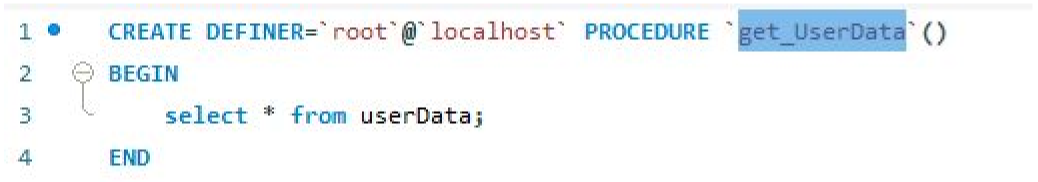
MySQL with Node.js

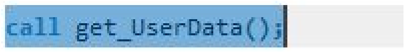
1. Calling the Stored Procedure

Step 1: Create the Stored Procedure on your MySQL server



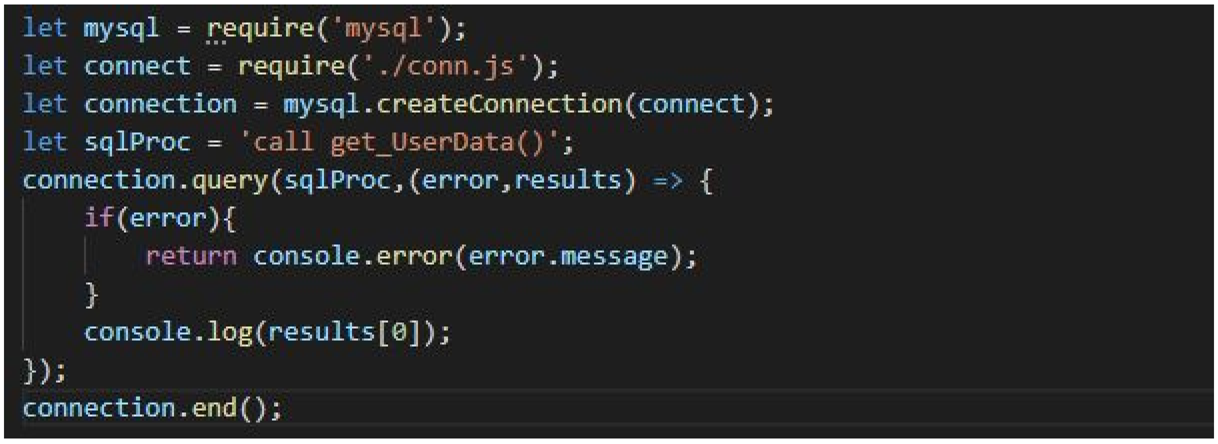
```
CREATE DEFINER=`root`@`localhost` PROCEDURE `get_UserData`()
BEGIN
    select * from userData;
END
```

