are approaches for providing a strong and efficient data communication connection that should be included in the data processing [73]. To avoid network collateral damage, the data received from security-critical sectors must be treated with the utmost priority. Data priority must be taken very seriously if there is to be no damage to other parts of the network.

- **Latency:** Latency in V2X communication can be caused by several things, such as what information to gather and filter, what data to analyze, and what to send and receive. Therefore, all issues associated with communication delays in V2X must be addressed so that safety and security-critical situations may be managed in real-time [74].

B. SECURITY REQUIREMENT

- **Authentication:** For reasons of security, it is crucial to ensure that messages are coming from a trusted source. Vehicle identification is crucial in IoV due to the high mobility and transient nature of vehicles. Each vehicle must be verified before it can begin sending and receiving data once connected to the network. To prove authenticity, you could use a private key and a certificate of the vehicle, or you could use a fake name.
- **Authorization:** Access control, or authorization, is the process of deciding whether any entity in V2X is permitted to access resources or other entities, such as read or write data, execute programmes, or operate actuators. Authorization also involves blocking or revoking access, particularly for malevolent entities . There is a need. In smart city networks, a secure authentication method for municipal-based V2X communication is needed.
- **Data security:** Data security is paramount, especially when it comes to keeping sensitive information from falling into the wrong hands. Both the data and the resources themselves need to be shielded from prying eyes. To keep messages private, they need to be both signed and encrypted, not just signed.
- **Trust mechanism:** Secure and controlled collaborative communication of V2X in smart cities requires a trustworthy mechanism to build automated trust based on the historical result same as in human nature. This mechanism requires issuing a unique identity and behaviour to each device in heterogeneous IoT networks

V. OVERVIEW OF THE PROPOSED FRAMEWORK

The proposed framework is composed with the help of a layered approach to ensure modularity, scalability, and seamless integration of the smart city v2x collaborative ecosystem . The first layer is the V2X Collaborative Layer, which comprises a heterogeneous set of IoT-enabled nodes, including vehicular nodes (e.g., buses, cars, and service vehicles) and smart citizen nodes (e.g., passenger mobile devices, public kiosks, and connected wearables). This layer facilitates bidirectional data exchange within the V2X ecosystem, enabling real-time communication between vehicles, infrastructure, and end users. The second layer is the Dynamic Smart Contract Layer, which is responsible for the creation and deployment of dynamic smart contracts based on contextual input data. This layer also integrates with a multichain blockchain platform to ensure decentralized, secure, and transparent execution of smart contracts across various V2X services.Figure 2 shows Proposed AI-Driven Smart Contract Framework For Decentralized V2X communication.In the next section, we provide a detailed discussion of the layered approach used in the proposed framework.

VI. PROPOSED SYSTEM ARCHITECTURE

The proposed framework is implemented with a layered architectural approach to provide flexibility, adaptability, and scalability to the smart city transportation ecosystem. This layered architectural approach enables the seamless integration of V2X communications, context-aware data processing, and automated smart contract administration. This section fully explains the proposed system architecture, highlighting each layer's responsibility and contribution to the overall structure. In this section, we discuss the key submodules within each layer, outlining their functional responsibilities, interactions, and the algorithmic approaches employed to support real-time decision-making and autonomous operations.

A. V2X COLLABORATIVE LAYER

The V2X Collaborative Layer serves as the foundation for the proposed framework. This layer offers bidirectional, real-time networking throughout the V2X smart city ecosystem. We Used General Transit Feed Specification (GTFS) APIs [75] In order to incorporate this layer into public transportation data formats. GTFS APIs provide controlled and standardized datasets of vehicle-to-smart-citizen data, which include timetables, routes, stop locations, and fare information. In this layer, the representations of vehicular nodes and smart citizen nodes are defined as follows:

1) Vehicular Nodes are represented by live bus instances extracted from GTFS APIs have real-time data with features such as trip updates, positions, and prices, etc. In this research article we primarily focused on vehicle positions, fare prices, and passenger count from the GTFS real-time data to mimic the smart behavior of these features and effectively inorder to implement the vehicular nodes within our simulated V2X environment.
2) We implemented the X nodes as client nodes, representing smart citizens within the ecosystem. These nodes receive data from the subscribed GTFS services, allowing them to access features such as stop updates, route information, and trip schedules, thereby simulating real-time interaction between smart citizens and the public transport system in the V2X environment

After establishing the collaborative interaction pipeline between vehicular nodes and smart citizen (X) nodes, there

