Marmara University

CSE4074 Computer Networks Project

Students:

Mustafa Yanar – 150118048

Yiğit Göksel – 150119053

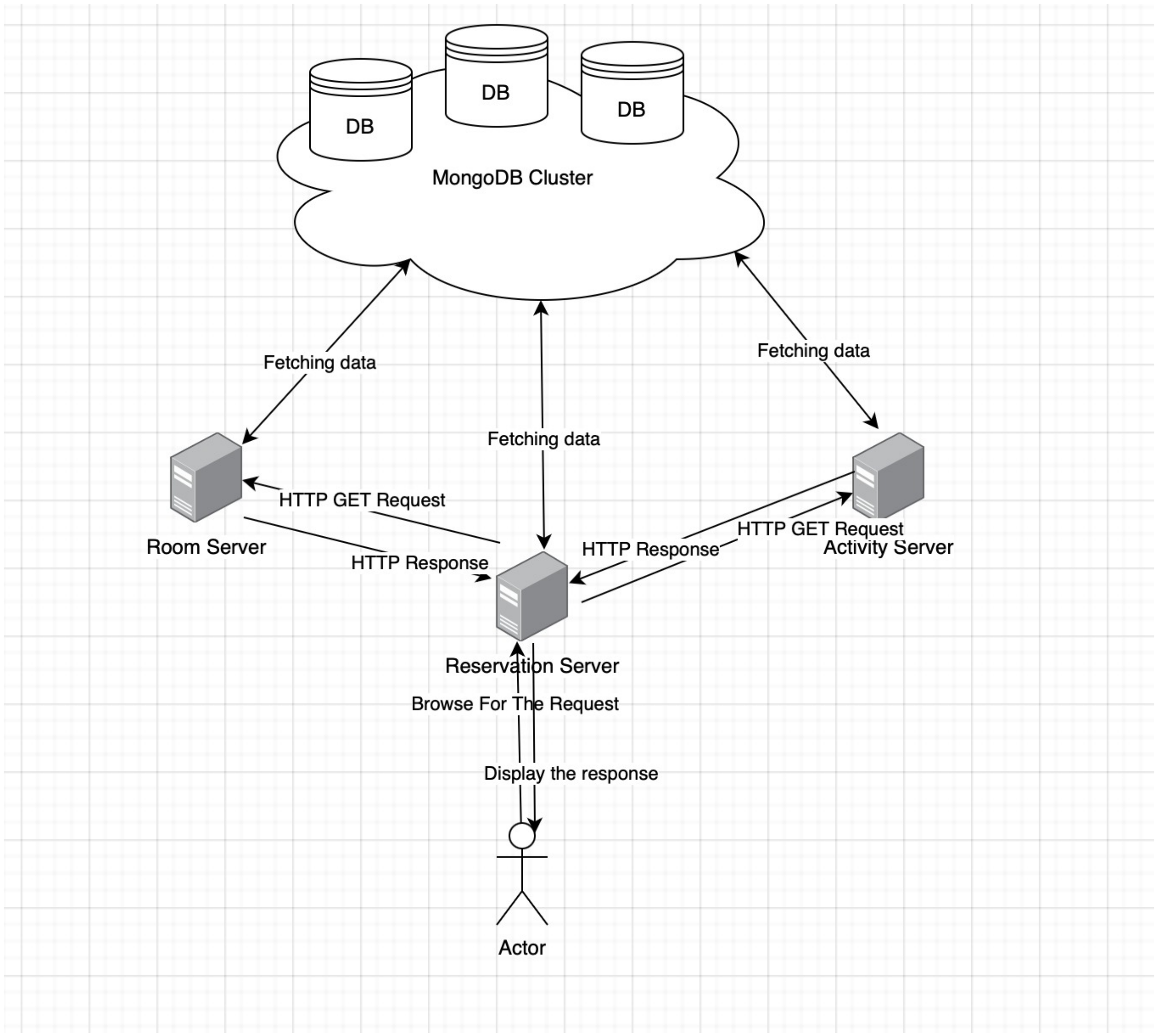
Emir Said Haliloğlu – 150119039

Instructor: Ömer Korçak

Introduction:

We have made 3 servers to simulate a reservation backend system. Room server is on port number 8080. It is responsible for creating room, removing room, reserving room for given activity name and date information and list availability of the room for given day. Activity server is on port 8081. It is responsible for creating activity, removing activity, and checking activity existence. And the main server which user will be use in front end is reservation server. It is on port number 8082; it has all required request to reserve a server such as listing available days and display details of reservation.

