

## Project Development Phase and Model Performance Test

Date	04 November 2023
Team ID	NM2023TMID00389
Project Name	Blockchain Technology For Electronic Health Records

### Model Performance Testing:


Project team shall fill the following information when working for blockchain.

S.No.	Parameter	Values	Screenshot
1.	Information gathering	Setup all the Prerequisite:	The Screenshots of information gathering are all given below.
2.	Extract the zip files	Open to vs code	The Screenshots of extract the zip file are all given below.

3.	Remix Idle Platform exploring	<p>Deploy the smart contract code</p> <p>Deploy and run the transaction.</p> <p>Byselecting the environment - inject the MetaMask.</p>	The Screenshots of extract the zip file are all given below.
4	Open file explorer	<p>Open the extracted file and click onthe folder.</p> <p>Open src, and search for utiles.</p> <p>Open cmd enter commands</p> <p>1.npm install</p> <p>2.npm bootstrap</p> <p>3.npm start</p>	The Screenshots of extract the zip file are all given below.
5	LOCALHOST IPADDRESS	Copy the address and open it to chrome so you can see the front end of your project.	The Screenshots of extract the zip file are all given below.

# 1. INFORMATION GATHERING

Screenshot 1



## Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

⌚ 10 minutes to prepare  
⌚ 1 hour to collaborate  
👤 2-8 people recommended

**Before you collaborate**

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

⌚ 10 minutes

- Team gathering**  
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.
- Set the goal**  
Frame about the problem you'll be focusing on solving in the brainstorming session.
- Learn how to use the facilitation tools**  
Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#)

**Define your problem statement**

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

⌚ 5 minutes

**PROBLEM**

Many users and doctors prefer paper records. They prefer to keep the medical records in a file system. Some medicine shops are not 100% paperless. Most medicine clinics use prioritization to keep records of their medicines. Patients also keep the paper works for their family purposes. So adapting to a total paperless blockchain network is a challenging task.

**Key rules of brainstorming**

To run an smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

Screenshot 2

**Brainstorm**

Write down any ideas that come to mind that address your problem statement.

⌚ 10 minutes

**TIP**  
You can attach a sticky note and fill the panel, which is easy to move and change.

**Group ideas**

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

⌚ 20 minutes

**TIP**  
Add a commentable tag to sticky notes to make it easier to find, remove, organize, and categorize important ideas as they're added your board.

**HARESH KUMAR, P**

Doctors should use small blockchains in order to get habituated to the latest technology.

Publishing or sharing documents online is the most obvious and cheapest option to go paperless.

They should try to use minimum papers. For X-ray plates and other surgery-related documents, they can use paper.

for prescriptions and other file records, the health sector should adopt blockchains as it is easy to store.

**SANJAY KUMAR, S**

Emphasize the positive environmental impact of reducing paper usage and going green to gain public and corporate support.

Share examples of successful paperless transitions in similar industries to inspire others to follow suit.

**LIVINGSTON, I**

Offer training and resources to help individuals and organizations understand the benefits of paperless methods and how to use them effectively.

Develop user-friendly, intuitive software and tools that make the transition to paperless methods easier for people of all tech-skill levels.

Implement robust data security measures to alleviate concerns about privacy and data breaches.

This could include encryption, secure cloud storage, and access controls.

**ASWIN, N**

Develop programs to enhance digital literacy, particularly for older or less tech-savvy individuals.

**ARUL, S**

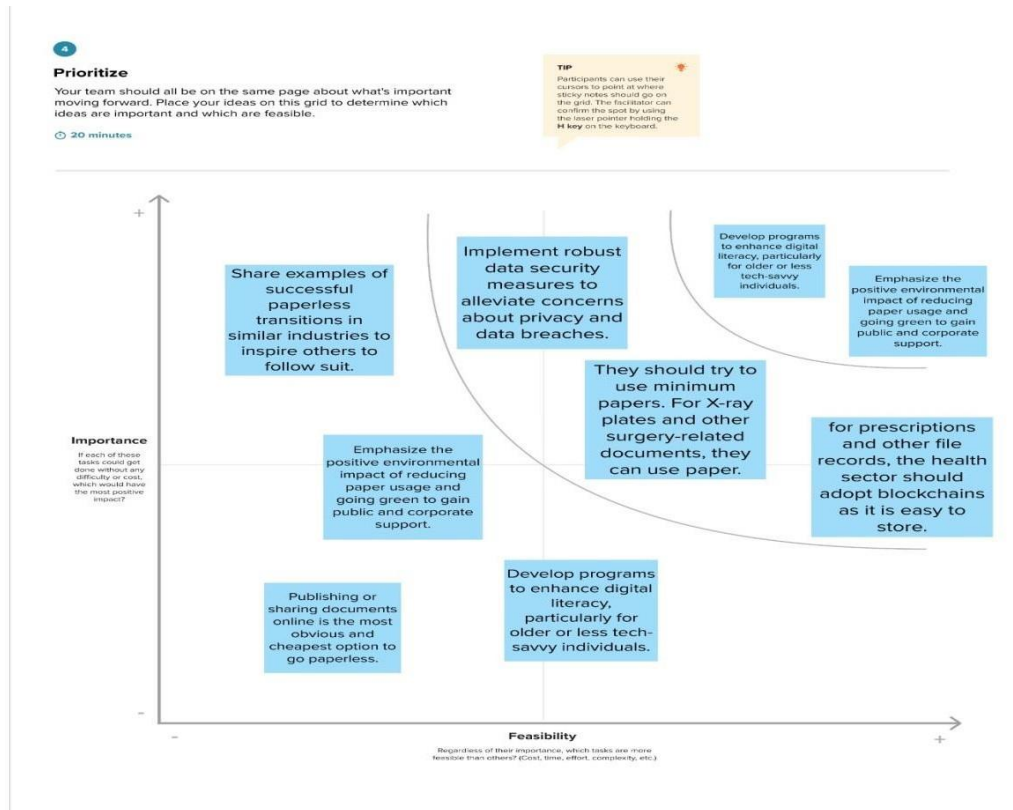
Highlight the cost-saving benefits of going paperless, including reduced paper, ink, and storage costs.

Consider offering incentives or subsidies for adopting paperless methods.

Continually update and improve digital systems to make them more efficient and user-friendly.

Encourage collaboration between organizations, businesses, and government agencies to collectively work towards paperless goals.

