# Prova tècnica 3.0

Es tracta de desenvolupar una API en el llenguatge i framework que et sentis més còmode. Un fragment de codi per una màquina de cobrament automàtic.

El teu codi ha de rebre una petició HTTP, WebSockets o com et vagi millor. Llegir les dades i processar-les.

Exemple petició de pagament amb targeta:

{"amount": 14527, "currency": "eur", "card\_num": 4000000000001000}

Exemple petició pagament en metàl·lic

{"amount": 14527, "currency": "eur", "coin\_types": {"10000": 2}}

Validacions:

* Targeta: Cal comprovar que el número de la targeta és vàlid, abans d’enviar el pagament al banc. Pots aplicar l’algoritme de Luhn’s (o utilitza una llibreria que ja ho incorpori l’apliqui).
* Metàl·lic: El mòdul de cobrament s’encarrega de validar les monedes i ja ha informat del que s’ha introduït, no cal validar res més.

Enviament:

* Targeta: Si la validació de la targeta es true, s’envia una petició al banc. Aquest contesta si s’ha realitzat o no i el motiu en un codi d’error
* {  
   "success": false,  
   "error": 702  
  }
* o
* {  
   "success": true  
  }
* Metàl·lic: Cal tornar el canvi. La màquina funciona bé i disposa de totes les monedes d’Euro possibles, partint d’això, has de trobar la fórmula o l’algoritme perquè torni el mínim de monedes possible. Seguint l’exemple tornaria:
* {  
   "success": true,  
   "amount": 5473,  
   "coin\_types": {  
   "1": 1,  
   "2": 2,  
   "20": 1,  
   "50": 1,  
   "200": 2,  
   "5000": 1  
   }  
  }

## Punts de millora

La prova ha acabat. Anota el temps que has tardat. A partir d’aquí si tens ganes de programar més, pots fer el que vulguis:

* Guardar les transaccions en una memòria persistent
* Fer un front, per introduir els pagaments
* Calcular el canvi perquè tingui en compte la quantitat de monedes disponibles.
* …el que vulguis aportar.

Tots els punts de millora es valoraran per donar més context a les decisions, però no representen un avantatge davant dels que només han fet la primera part.

Gràcies.