ENSET

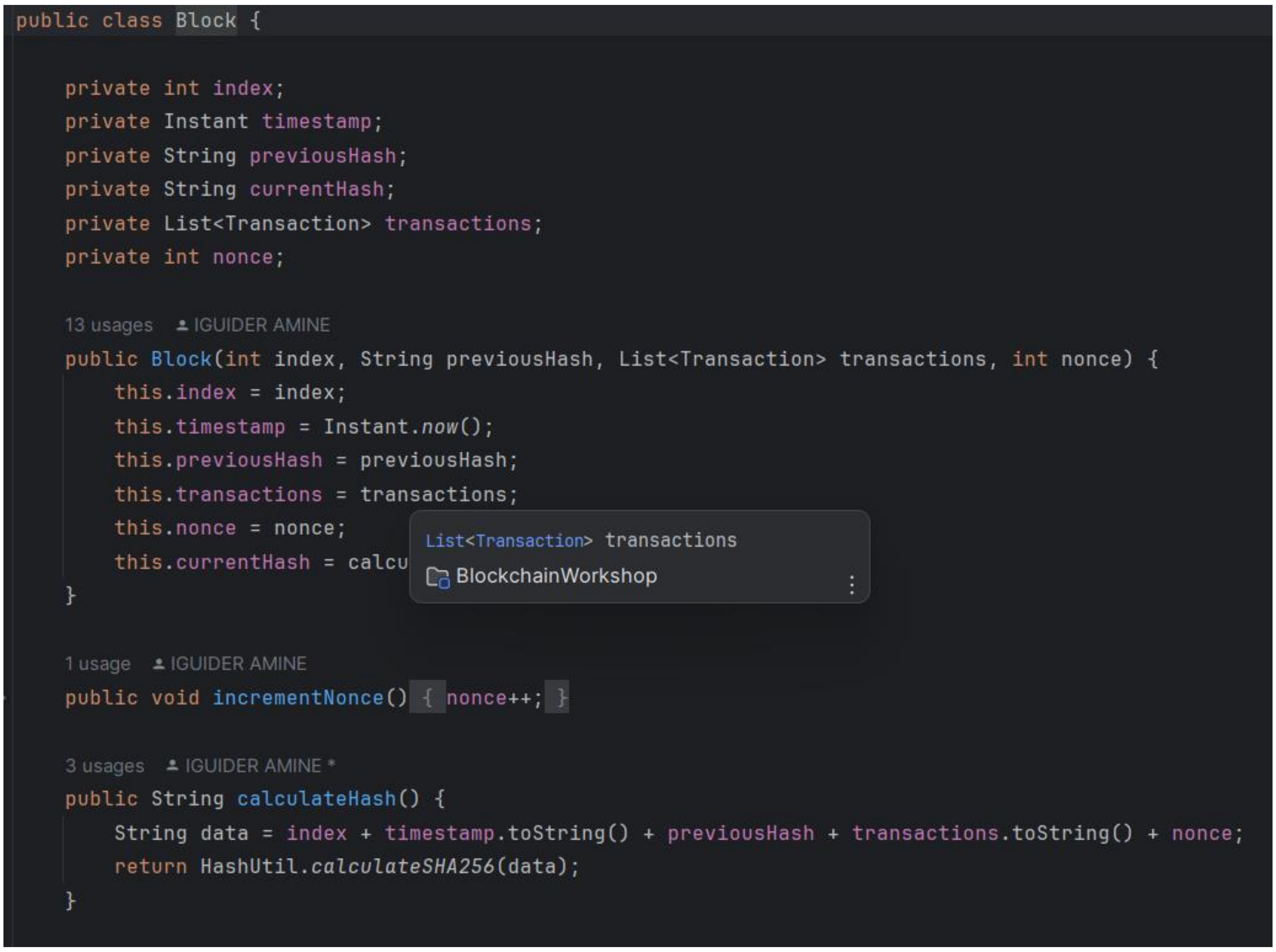
ECOLE NORMALE SUPÉRIEURE DE L'ENSEIGNEMENT TECHNIQUE DE MOHAMMEDIA
UNIVERSITÉ HASSAN II DE CASABLANCA

Rapport

Fait par : IGUIDER Amine

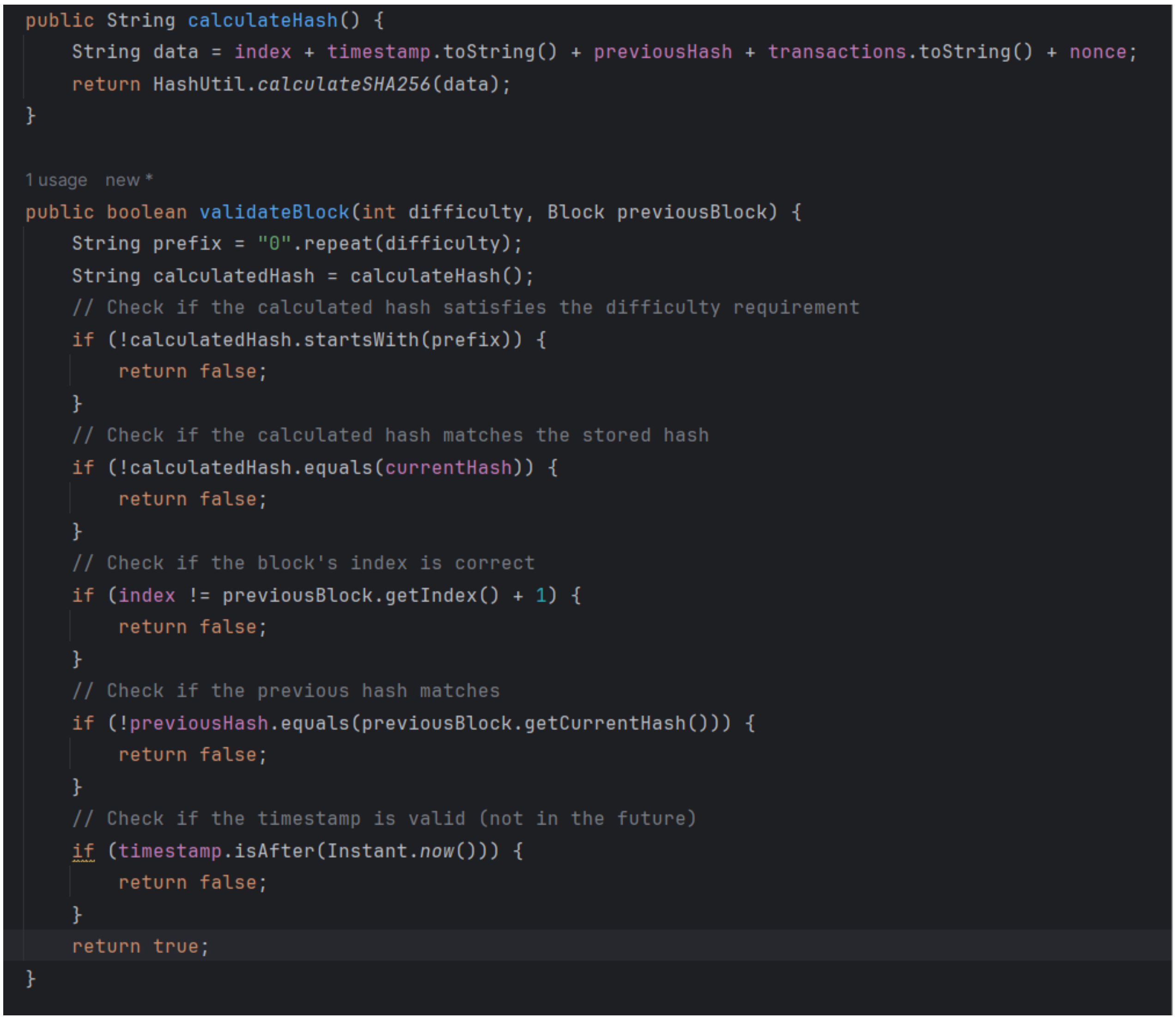
Filière : Master 1 SDIA

1. Block Class



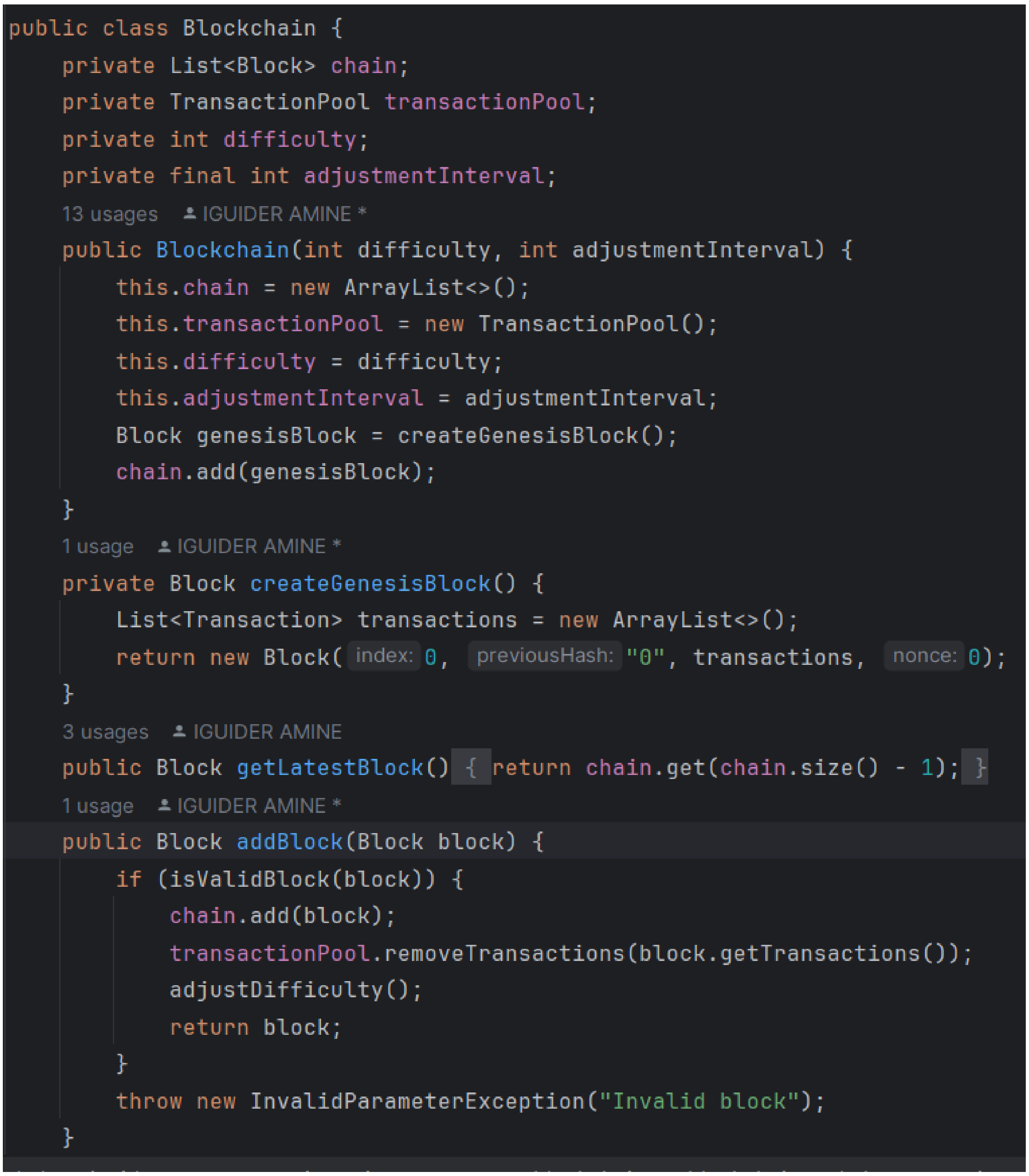
```
public class Block {

    private int index;
    private Instant timestamp;
    private String previousHash;
    private String currentHash;
    private List<Transaction> transactions;
    private int nonce;

    13 usages   IGUIDER AMINE
    public Block(int index, String previousHash, List<Transaction> transactions, int nonce) {
        this.index = index;
        this.timestamp = Instant.now();
        this.previousHash = previousHash;
        this.transactions = transactions;
        this.nonce = nonce;
        this.currentHash = calcu
    }

    1 usage   IGUIDER AMINE
    public void incrementNonce() { nonce++; }

    3 usages   IGUIDER AMINE *
    public String calculateHash() {
        String data = index + timestamp.toString() + previousHash + transactions.toString() + nonce;
        return HashUtil.calculateSHA256(data);
    }
```

```
List<Transaction> transactions
BlockchainWorkshop
```

