Team 1 Presents

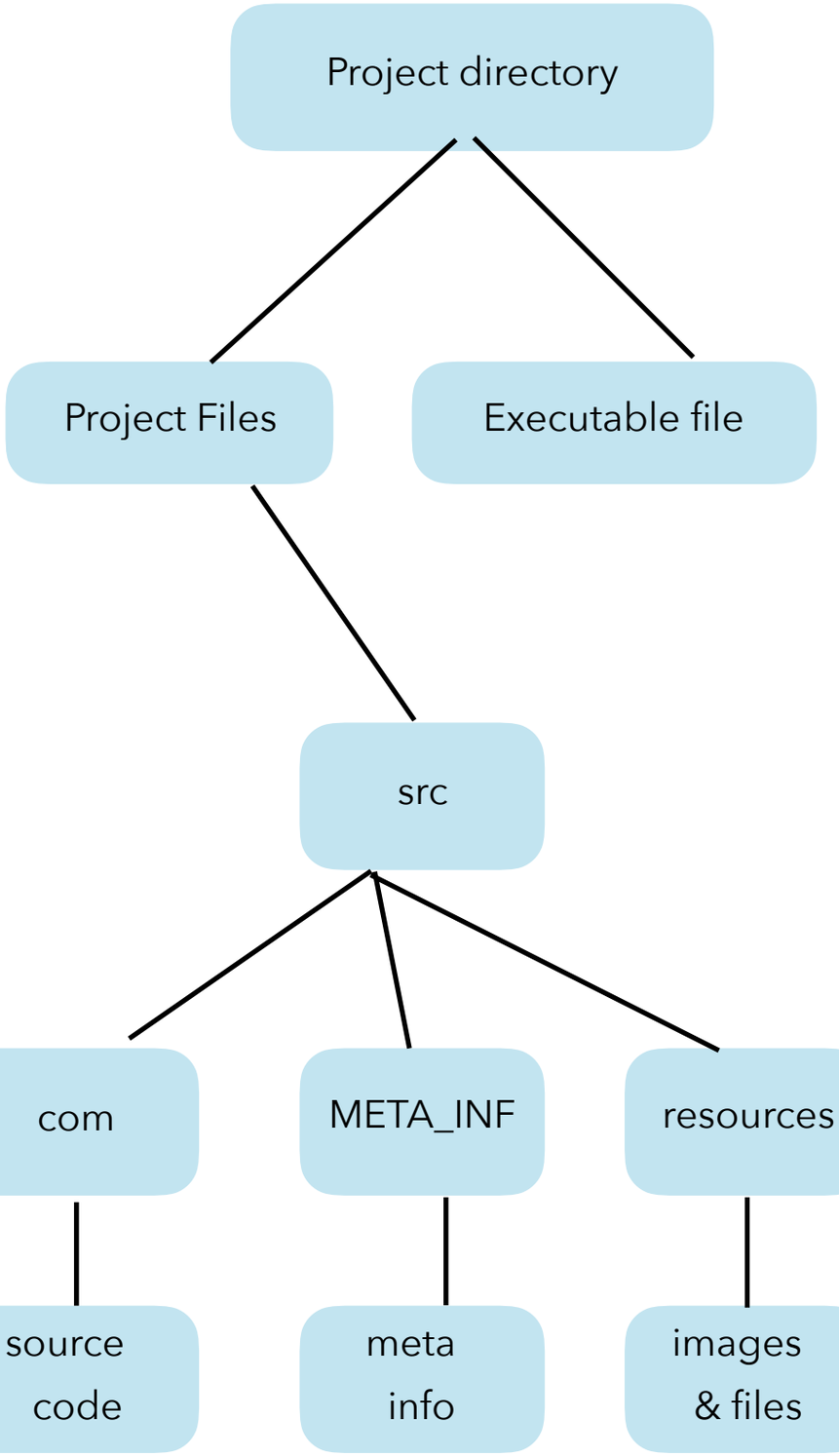
# MEDI-CHAIN

A Blockchain implementation for the medical profession

# TEAM MEMBERS

- Parveen .
  2018A7PS0623H
- Anirudh Agrawal
  2018A7PS0099H
- Ansh Gupta
  2018A7PS0338H
- Dhruv Adlakha
  2018A7PS303H
- Manu Gupta
  2018A7PS0316H

