# ARCHITECTURE

The three main architecture patterns are [1]

1. Layer pattern
2. Pipeline and filter pattern
3. Blackboard pattern

## Layer pattern:

Layer pattern architecture helps to structure applications that can be decided into different subtask .The layers can be divided according to the OSI layers, different components (modules) ,protocols . A large system system need decomposition. Blockchain is a very vast system and its implementation has very different components which can be divided into different layers , each layers can be represented by a CRC card having class name as the layer name . In our blockchain implementation we have mainly 4 layers :

1. Ui level
2. Blockchain internal connection
3. Network connectivity
4. Transaction detail verification

Ui layer mostly deals with local db retrieval ,bill generation and the user interface for the customer.

Blockchain internal connection deals with the wallet configuration ,storage of transaction and admin functionality. Network connectivity layer deals with the protocols used for the blockchain network connectivity. Transaction detail verification layer deals with POA and digital signature.

## Pipeline and filter pattern:

This architecture provides structure for system that process a stream of data.each step is encapsulated with a filter component and the best example of this system is a parser.

This type of system is used when we want to compute a stream of data for ex lets take the parser of a compiler , the ascii value is passed bit by bit from a lexical analyser which converts it to a token stream then in the next step it is passed through the syntax analyser in which it is filtered and then checked for any syntax error and so on.

Filter components are the processing units of the pipeline. A filter enriches, refines or transforms its input data. It enriches data by computing and adding information, refines data by concentrating or extracting information, and transforms data by delivering the data in some other representation. A concrete filter implementation may combine any of these three basic principles.

1. The pipeline element pulls output data from the filter
2. The previous pipeline element pushes new input data to the filter
3. filter is the most active member it is responsible for pushing in and pulling out the data from the pipeline.

The CRC card of this data can be represented using two cards with class filter and pipeline.

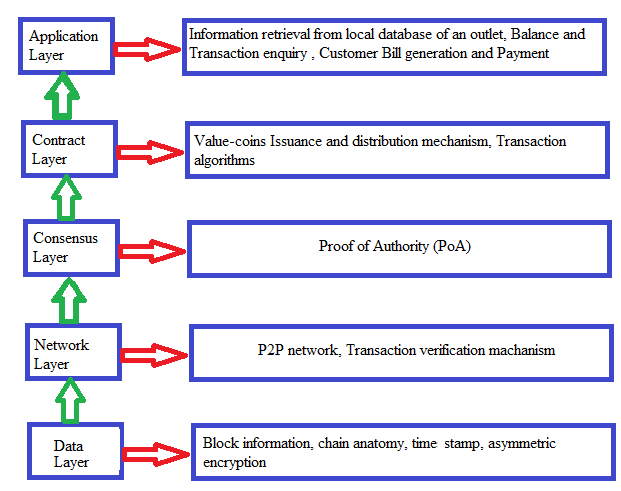
The data flow diagram can be made to explain the flow of the process between the pipeline and filter.

## Blackboard pattern:

The blackboard architecture pattern is useful for problems in which the solution is non deterministic. For e.g. let us consider the speech recognition technique where the speech is converted to text by prediction of the wavelength of the voice. In this architecture is divided into two components blackboard and knowledge source. Speech recognition system has diverse knowledge sources that transform several hypotheses at the same level and with contiguous time intervals to a single hypothesis on the next higher level. For example, a phrase is built from a selection of words that together span the time interval corresponding to the phrase. Other knowledge sources predict new hypotheses at the same level. For example, one knowledge source predicts possible words that might syntactically precede or follow a given phrase. We also define a knowledge source that verifies the predicted hypotheses based on information at the next lower level. This calculates the consistency between a predicted word and the set of segments that span the same time interval. The control component runs a loop that monitors the changes on the blackboard and decides what action to take next. It schedules knowledge source evaluations and activations according to a knowledge application strategy. The basis for this strategy is the data on the blackboard.

This technique cannot be used for our implementation of block chain as we need to have a deterministic solution.

**Blockchain Technology Architecture**

**Fig. 1.** Blockchain Technology Architecture

There are five conceptual layers of blockchain: data layer, network layer, consensus layer, contract layer and application layer [2].

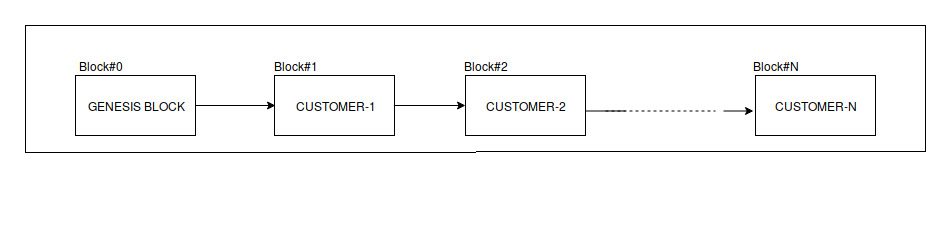
The *data* layer contains the block data, structure of the chain, customer address data, time stamp of block creation and details of asymmetric encryption. It is because of the data layer that transactions can happen transparently in a non-repudiated and authenticated manner. The *networking* layer makes sure that each node receives transactions. The *consensus* layer makes sure that each node agrees on the same transactions to modify their local state. The *contract* layer is responsible for checking the credit limit of the customer in order to make a successful transaction. It ensures a safe and secure transaction, issuance and distribution of Value-coins by the genesis block. As for the *application* layer, it processes transactions. Given a transaction and a state, the application will return a new state. Each transaction abides by the norms of the contract layer and modifies the state according to the specific transaction rules.

