KU LEUVEN

Faculteit Industriële Ingenieurswetenschappen
Projectlab bachelor elektronica-ICT
J. Lannoo, L. Espeel

Verslag E-ink room reservation display

31/05/2019

Baptiste Pattyn, Michel Dequick, Stijn Declerck, Ine Vanderhaeghe

E-ink room reservation display

1 Inhoud

Inhoudsopgave

2 Probleemstelling

We willen een display maken voor klaslokalen die dynamisch kan veranderen per uur. Op het scherm moeten verschillende zaken komen: het nummer van het klaslokaal, de datum, op welke uren het lokaal bezet is op die dag, welk vak op dit moment gegeven wordt en de docent die dit vak geeft.

We gebruiken hiervoor een E-ink display omdat dit het meest energiezuinig is. Dit komt omdat de display enkel voeding nodig heeft om het scherm te veranderen. We moeten het scherm maar om het uur aanpassen, dus alle tijd daartussen heeft een E-ink display geen voeding nodig. Een ander voordeel van een E-ink display is dat het geen licht uitzendt, maar het reflecteert licht zoals een blad papier. Hierdoor is het gemakkelijker leesbaar, ook als er in de omgeving veel licht is.
Een E-ink display bestaat uit kleine gebieden die dipolen zijn (deze worden gebruikt als de pixels van de afbeelding). De positieve kant van de gebieden bestaat uit wit plastiek, de negatieve kant is zwart plastiek. Deze gebieden bevinden zich in een bubbel van olie zodat ze gemakkelijk kunnen omdraaien, en zitten tussen transparante elektrode lagen. Wanneer nu op die elektrode lagen een spanning wordt gezet, kan bepaald worden welke delen van het scherm zwart zijn, en welke wit. Op die manier kan een zwart-wit afbeelding op de display geprogrammeerd worden.

Om dit te realiseren zijn er moeten we met verschillende dingen rekening houden.

We moeten een E-ink display implementeren. Hierbij moeten we bekijken als hier burn-in of ghosting kan optreden. Ook moeten we rekening houden met de grootte van de memory in de display.

Er moet een databank opgezet worden waar alle data van de lokalen in opgeslagen is. Deze databank moet bereikbaar zijn via wifi.

Er moet een connectie gemaakt worden via een wifi module van de databank naar de display.

3 Aanpak

Eerst hebben we alle deelopdrachten op een rijtje gezet en een planning gemaakt. Daarbij hebben we ook een taakverdeling gemaakt.

Dit zijn de grootste taken:

Een library maken voor de display – Michel
Eerst moesten we bekijken hoe we een afbeelding op de display konden krijgen. Op het internet vonden we een voorbeeld van een E-ink display die gemaakt was met een Arduino. Wij maken gebruik van een mbed, dus we hebben de code moeten aanpassen. Als eerste hebben we geprobeerd om een afbeelding op de display te zetten en om een afbeelding die er op staat te kunnen wissen. Daarna hebben we een ontwerp gemaakt van wat we allemaal op het scherm wilden zetten en waar. We hebben dan ook geprobeerd om dit op de display te krijgen.

Een database opstellen met een wifi access point – Baptiste
We hebben een database opgesteld op een Raspberry Pi. Daar hebben we ook een wifi access point op geïmplementeerd zodat de display met de databank kan communiceren. Eens de databank opgevuld was konden we testen als we de data konden in een tabel zetten per lokaal.

Wifi connectiviteit maken – Stijn (en Ine)
Het is belangrijk om via wifi te kunnen communiceren tussen de E-ink display en de databank. Hiervoor moesten we onderzoeken hoe een HTTP GET request werkt en het hoe we het zelf konden implementeren in onze toepassing.

Er waren ook enkele kleinere taken.

De databank invullen – Ine
De databank moest opgevuld worden met fictieve data over de lokaalbezetting zodat we een proof of concept konden maken om voor te stellen op de presentatie van ons project.

Het verslag en de powerpointpresentatie opstarten – Ine
Om het verslag te maken heeft iedereen het deel ingevuld waar hij zelf meest aan gewerkt heeft aangevuld.

