

# IBA BOOKING SYSTEM

Peter & Pernille

● PHP 65.6%

● HTML 34.0%

● Vue 0.4%

A horizontal bar chart at the bottom of the slide. It consists of a long thin rectangle divided into three segments. The first segment on the left is dark blue and represents PHP at 65.6%. The second segment in the middle is orange and represents HTML at 34.0%. The third segment on the right is very thin, dark blue, and represents Vue at 0.4%. Each segment has a small colored circle next to its label and percentage.

# Case

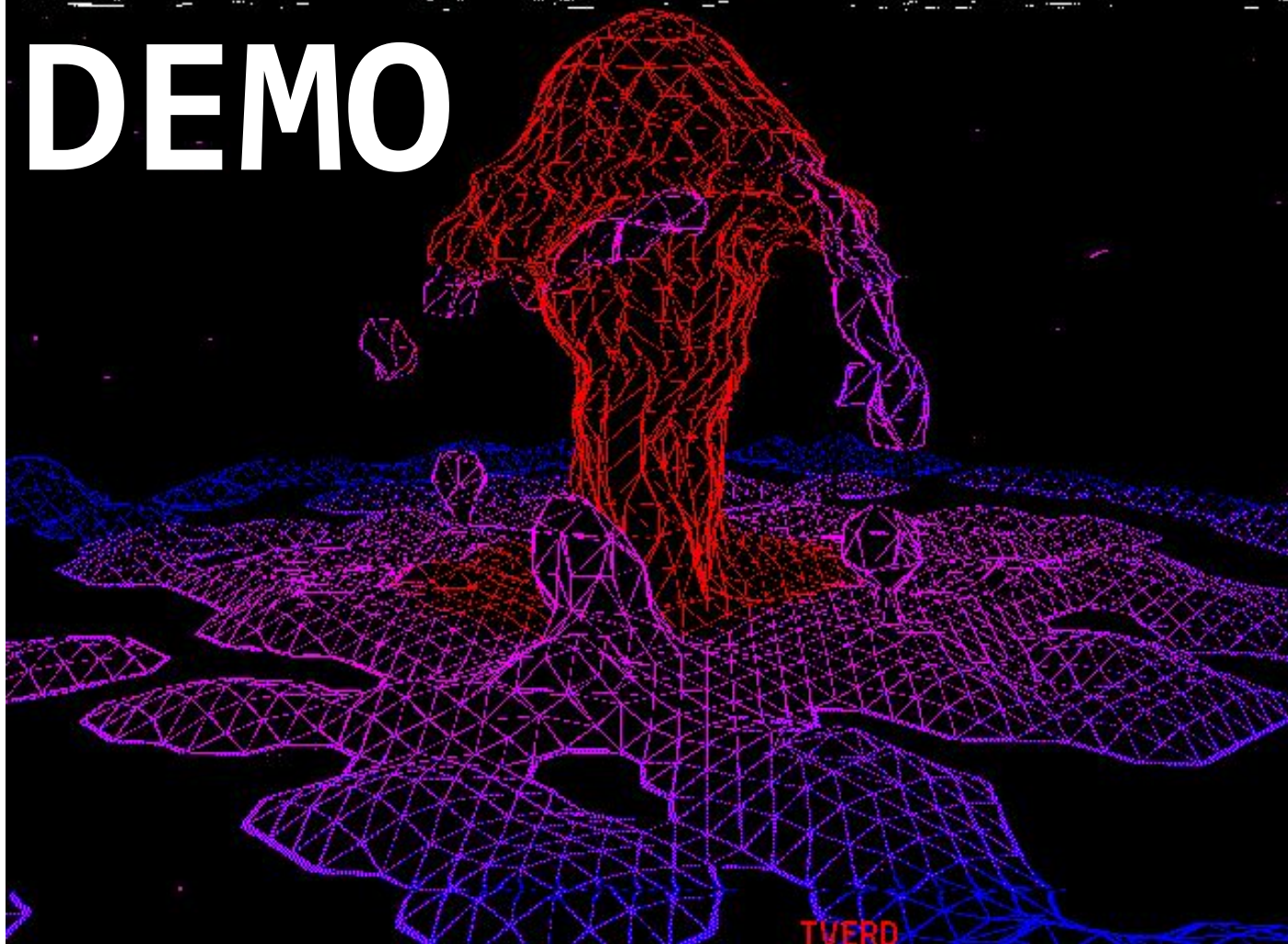
*Hvordan virker vores booking system?*

Oprette Rekvirenter Lokaler/Klasser

Reservationer baseret på rekvirenter & lokaler

Konfigurere lokaler (ULTRA-beta)

