1. Problem Definition (6 points)

Hypothetical AI Problem:
Predicting student dropout risk in university students.

Objectives:

1. Identify students at high risk of dropping out early in the semester.
2. Provide insights to academic advisors for timely intervention.
3. Improve overall student retention and graduation rates.

Stakeholders:

- University administration
- Students

Key Performance Indicator (KPI):

- **Recall (True Positive Rate)** for correctly identifying students at risk of dropping out.

2. Data Collection & Preprocessing (8 points)

Two Data Sources:

1. **Academic records** – attendance, GPA, course completion rates.
2. **Learning management system (LMS) activity logs** – logins, assignment submissions, forum participation.

One Potential Bias:

- Students with limited internet access or disabilities may appear "inactive" in LMS logs, leading to biased dropout predictions.

Three Preprocessing Steps:

1. **Handle missing values** – fill or drop incomplete academic records.
2. **Normalize features** – scale GPA, participation scores to uniform range.
3. **Encode categorical variables** – e.g., course type, major, using one-hot encoding.

3. Model Development (8 points)

Model Choice:

- **Random Forest Classifier**
  Justification: Robust to overfitting, handles mixed data types, interpretable, and good for small to medium-sized datasets.

Data Split Strategy:

- 70% Training
- 15% Validation
- 15% Test

Two Hyperparameters to Tune:

1. `n_estimators` – Number of trees in the forest (affects accuracy and runtime).
2. `max_depth` – Limits the depth of trees to control overfitting.

4. Evaluation & Deployment (8 points)

Two Evaluation Metrics:

1. **Recall** – Ensures we catch most students at risk.
2. **F1 Score** – Balances recall and precision, especially useful if the dropout class is imbalanced.

What is Concept Drift?

- When the statistical properties of input data change over time, causing model performance to degrade.
  How to Monitor It Post-Deployment:

- Track prediction accuracy over time and set alerts for sudden drops. Use periodic re-training with new data.

One Technical Deployment Challenge:

- **Integration with multiple data sources** – synchronizing student records, LMS data, and external systems without data lag or conflicts.

Part 2: Case Study Application (40 points)

Scenario: A hospital wants an AI system to predict patient readmission risk within 30 days of discharge.

Problem Scope (5 points)

Problem:
Develop an AI model that predicts whether a patient will be readmitted to the hospital within 30 days of discharge.

Objectives:

1. Reduce avoidable patient readmissions.
2. Improve resource allocation and discharge planning.
3. Enhance patient care through proactive follow-up.

Stakeholders:

- Hospital administrators
- Doctors and care teams

Data Strategy (10 points)

Proposed Data Sources:

- **Electronic Health Records (EHRs):** Diagnosis, treatment history, medications, vital signs.
- **Demographic Information:** Age, gender, location, socio-economic status.

Two Ethical Concerns:

1. **Patient Privacy:** Sensitive health data must be protected under regulations like HIPAA.
2. **Algorithmic Bias:** Certain groups (e.g., older patients or minorities) may be misrepresented, leading to unfair treatment decisions.

Preprocessing Pipeline & Feature Engineering:

1. **Handle Missing Values:** Fill missing lab test values or use imputation strategies.
2. **Encode Categorical Variables:** Diagnosis codes, discharge locations → one-hot encoding.
3. **Feature Engineering:**
    - "Number of previous admissions in last year"
    - "Days since last hospital visit"
    - "Comorbidity score"
    - "Average length of stay"

Model Development (10 points)

Model Chosen:

- **Logistic Regression**

Justification:

- High interpretability—important in healthcare.
- Outputs probabilities for easy threshold tuning.
- Lightweight and requires less computational power.

Hypothetical Confusion Matrix (for 200 patients):

	Predicted: Yes	**Predicted: No**
Actual: Yes	50	10
Actual: No	20	120

Precision = 50 / (50 + 20) = **0.714**
Recall = 50 / (50 + 10) = **0.833**

Deployment (10 points)

Integration Steps into Hospital System:

1. **API Development:** Serve the model via a secure REST API.
2. **EHR Integration:** Connect the API to hospital EHR systems to trigger predictions at discharge.
3. **User Interface:** Design an alert system for doctors and care coordinators.
4. **Monitoring Dashboard:** Track model performance, drift, and usage.

