Structured project plan

1. Problem Definition and Business Framing

Define the central business question: *"Can routine patient data be used to triage individuals at high risk of stroke for early screening?"*

Clarify this as a **risk triage** task, not medical prevention.

Business KPIs: recall at a fixed precision threshold, AUROC, and false-negative cost framing.

---

2. Data Description and Versioning

Dataset: Healthcare Stroke dataset (~5,000 rows, 11 columns).

Register raw data in Azure Data Lake Storage Gen2 under `/raw/healthcare-stroke/`.

Version data via Azure ML `Dataset` assets (`stroke_raw@v1`, `stroke_clean@v1`, etc.).

Track every transformation in Azure ML + MLflow.

Enable scans in **Microsoft Purview** for both data and model lineage.

---

3. Exploratory Data Analysis (EDA)

Perform descriptive and inferential EDA separately from feature engineering.

Generate summary statistics, correlations, imbalance checks, and missingness reports.

Export an **HTML EDA report** (using `ydata-profiling`) and a Markdown executive summary.

Register both as artifacts and log to MLflow under `eda@v1`.

Add optional **differential privacy** noise for DP-protected aggregates (ε = 1.0).

---

4. Domain Feature Engineering

Engineer medical priors: `age_over_60`, `age_over_80`, `glucose_above_150`, `bmi_category`, `is_overweight`, etc.

Create feature sets and version them:

- `stroke_features@v1`: imputed BMI
- `stroke_features@v2`: rows with missing BMI dropped

Store features in **Azure ML Feature Store**, backed by ADLS.

Include a YAML artifact that documents column-level derivations for lineage capture.

---

5. Data Lineage and Governance

Implement **end-to-end lineage** through Azure ML and **Purview** integration.

Each pipeline step declares formal inputs/outputs:

`raw → clean → feature_set → train/test → model → endpoint → inference → monitoring`.

Register these as Azure ML assets so lineage graphs populate automatically.

Enable Purview scans on ADLS, AML Datasets, Models, and Endpoints.

Use AML's lineage view and Purview's data map for screenshots.

Maintain a YAML mapping of asset → version → owner → Purview GUID.

Tag sensitive columns (`HealthData`, `Confidential`) and store ε/δ for DP releases.

---

6. Pipeline Design (Azure ML Pipelines)

Define modular pipeline steps:

1. Data Ingestion →
2. EDA Generation →
3. Feature Engineering →
4. Split →
5. Train →
6. Evaluate →
7. Register Model →
8. Deploy →
9. Monitor

Each step logs metrics and outputs to MLflow and registers as AML assets.

Pipeline orchestrated with **Azure ML Pipelines YAML**, tracked in Git.

Store all environment configurations (`conda.yaml`) for reproducibility.

---

7. Experiments and Model Training

Train using **RandomForest**, **LightGBM**, and **XGBoost** on different feature versions.

Experiment matrix:

- Dataset variants (BMI-imputed vs BMI-dropped)
- Feature versions (v1, v2)
- Models (RF, XGB, LGBM)

  Use **CodeCarbon** to measure CO_2 emissions for each run.

  Log to MLflow with tags: `dataset_version`, `feature_version`, `model_type`, `carbon_emission`.

  After manual experiments, run **Azure AutoML** to compare best-performing model families.

  Record leaderboard in MLflow and Azure ML.

---

8. Model Evaluation

Evaluate on held-out test set using:

- F1-score, ROC-AUC, PR-AUC
- Calibration and confusion metrics
- Carbon efficiency (CO_2e vs AUC trade-off)

  Export performance plots and log them to MLflow.

  Visualize feature importances and SHAP summary for transparency.

---

9. Deployment

Deploy best model using **Azure ML Managed Online Endpoint**.

Include `score.py` with pydantic schema validation and example payload.

Enable **Application Insights** logging for inference telemetry.

Pin environment versions for reproducibility.

Tag deployment metadata for lineage tracing (`model_version`, `endpoint_name`, `build_commit`).

---

10. Model Monitoring

Implement two layers:

- **Batch drift detection**: scheduled AML job comparing prediction and feature distributions (PSI, JS divergence).
- **Live monitoring**: enable Azure ML Data Drift Monitor for age, glucose, and smoking status.

  Store drift metrics in MLflow and Purview.

  Visualize one dashboard snapshot in presentation.

---

11. Test Set Skew Experiment

Create a "Clinic B" simulation by skewing `age` and `smoking_status` distributions or adding unseen similar records.

Deploy model inference on both original and skewed datasets.

Compare metrics and drift metrics to demonstrate monitoring and robustness.

Log both runs and lineage (`test_original@v1`, `test_skewed@v1`) in AML + Purview.

---

12. Differential Privacy and Governance

Apply **OpenDP/SmartNoise** for releasing EDA summaries with DP noise.

Declare DP parameters (ε, δ) and include results as metadata in Purview.

Explain in slides that privacy is demonstrated on aggregate reports, not model weights.

---

13. Carbon and Efficiency Metrics

Use **CodeCarbon** to track emissions for every training run.

Plot emissions vs model quality metrics.

Add one slide showing “accuracy vs energy cost” trade-off.

---

14. Inference Speed Comparison

Convert tree models (LightGBM/XGBoost) to **ONNX format**.

Benchmark inference latency for batch sizes {1, 32, 256}.

Present latency-quality trade-offs to show production readiness.

---

15. Data Lineage Visualization (for Presentation)

Show the Purview graph:

```
ADLS/raw → dataset:clean@v1 → feature-set:v2 → dataset:train@v1
→ model:rf@v3 → endpoint:stroke-risk-blue → inference@2025-11-02 → mon
itoring@daily
```

Add column-level lineage snippet showing engineered features and their origins.

Show AML “Lineage” screenshot for a registered model asset linking all upstream datasets.

Include one table slide: Asset | Version | Owner | Type | Purview GUID.

---

16. Final Deliverables

1. Azure ML workspace assets (datasets, models, feature sets, endpoints)
2. Git repository with source code, pipeline YAMLs, conda files, and README with `az ml` CLI commands
3. MLflow experiment screenshots
4. Purview lineage screenshots and lineage YAML
5. PPT presentation including:
    - Business question
    - EDA summary
    - Feature engineering

- Pipeline diagram
- Experiment results + AutoML comparison
- Monitoring dashboard
- Data lineage visual
- Energy/latency comparison
- Team contribution slide

6. Demo video showing inference and monitoring under normal and skewed test data.

   Opening credits: *“A Film by S-2D Studio and Team.”* , followed by the contribution from each team members.
