Accés a una web utilitzant un QR temporal

Joel Rodríguez Santana

Resum– Aquest projecte té com a objectiu la creació d'un sistema d'accés i control de la visualització de continguts web, fent servir codis QR temporals. Es tracta de codis QR amb diferents característiques associades com per exemple una data de venciment o una duració. La idea és que els clients, que per exemple volen accedir al guiat d'una exposició, puguin obtenir el recurs web simplement escanejant el codi bidireccional. Un cop l'usuari realitza aquesta acció, el sistema s'encarrega de gestionar el procés de validació i control mitjançant tokens criptogràfics. Aquestes peces d'informació, que van associades als QRs, són les que permeten exercir una gestió dels accessos als recursos. Per a que tot aquest sistema funcioni, són necessaris dos mòduls. El primer és el mòdul gestor de codis QR, que s'encarrega de crear, modificar i imprimir els codis. El segon és el mòdul autenticador, aquest té la funció d'activar i validar els tokens. Aquests són els elements principals que constitueixen el sistema i permeten controlar l'accés als continguts.

Paraules clau– codi de resposta ràpida (Codi QR), HyperText Transfer Protocol, token criptogràfic, Uniform Resource Locator.

Abstract– This project aims to create a system for accessing and controlling the display of web content, using temporary QR codes. These are QR codes with different associated characteristics such as an expiration date or a duration. The idea is that customers, who for example want to access the guidance of an exhibition, can obtain the web resource simply by scanning the bidirectional code. Once the user performs this action, the system is responsible for managing the validation and control process using cryptographic tokens. These pieces of information, which are associated with QRs, are the ones that allow you to manage access to resources. For this whole system to work, two modules are needed. The first is the QR code manager module, which is responsible for creating, modifying and printing the codes. The second is the authentication module, which has the function of activating and validating tokens. These are the main elements that make up the system and allow access to content.

Keywords– quick Response Code (QR code), HyperText Transfer Protocol, cryptographic token, Uniform Resource Locator.

1 Introducció - Context del treball

Avui dia existeixen diferents mecanismes que permeten donar accés a continguts protegits. Alguns d'aquests mètodes serien l'ús de credencials d'accés, autenticació via SMS o verificació via correu electrònic. Aquests exemples de mètodes d'autenticació són totalment correctes, però requereixen que l'usuari realitzi accions enrevessades a més de proporcionar dades privades. Existeixen aplicacions pràctiques, on es poden simplificar aquests tipus de processos de validació d'accés a continguts, donant així una millor experiència d'usuari i mantenint un bon nivell de seguretat.

Una solució a la problemàtica comentada anteriorment, seria la utilització de codis QR temporals. Aquest mecanisme permet donar accés als continguts de forma senzilla, simplement escanejant el codi amb un *smartphone*, a més d'oferir la possibilitat d'establir una durada màxima. Una possible implementació del sistema, on es poden apreciar els avantatges que proporciona respecte d'altres mecanismes, és en l'autenticació d'usuaris que volen accedir al guiat d'una exposició. Un cop s'ha generat el codi QR amb les diferents característiques especificades, l'usuari es limita a escanejar-lo i accedir a la informació. Tots els processos de validació i control queden amagats darrere del sistema validador, sense que el client hagi de realitzar cap acció per autenticar-se, millorant així l'experiència de l'usuari.

• E-mail de contacte: joel.rodriguezs@autonoma.cat
• Mención realitzada: Tecnologies de la Informació
• Treball tutoritzat per: Marta Prim Sabrià (Microelectrònica i Sistemes Electrònics)
• Curs 2021/22

El propòsit d'aquest treball és estudiar i desenvolupar un sistema d'accés a continguts web, específicament, un *software* que realitzi aquest control sobre les peticions que faran els usuaris a través de la lectura de codis QR. Com ja s'ha introduït anteriorment, en aquest mateix apartat, aquests codis disposen de característiques extra associades. Això vol dir que aquestes peces d'informació, no només emmagatzemaran una adreça web, sinó que també disposaran d'una durada màxima, d'una data de venciment, etc.

