Contents

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 catelogue 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 (migght 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

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

1.1.2 Interview

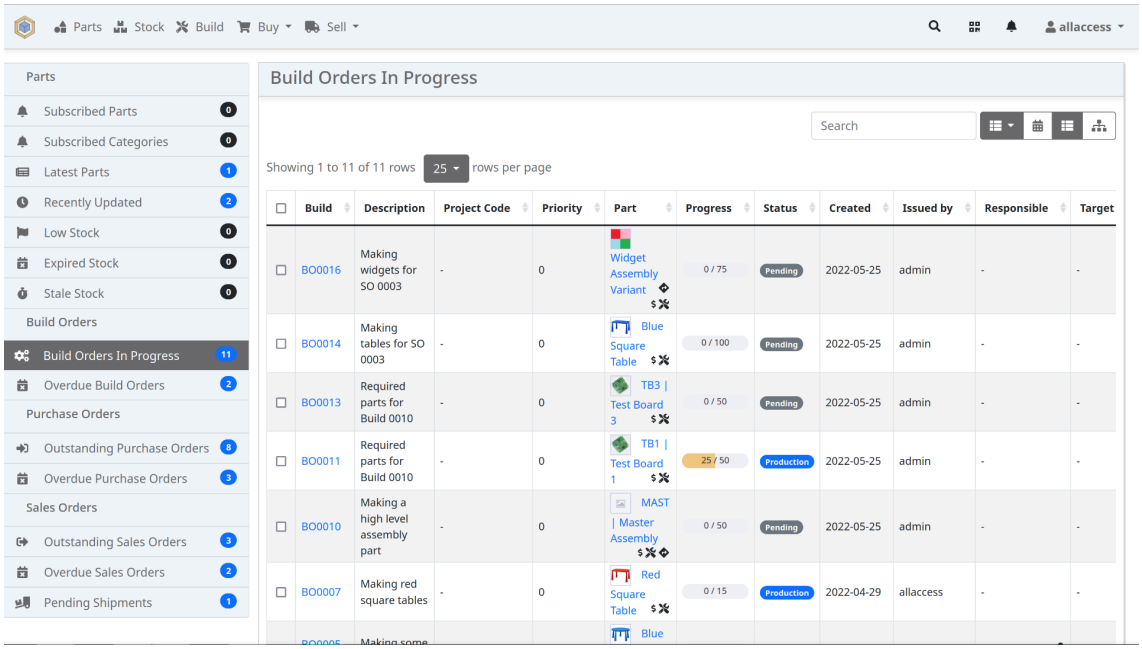
1.1.3 Existing similar solutions

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.

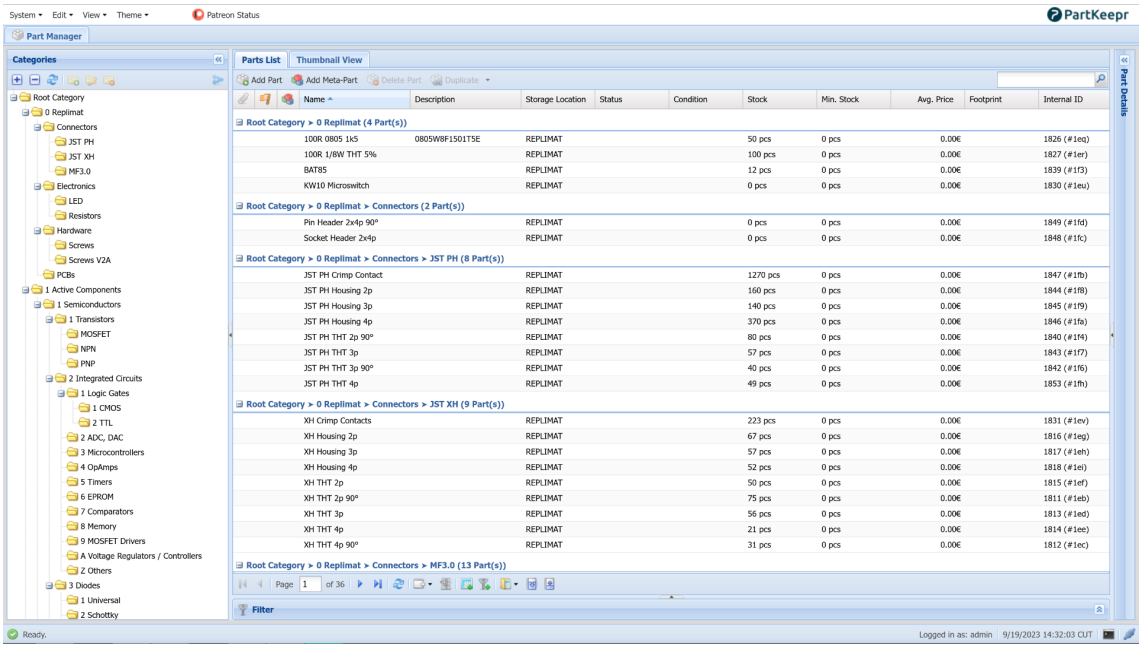
Below is a screenshot of the InvenTree homepage.



