

Knight Foundation School of Computing and Information Sciences

Fall 2025 Capstone 1 - Computer Science



Students: Diego Martinez, Christian Cevallos, Jhonny Felix
Instructor/Faculty: Masoud Sadjadi, Florida International University

PROBLEM

Modern news consumption is heavily influenced by bias, framing, and selective language, making it difficult for readers to interpret information objectively. Articles may be factually correct but still present information in a biased way. Readers must also deal with manually evaluating source credibility, language tone, and framing of arguments

CURRENT SYSTEM

Most existing approaches fail to capture bias effectively:

- 1. Manual Interpretation:** Users must detect bias themselves and there is no structure or measurable feedback.
- 2. External Fact-Checking Websites/Tools:** Users leave the article, search fact-checking platforms, and hope the specific claim has already been reviewed. These services cannot keep up with the volume of new content and rarely provide real-time analysis.

- 3. Source Reputation:** Judge entire websites instead of individual articles.

SOLUTION

Veritas improves on these systems in several ways:

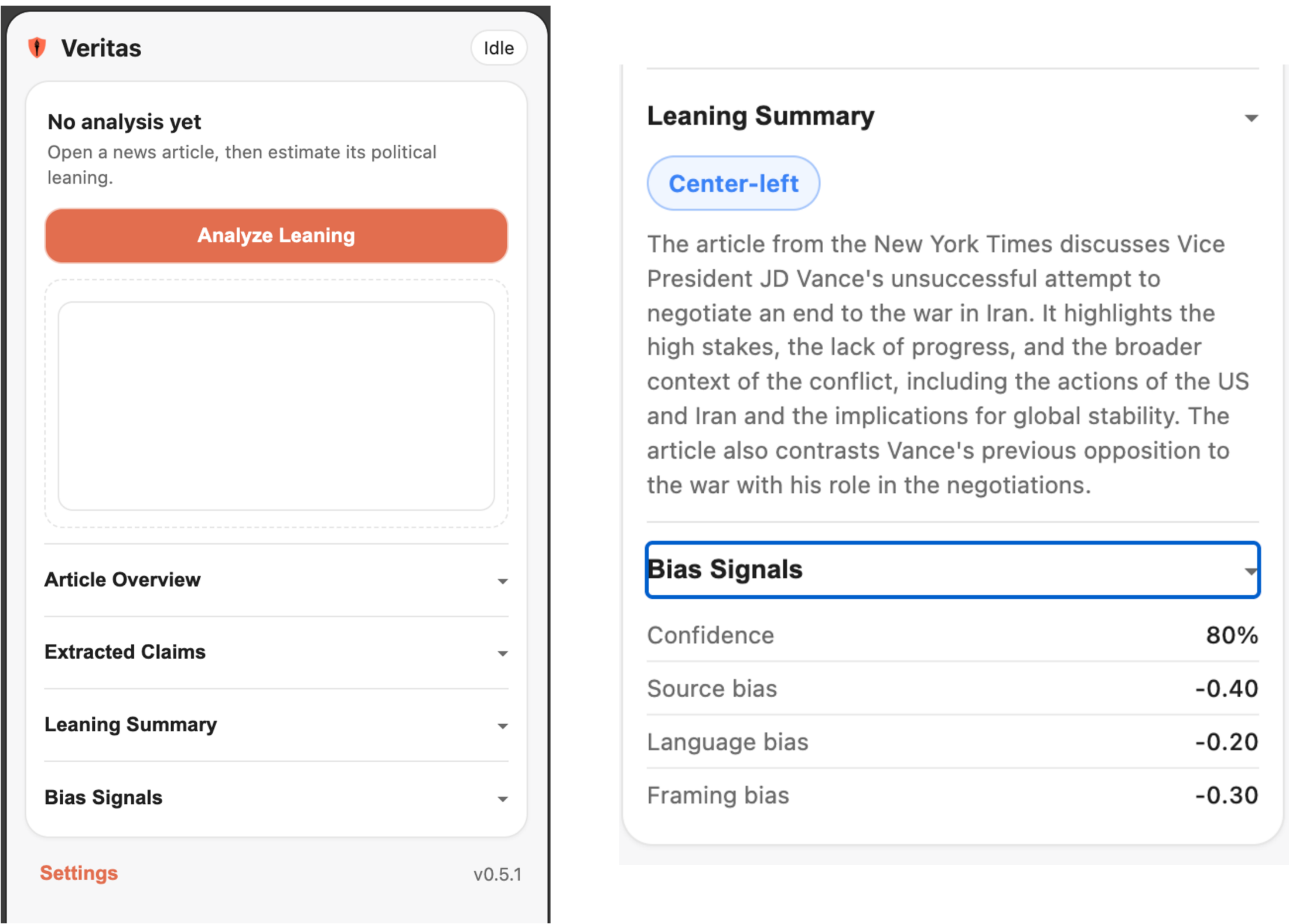
- 1. Signal-Based Bias AnalysisEvaluates:**
 - a. Source Bias
 - b. Language Bias
 - c. Framing Bias
- 2. Real-Time Analysis**
 - a. Runs directly in the browser via extension
 - b. One-click analysis of any article
- 3. Structured Claim Extraction**
 - a. Identifies important statements before analysis
 - b. Ensures high-signal content is evaluated
- 4. Explainable Results**
 - a. Returns:
 - i. Verdict (Left, Center, Right, etc.)
 - ii. Summary explanation
 - iii. Confidence score
 - iv. Bias breakdown

