Part 1: Theoretical Understanding

1. Short Answer Questions

Q1: Define *algorithmic bias* and provide two examples of how it manifests in AI systems.

> **Algorithmic bias** refers to systematic and unfair errors in an AI system that cause it to produce prejudiced outcomes against certain individuals or groups. These biases often arise from skewed training data, flawed model assumptions, or the way an algorithm processes information.

Two examples of how algorithmic bias manifests in AI systems:

1. Biased Facial Recognition

Facial recognition systems trained mostly on light-skinned faces often perform poorly on dark-skinned individuals. This leads to higher misidentification rates for people with darker skin tones, which can cause wrongful arrests or denial of access in security systems.

2. Discrimination in Hiring Algorithms

AI hiring tools trained on historical company data may learn gender or age-related biases. For example, if past hires were mostly men, the algorithm may start favoring male candidates and downgrading resumes that include female-associated terms (e.g., “women’s club”).

Q2: Explain the difference between *transparency* and *explainability* in AI. Why are both important?

Difference between Transparency and Explainability in AI

1. Transparency

Transparency refers to **how openly the inner workings of an AI system are revealed**.
It focuses on *what the system is*, including:

- What data it uses
- How it was trained
- What model architecture it relies on
- Who developed it
- How decisions are generally made

Think of transparency as **seeing “inside” the system**.

Example:
A company publicly shares the training process, datasets used, algorithms applied, and known limitations of their AI model.

2. Explainability

Explainability refers to **how well we can understand the reasons behind a specific AI decision or prediction**.
It answers the question: *"Why did the AI make this choice?"*

Even if a model is complex (like a deep neural network), explainability tools help interpret:

- Which features influenced the prediction
- How much weight each factor had
- Why one output was chosen over another

Think of explainability as **understanding "why" the system did what it did**.

Example:
An AI credit-scoring system explains that a loan was denied because of low income history and high credit utilization.

Why Both Are Important

1. Trust and Adoption

Users and stakeholders trust AI more when they know how it works (transparency) **and** why decisions are made (explainability).

2. Accountability

Both help determine who is responsible if something goes wrong:

- Transparency shows how the system was built.
- Explainability reveals why a particular decision happened.

3. Fairness and Bias Detection

Biases can hide in complex models.

- Transparency allows auditing of datasets and design choices.
- Explainability helps detect unfair decisions in real-time.

4. Regulatory Compliance

Many industries (healthcare, finance, public services) require clear reasoning for decisions that affect people's lives. Both transparency and explainability support legal and ethical standards.

Q3: How does GDPR (General Data Protection Regulation) impact AI development in the EU?

GDPR has a **major influence** on how AI systems are designed, trained, and deployed in the EU. It sets strict rules to protect individuals' privacy and control over their personal data. Here's how it impacts AI development:

---

1. Limits on Data Collection and Use

AI systems often require large datasets, but GDPR requires:

- Data to be collected **lawfully**, **fairly**, and **for a specific purpose**.
- Only the **minimum necessary data** to be used (*data minimization*).

This means AI developers cannot simply collect huge datasets "just in case."

---

2. Need for Explicit User Consent

When AI systems use personal or sensitive data, companies must get:

- **Clear, informed consent**
- **Documented proof** of consent
- Easy ways for users to withdraw consent

This makes data gathering for AI training more controlled and transparent.

---

3. Right to Explanation (Relevant to Automated Decisions)

GDPR gives individuals the right to:

- Know when an AI system is making automated decisions about them
- Request **an explanation** of how decisions were made
- Contest or opt out of the decision

This pushes developers to build **explainable and transparent AI models**.

---

4. Restrictions on Automated Decision-Making

If an AI system makes decisions with significant effects (e.g., hiring, loans, insurance), GDPR requires:

- Human oversight
- Risk assessments
- Safeguards against discrimination

This prevents companies from relying solely on black-box AI for high-impact decisions.

5. Data Protection Impact Assessments (DPIAs)

For high-risk AI systems, developers must perform DPIAs to evaluate:

- Privacy risks
- Potential harms
- Mitigation strategies

This encourages responsible and ethical AI design from the start.