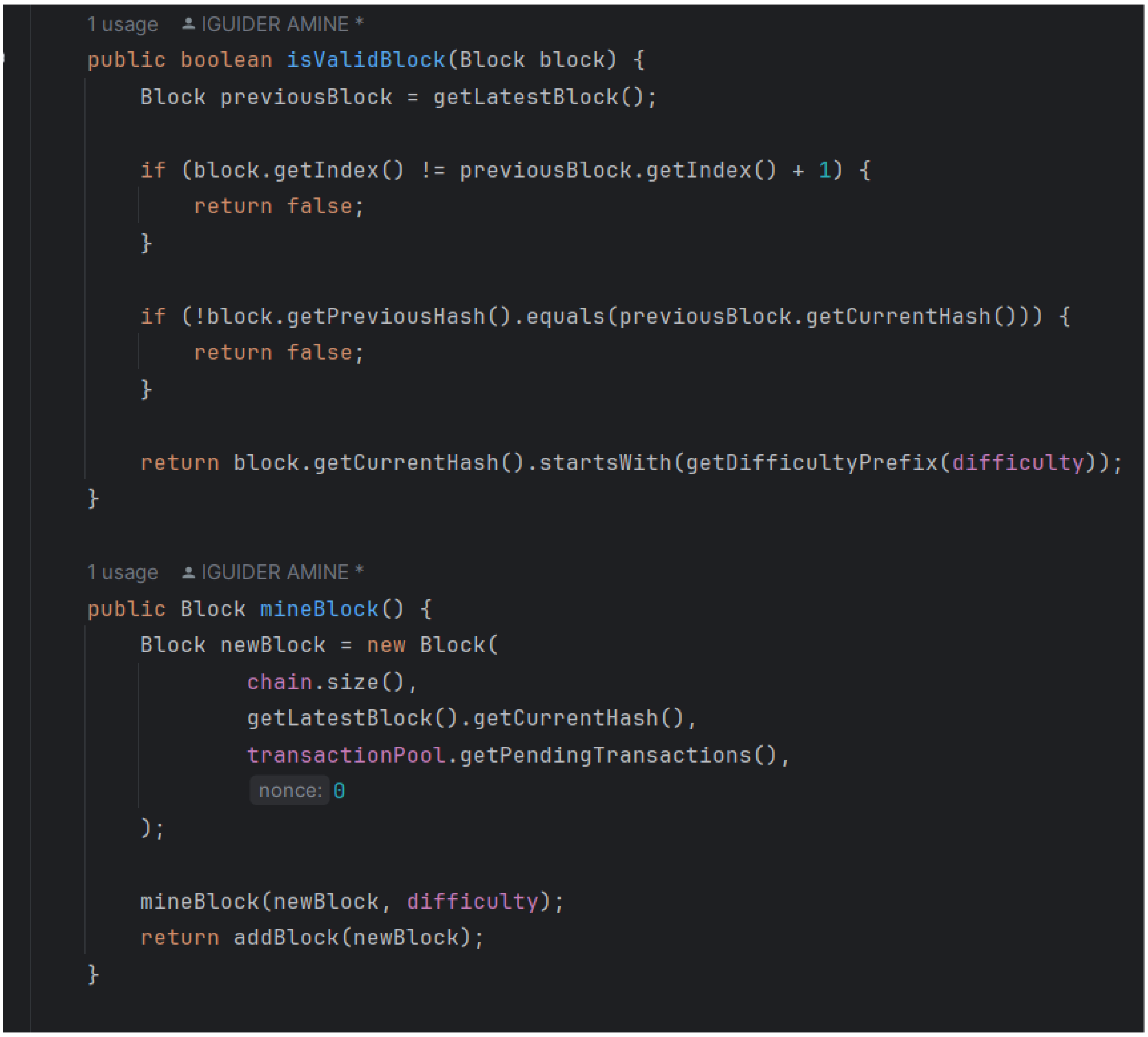


```
public String calculateHash() {
    String data = index + timestamp.toString() + previousHash + transactions.toString() + nonce;
    return HashUtil.calculateSHA256(data);
}

1 usage   new *
public boolean validateBlock(int difficulty, Block previousBlock) {
    String prefix = "0".repeat(difficulty);
    String calculatedHash = calculateHash();
    // Check if the calculated hash satisfies the difficulty requirement
    if (!calculatedHash.startsWith(prefix)) {
        return false;
    }
    // Check if the calculated hash matches the stored hash
    if (!calculatedHash.equals(currentHash)) {
        return false;
    }
    // Check if the block's index is correct
    if (index != previousBlock.getIndex() + 1) {
        return false;
    }
    // Check if the previous hash matches
    if (!previousHash.equals(previousBlock.getCurrentHash())) {
        return false;
    }
    // Check if the timestamp is valid (not in the future)
    if (timestamp.isAfter(Instant.now())) {
        return false;
    }
    return true;
}
```

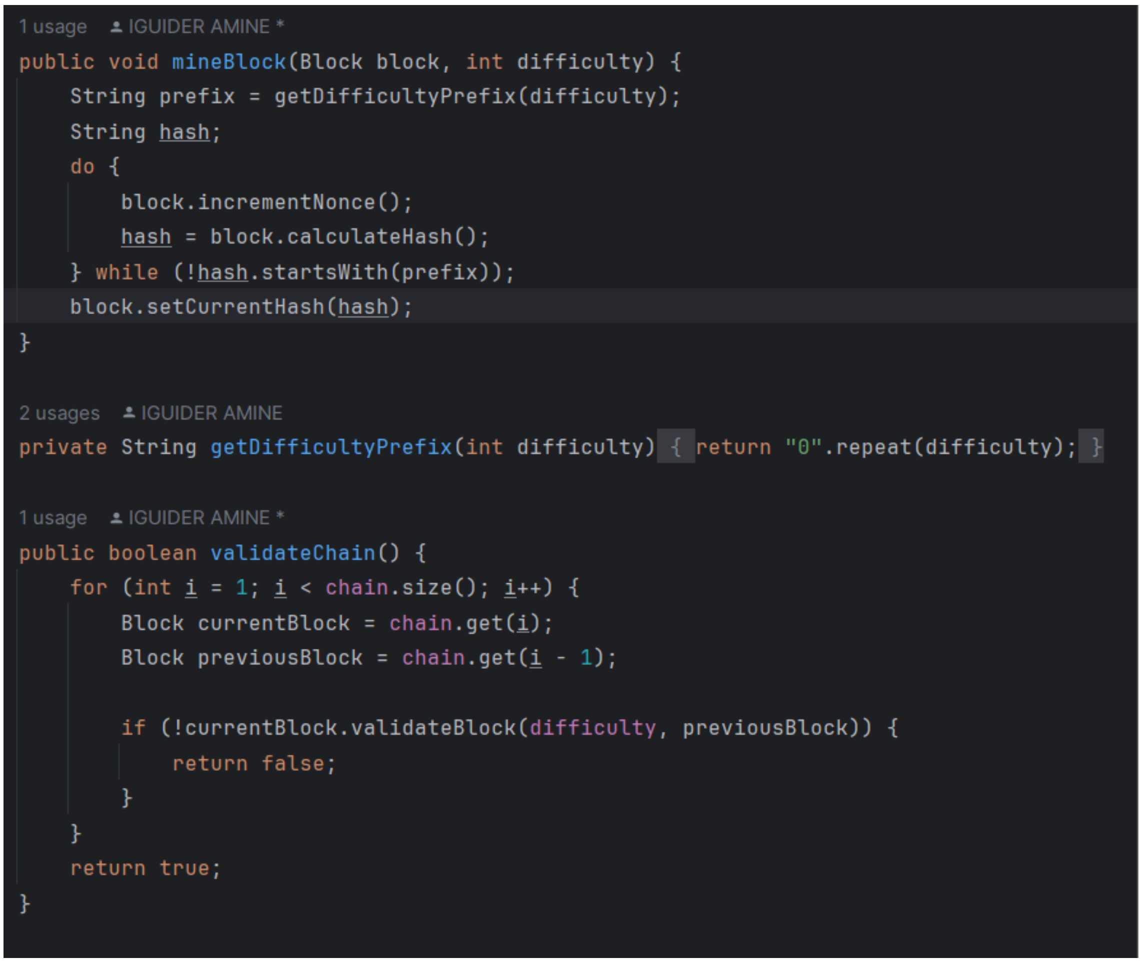
2. Blockchain Class



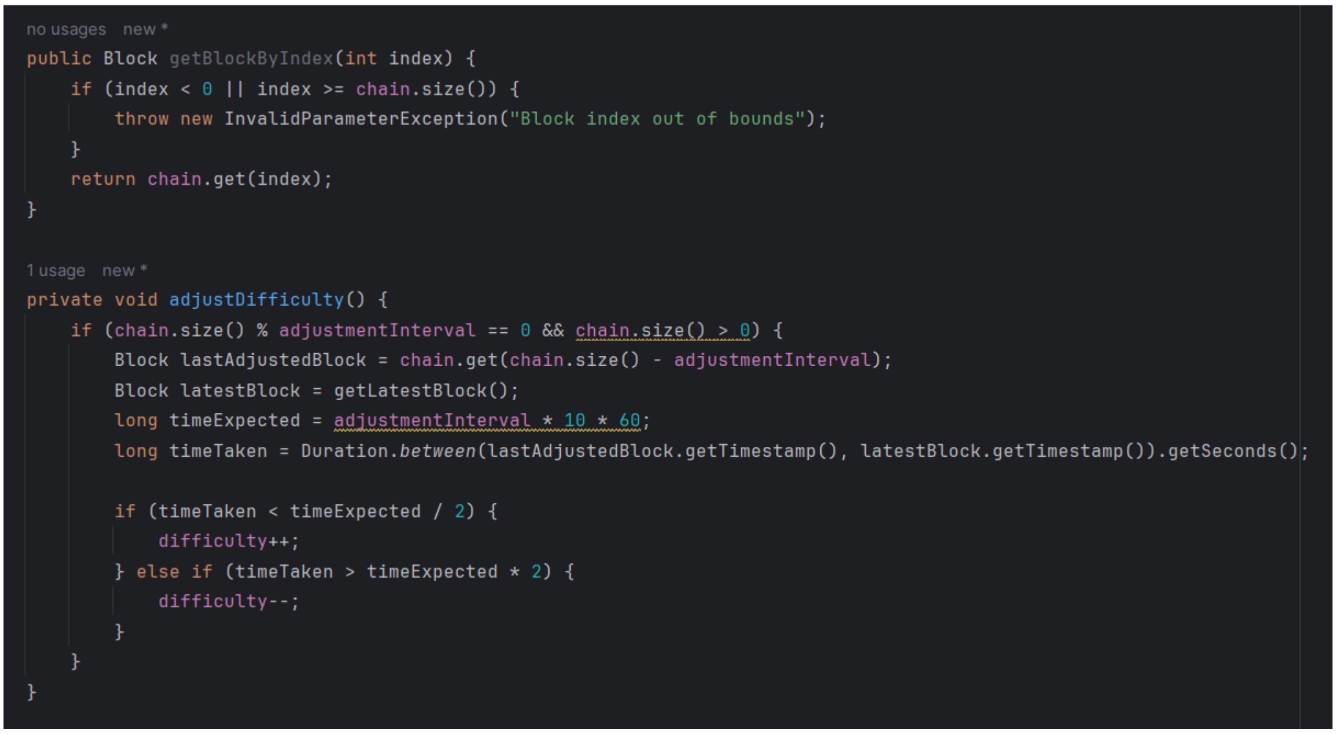
```java
public class Blockchain {
    private List<Block> chain;
    private TransactionPool transactionPool;
    private int difficulty;
    private final int adjustmentInterval;
    13 usages   IGUIDER AMINE *
    public Blockchain(int difficulty, int adjustmentInterval) {
        this.chain = new ArrayList<>();
        this.transactionPool = new TransactionPool();
        this.difficulty = difficulty;
        this.adjustmentInterval = adjustmentInterval;
        Block genesisBlock = createGenesisBlock();
        chain.add(genesisBlock);
    }
    1 usage   IGUIDER AMINE *
    private Block createGenesisBlock() {
        List<Transaction> transactions = new ArrayList<>();
        return new Block( index: 0,  previousHash: "0", transactions,  nonce: 0);
    }
    3 usages   IGUIDER AMINE
    public Block getLatestBlock() { return chain.get(chain.size() - 1); }
    1 usage   IGUIDER AMINE *
    public Block addBlock(Block block) {
        if (isValidBlock(block)) {
            chain.add(block);
            transactionPool.removeTransactions(block.getTransactions());
            adjustDifficulty();
            return block;
        }
        throw new InvalidParameterException("Invalid block");
    }
```



```
1 usage   IGUIDER AMINE *
public boolean isValidBlock(Block block) {
    Block previousBlock = getLatestBlock();

    if (block.getIndex() != previousBlock.getIndex() + 1) {
        return false;
    }

    if (!block.getPreviousHash().equals(previousBlock.getCurrentHash())) {
        return false;
    }

    return block.getCurrentHash().startsWith(getDifficultyPrefix(difficulty));
}

1 usage   IGUIDER AMINE *
public Block mineBlock() {
    Block newBlock = new Block(
            chain.size(),
            getLatestBlock().getCurrentHash(),
            transactionPool.getPendingTransactions(),
            nonce: 0
    );

    mineBlock(newBlock, difficulty);
    return addBlock(newBlock);
}
```