## Screenshot 3



## Screenshot 4

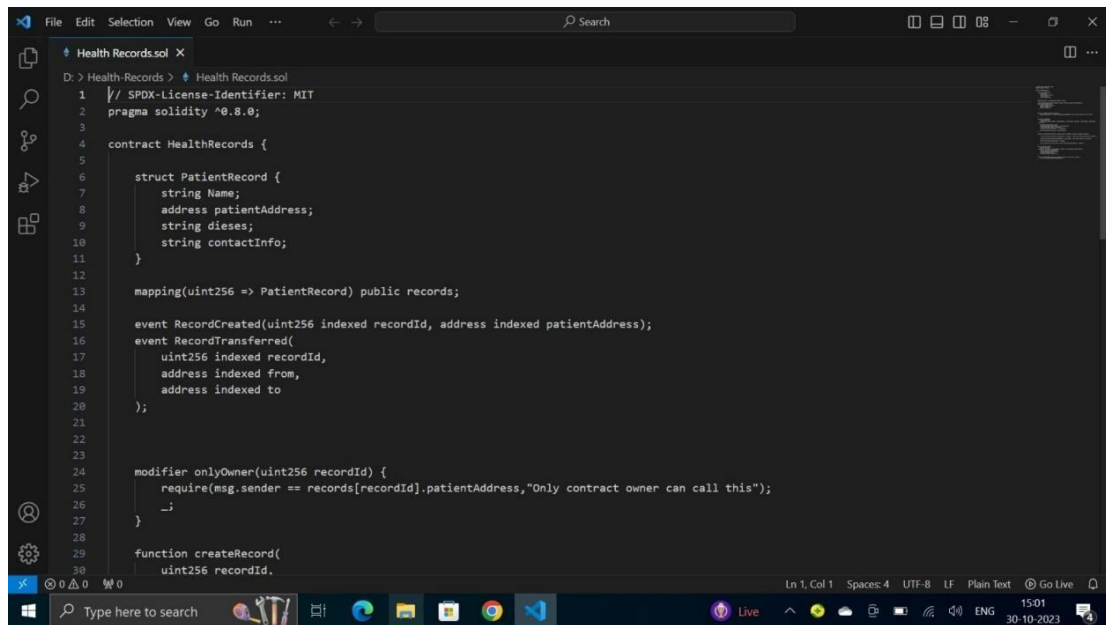
### Empathy map for: Blockchain Technology For Electronics Health Records

This empathy map aims to understand the thoughts, feelings, needs, and actions of users in relation to blockchain technology for electronic health records. It will help in gaining insights into their experiences and identifying opportunities for improvement.



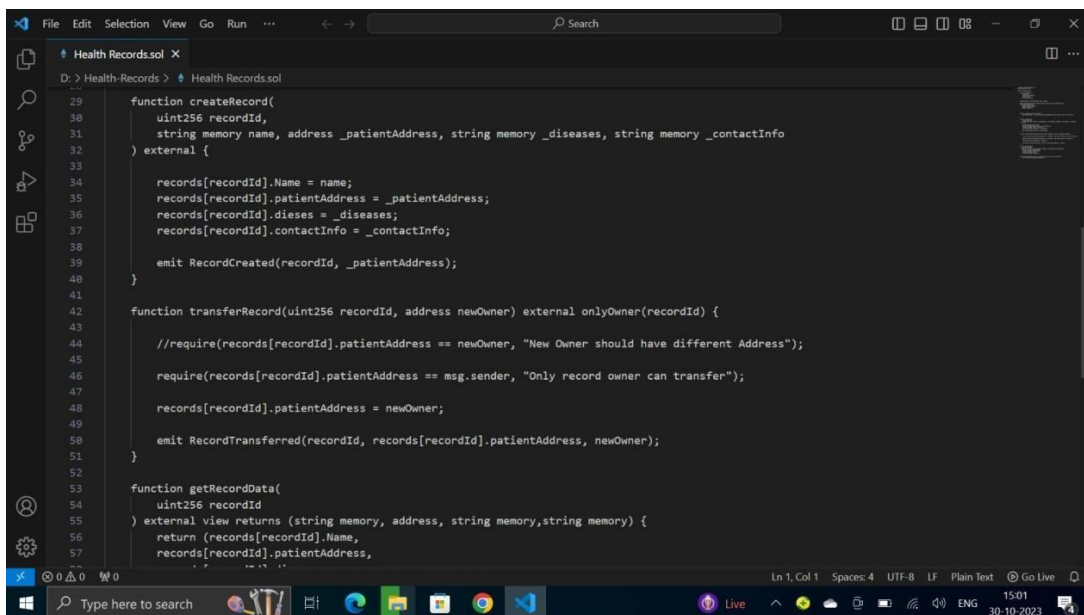
## 2. EXTRACT THE ZIP FILE

Screenshot 1



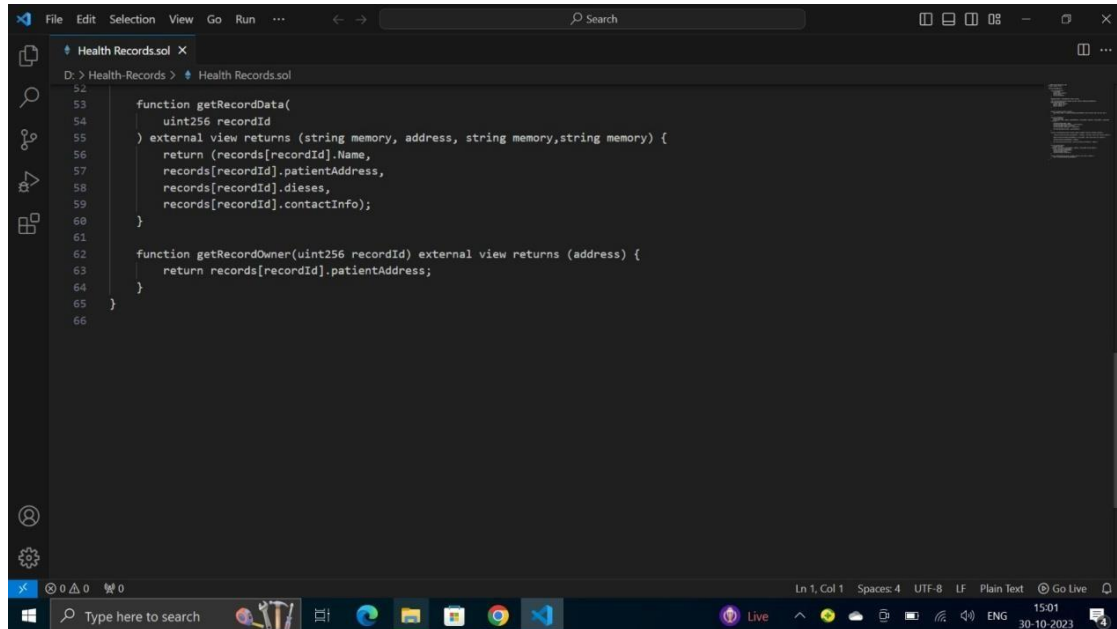
```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 contract HealthRecords {
5
6     struct PatientRecord {
7         string Name;
8         address patientAddress;
9         string diseases;
10        string contactInfo;
11    }
12
13    mapping(uint256 => PatientRecord) public records;
14
15    event RecordCreated(uint256 indexed recordId, address indexed patientAddress);
16    event RecordTransferred(
17        uint256 indexed recordId,
18        address indexed from,
19        address indexed to
20    );
21
22
23
24    modifier onlyOwner(uint256 recordId) {
25        require(msg.sender == records[recordId].patientAddress, "Only contract owner can call this");
26        _;
27    }
28
29    function createRecord(
30        uint256 recordId,
```

