

Bhartiya Vidya Bhavan’s

Sardar Patel Institute of Technology, Mumbai-400058

Department of Computer Science and Engineering

**OEIT1:Blockchain Technology and Applications**

**Lab5A: Ethereum Blockchain and DApp**

**Building an auction DApp (Do It Yourself)**

**Name of Student: Adwait Purao**

**UCID: 2021300101**

**Branch: COMPS B**

**Objective: Building an auction DApp**

**Outcomes:** After successful completion of lab students should be able to

Implement an Ethereum private blockchain and an auction DApp

Write a smart contract using Solidity Language

Compile and run the DApp on Ethereum Blockchain

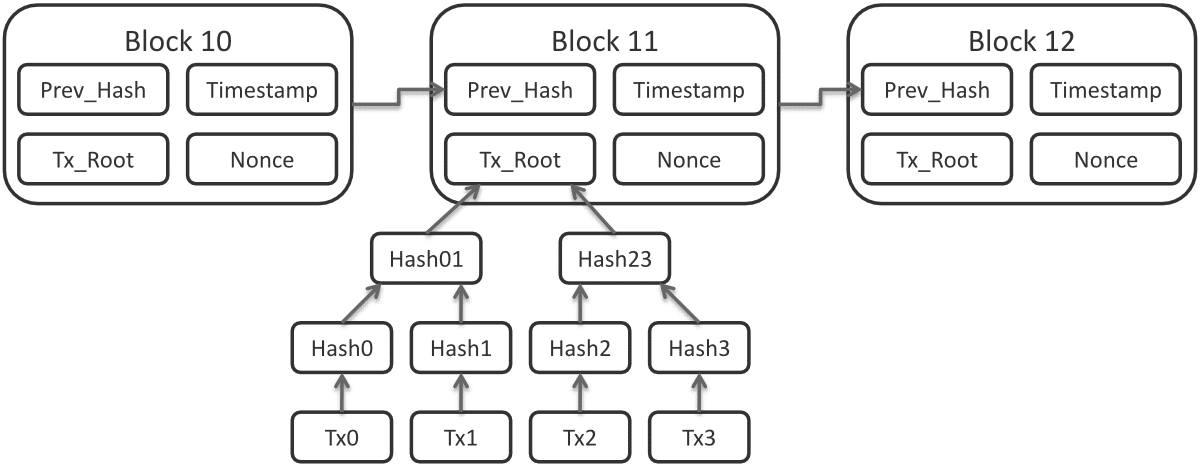
Use Web3/REST API

**System Requirements:**

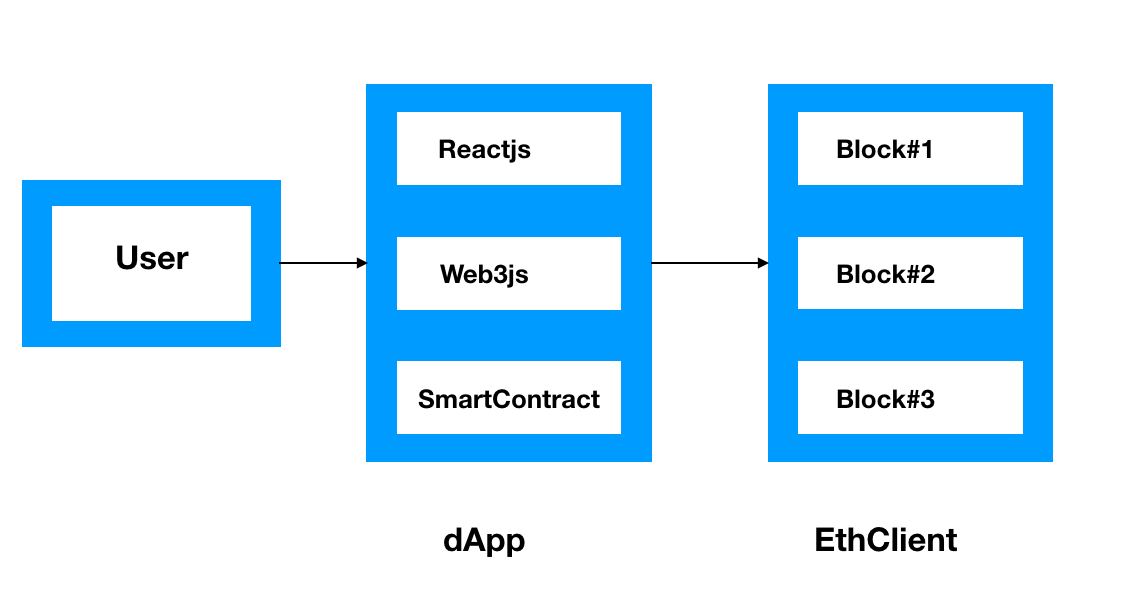
PC (C2D, 8GB RAM, 100GB HDD space and NIC),Ubuntu Linux 14.04/20.04