REQUIREMENTS

3.4 Software Requirements			
Category	Software / Tool	Version or Equivalent	Usage
Operating System	Windows, macOS, or Linux	Recent stable release	Development environment
Web Browser	Google Chrome	Current stable version	Running the extension popup
Browser Dev Tools	Chrome Developer Tools	Built in	Inspecting popup behavior and page extraction
Programming Language	Python	3.10 or later	Backend services and analysis pipeline
Backend Framework	FastAPI	Current stable version	/extract and /analyze API endpoints
NLP Library	spaCy	Current compatible version	Sentence segmentation and claim extraction
Date Parsing	python-dateutil	Current compatible version	Publication date normalization
Environment Loader	python-dotenv	Current compatible version	Loading API keys from .env
AI SDK	google-genai	Current compatible version	Gemini model communication
Data Validation	Pydantic	Current stable version	Request and response schemas
Extension Stack	JavaScript, HTML, CSS	ES6 or later	Popup UI and client-side extraction
Version Control	Git	Current stable version	Source control
Repository Hosting	GitHub or similar	Web account	Collaboration and documentation

- 1. Launch Extension**
User opens Veritas → popup initializes → ready to analyze article
- 2. Extract Article Content**
Extension captures URL, title, text, and metadata from active page
- 3. Extract Claims**
Backend cleans text → segments sentences → ranks high-signal claims
- 4. Display Overview**
UI shows source, publication date, and number of claims detected
- 5. Display Claims**
UI lists extracted claims in ranked order
- 6. Analyze Political Leaning**
Backend sends article + claims to Gemini → computes:
 - Verdict (Left → Right spectrum)
 - Confidence
 - Source, Language, Framing bias
- 7. Display Leaning**
UI shows verdict + short explanation
- 8. Display Bias Signals**
UI shows:
 - Confidence (%)
 - Source Bias
 - Language Bias
 - Framing Bias
- 9. Settings Management**
User toggles UI sections and theme preferences

IMPLEMENTATION



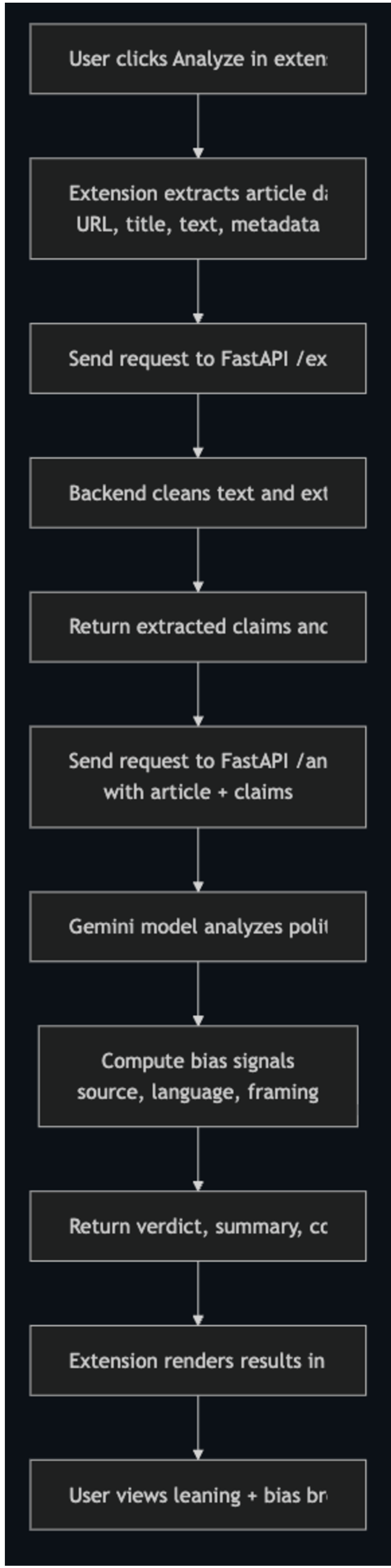
```
(.venv) diegomartinez@Mac Veritas % uvicorn main:app
INFO: Started server process [89906]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO: 127.0.0.1:62657 - "POST /extract HTTP/1.1" 200 OK
```

```
try:
    nlp = spacy.load("en_core_web_sm")
except Exception:
    nlp = None
```

