Low Level Design

Inventory Management in Supply Chain using Blockchain technology

Document Control

Change Record:

Version	Author	Comments
0.1	Dhruv Garg, Akshata Bhaangi	Introduction & Architecture defined
0.2	Piyush Deshpande	Architecture & Architecture Description appended and updated
0.3	Ketan Mahalley, Nikita Madhyan	Unit Test Cases defined and appended

Contents

1. Introduction

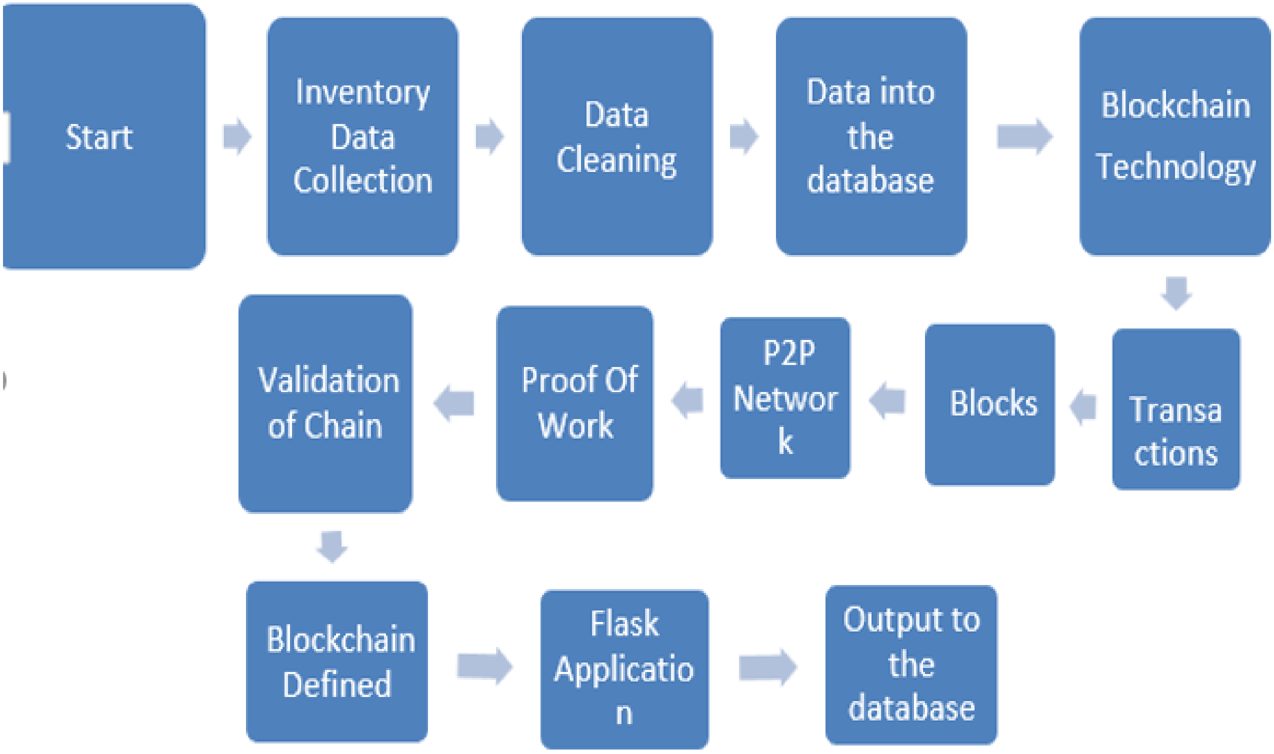
1.1. What is Low-Level design document?

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for Food Recommendation System. LLD describes the class diagrams with the methods and relations between classes and program specs. It describes the modules so that the programmer can directly code the program from the document.

1.2. Scope

Low-level design (LLD) is a component-level design process that follows a step-by-step refinement process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work

2. Architecture



3. Architecture Description

3.1. Inventory data collection:

By using automated data collection in inventory management, companies can easily track the number of items in their warehouses at any given time. When this is linked to existing financial reporting or other tracking systems, companies can better track stock availability, packing needs and operational costs.

