**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)

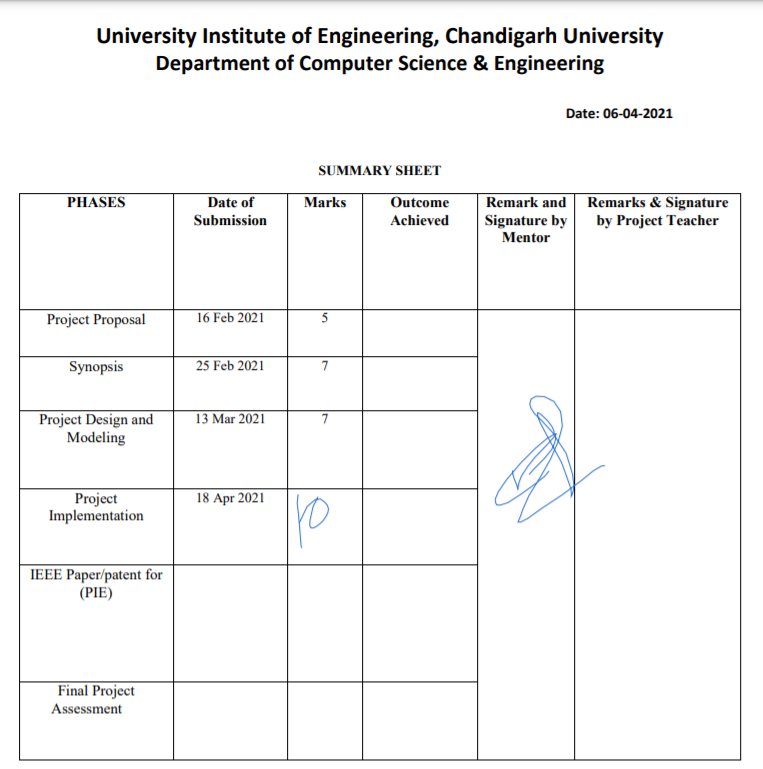
# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

# Chandigarh University, Gharuan,

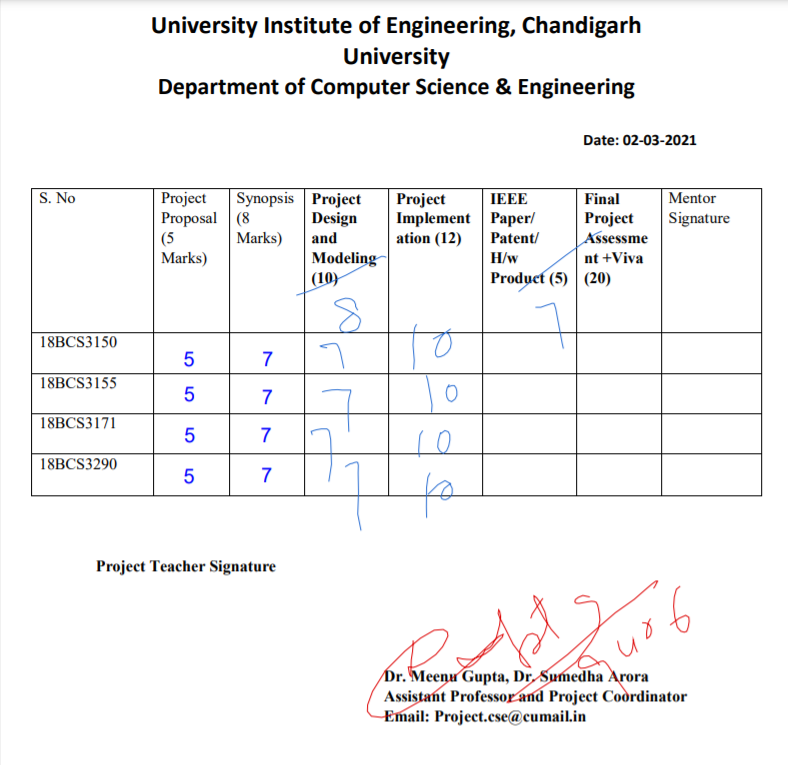
# June 2021

**FINAL REPORT**

|  |  |  |
| --- | --- | --- |
| SERIAL NO. | TOPIC | PAGE NO. |
| 1 | SUMMARY SHEET | 3 |
| 2 | MARKSHEET | 4 |
| 3 | PROJECT PROPOSAL | 5 |
| 4 | PROJECT SYNOPSIS | 6-12 |
| 5 | LOG REPORTS | 13-16 |
| 6 | FIRST PROGRESS REPORT | 17-26 |
| 7 | SECOND PROGRESS REPORT | 27-38 |

**SUMMARY SHEET**:

**MARKSHEET**:



**DOG ADOPTION DECENTRALIZED APPLICATION USING BLOCKCHAIN**

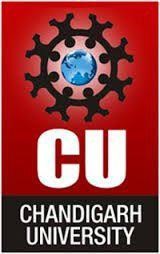
Submitted in partial fulfillment of the requirements for the award of degree of

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE & ENGINEERING**

**Submitted By:**



**Submitted to:**

Er.Navpreet Kaur Walia(E7347) (Group-A) ANIL TH AKUR 18BCS3150

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

**Introduction:**

We will be making a Dog Adoption Application, which will be basically a Decentralized Application running on Blockchain Technology. It will not be using DBMS for storage of data and will be highly secured. We will be associating an Ethereum address with the pet to be adopted.

Decentralized application:

Decentralized applications (DApps) are digital applications or programs that exist and run on a [blockchain](https://www.investopedia.com/terms/b/blockchain.asp) or P2P network of computers instead of a single computer, and are outside the purview and control of a single authority.

DApps can run on a P2P network or a blockchain network. For example, BitTorrent, [Tor](https://www.investopedia.com/terms/t/tor.asp) and Popcorn Time are applications that run on computers that are part of a P2P network, whereby multiple participants are consuming content, feeding or seeding content, or simultaneously performing both functions.

Technology Used: **Blockchain**

A blockchain, originally block chain, is a growing list of [records,](https://en.wikipedia.org/wiki/Record_(computer_science)) called block*s*, that are linked using [cryptography.](https://en.wikipedia.org/wiki/Cryptography) Each block contains a [cryptographic hash](https://en.wikipedia.org/wiki/Cryptographic_hash_function) of the previous block, a [timestamp,](https://en.wikipedia.org/wiki/Trusted_timestamping) and transaction data (generally represented as a [Merkle tree)](https://en.wikipedia.org/wiki/Merkle_tree). By design, a blockchain is resistant to modification of its data. This is because once recorded, the data in any given block cannot be altered retroactively without alteration of all subsequent blocks.

Blockchain Fundamentals:

1. **Disintermediation:** The transactions are not verified by a single central gatekeeper and this may reduce the costs of building an infrastructure and its maintenance, and there may be some performance gains.
2. **Transaction interaction:** Complicated and interconnected transactions can be implemented by using smart contracts. Blockchain structure provides an easy and modular base for public key infrastructures and blind signatures.
3. **More control:** Enterprise blockchain-based applications are usually designed to connect different organizations or trading partners. If even a part of the solution is centralized, each organization must trust the entity controlling that part. Fully decentralized applications don’t have this problem, because a decentralized structure allows each party to run the app without having to trust any other party. This usually results in faster adoption of the application.
4. **Open source:** All DApp code is supposed to be open source, which means it’s transparent and can be viewed by anyone who wants to verify the developers’ claims on what it does.
5. **Less downtime:** dApps are more robust and flexible than centralized applications since they don’t require connectivity to a single centralized server to run. This means that enterprises can ensure minimal interruptions and downtime for maximum business continuity and resilience.
6. **Data is never lost:** Once information is added to the blockchain, it’s stored permanently, which means dApps are more resistant to modifications or restrictions.

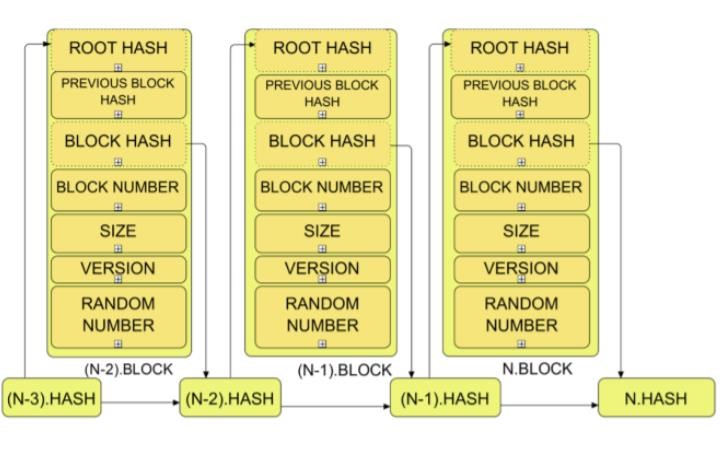
**Feasibility Study:**

This section provides the feasibility study of taking advantage of blockchain technology in decentralized pet adoption applications. By speaking of feasibility, we mean a cost-effective, scalable, secure, and easy-to-deploy system. The system should support many people, depending on the business size or target group population. The security level should not be lower than non-blockchain solutions.

