

Contents

1	Analysis	2
1.1	Problem Identification	2
1.1.1	Problem Description	2
1.1.2	Interview	2
1.1.3	Existing similar solutions	2
1.1.4	Features to be incorporated into solution	5
1.1.5	Feedback from stakeholders	5
1.2	Requirements	5
1.2.1	Stakeholder requirements	5
1.2.2	Software and hardware requirements	5
1.2.3	Success requirements	5
2	Design	5
2.1	User Interface Design	5
2.1.1	Usability Features	5
2.1.2	Feedback from stakeholder	5
2.2	Modular breakdown	5
2.3	Algorithms	5
2.4	Data Dictionary	5
2.5	Inputs and outputs	5
2.6	Validation	5
2.7	Testing	5
2.7.1	Methods	5
2.7.2	Test Plan	5
3	Implementation	5
3.1	First Iteration	5
4	Testing	5
5	Evaluation	5

1 Analysis

1.1 Problem Identification

1.1.1 Problem Description

Popular inventory management solutions are relatively expensive, and may be out of reach for individuals or small schools. Inventory systems have numerous benefits for businesses and individuals alike; a business may choose to track their supply levels where an individual may wish to catalogue their DVD collection.

My goal is to create a web-based application aimed at both businesses and individuals to manage inventory, with additional modern features such as automatic item re-ordering when stocks are running low.

Traditional inventory management solutions are typically single-user at best, whereas I am to create a multi-user, collaborative environment.

An inventory system should be able to:

time consuming to add data not user friendly

- be easy to use (intuitive) - catalogue of inventory, re-order for you - be cross-platform, Fast - scan using a phone (no external hardware needed) - alert / re-order when stocks are running low. - purchase links - stretch: source data from amazon or equivalent instead of typing it manually - search engine for catalogued and new Parts - provides with options for where to purchase certain goods - button to re-order
- smart device?????? - predict when stocks will run out. - support for consumable and non-consumable goods. - source data from external sources - like monzo projection of when it will run out - how much you are spending each month on goods - nfc support to easily scan / etc items (might be too hard on iOS)

Barcode check in / out

- monzo integration

- budgeting - figure projections as well

clearly define what the APP will feature.

Think about

- potential users - how does the app cater to their needs - different features etc

1.1.2 Interview

1.1.3 Existing similar solutions

InvenTree <https://inventree.org/>

Overview

InvenTree is an **open-source** inventory management system, providing *low level stock control and part tracking*. It uses a Python/Django database backend and provides both a **web-based interface** as well as a REST API for interacting with other services. InvenTree also has a powerful plugin system for custom applications and other extensions.

Below is a screenshot of the InvenTree homepage.

Build	Description	Project Code	Priority	Part	Progress	Status	Created	Issued by	Responsible	Target
BO0016	Making widgets for SO 0003	-	0	Widget Assembly Variant	0 / 75	Pending	2022-05-25	admin	-	-
BO0014	Making tables for SO 0003	-	0	Blue Square Table	0 / 100	Pending	2022-05-25	admin	-	-
BO0013	Required parts for Build 0010	-	0	TB3 Test Board 3	0 / 50	Pending	2022-05-25	admin	-	-
BO0011	Required parts for Build 0010	-	0	TB1 Test Board 1	25 / 50	Production	2022-05-25	admin	-	-
BO0010	Making a high level assembly part	-	0	MAST Master Assembly	0 / 50	Pending	2022-05-25	admin	-	-
BO0007	Making red square tables	-	0	Red Square Table	0 / 15	Production	2022-04-29	allaccess	-	-

Parts applicable to my solution

- concept is similar (web-based), but I'm doing a different approach.
- not indented for stock control

