Project Overview

🏨 Hotel Booking Cancellation Prediction

Problem Statement

Hotels lose significant revenue when guests cancel bookings at the last minute, leaving rooms unoccupied with no time to rebook them.

Solution

Machine learning system that predicts cancellation risk before it happens, enabling proactive management.

Business Impact

- Detect 87.55% of potential cancellations
- Reduce revenue loss by up to €32,400/month per hotel
- Enable proactive customer engagement

Technology Stack

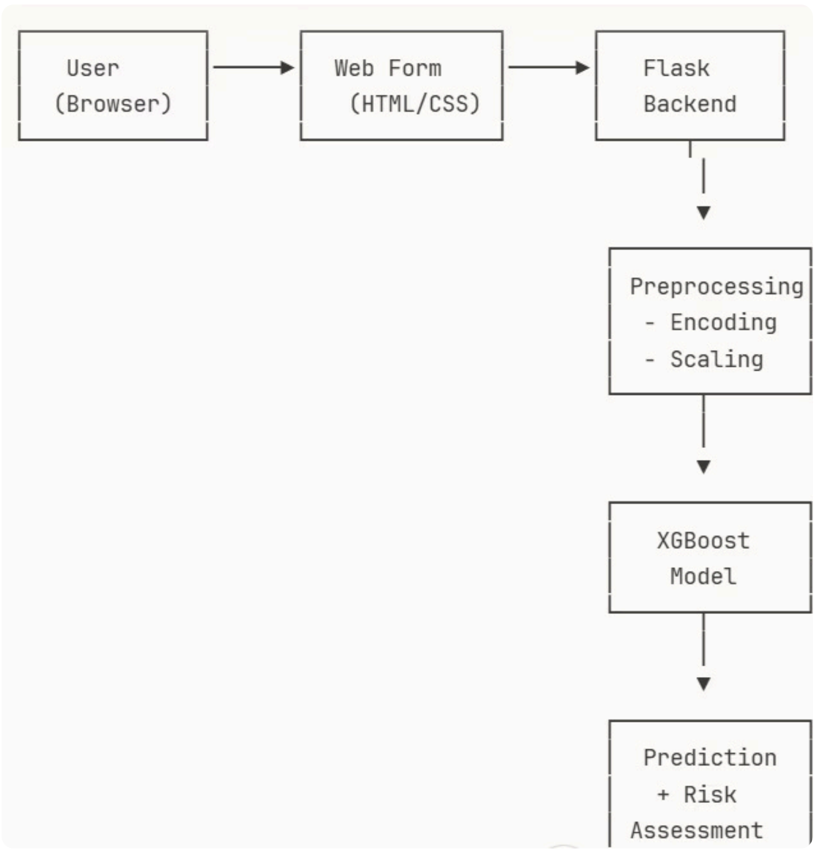
- **Model:** XGBoost Classifier
- **Deployment:** Flask Web Application (Local)
- **Frontend:** HTML, CSS, JavaScript
- **Language:** Python 3.x

Dataset

- 119,390 hotel bookings (2015-2017)
- 234 features after preprocessing
- Target: is_canceled (63% not canceled, 37% canceled)

System Architecture

High-Level Architecture



Data Flow

1. User Input → 20+ booking parameters
2. Feature Engineering → Calculate derived features
3. Encoding → One-hot encoding for categorical variables
4. Scaling → StandardScaler for 23 numeric features
5. Model Prediction → XGBoost outputs probability
6. Risk Assessment → Classify as Low/Medium/High
7. Display Results → Show prediction + recommendations

Dataset & Preprocessing

Dataset Description

Source

Hotel booking data from two Portuguese hotels (2015-2017)

Size

- **Total Records:** 119,390 bookings
- **Features:** 32 original → 234 after encoding
- **Target Distribution:**
  - Not Canceled: 75,166 (63%)
  - Canceled: 44,224 (37%)

Key Features

Feature	Type	Description	Example
lead_time	Numeric	Days between booking and arrival	30
adr	Numeric	Average daily rate (€)	100.50
previous_cancellations	Numeric	Past cancellations by guest	0, 1, 2...
deposit_type	Categorical	No Deposit / Non Refund / Refundable	No Deposit
market_segment	Categorical	Online TA / Direct / Corporate	Online TA
hotel	Categorical	City Hotel / Resort Hotel	Resort Hotel

Data Preprocessing Steps

1. Data Cleaning

Missing Values:

```
company → 0 (no company)
agent → 0 (no agent)
country → mode (most frequent)
children → median
```

Duplicates: Removed 8,031 duplicate rows

Outliers (Capping):

```
lead_time: capped at 365 days
adr: negative → 0, > 1000 → 1000
adults: 0 → 1 (minimum 1 adult per booking)
```

2. Feature Engineering

Created 5 new features:

```
total_guests = adults + children + babies
total_nights = weekend_nights + week_nights
is_family = 1 if (children + babies > 0)
stay_duration_category = Short/Medium/Long
season = Winter/Spring/Summer/Autumn (from month)
```

3. Encoding

Label Encoding:

- arrival_date_month → 1-12

One-Hot Encoding:

- hotel, meal, market_segment, distribution_channel
- reserved_room_type, deposit_type, customer_type
- country (180+ columns)

Result: 234 features total

4. Feature Scaling

StandardScaler applied to 23 numeric features:

```
Features = (X - mean) / std_deviation
```

Important: One-hot encoded features (0/1) were NOT scaled

Model Development

Machine Learning Model

Model Selection Process

Tested 4 algorithms:

Model	Accuracy	Precision	Recall	F1-Score	Selected
Random Forest	89.23%	88.97%	80.97%	84.78%	❌
XGBoost	**87.90%**	**81.23%**	**87.55%**	**84.27%**	✅
LightGBM	85.64%	77.80%	85.66%	81.54%	❌
Logistic Regression	82.50%	73.51%	77.20%	75.11%	❌

Why XGBoost?

