**ONLINE AUCTION SYSTEM**

**REVIEW-3 REPORT**

*Submitted in partial fulfillment of the*

*requirement for the award of the*

*Degree of*

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

*by*

**Rohith Reddy Dirisala (19BCE7303)**

**Hruthik Yadav Madati (19BCN7092)**

*Under the Guidance of*

**DR. AJAY KUMAR MALLICK**



SCHOOL OF ELECTRONICS ENGINEERING

VIT-AP UNIVERSITY

AMARAVATI- 522237

*JANUARY 2021*

**CERTIFICATE**

This is to certify that the Capstone Project work titled “**ONLINE AUCTION SYSTEM**” that is being submitted by **Rohith Reddy Dirisala(19BCE7303)** and **Hruthik Yadav Madati(19BCN7092)** is in partial fulfillment of the requirements for the award of Bachelor of Technology, is a record of bonafide work done under my guidance. The contents of this Project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University for award of any degree or diploma and the same is certified.

Dr. AJAY KUMAR MALLICK

Guide

**The thesis is satisfactory / unsatisfactory**

**Internal Examiner External Examiner**

**Approved by**

**PROGRAM CHAIR DEAN**

B. Tech. CSE School Of Computer Science and Engineering

**ACKNOWLEDGEMENTS**

**ABSTRACT**

In this paper an efficient system is presented which can facilitate easy bidding and listing of various items worldwide . Traditional Auction houses have been around for a very long time. But they have certain shortcomings such as the scope of the bidding which is mostly restricted to local population. So, we are building a website for the auctions to be available for larger section of people and can be completed effectively.

The system is designed to allow users to set up their products for auctions and bidders to register and bid for various products available for bidding. Since the items are put to bidding online, we cannot the verify the entities put by regular users online. Hence, we are considering to allow the items of enterprises only to be put on the website for bidding. This ensures the quality of the products put on the website.

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No.** | **Chapter** | **Title** | **Page Number** |
| **1.** |  | **Acknowledgement** | **1** |
| **2.** |  | **Abstract** | **3** |
| **3.** | **1**  **1.1** | **Introduction**  **Objectives** | **6** |
| **4** | **2**  **2.1**  **2.2**  **2.4** | **ONLINE AUCTION SYSTEM**  **Proposed System**  **Working Methodology**  **System Details** | **7**  **8**  **10** |
| **5** | **3** | **Cost Analysis** | **16** |
| **6** | **4** | **Appendix** | **17** |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **Page No.** |
| **1** | **System Use-case diagram** | **7** |
| **2**  **3** | **ER diagram**  **Login Page** | **8**  **10** |
| **4** | **Userlogin page** | **11** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| **5** | **SellerLogin page** | **12** |
| **6** | **Full Bid page** | **13** |
| **7** | **Bid Items table** | **14** |
|  |  |  |
|  |  |  |
|  |  |  |
| **8**  **9**  **10** | **User info table**  **Seller info table**  **Table of Cost Analysis** | **14**  **15**  **16** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**CHAPTER 1**

**INTRODUCTION**

In a traditional auction, potential buyers get the chance to examine items up close and ask questions before purchasing. Bidders get an idea about what type of items will go up for auction and can estimate how much items may go for. Traditional auctions allow buyers to discuss items with the seller and get details about them before the auction begins. Traditional auctions also allow buyers to make educated purchasing decisions in person.

The main difference between traditional auctions and online auctions is that online auctions are meant for convenience. Items are listed on auction house websites in a catalog fashion, and buyers can click on them to learn more details about them. An online auction may be less expensive for sellers due to the elimination of costs associated with holding a traditional auction, but may limit buyers.Once buyers see items they are interested in, they can bid for purchases using their credit card information.

Online auctions are available to more potential buyers, not just buyers located in a specific area near the auction house. Having a wider range of bidders during an auction can be beneficial to the seller.

* 1. **Objectives**

The following are the objectives of this project:

* To design an efficient system which can automatically manage the listed items for bidding and help sellers from remote places to list their items online for bidding
* The Auction system is built to facilitate many users to concurrently bid on a certain item/service
* It simplifies the process of auction by handling all the transactions between the user and the seller using a robust web application system

**1.3 Organization of the Report**

The remaining chapters of the project report are described as follows:

* Chapter 2 contains the proposed system, methodology and software details.
* Chapter 3 gives the cost involved in the implementation of the project.
* Chapter 4 discusses the results obtained after the project was implemented.
* Chapter 5 concludes the report.
* Chapter 6 consists of codes.

**CHAPTER 2**

**ONLINE AUCTION SYSTEM**

This Chapter describes the proposed system, working methodology, software and hardware details.

**2.1 Proposed System**

The following block diagram (figure 1) shows the system architecture of this project.

Admin

Seller

Customer

Figure 1 System Use-case Diagram



**Figure 2 ER diagram**

**2.2 Working Methodology**

The system has two sets of users

Users

Sellers

Depending on the type of users , various functionalities are embedded in the system

Users have the majority of the functionality in the website since everything in the system is designed around the user accounts . One of the primary feature of the user account type is the Bidding functinality which enables the user accounts to bid on any item of their choice unless they have the required credits in their account. Some other features include viewing the seller of a particular bid , time remaining for the bid, current bid amount . The user can also view their information stored in the database.

Sellers have very limited functionality given that it is not intended for any bidding activites.It is just used for listing an item, viewing the listed items , viewing the current Bid amount , current Bidder . The sellers also could view their information stored on the website using the view accountInfo option.