And test for the Stored procedure execution also

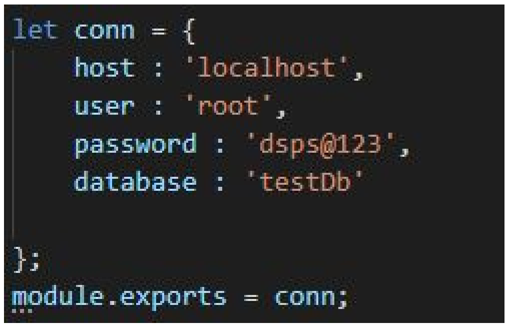


```
call get_UserData();
```

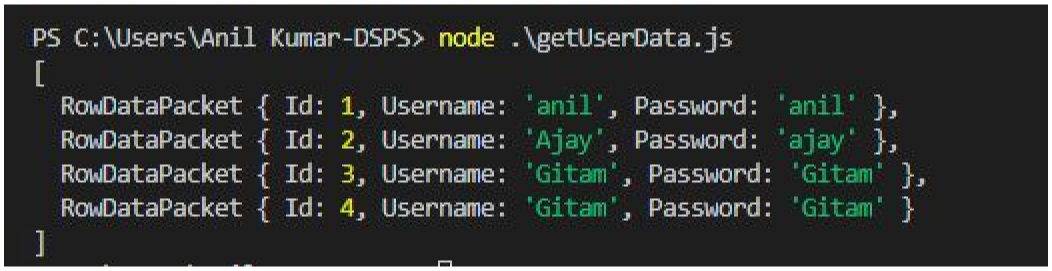
Step 2: Front end application and Call this procedure using Node.js



```
let mysql = require('mysql');
let connect = require('./conn.js');
let connection = mysql.createConnection(connect);
let sqlProc = 'call get_UserData()';
connection.query(sqlProc,(error,results) => {
    if(error){
        return console.error(error.message);
    }
    console.log(results[0]);
});
connection.end();
```



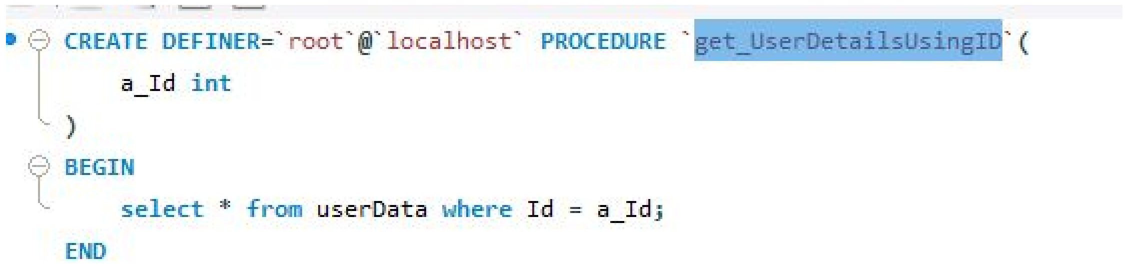
```
let conn = {
    host : 'localhost',
    user : 'root',
    password : 'dsps@123',
    database : 'testDb'

};
module.exports = conn;
```



```
PS C:\Users\Anil Kumar-DSPS> node .\getUserData.js
[
  RowDataPacket { Id: 1, Username: 'anil', Password: 'anil' },
  RowDataPacket { Id: 2, Username: 'Ajay', Password: 'ajay' },
  RowDataPacket { Id: 3, Username: 'Gitam', Password: 'Gitam' },
  RowDataPacket { Id: 4, Username: 'Gitam', Password: 'Gitam' }
]
```