1. ✅ **Best F1-Score** (84.27%) - balanced performance
2. ✅ **High Recall** (87.55%) - detects most cancellations
3. ✅ **Handles Imbalance** - scale_pos_weight parameter
4. ✅ **Built-in Regularization** - prevents overfitting
5. ✅ **Feature Importance** - interpretable results

Handling Class Imbalance

Compared 4 strategies:

Strategy	Recall	Precision	F1	Decision
Original + Weights	85.61%	78.22%	81.75%	✅ Selected
SMOTE	90.73%	72.12%	80.36%	❌ Low precision
Undersampling	92.66%	69.87%	79.67%	❌ Data loss
SMOTE + Tomek	90.23%	72.45%	80.11%	❌ No improvement

Selected Approach: `scale_pos_weight = 1.7`

Hyperparameter Tuning

Method: RandomizedSearchCV with 5-Fold Cross-Validation

Parameters Tested:

```
{
    'n_estimators': [100, 200, 300, 400],
    'max_depth': [4, 6, 8, 10],
    'learning_rate': [0.01, 0.05, 0.1, 0.15],
    'subsample': [0.7, 0.8, 0.9, 1.0],
    'colsample_bytree': [0.7, 0.8, 0.9, 1.0],
    'reg_alpha': [0, 0.01, 0.1],
    'reg_lambda': [1, 1.5, 2]
}
```

Best Parameters:

```
{
    'n_estimators': 300,
    'max_depth': 8,
    'learning_rate': 0.1,
    'subsample': 0.8,
    'colsample_bytree': 0.8,
    'reg_alpha': 0.01,
    'reg_lambda': 1,
    'scale_pos_weight': 1.7
}
```

Improvement:

- Before: F1 = 81.75%
- After: F1 = 84.27%
- **Gain: +2.52%**

Model Evaluation

Confusion Matrix:

```
              Predicted
           Not Cancel | Canceled
Actual Not  |  13,243   |  1,790  |
Actual Yes  |   1,100   |  7,745  |
```

Key Metrics:

- **Accuracy:** 87.90%
- **Precision:** 81.23% (81% of predicted cancellations are correct)
- **Recall:** 87.55% (detects 87.55% of actual cancellations)
- **F1-Score:** 84.27% (balanced performance)
- **AUC-ROC:** 95.63%

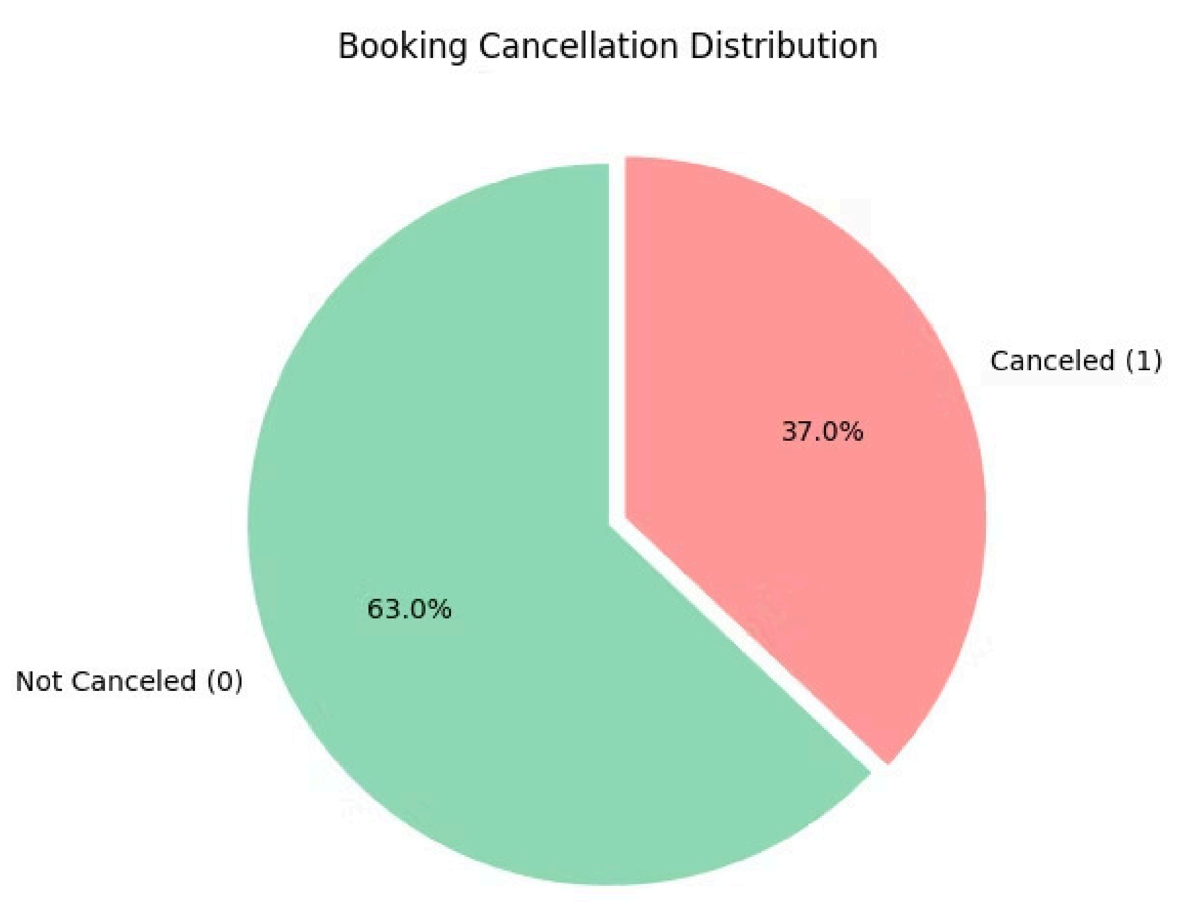
Cross-Validation (5-Fold):