Screenshot 2



```
29    function createRecord(
30        uint256 recordId,
31        string memory name, address _patientAddress, string memory _diseases, string memory _contactInfo
32    ) external {
33
34        records[recordId].Name = name;
35        records[recordId].patientAddress = _patientAddress;
36        records[recordId].diseases = _diseases;
37        records[recordId].contactInfo = _contactInfo;
38
39        emit RecordCreated(recordId, _patientAddress);
40    }
41
42    function transferRecord(uint256 recordId, address newOwner) external onlyOwner(recordId) {
43
44        //require(records[recordId].patientAddress == newOwner, "New Owner should have different Address");
45        require(records[recordId].patientAddress == msg.sender, "Only record owner can transfer");
46
47        records[recordId].patientAddress = newOwner;
48
49        emit RecordTransferred(recordId, records[recordId].patientAddress, newOwner);
50    }
51
52
53    function getRecordData(
54        uint256 recordId
55    ) external view returns (string memory, address, string memory, string memory) {
56        return (records[recordId].Name,
57            records[recordId].patientAddress,
```

### Screenshot 3

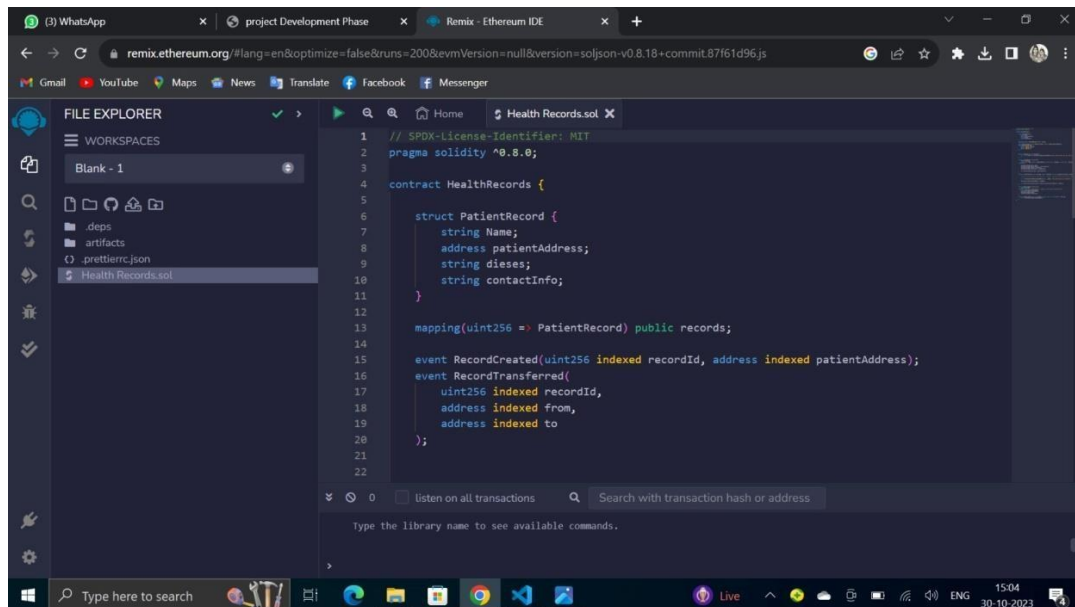


This screenshot shows a Solidity IDE with a file named `HealthRecords.sol` open. The code defines two external view functions. The first function, `getRecordData`, takes a `uint256 recordId` and returns a tuple of `(string memory, address, string memory, string memory)`. It returns the `Name`, `patientAddress`, `dieses`, and `contactInfo` of the record. The second function, `getRecordOwner`, takes a `uint256 recordId` and returns the `patientAddress` of the record.

```
52
53 function getRecordData(
54     uint256 recordId
55 ) external view returns (string memory, address, string memory, string memory) {
56     return (records[recordId].Name,
57           records[recordId].patientAddress,
58           records[recordId].dieses,
59           records[recordId].contactInfo);
60 }
61
62 function getRecordOwner(uint256 recordId) external view returns (address) {
63     return records[recordId].patientAddress;
64 }
65
66
```

