1: Register & Login
Primary Actor: Student
Preconditions: Internet access, valid campus email (if SSO not used).
Main Success Scenario:
• User navigates to Register page and selects 'Register with email' or 'Sign in with SSO'.
• User fills required fields and submits.
• System validates input and sends email verification (if email/password).
• User clicks verification link; system activates account.
• User signs in and is routed to dashboard.
Extensions:
E1: Invalid email format → show field error. E2: Email already exists → show account recovery option.
Frequency: Daily

UC4: QR Recycling Check-In & Eco Points
Primary Actor: Student
Preconditions: Logged in and camera permission granted.
Main Success Scenario:
• Student opens QR scanner in app and scans bin QR.
• Client sends signed QR payload to server for validation.
• Server validates signature, checks rate-limit, awards points, and logs transaction.
• Server updates user's eco-point account and recalculates leaderboard.
• Student receives confirmation and points summary.
Extensions:
E1: Invalid QR → show error and report option. E2: Duplicate scan within restricted window → show message with next eligible time.

Appendix A: Glossary

Eco-point: A unit of reward for sustainable actions logged in the system.
SSO: Single Sign-On service such as OAuth2 or SAML.
PWA: Progressive Web App.
Admin: University staff with elevated privileges for moderation and configuration.

Appendix B: Analysis Models

Included in the project deliverables: Use Case diagrams, high-level class diagram, and data flow diagrams (to be attached).

Appendix C: To Be Determined

• Integration specifics with campus SSO (SAML endpoints, attributes) - TBD
• Partner API contract for reward redemptions - TBD
• Exact eco-point to reward mapping and campaign rules - TBD