```
    return $this->belongsTo(User::class);
  }

  protected static function booted()
  {
    static::addGlobalScope(new UserScope());
  }
}
```

And this trait is being used in the `Subscriber` model:

```
class Subscriber extends BaseModel
{
  use HasUser;
}
```

Now anytime you write a subscriber query, a `where user_id = ?` will be added. Let's discuss the advantages and disadvantages of this solution. First, the disadvantages:

- It's hidden. It's magic. Yes, it is.
- It's harder to understand by newcomers. Yes, it is. However, in my experience, if the project has such a critical logic hidden by a scope (or something similar), it will come up in the first hour of the first day.

Now, the advantages:

- If you don't use this scope, you have to write a where statement in **EVERY QUERY** for the next X years. What happens if you forget it? Users will see other users' subscribers. This is a clear violation of the GDPR and possibly some other laws. Please note that there is an $877 million GDPR fine because of a poorly implemented cookie contest or a $255 million one caused by a poor explanation of a data processing practice. $255 million for a wrong description! And, of course, newcomers will forget to filter by user,