Project Survey Report-2

Team Members-
Tanisha Sahu Prajakta Darade Sondarva Princy Ganvir Devendra

Problem Statement:
Write a brief report on what software development process model you want to use for your project and why. Justify your choice based on the features you want to include from existing off-the-shelf commercial products (if any) and the new features you want to implement.

Some **Features of our project-**

- We want to create a Progressive Web App that will act as a document parser.
- The user will provide a document(s) (image format or pure text format) and a query as input and at the backend the most suitable documents will be retrieved with the exact position of query in it.

Software development process model-

Agile is a software development methodology that prioritizes flexibility and customer collaboration. It is based on the Agile Manifesto, a set of values and principles for software development that emphasizes adaptive planning, early delivery, and continuous improvement.

Agile model is a combination of incremental and iterative process models which means that software is developed in short cycles, with regular feedback from customers and opportunities for adjustment. This would open doors for improvements in our Web app at every stage of development. Agile methodologies prioritize delivering working software over comprehensive documentation, and they value individuals and interactions over processes and tools.

Here are the key characteristics of Agile:

Iterative and incremental development: Agile emphasizes delivering working software in small increments, with frequent feedback and opportunities for adjustment.

Flexibility: Agile methodologies are designed to be flexible, with the ability to adapt to changing requirements and priorities. This makes them well-suited for our project.

Customer collaboration: Agile prioritizes regular communication and collaboration with customers to ensure that the software being developed meets their needs.This helps us to ensure that the final product is what the customer actually wants and needs.

Continuous delivery: Agile values delivering working software early and often, which helps to provide value to customers as soon as possible and to get feedback on the product.

Continuous improvement: Agile methodologies encourage continuous improvement through regular reflection and adjustment of processes and practices. This helps us to ensure that the product being developed is of the highest quality.

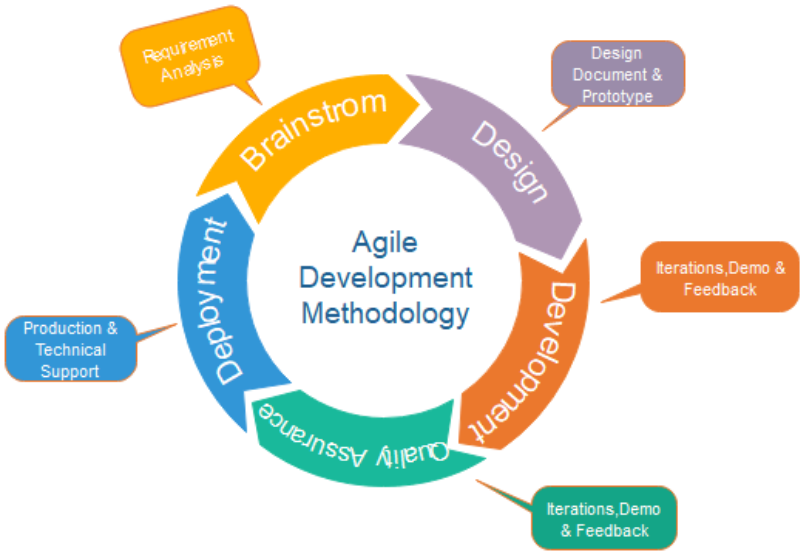

Fig. Agile Model

Transparent and open communication: Agile methodologies emphasize transparency and open communication, which helps to build trust and to ensure that everyone is on the same page throughout the project.

The Agile model provides a flexible and customer-focused approach to software development that is well suited for our project with rapidly changing requirements and tight timelines. By following an Agile methodology, we can deliver high-quality software that meets the evolving needs of the business and customers.

Considering all these properties of an Agile Model,it seems the best fit model for our project hence we choose to move ahead with this model.

We would look for the requirements initially, design the overall structure on the basis of these features then implement these features in the first stage of our development process.

In the later stage we would modify the existing features for better user experience if required. Otherwise we would implement advanced features for query retrieval with multiple filters and iterate over this process.

- In the first iteration we would be working on the extraction of text from images for the documents in the form of an image.We would be using different techniques such as OCR(optical character recognition) combined with Machine Learning Techniques for efficient text extraction.
- Then moving on to the second iteration we would be working for searching the input text in the extracted text from the image.(use of various python libraries or natural language processing models)
- The third iteration would include the creation of a frontend and the integration of the frontend with the text extracting as well as searching the backend we would have developed.

Since our methodology alligns with agile model, we decide to move on with it for our web-app.

- The purpose of a document parser is to automatically extract relevant information from unstructured text documents, such as emails, contracts, and reports. By doing so, a document parser can help organizations to save time, reduce costs, improve accuracy, and enhance productivity.
- Benefits:
- Increased efficiency and productivity: A document parser can automate the process of extracting information from large volumes of unstructured text documents, which can save time and reduce manual labor costs.
-
- Improved accuracy and consistency: A document parser can reduce the risk of errors and inconsistencies that can occur when information is manually extracted from documents.
- Enhanced data analysis: A document parser can enable organizations to quickly and easily analyze large amounts of data that would otherwise be difficult or time-consuming to process.
- Cost savings: A document parser can help to reduce costs associated with manual data entry, data processing, and data analysis.