Ensuring Regulatory Compliance (e.g., HIPAA):

- Encrypt all data in transit and at rest.
- Use access control and authentication mechanisms.
- Maintain audit trails for who accessed predictions and when.
- Anonymize patient data during model training.

Optimization (5 points)

Method to Address Overfitting:

- **Cross-validation** combined with **L2 Regularization (Ridge Penalty)** to penalize overly complex models and ensure generalization to unseen patient data.

Part 3: Critical Thinking (20 Points)

Ethics & Bias (10 points)

Q1: How might biased training data affect patient outcomes in the case study?

Biased training data can lead to **unfair or harmful predictions**. For example, if the data underrepresents certain groups (e.g., low-income patients, minorities, or elderly), the model might **underestimate their readmission risk**, resulting in fewer follow-ups or interventions. This can **worsen health outcomes**, increase disparities, and reduce trust in the healthcare system.

Q2: Suggest 1 strategy to mitigate this bias.

Use **bias detection and fairness-aware algorithms**. One effective strategy is:

> **Stratified sampling during data preparation**, ensuring balanced representation of different demographics (e.g., gender, age, ethnicity).
> This helps the model learn patterns equitably across diverse patient groups.

Additional mitigation can include:

- Using **Fairlearn** or **AIF360** libraries to evaluate fairness metrics.
- Regular **bias audits** post-deployment.

Trade-offs (10 points)

Q1: Discuss the trade-off between model interpretability and accuracy in healthcare.

In healthcare, **interpretability is often prioritized over raw accuracy**. A highly accurate model like a deep neural network might be a "black box," making it hard for doctors to trust or act on predictions. On the other hand, interpretable models like logistic regression or decision trees:

- Allow clinicians to understand *why* a prediction was made,
- Provide **transparency** and support **informed decision-making**,
- Align with **ethical and legal standards** like explainability under data regulations.

Thus, while some accuracy might be sacrificed, the **trust, accountability, and usability** gained through interpretability is essential in healthcare.

Q2: If the hospital has limited computational resources, how might this impact model choice?

Limited computational resources may require:

- Choosing **lightweight models** (e.g., Logistic Regression, Naive Bayes) over computationally expensive ones (e.g., XGBoost, Neural Networks).
- Avoiding models that require **GPU acceleration** or **long training times**.
- Preferring **batch predictions** (e.g., once a day) over real-time predictions.

This ensures the system remains **efficient, affordable, and maintainable** without compromising critical hospital operations.

Part 4: Reflection & Workflow Diagram (10 Points)

Reflection (5 points)

Q1: What was the most challenging part of the workflow? Why?

The most challenging part was the **data preprocessing phase**. This is because:

- Healthcare data is often **incomplete**, **inconsistent**, and **highly sensitive**.
- We had to balance between **preserving data quality** and ensuring **patient privacy**.
- Designing an effective **feature engineering pipeline** required domain knowledge, which added complexity.

Q2: How would you improve your approach with more time/resources?

With more time and resources, we would:

- **Collaborate with healthcare professionals** to refine our features and validate the model's outputs.
- **Explore advanced techniques** like synthetic data generation (e.g., using GANs) to augment limited datasets.
- **Build a continuous monitoring system** for evaluating model drift and fairness in real-time.

Diagram (5 points)

Here is a **flowchart of the AI Development Workflow**. You can recreate this using tools like Draw.io, Lucidchart, or even by hand and export it as an image.

```
+--------------------------+
|  1. Problem Definition   |
+--------------------------+
            ↓
+--------------------------+
|  2. Data Collection      |
+--------------------------+
            ↓
+--------------------------+
|  3. Data Preprocessing   |
+--------------------------+
            ↓
+--------------------------+
|  4. Model Development    |
+--------------------------+
            ↓
+--------------------------+
|  5. Evaluation           |
+--------------------------+
            ↓
+--------------------------+
|  6. Deployment           |
+--------------------------+
            ↓
+--------------------------+
|  7. Monitoring & Feedback|
+--------------------------+
```