

Ember.js

Ember.js adalah Framework Javascript berbasis open source yang digunakan pada client-side yang dipakai untuk mengembangkan aplikasi web dan menggunakan pola arsitektur MVC (Model View Controller). Salah satu kelebihan dari web framework ini dibandingkan dengan web framework lainnya adalah kecepatan loading website saat digunakan. Hal ini dikarenakan framework Ember bersifat asinkron yang mana Ember akan memuat lebih sedikit sumber daya eksternal seperti gambar dan data. Ember menggunakan mekanisme DS.store yang berarti saat web dijalankan hal pertama yang di-load adalah JS, Html dan CSS. Hal lain seperti data dari database akan di-load belakangan.

Untuk penginstalasian framework ini terbilang cukup mudah. Pertama tama kita harus menginstall nodeJS. File instalasi dapat didownload pada website resmi mereka, dan install seperti kita menginstall aplikasi biasa. Untuk mengecek apakah nodeJS sudah terinstall atau belum, kita dapat menjalankan *Command* “node -v” untuk melihat versi dari nodeJS.

```
C:\Users\Asus>node -v  
v14.15.4
```

Saat nodeJS terinstall, npm juga otomatis akan terinstall.

```
C:\Users\Asus>npm -v  
6.14.10
```

Selanjutnya kita akan install emberJS melalui terminal dengan menjalankan *Command* berikut

```
C:\Users\Asus>npm install -g ember-cli  
npm WARN deprecated resolve-url@0.2.1: https://github.com/lydell/resolve-url#deprecated  
npm WARN deprecated urix@0.1.0: Please see https://github.com/lydell/urix#deprecated  
npm WARN deprecated @babel/polyfill@7.12.1: ⚠️ This package has been deprecated in favor of separate inclusion of a polyfill and regenerator-runtime (when needed). See the @babel/polyfill docs (https://babeljs.io/docs/en/babel-polyfill) for more information.  
npm WARN deprecated core-js@2.6.12: core-js@3 is no longer maintained and not recommended for usage due to the number of issues. Please, upgrade your dependencies to the actual version of core-js@3.  
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >4 <4.3.1 have a low-severity ReDos regression when used in a Node.js environment. It is recommended you upgrade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)  
C:\Users\Asus\AppData\Roaming\npm\ember -> C:\Users\Asus\AppData\Roaming\npm\node_modules\ember-cli\bin\ember  
+ ember-cli@3.24.0  
updated 3 packages in 102.838s
```

Selanjutnya cek apakah ember sudah terinstall dengan Command berikut

```
C:\Users\Asus>ember -v  
ember-cli: 3.24.0  
node: 14.15.4  
os: win32 x64
```

Setelah itu kita buat sebuah project baru melalui terminal dengan command berikut

```
D:\>cd tesember

D:\tesember>ember new World-Time
installing app
Ember CLI v3.24.0

Creating a new Ember app in D:\tesember\World-Time:
  create .editorconfig
  create .ember-cli
  create .eslintignore
  create .eslintrc.js
  create .prettierignore
  create .prettierrc.js
  create .template-lintrc.js
  create .travis.yml
  create .watchmanconfig
  create README.md
  create app\app.js
  create app\components\.gitkeep
  create app\controllers\.gitkeep
  create app\helpers\.gitkeep
  create app\index.html
  create app\models\.gitkeep
  create app\router.js
  create app\routes\.gitkeep
  create app\styles\app.css
  create app\templates\application.hbs
  create config\ember-cli-update.json
  create config\environment.js
  create config\optional-features.json
  create config\targets.js
  create ember-cli-build.js
  create .gitignore
  create package.json
  create public\robots.txt
  create testem.js
  create tests\helpers\.gitkeep
  create tests\index.html
  create tests\integration\.gitkeep
  create tests\test-helper.js
  create tests\unit\.gitkeep
  create vendor\.gitkeep
```

Maka project baru EmberJS sudah berhasil dibuat.