- F1-Score: 84.19% ± 0.5%
- Consistent across folds → No overfitting

Visualization

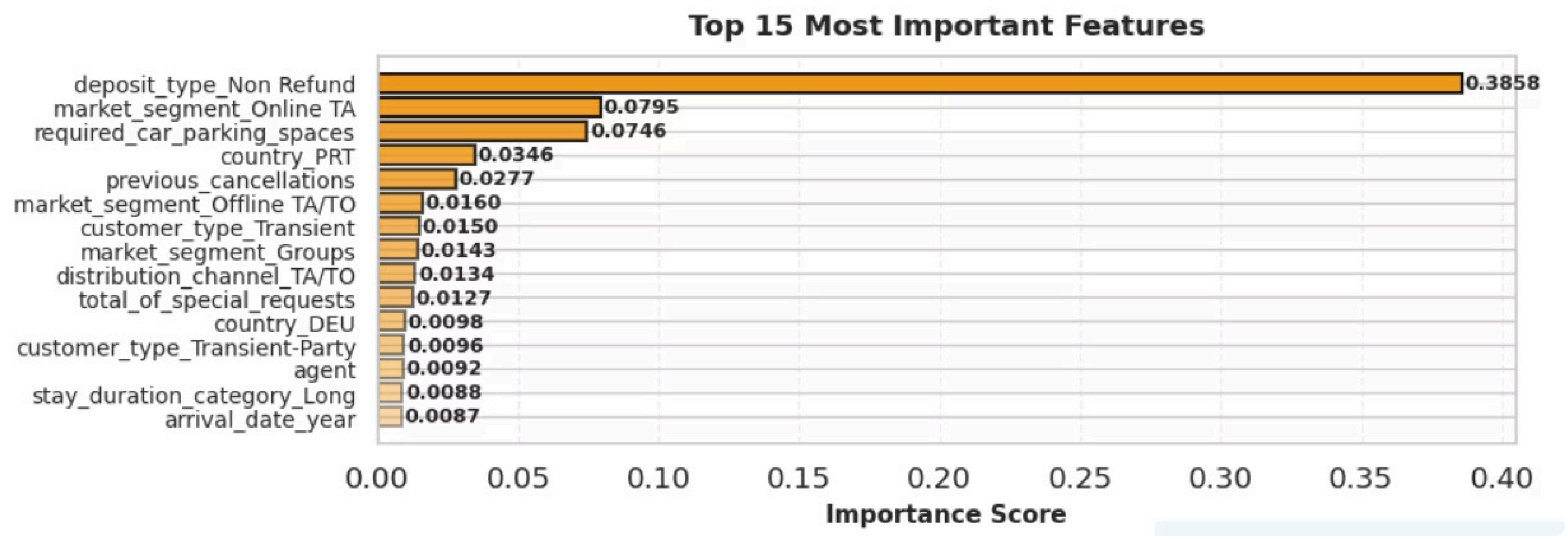
Data Visualizations

1. Class Distribution

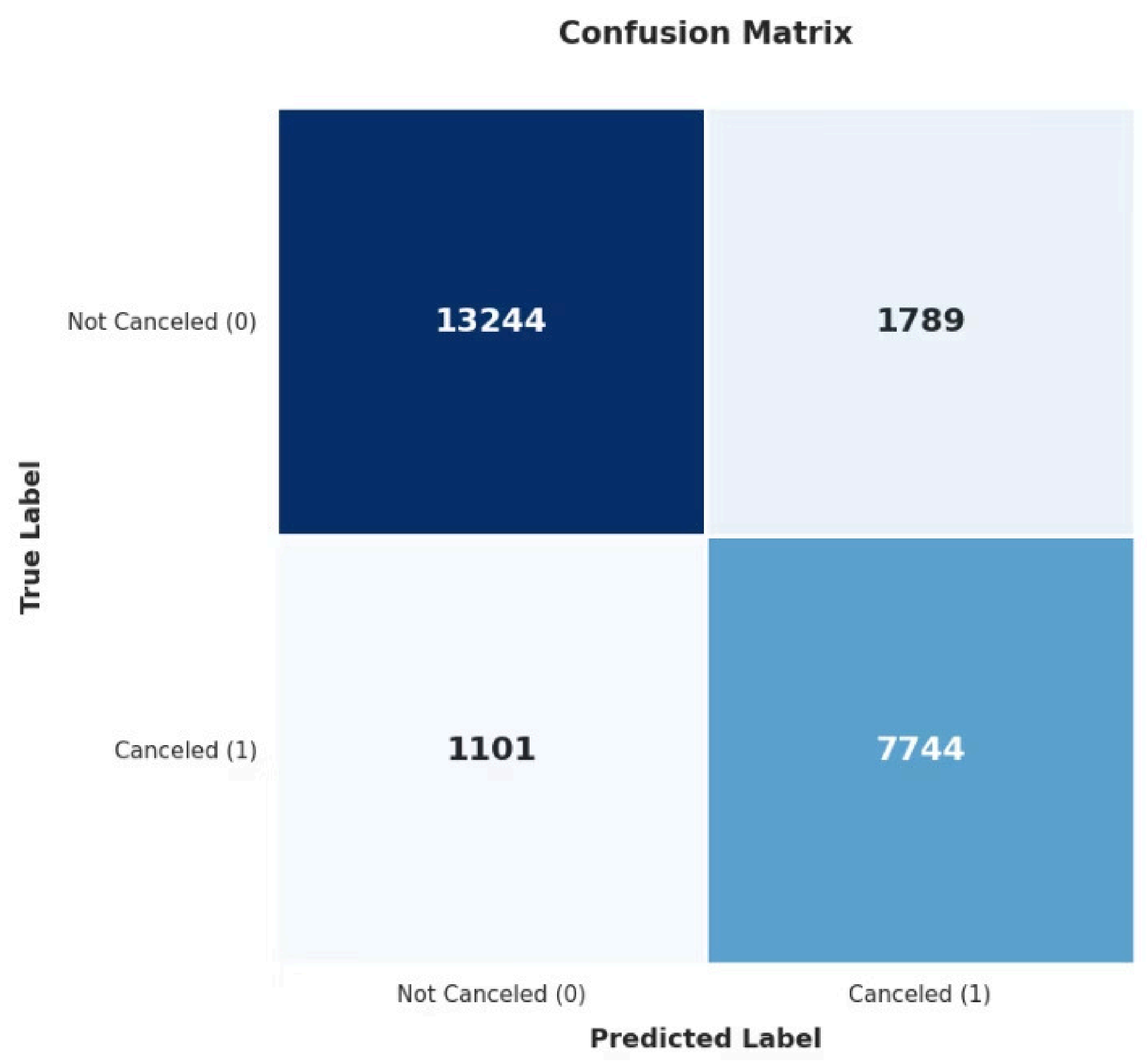


The **Pie Chart** shows that **~63% of bookings were not canceled** and **~37% were canceled**.

2. Feature Importance (Top 15)



3. Confusion Matrix

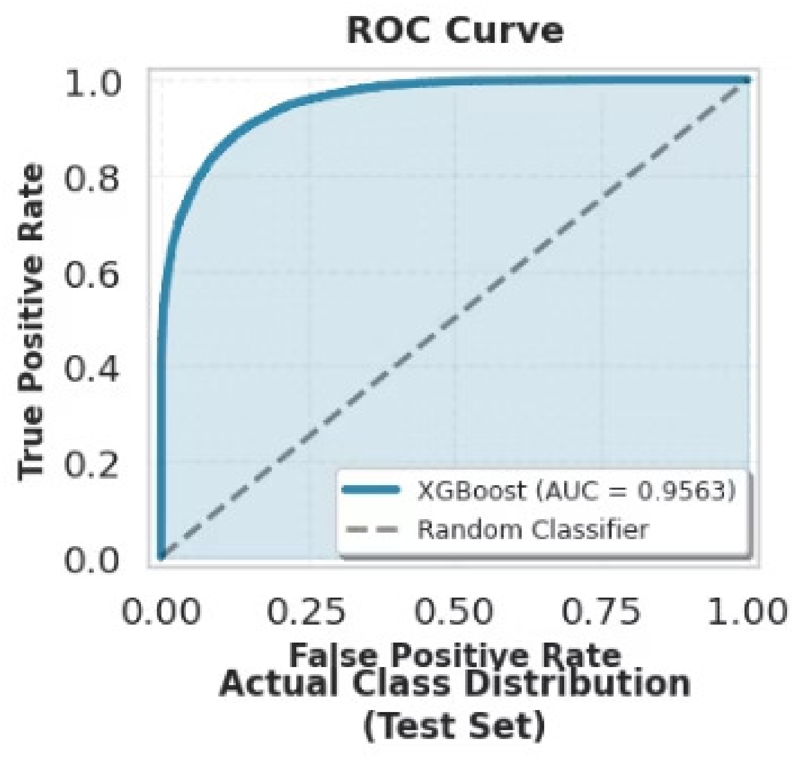


Confusion Matrix Summary (XGBoost)

- The model correctly predicted most bookings.
- **13,244** bookings were correctly classified as *not canceled.*
- **7,744** bookings were correctly classified as *canceled.*
- It made some mistakes:
  - **1,789** bookings were predicted as canceled but were not.
  - **1,101** bookings were canceled but the model missed them.

Overall, the model performs well in predicting cancellations.

4. ROC Curve



ROC Curve Summary (XGBoost)

- The model shows strong performance with a high **AUC = 0.956**.
- It can distinguish well between canceled and non-canceled bookings.
- The curve being far above the diagonal means the model is much better than random guessing.

