Part 1: Short Answer Questions (30 points)

1. Problem Definition (6 points)

- Define a hypothetical AI problem (e.g., "Predicting student dropout rates").
- List **3 objectives** and **2 stakeholders**.
- Propose **1 Key Performance Indicator (KPI)** to measure success.

Answer:

AI Problem: Predicting patient no-show appointments in outpatient clinics.

Objectives:

1. Reduce the number of missed appointments.
2. Improve scheduling efficiency.
3. Help clinicians better plan their day.

Stakeholders:

1. Clinic administrators
2. Patients

Key Performance Indicator (KPI): How much the number of missed appointments goes down after using the AI system.

Question:

2. Data Collection & Preprocessing (8 points)

- Identify **2 data sources** for your problem.
- Explain **1 potential bias** in the data.
- Outline **3 preprocessing steps** (e.g., handling missing data, normalization).

Answer:

Data Collection & Preprocessing

Data Sources:

1. Appointment booking records from the clinic's internal system.
2. Patient reminders and communication logs (like SMS or email response data).

Potential Bias: The data might favor patients who use digital communication more often, which could lead to unfair predictions for those without phones or internet access.

Preprocessing Steps:

1. Fill in missing details like age or appointment type if possible.
2. Turn text-based features (like appointment reasons) into numbers using encoding.

3. Normalize date/time features so they’re easier for the model to understand.

<u>Question</u>

3. Model Development (8 points)

- Choose a model (e.g., Random Forest, Neural Network) and justify your choice.
- Describe how you would split data into training/validation/test sets.
- Name **2 hyperparameters** you would tune and why.

<u>Answer</u>

3. Model Development

Model Choice: I'll use a **Random Forest** model. It works well with tabular data like appointment records, handles missing values better than some other models, and helps avoid overfitting by using many decision trees.

Data Splitting: Split the dataset into:

- 70% for training
- 15% for validation (to fine-tune the model)
- 15% for testing (to check performance on new data)

Hyperparameters to Tune:

1. **Number of trees** – More trees might improve accuracy but also increase computation time.
2. **Maximum depth of trees** – Controls how complex each tree is, which affects overfitting.

<u>Question:</u>

4. Evaluation & Deployment (8 points)

- Select **2 evaluation metrics** and explain their relevance.
- What is **concept drift**? How would you monitor it post-deployment?
- Describe **1 technical challenge** during deployment (e.g., scalability).

<u>Answer:</u>

4. Evaluation & Deployment

Evaluation Metrics:

1. **Accuracy** – Tells us the overall percentage of correct predictions. It's a good starting point to see if the model is working well.
2. **Recall** – Helps measure how many of the actual no-shows the model correctly caught. This is important because missing a no-show could mean an empty appointment slot that could’ve been given to someone else.

What is Concept Drift? Concept drift happens when patterns in the data change over time. For example, during flu season, patient behavior might shift. What worked last month might not work as well now.

How to Monitor It: Keep tracking the model's performance over time. If accuracy or recall starts dropping, retrain the model with the latest data.

Deployment Challenge – Scalability: The model might work well during testing, but once it's live and handling thousands of daily appointments, performance could slow down or crash. You'd need a setup that can scale, like cloud deployment or load balancing.

Part 2: Case Study Application (40 points)

Question:

1. **Problem Scope (5 points):** Define the problem, objectives, and stakeholders.

Answer:

Problem Scope (5 points)

Problem Statement: The hospital wants to reduce the chances of patients being readmitted within 30 days after they leave. The AI system will predict which patients are at high risk of readmission, so that doctors can give them extra care or follow-up plans.

Objectives:

1. Identify patients at high risk of readmission.
2. Help doctors make better discharge and follow-up decisions.
3. Lower overall hospital readmission rates and improve patient outcomes.

Stakeholders:

1. Doctors and nurses who use the system for care decisions.
2. Hospital administrators who track performance and costs

Question:

Data Strategy (10 points):

1. Propose data sources (e.g., EHRs, demographics).
2. Identify **2 ethical concerns** (e.g., patient privacy).
3. Design a preprocessing pipeline (include feature engineering steps).

Answer:

Data Strategy (10 points)

Proposed Data Sources:

1. Electronic Health Records (EHRs) – These include diagnoses, previous admissions, medications, and lab results.
2. Patient demographics and discharge summaries – Age, gender, living situation, and follow-up care plans.

Ethical Concerns:

1. **Patient privacy** – Handling personal health data must follow strict rules to protect sensitive information.
2. **Bias in historical data** – If past care was uneven (e.g., some groups received less follow-up), the model might learn those same unfair patterns.

Preprocessing Pipeline:

1. **Data cleaning** – Remove or fix incomplete and inconsistent entries, such as missing discharge dates or diagnosis codes.
2. **Feature engineering** – Create new features like "number of hospital visits in the past year" or "time since last admission."
3. **Encoding and scaling** – Convert categorical features (like diagnosis type) into numbers and scale numerical features (like age) to help the model learn better.

Question:

Model Development (10 points):

- Select a model and justify it.
- Create a confusion matrix and calculate precision/recall (hypothetical data).

