**Introduction:**

Explored by prototyping a Pay-As-You-Go consumption-based Application where payments are carried out in Ethereum ($ETH) by deploying a Smart Contract on one of the Ethereum Rinkeby Test Network. Tried to tick one or few check boxes within the concepts such as multi-service, microservices, App containerization using docker and deployed on SAP BTP Kyma Runtime with multiple replicas of an application which uses one of the SAP AI Business Services.

**Components of Prototype:**

1. Applications:

Application one uses MetaMask Wallet to communicate a Ethereum Contract which was deployed on Rinkeby Test Network (not the Mainnet) for payments.

Application two uses SAP AI Business Service – Invoice Object Recommendation (IOR) to provide G/L Account Recommendations for the user input file.

These two applications have their own dependencies, programming languages, libraries, scalability requirements and can be deployed and consumed independently

1. Smart Contract:

Built a simple smart contract using Solidity language. Tested used Ethereum Remix. Deployed on Rinkeby Test Network using https://infura.io/. Purpose of this contract is to send the payments across accounts.

1. React:

Used a boiler plate code from GitHub for creating a React Application. This is the UI to interact with MetaMask and to upload a file for G/L recommendations which used Invoice Object Recommendation Service (IOR).

1. Node.js:

Application two was built using Node.js, used Multer library to handle the uploaded file. This application handles the consumption of the IOR and to send back the response to Application One.

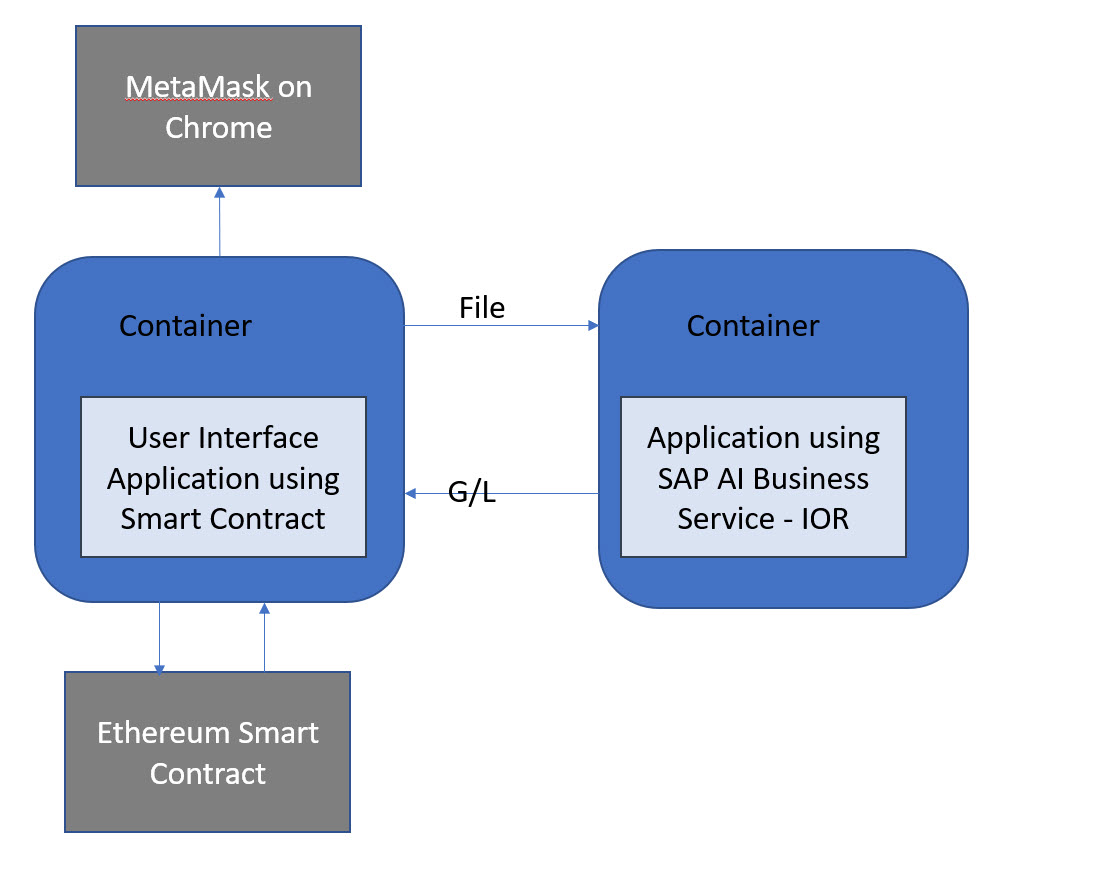
1. Docker:

Docker/Dockerfile allowed to create Images for these applications to run it independently. I have used Docker-compose file to enable these two images/containers to communicate via Network Drivers to run it on localhost.