# Project Structure



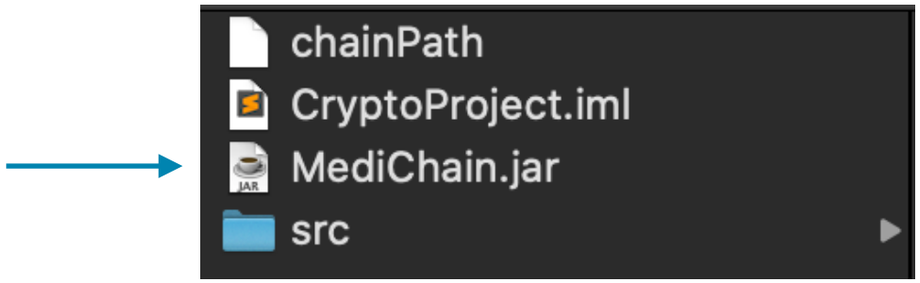

The  project directory is divided into two parts:

- Project files
- An executable jar file

The project files is further divided as:

- src
  - com (contains the source code of the project)
  - META-INF (contains meta info of the project)
  - resources (contains the image and files used in project)

# Running the app

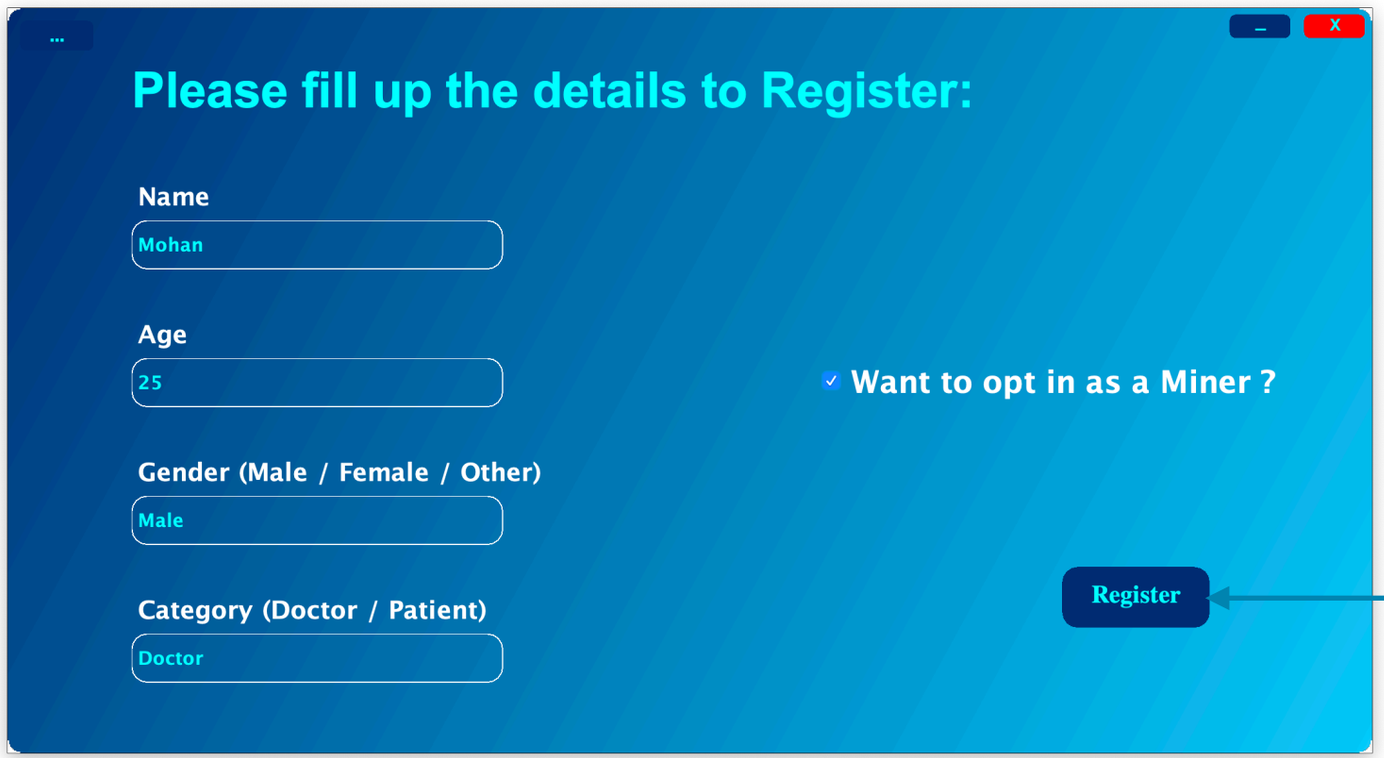



To run the app, you need to double click the ‘MediChain.jar’ executable file present inside the project folder.


