SUBMITTED BY

KARAN MANCHANDIA

DISTRIBUTED SYSTEMS

PROJECT – 2

Publisher Subscriber model in Docker

PROJECT CONTENTS:

INTRODUCTION

The project mainly focuses on implementing the publisher subscriber model. The publisher subscriber model is one of the topics of Indirect Communication in the Distributed Systems area. The project implements the publish(), subscribe() and notify() aspects of the Pub/Sub system. Apart from the Publisher Subscriber model, the project also involves developing an online compiler web application. The pub/sub offers many-to-many relationship i.e, any number of subscribers can subscribe to one publisher and any one publisher can publish to any number of subscribers. All the phases of the project are implemented in Docker.

A pub/sub system is a system where subscriber subscribe to the publishers and the publishers publish the content. Whenever a subscribe subscribes, he will get all the notifications of the topic of his interest from the publisher. Since this project is built on Topic-based and Rendezvous routing, the subscriber subscribes to topics and publishers publish articles on topics. Whenever the publisher publishes a topic, it will notify to the subscribers of interest. The routing is done between the subscribers and publishers based on Rendezvous routing.

The online compiler takes in a piece of code from web interface and compiles it on the local machine and gives back the output from the local machine to the web interface.

SOFTWARE REQUIREMENT SPECIFICATION

Software requirements specification describes the requirements for external functioning of the application. The Software and hardware requirements are mentioned below. Each of them have been used for one or the other phases of the project.

SOFTWARE REQUIRMENTS:

OPERATING SYSTEM: Windows 7/8/8.1/10
WEB FRAMEWORK: Flask
WEB STACK: Python, JavaScript, HTML, CSS, SQL, Ajax Jquery

But all these are not required for the implementation of the project. Docker takes care of all the dependencies for the execution of the project.

- Docker is a program that performs Operating-System level virtualization using the concept of containers, also known as Containerization.
- Containers are a light weight version of the Virtual Machines.
- Different containers are isolated from each other, which sets it apart from the virtual machines.
- Each container of the docker system are run using an ‘image’.

SOFTWARE DEVELOPMENT LIFE CYCLE

Analysis:

In this stage we analyze all the requirements for the project and prepare plan on phases for the application to be made in.

Design:

A design is made for the implementation of each phase of the project. Proper web stack is chosen based on our requirements collected in the Analysis stage. A complete design of the project and deadlines are made by this stage.

Implementation:

The design formed in the previous stage is put to work in this stage. The implementation is done with the web-stack decided. The web-application is developed in this stage.

Testing:

The developed web-application is put to test in this stage. The testing has been done manually for this project

Deploying Docker:

The entire project made from all the previous stages is virtualized using the Docker software. A docker image is made for each phase of the project and the projects are run on these images virtually. The docker system has been discussed in the Introduction part of the documentation.

PROGRAMMING LANGUAGES, PLATFORMS USED AND CHOICE OF TECHNOLOGIES

FLASK:

Flask is a web framework written in python.

PYTHON:

Python has been mainly used for the server side implementation of the web application. The main part of the python programs is to process the requests passed to it from the user, process them and send them back.

HTML5:

HTML is mainly used to implement the design of the web pages on which the user will be working. The HTML implements the UI for the user(client). The reason for choosing html is its most amazing feature that is semantics. The new tags like <aside> and <nav> makes HTML more meaningful.

JAVASCRIPT:

JavaScript is mainly used to call the invoke methods on the server side and get result to display it on the UI of the client.

CSS:

The reason for using CSS (Cascading Style Sheets) for designing the front end is that CSS allows us to separate the content from the presentation layer. By using CSS the HTML page size is smaller and hence it can load faster.

JQUERY AJAX:

Ajax is an asynchronous javascript and XML language. It is used to make calls to the server side which is embedded in the javascript code. The main reason for using Jquery Ajax is that it increases performance and speed. It also reduces the traffic load between the client and the server.

My SQL:

The flexibility of using open source software is the main reason for using my sql database. Scalability is another reason for using my sql database.

DOCKER:

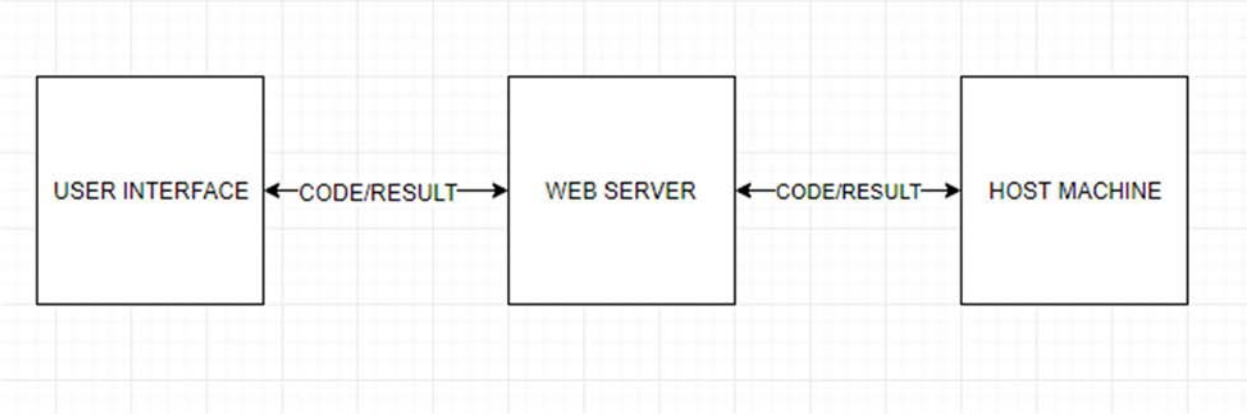
Docker is used to virtualize the entire project implemented using the above languages and platforms. Docker makes the project independent of requirements of the project, since the user can simply run the project on his system using image. For phase 2, the NGINX has been used as the server.

ARCHITECTURE

There are different architectures for different phases of the project.

For the PHASE-1, the user sends a code as request to the server, where the server compiles and executes the code on the local machine and sends back the execution result from the local machine.

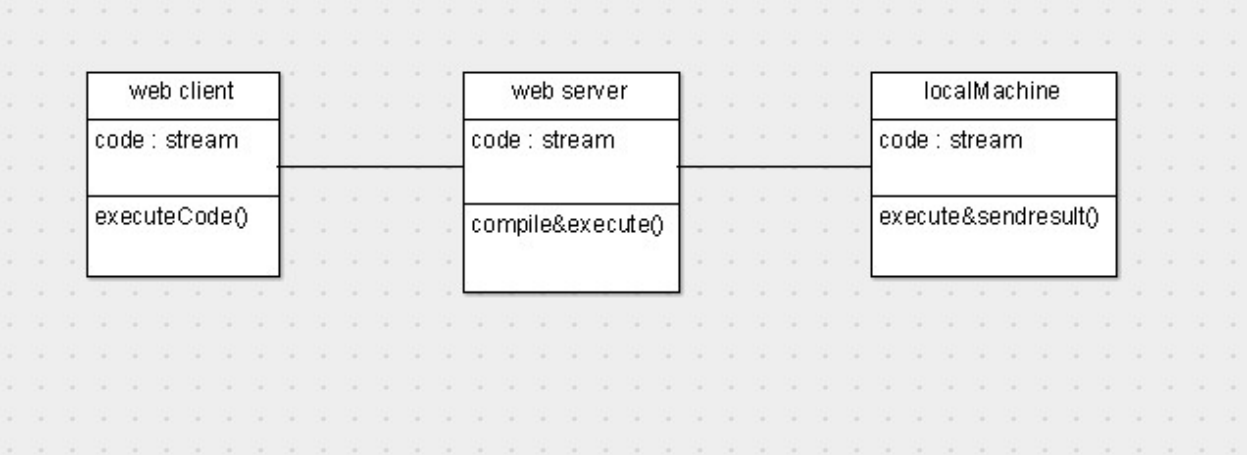
Architecture Diagram:



the above diagram represents the architecture diagram of phase-1

Class Diagram:.