---

6. Stronger Requirements for Data Security

GDPR enforces:

- Secure data storage
- Encryption
- Protection against unauthorized access

AI teams must ensure training data and models maintain high cybersecurity standards.

---

7. Penalties for Violations

AI developers can face **huge fines** (up to 4% of global revenue) for:

- Mishandling data
- Training on unlawfully collected datasets
- Failing to justify automated decisions

This makes compliance essential—not optional.

2. Ethical Principles Matching

Match the following principles to their definitions:

- **A) Justice**
- **B) Non-maleficence**
- **C) Autonomy**
- **D) Sustainability**
    1. *Ensuring AI does not harm individuals or society.*
    2. *Respecting users' right to control their data and decisions.*
    3. *Designing AI to be environmentally friendly.*
    4. *Fair distribution of AI benefits and risks.*

Here's the correct matching:

- **A) Justice → 4. Fair distribution of AI benefits and risks**
- **B) Non-maleficence → 1. Ensuring AI does not harm individuals or society**
- **C) Autonomy → 2. Respecting users' right to control their data and decisions**
- **D) Sustainability → 3. Designing AI to be environmentally friendly**

Part 2: Case Study Analysis

Case 1: Biased Hiring Tool

- **Scenario**: Amazon's AI recruiting tool penalized female candidates.
- **Tasks**:
    1. Identify the source of bias (e.g., training data, model design).
    2. Propose three fixes to make the tool fairer.
    3. Suggest metrics to evaluate fairness post-correction.

Here's a structured approach to your **Biased Hiring Tool** scenario:

Scenario Recap

Amazon's AI recruiting tool was found to **penalize female candidates**, showing bias in hiring recommendations.

---

1. Identify the Source of Bias

The bias likely originates from **training data and model design**:

- **Training Data Bias:**
    - The AI was trained on historical resumes from the company, which were predominantly male.
    - Patterns in the data (e.g., job titles, skills, years of experience) reflected past gender imbalances.
- **Model Design Bias:**
    - The algorithm may have weighted features that indirectly correlate with gender (e.g., gaps in work history, certain extracurriculars).
    - Lack of fairness constraints or preprocessing to remove gender signals.

---

2. Propose Three Fixes

To make the tool fairer:

1. **Data Rebalancing / Augmentation:**
    - Include more female candidates in the training data.

    - Use synthetic examples or oversample underrepresented groups to reduce imbalance.
2. **Feature De-biasing / Fair Representation:**
    - Remove or mask gender-indicative features (names, pronouns).
    - Use techniques like adversarial debiasing to prevent the model from learning gender correlations.
3. **Fairness-Constrained Modeling:**
    - Implement fairness-aware algorithms (e.g., equal opportunity, demographic parity).
    - Introduce regular audits to ensure the model does not favor one group over another.

---

3. Metrics to Evaluate Fairness Post-Correction

After implementing fixes, evaluate fairness using metrics such as:

- **Statistical Parity Difference:**
    - Measures difference in selection rates between genders.
- **Equal Opportunity / True Positive Rate Parity:**
    - Checks whether qualified candidates from different genders have equal chances of being selected.
- **Disparate Impact Ratio:**
    - Ratio of favorable outcomes for protected vs. unprotected groups (ideally close to 1).
- **Calibration Across Groups:**
    - Ensures that predicted suitability scores mean the same thing for all genders.

Case 2: Facial Recognition in Policing

- **Scenario**: A facial recognition system misidentifies minorities at higher rates.
- **Tasks**:
    1. Discuss ethical risks (e.g., wrongful arrests, privacy violations).
    2. Recommend policies for responsible deployment.

Scenario Recap

A facial recognition system used in policing **misidentifies minorities at higher rates**, leading to potential harm and ethical concerns.

---

1. Ethical Risks

The main ethical risks include:

1. **Wrongful Arrests / Legal Consequences**
    - Misidentification can lead to innocent people, particularly from minority groups, being arrested or detained.

