Cryptography (BITS F463)
Project Report

Vamshi Duvva (2019A7PS0095H)
Abhigna Srikala (2019A7PS0047H)

Problem statement

Making a blockchain-based solution to address issues in Healthcare.

The healthcare system has a major loophole in the current system of using handwritten medical records and prescriptions. An individual may produce illicit drugs under a false prescription. Medical bills may be spoofed in order to claim insurance money or corporate reimbursement.

Manual filing of diagnoses may or may not be properly accounted for.
Another drawback of the current system is the way in which an individual's medical history is stored. It is generally done by the manual filing of diagnoses that may or may not be properly accounted for.

A secure way is to use a blockchain-based solution to store medical information.

Blockchain introduction and how blockchain technology solves the problem you have selected

Blockchain is a method of storing data in such a manner that it is impossible to change, hack, or cheat. It is a digital log of transactions that is copied and distributed throughout the network of systems. Blockchain technology is disrupting established industries and the ways in which we interact financially.

Most of the existing healthcare system uses a centralized database system. To store large data it is advisable to use decentralized data storage techniques as well as an information system. There is no automatic attack recovery in central data architectures. The decentralized architecture (blockchain) provides automatic data recovery from different attacks.

Through blockchain technology, this portal maintains a digital copy of patients' medical history, diagnoses, prescriptions, and other records making it more secure. Accountability is maintained as the address of all the concerned parties are recorded at every step, while maintaining the security of each user's medical records.

How have you implemented zero-knowledge-proof in your project?

We used zero knowledge proof to verify the patient password in the Add and View options. We choose two numbers p and g. Here p can be a large prime and g is a generator. For simplicity we have used p=11 and g=2.
Now a variable y is calculated as

1. a random number between 0 and 1 is chosen and $h = g^r mod(p)$ is calculated.
2. a random bit b is generated
3. User is prompted for s=(r+bx)mod(p-1) where x is the password.
4. Now $g^s mod(p)$ should be equal to $hy^b mod(p)$

Both the values must be equal for the patient to be verified. Hence the password of the patient is verified without actually getting to know what it is.

```
public static Boolean zeroKnowledgeProof(int y1)
    {

        Random rand = new Random();
        Scanner sc=new Scanner(System.in);
        System.out.println("\nKindly verify yourself as a user");
        System.out.println("Zero Knowledge Proof");
        System.out.println("Choose a random number between 0 and 9(r): ");
        int r = sc.nextInt();
        System.out.println("computing h=(2^r)(mod p) [ h=(2*" + r + ")(mod " + p + ") ]");
        int h = ((((int)Math.pow(2,r)) % p) + p) % p;
        System.out.println(" h : "  +  h);
        // change to variables i.e p = 11 */
        //int h=sc.nextInt();
        //System.out.println("h is "+ h );
        int b=rand.nextInt(2);
        System.out.println("Random bit(b) is: "+ b );
        System.out.println("compute  s=(r+ b*x)mod(10).Here x is password(known): ");
        // print with the calculated values */
        int s=sc.nextInt();
        System.out.println("s is : " + s);
        int val1=expo(2,s,11);
        int val2=(h*expo(y1,b,11))%11;
        //int val3=(h*expo(y2,b,11))%11;A

        if(val1==val2)
        {
            System.out.println("Zero Knowledge Proof Successful.You are verified as registered user\n");
            return true;
        }

        else
        {
            System.out.println("Zero Knowledge Proof Failed.Please try again\n");
            return false;
        }
    }
```

A detailed explanation of functions written in the code using flowcharts along with some UML diagrams.

createBlock() :- Create a block with information related to a document.

```
public static String createBlock(ArrayList<Record> record,String previousHash,Record data)
{
    System.out.println("creating block...");

    Block block = new Block(record, previousHash,data);

    block.mineBlock(difficulty);
    if (verifyTransaction(block))
    {
        blockchain.add(block);
    }

    return block.getBlockHash();
}
```

verifyTransaction() :-It will verify whether the person trying to access the documents is authorized or not.

```
public static boolean verifyTransaction(Block block)
{
    for(int i=1;i<blockchain.size();i++)
    {
        if(!(blockchain.get(i).getPreviousHash()==blockchain.get(i-1).getBlockHash()))
        {
            return false;
        }

    }

    if(blockchain.size()>0)
    {
        if (!(blockchain.get(blockchain.size()-1).getBlockHash() == block.getPreviousHash()))
        {
            return false;
        }

    }
    return true;
}
```

