Paperless QR Code Based Picking of Critical Part with Automated Exception Workflow and Traceability

Dr.N. Baggyalakshmi[1*], K. Jeswanthika[2], Dr.R. Revathi[3]

[1*] Assistant Professor, Department of Computer Science, PSGR Krishnammal College for Women, Coimbatore.
E-mail: baggyanethra@gmail.com
[2] Student, B.sc Computer Science, PSGR Krishnammal College for Women, Coimbatore. E-mail: jeswan24@gmail.com
[3] Assistant Professor, Department of Computer Science, Karpagam Academy of Higher Education, Coimbatore.
E-mail: ravathilakshay@gmail.com

***Abstract*---** A mobile application, commonly referred to as an app, is a type of application software designed to run on a mobile device. It allows users to handle information instantly and in real-time, serving as an excellent communication channel between organizations and their users or clients. The app is a tool that enables businesses to manage their maintenance activities by analyzing and monitoring all tasks updated on a daily, weekly, and monthly basis. Maintenance data and a list view of the required CSV file are present, and the necessary check boxes are checked, displaying the filtered items as a list view. This list can be loaded and sent to the next screen using the "Load list" button. Stakeholders are individuals who either care about or have a vested interest in your project. They are the people actively involved in the project's work. The list of screens is displayed, and an item from the list must be selected to proceed. By using the "PICK" button, the barcode scanner is opened to select the item to pick. If the item is matched, then clicking the "Confirm Picking" button completes the process or "Wrong part picked" will appear. If the item is not available, a message will appear stating "Out of stock."
***Keyword*---** Mobile Application, QR Code, Stack Holder, Business.

I. INTRODUCTION

Picking critical parts using a paperless QR code-based system involves an automated exception workflow and traceability to ensure the QR codes are maintained according to the provided list, with updates on a daily, weekly, and monthly basis [1]. The picking process is executed by selecting items from the list. The application comprises multiple modules or screens containing checklists with lists of items. We utilize a respective barcode scannerto generate the QR codes and display the list in a list view. When the picking is confirmed,the required field's result automatically enters the table and is sent to a designated email address for review [2]. In case of any abnormalities, both the reviewer and the client are notified, and appropriate actions are taken to address them. The app is designed to auto-filldata in the table and ensure all the listed items meet the client's requirements, displaying the output accordingly. The primary objective of this mobile app is to assist manufacturingunits in accurately assembling machine parts.

A mobile application, most commonly referred to as an app, is a type of application software designed to run on a mobile device [3]. It allows users to handle information instantly and in real-time, serving as an excellent communication channel between organizations and their users or clients [4]. A paperless QR code-based picking of critical parts with automated exception workflow and traceability is a tool that enables businesses to manage their maintenance activities by analyzing, monitoring, and documenting all the tasks involved in sustaining an organization. Maintenance data and a list view of selection files are tools used by technicians to document equipment maintenance inspections. Equipment maintenance involves the continuous process of checking and servicing operating equipment to ensure businesses can operate without interruption. The mobile application maintains the list, updating daily, weekly, and monthly selection files from CSV with access to the SAF root directory [5]. Once the pickingis confirmed, the required details are automatically entered into the table. The results of thechecks are sent to a common email id after the process completion, while any abnormalitiesare communicated to stakeholders for action, as well as to specific clients according to theirrequirements.

The existing system relies on a manual process that involves using a QR scanner to update maintenance details on paper sheets. However, this system has some drawbacks. After a long day of work, managing the physical paper sheets can become cumbersome, and retrieval of information may not be as efficient as needed. However, as the organization and maintenance tasks grow, relying solely on a manual system might become more challenging. Handling large volumes of paper sheets could lead to misplaced or lost

information, which may result in inefficiencies and errors. Furthermore, manual data entry is prone to human errors, which could impact the accuracy and reliability of the recorded data. Disadvantages of Existing System

- Increase time consumption.
- Prone to Human Errors.
- Space and Storage Requirements.
- Increased Risk of Data Loss.

II. LITERATURE REVIEW

Larsen et al [6] The Connecticut Department of Transportation's first annual administrative report on pavements for the calendar year 2019 is shown below. This report offers information on Connecticut's pavement conditions—both present and from the previous few years—to executive level management and other parties. The funding sources, paving initiatives, and anticipated future activity in Connecticut as a result of utilizing the CTDOT Pavement Management program are all included in this description. The current state of the pavements for two roadway systems—the state- and town-maintained National Highway System (NHS) designated roadways in Connecticut and the complete CTDOT-maintained roadway network—is summarized in this yearly pavement report.