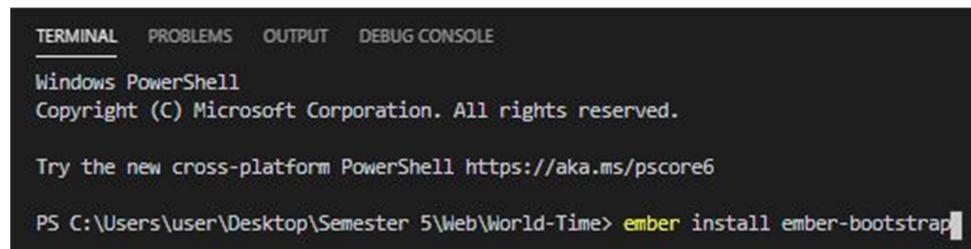
Komponen-komponen website pada framework ini dapat dikerjakan secara terpisah. Komponen ember pada dasarnya adalah sandboxed potongan UI yang dapat digunakan kembali (Reuseable) sehingga lebih efisien bagi pengembang/developer. Pada code tugas kami, komponen tersebut dapat dilihat sebagai berikut:

```
app > templates > ~ application.hbs > ⚡ script > ⚡ show > 📁 Tik > 🎨 constructor
  1   {{page-title "World Time"}}
  2
  3   <div class="col-12 page">
  4     <div class="row">
  5       <DropDown/>
  6       <ButtonMode/>
  7     </div>
  8   </div>
  9   <div id ="loading"></div>
 10   <DateDay/>
 11   <DateHours/>
 12
```

application.hbs merupakan body dari html, disini kita dapat melakukan pengisian html dan script, namun berbeda dengan html biasa, disini setiap bagian dipecah menjadi komponen tertentu, seperti menggunakan class pada suatu block code, sehingga bisa dipanggil kapan dan dimana saja dan bersifat *reusable*. Untuk komponen sendiri terdiri dari DropDown, ButtonMode, DateDay, dan DateHours. Komponen ini di deklarasikan dengan huruf besar pada nama awalan mereka (meskipun ketika kita membuat komponen kita memberikan nama dalam huruf kecil) dan di akhiri dengan “/”, sebagai tambahan untuk memakai komponen tersebut jangan lupa menghilangkan tanda “-“.

```
app > components > drop-down.hbs > BsDropdown > dd.toggle#dropdownMenuButton.position1.btn.btn-outline-light.dropdown-toggle.dropstyle
1  <div class="dropdown col-6">
2    <BsDropdown @tagName="span" as |dd| >
3      <dd.toggle class="position1 btn btn-outline-light dropdown-toggle dropstyle" type="button" id="dropdownMenuButton" data-bs-toggle="dropdown"
4        | Your Location
5        </dd.toggle>
6        <CountryList/>
7      </BsDropdown>
8    </div>
```

Komponen terletak pada folder app > components. Disinilah semua komponen yang reusable diletakkan, penamaan folder disini adalah semua huruf kecil, dan pemisahan kata menggunakan tanda kurang (-), dan untuk pemanggilannya adalah dengan menghilangkan tanda kurang nya dan setiap awalan kata menggunakan kapital dan ditaruh dalam sebuah tag (< />) , misal untuk dropdown, penamaan filenya adalah drop-down.hbs, maka pemanggilannya di application.hbs adalah <DropDown/>. Untuk dropdownnya sendiri menggunakan sebuah komponen yang diinstall dari ember-bootstrap, dimana ember bootstrap merupakan komponen yang disediakan oleh ember.js supaya pemanggilan komponen bootstrap lebih mudah dilakukan. Cara menginstallnya adalah dengan memasukkan *command* berikut



```
TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\user\Desktop\Semester 5\Web\World-Time> ember install ember-bootstrap
```

Setelah itu komponen bisa dipanggil, sama seperti menggunakan framework bootstrap biasa, namun bedanya kita memanggilnya kedalam tag, disini untuk tombol dropdown, kita memanggilnya dengan tag <BsDropdown> kemudian didalamnya kita bisa mengatur isinya, disini untuk itemnya dipisahkan dengan dimasukkan kedalam komponen CountryList.

```

app > components > country-list.hbs > dd.menu.dropdown-menu
1  <dd.menu class="dropdown-menu" aria-labelledby="dropdownMenuButton">
2    <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Eniwetok (GMT - 12)</a></menu.item>
3    <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Samoa (GMT - 11)</a></menu.item>
4    <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Honolulu (GMT - 10)</a></menu.item>
5    <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Alaska (GMT - 9)</a></menu.item>
6    <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">California (GMT - 8)</a></menu.item>
7    <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Alberta (GMT - 7)</a></menu.item>
8    <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Mexico City (GMT - 6)</a></menu.item>
9    <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Santiago (GMT - 4)</a></menu.item>
10   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Ottawa (GMT - 3)</a></menu.item>
11   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Grytviken (GMT - 2)</a></menu.item>
12   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Praia (GMT - 1)</a></menu.item>
13   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">London (GMT)</a></menu.item>
14   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Vienna (GMT + 1)</a></menu.item>
15   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Baghdad (GMT + 3)</a></menu.item>
16   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">New Delhi (GMT + 5)</a></menu.item>
17   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Thimpu (GMT + 6)</a></menu.item>
18   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Jakarta (GMT + 7)</a></menu.item>
19   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Beijing (GMT + 8)</a></menu.item>
20   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Tokyo (GMT + 9)</a></menu.item>
21   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Hagåtña (GMT + 10)</a></menu.item>
22   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Honolulu (GMT + 11)</a></menu.item>
23   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Kingston (GMT + 12)</a></menu.item>
24   <menu.item><a class="dropdown-item" href="#" onclick = "getText(this)">Your Location</a></menu.item>
25 </dd.menu>