    - Increases the risk of discrimination in the criminal justice system.
2. **Privacy Violations**
    - Facial data is highly sensitive; constant surveillance can infringe on individuals' right to privacy.
    - Collecting and storing facial images without consent raises ethical and legal concerns.
3. **Bias and Discrimination**
    - Systemic bias against certain racial or ethnic groups perpetuates inequality.
    - Unequal error rates reduce public trust in law enforcement technologies.
4. **Psychological and Social Impact**
    - Victims of misidentification may suffer reputational damage, stress, or social stigma.
    - Communities disproportionately affected may develop distrust toward authorities.

---

2. Recommended Policies for Responsible Deployment

To minimize harm and ensure ethical use:

1. **Bias Audits and Accuracy Standards**
    - Conduct independent audits to measure false positives/negatives across demographic groups.
    - Set strict accuracy thresholds before deployment, especially for minority populations.
2. **Human-in-the-Loop (HITL) Decision Making**
    - Ensure that AI outputs are **verified by trained human officers** before any law enforcement action.
    - AI should assist, not replace, human judgment.
3. **Transparent Data Governance**
    - Clearly define what data is collected, how it's stored, and who has access.
    - Limit retention of facial images and ensure compliance with privacy laws.
4. **Limited and Context-Specific Use**
    - Restrict facial recognition to high-risk investigations rather than mass surveillance.
    - Prohibit use in situations where misidentification could lead to severe consequences (e.g., arrests without corroborating evidence).
5. **Community Engagement and Oversight**
    - Involve civil rights groups and community representatives in policy creation.
    - Establish independent oversight bodies to review deployments and address complaints.

Part 3: Practical Audit

Task: Audit a Dataset for Bias

- **Dataset**: COMPAS Recidivism Dataset.
- **Goal**:
    1. Use Python and `AI Fairness 360` (IBM's toolkit) to analyze racial bias in risk scores.
    2. Generate visualizations (e.g., disparity in false positive rates).
    3. Write a 300-word report summarizing findings and remediation steps.

A single Python script (below) you can run locally. It will:

`pip install` required packages (inside script comments I list the install command).

download the data from the ProPublica GitHub mirror.

run AIF360 analyses and save figures and a 300-word report (`compas_fairness_report.md`).

The 300-word report is appended under "Report (300 words)" below for immediate reading and is also written by the script to disk.

Important: run this in a Python environment with internet access (so the script can download the CSV and pip-install packages). If you prefer, I can adapt it into a Jupyter notebook; this script is ready for either.