When the app starts running, you will see the following welcome screen.

# Registering as a new user


If you are a first time user, or you want to make a new account press the ‘New User ?’ button.

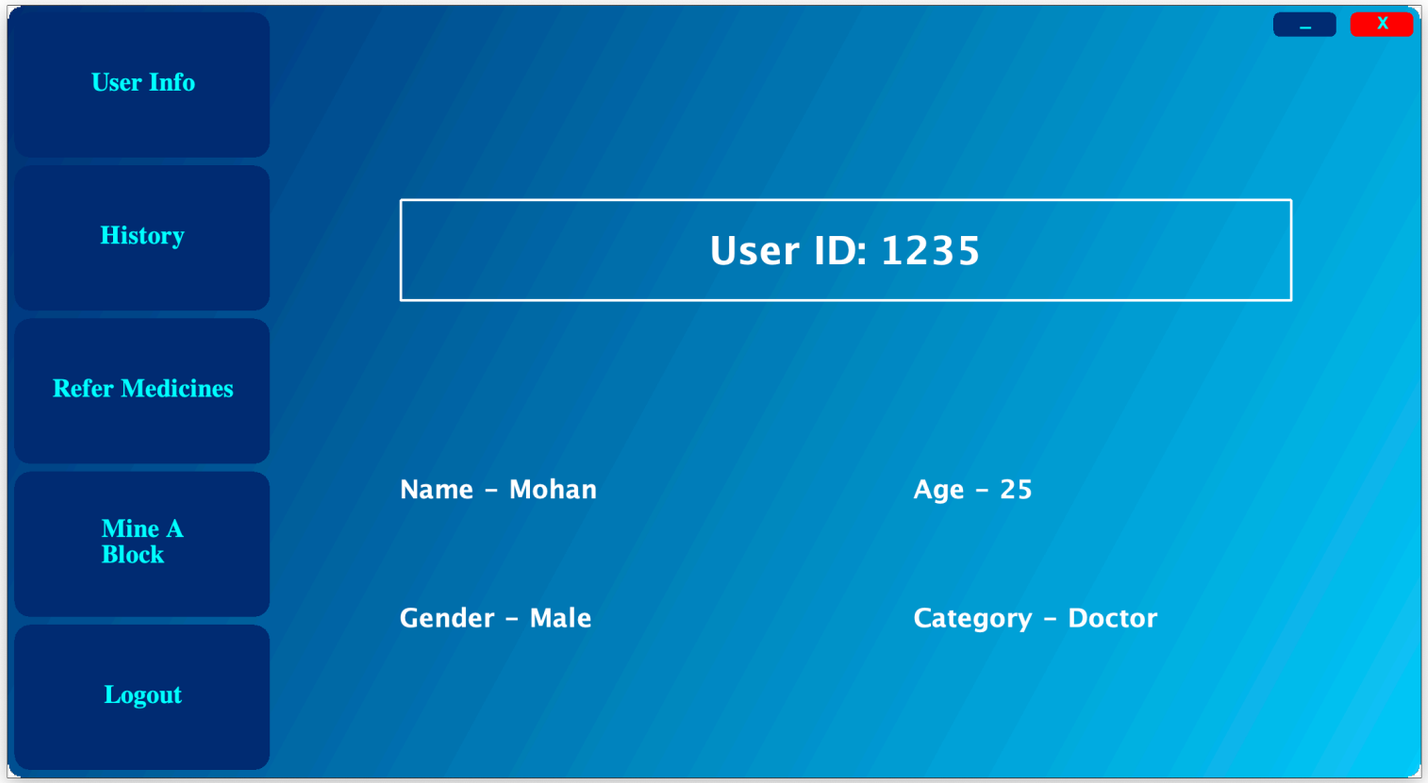



Fill in all of your details appropriately. You may opt in as a miner of blockchain by clicking the check box. After filling all the details click the ‘Register’ button.

