

SUBMITTED BY

KARAN MANCHANDIA

DISTRIBUTED SYSTEMS

PROJECT – 2

Publisher Subscriber model in Docker

PROJECT CONTENTS:

1. Introduction
2. Software Requirement Specifications
3. Software Development Life Cycle
4. Programming Languages and Platforms used
5. Architecture
6. Code Snippets
7. Working Screenshots
8. Future Enhancements
9. Conclusion
10. References

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 subscriber 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:

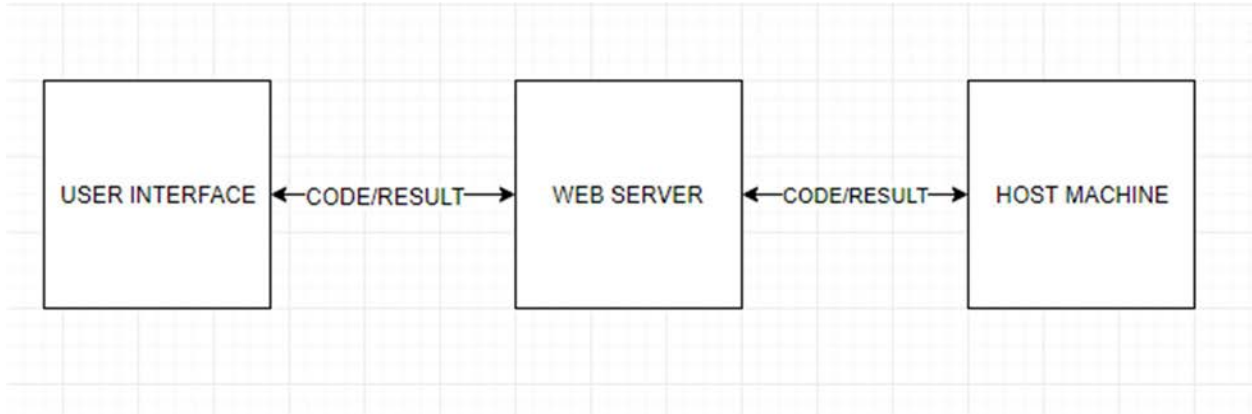
Docker is used to virtualize the entire project implemented using the above languages and platforms. Docker makes the project independent of requirements of the project, since the user can simply run the project on his system using image. For phase 2, the NGINX has been used as the server.

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.

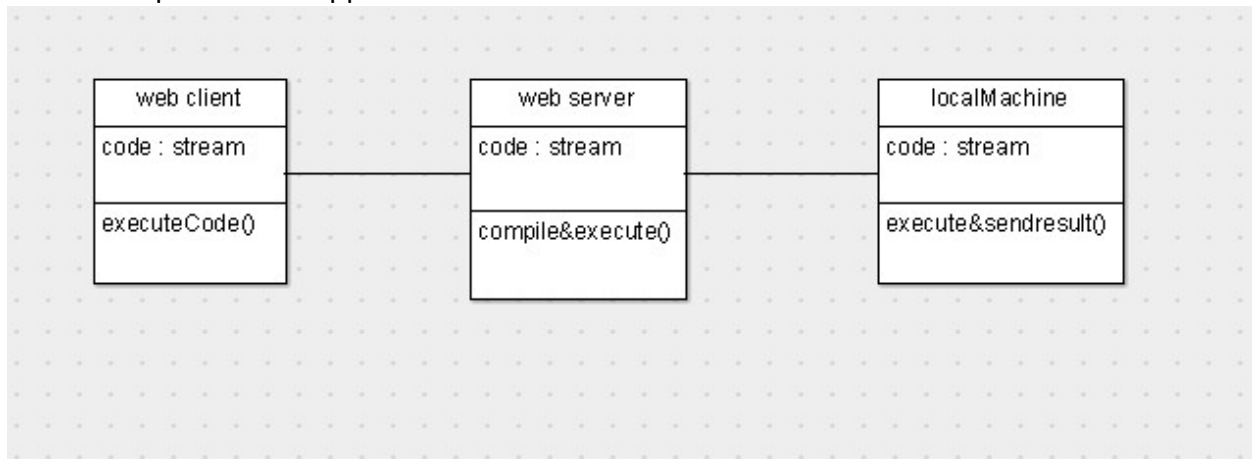
Architecture Diagram:



the above diagram represents the architecture diagram of phase-1

Class Diagram:.

