BCDV - 1011
DESIGN PATTERNS FOR BLOCKCHAIN

Team Members

Feiya Wang	101036160
Muzammil	101281406
Neraj Obla KumarBabu	101275194
Patterson Thamba	101280376
Vidyavathy Venkataseshadri	101266772

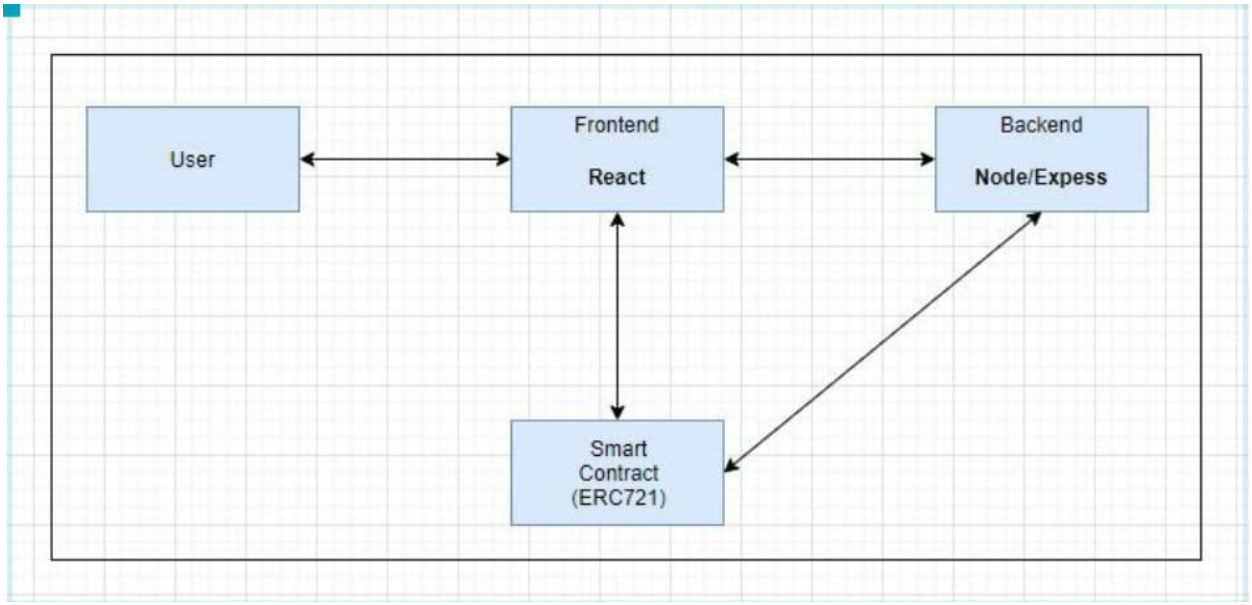
Architectural Documentation for Arts Provenance Project

Digital Art Platform:

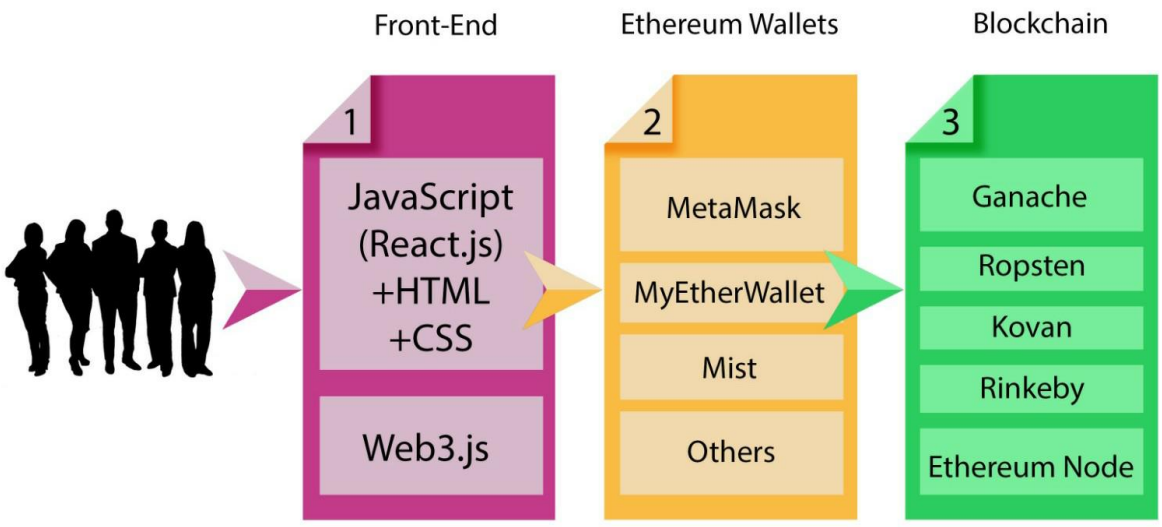
The decentralized digital art platform provides the artist with a place to sell their arts, and thereby a buyer finds a marketplace. It has the following features:

- It allows artists to easily sell their product without a third party, with no hidden fee during the transaction
- All art trading history is available in the blockchain; it is transparent
- The user can browse digital arts in the art gallery
- Each piece of art shows its product information
- The product owner can resell the art with the desired price
- Once a deal has been completed, the ownership of the art will be transferred, and the buyer will pay the seller for the art

Technical Architecture:



The web app is built with reactJS and web3 while the backend is built using NodeJS and Express.The smart contract is deployed using Truffle and Ganache.The smart contract functionalities have been tested using truffle suite. The generated abi is stored in the backend. ReactJS uses the abi from backend to interact with the smart contract. Web3 module is used in the react app to call the functionalities of the smart contract.



The above figure shows the techstack visual representation of the project.The frontend uses ReactJS along with HTML and CSS for building the app and web3 for interacting with the smart contract functionalities. Next phase would be the Ethereum wallets such as Metamask and MyEtherWallet. A Metamask browser extension is highly recommended for using this app. The smart contract is tested compatible to be deployable in most of the test nets some prominent ones being Ropsten and Rinkby. The entire app was built and tested using the local blockchain.

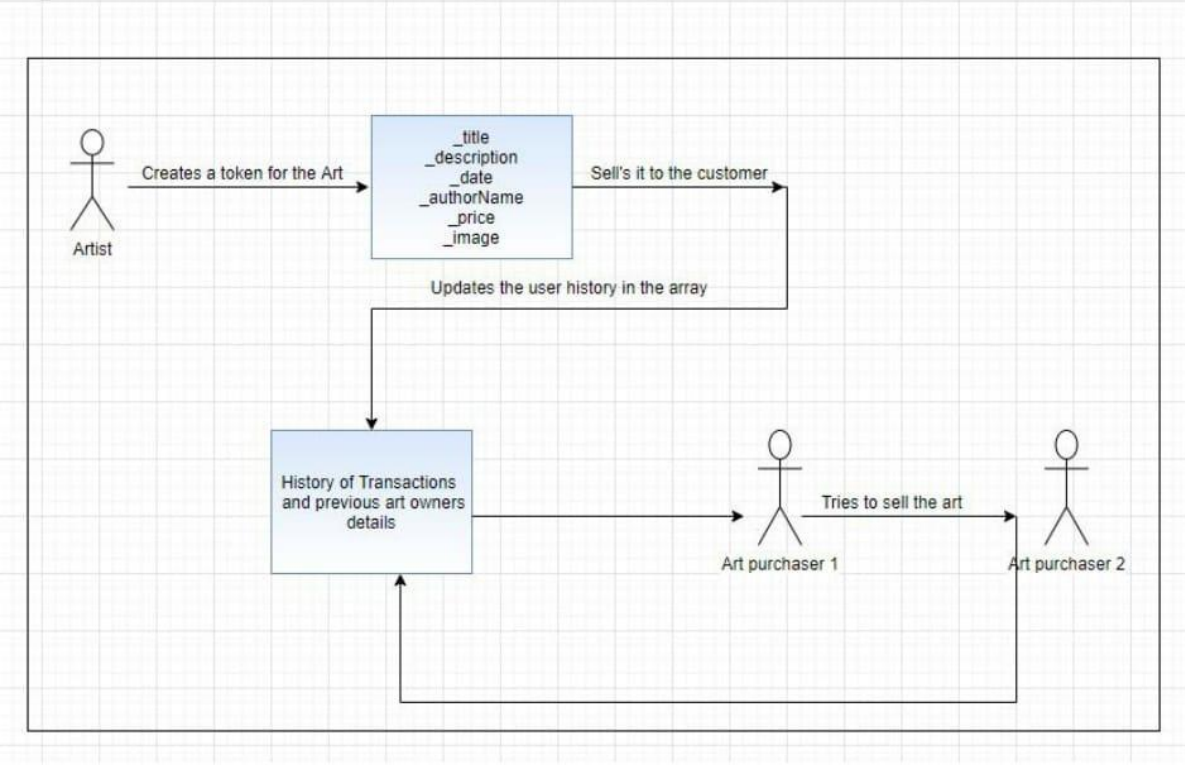
Data

Name	**Type**	**Structure**	**Purpose**
Name, Symbol	String	n/a	Token Information
ownedTokensCount	mapping	(address => uint)	No. of tokens owned by owner
tokenOwner	mapping	(uint => address)	TokenID corresponding to owner
operatorApproval	mapping	(address => mapping (address => bool)	Enable or disable third party (operator) to manage assets
Art	struct	{ id, title, description, date, price, author, owner, status etc }	Each art has its own token attributes.
ArtTxn	struct	{id,price,seller,buyer,txnDate,status}	Keeps the information record of transaction history
status	Struct field	0 or 1	For sale (or) sold

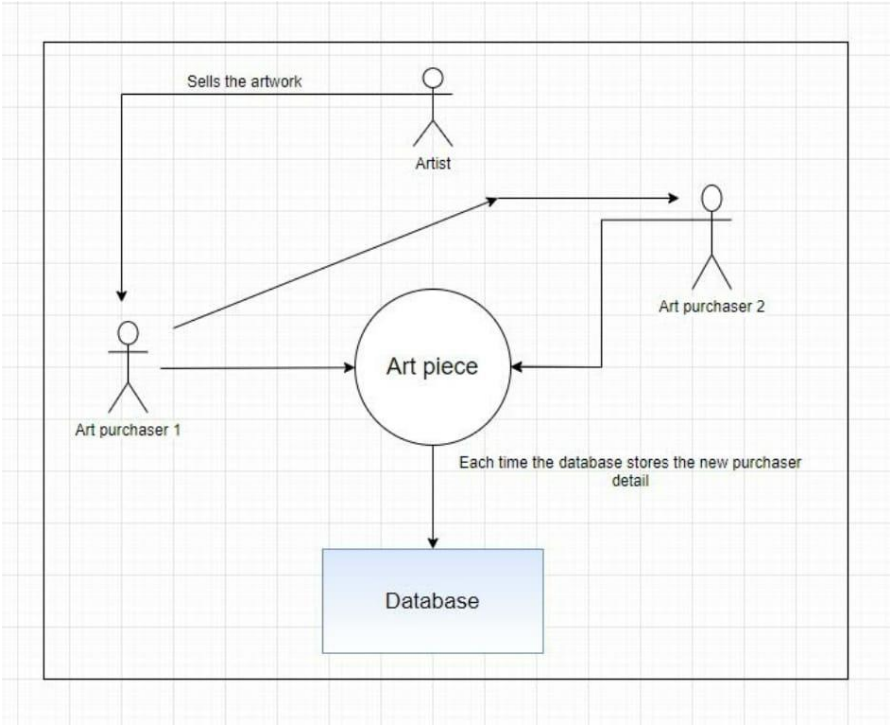
Functions:

Function Name	Description
ERC721 functions ()	Used to transfer ownership, check balance, get approved, check owner, checks if the address is approved for operating, etc.
CreateTokenAndSellArt()	Used to publish and sell the artwork by authors
buyArt()	Used to place an order for digital art, verifies all the conditions and then updates the ownership info.
resellArt()	Used when art owner wants resell the art piece at their own price
findArt()	Used to find art details by passing token Id
findAllPendingArt()	Used to find all art pieces for sale in the gallery
getArtAllTxn()	Used to retrieve transaction history and details

Flow Diagram:



State Diagram:

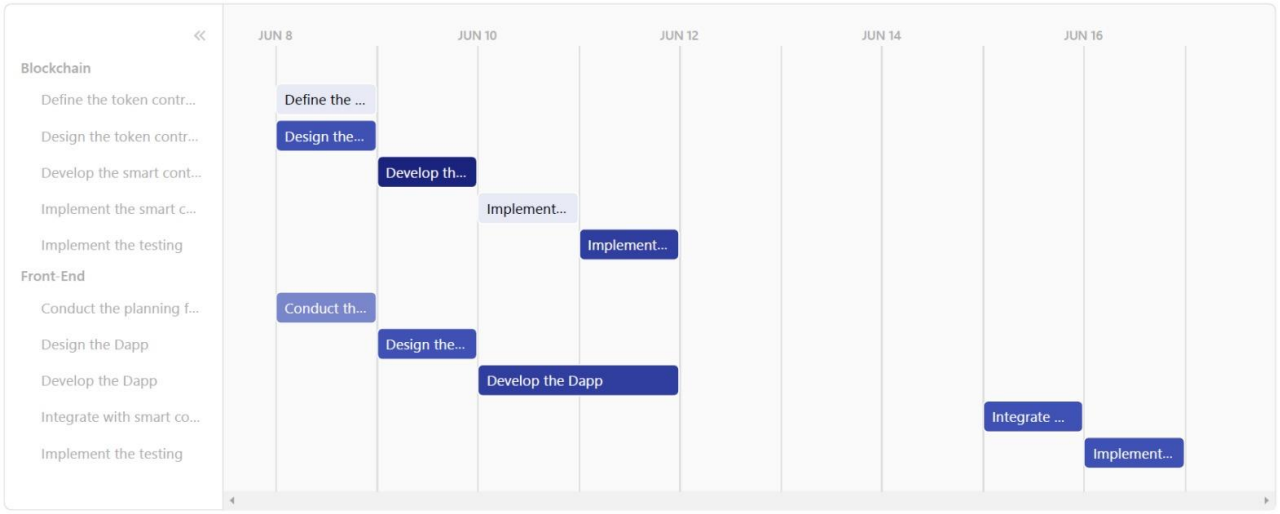


Project Plan

Decomposition and Time Estimation:

The project is decomposed to the major tasks and sub-tasks. Each party is assigned with certain tasks, and estimated hours for every subtask we require to complete are displayed.

