Product Requirements Document: SubSentry

Product Name: SubSentry - Subscription Tracker
Version: 1.0
Document Owner: Product Management Team
Last Updated: November 18, 2025
Status: Active Development

Executive Summary

SubSentry is a web-based subscription management application designed to help users track, manage, and optimize their recurring subscription expenses. The application addresses the growing challenge of subscription fatigue and hidden costs by providing a centralized platform for monitoring all subscription services, tracking renewal dates, and analyzing spending patterns.

Product Vision

To empower users to take control of their subscription expenses by providing intuitive tracking, timely renewal reminders, and actionable spending insights, ultimately helping them make informed decisions about their subscription portfolio.

Problem Statement

Modern consumers subscribe to multiple services across entertainment, software, fitness, education, and cloud storage platforms. Key pain points include:

- Difficulty tracking multiple subscriptions across different billing cycles
- Unexpected charges from forgotten or unused subscriptions
- Lack of visibility into total monthly and annual subscription costs
- Missing renewal dates leading to unwanted auto-renewals
- No centralized view of subscription spending by category

Target Audience

1. **Primary Users:** Young professionals and students (ages 18-35) managing 5-15 active subscriptions
2. **Secondary Users:** Families tracking shared subscription services
3. **Tertiary Users:** Small business owners managing software and service subscriptions

User Characteristics:

- Tech-savvy individuals comfortable with web applications
- Budget-conscious users seeking expense optimization
- Users with multiple streaming, software, and digital service subscriptions
- Indian market focus (INR currency support)

Goals and Success Metrics

Business Goals

1. Help users identify and eliminate unused subscriptions
2. Improve user financial awareness regarding recurring expenses
3. Provide value through simple, accessible subscription management

Success Metrics

Metric	Target
Active users tracking subscriptions	1,000+ users in first quarter
Average subscriptions tracked per user	8-12 subscriptions
User retention rate	60% after 30 days
Average cost savings identified	₹2,000/month per user
Feature adoption (filtering)	40% of active users

Table 1: Key Performance Indicators

Product Features and Requirements

1. Subscription Management

Priority: P0 (Must Have)

Description: Core functionality allowing users to add, view, and delete subscription entries.

Functional Requirements:

- FR1.1: Users shall be able to add new subscriptions with the following required fields:
  - Service name (text input)
  - Cost in Indian Rupees (numeric input)
  - Billing cycle (dropdown: Monthly, Quarterly, Yearly)
  - Next renewal date (date picker)
  - Category (dropdown with predefined options)
- FR1.2: Users shall be able to view all subscriptions in a list format with complete details
- FR1.3: Users shall be able to delete any subscription with a single click
- FR1.4: System shall validate all required fields before allowing submission
- FR1.5: System shall store subscription data locally using browser localStorage

User Stories:

- As a user, I want to add my Netflix subscription so I can track when it renews
- As a user, I want to delete cancelled subscriptions so my list stays current
- As a user, I want to see all my subscription details at a glance

2. Category Organization

Priority: P0 (Must Have)

Description: Categorization system for organizing subscriptions by service type.

Supported Categories:

- Entertainment (streaming services, gaming)
- Software (productivity tools, development platforms)
- Fitness (gym memberships, fitness apps)
- Education (online courses, learning platforms)
- Cloud Storage (file storage services)
- Other (miscellaneous services)

Functional Requirements:

- FR2.1: System shall provide predefined category options in dropdown
- FR2.2: Category field shall default to "Entertainment"
- FR2.3: Each subscription must be assigned to exactly one category

3. Billing Cycle Support

Priority: P0 (Must Have)

Description: Support for multiple billing frequencies to accommodate different subscription models.

Supported Billing Cycles:

- Monthly (billed every month)
- Quarterly (billed every 3 months)
- Yearly (billed annually)

Functional Requirements:

- FR3.1: System shall support Monthly, Quarterly, and Yearly billing cycles
- FR3.2: System shall correctly calculate monthly equivalent costs for all billing cycles
- FR3.3: Billing cycle shall default to "Monthly"

4. Dashboard and Analytics

Priority: P0 (Must Have)

Description: Real-time dashboard displaying key subscription metrics and spending analysis.

Dashboard Metrics:

Metric	Calculation
Total Subscriptions	Count of active subscriptions
Monthly Cost	Sum of monthly equivalent costs
Yearly Cost	Monthly cost × 12
Renewing Soon	Count of subscriptions renewing within 7 days

Table 2: Dashboard metrics and their calculations

Functional Requirements:

- FR4.1: Dashboard shall display total number of active subscriptions
- FR4.2: Dashboard shall calculate and display total monthly cost (converting quarterly and yearly to monthly equivalents)
- FR4.3: Dashboard shall calculate and display projected yearly cost
- FR4.4: Dashboard shall show count of subscriptions renewing within the next 7 days
- FR4.5: Dashboard metrics shall update in real-time when subscriptions are added or deleted
- FR4.6: All costs shall be displayed in Indian Rupees (₹) with proper formatting

5. Subscription Filtering

Priority: P1 (Should Have)

Description: Filter functionality to view subscriptions by category.

