# Final Report: Marriage Likelihood Prediction — AI Development Workflow

**Course:** AI for Software Engineering
**Student / Author:** Bikila Keneni

## 1. Executive Summary

This project demonstrates the full AI Development Workflow applied to a creative, ethically mindful problem: predicting the **likelihood of a couple eventually marrying** based on quantified relationship features. The work is educational and experimental — it explores how behavioral, cultural, and socio-economic factors can be modeled to surface patterns that may correlate with relationship longevity. The model is not deterministic, does not claim to predict destiny, and the author emphasizes that faith and free will (Providence) are ultimately decisive in human relationships.

Key deliverables:

- Synthetic dataset representing couple traits (saved at `data/processed/marriage_data.csv`)
- End-to-end Jupyter notebooks for data generation, exploration, model training and evaluation (`notebooks/00-03`)
- Trained model artifact (`models/marriage_model.pkl`)
- Evaluation metrics and discussion

## 2. Problem Definition

### 2.1 Problem statement

Predict whether a couple is likely to marry (binary classification) within the relationship lifecycle based on features such as communication, financial stability, cultural/religious alignment, family acceptance, and emotional support.

### 2.2 Objectives

1. Build a reproducible pipeline from data generation to model deployment-ready artifacts.
2. Identify key relationship features correlated with a high likelihood of marriage.
3. Demonstrate ethical considerations and biases in relationship-focused AI.

### 2.3 Stakeholders

- End users (couples interested in relationship insights)
- Educators / researchers of social dynamics and relationship counseling
- Platform developers (future web application for matchmaking insight)

### 2.4 Key Performance Indicator (KPI)

- **Recall for the "High likelihood" class** — priority is to correctly identify couples truly likely to marry (minimize false negatives on positives), because missing a true positive is more harmful for targeted interventions or counseling suggestions.

---

# 3. Data Collection & Preprocessing

## 3.1 Data sources (this project)

For the assignment we used a **synthetic dataset** generated to reflect plausible distributions of relationship traits. In a production setting, recommended sources would include:

- Survey responses from consenting couples (structured questionnaires)
- Interaction logs on a couples-focused platform (activity, message sentiment)
- Demographic/contextual data (age, cultural background, family acceptance)

## 3.2 Potential biases

- **Sampling bias:** Synthetic sampling or convenience-sampled surveys may not represent diverse cultures or relationship norms.
- **Response bias:** Couples reporting on their relationship may overstate positive traits.
- **Cultural bias:** Models trained on one cultural context risk misclassifying couples from different cultures because behaviors and norms differ.

## 3.3 Preprocessing pipeline

1. **Directory & file checks:** Ensure `data/processed` exists and that the dataset CSV is present.
2. **Missing value handling:** Numeric features imputed with median values; categorical data handled via one-hot encoding when present.
3. **Scaling:** Standard scaler on numerical features for models sensitive to scale.
4. **Train/test split:** 80/20 split used in notebooks for development.
5. **Feature engineering:** Derived composite features (e.g., weighted communication & emotional support score) were tested to improve signal quality.

---

# 4. Model Development

## 4.1 Model choice

A **Random Forest Classifier** (or simpler logistic-regression baseline in some notebook variations) is used. Reasons:

- Handles mixed numeric features well
- Robust to noise and outliers
- Provides feature importances for interpretability (important in socially-sensitive contexts)

## 4.2 Hyperparameters considered

- `n_estimators` (number of trees) — balancing accuracy and compute cost.
- `max_depth` — controlling overfitting by limiting tree depth.

### 4.3 Training procedure

- Feature matrix `X` and label `y` read from `data/processed/marriage_data.csv`.
- StandardScaler applied to `X`.
- Data split into train/test (80/20).
- Model fit on training set and evaluated on the test set.

---

## 5. Evaluation

### 5.1 Reported metrics (final run)

**Confusion Matrix (test set):**

- True Negatives (Low predicted Low): 26
- False Positives (Low predicted High): 3
- False Negatives (High predicted Low): 26
- True Positives (High predicted High): 5

**Classification report:**

```
              precision    recall  f1-score   support

           0       0.50      0.90      0.64        29
           1       0.62      0.16      0.26        31

    accuracy                           0.52        60
   macro avg       0.56      0.53      0.45        60
weighted avg       0.56      0.52      0.44        60
```

### 5.2 Interpretation

- The model is conservative and good at detecting the **Low** class (recall 0.90) but poor at detecting **High** class (recall 0.16).
- This suggests a strong imbalance or weak signal for the positive class in the synthetic data or that the modeling approach needs more expressive features and better sampling.

### 5.3 Improvements & next steps

1. **Balance the dataset** (oversample positive cases or use targeted data collection).
2. **Feature enrichment**: include more behavioral signals (message sentiment, duration of relationship, timestamps) and social signals (family approvals, cultural markers).
3. **Model experimentation**: try gradient boosting (XGBoost/LightGBM) and calibrated probability thresholds.
4. **Fairness checks**: evaluate metrics across subgroups (culture, religion, age) to detect disparate impact.

---

## 6. Deployment & Operational Considerations

### 6.1 Integration

The notebook includes a minimal `src/app/app.py` demonstrating loading the model and making predictions. Future work includes building a secure web front-end where both partners supply answers and the model returns an interpreted probability and guidance (not verdicts).

### 6.2 Data privacy & compliance

- Any real deployment must use **consensual data collection**, anonymization, and secure storage.
- Storage and processing should comply with applicable laws (GDPR, local privacy laws).

### 6.3 Monitoring & concept drift

Regular retraining and monitoring are required. Monitor performance metrics and data distributions to detect drift. Add pipelines to gather fresh labeled feedback and re-evaluate.

---

## 7. Ethics & Reflection

### 7.1 Ethical concerns

- Predicting intimate life outcomes can harm privacy and autonomy.
- Potential misuse: automated matchmaking decisions or stigmatization.
- Cultural insensitivity if models are not localized.

### 7.2 Mitigation strategies

- Use anonymization and opt-in policies.
- Present results as **probabilistic suggestions** with clear disclaimers.
- Conduct fairness audits and seek diverse data sources.

### 7.3 Reflection

The most challenging part was designing realistic features that capture the nuance of relationships. With more time the project could expand to multi-modal signals (text, voice sentiment, interaction timelines) and a carefully designed human-in-the-loop workflow for feedback and correction.

---

## 8. Appendices

### A. Files in the repository (key)

- `notebooks/00-generate-data.ipynb` — synthetic data generation
- `notebooks/01-data-exploration.ipynb` — EDA and visualizations
- `notebooks/02-model-training.ipynb` — preprocessing and model training
- `notebooks/03-evaluation.ipynb` — final evaluation and confusion matrix
- `src/data/preprocess.py`, `src/models/train_model.py`, `src/eval/evaluate.py`, `src/app/app.py` — supporting scripts
- `models/marriage_model.pkl` — saved model artifact (if created)
- `data/processed/marriage_data.csv` — processed dataset used for training

## B. How to reproduce (short)

```
python -m venv venv
source venv/Scripts/activate  # Windows Git Bash
pip install -r requirements.txt
jupyter notebook   # run notebooks in order 00 -> 03
python src/models/train_model.py
python src/eval/evaluate.py
```

---

*End of report.*