Cloud-Printer

OLOD: Internet of Things - academiejaar: 2020

docent: Frederick Rogiers

studenten

Paulien De Fraine

Emmeline Martens

2NMD

inhoudsopgave

1. Discover

Briefing en uitleg van je concept
Voor IoT moesten we als werkstuk zelf een project kiezen dat we willen onderzoeken en dan uitvoeren. En wij hebben gekozen voor een Cloud Printer. Specifiek is dit dat we van de Raspberry Pi een wireless local printer server maken waar je via een html-interface makkelijk je bestand kan afprinten. Voor het printen maken we onze eigen api zodat printcommando's extern kunnen aangevraagd worden. Ons uiteindelijke doel is dat we oude niet-draadloze printers omzetten in een draadloze printer met behulp van een Raspberri Pi.We ook in staat zijn via mobiele telefoons te printen etc. Als eventuele uitbreiding dachten we ook aan 3D objecten te printen. Dit aan de hand van de Raspberry Pi ook te koppelen aan een 3D printer. We zullen eerst de andere dingen aanpakken en daarna eventueel deze uitbreiding uitvoeren.

Voor ons werkstuk moesten we rekening houden met de volgende factoren.

Briefing

IoT Werkstuk
Repository
Maak een GitHub-repository met onderstaande structuur.
Mappen & Bestanden
«github-repo-naam»/
docs/
productiedossier.pdf
presentatie.pdf
app/

Dossier

Het definitief dossier moet ingediend worden op 05/06/2020 voor 23:59 op je github-repo.

Link van de repo moet je doorsturen naar dhr. F. Roegiers.
Werk je met een private repo? Geen probleem, maar geef me dan wel toegangsrechten (Github-gebruiker: https://github.com/rogerthat-be
 (Koppelingen naar een externe site.)

De planning voor het mondeling examen wordt onder meer opgesteld op basis van de ingeleverde links.

Code

De bron
code moet definitief ingediend worden op 08/06/2020 voor 23:59 op je github-repo.
Productiedossier
Het ontwerp en de ontwikkeling wordt gedocumenteerd in een productiedossier.
Richtlijnen productiedossier
Inhoud voor dit opleidingsonderdeel IoT:

1. Discover

1.1 Briefing en uitleg van je concept

2. Define

2.1 Analyse

2.2 Noodzakelijke soft- en hardware

2.3 Inspiratie

3. Design

3.1 Visual Designs

4. Develop

4.1 Code snippets van belangrijke gedeeltes
met bijschrift / toelichting

4.2 Schermafbeeldingen uitwerking

5. Deliverables

5.1 Handleiding als je het wil namaken

5.2 Timesheet

Specs:

- Papier

- A4 (staand of liggend)

- Paginanummering

Schermopname / video-opname
Maak een scherm- en/of video-opname van de uitwerking en host deze op vimeo/youtube.
Plaats een link naar de video in de readme-file van je github repo.
Belangrijk hier, is dat men na het bekijken van de video (of meerdere video's), kan begrijpen wat de bedoeling is, wat er gebeurt en hoe dit project werkt.

Presentatie
De voorstelling gebeurt aan de hand van een presentatie (alleen of per twee) met live-voorbeelden / schermopnames / scherm delen /

2. Define

2.1 Analyse

In deze opdracht proberen we een via de Raspberry Pi een wireless connectie me de lokale printer server te maken. Dit zal gebeuren via een html interface. Dit willen we dan uitbreiden naar niet-draadloze en verouderde printers.
Dit proces gaan we ook vastleggen op de door de school voorziene camera.
Zou dit werken, gaan we ook proberen om een eventuele verbinding te maken met een 3D printer.

Concurrentie

De grootste concurrentie die we hebben is Google Cloud Printing. Natuurlijk kunnen we hier niet meteen tegenop, maar zij stoppen de aanbieding van deze service eind 2020.

Wat gaan we doen?

We maken hierbij een html pagina waarmee de gebruiker de bestanden kan doorsturen naar de printer in kwestie.