Parts applicable to my solution

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

PartKeepr `https://partkeepr.org/`



Overview

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

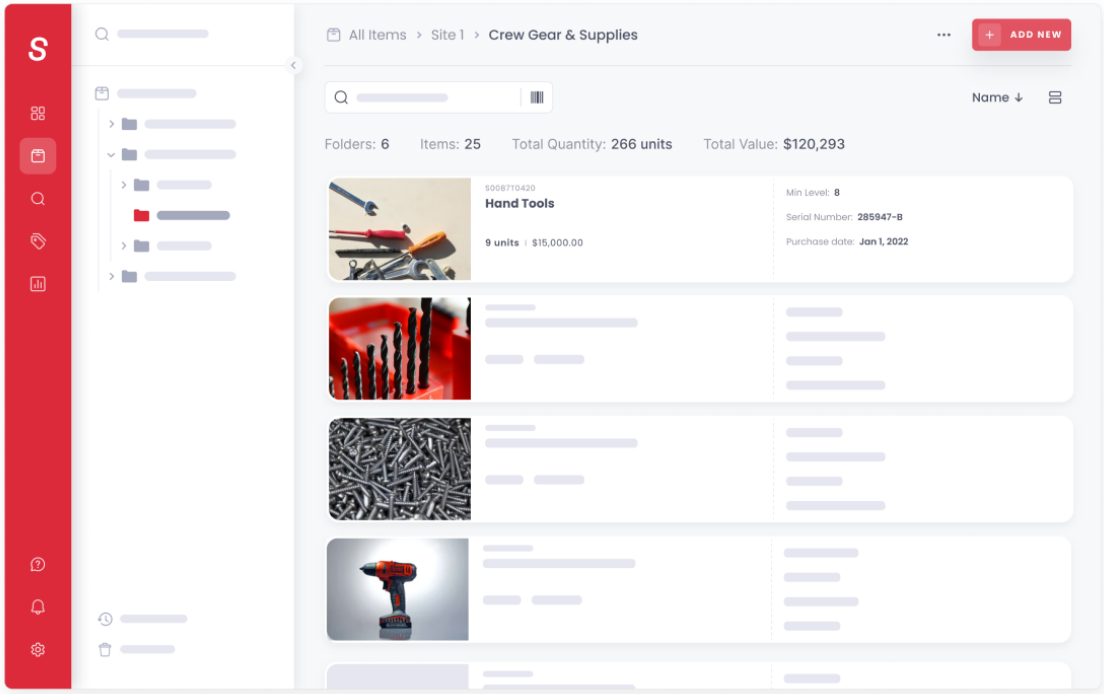
- Fast Part Searching
- Ability to add complete part database
- Keeping track of stock

- 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 inviduals.

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

Parts applicable to my solution

I hope to implement the following features from Sortly:

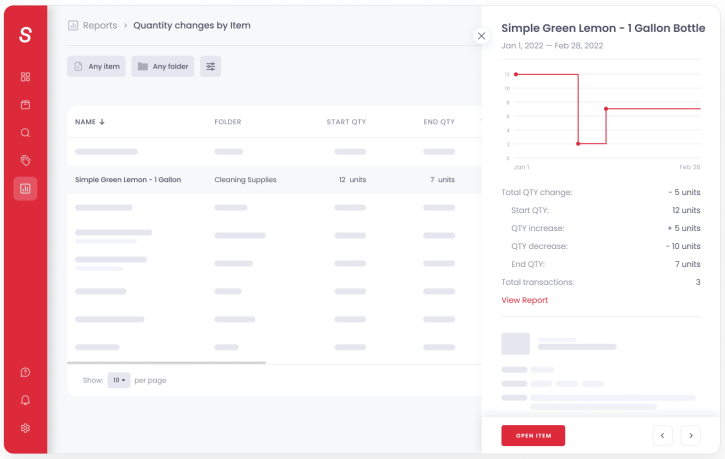
- Web based interface

  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