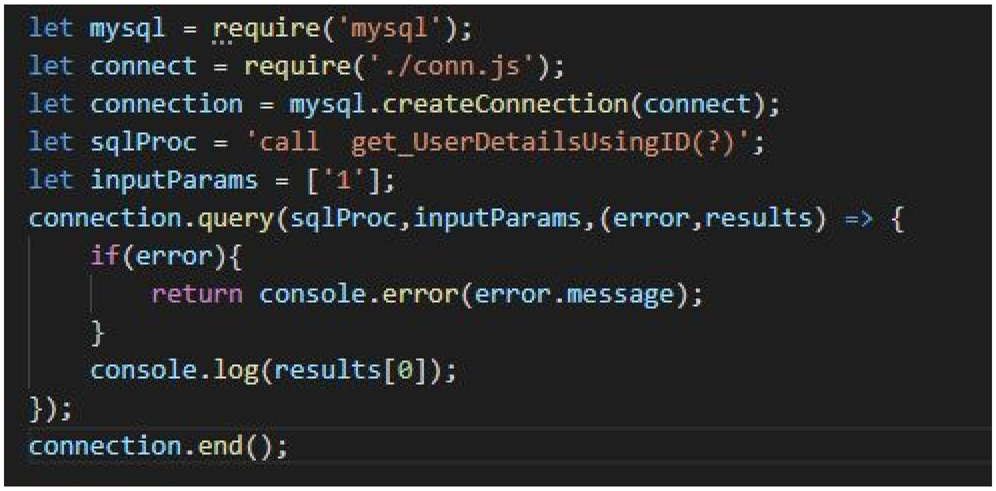
Example with Filters

Step1: First let;s create the stored procedure on your server

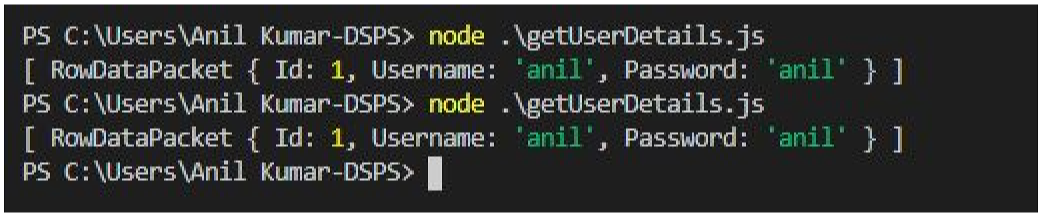


```
CREATE DEFINER=`root`@`localhost` PROCEDURE `get_UserDetailsUsingID`(
    a_Id int
)
BEGIN
    select * from userData where Id = a_Id;
END
```

Step 2: Go to front end application and call this procedure, save it into an object

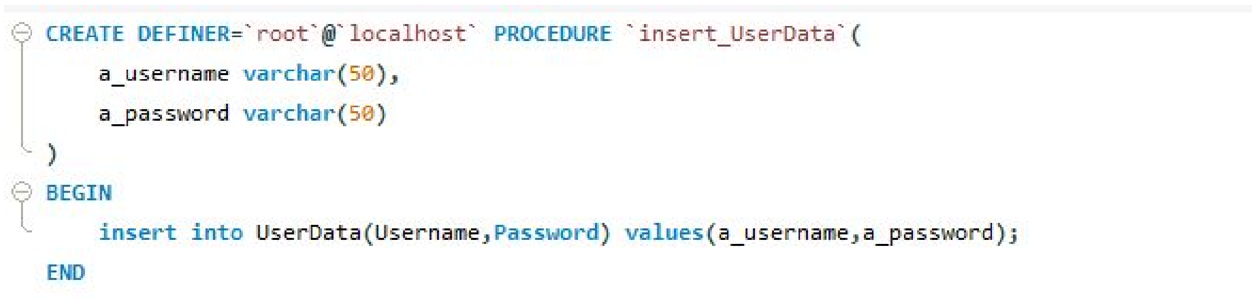


```
let mysql = require('mysql');
let connect = require('./conn.js');
let connection = mysql.createConnection(connect);
let sqlProc = 'call  get_UserDetailsUsingID(?)';
let inputParams = ['1'];
connection.query(sqlProc,inputParams,(error,results) => {
    if(error){
        return console.error(error.message);
    }
    console.log(results[0]);
});
connection.end();
```

