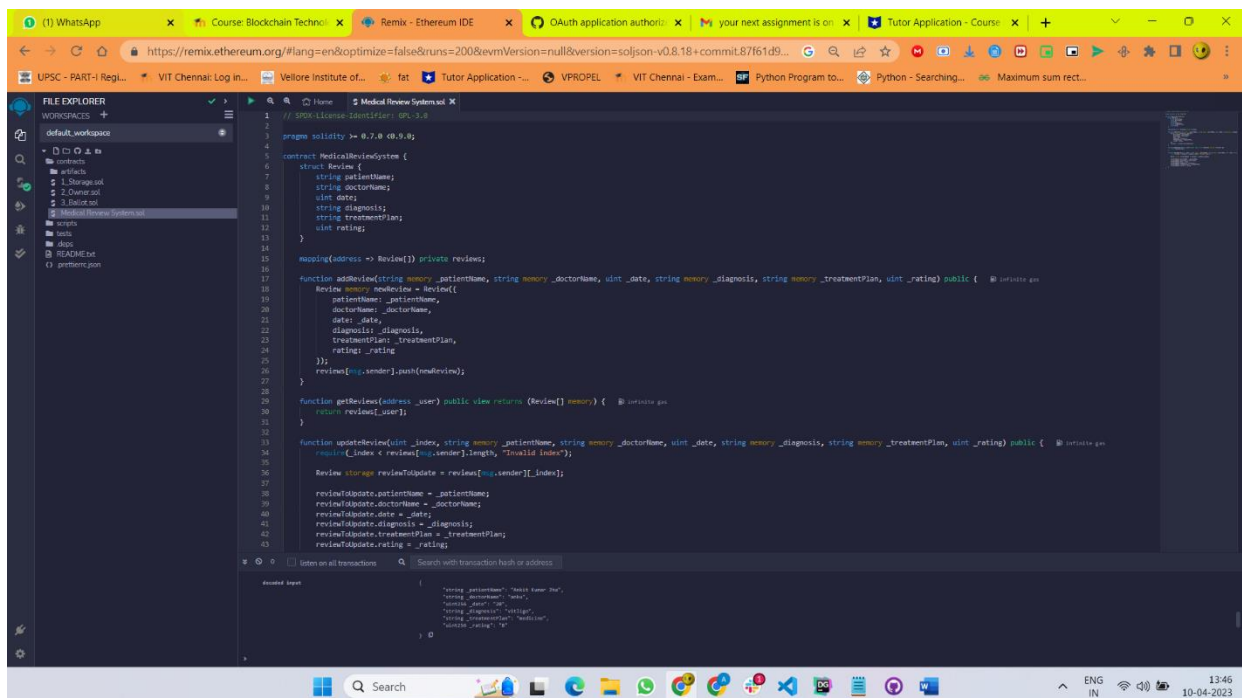


NAME: ANKIT KUMAR JHA  
REG.NO.20BPS1050

## DIGITAL ASSIGNMENT 1

Design a medical review system using solidity.



CODE:

```
// SPDX-License-Identifier: GPL-3.0

pragma solidity >= 0.7.0 <0.9.0;

contract MedicalReviewSystem {
    struct Review {
        string patientName;
        string doctorName;
        uint date;
        string diagnosis;
        string treatmentPlan;
        uint rating;
    }

    mapping(address => Review[]) private reviews;
```

```

function addReview(string memory _patientName, string memory _doctorName, uint _date, string memory
_diagnosis, string memory _treatmentPlan, uint _rating) public {

    Review memory newReview = Review({
        patientName: _patientName,
        doctorName: _doctorName,
        date: _date,
        diagnosis: _diagnosis,
        treatmentPlan: _treatmentPlan,
        rating: _rating
    });
    reviews[msg.sender].push(newReview);
}

function getReviews(address _user) public view returns (Review[] memory) {
    return reviews[_user];
}

function updateReview(uint _index, string memory _patientName, string memory _doctorName, uint _date, string
memory _diagnosis, string memory _treatmentPlan, uint _rating) public {
    require(_index < reviews[msg.sender].length, "Invalid index");

    Review storage reviewToUpdate = reviews[msg.sender][_index];

    reviewToUpdate.patientName = _patientName;
    reviewToUpdate.doctorName = _doctorName;
    reviewToUpdate.date = _date;
    reviewToUpdate.diagnosis = _diagnosis;
    reviewToUpdate.treatmentPlan = _treatmentPlan;
    reviewToUpdate.rating = _rating;
}
}

```

## addReview() functionality:

The screenshot displays the Remix IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' panel shows the 'MEDICALREVIEWSYSTEM AT 0xD91...39138' contract. The 'addReview' function is selected, and its parameters are filled in: \_patientName: 'Akhil Kumar Jha', \_doctorName: 'anku', \_date: '20', \_diagnosis: 'vittigo', \_treatmentPlan: 'medicine', and \_rating: '8'. The 'transact' button is highlighted. On the right, the Solidity code for the 'MedicalReviewSystem' contract is visible. Below the code, the 'Transaction mined and execution succeed' message is shown, along with transaction details such as hash, from, to, gas, transaction cost, execution cost, input, and decoded input. The decoded input shows the parameters passed to the 'addReview' function.