Blockchain, by its nature, cannot be applied to all systems as a modular off-the-shelf solution. If blockchain is not useful for a project, it cannot be feasible, too. Blockchain-based systems may also have the following valueadded characteristics. These values can make the blockchain-based alternatives preferable depending on the use case scenarios.

Security and Reliability:

The security services that the blockchain provides is compared with other database solutions. The availability and the fault tolerance of the system is high as all the nodes keep a copy of the records and check each other to make a stable system. The blockchain provides transparency with anonymity. The privacy is not aimed but can be implemented. Each block keeps the hash of the previous block and this eventually provides a chain of blocks that are linked to each other.



Merkle tree is used in order to keep the integrity of the records. Each block holds more than one transaction. Firstly, hash values of each transaction are taken and paired with the hash of the other transaction. Pairs of hashes are then combined till it results in a single root hash. Transactions can easily be verified by this structure.

Cost Effective:

We are making a decentralized application which will provide a higher transaction speed, which translates to cost reduction. Here the shop owner does not need to use heavy servers and spend time in managing and maintaining the servers and data.

Risk factors:

Possibilities always come along with risks. Each core function of blockchain has several significant threats that need to be evaluated and counter measured before implementation. Not always these risks will be purely technical, because risks can also arise from legal, economical, even cultural areas. One of the main concerns regarding decentralized networks is always about their control.

**Methodology/ Planning of work**:

It will be basically a Decentralized Application running on Blockchain Technology. Unlike traditional applications & websites, this will be using Blockchain Technology as a back-end instead of SQL/Oracle Databases. This Dapp will associate Ethereum address to the Dogs to be Adopted. We will be making use of Ethereum for handling pet adoption in which we will associate an Ethereum address with the pet to be adopted.

For achieving our Objective, we will be learning the essential tools that will be used in the project development. Languages like Solitude are needed for making Ethereum contracts will be learned and applied. We will be studying existing decentralized applications from sites like truffle suite and YouTube tutorials.

Tools such as ganache and NodeJS will also be implemented at the backend, where as HTML will be used as a GUI frontend website. Ganache will be used for maintain Ethereum accounts and transactions. For Ethereum contract transactions, MetaMask Firefox extension will be used. Metamask is nothing but just a browser extension crypto-wallet.

**Module & Team Member wise Distribution of work:**

We believe in Teamwork; therefore approx. equal work is divided between us.

1. Anil Thakur – Back-end Designing + Programming + 25% documentation
2. Aniket Singh – Front-end Programming + 25% documentation
3. Priti Pandey – Back-end Programming + 25% documentation
4. Kashish Kumar – Resource Management + 25% documentation

**Innovations in Project:**

As this is a Blockchain based project, no traditional database will be used. Since Blockchain is a new and trending technology, we are trying to gather as much information and knowledge we can and this will be challenging to work on this project. As a part of innovation, we will be trying to make our project optimized and light as much as possible.

**Software and Hardware Requirements:**

# Software

1. Windows 10
2. NodeJS
3. Node Packet Manager(npm)
4. Ganache
5. Solidity - Programming Language
6. JavaScript, HTML
7. Truffle Suite
8. MetaMask FireFox extension 9) Firefox browser

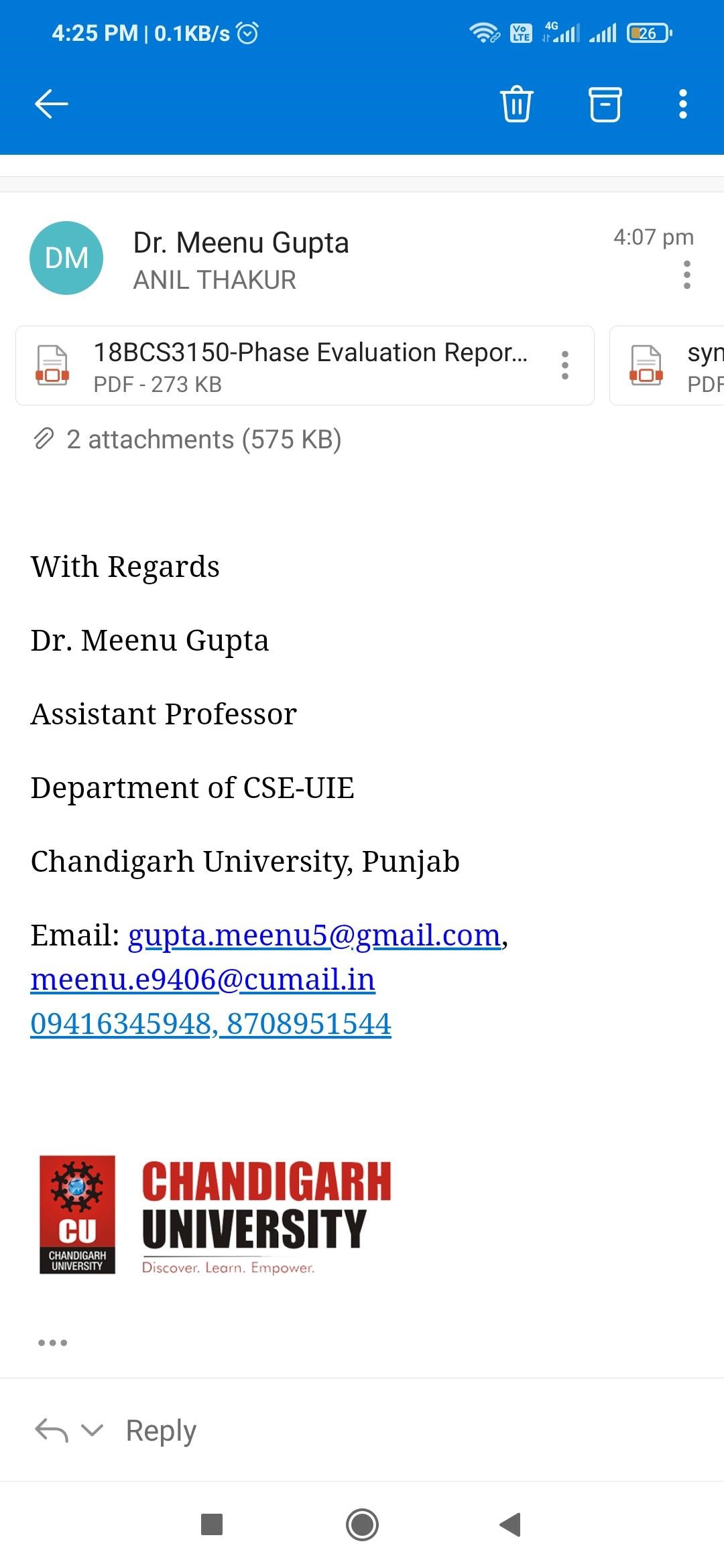
# Hardware

1. 4GB Ram
2. Minimum Dual-Core Processor
3. Normal internet connection

**Bibliography:**

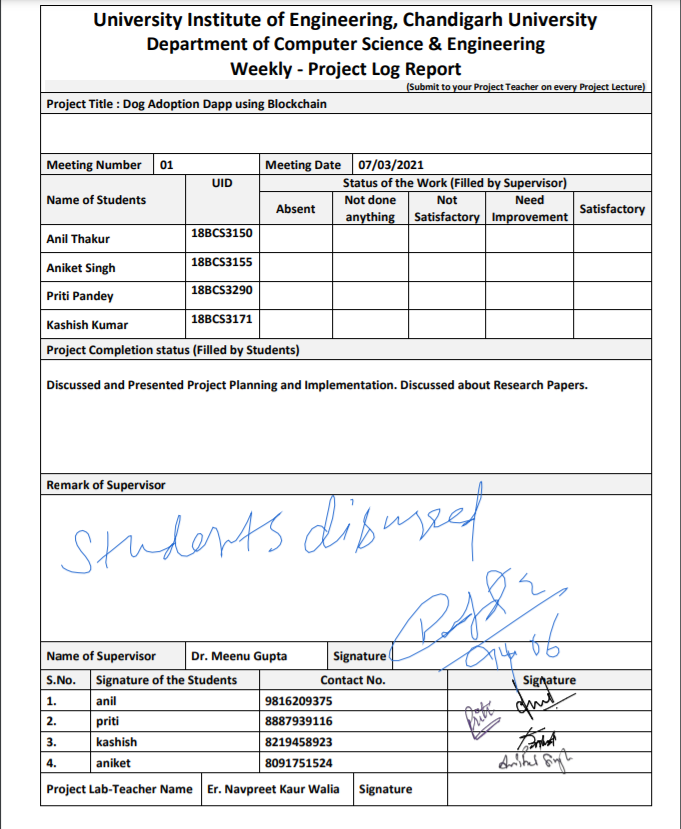
Dapp University – YouTube Channel

Truffle Suite -<https://www.trufflesuite.com/><https://www.investopedia.com/terms/b/blockchain.asp>