# Logging in to your account

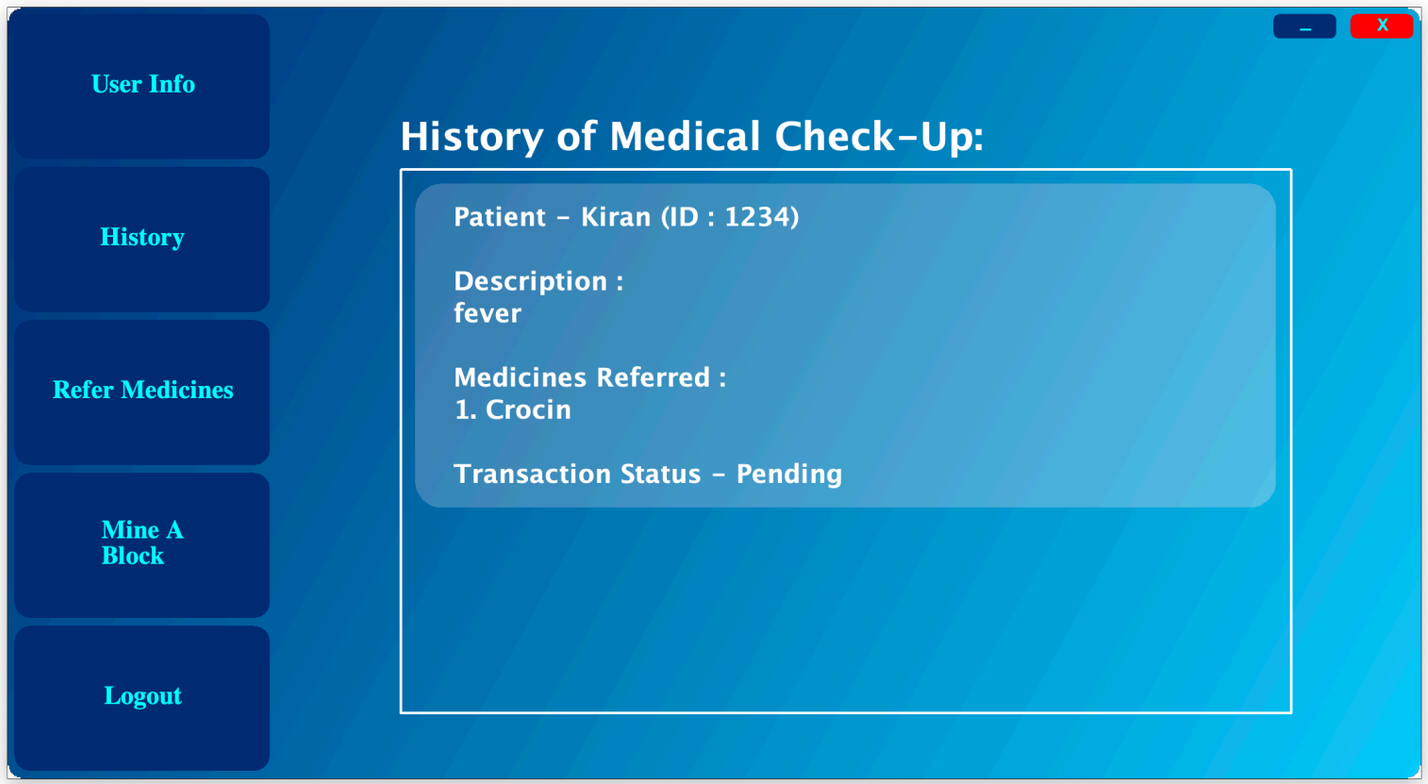

If you are an existing user, you need to enter your user id in the given field and then press the ‘Login’ button to proceed.

# User Home Page




After clicking the 'Register' or 'Login' button, you would redirected to the shown page. You should keep a note of your user id for your future reference or for login.