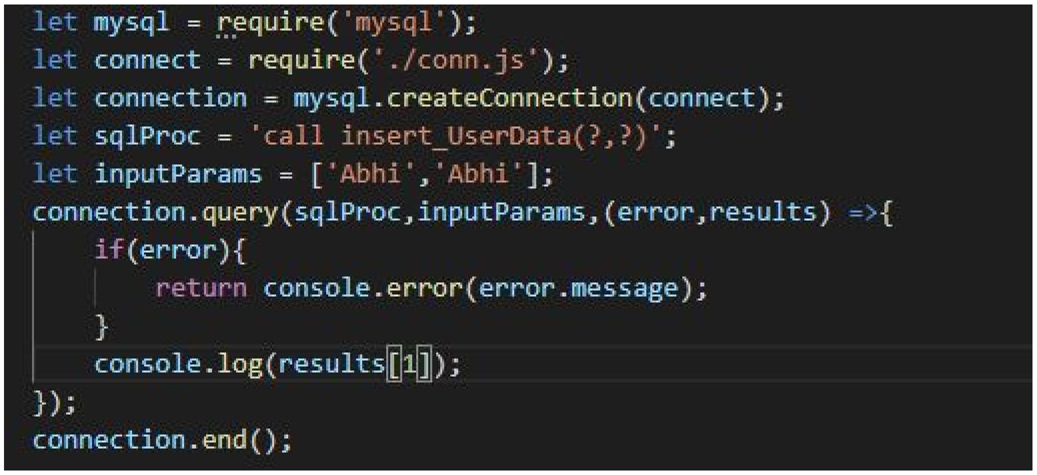


```
PS C:\Users\Anil Kumar-DSPS> node .\getUserDetails.js
[ RowDataPacket { Id: 1, Username: 'anil', Password: 'anil' } ]
PS C:\Users\Anil Kumar-DSPS> node .\getUserDetails.js
[ RowDataPacket { Id: 1, Username: 'anil', Password: 'anil' } ]
PS C:\Users\Anil Kumar-DSPS>
```

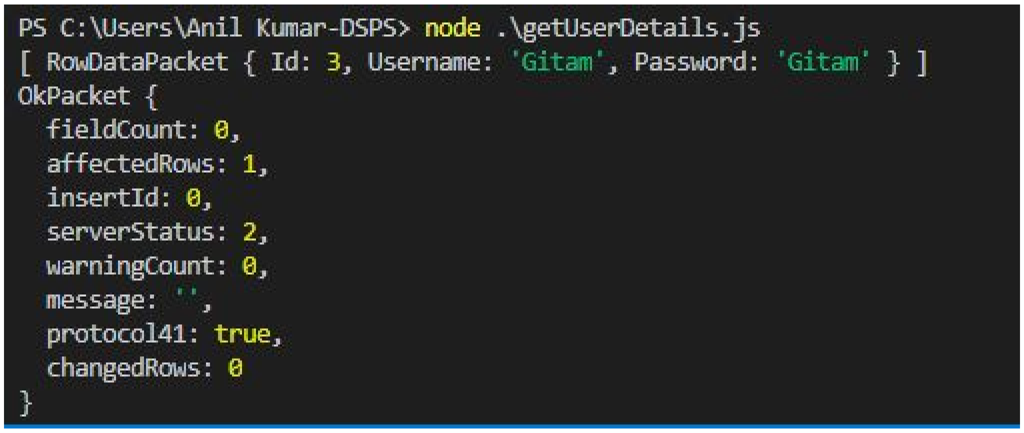
Task : Write the Procedure for insertion and call the procedure from the Node.js application



```
CREATE DEFINER=`root`@`localhost` PROCEDURE `insert_UserData`(
    a_username varchar(50),
    a_password varchar(50)
)
BEGIN
    insert into UserData(Username,Password) values(a_username,a_password);
END
```