## 3. Remix Ide platform Explorting

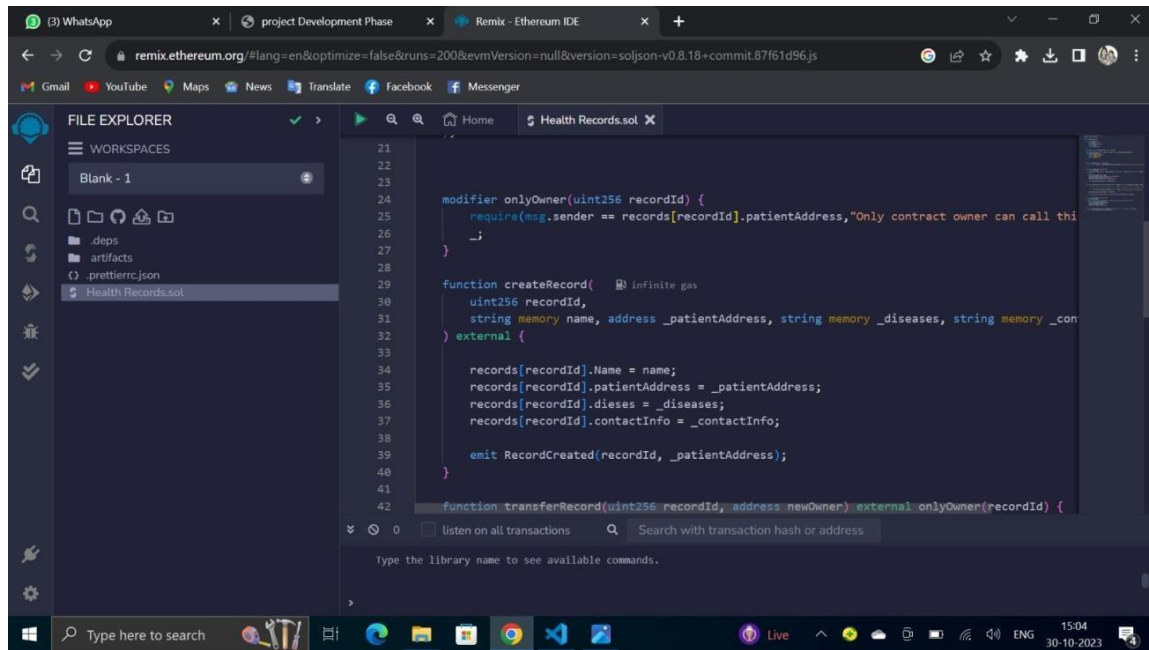
### Screenshot 1



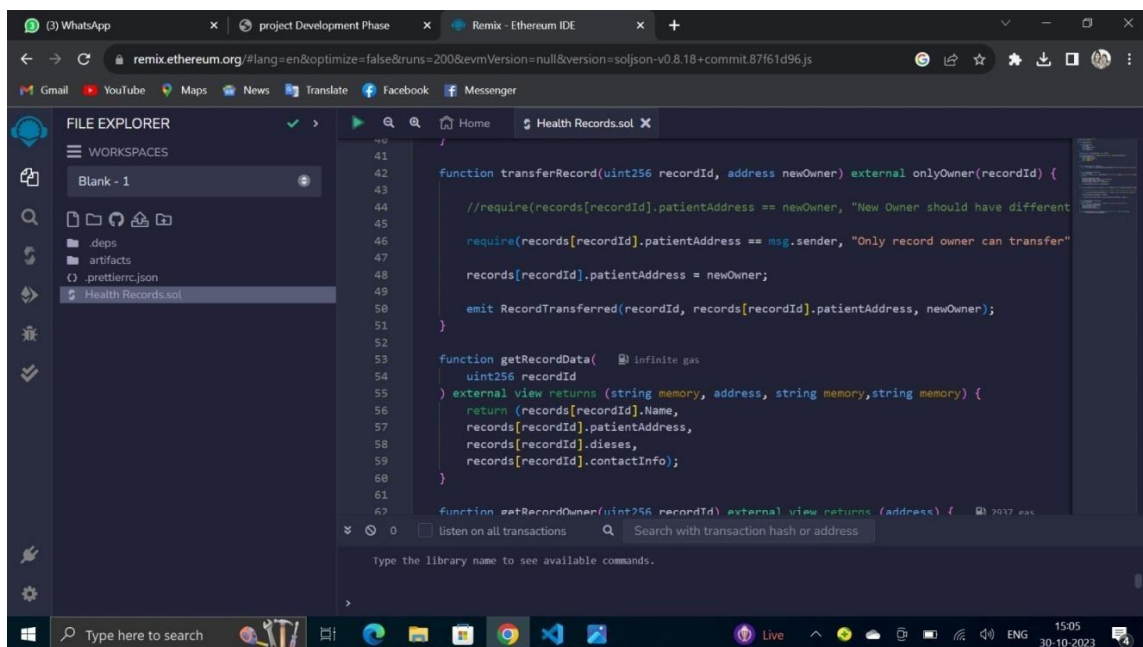
This screenshot shows the Remix IDE platform with the `HealthRecords.sol` file open. The code defines a `HealthRecords` contract with a `PatientRecord` struct. The struct has four fields: `string Name`, `address patientAddress`, `string dieses`, and `string contactInfo`. The contract has a `mapping(uint256 => PatientRecord) public records` and two events: `RecordCreated` and `RecordTransferred`. The `RecordCreated` event has parameters `uint256 indexed recordId` and `address indexed patientAddress`. The `RecordTransferred` event has parameters `uint256 indexed recordId`, `address indexed from`, and `address indexed to`.

