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 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 intend to create a multi-user, collaborative environment.

In my view, an inventory system should be:

- Easy for end users to use.
- Cross platform
- Performant interface
- Efficient in terms of adding data
- Allow for easy cataloguing of inventory
- Allow for item scanning using QR codes / barcodes
- Be able to source data from external sources
- Support both consumable and non-consumable goods.

1.1.2 Stakeholders

Stakeholder	Description	Current Use	Requirements
Claire Foley	Senior Leadership Team at The Village Prep School	Library books; currently tracked on paper	TBD
TBD	Freelance photographer	Excel Spreadsheet	TBD

1.1.3 Interview

Question Set

- What would you consider your skill level to be regarding technology?
- Do you currently have a way to manage inventory?
  - If so, what is your current solution?
    * What aspects of this solution do you like?
    * What aspects of this solution do you dislike?
  - What features would you **require** in a custom solution?
  - What features would **enchance** your experience?

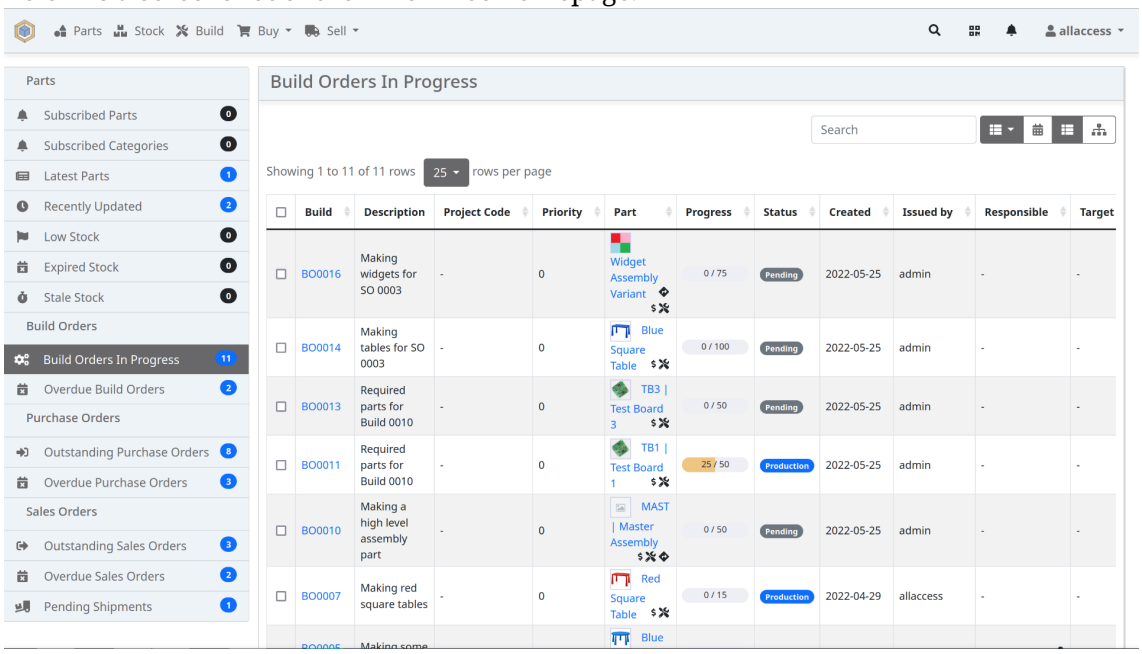
1.1.4 Existing similar solutions

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.