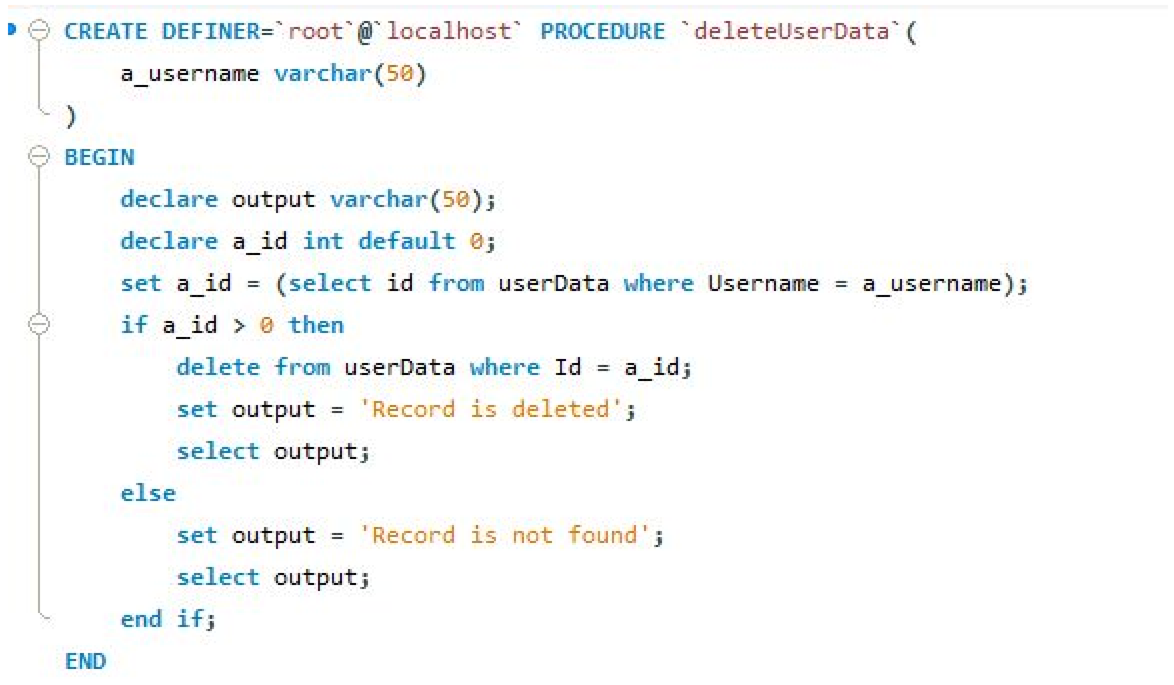


```
let mysql = require('mysql');
let connect = require('./conn.js');
let connection = mysql.createConnection(connect);
let sqlProc = 'call insert_UserData(?,?)';
let inputParams = ['Abhi','Abhi'];
connection.query(sqlProc,inputParams,(error,results) =>{
    if(error){
        return console.error(error.message);
    }
    console.log(results[1]);
});
connection.end();
```



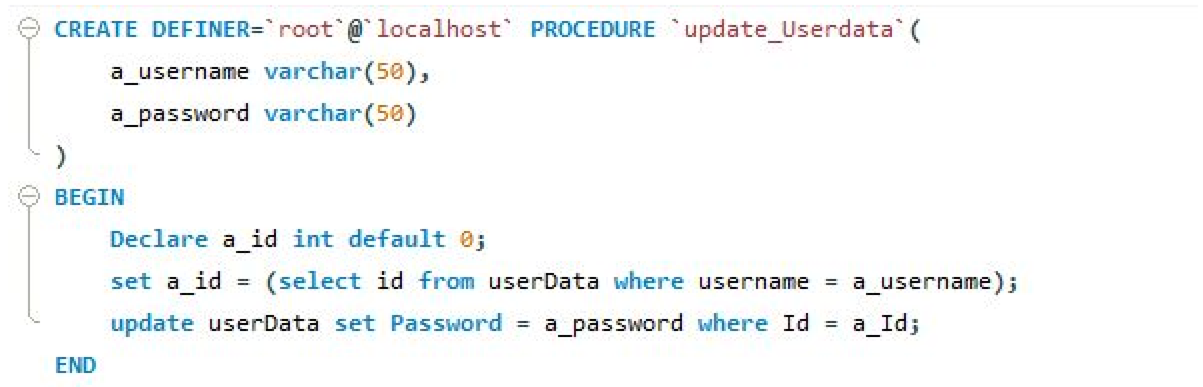
```
PS C:\Users\Anil Kumar-DSPS> node .\getUserDetails.js
[ RowDataPacket { Id: 3, Username: 'Gitam', Password: 'Gitam' } ]
OkPacket {
  fieldCount: 0,
  affectedRows: 1,
  insertId: 0,
  serverStatus: 2,
  warningCount: 0,
  message: '',
  protocol41: true,
  changedRows: 0
}
```