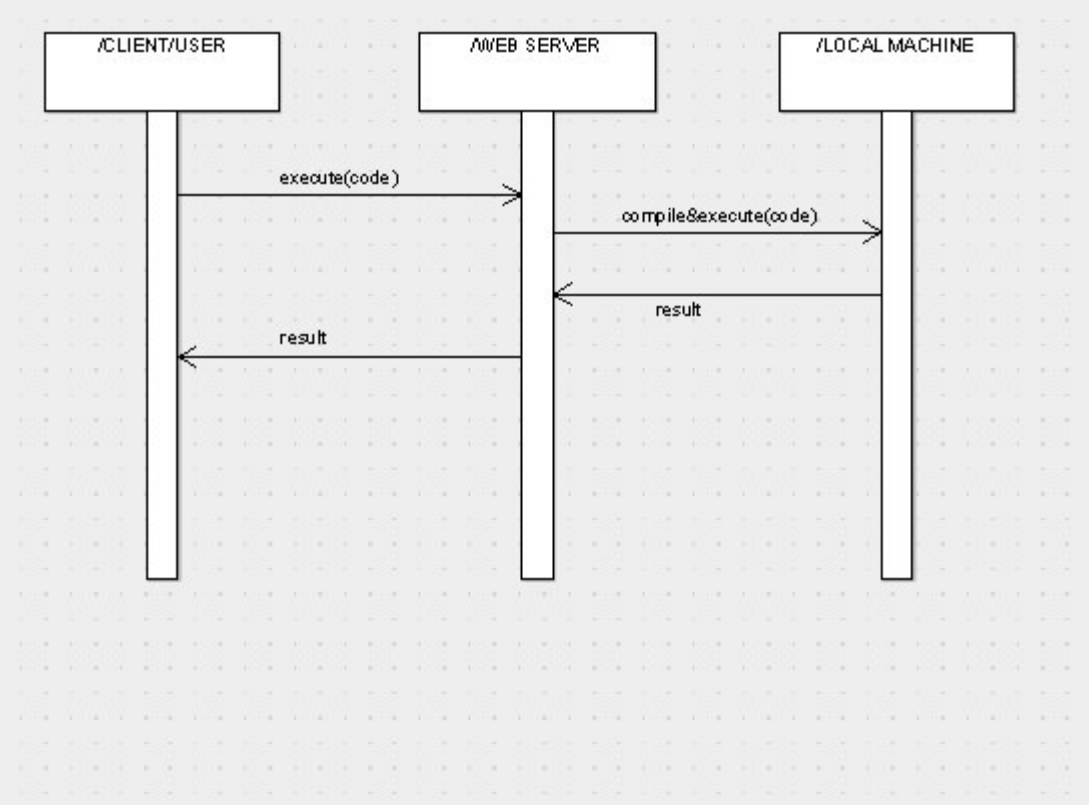
Class Diagram gives the static view of the model which helps in visualizing and documenting different aspects of the application.



The above diagram represents the class diagram of the phase-1

Sequence Diagram:

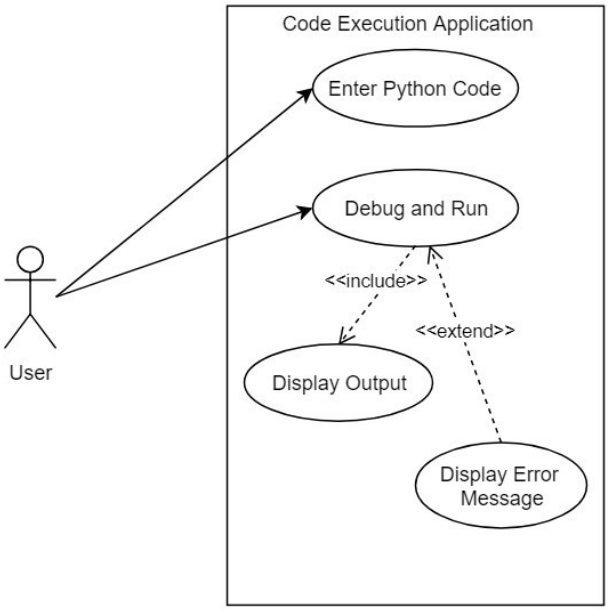
The sequence diagram gives the systematic flow of the web application. The sequence diagrams represents the object interactions arranged in time space.



The above diagram represents the sequence diagram of phase-1

USE CASE DIAGRAM:

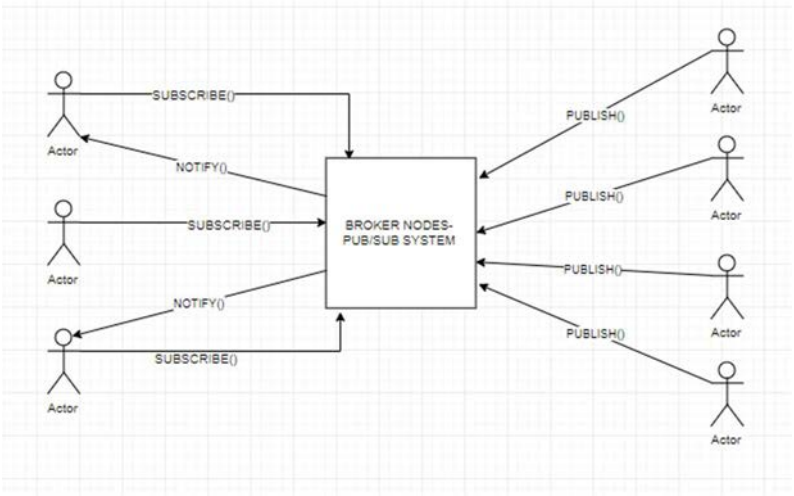
The use case diagrams are used to represent the set of actions that can be performed by the user. We have two kinds of users: Subscribers and Publishers.



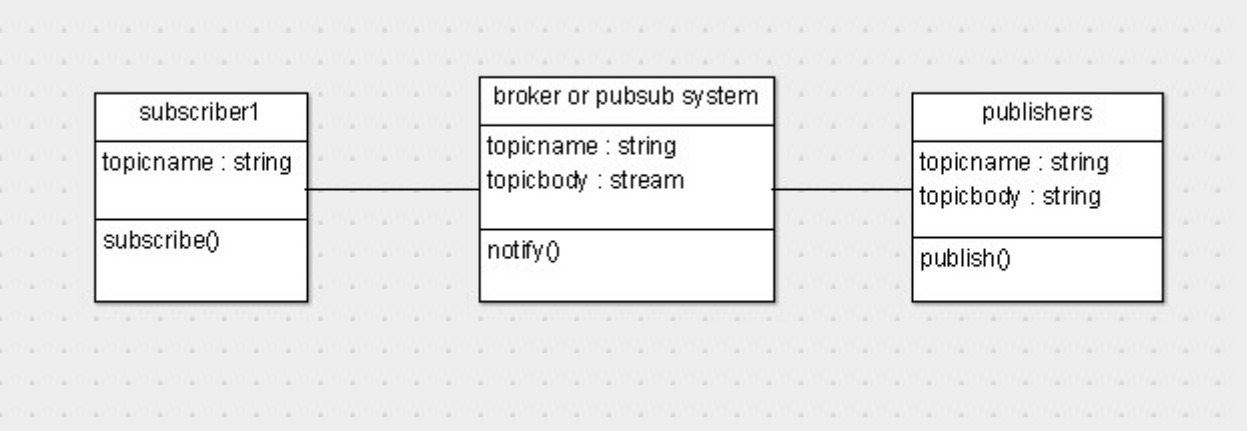
–

For the PHASE-2 and 3, different subscribers and publishers share content through the pub/sub architecture. The Subscribers subscribe to different topics and publishers publish articles on various topics which are routed to the subscribers through rendezvous routing in phase 3. The Architecture and UML diagrams are represented below.