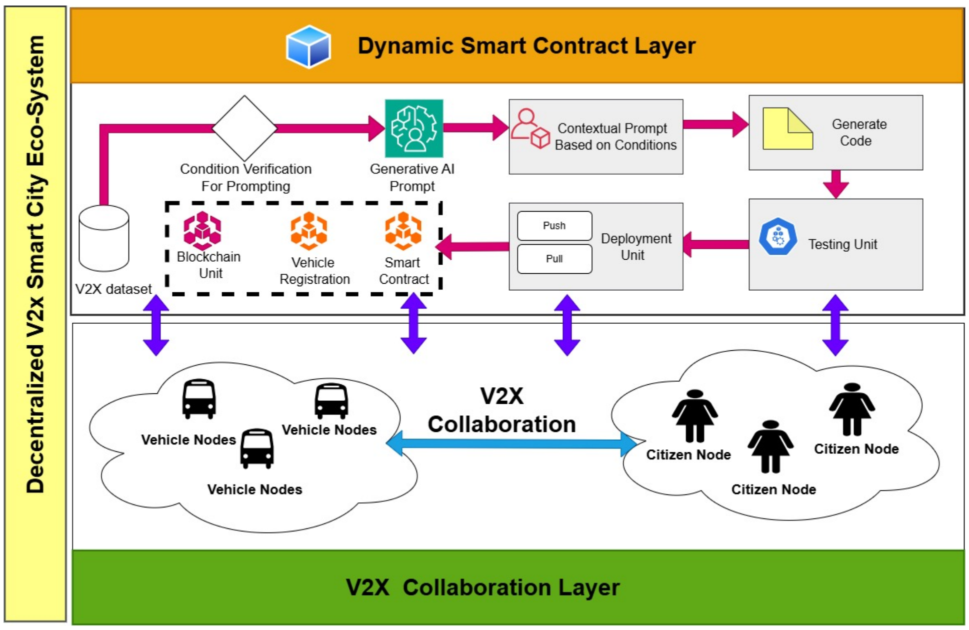

FIGURE 2: Proposed AI-Driven Smart Contract Framework For Decentralized V2X communication

arises a crucial need to enforce decentralized rules on fare pricing and ensure secure communication within the V2X data exchange process. While the collaborative layer enables real-time data sharing and interaction, it must be governed by a trusted, tamper-resistant mechanism to guarantee fairness, transparency, and reliability across all transactions.To address this, the proposed framework integrates a Dynamic Smart Contract Layer that autonomously manages fare policies through decentralized execution.

B. DYNAMIC SMART CONTRACT LAYER

Dynamic Smart Contract layer is responsible to enable secure data exchange and dynamic V2X rule enforcement through smart contracts, the proposed framework integrates blockchain technology using the Multichain platform. Since Multichain does not support native smart contract functionality, we incorporated a mechanism for dynamic and decentralized smart contract creation and deployment using Python code, powered by Generative AI (GPT-4).Multichain provides several security features, one of which is the stream functionality. This feature is primarily used for recording and validating key security attributes related to smart contract creation and deployment, ensuring transparency, traceability, and tamper-resistant logging within the blockchain network. is used for recording and validating key security attributes related to smart contract creation and deployment through its stream feature.

1) Dynamic V2x Rule Enforcement

In the proposed framework, smart contracts are not fixed code modules but are instead developed on the basis of contextual data captured from the V2X environment . They possess the capability to adapt their internal logic during real-time operations through the integration of an AI component that is responsible for predicting the fare prices and the generative AI system that automates the code modification and deployment process. Initially, a dummy template of the smart contract is created and registered on the Multichain blockchain platform to define a fixed input-output interface and establish a base structure for future dynamic logic updates. This template acts as a placeholder and a deployment-ready shell that can be modified on demand by the system.When specific contextual triggers are observed such as low occupancy, high vehicle availability, or off-peak hours the FareAI model generates a prediction of the optimal discount value d using a set of real-time inputs d (time of day, passenger count, number of active vehicles, weather). This predictive model is built using supervised regression techniques to learn a function $f(x)$ that maps the current operating context to an appropriate pricing policy.

A key point in this framework is that the predicted discount value $d = f(x)$ is not used in isolation to change the smart contract logic. Instead, it is embedded into a context-rich prompt, which also includes supporting real-time features and AI-derived explanatory factors (e.g., “low ridership,” “off-peak,” “rainy weather”). This prompt is sent to the Generative AI model (e.g., GPT-4), which acts not as a decision-maker, but as a code generation engine, governed by structured and explainable inputs. This ensures that the internal logic update is not only automated but also safe, context-aware, and consistent with operational goals. The dynamic smart contract logic update can be conceptually represented as:

$$\texttt{UpdateFare}(r, t, d) \Rightarrow \texttt{ApplyDiscount}(r,\, t,\, d = f(x))$$

Where:

- $\texttt{UpdateFare}(r, t, d)$: The trigger event to update the smart contract logic.
- r: Route identifier (e.g., R1, R2).
- t: Time period (e.g., Morning, Evening).
- d: Discount percentage.
- $f(x)$: AI function mapping contextual features to d.
- x: Features including:
  - Hour of the day
  - Passenger count
  - Active vehicle count
  - Weather condition
  - Weekend/weekday status
- $\texttt{ApplyDiscount}$: The new smart contract logic applied.