Els continguts d'aquest article es poden agrupar en dos grans blocs. En el primer, es troba el que seria tota la introducció i la part de la gestió del desenvolupament. Consta d'un total de cinc apartats, tenint en compte aquest mateix, on es defineix el projecte i com es durà a terme, respecte a la planificació i metodologia. En el segon bloc, es considera la part més centrada en el desenvolupament. Pel que fa a aquesta porció del document, s'elabora una anàlisi precisa de tot el sistema, de com interactuen les diferents parts que el componen i de les tecnologies utilitzades.

Per finalitzar amb l'apartat, comentar breument les principals motivacions per les quals ha sigut seleccionat aquest treball. El principal motiu és que la temàtica em va semblar interessant, crec que l'ús de codis QR és un recurs que està molt estès actualment i que perdurarà al llarg del temps. A més, sempre he tingut curiositat de saber com s'utilitzen aquestes peces d'informació en les aplicacions web i també de conèixer les oportunitats que ofereixen.

2 ESTAT DE L'ART

Actualment, existeixen diferents mètodes per donar un accés controlat a continguts web que no són públics, per als quals es necessita alguna mena d'acreditació. La implementació d'aquests varia segons si es volen limitar temporalment els accessos i a més es pretén aplicar en un context concret, com és el de donar accés al guiat d'un museu o centre d'exposicions.

Per una banda, es troba un dels mecanismes més utilitzats, que és l'accés a través d'un compte d'usuari. L'individu que vol accedir als continguts s'ha de crear un compte, el qual li atorgarà accés a la informació protegida. Si a més es vol donar accés temporalment, existeixen mecanismes que de forma interna permeten controlar els continguts accessibles per a cada client. Aquest sistema és totalment funcional i a més permet extreure dades al client, dades que posteriorment poden ser fetes servir per a publicitat personalitzada o per prendre decisions empresarials estratègiques, però per al context sobre el qual estem treballant és innecessari.

Addicionalment, existeix la verificació d'accés basada en certificats digitals, certificats que permeten la identificació d'usuaris i màquines [1]. Aquests contenen diferents camps amb informació variada, però els elements principals són la clau pública de l'usuari i la signatura d'una entitat certificadora. En aquest sistema, l'usuari entrega el certificat al servidor, aquest valida la signatura i comprova l'entitat certificadora. Si l'entitat que ha signat el certificat és considerada de confiança, llavors es dona accés, si no és així no es permet visualitzar els continguts. Succeeix el mateix que en l'anterior mecanisme, el sistema és completament funcional, però per al context en el qual estem actuant és redundant.

Una altra possibilitat existent, és un sistema d'autenticació basat en codis d'accés. Es tracta d'un mètode força senzill d'implementar, on el servidor genera un codi d'accés que posteriorment l'usuari utilitzarà per visualitzar els continguts. Aquest mecanisme és el més semblant a l'implementat en aquest projecte, però amb la diferència de què l'usuari ha d'introduir manualment la clau d'accés cada cop que vulgui visualitzar els continguts protegits de la web.

3 OBJECTIU

L'objectiu del projecte és facilitar als usuaris la visualització de continguts web i controlar l'accés mitjançant l'ús d'un sistema d'accés basat en codis QR temporals. La idea és que els usuaris puguin accedir a la informació sense necessitat de descarregar cap aplicació. Una de les parts del sistema és la generació dels codis. Els QRs generats, a més de contenir una adreça URL, tindran associades altres característiques com una durada màxima o una data de caducitat en cas de no ser utilitzats.

A banda de generar els codis, també necessitem validarlos. Aquesta és la part principal del projecte, la creació d'un sistema que autentiqui els accessos que es duen a terme. Aquest sistema haurà de verificar diferents aspectes per garantir que els accessos a la informació es fan de forma legítima. Les principals característiques que s'hauran de verificar són: la validesa dels codis amb els quals s'intenta accedir a les dades, les dates de venciment, el temps de durada màxima assignat i si el codi ja ha sigut emprat.

4 METODOLOGIA

La metodologia seleccionada, per dur a terme el desenvolupament del projecte, ha sigut Kanban. Aquesta metodologia d'origen japonès, proporciona un sistema que permet la gestió del flux de treball des del principi fins al final del projecte [2]. Es tracta d'una metodologia àgil, caracteritzada per l'ús d'un taulell i targetes en la gestió dels processos. En el taulell trobem diferents columnes, segons les implementacions el nombre de columnes pot variar, però en les implementacions més bàsiques com a mínim sempre trobarem: to do, in progress i done. Un cop definida l'estructura, les diferents tasques són representades per les targetes, que s'ordenen per prioritat.