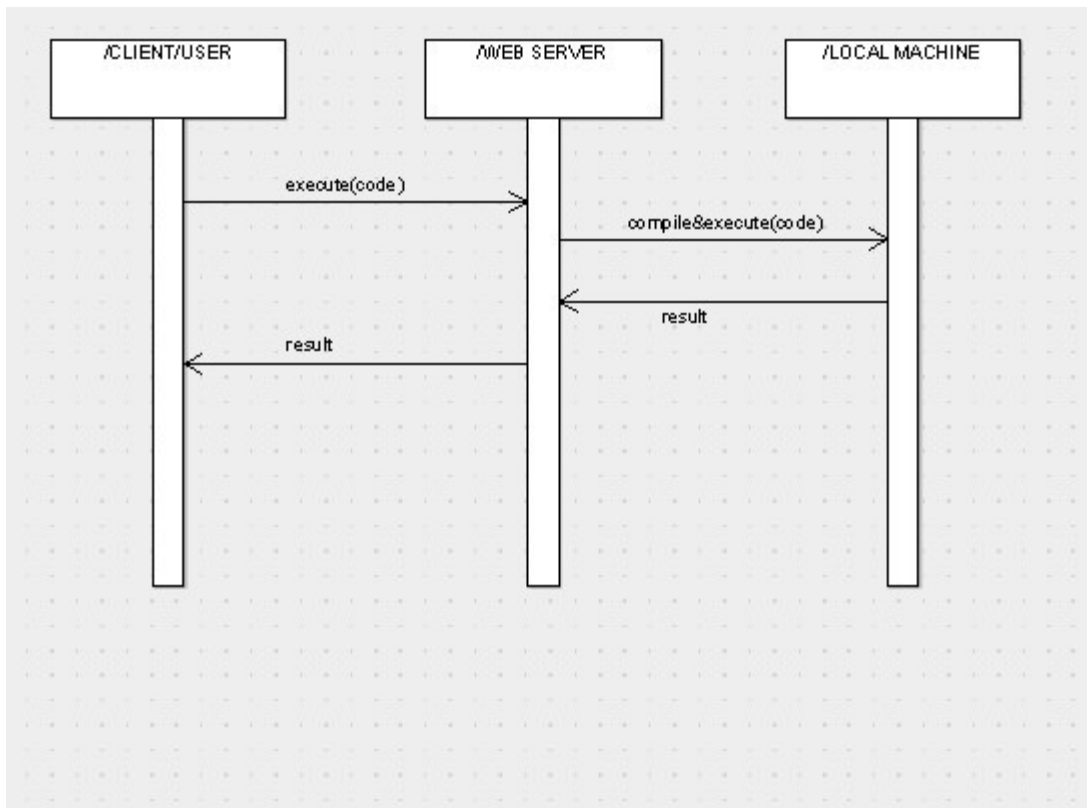
Class Diagram gives the static view of the model which helps in visualizing and documenting different aspects of the application.



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

Sequence Diagram:

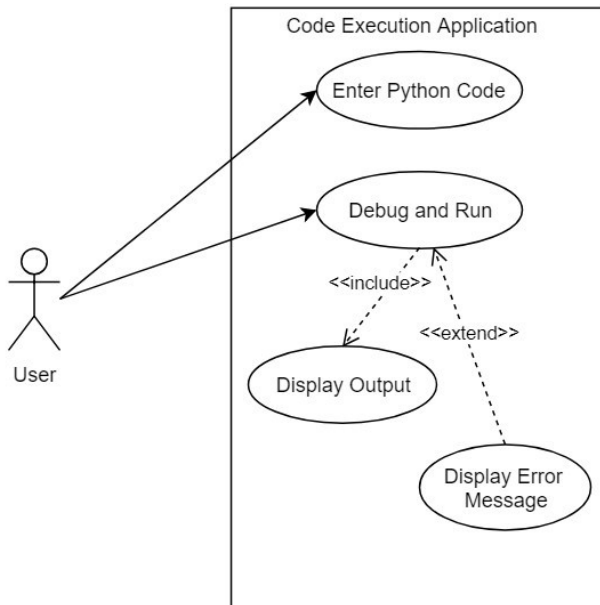
The sequence diagram gives the systematic flow of the web application. The sequence diagrams represents the object interactions arranged in time space.



