Mechanistic Coherence Module v2.1 — MVP Engineering Task

📌 Overview

The goal of this task is to develop a **Minimum Viable Product (MVP)** of the **Mechanistic Coherence Module**, a computational pipeline that evaluates the **biological plausibility and internal consistency** of a drug candidate based on molecular, pathway, and clinical evidence. This system will trace the biological narrative from **genetic variant → protein function → pathway disruption → tissue phenotype → drug response** using real models, and generate a **scored, interpretable output** that supports downstream drug development decisions.

🎯 Primary Goals

- Integrate **biologically grounded models** into a modular, testable pipeline.
- Generate a final JSON report scoring the coherence of a variant/drug's mechanism of action.
- Provide enough traceability, fallback logic, and logging to support future expansion.

✅ Core Models to Integrate (MVP Scope)

Category	Required Model/Tool
Genotype → Phenotype	**AlphaMissense** (via HuggingFace)
Sequence Manipulation	**BioPython**
Known Variant Annotation	**ClinVar API**
Pathway Impact & Simulation	**GSEApy** (MSigDB or Reactome gene sets)
Protein Structure & Function	**ProteinMPNN**
Clinical & Biomarker Relevance	**GTEx API** + basic translator module

All other advanced or optional models are deferred until the MVP is complete.

📂 Key Modules and Expected Behavior

Module	Description	Acceptance Test
`enhanced_input_processor.py`	Validates and enriches inputs; parses variants; applies to sequences	✅ Returns validated dict given `ex_variants.json`, logs QC warnings
`variant_impact_analyzer.py`	Runs AlphaMissense + ClinVar to assess variant effect	✅ Returns `{ pathogenicity_score, confidence, known_annotations[] }`
`pathway_dynamics_analyzer.py`	Performs real gene set enrichment using GSEApy	✅ Returns `{ pathway_enrichment: { pathway_name: score, ... } }`
`structure_function_integrator.py`	Uses ProteinMPNN to analyze structural plausibility and druggability	✅ Returns `{ binding_site_score, model_used }`
`clinical_translator.py`	Maps tissue-specific gene effects to clinical relevance and biomarker use	✅ Returns `{ biomarker_potential, therapeutic_area }`

`coherence_analyzer.py`	Aggregates evidence and checks biological consistency across layers	✅ Returns `{ cross_scale_consistency : float }`
`intelligent_coherence_scorer.py`	Combines all scores into overall score and coherence category	✅ Returns `{ overall_score: float, coherence_category: str }`
`output_formatter.py`	Formats the final JSON report with metadata, trace, and scores	✅ Output matches schema below
`enhanced_pipeline_orchestrator.py`	Main orchestrator that runs the full pipeline end-to-end	✅ Given example input, returns final output in <5 min

---

🧪 Final Output Schema (Simplified)


```
{
  "compound_analysis": {
    "mechanistic_coherence": {
      "overall_score": 0.84,
      "coherence_category": "HIGH_COHERENCE",
      "category_scores": {
        "genotype_pathogenicity": 0.76,
        "structure_function": 0.81,
        "pathway_enrichment": 0.89,
        "clinical_translatability": 0.78
```

```
      },
      "causal_trace": [
        {
          "step": 1,
          "level": "molecular",
          "description": "TP53_R175H disrupts DNA binding domain"
        },
        {
          "step": 2,
          "level": "pathway",
          "description": "Downstream p53 signaling pathway
suppression"
        }
      ]
    }
  }
}
```

🧪 Benchmark Cases (Must Pass in CI)

AlphaMissense + ClinVar

Variant	Expected `pathogenicity_score`	ClinVar Annotation
TP53_R175H	0.85–0.95	Pathogenic
CFTR_ΔF508	0.80–0.90	Pathogenic
BRCA1_185delAG	0.90–1.00	Pathogenic

GSEApy Pathway Enrichment

Pathway	Minimum Expected Score
p53_signaling	> 0.70
DNA_repair	> 0.65
Apoptosis	> 0.60

Use `pytest` to enforce score ranges and expected keys.

🔧 Tech Implementation Guidelines

General Design Principles

- **Modularity:** Each module should be a callable Python class or function with clear I/O.
- **Fallbacks:** If a model fails (e.g. ClinVar timeout), return defaults and log it in `logs/`
- **Asynchronous I/O:** Use `aiohttp` or similar for API-heavy calls (ClinVar, GTEx).
- **Caching:** Store ClinVar and GTEx results locally (`cache/`) to avoid repeat calls.

Sequence of Execution

1. Parse + validate variant and context input (`enhanced_input_processor.py`)
2. Predict pathogenicity via AlphaMissense + ClinVar lookup
3. Apply variant to reference sequence (BioPython)
4. Run GSEApy on gene list
5. Run ProteinMPNN on protein `.pdb` structure
6. Run tissue-specific biomarker relevance check (GTEx)
7. Aggregate and score outputs
8. Format output JSON with trace

🧱 Environment Spec (Conda)

Save this as `mech-coherence-env.yml` and install via:

bash

```
conda env create -f mech-coherence-env.yml
conda activate mech-coherence
```

```
name: mech-coherence
channels:
  - conda-forge
dependencies:
  - python=3.9
  - pip
  - gseapy=1.0.6
  - transformers=4.30.0
  - pytorch=2.0.0
  - biopython=1.81
  - pandas=2.0.0
  - numpy=1.24.0
  - requests=2.31.0
  - aiohttp=3.8.0
  - pytest=7.1.0
```

📁 Suggested Project Structure

```
project_root/
├── modules/
|   ├── enhanced_input_processor.py
|   ├── variant_impact_analyzer.py
|   ├── pathway_dynamics_analyzer.py
|   ├── structure_function_integrator.py
|   ├── clinical_translator.py
|   ├── coherence_analyzer.py
```

```
|   └── intelligent_coherence_scorer.py
├── orchestrator/
|   └── enhanced_pipeline_orchestrator.py
├── output/
├── examples/
|   ├── ex_variants.json
|   ├── gene_list.csv
|   └── protein_42.pdb
├── tests/
|   └── test_*.py
├── cache/
├── logs/
├── mech-coherence-env.yml
└── README.md
```

✅ Completion Checklist

- Each module returns structured, valid output
- All `pytest` tests pass, including benchmarks
- End-to-end JSON is generated in < 5 minutes
- Environment installs cleanly from `mech-coherence-env.yml`
- Fallbacks and error logs are properly handled

🚀 Next Steps After MVP (Future Phases)

Once the MVP is complete and stable, we will explore:

- Enformer integration for expression prediction
- AlphaFold DB integration for structure retrieval
- Cross-variant interactions (epistasis)
- Real-time data viewer with Streamlit
- Literature-based causal inference layer