Task : Write the Procedure for Delete and call the procedure from the Node.js application

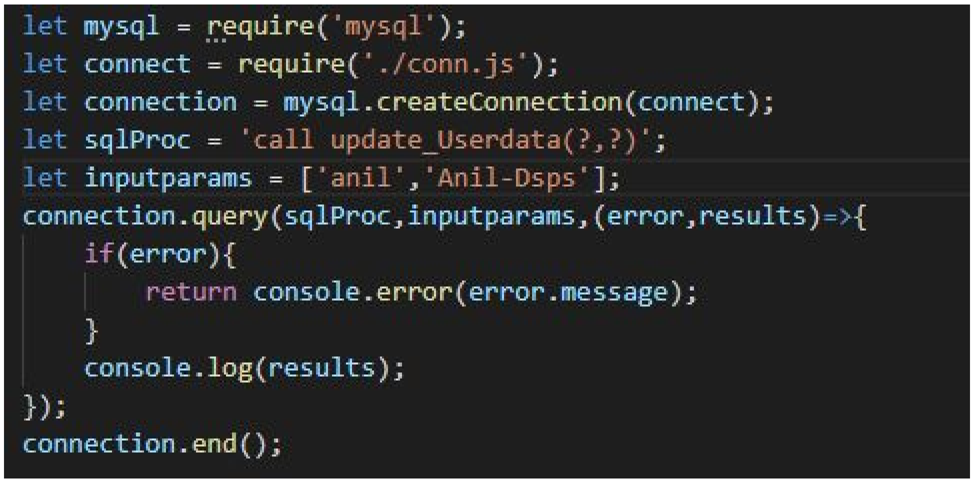


```sql
CREATE DEFINER=`root`@`localhost` PROCEDURE `deleteUserData`(
    a_username varchar(50)
)
BEGIN
    declare output varchar(50);
    declare a_id int default 0;
    set a_id = (select id from userData where Username = a_username);
    if a_id > 0 then
        delete from userData where Id = a_id;
        set output = 'Record is deleted';
        select output;
    else
        set output = 'Record is not found';
        select output;
    end if;
END
```

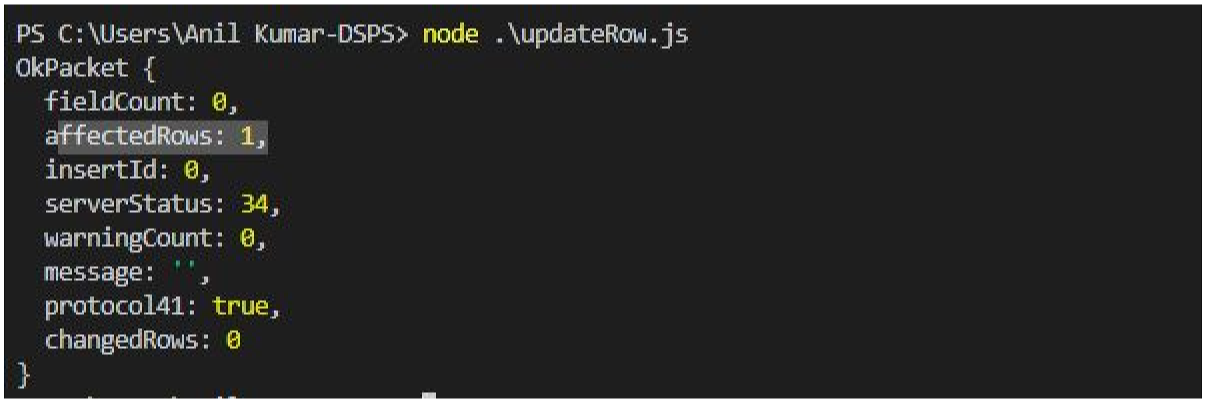
Task : Write the Procedure for Update and call the procedure from the Node.js application



```sql
CREATE DEFINER=`root`@`localhost` PROCEDURE `update_Userdata`(
    a_username varchar(50),
    a_password varchar(50)
)
BEGIN
    Declare a_id int default 0;
    set a_id = (select id from userData where username = a_username);
    update userData set Password = a_password where Id = a_Id;
END
```

