# Model Testing Report

Date: 27 July 2025

## 1. Introduction

This report documents the testing of a machine learning API built using FastAPI and an XGBoost model. The testing was carried out in Google Colab and covered the following tasks:  
- Unit testing for core functions  
- Endpoint testing of the API  
- Load testing to simulate concurrent user requests

## 2. Files Used

- `inference\_pipeline.py`: Handles model loading, preprocessing, and prediction.  
- `api\_app.py`: Defines the FastAPI application and endpoints.  
- `test\_api.py`: Contains sample unit and API tests.  
- `best\_xgboost\_model.pkl`: Serialized trained XGBoost model.  
- `requirements.txt`: Python dependencies.

## 3. Unit Testing

The following unit tests were implemented and passed:  
- `preprocess()` function to ensure input reshaping.  
- `predict()` function to check model returns a valid numeric result.  
All tests were executed in Google Colab using standard `assert` statements.

## 4. API Endpoint Testing

API endpoints were tested using FastAPI's `TestClient` as follows:  
- `GET /health`: Checked service readiness.  
- `POST /predict`: Verified prediction on a single input.  
- `POST /predict/batch`: Verified batch prediction handling.  
All endpoints returned valid responses with expected structure and status codes (200).

## 5. Load Testing

Basic load testing was done using Python's `threading` module. Ten parallel threads were spawned to hit the `/predict` endpoint concurrently. All threads completed successfully once the model was manually injected into the FastAPI app.

## 6. Conclusion

All three tasks — unit testing, endpoint testing, and load testing — were successfully completed. The model responds as expected and the API is ready for deployment or further integration.