Deployment & Usage

Local Deployment (Flask)

Technology Stack:

- Backend: Flask (Python)
- Frontend: HTML, CSS, JavaScript
- Model: XGBoost (Pickle file)

Installation:

```
# 1. Clone repository
git clone https://github.com/username/hotel-prediction.git
cd D:\Hotel_Prediction_website

# 2. Install requirements
pip install -r requirements.txt

# 3. Run application
python app.py

# 4. Open browser
http://127.0.0.1:5000
```

User Guide

Step 1: Enter Booking Details

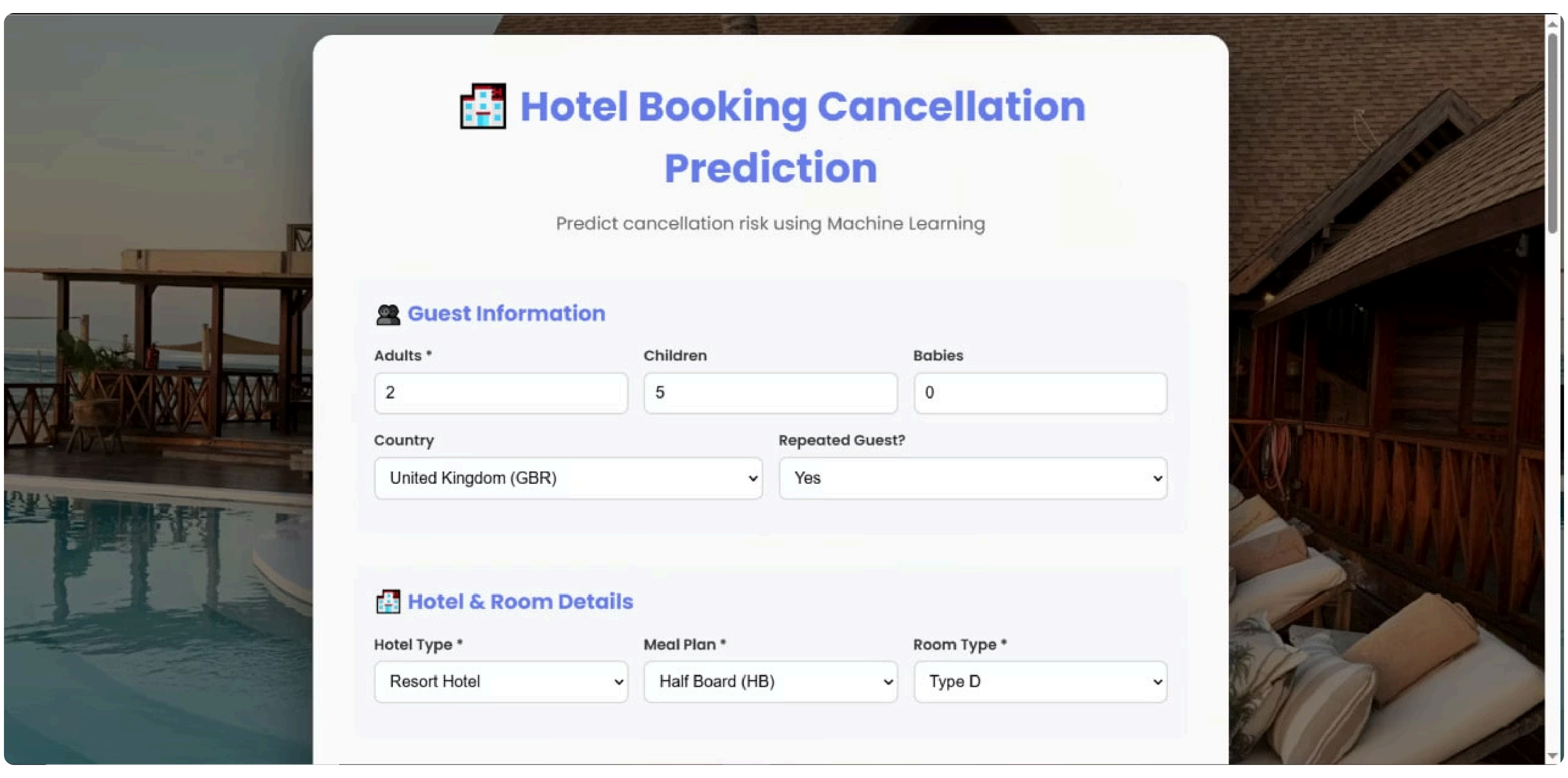
- Guest information (adults, children, babies)
- Hotel type and room details
- Booking dates and duration
- Market segment and distribution channel
- Guest history (previous bookings/cancellations)
- Financial details (ADR, deposit type)