- Scalability: A document parser can handle large volumes of documents, making it a scalable solution for organizations that need to process a large number of documents.
-
-
- Objectives:
- To develop a document parser that can accurately extract information from unstructured text documents.
- To improve the efficiency and productivity of data processing tasks.
- To reduce errors and inconsistencies in data extraction.
- To enhance data analysis capabilities.
- To reduce costs associated with manual data entry and processing.
- Goals:
- To achieve a high level of accuracy in data extraction from unstructured text documents.
- To develop a document parser that can handle a large volume of documents.
- To reduce the time and effort required to process large volumes of data.
- To enable organizations to quickly and easily analyze large amounts of data.
- To provide a scalable solution that can meet the needs of organizations of all sizes.
-
-

Extraction of structured data: A document parser can extract structured data from unstructured documents such as PDFs, Word documents, and web pages. This can include information such as names, addresses, phone numbers, dates, and other types of structured data.

Categorization of documents: A document parser can categorize documents based on their content or structure. For example, it can identify whether a document is a legal contract, a sales report, or a medical record.

Summarization of documents: A document parser can summarize the content of a document to provide a brief overview of its key points.

Keyword extraction: A document parser can extract important keywords from a document to aid in indexing and searching.

Language translation: A document parser can translate the content of a document from one language to another.

Data validation: A document parser can validate the accuracy and completeness of data in a document.

Data normalization: A document parser can convert data in a document to a standardized format for consistency and ease of use.

Integration with other systems: A document parser can integrate with other systems, such as a database or a content management system, to automate data entry and management.

Document redaction: A document parser can automatically identify and redact sensitive information, such as personal identification numbers, credit card numbers, or medical records, to protect the privacy of individuals.

Keyword search: A document parser can allow users to search for specific keywords or phrases within a document, making it easier to find relevant information.

Data aggregation: A document parser can aggregate data from multiple documents, such as sales reports or financial statements, into a single database or report.

Image and video analysis: A document parser can analyze images and videos embedded in a document, such as product photos or security camera footage, to extract information or identify objects or people.

Document summarization: A document parser can automatically generate a summary of a long or complex document, providing a concise overview of its main points.

Text-to-speech conversion: A document parser can convert text from a document into speech, allowing users to listen to a document instead of reading it.

Natural language processing: A document parser can use natural language processing techniques, such as part-of-speech tagging or named entity recognition, to analyze the syntax and structure of a document and extract meaning from it.

1. **Performance requirements:** The document parser may have performance requirements, such as processing speed, memory usage, and scalability. These

requirements should be clearly defined and documented in the software requirements specification.

2. **Legal and regulatory requirements:** The document parser may need to comply with legal and regulatory requirements, such as data privacy and security regulations. These requirements should be clearly defined and documented in the software requirements specification.

Acceptance criteria are a set of predefined conditions or requirements that a document parser must meet in order to be considered acceptable for use. The use of acceptance criteria in document parser development can help ensure that the final product meets the needs of its users and performs its intended functions effectively. By using acceptance criteria in document parser development, developers can ensure that the final product meets the specific needs and requirements of its users. This can help improve user satisfaction and increase the overall effectiveness of the document parser.

The functions of a document parser typically include:

Text extraction: The document parser must be able to extract text from the input document, even when the text is embedded within images, tables, or other non-textual elements.
Text normalization: The document parser must normalize the extracted text to remove noise and inconsistencies, such as extra white spaces, punctuation, and capitalization.

Entity recognition: The document parser must be able to recognize entities within the normalized text, such as names, dates, locations, and organizations.

Relationship extraction: The document parser may need to identify relationships between entities within the normalized text, such as identifying that a person is the CEO of a company.

Sentiment analysis: The document parser may need to perform sentiment analysis to determine the tone of the text, such as whether the text expresses positive, negative, or neutral sentiment.

Categorization: The document parser may need to categorize the input document into predefined categories, such as classifying a contract as a sales contract or a non-disclosure agreement.

Data integration: The document parser must be able to integrate with downstream data storage systems, such as databases and data warehouses, to store the extracted information in a structured format.

Feedback mechanism: The document parser may need to provide a feedback mechanism for users to verify and correct the extracted information to improve the accuracy of the parser.

These functions are typically implemented using various natural language processing (NLP) techniques, such as statistical analysis, machine learning, and rule-based approaches, depending on the requirements of the application.

Week 1:

Day 1-2: Research and select appropriate libraries for document parsing
Day 3-5: Plan the system architecture and data flow for the document parser
Week 2:

Day 1-3: Develop and test the document parser using the selected libraries
Day 4-5: Develop and implement necessary algorithms for data extraction and processing
Week 3:

Day 1-2: Develop a simple user interface for the document parser
Day 3-4: Thoroughly test and validate the document parser to ensure it meets the requirements
Day 5: Prepare documentation for the project
Week 4:

Day 1-2: Deploy the document parser in the target environment
Day 3-4: Provide user training and support for the document parser
Day 5: Review the project, gather feedback, and make necessary updates and improvements
Note that this schedule assumes that the project has a limited scope and that there is a dedicated team working on the project full-time. If the project is more complex or involves more stakeholders, it may take longer to complete. It's also important to

regularly review and adjust the schedule as needed to ensure the project stays on track and meets its goals.

Image doc
Tables
Lists

Features Implemented	Features left
Image to text	Need to format data in pdf line by line or in lists
Sentiment analysis of text	Text to speech(tts)
Emails Phone numbers Dates Credit Cards	Hyperlinks
Done with putting data in pdf and downloading the pdf	Semantic word search
Voice Input	Video
Bounding Boxes	Figures in pdf
Highlighting the text	Barcode qrcode
Summarization	Text details(font, size)
	Can translate text in doc to another language
	Titles (Subtopics)