# Functional Specification for Checklist Management Application

## 1. Overview

This document outlines the functional specification for the Checklist Management Application. The application will allow users to create and manage checklist templates and instances for repetitive tasks, such as onboarding new clients, preparing for events, or managing projects. It will be built using FastHTML framework.

## 2. Functional Requirements

### 2.1 Template Management

1. Users can create checklist templates with the following attributes:  
 - Title  
 - Description (short and long)  
 - Steps with descriptions, reordering via drag-and-drop, and reference materials (links or small images).  
2. Templates are categorized using tags (category, sub-category, sub-sub-category).  
3. Templates can be copied to create new templates.  
4. Templates can be imported via Excel/Google Sheets uploads.  
 - The app will provide a template structure for uploads.  
 - It will intelligently process uploads that deviate slightly from the prescribed format.  
5. Version control will be available for templates, allowing users to track changes and revert to previous versions.

### 2.2 Checklist Instance Management

1. Users can create checklist instances from templates.  
2. Checklist instances cannot be edited after creation, but step statuses and notes can be updated.  
3. Checklist statuses include:  
 - 'Active'  
 - 'Completed'  
 - 'Not Started'  
4. Step statuses include:  
 - 'Not Started'  
 - 'In Progress'  
 - 'Completed'  
 - Status values are configurable globally and per user, but only the template creator can modify them.

### 2.3 User Authentication and Permissions

1. Users can authenticate via:  
 - Native email/password login.  
 - Social login (Google, Facebook, Microsoft).  
2. Templates and instances can be shared with the following permissions:  
 - Public (view-only).  
 - Private (user-only).  
 - Shared with specific users (view or edit permissions).  
3. Editors have the same permissions as creators, except they cannot modify status values.

### 2.4 Analytics and Reporting

1. Analytics will be provided for templates, checklist instances, and steps.  
2. Metrics include:  
 - Creation date, status change timestamps, step durations.  
 - Usage statistics and user engagement.  
 - Tag-based analytics.  
3. Visualizations include charts, graphs, and dashboards.  
4. Generated insights will provide automated analysis of trends and metrics.

### 2.5 Public Template Library

1. Users can share templates to a public library.  
2. Public templates are searchable and categorized by tags.  
3. An AI-based mechanism will flag inappropriate content.

### 2.6 Media and Attachments

1. V1 will support:  
 - Links.  
 - Small images (size limit to be determined, suggested 1024x1024px, 2MB).

### 2.7 Mobile and Localization

1. The application will be mobile-friendly out of the box.  
2. Future development may include a native mobile app.  
3. Initial deployment will support English only, with plans for multi-language support in future versions.

## 3. Technical Stack

1. Frontend:  
 - FastHTML  
2. Backend:  
 - FastHTML  
3. Hosting:  
 - Heroku.  
4. Authentication:  
 - Native Passwordless, and social logins (Google, Facebook, Microsoft).

Here's the revised terminology, removing Template and updating Instance:

**Core Concepts:**

* **Checklist**: A reusable set of steps that serves as a blueprint
* **Checklist\_Instance**: A specific usage of a checklist for tracking actual progress
* **Step**: An individual item within a checklist or checklist\_instance that needs to be completed

**Status Terms:**

* **Active**: A checklist\_instance that is currently in use
* **Not Started**: Initial state for checklist\_instances or steps
* **In Progress**: Work has begun on a step
* **Completed**: All work is finished on a checklist\_instance or step

**User Roles:**

* **Creator**: User who creates a checklist
* **Editor**: User with permission to modify checklist\_instances but not status values
* **Viewer**: User with read-only access

**Technical Terms:**

* **FastHTML**: The web framework we're using
* **MonsterUI**: UI component library for building the interface
* **SQLite**: Our initial database choice
* **HTMX**: Tool for dynamic UI updates

**Data Structure Terms:**

* **Tag**: Category identifier for checklists (category, sub-category, sub-sub-category)
* **Reference Material**: Links or images attached to steps
* **Version**: A saved state of a checklist

Instance user workflow and design:

Here's a comprehensive summary of the checklist instance design and workflow:

1. User Navigation Flow:

- Main Page

\* Tabs: "My Checklists" and "Active Instances"

\* Checklists show instance count badges

\* Active Instances section with filters (status/date/checklist)

- Checklist View

\* Tabs: "Details" and "Instances"

\* "New" button for creating instances

\* List of related instances with status and progress

2. Instance Creation:

- Modal Form Fields:

\* Instance name (required)

\* Description (optional)

\* Start date (default: today)

\* Target completion date (optional)

\* Initial status (default: "Not Started")

3. Instance List View:

- Per-instance displays:

\* Name/identifier

\* Status (color-coded)

\* Creation & last updated dates

\* Progress indicator

\* Quick actions (View/Continue)

- Filtering options:

\* By status (Active/Completed/Not Started)

\* By date

\* By name

\* Sort options available

4. Single Instance View:

- Header

\* Instance name

\* Parent checklist link

\* Dates (creation/target)

\* Progress bar

\* Status selector

\* Complete button

- Steps Display

\* Original step text (non-editable)

\* Status selector per step

\* Expandable notes

\* Original reference materials

\* Update timestamps

\* Ordered as per original

\* Active step highlighting

- Navigation

\* Back to checklist

\* Next/Previous instance

\* Jump to incomplete steps

Would you like to proceed with the database schema design to support this workflow?