CRITICALRIVER

Document Layout Analysis

Problem Statement & Solution

Problem Statement:

Develop an accurate document layout analysis model to automatically identify and categorize textual and non-textual elements within document images. The model should handle diverse document types, languages, and layouts, outputting a structured representation of the layout.

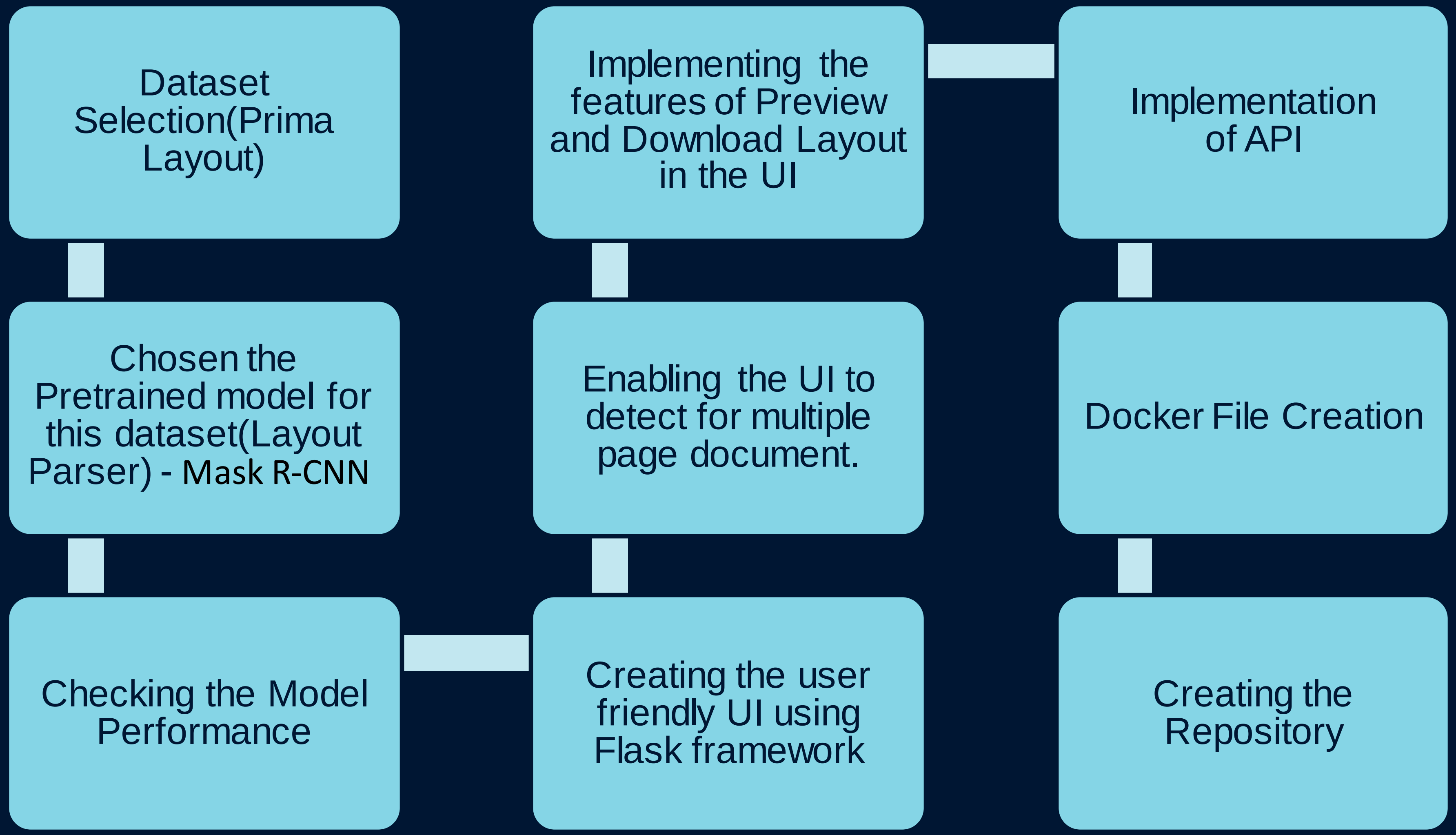
Solution:

The problem statement has been addressed through two solutions: a user-friendly Flask web application and a versatile Flask RESTful API. The web application enables seamless upload of PDFs or images, leveraging the Detectron2 layout analysis model to identify and visualize layout elements. This interactive interface allows users to understand the spatial arrangement of text regions, images, and tables in their documents. On the other hand, the RESTful API provides layout analysis functionality for PDFs and images, returning JSON responses with annotated bounding boxes. This API facilitates easy integration with various applications and services, ensuring scalability and adaptability. Both solutions empower users to gain valuable insights into their document layouts, fostering optimized OCR, document processing, and information extraction workflows.

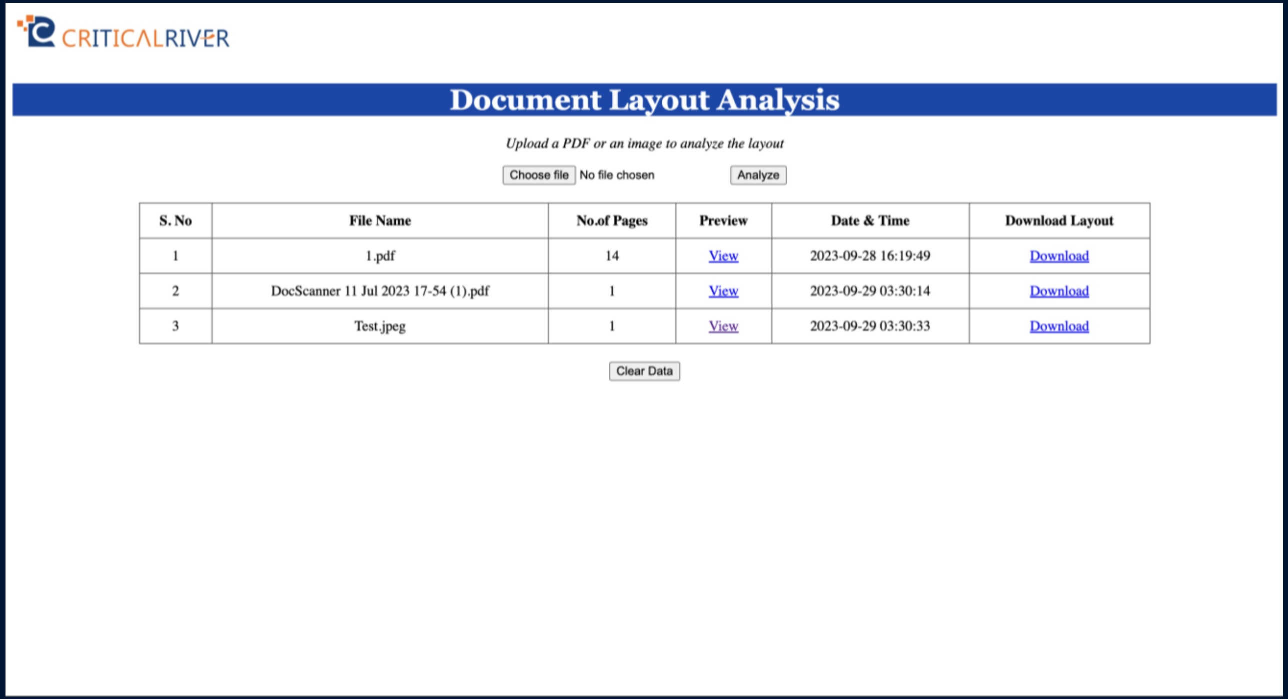
Steps Involved:

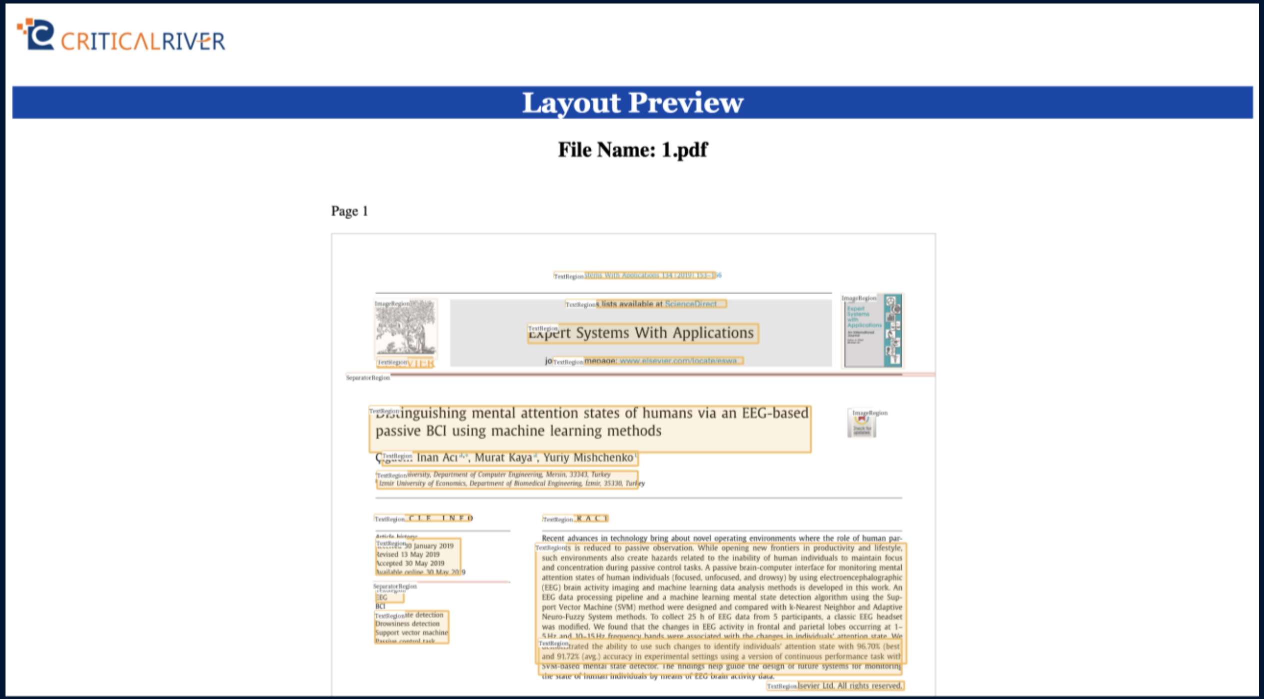
1. **Dataset Selection:** The Prima Dataset was chosen for its diverse content, which includes the scanned and native images/pdfs.
2. **Model Selection:** Chosen a pretrained layout analysis model(layout parser trained using Detectron2), that can accurately detect and classify text regions, images, tables, and other elements.
3. **Model Training and Evaluation:** We can train the model if we have the enough computational resources.
4. **Visualization:** Visualize the detected layout elements by drawing bounding boxes around them on the document images.
5. **Web Application Development:** Created a user-friendly Flask web application allowing users to upload PDFs or images for layout analysis.
6. **API Implementation:** Set up a Flask RESTful API to provide layout analysis functionality for external applications and services.
7. **Integration & Testing:** Integrate the model into the web application and API, ensuring smooth functionality and conducting thorough testing.
8. **Deployment:** Deploy both the web application and RESTful API to suitable hosting environments for public access.
9. **Documentation & Maintenance:** Create comprehensive documentation and establish a maintenance plan for long-term sustainability.

