Laravel Test

1. Service providers are the central place of all Laravel application for registering things. The Laravel app and it's core are registered via service providers.

2. There are already some service providers set by default but you can access them and creating new ones by typing `php artisan make:provider.`

3. Middleware is like a filter for HTTP requests. When you call one of them after a route, this one will check if a condition is validated. If it is, it will continue to the set route, if not, it will redirect you.

4. A Laravel event is an observer. An event lets you fire a set of listeners that are separated from one another. Like the other part of the test, an event could be a detection of a sent email, and the listeners could be: send a Slack message telling that the mail has been sent, or a automated response for a received mail.

5. Eloquent and Query Builder are used together mostly in the Controllers. Eloquent allows the user to use a model to refer to a table, so it simplifies the use of Query Builder when you have to fetch data from the database.

6. The Relationships listed on Eloquent are:
    a. One To One
    b. One To Many
    c. One To Many (Inverse)
    d. Many To Many (needs an extra table many_many)
    e. Has One Through (has 3 tables connecting one another)
    f. Has Many Through (has 3 tables but only 2 connecting one another)

7. The Polymorphic Relations allows the target model to belong to more than one type of model with only one connection. The Polymorphic Relations are:
    a. One To One
    b. One To Many
    c. Many To Many
8. The Collection is a class that allows working with arrays of data. Easily created using collected(...). (Laravel Collection != Eloquent Collection)
9. `php artisan make:policy NameOfThePolicy`
10. The Model Observers are used to group all the events into a single class. The Model Events are events provided by Eloquent that tells the Observer the model state (retrieving, creating, etc).

11. Depending on the relationship it goes like this:

```
public function model_to_connect()
    {
          return $this->Relationship('App\ModelToConnect');
     }
```

12. Queues allow you to defer the processing of a time consuming task, such as sending an email, until a later time. (Just as JS). Laravel provides a unified API for backend services.

13. Units tests are written from the programmer's perspective. They are use to check if a method is doing their tasks. Functional test are written from the user's perspective. They ensure that the system is functioning as users are expecting it to. 2 separate units tests may work, but when you mash them together something might break.

14. Lifecycle hooks are the defined methods which get executed in a certain stage of the Vue.js object lifespan. Vue explains it better: https://es.vuejs.org/images/lifecycle.png. They are methods very similar like the Model Events methods of Laravel that tells you the model state.

15. Events on Vue.js are triggered using v-on: or @. The v-on:event or @event accept the names of methods so they can be triggered.