4 Implementatie

4.1 Database en Wifi Access Point

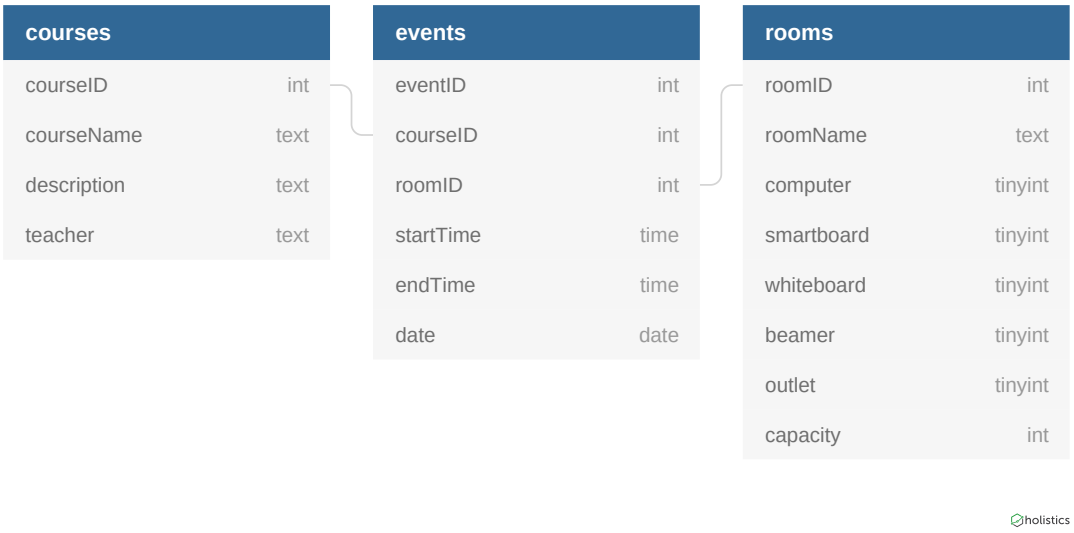
Voor het opslaan van de gegevens is er een database opgezet op een Raspberry Pi model B+ door gebruik te maken van phpMyAdmin. Hier draait een SQL server op waarin alle gegevens voor de komende events van verschillende lokalen kunnen opgeslagen worden. Verder zijn alle gegevens van de courses er ook opgeslagen en de algemene info van de rooms. Om ervoor te zorgen dat onze MBED toegang kan krijgen tot deze database hebben we een Wifi Access Point (WAP) op de Raspberry Pi gezet, waarmee de MBED zich kan verbinden via een ESP8266 Wifi chip. Het opvragen van de gegevens door de mbed gebeurt via een GET request op de HTTP poort van de Raspberry Pi (poort 80).

4.1.1 Database

Deze database genaamd roomdb bestaat uit 3 verschillende tables: courses, events en rooms. De structuur van de database en de relaties tussen de verschillende tables staat afgebeeld in figuur 2. Op de Raspberry Pi staan verschillende PHP files die ervoor zorgen dat de webserver kan verbinden met de database. Een belangrijke file hierbij is dbConnect.php 1, in deze file staan de gegevens die nodig zijn om te kunnen verbinden met de database. In deze file declareren we verschillende variabelen: `$host`, `$user`, `$passworden` `$databse`. Deze worden dan gebruikt in een `mysqli_connect()` commando om een verbinding op te zetten met de databse. Hierna wordt er nog een controle gedaan om te kijken of de verbinding tot stand is gekomen, indien dit niet het geval is dan verkrijgen we een error. Deze code is enkel bedoeld voor tijdens het opzetten van de webserver en de database en dient verwijderd te worden indien de code definitief in gebruik wordt genomen. Als `$host` kiezen we het local host adres 127.0.0.1 omdat de database zich op de Raspberry Pi zelf bevindt. De `$user` en het `$password` zijn specifiek voor elke database en hangen af van welke user je ingesteld hebt op je SQL server.

```
<?php
$host = "127.0.0.1";
$user = "root";
$password = "password" ;
$database = "roomdb";
$conn = mysqli_connect($host, $user, $password, $database);
// delete when ready to launch
if(!$conn){
die("Connection failed: " . $conn ->connect_error);
}
?>
```

Figuur 1: dbConnect.php



Figuur 2: Database structure

4.1.2 Wifi Access Point op Raspberry Pi

Voor het opzetten van de WAP op de Raspberry Pi hebben we een online step-by-step guide gevolgd [1]. De belangrijkste services die we hiervoor installeren op de Raspberry Pi zijn `hostapd` en `dnsmasq`.