Requirements

De service moet de printer kunnen selecteren en de bestanden opvragen en doorsturen naar de geselecteerde printer.
Noodzakelijke soft- en hardware

2.2 Noodzakelijke soft- en hardware

Bij deze uitwerking maken we gebruik van de Raspberry Pi 3b+ en 4, verscheidene printers en eventueel ook een 3D printer. Ook mogen we gebruik maken van de Raspberry Pi Camera V2 8MP. Deze werd uitgeleend door de school.
Als programmeertalen gebruiken we de talen Python en HTML en CSS.

Door onderzoek hebben we gevonden dat er een cloud printing project bestaat.:https://pypi.org/project/cloudprint/

2.3 Inspiratie

Eerst had Paulien het idee om al te werken met een draadloze verbinding met de printer. Dit uit eerdere problemen met bedrade printers waarbij de kabel die de printer met de computer verbindt vaak snel kapot ging en geen verbinding maakte tenzij in een bepaalde hoek gedraaid. Emmeline wou dan een uitbreiding naar 3D printing, aangezien zij daar een grote interesse voor toont. We zagen printen dus heel breed en dachten al aan tal van mogelijkheden wat onze Raspberri Pi printer allemaal zou kunnen verwezelijken en printen.

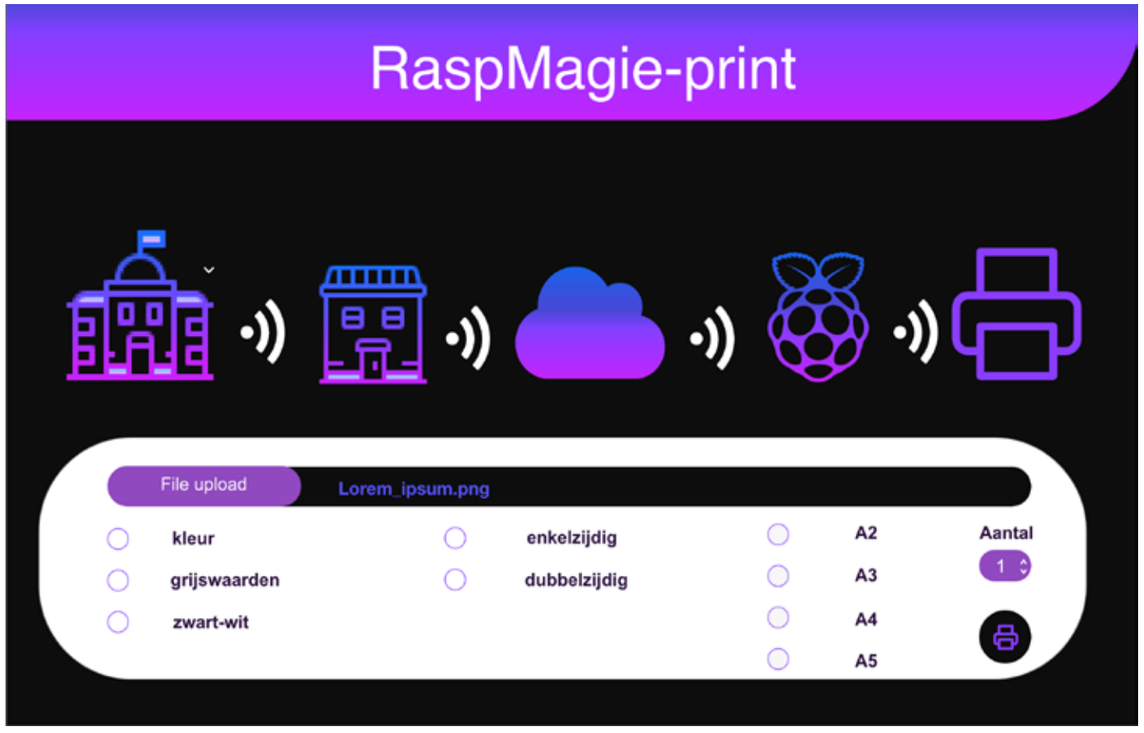
Uiteindelijk zijn we gebleven bij de basis om gewone printers te koppelen die cloud printing ondersteunen. Maar de uiteindelijke inspiratie kwam van bedrade printers te koppelen aan een draadloze service omdat veel mensen dit zelfde probleem ondervinden als Paulien heeft gehad. Zodat we ook in staat zijn om zelf via onze smarthphone te kunnen printen via het mobiel toestel.

