

**QUALITY CHECK OF ENGINEERING DOCUMENTS – INGENIA**

**Daniel Halls (Client)**

**Georg Grossmann (Mentor)**

**FINAL PROJECT REPORT**

**GROUP 2**

**By: -**

**Vinit Dave (110336340)**

**Sindhuja Reddy Gade (110334632)**

**Sujan Gaha Magar (110355716)**

**Madeline Goldy (110369159)**

**Li Guan (110370777)**

**Date – 25/10/2023**

**Executive Summary**

This project report details the efforts of our team, in collaboration with Ingenia, to address the identified challenges in the manual quality assessment of their engineering documents. As a leading design and engineering firm in South Australia, Ingenia recognized that the existing manual system, characterized by inconsistencies and recurrent issues, was not in line with their commitment to innovation and excellence.

The primary objective of this project report is to detail the strategies and methodologies employed to refine the assessment process, minimize manual dependencies, and elevate quality standards. Our approach was multifaceted, involving meticulous planning, innovative design, and stringent execution. A notable milestone was the creation of a digital checklist design prototype using Figma. Integral to the project was detailing the technical requirements essential for full implementation. Additionally, to implement the automation of checklist tasks, we explored the potential of Optical Character Recognition (OCR) technologies. Our evaluations, as documented in this report, concluded that PaddleOCR outperforms its counterparts (Tesseract), especially in extracting text from intricate engineering drawings. While OCR has made significant strides, there's still a clear need for models that can understand and organize data contextually, especially in specialized fields like engineering. Future research should aim to bridge this gap, ensuring that the technology is both advanced and accessible.

**Table of Contents**

[**1.** **Introduction** 3](#_Toc149860774)

[**2.** **Implementation Approach** 5](#_Toc149860775)

[**3.** **Solution Design** 6](#_Toc149860776)

[**3.1.** **Stakeholder Analysis & Swim Lane Diagram** 7](#_Toc149860777)

[**3.2.** **AS-IS Process and Diagram** 8](#_Toc149860778)

[**3.3.** **TO-BE Process and Diagram** 9](#_Toc149860779)

[**4.** **Digitalization of Checklist** 10](#_Toc149860780)

[**4.1.** **Overview** 10](#_Toc149860781)

[**4.2.** **Process Flow** 10](#_Toc149860782)

[**4.3.** **Challenges** 12](#_Toc149860783)

[**4.4.** **Detailed Technical Requirements for a Full Implementation** 14](#_Toc149860784)

[**5.** **Automation of Checklist Tasks** 17](#_Toc149860785)

[**5.1.** **Overview** 17](#_Toc149860786)

[**5.2.** **Methodology** 17](#_Toc149860787)

[**5.2.1 Information Block** 18](#_Toc149860788)

[**5.2.2 Preprocessing the Data** 19](#_Toc149860789)

[**5.2.3 OCR Library Test and Evaluation** 20](#_Toc149860790)

[**5.2.4 Result Analysis** 22](#_Toc149860791)

[**5.3.** **Challenges and Opportunities** 23](#_Toc149860792)

[**6.** **Conclusion** 24](#_Toc149860793)

[**6.1.** **Project Outcomes** 24](#_Toc149860794)

[**6.2.** **Key Learnings** 25](#_Toc149860795)

[**7.** **References** 25](#_Toc149860796)

[**8.** **Appendix** 27](#_Toc149860797)

[**Appendix A - Figma Wireframes** 28](#_Toc149860798)

[**A-1: Login Page** 28](#_Toc149860799)

[**A-2: Home page** 29](#_Toc149860800)

[**A-3: Checklist Management Pages** 30](#_Toc149860801)

[**A-4: Upload Document Page** 32](#_Toc149860802)

[**A-5: OCR Extract Pages** 34](#_Toc149860803)

# **Introduction**

Ingenia, a company that offers specialized design and engineering services in South Australia, has consistently been at the forefront for its commitment to delivering full lifecycle project support from initiation to completion. Known for its innovative, practical, and cost-effective solutions, the company has recently encountered a pressing concern within its workflow, specifically regarding the quality assessment process for its engineering documents that demands prompt attention.

**Challenges Faced by Ingenia in the Quality Assessment Process**:

* **Manual Quality Checks**: Ingenia's experts manually check engineering work using checklists and a PDF annotation tool, focusing on format, accuracy, and standards.
* **Reviewer Reliance**: The quality assessment process relies heavily on the expertise and time of individual reviewers, leading to inconsistent evaluations and variable timelines.
* **Minimal Automation**: With little automation, tasks like sending reminders and tracking evaluations are done manually, causing delays and mistakes.
* **Scattered Documentation**: Documents are either physical or scattered digitally, making it hard to manage versions and access files efficiently, causing confusion.
* **Collaboration Issues**: There's a lack of tools for seamless collaboration, with crucial interactions often happening outside the established processes.
* **Inefficient Reporting**: Generating reports on document quality is labour-intensive, making it difficult to gain clear insights into the quality of work.

**Motivation Behind the Project**: Recognizing these challenges, Ingenia is driven to transform its quality assessment process. There is a compelling need to reduce the dependence on manual checks, bolster accuracy, and make the entire process more streamlined. The transition to a digital and automated system promises enhanced efficiency and a consistently high standard of quality checks. This project resonates with Ingenia's pioneering spirit and its unwavering commitment to addressing operational challenges.

**Key Solutions Offered in This Report:**

**1. Design Prototype and Requirements for Full Implementation of Digitalised Checklist**

This project report outlines the design and strategic approach to digitalize Ingenia's quality checklist. By leveraging tools like Figma, a prototype is developed, offering a glimpse of the envisioned digital checklist. The requirements segment delves into the detailed technical requirements to transform the traditional paper-based checklist into a fully functional web-based digital platform.

**2. Research on Automation of Checklist Tasks**

A focal point of this project report is the exploration of automation technologies, specifically Optical Character Recognition (OCR). Two primary OCR libraries, Tesseract and PaddleOCR, are under scrutiny to determine their suitability and efficacy for Ingenia's requirements.

**Resources**: For this project, Ingenia has provided several engineering drawings that require quality checks. These are complemented by the quality checklists that Ingenia routinely uses for assessments. These vital resources underpin the solutions proposed in this project report.

This project report, while exhaustive, paves the way for the ongoing quest towards a more streamlined, automated, and digitized quality check procedure.

# **Implementation Approach**

Our group’s approach to accomplishing this project was to have regular weekly meetings with our mentor to discuss the progress of our project. In these meetings, we discussed our individual developments related to the project. Each member documented their progress in the weekly ongoing progress reports. We also had meetings internally with our team members whenever necessary to discuss our individual developments and to divide our tasks. We have shared our feedback, asked questions, and suggestions in our internal WhatsApp group. Below is the implementation approach that contributed to our project’s success.

**Project Initiation:**

* In week 1, we determined the goals and parameters of the project.
* We organised and designated our roles and duties.
* Arranged a kick-off meeting to present the initiative.

**Needs Evaluation:**

* In week 2 and 3, to comprehend the context of the project, we gathered and did initial research.
* To guarantee alignment, we had regular meetings with the stakeholders of our project.

**The phase of Planning:**

* In week 4, we created a thorough project plan that outlines the resources, deadlines, and milestones.
* Set up project planning meetings every week to assess our progress.
* Presented our project plan to our client and received some valuable feedback to improve our project’s scope.