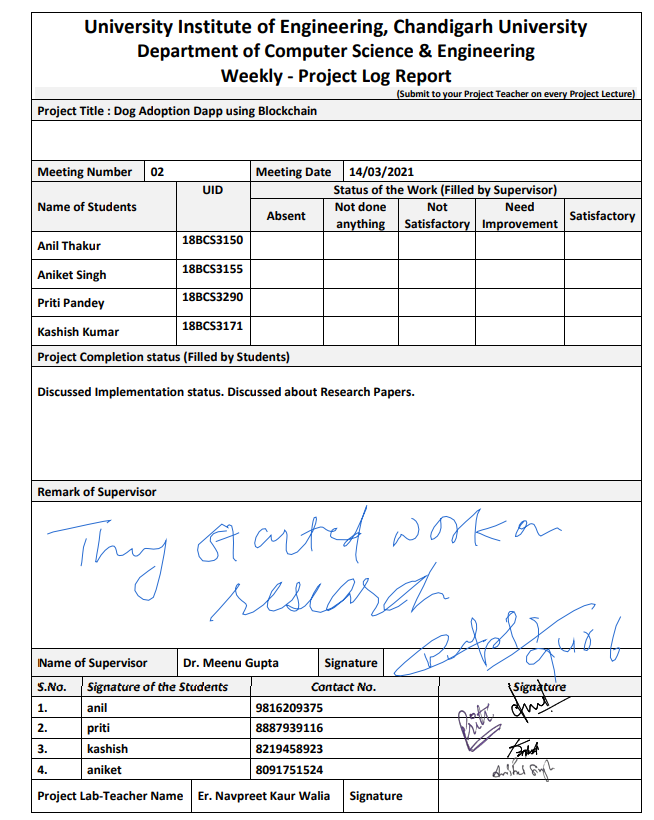


**LOG REPORTS:**

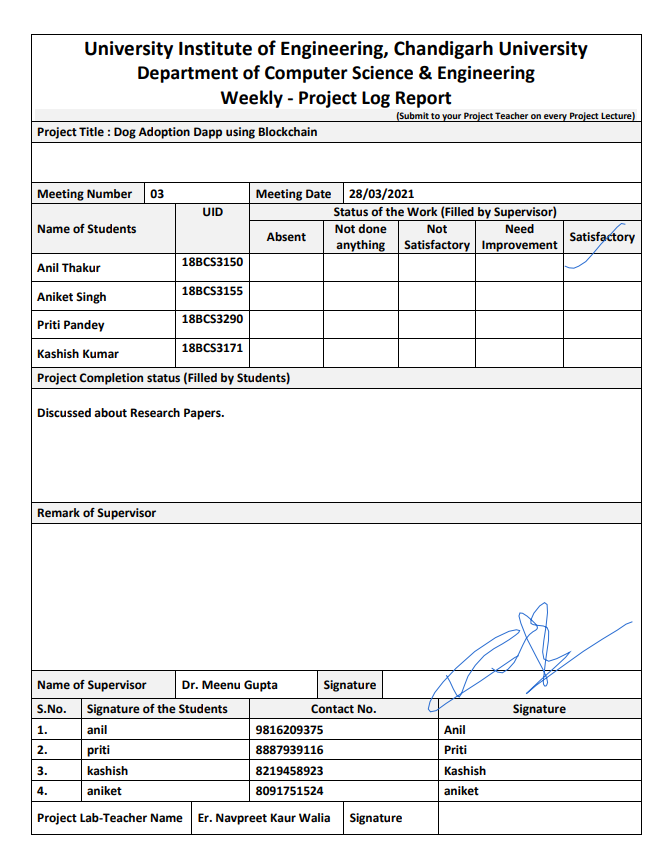
**LOG 1:**



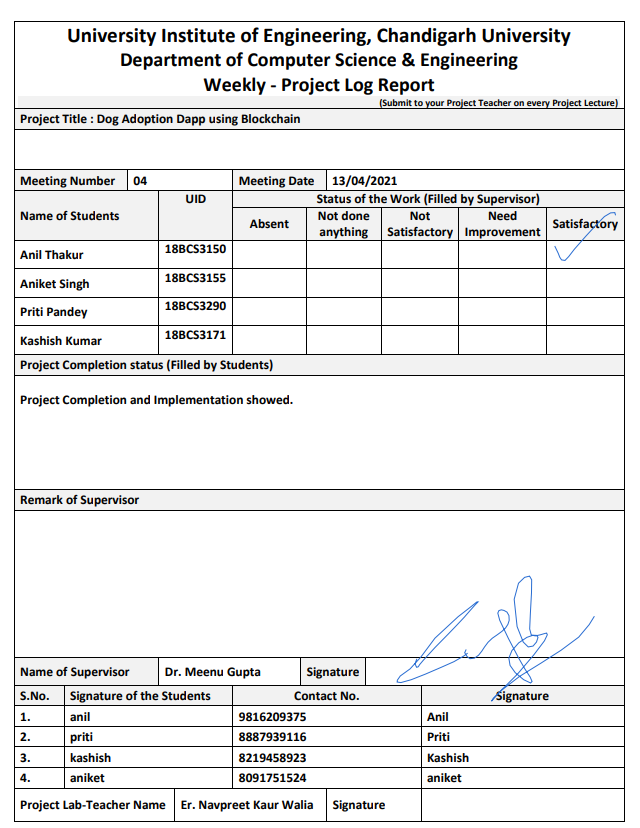
**LOG 2:**



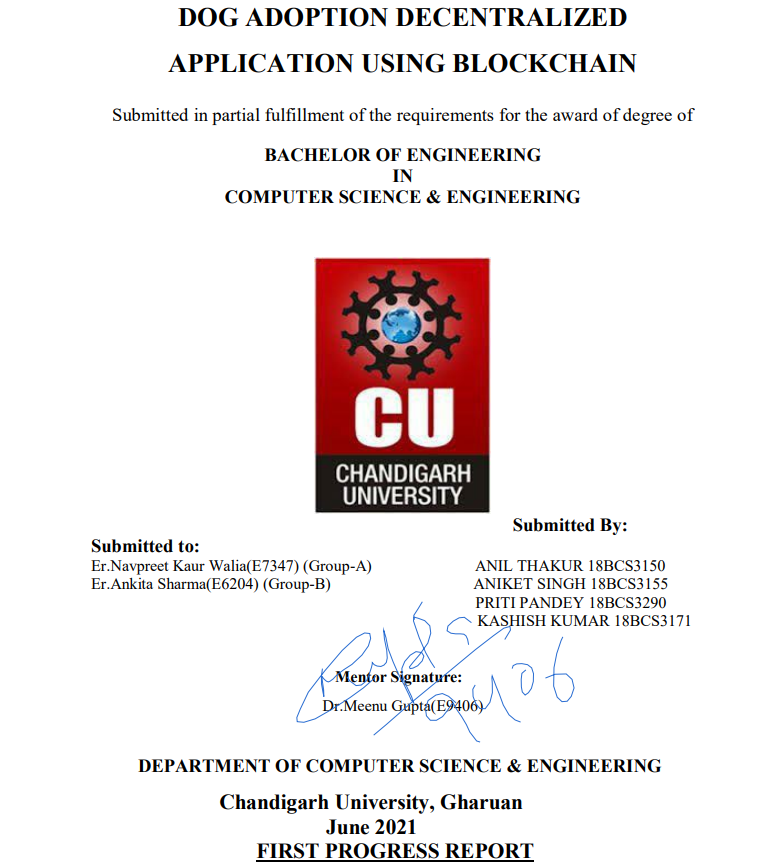
**LOG 3:**



**LOG 4:**



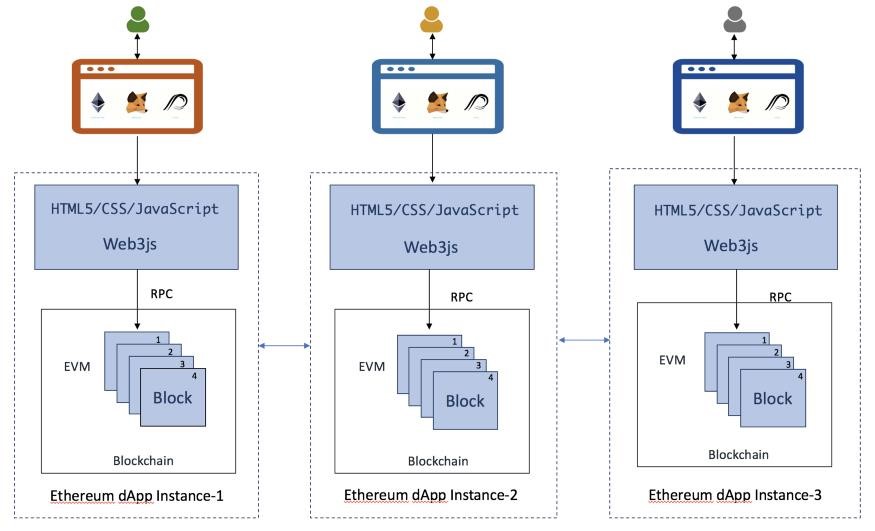
**PROJECT REPORT:**



**Project Design:**

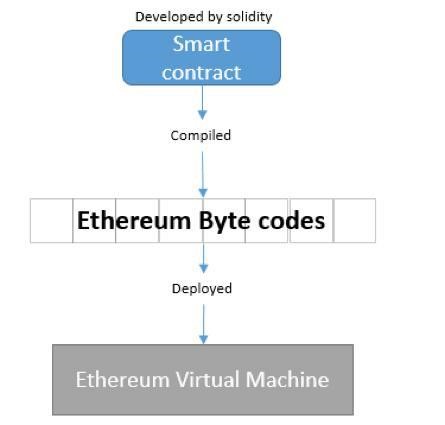
# The Blockchain is our Database

[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
2. **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.

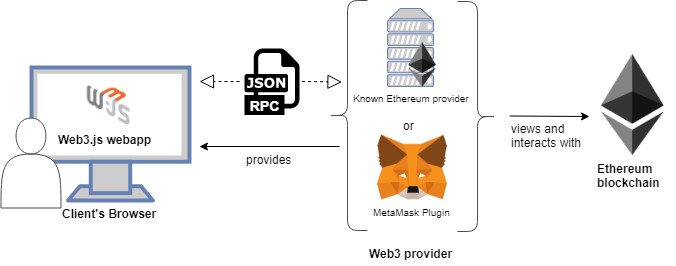
**Innovation in model/design/solution:**

**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.

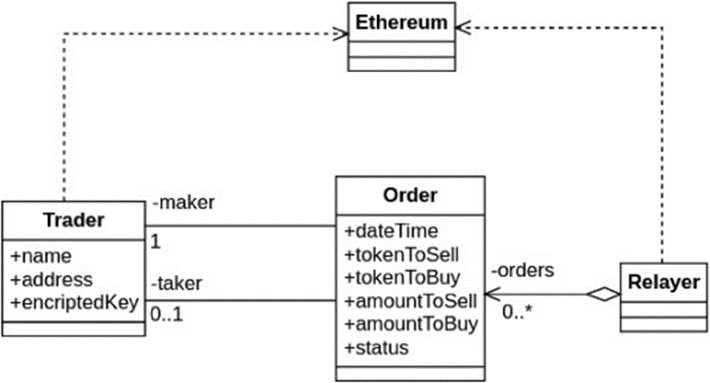
**Flowchart representation:**

Below is the diagrammatic representation of our project

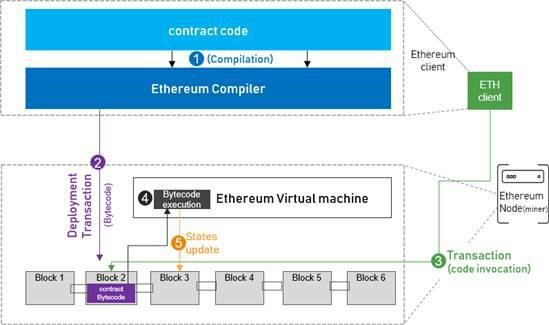


We will be using Ethereum Blockchain for our project development which will be connected through Metamask which is our Web3 provider for client’s browser.

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.



