**SYSTEM DESIGN**

**SYSTEM ARCHITECTURE:**

The system architecture defines the overall structure of the hybrid fraud detection model and its interaction with users and subsystems. It comprises the following major components:

1. **Data Source** – The raw dataset containing credit card transaction records is collected from public repositories such as Kaggle.
2. **Data Preprocessing Module** – Handles cleaning, normalization, feature selection, and class balancing using SMOTE.
3. **Hybrid Classification Model** – Integrates multiple machine learning algorithms, including Logistic Regression, Random Forest, SVM, and XGBoost. These classifiers are trained and combined through an ensemble voting mechanism.
4. **Prediction Module** – Performs both single-transaction and batch predictions using the trained hybrid model.
5. **Flask Web Interface** – Serves as the user interaction layer that allows uploading transaction data and displaying results visually.
6. **Result Visualization** – Generates confusion matrices, ROC curves, and performance summaries for model evaluation.

**System Architecture Diagram**

**DATA FLOW DIAGRAM (DFD):**

The Data Flow Diagram (DFD) depicts how data moves through the system, from input to final output.

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

**DFD Level 0 – Context Diagram:**

At the highest level, the system interacts with an external **User** who provides input data (transaction details or dataset). The system processes the input using the hybrid model and returns the result (fraudulent or legitimate).

**DFD Level 1 – Detailed Data Flow:**

**UML DIAGRAMS**

The Use Case Diagram shows the interaction between the **User** and the **Fraud Detection System**.

**Actors:**

* User (Admin / Analyst)

**Use Cases:**

* Upload Dataset
* Train Model
* View Accuracy Results
* Perform Single Prediction
* Perform Batch Prediction
* View Visual Reports (Confusion Matrix, ROC)

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

**GOALS:**

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

**SEQUENCE DIAGRAM**

The Sequence Diagram describes the interaction between the system components during prediction.

User → Web Interface: Upload dataset / Input data

Web Interface → Model Controller: Send data for preprocessing

Model Controller → ML Engine: Run hybrid model (RF, SVM, LR, XGB)

ML Engine → Model Controller: Return prediction

Model Controller → Web Interface: Display prediction result

Web Interface → User: Show fraud status and graphs