**Creation and Design:**

* From week 5 until week 13, we distributed relevant sections of our project deliverables according to each member's area of competence and expertise.
* Used online communication tools and team gatherings to foster better collaboration.

**Execution and Examination:**

* Followed the set project plan when executing the deliverables.
* Examined the progress often to find and fix any problems we had encountered.

**Guarantee of Quality:**

* We made sure that the project satisfied established quality requirements.
* Reviewed our submissions regularly and resolved any differences.
* Discussed with our mentor, Georg, about the project's progress for improvements.

**Delivering Final Report:**

* Divided our project's final report sections to each team member.
* To guarantee a seamless final report, we collaborated to ensure a better outcome.

**Project Finish:**

* Examined the project report in relation to its original goals and outcomes.

**Project Report and Presentation:**

* In week 13, we planned a presentation to highlight the project's achievements.
* Demonstrated the project to our client and our mentor.

1. **Solution Design**

## **Stakeholder Analysis & Swim Lane Diagram**

The analysis is assuming that this project will be implemented in future with the below given stakeholders:

1. **Project Sponsors (Ingenia)**

Project Sponsors (Ingenia) are the primary stakeholder with a vested interest in the successful implementation of the Quality Check of Engineering Documents project. Their concerns include ensuring the precision, uniformity, and compliance of engineering documents. They are deeply invested in the automation of document review processes to enhance efficiency, reduce human errors, and ensure adherence to industry standards. Ingenia's reputation and credibility heavily rely on the quality of engineering documents they provide to their clients. Therefore, they are deeply concerned about the accuracy and consistency of these documents. Their role in the project is crucial as they provide the initial checklist, define project goals, and validate the end product.

1. **End Users (Engineers- Ingenia)**

The end users are the ones who will directly interact with the application. They are concerned with having a user-friendly interface, efficient functionality, and the ability to easily manage and assess engineering documents. Their feedback on usability, feature requirements, and overall experience is vital. Engineers need tools that aid their documentation processes, and project managers require efficient tracking mechanisms. Draftspersons are interested in seamless collaboration, and administration and management need reliable reporting. Addressing their needs is crucial for the app's success.

1. **Software Developers**

Software developers play a pivotal role in the project. They are responsible for providing necessary software or hardware components, ensuring integration requirements, and maintaining long-term business relationships. Their expertise is crucial for the technical implementation of the project, especially in aspects related to OCR integration, app development, and seamless collaboration with tools like Drawboard Projects.

1. **Project Manager (Ingenia)**

The project manager ensures to take the requirements from the project document and goals. And publish the Project Goals and Sprint Planning. Create JIRA Tickets and assign to the software developers’ team. Project Managers are also responsible for coordination and Sprint Reviews. They also validate the end product and in turn sends it to the project sponsors for final sign off.

A diagram of a flowchart

Description automatically generated

Figure : Swim Lane Diagram

## **AS-IS Process and Diagram**

To guarantee that engineering designs, plans, and reports are accurate, comprehensive, and compliant, quality checks of engineering documents are essential. There are several processes in this procedure, and people with experience in document control or engineering specifically usually handle them. Currently, Ingenia is following a manual process of quality checking for engineering documents. Initially, the quality check reviewers receive the required document from the end users, then they check the quality and compliance of the document according to the company standards and specific document checklist. If there is any error in the document, then the reviewer sends the document to the employees to fix the mistakes. After the errors are corrected, the quality check reviewer rechecks the document again and if none found, the document is forwarded to the end user for usage. The below AS-IS diagram demonstrates the detailed process that Ingenia is currently following.

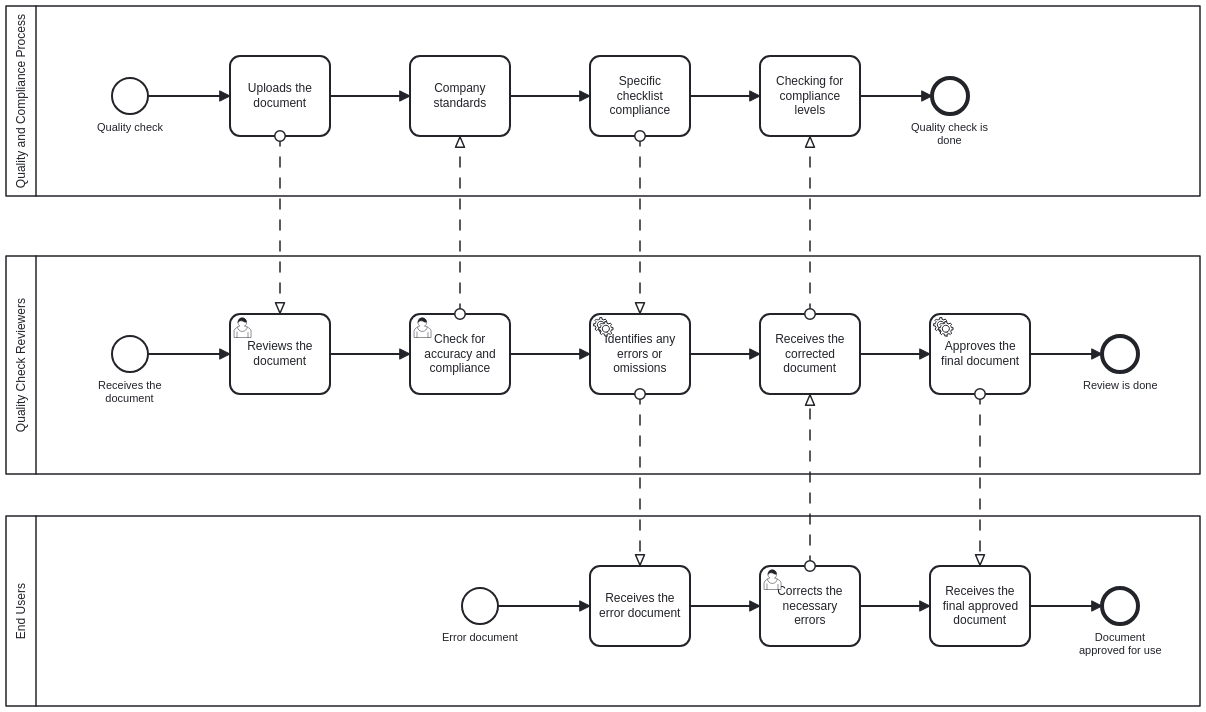


Figure : As-Is Process Diagram

## **TO-BE Process and Diagram**

The To-Be process that we recommend to Ingenia is to use an application that can automatically check for the quality of the document according to the company standards and checklist compliance. The To-Be process would be that the end user initially accesses the application, logs in with the credentials, and access the user page. From then, the end user would create a new project and selects the specific checklist that is relevant to the uploaded document then in OCR extract section, they need to select correct paper type for the information block. Then the checks for checklist completion are done and then the end user would be able to access the checked document which is ready for use. The user database extracts or inputs the data of every check that is being done in the app platform. It stores the user credentials for confirmation for logins, stores the uploaded document for future use, extracts the one of the stored checklist types when selected accordingly and checks with the relevant checklist stored and keeps a copy of the final checked document to access it later.