Here, the generative AI produces the logic to apply the updated fare rule, modifying only the relevant internal structure of the contract while keeping the outer interface stable. To prevent invalid or unsafe changes, the generated code is validated before deployment using internal testing and domain constraints.Finally, the updated contract is deployed back to the system, and all metadata related to the change including discount value, timestamp, hash, route ID, and update context is immutably recorded on the Multichain blockchain using its stream functionality. This logging mechanism provides traceability, tamper resistance, and accountability for all contract updates. Algorithm 1 shows the detailed algorithmic workflow for this AI-guided smart contract adaptation process.

Algorithm 1 Dynamic Smart Contract

```
1:  Init V2X stream, FareAI model, GenAI module
2:  while system active do
3:      Read input features:
        $x_1$ = time, $x_2$ = passenger count,
        $x_3$ = vehicle count, $x_4$ = weather, $x_5$ = day type
4:      Predict discount $d = f(x_1, x_2, x_3, x_4, x_5)$ using FareAI
5:      if $d$ indicates update needed then
6:          Generate reasons $e$ (e.g., low riders, off-peak)
7:          Build prompt $p$ using route $r$, time $x_1$, discount $d$, reasons $e$
8:          Send $p$ to GenAI
9:          Receive new contract logic $c$
10:         Validate $c$ for syntax and policy
11:         if valid then
12:             Deploy $c$ to contract layer
13:             Log ($r$, $x_1$, $d$, hash) to blockchain
14:         end if
15:     end if
16:     Wait for next data cycle
17: end while
```

2) Secure Data Exchange through Blockchain

In the proposed V2X smart city ecosystem, we integrate the permissioned blockchain multichain platform. This integration provides the core security features to the communication mechanism between V2x nodes by enabling decentralized identity management and cryptographic message validation.Through the use of public-private key pairs and digital signatures, each participating node of a vehicle (V) and citizen (x) can securely authenticate, verify, and exchange information with blockchain using the blockchain mechanism. The Multichain platform supports fine-grained access control and append-only data streams, ensuring that all transactions, including message exchanges and smart contract creation and deployment, remain tamper-resistant, auditable, and secure by design. To initialize the system and enable trusted data exchange across the V2X ecosystem, the message exchange pipeline begins with the V2X node registration process and the smart contract for security verification purposes. Algorithmm 2 shows the implementation logic of the registration process.

To enable secure and trustworthy communication in the V2X environment, the first essential step is the blockchain-based registration of all V2X nodes. In our framework, it is assumed that all legitimate V and X nodes are pre-configured with knowledge of the private blockchain's identity or digital signature. This pre-trusted association allows V2X nodes to send registration requests directly to the blockchain platform. Figure3 illustrates the blockchain-based registration process for vehicle nodes, citizen nodes, and smart contracts. To manage registration and security attributes efficiently, the framework utilizes three dedicated Multichain streams, each responsible for handling different types of security-related data:

1) **Vehicular Registration Stream:** Responsible for managing the registration of vehicle nodes. It stores and verifies security tokens, vehicle metadata (e.g., route ID, operator ID), and other attributes required for authenticated participation.
2) **X-Registration Stream:** Handles the registration of smart citizen nodes. It maintains registration tokens and identification metadata associated with mobile devices, kiosks, or wearable devices that participate in the V2X communication.
3) **Smart Contract Stream:** Maintains records of the latest deployed smart contracts, including logic hashes, timestamps, deployment triggers, and associated policy metadata. This ensures transparency and immutability in contract versioning and execution history.

Together, these streams provide a secure, scalable, and auditable mechanism for managing trust and identity across the decentralized V2X smart city ecosystem.Figure 3 illustrates the V2X registration workflow, where both Vehicle(V) and Citizen(X) nodes initiate registration requests by incorporating the known shared keys in the request message and encrypted it with the public key of the private blockchain as shown in equation1. This ensures that only legitimate nodes can participate in the network.