Internet connectivity,Python Cryptography and Pycrypto,Nodejs, Truffle,Ganache-cli

, solidity,REST API



**Figure-1: Blockchain Implementation**



**Figure-2: Ethereum and DApp**

Procedure:

Stage-0: **Create an Ethereum Dapp with React and Docker by Shubham Chadokar**

Read this article carefully.

<https://medium.com/hackernoon/create-an-ethereum-dapp-with-react-and-docker-211223005f17>

Step-1: Install docker and docker-compose

$sudo apt-get install docker.io docker-compose

Step-2: Check for docker images

$sudo docker images

Step-3: Check for running container

$sudo docker ps

Step-4: Clone the project using this GitHub link

$mkdir lab5a

$git clone https://github.com/schadokar/docker-ethereum.git

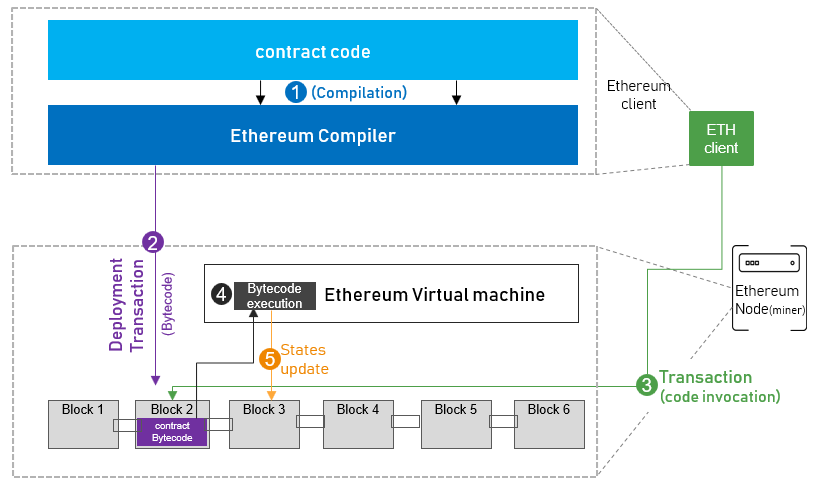
Step-5: Build the DApp project

$sudo docker-compose build

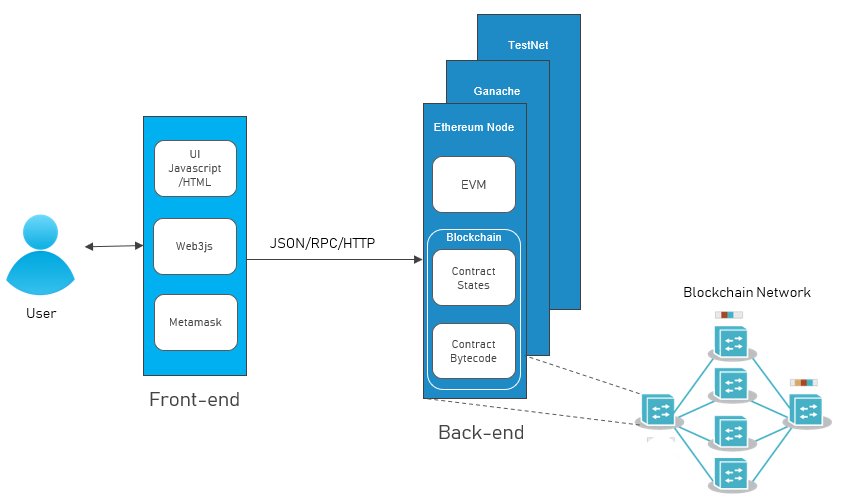
Follow the instructions as per the article.

Take screenshots and complete it.