A mesura que el projecte es duu a terme, les diferents tasques es desplacen d'esquerra a dreta fins a arribar a l'última columna del taulell. Algunes altres característiques comunes són la limitació de tasques per columna, proporcionant control sobre el flux de treball, també es pot considerar la utilització de colors en les targetes, per classificar les tasques segons tipus o també es poden utilitzar els colors per l'assignació d'equips de treball.

Kanban ens permet visualitzar en qualsevol moment les fases i el flux de treball del projecte. Gràcies a això és senzill realitzar un seguiment precís de tot el desenvolupament, a més de modificar tasques, detectar i corregir problemes, optimitzar processos i augmentar la productivitat.

El primer pas per començar a aplicar aquesta metodologia és trobar una eina que permeti crear un taulell amb les diferents columnes i targetes, existeixen moltes aplicacions que ofereixen aquest tipus de solució, pel desenvolupament

d'aquest treball el *software* escollit ha sigut Trello. Un cop escollit l'entorn, el següent pas consisteix a definir les columnes i les diferents tasques que conformaran les targetes. Per acabar, només quedaria definir altres aspectes com el màxim nombre de tasques per columna i establir una classificació per colors de les tasques, per a aconseguir una millor eficiència i productivitat en la implementació de la metodologia. En aquesta implementació en concret, s'ha fet servir una classificació de les tasques basada en quatre agrupacions. Específicament, les etiquetes són: planificació (blau), disseny (vermell), desenvolupament (taronja) i test i correcció d'errors (morat). Cadascuna té assignat un color, el qual permet identificar fàcilment en quin tipus de procés s'està treballant.

5 PLANIFICACIÓ

La planificació del projecte s'ha dut a terme mitjançat l'ús d'un diagrama de Gantt, que es pot visualitzar en l'annex. Aquesta eina ofereix informació important a l'hora d'executar un projecte, permet conèixer l'estat general de les tasques que s'han de dur a terme, qui ha de realitzar-les i quan ha de realitzar-les.

Per estructurar el diagrama, s'ha dividit el projecte en cinc fases principals i una extra especificada en l'apèndix, cadascuna d'aquestes fases té definida una durada màxima. En primer lloc, es troba l'etapa de planificació, on es defineix el projecte, les pautes a seguir durant el desenvolupament, el calendari i es duen a terme correccions sobre la proposta principal. En la següent fase es realitza la part de disseny, es comença el desenvolupament i es fan els primers testos, bàsicament s'estructuren les diferents parts del sistema i es comença a desenvolupar el codi. En tercer lloc, es continua amb el desenvolupament i els testos. A continuació s'inicia un període centrat a dur a terme parts secundàries del sistema i en finalitzar l'informe. Finalment, es porten a cap les darreres modificacions sobre la presentació i el dossier, es procedeix a finalitzar el projecte amb les entregues dels últims documents.

6 DESENVOLUPAMENT

En aquest apartat, s'analitzaran els diferents mòduls que componen tot el sistema. Els continguts s'estructuraran en dos subapartats, un per a cada component: mòdul gestor i mòdul autenticador. La idea principal d'aquesta secció és descompondre el sistema i veure quins elements existeixen, quines funcions tenen, com es comuniquen i quines tecnologies s'utilitzen.

6.1 Codificació del mòdul gestor de codis QR

A continuació, es realitzarà un estudi a la part del sistema encarregada de totes les gestions relacionades amb els codis QR. En aquest cas, l'element principal és el *front-end*, que permet operar amb els QRs, però existeixen altres peces com el *back-end* i la base de dades que també es comentaran. Es tracta d'un mòdul que el faran servir, principalment, els treballadors de l'establiment, així que s'han tingut algunes consideracions a l'hora de dissenyar el funcionament i l'aparença. La idea és que els operaris treballin amb un software senzill, que no sigui difícil d'utilitzar i que no doni problemes. Sempre es prioritza la fiabilitat del sistema, a més de la satisfacció i la bona experiència tant del client com del treballador.

