**INTERNSHIP : PROJECT REPORT**

| Internship Project Title | RIO 125: Setup a docker container for application development using Blockchain Network |
| --- | --- |
| Project Title | Setup a docker container for application development using Blockchain Network |
| Name of Organisation | Tata Consultancy Services |
| Name of Industry Mentor | Debashis Roy |
| Name of Institute | Sister Nivedita University |

| Start Date | End Date | Total Effort (hrs.) | Project Environment | Tools Used |
| --- | --- | --- | --- | --- |
| 19/02/2022 | 15/04/2022 | 133.5 | UBUNTU (19.04) | Eclipse, JDK, Vim, Solidity, Solc, Ganache, Ethereum, Truffle, Docker, Git, Github, etc |
| Project Synopsis:  **Docker** is a containerization platform that packages an application all its dependencies together in the form of a docker container to ensure that the following application works seamlessly in any environment.  Docker container image is a lightweight, standalone, executable package of software that has everything one can need to run an application – code, runtime, system tools, system libraries, and settings. The isolation and security allows us to run many containers simultaneously on a given host without interference.  Containers are lightweight because they don’t need extra load of a hypervisor, but can run directly within the host machine’s kernel. This means we can run more containers on a given hardware combination than if we were to use virtual machines. One can also run Docker Containers within host machines that are actually virtual machines.  Docker Container Key Points:   1. VM is virtualization of Hardware, while Container is virtualization of OS. 2. Containers are all about the ability to make quick, iterative deployments of our applications. 3. Using containers changes how we develop, test and deploy applications. 4. Docker helps us to test the code before we deploy it to production as soon as possible. 5. Since docker containers are lightweight, they are very easily scalable. 6. Docker containers run everywhere: Linux, Windows, Cloud, Serverless, Data Centres, etc. We can run the same containers everywhere. 7. Docker has the ability to reduce the size of development by providing a smaller footprint of the operating system via containers. | | | | |
| Components of Docker:   1. **Docker Engine** - The Docker engine is a part of Docker which creates and runs the Docker containers. The docker container is a live running instance of a docker image. 2. **Docker Image** - The Docker Image is building the block of a docker or docker image is a read-only template with instructions to create a Docker container. 3. **Docker Hub** - A Docker Hub is a registry which is used to host various docker images. 4. **Docker Compose** - A Docker compose is used to define applications using multiple docker containers. 5. **Docker Daemon** - Docker Daemon is simply a part of Docker Engine which is a persistent background process that manages the containers on a single host. 6. **Hypervisor** - A hypervisor also known as a virtual machine monitor(VMM) is a software that creates and runs virtual machines. | | | | |
| Solution Approach:  (Advisory followed in the presentation given on the RIO Platform)   * Creating a docker file with the following specifications:  1. Installing git (for version control) 2. Installing vim (required for editing the files) 3. Installing build-essential 4. Installing OpenJDK (Java Development Kit) 5. Configuring OpenJDK 6. Setting the environment variables if not already configured in the GUI setup. 7. Installing Eclipse IDE and configuring the same in GUI setup 8. Installing the YAKINDU plugin for Eclipse to support solidity 9. Set the YAKINDU plugin 10. Installing EVM and configuring the same 11. Installing Solc and configuring the same  * Creating a docker file and creating an image * Once the Image is created and operating normally, test the image * Once the Image is up & running, update the Docker file and add the following  1. Install and configure NordJS 2. Install and configure truffle packages 3. Install and configure testrpc 4. Initialise truffle projects 5. Deploy the contracts 6. Create dApp 7. Launch dApp server 8. Install and configure Ganache 9. Exposing the port  * Configure the environment variables if needed * Create the docker image after updating the Docker files * Create a blockchain network with Ganache using the image created after updating the Docker file. * Test the workspace and interface * Perform transactions | | | | |
| Assumptions:   * The OS used is Linux (Ubuntu). * Git is pre-installed and a root folder is created. * Dependencies are updated. * Docker is updated. * The IDE used is Eclipse and the text editor used is Vim. * Docker’s official GPG key is used for security purposes. * Captured Runtime execution for some to observe the Process Scheduling. | | | | |
| Project Diagram: | | | | |
| Algorithms:     * Setting base Ubuntu as Ubuntu 19.04 * Running an update and installing git, vim, curl, OpenJDK * Installing JRE * Installing the Eclipse IDE * Installing Ethereum * Installing NodeJS and NPM * Installing Ganache, Express, Solc * Updating the system again after the installations are done * Installing Truffle * Settings contracts & migrations | | | | |
| Outcome:   * Docker Image is running successfully * Image has been tested, and running * Blockchain container has been creating using Ganache CLI * The platform was tested using smart contracts * The application development container is up and running | | | | |
| Exceptions Considered:   * The OS must be up-to-date * Environment must be configured * Environment variable and Path variable must be set up | | | | |
| Enhancement Scope:   * The reusability can be enhanced * The security can be made more versatile to ensure protection against vulnerabilities. | | | | |
| Link to the code and executable file: https://github.com/FrozenRetard/DockerContainer\_TCSRIO | | | | |