**DOG ADOPTION DECENTRALIZED APPLICATION USING BLOCKCHAIN**

Submitted in partial fulfillment of the requirements for the award of degree of

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE & ENGINEERING**



**Submitted By:**

**Submitted to:**

Er.Navpreet Kaur Walia(E7347) (Group-A)

Er.Ankita Sharma(E6204) (Group-B)

ANIL THAKUR 18BCS3150

ANIKET SINGH 18BCS3155

PRITI PANDEY 18BCS3290

KASHISH KUMAR 18BCS3171

**Mentor Signature:**

Dr.Meenu Gupta(E9406)

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**Chandigarh University, Gharuan**

**June 2021**

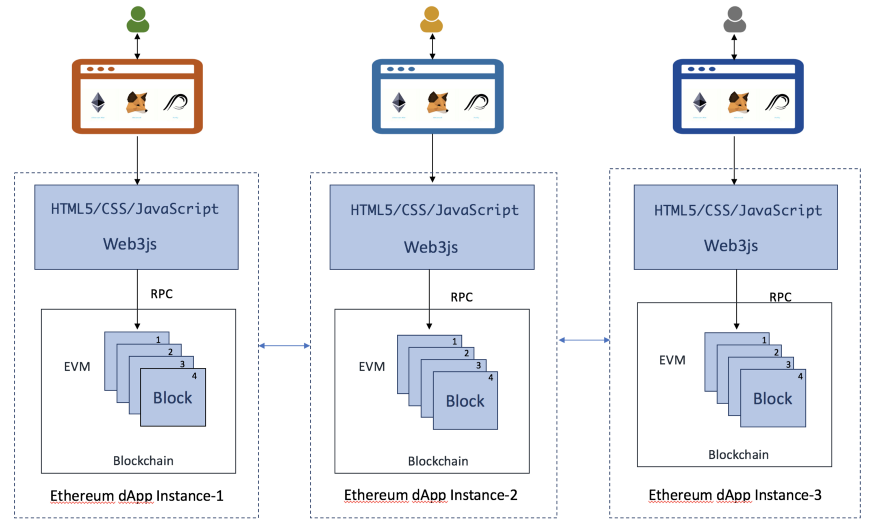
**FIRST PROGRESS REPORT**

**Project Design:**

**The Blockchain is our Database**

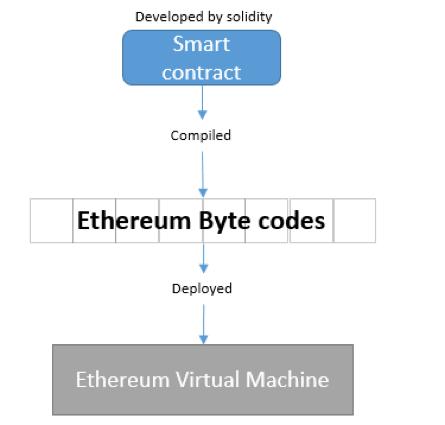
[A blockchain is a chain of blocks or pieces of data that store data and transactions.](https://towardsdatascience.com/a-deep-dive-into-blockchain-d1eb753fb74c) Everything on the blockchain is accessible to anyone and can be used in any form. The way this works is that every transaction on the blockchain is verified over a bunch of nodes, maintained by miners.

 For Ethereum , the library Web3.js is used to communicate with the blockchain. The way this works is that it uses something called the JSON-RPC protocol to query the blockchain. The JSON-RPC is a remote procedure call that sends JSON data to the blockchain that tells it to retrieve something specific. The heart of these dapps are the smart contracts, and are the principal functionality component of the app.



**Smart Contracts: The Principal Functionality**

Smart contracts are immutable pieces of code that run decentralized applications. They have discrete functions with certain capabilities that anyone can access and see. A smart contract is nothing but an agreement between multiple parties. A user can agree to the agreement by signing the contract with their private key. The fact that they are immutable and can’t change is what makes them so powerful.