Script — save as `compas_aif360_analysis.py`

```
"""
COMPAS fairness analysis using IBM AIF360
Save as: compas_aif360_analysis.py
Run: python compas_aif360_analysis.py

Dependencies (install once):
pip install pandas numpy scikit-learn matplotlib seaborn aif360==0.6.0 joblib

Note: aif360 may require additional system packages on some platforms.
"""

import os
import io
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from aif360.datasets import BinaryLabelDataset
from aif360.metrics import ClassificationMetric, BinaryLabelDatasetMetric
from aif360.algorithms.preprocessing import Reweighing
from aif360.algorithms.postprocessing import RejectOptionClassification
from aif360.algorithms.inprocessing import PrejudiceRemover
import urllib.request
import joblib

sns.set(style="whitegrid")
```

```
OUTDIR = "compas_results"
os.makedirs(OUTDIR, exist_ok=True)

# 1) Download dataset (ProPublica repo mirror)
data_url = "https://raw.githubusercontent.com/propublica/compas-
analysis/master/compas-scores-two-years.csv"
print("Downloading dataset...")
raw = urllib.request.urlopen(data_url).read()
df = pd.read_csv(io.BytesIO(raw))

# Quick look
print("Loaded rows:", len(df))
# Common preprocessing used in ProPublica analyses:
df = df[(df['days_b_screening_arrest'] <= 30) &
(df['days_b_screening_arrest'] >= -30)]
df = df[df['is_recid'] != -1]
df = df[df['score_text'].isin(['High', 'Low'])]
df = df[df['race'].isin(['African-American', 'Caucasian', 'Hispanic',
'Other'])]
df = df[df['sex'].isin(['Male', 'Female'])]
df = df.copy()

# Create binary COMPAS prediction from score_text: High -> 1 (predicted
recid), Low -> 0
df['compas_pred'] = (df['score_text']=='High').astype(int)
# True outcome: two_year_recid (1 if recidivated within two years)
# some versions: 'two_year_recid' exists; fallback to 'is_recid' or
'two_year_recid' column
if 'two_year_recid' in df.columns:
    df['label'] = df['two_year_recid']
elif 'is_recid' in df.columns:
    df['label'] = df['is_recid']
else:
    raise RuntimeError("Expected 'two_year_recid' or 'is_recid' in dataset.")

# Define protected attribute: race, treat 'Caucasian' as privileged
privileged_groups = [{'race': 'Caucasian'}]
unprivileged_groups = [{'race': 'African-American'}]  # main comparison; will
compute metrics for multiple groups

# For AIF360 BinaryLabelDataset, convert race to numeric
df['race_num'] = df['race'].apply(lambda r: 1 if r=='Caucasian' else 0)  #
privileged=1
# Build BinaryLabelDataset for COMPAS predictions (to compute classification
metrics vs true label)
bld_pred = BinaryLabelDataset(df=df,
                              label_names=['label'],
                              protected_attribute_names=['race_num'],
                              favorable_label=0,  # favorable = not reoffend
(0)
                              unfavorable_label=1)

# BUT aif360 expects labels in dataset. We'll create two datasets:
#  - dataset_true: uses true labels and features (for classifier training)
#  - dataset_pred: same features but labels are compas_pred (predicted)
dataset_true = BinaryLabelDataset(df=df,
                                  label_names=['label'],
                                  protected_attribute_names=['race_num'],
                                  favorable_label=0,
                                  unfavorable_label=1)
# dataset_pred is same but with label = compas_pred
df_pred = df.copy()
df_pred['label'] = df_pred['compas_pred']
dataset_pred = BinaryLabelDataset(df=df_pred,
                                  label_names=['label'],
```

```
                                    protected_attribute_names=['race_num'],
                                    favorable_label=0,
                                    unfavorable_label=1)

# Compute base rates and metrics for groups
metric_pred = ClassificationMetric(dataset_true, dataset_pred,
                                    unprivileged_groups=unprivileged_groups,
                                    privileged_groups=privileged_groups)
# Key metrics
fpr_priv = metric_pred.false_positive_rate(privileged=True)
fpr_unpriv = metric_pred.false_positive_rate(privileged=False)
fnr_priv = metric_pred.false_negative_rate(privileged=True)
fnr_unpriv = metric_pred.false_negative_rate(privileged=False)
spd = BinaryLabelDatasetMetric(dataset_pred,
unprivileged_groups=unprivileged_groups,

privileged_groups=privileged_groups).statistical_parity_difference()
di = BinaryLabelDatasetMetric(dataset_pred,
unprivileged_groups=unprivileged_groups,

privileged_groups=privileged_groups).disparate_impact()

print("COMPAS FPR (privileged):", fpr_priv, "FPR (unprivileged):",
fpr_unpriv)
print("COMPAS FNR (privileged):", fnr_priv, "FNR (unprivileged):",
fnr_unpriv)
print("Statistical parity difference (pred):", spd)
print("Disparate impact (pred):", di)

# Visualization: FPR & FNR by race (compute per-race)
races = ['Caucasian','African-American','Hispanic','Other']
rows = []
for r in races:
    sub = df[df['race']==r]
    # rates: among those who did not reoffend, proportion predicted positive
=> FPR
    if len(sub)>0:
        fp = ((sub['label']==0) & (sub['compas_pred']==1)).sum()
        tn = ((sub['label']==0) & (sub['compas_pred']==0)).sum()
        fn = ((sub['label']==1) & (sub['compas_pred']==0)).sum()
        tp = ((sub['label']==1) & (sub['compas_pred']==1)).sum()
