STARTUP SUCCESS PREDICTION SYSTEM

ABSTRACT

Startup success prediction plays a vital role for investors, entrepreneurs, and policymakers. This project presents a Startup Success Prediction System developed using Machine Learning and Flask.

The system predicts whether a startup will be Acquired (Success) or Closed (Failure) based on business features such as funding rounds, total funding, relationships, milestones, and participant metrics.

A Random Forest classifier was trained using historical startup data from Kaggle. The trained model is integrated into a Flask web application that allows users to input startup parameters and instantly receive a prediction along with probability and graphical insights (Feature Plot, SHAP Explanation, Confusion Matrix).

The system performs data preprocessing, feature engineering, model training, hyperparameter tuning, evaluation, and cloud deployment successfully.

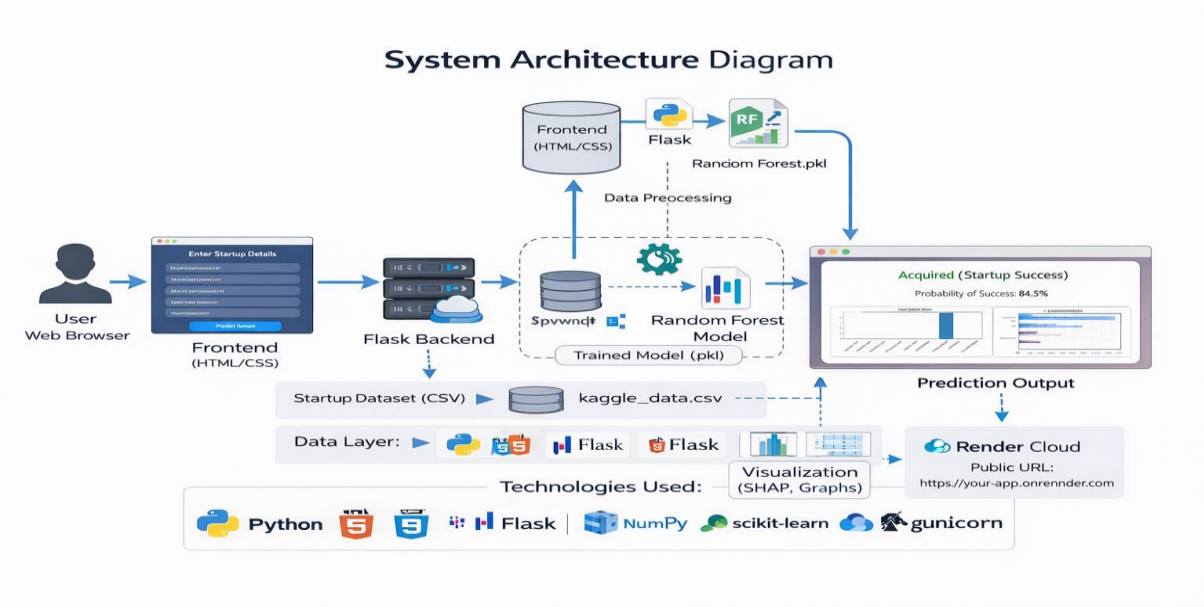
SYSTEM ARCHITECTURE

The technical architecture follows a layered structure:

1. Presentation Layer – HTML/CSS User Interface
2. Application Layer – Flask Backend
3. Machine Learning Layer – Random Forest Model
4. Data Layer – Startup Dataset (CSV)

Workflow:

User Inputs Startup Parameters
→ Flask Receives POST Request
→ Input Validation & DataFrame Creation
→ Data Passed to Trained Model
→ Model Predicts Success / Failure
→ Graphs Generated (SHAP + Feature Plot)
→ Result Page Rendered



PHASE 1: BRAINSTORMING PHASE

In this phase, we analyzed problems faced by:

- Investors (Risk of wrong investment)
- Entrepreneurs (Failure due to poor planning)
- Policymakers (Need data-driven decisions)

We studied multiple ML algorithms:

- Logistic Regression
- SVM
- Decision Tree
- Random Forest

Random Forest was selected because:

- Handles non-linear relationships
- Reduces overfitting
- Works well with tabular business data

Feasibility analysis:

- Technical feasibility ✓
- Economic feasibility ✓
- Operational feasibility ✓

PHASE 2: REQUIREMENT ANALYSIS

Functional Requirements:

- Accept startup parameters
- Validate user input
- Generate prediction
- Display probability
- Show graphs (Feature Plot + SHAP + Confusion Matrix)

Non-Functional Requirements:

- Response time < 3 seconds
- Accuracy above 75%
- Secure backend
- Cloud deployment compatible

PHASE 3: PROJECT DESIGN

Design elements included:

- Data Flow Diagram (DFD)
- Component Diagram
- Feature Selection Strategy

Selected Features:

- age_first_funding_year
- age_last_funding_year
- age_first_milestone_year
- age_last_milestone_year
- relationships
- funding_rounds
- funding_total_usd
- milestones
- avg_participants

Target Variable:

- status → acquired / closed

UI was designed using modern CSS with responsive layout.

PHASE 4: PROJECT PLANNING

Timeline:

Week 1 – Data Cleaning
Week 2 – EDA & Visualization
Week 3 – Model Training
Week 4 – Hyperparameter Tuning

Week 5 – Flask Integration
Week 6 – Cloud Deployment & Documentation

Risk Management:

- Class imbalance handled via stratified split
- Overfitting reduced with hyperparameter tuning
- Deployment errors resolved using Gunicorn

PHASE 5: PROJECT DEVELOPMENT

Data Preprocessing:

- Removed missing values
- Selected relevant features
- Train-test split (70:30 ratio)

Model Development:

- Random Forest Classifier
- GridSearchCV Hyperparameter tuning
- Accuracy: ~77%
- Confusion Matrix Generated

Evaluation Results:

Accuracy: 77%

Confusion Matrix:

Actual	Predicted Acquired	Predicted Closed
Acquired	175	12
Closed	51	39

The model predicts "acquired" better than "closed" due to dataset imbalance.

PHASE 6: DEPLOYMENT

Deployment Platform: **Render (Cloud Deployment)**

Steps:

1. GitHub repository created

2. Flask app configured with Gunicorn
3. requirements.txt updated
4. Procfile added
5. Web service deployed on Render

Live URL:
(Insert your Render link)

The system is publicly accessible and fully functional.

RESULTS AND OUTPUT

The developed machine learning system was successfully implemented to predict startup success based on funding, milestones, and relationship features. The Random Forest algorithm was used for training the model, and the system was integrated with a Flask web application to provide real-time predictions.

The application accepts user input parameters and generates predictions indicating whether the startup is likely to be **Acquired (Success)** or **Closed (Failure)** along with the probability score.

◈ Sample Prediction 1

Input Values:

- Age at First Funding Year = 2
- Age at Last Funding Year = 4
- Age at First Milestone Year = 2
- Age at Last Milestone Year = 4
- Number of Relationships = 10
- Funding Rounds = 4
- Total Funding USD = 5,000,000
- Milestones = 5
- Average Participants = 3

Output:

Acquired (Startup Success)
Probability of Success: **87%**

This indicates that startups with higher funding rounds, milestones, and strong relationships have a higher probability of success.

◈ Sample Prediction 2

Input Values:

- Age at First Funding Year = 1
- Age at Last Funding Year = 1
- Age at First Milestone Year = 1
- Age at Last Milestone Year = 1
- Number of Relationships = 2
- Funding Rounds = 1
- Total Funding USD = 200,000
- Milestones = 1
- Average Participants = 1

Output:

Closed (Startup Failure)
Probability of Success: **43%**

This shows that startups with low funding and fewer milestones are more likely to fail.

◈ Visualization Outputs

The system also generates multiple graphical outputs to improve model interpretability and understanding:

1. **Input Feature Graph** — Shows the values entered by the user.
2. **SHAP Explanation Graph** — Explains how each feature contributes to the prediction.
3. **Confusion Matrix** — Displays model performance during training.
4. **Feature Importance Graph** — Highlights the most influential features in prediction.