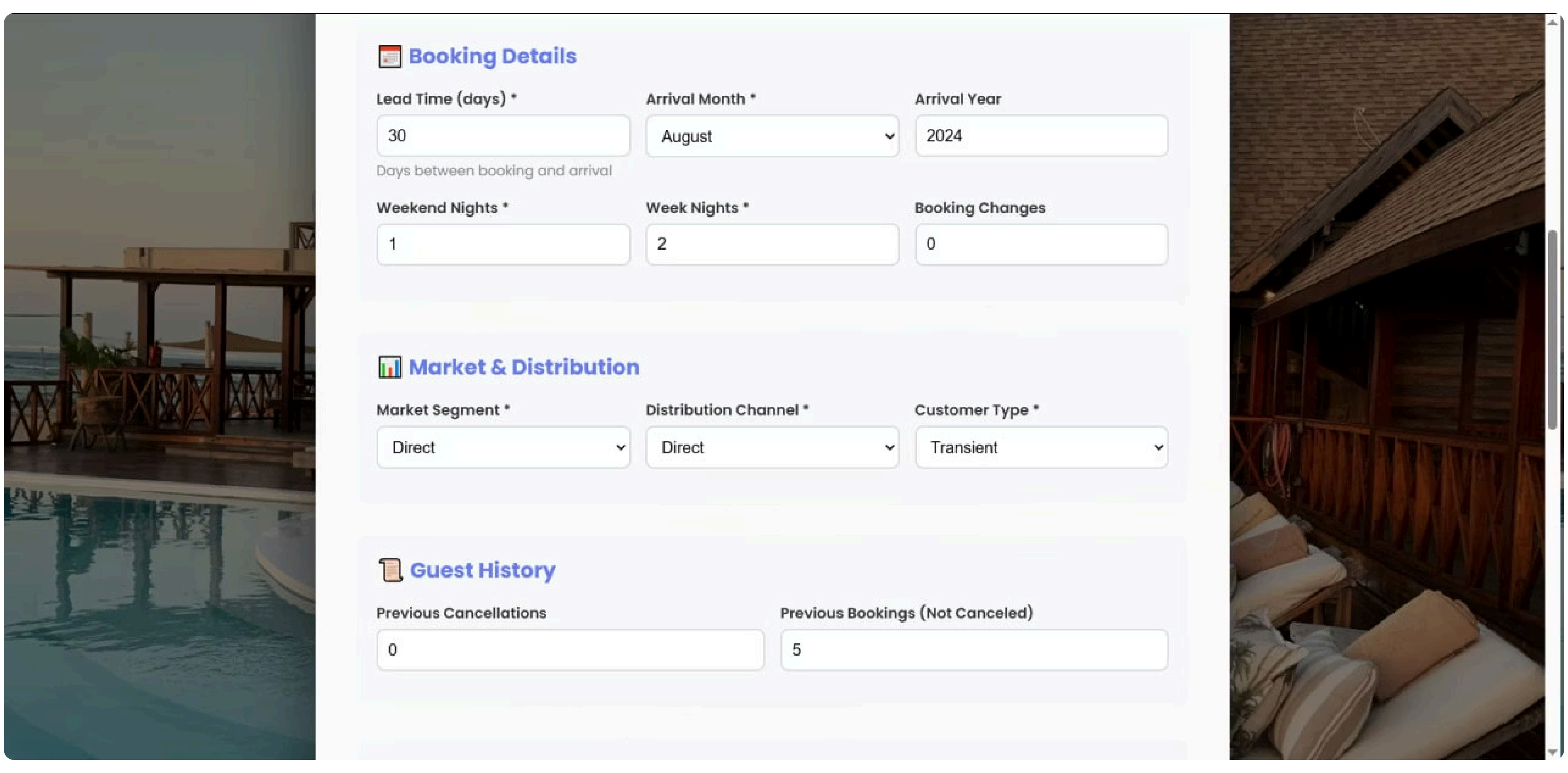
Step 2: Click "Predict Cancellation Risk"