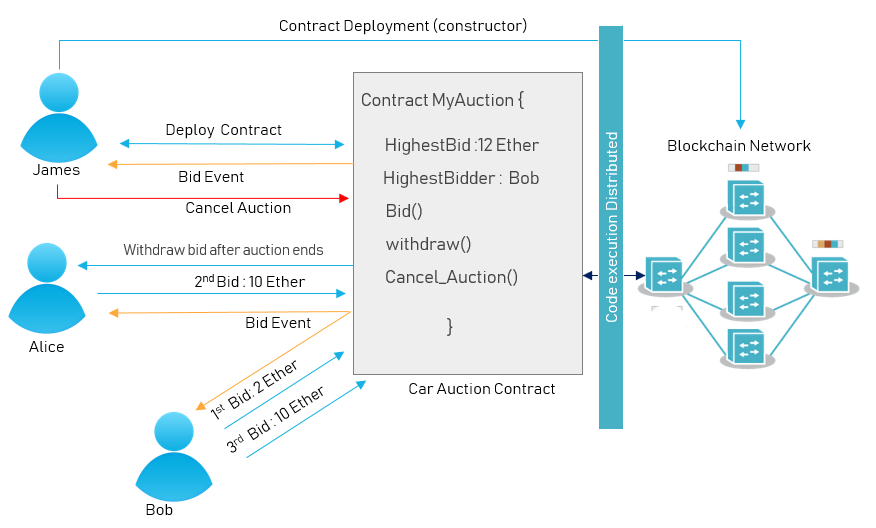
Stage-1:



**Figure-3: Ethereum virtual machine and smart contracts**



**Figure-4: DApp architecture**



**Figure-5: Auction Contract in Solidity**

**Procedure for stage-1:**

Refer to Peer-to-Peer Auctions in Ethereum [1], chapter 4 of Blockchain by Example online free learning edition by Packt Publications.

Auction DApp summary:

DApp is a web application that enables users to start in auctions using ether.

* Write (in Solidity) and compile the auction contract

Smart Contract auction.sol:

pragma solidity ^0.4.24;

contract Auction {

address internal auction\_owner; uint256 public auction\_start; uint256 public auction\_end; uint256 public highestBid; address public highestBidder;

enum auction\_state { CANCELLED, STARTED

}

struct car { string Brand;

string Rnumber;

}

car public Mycar; address[] bidders;

mapping(address => uint) public bids; auction\_state public STATE;

modifier an\_ongoing\_auction() { require(now <= auction\_end);

\_;

}

modifier only\_owner() { require(msg.sender == auction\_owner);

\_;

}

function bid() public payable returns (bool) {} function withdraw() public returns (bool) {} function cancel\_auction() external returns (bool) {}

event BidEvent(address indexed highestBidder, uint256 highestBid); event WithdrawalEvent(address withdrawer, uint256 amount);

event CanceledEvent(string message, uint256 time);

}

* Interact with your contract through a web page
* Deploy our smart contract on different environments and set up a local blockchain

Accounts generated using Ganache CLI

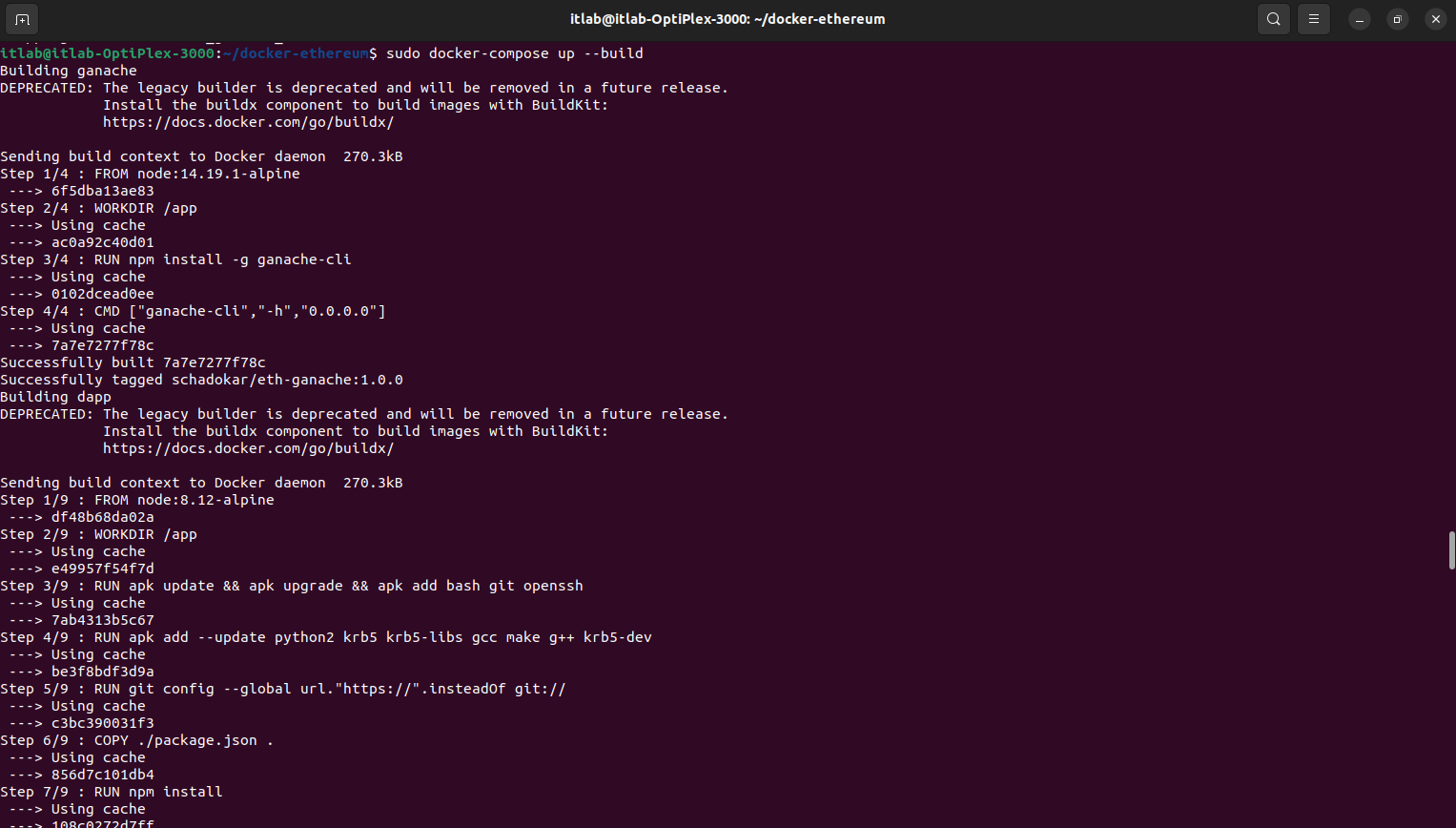
Set compiler to older version.