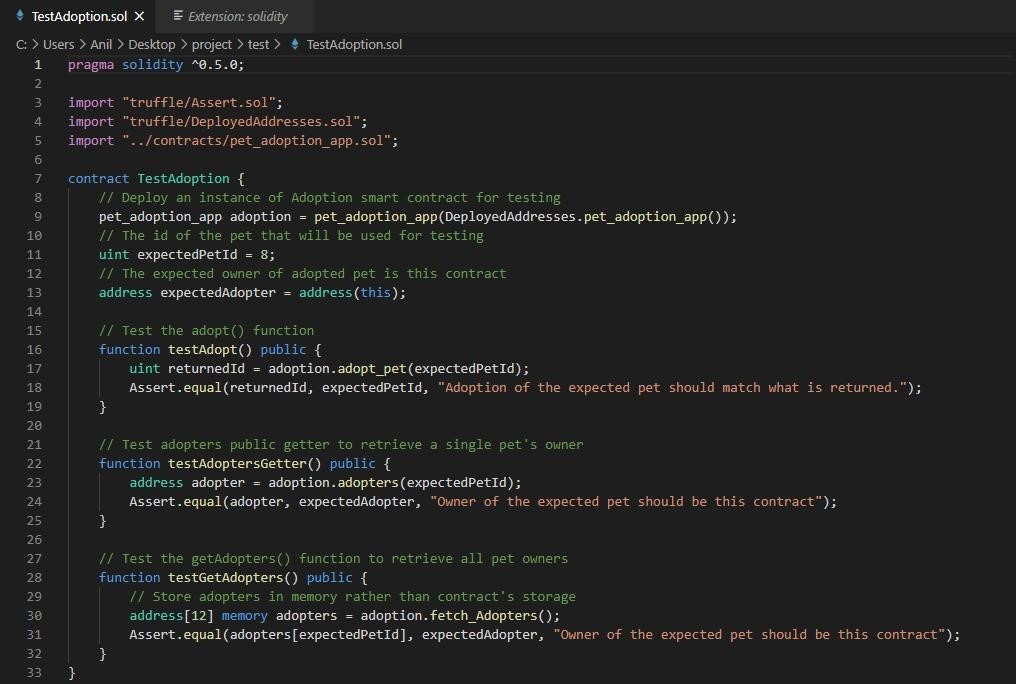
The below diagram represents the detailed working of our model.



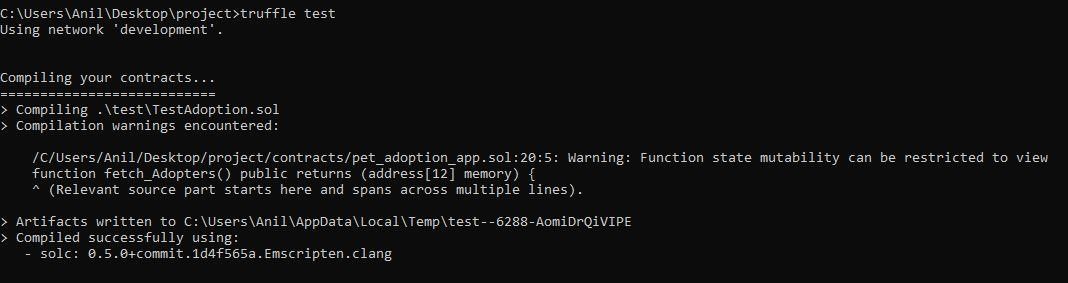
**Test cases:**

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







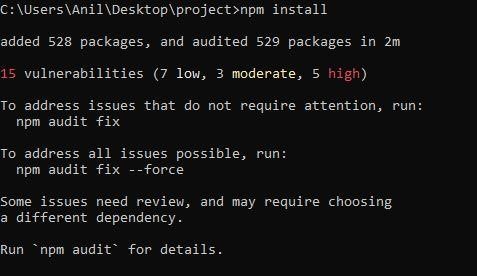
**Implementation:**

**BACKEND:**

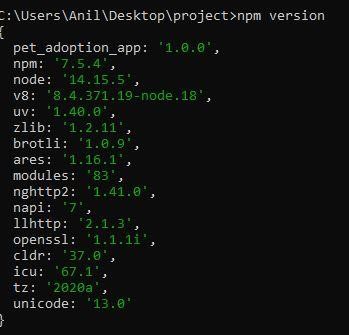
At the backend we will begin by setting up the development environment and installing necessary software tools. We have installed Node.js and on node.js command line we will be running all our commands.

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



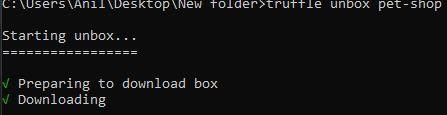
These are the packages npm installs:



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

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

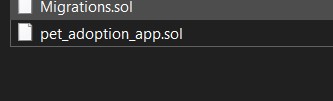
For our project we will installing all the required packages by npm(node packet manager) by downloading truffle-box named as: pet-shop which includes the basic project structure.