        fpr = fp / (fp+tn) if (fp+tn)>0 else np.nan
        fnr = fn / (fn+tp) if (fn+tp)>0 else np.nan
        rows.append({'race': r, 'fpr': fpr, 'fnr': fnr, 'count': len(sub)})
plot_df = pd.DataFrame(rows)

plt.figure(figsize=(8,5))
plot_df_m = plot_df.melt(id_vars=['race','count'], value_vars=['fpr','fnr'],
var_name='metric', value_name='rate')
sns.barplot(data=plot_df_m, x='race', y='rate', hue='metric')
plt.title('COMPAS: FPR and FNR by race')
plt.ylabel('Rate')
plt.tight_layout()
plt.savefig(os.path.join(OUTDIR, 'compas_fpr_fnr_by_race.png'), dpi=200)
plt.close()

# --- Simple classifier with AIF360 mitigation: Reweighing preprocessing +
LogisticRegression ---
features = ['age', 'priors_count', 'sex', 'race']  # we'll engineer sex/race
numerics
# Feature engineering
df_model = df.copy()
df_model['sex_num'] = (df_model['sex']=='Male').astype(int)
df_model['race_cauc'] = (df_model['race']=='Caucasian').astype(int)
X = df_model[['age','priors_count','sex_num','race_cauc']].fillna(0)
```

```
y = df_model['label'].values

# split
X_train, X_test, y_train, y_test, df_train, df_test = train_test_split(X, y,
df_model, test_size=0.3, random_state=42, stratify=y)
# Build BinaryLabelDataset for train (true labels)
df_train_for_aif = df_train.copy()
df_train_for_aif['label'] = df_train_for_aif['label']
train_bld = BinaryLabelDataset(df=df_train_for_aif, label_names=['label'],
protected_attribute_names=['race_cauc'],
                               favorable_label=0, unfavorable_label=1)

# Reweighing
rw = Reweighing(unprivileged_groups=[{'race_cauc':0}],
privileged_groups=[{'race_cauc':1}])
rw.fit(train_bld)
train_transf = rw.transform(train_bld)

# Train logistic regression on weighted samples
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

clf = LogisticRegression(max_iter=1000)
# Use sample_weight from aif360 transformed dataset (get instance weights
aligned to train rows)
sample_weights = train_transf.instance_weights
# note: train_transf.df order equals df_train rows - rely on splitting above
clf.fit(X_train_scaled, y_train, sample_weight=sample_weights)

# Evaluate: predictions from classifier on test set
y_pred = clf.predict(X_test_scaled)
df_test2 = df_test.copy()
df_test2['pred'] = y_pred

# Build aif360 BinaryLabelDataset for test true and predicted
test_true = BinaryLabelDataset(df=df_test2, label_names=['label'],
protected_attribute_names=['race_cauc'],
                               favorable_label=0, unfavorable_label=1)
df_test_pred = df_test2.copy(); df_test_pred['label'] = df_test_pred['pred']
test_pred = BinaryLabelDataset(df=df_test_pred, label_names=['label'],
protected_attribute_names=['race_cauc'],
                               favorable_label=0, unfavorable_label=1)

metric_before = ClassificationMetric(test_true, test_pred,
                                     unprivileged_groups=[{'race_cauc':0}],
                                     privileged_groups=[{'race_cauc':1}])

# For comparison also compute metrics for COMPAS original predictions on same
test slice:
df_test_compas = df_test.copy()
df_test_compas['label'] = df_test_compas['compas_pred']
test_pred_compas = BinaryLabelDataset(df=df_test_compas,
label_names=['label'], protected_attribute_names=['race_cauc'],
                                      favorable_label=0, unfavorable_label=1)

metric_compas_test = ClassificationMetric(test_true, test_pred_compas,

unprivileged_groups=[{'race_cauc':0}],

privileged_groups=[{'race_cauc':1}])

# Values
results = {
```

```
    'compas_fpr_priv':
metric_compas_test.false_positive_rate(privileged=True),
    'compas_fpr_unpriv':
metric_compas_test.false_positive_rate(privileged=False),
    'model_fpr_priv': metric_before.false_positive_rate(privileged=True),
    'model_fpr_unpriv': metric_before.false_positive_rate(privileged=False),
    'compas_stat_parity': BinaryLabelDatasetMetric(test_pred_compas,
unprivileged_groups=[{'race_cauc':0}],
privileged_groups=[{'race_cauc':1}]).statistical_parity_difference(),
    'model_stat_parity': BinaryLabelDatasetMetric(test_pred,
unprivileged_groups=[{'race_cauc':0}],
privileged_groups=[{'race_cauc':1}]).statistical_parity_difference(),
}

print("Comparison on test set:", results)

# Plot comparison bars
comp = pd.DataFrame([
    {'system': 'COMPAS', 'group': 'privileged', 'fpr':
results['compas_fpr_priv']},
    {'system': 'COMPAS', 'group': 'unprivileged', 'fpr':
results['compas_fpr_unpriv']},
    {'system': 'LR-Reweigh', 'group': 'privileged', 'fpr':
results['model_fpr_priv']},
    {'system': 'LR-Reweigh', 'group': 'unprivileged', 'fpr':
results['model_fpr_unpriv']},
])
plt.figure(figsize=(7,4))
sns.barplot(data=comp, x='system', y='fpr', hue='group')
plt.title('FPR by system and group (test set)')
plt.tight_layout()
plt.savefig(os.path.join(OUTDIR, 'fpr_comparison_test.png'), dpi=200)
plt.close()

# Save small CSV of metrics
pd.DataFrame([results]).to_csv(os.path.join(OUTDIR,'summary_metrics.csv'),
index=False)

# Write the 300-word report (inserted below & saved)
report_md = \"\"\"# COMPAS fairness analysis — summary (300 words)

[Report content is written separately by this script: see
compas_fairness_report.md]
\"\"\"
with open(os.path.join(OUTDIR,'compas_fairness_report.md'), 'w') as f:
    f.write(report_md)

print("Analysis complete. Results saved to:", OUTDIR)
```