```

Di Countrylist kita akan mendefinisikan menu-menu yang bisa dipilih ketika tombol dropdown di klik, dimana setiap pilihan dimasukkan kedalam tag <menu.item> dan diberi atribut onclick untuk melakukan aksi apa yang terjadi jika menu di klik, untuk fungsi getText itu dituliskan didalam application.hbs

```

app > components > button-mode.hbs > ...
1  <div class="col-6">
2    <BsButton type="button" class="position2 btn btn-light" onclick="gantiMode()" id ="mode">Full Mode</BsButton>
3  </div>
4
5
6

```

Komponen selanjutnya adalah ButtonMode, dimana fungsinya adalah untuk menampilkan waktu hanya dalam bentuk jam dan menit, atau lengkap dengan detiknya. Untuk komponen buttonnya juga diambil dari ember-bootstrap dengan tag <BsButton>

```

app > components > date-day.hbs > ...
1  <div class="col-12 position3">
2    <p id = "tampilHari"></p>
3  </div>
4

```

Kemudian komponen DateDay, dimana fungsinya adalah untuk menampilkan hari dan tanggal lengkap sesuai negara yang dipilih di DropDown

```

app > components > date-hours.hbs > ...
1  <div class="col-12 position4">
2    <p id = "waktu"></p>
3  </div>
4

```

Yang terakhir adalah komponen DateHours, dimana fungsinya adalah untuk menampilkan jam dibawah tanggal, kemudian komponen-komponen tadi dipanggil kedalam application.hbs seperti pada foto pertama. Untuk semua fungsi javascriptnya diatur didalam application.hbs

Sebelum membuat tampilan, kami menambahkan perintah onClick. Perintah onClick ini akan dijalankan ketika user melakukan click pada elemen yang diberikan, Pada tugas kami memakai onclick pada button yang mana ketika button di-click function yang berada di dalam onClick akan dijalankan.

```
~ application.hbs ~ date-hours.hbs ~ button-mode.hbs X ~ drop-down.hbs ~ country-list.hbs
app > components > ~ button-mode.hbs > div.col-6 > BsButton#mode.position2.btn.btn-light
You, 4 days ago | 2 authors (You and others)
1 <div class="col-6">
2   <BsButton type="button" class="position2 btn btn-light" onclick="gantiMode()" id ="mode">Seconds Mode</BsButton>
3 </div>
4
5
6
```

pada gambar dibagian button-mode kami memberikan nama fungsi gantiMode() dengan id “mode”, jadi kami juga harus memastikan pada saat membangun function dijavascript, function tersebut memiliki nama yang sama yaitu gantuMode().

```
40 //fungsi ganti mode
41 function gantiMode(){
42   if(document.getElementById("mode").innerText == "Full Mode")document.getElementById("mode").innerText = "Normal Mode"
43   else if(document.getElementById("mode").innerText == "Normal Mode")document.getElementById("mode").innerText = "Full Mode"
44 }
```

function di atas akan menjalankan perintah dimana ketika user meng-click button, secara otomatis kita akan mengambil teks yang ada pada button bedasarkan id button yang telah di deklarasikan diawal (“mode”) ketika teks yang sama ambil lalu saya cocokan adalah seconds mode maka saya hanya tinggal mengubah innertext menjadi “Normal Mode”, begitu pula untuk sebaliknya.

