🛠️ Problem Solving Log – Mumbai House Price Prediction App

This file documents the real-world challenges faced while building the Mumbai House Price Prediction App — and how each was resolved to showcase problem-solving, debugging, and real-world engineering skills.

✅ 1. Data Quality & Feature Selection

Problem:
The raw Mumbai dataset had too many missing values and low-quality features.

Solution:
Manually selected 18+ core features, removed missing rows, engineered new features like `price_per_sqft`, and normalized square footage.

✅ 2. Location Had No Impact on Predictions

Problem:
Changing location had no effect — model was over-relying on `sqrt` (square footage).

Solution:
Introduced `sqrt × location` interaction features + switched to XGBoost to capture non-linear relationships.

✅ 3. High MAE Despite Good R^2

Problem:
Even after tuning, the model had high MAE (> ₹50L).

Solution:
Removed extreme outliers, tuned XGBoost hyperparameters, and improved data scaling. Final MAE dropped to ₹29.5L.

✅ 4. API Not Working on Mobile

Problem:
Flutter app failed to connect to the Flask API when installed on a real device.

Solution:
Replaced `localhost` with the live Render API URL and added `INTERNET` permission in AndroidManifest.xml.

✅ 5. UI Was Too Basic

Problem:
Original app UI looked too plain and stretched full width on web.

Solution:
Rebuilt UI with modern card layout, dark mode, GetX for state management, form validation, and Indian currency formatting.

✅ 6. Build Failed: JDK/Gradle Issues

Problem:
Android Studio threw build errors due to missing or misconfigured JDK and SDK tools.

Solution:
Installed JDK 17, fixed JAVA_HOME and ANDROID_HOME paths, installed required cmdline-tools and accepted all Android SDK licenses.

✅ 7. Real Users Said Predictions Were Off

Problem:
Real Mumbai locals provided feedback that predicted prices felt outdated.

Solution:
Added inflation adjustment logic for 13.5% over 7 years in the backend prediction output to better reflect today's prices.

✅ 8. Location Influence Still Too Weak in Predictions

Problem:
Even with one-hot encoding, the model didn't respond well to location changes. It still leaned heavily on square footage.

Solution:
Created `sqrt × location_xx` interaction features to allow the model to learn the varying impact of size across different locations.

✅ 9. Model Accuracy Still Not Ideal (MAE ₹41L+)

Problem:
First enhanced XGBoost model had improved R^2 but still high MAE (~₹41L).

Solution:

- Removed outliers (price > ₹10 Cr or < ₹10L)
- Tuned XGBoost with:
  `n_estimators=300`, `learning_rate=0.05`, `max_depth=8`, `subsample=0.8`, `colsample_bytree=0.8`

✅ Final Result:

- R^2 Score: **0.802**
- RMSE: ₹57.2 Lakh

- MAE: ₹29.5 Lakh

✅ 10. Seamless Upgrade of Model Without Breaking App

Problem:
Deploying a new model could've required Flutter and API changes.

Solution:
Kept the input feature structure unchanged. Only updated `mumbai_price_model.pkl` and `model_columns.pkl`. Render auto-deployed the new model without any additional changes to app code.

📌 This log shows iterative problem-solving, applied ML debugging, and full-stack deployment understanding — ideal for portfolios, resumes, or interviews.