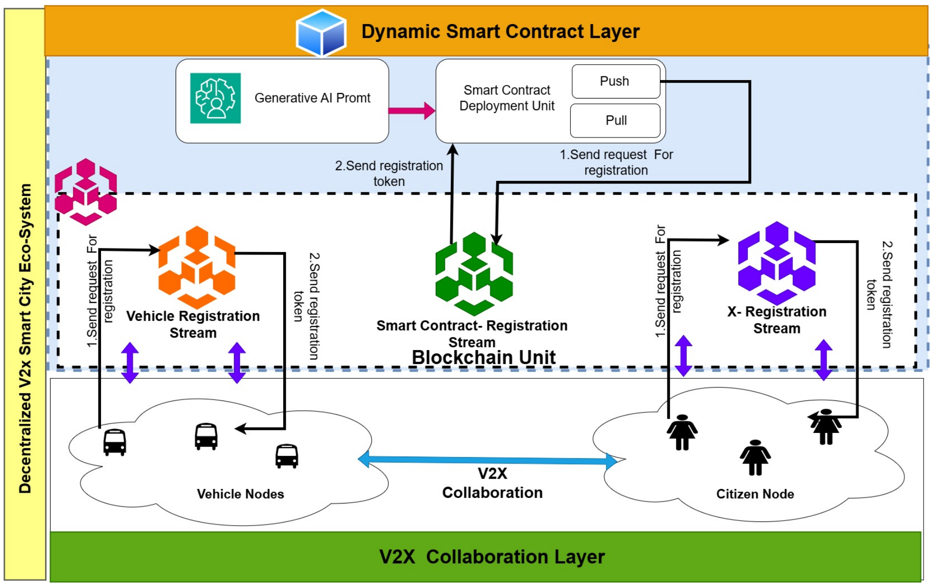
Dynamic Smart Contract Layer
Generative AI Promt
Smart Contract Deployment Unit
Push
Pull
2.Send registration token
1.Send request For registration
1.Send request For registration
Vehicle Registration Stream
2.Send registration token
Smart Contract- Registration Stream
Blockchain Unit
1.Send request For registration
X- Registration Stream
2.Send registration token
Decentralized V2x Smart City Eco-System
Vehicle Nodes
V2X Collaboration
Citizen Node
V2X Collaboration Layer

FIGURE 3: Blockchain based Registration Process for Vehicle nodes(V), Citizen nodes(X), and Smart contracts

$$Message = \begin{bmatrix} (RMessage)|| \\ (SharedKey) \end{bmatrix}_{public} \tag{1}$$

In the next step blockchain first verify the legitimacy by verifying the hash of of the shared key against the pre-known identity signature by decrypting the message with the private key of private blockchain. Upon successful verification, a registration token is issued to the respective V2X node and stored securely in the corresponding blockchain stream. This token is then used for subsequent authentication and verification during message exchanges and smart contract interactions.Algorithm 2 shows the algorthmic appraoch of registration process .

VII. DATASET CREATION AND TRAINED MODEL SELECTION

For predicting dynamic fare changes using the FareAI module in the proposed V2X framework, we use the General Transit Feed Specification (GTFS-RealTime) APIs [76], which provide standard formats for public transit data, such as vehicle positions, schedules, and fare structures, to model realistic public transportation behavior. We extract the most relevant variables for predicting fare pricing from the GTFS stream. A feature vector is the mathematical representation of certain attributes: From the GTFS feed, we extract the most relevant attributes for fare pricing predictions. These attributes are represented as a feature vector:

$$x = \{x_1, x_2, x_3, x_4, x_5\}$$

Where:

- x_1: Hour of the day (time)
- x_2: Estimated passenger count
- x_3: Active vehicle count on the route

Algorithm 2 V2X Node Registration

```
1:  procedure RegisterNode($Type, ID, Key$)
2:      $Msg \leftarrow [ID||Key]$
3:      $Enc \leftarrow Encrypt(Msg, BlockchainPubKey)$
4:      Send $Enc$ to GetStream($Type$)
5:      $Dec \leftarrow Decrypt(Enc, BlockchainPrivKey)$
6:      if CheckKey($Dec.Key$) is valid then
7:          $Token \leftarrow$ CreateToken($ID$)
8:          Save $Token$ in GetStream($Type$)
9:          Send $Token$ to node
10:     else
11:         Reject request
12:     end if
13: end procedure
14: function GetStream($Type$)
15:     if $Type$ is Vehicle then
16:         return VehicleStream
17:     else if $Type$ is Citizen then
18:         return CitizenStream
19:     else
20:         return ContractStream
21:     end if
22: end function
```

- x_4: Weather condition index (e.g., clear, rain, snow)
- x_5: Day type (weekday/weekend)

These variables are either directly pulled from GTFS or estimated based on known historical patterns. The **target variable** d represents the *discount factor* suggested for fare adjustment, where:

$$d \in [0, 0.5]$$

indicates a discount between 0% and 50%. The relationship is modeled as:

$$d = f(x_1, x_2, x_3, x_4, x_5)$$

After extracting these feature values, the corresponding fare discount d is predicted using a trained machine learning model. This forms the basis of our **FareAI** component, which supports dynamic pricing decisions in the smart contract system.