Pertama-tama kita akan coba untuk menampilkan waktu dari perangkat user dengan mendeklarasikan variabel baru untuk menyimpan date pada perangkat user dengan memanggil fungsi new Date(). Untuk menyimpan informasi dari date, kita akan memakai class yang bernama “Tik”, dimana class ini akan menyimpan jam, menit, detik, hari, dsb. Pada gambar dapat kita lihat bahwa bulan dan hari yang kita ambil masih berbentuk angka sementara yang kita inginkan adalah berbentuk teks, maka langkah selanjutnya yang akan kita lakukan adalah mencoba untuk membuat sebuah variabel yang bisa menyimpan nama hari dan bulan kita dalam bentuk string supaya kita mengkonversinya dari angka menjadi bentuk teks dengan cara mengakses array weekday dan month sesuai dengan angka yang kita dapatkan saat kita memanggil misalnya .getMonth()

```

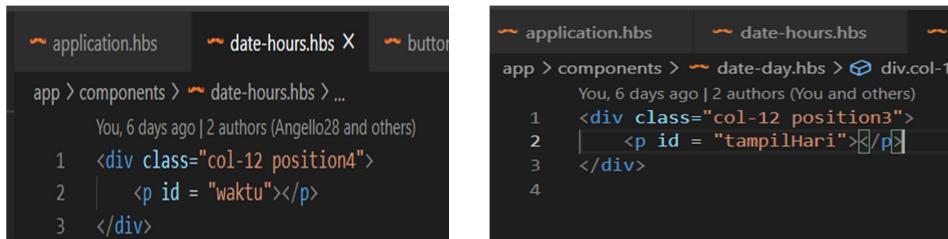
function show() {
    //untuk tanggal, hari dan bulan
    var weekDay = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"];
    var Month = ["January", "February", "March", "April", "June", "July", "August", "September", "October", "November", "December"];
    var date = new Date()
    ...

    //untuk jam default kita
    ...
}

class Tik {
    constructor(detik, menit, jam, tanggal, hari, bulan, tahun) {
        this.detik = detik;
        this.menit = menit;
        this.jam = jam;
        this.tanggal = tanggal;
        this.hari = hari;
        this.bulan = bulan;
        this.tahun = tahun;
    }
}
var jadwal = new Tik(date.getSeconds(), date.getMinutes(), date.getHours(), date.getDate(), date.getMonth(), date.getFullYear())

```

Selanjutnya kita akan menampilkan waktu yang sudah tersimpan dengan memanggil komponen “DateDay” dan “DateHours”. Kita juga memberikan id kepada masing masing komponen, id yang ada pada element p diperlukan membuat tulisan yang berisi jam yang sedang berjalan dari javascript.



Agar jam, menit, dan detik dapat ditampilkan kita akan mengambil id yang ada di element p pada komponen date-hours.hbs Kemudian memasukkan inner text mereka. Innertext mereka pun dimasukan dengan class yang sudah di deklarasikan diawal, pada gambar dibawah ini kita bisa melihat bagaimana weekDay dan Month melakukan konversi dari angka menjadi string seperti yang kita inginkan.

```

//menampilkan hari kita
document.getElementById("tampilHari").innerText = weekDay[jadwal.hari].toString() + " "
+ jadwal.tanggal.toString() + " " + Month[jadwal.bulan] + " "
+ jadwal.tahun.toString()

//menampilkan jam kita sesuai dengan mode yang kita pilih
if(document.getElementById("mode").innerText == "Seconds Mode"){
    document.getElementById("waktu").innerText = jadwal.jam.toString().padStart(2, "0") + ":" 
    + jadwal.menit.toString().padStart(2, "0") + ":" 
    + jadwal.detik.toString().padStart(2, "0")
} else if(document.getElementById("mode").innerText == "Normal Mode"){
    document.getElementById("waktu").innerText = jadwal.jam.toString().padStart(2, "0") + ":" 
    + jadwal.menit.toString().padStart(2, "0")
}

```

Untuk menampilkan background sesuai dengan jam yang sedang berjalan, kita tinggal melakukan percabangan yang mana ketika jam berada dalam range 18 – 23 dan 0 - 6 maka kita akan set background menjadi gelap dan ketika jam kita diluar dari range tersebut maka background menjadi cerah

```

if( (jadwal.jam >=18 && jadwal.jam <= 23) || (jadwal.jam >=0 && jadwal.jam <= 6))
document.body.style.backgroundImage = "url(..assets/images/night.jpg)";
|
else document.body.style.backgroundImage = "url(..assets/images/day.jpg)";