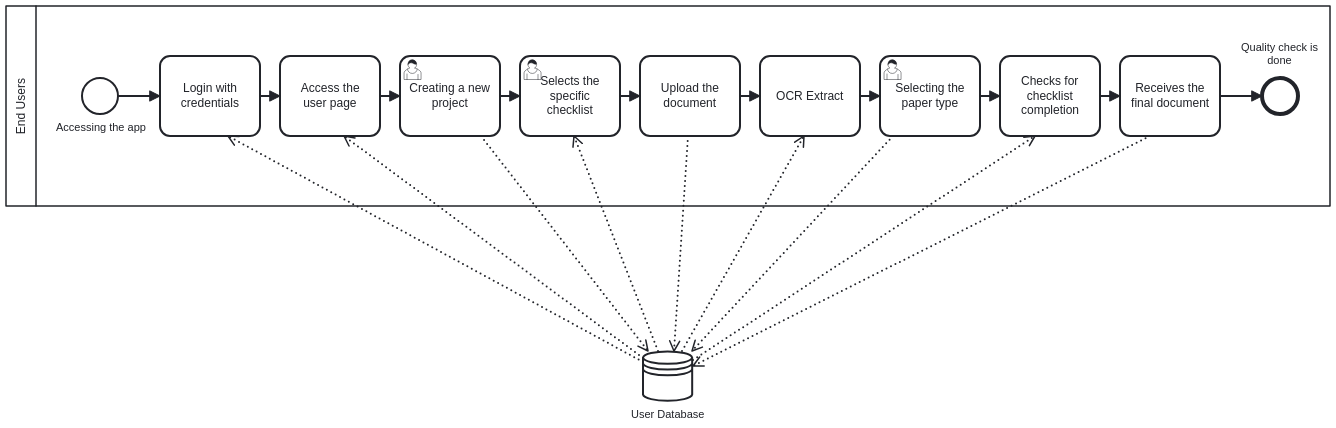


Figure : To-Be Process Diagram

1. **Digitalization of Checklist**

## **Overview**

The digitalization of checklists has emerged as a pivotal solution in streamlining processes, enhancing efficiency, and minimizing errors in various domains. In the context of engineering document quality management, the integration of a digital checklist system promises to revolutionize how project data is handled. This transition is exemplified through the Figma prototype, which serves as the visual representation of the solution, offering a tangible interface for users and stakeholders to interact with the system. The Figma prototype encompasses a manager's view, emphasizing the importance of user roles and access configuration.

[**FIGMA ACCESS LINK**](https://www.figma.com/proto/LdNRWXMTB6DzYL4rBy1nQB/Figma-basics?type=design&node-id=624-661&t=YOGyxjXx5yKsCiFM-0&scaling=contain&page-id=1669%3A162202&starting-point-node-id=624%3A661)

## **Process Flow**

* **Login Page:**

The journey begins with the user logging in or creating an account, establishing a secure and personalized environment for checklist management. (See Appendix A - Figma Wireframes A-1 for the view of Login Page)

* **Home Page:**

The home page serves as the central dashboard, offering an overview of crucial statistics such as total uploaded files, recent transmittals, and incomplete items. Additionally, the pop-up cards offer detailed insights into projects, including due dates, status, owners, assignees, and last modifications. This page acts as a launchpad, providing shortcut steps to essential functionalities like the checklist management page and upload document page. (See Appendix A - Figma Wireframes A-2 for the view of Home Page)

* **Checklist Management Page:**

Users navigate to the Checklist Management page, a hub where they organize and oversee their checklists. The "Checklist Overview" consolidates information about checklist names, associated projects, assignees, creation details, and modifications. Additionally, users can configure checklists for specific checklist items and set default checklist types through a sub-page. (See Appendix A - Figma Wireframes A-3 for the view of Checklist Management Pages)

* **Upload Document Page:**

Engineers upload engineering drawings, choosing to either add to an existing project or create a new one. This page integrates checklist tasks based on the user's configurations, ensuring adherence to quality metrics. New users are seamlessly guided through the process of creating a new project. (See Appendix A - Figma Wireframes A-4 for the view of Upload Document Pages)

* **OCR Extract Page:**

Following the submission of documents, the OCR extract page initiates a multi-step process. Initially, it presents an information block, drawing, extracted metadata, and data checks. The subsequent step involves generating a checklist report table, reflecting user-configured checklists. Manual modifications can be made to checked icons and data checks from Initial process, ensuring flexibility and accuracy. After the documents undergo processing, the OCR extract page offers a summary of processed files, including names, projects, statuses, and upload details. Additionally, project visibility and addition are streamlined through a dedicated view projects button, this feature allows user to see file competition status and go back to the documents undergo processing for pending tasks or even completed tasks. (See Appendix A - Figma Wireframes A-5 for the view of OCR Extract Pages)

A diagram of a process

Description automatically generated*Figure 4: User flow diagram*

## **Challenges**

1. **Integration Complexity**

Integrating the checklist system with existing project management tools, databases, or platforms can be complex. Ensuring seamless interoperability without disrupting ongoing processes is crucial.

Possible Solution: Conduct thorough compatibility assessments and provide clear integration documentation. API (Application Programming Interface) standards should be followed to facilitate smooth data exchange between systems (Balsari et al., 2018).

1. **Configurability**

Different engineering projects may have unique checklist requirements. Creating a system that caters to this diversity while maintaining configurability is challenging.

Possible Solution: Design the checklist system with a modular and configurable architecture. Allow users to customize checklist templates, create project-specific configurations, and easily adapt the system to different workflows.

1. **Feedback Incorporation**

Effectively incorporating user feedback and iterating the system to align with evolving needs is an ongoing challenge.

Possible Solution: Implement a feedback loop mechanism, such as a dedicated feedback iteration page, where users can provide input. Regularly analyse feedback, prioritize feature requests, and release updates to address user concerns and improve the system iteratively.

1. **User Roles and Access Configuration**

Managing user roles and access levels is crucial, especially in environments where different stakeholders, such as engineers and managers, require different levels of access.

Possible Solution: Implement Role-Based Access Control (RBAC) to define roles such as engineers, managers, and administrators (Lu et al., 2015). Each role should have specific permissions tailored to their responsibilities. This ensures that users only have access to the functionalities relevant to their roles.

1. **User Adoption**

Transitioning from traditional paper-based or manual checklists to a digital system might face resistance from users accustomed to established workflows. Some may find it challenging to adapt to new technologies and interfaces.

Possible Solution: To address this, a comprehensive training program and user onboarding process should be implemented. The user interface should be designed with simplicity in mind, ensuring an intuitive experience for both new and experienced users.

1. **Efficient Workflow Management**

Ensuring an efficient workflow that aligns with project requirements and deadlines can be challenging.

Possible Solution: Design the system to allow project managers to set deadlines, prioritize tasks, and track progress efficiently. Incorporate visual cues, such as progress bars and status indicators, to provide users with a quick overview of project statuses.

1. **Auditability and Compliance**

Meeting auditing requirements and ensuring compliance with industry standards is essential, especially in regulated industries.

Possible Solution: Implement features that enable audit trails, logging of user actions, and generation of compliance reports. This ensures that the system can withstand audits and adhere to industry-specific regulations.

1. **Data Security**

The digitalization of checklists involves handling sensitive project data. Ensuring the security and privacy of this information is a paramount concern.

Possible Solution: Implement robust encryption methods, secure data storage practices, and access controls. Regular security audits and compliance with industry standards contribute to a secure system.

