

# Security In Computing: Implementation Report

[Group Members](#)

[Implementation Details](#)

[Setup](#)

[Node](#)

[Ganache and MetaMask](#)

[Simulating user interactions](#)

[Seller](#)

[Buyer](#)

[Authority](#)

[Bank](#)

[Bibliography](#)

## Group Members

- Justin Naismith - s3605206 (WED-9:30-10:30)
- Siddhesh Kale - s3854666 (FRI-10:30-11:30)

## Implementation Details

Our J&S Real Estate application is a blockchain marketplace that uses Smart Contracts to facilitate transactions between accounts.

React JS, Solidity and Ganache were used to develop this application. React is the front end, controlling the routes and the user input. React interacts with the blockchain through the MetaMask extension, which allows us to use Ethereum in the browser. The blockchain itself is running on Ganache. Contracts are created using Solidity and tested using Chai.

The most important code is the `Marketplace.sol` and `App.tsx`. These two files contain the code vital to the blockchain running, and contain comments explaining what the code is doing. The majority

of the rest of the code is React components for displaying the data we are transacting in the blockchain marketplace.

## Setup

The instructions on running the application and its dependencies are outlined in the README.md file.

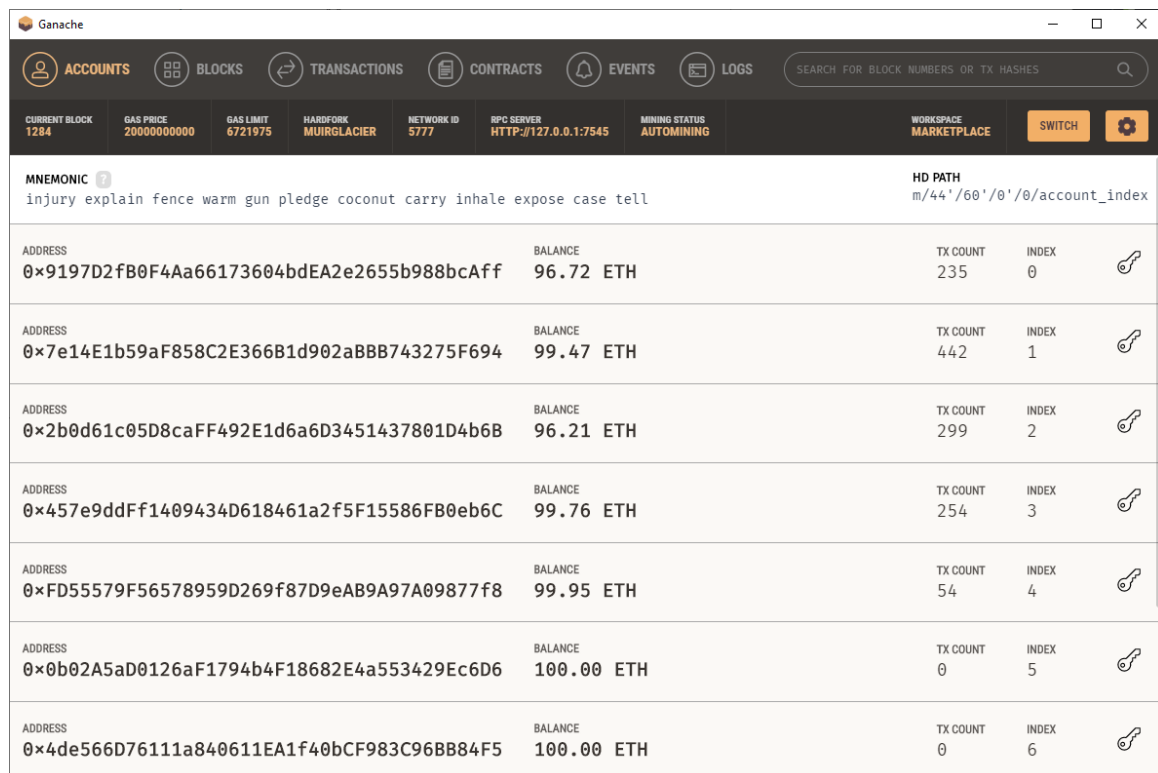
The following steps are required to run the application.