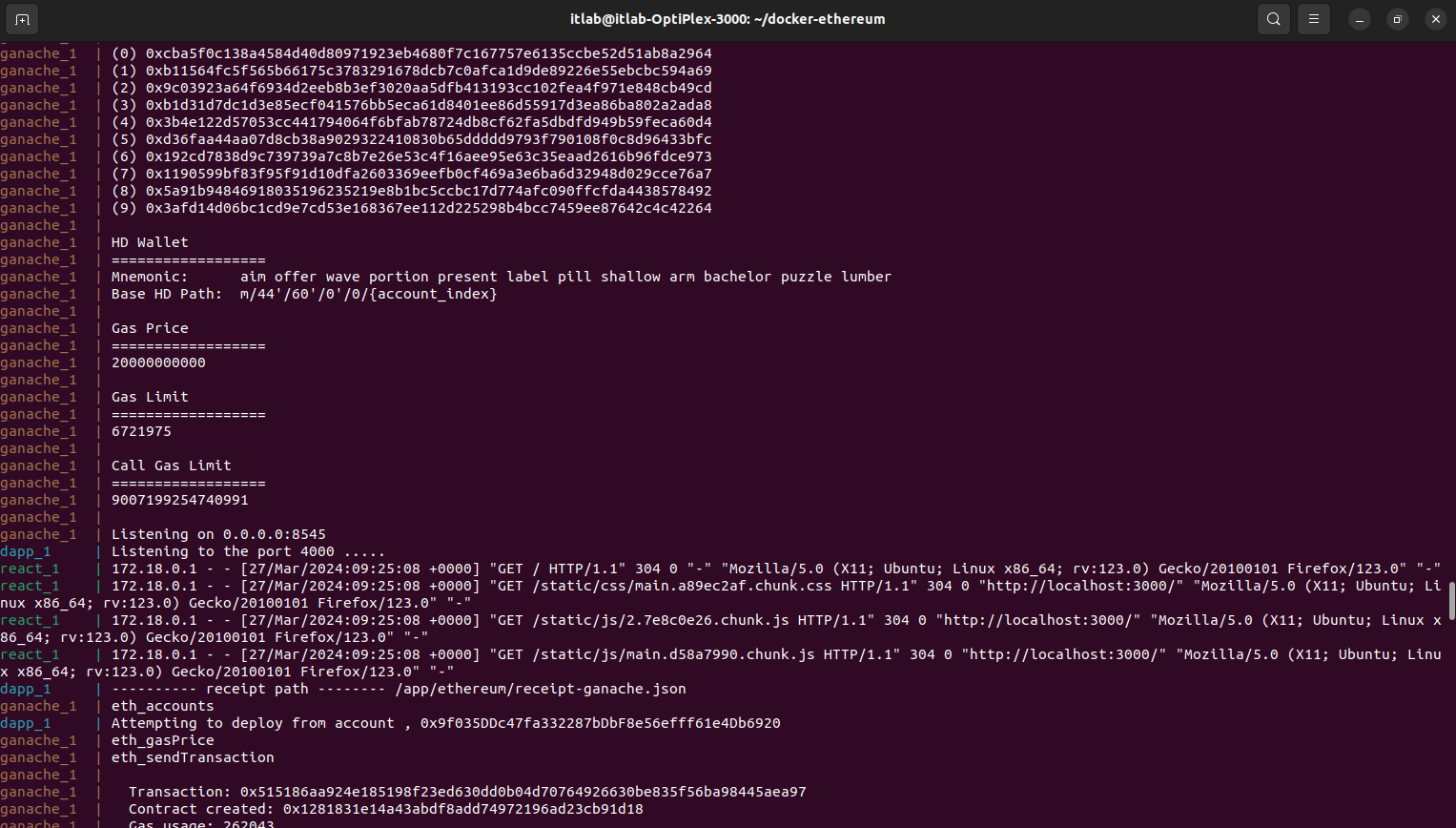
Remix Environment set to Web3 Provider.