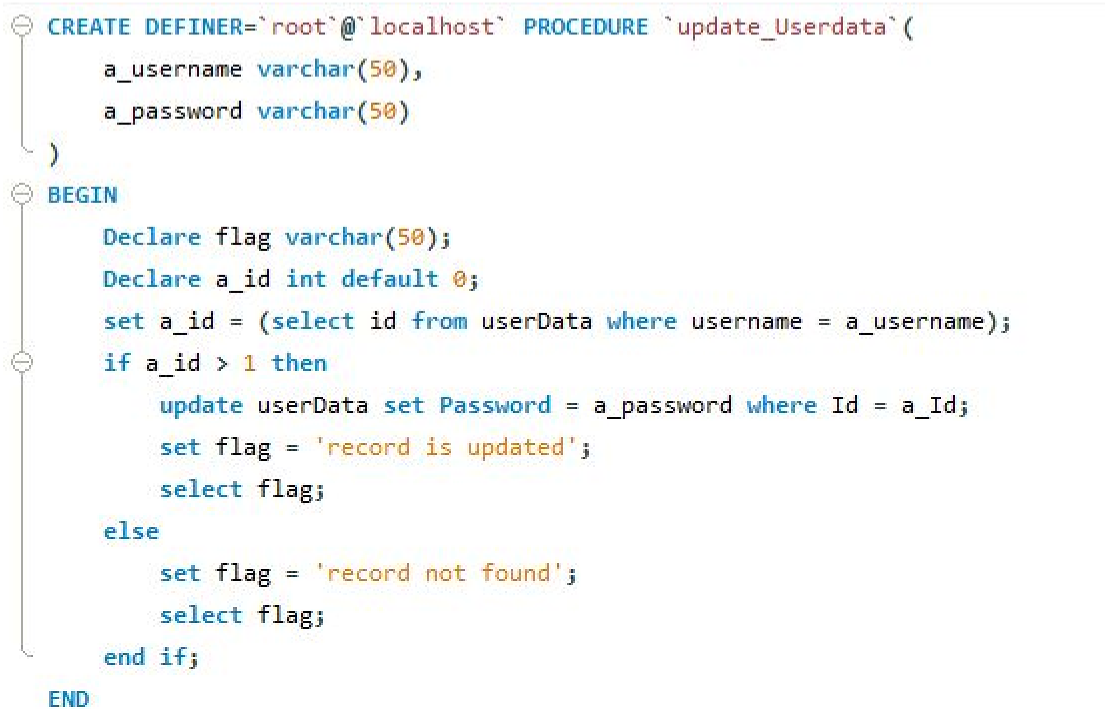


```
let mysql = require('mysql');
let connect = require('./conn.js');
let connection = mysql.createConnection(connect);
let sqlProc = 'call update_Userdata(?,?)';
let inputparams = ['anil','Anil-Dsps'];
connection.query(sqlProc,inputparams,(error,results)=>{
    if(error){
        return console.error(error.message);
    }
    console.log(results);
});
connection.end();
```

