



Auction Dapp Part2

We completed our contract development part with truffle. Next we need to integrate this contract development with web development part

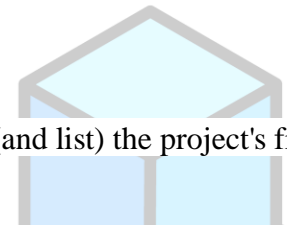
Setting up the development environment with Express

We have already create project directory,change to the root of the directory where the project is located and then type the following into a terminal :

```
$ express --view=ejs
```

The express generator will create (and list) the project's files.

Output



```
karthika@karthika-Lenovo-ideapad-320-15IKB:~/Desktop/CED_AuctionDapp$ express -e
warning: option '--ejs' has been renamed to '--view=ejs'

create : public/
create : public/javascripts/
create : public/images/
create : public/stylesheets/
create : public/stylesheets/style.css
create : routes/
create : routes/index.js
create : routes/users.js
create : views/
create : views/error.ejs
create : views/index.ejs
create : app.js
create : package.json
create : bin/
create : bin/www

install dependencies:
$ npm install

run the app:
$ DEBUG=ced-auctiondapp:* npm start

karthika@karthika-Lenovo-ideapad-320-15IKB:~/Desktop/CED_AuctionDapp$
```



Auction Dapp Part2

Directory structure

The project directory contains the following files and folders.

- ★ **app.js**: routing specification.
- ★ **bin/**: contains server configuration file.
- ★ **node_modules/**: node packages needed for the project are stored here.
- ★ **package.json**: lists the packages used for the projects and any associated scripts.
- ★ **package-lock.json**: lists out the exact version and source of the packages used for the project.
- ★ **public/**: js script, picture, css (can be accessed without server).
- ★ **routes/**: routes js files are stored here.
- ★ **views/**: contains UI files

Configure truffle with express

We have our contract development with truffle framework, so we need to configure this part with our web app, for that change app.js file with following manner

```
port web3 library
Web3 = require('web3');

port compiled output of auction contract from build directory
MyContractJSON = require(path.join(__dirname,
'build/contracts/Auction.json'));

//Establish connection with local geth private chain

=new Web3(new
Web3.providers.HttpProvider("http://localhost:8545"));

et contract address, from network id 4002 (here 4002 is the network
id of geth private chain)
contractAddress = MyContractJSON.networks['4002'].address;

int Contract Address
ble.log(contractAddress);

et abi
t abi = MyContractJSON.abi;

reating contract object
ntract = new web3.eth.Contract(abi, contractAddress);
```



Create express based Web App

- First we need to create routes for a get auction details in a module named **getAuctionDetails.js**. The code first imports the Express application object, uses it to get a Router object and then Last of all the module exports the Router object.

```
var express = require('express');
var router = express.Router();
// getting data from blockchain
// and rendering it to the auction view
router.get('/', function (req, res, next) {
  //For getting accounts from local geth private chain
  web3.eth.getAccounts().then((data1) => {
    console.log(data1);
    //call getAuctionDetails from deployed contract for getting
    auction details
    MyContract.methods.getAuctionDetails().call({ from: data1[0]
  }).then(function (data) {
    //call Mycar from deployed contract for getting car details
    MyContract.methods.Mycar().call({ from: data1[0]
  }).then(function (car) {
    MyContract.methods.auction_status().call({ from: data1[0]
  }).then(function (state) {
    if (state == 1)
      astatus = "Live"
    else if (state == 0)
      astatus = "Not Live"
    //data passing to the auction.ejs

    HighestBid = web3.utils.fromWei(data[1], 'ether')
    res.render("auction", { data: data, state: astatus, car: car,
      data1: data1, HighestBid: HighestBid });

  })
})

})
});

module.exports = router;
```