The above diagram represents the sequence diagram of phase-1

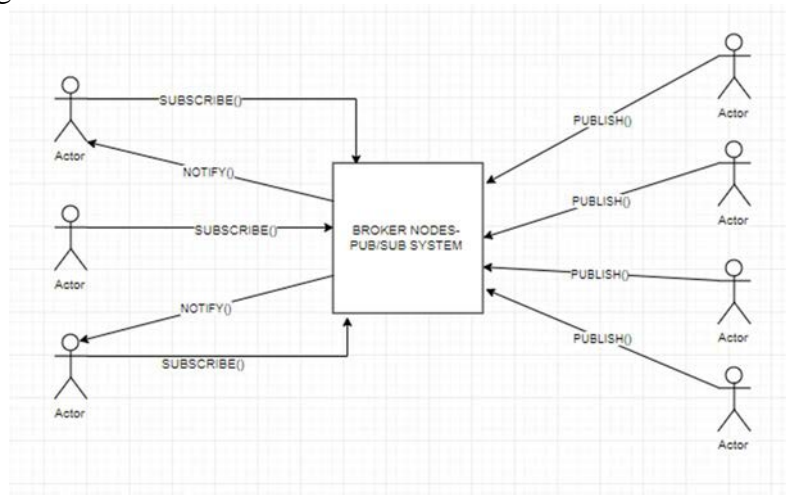
USE CASE DIAGRAM:

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.

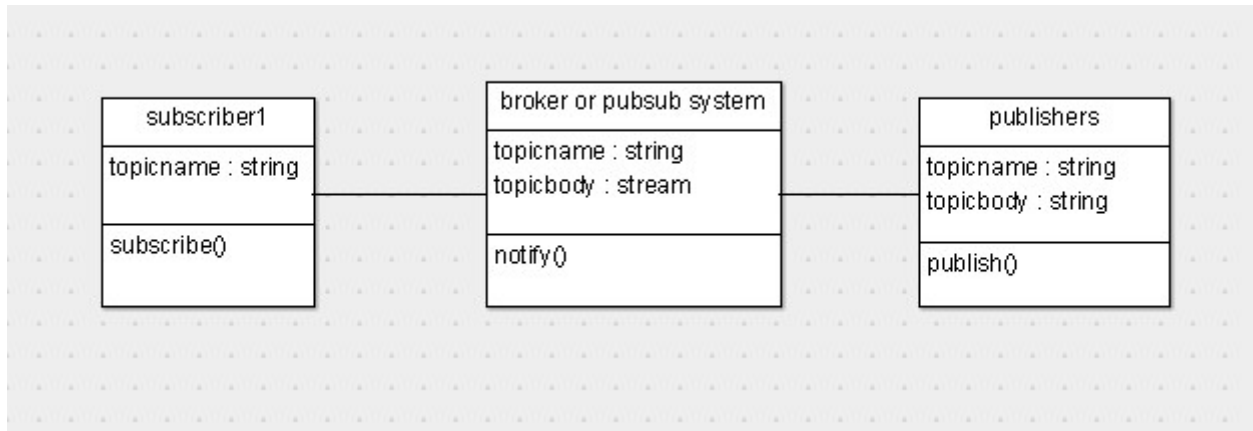


For the PHASE-2 and 3, different subscribers and publishers share content through the pub/sub architecture. The Subscribers subscribe to different topics and publishers publish articles on various topics which are routed to the subscribers through rendezvous routing in phase 3. The Architecture and UML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

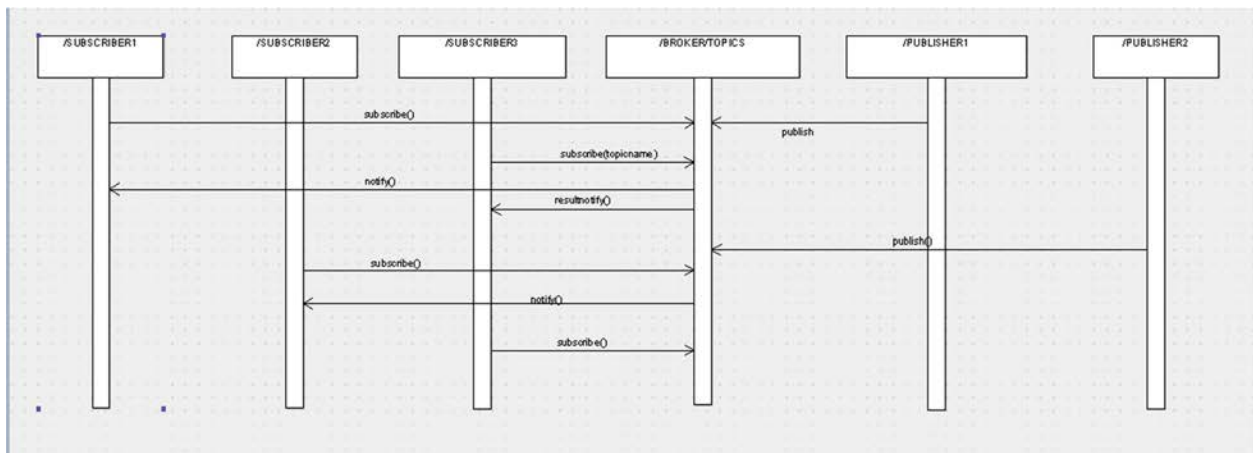
Architecture Diagram:



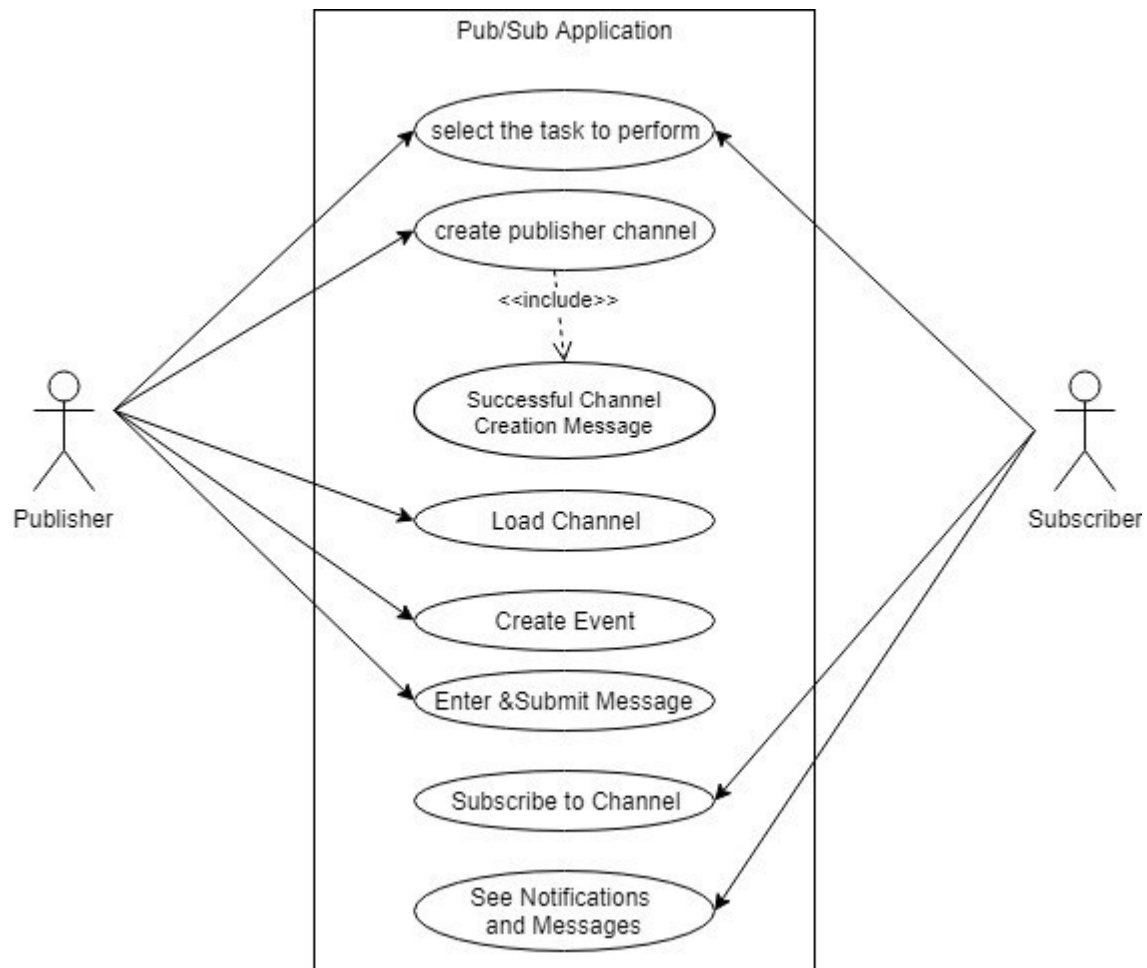
Class Diagram:



Sequence Diagram:

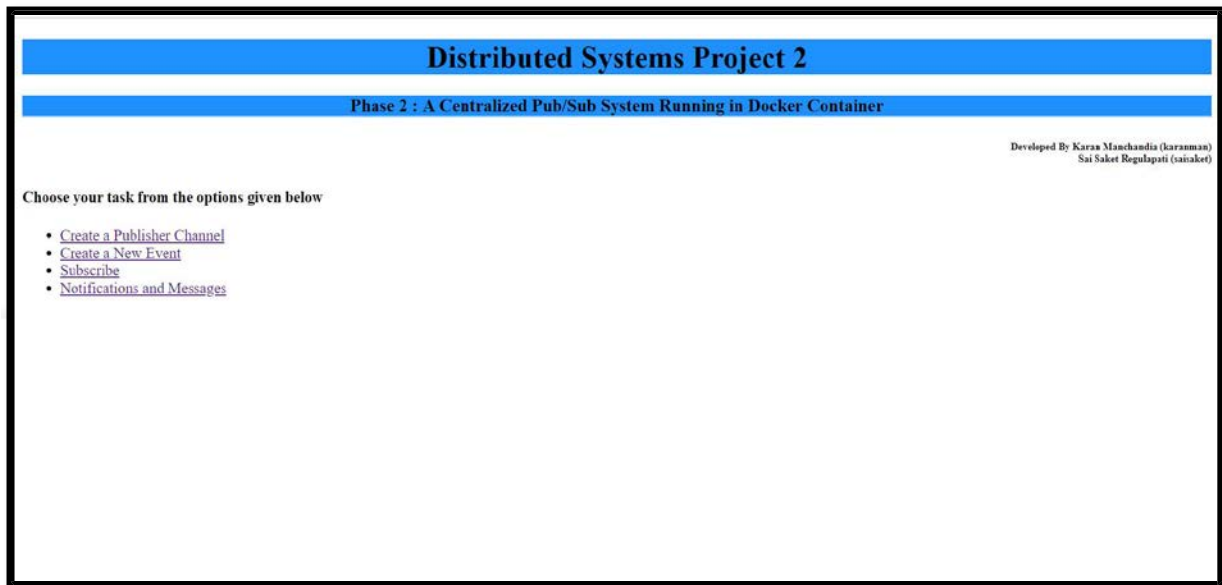


User Case Diagram:

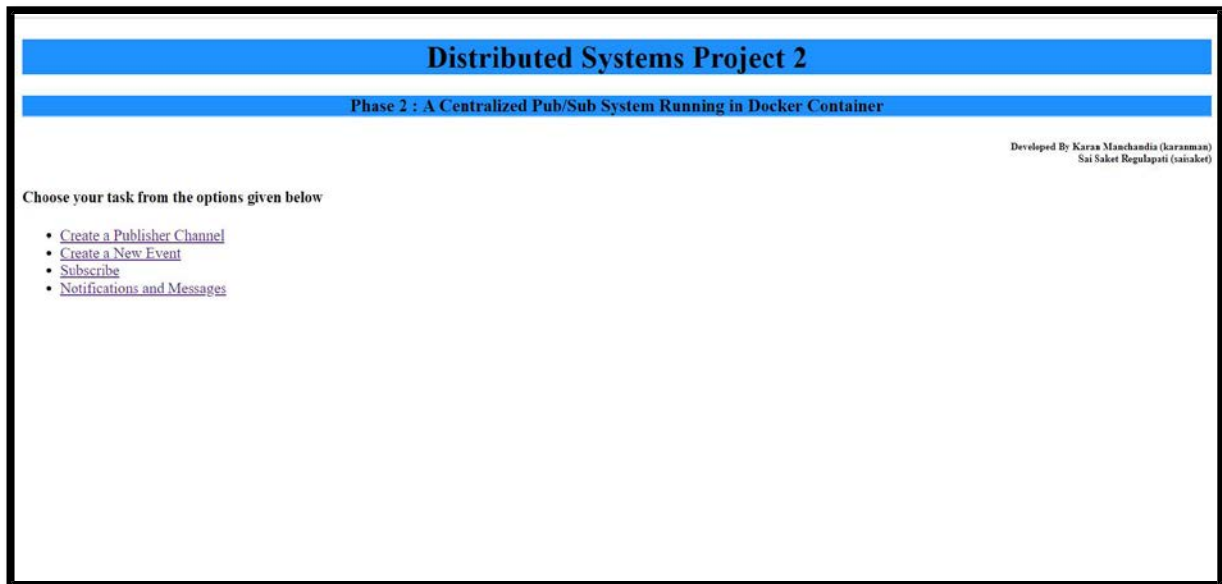


SCREENSHOTS

Home page of the Pub/Sub system.