## **Detailed Technical Requirements for a Full Implementation**

In the realm of software development, conceptualizing a solution represents only the initial step. Bringing this vision to life requires a holistic approach that addresses every technical facet. The following elucidation provides the comprehensive technical requirements for a complete implementation:

**1. Constructing the Digital Infrastructure - System Architecture & Components**

To lay a solid foundation for Ingenia's web-based app, several architectural and component choices are paramount. These are:

* **Web Server Selection:** The web server plays a pivotal role in ensuring the application remains responsive and can handle concurrent requests. For Ingenia's web-based app, it's recommended to use **Nginx** which is a high-performance web server that can manage web traffic efficiently (Soni, 2016). It's designed to be fast and is known for handling multiple requests simultaneously without a dip in performance.
* **Data Management:** A secure storage hub is imperative for data management. For Ingenia, the **PostgreSQL** database server is recommended for its extensibility and advanced data types (Juba & Volkov, 2019). It allows user to store data securely and access it quickly, which is vital for good performance and reliability.
* **Driving the Core (Backend):** The core operations should be carefully governed to be able to handle increased workloads without performance dips. For Ingenia, it is recommended to use **Node.js with Express.js**. Node.js is a runtime for executing JavaScript server-side, and Express.js is a framework for building web applications on top of Node.js (Peters, 2017). They're lauded for their non-blocking, event-driven architecture which makes applications built on them very scalable.
* **User Experience (Frontend):** Ingenia's web-based app needs a dynamic interface. React, a free, open-source frontend library for building user interfaces, is the recommended frontend framework for its component-based architecture and efficient state management (Gackenheimer, 2015).

**2. Features and Functionalities - Functional Requirements**

For Ingenia's web-based app to serve its intended purpose effectively, there are essential features and functionalities to consider. They include:

* **Secured Access:** Users accessing the web-based app should have a secure login mechanism. It's recommended to use **OAuth 2.0** which is an authentication protocol that enables secure authorization (Fett et al., 2016). It means that users can log in securely and their data is protected.
* **Navigational Interface:** A user-friendly dashboard and homepage are crucial.
* **Document Management:** A system to upload, store, and manage various document formats is vital.
* **Checklist Creation & Association:** The web-based app should facilitate dynamic checklist management and association with documents or projects.
* **Automated Text Extraction:** As Ingenia seeks to digitalize checklists, **PaddleOCR**, a tool that can extract text from various document formats, is the recommended OCR Library for its accuracy and extensibility (Sathyanarayanan et al., 2020).

**3. Assuring Quality & Robustness - Non-Functional Requirements**

Beyond the basic functionalities, the quality and robustness of the web-based app play a crucial role in user satisfaction and system reliability. Important considerations are:

* **Performance:** Rapid uploads, processing, and retrievals are vital.
* **Scalability:** Ingenia's web-based app should be scalable. Using **Amazon Web Services (AWS)** is recommended as it offers a wide range of cloud services such as compute power, database storage, content delivery, and other functionality to help businesses scale and grow (Mathew & Varia, 2014).
* **Security:** Regular security audits, and data integrity are crucial.

**4. Seamless Integrations**

To enhance Ingenia’s web-based app capabilities and offer a holistic user experience, integrating third-party APIs is often required. **RESTful APIs** are recommended. These are protocols that enable different software applications to communicate with each other, making the app more versatile and connected (Patni, 2017).

**5. Testing and Quality Assurance**

Before launching Ingenia's web-based app, it's vital to ensure its stability, performance, and adherence to requirements. This involves:

* **Rigorous testing**: This encompasses unit, integration, and user acceptance testing.
* **Continuous Integration and Delivery:** To keep Ingenia's web-based app optimized, it's recommended to deploy **Jenkins.** Jenkins is an automation tool that will help deploy updates and integrate new code seamlessly, which means the app stays up to date with less manual effort (Seth & Khare, 2015).

**6. Hosting & Deployment**

Ingenia's web-based app should be easily accessible. Alongside choosing a reliable web hosting solution and securing a domain name, integrating with a Content Delivery Network (CDN) is vital. **Cloudflare** is the recommended CDN for its security emphasis and performance-enhancing capabilities (Dewi et al., 2019). Cloudflare CDN will distribute the app’s content across the globe, reducing loading times and improving security against attacks.

1. **Automation of Checklist Tasks**

## **Overview**

OCR technology is useful for processing scanned documents and images, including engineering drawings, but it becomes challenging with detailed technical drawings. Specialized adjustments are necessary to identify key text accurately, and further actions are required to ensure data accuracy after OCR. Extracting precise data from Geometric Dimensioning and Tolerancing (GD&T) boxes is vital for automated processes and high-quality outputs in engineering. The information in these boxes is essential for computer-aided manufacturing (CAM) software. Currently, there is no widely available open-source solution capable of effectively managing this entire process.

## **Methodology**

The focus of the project is on testing Optical Character Recognition (OCR) technology for extracting information from Engineering Drawings. The information in engineering drawings is categorized into two main parts: information blocks and tables, and Geometric Dimensioning and Tolerancing (GD&T) information. The project emphasizes OCR technology for automating the extraction of text from the information block section of PDF-format engineering drawings provided by Ingenia. The goal is to assess OCR libraries' ability to accurately extract text from this common file format. The emphasis on information blocks is due to the fact that it does not require specialized technical knowledge related to images, making it a suitable area for OCR testing.

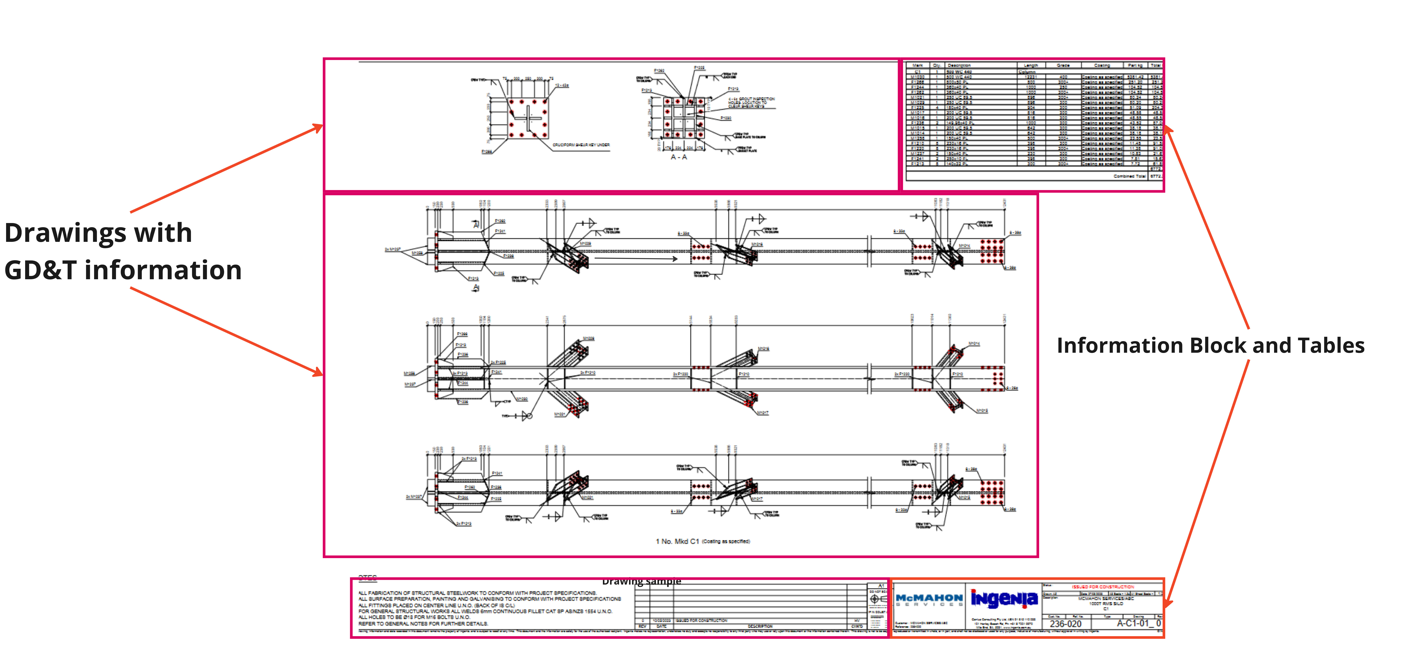


Figure 5. Major parts of Engineering Drawings

### **5.2.1 Information Block**

A mechanical drawing's "Information Block" section includes important textual elements that provide background and information about the drawing's contents. Understanding the goal of the drawing depends heavily on this section.