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 contract HealthRecords {
5
6     struct PatientRecord {
7         string Name;
8         address patientAddress;
9         string dieses;
10        string contactInfo;
11    }
12
13    mapping(uint256 => PatientRecord) public records;
14
15    event RecordCreated(uint256 indexed recordId, address indexed patientAddress);
16    event RecordTransferred(
17        uint256 indexed recordId,
18        address indexed from,
19        address indexed to
20    );
21
22
```

Screenshot 2

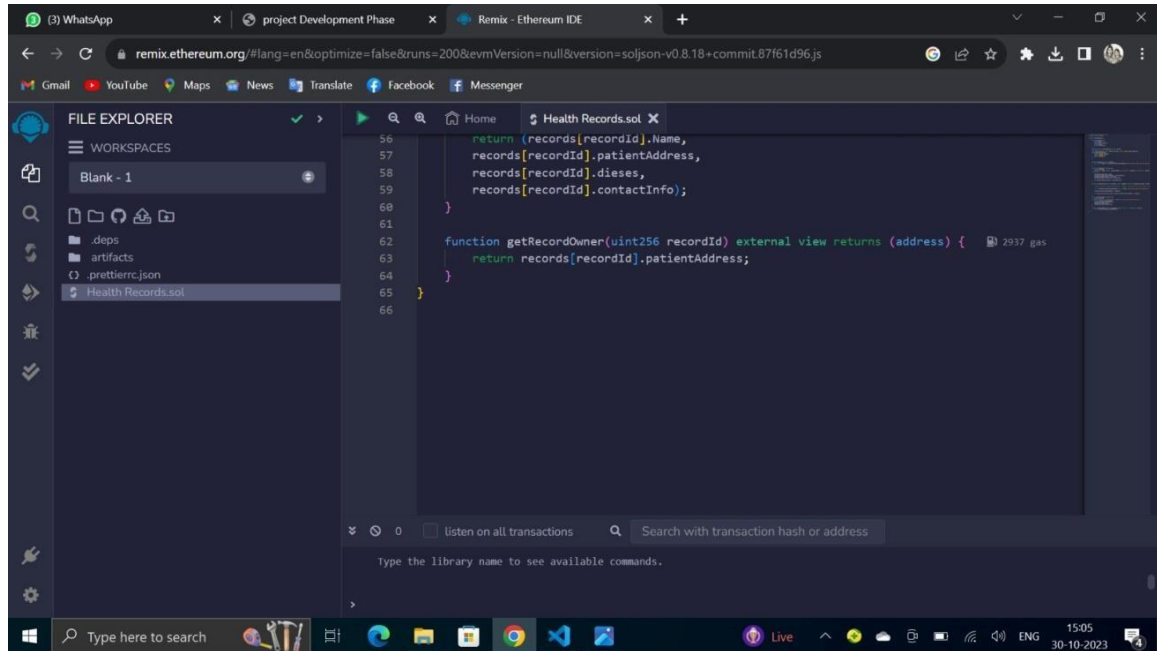


Screenshot 3

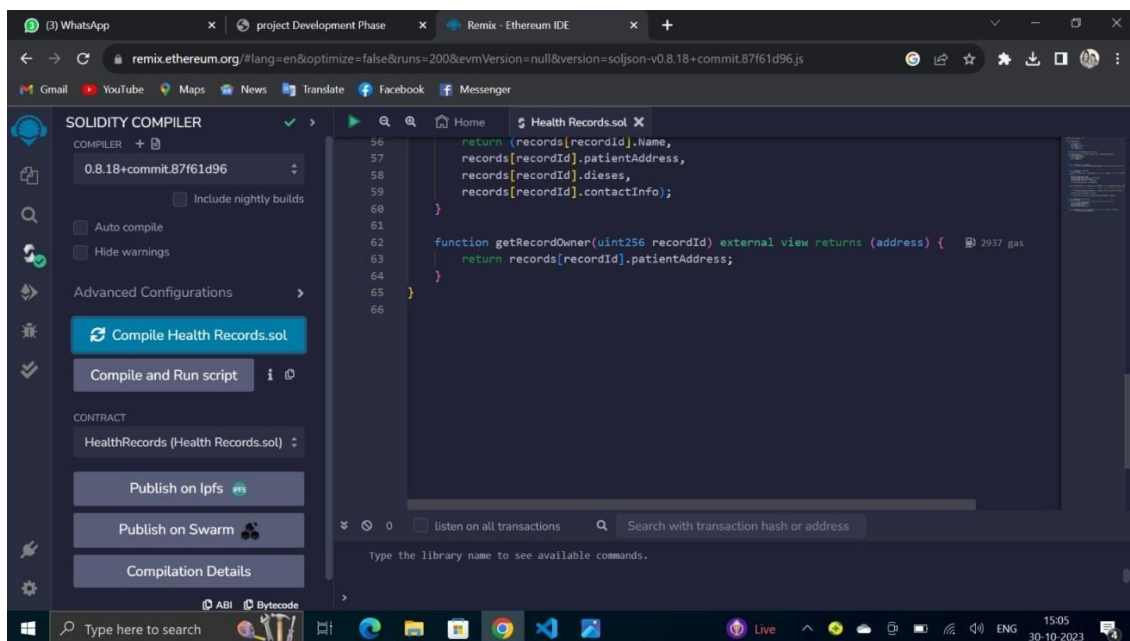




Screenshot 4

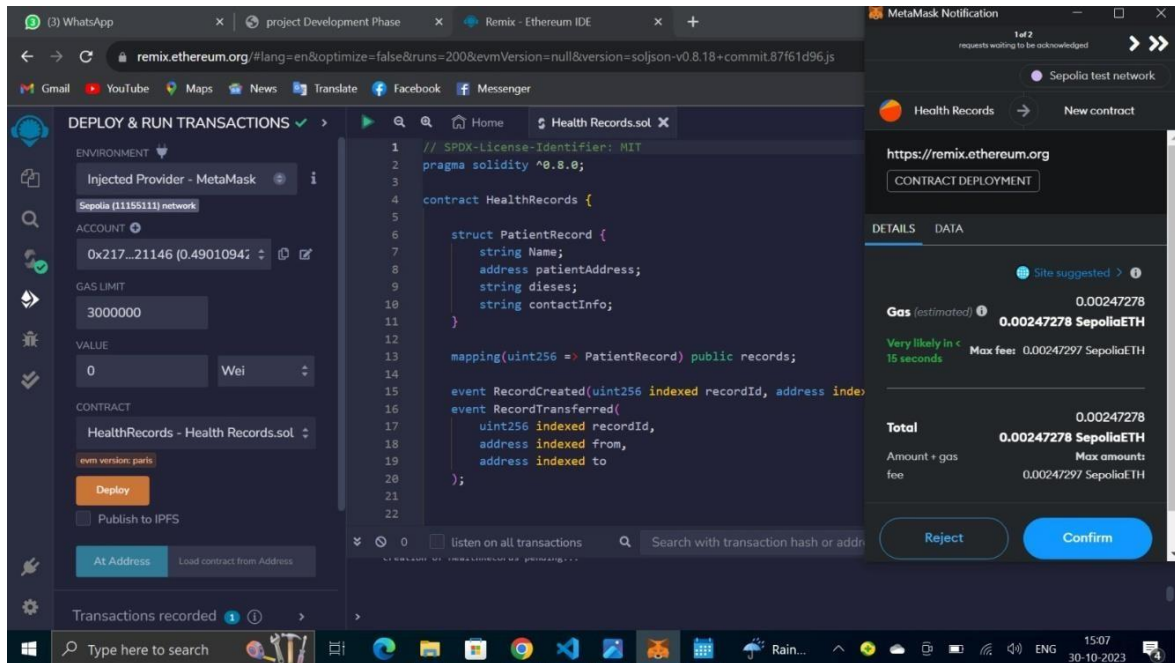


Screenshot 5

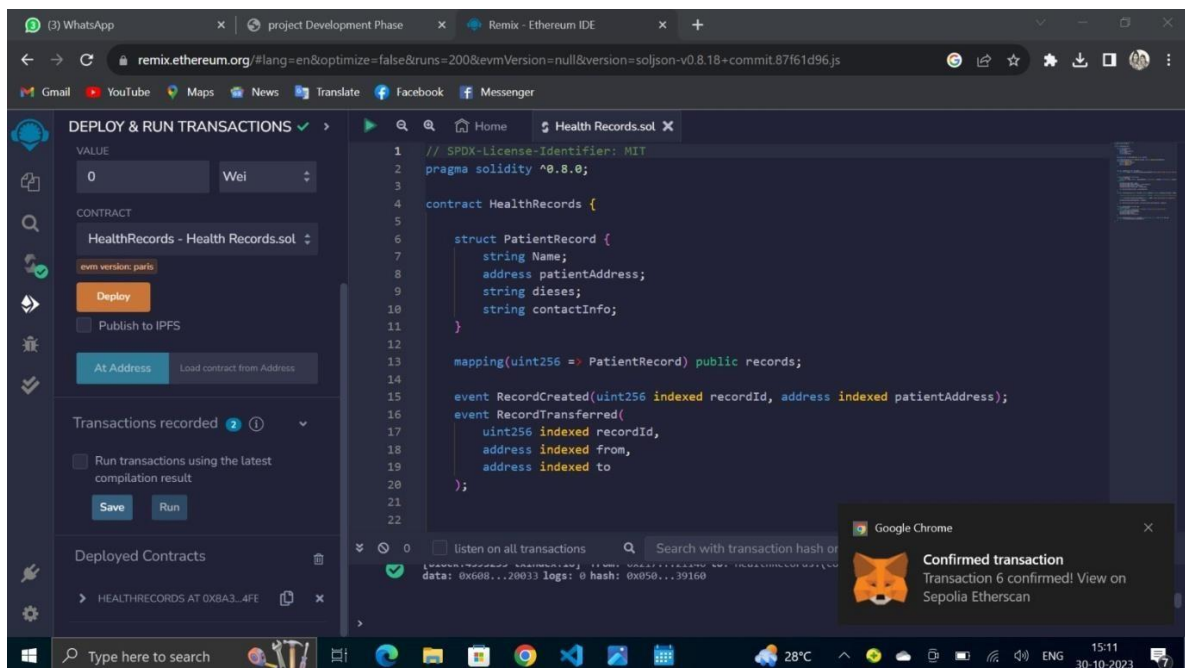




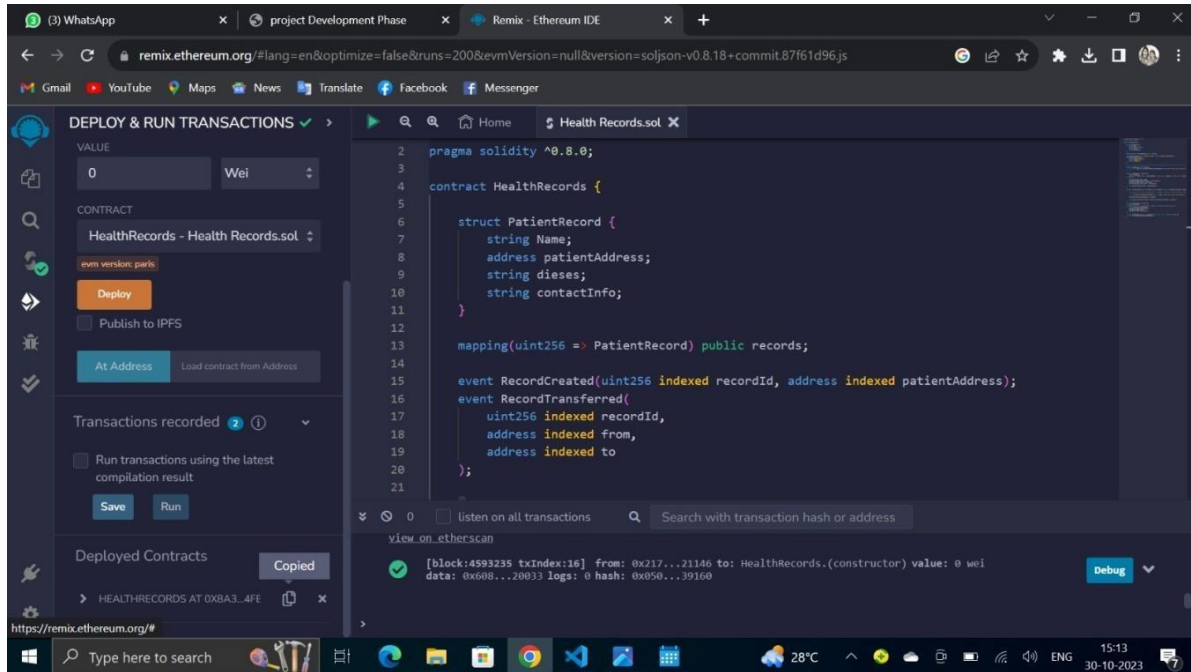