A. MODEL SELECTION

In the FareAI module of our V2X framework, we employ the Gradient Boosting Regressor (GBR) to predict dynamic fare adjustments. Gradient Boosting is a machine learning technique that uses an ensemble method to create models by improving each other's errors. To be more specific, it does so by putting together multiple weak learners, which are generally decision trees, to form one strong predictive model. In each iteration, a new tree is built to predict the residuals (errors) of the sequence of trees that came before it, which is done by the new tree only focusing on the areas where the model's performance needs to be improved. This process goes on step by step until a certain number of trees is reached or the improvements are no longer notable.Several characteristics make GBR particularly suitable for our fare prediction task:

1) Handling Non-Linear Relationships: Public transportation data often exhibit complex, non-linear interactions among variables such as time of day, passenger count, and weather conditions. GBR excels at capturing these intricate patterns, leading to more accurate predictions.
2) Flexibility with Various Data Types: GBR can seamlessly process heterogeneous data, including both numerical and categorical variables, without extensive preprocessing. This capability is advantageous when dealing with diverse datasets like those derived from GTFS APIs.
3) Robustness to Overfitting: By adjusting parameters like learning rate and tree depth, GBR can mitigate overfitting, ensuring that the model generalizes well to unseen data—a critical aspect for real-time fare prediction systems.
4) Automatic Feature Selection: During training, GBR inherently assigns varying importance to features, effectively performing feature selection. This means that more influential variables receive higher weights, enhancing the model's interpretability and performance.

By leveraging the strengths of the Gradient Boosting Regressor, the FareAI module can provide precise and adaptive fare adjustments, enhancing the efficiency and responsiveness of the V2X smart city ecosystem.

VIII. GENERATIVE AI INTEGRATION WITH AI-BASED CONTEXTUAL PROMPTING

The integration of Generative AI with the predictive FareAI model forms the core of the system's dynamic logic generation capability. The FareAI model, trained using a supervised regression approach on synthetic V2X data, continuously analyzes real-time input features such as time, passenger volume, vehicle availability, and environmental conditions. When a significant change in operational conditions is detected—such as unusually low occupancy or high vehicle availability—FareAI predicts a suitable discount value $d = f(x)$.However, the FareAI model is limited to numeric prediction and lacks the ability to dynamically construct executable code. To address this, the predicted value d is combined with relevant features x to form a structured natural language prompt $P(x, d)$. This prompt includes both quantitative indicators and semantic cues that describe the current context (e.g., *"low ridership," "rainy weather," "weekday," "off-peak hours"*). The Generative AI model G then transforms this input prompt into Python-style smart contract logic:

$$G(P(x, d)) \rightarrow \texttt{contract logic}(x, d)$$

This function G allows the system to adapt the contract's internal rules to the real-time context. The modularity and adaptability offered by this integration cannot be replicated using static logic blocks or if-else templates. Moreover, the generated code is reviewed and validated before deployment to ensure correctness and compliance. This hybrid design—combining predictive modeling with generative reasoning—offers a powerful automation pipeline that enhances smart contract responsiveness, operational intelligence, and auditability within dynamic V2X ecosystems.

~~In the Proposed framework we implement the secure exchange of data in V2X collaborative enviroment through Multichain blockchain platform . Multichain is a permissioned blockchain platform that focuses on security, scalability, and governance. It is designed to create and use private blockchain applications. It makes decentralized stream-based data structures possible, which are great for keeping records in order and with timestamps, like logs from IoT and V2X systems.~~Multichain is a permissioned blockchain platform that focuses on security, scalability, and governance. Secure data exchange in the V2X environment involves two key processes: the registration of vehicular nodes and smart citizen (X) nodes, and the establishment of a data publishing pipeline using Multichain's stream feature. This pipeline enables the structured and tamper-resistant publication of data into blockchain streams, ensuring authenticated, timestamped, and verifiable communication between V2X entities.

As shown in Figure-2, the proposed architecture for V2x communication in smart cities is made up of three layers: the perception layer, the controller layer, and the application layer. The perception layer is in charge of collecting the data from the sensing or data nodes of V2X network and passing them on to the controller layer. At the controller layer,