Figure 6. Information block

Usually, the "Information Block" contains the following:

* **Title:** This gives a brief overview of the drawing's contents, making it simpler to recognise the part, assembly, or system that it represents.
* **Drawing Number:** A unique reference number that identifies the drawing and helps with organisation and recordkeeping.
* **Revision Information:** This is where any updates or revisions to the drawing are noted, along with revision numbers, dates of modification, and explanations of the changes.
* **Material and Finish Specifications:** Details on the surface finishes, coatings, and treatments applied to the component or assembly, as well as the materials utilised in it.
* **Tolerances and Dimensions:** Accurate measurements, tolerances, and geometric dimensioning and tolerancing (GD&T) information that directs manufacturers and engineers through the manufacturing process.
* **Company or Project Information:** Details about the company or undertaking for which the drawing is created.
* **Notes and Special Instructions:** Extra annotations, observations, or special instructions pertaining to the drawing, such as assembly instructions, safety warnings, or other crucial details.
* **Part Number:** When ordering or identifying certain components, the part number—if it differs from the drawing number—is frequently given here.
* **Scale:** Information on the scale of the drawing in respect to the part's real size.
* **Name of Designer/Engineer:** The name of the person or group in charge of making the drawing.

### **5.2.2 Preprocessing the Data**

The process begins by opening a PDF in Jupyter Notebook and using the Fitz library (PyMuPDF) to convert the PDF into an image with a specified DPI. The focus is on extracting information from a specific block within the image. Two approaches are considered: identifying the block's position without altering the image or dividing the image into parts and extracting the target section from the bottom right corner. The chosen method involves dividing the image, converting it to grayscale, applying thresholding and morphological operations to create a binary image, and using the findContours function to efficiently locate and extract the desired information block. These steps prepare the image for further analysis and data extraction.

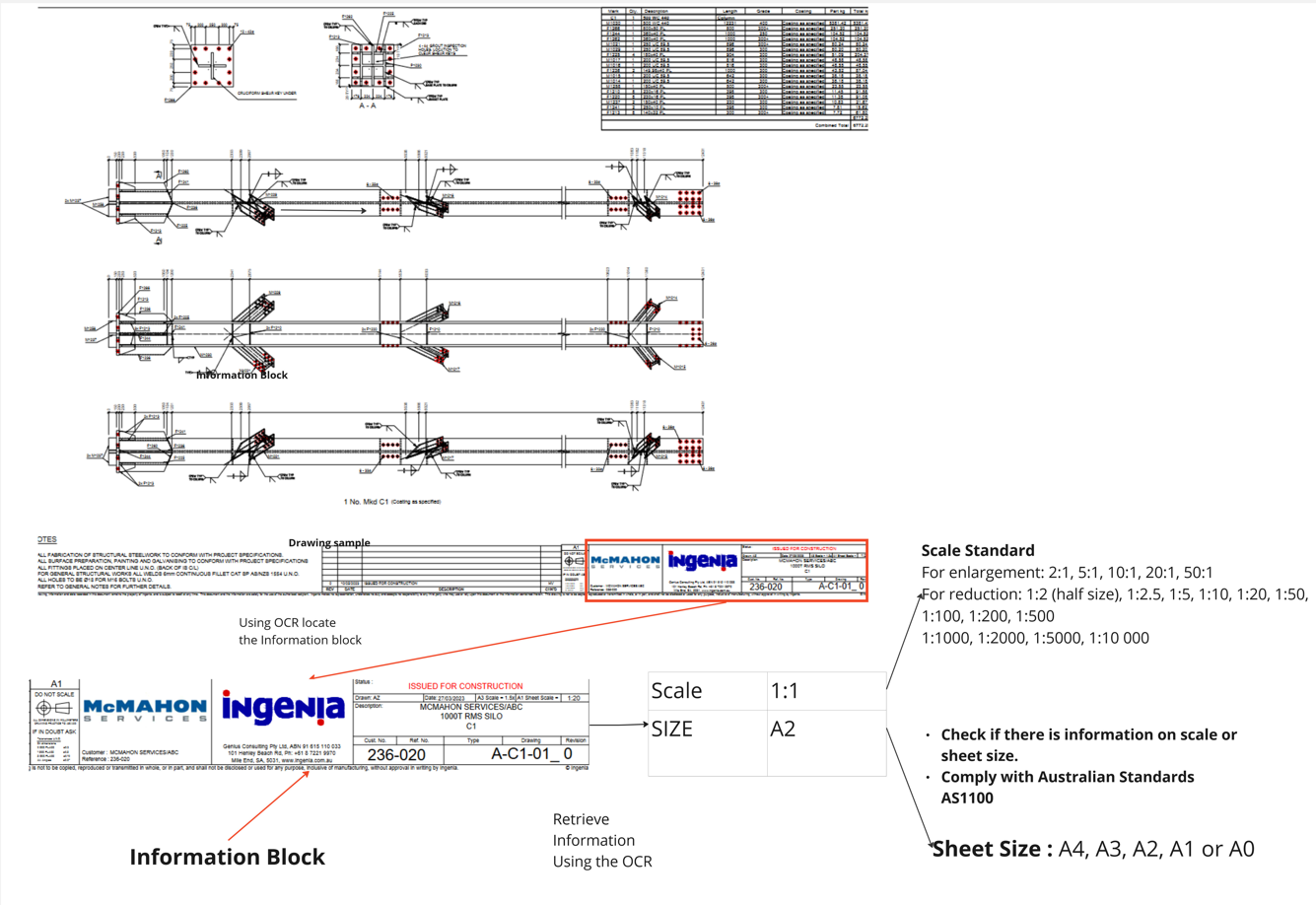


Figure 7. Task Automation Pipeline

### **5.2.3 OCR Library Test and Evaluation**

In our OCR library evaluation, we compared two popular OCR libraries: Tesseract and PaddleOCR, highlighting the distinct methodologies used by each library to extract text from images.

**Tesseract:** Tesseract is a widely used OCR library that excels at recognizing text within images. To extract text from a rectangular object using Tesseract, we start by identifying the rectangle boxes in the image that enclose the text regions. After this detection process, we separate the text properly to maintain the line structure. To perform the text extraction, we use the pytesseract.image\_to\_string(roi, config='--psm 6') function, where roi represents the region of interest, which is the detected text region in the rectangle. The config='--psm 6' parameter specifies the page segmentation mode, treating the image as a single block of text. This process allows us to extract the information or text from the rectangular object effectively. Finally, we store the extracted text in a DataFrame for further analysis or processing (*Pytesseract*, 2022).