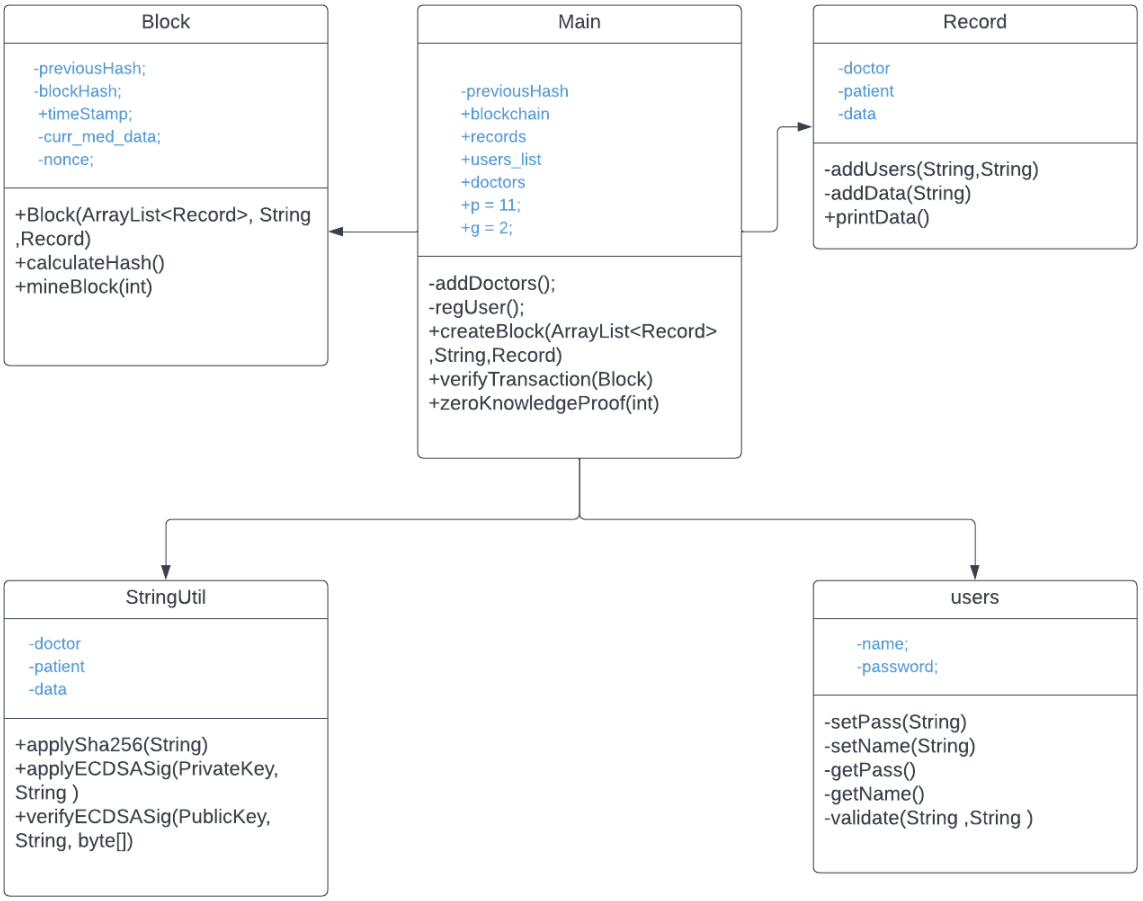
mineBlock() :- It includes new blocks to the blockchain.

```
public void mineBlock(int difficulty) {

    String target = StringUtil.getDificultyString(difficulty); //Create a string with difficulty * "0"

    while(!blockHash.substring( 0, difficulty).equals(target)) {
        nonce ++;
        blockHash = calculateHash();
    }
    System.out.println("Block Mined!!! : ");
    System.out.println("Hashed Block : " + blockHash);
}

public String getPreviousHash() {

    return previousHash;

}
```