```

Kita akan melakukan konversi jam untuk setiap negara berdasarkan jam yang sedang berjalan di tempat user, kita akan dapatkan terlebih dahulu utc timenya, dimana kita tinggal mengambil waktu yang sedang berjalan (.getTime()) lalu kita tambahkan hasil perhitungan dari offset user yang telah dikalikan dengan 60.000 (milidetik). Setelah utctime telah kita dapat kita akan menambahkan utctime kita dengan offset dari setiap negara yang sebelumnya telah dikalikan dengan 3.600.000 (60 menit * 60000 milidetik) lalu kita simpan kedalam suatu variabel, karena jam untuk setiap negara yang akan disimpan berjumlah banyak, maka disini kita akan menyimpannya dalam bentuk array.

```

53 var utcTime = date.getTime() + (date.getTimezoneOffset() * 60000);
54 var GMT = [new Date(utcTime + (3600000 * obj[0])), new Date(utcTime + (3600000 * obj[1])), new Date(utcTime + (3600000 * obj[2])), new Date(utcTime
55     new Date(utcTime + (3600000 * obj[4])), new Date(utcTime + (3600000 * obj[5])), new Date(utcTime + (3600000 * obj[6])), new Date(utcTime
56     new Date(utcTime + (3600000 * obj[8])), new Date(utcTime + (3600000 * obj[9])), new Date(utcTime + (3600000 * obj[10])), new Date(utcTime
57     new Date(utcTime + (3600000 * obj[12])), new Date(utcTime + (3600000 * obj[13])), new Date(utcTime + (3600000 * obj[14])), new Date(utcTime
58     new Date(utcTime + (3600000 * obj[16])), new Date(utcTime + (3600000 * obj[17])), new Date(utcTime + (3600000 * obj[18])), new Date(utcTime
59     new Date(utcTime + (3600000 * obj[20])), new Date(utcTime + (3600000 * obj[21])), new Date(utcTime + (3600000 * obj[22])), new Date(utcTime
60     new Date(utcTime + (3600000 * obj[24])), ]

```

Dibawah ini merupakan dictionary yang berisi nama setiap negara dan offset dari negara tersebut. ini digunakan untuk mengkonversi waktu user dengan waktu di negara lain. Kita akan membuat sebuah fungsi penyesuaian. Fungsi ini akan mengambil jam menit dan detik, Kemudian memasukanya kembali kedalam variabel jadwal (variabel jadwal adalah class TIK). Function penyesuaian ini di buat diluar function show() kita.

```

78 //jam untuk setiap negara
79 var country = { "Eniwetok (GMT - 12)" : 0, "Samoa (GMT - 11)" : 1, "Honolulu (GMT - 10)" : 2,
80     "Alaska (GMT - 9)" : 3, "California (GMT - 8)" : 4, "Alberta (GMT - 7)" : 5, "Mexico City (GMT - 6)" : 6,
81     "New York (GMT - 5)" : 7, "Santiago (GMT - 4)" : 8, "Ottawa (GMT - 3)" : 9,
82     "Grytviken (GMT - 2)" : 10, "Praia (GMT - 1)" : 11, "London (GMT)" : 12, "Vienna (GMT + 1)" : 13,
83     "Athena (GMT + 2)" : 14, "Baghdad (GMT + 3)" : 15, "Abu Dhabi (GMT + 4)" : 16,
84     "New Delhi (GMT + 5)" : 17, "Thimpu (GMT + 6)" : 18, "Jakarta (GMT + 7)" : 19,
85     "Beijing (GMT + 8)" : 20, "Tokyo (GMT + 9)" : 21, "Hagåtña (GMT + 10)" : 22,
86     "Honolulu" : 23, "Kingston (GMT + 12)" : 24, "Eniwetok" : 0, "Samoa" : 1, "Honolulu" : 2,
87     "Alaska" : 3, "California" : 4, "Alberta" : 5, "Mexico City" : 6, "New York" : 7, "Santiago" : 8, "Ottawa" : 9,
88     "Grytviken" : 10, "Praia" : 11, "London" : 12, "Vienna" : 13, "Athena" : 14, "Baghdad" : 15, "Abu Dhabi" : 16,
89     "New Delhi" : 17, "Thimpu" : 18, "Jakarta" : 19, "Beijing" : 20, "Tokyo" : 21, "Hagåtña" : 22,
90     "Honolulu" : 23, "Kingston" : 24
91 }
92
93 //mengambil waktu
94 if(document.getElementById("dropdownMenuButton").innerText in country){
95     jadwal = penyesuaian(jadwal, GMT[country[document.getElementById("dropdownMenuButton").innerText]]);
96 }