Use case – Process Flow



Results

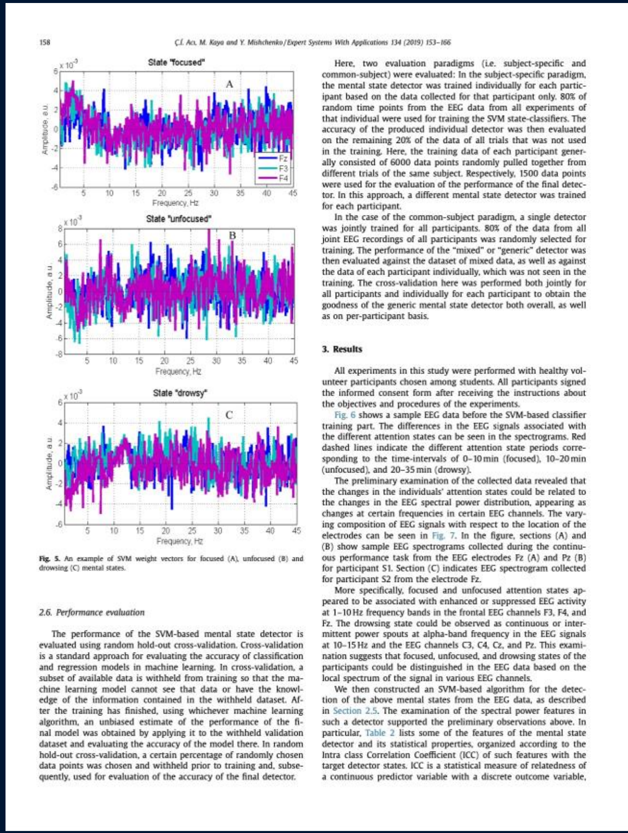


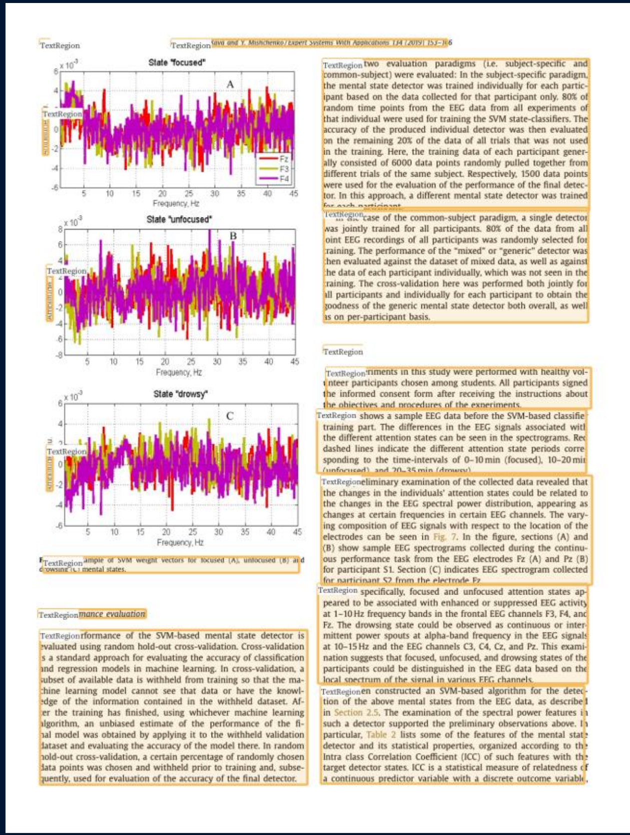