3.2. Data Cleaning

By using automated data collection in inventory management, companies can easily track the number of items in their warehouses at any given time. When this is linked to existing financial reporting or other tracking systems, companies can better track stock availability, packing needs and operational costs. Data cleaning unit helps in fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within the dataset.

3.3. Data in to Database

Our data contains fields like Storage date, Fixed cost, Total Cost of Storage, Product ID, etc.

3.4. Blockchain Technology

Blockchain is a simple, automated and safe way to pass information from point A to point B. On one end, the party creating a block initiates the transaction. The block is then verified by thousands, or even millions, of computers spread across the internet. Once verified, the block is added to a chain, which is stored across the network, creating a unique record with a unique history that is decentralized and distributed. From this point, it would be virtually impossible for hackers to falsify a single record because they would need to access and alter the data on all linked computers at precisely the same time.

3.5. Validation of Chain

The validation chain is a middleware, and it *should* be passed to an Express route handler. When the middleware runs, it will run each validator or sanitizer in the order they were specified; this means if a sanitizer is specified before a validator, the validator will run with the sanitized value.

3.6. Proof of Work

It is a type of cryptographic proof in which one party (in inventory management system) shows the verifiers that a specified amount of computing effort has been done. Following that, with minimum effort on their part, verifiers can authenticate the spending.

3.7. P2P network

The blockchain technology that underpins it takes advantage of the power of peer-to-peer networks to create a shared and trusted record of transactions. Blockchain, as a distributed ledger system, records transactions as an immutable timestamped digital block with senders and recipients identified.

3.8. Flask Application

Flask is one of the micro-categories. framework's Micro-frameworks are often frameworks that do not rely on external libraries. This has both advantages and disadvantages. The framework is light, there are few dependencies to update, and there are few security problems to check for. The disadvantages are that you will have to do more work on your own at times, or grow your list of dependents by adding plugins.

4. Unit Test Cases

Test Case Description	Pre-Requisite	Expected Result
Check to see if the user can access the Application URL.	The URL of the application should be specified.	The user should be able to view the application's URL.
When the URL is browsed, check to see if the application loads completely for the user.	1. The application's URL is available. 1. 2. The application has been launched.	When the URL is reached, the application should entirely load for the user.
Check to see if the user can register for the app.	The application is usable.	The user should be able to register for the app.
Check if the user can successfully log in to the application.	1. The application is usable. 1. 2. The user has registered with the application.	The user must be able to log in to the programme successfully.
Check to determine if the user can view the input fields when they log in.	1. The application is usable. 2. The user has registered with the application. 3. The user has successfully logged into the programme.	When logging in, the user should be able to see the input fields.
Check that the user has access to all input fields.	1. It is possible to use it. 2. The user has downloaded and installed the programme. 1. 3. The programme is open and the user is logged in.	Every input field should be editable by the user.
Make sure the user receives it. The inputs will be submitted when you click the Submit button.	1. It is possible to use it. 2. The user has downloaded and installed the programme. 3. The programme is open and the user is logged in.	To submit inputs, the user needs have a Submit button.

Check if the user is given with suggested results after pressing submit.	1. The application is usable. 2. The user has registered with the application. 1. 3. The user has successfully logged into the programme.	After pressing submit, the user should be given with suggested results.
Check to see if the proposed outcomes are in line with the user's choices.	1. The application is usable. 2. The user has registered with the application. 3. The user has successfully logged into the programme.	The suggested outcomes should be in line with the user's choices.
Check to see if the user can filter the suggested results.	1. The application is usable. 2. The user has registered to the application 3. User is logged in to the application	Additionally, the user should be able to filter the suggested results.
Verify whether KPIs modify as per the user inputs for the user's health	1. Application is accessible 2. User is signed up to the application User is logged in to the application	KPIs should modify as per the user inputs for the user's health
Verify whether the KPIs indicate details of the suggested recipe	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	The KPIs should indicate details of the suggested recipe

6 LOW LEVEL DESIGN (LLD)