**PaddleOCR:** PaddleOCR is another robust OCR library known for its efficiency in text extraction from images. In the context of PaddleOCR, we employ a component known as PPStructure, which plays a pivotal role in table detection and extraction. This component utilizes a combination of specific models tailored for various aspects of the OCR process, ensuring comprehensive and accurate results (PaddlePaddle, n.d.).

For the task of table extraction, PaddleOCR harnesses the power of the following models:

1. **Single Line Text Detection (DB):** The DB model is employed to detect single lines of text within an image, effectively identifying the boundaries and positions of these text lines.
2. **Single Line Text Recognition (CRNN):** The CRNN model is dedicated to recognizing and transcribing the individual text lines identified by the DB model, ensuring the accurate extraction of textual content.
3. **Table Structure and Cell Coordinate Prediction (SLANet):** SLANet is specifically designed to detect the structure of tables and predict the coordinates of individual cells within the tables. This enables the precise extraction of tabular data, making it a crucial component for structured data extraction.

These models collectively enable PaddleOCR to excel in tasks such as table detection and extraction, providing a comprehensive solution for extracting structured information from images.

### **5.2.4 Result Analysis**

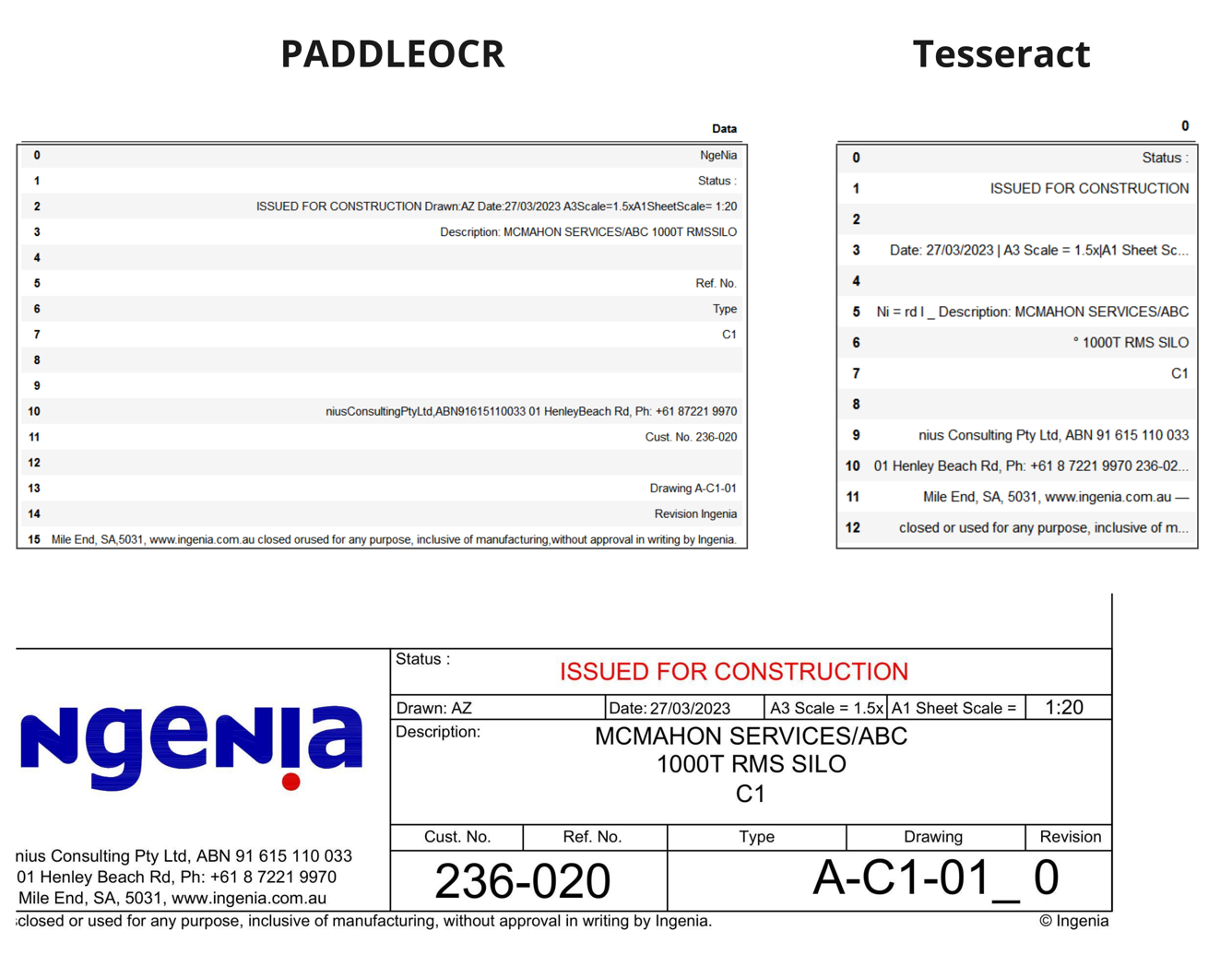


Figure 8 Result Comparison

Upon analysing the results from both PaddleOCR and Tesseract, it becomes evident that PaddleOCR consistently delivers more complete and accurate text detection. The key factor contributing to this superior performance lies in the models employed within the PPStructure component of PaddleOCR.

The models used in PaddleOCR, particularly within the PPStructure, are trained on extensive images having the table. This training data includes various examples of tabular content, enabling the models to learn and recognize the specific patterns, structures, and nuances present in tables. As a result, PaddleOCR excels in accurately detecting and extracting text from tables, as its models have been fine-tuned for this specific task.

In contrast, Tesseract, while a powerful OCR tool, may not be as specialized for table data, and its models might not possess the same level of proficiency in table-related text extraction. Consequently, PaddleOCR outperforms Tesseract when it comes to accurately recognizing and extracting text within tables.

## **Challenges and Opportunities**

OCR, or Optical Character Recognition, enables us to identify text within images or tables. However, the challenge lies in making sense of this text by organizing it into a meaningful context. Think of it like this: OCR can read individual characters, but it does not inherently understand how these characters form words, sentences, or the structure of a document. It is like having a jumbled puzzle; you can see the pieces, but you need to put them together to create a clear picture.

This disorganization makes it difficult to validate the extracted information. If text is not correctly structured, it is challenging to determine which words belong together, which section they are part of, and how they relate to one another. It is a bit like trying to understand a story without knowing the order of events or the roles of different characters.

It becomes essential to use a model trained on engineering drawings to address the problem of disorganised text derived from pictures or tables. Although OCR is capable of recognising text and characters, this model can do more. It comprehends the precise arrangement and structure of engineering drawings, identifying the relationships between various components. With the use of this specialised model, the extracted text may be arranged such that it makes sense when seen in the context of engineering drawings. It is aware of the terms, symbols, and dimensions that go together and in what sequence. This significantly improves our capacity to verify and understand the data, which makes it important in contexts like engineering and technical documentation where accuracy and precision are crucial.