PartKeeper <https://partkeeper.org/>

Name	Description	Storage Location	Status	Condition	Stock	Min. Stock	Avg. Price	Footprint	Internal ID
100R 0805 1K5	0805WRF1501T5E	REPLIMAT			50 pcs	0 pcs	0.00E		1826 (#1eq)
100R 1/8W THT 5%		REPLIMAT			100 pcs	0 pcs	0.00E		1827 (#1er)
BAT5		REPLIMAT			12 pcs	0 pcs	0.00E		1839 (#1f3)
KW10 Microswitch		REPLIMAT			0 pcs	0 pcs	0.00E		1830 (#1eu)
Pin Header 2x4p 90°		REPLIMAT			0 pcs	0 pcs	0.00E		1849 (#1fd)
Socket Header 2x4p		REPLIMAT			0 pcs	0 pcs	0.00E		1848 (#1fk)
JST PH Crimp Contact		REPLIMAT			1270 pcs	0 pcs	0.00E		1847 (#1tb)
JST PH Housing 2p		REPLIMAT			160 pcs	0 pcs	0.00E		1844 (#18)
JST PH Housing 3p		REPLIMAT			140 pcs	0 pcs	0.00E		1845 (#19)
JST PH Housing 4p		REPLIMAT			370 pcs	0 pcs	0.00E		1846 (#1ta)
JST PH THT 2p 90°		REPLIMAT			80 pcs	0 pcs	0.00E		1840 (#1ta)
JST PH THT 3p		REPLIMAT			57 pcs	0 pcs	0.00E		1843 (#177)
JST PH THT 3p 90°		REPLIMAT			40 pcs	0 pcs	0.00E		1842 (#16)
JST PH THT 4p		REPLIMAT			49 pcs	0 pcs	0.00E		1853 (#1th)
XH Crimp Contacts		REPLIMAT			223 pcs	0 pcs	0.00E		1831 (#1ev)
XH Housing 2p		REPLIMAT			67 pcs	0 pcs	0.00E		1816 (#1eg)
XH Housing 3p		REPLIMAT			57 pcs	0 pcs	0.00E		1817 (#1eh)
XH Housing 4p		REPLIMAT			52 pcs	0 pcs	0.00E		1818 (#1ei)
XH THT 2p		REPLIMAT			50 pcs	0 pcs	0.00E		1815 (#1ef)
XH THT 2p 90°		REPLIMAT			75 pcs	0 pcs	0.00E		1811 (#1eb)
XH THT 3p		REPLIMAT			56 pcs	0 pcs	0.00E		1813 (#1ed)
XH THT 4p		REPLIMAT			21 pcs	0 pcs	0.00E		1814 (#1ee)
XH THT 4p 90°		REPLIMAT			31 pcs	0 pcs	0.00E		1812 (#1ec)

Overview

PartKeeper is an open-source inventory management system with a focus on electronic components. It is designed around four main principles:

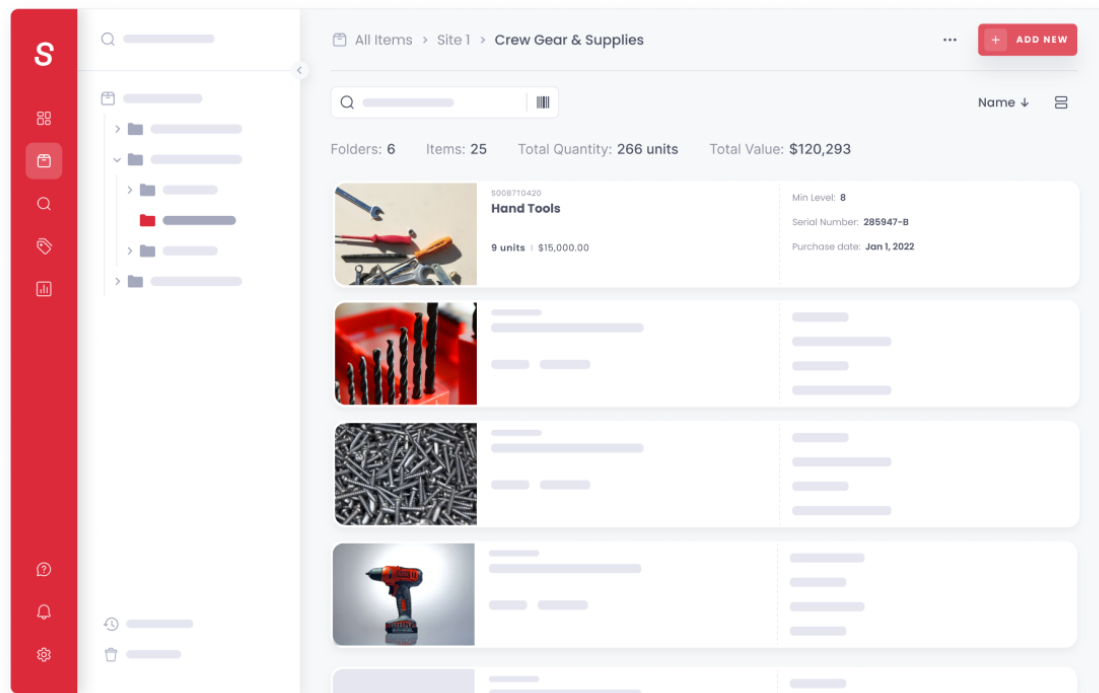
- Fast Part Searching
- Ability to add complete part database
- Keeping track of stock

- Ease of use

Parts applicable to my solution

Like PartKeepr, I hope to implement a web-based interface. However, I am using a different approach as my solution will not be tailored specifically to electronic components.

Sortly <https://www.sortly.com/solutions/inventory-management-software/>



Overview

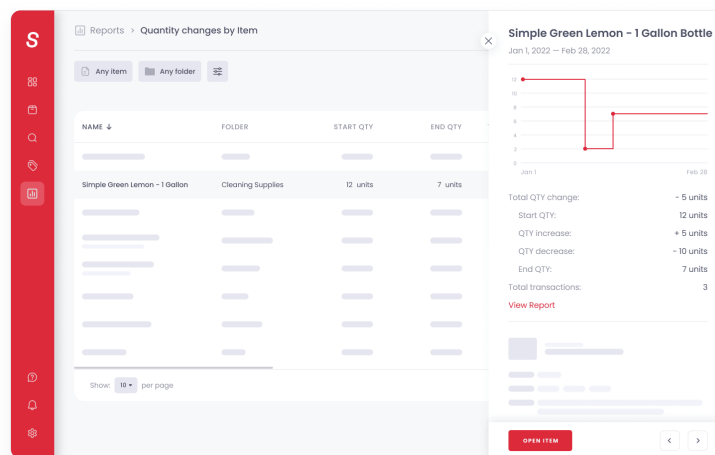
Sortly is a proprietary cloud-based inventory management system with a focus on small businesses and individuals.

It has two plans available, an always free plan with limited functionality and a paid plan with a more complete feature-set.

Parts applicable to my solution

I hope to implement the following features from Sortly:

- Web based interface
Allows for easy access.
- Ability to create QR codes to stick on items/containers
Allows for easily unit selection in the interface.
- Real-time reporting insights



Allows for added insight into usage patterns for particular units.

1.1.4 Features to be incorporated into solution

1.1.5 Feedback from stakeholders

1.2 Requirements

1.2.1 Stakeholder requirements

1.2.2 Software and hardware requirements

1.2.3 Success requirements

2 Design

2.1 User Interface Design

2.1.1 Usability Features

2.1.2 Feedback from stakeholder

2.2 Modular breakdown

2.3 Algorithms

2.4 Data Dictionary

2.5 Inputs and outputs

2.6 Validation

2.7 Testing

2.7.1 Methods

2.7.2 Test Plan

3 Implementation

3.1 First Iteration

4 Testing

5 Evaluation