### Node

1. Install node.
2. Install the node modules using the `node i` command.

### Ganache and MetaMask

1. Download and configure Ganache. Found at this [link](#).
2. Run the ganache application, select the create a new workspace option and name its **marketplace**.
3. Press the add project button and select the `truffle-config.js` file and save the workspace.
4. Doing so should present with the following screen:



The screenshot shows the Ganache application window. At the top, there's a navigation bar with icons for ACCOUNTS, BLOCKS, TRANSACTIONS, CONTRACTS, EVENTS, and LOGS. Below this is a status bar with various metrics like CURRENT BLOCK (1284), GAS PRICE (2000000000), GAS LIMIT (6721975), HARDFORK (MUIRGLACIER), NETWORK ID (5777), RPC SERVER (HTTP://127.0.0.1:7545), MINING STATUS (AUTOMINING), and a WORKSPACE MARKETPLACE button. The main area displays a list of accounts with their mnemonics, addresses, balances, transaction counts, and indices.

ACCOUNTS	BLOCKS	TRANSACTIONS	CONTRACTS	EVENTS	LOGS
CURRENT BLOCK: 1284   GAS PRICE: 2000000000   GAS LIMIT: 6721975   HARDFORK: MUIRGLACIER   NETWORK ID: 5777   RPC SERVER: HTTP://127.0.0.1:7545   MINING STATUS: AUTOMINING   WORKSPACE MARKETPLACE   SWITCH					
MNEMONIC: injury explain fence warm gun pledge coconut carry inhale expose case tell   HD PATH: m/44'/60'/0'/0/account_index					
ADDRESS: 0x9197D2fB0F4Aa66173604bdEA2e2655b988bcAff		BALANCE: 96.72 ETH		TX COUNT: 235	INDEX: 0
ADDRESS: 0x7e14E1b59aF858C2E366B1d902aBBB743275F694		BALANCE: 99.47 ETH		TX COUNT: 442	INDEX: 1
ADDRESS: 0x2b0d61c05D8caFF492E1d6a6D3451437801D4b6B		BALANCE: 96.21 ETH		TX COUNT: 299	INDEX: 2
ADDRESS: 0x457e9ddFf1409434D618461a2f5F15586FB0eb6C		BALANCE: 99.76 ETH		TX COUNT: 254	INDEX: 3
ADDRESS: 0xFD55579F56578959D269f87D9eAB9A97A09877f8		BALANCE: 99.95 ETH		TX COUNT: 54	INDEX: 4
ADDRESS: 0x0b02A5aD0126aF1794b4F18682E4a553429Ec6D6		BALANCE: 100.00 ETH		TX COUNT: 0	INDEX: 5
ADDRESS: 0x4de566D76111a840611EA1f40bCF983C96BB84F5		BALANCE: 100.00 ETH		TX COUNT: 0	INDEX: 6

5. Install the MetaMask browser extension which will be used to interact directly with Ganache. The extension can be downloaded [here](#).
6. Create a test network as shown in the following image:



8. Replace the addresses located in `src/contracts/Marketplace.sol` and `src/utis/addresses.ts` with the ones from your Ganache.

```
src > utis > addresses.ts > ...
1 // Justin's addresses
2 export const Marketplace = '0x9197D2fB0F4Aa66173604bdEA2e2655b988bcAff';
3 export const Seller = '0x7e14E1b59aF858C2E366B1d902aBBB743275F694';
4 export const Authority = '0x2b0d61c05D8caFF492E1d6a6D3451437801D4b6B';
5 export const Buyer = '0x457e9ddFf1409434D618461a2f5F15586FB0eb6C';
6 export const Bank = '0xFD55579F56578959D269f87D9eAB9A97A09877f8';
7
8 // Siddhesh's addresses
9 // export const marketplace = '0x728Ad52C853e97Bb907F83F842043567266e3483';
10 // export const Seller = '0x386c0b9C66a334cEedf0059b1B4E44E43C2821a6';
11 // export const Buyer = '0x728Ad52C853e97Bb907F83F842043567266e3483';
12 // export const Authority = '0x2E11fF485F0B543222e5cb392395f3ec748E37B8';
13 // export const Bank = '0x5c5857f8cEB15DA1DF8d0217EcA61c8B19db501E';
```

```
constructor() public {
  name = 'SICI Real Estate Marketplace';
  // Justins addresses
  marketplace = 0x9197D2fB0F4Aa66173604bdEA2e2655b988bcAff;
  seller = 0x7e14E1b59aF858C2E366B1d902aBBB743275F694;
  authority = 0x2b0d61c05D8caFF492E1d6a6D3451437801D4b6B;
  buyer = 0x457e9ddFf1409434D618461a2f5F15586FB0eb6C;
  bank = 0xFD55579F56578959D269f87D9eAB9A97A09877f8;

  // Sids addresses
  // seller = 0x386c0b9C66a334cEedf0059b1B4E44E43C2821a6;
  // authority = 0x2E11fF485F0B543222e5cb392395f3ec748E37B8;
  // buyer = 0x728Ad52C853e97Bb907F83F842043567266e3483;
  // bank = 0x5c5857f8cEB15DA1DF8d0217EcA61c8B19db501E;
}
```

9. Run the following commands in the terminal.

```
truffle compile
truffle migrate
truffle deploy
npm start
```

## Simulating user interactions

### Seller

- To simulate Seller actions, swap to the seller account on MetaMask and refresh the page.

- If the Seller is logged in successfully the above should be visible.
- A seller can then apply for a permit by providing the necessary information.

## Buyer

- To simulate Buyer actions, swap to the seller account on Meta mask and refresh the page.

- If the Buyer is logged in successfully the above should be visible.
- A buyer can then apply for a loan by providing the necessary information.

## Authority

- To simulate Authority actions, swap to the Authority account on MetaMask and refresh the page.

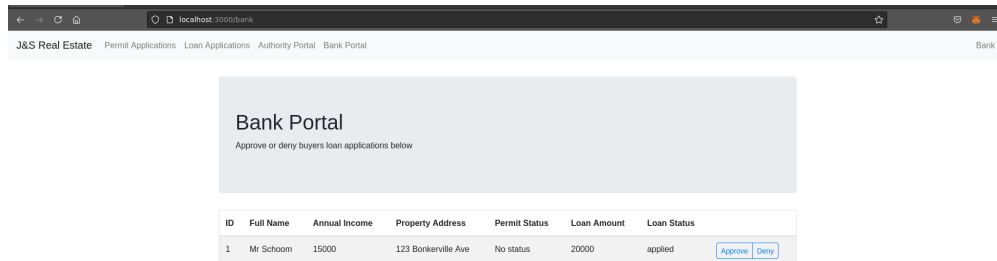
ID	Property Address	Document	Licence Number	Status
1	123 Borkenville	data.pdf	L1234	approved

- If the Authority is logged in successfully the above should be visible.

- The Authority can then Approve or Deny the the Permit applications.

## Bank

- To simulate Bank actions, swap to the Bank account on MetaMask and refresh the page.



- If the Bank is logged in successfully the above should be visible.
- The Bank can then Approve or Deny the Loan applications.

## Bibliography

- Dapp University. (2019, August 12). Intro To Blockchain Programming (Ethereum, Web3.js & Solidity Smart Contracts) [FULL COURSE]. YouTube. <https://youtu.be/VH9Q2lf2mNo>
- (2019, July 20). dappuniversity/marketplace. GitHub. <https://github.com/dappuniversity/marketplace/tree/part-3>

Security In Computing: Implementation Report