A library or the package that is worth investigating is eDOCr, which is a packaged version of Keras-OCR that is intended to expedite the complete digitisation of mechanical engineering drawings (EDs). Keras is a deep learning library and exploring deep learning entails exploring the most advanced areas of machine learning. It requires specific expertise in Python programming as well as a thorough comprehension of the technicalities of deep learning methodologies. However, we had difficulties testing this specific library because to the technical difficulties related to Keras and the unique needs of the eDOCr tool (Toro et al., 2023).

1. **Conclusion**

## **Project Outcomes**

The project can be implemented into 2 phases as below based on the project goals:

**Phase 1: Digitalizing Ingenia's Checklist**

**Requirements Gathering:**

* Detailed understanding of the checklist provided by Ingenia.
* Discussions with Ingenia to clarify any ambiguities in the checklist.

**App Development:**

* Choose appropriate programming languages and frameworks for web and mobile platforms.
* Develop a user-friendly interface for uploading, organizing, and tracking engineering documents.
* Implement checklist management system allowing Ingenia and users to create, modify, and assign checklists.

**Quality Metrics Integration:**

* Implement algorithms for AI analysis (e.g., quality score, readability, consistency).
* Integrate natural language processing algorithms for grammar and syntax checks.
* Develop a reporting system for users to view quality metrics.

**Phase 2: OCR for Drawing Automation**

**Research and Development:**

* Investigate OCR technologies suitable for engineering drawings.
* Develop algorithms to automate checklist items related to drawings (e.g., scale verification, legibility).

**Integration and Testing:**

* Integrate OCR functionalities into the app.
* Test OCR automation with sample engineering drawings to ensure accuracy.

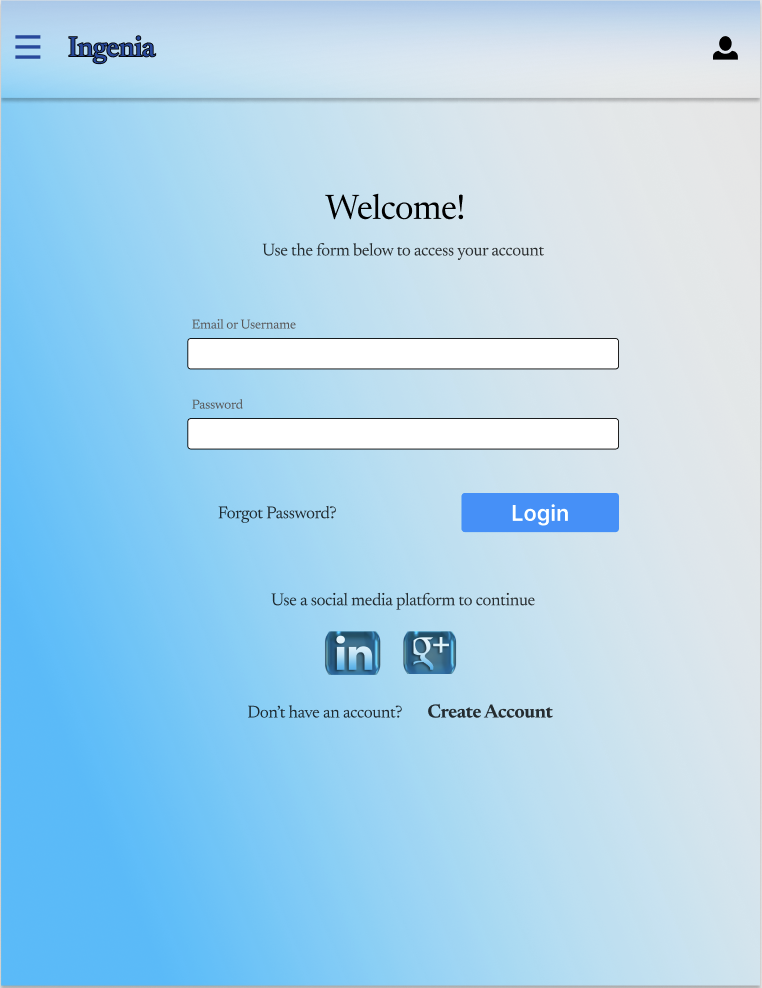
## **Key Learnings**

The major key learning is as we had adopted the Agile Approach, we should have done more stakeholder meetings as a part of weekly Sprints which would have provided us with more insights on the project goals. As we already have done this Sprint Planning with the Georg on a weekly basis.

1. **References**
2. Balsari, S., Fortenko, A., Blaya, J. A., Gropper, A., Jayaram, M., Matthan, R., Sahasranam, R., Shankar, M., Sarbadhikari, S. N., Bierer, B. E., Mandl, K. D., Mehendale, S., & Khanna, T. (2018). Reimagining Health Data Exchange: An Application Programming Interface-Enabled Roadmap for India. *Journal of Medical Internet Research*, *20*(7), e10725–e10725. https://doi.org/10.2196/10725
3. Dewi Estri, J. H., Umar, R., & Riadi, I. (2019). Implementation of cloudflare hosting for speeds and protection on the website. *Universitas Ahmad Dahlan*.
4. Fett, D., Küsters, R., & Schmitz, G. (2016, October). A comprehensive formal security analysis of OAuth 2.0. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (pp. 1204-1215).
5. Gackenheimer, C. (2015). *Introduction to React*. Apress.
6. Hardt, D. (2012). *The OAuth 2.0 authorization framework* (No. rfc6749).
7. Juba, S., & Volkov, A. (2019). *Learning PostgreSQL 11: A beginner's guide to building high-performance PostgreSQL database solutions*. Packt Publishing Ltd.
8. Lu, H., Hong, Y., Yang, Y., Duan, L., & Badar, N. (2015). Towards user-oriented RBAC model. *Journal of Computer Security, 23*(1), 107–129. https://doi.org/10.3233/JCS-140519
9. Mathew, S., & Varia, J. (2014). Overview of amazon web services. *Amazon Whitepapers, 105*, 1-22.
10. PaddlePaddle. (n.d.). PaddleOCR/README\_en.md at release/2.7 · PaddlePaddle/PaddleOCR.GitHub. <https://github.com/PaddlePaddle/PaddleOCR/blob/release/2.7/README_en.md>
11. Patni, S. (2017). *Pro RESTful APIs*. Apress.
12. Peters, C. (2017). Building rich internet applications with node. js and express. js. *Rich Internet Applications w/HTML and Javascript,* 15.
13. pytesseract. (2022, August 16). PyPI. <https://pypi.org/project/pytesseract/>
14. Sathyanarayanan, A., Srivatsan, M., & Meenakshi, K. (2022, January). Smart Web Dictionary using OCR. In *2022 International Conference on Computer Communication and Informatics (ICCCI)* (pp. 1-5). IEEE.
15. Seth, N., & Khare, R. (2015, December). ACI (automated Continuous Integration) using Jenkins: Key for successful embedded Software development. In *2015 2nd International Conference on Recent Advances in Engineering & Computational Sciences (RAECS)* (pp. 1-6). IEEE.
16. Soni, R. (2016). *Nginx.* Berkeley: Apress.
17. Villena Toro, J., Wiberg, A., & Tarkian, M. (2023). Optical character recognition on engineering drawings to achieve automation in production quality control*. Frontiers in Manufacturing Technology, 3*. <https://doi.org/10.3389/fmtec.2023.1154132>
18. **Appendix**

