The Publisher

Difficulty	Medium
Skills	API, Architecture, Backend, Data Structures, documentation, git, github, laravel, php
Property	

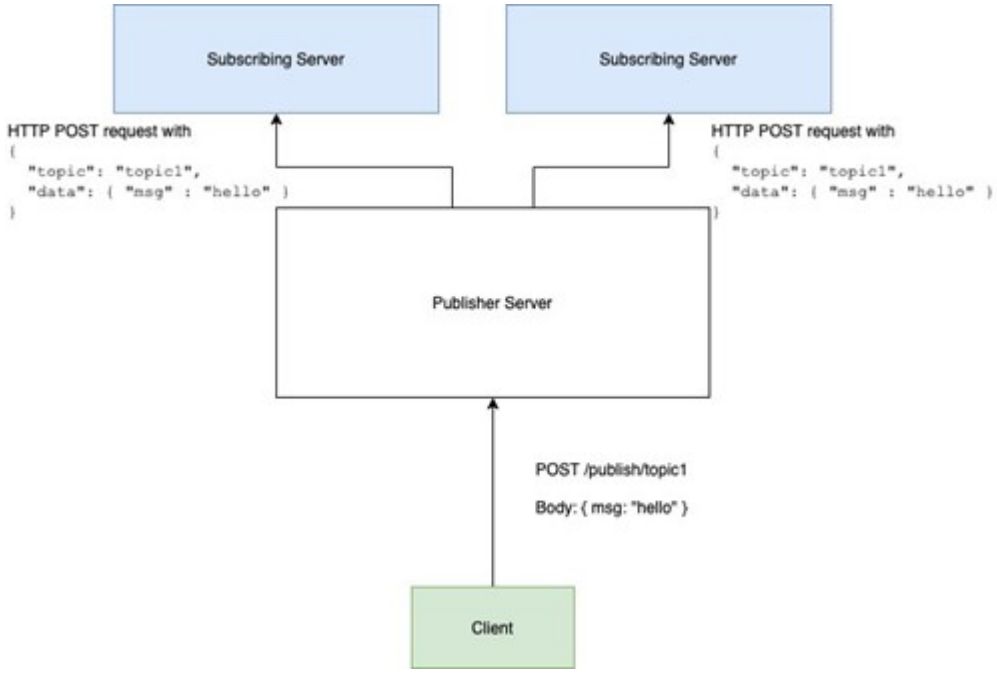
Description

Write a description for the interview question here.

For this challenge we'll be creating a HTTP notification system.

A server (or set of servers) will keep track of topics->subscribers where a topic is a string and a subscriber is an HTTP endpoint. When a message is published on a topic, it should be forwarded to all subscriber endpoints.

Sample Flow



Publisher Server Endpoints

Create a subscription

`POST /subscribe/{topic}`

Expected Body

Success Response

```
{
    url:string,
    topic:string
}
```

Example Request & Response

```
POST /subscribe/topic1
// body
{
    url: "http://mysubscriber.test"
}
```

Response

```
{
    url: "http://mysubscriber.test",
    topic: "topic1"
}
```

Publish Message to Topic

`POST /publish/{topic}`

POSTing to this endpoint should send HTTP requests to any current subscribers for the specified {topic} . It is valid
to publish to topics with no subscribers. If there are multiple subscribers they should all be notified.

Expected Body

```
// must be a javascript object {}, it can contain any keys and have nested data
{
    [key: string]: any
}
```

Expected Response

Should give a meaningful HTTP response code based on whether the publish was successful or not

Payload Sent to Subscribers

```
{
    topic: string
    data: object // whatever data was sent in the publish body
}
```

Testing

Write a test that starts the publisher server on port 8000 and another
server is running on port 9000 (subscriber). The subscriber will be getting data forwarded to it when its corresponding topic is published, which it will then receive and print the data to verify everything is working at the
test1 and test2 endpoints.

Submission

- Create a public GitHub repository and commit your work there.
- Document the setup process of this app in the readme.md file that comes with every fresh Laravel installation. It will be impossible to access your work if this is not done

Bonus Test (Optional, Highly Recommended)

Add more test to test the models are being instantiated as they should.

Bonus Task (Optional, Recommended)

Document API on postman