K B A

Auction Dapp Part2

- Next we'll do our templating inside of the views folder and the rest is pretty standard Node practices. Here **ejs** is our view engine, **ejs** is very easy to setup.
- Here we need an **auction.ejs** file for displaying auction details

```
<!DOCTYPE html>
<html>

<head>
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>KBA Auction Dapp on Ethereum</title>

  <link rel='stylesheet' href='/stylesheets/style.css' />
  <h1>Auction Dapp on Ethereum</h1>
  <p class="stylename1">A vehicle's owner deploys the contract
to the blockchain and becomes the auction owner. The
  auction is open immediately after the contract deployment,
  and once the bidding period is over, the highest bidder
  wins the auction, and the other participants withdraw their
bids.</p>
</head>

<body>
  <table>

    <td>

      <h2>Auction Details</h2>
      <br>
      <div id="details">
        <p><b>Car Brand</b> : <%= car[0] %></p>
        <br>
        <p><b>Car Registration Number</b> : <%= car[1] %></p>
        <br>
        <p><b>Current Car Owner</b> : <%= car[2] %></p>
        <br>
        <p><b>Auction Owner</b> : <%= data[3] %></p>
        <br>
        <p id="endingTime"><b>Auction End</b> : <%= data[0]
%></p>
        <br>
        <p><b>Highest Bid</b> : <%= HighestBid %> </p>

      <br>
      <p><b>Highest Bidder</b> : <%= data[2] %></p>
      <br>
      <p><b>Auction Status</b> : <%= state %></p>

    </td>

  </table>

</body>
</html>
```



Auction Dapp Part2

```
</tr>
</table>
<div class="footer">&copy; Copyright 2019, Kerala Blockchain
Academy</div>
</body>

</html>
```

- Here we need to add style.css file under public/stylesheets

```
body {
  background-image: url("https://images.unsplash.com/photo-
1508796079212-a4b83cbf734d?ixlib=rb-
1.2.1&ixid=eyJhcnBfaWQiOjE5MDd9&w=1000&q=80");
  background-attachment: fixed;
  border: 2px solid black;
  padding: 25px;
  /* Full height */
  height: 100%;
  /* Center and scale the image nicely */
  background-position: center;
  background-repeat: no-repeat;
  background-size: cover;

  background-color: #cccccc;
  font-family: 'Raleway', 'Source Sans Pro', 'Arial';
}
h1 {
  margin: 0em 0 0.5em 0;
  color: #291c4d;
  font-weight: bold;
  font-family: 'Ultra', sans-serif;
  font-size: 45px;
  font-style: italic;
  text-align: center;
  line-height: 42px;
  font-variant: capitalize;
  text-shadow: 0 2px white, 0 3px #777;
  border: 7px solid rgb(248, 244, 244);
  border-radius: 8px;
}

input[type=button], input[type=submit], input[type=reset] {
  background-color: rgb(170, 142, 168);
  border: none;
  color: black;
  padding: 12px 32px;
  font-size: 16px;
```



Auction Dapp Part2

```
font-weight: bold;
margin: 4px 2px;
cursor: pointer;
border: 2px solid rgb(15, 14, 14);
border-radius: 2px;
}
select {
padding: 5px;
margin: 0;
-webkit-border-radius: 8px;
-moz-border-radius: 8px;
border-radius: 8px;
-webkit-box-shadow: 0 3px 0 #ccc, 0 -1px #fff inset;
-moz-box-shadow: 0 3px 0 #ccc, 0 -1px #fff inset;
box-shadow: 0 3px 0 #ccc, 0 -1px #fff inset;
background: #f8f8f8;
color: rgb(19, 12, 12);
border: 2px solid black;
outline: none;
display: inline-block;
-webkit-appearance: none;
-moz-appearance: none;
appearance: none;
cursor: pointer;
font-weight: bold;
font-size: 15px;
}
p {
font-weight: bold;
font-family: 'Ultra', sans-serif;
font-size: 16px;
font-style: italic;
font-variant: capitalize;
}
h2 {
margin: 1em 0 0.5em 0;
color: #ffffff;
font-weight: bold;
font-family: 'Ultra', sans-serif;
font-size: 25px;
font-style: italic;
text-align: center;
line-height: 42px;
font-variant: capitalize;
text-shadow: 0 2px white, 0 3px #777;
border: 7px solid rgb(31, 29, 29);
border-radius: 8px;
background-color: rgb(85, 180, 197);
}
.stylenamel {
font-family: Georgia, "Times New Roman", Times, serif;
font-size: 30px;
font-style: italic;
```



