Installation

- Node js
- Npm
- Ganache
- Truffle
    - Npm install -g truffle
    - truffle version

Steps

1. Open the dir(pet-shop-daap) in VSCode
2. Truffle initialize

   → truffle init / truffle unbox pet-shop(used this)
3. Install Extension (Solidity) in VSCode
4. Adoption.sol

```
   // SPDX-License-Identifier: MIT
pragma solidity ^0.5.16;

contract Adoption {
    address[16] public adopters;

    function adopt(uint petId) public returns (uint) {
        require(petId >= 0 && petId <= 15, "Invalid pet ID");
        adopters[petId] = msg.sender;
        return petId;
    }

    function getAdopters() public view returns (address[16] memory) {
        return adopters;
    }
}
```

```
B. UserRegistration.sol
// SPDX-License-Identifier: MIT
pragma solidity ^0.5.16;

contract UserRegistration {
    mapping(address => string) public userEmails; // Mapping to store user
emails

    // Function to register a user
    function registerUser(string memory email) public {
        require(bytes(email).length > 0, "Email cannot be empty");
        userEmails[msg.sender] = email;
    }

    // Function to retrieve the user's email
    function getEmail() public view returns (string memory) {
        return userEmails[msg.sender];
    }
}
```

    a. truffle compile

5. 2_deploy_contracts

```
      var Adoption = artifacts.require("Adoption");

module.exports = function(deployer) {
  deployer.deploy(Adoption);
};
```

```
B. 3_deploy_contracts
      const UserRegistration = artifacts.require("UserRegistration");

module.exports = function(deployer) {
    deployer.deploy(UserRegistration);
};
```

6. Open Ganache(quickstart)
7. Open terminal
    a. truffle migrate

8. Create a file (testAdoption.test.js) in test directory

```
const Adoption = artifacts.require("Adoption");

contract("Adoption", (accounts) => {
    let adoption;
    let expectedAdopters;

    before(async () => {
        adoption = await Adoption.deployed();
    });

    describe("adopting a pet and retrieving account addresses",
async () => {
        before("adopt a pet using account[0]", async () => {
            await adoption.adopt(8, {from: accounts[0]});
            expectedAdopters = accounts[0];
        });

        it("can fetch the address of an owner by pet id", async ()
=> {
            const adopter = await adoption.adopters(8); // change
`adopter` to `adopters`
            assert.equal(adopter, expectedAdopters, "The owner of
the pet should be the first account.");
        });

        it("can fetch the collection of all pet owner addresses",
async () => {
            const adopters = await adoption.getAdopters();
            assert.equal(adopters[8], expectedAdopters, "The owner
of the adopted pet should be in the collection.");
        });
    });
});
```

```
truffle test
```

Open app.js (src>js dir)

Update the functions in the app.js file

```
App = {
    web3Provider: null,
    contracts: {},

    init: async function() {
        await App.initWeb3();
        await App.initContracts(); // Initialize both contracts
        await App.loadPets();
        App.bindEvents();
    },

    initWeb3: async function() {
        if (window.ethereum) {
            App.web3Provider = window.ethereum;
            try {
                await window.ethereum.request({ method:
'eth_requestAccounts' });
            } catch (error) {
                console.log("User denied account access");
            }
        } else if (window.web3) {
            App.web3Provider = window.web3.currentProvider;
        } else {
            App.web3Provider = new
Web3.providers.HttpProvider('http://localhost:7545');
        }
        web3 = new Web3(App.web3Provider);
    },

    initContracts: async function() {
        // Initialize Adoption contract
        const adoptionData = await $.getJSON('Adoption.json');
        App.contracts.Adoption = TruffleContract(adoptionData);
        App.contracts.Adoption.setProvider(App.web3Provider);
        // Initialize UserRegistration contract
        const userRegistrationData = await
$.getJSON('UserRegistration.json');
        App.contracts.UserRegistration =
TruffleContract(userRegistrationData);
        App.contracts.UserRegistration.setProvider(App.web3Provider);
    },

    bindEvents: function() {
        $(document).on('click', '.btn-adopt', App.handleAdopt);
    },

    registerUser: function() {
        let email = document.getElementById('email').value;

        web3.eth.getAccounts(async function(error, accounts) {
            if (error) {
                console.log(error);
                return;
            }

            let account = accounts[0];
            const instance = await
App.contracts.UserRegistration.deployed();

            try {
                await instance.registerUser(email, { from: account });
                alert("Registration successful!");
            } catch (err) {
                console.error(err.message);
            }
        });
    },

    signInUser: function() {
        let email = document.getElementById('email').value;

        web3.eth.getAccounts(async function(error, accounts) {
            if (error) {
                console.log(error);
                return;
            }

            let account = accounts[0];
            const instance = await
App.contracts.UserRegistration.deployed();

            try {
                const storedEmail = await instance.getEmail.call({ from:
account });

                if (storedEmail === email) {
                    // alert("Sign-in successful!");
                    window.location.href = "main.html"; // Redirect to
main.html on sign-in
                } else {
                    alert("User not registered. Please register first.");
                }
            } catch (err) {
                console.error(err.message);
            }
        });
    },

    handleAdopt: function(event) {
        event.preventDefault();

        var petId = parseInt($(event.target).data('id'));
        var adoptionInstance;

        web3.eth.getAccounts(function(error, accounts) {
            if (error) {
                console.log(error);
            }

            var account = accounts[0];

            App.contracts.Adoption.deployed().then(function(instance) {
                adoptionInstance = instance;
                return adoptionInstance.adopt(petId, { from: account });
            }).then(function(result) {
                console.log('Adoption successful:', result);
                return App.markAdopted();
            }).catch(function(err) {
                console.log('Adoption error:', err.message);
            });
        });
    },

    markAdopted: function() {
        var adoptionInstance;

        App.contracts.Adoption.deployed().then(function(instance) {
            adoptionInstance = instance;
            return adoptionInstance.getAdopters.call();
        }).then(function(adopters) {
            for (let i = 0; i < adopters.length; i++) {
                if (adopters[i] !==
'0x0000000000000000000000000000000000000000') {

$('.panel-pet').eq(i).find('button').text('Success').attr('disabled',
true);
                }
            }
        }).catch(function(err) {
            console.log(err.message);
        });
    },

    loadPets: function() {
        $.getJSON('pets.json', function(data) {
            var petsRow = $('#petsRow');
            var petTemplate = $('#petTemplate');

            for (let i = 0; i < data.length; i++) {
                petTemplate.find('.panel-title').text(data[i].name);
                petTemplate.find('img').attr('src', data[i].picture);
                petTemplate.find('.pet-breed').text(data[i].breed);
                petTemplate.find('.pet-age').text(data[i].age);
                petTemplate.find('.pet-location').text(data[i].location);
                petTemplate.find('.btn-adopt').attr('data-id',
data[i].id);

                petsRow.append(petTemplate.html());
            }
        });
    },
};

// Initialize on page load
$(window).load(function() {
    App.init();
});
```