Screenshot 6



Screenshot 7

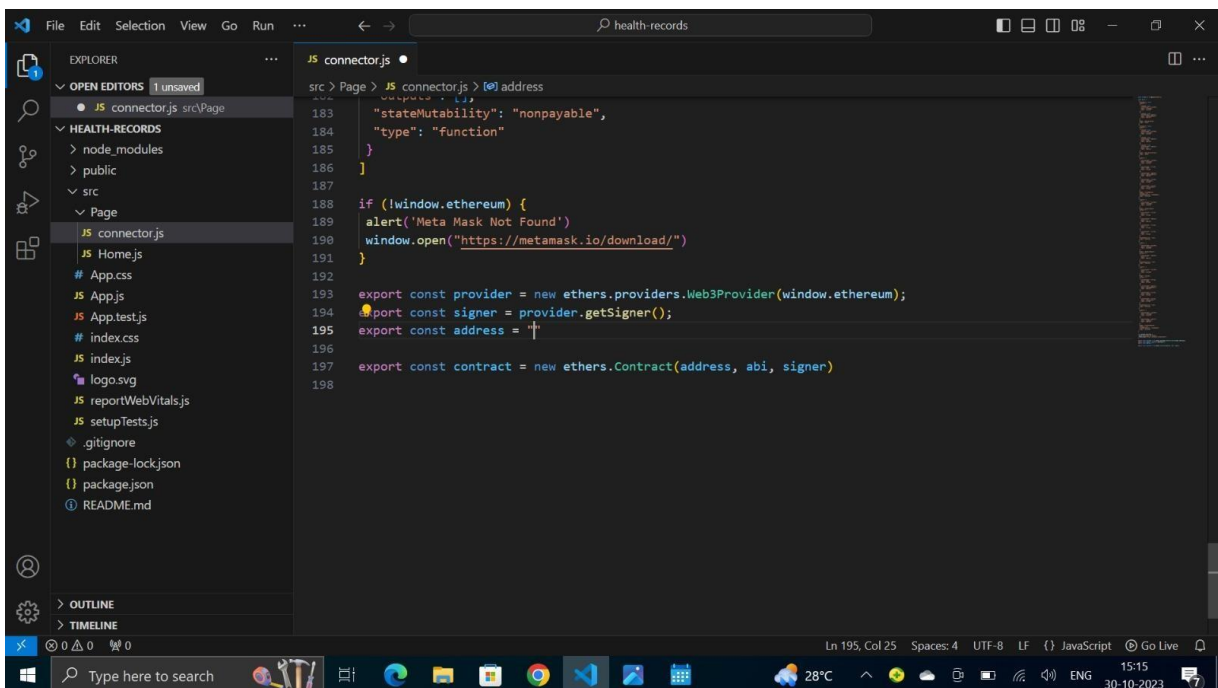


## Screenshot 8

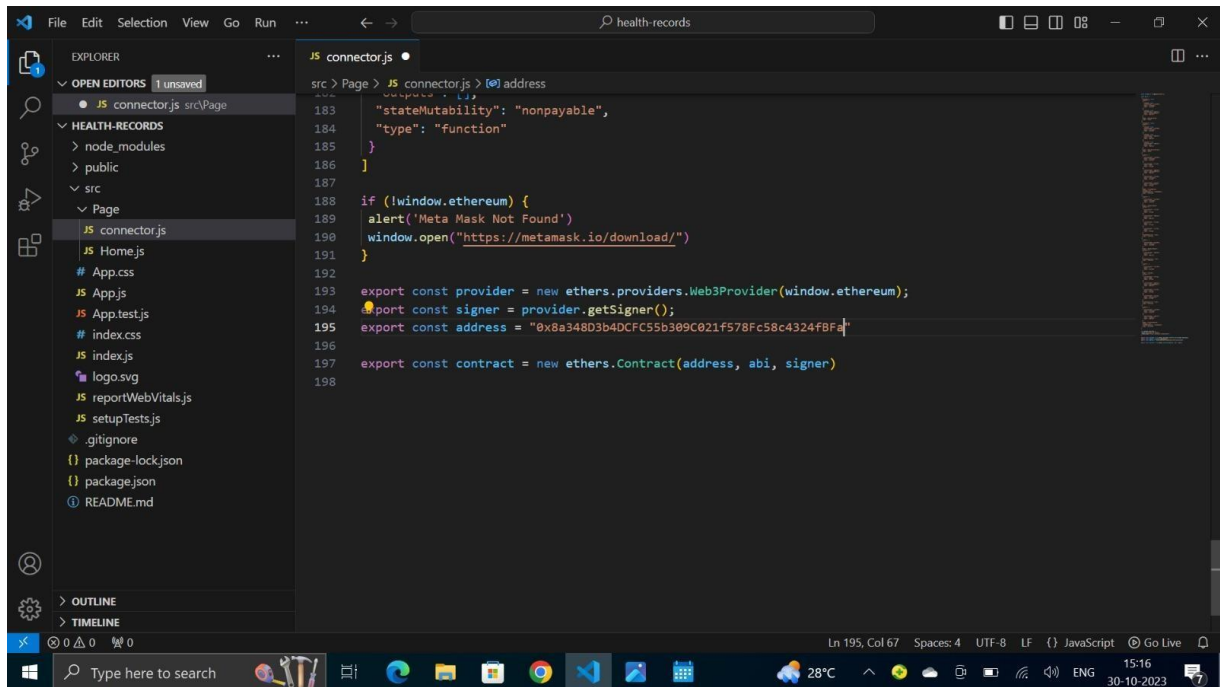


## 4. OPEN THE FILE EXPLORER

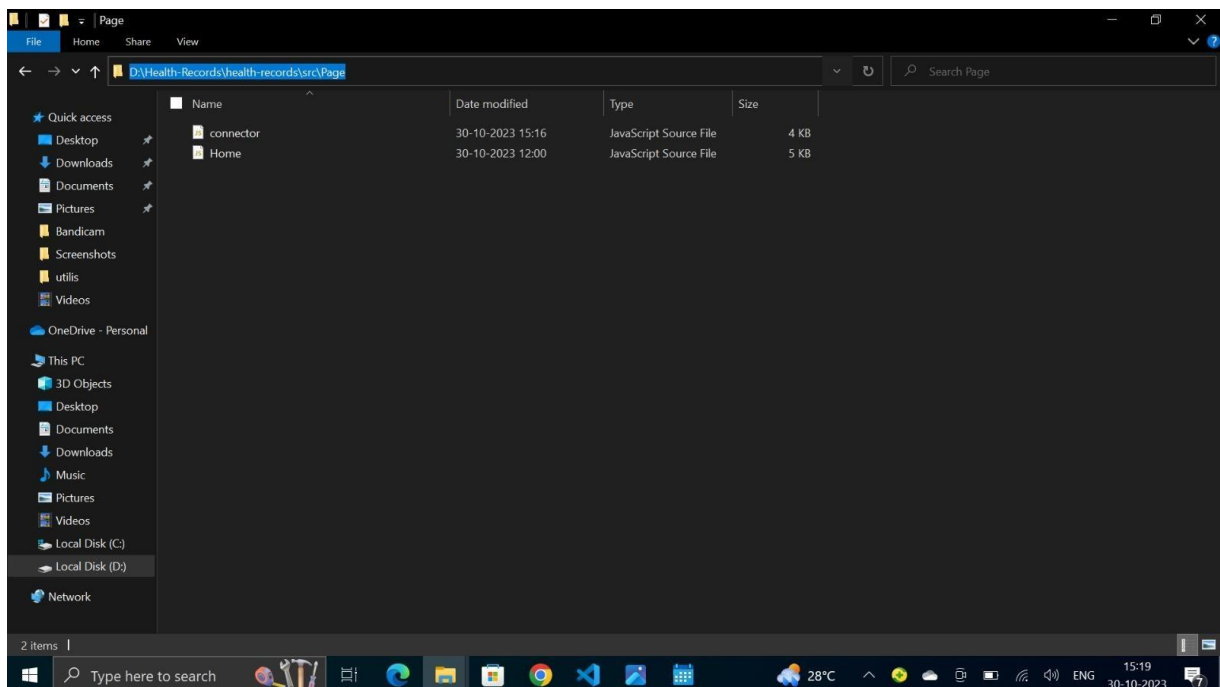
### Screenshot 1



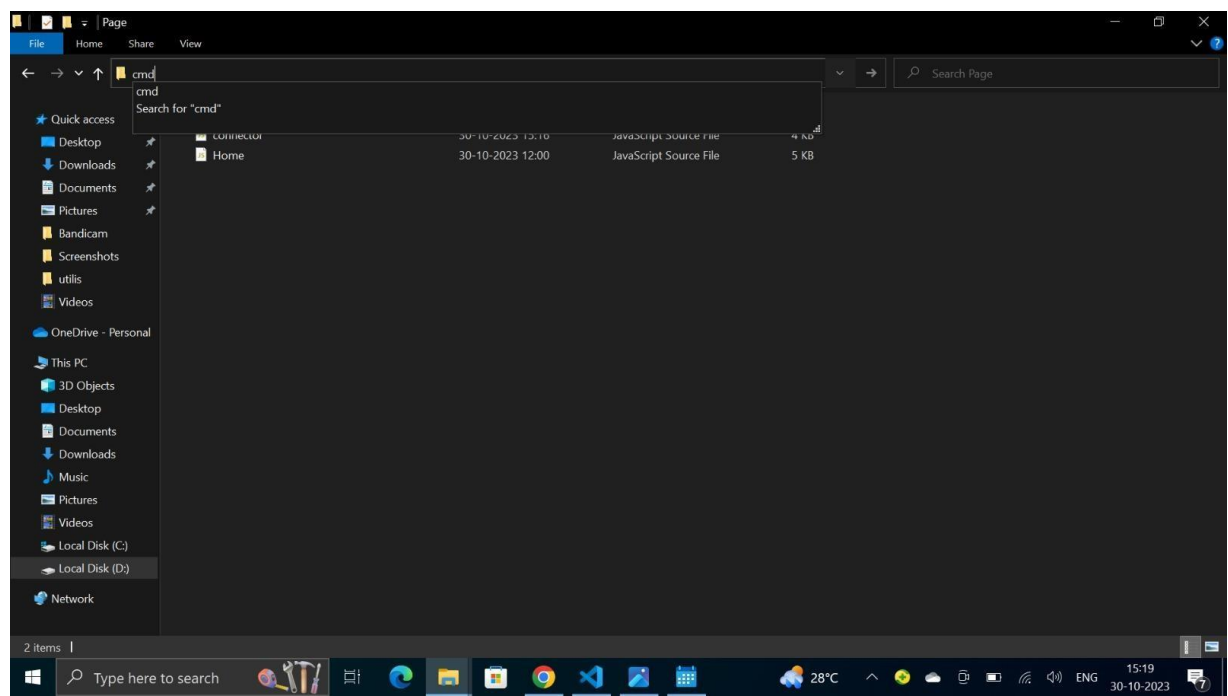
Screenshot 2



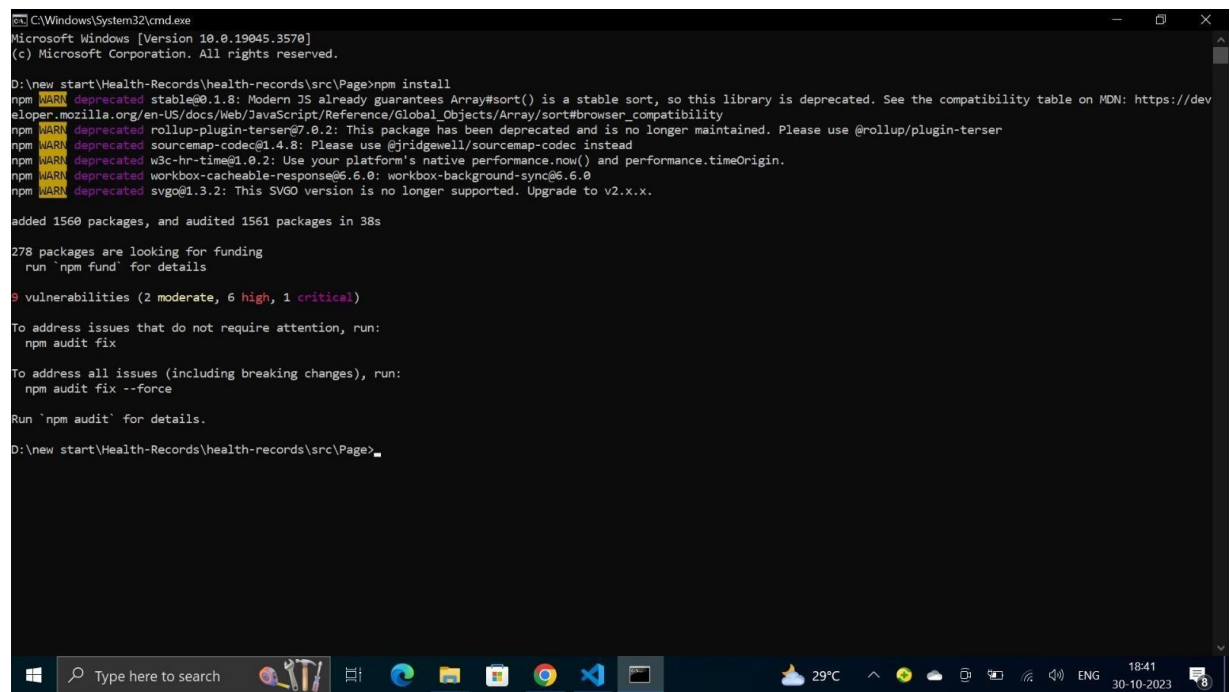
Screenshot 3



Screenshot 4



Screenshot 5



## Screenshot 6