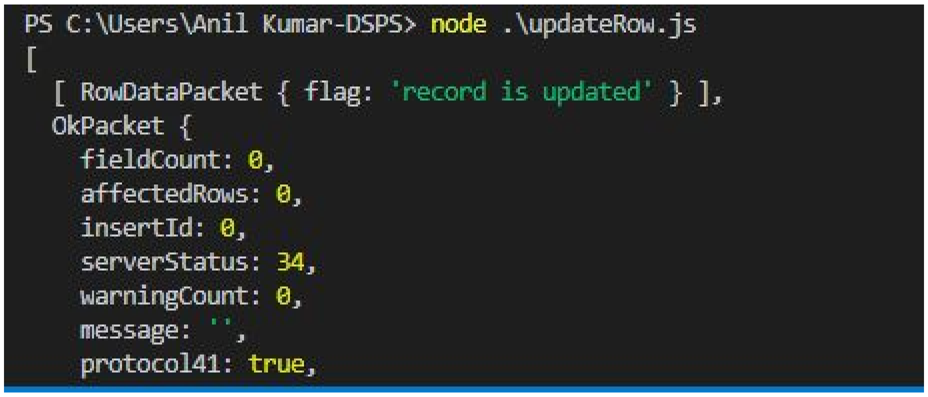


```
PS C:\Users\Anil Kumar-DSPS> node .\updateRow.js
OkPacket {
  fieldCount: 0,
  affectedRows: 1,
  insertId: 0,
  serverStatus: 34,
  warningCount: 0,
  message: '',
  protocol41: true,
  changedRows: 0
}
```

Change the Procedure on MySQL server and again execute the application



```
CREATE DEFINER=`root`@`localhost` PROCEDURE `update_Userdata`(
    a_username varchar(50),
    a_password varchar(50)
)
BEGIN
    Declare flag varchar(50);
    Declare a_id int default 0;
    set a_id = (select id from userData where username = a_username);
    if a_id > 1 then
        update userData set Password = a_password where Id = a_Id;
        set flag = 'record is updated';
        select flag;
    else
        set flag = 'record not found';
        select flag;
    end if;
END
```

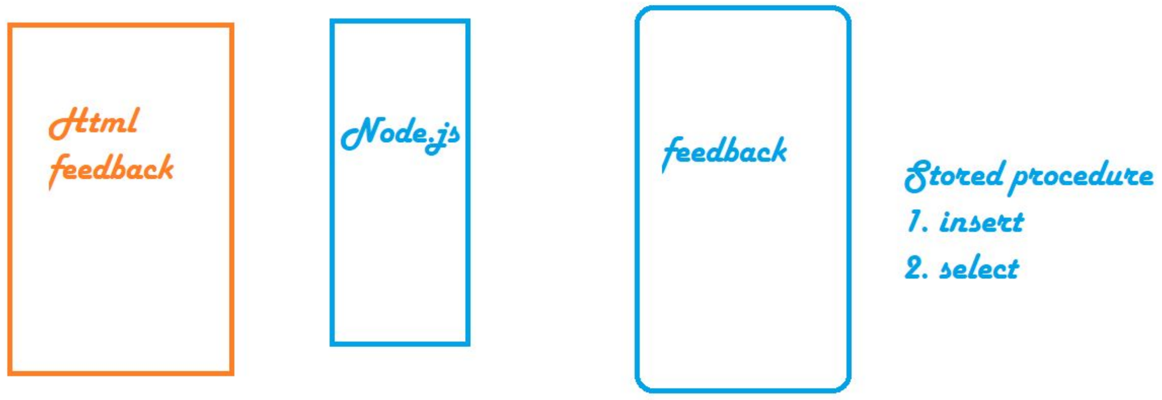


```
PS C:\Users\Anil Kumar-DSPS> node .\updateRow.js
[
  [ RowDataPacket { flag: 'record is updated' } ],
  OkPacket {
    fieldCount: 0,
    affectedRows: 0,
    insertId: 0,
    serverStatus: 34,
    warningCount: 0,
    message: '',
    protocol41: true,
```

Task: Create a Small Feedback form with two input fields

1. EmailId
2. Feedback

First of all you need to create the table on a database with few fields.



Two packages to be installed in your application

a. npm install express
b. npm install body-parser