**2.4 System Details**

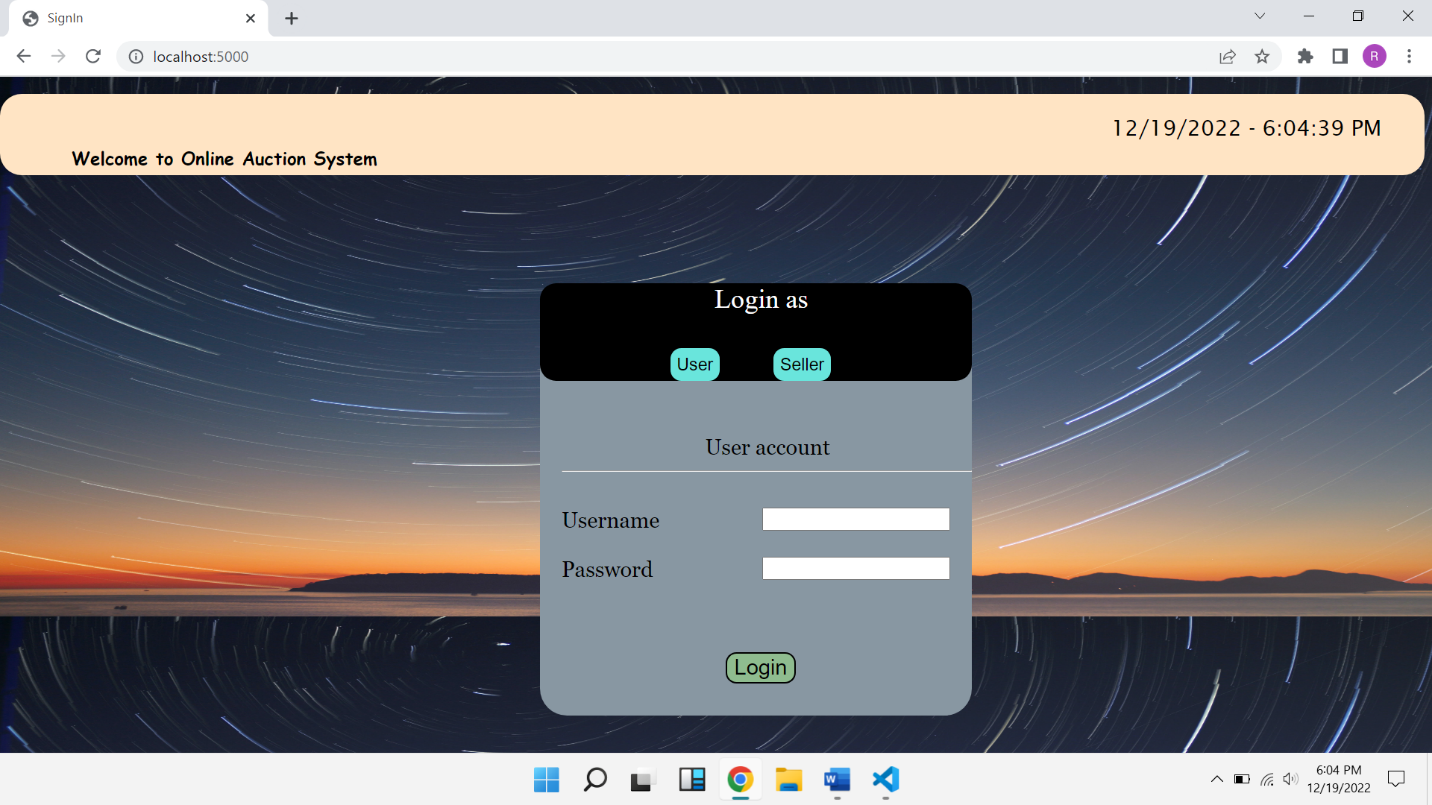
This section describes the software and hardware details of the system:

The overview of the web application could be divided into three components for our easy understanding