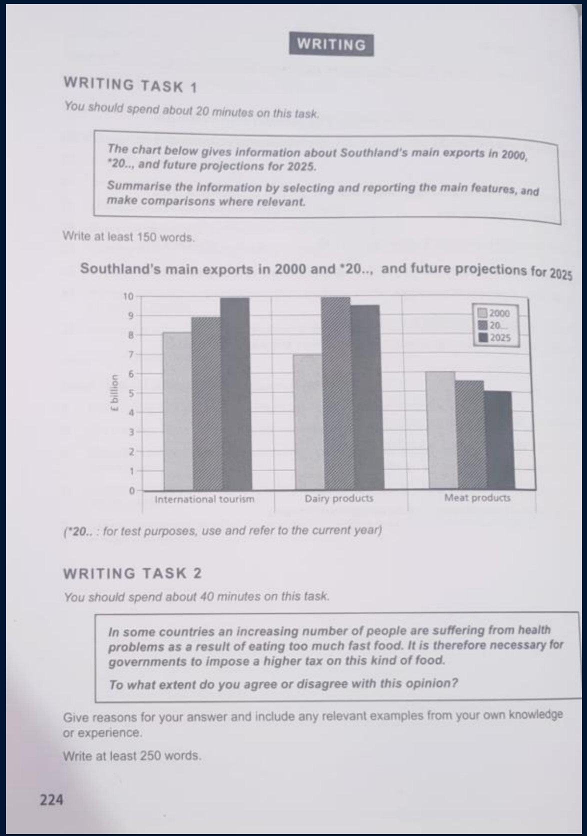
UI of Main Page

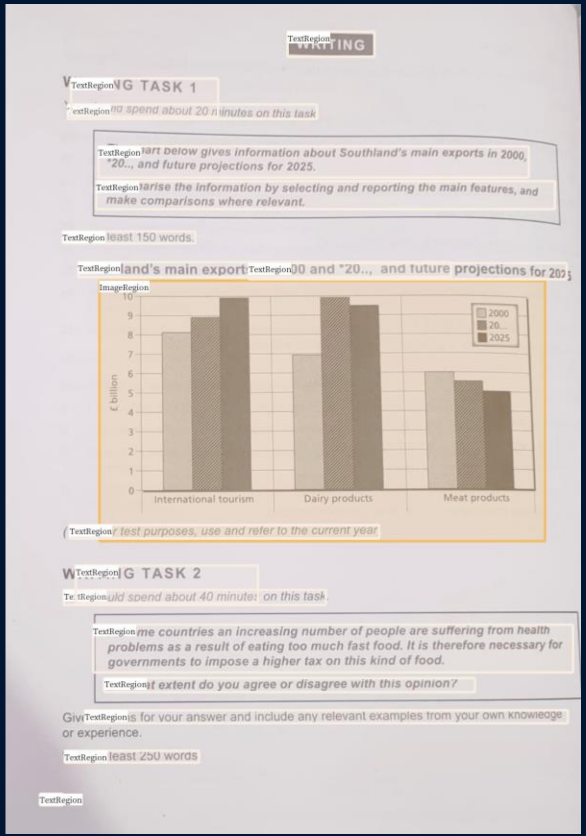
UI of Layout Preview

Results - Cont.





