## 📄 Legal Agreement Analyzer (LLM-Powered) – Product Requirements Document

**Author:** [Your Name]  
**Date:** July 11, 2025  
**Product Area:** OCI – Legal Ops/AI Platform  
**Feature Name:** LLM-powered Legal Agreement Analyzer

### 🧩 1. Overview

OCI customers frequently sign complex legal agreements that often differ subtly from prior versions. These differences may expose OCI to risk, contradict existing platform capabilities, or introduce new requirements.

This feature leverages **Large Language Models (LLMs)** to:

* Analyze a new legal agreement.
* Compare it against a repository of existing agreements.
* Cross-reference a feature-clause mapping stored in an **OCI NoSQL DB**.
* Incorporate manual feedback to refine output accuracy.
* Generate a report on the **differences**, **missing mappings**, and **confidence metrics (Model temperature, F1 score)**.

### 🎯 2. Goals & Objectives

| Goal | Description |
| --- | --- |
| 🧠 Use LLM for clause comparison | Automatically identify variations in clauses vs existing agreements. |
| 🧩 Clause-to-feature mapping | Use mapping DB to relate clauses to platform capabilities. |
| 📝 Generate comparative summary | Highlight differences and unrecognized clauses. |
| 🧪 Feedback Integration | Allow human reviewers to validate and correct output. |
| 📈 Evaluation metrics | Include LLM temperature setting, F1 score, and additional AI performance metrics. |

### 🧪 3. Functional Requirements

#### 3.1 Input

* **New Agreement Document** (uploaded or provided as plain text / PDF).
* **Existing Agreements Repository**: Accessible via OCI Object Storage or other internal system.
* **Clause-Feature Mapping DB**: OCI NoSQL DB table in the format:
* | clause\_id | clause\_text\_fragment | platform\_feature |
* | ———- | ———————- | —————– |

#### 3.2 Processing Logic (LLM-enabled)

* Segment the new agreement into logical clauses (via structural hints: numbering, headers, etc.).
* For each clause:
  1. Map clause to known platform feature using the NoSQL mapping DB.
  2. Compare the clause to semantically similar clauses from past agreements using vector embeddings or similarity search (e.g., OCI Data Science + AI Vector Search).
  3. Identify differences (insertions, removals, modifications) with a semantic threshold.
  4. Flag any clauses that do not map to an existing feature.
* Allow human reviewer to validate and correct clause mappings.
* Track and return model temperature, F1 score, and reviewer agreement percentage.

#### 3.3 Output

A structured report in JSON and optional PDF with:

{  
 "differences": [  
 {  
 "clause\_id": "7.3",  
 "mapped\_feature": "Data Sovereignty",  
 "difference\_type": "modified",  
 "new\_clause": "Customer data shall not leave the EU...",  
 "most\_similar\_existing\_clause": "Customer data may be replicated globally..."  
 }  
 ],  
 "unmapped\_clauses": [  
 {  
 "clause\_text": "OCI must provide weekly compliance audit logs.",  
 "reason": "Not found in clause-to-feature DB"  
 }  
 ],  
 "model\_temperature": 0.3,  
 "evaluation\_metrics": {  
 "f1\_score": 0.87,  
 "precision": 0.89,  
 "recall": 0.85,  
 "reviewer\_agreement": 0.92  
 }  
}

### 🧰 4. Non-Functional Requirements

| Requirement | Description |
| --- | --- |
| ⚖️ Accuracy | LLM outputs must meet F1 score threshold ≥ 0.85 |
| 🔒 Security | Agreements and mapping DB are confidential; must use OCI Vault, proper IAM policies |
| ⏱️ Latency | End-to-end analysis must complete < 2 minutes for <20-page documents |
| 📚 Auditability | Output must link each finding to evidence (agreement section, previous clause reference) |
| 🧪 Review Flow | Must support a reviewer UI for capturing feedback on output accuracy |

### 📊 5. Metrics of Success

#### Additional Suggested LLM Output Accuracy Metrics:

| Metric | Description | Learn More |
| --- | --- | --- |
| Calibration Error | Measures how well the model’s confidence scores match actual accuracy. | [Wikipedia: Calibration (statistics)](https://en.wikipedia.org/wiki/Calibration_(statistics)) |
| BLEU / ROUGE Scores | For evaluating text similarity between model output and expected text. | [BLEU](https://cloud.google.com/natural-language/automl/docs/evaluate" \l "bleu) / [ROUGE](https://en.wikipedia.org/wiki/ROUGE_(metric)) |
| Exact Match Ratio | Percentage of clauses where output exactly matches validated gold standard. | [SQuAD Evaluation](https://rajpurkar.github.io/SQuAD-explorer/) |
| Edit Distance (Levenshtein) | Measures how much text needs to be changed to match a reference clause. | [Levenshtein Distance](https://en.wikipedia.org/wiki/Levenshtein_distance) |
| Semantic Similarity Score | Quantifies closeness of meaning between predicted and reference clauses. | [Sentence Transformers](https://www.sbert.net/) |
| Token-level Confidence Average | Average probability assigned to generated tokens in output. | [OpenAI Cookbook: Logprobs](https://github.com/openai/openai-cookbook/blob/main/examples/How_to_interpret_token_probabilities.ipynb) |