Auction Dapp Part2

```
font-weight: bold;
font-variant: capitalize;
color: rgb(75, 21, 43);
background-color: rgb(238, 231, 231);
border: 8px solid rgba(140, 145, 190, 0.911);
border-radius: 8px;
}
.stylename {
font-family: Georgia, "Times New Roman", Times, serif;
font-size: 25px;
font-style: italic;
font-weight: bold;
font-variant: small-caps;
color: rgb(10, 5, 19);
}
td:nth-of-type(1) {
padding-right: 400px;
}
input[type=text] {
width: 30%;
padding: 12px 20px;
margin: 8px 0;
box-sizing: border-box;
}
```

- Next step is to add the routes to the middleware chain. We do this in app.js.

```
//Then we require() modules from our routes directory.
//These modules/files contain code for handling particular sets
of related "routes" (URL paths)
var auctionRouter = require('./routes/getAuctionDetails');
```

- Next we need to add our (previously imported) route-handling code to the request handling chain

```
//we add our (previously imported) route-handling code to the
request handling chain
app.use('/', auctionRouter);
```



Auction Dapp Part2

-

Install Project Dependencies

Install all project dependencies using following command

```
$ npm install
$ npm install web3@1.0.0-beta.48 --save
```

Run Project

Make sure that your ethereum client, geth running on background, Set your auction deployment details in **2_deploy_auction.js**, set coinbase as auction owner address(in migrations folder) and run your migration again with following command

```
$ truffle migrate --reset
```

for running our web application use following command

```
$ npm start
```

Output

```
karthika@karthika-Lenovo-ideapad-320-15IKB:~/CED_Auction_Dapp$ npm start
> nodejs@0.0.0 start /home/karthika/CED_Auction_Dapp
> node ./bin/www
```

Navigate to <http://localhost:3000/>



K B A

Auction Dapp Part2

← → localhost:3000 ☆

Auction Dapp on Ethereum

A vehicle's owner deploys the contract to the blockchain and becomes the auction owner. The auction is open immediately after the contract deployment, and once the bidding period is over, the highest bidder wins the auction, and the other participants withdraw their bids.

Auction Details

Car Brand : BMW

Car Registration Number : RN00091

Auction Owner : 0x3f13B0B277013D83a672929318674DF31AAAb43C7

Auction Start : 1553581821

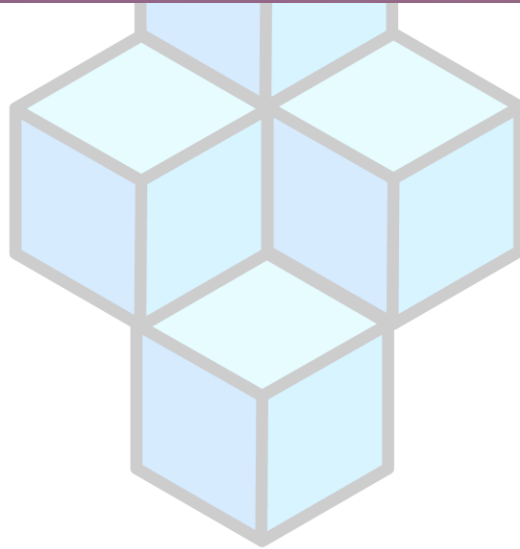
Auction End : 1553582121

Highest Bid : 0

Highest Bidder : 0x00

Auction Status : Live

© Copyright 2019, Kerala Blockchain Academy



K

B

A