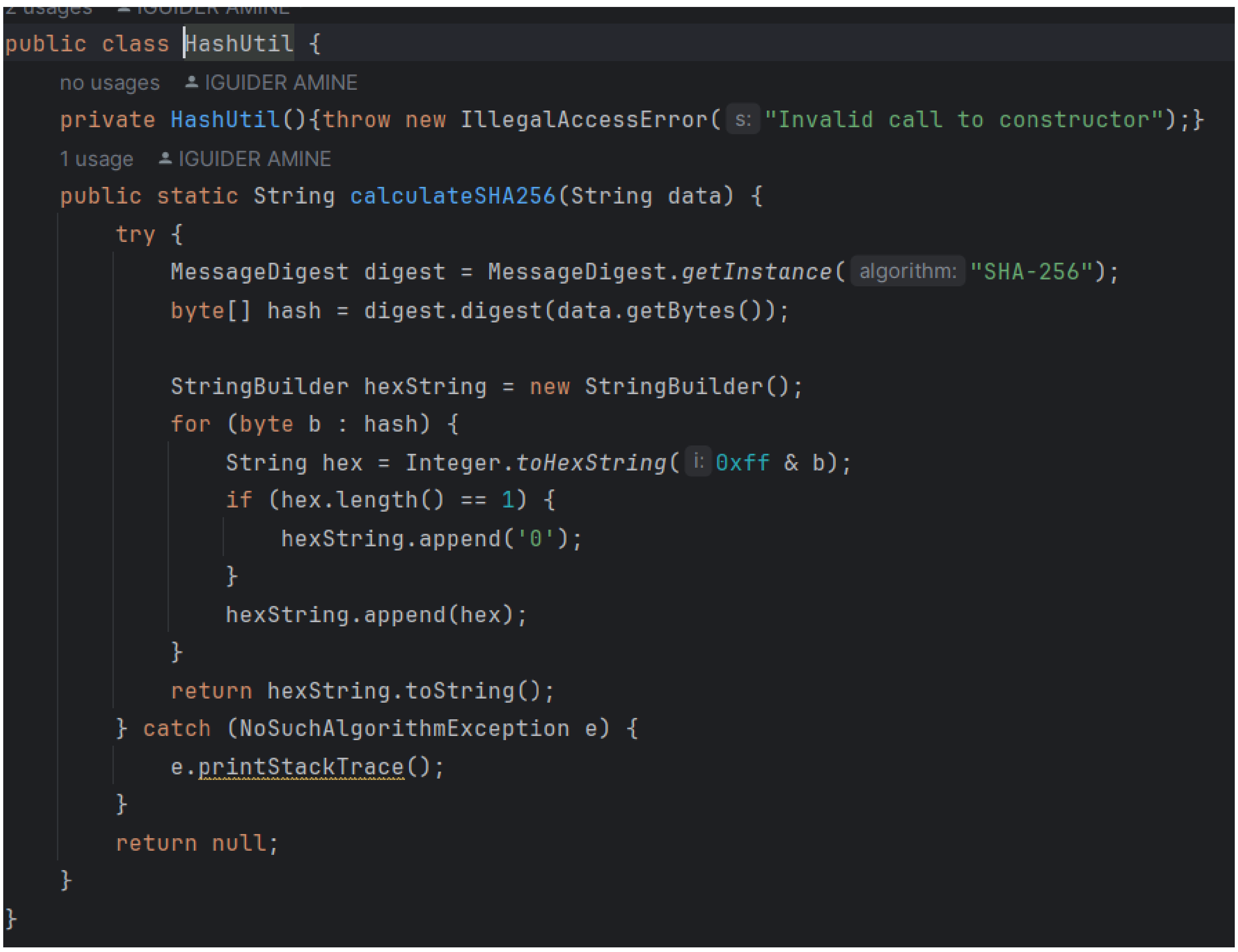


```
1 usage   IGUIDER AMINE *
public void mineBlock(Block block, int difficulty) {
    String prefix = getDifficultyPrefix(difficulty);
    String hash;
    do {
        block.incrementNonce();
        hash = block.calculateHash();
    } while (!hash.startsWith(prefix));
    block.setCurrentHash(hash);
}

2 usages   IGUIDER AMINE
private String getDifficultyPrefix(int difficulty) { return "0".repeat(difficulty); }

1 usage   IGUIDER AMINE *
public boolean validateChain() {
    for (int i = 1; i < chain.size(); i++) {
        Block currentBlock = chain.get(i);
        Block previousBlock = chain.get(i - 1);

        if (!currentBlock.validateBlock(difficulty, previousBlock)) {
            return false;
        }
    }
    return true;
}
```



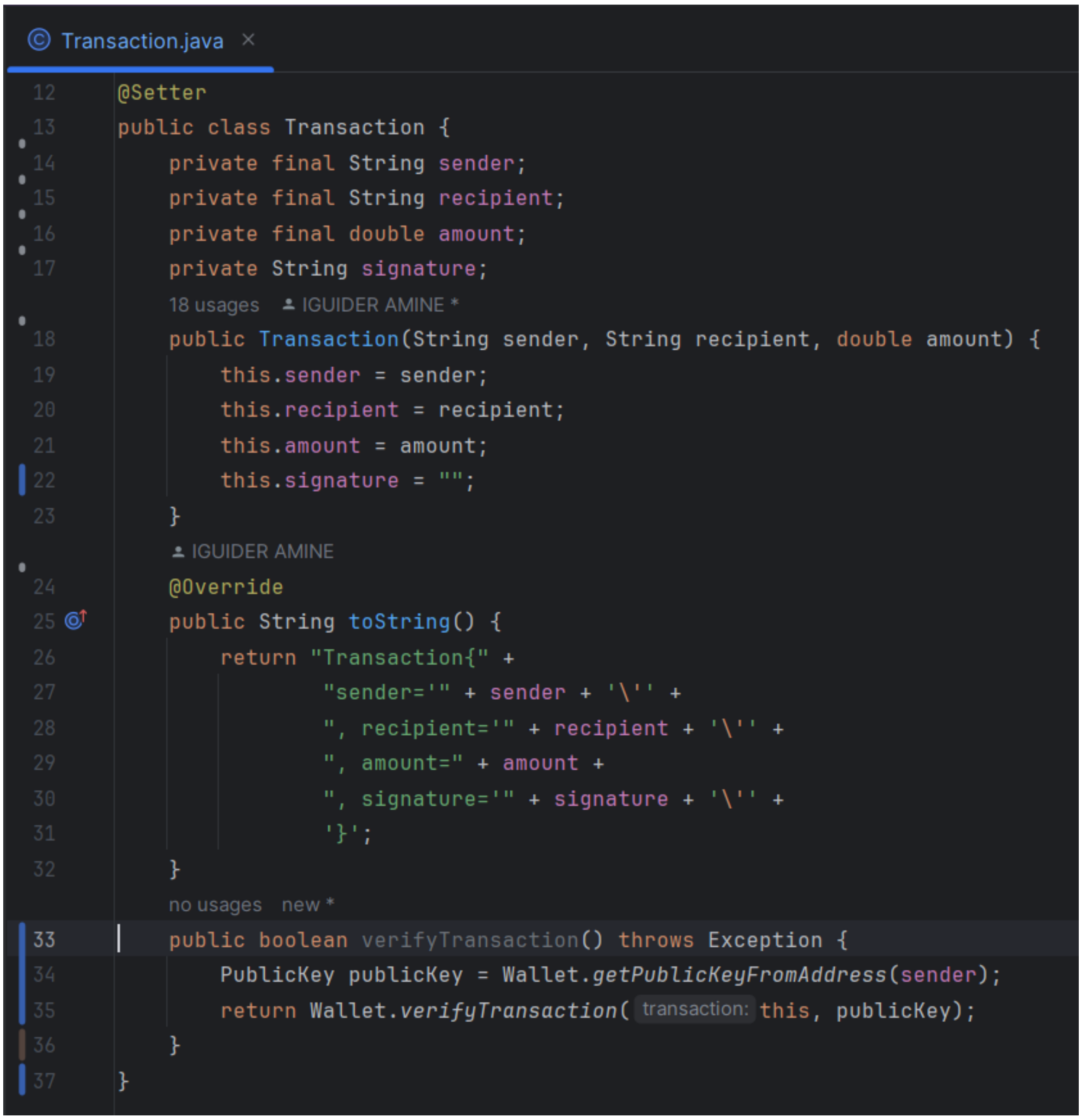
```
no usages   new *
public Block getBlockByIndex(int index) {
    if (index < 0 || index >= chain.size()) {
        throw new InvalidParameterException("Block index out of bounds");
    }
    return chain.get(index);
}

1 usage   new *
private void adjustDifficulty() {
    if (chain.size() % adjustmentInterval == 0 && chain.size() > 0) {
        Block lastAdjustedBlock = chain.get(chain.size() - adjustmentInterval);
        Block latestBlock = getLatestBlock();
        long timeExpected = adjustmentInterval * 10 * 60;
        long timeTaken = Duration.between(lastAdjustedBlock.getTimestamp(), latestBlock.getTimestamp()).getSeconds();

        if (timeTaken < timeExpected / 2) {
            difficulty++;
        } else if (timeTaken > timeExpected * 2) {
            difficulty--;
        }
    }
}
```