## **Appendix A - Figma Wireframes**

### **A-1: Login Page**



### **A-2: Home page**

A screenshot of a computer

Description automatically generated

### **A-3: Checklist Management Pages**

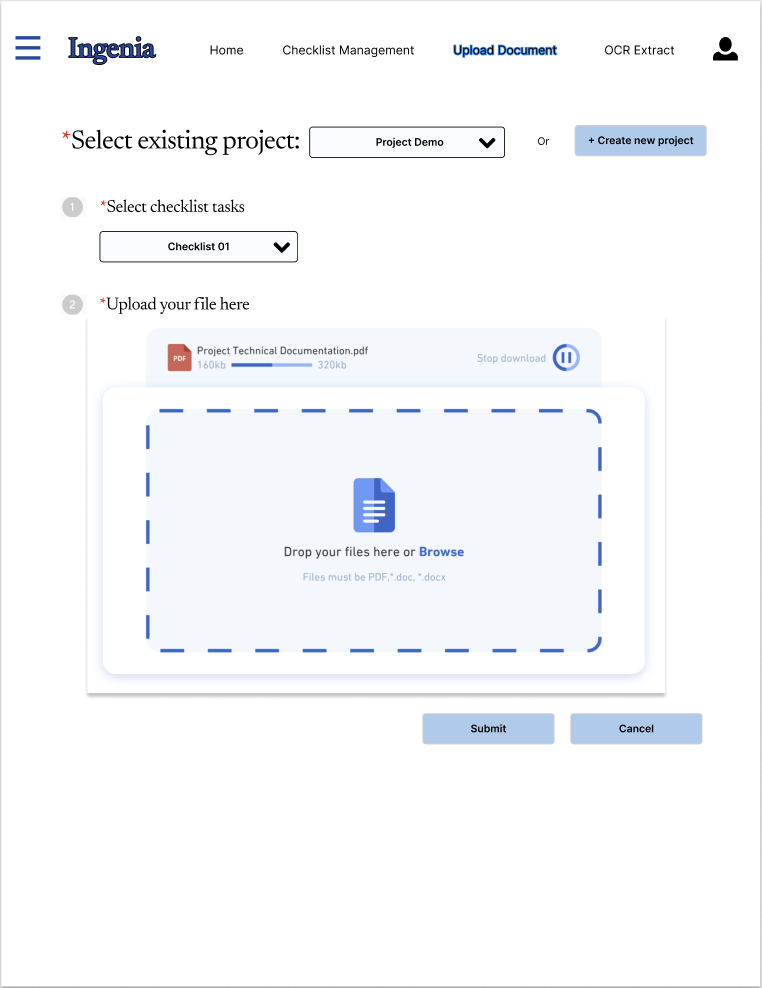
A screenshot of a checklist

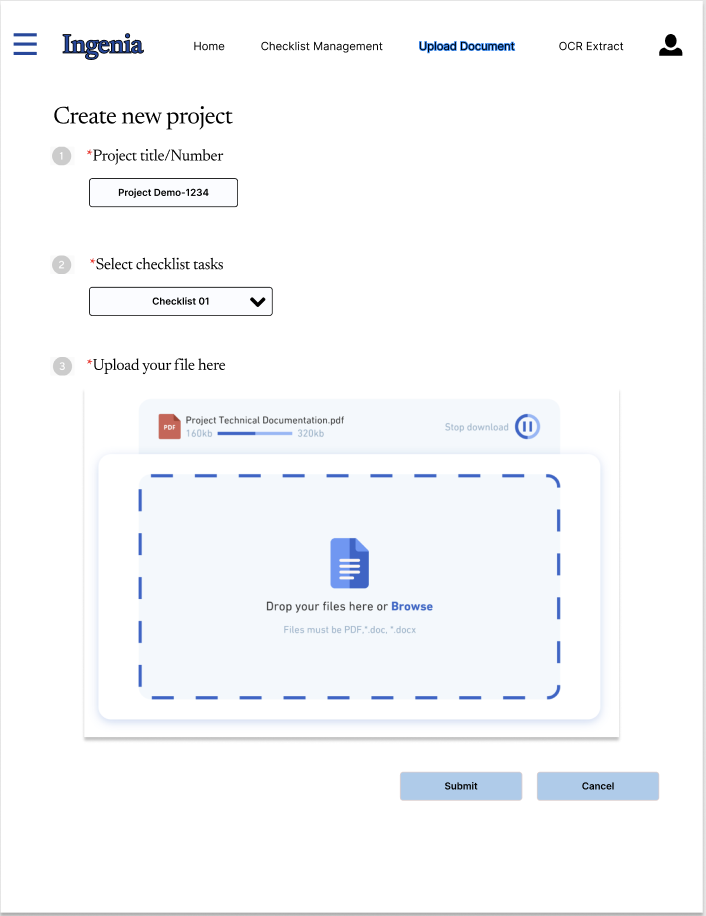
Description automatically generated

A screenshot of a checklist

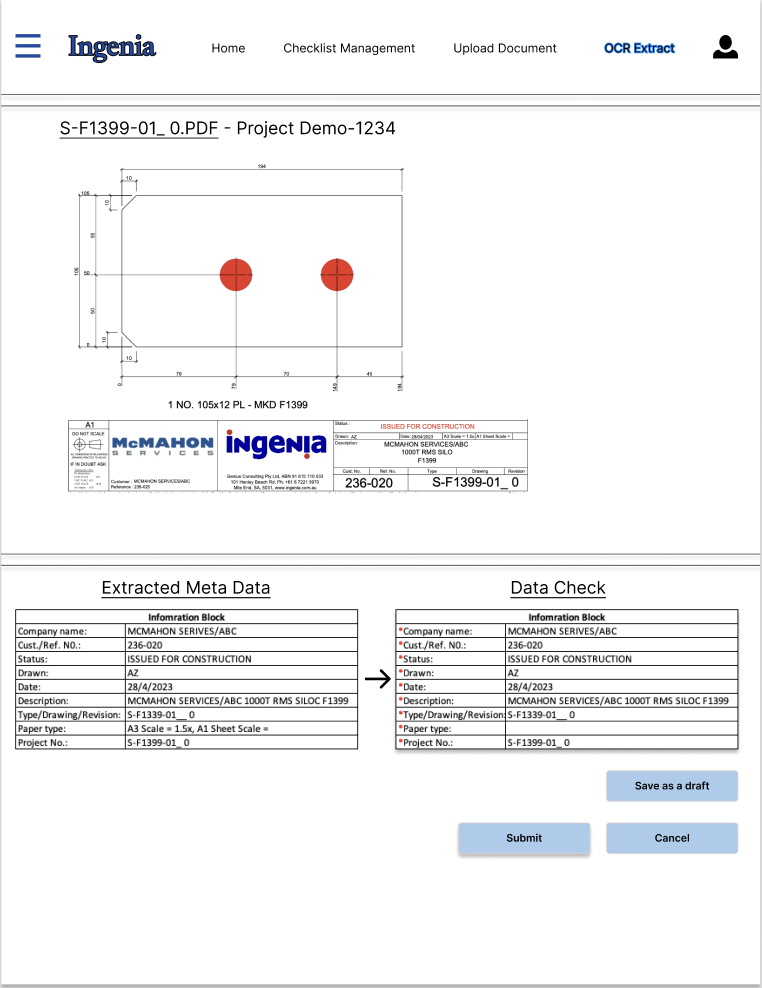
Description automatically generated

### **A-4: Upload Document Page**



****

### **A-5: OCR Extract Pages**



A screenshot of a checklist

Description automatically generated