3. Design

3.1 Visual Designs

XD-link: https://xd.adobe.com/view/3e45cc3f-898a-4c4a-4f59-7097338e04d9-4874/

Homescreen Cloud Printer



Succes Cloud Printer

4. Develop

4.1 code snippets

De bronnen die we zouden gebruiken voor ons project te verwezelijken zijn

https://developers.google.com/cloud-print/docs/pythonCode

schermafbeeldingen uitwerking

```
def EncodeMultiPart(fields, files, file_type='application/xml'):
    """Encodes list of parameters and files for HTTP multipart format.

    Args:
      fields: list of tuples containing name and value of parameters.
      files: list of tuples containing param name, filename, and file contents.
      file_type: string if file type different than application/xml.
    Returns:
      A string to be sent as data for the HTTP post request.
    """
    lines = []
    for (key, value) in fields:
      lines.append('--' + BOUNDARY)
      lines.append('Content-Disposition: form-data; name="%s"' % key)
      lines.append('')  # blank line
      lines.append(value)
    for (key, filename, value) in files:
      lines.append('--' + BOUNDARY)
      lines.append(
          'Content-Disposition: form-data; name="%s"; filename="%s"'
          % (key, filename))
      lines.append('Content-Type: %s' % file_type)
      lines.append('')  # blank line
      lines.append(value)
    lines.append('--' + BOUNDARY + '--')
    lines.append('')  # blank line
    return CRLF.join(lines)
```

beschrijving: Dit verwijst naar hoe we onze inputvelden zouden koppelen en oproepen om zo naar onze printer te sturen

```
def GetUrl(url, tokens, data=None, cookies=False, anonymous=False):
  """Get URL, with GET or POST depending data, adds Authorization header.

  Args:
    url: Url to access.
    tokens: dictionary of authentication tokens for specific user.
    data: If a POST request, data to be sent with the request.
    cookies: boolean, True = send authentication tokens in cookie headers.
    anonymous: boolean, True = do not send login credentials.
  Returns:
    String: response to the HTTP request.
  """
  request = urllib2.Request(url)
  if not anonymous:
    if cookies:
      logger.debug('Adding authentication credentials to cookie header')
      request.add_header('Cookie', 'SID=%s; HSID=%s; SSID=%s' % (
          tokens['SID'], tokens['HSID'], tokens['SSID']))
    else:  # Don't add Auth headers when using Cookie header with auth tokens.
      request.add_header('Authorization', 'GoogleLogin auth=%s' % tokens['Auth'])
  request.add_header('X-CloudPrint-Proxy', 'api-prober')
  if data:
    request.add_data(data)
    request.add_header('Content-Length', str(len(data)))
    request.add_header('Content-Type', 'multipart/form-data;boundary=%s' % BOUNDARY)
```



```
# In case the gateway is not responding, we'll retry.
retry_count = 0
while retry_count < 5:
  try:
    result = urllib2.urlopen(request).read()
    return result
  except urllib2.HTTPError, e:
    # We see this error if the site goes down. We need to pause and retry.
    err_msg = 'Error accessing %s\n%s' % (url, e)
    logger.error(err_msg)
    logger.info('Pausing %d seconds', 60)
    time.sleep(60)
    retry_count += 1
    if retry_count == 5:
      return err_msg
```

http://www.pykota.com/software/pkipplib/

schermafbeeldingen uitwerking

beschrijving: Dit is hoe we een api kunnen krijgen die ons met de printer verbindt in dit geval de google cloud printer

