SoC-PLAYTIME

**Table of Contents**

[Inleiding 1](#_Toc431854378)

[Aantal personen: 2 1](#_Toc431854379)

[Vereisten 1](#_Toc431854380)

[Front End 1](#_Toc431854381)

[Back End 1](#_Toc431854382)

[Uitwerking 2](#_Toc431854383)

[Te gebruiken technologieën 2](#_Toc431854384)

[Vereisten 2](#_Toc431854385)

# Inleiding

SoC staat voor System On Chip en verwijst naar bvb.een Raspberry Pi , of Intel Edison.

Het is zeker niet nieuw om een deze SoC’s met behulp van bestaande libraries te combineren met node en angular. Raspberry Pi 2 voorziet bovendien integratie mogelijkheden met Windows10. Informatie vind je bijvoorbeeld [hier](https://blogs.windows.com/buildingapps/2015/08/10/hello-windows-10-iot-core/). Waarom dan niet een test case bouwen?

Denk aan een mogelijke toepassing en werk een voorstel uit met verschillende real-time sensoren of actuatoren. Bedoeling is de feasibility van het voorstel uit te testen.   
Voorbeeld binnen Home automation:

* Aansturen van lichten of thermostaat of .. via één of meerdere mobiele apparaten of sensoren.
* Bijhouden van historiek (temperatuur of verbruik).

## Aantal personen: 2

# Vereisten

## Front End

De front van de applicatie bestaat uit verschillende delen.

* De welkom pagina toont de controllers voor het aansturen van het device. Dit kan op een map zijn.
* Real-time weergave van de status van de devices.
* Voorprogrammering van device status op bepaalde tijdstippen.
* Rapport pagina (voor historiek of rendement) met verschillende filter mogelijkheden: datum, maximum en minimum waarden.
* Bewaren van een rapport in pdf.

## Back End

* Alle data wordt opgeslagen en is voorzien van een timestamp.
* Meerdere personen kunnen in real-time aansturen of rapporten uit vragen.
* Een administrator kan personen toegang geven of verhinderen, kan back-up’s uitvoeren voor het wegschrijven van historiek, kan de voorprogrammering van een device doen.

# Uitwerking

## Te gebruiken technologieën

|  |  |
| --- | --- |
| **Front End** 8pt van de 20pt | **Back End**\*7pt van de 20pt |
| * CSS Preprocessor, * taskRunnner, * front -end testing, * LocalStorage, * Maps, * gebruik van framework naar keuze (angular). | * volledig asynchroon in javascript – node.js, * persistentie in een NoSQL database (mongodb), * sockets voor multi-user integratie(socket-io), * autorisatie, authenticatie, * minstens één integratie en één unittest(jade), * deployment op cloud naar keuze(azure, heroku), * gebruik van framework naar keuze (express). * IoT device met WiFi adapter en eventueel randapparatuur zoals relay, switch ..   \* Default zaken, zoals gezien tijdens de lessen, staan tussen haakjes maar mogen aangepast worden. |

Noot:

* Wie geen Front End volgt kan gebruik maken van een standaard HTML, Bootstrap of Express lay-out.   
  Eventueel kan een student, die enkel Front End volgt, deze front-end aanmaken als project.
* Wie geen Back End volgt kan alle data stockeren in een lokale JSON file.  
  Eventueel kan een student, die enkel Back End volgt, deze backend aanmaken als project.

## Vereisten

* Programmeer in het Engels
* Javascript volgens de regels van de kunst (jshint als evaluatie tool)
* CSS volgens de regels van de kunst (csslint als evaluatie tool)
* Conventies:
  + Gebruik de juiste css naamgevingen.
  + Gebruik duidelijke code.
* Performantie:
  + minifying,
  + concat
  + performante selectoren,
  + performante javascript,
  + foutcontrole op kritische onderdelen,
  + zoveel mogelijk DRY en abstract (= vermijden van herhalingen)
* Mooie, acceptabele (UI) en gebruiksvriendelijk(UX) userinterface voor zowel mobile of desktop gebruik.
* Gedocumenteerd:
  + Waar nodig, als verduidelijking, in de code
  + Aanvullend document (min. 2 bladzijden) met de nodige paswoorden, online URL’s, werkverdeling, schatting van werkuren, grootste moeilijkheden(2) en successen(2), eventueel architectuur en database schema, eventuele referenties en persoonlijke conclusie
* Project beschikbaar op een source control systeem naar keuze.
* IoT communicatie naar keuze via pub/sub. (AMQP, MQTT)
* Indien datum: ten laatste afgewerkt in de week vóór het examen. Tussentijdse evaluaties zijn mogelijk.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_