```



```

155 //sesuaikan jam dengan setiap wilayah
156 function penyesuaian(c, value){
157     c.detik = value.getSeconds();
158     c.menit = value.getMinutes();
159     c.jam = value.getHours();
160     c.tanggal = value.getDate();
161     c.hari = value.getDay();
162     c.bulan = value.getMonth();
163     c.tahun = value.getFullYear();
164     return c
165 }

```

Selanjutnya kita akan mengatur innertext yang ada pada dropdown dimana ketika nama suatu negara dipilih yang akan ditampilkan pada Button Dropdown hanya nama negaranya saja tanpa offsetnya. kita akan mengimport jquery terlebih dahulu untuk fungsi kita.

```
15 <script src='http://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js'></script>
16 )<script>...
166 </script>
```

Selanjutnya kita akan membuat fungsi baru bernama function getText(obj) dimana function ini berfungsi untuk melakukan perubahan pada teks kita. Kita akan melakukan perulangan sepanjang teks. Fungsi perulangan diatas untuk mengambil setiap char kemudian menyimpannya ke dalam temp. Sebelum disimpan kedalam temp. Disini kita akan mengecek terlebih dahulu apakah ada spasi, jika tidak ada maka kita akan simpan sampai habis, tetapi jika ada kita akan mengecek apakah ada terdapat kata “GMT” hal ini dilakukan supaya ketika kita mengklik negara tertentu yang muncul hanyalah “nama negara” saja dan “(GMT-n)” nya tidak ikut di tampilkan. Setelah itu kita akan mengecek apakah inner text dari dropdown list ada di dalam dictionary. Jika tertera maka konversi tinggal dijalankan.

```
function getText(obj){
    var t = $(obj).text();
    var temp ='';
    for(var i = 0 ; i < t.length ;i++){
        if(t.charAt(i) != ' '){
            temp+=t.charAt(i);
        }else{
            if(t.charAt(i+2)=='G'&&t.charAt(i+3)=='M'&&t.charAt(i+4)=='T'){
                i += t.length;
                break;
            }else{
                temp+=t.charAt(i);
            }
        }
    }
    document.getElementById("dropdownMenuButton").innerText = temp;
}

if(document.getElementById("dropdownMenuButton").innerText in country){
    jadwal = penyesuaian(jadwal, GMT[country[document.getElementById("dropdownMenuButton").innerText]]);
}
```

Keunggulan lainnya dari framework ini adalah deploying yang mudah dilakukan, yaitu hanya dengan drag and drop ke Netlify. Netlify merupakan layanan hosting Webapp maupun Website yang direkomendasikan oleh situs resmi Ember.js. berikut adalah langkah mudah yang dapat kita lakukan.

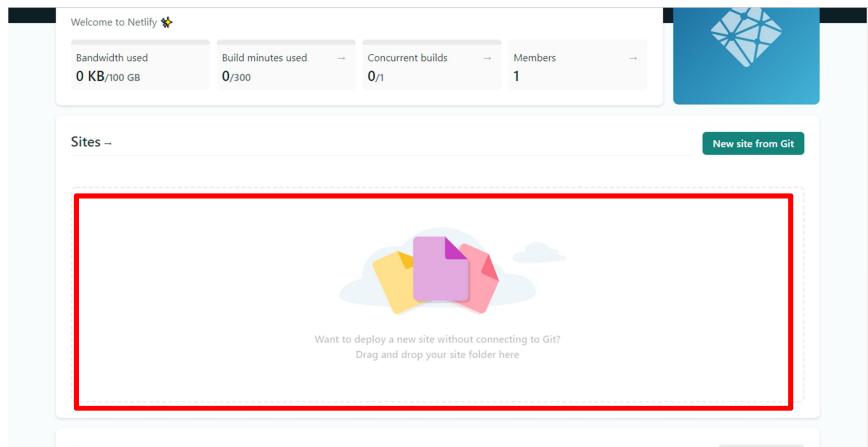
Pertama build environment dari folder dist/ dengan menjalankan command **ember build –environment=production**

```
D:\Mikroskil-S5\SEMESTER-5\Web\World-Time>ember build --environment==production
Running without permission to symlink will degrade build performance.
See https://cli.emberjs.com/release/appendix/windows/ for details.

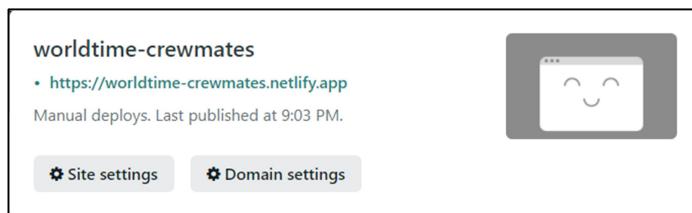
Environment: =production
cleaning up...
Built project successfully. Stored in "dist/".

D:\Mikroskil-S5\SEMESTER-5\Web\World-Time\World-Time>_
```