3. Hashing Function

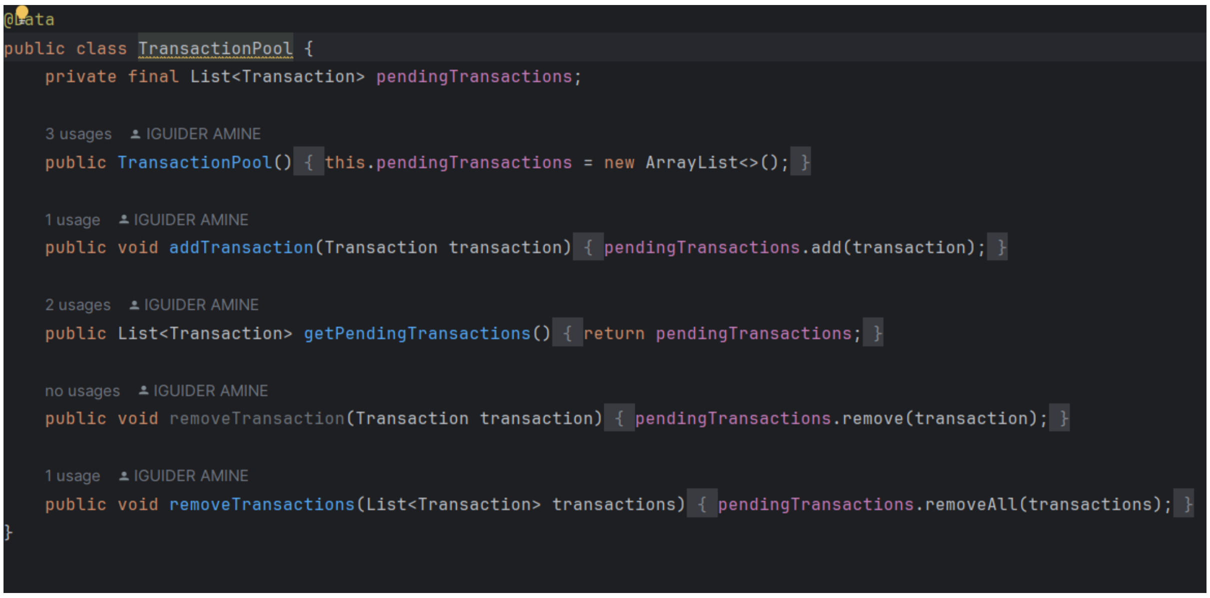


```
2 usages   IGUIDER AMINE
public class HashUtil {
    no usages   IGUIDER AMINE
    private HashUtil(){throw new IllegalAccessError( s: "Invalid call to constructor");}
    1 usage   IGUIDER AMINE
    public static String calculateSHA256(String data) {
        try {
            MessageDigest digest = MessageDigest.getInstance( algorithm: "SHA-256");
            byte[] hash = digest.digest(data.getBytes());

            StringBuilder hexString = new StringBuilder();
            for (byte b : hash) {
                String hex = Integer.toHexString( i: 0xff & b);
                if (hex.length() == 1) {
                    hexString.append('0');
                }
                hexString.append(hex);
            }
            return hexString.toString();
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

4. Transaction Pool


Transaction.java

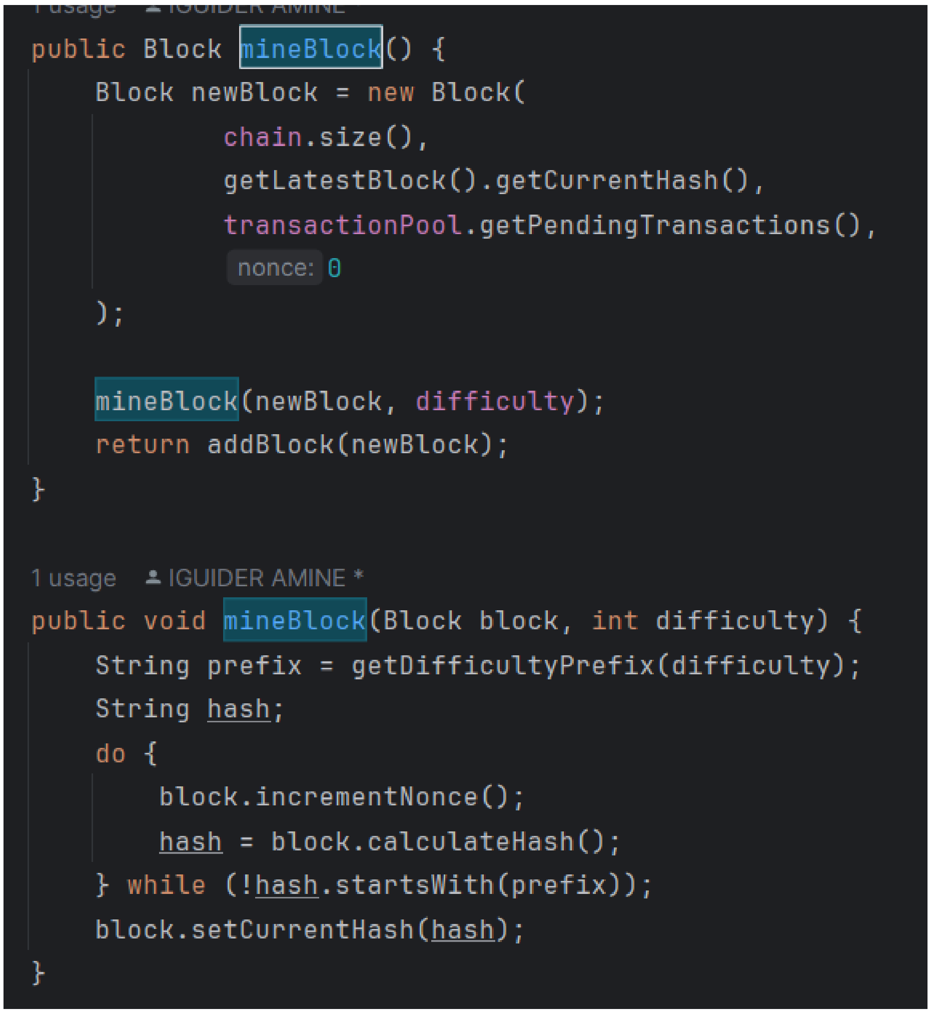
```java
@Setter
public class Transaction {
    private final String sender;
    private final String recipient;
    private final double amount;
    private String signature;
    18 usages  IGUIDER AMINE *
    public Transaction(String sender, String recipient, double amount) {
        this.sender = sender;
        this.recipient = recipient;
        this.amount = amount;
        this.signature = "";
    }
    IGUIDER AMINE
    @Override
    public String toString() {
        return "Transaction{" +
                "sender='" + sender + '\'' +
                ", recipient='" + recipient + '\'' +
                ", amount=" + amount +
                ", signature='" + signature + '\'' +
                '}';
    }
    no usages   new *
    public boolean verifyTransaction() throws Exception {
        PublicKey publicKey = Wallet.getPublicKeyFromAddress(sender);
        return Wallet.verifyTransaction( transaction: this, publicKey);
    }
}
```



```java
@Data
public class TransactionPool {
    private final List<Transaction> pendingTransactions;

    3 usages  IGUIDER AMINE
    public TransactionPool() { this.pendingTransactions = new ArrayList<>(); }

    1 usage  IGUIDER AMINE
    public void addTransaction(Transaction transaction) { pendingTransactions.add(transaction); }

    2 usages  IGUIDER AMINE
    public List<Transaction> getPendingTransactions() { return pendingTransactions; }

    no usages  IGUIDER AMINE
    public void removeTransaction(Transaction transaction) { pendingTransactions.remove(transaction); }

    1 usage  IGUIDER AMINE
    public void removeTransactions(List<Transaction> transactions) { pendingTransactions.removeAll(transactions); }
}
```

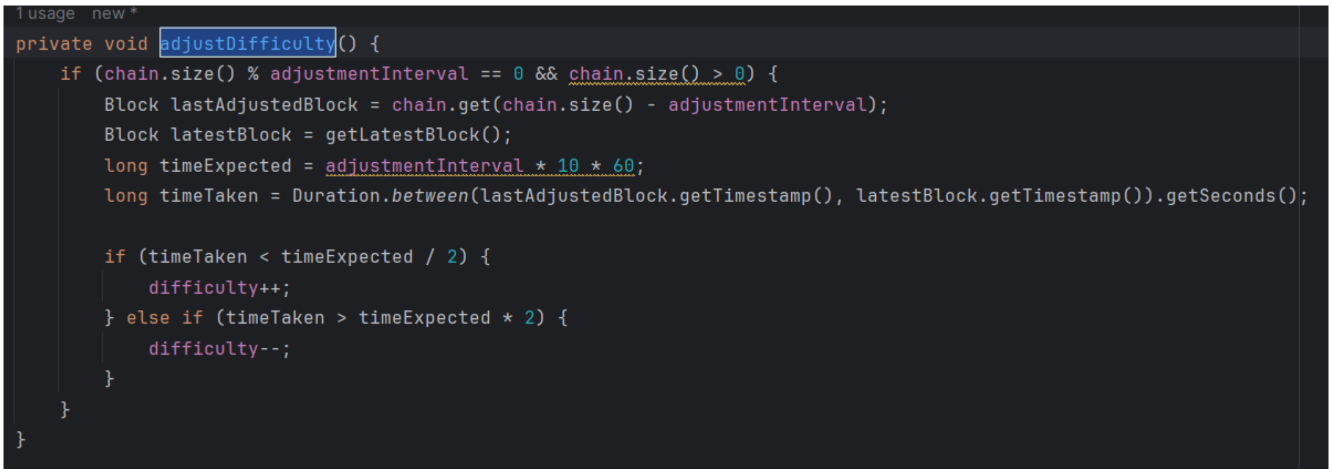
5. Proof of Work Implementation

Method : `mineBlock(Block block, int difficulty)



```
1 usage  IGUIDER AMINE
public Block mineBlock() {
    Block newBlock = new Block(
            chain.size(),
            getLatestBlock().getCurrentHash(),
            transactionPool.getPendingTransactions(),
            nonce: 0
    );