6.1.1 Front-end aplicació gestora de QRs

Es tracta de la interfície d'usuari, creada amb HTML i JS, que s'encarrega de totes les interaccions entre els usuaris i el servidor. S'han fet servir aquestes tecnologies per l'alta compatibilitat que presenten amb la majoria de cercadors, el bon funcionament i l'optimització; un altre motiu important és l'existència de grans comunitats que donen suport i aporten documentació. Com es pot veure en la figura 1, l'aplicació disposa d'una composició basada en dos elements. El primer és una barra d'eines que es troba a la part superior de la pantalla, el segon són els requadres on apareixen els camps necessaris per dur a terme les operacions, aquestes figures es troben sota l'eina de navegació.

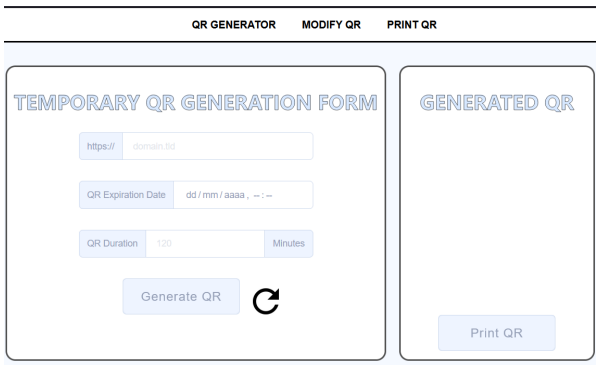

Fig. 1: *Front-end* de l'aplicació gestora de QRs

Barra de navegació
Pel que fa al primer element, aquest permet a l'usuari escollir quina eina vol fer servir. Per defecte, en el primer accés a la web, sempre sortirà l'eina de generació de codis QR com l'escollida. Aquesta barra de navegació permet visualitzar només un dels instruments alhora, d'aquesta forma s'aconsegueix que la pantalla no se saturi amb la mostra de molts objectes al mateix temps, evitant errades i facilitant la utilització.

Eina generadora de codis QR
El segon mòdul d'aquest *front-end*, es compon d'uns requadres que contenen els camps del formulari que permetran dur a terme diferents accions segons l'eina seleccionada. Per al QR Generator, apareixen dos objectes encarregats de realitzar la generació dels QR. El primer és un formulari que disposa de tres camps d'entrada per definir valors com la data d'expiració, la duració de l'accés o l'adreça URL efectiva. L'altre requadre mostra la imatge del codi QR que s'ha generat. Quan s'omplen tots els camps i es prem el botó d'enviar, es duen a terme dues accions internes que són importants per al funcionament del sistema. Mitjançant el llenguatge JS, primer es llança una petició al servidor, per generar un codi identificador únic per al QR que s'acaba de crear. Un cop s'ha obtingut aquest valor alfanumèric, es pot elaborar l'adreça que emmagatzemarà el codi bidireccional. Per exemple, en

l'entorn de proves, aquesta direcció tenia una estructura similar a la del diagrama de la figura 2. D'aquesta forma, quan l'usuari intenti accedir als continguts, remetrà una petició a l'API validadora, que posteriorment reencaminarà a l'adreça efectiva introduïda al formulari de generació. El segon mètode que s'executa després de prémer el botó de *submmit*, és la inserció de la informació del QR a la base de dades. Un cop realitzades les verificacions necessàries sobre els camps omplerts, s'executa una petició al servidor per emmagatzemar les dades.

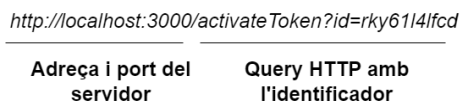

Fig. 2: Format de la petició HTTP a l'API validadora

Modificació de codis QR existents
El següent mòdul disponible és l'encarregat de modificar els codis QR existents a la base de dades. En aquest cas la composició també està basada en dos requadres. El primer, permet cercar el QR que es vol alterar. Per fer això, és necessari introduir el codi identificador sobre el qual es volen realitzar els canvis. Un cop s'ha fet servir un identificador vàlid, només s'ha de polsar el botó de cercar i automàticament es farà una consulta, a la base de dades, que proporcionarà la informació actual que emmagatzema el QR. Aquestes dades es poden consultar en el segon requadre, que es troba a sota del primer. L'exploració a la base de dades, no es fa directament des del codi JS del front-end, sinó que en el moment de fer clic al botó de cercar, s'executa una funció que posteriorment fa una petició al back-end. Un cop s'ha obtingut la resposta, és quan es poden carregar les dades.