1. Frontend/Client side
2. Backend/Server side
3. Database

**I. Frontend/Client side**

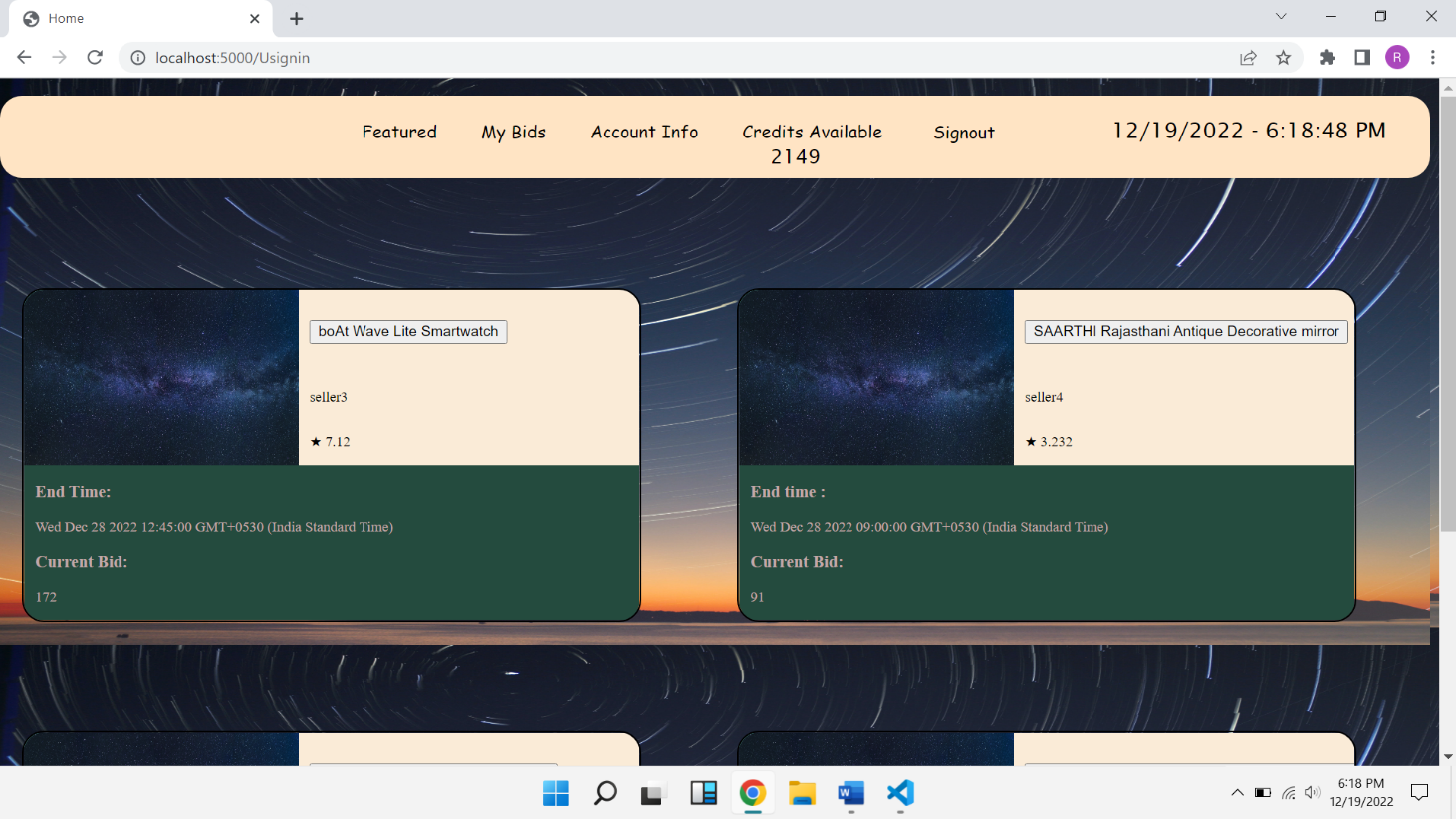
**Login page:**

****

**Figure3**

* A **<span> element** is used to display the live time at the top right corner of the page using a javascript function which keeps on executing every 500 milliseconds to display the live time accurately.
* A form is created for two different types of account namely **user and seller** which has two **post** methods for each separate type.
* The two forms transistion without a page reload with the help of javascript display properties .