1. **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.

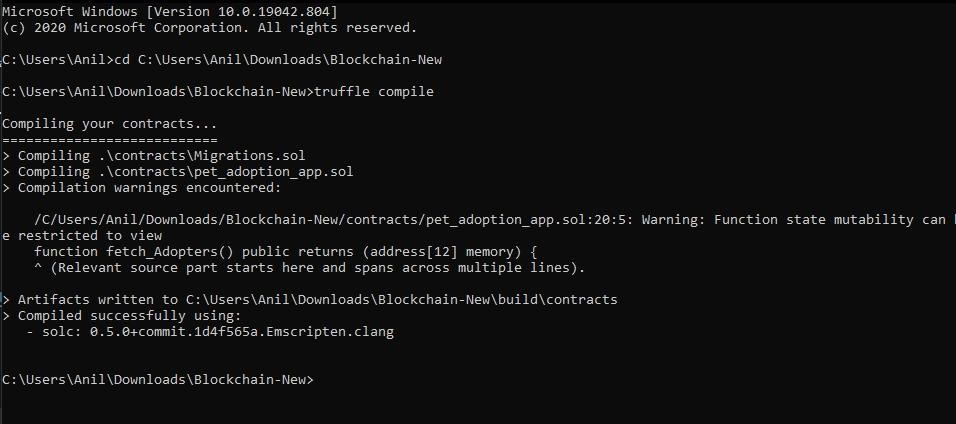
Below is the screen shot of the contract.



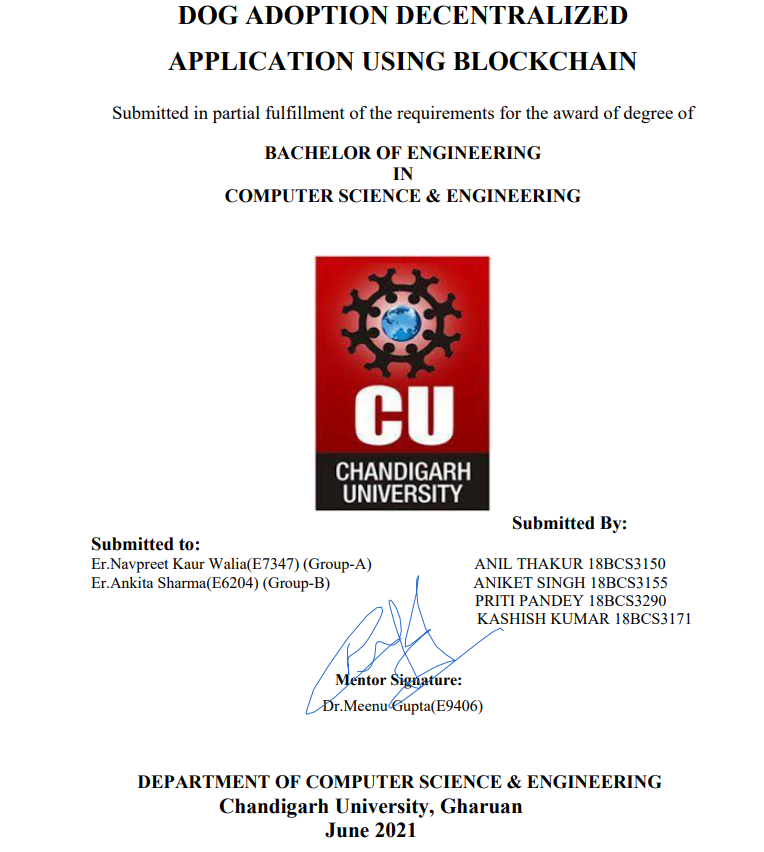
We've defined a single variable: adopters. This is an array of Ethereum addresses. Arrays contain one type and can have a fixed or variable length. In this case the type is address and the length is 12.

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.



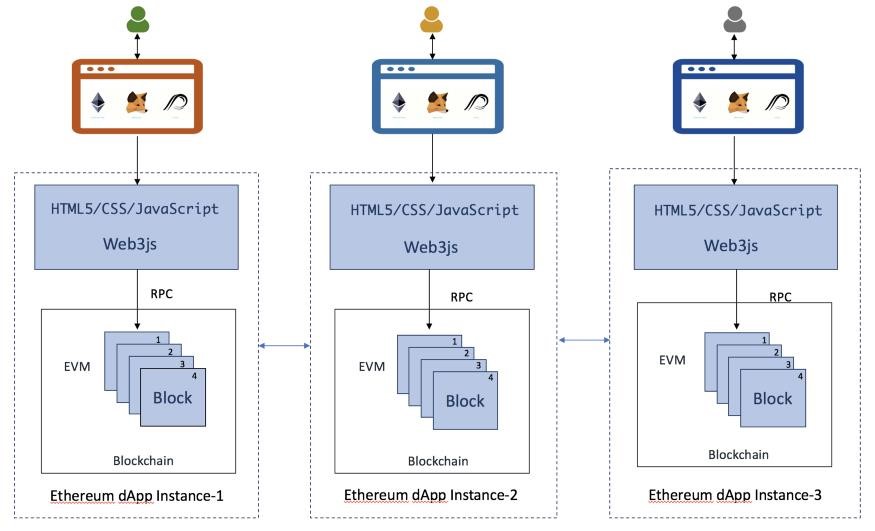
**SECOND PROGRESS REPORT**

**INTRODUCTION:-**

Dog Adoption using Blockchain . This project focus on online adoption of dogs using secure peer to peer transaction using blockchain . As Dogs are part of our livehood , more dogs got hurted on roads ,rather, got killed by hunger. This project aim to save those or help them to find new secure place . In this dog will be adopted through online transaction using blockchain. A blockchain is a chain of blocks or pieces of data that store data and transactions. It is a distributed network that is used for various different purposes. The recorded information is opened and can be viewed by everyone in blockchain. 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.

## **The Blockchain is our Database**

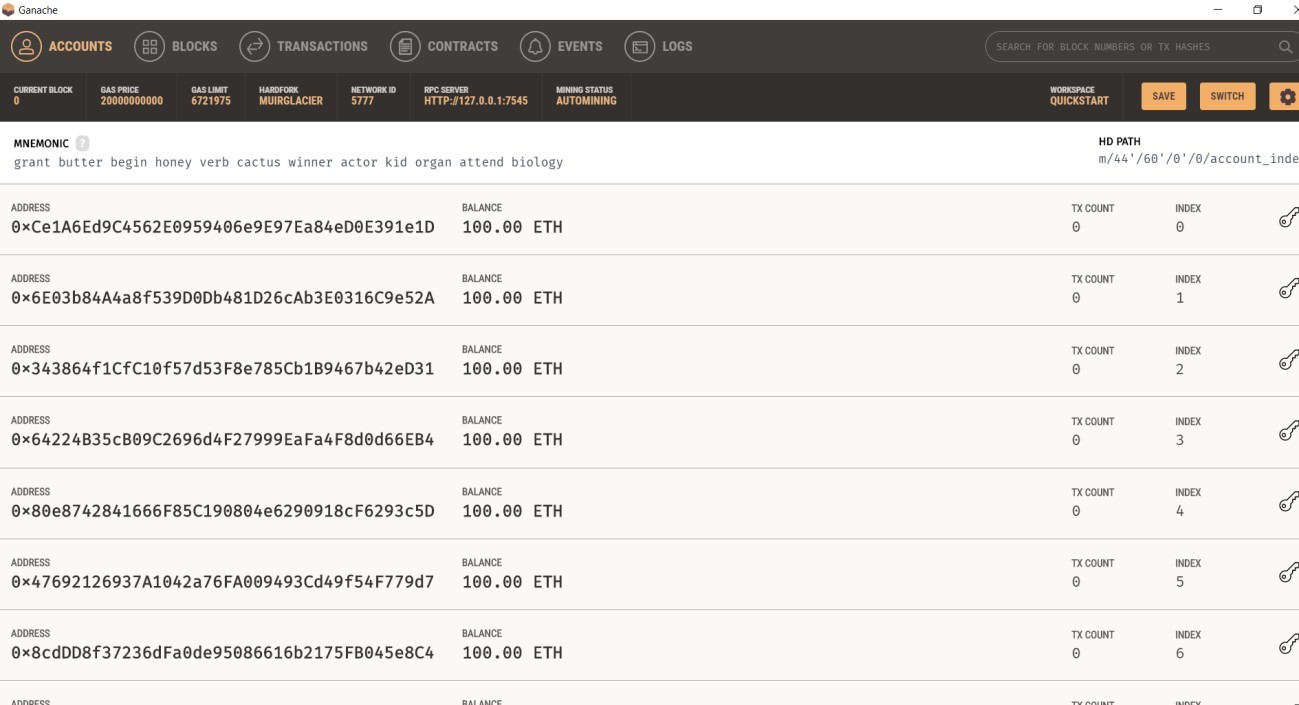
A blockchain is a chain of blocks or pieces of data that store data and and transactions . Everythi ng on the blockchain is accessible to anyone and can be used in any form. The way this works is that every transaction on the blockchain is verified over a bunch of nodes, maintained by miners. For Ethereum , 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.