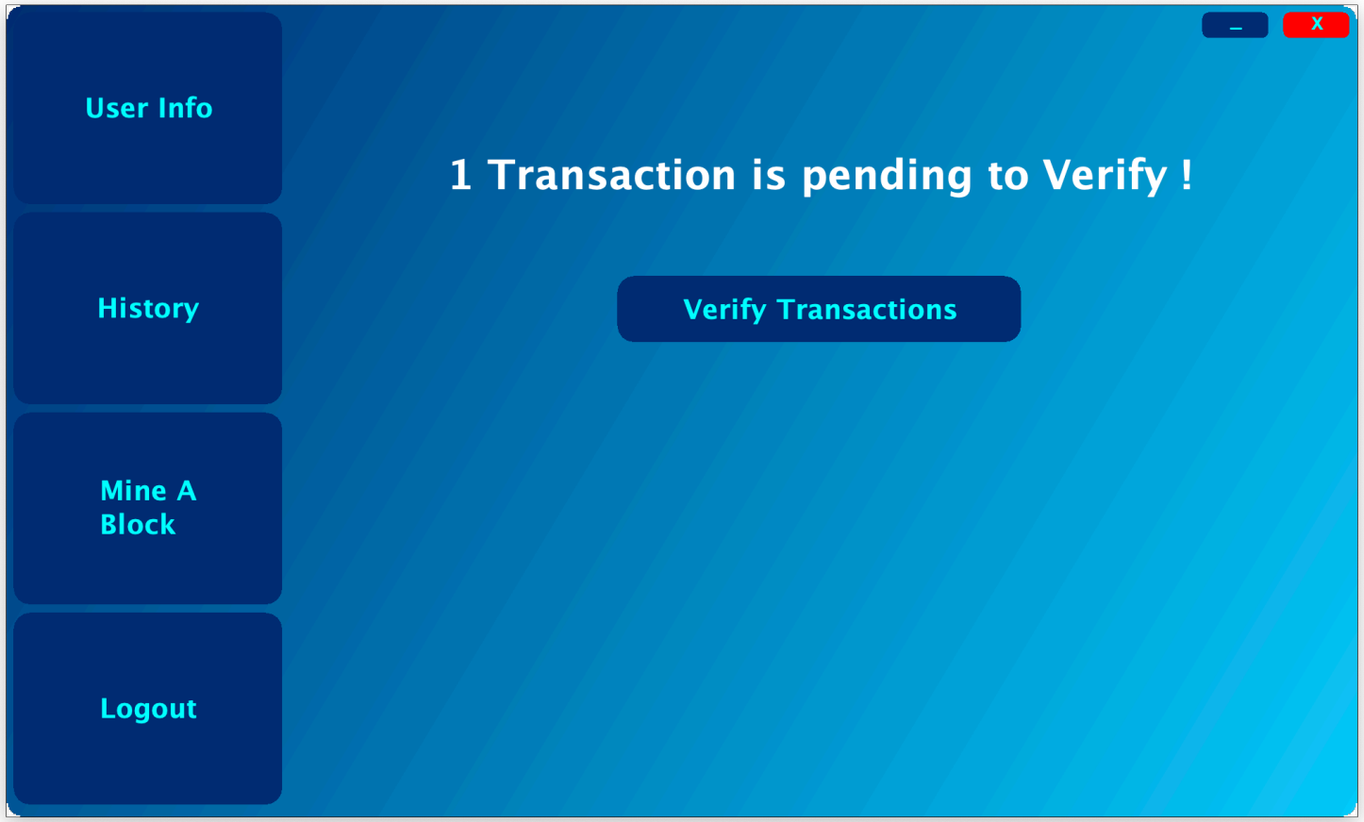
Pressing the 'User Info' button lands you to the shown page. This page shows the summary of the user details.




Pressing the 'History' button takes you to the shown page.

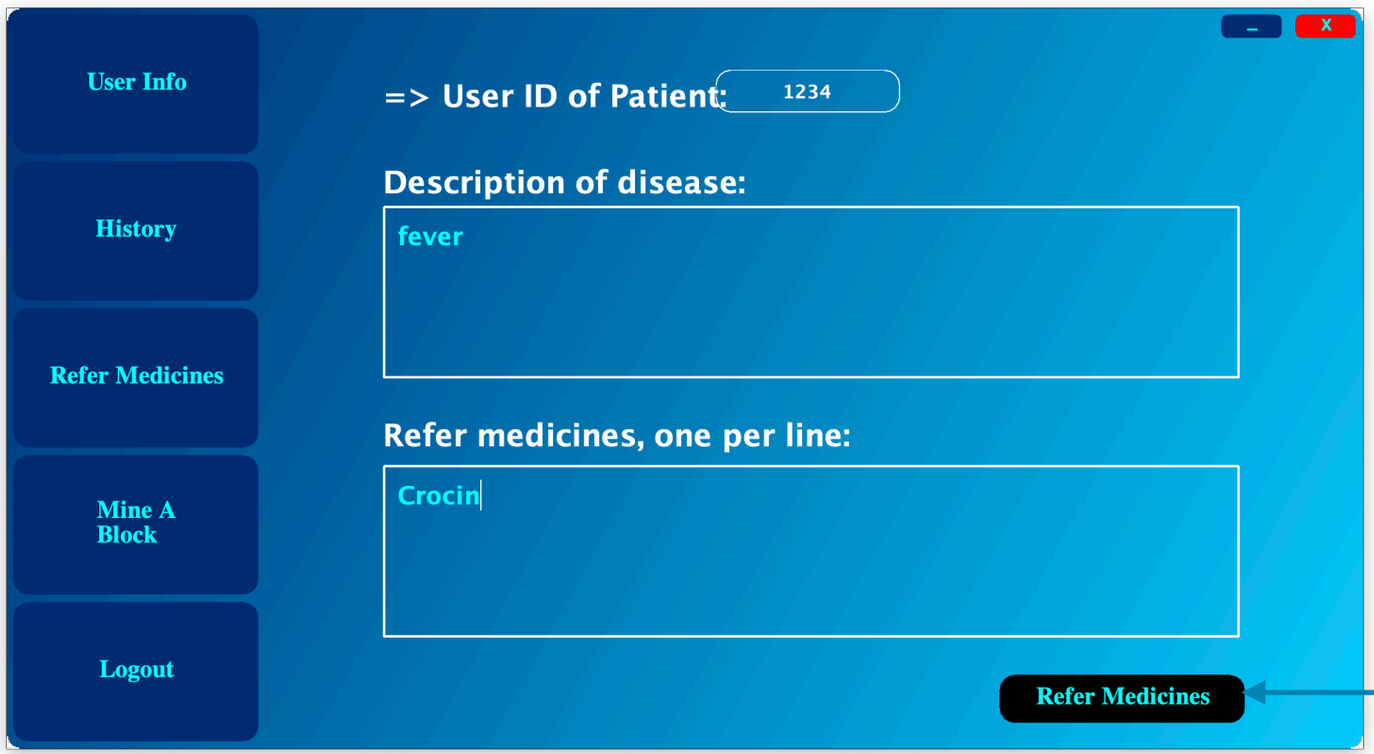
This page consists of all the visits that a patient has made (if the account belongs to the patient) or all the treatments that the doctor has made so far.
(Note: Only successful and pending transactions would be visible.)

