**Top Priority (This Week) – Get the Core Working**

**1. Rebuild & Document Your Codebase**

* ~~Reconstruct your scripts (clean\_dwi.py, build\_connectomes.py) from backups or ChatGPT memory (already done above).~~
* ~~Add back all~~ **~~docstrings~~**~~,~~ **~~comments~~**~~, and structure it like a Python package (e.g., src/, scripts/, data/).~~
* ~~Initialize a~~ **~~Git repository~~**~~:~~
* ~~git init~~
* ~~git commit -m "Initial commit after system restore"~~

**2. Set Up a Robust Backup (Avoid OneDrive)**

* ~~Use~~ [~~GitHub~~](https://github.com/) ~~or~~ [~~GitLab~~](https://gitlab.com/) ~~as a remote backup:~~
* ~~git remote add origin <your\_repo\_url>~~
* ~~git push -u origin main~~

**R-GIANT Development Roadmap – Phase 2**

**1. OASIS Data Preparation**

**1.1 Decide on data structure**

* **Determine whether to work directly from BIDS format or unpack into a simplified structure**
* **Evaluate trade-offs between flexibility and complexity**

**1.2 Download necessary OASIS data**

* **Use wget, curl, or a browser-based approach to pull data**
* **Only download DWI, T1, and relevant metadata (optionally PET later)**
* **Organize into a consistent local directory for testing**

**1.3 Build a script for batch formatting**

* **Automate identification and structuring of participant/session pairs**
* **Build wrapper to call preprocessing functions (e.g., run\_cleaning\_pipeline)**
* **Design for compatibility with future HPC scaling**

**2. Data Preprocessing and Feature Extraction**

**2.1 Test pipeline on selected participants**

* **Run cleaning and connectome generation on 1–3 participants**
* **Log and verify each intermediate output (e.g., cleaned DWI, connectome matrix, node features)**

**2.2 Extract node features**

* **Identify all relevant FreeSurfer stats and PET features**
* **Design a way to associate features with brain regions/nodes**
* **Standardize how features are saved (e.g., per participant .csv or .pt file)**

**2.3 Handle different node types**

* **Explore options for modeling graphs with heterogeneous node features**
* **Decide whether to use node masks, separate node classes, or type embeddings in the model**

**3. HPC Scaling**

**3.1 Move pipeline to HPC**

* **Transfer code and partial dataset**
* **Write SLURM job array or other parallel execution strategy**
* **Handle logs, errors, and outputs per participant/session**

**3.2 Clean the full dataset**

* **Run batch pipeline across all valid participant/session pairs**
* **Collect all outputs in a structured dataset for training**

**4. Model Training Phase**

**4.1 Finalize dataset format for GNN input**

* **Combine connectome matrices, node features, and labels**
* **Align temporal sessions for longitudinal input (if needed)**

**4.2 Begin training R-GIANT model**

* **Start with baseline forward pass**
* **Verify input handling and training stability**
* **Gradually scale to full longitudinal training with prediction objectives**

**🧠 Model Development Roadmap**

**🔹 Phase 1: Baseline GNN Classifier**

* Task: Diagnosis classification based on **single connectome**.
* Model:
* GNN → latent\_vector → MLP → diagnosis
* Input: N x N matrix + optional N x D node features.
* Output: Binary or multi-class label.

**🔹 Phase 2: Longitudinal Sequence**

* Aggregate sessions per subject: latent\_vector\_t1, t2, ..., tn
* Feed into Transformer or RNN (LSTM).
* Interpolate or mask missing sessions.
* Predict next diagnosis or CDR score.