1. Kyma:

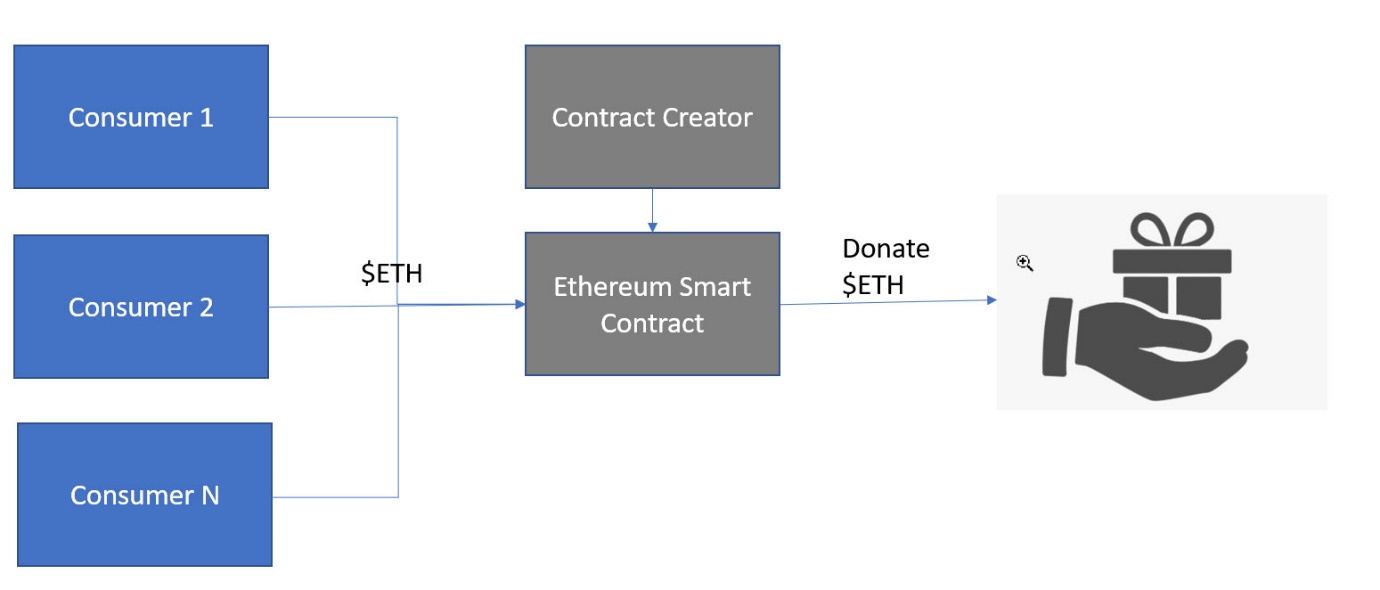
Used Kompose tool to convert docker-compose file. Edited the number of replicas and hosted it on SAP BTP Kyma Runtime as described in this Blog by [Gunter Albrecht](https://people.sap.com/gunteralbrecht)

**Flow:**

****

**Premises:**

Pay-As-You-Go Model: Users will pay a minimum of 0.11 $ETH for using the application. All payments are accumulated in the Smart Contract till the Contract Creator decides to donate 90% of the amount to a Charity.

****

**Demo:**

**Deployment:**

1. Kyma:
2. Smart Contract:

**Output:**

1. Transactions and Account Balances:

* User 1 pays 0.25 $ETH for using the Service.
* User 2 pays 0.5 $ETH for using the Service.
* Creator donates the accumulated amount.
* User 1: 16.848834 ETH – 0.25 ETH ≈ 16.598834 ETH (minus gas fees)
* User 2: 7.5 ETH – 0.5 ETH ≈ 7 ETH (minus gas fees)
* Creator: 16.745724 ETH + (0.75 \* 10/100) ETH ≈ 16.820724 ETH (minus gas fees)
* Charity: 22.656 ETH + (0.75 \* 90/100) ETH = 23.331 ETH

1. Accounts and Transaction Information: Account Information and transaction hash can be found at https://rinkeby.etherscan.io/

* User 1: 0xdcEeF70ebd5e3F07d57712cCe5070B16A2662246
* User 2: 0x9507109C401ef44cEa61E6a423EB285FEAc25715
* Creator: 0xc172eDC818976CaBf0c958AE452FB46C74a033C0
* Charity: 0x8f730fB15772EC8fFCE0954d45baDbeFdbEDA8a8
* Contract Address: 0x11b02bE7e2FBBfa3618Dedddbc01016a214A002D

G/L Recommendations:

* User 1 file output:
* User 2 file output:

**Source Code:** @ GitHub -> User: “shahidla” -> Repository: “ETH”

Folder Information:

Refer folder: eth->contract is used to deploy the Smart Contract.

Refer folder: eth->ethpayment is used to build React Application.

Refer folder: eth->mlservice is used to build SAP AI Business Service – IOR Application.

Refer folder: eth for docker-compose and Kompose files.

**Conclusion:**

This has been a good start for me to understand and explore Cryptocurrency, Docker, Kubernetes, and their tools such as Kompose and ArgoCD respectively. Still working on my fundamentals, trying to draw a picture with Coding and exploring CI/CD.

**Disclaimer:**

This is a prototype type built as part of learning. Bugs do exist. Coding standards, Authorizations and Performance are important and essential aspects for any application development which are not considered as part of this Prototype.

**References:**

<https://developers.sap.com/tutorial-navigator.html?tag=products:technology-platform/sap-business-technology-platform/sap-btp-kyma-runtime>

<https://developers.sap.com/tutorial-navigator.html?tag=products:technology-platform/sap-ai-business-services/invoice-object-recommendation>

<https://blogs.sap.com/2021/03/29/kyma-kubernetes-kanban-deploying-a-multi-service-application-on-sap-btp/>