Design Document:



This picture shows how our system works.

We used java programming language to do this assignment. And MongoDB cluster account to keep data.

Implementing

Room Server:

The RoomServer class is a server that listens for incoming client connections on a specified port and processes HTTP requests received from clients. When a client connects to the server, it reads the HTTP request and determines which action to take based on the endpoint and request parameters included in the request. The server connects to a MongoDB database to store and retrieve information about rooms, reservations, and other data.

The isBetween and isBetweenWithDuration methods are helper methods that check whether a given value is within a certain range. The getDay method converts an integer value representing a day of the week into the corresponding day of the week as a string.

The main method of the RoomServer class listens for incoming client connections and processes each request in a loop. It reads the HTTP request from the client, parses the request to determine the endpoint and request parameters, and then connects to the MongoDB database. Depending on the endpoint specified in the request, it performs the appropriate action, such as adding a room to the database, removing a room from the database, making a reservation for a room, or checking the availability of a room. The server sends an appropriate HTTP response back to the client based on the outcome of the request.

Activity Server:

The ActivityServer class is a server that listens for incoming client connections on a specified port and processes HTTP requests received from clients. When a client connects to the server, it reads the HTTP request and determines which action to take based on the endpoint and request parameters included in the request. The server connects to a MongoDB database to store and retrieve information about activities and other data.

The main method of the ActivityServer class listens for incoming client connections and processes each request in a loop. It reads the HTTP request from the client, parses the request to determine the endpoint and request parameters, and then connects to the MongoDB database. Depending on the endpoint specified in the request, it performs the appropriate action, such as adding an activity to the database, removing an activity from the database, or checking the availability of an activity. The server sends an appropriate HTTP response back to the client based on the outcome of the request.

Reservation Server:

The ReservationServer class is a server that listens for incoming client connections on a specified port and processes HTTP requests received from clients. When a client connects to the server, it reads the HTTP request and determines which action to take based on the endpoint and request parameters included in the request. The server connects to a MongoDB database to store and retrieve information about reservations and other data.

The main method of the ReservationServer class listens for incoming client connections and processes each request in a loop. It reads the HTTP request from the client, parses the request to determine the endpoint and request parameters, and then connects to the MongoDB database. Depending on the endpoint specified in the request, it performs the appropriate action, such as reserving a room for a specific activity on a specific day and time or checking the availability of a room. The server sends an appropriate HTTP response back to the client based on the outcome of the request.

This server communicates with the other servers by their socket. It requests and listen the response of that servers.

For activity server:

```
Socket activitySocket = new Socket("localhost", 8081);
```

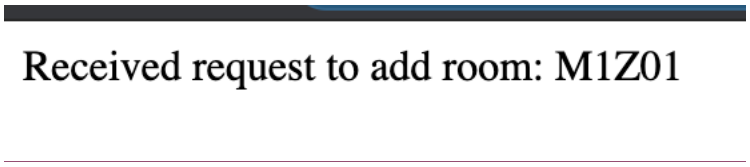
For room server:

```
Socket roomSocket = new Socket("localhost", 8080);
```

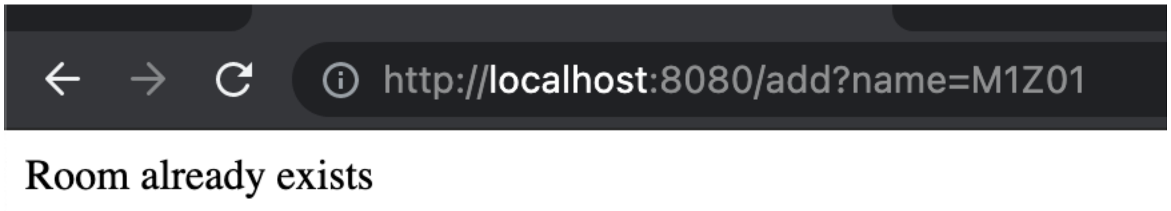
Sample Output Screenshots:

Room Server:

http://localhost:8080/add?name=M1Z01

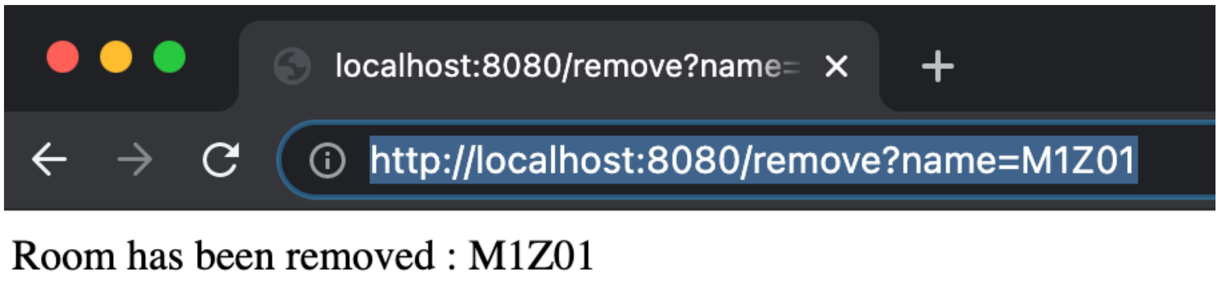


Received request to add room: M1Z01



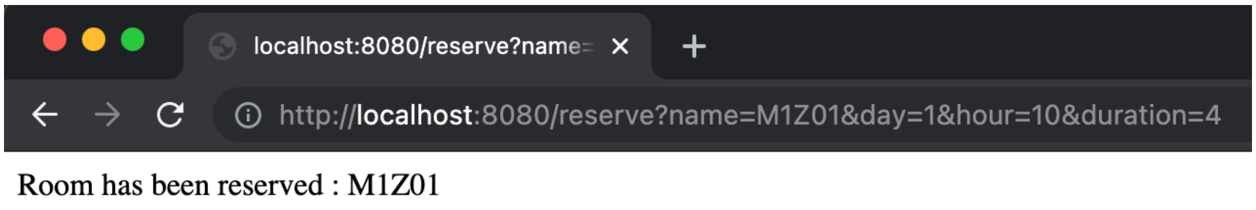
http://localhost:8080/add?name=M1Z01

Room already exists



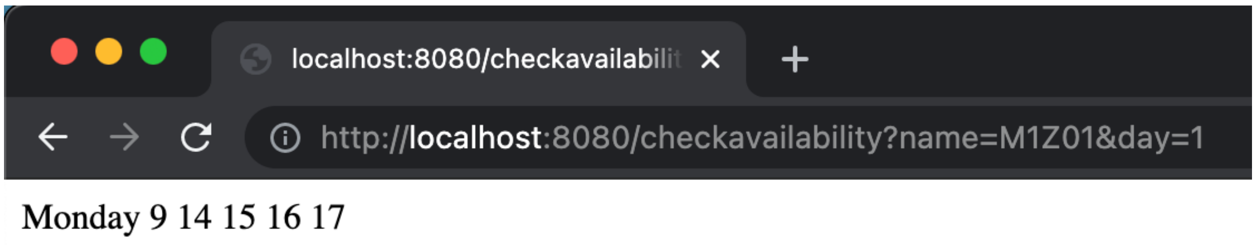
http://localhost:8080/remove?name=M1Z01

