Proposed Solution Template

Prosperity Prognosticator: ML For Startup Success Prediction

1. Solution Title

Prosperity Prognosticator – A machine learning web application for predicting startup success using historical data and Random Forest classification.

2. Solution Description

The proposed solution is a full-stack machine learning application built using Python and Flask. It processes startup data through an ML pipeline — from raw CSV data to a trained Random Forest model — and serves predictions through an interactive web interface. Users enter startup parameters via a form, and the system returns a clear success or failure prediction.

3. Solution Components

Component	Description	File
Data Collection	**Download startup_data.csv from Kaggle**	**startup_data.csv**
Data Preprocessing	**Cleaning, encoding, feature selection**	**notebook (.ipynb)**
Exploratory Data Analysis	**Statistical and visual analysis of dataset**	**notebook (.ipynb)**
Model Training	**Train 6 ML algorithms and compare accuracy**	**notebook (.ipynb)**
Hyperparameter Tuning	**GridSearchCV for Random Forest optimization**	**notebook (.ipynb)**
Model Saving	**Save best model using Pickle**	**random_forest_model.pkl**
Web Backend	**Flask routing and prediction logic**	**app.py**
Prediction Form	**Dynamic form for entering startup metrics**	**index.html**
Result Page	**Display success/failure prediction result**	**result.html**

4. Proposed Workflow

- Step 1: Download and explore the Kaggle startup dataset
- Step 2: Preprocess data — handle missing values, encode categoricals
- Step 3: Perform Exploratory Data Analysis with charts
- Step 4: Train and compare 6 ML classification algorithms
- Step 5: Apply hyperparameter tuning to improve the best model
- Step 6: Identify and select top features for final model
- Step 7: Save the optimized model as a .pkl file
- Step 8: Build Flask app with routing and HTML templates
- Step 9: Integrate saved model into Flask prediction endpoint
- Step 10: Test application end-to-end and deploy locally

5. Expected Outcomes

- A working ML web application predicting startup success with 80%+ accuracy
- Clean, documented Jupyter Notebook covering full ML pipeline
- Flask web application with home, prediction form, and result pages
- Saved Random Forest model (.pkl) for deployment