Webpage to create the topic:



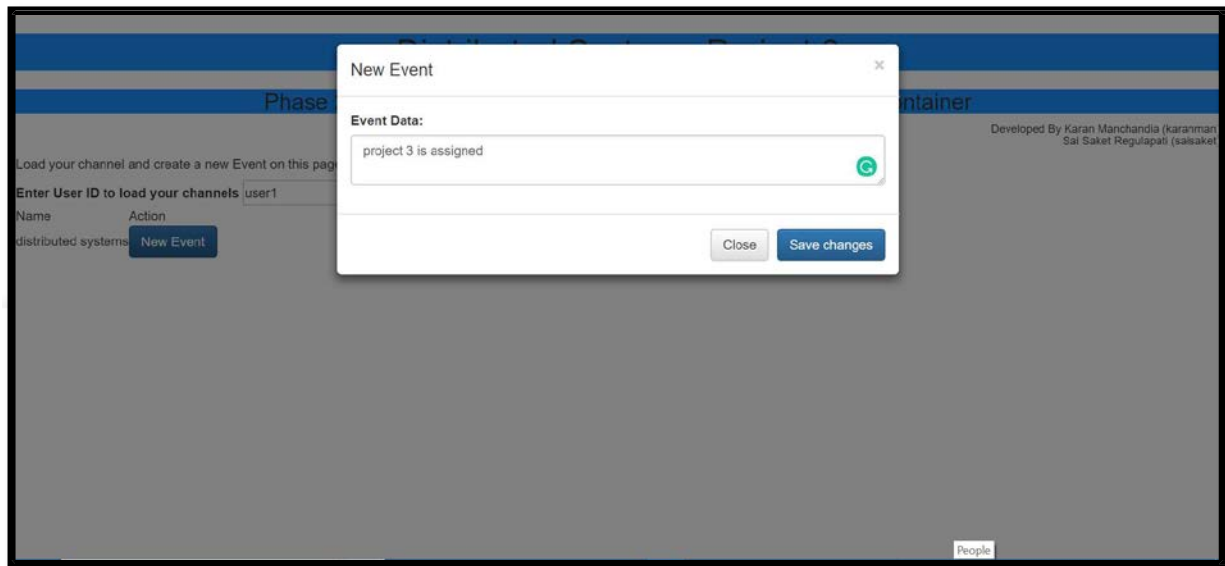
Webpage to Create a new event (publish()):

The screenshot shows a web application titled "Distributed Systems Project 2" with a subtitle "Phase 2 : A Centralized Pub/Sub System Running in Docker Container". The page is developed by Karan Manchandia (karanman) and Sai Saket Regulapati (saisaket). The main heading is "Load your channel and create a new Event on this page". Below this, there is a form with a label "Enter User ID to load your channels" and a text input field containing "user1". To the right of the input field is a "Load" button. Below the input field, there is a table with two columns: "Name" and "Action". The table has one row with the text "distributed systems" under the "Name" column and a "New Event" button under the "Action" column. At the bottom right of the page, there is a "People" button.

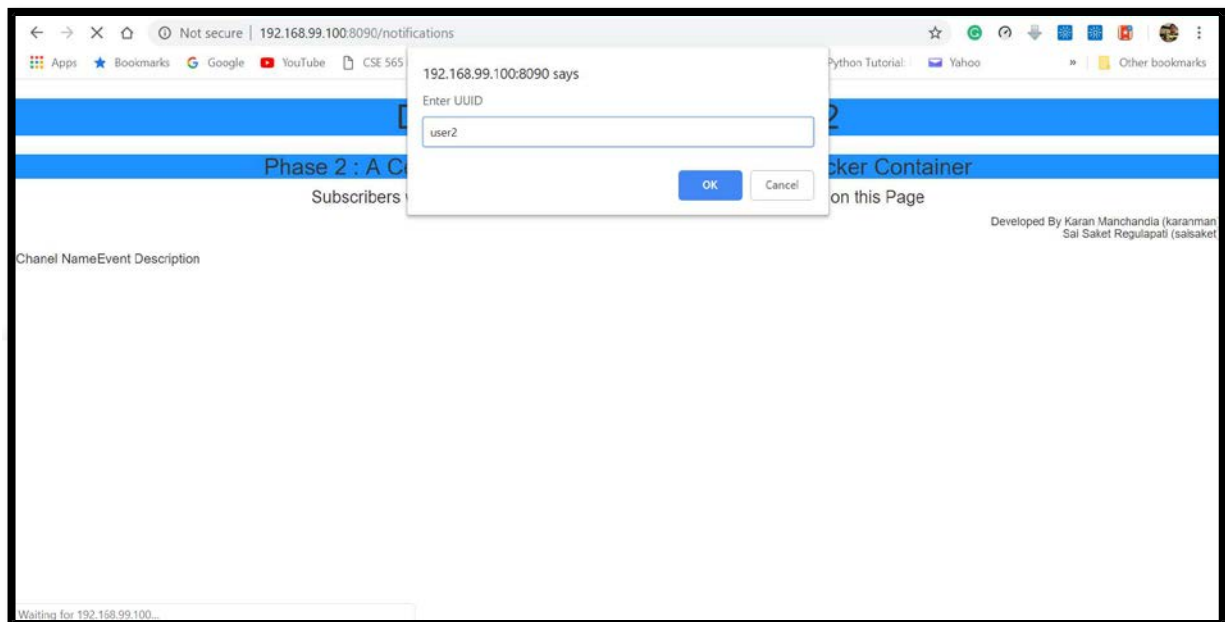
Webpage for the subscribers to subscribe:

The screenshot shows a web application titled "Distributed Systems Project 2" with a subtitle "Phase 2 : A Centralized Pub/Sub System Running in Docker Container". The page is developed by Karan Manchandia (karanman) and Sai Saket Regulapati (saisaket). The main heading is "Subscribe to an event on this page". Below this, there is a form with a label "Enter your client User ID" and a text input field containing "user2". Below the input field, there is a table with three columns: "Chanel Name", "Chanel Owner", and "Action". The table has one row with the text "distributed systems" under the "Chanel Name" column, "user1" under the "Chanel Owner" column, and a "Subscribe" button under the "Action" column. At the bottom right of the page, there is a "People" button.

A Publisher publishing an event:



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



Subscriber getting the notification:

The screenshot shows a web application titled "Distributed Systems Project 2" with a subtitle "Phase 2 : A Centralized Pub/Sub System Running in Docker Container". The main heading is "Subscribers will be notified and can Read Messages from Publishers on this Page". On the right, it says "Developed By Karan Manchandia (karanman) Sai Saket Regulapati (saisaket)". On the left, there is a table with two columns: "Chanel Name" and "Event Description". The table contains one row with the text "distributed systemsproject 3 is assigned".

Chanel Name	Event Description
distributed systemsproject 3	distributed systemsproject 3 is assigned

Creating a new Publisher:

The screenshot shows the same web application as above, but with a different heading: "Enter a User ID and Create your Channel". On the right, it says "Developed By Karan Manchandia (karanman) Sai Saket Regulapati (saisaket)". On the left, there is a form with the following fields and buttons:

- User Identifier:
- Enter Chanel Name:
-

Subscriber subscribing to Publishers:

The screenshot shows a web application titled "Distributed Systems Project 2" with a subtitle "Phase 2 : A Centralized Pub/Sub System Running in Docker Container". The main heading is "Subscribe to an event on this page". On the right, it says "Developed By Karan Manchandia (karanman) Sai Saket Regulapati (saisaket)". On the left, there is a form to "Enter your client User ID" with the value "user2". Below this is a table with columns "Chanel Name", "Chanel Owner", and "Action".

Chanel Name	Chanel Owner	Action
iss	user3	Subscribe
iss	user3	Subscribe
distributed systems	user1	Subscribe

At the bottom left, there is a URL bar showing "192.168.99.100:8090/adverts_form#".

Publisher writing an event (publishing):

The screenshot shows the same web application as before, but with a "New Event" modal open. The modal has a title "New Event" and a close button. Below the title is a text input field labeled "Event Data:" containing the text "there is a iss hold on your student account". To the right of the input field is a red circle with the number "2". At the bottom of the modal are two buttons: "Close" and "Save changes". In the background, the web application shows a form to "Enter User ID to load your channels" with the value "user3". Below this is a table with columns "Name" and "Action".

Name	Action
iss	New Event
iss	New Event

Subscriber getting notifications:

Distributed Systems Project 2

Phase 2 : A Centralized Pub/Sub System Running in Docker Container

Subscribers will be notified and can Read Messages from Publishers on this Page

Developed By Karan Manchandia (karanman)
Sai Saket Regulapati (saisaket)

Chanel Name	Event Description
distributed systems	project 3 is assigned
iss	there is a iss hold on your student account
iss	there is a iss hold on your student account
iss	there is a iss hold on your student account
iss	there is a iss hold on your student account
iss	there is a iss hold on your student account
iss	there is a iss hold on your student account
iss	there is a iss hold on your student account
iss	there is a iss hold on your student account
iss	there is a iss hold on your student account