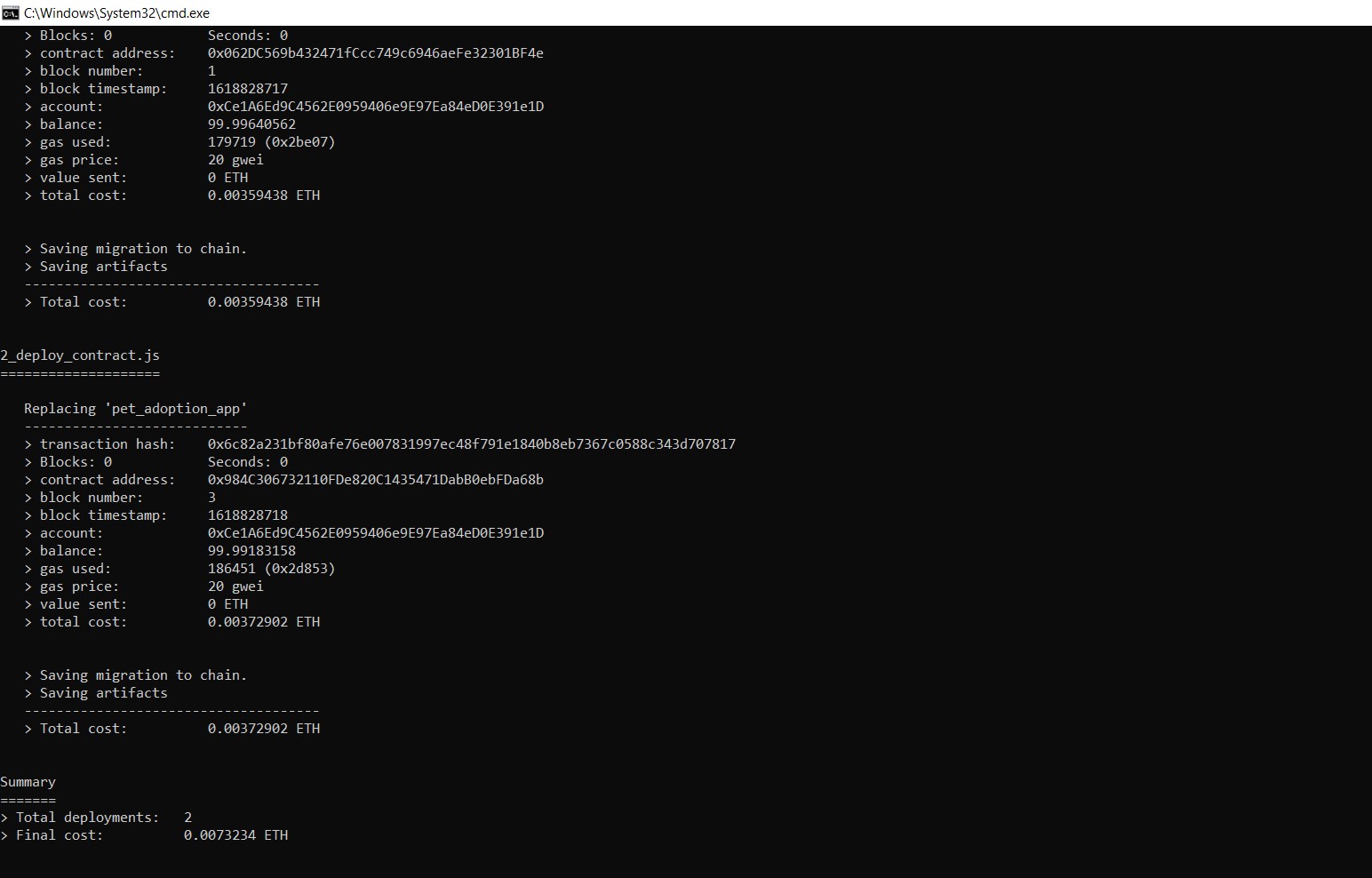
# Implementation (100%):-

**Screenshots and Explanation:-**

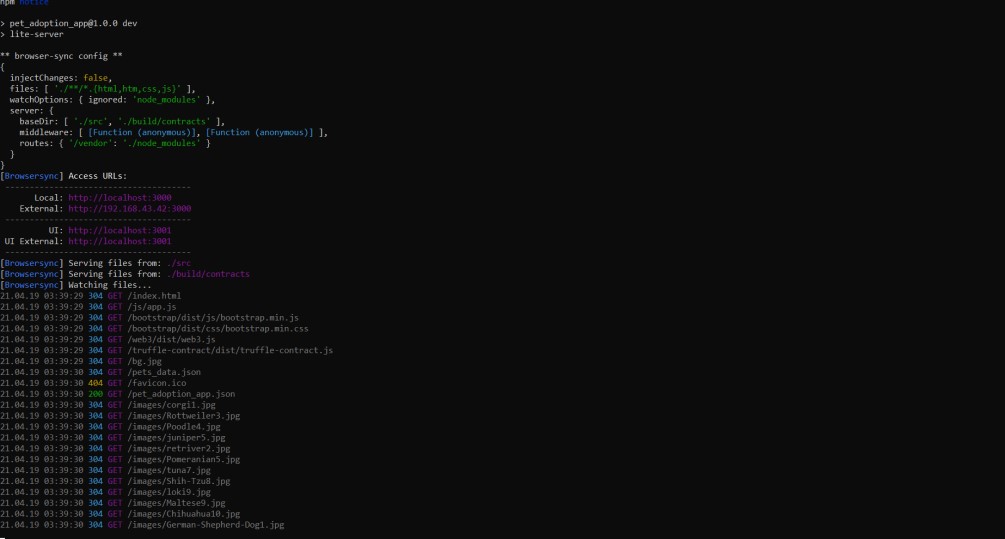
Transaction will be shown over here after success of adoption. This will be providing 100 ethers for adoption.