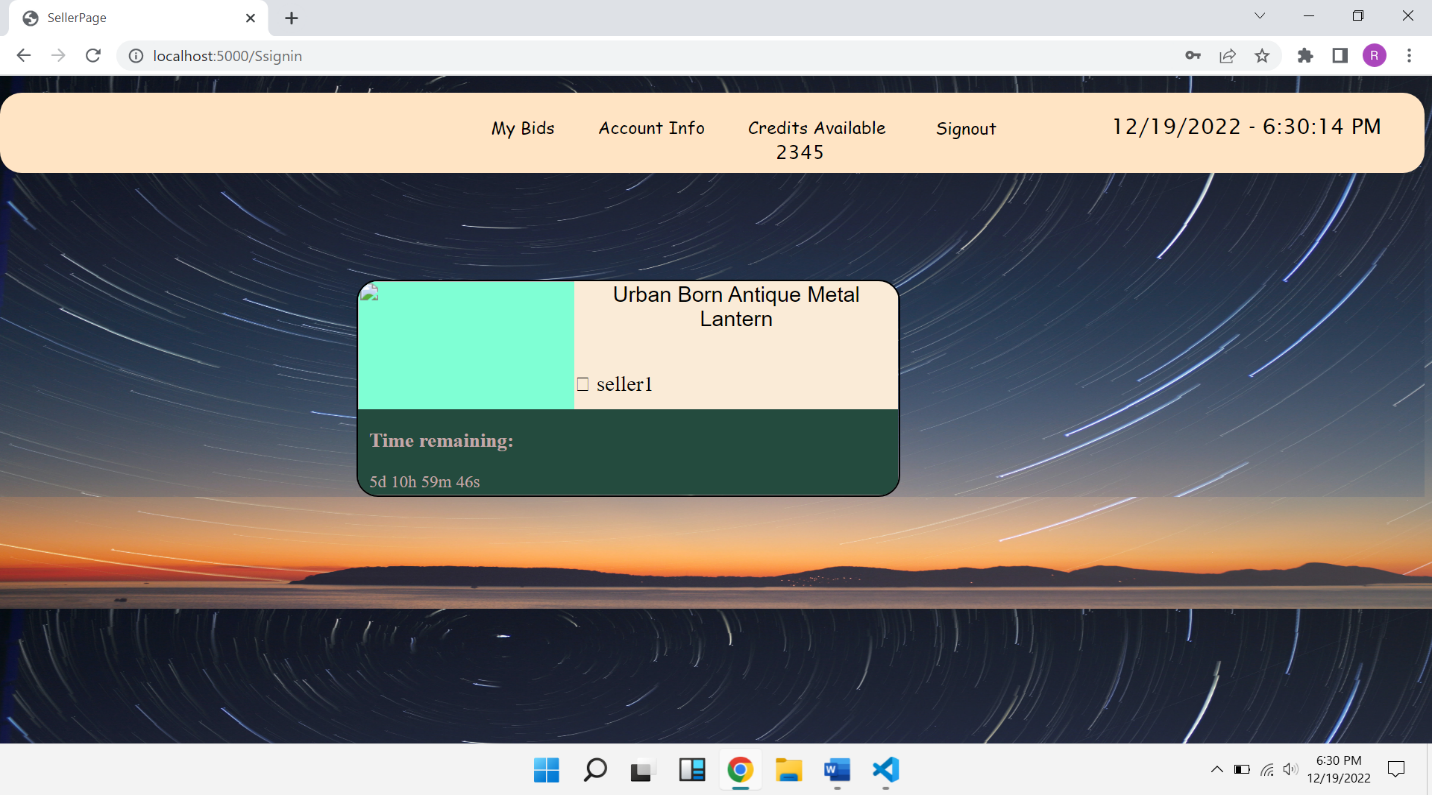
**User login Page:**



**Figure4**

* This is the **home page** for a User
* The **navigation bar** consists of
  + - Featured
    - My Bids
    - Account Info
    - Credis Available to a user
    - Signout button
    - Live time element
* The navigation bar helps the user to navigate through various modules with ease without having to reload the page entirely
* Then the ongoing Bids are displayed using a <div> template and styled using external css file

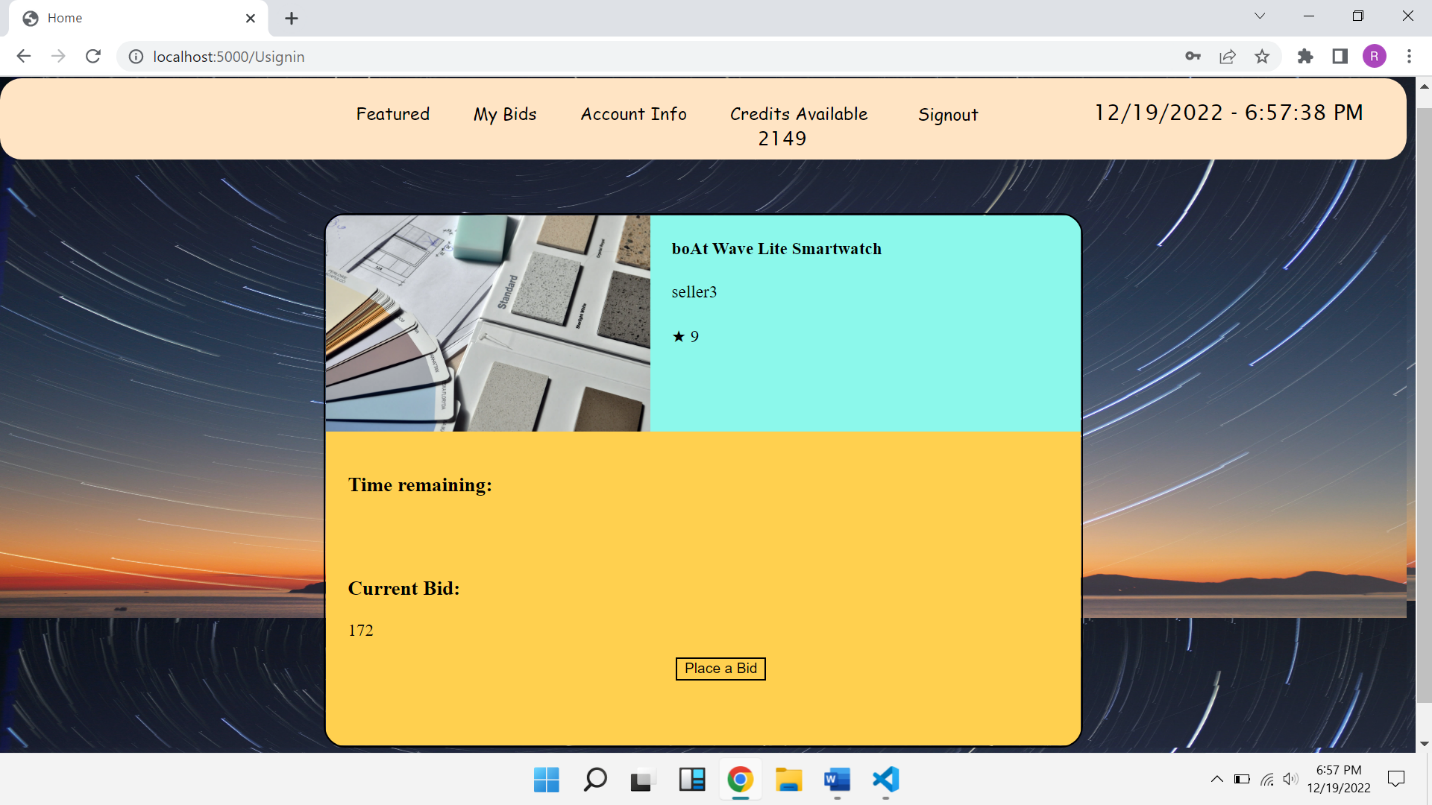
**Seller Login page:**

****

**Figure5**

* Here the navigation bar consists of
* My Bids
* AccountInfo
* Credits Available
* Signout
* The Bids of the seller are display using a div element to display the **Bid title, time remaining**