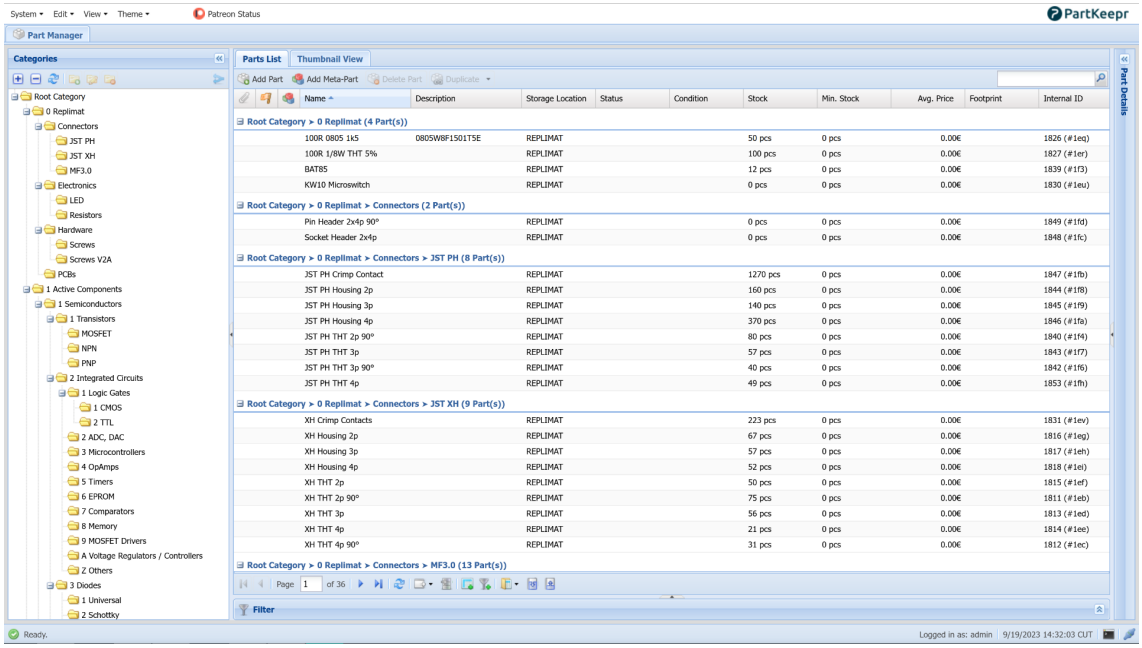
Below is a screenshot of the InvenTree homepage.



Parts applicable to my solution

The core concept is similar (the application is web-based), but my solution will be more generalized that just stock control.

PartKeepr `https://partkeepr.org/`



Overview

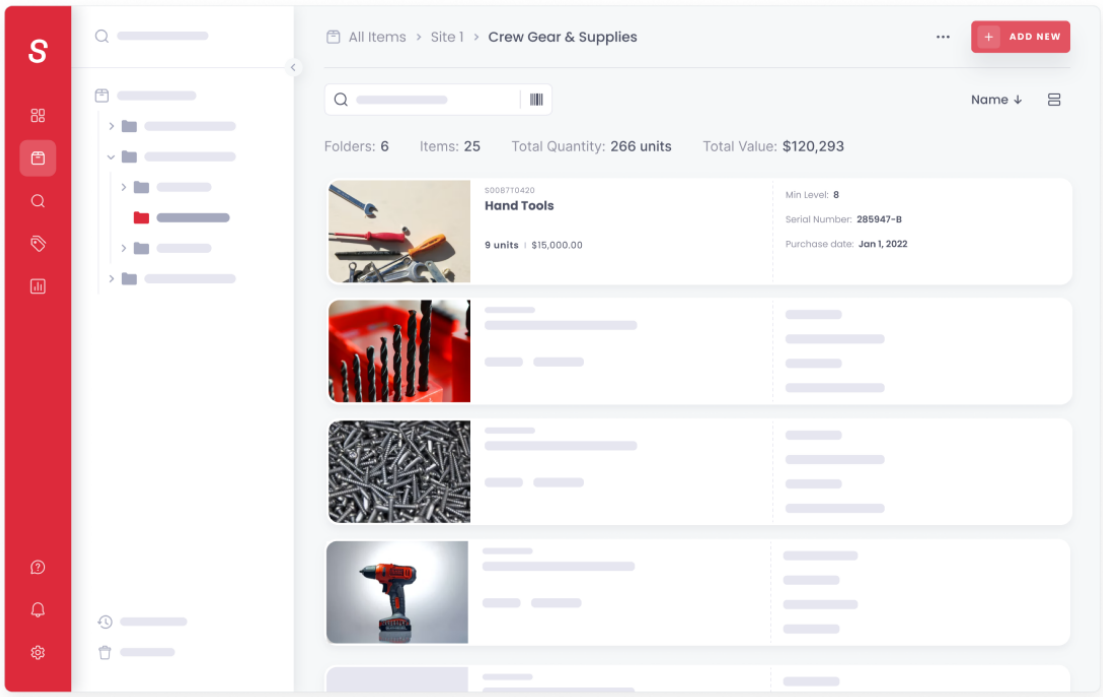
PartKeepr is an open-source inventory management system with a focus on electronic components. It is designed around four main principles:

- 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 individuals.

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.
- Barcode support
  - Allows end users to print off QR codes to stick to items
  - Which can be scanned in-app to easily perform actions on the item.
- Real-time reporting insights
  - Allows for added insight into usage patterns for particular units.

1.1.5 Features to be incorporated into solution

1.1.6 Feedback from stakeholders

1.2 Requirements

1.2.1 Stakeholder requirements

1.2.2 Software and hardware requirements

System Requirements

Hardware	Justification
Input Device *(Touchscreen or Keyboard+Mouse)*	An input device will help users navigate through the program as it will have a graphical user interface (GUI). This will allow the user to interact with buttons, menus and icons.

Software Requirements

Software	Justification
Operating System *(Windows, MacOS, Linux, ChromeOS, iOS, iPadOS, Android)*	An operating system is required to... TBD
A web browser *(eg. Chrome, Firefox, Microsoft Edge, Safari)*	TBD

1.2.3 Success requirements

The overall objectives for the system.
To measure the overall effectiveness of the system, targets must be set before writing the program. These targets will help in the evaluation stage to determine weather our objectives have been met. These objectives will be **SMART**, i.e:

- **Specific**
  What objective needs to be accomplished?
- **Measurable**
  How can we quantify this objective?
  How will the success of this objective be measured? (quantitatively or qualitatively)
- **Achievable**
  Is this objective achievable and realistic? If so, how to you plan to achieve them?
- **Relevant**
  How does this objective benefit the end-users of this application as a whole?
  Why has this goal been set?
- **Timely**
  Can this objective be completed within an appropriate time frame?
  At what stage in the software development lifecycle will you start implementing this goal?
  In which order will any sub-objectives be completed?

The Project's SMART Objectives

1. **To produce a solution for cataloguing a school library and recording users and books borrowed**
   At the end of the project, I will evaluate against my success criteria and determine weather this objective has been met. On the software side, I will be using React, Expo and PostgreSQL. This objective will be the main objective for this project. This objective must be completed by March 2024.
2. **To produce a solution including a database that can store details of books, borrowers, loans and returns**
3. **To produce an intuitive and easy to use solution**
   I will evaluate my success on this objective by having a new user without any prior training or advice use the system and try to carry out a number of tasks without any assistance. If the user is able to successfully complete the tasks I will consider the system to be intuitive and easy to use and therefore this objective satisfied. To achieve this I will design my system to have a consistent layout based on **Material Design 2**, (`https://m2.material.io/`) the design language used by Google products and many apps running on the Android operating system. I will also use language that is a) appropriate for the situation the product will be deployed in (with young children) and b) easy to understand (so that children can interact with the system) I will also use meaningful error messages so that the user has a clear understanding of the problem that has occurred.
4. **To produce a solution that features a fully searchable catalogue**
5. **To produce a solution that features reporting for overdue and/or lost books**
6. **To produce a solution that includes a curated "suggested reading list" for each borrower**

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