Room has been removed : M1Z01



http://localhost:8080/reserve?name=M1Z01&day=1&hour=10&duration=4

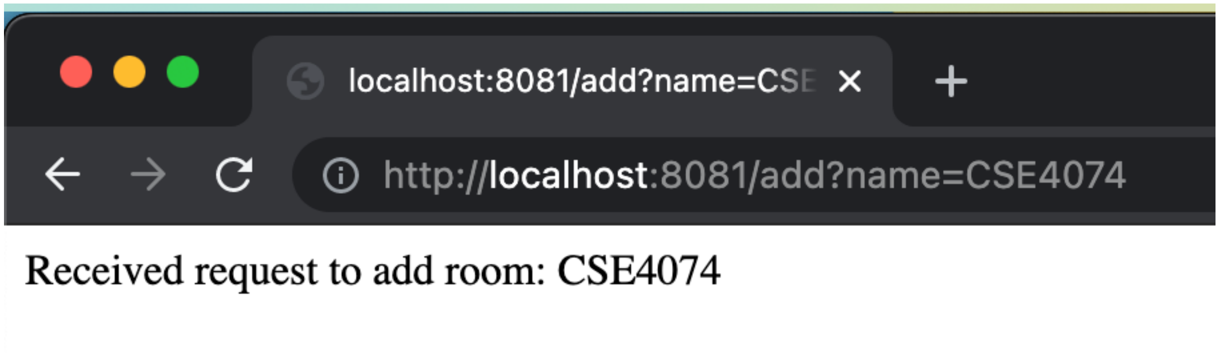
Room has been reserved : M1Z01



http://localhost:8080/checkavailability?name=M1Z01&day=1

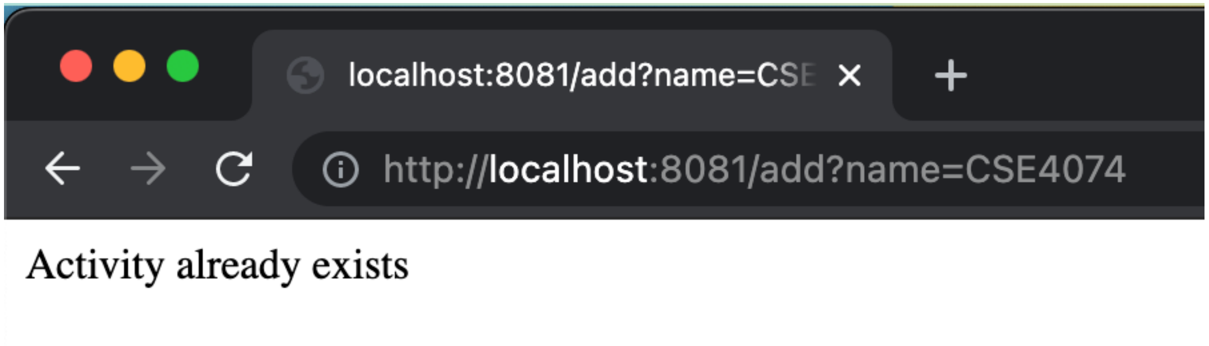
Monday 9 14 15 16 17

Activity Server:


localhost:8081/add?name=CSE

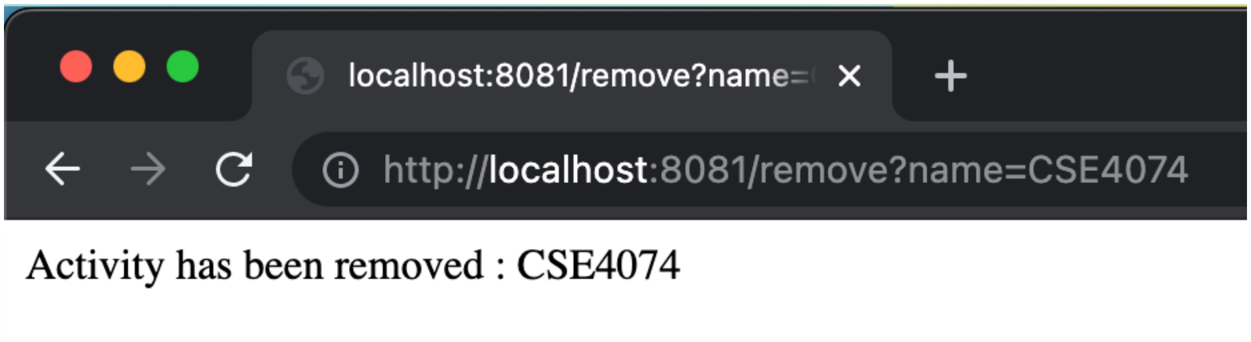
http://localhost:8081/add?name=CSE4074

Received request to add room: CSE4074


localhost:8081/add?name=CSE

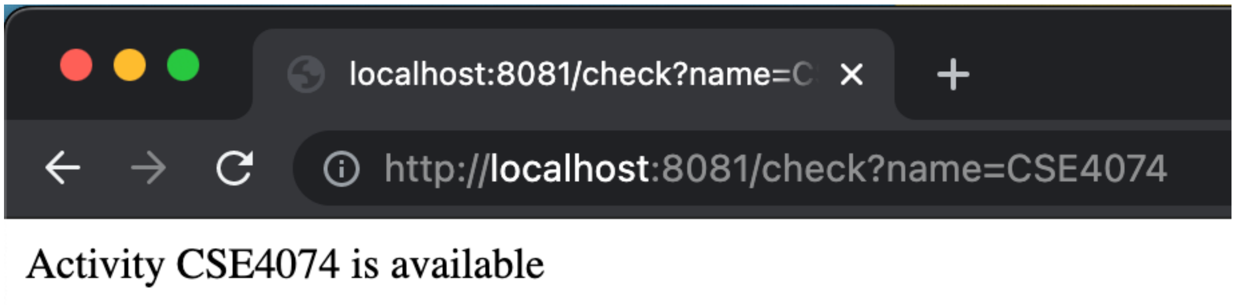
http://localhost:8081/add?name=CSE4074

Activity already exists


localhost:8081/remove?name=

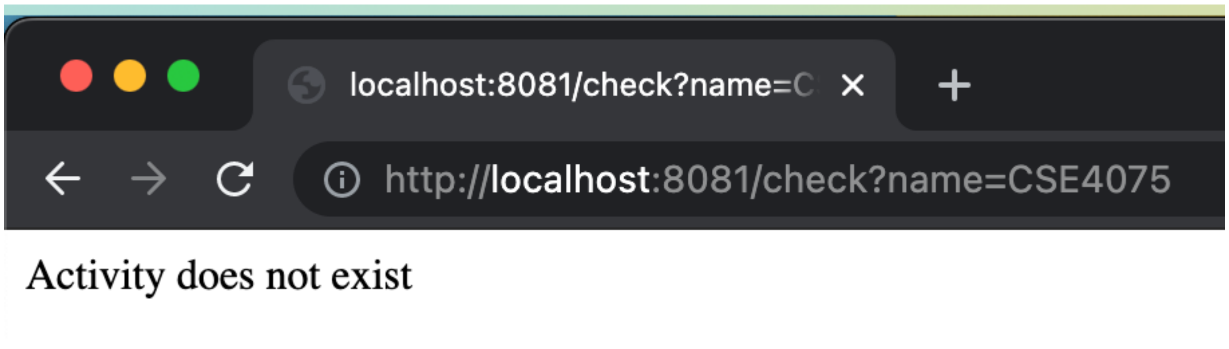
http://localhost:8081/remove?name=CSE4074

Activity has been removed : CSE4074


localhost:8081/check?name=C

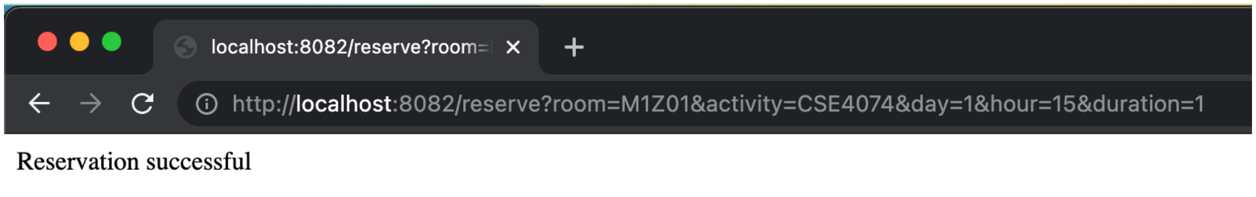
http://localhost:8081/check?name=CSE4074