CUPS' API :

```
# CUPS' API
from pkipplib import pkipplib

# Create a CUPS client instance
# cups = pkipplib.CUPS(url="http://server:631, \
#                      username="john", \
#                      password="5.%!oyu")
cups = pkipplib.CUPS()

# High level API : retrieve info about job 3 :
answer = cups.getJobAttributes(3)
print answer.job["document-format"]
# That's all folks !

# Lower level API :
request = cups.newRequest(pkipplib.IPP_GET_PRINTER_ATTRIBUTES)
request.operation["printer-uri"] = ("uri",
                                    cups.identifierToURI("printers", "HP2100"))
for attribute in ("printer-uri-supported",
                  "printer-type",
                  "member-uris") :
    # IMPORTANT : here, despite the unusual syntax, we append to
    # the list of requested attributes :
    request.operation["requested-attributes"] = ("nameWithoutLanguage", attribute)

# Sends this request to the CUPS server
answer = cups.doRequest(request)

# Print the answer as a string of text
print answer
```

beschrijving: Zo kunnen we beroep doen op de APi van cups en zo onze cups koppelen met onze server

Creating :

```
# Building.py   EXAMPLE
from pkipplib import pkipplib

# Create a CUPS_GET_DEFAULT request
request = pkipplib.IPPRequest(operation_id=pkipplib.CUPS_GET_DEFAULT)
request.operation["attributes-charset"] = ("charset", "utf-8")
request.operation["attributes-natural-language"] = ("naturalLanguage", "en-us")

# Get the request as binary datas
ippdatas = request.dump()

# Parse these datas back
newrequest = pkipplib.IPPRequest(ippdatas)
newrequest.parse()

# Of course, the result of parsing matches what we created before.
print newrequest.operation["attributes-natural-language"]
```

beschrijving: zo spreken we de effetcieve functies aan van de Cups Api en zijn we zo in staat ons verzoek om te printen te sturen

5. Deliverables

5.1 Handleiding om het project na te maken

Instructies:
Voor deze app willen we onze draadloze en bedraade printers in verbinding zetten met de raspberry pi en deze de kans geven om heel wat af te printen zonder een account te hebben. Hierbij maken we gebruik van cups.
Automatisch zal de app de dichtstbijzijnde printer selecteren, waarna u uw document kan uploaden, u opties aanpassen en hoeveel keer moet printen kan kiezen. Wanneer u op print druk gaat u naar een succes pagina met een camera die het printen vastlegd.

STAP 1: Github Repo maken met Cloud Printer als naam

STAP 2: Mappenstructuur opstellen

STAP 3: Maken van Homepagina en Succespagina

STAP 4: Stylen van Homepagine en Succespagina met CSS

STAP 5: Flask server opstellen (test)

STAP 6: Cups instaleren + printer via wifi koppelen

STAP 7: downloaden van ppiklib via pip en dowloaden van cups code

STAP 8: Cups Api oproepen

STAP 9: fieds aanspreken en koppelen aan functies cups

STAP 10: kunnen printen vanaf een webserver

5.2 Timesheet

21 april: kiezen onderwerp + research gedaan op voorrand door beide

23 april: kijken wat we nodig hadden van materiaal voor ons project en
en in de lijst zetten

1 mei : maken dossier Emmeline + layout
onderzoeken hoe door Paulien

6 mei: Verdeling van Discover en Define fase normaal met 2

8 mei: Begin dossier maken via Google Docs (mooie verdeling)
Paulien Discover fase + Define fase + estetische aanpassingen
Emmeline

12 mei: Samen Design maken in XD Emmeline werkte het uiteindelijk

13 mei: Beginnen HTML Templates en CSS stylings

20 mei: bespreken van wat we al hebben met de docent en showen
het design en hier nog een hele middag aan verder werken +
onderzoeken + opmaak verder afweren in HTML en CSS

4 juni: Snippets proberen en api implementering (zonder succes), flask
server

5 juni: Google Cloud Printer gekoppeld, pkipplib installeren,
cups installeren, (api mplemetatie gelukt)


artevelde hogeschool