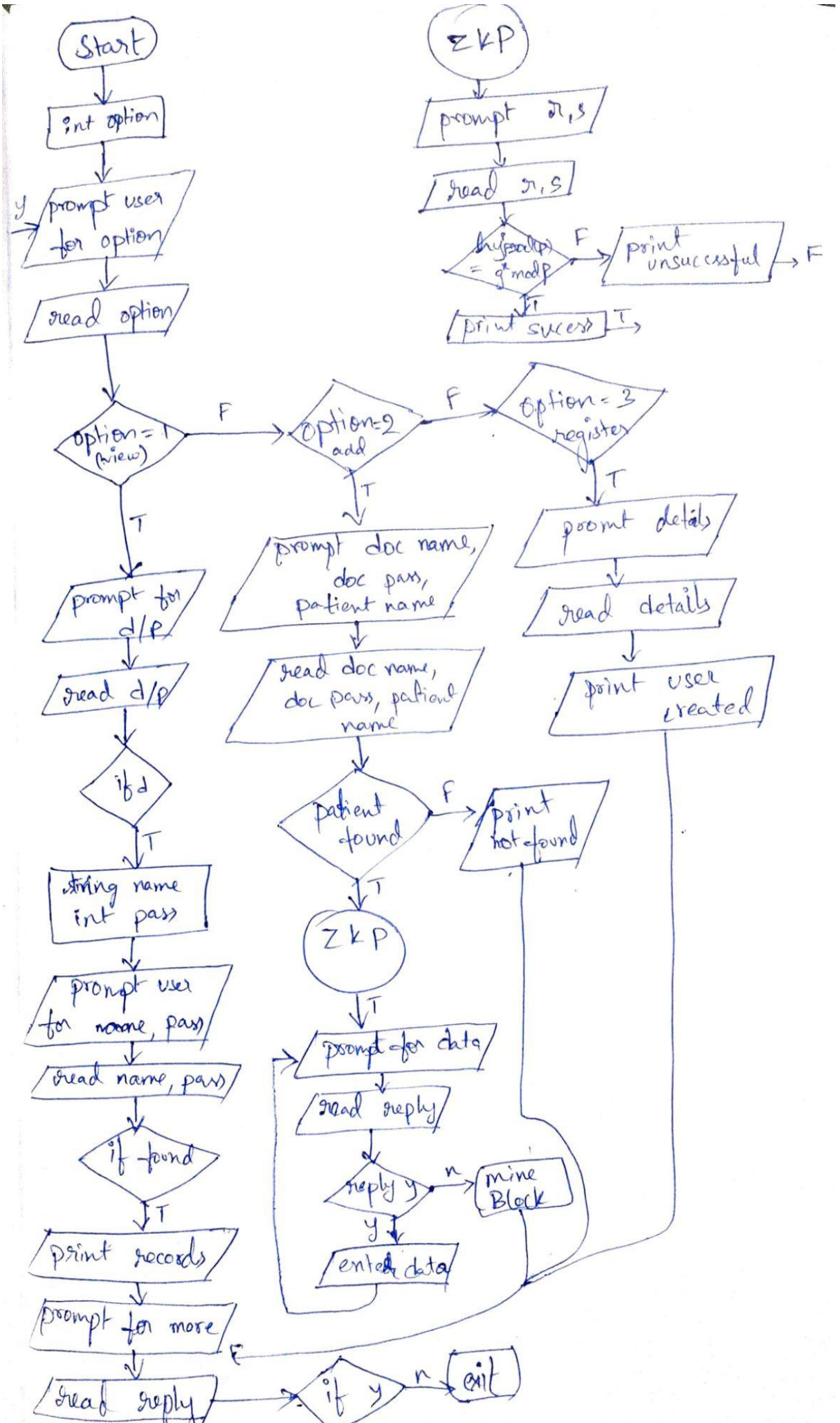
viewUser() :- It lists all the transactions of a user.

```
System.out.println("Are you a doctor or patient?(d/p)");
sc = new Scanner(System.in);
String value = sc.nextLine();
if(value.equals("d"))
{
    sc = new Scanner(System.in);
    System.out.println("Enter your name:");
    String doct = sc.nextLine();
    System.out.println("Enter your pass:");
    String pass = sc.nextLine();
    int f1=0;
    for(int i=0;i<doctors.size();i++)
    {
        if(doctors.get(i).getName().equals(doct)&&doctors.get(i).getPass().equals(pass))
        {
            for(int k=0;k<blockchain.size();k++)
            {
                if(blockchain.get(k).docname().equals(doct))
                {
                    System.out.println("Doctor:"+doct);
                    System.out.println("Patient:"+blockchain.get(k).patient_name());
                    System.out.println("His Medical Data:");
                    System.out.println("TimeStamp at which data  was recorded:"+blockchain.get(k).timeStamp);
                    blockchain.get(k).printData();
                    System.out.println();
                    f1=1;

                }
            }
            if(f1==1)
                break;
        }
    }
    if(f1==0)
        System.out.println("Doctor Not Found");
}
```

