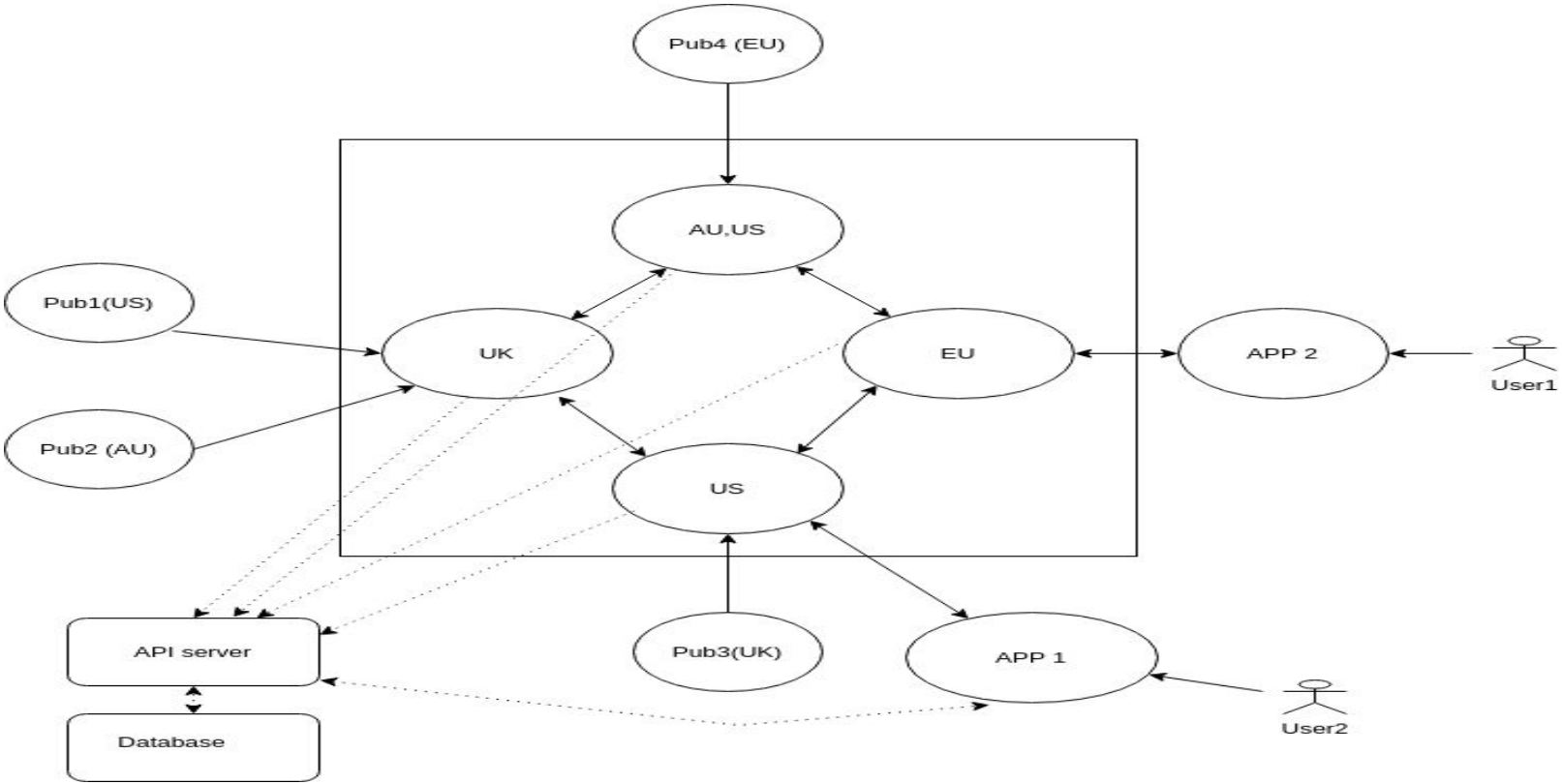
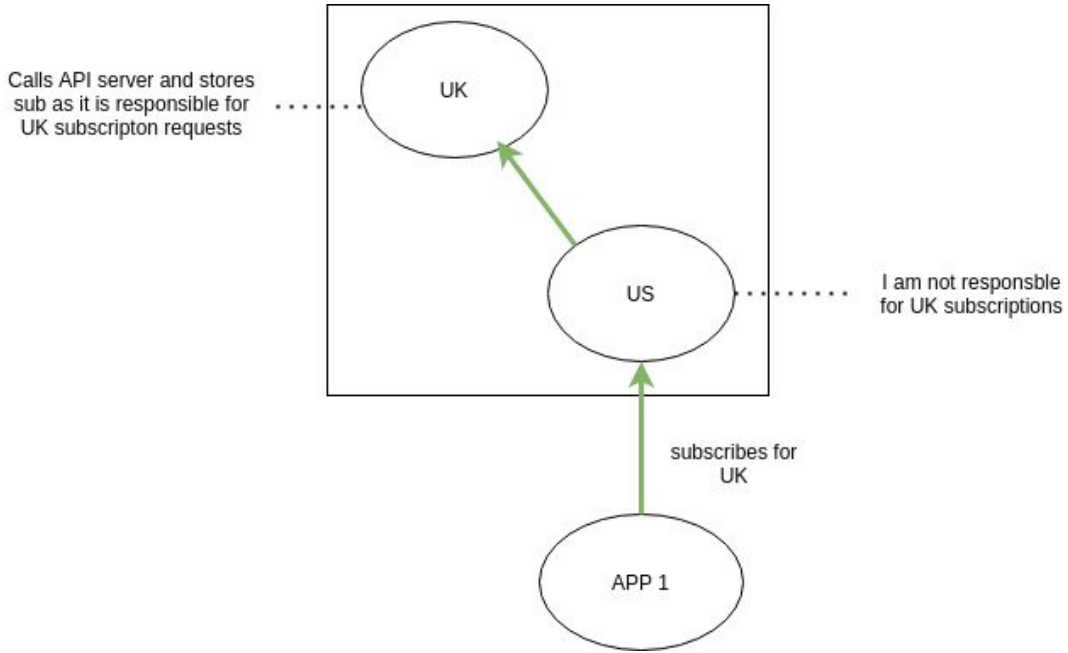
Distributed Pub Sub