**Project phase development:**

* Setting up the development environment
* Creating a Truffle project using a Truffle Box
* Writing the smart contract
* Compiling and migrating the smart contract
* Testing the smart contract
* Creating a user interface to interact with the smart contract
* Interacting with the dapp in a browser

**Software’s to install:**

* Node.js
* Ganache
* Truffle
* Metamask

**Modules:**

1. **Backend** (Creating a Truffle project using a Truffle Box)

Will be working with truffle and Writing the smart contract

1. **Migration**

In this module the smart contracts will be compiled and migrated to [Ganache](https://www.trufflesuite.com/ganache), a personal blockchain for Ethereum development.

1. **Frontend** (Interacting with the dapp in a browser)

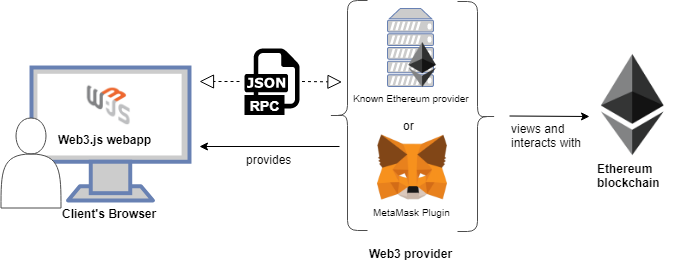
In this module we will be creating user-interface to interact with the contracts made in the backend module of the project.

**Innovation in model/design/solution:**

**Innovation:**

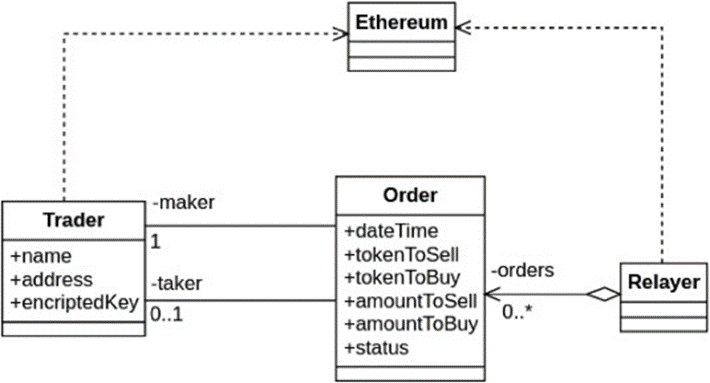
In the client server model i.e., in centralized application at a single point of failure the whole system collapses and becomes more prone to attackers to easily get access into the network. To overcome this problem decentralized application is used which is more secure and efficient. It eliminates the drawbacks of centralized system making it more efficient than the existing one. This project aims at adopting new technology i.e., Blockchain and replacing the existing system and its flaws.

**Flowchart representation:**

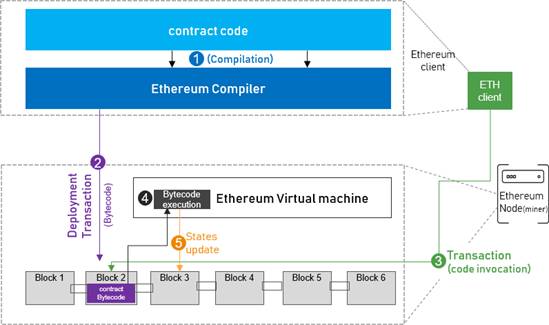
Below is the diagrammatic representation of our project

We will be using Ethereum Blockchain for our project development which will be connected through Metamask which is our Web3 provider for client’s browser.

Below is the class diagram representing the adoption process of the dogs. Here we can see that both the trader and buyer are dependent on the Ethereum to make the process of dog adoption successful. The buyer is having one to many relationships i.e., he/she can adopt more than one dog.