# DEMO



# Rekvirent Controller

```
class RekvirentController extends Controller
{
    public function index()
    {
        $rekvirenter = Rekvirent::all();
        return view('rekvirent.index', compact('rekvirenter'));
    }

    public function create() {
        return view('rekvirent.create');
    }

    public function store()
    {
        Rekvirent::create(
            request()->validate([
                'name' => ['required', 'min: 2'],
                'initialer' => ['required'],

            ]));
        return redirect('/rekvirent');
    }
}
```

```
public function show(Rekvirent $rekvirent)
{
    return view('rekvirent.show', compact('rekvirent'));
}

public function edit(Rekvirent $rekvirent)
{
    return view('rekvirent.edit', compact('rekvirent'));
}

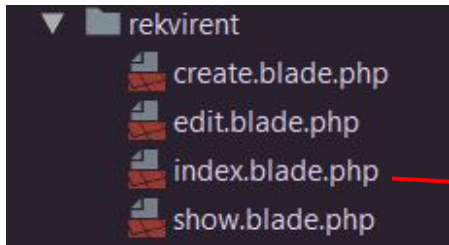
public function update(Rekvirent $rekvirent)
{
    $rekvirent->update(request(['name', 'initialer']));

    return redirect('/rekvirent');
}

public function destroy(Rekvirent $rekvirent)
{
    $rekvirent->delete();

    return redirect('/rekvirent');
}
}
```

# Rekvirent View




```
@extends('layouts.app')

@section('content')
<main class="show">
<center>
  <h1>Rekvirenter</h1>
  <table>
    <tr>
      <td><b>Navn</b></td>
      <td><b>initialer</b></td>
      <td><b>Type</b></td>
      <td><b>Oplysninger</b></td>
      <td><b>Rediger</b></td>
    </tr>
    @foreach($rekvirenter as $rekvirent)
      <tr>
        <td>{{ $rekvirent->name }}</td>
        <td>{{ $rekvirent->initialer }}</td>
        <td>{{ $rekvirent->type }}</td>
        <td><a href="/rekvirent/{{ $rekvirent->id }}"><i class="fas fa-info-circle"></i></td>
        <td><a href="/rekvirent/{{ $rekvirent->id }}/edit"><i class="fas fa-edit"></i></a></td>
      </tr>
    @endforeach
  </table>
</center>
</main>
@endsection
```

# Forside view

```
<div class="card">
  <table>
    <tr>
      <td><b>Lokale</b></td>
      <td><b>Reservations dato</b></td>
      <td><b>Rekvirent</b></td>
    </tr>
    @foreach (App\Reservation::all() as $reservation)
      <tr>
        <td>{{ $reservation->lokale }}</td>
        <td>{{ $reservation->datotid }}</td>
        <td>{{ $reservation->rekvirent }}</td>
      </tr>
    @endforeach
  </table>
</div>
```

## Dashboard



Velkommen Qwerty

Slettede Reservationer

Nylige Reservationer

Lokale	Reservations dato	Rekvirent
4.01	0123-03-12 12:03:00	nml
4.02	1995-03-12 13:00:00	mmd
4.03	1995-03-12 13:00:00	QWERT
4.03	2019-03-12 09:00:00	Dal
4.03	2019-12-20 13:00:00	ytq
4.03	2020-12-20 13:00:00	PM
4.04	2019-02-20 13:00:00	HDD
4.05	2019-09-19 13:00:00	ssd
4.08	2019-08-30 16:00:00	dal
SQL	2019-02-14 14:30:00	SQL
109	2019-02-20 13:00:00	MMQ
dds	2019-02-20 13:00:00	DDS
9.05	2019-02-14 14:00:00	oh



# MVC - Routes ~ web.php

I laravel kan man nemt lave CRUD. Man kan lave en controller med de metoder som CRUD skal bruge ved hjælp af artisan commanden:

```
php artisan make:controller NewController --resource
```

```
Route::get('/', function () {  
    return view('welcome');  
});  
Auth::routes();  
Route::get('/home', 'HomeController@index')->name('home');  
Route::resources([  
    'lokale' => 'LokaleController',  
    'reservation' => 'reservationController',  
    'person' => 'PersonController',  
    'rekvirent' => 'RekvirentController',  
    'config' => 'ConfigController',  
]);
```

Actions Handled By Resource Controller

Verb	URI	Action	Route Name
GET	/photos	index	photos.index
GET	/photos/create	create	photos.create
POST	/photos	store	photos.store
GET	/photos/{photo}	show	photos.show
GET	/photos/{photo}/edit	edit	photos.edit
PUT/PATCH	/photos/{photo}	update	photos.update
DELETE	/photos/{photo}	destroy	photos.destroy