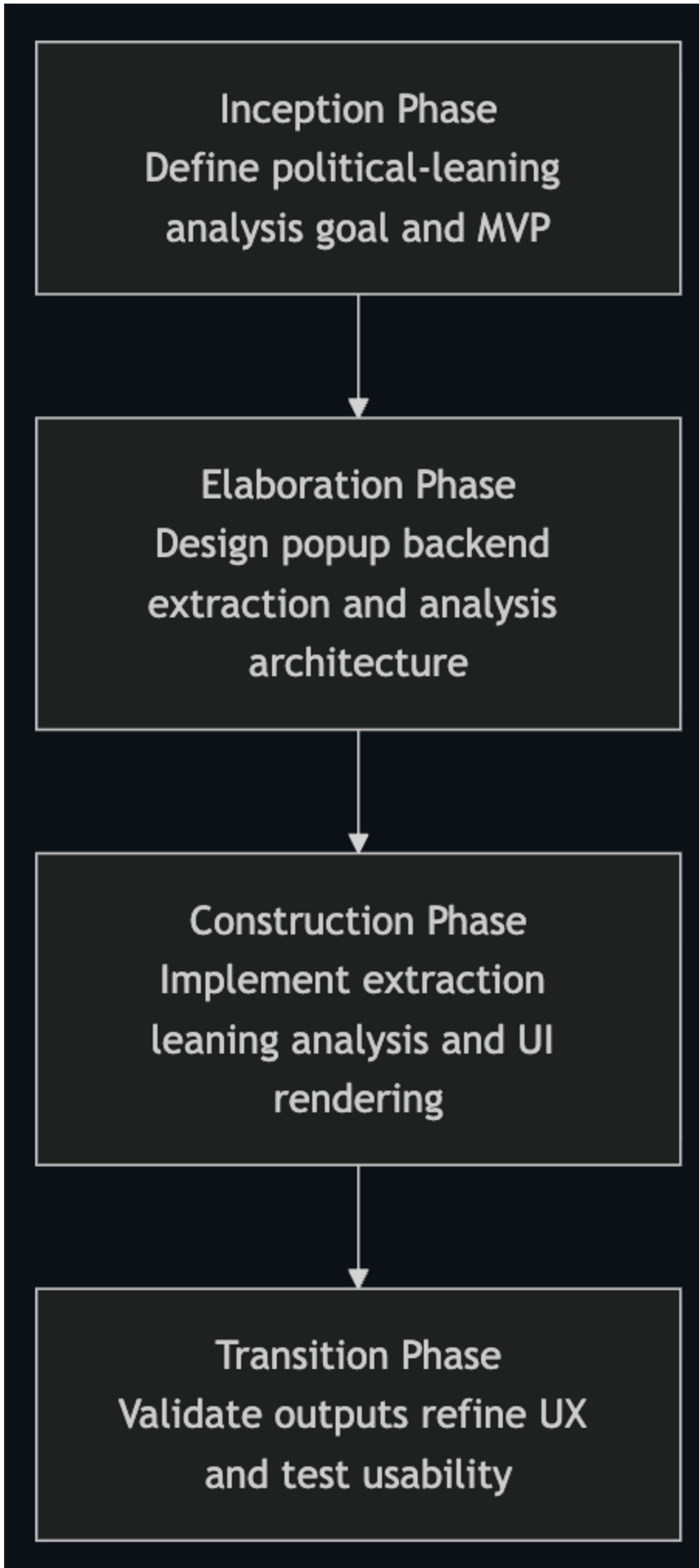
```
RAW GEMINI OUTPUT: {"verdict": "Center-left", "summary": "An article from the New York Times about Vice President Vance's unsuccessful attempt to negotiate an end to the war in Iran. It highlights the high stakes, the lack of progress, and the broader context of the conflict, including the actions of the US and Iran and the implications for global stability. The article also contrasts Vance's previous opposition to the war with his role in the negotiations.", "confidence": 0.7, "source_bias": -0.4, "language_bias": -0.2, "framing_bias": -0.3}
```

SYSTEM DESIGN

Workflow



Unified Software Development Process



Verification

- Backend validation using Pydantic schemas and structured request/response testing
- Extraction testing to ensure accurate article text, metadata, and claim ranking
- API endpoint verification (/extract, /analyze) with consistent JSON responses
- AI output validation for correct format, value ranges, and leaning classification
- UI checks confirming proper rendering of claims, verdict, and bias signals

References

[Github Repository](#)

[Google Gemini Docs](#)

[Chrome Devtool Documentation](#)

[FastAPI Documentation](#)

SUMMARY

This Capstone project implements an automated political bias and credibility analysis system designed to evaluate online articles in real time. Veritas functions as a browser extension that extracts the visible text of an article, identifies key claims, and forwards structured article data to an analysis pipeline. The system uses a Chrome extension for user-side interaction, a FastAPI backend for claim extraction and data processing, and a Gemini-based analysis component that estimates political leaning and produces a verdict, confidence score, bias signals, and a short explanatory summary. Development follows the Unified Software Development Process, progressing through iterative and incremental cycles that emphasize requirements analysis, architectural design, implementation, and continuous evaluation. This structured approach supports rapid validation of the minimum viable product, clear separation of responsibilities across components, and a scalable foundation for future enhancements.