Functional Requirements:

- FR5.1: Users shall be able to filter subscriptions by category using a dropdown
- FR5.2: Filter shall include "All Categories" option to show all subscriptions
- FR5.3: Filter shall update the subscription list dynamically without page reload
- FR5.4: Dashboard metrics shall reflect filtered subscriptions

6. Data Persistence

Priority: P0 (Must Have)

Description: Local storage implementation for saving user data across sessions.

Functional Requirements:

- FR6.1: System shall save all subscription data to browser localStorage
- FR6.2: System shall automatically load saved subscriptions on page load
- FR6.3: System shall persist data across browser sessions
- FR6.4: Data shall remain available until user clears browser data

Technical Specifications:

- Storage method: Browser localStorage API
- Data format: JSON serialization
- Storage key: "subscriptions"

User Interface Design

Design Principles

1. **Simplicity:** Clean, uncluttered interface focused on core functionality
2. **Clarity:** Clear visual hierarchy with prominent dashboard metrics
3. **Responsiveness:** Adaptive layout for desktop and mobile devices
4. **Accessibility:** Proper form labels, semantic HTML, keyboard navigation support

Visual Design

Color Scheme:

- Primary color: Teal (#21808D) for actions and interactive elements
- Background: Cream (#FCFCF9) for reduced eye strain
- Text: Dark slate (#13343B) for readability
- Accents: Red for delete actions, orange for warnings

Typography:

- Font family: System fonts (-apple-system, BlinkMacSystemFont, Segoe UI, Roboto)
- Hierarchy: Clear distinction between headings, body text, and labels

Layout Components

1. **Header:** Application title with emoji icon (SubSentry)
2. **Dashboard Cards:** Four-column grid displaying key metrics
3. **Filter Section:** Dropdown for category filtering
4. **Add Subscription Form:** Card-based form with all required fields
5. **Subscription List:** Card-based list view with individual subscription details

Responsive Behavior

- Desktop: Multi-column dashboard, side-by-side form and list
- Tablet: Two-column dashboard, stacked form and list
- Mobile: Single-column layout, full-width components

Technical Requirements

Frontend Technologies

- HTML5 for semantic structure
- CSS3 for styling with custom properties (CSS variables)
- Vanilla JavaScript (ES6+) for functionality
- No external dependencies or frameworks required

Browser Support

Browser	Minimum Version
Chrome	90+
Firefox	88+
Safari	14+
Edge	90+

Table 3: Supported browsers

Performance Requirements

- NFR1: Page load time shall be under 2 seconds on 3G connection
- NFR2: UI updates (add/delete) shall complete within 100ms
- NFR3: Application shall support up to 100 subscriptions without performance degradation
- NFR4: LocalStorage operations shall complete within 50ms

Data Storage Limitations

- Maximum storage: ~5-10MB (localStorage browser limit)
- Estimated capacity: 500-1000 subscriptions before reaching limits
- No backend synchronization or cloud backup

Accessibility Requirements

- WCAG 2.1 Level AA compliance target
- Keyboard navigation for all interactive elements
- Proper ARIA labels for form fields
- Minimum contrast ratio of 4.5:1 for text
- Form validation with clear error messages

User Workflows

Add New Subscription Workflow

1. User navigates to "Add New Subscription" form
2. User enters service name (e.g., "Netflix")
3. User enters monthly cost in rupees (e.g., "499")
4. User selects billing cycle from dropdown (e.g., "Monthly")
5. User selects next renewal date from date picker
6. User selects category from dropdown (e.g., "Entertainment")
7. User clicks "Add Subscription" button
8. System validates all required fields
9. System saves subscription to localStorage
10. System updates dashboard metrics
11. System adds new card to subscription list
12. User sees confirmation of successful addition

Delete Subscription Workflow

1. User locates subscription card in list
2. User clicks "Delete" button on subscription card
3. System removes subscription from localStorage
4. System updates dashboard metrics
5. System removes card from subscription list
6. User sees updated list without deleted subscription

Filter Subscriptions Workflow

1. User clicks category filter dropdown
2. User selects desired category (e.g., "Entertainment")
3. System filters subscription list to show only matching category
4. System updates dashboard to reflect filtered metrics
5. User views filtered results
6. User can select "All Categories" to reset filter

Edge Cases and Error Handling

Input Validation

- Empty required fields: Display browser validation message
- Negative cost values: Prevent submission with validation
- Invalid date selection: Browser date picker prevents invalid dates
- Special characters in service name: Allow all characters for flexibility