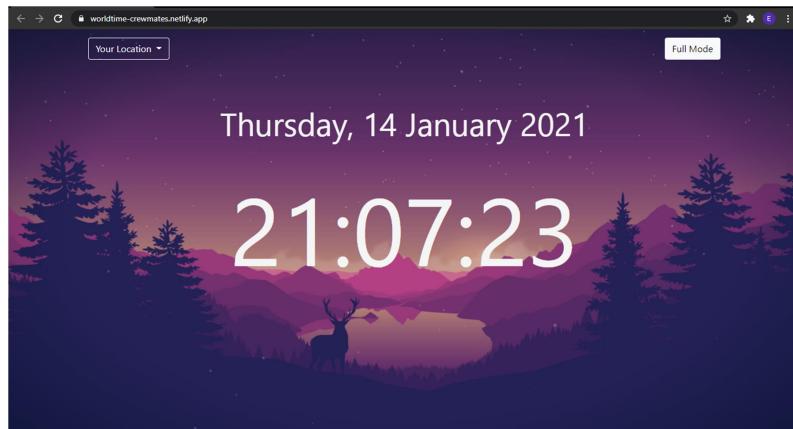
Selanjutnya buka situs Netlify, lalu drag n drop folder dist ke Netlify.



Tunggu hingga proses upload selesai, hingga muncul gambar berikut:



Maka web sudah berhasil di deploy.



TAMBAHAN :

EmberJS merupakan framework yang menggunakan template Handlebars dalam penyusunan struktur Website. Handlebars merupakan bahasa templating sederhana. Pada Handlebars, kita akan mengenal yang namanya Handlebars Expression. Ini merupakan salah satu unit dasar pada template Handlebars. Handlebars Expression ini ditandai dengan buka tutup kurung kurawal ganda {{ }}. Kita akan coba buat sebuah contoh sederhana mengenai ekspresi ini. Sebelum itu kita akan membuat sebuah project ember baru. Kita hanya memasukan command **ember new (nama project)** pada terminal.

```

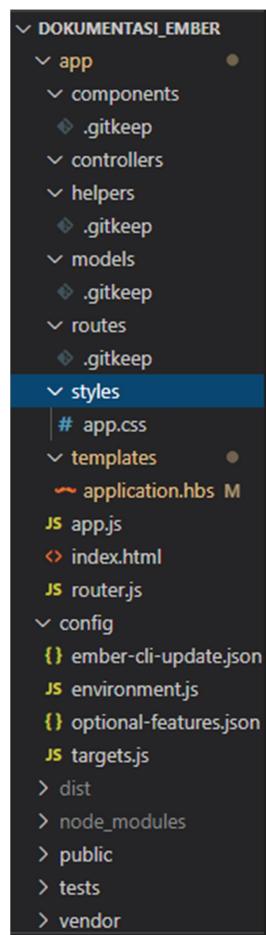
D:\tesember>ember new Dokumentasi_Emberr
installing app
ember CLI v3.24.0

Creating a new Ember app in D:\tesember\Koleksi_Dokumentasi_Emberr:
create .editorconfig
create .ember-cli
create .eslintrcignore
create .eslintrc.js
create .prettierrcignore
create .prettierrc.js
create .template-lintrc.js
create .travis.yml
create .watchmanconfig
create README.md
create app\app.js
create app\components\.gitkeep
create app\controllers\.gitkeep
create app\helpers\.gitkeep
create app\index.html
create app\models\.gitkeep
create app\routes\.gitkeep
create app\styles\.gitkeep
create app\templates\app.css
create config\ember-cli-update.json
create config\environment.js
create config\optional-features.json
create config\targets.js
create ember-cli-build.js
create .gitignore
create package.json
create public\robots.txt
create testem.js
create tests\helpers\.gitkeep
create tests\index.html
create tests\integration\.gitkeep
create tests\test-helper.js
create tests\unit\.gitkeep
create vendor\.gitkeep

Installing packages... This might take a couple of minutes.
npm: Installing dependencies ...