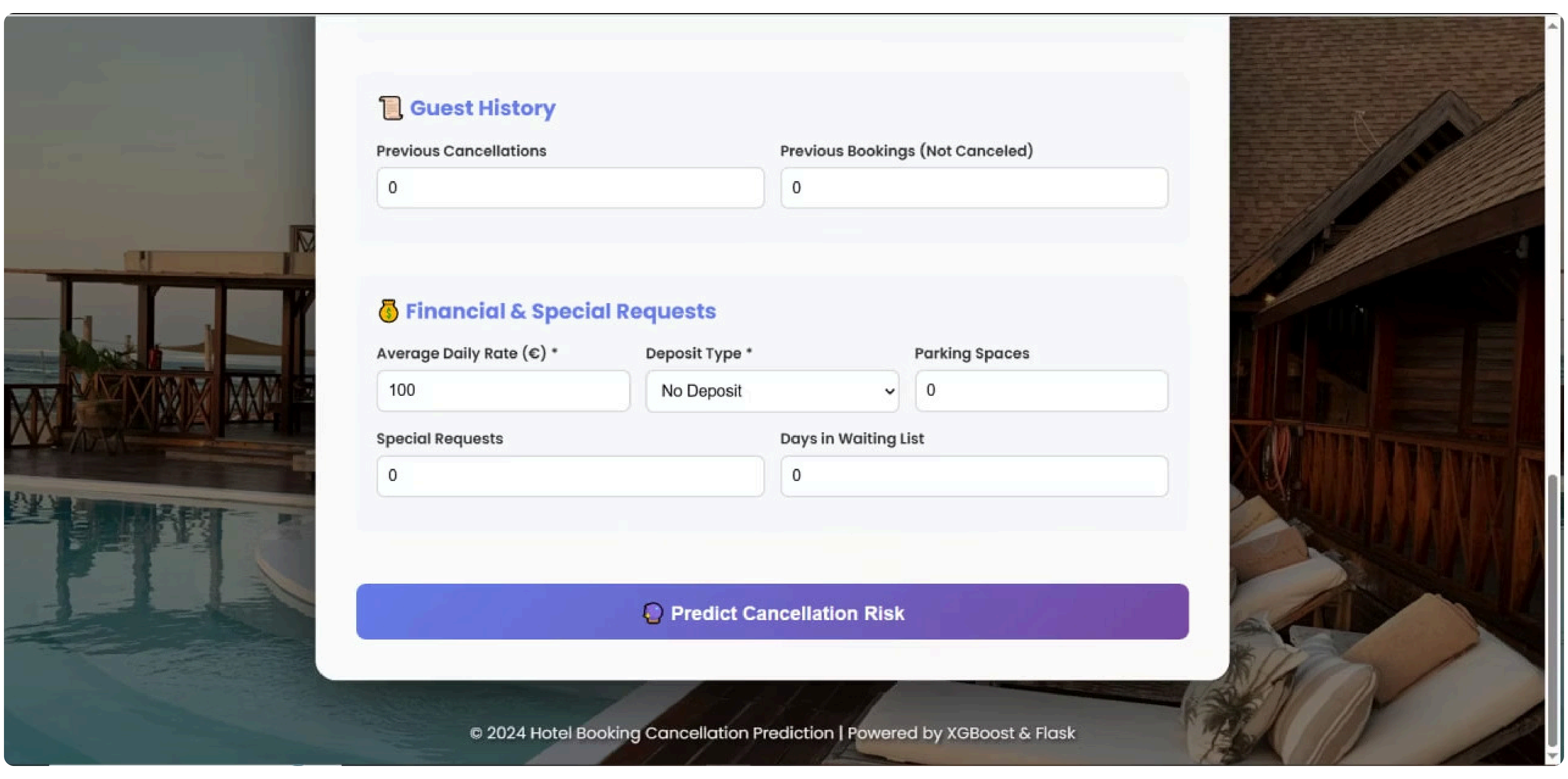
Step 3: View Results

- Risk Level: 🟢 Low / 🟡 Medium / 🔴 High
- Cancellation Probability: XX.XX%
- Recommended Actions for hotel