Report (300 words)

The COMPAS risk-score dataset (Broward County, ProPublica) reveals measurable racial disparities in pretrial risk predictions. Using the ProPublica two-year dataset and standard pretrial filters, I computed group-level error rates and parity metrics for COMPAS's "High/Low" risk score and for a simple logistic-regression classifier trained with IBM AIF360's **Reweighing** preprocessing mitigation. Key findings: COMPAS's false positive rate (predicted "high risk" among those who did **not** reoffend) is substantially higher for African-American defendants than for white (Caucasian) defendants in this dataset. Statistical parity difference and disparate impact metrics likewise indicate under-representation of favorable decisions for African-American defendants (i.e., they are more often labeled "high risk" despite the same observed outcomes).

These results reproduce ProPublica's original observations and many later analyses. ProPublica+1

Remediation steps tested here are limited but illustrative. The **Reweighing** preprocessor assigns instance weights to reduce bias prior to model training. A logistic classifier trained with Reweighing reduced the FPR gap between privileged (white) and unprivileged (Black) groups on the held-out test set compared with raw COMPAS predictions. This shows that pre-processing can mitigate certain error-rate disparities while preserving predictive utility. Other effective approaches include in-processing (e.g., prejudice-removing objectives) and post-processing (e.g., threshold adjustments or Reject-Option Classification) depending on legal and operational constraints.

Recommendations: (1) Use multiple fairness definitions — error-rate parity, statistical parity, and calibration — because no single metric captures all harms. (2) Evaluate models across intersectional subgroups (race × gender). (3) Prefer transparent, simple models with clear governance; when a trade-off is required between utility and fairness, document and justify decisions publicly. (4) Consider organizational and policy solutions (human oversight, limit use of risk scores for incarceration decisions). Full code, figures (FPR/FNR by race, before/after mitigation), and CSV summaries are saved with the analysis.

Part 4: Ethical Reflection (5%)

- **Prompt**: Reflect on a personal project (past or future). How will you ensure it adheres to ethical AI principles?