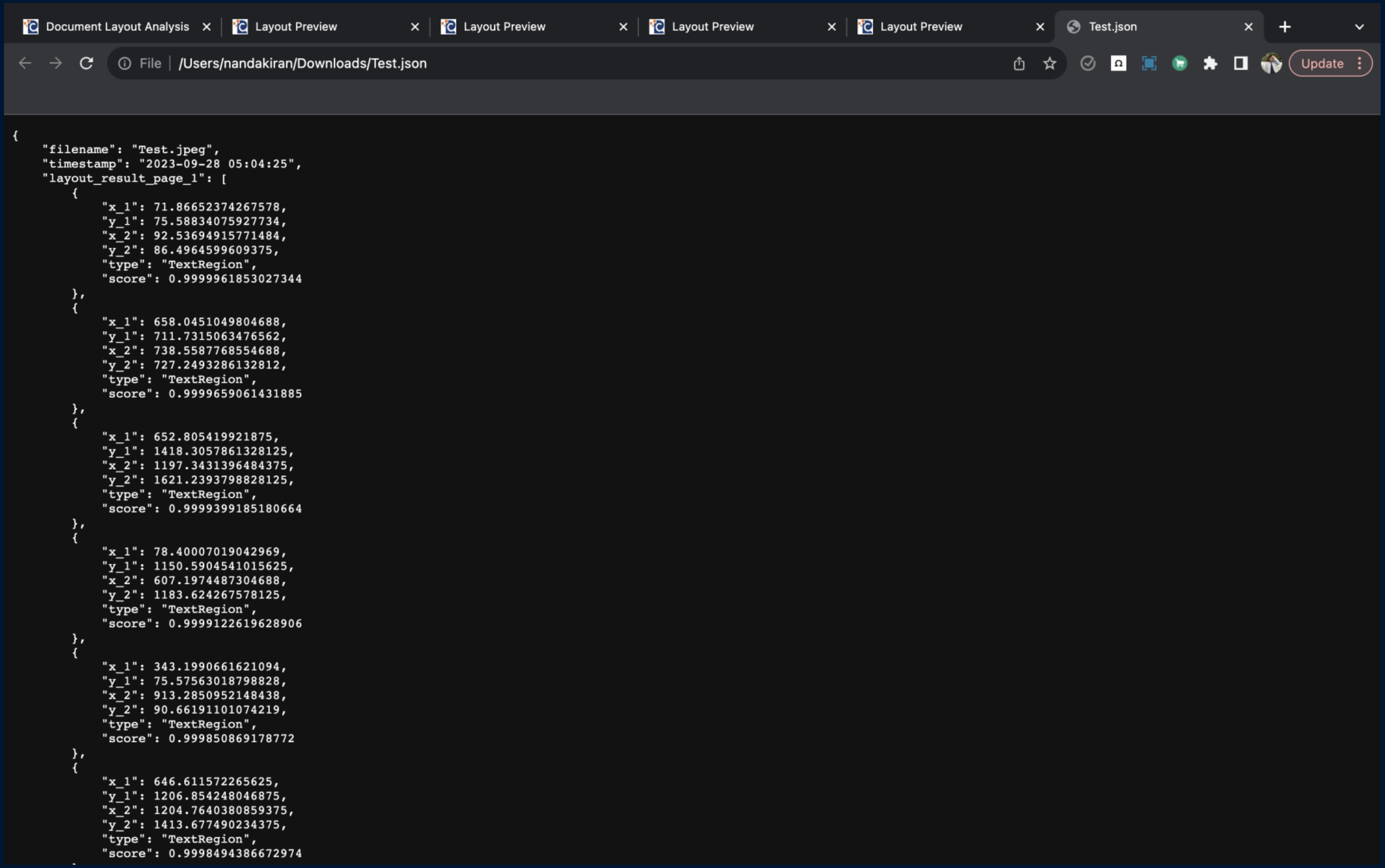




Native Image and Its Output

Scanned Image and Its output

Results- Cont.




```
{
    "filename": "Test.jpeg",
    "timestamp": "2023-09-28 05:04:25",
    "layout_result_page_1": [
        {
            "x_1": 71.86652374267578,
            "y_1": 75.58834075927734,
            "x_2": 92.53694915771484,
            "y_2": 86.4964599609375,
            "type": "TextRegion",
            "score": 0.9999961853027344
        },
        {
            "x_1": 658.0451049804688,
            "y_1": 711.7315063476562,
            "x_2": 738.5587768554688,
            "y_2": 727.2493286132812,
            "type": "TextRegion",
            "score": 0.9999659061431885
        },
        {
            "x_1": 652.805419921875,
            "y_1": 1418.3057861328125,
            "x_2": 1197.3431396484375,
            "y_2": 1621.2393798828125,
            "type": "TextRegion",
            "score": 0.9999399185180664
        },
        {
            "x_1": 78.40007019042969,
            "y_1": 1150.5904541015625,
            "x_2": 607.1974487304688,
            "y_2": 1183.624267578125,
            "type": "TextRegion",
            "score": 0.9999122619628906
        },
        {
            "x_1": 343.1990661621094,
            "y_1": 75.57563018798828,
            "x_2": 913.2850952148438,
            "y_2": 90.66191101074219,
            "type": "TextRegion",
            "score": 0.999850869178772
        },
        {
            "x_1": 646.611572265625,
            "y_1": 1206.854248046875,
            "x_2": 1204.7640380859375,
            "y_2": 1413.677490234375,
            "type": "TextRegion",
            "score": 0.9998494386672974
```


JSON Output Format

Benefits

- **Improved Document Understanding:** Layout analysis provides a comprehensive view of document structure, enhancing comprehension of content arrangement and relationships.
- **Efficient Information Extraction:** Accurate layout analysis streamlines data extraction, enabling faster and more precise retrieval of text, tables, and key information.
- **Optimized OCR Performance:** Precise text region detection improves OCR accuracy, reducing errors and enhancing the overall performance of optical character recognition.
- **User-Friendly Interactions:** Interactive web applications and APIs allow users to easily analyze document layouts, offering a seamless and intuitive user experience.
- **Versatility and Integration:** Utilizing diverse datasets, like Prima, ensures adaptability to different document types, while the RESTful API enables effortless integration with other applications and services.

Future Work

- Training the existing pre trained layout parser model with our custom dataset so to integrate smoothly in our pipeline.
- Training the model in our local Environment to have more access towards the code.

THANKS!

NandaKiran Velaga
nandakiran.velaga@criticalriver.com