```

Tunggu hingga proses selesai



Di atas merupakan bagian yang kita dapatkan saat pertama kali membuat project baru. Untuk mebangun web kita, kita dapat menambahkan file .hbs maupun .js pada folder-folder didalam folder app. Folder-folder tersebut antara lain:

1. Components, untuk penempatan komponen website yang reusable yang biasanya kita panggil ke application.hbs
2. Controllers, untuk penempatan file yang berhubungan dengan logika dari web
3. Helpers, untuk penempatan file yang berhubungan dengan perhitungan
4. Models, untuk penempatan file yang berhubungan dengan data.
5. Routes, untuk penempatan file yang berhubungan tentang rute antar file

Selanjutnya kita dapat memulai pengkodean sederhana terkait penggunaan ekspresi pada EmberJS.

```
app > templates > ~ application.hbs > ...
1   {{page-title "DokumentasiEmber"}}
2
3   <h1>hallo, Nama Saya {{Name}}</h1>
4   <h2>Umur saya adalah {{Umur}}</h2>
```

Kita akan mencoba menggunakan ekspresi Name untuk nama

```
app > controllers > JS application.js > default > ↗ Umur
1   export default Ember.Controller.extend({
2     Name: "Erwin",
3     Umur: "20 Tahun"
4   });
5
```

Pendeklarasian nilai dari ekspresi kita lakukan pada bagian controller. Kita akan membuat sebuah file JS untuk pendeklarasiannya. Pendeklarasian nilai ekspresi dilakukan didalam Export default ember.controller, kode ini digunakan agar segala sesuatu yang ada pada file dapat panggil keluar.

Berikut hasil yang ditampilkan ketika dijalankan.



hallo, Nama Saya Erwin

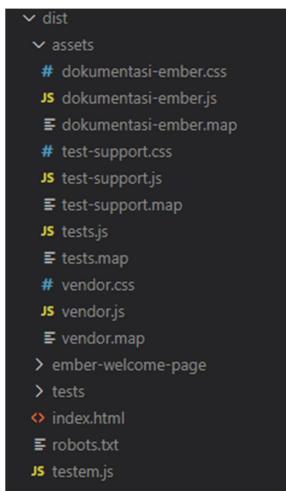
Umur saya adalah 20 Tahun

Untuk ekspresii {{page-title}} merupakan Addon yang kita dapatkan ketika kita meng-install Framework Ember. Sehingga dapat langsung digunakan.

Di dalam folder App juga terdapat index.html. File app / index.html merupakan tempat dimana struktur DOM dasar diletakkan. File ini juga merupakan tempat dimana semua template, style, dan javascript disatukan. File index.html menyertakan hook - {{content-for 'head'}} dan {{content-for 'body'}} - yang dapat digunakan oleh addons untuk memasukkan konten ke dalam Head maupun body web. Hook ini tidak akan berpengaruh terhadap web ketika tidak menggunakan addon tertentu, namun ketika kita menghapusnya web tidak akan berjalan dengan normal.

```
app > index.html > ...
1   !DOCTYPE html
2   <html>
3     <head>
4       <meta charset="utf-8">
5       <meta http-equiv="X-UA-Compatible" content="IE=edge">
6       <title>DokumentasiEmber</title>
7       <meta name="description" content="">
8       <meta name="viewport" content="width=device-width, initial-scale=1">
9
10      {{content-for "head"}}
11
12      <link integrity="" rel="stylesheet" href="{{rootURL}}assets/vendor.css">
13      <link integrity="" rel="stylesheet" href="{{rootURL}}assets/dokumentasi-ember.css">
14
15      {{content-for "head-footer"}}
16    </head>
17    <body>
18      {{content-for "body"}}
19
20      <script src="{{rootURL}}assets/vendor.js"></script>
21      <script src="{{rootURL}}assets/dokumentasi-ember.js"></script>
22
23      {{content-for "body-footer"}}
24    </body>
25  </html>
```

Struktur yang ada pada index.html akan mengarah ke file yang ada pada folder ***dist/assets***



Saat web dijalankan, file index.html akan ditampilkan. Sistem build-in tools pada ember akan melakukan penyesuaian terhadap index.html melalui file-file dari folder assets tersebut baik dari styling, pengisian konteks, maupun logika yang kita buat. Jadi kita tidak disarankan untuk melakukan perubahan pada index.html.