En el requadre inferior trobem la informació que pot ser modificada per l'operari. El conjunt de dades és el següent: l'URL associada al QR, la data de venciment i la durada. Un cop s'han dut a terme tots els canvis necessaris, només quedarà fer clic al botó d'actualitzar. De forma similar a l'anterior cas, es crida una funció JS encarregada d'enviar una petició al servidor amb tots els canvis a implementar. Per a l'usuari només seran visibles els canvis que hagi introduït al formulari, però de forma interna també s'actualitzen els flags. Això es fa per garantir el correcte funcionament del sistema, un cop s'utilitzi el QR modificat. Els flags que es canvien són: firstAcces, used i active. Un altre paràmetre que també es veu afectat és tokenKey. Un cop es realitza una modificació, tots aquests atributs prenen els valors base. Amb això s'aconsegueix que l'API autenticadora generi un nou token amb les noves característiques.

Selecció i impressió dels codis QR
A continuació, s'estudiarà l'últim mòdul que hi ha a l'aplicació web. En aquest cas es tracta del responsable d'imprimir els codis QR. Un altre cop la composició es basa en dos requadres. En el primer es troba un formulari que permet cercar el QR que es vol imprimir. Com succeïa en el cas anterior, un cop l'operari ha introduït un codi correcte i prem el botó de cercar, s'executa un codi JS des del front-end que envia una consulta cap al servidor. Quan s'obté la resposta del servidor, apareix el segon requadre que mostra el QR i un botó que permet carregar la imatge en una plantilla. En fer clic al botó *LOAD*, apareix una nova finestra amb la plantilla i la figura integrada [3]. Finalment, per imprimir tota la plantilla, es pot fer servir un nou botó que apareix a sota del requadre, o també es pot fer manualment amb el cercador.

Tots els formularis que són utilitzats pels treballadors, en els diferents mòduls, realitzen comprovacions sobre els inputs. Abans de trametre cap petició, es busquen possibles errors com valors impossibles, camps buits o codis identificadors incorrectes. Si es detecta qualsevol error d'aquest tipus, es mostren missatges informatius indicant el problema i probable solució.

6.1.2 Back-end aplicació gestora de QRs

En el costat del servidor, s'utilitza un entorn Node.js que gestiona peticions, interacciona amb la base de dades i amb el *front-end*. S'ha escollit aquesta tecnologia, basada en JavaScript, perquè ofereix característiques que són interessants a l'hora de crear una aplicació web, algunes d'aquestes particularitats són la velocitat, l'escalabilitat o els esdeveniments asíncrons.

Pel que fa a l'aplicació, es fan servir tres llibreries per fer funcionar el sistema. La primera és Express, que permet estructurar l'aplicació web a més d'oferir funcionalitats d'enrunament, *middleware*, gestió de peticions HTTP, *cookies* i sessions. Un *framework* molt útil i que facilitat l'elaboració d'aplicacions web. En segon lloc, es troba Path, aquesta llibreria proporciona característiques per treballar amb els diferents directoris i arxius que componen el sistema. I per últim Mongoose, un mòdul que fa de connector entre l'aplicació i la base de dades, permeten realitzar connexions amb el sistema d'emmagatzematge, crear esquemes de dades i dur a terme consultes. A continuació es farà un repàs a les principals funcions de l'aplicació, detallant que fan i com funcionen.

Posada en marxa del servidor i connexió a la base de dades
La primera acció que es pot veure en el codi és la posada en marxa del servidor, mitjançant Express. Es defineix una constant *app*, per gestionar peticions, i també un port per a les connexions. Després d'iniciar el servidor, es continua amb la connexió a la base de dades mitjançant els mètodes que proporciona Mongoose. L'últim que es duu a terme en aquesta secció de codi és la creació d'un esquema de dades. Aquesta estructura conté tota la informació que es guardarà sobre cadascun dels QRs. En aquesta implementació s'emmagatzemen les següents dades: codi identificador, URL efectiva, data d'expiració, duració de l'accés, un *flag* que indica si el QR ha sigut utilitzat, un *flag* que indica el primer accés, un *flag* que indica si el token encara és actiu i la clau de la signatura criptogràfica de 256 bits que servirà per protegir al token de possibles atacs.