## getReview() functionality:

The screenshot shows the Remix IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' panel is open, displaying the 'getReviews' function. The 'call' button is highlighted, and the 'Low level interactions' section shows the transaction details. The main editor displays the Solidity code for the 'MedicalReviewSystem' contract, which includes a 'Review' struct and a 'getReviews' function. The output panel on the right shows the decoded output of the transaction, which is a tuple of strings and uints representing a review record.

```
1 // SPDX-License-Identifier: GPL-3.0
2
3 pragma solidity >= 0.7.0 <0.9.0;
4
5 contract MedicalReviewSystem {
6     struct Review {
7         string patientName;
8         string doctorName;
9         uint date;
10        string diagnosis;
11        string treatmentPlan;
12    }
13
14    function getReviews() public view returns (Review[] memory) {
15        // ... (implementation details) ...
16    }
17 }
```

Transaction details for 'getReviews':

- From: 0x5838D6a701c568545dCf8B3fC8B75F56beddC4
- To: MedicalReviewSystem.getReviews(address)
- Data: 0xdef...eddc4

Decoded output:

```
[{"string_patientName": "Ankit Kumar Sharma",
  "string_doctorName": "Ankur",
  "uint256_date": "20",
  "string_diagnosis": "vitiligo",
  "string_treatmentPlan": "medicine",
  "uint256_rating": "8"}]
```

## updateReview() functionality:

The screenshot shows the Remix IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' panel is open, displaying the 'updateReview' function. The 'call' button is highlighted, and the 'Low level interactions' section shows the transaction details. The main editor displays the Solidity code for the 'MedicalReviewSystem' contract, which includes a 'Review' struct and an 'updateReview' function. The output panel on the right shows the decoded input of the transaction, which is a tuple of strings and uints representing a review record.

```
1 // SPDX-License-Identifier: GPL-3.0
2
3 pragma solidity >= 0.7.0 <0.9.0;
4
5 contract MedicalReviewSystem {
6     struct Review {
7         string patientName;
8         string doctorName;
9         uint date;
10        string diagnosis;
11        string treatmentPlan;
12    }
13
14    function updateReview(uint256 _index, string _patientName, string _doctorName, uint256 _date, string _diagnosis, string _treatmentPlan, uint256 _rating) public {
15        // ... (implementation details) ...
16    }
17 }
```

Transaction details for 'updateReview':

- From: 0x5838D6a701c568545dCf8B3fC8B75F56beddC4
- To: MedicalReviewSystem.updateReview(uint256,string,string,uint256,string,string,uint256)
- Data: 0xd91...39138

Decoded input:

```
{
  "uint256_index": "0",
  "string_patientName": "Ankit",
  "string_doctorName": "Ankur",
  "uint256_date": "20",
  "string_diagnosis": "vitiligo",
  "string_treatmentPlan": "medicine",
  "uint256_rating": "8"
}
```

## getReview() functionality:

The screenshot displays the Remix IDE interface, showing the deployment and execution of a smart contract. The left sidebar contains the 'DEPLOY & RUN TRANSACTIONS' panel, which is divided into sections for 'updateReview' and 'getReviews'. The 'updateReview' section shows a transaction being executed with the following parameters:

- \_index: 0
- \_patientName: Ankit
- \_doctorName: anku
- \_date: 20
- \_diagnosis: vitiligo
- \_treatmentPlan: medicine
- \_rating: 9

The 'getReviews' section shows a transaction being executed with the following parameters:

- \_index: 0
- \_patientName: Ankit
- \_doctorName: anku
- \_date: 20
- \_diagnosis: vitiligo
- \_treatmentPlan: medicine
- \_rating: 9

The 'Low level interactions' section shows the transaction data for the 'getReviews' call, including the transaction hash and the function being called.

The main editor on the right displays the Solidity code for the 'MedicalReviewSystem' contract. The code includes a 'Review' struct and a 'getReviews' function. The 'getReviews' function is highlighted with a red circle.

```
1 // SPDX-License-Identifier: GPL-3.0
2
3 pragma solidity >= 0.7.0 <0.9.0;
4
5 contract MedicalReviewSystem {
6     struct Review {
7         uint256 _index;
8         string _patientName;
9         string _doctorName;
10        string _date;
11        string _diagnosis;
12        string _treatmentPlan;
13        string _rating;
14    }
15
16     Review[] reviews;
17
18     function getReviews(uint256 _index) public view returns (Review memory) {
19         return reviews[_index];
20     }
21 }
```

The bottom status bar shows the transaction hash and the function being called: [call] from: 0x58380a701c568545dcfc803fc8875f56beddC4 to: MedicalReviewSystem.getReviews(address) data: 0x4ef...eddc4