# User Home Page (optional)




If you have opted as a miner, then clicking 'Mine a Block' button lands you to this page.

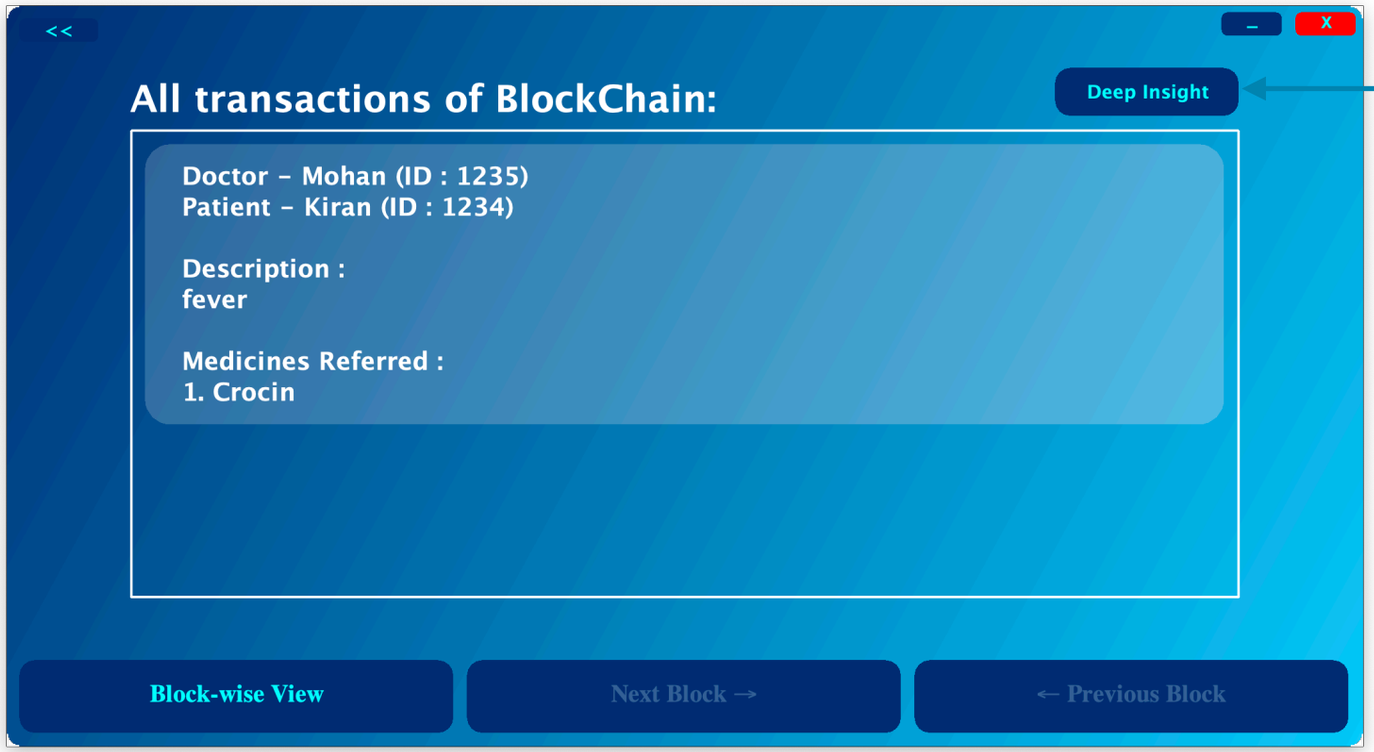
On clicking the 'Verify Transactions' button the mining process will begin and all the pending transactions would be first verified and then saved in the block which furthers gets mined and joins the blockchain.
(Note: This process may take a while depending on the number of pending transactions.)