    mineBlock(newBlock, difficulty);
    return addBlock(newBlock);
}

1 usage  IGUIDER AMINE *
public void mineBlock(Block block, int difficulty) {
    String prefix = getDifficultyPrefix(difficulty);
    String hash;
    do {
        block.incrementNonce();
        hash = block.calculateHash();
    } while (!hash.startsWith(prefix));
    block.setCurrentHash(hash);
}
```

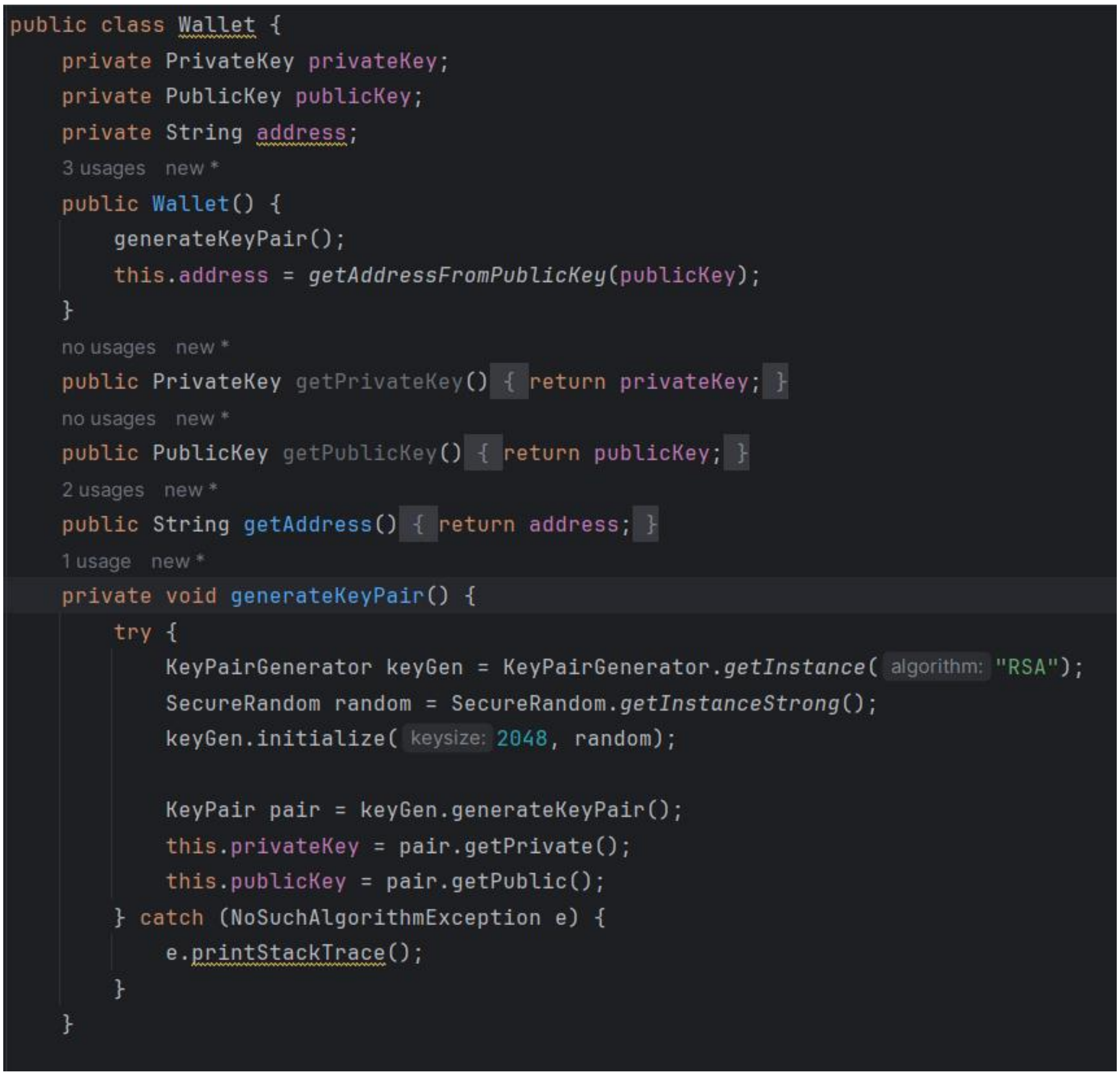
Method : `adjustDifficulty()`



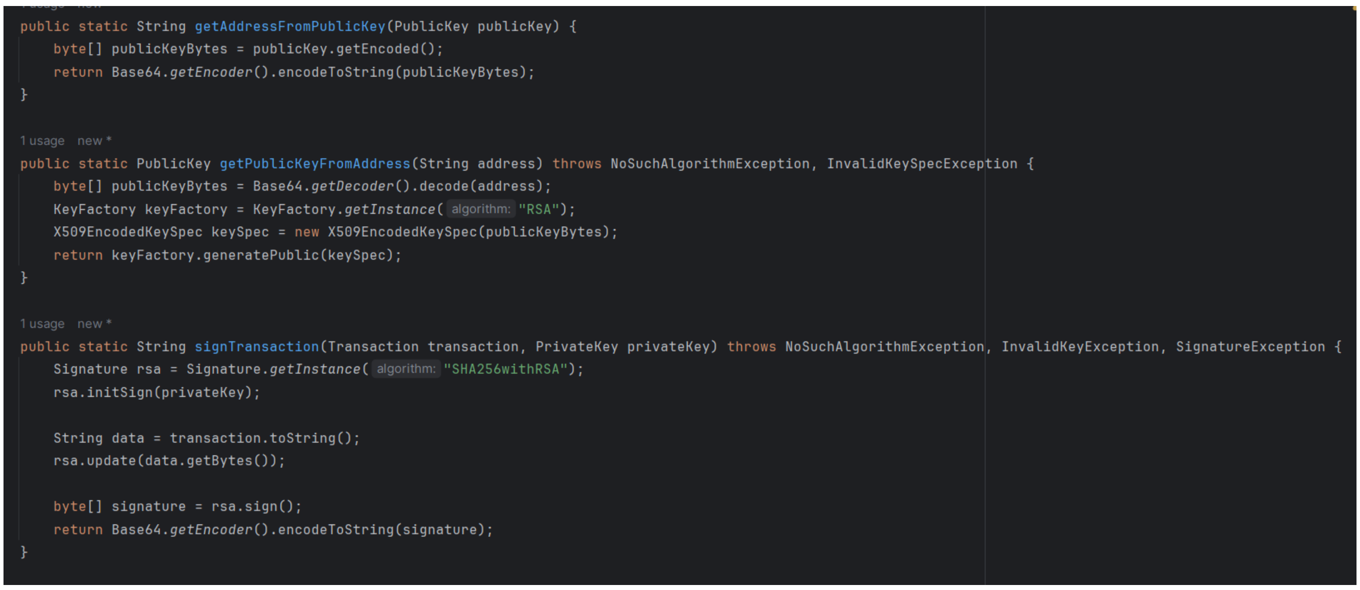
```
1 usage  new *
private void adjustDifficulty() {
    if (chain.size() % adjustmentInterval == 0 && chain.size() > 0) {
        Block lastAdjustedBlock = chain.get(chain.size() - adjustmentInterval);
        Block latestBlock = getLatestBlock();
        long timeExpected = adjustmentInterval * 10 * 60;
        long timeTaken = Duration.between(lastAdjustedBlock.getTimestamp(), latestBlock.getTimestamp()).getSeconds();

        if (timeTaken < timeExpected / 2) {
            difficulty++;
        } else if (timeTaken > timeExpected * 2) {
            difficulty--;
        }
    }
}
```

6. Wallet Management

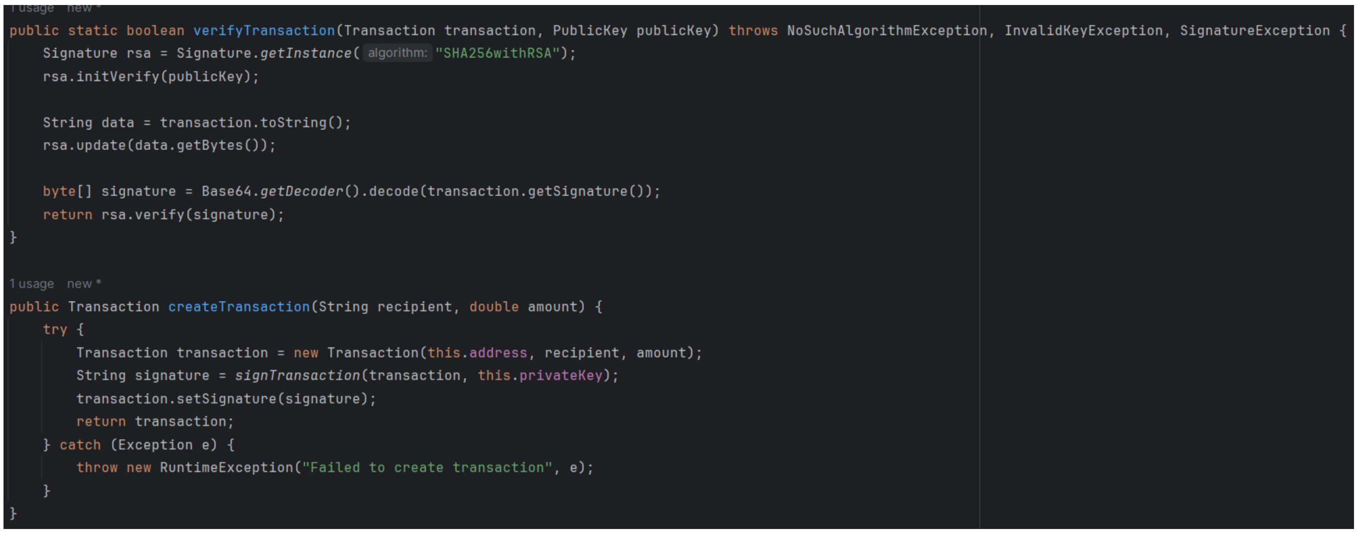
Class Wallet



```
public class Wallet {
    private PrivateKey privateKey;
    private PublicKey publicKey;
    private String address;
    3 usages   new *
    public Wallet() {
        generateKeyPair();
        this.address = getAddressFromPublicKey(publicKey);
    }
    no usages   new *
    public PrivateKey getPrivateKey() { return privateKey; }
    no usages   new *
    public PublicKey getPublicKey() { return publicKey; }
    2 usages   new *
    public String getAddress() { return address; }
    1 usage   new *
    private void generateKeyPair() {
        try {
            KeyPairGenerator keyGen = KeyPairGenerator.getInstance( algorithm: "RSA");
            SecureRandom random = SecureRandom.getInstanceStrong();
            keyGen.initialize( keysize: 2048, random);

            KeyPair pair = keyGen.generateKeyPair();
            this.privateKey = pair.getPrivate();
            this.publicKey = pair.getPublic();
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        }
    }
```



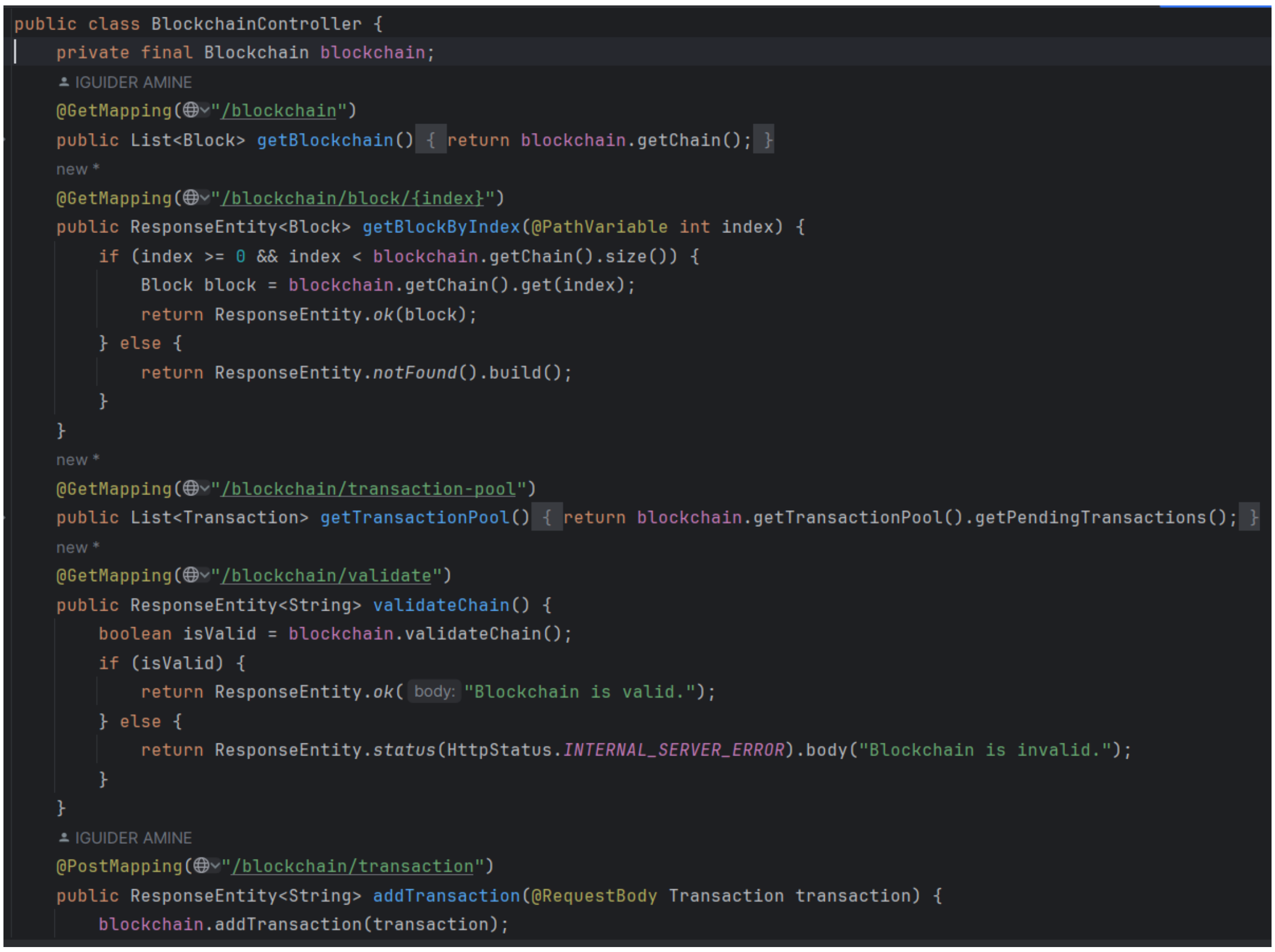
```
public static String getAddressFromPublicKey(PublicKey publicKey) {
    byte[] publicKeyBytes = publicKey.getEncoded();
    return Base64.getEncoder().encodeToString(publicKeyBytes);
}