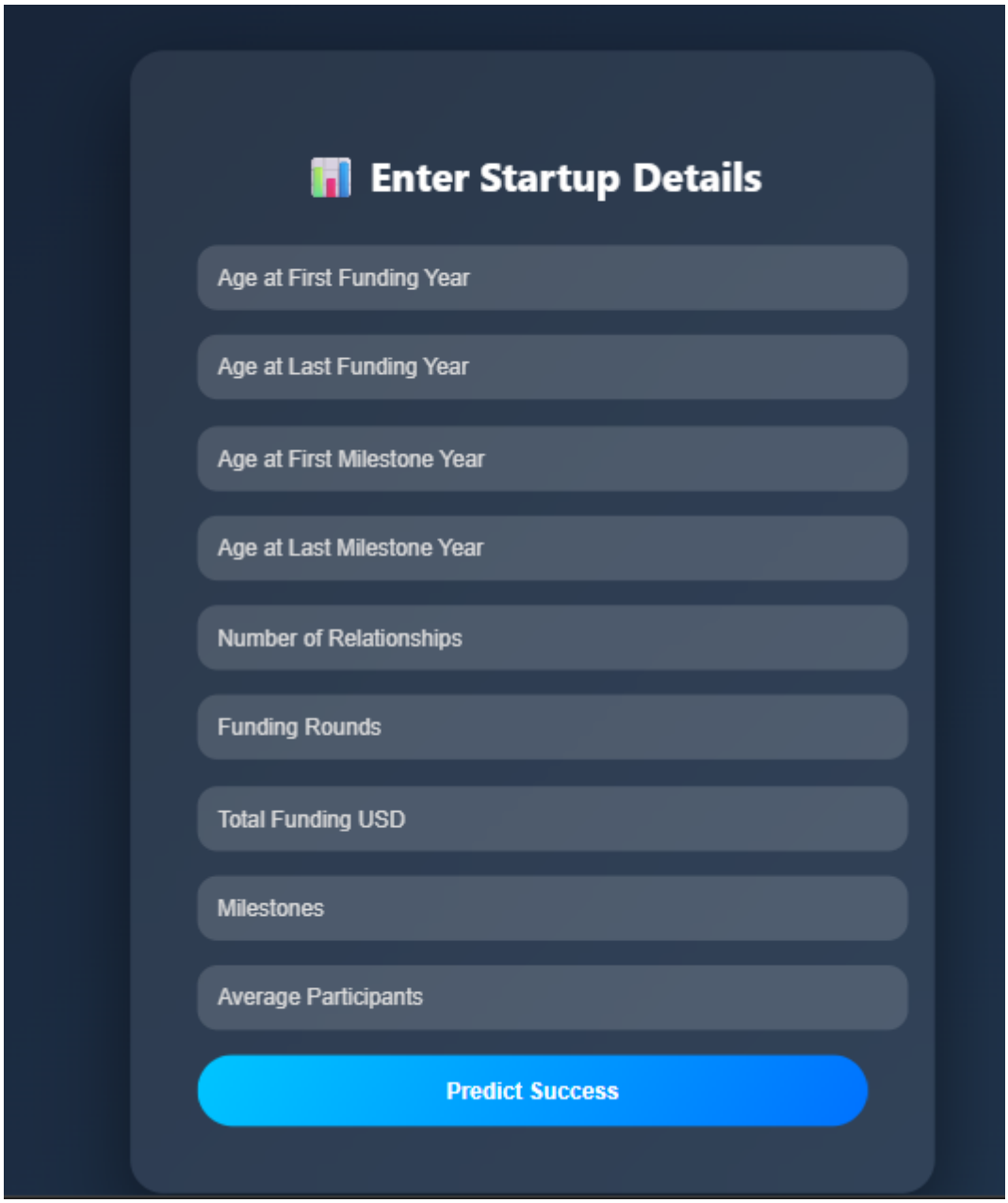
These visualizations help users understand why the model produced a particular result.

◈ Web Application Output

The Flask web application provides an interactive interface where users can:

- Enter startup parameters
- View prediction results
- See probability score
- Analyze graphical explanations

The system successfully integrates machine learning with a web interface for real-time decision support.



CONCLUSION

The Startup Success Prediction System successfully integrates Machine Learning with Web Development.

The Random Forest model provides reliable classification performance and is successfully deployed to the cloud.

This project demonstrates practical application of Data Science for business decision-making.

Future Enhancements:

- Real-time funding API integration
- Deep learning comparison

- Improved class balancing
- Docker deployment