Data Scenarios

- First-time user (empty state): Display empty subscription list with invitation to add first subscription
- Browser with localStorage disabled: Application will not function; consider graceful degradation message
- localStorage quota exceeded: Alert user to delete subscriptions or clear data
- Corrupted localStorage data: Clear and reinitialize with empty array

Browser Compatibility

- Older browsers without localStorage: Display compatibility warning
- Browsers with JavaScript disabled: Display message requiring JavaScript

Future Enhancements (Out of Scope for v1.0)

Phase 2 Features

1. **Renewal Notifications:** Browser push notifications for upcoming renewals
2. **Spending Charts:** Visual graphs showing spending trends over time
3. **Budget Alerts:** Set spending limits and receive warnings
4. **Custom Categories:** Allow users to create custom category tags
5. **Notes Field:** Add optional notes/comments to each subscription

Phase 3 Features

1. **Cloud Sync:** Backend integration for cross-device synchronization
2. **User Accounts:** Authentication and multi-device support
3. **Expense History:** Track historical payments and renewal records
4. **Price Comparison:** Suggest cheaper alternatives for services
5. **Shared Subscriptions:** Track family/group subscription sharing
6. **Export Functionality:** Export data to CSV or PDF

Technical Debt Considerations

- Migrate from localStorage to IndexedDB for better performance and capacity
- Implement proper state management for complex interactions
- Add unit tests for core calculation functions
- Implement proper form validation library
- Add service worker for offline functionality

Security and Privacy

Data Privacy

- All data stored locally on user's device
- No data transmission to external servers
- No user tracking or analytics in v1.0
- No personally identifiable information (PII) collected

Security Considerations

- Client-side only implementation reduces attack surface
- No sensitive data (passwords, credit card info) stored
- localStorage data accessible to user via browser dev tools
- XSS protection through proper input sanitization

Deployment and Release

Deployment Method

- Static HTML file deployment
- Can be hosted on any web server or file hosting service
- No build process or compilation required
- Single-file application for easy distribution

Release Criteria

Pre-launch checklist:

1. All P0 features implemented and tested
2. Cross-browser testing completed (Chrome, Firefox, Safari, Edge)
3. Mobile responsiveness verified on iOS and Android
4. Accessibility audit passed (minimum WCAG 2.1 AA)
5. User acceptance testing completed with 5+ test users
6. Documentation prepared (user guide, README)

Post-Launch Monitoring

- User feedback collection through surveys or feedback form
- Bug tracking and prioritization system
- Usage pattern analysis (if analytics added)
- Feature request collection and evaluation

Dependencies and Constraints

Technical Dependencies

- Modern web browser with localStorage support
- JavaScript enabled in browser
- No external API dependencies
- No backend services required

Constraints

- Single-device limitation (no cross-device sync)
- localStorage capacity limits (5-10MB typical)
- No real-time notifications (requires browser permissions)
- India-focused (INR currency only in v1.0)

Testing Strategy

Test Coverage

Test Type	Coverage Areas
Functional Testing	All CRUD operations, calculations, filtering
UI Testing	Layout responsiveness, visual design, interactions
Browser Testing	Chrome, Firefox, Safari, Edge compatibility
Accessibility Testing	Keyboard navigation, screen reader support
Performance Testing	Load times, localStorage operations, rendering

Table 4: Testing coverage areas

Test Scenarios

Critical Path Testing:

1. Add subscription → Verify appears in list and dashboard updates
2. Delete subscription → Verify removed from list and dashboard updates
3. Filter by category → Verify correct subscriptions shown
4. Refresh page → Verify data persists from localStorage
5. Add multiple subscriptions → Verify calculations correct

Boundary Testing:

- Add subscription with maximum character lengths
- Add 100+ subscriptions to test performance

- Test with very large cost values (e.g., ₹999,999)
- Test with dates far in future/past

Glossary

- **Billing Cycle:** The frequency at which a subscription charges the user (monthly, quarterly, yearly)
- **localStorage:** Browser API for storing data persistently on user's device
- **Renewal Date:** The date when a subscription will automatically renew and charge
- **Monthly Equivalent:** Normalized monthly cost calculated from different billing cycles (quarterly cost ÷ 3, yearly cost ÷ 12)
- **CRUD:** Create, Read, Update, Delete - basic data operations
- **Responsive Design:** UI approach that adapts layout to different screen sizes

Approval and Sign-off

Role	Name	Approval Date
Product Manager	[Pending]	[Date]
Engineering Lead	[Pending]	[Date]
Design Lead	[Pending]	[Date]
QA Lead	[Pending]	[Date]

Table 5: Document approval tracking

Revision History

Version	Date	Author	Changes
1.0	2025-11-18	Product Team	Initial PRD creation

Table 6: Document revision history