Gregory et al [7] provide an Intelligent Process Automation (IPA) software solution that uses label information from basic warehouse photos to locate, recognize, and manage inventory storage logistics. It does this by utilizing state-of-the-art computer vision (CV) and other algorithmic techniques. Our pipeline may be demonstrated to replace the need for expensive, prone to error human input to the inventory tracking system when combined with a recently created robotic imaging device. This paper describes the deep learning-driven IPA's technical and practical applications. Although Mercedes-Benz U.S. International (MBUSI) had a specific need that needed to be addressed, the methods developed for this project could be used more broadly in other inventory management scenarios.

Sahi et al [8] In order to investigate: (a) the types of regulatory support people offered others; (b) how people felt receiving various forms of social feedback about their experiences; and (c) whether the support people offered others shaped how they felt receiving different feedback, the current study used two experiments in 2020 (N = 50, N = 268). Participants shared and responded to personal experiences. There was variation among participants when it came to the types of social reappraisals that help others change their thoughts about an emotional experience. These included temporal distancing, which emphasizes how things change over time, positive focus, which focuses on the positive, and perspective taking, which involves taking into account others' perspectives.

Luo et al [9] examined Stack Overflow (SO) and Reddit, two well-known online development communities, to offer insights on the features and difficulties of LCD from the viewpoint of practitioners. Method: We searched for pertinent topics in SO using two phrases relating to LCDs, and we were able to extract 73 entries. In the interim, we looked through three Reddit subreddits pertaining to LCDs and gathered 228 entries. We took information off of these postings and used the Constant Comparison technique to examine the features, advantages, drawbacks, and restrictions of LCD. We employed descriptive statistics to assess and present the data for the platforms and programming languages used in LCD, the implementation units in LCD, the supporting technologies of LCD, the kinds of applications generated by LCD, and the domains that use LCD.

Kim et al [10] Process frameworks that are used to implement cloud ERP are categorized according to the type of construction into four categories: software-as-a-service (SaaS), platform-as-a-service (PaaS), content-as-a-service (CaaS), and infrastructure-as-a-service (IaaS). These frameworks are defined using six derived processes, twenty-one activities, and many specific tasks. The final proposed process framework's process engineering features were further dissected and compared to on-premise ERP construction procedures to identify the differences and similarities. The theoretical underpinnings of standardized research on cloud ERP construction techniques are provided by this work. It can serve as a useful manual for stakeholders and a tool for process customization, giving details on certain duties and activities for every stage of development. From the standpoint of the client, it also helps build and disseminate dependable cloud-based ERP systems.

III. PROPOSED METHODOLOGY

The proposed system aims to introduce a paperless QR code-based picking process for critical parts, incorporating automated exception workflows and traceability. This modernapproach offers several advantages over the existing manual system. The paperless natureof the system eliminates the need for physical paper sheets, reducing the risk of misplacement or loss of important maintenance information. By utilizing QR codes for picking critical parts, employees can easily update maintenance details in a digital format, ensuring data accuracy and accessibility. Advantages of Proposed System

- Reduces time consumption.
- Reducing the need for manual data entry.
- Employees can easily update details in a digital format.
- Reducing the risk of misplacement

3.1. Application Specification

3.1.1. MIT App Inventor

Formerly offered by Google, the Massachusetts Institute of Technology (MIT) currently maintains MIT App Inventor, an IDE for building web applications. With App Inventor, you can create Android apps on a web browser with a real phone or an emulator that you can use on your screen. You can save your progress and access all of your project

information on the MIT App Inventor servers.

Developers can build Android apps with its help using a graphical user interface (GUI) that's reminiscent of Scratch and the Star Logo. You can build your app's components with the App Designer. You can build programme blocks that define the components' behaviour in the App Inventor Blocks Editor. Put programmes together visually, putting them together like a puzzle. You can test your work as you construct because the software appears on the phone step-by-step as you add pieces. You can create apps for Android devices even if you don't own one by utilising an emulator, which is software that mimics the phone's behaviour on a computer.

Constructionist learning theories, which provide the groundwork for App Inventor and the other projects, stress the importance of programming as a means of actively engaging with powerful concepts through learning. With App Inventor, the developers have made blocks for practically every function of an Android phone, in addition to "programming-like" functions like data storage, action repetition, and conditional execution. There are also blocks that can communicate with social media platforms. Through its Cloud DB component, App Inventor additionally facilitates the use of cloud-based data.