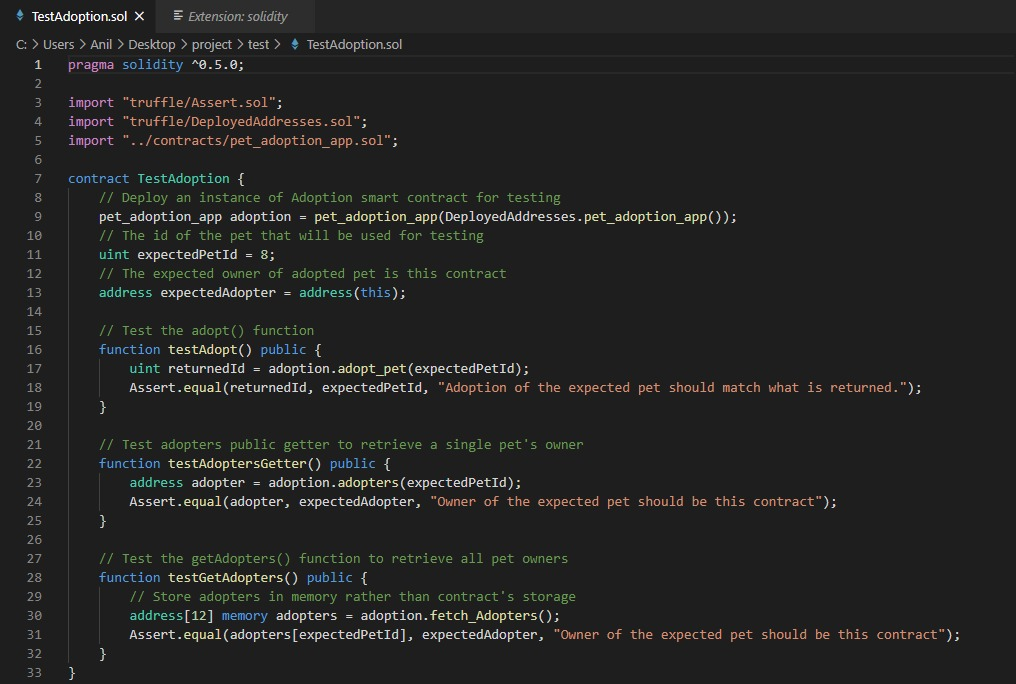
The below diagram represents the detailed working of our model.



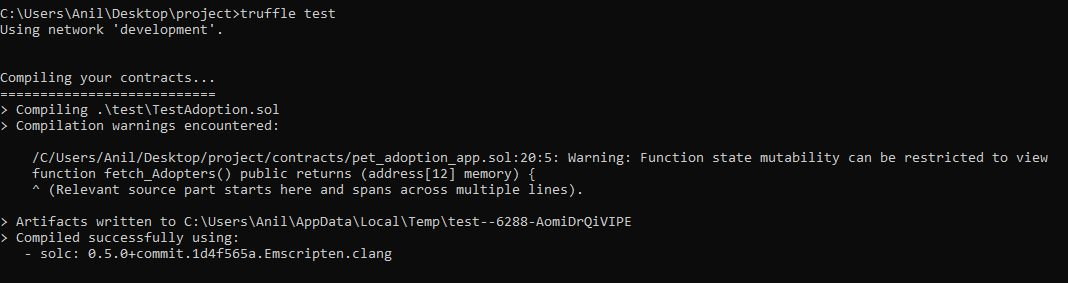
**Test cases:**

Truffle is very flexible when it comes to smart contract testing, we will be testing our contract using Solidity.