**Full Bid page:**

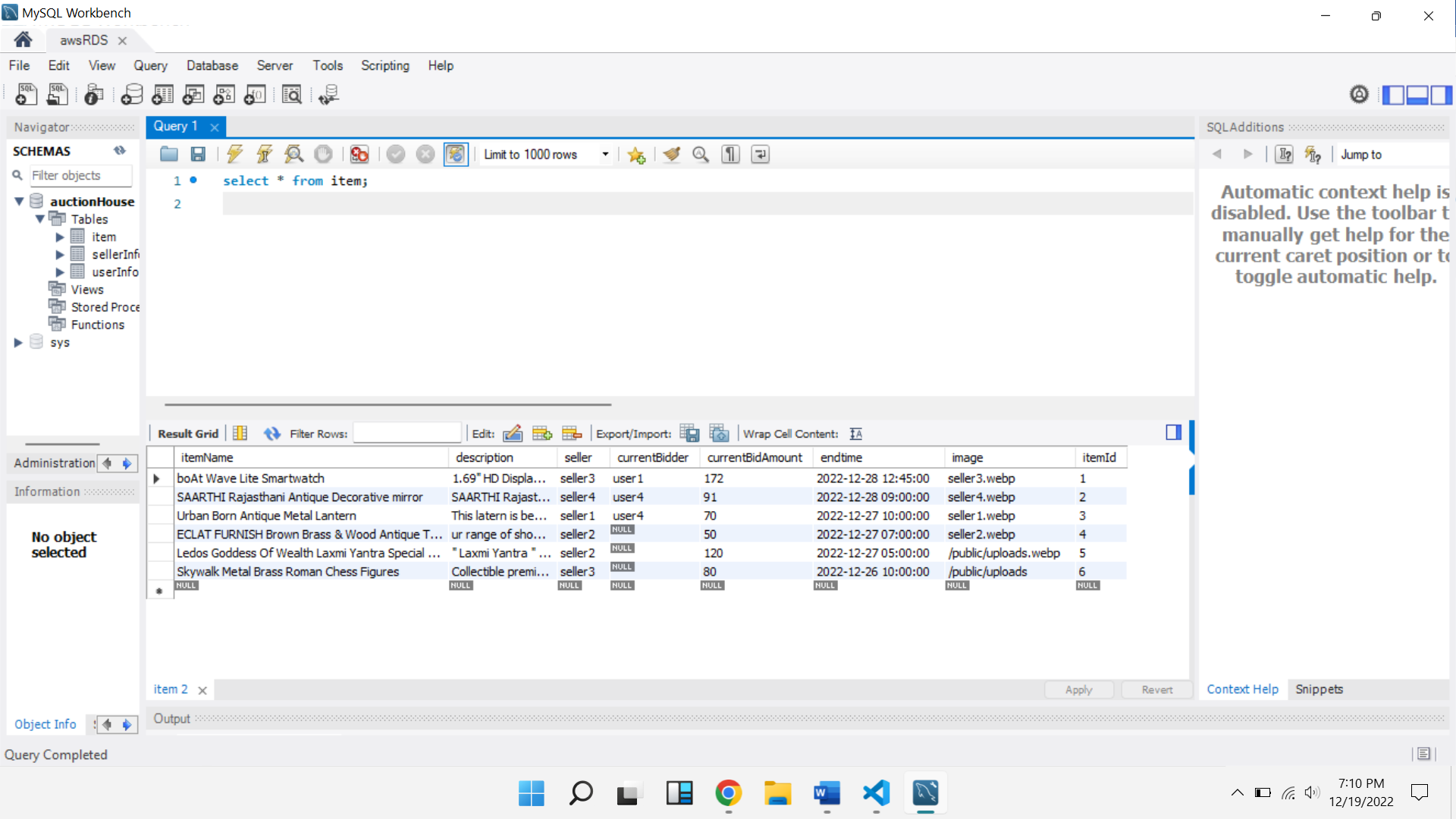
****

**Figure6**

* The Full Bid page consists of a **<div>** element which has
  + Image container for storing image
  + Title field
  + Time field
  + Bid field ,which is used to invoke functions related to Bidding