Answer:

Model Selection: A **Logistic Regression** model would be a good starting point. It's simple, fast, and works well when the goal is to predict a yes/no outcome, like whether a patient will be readmitted or not. Plus, it's easy for hospital staff to understand how the model makes decisions, which is important in healthcare.

Confusion Matrix (Hypothetical Data):

	Predicted: Readmit	**Predicted: No Readmit**
Actual: Readmit	50 (True Positive)	10 (False Negative)
Actual: No Readmit	20 (False Positive)	120 (True Negative)

Precision: = 50 / (50 + 20) = 0.71 This means that when the model predicts a readmission, it's right about 71% of the time.

Recall: = 50 / (50 + 10) = 0.83 This means the model is catching 83% of the actual readmission cases, which is important to avoid missed care opportunities.

Question:

Deployment (10 points):

- Outline steps to integrate the model into the hospital's system.
- How would you ensure compliance with healthcare regulations (e.g., HIPAA)?

Answer:

Deployment (10 points)

Steps to Integrate the Model into the Hospital's System:

1. **API Integration** – Package the model into an API (like using Flask or FastAPI) so it can connect with the hospital's patient data systems.
2. **Trigger Points** – Set it up to run automatically at key moments, like right before discharge.
3. **User Interface** – Build a simple dashboard or tool where doctors can see the readmission risk score and key factors behind it.
4. **Monitoring and Logs** – Track the model's predictions and real outcomes to catch errors and retrain when needed.

Compliance with Healthcare Regulations (e.g., HIPAA):

1. **Data Encryption** – Make sure all patient data is encrypted during storage and transfer.
2. **Access Control** – Limit who can access the model and patient data—only authorized users like doctors or IT staff.
3. **Audit Trails** – Keep detailed logs of who accessed what data and when. This helps stay transparent and accountable.

Question:

Optimization (5 points): Propose **1 method** to address overfitting.

Answer:

Optimization (5 points)

To help prevent overfitting, I would use **regularization**. It works by adding a small penalty to the model if it tries to place too much importance on certain features. This keeps things balanced and stops it from learning noise in the training data. Since I'm using logistic regression, I'd apply L2 regularization—it's simple and effective.

I'd also use **cross-validation**. That means I'd split the training data in different ways, train the model several times, and check how stable the results are. If the model performs well across all those runs, I know it's more likely to do well on new data too.

Part 3: Critical Thinking (20 points)

Question:
Ethics & Bias (10 points):

- How might biased training data affect patient outcomes in the case study?
- Suggest **1 strategy** to mitigate this bias.

Answer:

Ethics & Bias (10 points)

If the training data is biased, it could lead to unfair treatment of certain patients. For example, if the historical data shows that patients from rural areas or low-income backgrounds received less care, the model might learn to wrongly assume they are less likely to need follow-up support. That could cause it to ignore patients who actually need help, just because of where they live or their background.

To reduce this kind of bias, I would start by checking how the data is spread across different groups like age, gender, or location. Then I'd balance the training data so that underrepresented groups have enough examples. I'd also monitor fairness metrics during evaluation to catch any warning signs early.

Question:

Trade-offs (10 points):

- Discuss the trade-off between model interpretability and accuracy in healthcare.
- If the hospital has limited computational resources, how might this impact model choice?

Answer:

Trade-offs (10 points)

In healthcare, there's often a trade-off between **model interpretability** and **accuracy**. A simple model like logistic regression is easy to understand and explain to doctors—it shows how each factor (like age or past visits) affects the prediction. But it might not be as accurate as a more complex model like a deep neural network.

On the other hand, a more accurate model might be harder to explain. That can be risky in healthcare, where trust and transparency are important. Doctors need to understand why a model made a certain prediction so they can make informed decisions.

If the hospital has limited computational resources, that might also shape the choice of model. Heavier models like neural networks need more power and time to run. In this case, I would go for a lighter model that still gives solid performance, like logistic regression or a random forest with limited depth.

Part 4: Reflection & Workflow Diagram (10 points)

Question:

1. **Reflection (5 points):**
   - What was the most challenging part of the workflow? Why?
   - How would you improve your approach with more time/resources?

Answer:

Reflection (5 points)

The most challenging part of the workflow was the **data preprocessing stage**. It required careful thinking to clean and organize the data in a way that the model could understand, while also preserving important information. For example, I had to balance feature engineering with ethical awareness, especially when dealing with sensitive health data.

If I had more time and resources, I would invest in exploring more advanced models, like gradient boosting or ensemble techniques, and test how they compare in both accuracy and fairness. I'd also spend more time improving the user interface for doctors, making the tool more user-friendly and interpretable.

Question:

1. **Diagram (5 points):**
   - Sketch a flowchart of the AI Development Workflow, labeling all stages.

Answer:

Workflow Diagram (5 points)

Here's a simple description of the AI Development Workflow as a flowchart:

```
1. Problem Definition
   ↓
2. Data Collection
   ↓
3. Data Preprocessing
   ↓
4. Model Selection and Training
   ↓
5. Model Evaluation
   ↓
6. Deployment
   ↓
7. Monitoring and Maintenance
```

Each stage feeds into the next, and I would show arrows looping back from monitoring to model training, since improving the model over time is key.