1 usage  new *
public static PublicKey getPublicKeyFromAddress(String address) throws NoSuchAlgorithmException, InvalidKeySpecException {
    byte[] publicKeyBytes = Base64.getDecoder().decode(address);
    KeyFactory keyFactory = KeyFactory.getInstance( algorithm: "RSA");
    X509EncodedKeySpec keySpec = new X509EncodedKeySpec(publicKeyBytes);
    return keyFactory.generatePublic(keySpec);
}

1 usage  new *
public static String signTransaction(Transaction transaction, PrivateKey privateKey) throws NoSuchAlgorithmException, InvalidKeyException, SignatureException {
    Signature rsa = Signature.getInstance( algorithm: "SHA256withRSA");
    rsa.initSign(privateKey);

    String data = transaction.toString();
    rsa.update(data.getBytes());

    byte[] signature = rsa.sign();
    return Base64.getEncoder().encodeToString(signature);
}
```



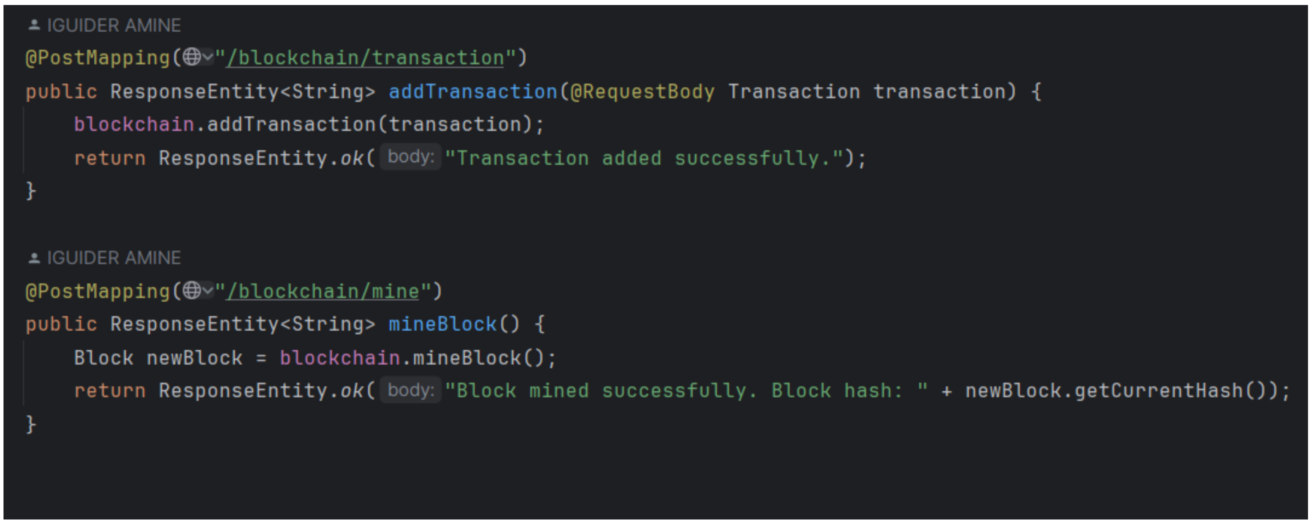
```
public static boolean verifyTransaction(Transaction transaction, PublicKey publicKey) throws NoSuchAlgorithmException, InvalidKeyException, SignatureException {
    Signature rsa = Signature.getInstance( algorithm: "SHA256withRSA");
    rsa.initVerify(publicKey);

    String data = transaction.toString();
    rsa.update(data.getBytes());

    byte[] signature = Base64.getDecoder().decode(transaction.getSignature());
    return rsa.verify(signature);
}

1 usage  new *
public Transaction createTransaction(String recipient, double amount) {
    try {
        Transaction transaction = new Transaction(this.address, recipient, amount);
        String signature = signTransaction(transaction, this.privateKey);
        transaction.setSignature(signature);
        return transaction;
    } catch (Exception e) {
        throw new RuntimeException("Failed to create transaction", e);
    }
}
```

7. Api

BlockchainController

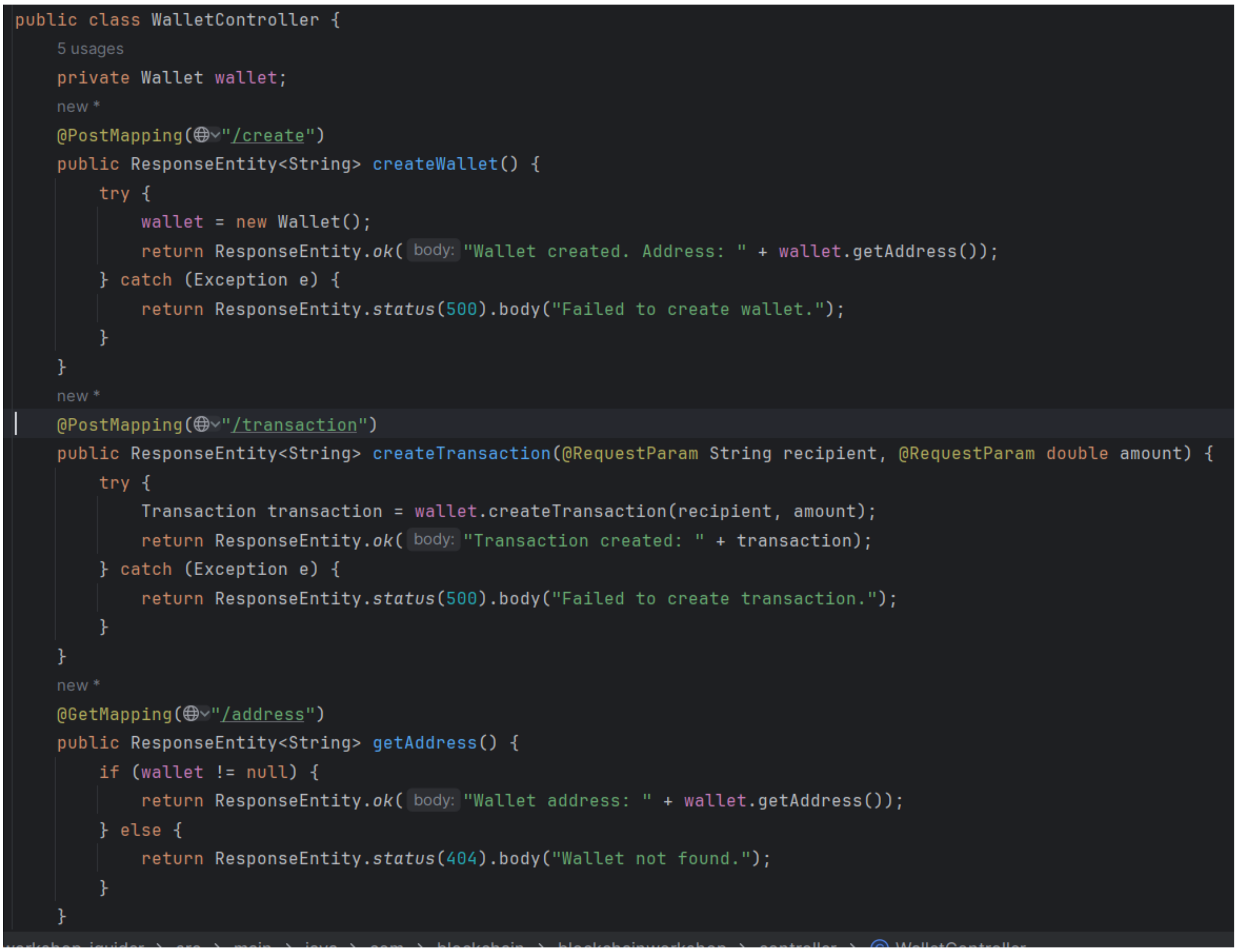


```
public class BlockchainController {
    private final Blockchain blockchain;
    IGUIDER AMINE
    @GetMapping("/blockchain")
    public List<Block> getBlockchain() { return blockchain.getChain(); }
    new *
    @GetMapping("/blockchain/block/{index}")
    public ResponseEntity<Block> getBlockByIndex(@PathVariable int index) {
        if (index >= 0 && index < blockchain.getChain().size()) {
            Block block = blockchain.getChain().get(index);
            return ResponseEntity.ok(block);
        } else {
            return ResponseEntity.notFound().build();
        }
    }
    new *
    @GetMapping("/blockchain/transaction-pool")
    public List<Transaction> getTransactionPool() { return blockchain.getTransactionPool().getPendingTransactions(); }
    new *
    @GetMapping("/blockchain/validate")
    public ResponseEntity<String> validateChain() {
        boolean isValid = blockchain.validateChain();
        if (isValid) {
            return ResponseEntity.ok( body: "Blockchain is valid.");
        } else {
            return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).body("Blockchain is invalid.");
        }
    }
    IGUIDER AMINE
    @PostMapping("/blockchain/transaction")
    public ResponseEntity<String> addTransaction(@RequestBody Transaction transaction) {
        blockchain.addTransaction(transaction);
```



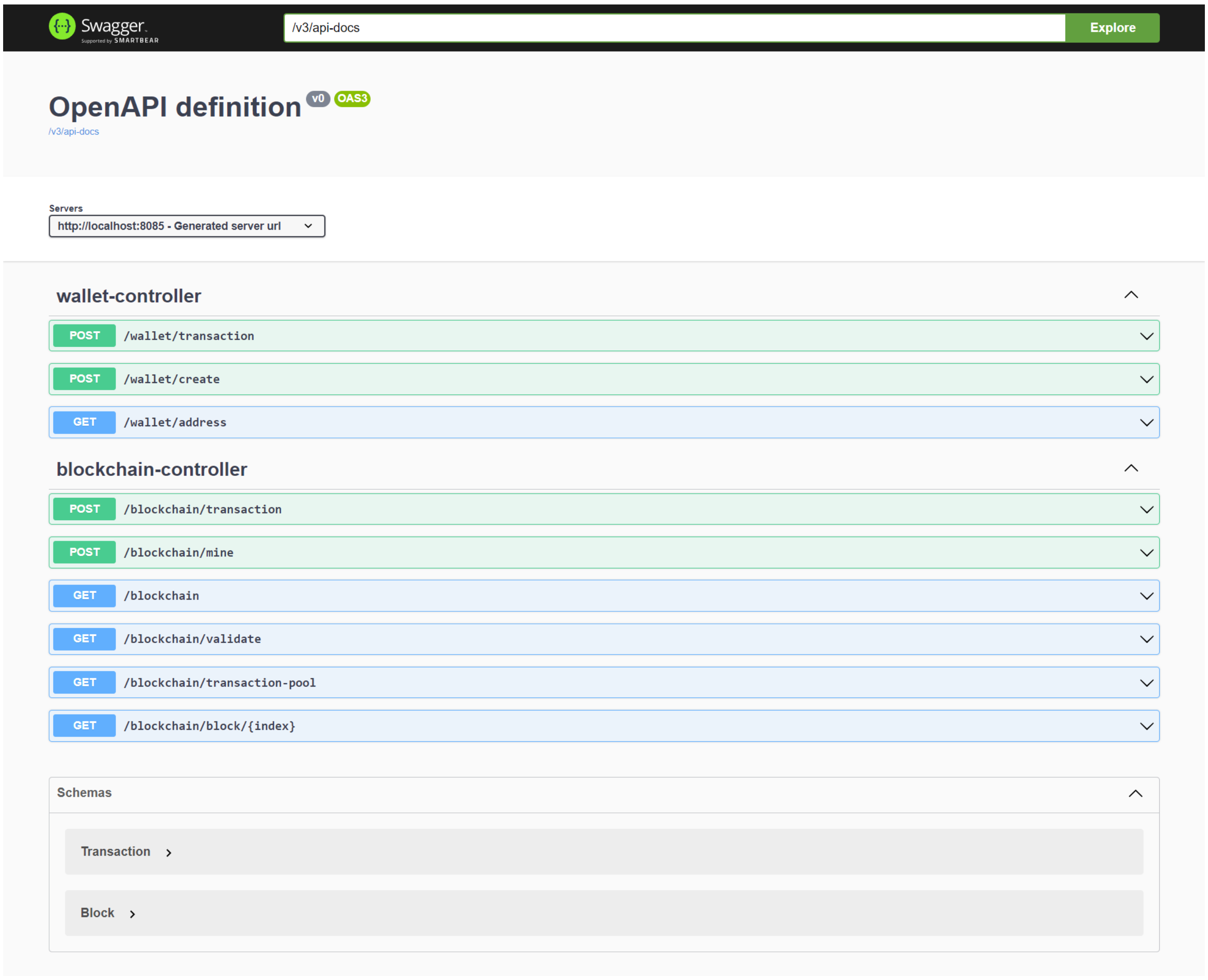
```
IGUIDER AMINE
@PostMapping("/blockchain/transaction")
public ResponseEntity<String> addTransaction(@RequestBody Transaction transaction) {
    blockchain.addTransaction(transaction);
    return ResponseEntity.ok( body: "Transaction added successfully.");
}