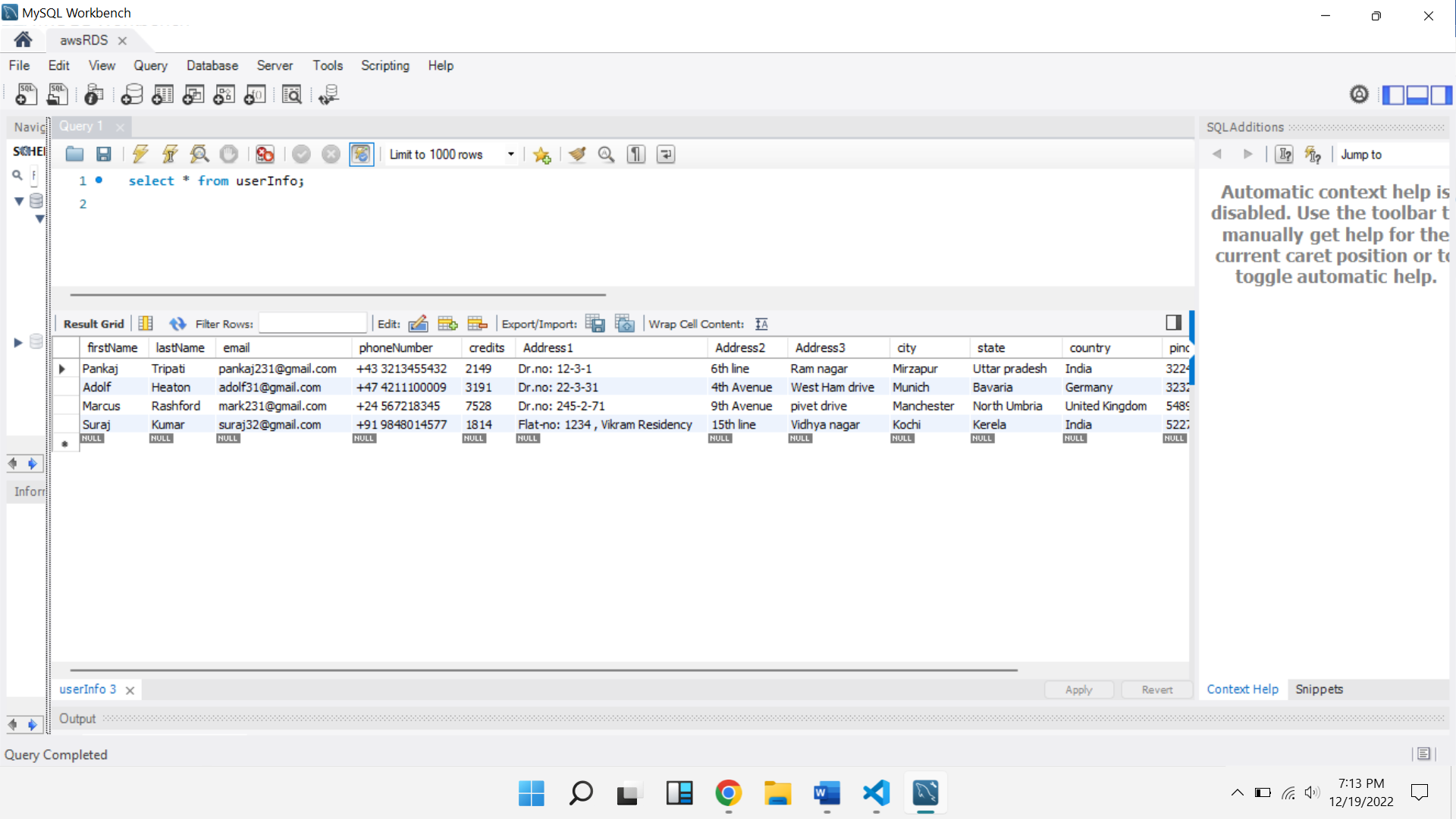
**Database:**

**Table1: bid items:**

****

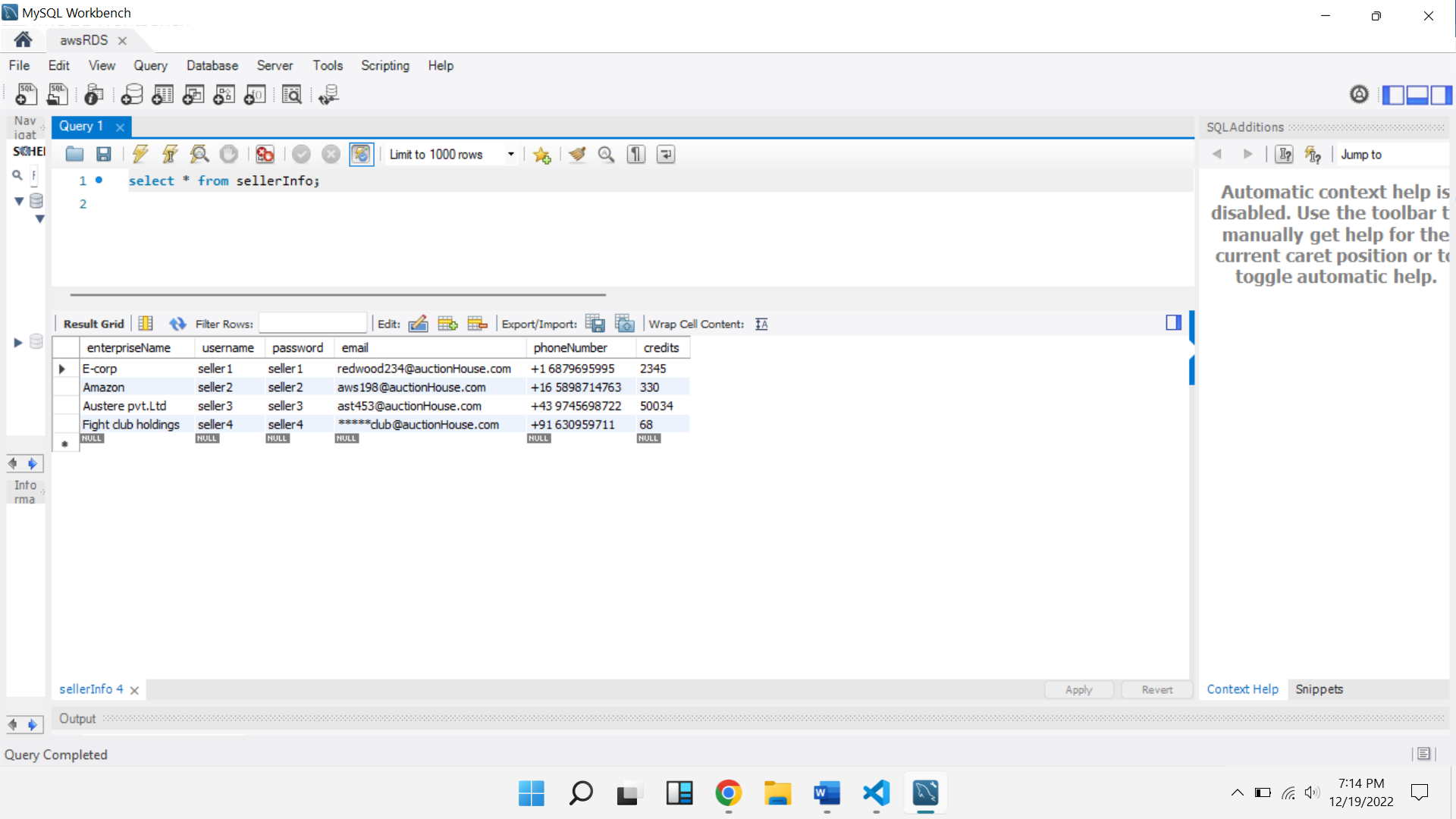
**Figure7**

**Table2: UserInfo:**

****

**Figure8**

**Table3: sellerInfo:**

****

**Figure9**

**Basic Routing used in Server side**

**App.post(‘/’) :** used to display index page on starting the server

**App.post(‘/userSignIn’):** used to validate the data taken from the userlogin form

**App.post(‘/sellerSignin’):** used to validate the data taken from the seller login form

**App.post(‘/bidPlace’):** used to validate the user credits and if successful bids an item in the favour of the user logged in

**App.post(‘/signOut’):** redirects the user back to login page/index page

**CHAPTER 3**

**COST ANALYSIS**

**3.1 List of components and their cost**

The costs of the various components used in this project are given below in Table 3.1.

**Figure10**

|  |  |
| --- | --- |
| **COMPONENT** | **Pricing/month** |
| AWS ec2 instances | ₹ 760 |
| AWS database instance(RDS) | ₹ 270 |

**CHAPTER 4**

**APPENDIX**

**Nodejs Code**

const { application, request } = require('express');

const mysql=require('mysql');

const express=require('express');

const ejs = require('ejs');

const multer=require('multer');

const app=express();

app.set('view engine', 'ejs');

app.set('views','./views');

app.use(express.static('public'));

app.use(express.json());

var bodyParser = require('body-parser');

const { render } = require('pug');

let alert=require('alert');

let window=require('window');

app.use(bodyParser.json());

app.use(bodyParser.urlencoded({extend:false}));

//database connection

const db = mysql.createConnection({

    host     : "auctiondatabase.c5xfieorp2bw.ap-south-1.rds.amazonaws.com" ,

    user     : "rohith26721",

    password : "Rohith1290",

    port     : "3306",

    database : "auctionHouse"

  });

  db.connect(function(err){

    console.log("Connected!");

    });

app.listen(5000);

//PAGE ROUTES....

let uname;

let sname;

let EnameSeller;

let itemId;

let path;

let iName;

let iDescription;

let iSeller;

let iImage;

let iTime;

 let iName1;

let iSeller1;

let iTime1;

let iName2;

let iSeller2;

let iTime2;

let iName3;

let iSeller3;

let iTime3;

let iName4;

let iSeller4;

let iTime4;

let iAmount1;

let iAmount2;

let iAmount3;

let iAmount4;