```
C:\Windows\System32\cmd.exe
npm WARN deprecated stable@0.1.8: Modern JS already guarantees Array#sort() is a stable sort, so this library is deprecated. See the compatibility table on MDN: https://dev
eloper.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/sort#browser_compatibility
npm WARN deprecated rollup-plugin-terser@7.0.2: This package has been deprecated and is no longer maintained. Please use @rollup/plugin-terser
npm WARN deprecated sourcemap-codec@1.4.8: Please use @jridgewell/sourcemap-codec instead
npm WARN deprecated w3c-hr-time@1.0.2: Use your platform's native performance.now() and performance.timeOrigin.
npm WARN deprecated workbox-cacheable-response@6.6.0: workbox-background-sync@6.6.0
npm WARN deprecated svgo@1.3.2: This SVGO version is no longer supported. Upgrade to v2.x.x.

added 1560 packages, and audited 1561 packages in 38s

278 packages are looking for funding
  run `npm fund` for details

9 vulnerabilities (2 moderate, 6 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

D:\new start\Health-Records\health-records\src\>npm install bootstrap

changed 1 package, and audited 1561 packages in 8s

278 packages are looking for funding
  run `npm fund` for details

9 vulnerabilities (2 moderate, 6 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

D:\new start\Health-Records\health-records\src\>
```

## Screenshot 7

```
C:\Windows\System32\cmd.exe www.BANDICAM.com

278 packages are looking for funding
  run `npm fund` for details

9 vulnerabilities (2 moderate, 6 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

D:\new start\Health-Records\health-records\src\>npm install bootstrap

changed 1 package, and audited 1561 packages in 8s

278 packages are looking for funding
  run `npm fund` for details

9 vulnerabilities (2 moderate, 6 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

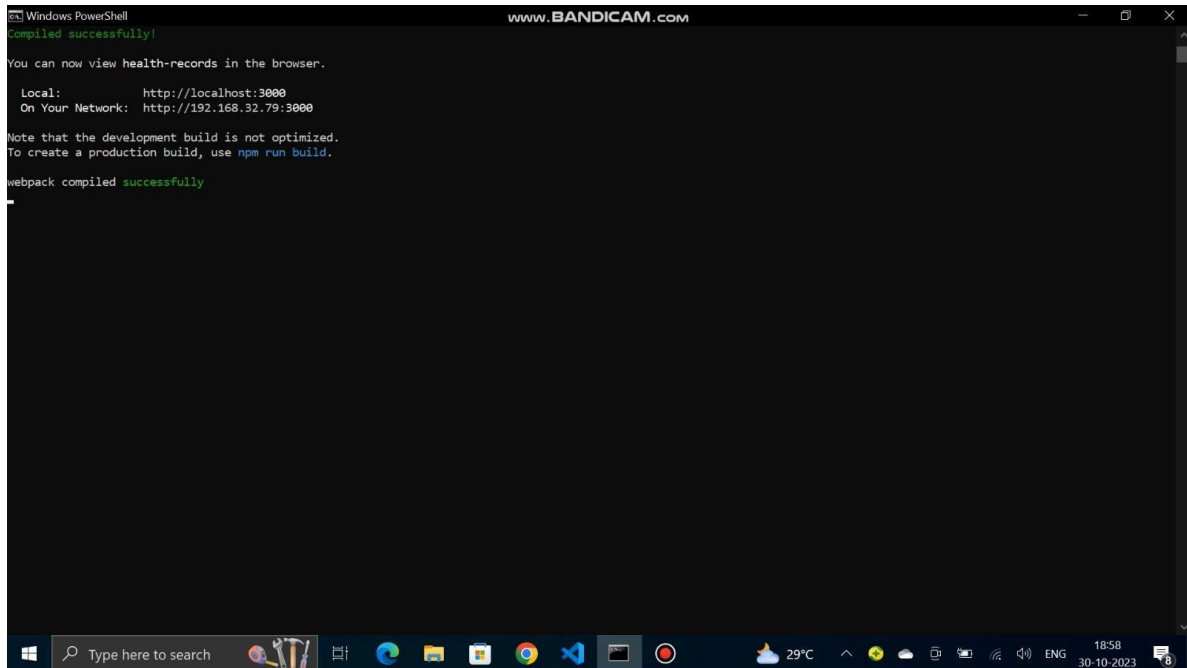
D:\new start\Health-Records\health-records\src\>npm start

> health-records@0.1.0 start
> react-scripts start

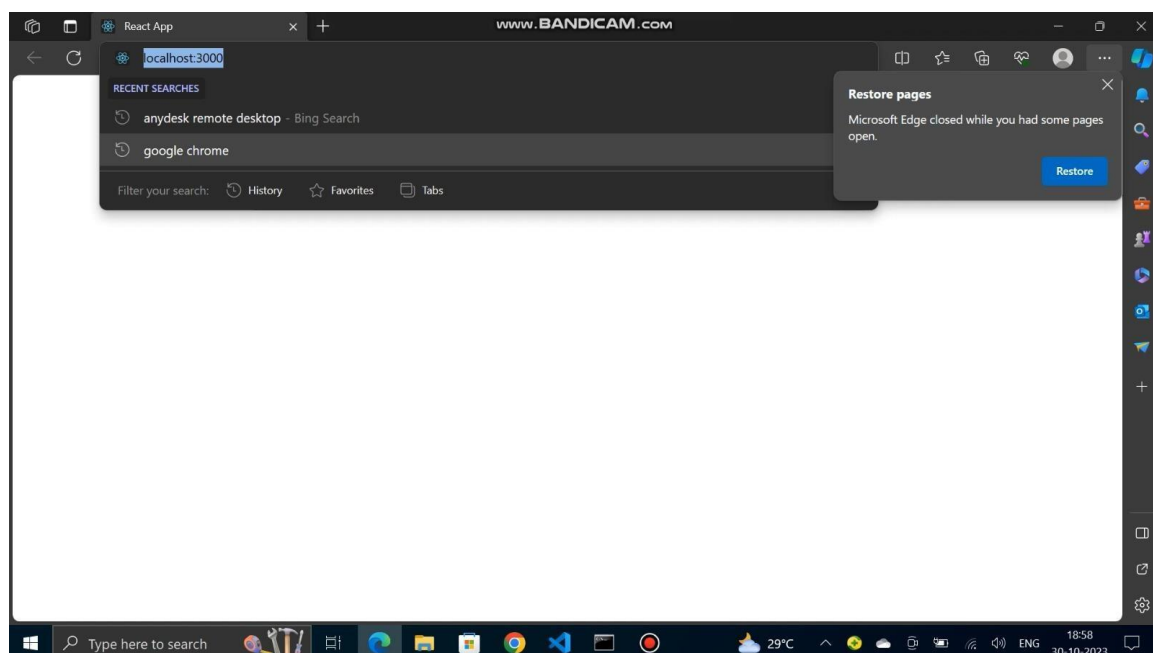
(node:12560) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' option.
(Use 'node --trace-deprecation ...' to show where the warning was created)
(node:12560) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Please use the 'setupMiddlewares' optio
n.
```

## 5. LOCALHOST IPADDRESS

Screenshot 1



Screenshot 2





## Screenshot 3