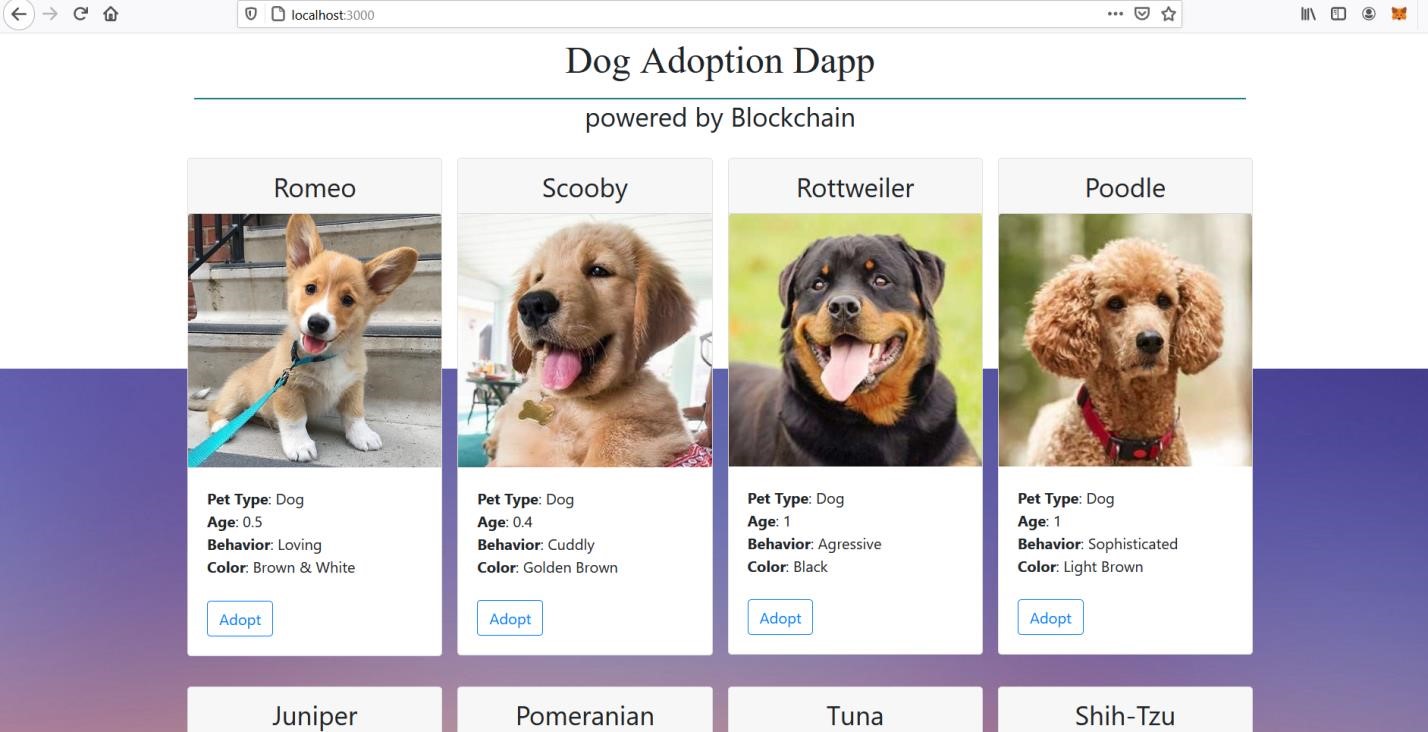
Truffle compile and migratation



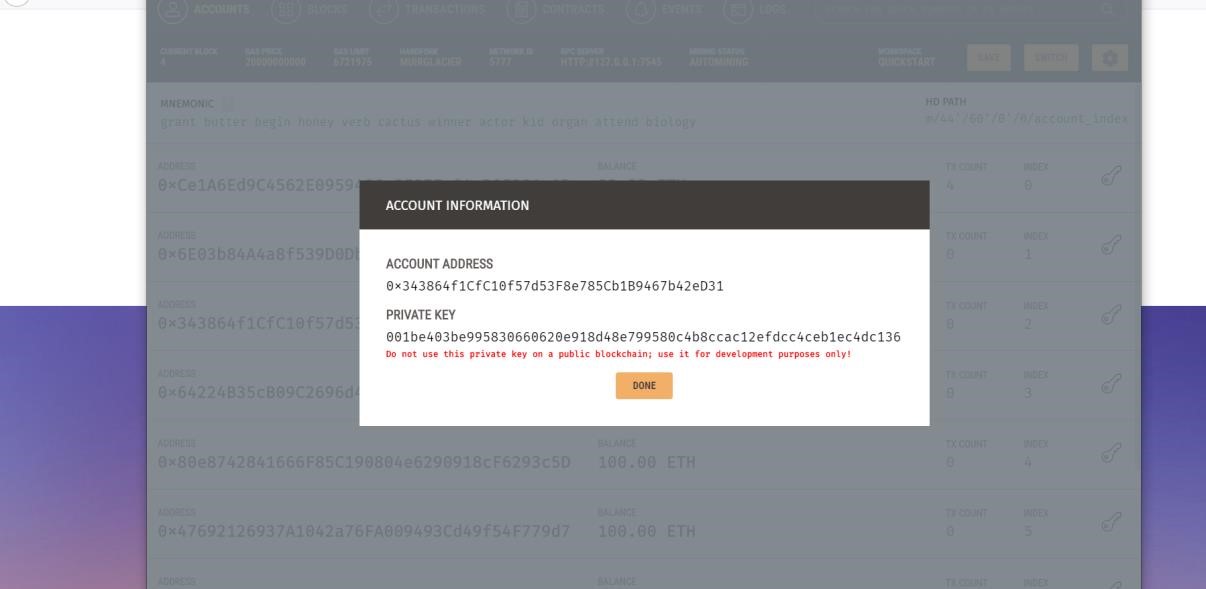
**truffle compiled successfully.**



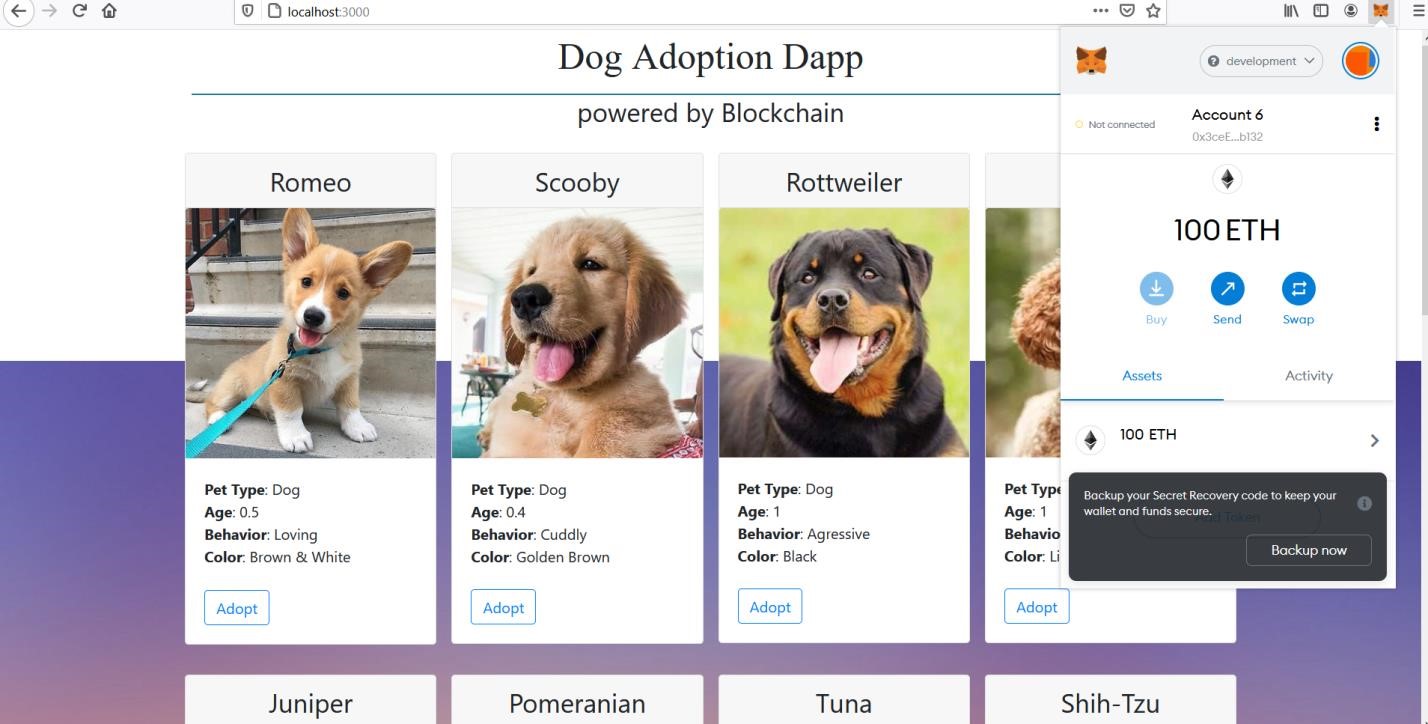
**Local host3000 site for dog adoption opened after truffle compilation. From here we can adopt dog using blockchain using ehtereum.**



**First we need to import any one account in metamask using private key show over here.this will import 100 ethers to our account used for adopting dogs.**



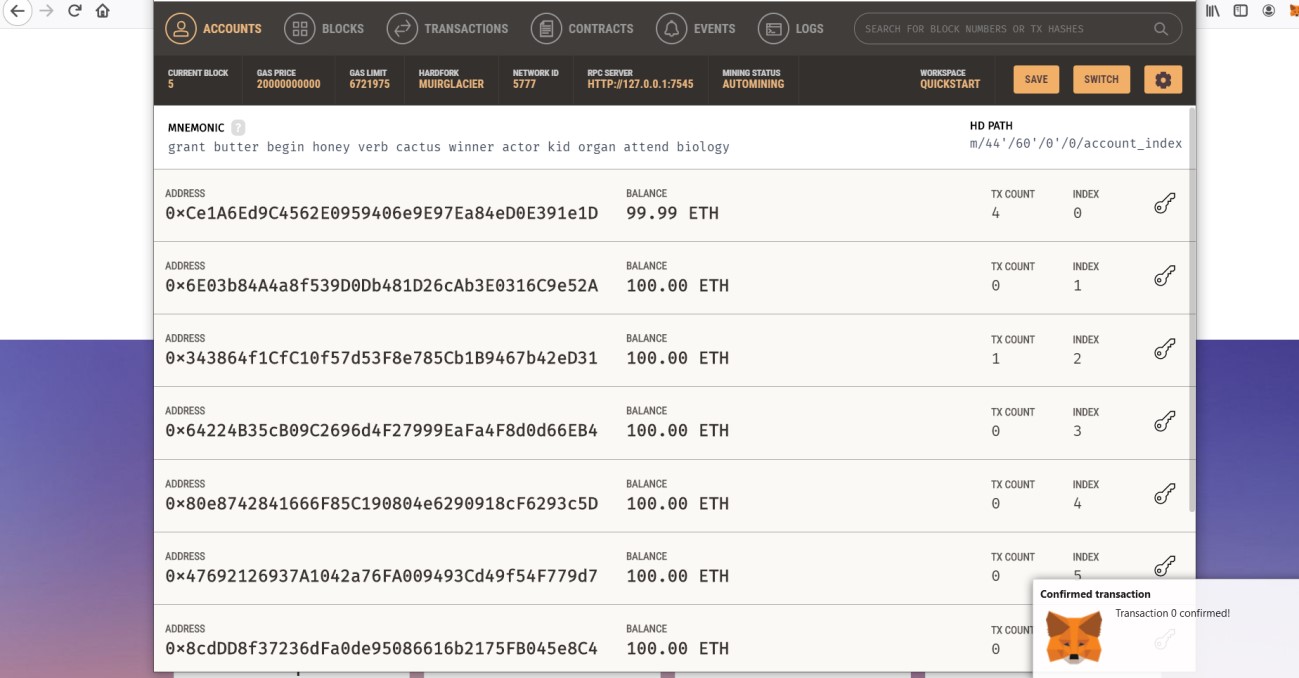
**After importing private key , we will get 100ETH in our account as shown**