Next Steps:

Open chrome and install metemask extension

Import wallet
Enter the mnemonic given in ganache in the metamask secret recovery phrase
Add network localhost
    New rpc url: http://127.0.0.1:7545
    Chainid:1337
    Curr: eth

Now run : **npm run dev**
To run the application locally
    - it should open the browser with the interface ready
    - will open a metemesk popup asking to connect to account

If needed : **truffle migrate --reset**
Click adopt button will ask for approval in the metamask wallet

index.html code:

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Pet Home - Sign In</title>
    <link href="css/bootstrap.min.css" rel="stylesheet">
</head>
<body>
    <div class="container">
        <h1 class="text-center">Welcome to Pet Home</h1>
        <hr/>
        <div class="row">
            <div class="col-xs-12 col-sm-8 col-sm-push-2">
                <h2 class="text-center">Sign In / Register</h2>
                <form id="registrationForm" onsubmit="return false;">
                    <div class="form-group">
                        <input type="text" id="name" class="form-control"
placeholder="Enter your name" required />
                    </div>
                    <div class="form-group">
                        <input type="email" id="email"
class="form-control" placeholder="Enter your email" required />
                    </div>
                    <button type="button" class="btn btn-primary"
onclick="App.registerUser();">Register</button>
                    <button type="button" class="btn btn-success"
onclick="App.signInUser();">Sign In</button>
                </form>
            </div>
        </div>
    </div>

    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></
script>
    <script src="js/web3.min.js"></script>
    <script src="js/truffle-contract.js"></script>
    <script src="js/app.js"></script>

    <script>
```

```
        $(window).load(function() {
            App.init(); // Initialize Web3 connection and MetaMask when
the registration page loads
        });
    </script>
</body>
</html>
```

Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Pet Home - Sign In</title>
    <link href="css/bootstrap.min.css" rel="stylesheet">
</head>
<body>
    <div class="container">
        <h1 class="text-center">Welcome to Pet Home</h1>
        <hr/>
        <div class="row">
            <div class="col-xs-12 col-sm-8 col-sm-push-2">
                <h2 class="text-center">Sign In / Register</h2>
                <form id="registrationForm" onsubmit="return false;">
                    <div class="form-group">
                        <input type="text" id="name" class="form-control"
placeholder="Enter your name" required />
                    </div>
                    <div class="form-group">
                        <input type="email" id="email"
class="form-control" placeholder="Enter your email" required />
                    </div>
                    <button type="button" class="btn btn-primary"
onclick="App.registerUser();">Register</button>
```

```
                    <button type="button" class="btn btn-success"
onclick="App.signInUser();">Sign In</button>
                </form>
            </div>
        </div>
    </div>

    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></
script>
    <script src="js/web3.min.js"></script>
    <script src="js/truffle-contract.js"></script>
    <script src="js/app.js"></script>

    <script>
        $(window).load(function() {
            App.init(); // Initialize Web3 connection and MetaMask when
the registration page loads
        });
    </script>
</body>
</html>
```