Architecture

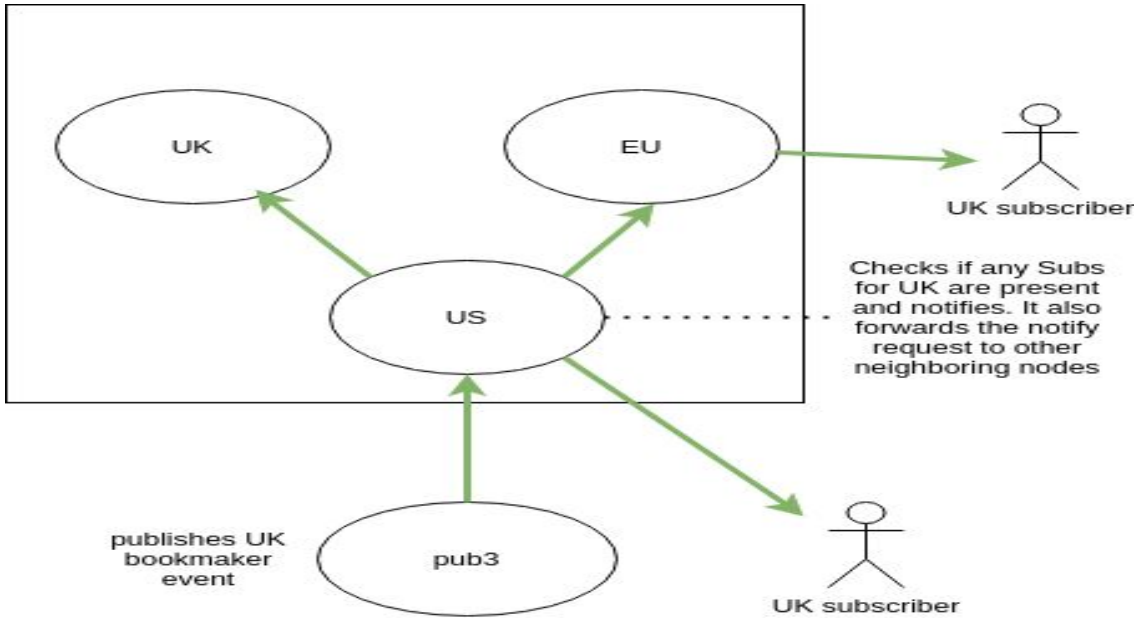


UK broker is a direct neighbor to US broker.

US broker can keep track of what their neighbor nodes handle and pass message only to UK in this case



Dynamic handling of message notification



Tech Stack

Architecture and Tech stack of the Distributed pub sub .

- I have utilized a total of 12 containers. 4 for publishers (API fetchers), one for Database, one for Api server(Express JS server), four for socket server(Broker nodes), and 2 for servers that serve the content to users(React App). Broker nodes takes care of all the socket related operations. The Api server interacts with the database to manage things such as authentication, and also save topics that have been advertised by the publishers. The publishers publish betting odds of bookmakers from four different regions. I have used the region as topics. The four different topics are EU bookmakers, US bookmakers, AU bookmakers and UK bookmakers.
- The publishers advertise their topics to the broker nodes. Then these topics are added to the Mongo database through the API server.
- The subscribers can subscribe to the topics that were advertised by the publishers using the Multi-select dropdown.
- Whenever a broker node receives a subscribe or unsubscribe request, it checks if it is responsible for the subscribe request. If yes, it calls the API server and saves the subscription. Otherwise, it forwards the subscribe request to its neighboring nodes
- Whenever a broker node receives a publish request. It checks if any direct subscribers are subscribed for that particular topic. If yes, it notifies those subscribers. Then, it forwards the notify request to neighboring nodes.

How to Run?

Run `sudo docker-compose build` from root of the project to build all the images. Run `sudo docker-compose up` from root to start all the containers using docker-compose