**Here dog adoption using blockchain successfully implemanted. As we can see success on Rottweiler Dog and transaction 0 confirmed on metamask as shown.**



**We can see new blocks after adopting dog and .1 ETH used as shown in first account**



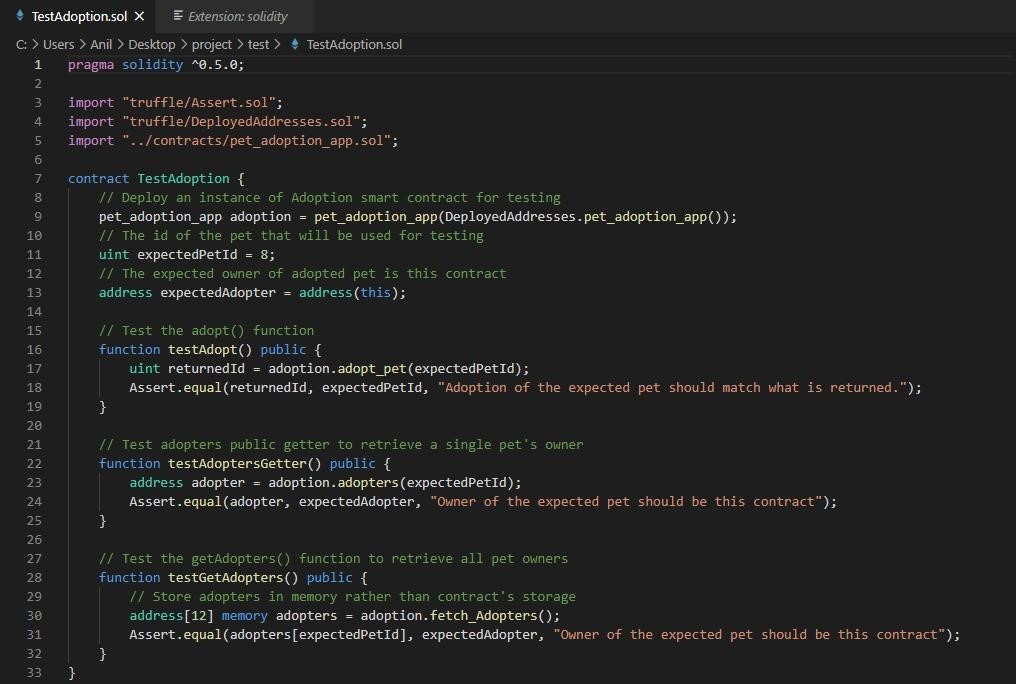
This is fully implementation of project Dog adoption using blockchain.

# 0utput validation and comparison:-

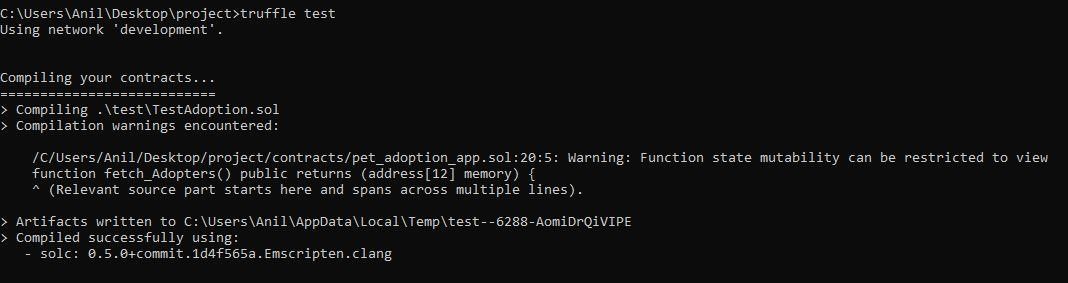
**Test cases:**

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





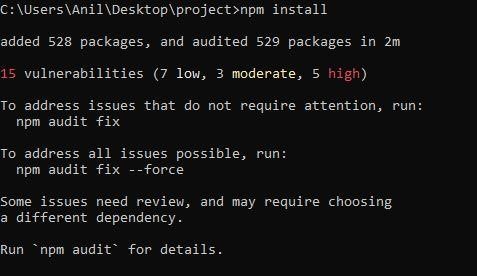


**BACKEND:**

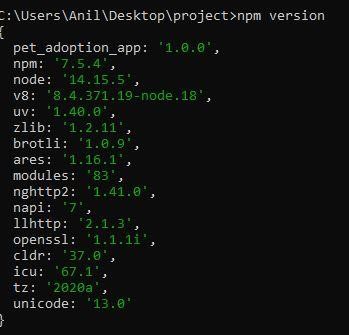
At the backend we will begin by setting up the development environment and installing necessary software tools. We have installed Node.js and on node.js command line we will be running all our commands.

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

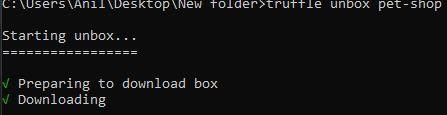
packages npm installs:



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

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

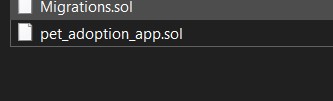
For our project we will installing all the required packages by npm(node packet manager) by downloading truffle-box named as: pet-shop which includes the basic project structure.



1. **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.

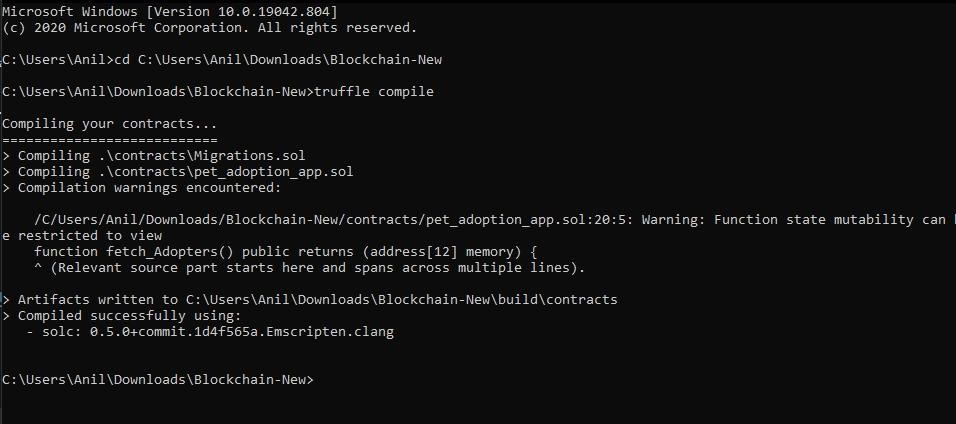
Below is the screen shot of the contract.



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.

**Application ,Compared and solutions:-**

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.

An application bugs exploitation emerges when there is an error in the smart contract code. This exploitation mainly occurs in smart contracts. It arises when developers fail to identify code errors in the decentralized application. Attackers are able to drain all the money from the contract wallet through simple code bugs. Smart contract applications are similar to web applications that run over the blockchain. Like web application bugs, they also comprise errors, however, these bugs can lead to serious challenges. For example, the DAO was able to raise $150m, whilst the attacker was able to steal about $60m due to code bugs.

# Team Work:-

Anil Thakur(18bcs3150)- Backend + 25% documentation.

Aniket singh(18bcs3155)- frontend + 25% documentation.

Priti pandey(18bcs3290)- backend + 25% documentation.

Kashish kumar(18bcs3171)- frontend + 25% documentation.

## Enhanced knowledge of team members:-

This project enhanced knowledge on new technology i.e. blockchain.

In this project team members worked on html, java script for frontend . This enhanced technical skills in java script and html.

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

Will be working with truffle and Writing the smart contract.

We worked on Node.js and on node.js command line, we will be running all our commands.

Enhanced knowledge on these software:-

* Node.js
* Ganache
* Truffle
* Metamask