DNSmasq
Deze service is in principe gewoon een DHCP (Dynamic Host Configuration Protocol) server. Deze zal ervoor zorgen dat we dynamisch IP adressen kunnen toekennen op ons netwerk zonder dat we alle apparaten manueel moeten verbinden. In de config file van dnsmasq (die zich onder het volgende path bevindt `/etc/dnsmasq.conf`) kennen we een range aan van IP adressen die onze Wifi kan gebruiken om toe te kennen aan apparaten. Verder bepalen we ook hoelang deze IP adressen kunnen gebruikt worden door de apparaten die zich verbinden met het Wifi netwerk. De config file ziet er dan als volgt uit 3 .

```
interface=wlan0
dhcp-range=192.168.0.11,192.168.0.30,255.255.255.0,24h
```

Figuur 3: dnsmasq.conf

Hostapd
Deze service zorgt ervoor dat er een WAP wordt opgezet. In de config file van hostapd (die zich onder het volgende path bevindt `/etc/hostapd/hostapd.conf`) stel je het SSID en het wachtwoord in van je WAP. Verder stel je hier ook een bridge in tussen je ethernet verbinding en je wlan0 van je Raspberry Pi. Dit zorgt ervoor dat je zowel via ethernet als via de WAP toegang kan krijgen tot je webserver. In deze file zorgen we ervoor dat we steeds op channel 7 zullen zitten van van de Wifi. We stellen ook de Wifi Protected Acces (WPA) versie in en bepalen welke vorm van management we zullen gebruiken voor het WPA wachtwoord. Voor deze opdracht gebruiken we WPA-PSK wat staat voor Pre-Shared Key. Een wachtwoord bestaat uit minimaal 8 en maximaal 63 tekens. De config file ziet er dan als volgt uit 4 .

```
interface=wlan0
driver=nl80211
bridge=br0
hw_mode=g
channel=7
wmm_enabled=0
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
wpa_pairwise=CCMP
ssid=teamzoepertoff
wpa_passphrase=stijnkanniks
```

Figuur 4: hostapd.conf

4.1.3 Webserver en GET request

De webserver die we op onze Raspberry Pi draaien is een apache 2.4 server die de mogelijkheid heeft om samen te werken met onze SQL database. Voor onze applicatie hebben we slechts vier van de zes aanwezige files nodig:

- `dbConnect.php`
- `event.php`
- `room.php`
- `roominfo.php`

De overige twee files (`index.php` en `showTable.php`) zijn er enkel indien we via een webbrowser toegang willen krijgen tot de lokaalbezetting. De inhoud van `dbConnect.php` hebben we hierboven al besproken, dus zullen we er hier niet meer verder op ingaan.

Bezetting Lokaal 02.85

Date	Start Time	End Time	Course Name	Course Description	Teacher(s)
2019-03-27	08:15:00	17:45:00	Projectlab bachelor elektronica-ICT	B-KUL-B3390N	Espeel Ludovic Lannoo Jonas
2019-04-03	08:15:00	17:45:00	Projectlab bachelor elektronica-ICT	B-KUL-B3390N	Espeel Ludovic Lannoo Jonas
2019-04-05	12:30:00	18:45:00	Projectlab bachelor elektronica-ICT	B-KUL-B3390N	Espeel Ludovic Lannoo Jonas
2019-04-05	08:15:00	10:15:00	C programmatie	B-KUL-B3391N	Van Waes Jonas
2019-04-04	13:30:00	15:30:00	Projectlab bachelor elektronica-ICT	B-KUL-B3390N	Espeel Ludovic Lannoo Jonas

Figuur 5: lokaalbezetting lokaal 02.85

5 Resultaten

Wanneer we uit de databank de info van het lokaal 02.85 filteren, krijgen we volgende figuur 5.

6 In het vervolg

tekst.

7 Conclusie

tekst.

Referenties

[1] SurferTim. How to use your raspberry pi as a wireless access point. https://thepi.io/how-to-use-your-raspberry-pi-as-a-wireless-access-point/, 2017. [Online; accessed 13-May-2019].