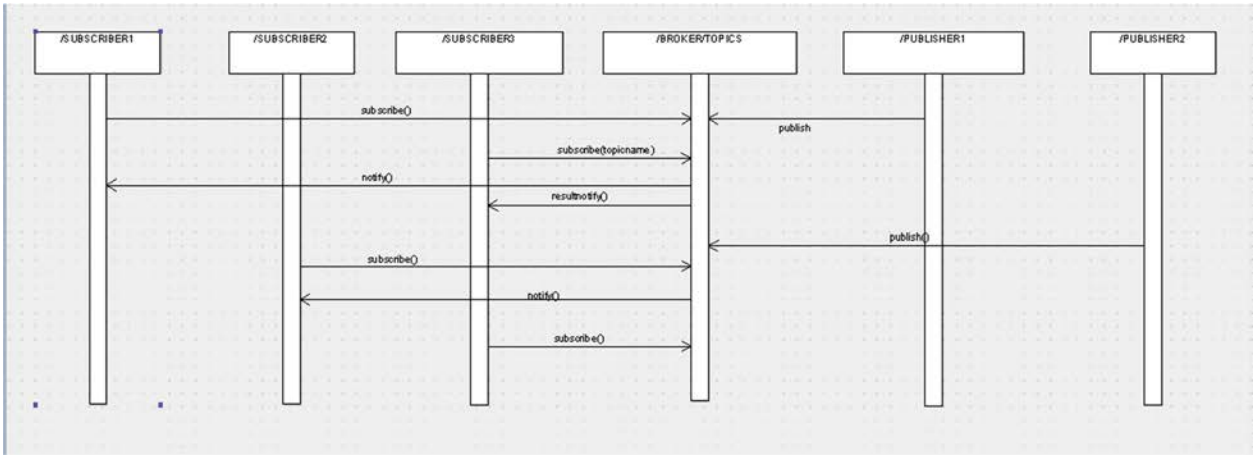
Architecture Diagram:



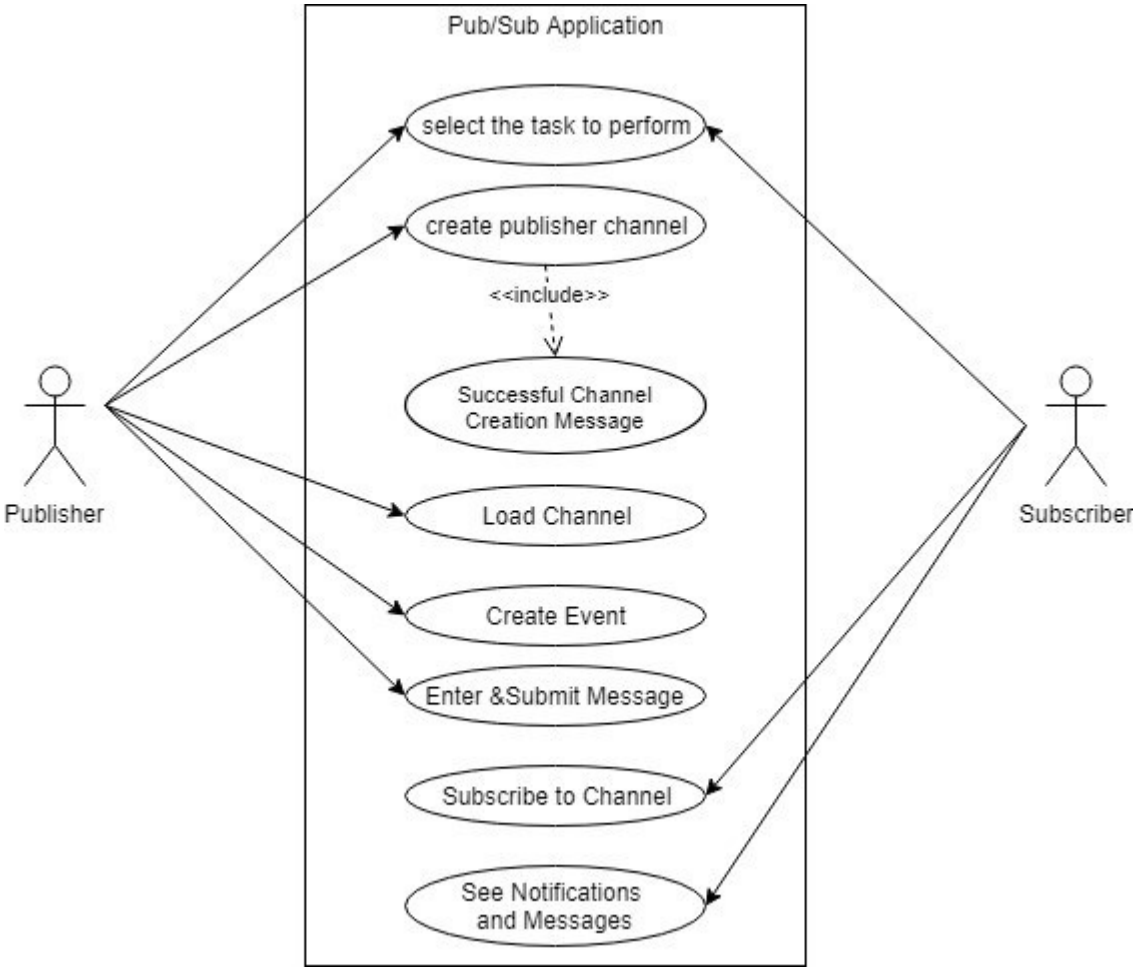
Class Diagram:



Sequence Diagram:

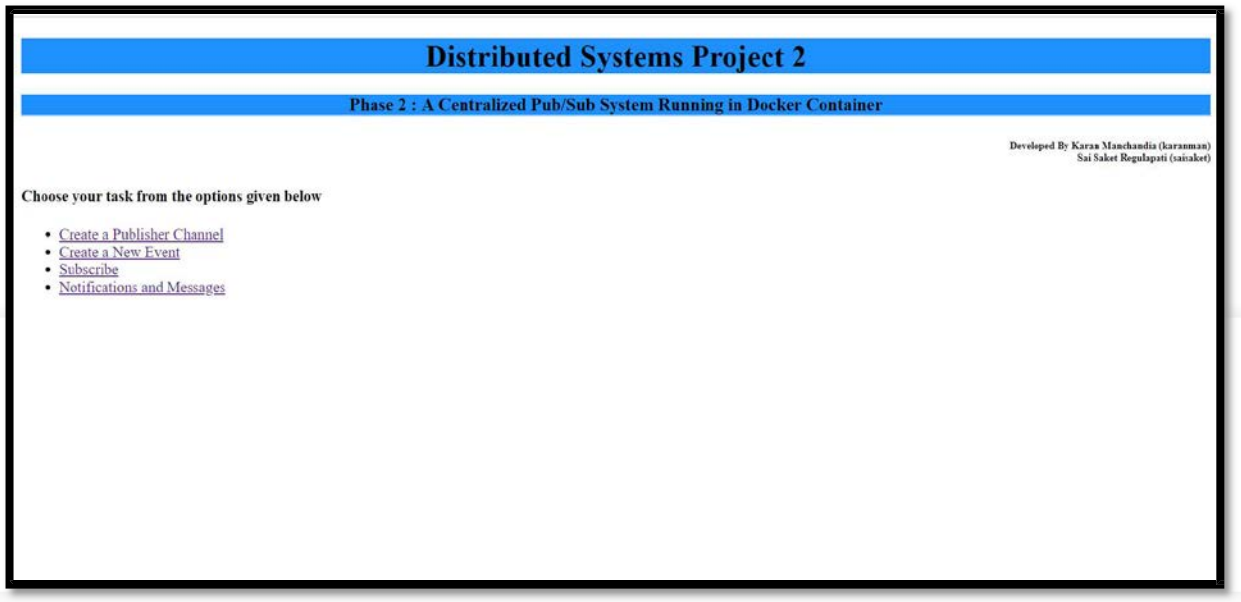


User Case Diagram:

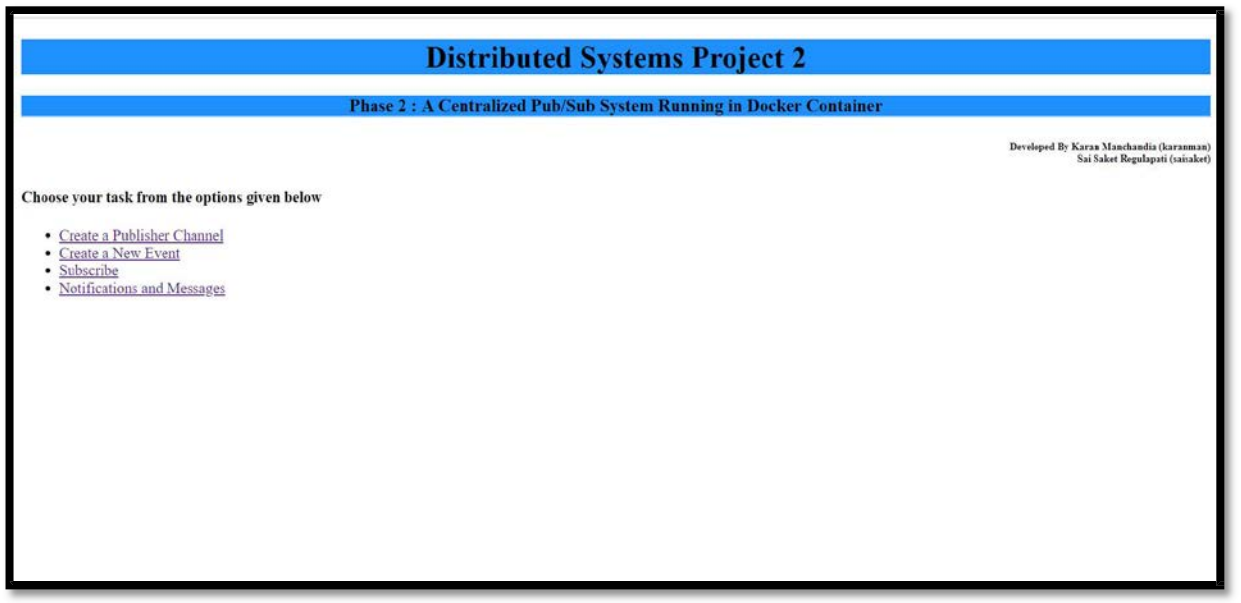


SCREENSHOTS

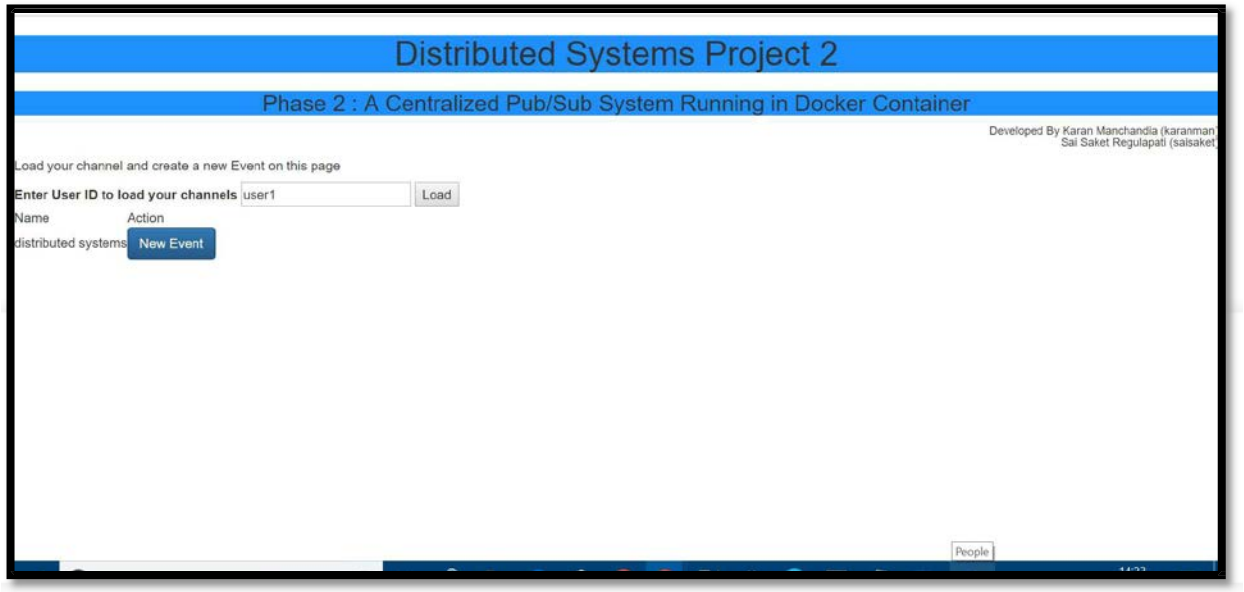
Home page of the Pub/Sub system.



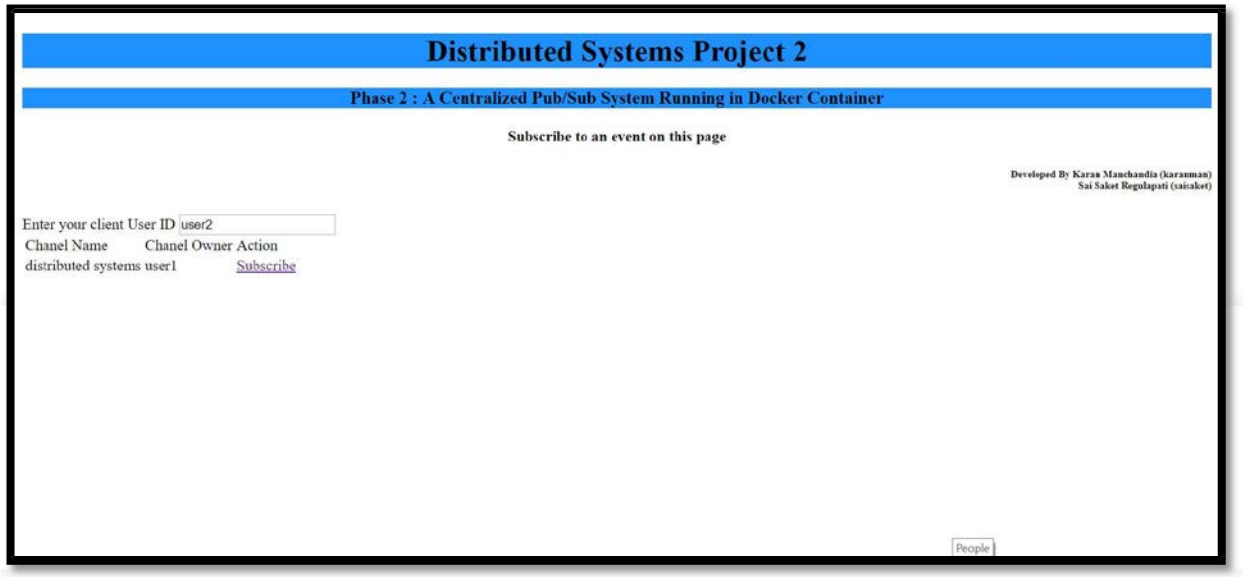
Webpage to create the topic:



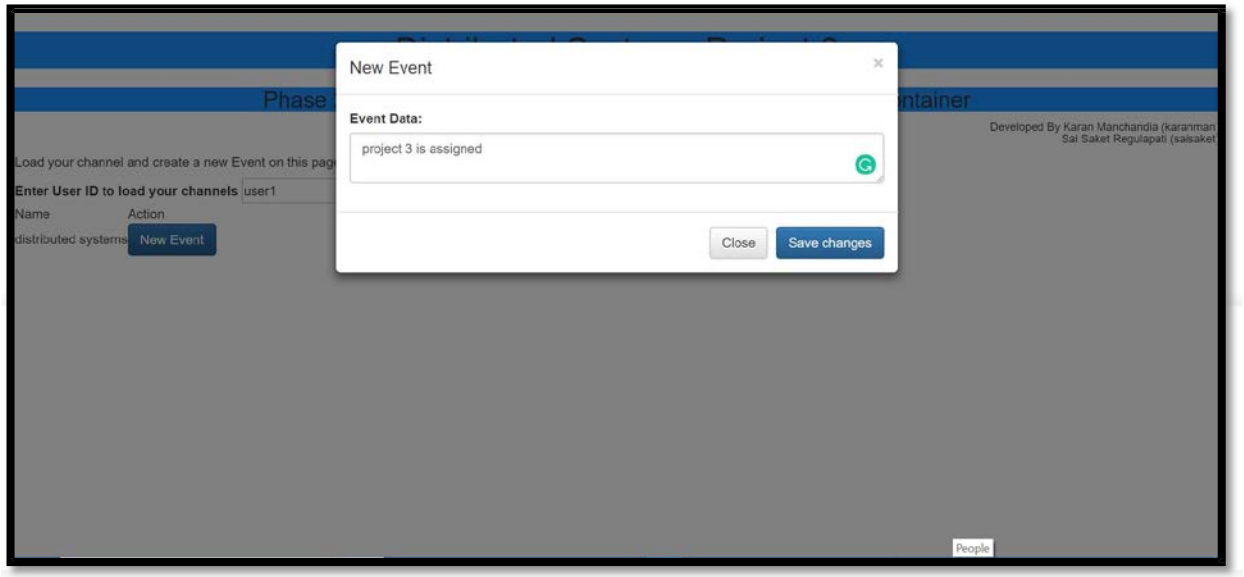
Webpage to Create a new event (publish()):



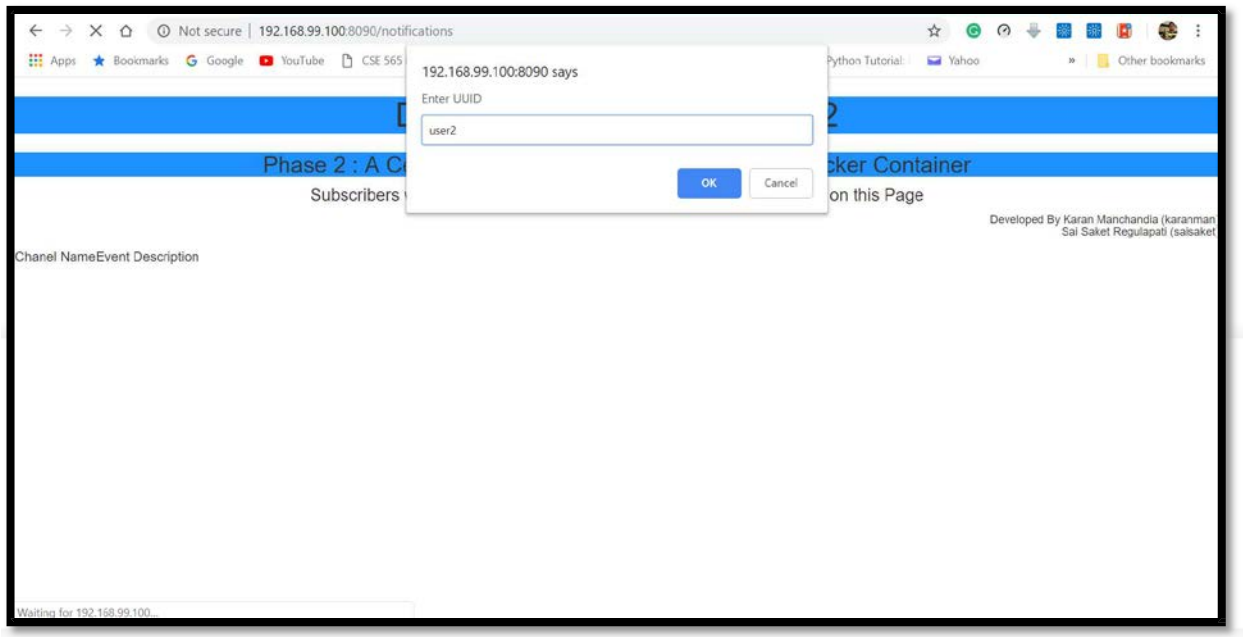
Webpage for the subscribers to subscribe:



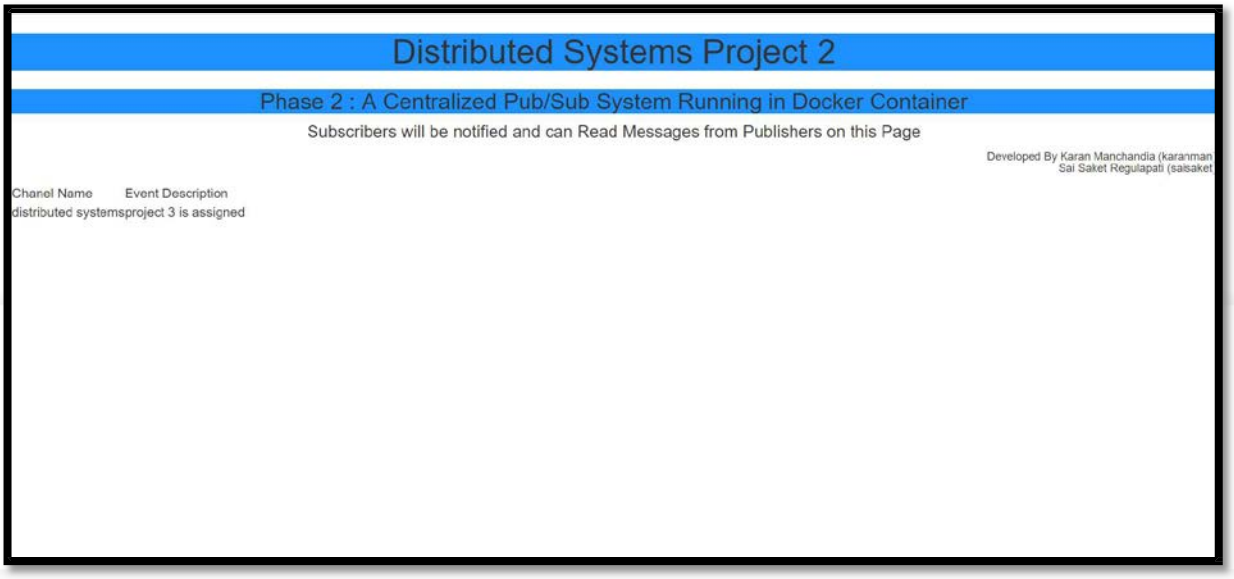
A Publisher publishing an event:



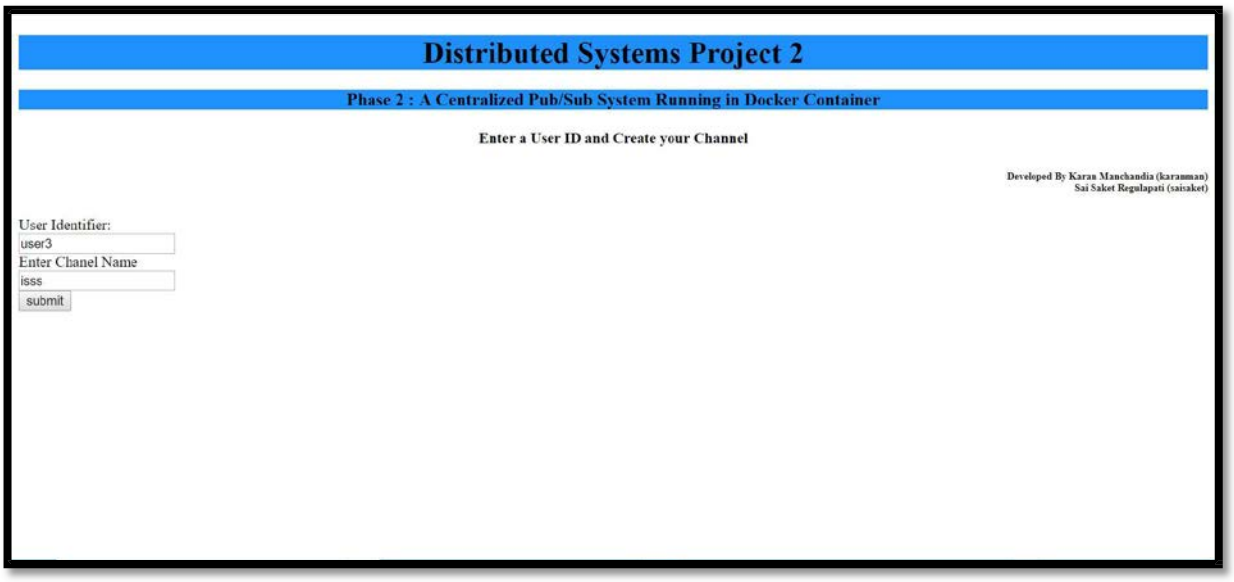
Enter User ID (subscriber ID) to see the event:



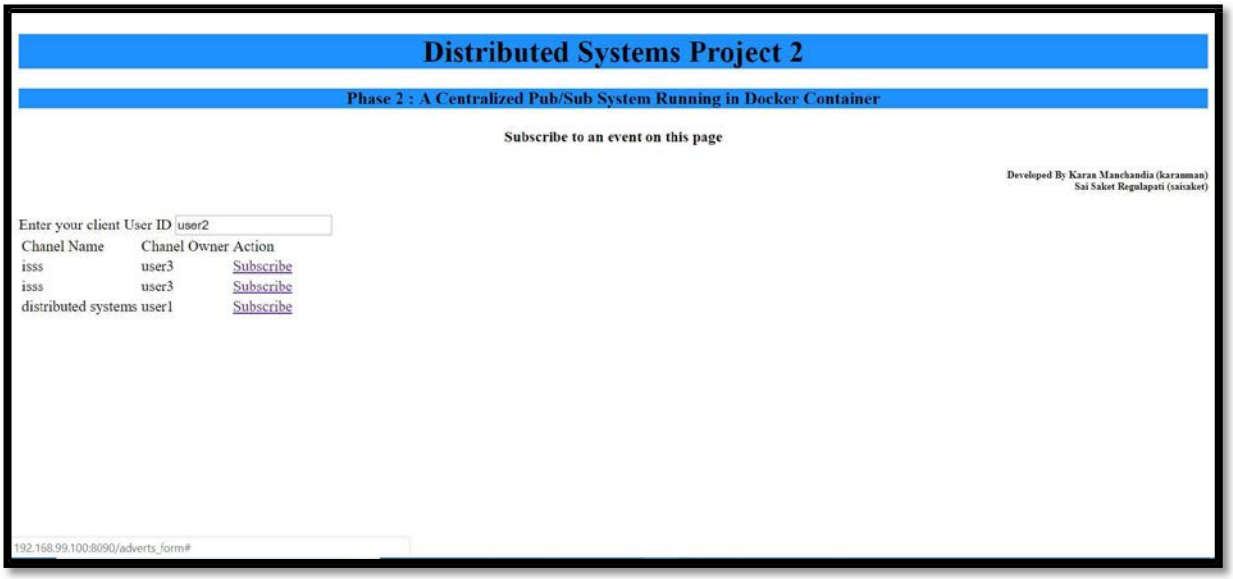
Subscriber getting the notification:



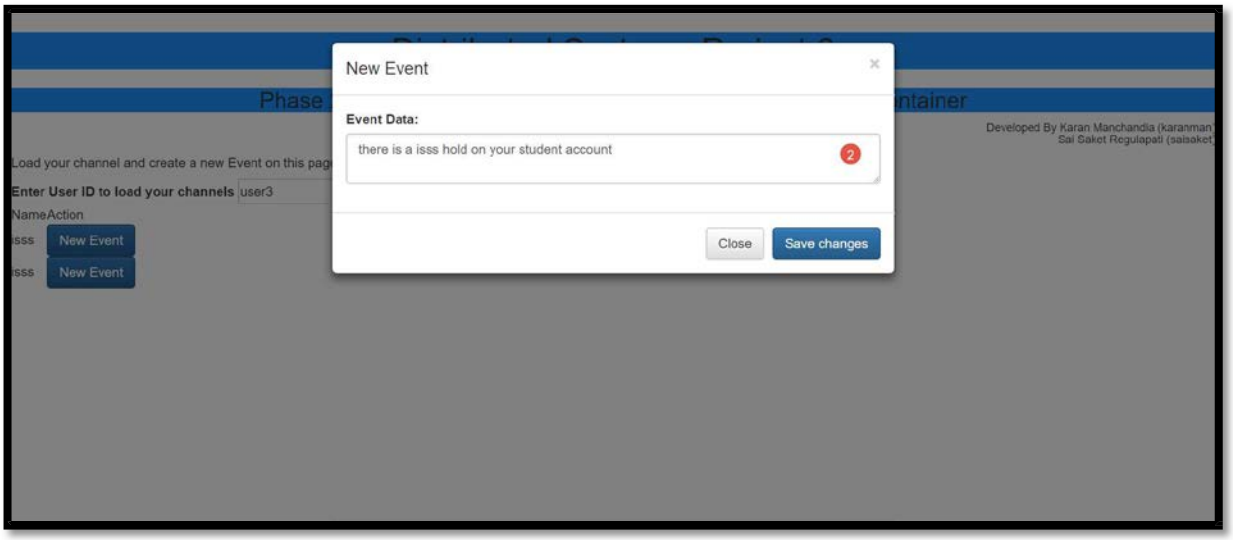
Creating a new Publisher:



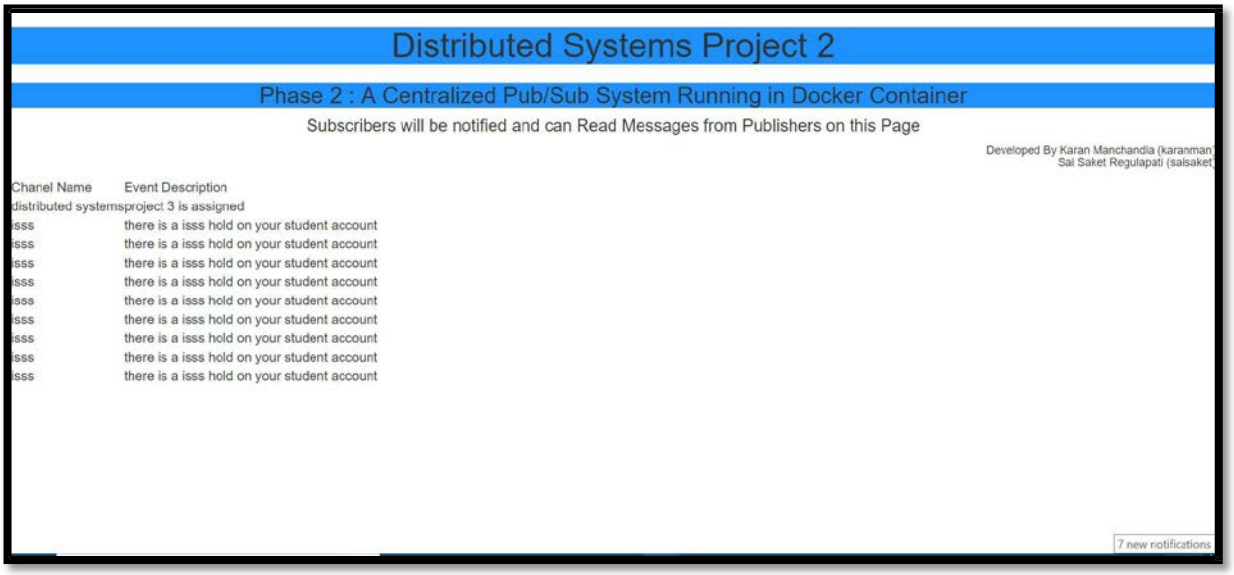
Subscriber subscribing to Publishers:



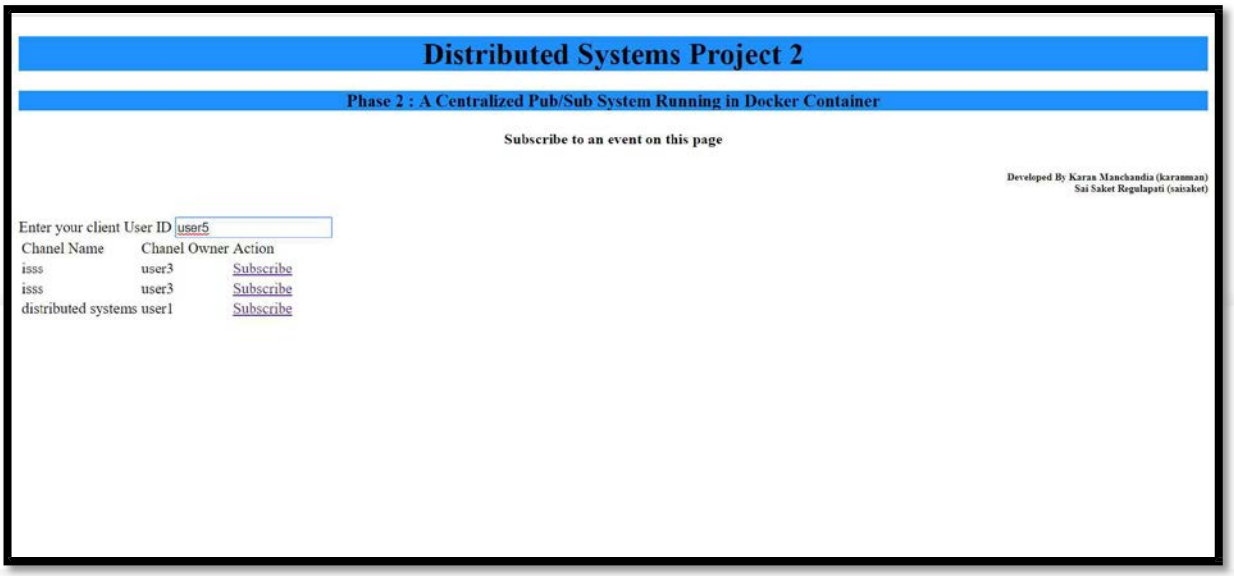
Publisher writing an event (publishing):



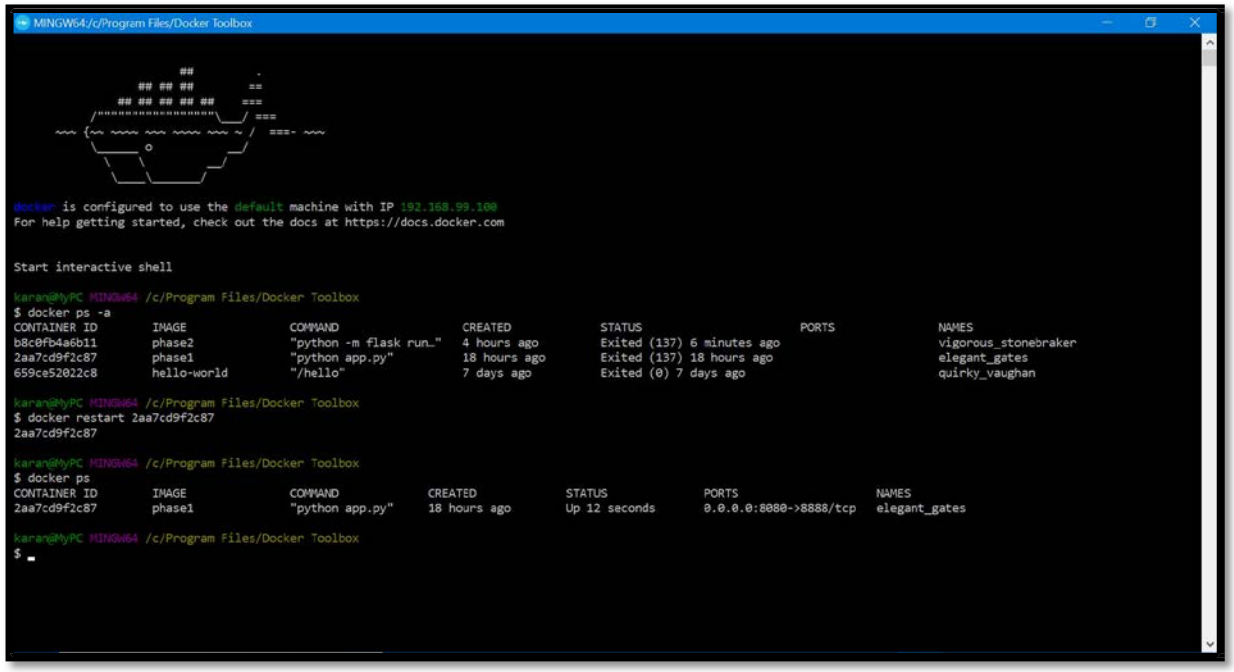
Subscriber getting notifications:

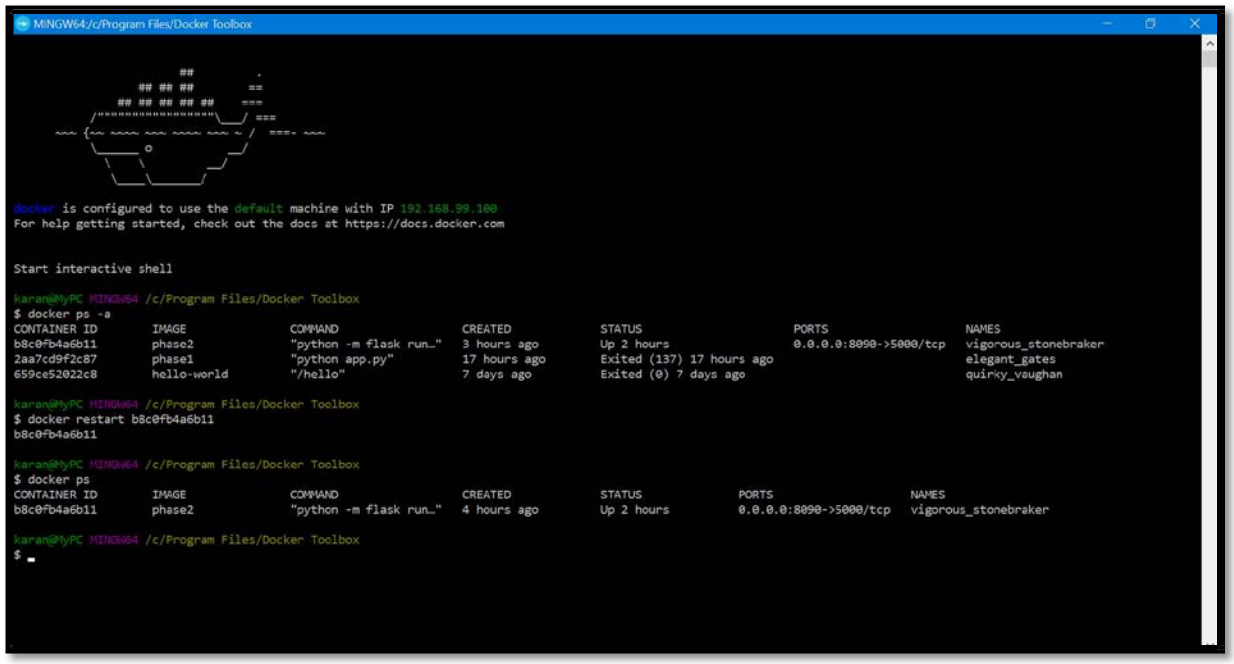


Now lets create a new user (user5) to demonstrate to demonstrate that multiple users can subscribe to a single publisher:

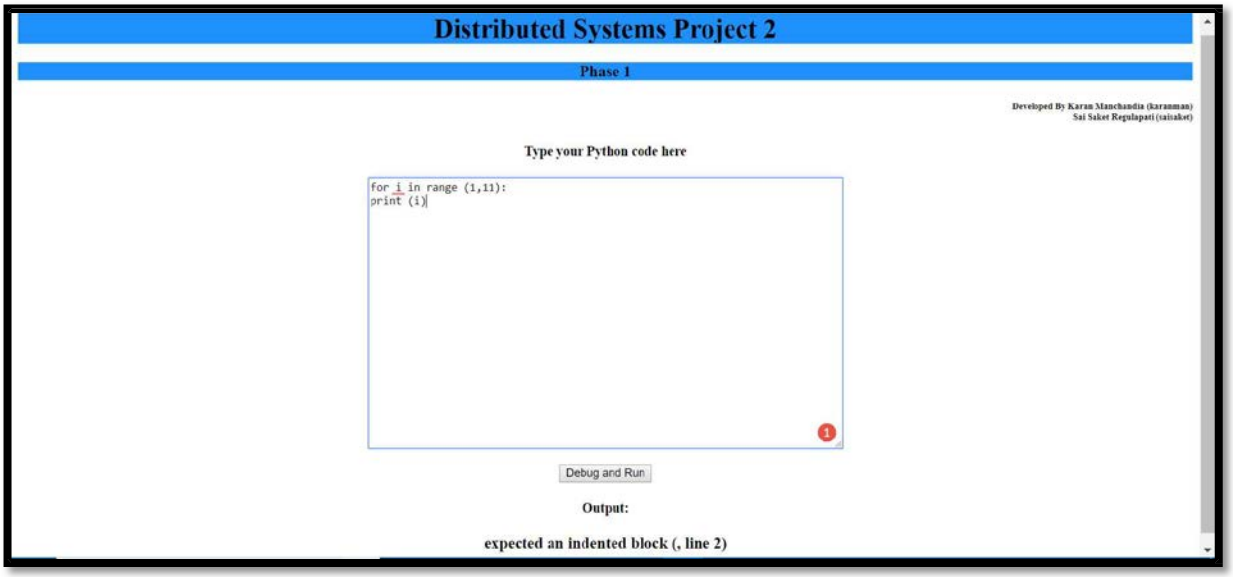


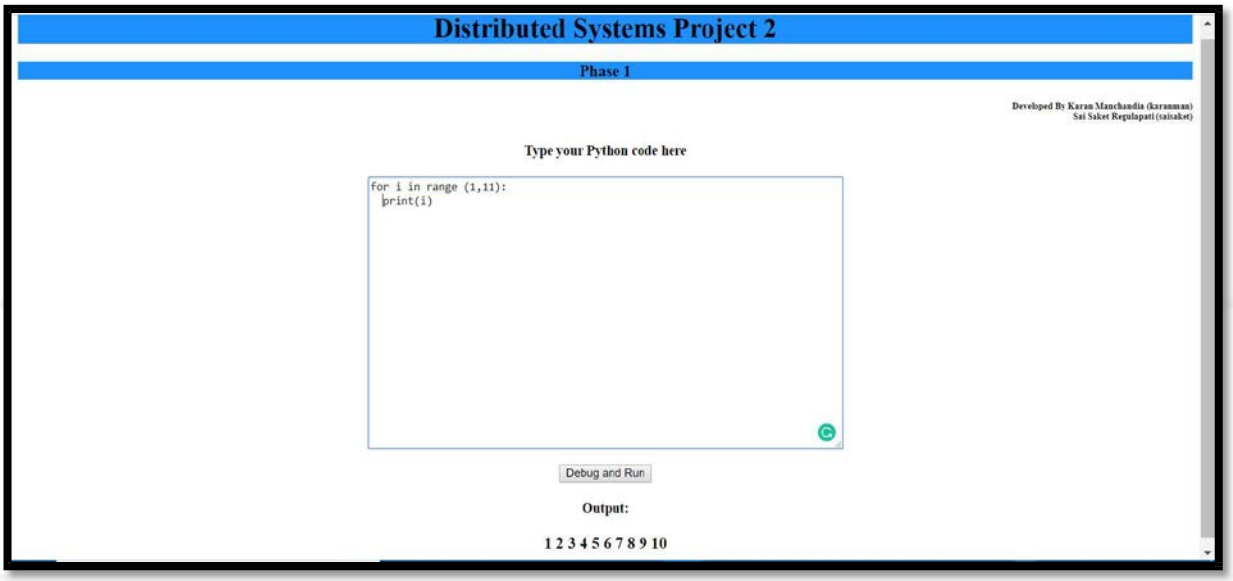
Images representing running state of Docker for phases 1 & 2:





Implementations of phase – 1:





CONCLUSION AND FUTURE ENHANCEMENTS

To optimize the performance of the Pub/Sub system use of web/system sockets can be very useful. All the subscribers can listen to the same port/socket where the publisher publishes articles. The match operation can be performed at the socket level. This reduces the time and the cost function of the Web Application.

Each phase of the web applications is designed in four stages analysis, design, implementation and testing. The web applications implement the pub/sub system and an online compiler model. The web application can be enhanced by the use of databases or web/system sockets.

REFERENCES

[1]: https://docs.docker.com/toolbox/
[2]: https://djangostars.com/blog/what-is-docker-and-how-to-use-it-with-python/
[3]: https://www.docker.com/resources/what-container
[4]: https://stackoverflow.com/
[5]: https://www.geeksforgeeks.org/