IGUIDER AMINE
@PostMapping("/blockchain/mine")
public ResponseEntity<String> mineBlock() {
    Block newBlock = blockchain.mineBlock();
    return ResponseEntity.ok( body: "Block mined successfully. Block hash: " + newBlock.getCurrentHash());
}
```

WalletController



```java
public class WalletController {
    5 usages
    private Wallet wallet;
    new *
    @PostMapping("/create")
    public ResponseEntity<String> createWallet() {
        try {
            wallet = new Wallet();
            return ResponseEntity.ok( body: "Wallet created. Address: " + wallet.getAddress());
        } catch (Exception e) {
            return ResponseEntity.status(500).body("Failed to create wallet.");
        }
    }
    new *
    @PostMapping("/transaction")
    public ResponseEntity<String> createTransaction(@RequestParam String recipient, @RequestParam double amount) {
        try {
            Transaction transaction = wallet.createTransaction(recipient, amount);
            return ResponseEntity.ok( body: "Transaction created: " + transaction);
        } catch (Exception e) {
            return ResponseEntity.status(500).body("Failed to create transaction.");
        }
    }
    new *
    @GetMapping("/address")
    public ResponseEntity<String> getAddress() {
        if (wallet != null) {
            return ResponseEntity.ok( body: "Wallet address: " + wallet.getAddress());
        } else {
            return ResponseEntity.status(404).body("Wallet not found.");
        }
    }
```

8. Test Api with Swagger



/v3/api-docs

OpenAPI definition v0 OAS3

/v3/api-docs

Servers
http://localhost:8085 - Generated server url

wallet-controller

POST `/wallet/transaction`
POST `/wallet/create`
GET `/wallet/address`

blockchain-controller

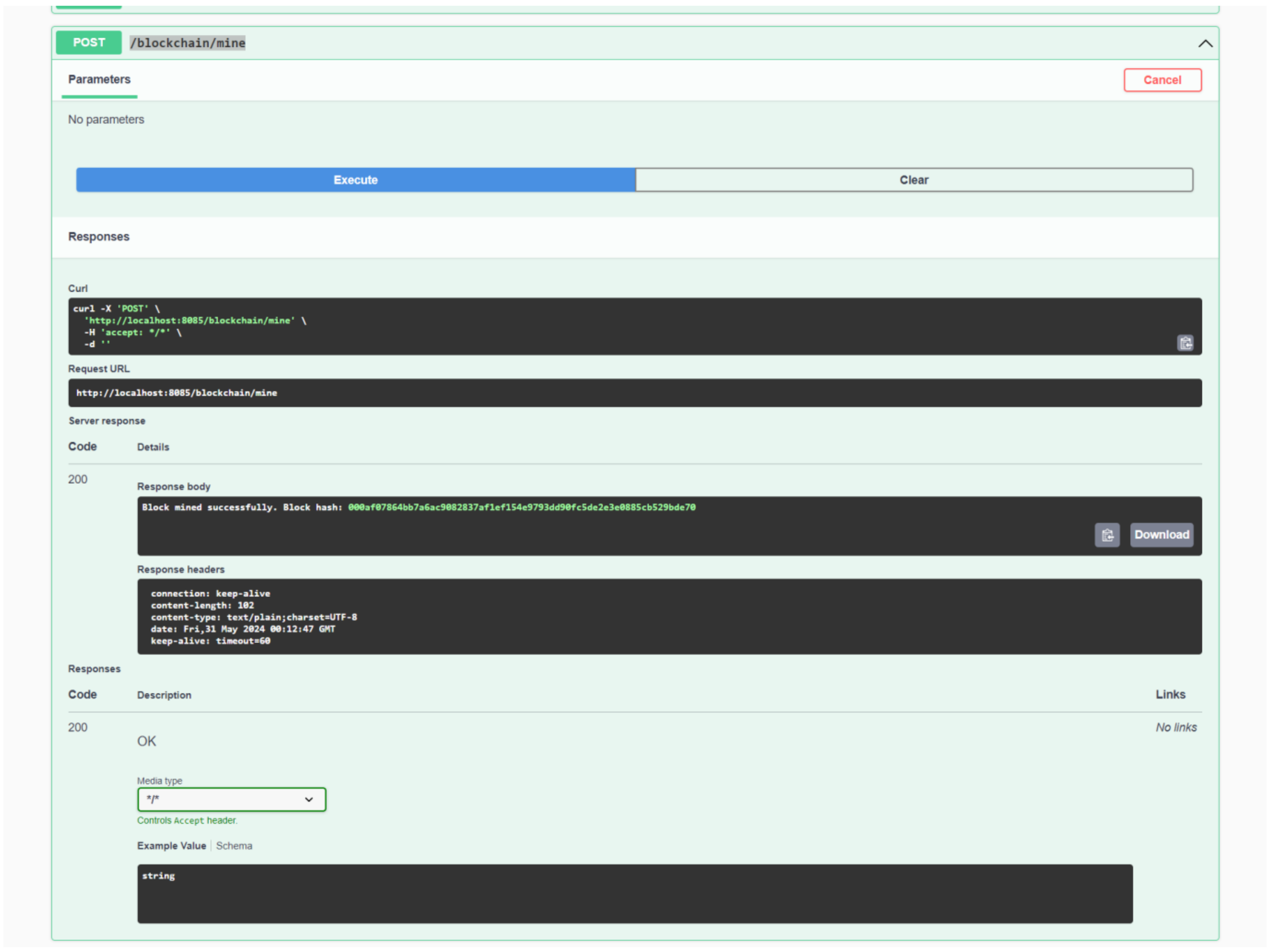
POST `/blockchain/transaction`
POST `/blockchain/mine`
GET `/blockchain`
GET `/blockchain/validate`
GET `/blockchain/transaction-pool`
GET `/blockchain/block/{index}`

Schemas

Transaction

Block

Test api : **/blockchain/mine**



No parameters

Responses

Curl

```
curl -X 'POST' \
  'http://localhost:8085/blockchain/mine' \
  -H 'accept: */*' \
  -d ''
```

Request URL

```
http://localhost:8085/blockchain/mine
```

Server response

Code	Details
200	Response body

```
Block mined successfully. Block hash: 000af07864bb7a6ac9082837af1ef154e9793dd90fc5de2e3e0885cb529bde70
```

Response headers

```
 connection: keep-alive
 content-length: 102
 content-type: text/plain;charset=UTF-8
 date: Fri,31 May 2024 00:12:47 GMT
 keep-alive: timeout=60
```

Responses

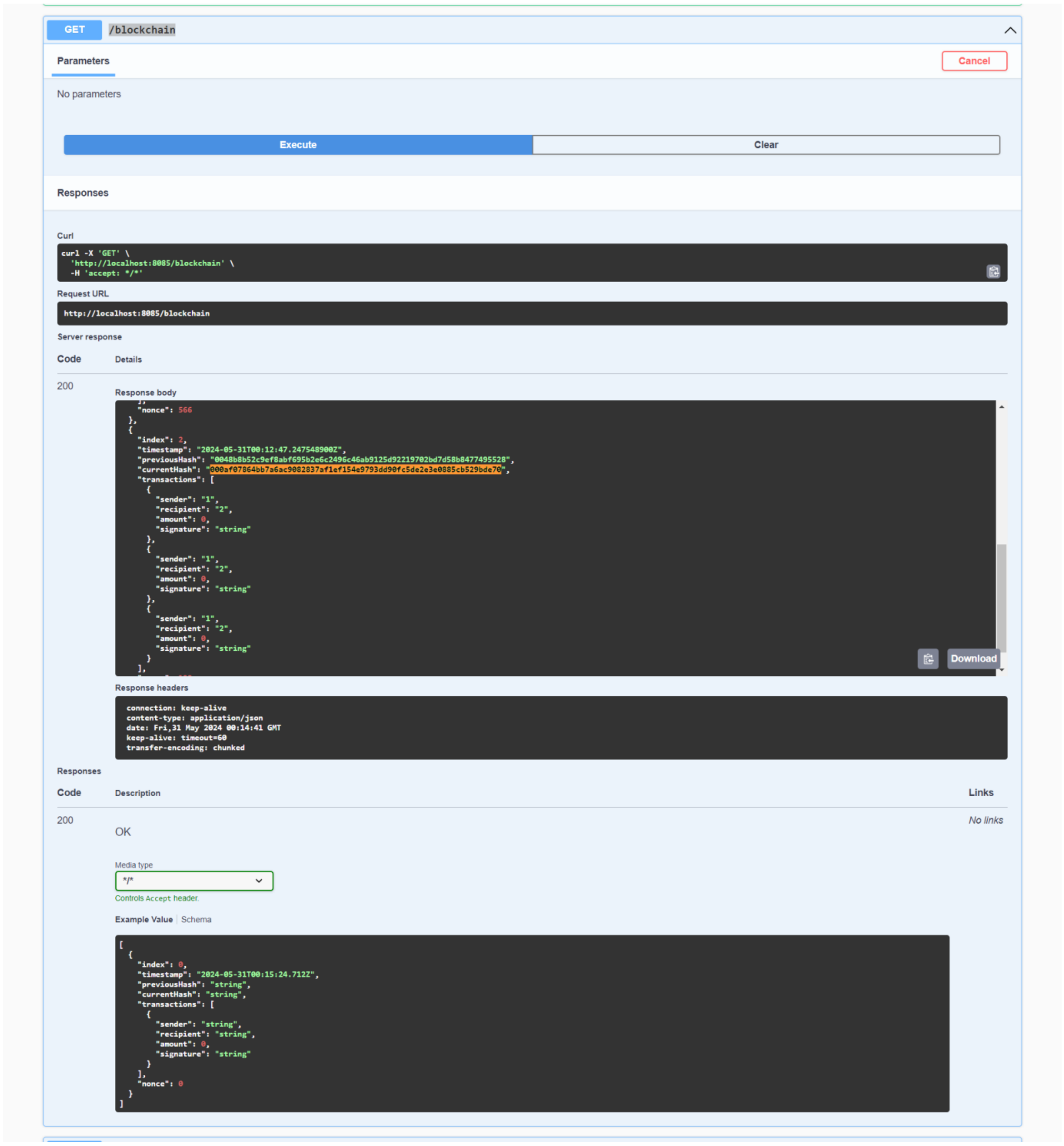
Code	Description	Links
200	OK	*No links*

Media type: */*

Controls `Accept` header.