If you are a doctor, you may prescribe medications to a patient by pressing the 'Refer Medicine' button and filling the required details. After entering the details you should press 'Refer Medicines' in order to add transaction.

Pressing the 'Logout' page will take you to the welcome screen.

# Viewing the whole blockchain



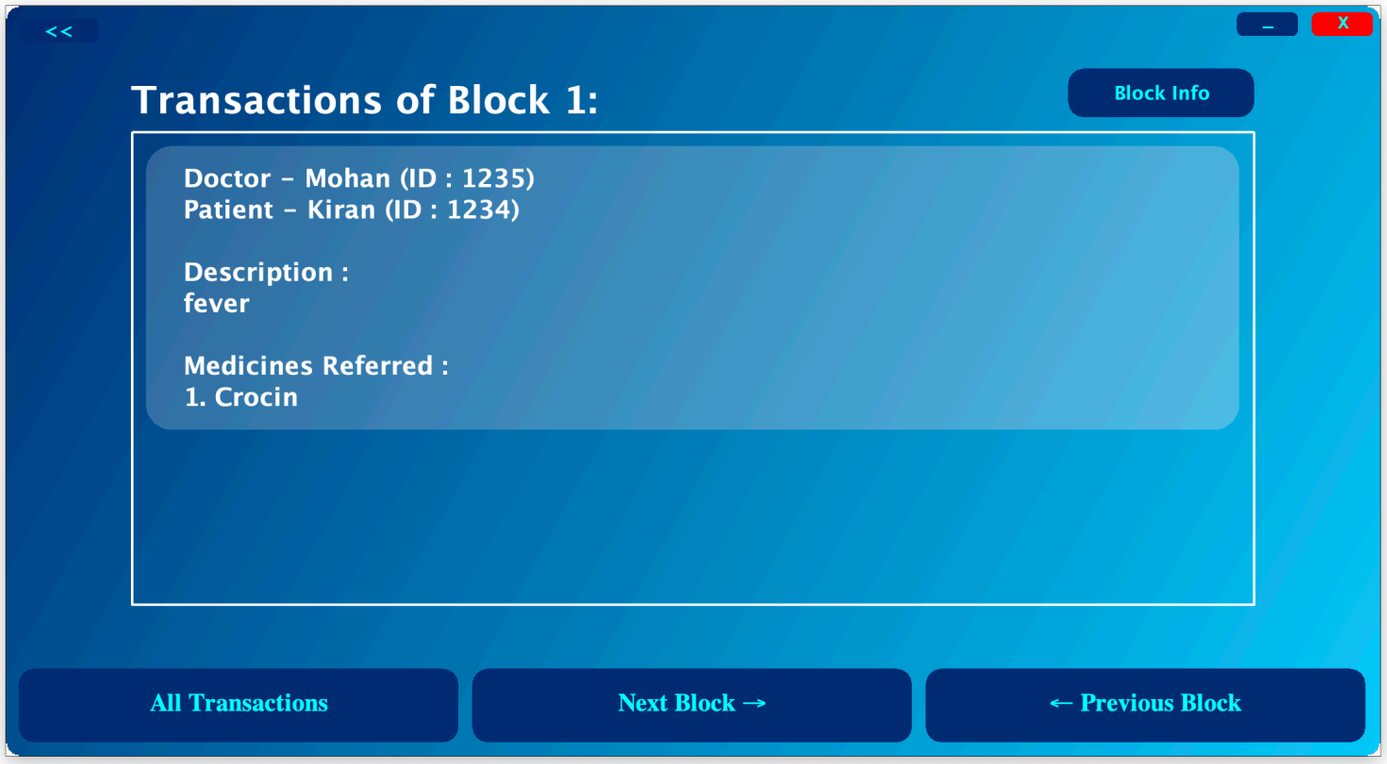

On clicking the 'View BlockChain' button, present on the welcome screen, you are taken to this page. Here you can see all the transactions present in the blockchain.


