Part 1: Short Answer Questions (30 points)

1. Problem Definition (6 points)

Hypothetical AI Problem:
"Predicting Student Dropout Rates in Online Courses"

Objectives:

1. Identify students at risk of dropping out based on engagement and performance data.
2. Enable timely intervention by academic advisors.
3. Improve course completion rates and learner retention.

Stakeholders:

- University academic advisors
- Online course platform administrators

Key Performance Indicator (KPI):

- **Recall rate** of the dropout prediction model (to minimize false negatives).

---

2. Data Collection & Preprocessing (8 points)

Two Data Sources:

1. Learning Management System (LMS) logs (login frequency, quiz submissions).
2. Student demographic and academic records.

Potential Bias:

- Underrepresentation of non-traditional learners (e.g., working adults, first-generation students) may skew predictions and limit model generalizability.

Three Preprocessing Steps:

1. **Handling missing data** – Impute missing quiz scores with average performance.
2. **Normalization** – Scale login frequency and session duration for consistent feature ranges.
3. **Encoding categorical variables** – Convert enrollment type or program into numeric format using one-hot encoding.

---

3. Model Development (8 points)

Chosen Model:

- **Random Forest**
  Justification: Robust to overfitting, handles feature importance, works well with both numerical and categorical data.

Data Splitting Strategy:

- 70% Training set
- 15% Validation set
- 15% Test set
  (Split using stratified sampling to maintain dropout ratio)

Two Hyperparameters to Tune:

1. `n_estimators` – Number of trees in the forest. Affects performance and speed.
2. `max_depth` – Controls tree growth to avoid overfitting.

---

4. Evaluation & Deployment (8 points)

Two Evaluation Metrics:

- **Recall:** Prioritizes catching at-risk students (true positives).
- **F1-Score:** Balances precision and recall for overall performance.

Concept Drift:

- Change in the statistical properties of input data over time (e.g., new learning behaviors post-COVID).
  Monitoring Strategy: Retrain the model periodically with updated logs, monitor performance drop via dashboard.

One Deployment Challenge:

- **Scalability:** Serving real-time predictions to thousands of concurrent students may require a load-balanced inference API.

🔹 Part 2: Case Study Application (40 points)

Scenario: Predicting 30-day Patient Readmission Risk

Problem Scope (5 points)

Problem:
Predict whether a discharged patient will be readmitted within 30 days.

Objectives:

1. Reduce unnecessary hospital readmissions.
2. Improve patient outcomes through early intervention.

Stakeholders:

- Hospital administrators
- Clinicians (doctors, nurses)

Data Strategy (10 points)

Proposed Data Sources:

- Electronic Health Records (EHRs): diagnoses, treatment history, vitals.
- Demographic data: age, gender, socioeconomic status.

Two Ethical Concerns:

1. **Patient privacy and consent** when handling sensitive EHR data.
2. **Algorithmic bias** leading to unfair treatment across demographic groups.

Preprocessing Pipeline:

1. **Data Cleaning** – Remove erroneous entries (e.g., age > 120).
2. **Feature Engineering** – Generate "time since last admission", count of chronic conditions.
3. **Normalization & Encoding** – Normalize lab results, encode diagnosis codes.

Model Development (10 points)

Selected Model:

- **Gradient Boosted Trees (e.g., XGBoost)**
  Justification: Excellent for tabular healthcare data, interpretable with SHAP, high predictive performance.

Confusion Matrix (Hypothetical Example):

	Predicted: Readmit	**Predicted: No Readmit**
Actual: Readmit	120 (TP)	30 (FN)
Actual: No Readmit	40 (FP)	210 (TN)

Metrics:

- **Precision** = 120 / (120 + 40) = 0.75
- **Recall** = 120 / (120 + 30) = 0.80

Deployment (10 points)

Integration Steps:

1. Develop API endpoint using Flask.
2. Secure API with role-based access.
3. Embed model into hospital dashboard for real-time risk flagging.

Compliance (HIPAA):

- Ensure **data encryption**, **audit trails**, and **access logs**.
- Limit model access to authorized healthcare personnel only.

Optimization (5 points)

Overfitting Solution:

- Use **cross-validation** and implement **regularization** (e.g., `lambda` parameter in XGBoost) to penalize overly complex models.

🔹 Part 3: Critical Thinking (20 points)

Ethics & Bias (10 points)

Effect of Bias:
If the training data underrepresents low-income patients, the model might under-predict their readmission risk, leading to reduced care and worse outcomes.

Mitigation Strategy:

- Use **IBM AI Fairness 360** to audit fairness metrics.
- Rebalance dataset or apply fairness-aware reweighting techniques during training.

Trade-offs (10 points)

Interpretability vs. Accuracy:

- A highly accurate deep learning model might be a “black box,” which is problematic in clinical decision-making.
- A simpler, interpretable model (e.g., logistic regression) builds clinician trust, but may sacrifice performance.

Impact of Limited Resources:

- Resource constraints may favor **lightweight models** (e.g., Decision Trees) over compute-heavy ones like deep neural networks.
- Also impacts real-time inference capability and data storage needs.

🔹 Part 4: Reflection & Workflow Diagram (10 points)

Reflection (5 points)

Most Challenging Part:
Designing an ethically responsible data pipeline—ensuring fairness, privacy, and regulatory compliance while achieving high performance.

Improvement Suggestions:

- Use a more diverse dataset.
- Allocate more time for hyperparameter tuning and fairness auditing.

Workflow Diagram (5 points)

graph TD

A[Problem Definition] --> B[Data Collection]

B --> C[Preprocessing]

C --> D[Model Development]

D --> E[Evaluation]

E --> F[Deployment]

F --> G[Monitoring & Feedback]