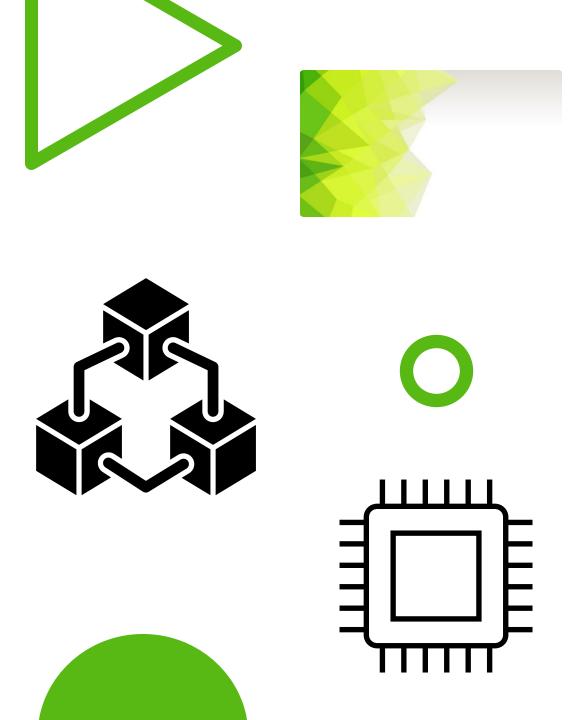
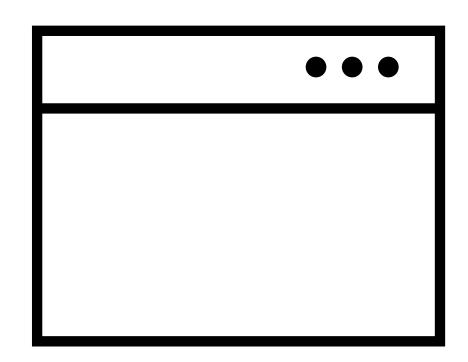
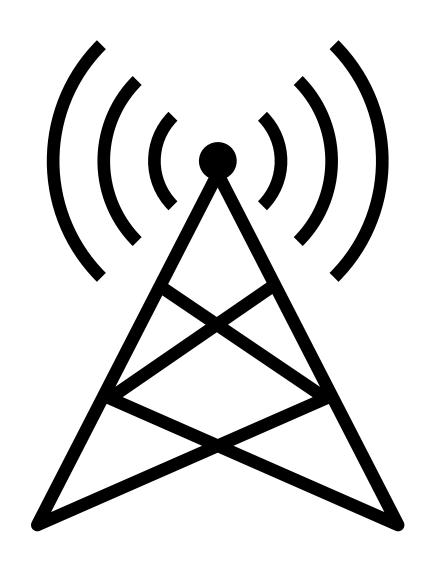
1 vaihe:kytkentä

C++





- Particle console
- WEB IDE



3 vaihe: rakennetaan webhook joka triggeröi koodin azurella

4: vaihe fuction appin rakentaminen ja storage accountin perustaminen azurelle

- AZURE
- HTTPS
- HTTP TRIGGER
- STORAGE ACCOUNT

• 5: vaihe koodataan trigger c sharpilla joka talentaa tietoa varastoon.



```
_______object
               . Coject to mirror
        peration == "MIRROR_X":
        mirror_mod.use_x = True
        mirror_mod.use_y = False
        mirror_mod.use_z = False
         _operation == "MIRROR_Y"
        lrror_mod.use_x = False
        lrror_mod.use_y = True
         lrror_mod.use_z = False
          operation == "MIRROR_Z"
          rror_mod.use_x = False
         lrror_mod.use_y = False
         __rror_mod.use_z = True
         selection at the end -add
          _ob.select= 1
          er ob.select=1
           ntext.scene.objects.action
          "Selected" + str(modified)
          irror ob.select = 0
         bpy.context.selected_obj
          mta.objects[one.name].se
vaihe 6 ohjelmoidaan käyttöliittymä html,css,js
```

```
vpes.Operator):
X mirror to the select
ject.mirror_mirror_x"
Pror X"
```

- 7 vaihe: ohjelmoidaan käyttöliittymään kutsu jolla haetaan azuren tietovarastosta sääaseman data.
- 8 vaihe ohjelmoidaan trigger joka rakentaa tietovarastoon tallenetusta datasta
- JSON rajapinnan (Rest API käyttöliittymän käyttöön



9 vaihe piiretään käyttöliittymään kaaviot JSON muodssa saadusta datasta

REACT APP