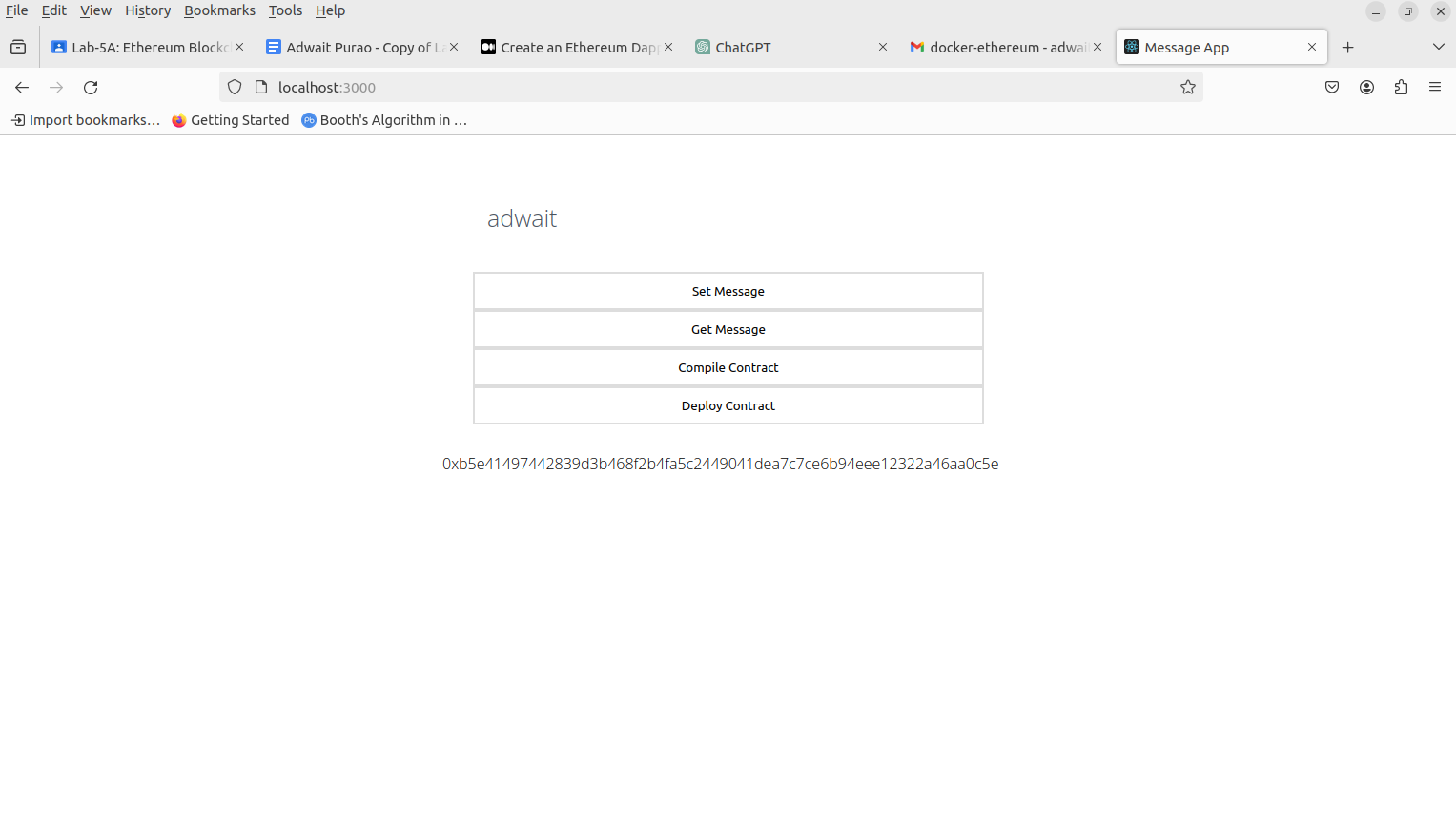
Deploy Smart Contract

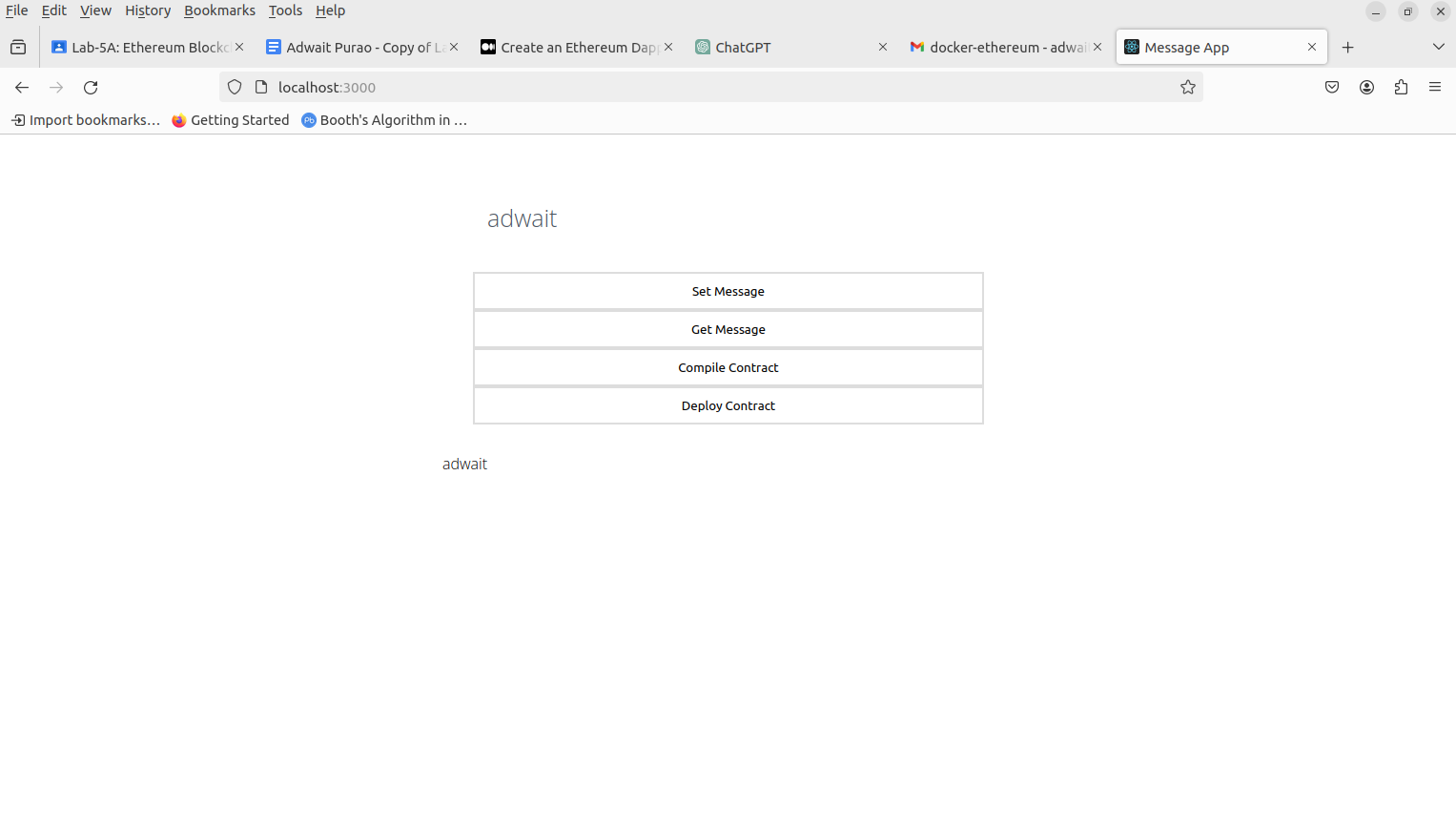
Set bid value and click bid button

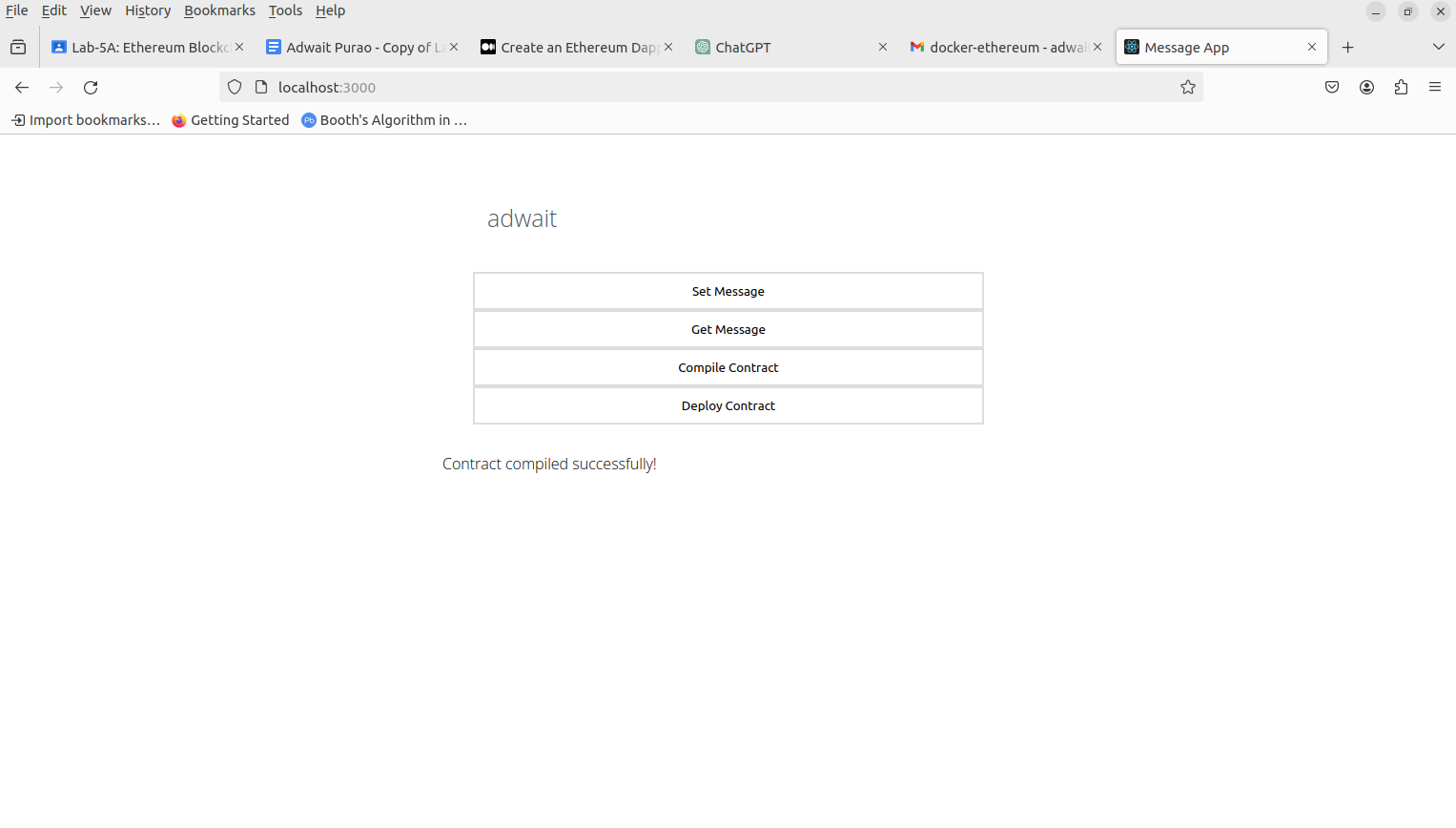
**Output:**

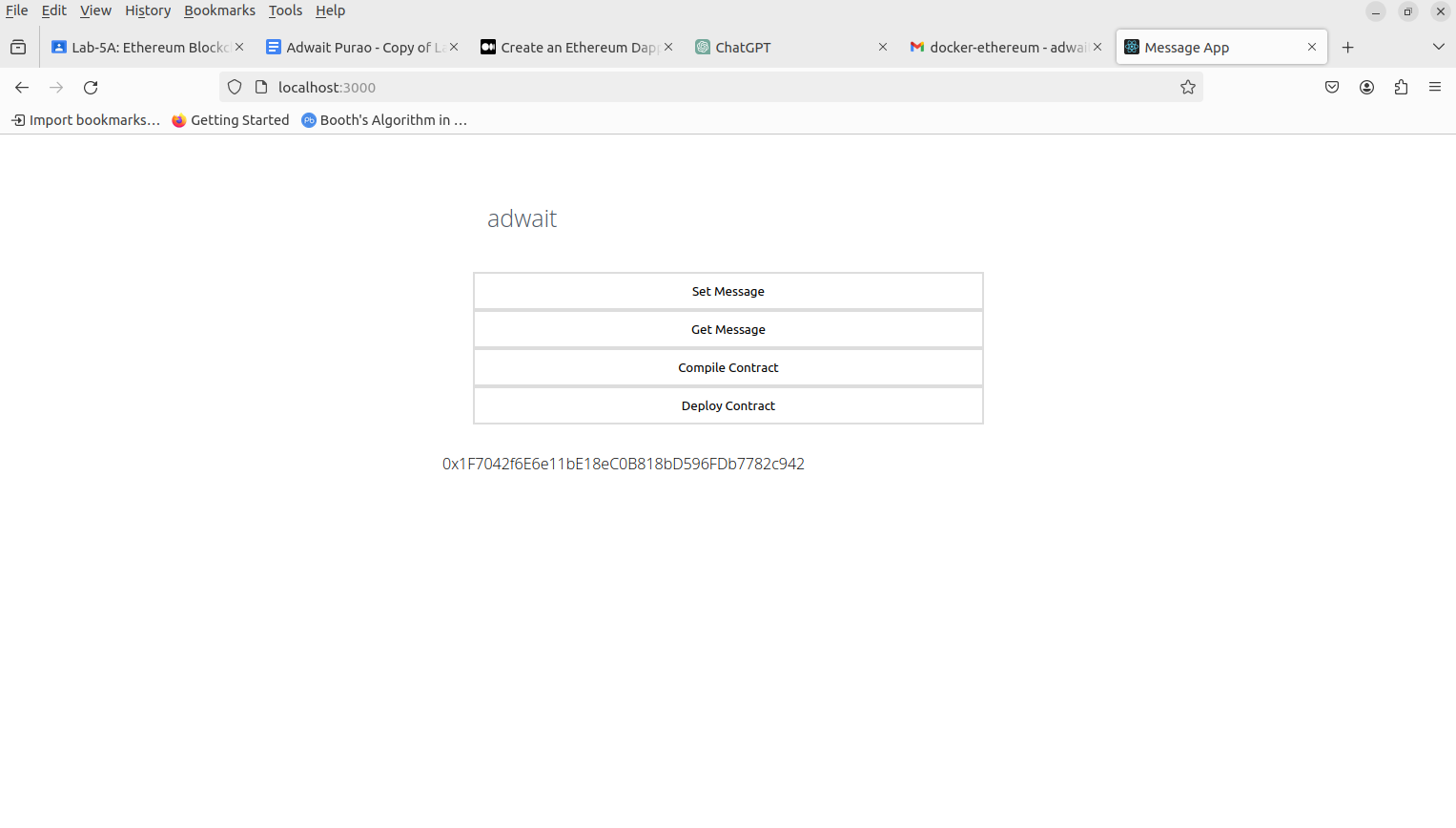
****











**Conclusion:**

Hence, we conclude that the provided Auction smart contract offers a robust framework for conducting peer-to-peer auctions on the Ethereum blockchain. With features for initializing auctions, managing states, facilitating bidding, withdrawal, and cancellation, along with the use of modifiers and events for contract integrity and transparency, it presents a comprehensive solution for decentralized auction applications. By enabling interaction through web interfaces, the contract ensures accessibility and usability for users, thus serving as a foundational component for diverse auction-based decentralized applications.

**References:**

[1] Create an Ethereum Dapp with React and Docker <https://medium.com/hackernoon/create-an-ethereum-dapp-with-react-and-docker-211223005f17>

[2] Peer-to-Peer Auctions in Ethereum

https://subscription.packtpub.com/book/big-data-and-business-intelligence/9781788475686/4

[3] How To Install Node.js on Ubuntu 20.04

[How To Install Node.js on Ubuntu 20.04 | DigitalOcean](https://www.digitalocean.com/community/tutorials/how-to-install-node-js-on-ubuntu-20-04)