3.1.2. Packaging Applications

When it comes to live programming, App Inventor isn't involved. The build server is responsible for transforming an App Inventor project into an Android application package (APK) that is ready for distribution, including the App Store.

Users can download the finished software to their device by scanning a QR code in their browser or save it to their PC for later distribution after using the Build option in software Inventor.

3.1.3. Apps with Multiple Screens

You can have many screens in an App Inventor app, and they're all viewable on the device at once. Designer new screens are announced. The open another screen block puts the active screen on a stack of suspended screens, suspends it, and then makes a new instance of the operand screen active.

By removing the active screen and selecting the top screen from the stack of suspended screens, the close screen block makes that screen active. There are variants of the screen opening and closing blocks that transfer values between screens to enable interscreen communication. There are two methods in which many screens engage with real-time development.

To start, the Companion and the browser are constantly in sync when you're editing one. When the user navigates to a different screen in the browser, the Companion receives the YAIL for that screen. Additionally, if the current screen in the Companion is changed (by means of the "open another screen" or "close screen" commands), the Companion notifies the browser through a particular message that the Blocks Editor should show the blocks corresponding to the new active screen.

For the Blocks Editor to be responsive, it must be able to sync with the Companion's active screen. When working in live development mode, the Companion's handling of multiple screens deviates slightly from how packaged programmes handle multiple screens. Suspended screens in packaged apps are instances of Forms that are still operating and can handle events like timer events that have nothing to do with the user interface.

3.1.4. Design App Inventor

Apps are built using components that contain capabilities of the Android software development kit. The components showcase a set of attributes, methods, and events. An app's components, including the layout of its UI elements, can be defined with the help of a drag-and-drop editor.

Parts resembling jigsaw puzzle pieces are linked to create programmes in a blocks programming language, which specifies the behaviours of components. Excruciating syntax mistakes are eliminated by blocks' forms. To help programmers remember the number and order of their operands and avoid mistakes like misspelt names and unbound variables, menu-based naming features are available. Another helpful feature is menu-based drawers of related blocks with labelled sockets.

You can easily incorporate Android device features like touch-based interfaces, location awareness, audio recording, language translation, social media, persistent data storage, cloud-based data storage, sprite-based gaming, and web service interactions into your apps with the help of blocks that give access to high-level behaviours. Ten blocks is all it takes to make a simple but engaging programme, and a few dozen blocks is all it takes to make a nontrivial useful app.

One characteristic of the App Inventor environment is the live development mode, which allows developers to work in real time with an app running on an Android device. This mode allows users to see their changes reflected in the app's UI and code blocks as they are made in the web browser. Additionally, you can examine the value—if any—produced by executing any block within the context of an already-running app. Because it does away with the time-consuming edit-build-install-test cycle that is normally involved with developing mobile apps, live development improves the experience of making, testing, and debugging apps.

Compiling apps to generate standard APK files is possible in the end, but the Companion, which acts as an interpreter for the App Inventor language, allows for browser interaction throughout live development. In addition to facilitating robust debugging tools that enable users to probe the app running on the mobile device, the Companion handles the Wi-Fi connection between browser programmes and external devices with the help of a rendezvous server.

The Designer is Divided into Several Areas

The Viewer is a white region towards the centre. Here you can lay out the various parts of your programme and

arrange them in the way you see fit. For instance, a line of text may end at a different location in your app than what you observe in the Viewer because the Viewer just provides a general idea of how the app will appear. You can preview your app's final look in App Inventor's built-in emulator or by downloading it to your phone ("Packaging the App for Downloading").

There is a list of components that you can select from in the Palette, which is to the left of the Viewer. You can access different parts of the Palette by clicking on their respective headings; for example, the Basic components are currently accessible. You can access the Media and Animation sections by doing the same.

The Components list, which displays all of your project's components, is located to the viewer's right. These items will be visible in the Viewer whenever you drag them into it. At the moment, the only item on the list is Screen1, which stands for the actual phone screen.

Clicking on a component in the Viewer will bring up its Properties, which are displayed in the area on the far right. You can edit the properties of each component to change their details. The screen's attributes, which include the title, background picture, and colour, are displayed at the moment (named Screen1).

Making a Label

Go to the Palette, click Label (which appears about down in the list of components), and drag it to the Viewer. You'll see a rectangular shape appear on the Viewer, with the words "Text for Label1."

Check out the Designer's Properties box over there on the right. It displays the label's attributes. Encircled by a box for the label's text, there is a property named Text around halfway down. Type "Already picked" in the text field and hit the Return key. The text will be seen in the Viewer after it changes.