**Blockchain Internal Configuration**

A blockchain is a transaction database shared by all nodes participating in a system based on the Bitcoin protocol. A full copy of a currency's block chain contains every transection ever executed in the currency. With this information, one can find out how much value belonged to each address at any point in history. Every block contains a hash of the previous block. This has the effect of creating a chain of blocks from the genesis to the current block. Each block is guaranteed to come after the previous block chronologically because the previous block's hash would otherwise not be known. Each block is also computationally impractical to modify once it has been in the chain for a while because every block after it would also have to be regenerated. These properties are what make blockchain transactions irreversible.

As we know that blockchain uses linked list data structure for storing data.Each block in blockchain

stores details about the previous block hash function Fig 1 gives the overview of connection between blockchain.



**Fig. 2.** Blockchain of Customer

In each block we store details about various details about user the details about user is as follows

**Previous Block Hash**: it contains the hash value of previous block in blockchain. It is used to make connection between different blocks by using Linked list data structure

**Nonce**: It is a 32 bit arbitrary random number that is typically used once. It is basically used for calculation of has value of current block.

**Current Balance**: It indicates the amount of un-spent Value-coins present in the wallet.

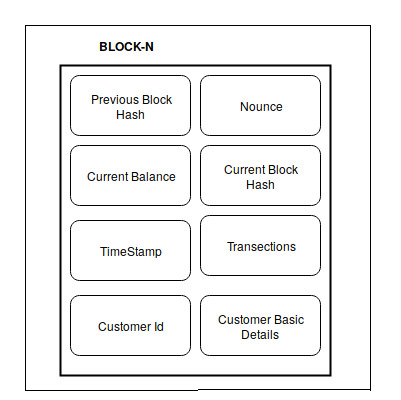
**Current Block Hash**: It is calculated by hashing the value of nonce, previous block hash, Timestamp value by applying SHA256 algorithm.

**Time Stamp**: Each block contains a Unix time timestamp. In addition to serving as a source of variation for the block hash, they also make it more difficult for an adversary to manipulate the block chain.

**Transactions:**Inside each block we store receipt of purchase inside a block in SQL format of fixed length.

**Customer id:**It is a uniquely identity for each customer in blockchain

**Customer Contact Details**: We store the basic contact details such as mobile number, name, address etc



**Fig. 3.** Magnified View of a Customer Block

**Application Layer: User Interface**

UI of our application will be made for Android using Android Studio. It will first have a login and signup page. New user have to login into our system and provide all the required details during the signing up. All the details asked will be useful for running our application successfully and reaching the expectation.

Once signup procedure is completed, user have to login again into the system via the application using their credentials. Customers can also change their credentials again afterwards according to their wish and comfort. Our application will be having following options:

* **Add coins(money) to their wallet.** Since all the payment will be done using our made currency ie ValueCoins. We must add money to our application in the wallet provided. It will convert all the currency into ValueCoins as per the rate decided before.
* **Add and delete something to and from cart respectively.** While in shops/marts customer can add some products into their cart by scanning the QR code on the product and keep it with them. And if they feel that some product are useless to them or not necessary, they can delete the item from their cart just using the remove option available to every item present in the cart and the corresponding item will be removed.
* **Check the list of items in cart.** Customer can at anytime see the number of items available in their cart and can decide whether to shop more or not. And can even mange their checklist and it will also show the total cost of the cart and can decide further.
* **Make some transactions.** After doing all the shopping they can just pay the amount, displaying below the cart. Payment will be made using the ValueCoins present in the wallet.
* **Check their balance.** After making all the payment or before shopping customer can check their balance.
* **See their past transactions.** It will show the list of transaction made. It will contain only up-to certain number or certain time. It will have details of the transaction such as coins spent, products taken, coins rewarded for shopping.
* **Change their credentials.** If one customer feels like one should change their credentials, they can by just providing certain details which will be available only with the authorized user.

**References:**

1. Pattern-oriented software architecture: A system of Patterns, volume 1; Wiley series
2. Gautier Marin https://blog.cosmos.network/understanding-the-value-proposition-of-cosmos-ecaef63350d
3. Satoshi Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System" www.bitcoin.org, 1998.
4. ShawnTabrizi,”A Next-Generation Smart Contract and Decentralized Application Platform”https://github.com/ethereum/wiki/wiki/White-Paper,2009