Below is test file named as TestAdoption.sol







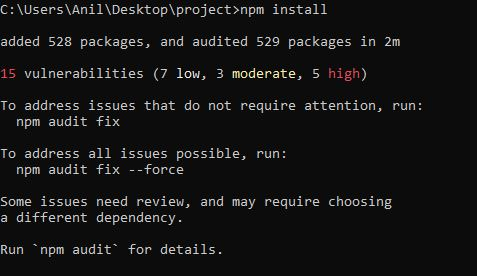
**Implementation:**

**BACKEND:**

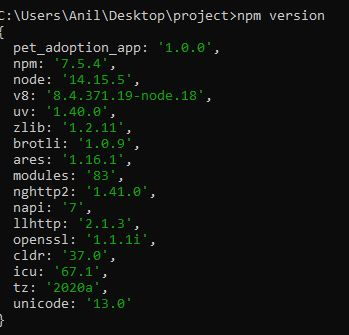
At the backend we will begin by setting up the development environment and installing necessary software tools. We have installed Node.js and on node.js command line we will be running all our commands.

**1. Setting up the development environment:**

Firstly, we will install npm (node packet manager) which will download and manage all the predefined packages of the node.js



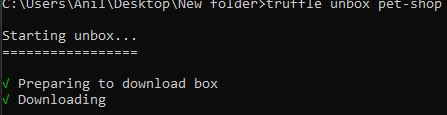
These are the packages npm installs:



After installing npm we will install truffle and verify it if properly installed or not.

**2. Creating a Truffle project using a Truffle Box**:

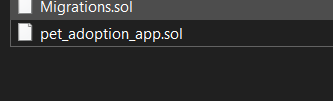
For our project we will installing all the required packages by npm(node packet manager) by downloading truffle-box named as: pet-shop which includes the basic project structure.



**3. Writing the smart contract:**

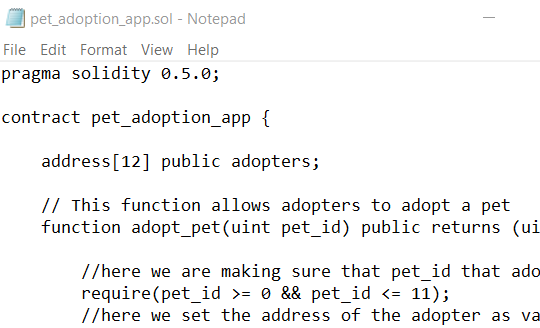
Now We'll start our dapp by writing the smart contract that acts as the back-end logic and storage.

Our contract name will be pet\_adoption\_app.



Language used to write contract will be solidity.

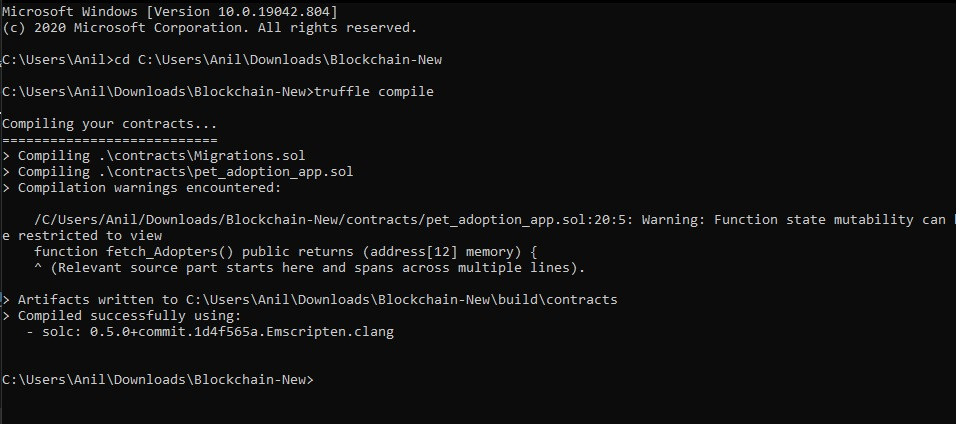
Below is the screen shot of the contract.



We've defined a single variable: adopters. This is an array of Ethereum addresses. Arrays contain one type and can have a fixed or variable length. In this case the type is address and the length is 12.

In this contract pet adoption function and retrieving he adopters function is included.

Now the contracts will be compiled.



The contract has been successfully compiled.