Activity CSE4074 is available

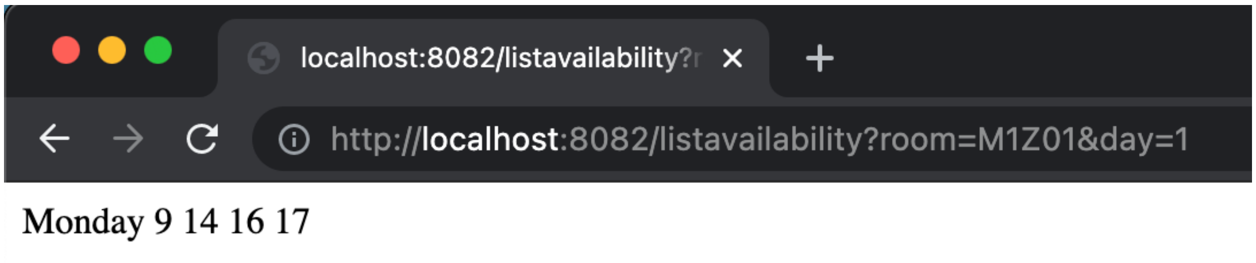

localhost:8081/check?name=C
http://localhost:8081/check?name=CSE4075

```
Activity does not exist
```

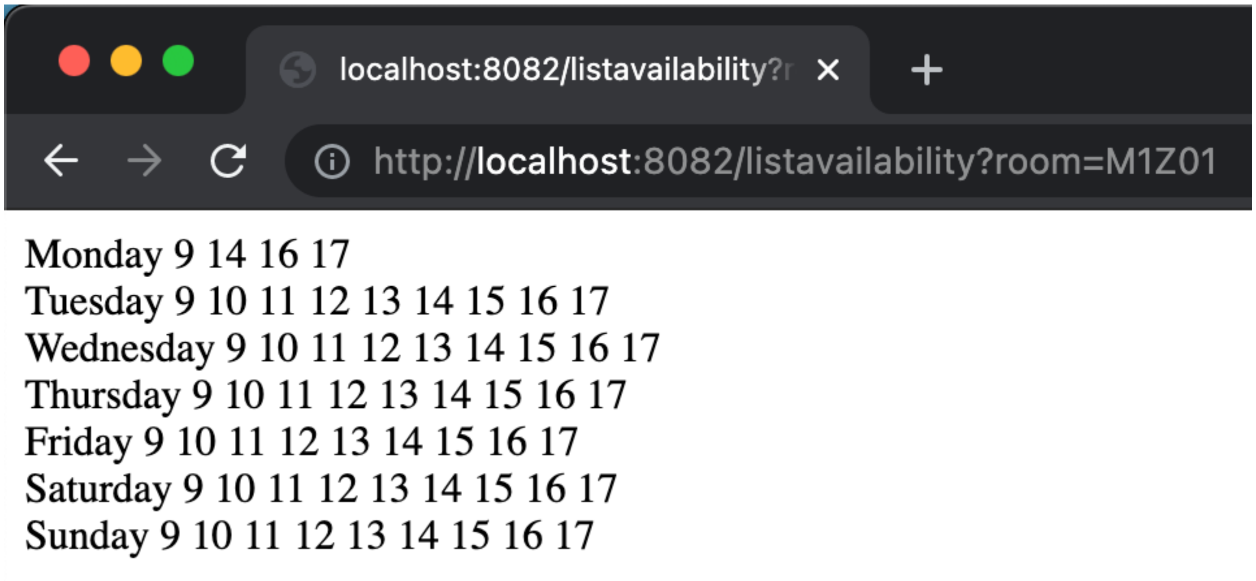
Reservation Server:


localhost:8082/reserve?room=
http://localhost:8082/reserve?room=M1Z01&activity=CSE4074&day=1&hour=15&duration=1

```
Reservation successful
```


localhost:8082/listavailability?r
http://localhost:8082/listavailability?room=M1Z01&day=1

```
Monday 9 14 16 17
```


localhost:8082/listavailability?r
http://localhost:8082/listavailability?room=M1Z01

```
Monday 9 14 16 17
Tuesday 9 10 11 12 13 14 15 16 17
Wednesday 9 10 11 12 13 14 15 16 17
Thursday 9 10 11 12 13 14 15 16 17
Friday 9 10 11 12 13 14 15 16 17
Saturday 9 10 11 12 13 14 15 16 17
Sunday 9 10 11 12 13 14 15 16 17
```