On clicking the 'Deep Insight' button you can see the following page. Here you can view:

- Prescription id
- Time Stamp
- Sender's address
- Receiver's address

# Viewing the whole blockchain(as blocks)


On pressing the 'Block-wise View' button, you can see the information related to the block. After pressing 'Transactions' button, you can view the transactions local to that block

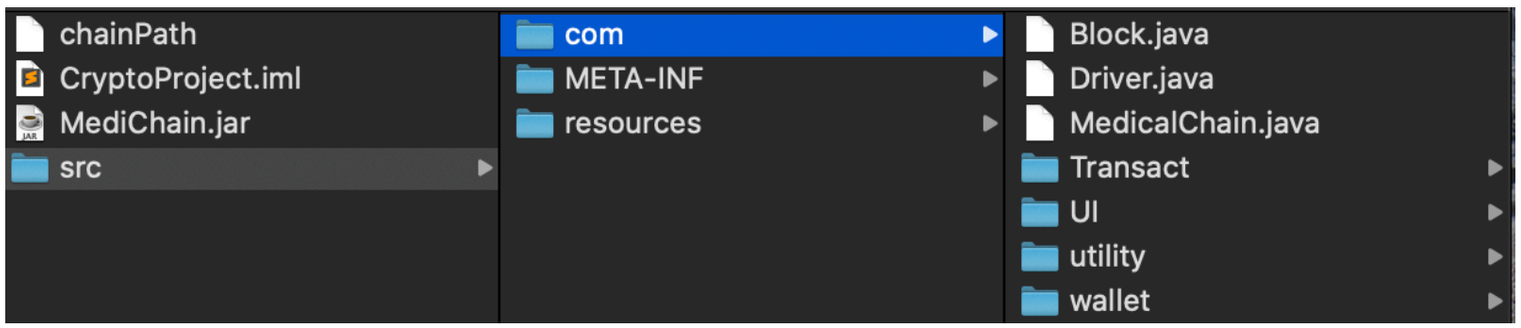



You may use the 'Next Block' and 'Previous Block' buttons to view the information regarding the respective blocks.

(Note: there are no transactions in the Genesis Block).

You may press the back button in the upper left corner to go to welcome screen.

# About the source code




The source code is contained in the src→com directory present inside the project directory.

- Block.java contains the code that contains a class Block and implements methods for:
  - getting transactions
  - adding transaction to the block
  - mining the block

- Driver.java is the driver class of the code, i.e. it contains the main function to run the code.

- UI directory contains various classes that are responsible for all the UI present in the app. The code present in this directory acts as an interface between user and blockchain.

- Transact directory contains Transaction.java which contains all the information related to a transaction (a visit to a doctor). It implements methods for:
  - taking user signature
  - verifying and processing transaction
  - zero knowledge proof

- The wallet directory contains User.java which contains all the information related to a user. It implements methods for:
  - get details of user
  - generating public and private keys of the user
  - helper functions for applying zero knowledge proof

- MedicalChain.java contains the blockchain, pending transactions and a list of all the users. It is a singleton class. It implements the methods for:
  - verifying the transactions (calls other methods present in Transaction.java)
  - Checking validity of chain

- The utility directory contains CommonConstants.java which contains some commonly used values in the project.

- The utility directory also contains Util.java that implements helper methods for:
  - Merkle tree root
  - hash algorithm
  - digital signature algorithm

# A brief about the implementation

- The basis of this implementation of blockchain is the use of Digital Signature Algorithm.
- Each user gets his/her pair of primary and private keys. Private keys are supposed to be stored on the device of that user (through generateKeys() method present in the User.java).
- A transaction is a visit to a doctor. (Note: a doctor might visit himself/herself).
- Each user may opt to become a miner.
- A block is mined by using the 'nonce method' where the number of prefix zeroes should be equal to the difficulty of the blockchain. (Note: Difficulty might be changed by changing the value of 'difficulty' variable present in the CommonConstants.java. Increasing the difficulty would increase the amount of computation required to mine the block.), by default the value of difficulty is 3.
- The Miner validates the transaction using zero knowledge proof where we use the nonce of the hash of the private key of the user as our secret number. By default zero knowledge proof undergoes 5 cycles for patient and doctor each.
- After the zero knowledge proof the transaction is validated against the signature of the doctor and the patient (using the verifySignature() method present in Transaction.java.
- If the signature of the transaction is validated then the transaction is added to the block that is under the process of mining.