| Metric | Goal |
| --- | --- |
| Avg F1 Score | ≥ 0.85 on test corpus |
| Reviewer Agreement | ≥ 90% agreement with model-generated output |
| Agreement difference coverage | ≥ 95% of unique clauses identified correctly |
| Clause-to-feature mapping recall | ≥ 90% of clauses matched to correct features |
| Latency | < 2 mins per agreement |
| Precision / Recall | ≥ 0.85 each |
| Feedback Turnaround Time | < 1 business day per document |

### 🛠️ 6. High-Level Scalable Architecture

#### Recommended LLMs for Legal Agreement Analysis:

| Model | Provider | Notes | OCI Usage Options |
| --- | --- | --- | --- |
| **OCI Generative AI (Cohere / Llama2 / Mixtral)** | Oracle | First-party integration, supports data residency & IAM controls | Available via OCI Data Science & OCI GenAI Service |
| **GPT-4 (via Azure OpenAI)** | Microsoft | Strong performance for long-context and legal text; secure enterprise access | Use via Oracle Interconnect with Azure or OCI AI service plugins |
| **Anthropic Claude 3** | Anthropic | High accuracy on structured documents, long-context support | Integrate via OCI Functions with external API Gateway |
| **Mistral Large / Mixtral** | Mistral AI | Open-weight options for customization, strong multilingual support | Deploy on OCI Supercluster or OCI Data Science (custom model) |
| **Google Gemini 1.5 Pro** | Google Cloud | Long context (1M+ tokens), suitable for large agreements | Integrate via OCI Functions or Container Instances |

#### Document Size Limitations:

* Most LLMs have token input limits that affect the maximum agreement length they can process in a single request:

| Model | Max Token Context | Approx. Page Count (at 400 words/page) |
| --- | --- | --- |
| GPT-4 Turbo | 128,000 tokens | ~300 pages |
| Claude 3 Opus | 200,000+ tokens | ~500 pages |
| Gemini 1.5 Pro | 1,000,000 tokens | ~2,500 pages |
| Mixtral / Llama 2 | 32,000 tokens | ~80 pages |

* **Workaround for large agreements**: Use chunking and parallel clause-level inference. Chunking must respect clause boundaries to preserve semantic integrity.
* **Recommendation**: Enforce a max file size (e.g., 50 pages / 20,000 words) per job in production unless a model with long-context capability is selected (e.g., Claude 3, Gemini).

#### Recommended LLMs for Legal Agreement Analysis:

| Model | Provider | Notes |
| --- | --- | --- |
| **OCI Generative AI (Cohere / Llama2 / Mixtral)** | Oracle | First-party integration, supports data residency & IAM controls |
| **GPT-4 (via Azure OpenAI)** | Microsoft | Strong performance for long-context and legal text; secure enterprise access |
| **Anthropic Claude 3** | Anthropic | High accuracy on structured documents, long-context support |
| **Mistral Large / Mixtral** | Mistral AI | Open-weight options for customization, strong multilingual support |
| **Google Gemini 1.5 Pro** | Google Cloud | Long context (1M+ tokens), suitable for large agreements |

User Uploads Agreement  
 ↓  
 OCI API Gateway  
 ↓  
OCI Function → Agreement Preprocessor (Clause Extractor)  
 ↓  
Vector Indexing (AI Vector DB) ↔ LLM Inference Engine ↔ OCI NoSQL Mapping DB  
 ↓  
Matching Engine (Diff Analyzer & Mapper)  
 ↓  
Review Feedback Loop (APEX UI / Internal Portal)  
 ↓  
 Output Generator (PDF/JSON/Report Store)

**Storage & Compute**:

##### Competitive Options in AWS:

* **S3** for agreement and report storage (equivalent to OCI Object Storage)
* **Amazon DynamoDB** for clause-feature mapping (equivalent to OCI NoSQL)
* **Amazon Bedrock** to access LLMs like Anthropic Claude, Cohere, and Meta’s Llama2
* **Amazon SageMaker** for custom model hosting and evaluation
* **Amazon Kendra** for semantic search and vector indexing
* **Amazon CloudWatch** and **X-Ray** for logging and monitoring

##### OCI Options:

* Object Storage: Agreement & output reports
* NoSQL: Feature-clause mappings
* Vector DB: Clause embeddings
* LLM Backend: OCI GenAI or external via secure APIs
* Logging: OCI Logging + Application Performance Monitoring