Product Roadmap Timeline

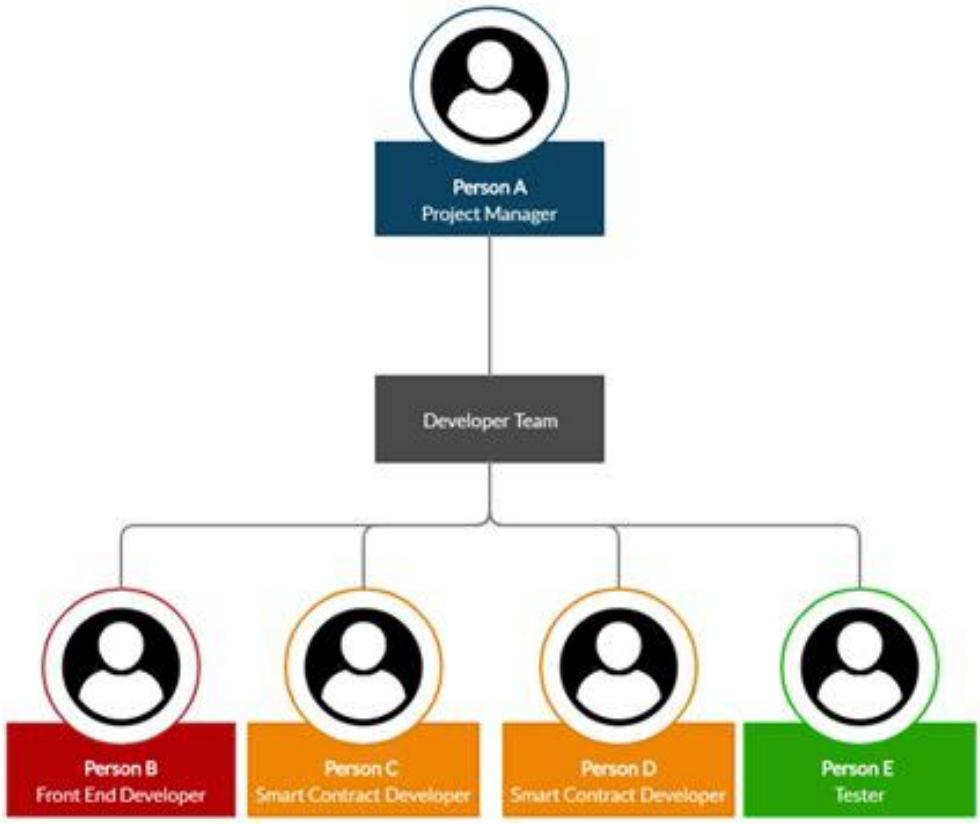


Product Blockchain

Group	Description	Start Day	End Day	Day/s to complete	Responsible Team	Hours
Blockchain 5	Define the token contract	6/8/2020	6/9/2020	1 day	Smart Control Developer	1
	Design the token contract	6/8/2020	6/9/2020	1 day	Smart Control Developer	5
	Develop the smart contract	6/9/2020	6/10/2020	1 day	Smart Control Developer	10
	Implement the smart contract code	6/10/2020	6/11/2020	1 day	Smart Control Developer	1
	Implement the testing	6/11/2020	6/12/2020	1 day	Smart Control Developer and tester	8
	Sum			5 days Sum		Sum 25
Front-End 5	Conduct the planning for Dapp	6/8/2020	6/9/2020	1 day	Front-End Developer	2
	Design the Dapp	6/9/2020	6/10/2020	1 day	Front-End Developer	5
	Develop the Dapp	6/10/2020	6/12/2020	2 days	Front-End Developer	8
	Integrate with smart contact	6/15/2020	6/16/2020	1 day	Front-End Developer	5
	Implement the testing	6/16/2020	6/17/2020	1 day	Front-End Developer and tester	5
	Sum			6 days Sum		Sum 25
	Sum			11 days Sum		Sum 50

Staffing model and cost:

All the parties participate in planning and designing phases; developers and testers are involved in development and testing. The structure of staffing as well as cost are shown in the following diagram.



The estimated hourly wage rate is listed below:

- Project Manager – $50/hr
- Front-End Developer – $40/hr
- Smart Contract Developer – $60/hr
- Tester – $20/hr

Cost Estimation:

Blockchain		Cost/hr				
Description	Hours	Project Manger	Front-End Developer	Smart Contract Developer	Tester	Sum
Define the token contract	1	$ -	$ 60.00	$ -	$ -	$ 60.00
Design the token contract	5	$ -	$ 300.00	$ -	$ -	$ 300.00
Develop the smart contract	10	$ -	$ 600.00	$ -	$ -	$ 600.00
Implement the smart contract code	1	$ -	$ 60.00	$ -	$ -	$ 60.00
Implement the testing	8	$ -	$ -	$ -	$ 160.00	$ 160.00
Sum	25	$ -	$ 1,020.00	$ -	$ 160.00	$ 1,180.00

Frontend		Cost/hr				
Description	Hours	Project Manger	Front-End Developer	Smart Contract Developer	Tester	Sum
Conduct the planning for dapp	2	$ -	$ -	$ 80.00	$ -	$ 80.00
Design the dapp	5	$ -	$ -	$ 200.00	$ -	$ 200.00
Develop the dapp	8	$ -	$ -	$ 320.00	$ -	$ 320.00
Integrate with smart contract	5	$ -	$ -	$ 200.00	$ -	$ 200.00
Implement the testing	5	$ -	$ -	$ -	$ 100.00	$ 100.00
Sum	25	$ -	$ -	$ 800.00	$ 100.00	$ 900.00

Job Title	Cost/Hr	Sum
Project Manger	$ 50.00	$ -
Front-End Developer	$ 40.00	$ 1,020.00
Smart Contract Developer	$ 60.00	$ 800.00
Tester	$ 20.00	$ 260.00
	Total	$ 2,080.00