let creditsUser;

let iUser1;

let iUser2;

let iUser3;

let iUser4;

let cBidname;

let cSeller;

let cEndTime;

app.get('/',function(req,res){

    res.render("index.ejs");

});

// user signin route....

// variable to take in userlogin values at the time of login request

app.post('/Usignin',function(req,res){

    uname=req.body.UserName;

    var pword=(req.body.PassWord);

    var firstnameUser;

    var lastnameUser;

    var emailUser;

    var phonenumberUser;

    var address1User;

    var address2User;

    var address3User;

    var cityUser;

    var stateUser;

    var countryUser;

    var pincodeUser;

    var usernameUser;

    var passwordUser;

    db.query("select \* from userInfo where username = ? ",[uname],function(err,results,fields){

    firstnameUser=results[0].firstName;

    lastnameUser=results[0].lastName;

    emailUser=results[0].email;

    phonenumberUser=(results[0].phoneNumber);

    creditsUser=results[0].credits;

    address1User=results[0].Address1;

    address2User=results[0].Address2;

    address3User=results[0].Address3;

    cityUser=results[0].city;

    stateUser=results[0].state;

    countryUser=results[0].country;

    pincodeUser=results[0].pincode;

    creditsUser=results[0].credits;

    });

    db.query("select \* from item",function(err,result,fields){

      if(err)

      console.log("value not updated");

      iName1=result[0].itemName;

      iSeller1=result[0].seller;

      iTime1=result[0].endtime;

      iUser1=result[0].currentBidder;

      iAmount1=result[0].currentBidAmount;

      iName2=result[1].itemName;

      iSeller2=result[1].seller;

      iTime2=result[1].endtime;

      iUser2=result[1].currentBidder;

      iAmount2=result[1].currentBidAmount;

      iName3=result[2].itemName;

      iSeller3=result[2].seller;

      iUser3=result[2].currentBidder;

      iTime3=result[2].endtime;

      iAmount3=result[2].currentBidAmount;

      iName4=result[3].itemName;

      iSeller4=result[3].seller;

      iUser4=result[3].currentBidder;

      iTime4=result[3].endtime;

      iAmount4=result[3].currentBidAmount;

     });

    db.query("select \* from userInfo where username = ? and password = ?",[uname,pword], function(err, result, fields) {

      if(err){

        console.log("error evaded");

        res.redirect('/');