```
string
```

Test api : **/blockchain**



GET /blockchain

Parameters

Cancel

No parameters

Execute

Clear

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8085/blockchain' \
  -H 'accept: */*'
```

Request URL

```
http://localhost:8085/blockchain
```

Server response

Code Details

200

Response body

```
      "nonce": 566
    },
    {
      "index": 2,
      "timestamp": "2024-05-31T00:12:47.247548900Z",
      "previousHash": "0048b8b52c9ef8abf695b2e6c2496c46ab9125d92219702bd7d58b8477495528",
      "currentHash": "000af07864bb7a6ac9082837af1ef154e9793dd90fc5de2e3e0885cb529bde70",
      "transactions": [
        {
          "sender": "1",
          "recipient": "2",
          "amount": 0,
          "signature": "string"
        },
        {
          "sender": "1",
          "recipient": "2",
          "amount": 0,
          "signature": "string"
        },
        {
          "sender": "1",
          "recipient": "2",
          "amount": 0,
          "signature": "string"
        }
      ],
```

Download

Response headers

```
connection: keep-alive
content-type: application/json
date: Fri,31 May 2024 00:14:41 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```

Responses

Code	Description	Links
200	OK	*No links*

Media type

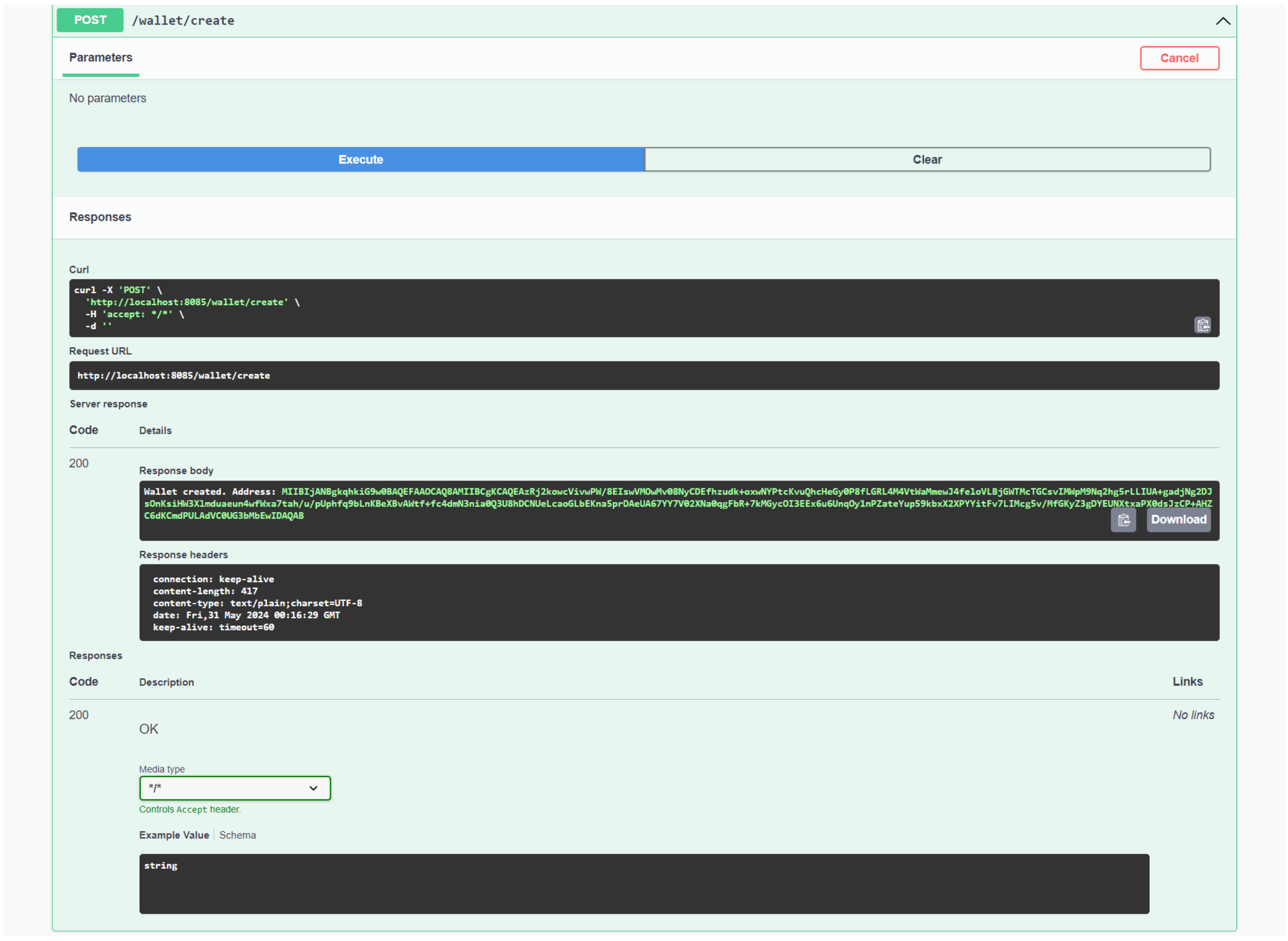
/

Controls Accept header.

Example Value | Schema

```
[
  {
    "index": 0,
    "timestamp": "2024-05-31T00:15:24.712Z",
    "previousHash": "string",
    "currentHash": "string",
    "transactions": [
      {
        "sender": "string",
        "recipient": "string",
        "amount": 0,
        "signature": "string"
      }
    ],
    "nonce": 0
  }
]
```

Test api : **/wallet/create**



No parameters

Responses

Curl

```
curl -X 'POST' \
  'http://localhost:8085/wallet/create' \
  -H 'accept: */*' \
  -d ''
```

Request URL

```
http://localhost:8085/wallet/create
```

Server response

Code	Details
200	Response body

```
Wallet created. Address: MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAzRj2kowcVivwPW/8EIswVMOwMv08NyCDEfhzudk+oxwNYPtcKvuQhcHeGy0P8fLGRL4M4VtWaMmewJ4feloVLBjGWTMcTGCsvIMWpM9Nq2hg5rLLIUA+gadjNg2DJ
sOnKsiHW3Xlmduaeun4wfWxa7tah/u/pUphfq9bLnKBeXBvAWtf+fc4dmN3nia0Q3U8hDCNUeLcaoGLbEKna5prDAeUA67YY7V02XNa0qgFbR+7kMGycOI3EEx6u6UnqOy1nPZateYup59kbxX2XPYYitFv7LIMcg5v/MfGKyZ3gDYEUNXtxaPX0dsJzCP+AHZ
C6dKCmdPULAdVC0UG3bMbEwIDAQAB
```

Response headers

```
connection: keep-alive
content-length: 417
content-type: text/plain;charset=UTF-8
date: Fri,31 May 2024 00:16:29 GMT
keep-alive: timeout=60
```

Responses

Code	Description	Links
200	OK	*No links*

Media type

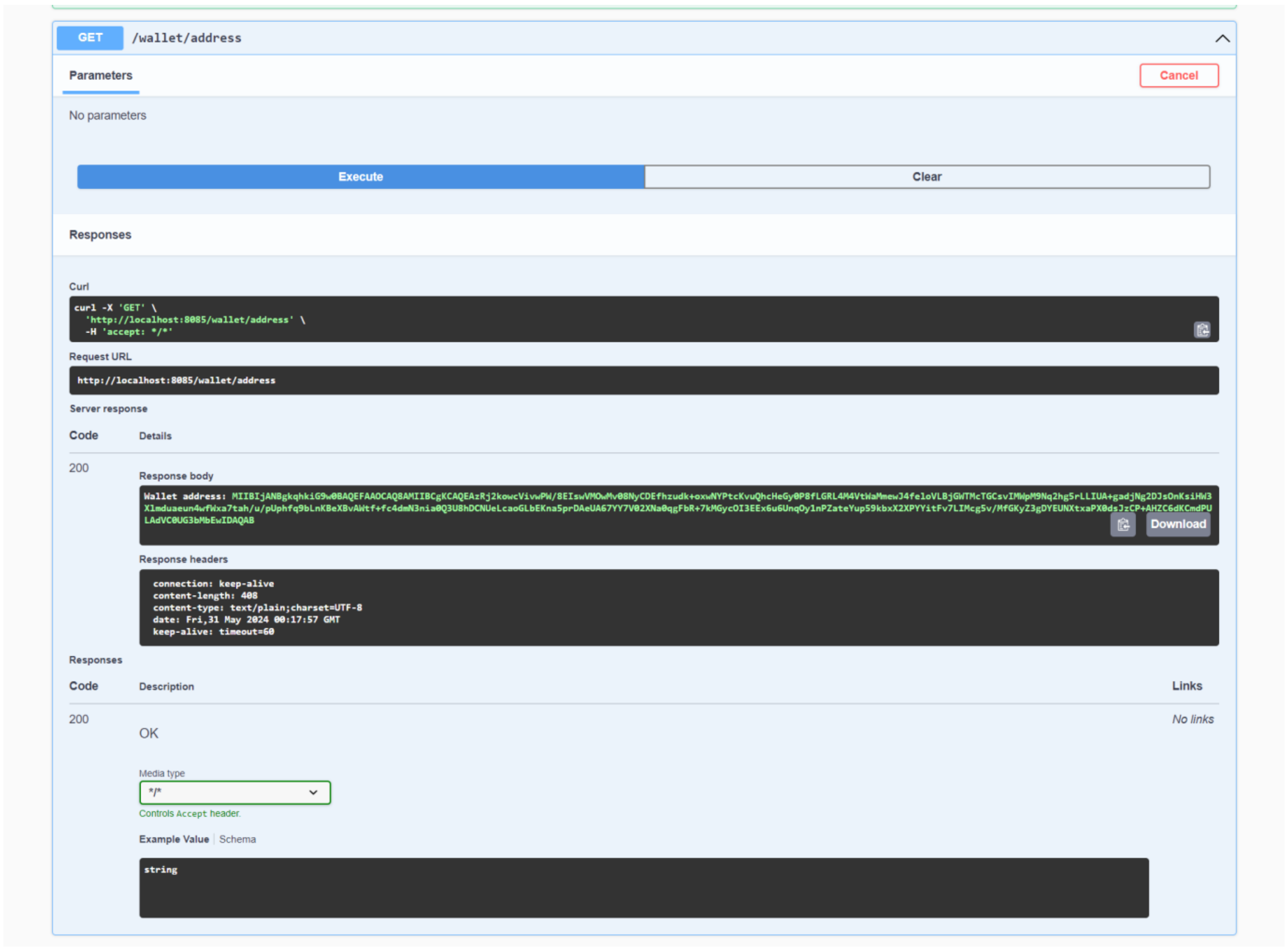
/

Controls `Accept` header.

Example Value | Schema

```
string
```

Test api : **/wallet/address**



GET /wallet/address

Parameters

Cancel

No parameters

Execute Clear

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:8085/wallet/address' \
  -H 'accept: */*'
```

Request URL

```
http://localhost:8085/wallet/address
```

Server response

Code	Details
200	Response body

```
Wallet address: MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAzRj2kowcVivwPW/8EIswVMOwMv08NyCDEfhzudk+oxwNYPtcKvuQhcHeGy0P8fLGRL4M4VtWaMmewJ4feloVLBjGWTMcTGCsvIMWpM9Nq2hg5rLLIUA+gadjNg2DJsOnKsiHW3
Xlmduaeun4wfWxa7tah/u/pUphfq9bLnKBeXBvAWtf+fc4dmN3nia0Q3U8hDCNUeLcaoGLbEKna5prDAeUA67YY7V02XNa0qgFbR+7kMGycOI3EEx6u6UnqOy1nPZateYup59kbxX2XPYYitFv7LIMcg5v/MfGKyZ3gDYEUNXtxaPX0dsJzCP+AHZC6dKCmdPU
LAdVC0UG3bMbEwIDAQAB
```

Download

Response headers

```
connection: keep-alive
content-length: 408
content-type: text/plain;charset=UTF-8
date: Fri,31 May 2024 00:17:57 GMT
keep-alive: timeout=60
```

Responses

Code	Description	Links
200	OK	*No links*

Media type

/

Controls Accept header.

Example Value | Schema

```
string
```