Project Planning Phase - III
Technology Stack (Architecture & Stack)

Date	18 October 2023
Team ID	PNT2023TMID592444
Project Name	Anticipating Business Bankruptcy
Maximum Marks	4 Marks

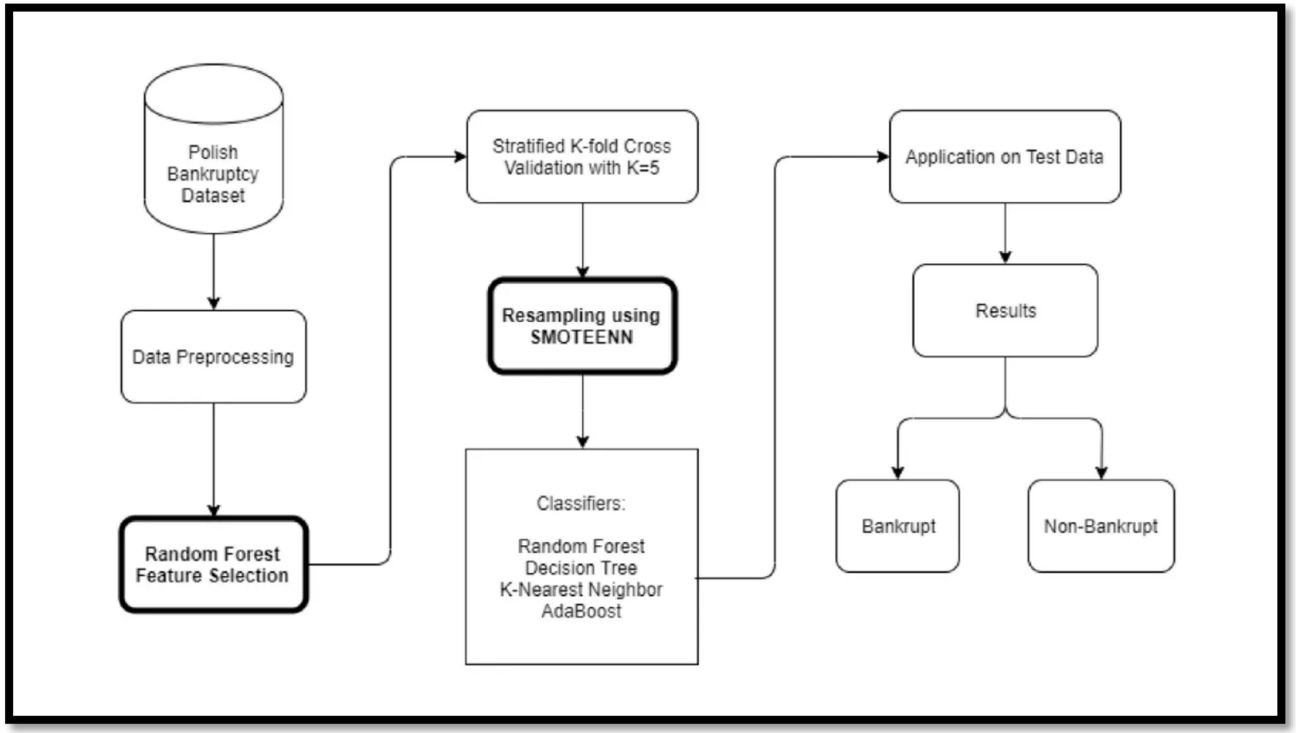
Polish Bankruptcy Dataset
Data Preprocessing
Random Forest Feature Selection
Stratified K-fold Cross Validation with K=5
Resampling using SMOTEENN
Classifiers: Random Forest Decision Tree K-Nearest Neighbor AdaBoost
Application on Test Data
Results
Bankrupt
Non-Bankrupt

Table-1 : Components & Technologies:

S. No.	Component	Description	Technology
1.	User Interface	How client interacts with the application (E.g., Website, Mobile App)	HTML, CSS, JavaScript, Node.js
2.	Application Logic - 1	Logic for a process in the application	Python
3.	Data Collection and Storage	Gather relevant financial and non-financial data, such as balance sheets, income statements, cash flow statements.	MySQL, NoSQL and Local storage
4.	Cloud Database	Database Service on Cloud	AWS, Azure, or GCP
5.	Financial Data APIs	APIs that provide real-time financial data, credit scores, and market trends	Bloomberg, Alpha Vantage, or Quandl
6.	Data Enrichment APIs	APIs that offer data enrichment for company profiles and financial information	Clearbit, Dun & Bradstreet
7.	Machine Learning Model	Build, train, and deploy machine learning models for bankruptcy prediction.	Scikit-Learn, TensorFlow, or PyTorch
8.	Infrastructure (Server / Cloud)	Application Deployment on Local System and Cloud	Local, Cloud Foundry, Kubernetes

Table-2: Application Characteristics:

S. No.	Characteristics	Description	Technology
1.	Open-Source Frameworks	Open-source frameworks used	Scikit-Learn, Apache Spark, Node.js
2.	Security Implementations	Implement robust security measures to protect sensitive financial data, user information, and the integrity of the application.	Data Encryption, access control policies, and compliance auditing tools
3.	Scalable Architecture	The application should be able to scale to handle a growing volume of data and user requests, especially during peak periods.	Cloud platforms like AWS, Azure, or GCP for infrastructure, AWS Elastic Container Service (ECS) for containerization, and AWS Auto Scaling for dynamic resource allocation.
4.	Availability	Ensuring minimal downtime and high availability. Continuously monitoring the health and performance of the system and receive alerts when issues arise.	Implementing redundant systems and failover mechanisms, leveraging open-source solutions like Keepalived or cloud provider's HA services. Open-source monitoring tools like Prometheus, Grafana, and cloud-native monitoring services.
5.	Performance	Optimize performance by implementing caching strategies for frequently accessed data. Ensuring that database queries are optimized for performance. Accelerate content delivery by leveraging CDN services.	Use in-memory data stores like Redis or Memcached. Utilize open-source database management systems like MySQL, PostgreSQL, or NoSQL databases like MongoDB. Cloud-based CDN providers like Cloudflare, Akamai, or AWS CloudFront.