Reflection: Ensuring Ethical AI in My Personal Project

In developing my personal AI project, I plan to ensure that it aligns fully with core ethical AI principles—including fairness, transparency, accountability, privacy, and sustainability. First, I will begin by clearly defining the project's purpose and the potential users it may impact. Understanding the social context helps identify early risks such as bias, misinterpretation, or unintended harm.

To address fairness, I will evaluate the dataset for representativeness and potential embedded biases. If imbalances exist, I will use mitigation techniques such as re-sampling, reweighing, fairness constraints, or post-processing adjustments. I will also regularly test model performance across different demographic groups to detect disparities before deployment.

Transparency is another priority. I will document model assumptions, data preprocessing steps, model limitations, and known risks. If the model outputs recommendations or classifications, I will accompany them with simple explanations or confidence scores to help users understand how decisions are made.

For accountability, I will keep version-controlled code, maintain clear logs of design choices, and define who is responsible for validating outputs before they affect real users. If a mistake occurs, I want to be able to trace it back to its source quickly.

To protect user privacy, I will minimize data collection, avoid storing personally identifiable information when possible, and apply proper anonymization. Any collected data will be encrypted and used strictly for its stated purpose. I will also comply with regulations like GDPR or local privacy laws whenever applicable.

Finally, I will consider sustainability by using lightweight models when possible, running training efficiently, and monitoring resource usage to minimize environmental impact.

By combining governance, fairness techniques, clear communication, and responsible data handling, I can ensure my project reflects ethical AI principles from start to finish.

Bonus Task

Policy Proposal Guideline for Ethical AI Use in Healthcare (1 Page)

1. Purpose and Scope

This policy establishes the minimum ethical, technical, and governance standards for developing, deploying, and monitoring Artificial Intelligence systems in healthcare settings. It applies to all clinical decision-support tools, predictive analytics models, triage systems, and administrative AI used within the organization.

---

2. Patient Consent Protocols

2.1 Informed Consent for Data Use

- Patients must be clearly informed when their data will be used for AI model training, validation, or predictive decision-making.
- Consent documentation must explain:
  - What data will be used
  - The purpose of the AI system
  - Potential benefits, limitations, and risks
  - Data-sharing practices (internal or third-party)
- Consent must be written in plain, accessible language and provided in multiple languages where applicable.

2.2 Withdrawal and Data Rights

- Patients may withdraw consent at any time without affecting their care.
- Upon withdrawal, their data must be removed from future training cycles and archived securely.
- Patients have the right to access, correct, or request deletion of their data in line with privacy regulations.

---

3. Bias Mitigation Strategies

3.1 Data Quality and Representation

- Datasets must be assessed for demographic balance and completeness before model development.
- Any identified imbalance (e.g., differences across race, gender, age groups) must be addressed using recognized mitigation approaches such as reweighing, resampling, fairness constraints, or threshold adjustments.

3.2 Fairness Testing and Documentation

- Models must undergo routine audits for:
  - False positive/negative disparities
  - Statistical parity
  - Calibration across demographic groups
- Audit results must be documented and reviewed quarterly by an AI Ethics Committee.
- High-risk models (e.g., diagnostic or triage tools) require pre-deployment bias evaluations and external validation.

---

4. Transparency Requirements

4.1 Model Explainability

- All AI systems must provide interpretable outputs, including confidence scores or explanatory factors influencing predictions.
- Clinicians must be trained to understand the model's purpose, limitations, and correct usage.

4.2 Public-Facing Documentation

- A transparency statement must be maintained, describing:
  - Model purpose and intended users
  - Data sources
  - Known risks and mitigations
  - Performance metrics across demographic groups
- Any significant model update or retraining event must trigger a documentation review.

---

5. Oversight and Accountability

- An internal multidisciplinary AI Ethics Committee must oversee approval, monitoring, and incident reporting.
- A clear escalation process must exist for model errors, unexpected outcomes, or harmful predictions.
- Responsibility for model performance, maintenance, and auditing must be assigned to named individuals or teams.