      }

      if(result.length>0){

         res.render("UserLogin.ejs",

         {"firstname" : firstnameUser ,

         "lastname" : lastnameUser,

         "email" : emailUser,

          "phonenumber" :phonenumberUser,

          "credits" :  creditsUser,

          "address1" : address1User,

          "address2" : address2User,

          "address3" : address3User,

          "city" :     cityUser,

          "state" : stateUser,

          "country" : countryUser,

          "pincode" : pincodeUser,

          "username" : uname,

          "password" : pword,

          "credits" : creditsUser,

          "name1":iName1,

          "seller1":iSeller1,

          "time1":iTime1,

          "name2":iName2,

          "seller2":iSeller2,

          "time2":iTime2,

          "name3":iName3,

          "seller3":iSeller3,

          "time3":iTime3,

          "name4":iName4,

          "seller4":iSeller4,

          "time4":iTime4,

          "amount1":iAmount1,

          "amount2":iAmount2,

          "amount3":iAmount3,

          "amount4":iAmount4,

        });

         console.log("user logged in successfully");

        }

         else

         res.sendStatus(404);

         res.end();

       });

});

// Seller signin route ...

app.post('/Ssignin',function(req,res){

    sname=req.body.Sid;

    let spword=(req.body.SPwd);

    db.query("select \* from sellerInfo where username = ? ",[sname],function(err,results,fields){

      EnameSeller=results[0].enterpriseName;

      emailSeller=results[0].email;

      phonenumberSeller=(results[0].phoneNumber);

      creditsSeller=results[0].credits;

      });

      db.query("select \* from item where seller = ? ",[sname],function(err,result,fields){

         iName=result[0].itemName;

         iDescription=result[0].description;

         iSeller=result[0].seller;

         iImage=result[0].image;

         iTime=result[0].endtime;

        });

    db.query("select \* from sellerInfo where username = ? and password = ?",[sname,spword], function(err, result, fields) {

         if(result.length>0){

          res.render("sellerLogin.ejs",

         {"enterpriseName" : EnameSeller ,

         "email" : emailSeller,

          "phonenumber" :phonenumberSeller,

          "credits" :  creditsSeller,

          "username" : sname,

          "password" : spword,

          "itemName" :iName,

          "description": iDescription,

          "seller": iSeller,

          "image" : iImage,

          "time" : iTime

        });

         console.log("seller logged in successfully");

        }

         else

         res.sendStatus(404);

       });

});

app.post('/item1Bid',function(req,res){

  let bid=req.body.item1BidAmount;

  let prevUser;

  if(bid>iAmount1&&creditsUser>=bid){

    creditsUser=creditsUser-bid;

    db.query("UPDATE userInfo SET credits = ? WHERE (username = ?);",[iAmount1,iUser1]);

    db.query("UPDATE item SET currentBidder = ?, currentBidAmount = ? WHERE (itemId = '1');",[uname,bid]);

    db.query("UPDATE userInfo SET credits = ? WHERE (username = ?);",[creditsUser,uname]);

    alert("congratulations!! your bid is successful");

  }

  else{

    alert("Enter the correct bid Amount");

    res.redirect('/Usignin');

  }

});

app.post('/item2Bid',function(req,res){

  let bid=req.body.item2BidAmount;

  if(bid>iAmount2&&creditsUser>=bid){

    creditsUser=creditsUser-bid;

    db.query("UPDATE userInfo SET credits = ? WHERE (username = ?);",[iAmount2,iUser2]);

    db.query("UPDATE item SET currentBidder = ?, currentBidAmount = ? WHERE (itemId = '2');",[uname,bid]);

    db.query("UPDATE userInfo SET credits = ? WHERE (username = ?);",[creditsUser,uname]);

    alert("congratulations!! your bid is successful");

  }

  else{

    alert("Enter the correct bid Amount");

  }

});

app.post('/item3Bid',function(req,res){

  let bid=req.body.item3BidAmount;

  if(bid>iAmount3&&creditsUser>=bid){

    creditsUser=creditsUser-bid;

    db.query("UPDATE userInfo SET credits = ? WHERE (username = ?);",[iAmount3,iUser3]);

    db.query("UPDATE item SET currentBidder = ?, currentBidAmount = ? WHERE (itemId = '3');",[uname,bid]);

    db.query("UPDATE userInfo SET credits = ? WHERE (username = ?);",[creditsUser,uname]);

    alert("congratulations!! your bid is successful");

  }

  else{

    alert("Enter the correct bid Amount");

  }

})

app.post('/item4Bid',function(req,res){

  let bid=req.body.item4BidAmount;

  if(bid>iAmount4&&creditsUser>=bid){

    creditsUser=creditsUser-bid;

    db.query("UPDATE userInfo SET credits = ? WHERE (username = ?);",[iAmount4,iUser4]);

    db.query("UPDATE item SET currentBidder = ?, currentBidAmount = ? WHERE (itemId = '4');",[uname,bid]);

    db.query("UPDATE userInfo SET credits = ? WHERE (username = ?);",[creditsUser,uname]);

    alert("congratulations!! your bid is successful");

  }

  else{

    alert("Enter the correct bid Amount");

  }

})

app.get('/signout',function(req,res){

    res.render("index.ejs");

})

app.post('/addBid',function(req,res){

  var newBidName=req.body.newBidName;

  var newBidId=req.body.newBidId;

  var newBidDescription=req.body.newBidDescription;

  const newBidT=req.body.newBidTime;

  var newBidAmount=req.body.newBidAmount;

  var imageLocation='/public/uploads';

  db.query("insert into item values(?,?,?,'NULL','0',?,?,?);",[newBidName,newBidDescription,sname,newBidT,imageLocation,newBidId],function(err,results,fields){

    });

});

**END OF THE REPORT**