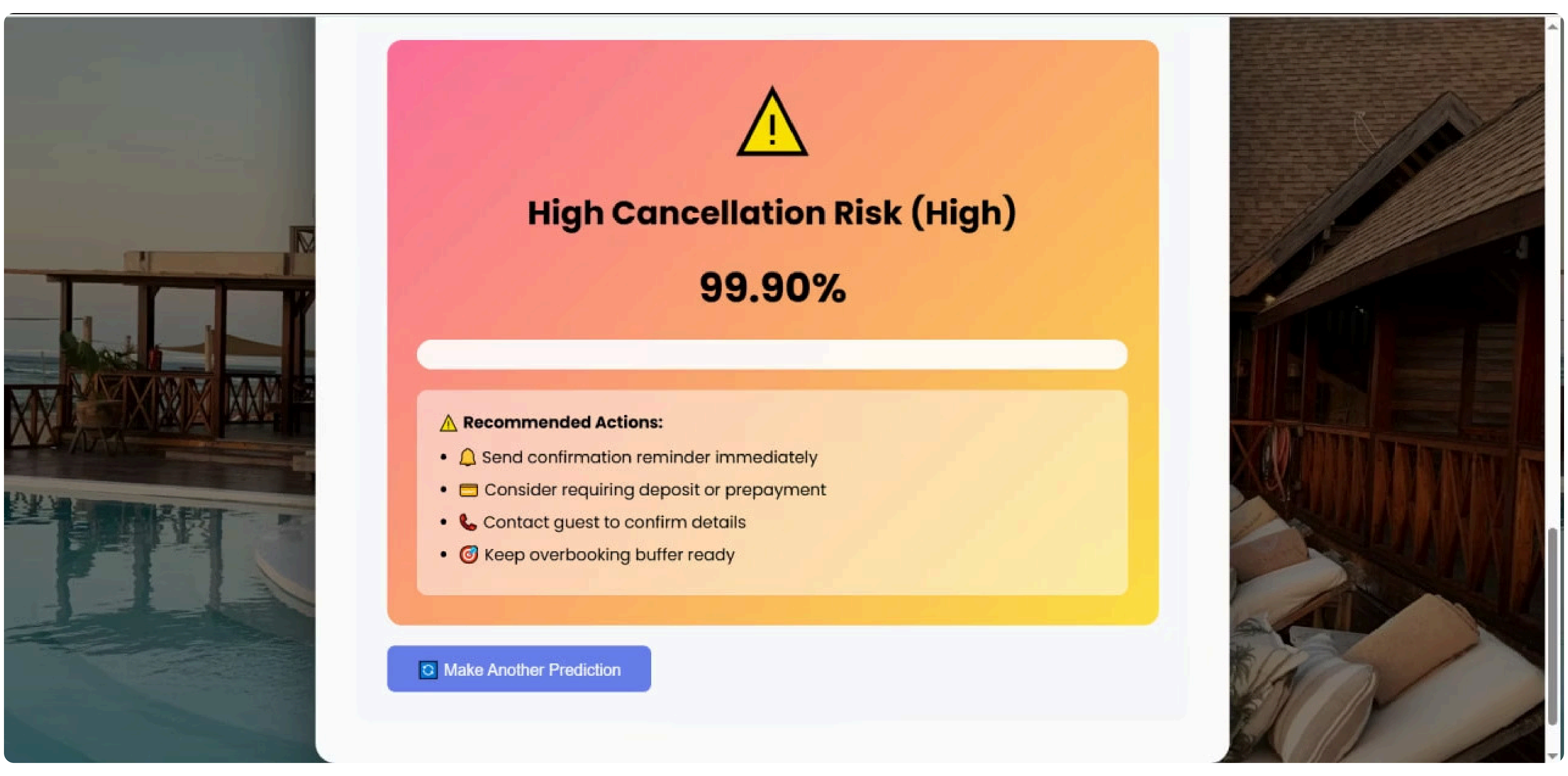
Screenshots



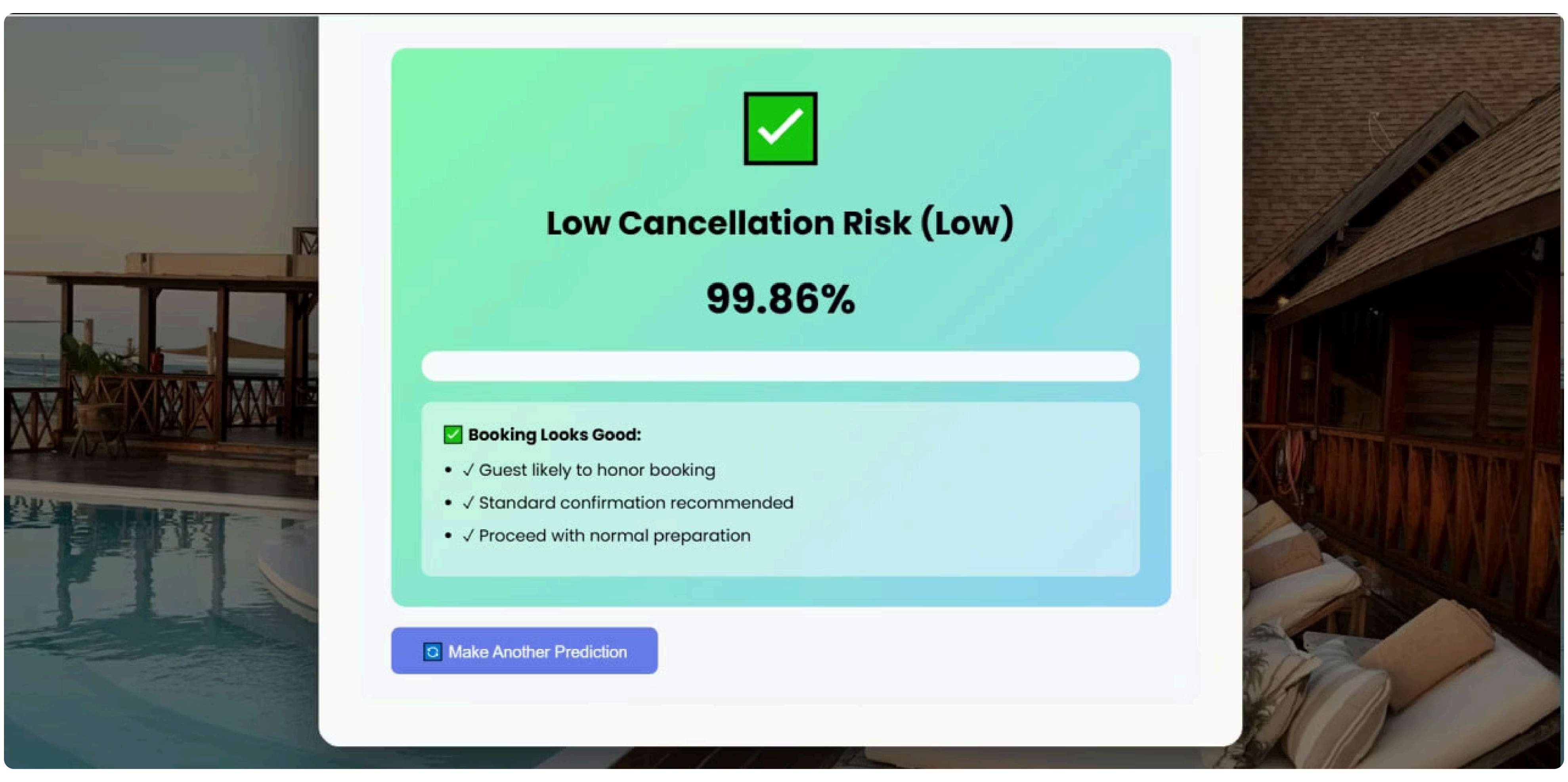




High Risk Case:



Low Risk Case:



Challenges & Lessons Learned

Challenge 1: Class Imbalance

Problem: 63:37 ratio might bias model

Solution:

- Tested SMOTE, Undersampling, Hybrid
- Selected scale_pos_weight approach
- Best balance between Precision & Recall

Challenge 2: Feature Scaling in Deployment

Problem: Model predictions were wrong (all 99% High Risk)

Cause: Training used scaled data, deployment used raw values

Solution:

- Saved StandardScaler as `scaler.pkl`
- Applied scaling to numeric features only
- Kept one-hot encoded features as 0/1

Challenge 3: High Dimensionality

Problem: 234 features after one-hot encoding

Solution:

- Feature importance analysis
- XGBoost handles high dimensions well
- Regularization prevents overfitting

Lessons Learned

1. Always save preprocessing objects (scalers, encoders)
2. Test deployment pipeline thoroughly
3. Class imbalance requires careful strategy selection
4. Recall > Precision for business-critical predictions

Conclusion

This project successfully demonstrated the use of machine learning for hotel booking cancellation prediction. The developed workflow—from data preprocessing to model evaluation—provides a strong foundation for further improvements. Future work could include:

- Hyperparameter tuning and testing alternative models to enhance prediction accuracy.
- Incorporating real-time data for dynamic prediction and hotel management.
- Deploying the model as an interactive tool or API for practical business use.

Overall, this project highlights the value of data-driven decision-making in the hospitality industry. By understanding patterns behind cancellations, hotels can improve operational efficiency, reduce revenue loss, and enhance customer satisfaction. The methodology and insights gained here can be extended to similar predictive analytics problems in the business domain.