7 new notifications

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

Distributed Systems Project 2

Phase 2 : A Centralized Pub/Sub System Running in Docker Container

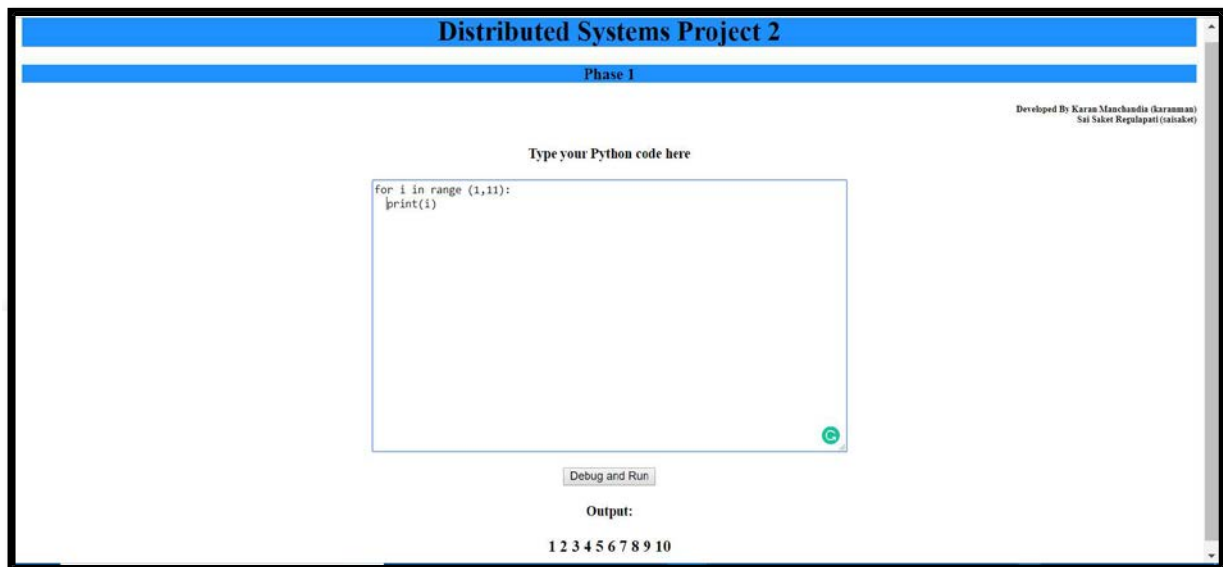
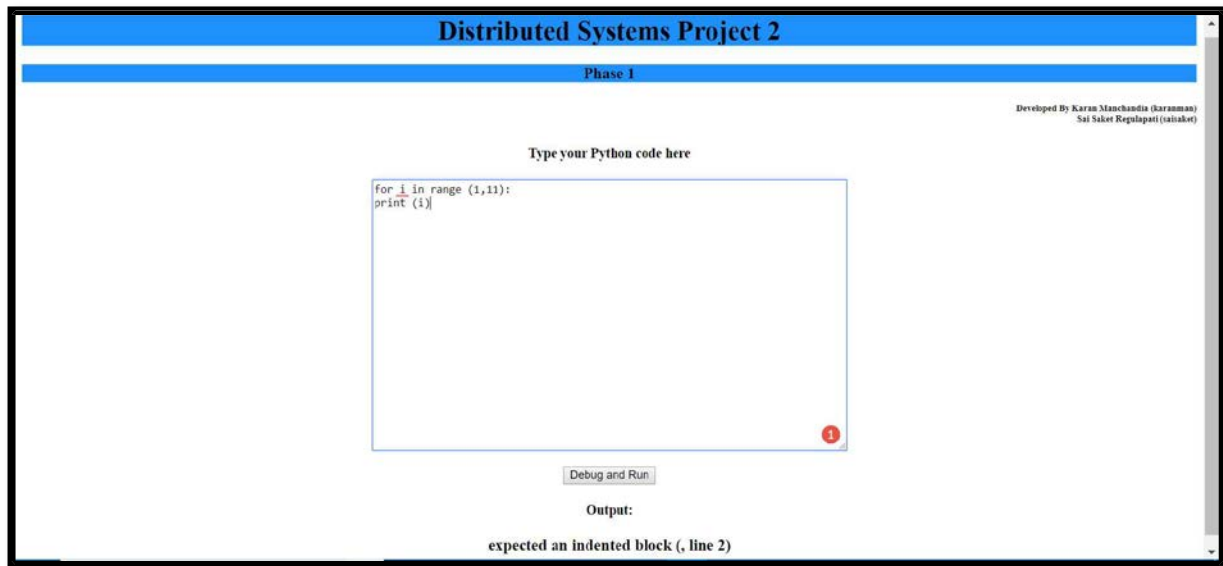
Subscribe to an event on this page

Developed By Karan Manchandia (karanman)
Sai Saket Regulapati (saisaket)

Enter your client User ID

Chanel Name	Chanel Owner	Action
iss	user3	Subscribe
iss	user3	Subscribe
distributed systems	user1	Subscribe

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/>