Change the Background Color of the label by clicking the box, whichcurrently reads None, to select a color from the list that appears. Select Blue. Also changethe Text Color of the label to Yellow.

Packaging the App for Downloading:

As you work in App Inventor, your app is stored on Google's internet servers because it is a cloud computing tool. Unlike with Word documents or audio files, your programme will still be accessible when you return to programme Inventor; no need to save anything on your PC. Additionally, you may test the app in real-time while it's linked to your phone (a process known as live testing) and there's no need to download anything to your phone.

The only problem is that if you disconnect your phone from App Inventor, the app running on the phone will stop, and you won't find an icon for it anywhere becauseit was never truly installed.

With proper packaging and installation, you can make your finished software run on any phone, regardless of whether it's linked to a computer or not. You must first verify that your mobile device supports app stores other than the Android Market. In most cases, you may accomplish this by navigating to your phone's Settings → Applications and then ticking the box that says "Unknown sources."

Next, return to App Inventor's Designer, find "Package for Phone," and choose "Download to Connected Phone." It may take up to a minute for the messages "Saving" and "Packaging" to appear. Just a few more seconds after the "Packaging" message goes away, the app will finish downloading to your phone.

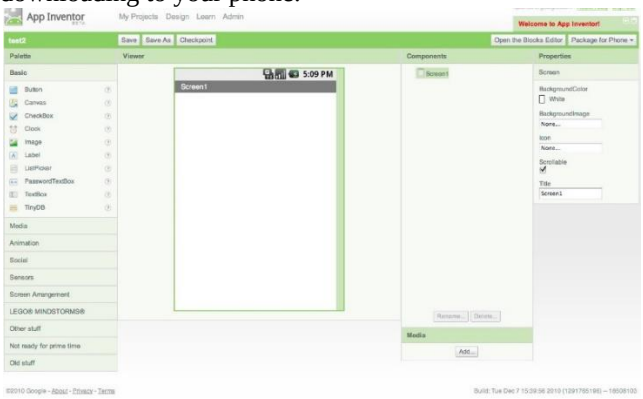

Figure 1 - App Inverter

IV. Results and Discussion

The primary input method for picking critical parts is through QR code scanning. The user interface should have a built-in QR code scanner that allows employeesto scan QR codes on equipment or parts to initiate the picking process. The proposed system can offer a user-friendly, efficient, and secure data entry process for maintenance.

Input Design

- Screen 1
- Screen 2
- Pick Screen

Screen 1

In this Screen 1 click a **"BROWSE "**button to fetch the data. Before initializing Screen 2 it directs us to internal storage and a particular CSV file is to be selected.

Figure 2 - Browse Button

Screen 2

After the required CSV file is selected it redirects us to screen 2 where the check box of LP2, LP3, DP2 and DP3 are present, and the required check boxes are checked so that it displays the filtered items as a list view and this list can be loaded and sent to the next screen with the **"Load list"**

button.

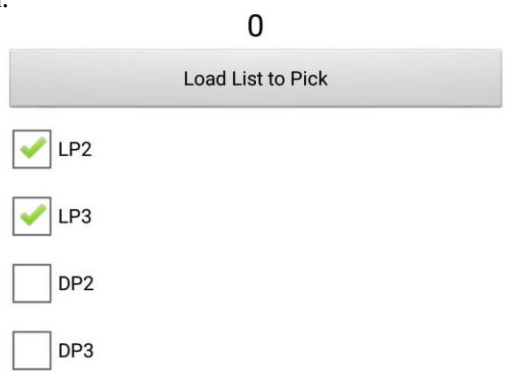

Figure 3 - Load List

Pick Screen

The list from the previous screen is displayed and an item of the list must be selected to proceed, and using the "PICK" button the barcode scanner is opened and the code is being scanned if the list item selected and the code scanned is matched then a button**"Confirm** Picking **"** appears ,when this button is clicked the table created in this same screen gets filled according to its respective field designated, if the list item does not matchwith the code scanner then a label appears as **"Wrong part picked"** is suppose list item isout of stock then a button **"Out of stock"** must be clicked such that it changes the pickingstatus in the table to the out of stock . once the required process is done the **"Send mail "button** is clicked so that using the activity starter the mail is initiated to the respective mailid.