### 🚧 7.

**Storage & Compute**:

##### Competitive Options in AWS:

* **S3** for agreement and report storage (equivalent to OCI Object Storage)
* **Amazon DynamoDB** for clause-feature mapping (equivalent to OCI NoSQL)
* **Amazon Bedrock** to access LLMs like Anthropic Claude, Cohere, and Meta's Llama2
* **Amazon SageMaker** for custom model hosting and evaluation
* **Amazon Kendra** for semantic search and vector indexing
* **Amazon CloudWatch** and **X-Ray** for logging and monitoring

##### OCI Options:

* Object Storage: Agreement & output reports
* NoSQL: Feature-clause mappings
* Vector DB: Clause embeddings
* LLM Backend: OCI GenAI or external via secure APIs
* Logging: OCI Logging + Application Performance Monitoring

**Appendix: Security Considerations – Claude 3 Access to OCI NoSQL Mapping DB**

Great question. Since **Claude 3** is hosted externally by Anthropic and not inside OCI, you **cannot give it direct access** to the OCI NoSQL DB for security, architectural, and design reasons.

Instead, the best practice is to **mediate access** through your OCI backend. Here’s how:

## 🔐 How to Give Claude 3 Access to OCI NoSQL Mapping DB (Indirect Access Pattern)

### ✅ Recommended Pattern: *Pre-fetch & Inject via Prompt*

#### 🔍 Example 1: Direct Mapping Comparison

You are reviewing a legal agreement. Below are mappings of clauses to platform features:  
- "Data must reside in EU" → "Data Sovereignty"  
- "Logs must be retained for 6 years" → "Compliance Retention"  
- "Customer may request support 24/7" → "Premium Support"  
  
Now evaluate the following clause and determine:  
1. Does it map to any known feature?  
2. If yes, which one?  
3. If no, say 'UNMAPPED' and explain why.  
  
Clause: "OCI will retain logs for only 3 years."

#### 🧠 Example 2: Clause-to-Feature Suggestion (Unknown Input)

Here is a database of clause-feature mappings:  
- "The platform must support SOC2 compliance." → "Compliance Frameworks"  
- "Customer has access to audit logs." → "Auditability"  
- "Platform will maintain 99.9% uptime." → "High Availability SLA"  
  
Now evaluate the clause:  
"Customer may request audit logs up to 6 months post-termination."  
  
Questions:  
1. What is the most likely platform capability this clause relates to?  
2. Rate your confidence from 1 to 10.  
3. If this clause introduces a new requirement, summarize it.

* **Query OCI NoSQL from your backend** (e.g., an OCI Function or Service running in OKE, Container Instances, or App Dev stack).
* **Prepare relevant mapping data** (i.e., clause-feature mappings).
* **Inject that data into Claude’s prompt** or use it as in-context examples. For instance, you could include a set of previously mapped clauses at the beginning of the prompt and follow them with a new clause to evaluate. Example:

Context:  
- "Encryption must be FIPS 140-2 compliant" → "Encryption Standards"  
- "Customer data should not cross regional boundaries" → "Data Sovereignty"  
- "99.9% uptime SLA must be maintained" → "High Availability"  
  
Now analyze this clause:  
"OCI must provide audit trails for privileged access."   
  
Questions:  
1. Which feature (if any) does this clause map to?  
2. If not mappable, state 'UNMAPPED' and suggest a new capability name.

🧩 Architecture Flow

User Uploads Agreement  
 ↓  
Backend queries OCI NoSQL Mapping DB  
 ↓  
Extract relevant clause-feature mappings (top-N or recent)  
 ↓  
Send prompt + mappings to Claude 3 API (via HTTPS)  
 ↓  
Claude processes and returns comparison/inference

### 📄 Example Prompt Format

You are reviewing a legal agreement. Below are mappings of clauses to platform features:  
- "Data must reside in EU" → "Data Sovereignty"  
- "Logs must be retained for 6 years" → "Compliance Retention"  
- ...  
  
Now evaluate the following clause and determine:  
1. Does it map to any known feature?  
2. If yes, which one?  
3. If no, say 'UNMAPPED' and explain why.  
  
Clause: "Client data may be mirrored across global regions for latency."

🛡️ Security & Control Considerations

* Do **not expose OCI NoSQL endpoints** or secrets to Claude directly.
* Use **OCI IAM policies** to allow only your backend function or app to read from the DB.
* **Throttle** and **cache** frequently used mappings for reuse.
* Log all external data transfers (prompt + response) using **OCI Logging** for audit.

🔄 Alternative: Vector Embedding + Retrieval

* Embed mapping clauses using **OCI Data Science + Vector DB**.
* Retrieve similar clauses for a new one via **semantic search**, then send both to Claude for judgment