UML:


Block
-previousHash;
-blockHash;
+timeStamp;
-curr_med_data;
-nonce;
+Block(ArrayList<Record>, String ,Record)
+calculateHash()
+mineBlock(int)
Main
-previousHash
+blockchain
+records
+users_list
+doctors
+p = 11;
+g = 2;
-addDoctors();
-regUser();
+createBlock(ArrayList<Record> ,String,Record)
+verifyTransaction(Block)
+zeroKnowledgeProof(int)
Record
-doctor
-patient
-data
-addUsers(String,String)
-addData(String)
+printData()
StringUtil
-doctor
-patient
-data
+applySha256(String)
+applyECDSASig(PrivateKey, String)
+verifyECDSASig(PublicKey, String, byte[])
users
-name;
-password;
-setPass(String)
-setName(String)
-getPass()
-getName()
-validate(String ,String)

Flowchart :


Start
int option
prompt user for option
read option
option = 1 (view)
F
option=2 add
F
option = 3 register
T
prompt for d/p
read d/p
if d
T
string name int pass
prompt user for name, pass
read name, pass
if found
T
print records
prompt for more
read reply
if y
n
exit
prompt doc name, doc pass, patient name
read doc name, doc pass, patient name
patient found
F
print not found
T
ZKP
T
prompt for data
read reply
reply y
n
mine Block
y
enter data
ZKP
prompt r, s
read r, s
F
print unsuccessful
F
print success
T
prompt details
read details
print user created

Screenshots of your working application.

```
1) view
2) add
3) register
```

Register :

```
3
Enter name:
vamshi
Enter pass:(only integer)
8
Verify pass:
8
User created...
Do you want to continue?(y/n)
```

Zero knowledge proof :

```
Kindly verify yourself as a user
Zero Knowledge Proof
Choose a random number between 0 and 9(r):
0
computing h=(2^r)(mod p) [ h=(2*0)(mod 11) ]
 h : 1
Random bit(b) is: 1
compute  s=(r+ b*x)mod(10).Here x is password(known):
8
s is : 8
Zero Knowledge Proof Successful.You are verified as registered user
```

Block mining :

```
Enter data: y/n
y
diabetics
Enter data: y/n
y
bp - 140/19
Enter data: y/n
n
creating block...
Block Mined!!! :
Hashed Block : 0000e326186933fa83f0efd581d09409022ec07b73a10f549bbaa6472e8a1175
Do you want to continue?(y/n)
```

Add :

```
2
Enter doc name:
gupta
Enter doc pass:
2
Enter patient name:
vamshi

Kindly verify yourself as a user
Zero Knowledge Proof
Choose a random number between 0 and 9(r):
0
computing h=(2^r)(mod p) [ h=(2*0)(mod 11) ]
 h : 1
Random bit(b) is: 1
compute  s=(r+ b*x)mod(10).Here x is password(known):
8
s is : 8
Zero Knowledge Proof Successful.You are verified as registered user

Enter data: y/n
y
diabetics
Enter data: y/n
y
bp - 140/19
Enter data: y/n
n
creating block...
Block Mined!!! :
Hashed Block : 0000e326186933fa83f0efd581d09409022ec07b73a10f549bbaa6472e8a1175
Do you want to continue?(y/n)
```

View :

```
1
Are you a doctor or patient?(d/p)
d
Enter your name:
gupta
Enter your pass:
2
Doctor:gupta
Patient:vamshi
His Medical Data:
TimeStamp at which data  was recorded:1650909546148
diabetics
bp - 140/19

Do you want to continue?(y/n)
```

```
1
Are you a doctor or patient?(d/p)
p
Enter your name:
vamshi

Kindly verify yourself as a user
Zero Knowledge Proof
Choose a random number between 0 and 9(r):
0
computing h=(2^r)(mod p) [ h=(2*0)(mod 11) ]
 h : 1
Random bit(b) is: 0
compute  s=(r+ b*x)mod(10).Here x is password(known):
0
s is : 0
Zero Knowledge Proof Successful.You are verified as registered user

Time:1650909546148
Doctor:gupta
Patient:vamshi
Patient's Medical Data:
diabetics
bp - 140/19

Do you want to continue?(y/n)
```