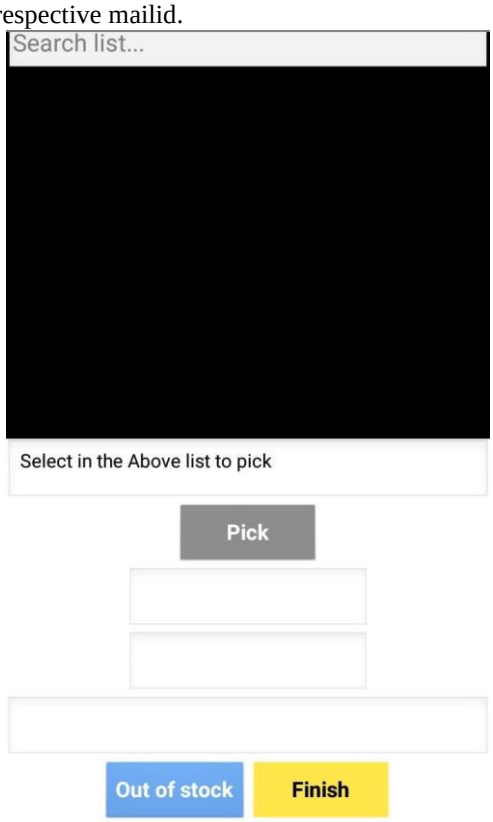

Figure 4 - Pick Screen

Figure 5 - Send Mail Button

Output Design

The output design should focus on delivering relevant data, reports, and notifications to users, ensuring effective communication and decision-making. After completing a picking task, the output design should confirm the successful completion anddisplay the relevant details, such as the picked item, date, and time of confirmation.

V. CONCLUSION

Improving the efficiency of machine part maintenance is the first key outcome. Critical parts can be selected using QR codes and an automatic exception workflow can be implemented. One way to keep tabs on a product's progress during its lifecycle is through traceability. The production process, beginning with the arrival of raw materials and ending with the delivery of final goods, can be tracked with traceability. A list of products is reviewed daily, and products themselves are reviewed daily. However, since they build every day, it is indisputable that ticking items off the list is important. The system's performance was also determined to be satisfactory throughout testing. Every required output has been produced. So, this technology makes it simple to automate all the consumption-related tasks. After reviewing the application, it is clear that it meets all requirements. To ensure client happiness, it functions as a file-sharing platform for important materials.

REFERENCE

[1] Hernandez, C., Aslankoohi, E., Frolikov, P., Li, H., Kurniawan, S., & Rolandi, M. (2023). Implementing QR codes in academia to improve sample tracking, data accessibility, and traceability in multicampus interdisciplinary collaborations. *Plos one*, *18*(4), 1-12.

[2] Zhang, S., Liao, J., Wu, S., Zhong, J., & Xue, X. (2021). A traceability public service cloud platform incorporating IDcode system and colorful QR code technology for important product. *Mathematical Problems in Engineering*, *2021*, 1-15.

[3] Chen, R., Yu, Y., Xie, S., Zhao, H., Liu, S., Ren, J., & Tan, H. Z. (2020). Rewritable and sustainable 2D barcode for traceability application in smart IoT based fault-tolerant mechanism. *Sustainability*, *12*(17), 1-14.

[4] Pipatprapa, A. (2019). QR Code on Mobile Platform for Improving Order Picking Process of Lean Factory Warehouse. *International Journal of Innovation, Management and Technology*, *10*(1), 56-60.
[5] Sung, H.C. (2020). Can online courts promote access to justice? A case study of the internet courts in China. *Computer Law & Security Review*, *39*.
[6] Larsen, D.A., Bernier, A. and Mahoney, J., (2020). Connecticut Annual Pavement Report. Bureau of Engineering and Construction Pavement Unit. *Connecticut Department of Transportation*.
[7] Gregory, S., Singh, U., Gray, J., & Hobbs, J. (2021). A computer vision pipeline for automatic large-scale inventory tracking. In *Proceedings of the 2021 ACM Southeast Conference*, 100-107).
[8] Sahi, R. S., He, Z., Silvers, J. A., & Eisenberger, N. I. (2023). One size does not fit all: Decomposing the implementation and differential benefits of social emotion regulation strategies. *Emotion*, *23*(6).
[9] Luo, Y., Liang, P., Wang, C., Shahin, M., & Zhan, J. (2021). Characteristics and challenges of low-code development: the practitioners' perspective. In *Proceedings of the 15th ACM/IEEE international symposium on empirical software engineering and measurement (ESEM)*, 1-11.
[10] Kim, H.S., Oh, D.S., & Kim, S.H. (2023). Cloud-based ERP construction process framework in the customer's perspective. *Computer Science and Information Systems*, *20*(1), 25-50.