Gestió de les peticions de recursos al *front-end*
Sota el codi de la secció anterior, es troben les funcions que gestionen els recursos del *front-end*. Es controlen totes les peticions dirigides cap a documents HTML, JS, CSS, etc. Per respondre aquestes peticions, es fa servir Express, concretament la funció *get* que serveix per respondre a les

sol·licituds HTTP GET.

Generació d'identificadors únics
Pel que fa a la generació dels identificadors únics comentada anteriorment, es reserva una funció encarregada de rebre aquest tipus de peticions i retornar els valors alfanumèrics corresponents. Per a aconseguir generar valors que siguin aleatoris i únics, es fan servir dues funcions: *Math.random()* i *Date.now*. Aquests dos mètodes es combinen per reduir les probabilitats de col·lisió. Amb aquesta combinació, per exemple evitem que *Date.now* generi dos números idèntics per a dues peticions que es processen en el mateix mil·lisegon [8]. A més s'aplica una conversió del valor a base 36 per augmentar el rang en el qual es treballa, es passa de tenir 10 símbols a 36 [9]. Fer servir aquesta codificació proporciona altres avantatges com identificadors més compactes, una presentació més neta i evitar errors amb símbols especials. Implementant aquestes tècniques maximitzem la propietat d'unicitat i assegurem un bon funcionament del sistema [10].

Emmagatzematge de dades
A l'hora d'emmagatzemar la informació relativa als codis QR, fem servir les funcions proporcionades per Mongoose. Es recullen els diferents paràmetres, enviats pel mètode *HTTP GET*, i es col·loquen seguint l'esquema de dades definit al començament del codi. Un cop s'han organitzat les dades, es pot realitzar la inserció i s'espera una resposta de la base de dades.

6.1.3 Base de dades de l'aplicació gestora de QRs

La tecnologia escollida per emmagatzemar la informació relativa als codis QR ha sigut MongoDB. És una base de dades basada en documents, no relacional, que funciona molt bé amb Node.js. El servidor serà l'encarregat d'interactuar amb aquest sistema d'emmagatzematge per fer les insercions, modificacions i eliminacions. Com que no fa servir consultes SQL, s'ha d'utilitzar la llibreria Moongose que proporciona els mètodes necessaris per dur a terme la connexió i les diferents operacions. Com es pot veure en la figura 3, l'estructura de dades feta servir en aquest cas està composta pels següents atributs: codi identificador, URL efectiva, data d'expiració, duració de l'accés, un *flag* que indica el primer accés, un *flag* que indica si el QR ha sigut emprat, la clau de la signatura criptogràfica i un *flag* que indica si el token encara és actiu.

```
object {8}
  code : 1g6kuyab7pk
  url : https://uab.cat
  expiration : 2022-05-20T12:00
  duration : 20
  firstAcces : 2022-05-18T17:56
  used : true
  tokenKey : 954613746472950224 0c7f674399c69062ae5e063628d1de4bd6265
             759ed4b2b
  active : false
```

Fig. 3: Format esquema base de dades

6.2 Codificació del mòdul autenticador

A continuació, es realitzarà un estudi a la part del sistema que s'encarrega de dur a terme la creació i autenticació dels tokens d'accés. La implementació s'ha fet utilitzant l'entorn de Node.Js i *Json Web Token* (JWT). Per finalitzar, s' explicarà en què consisteix el control i l'emmagatzematge dels tokens.

6.2.1 Json Web Token

Un dels elements principals d'aquest sistema són els JWT. Es tracta d'uns tokens d'accés que funcionen sota l'estàndard RFC 7519. Aquest patró permet que es realitzin intercanvis d'informació de forma segura entre dues parts diferents [11].

Un JWT està compost per tres parts diferenciades que analitzarem a continuació. La primera és el *header* o capçalera. Com es pot veure en la Figura 4, en la capçalera trobarem dues dades importants: el tipus d'algoritme que s'ha fet servir per dur a terme la signatura i el tipus de token. Per defecte, l'algoritme d'encriptació és HMAC-SHA256, però hi poden aparèixer altres com RSA. Un altre valor que pot prendre aquesta característica és *none*, si es defineix l'algoritme com a *none* llavors no es durà a terme cap signatura i el contingut quedarà desprotegit. Pel que fa al tipus de token, normalment sempre pren el valor JWT.

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

Fig. 4: Format del *header* d'un JWT

La segona part és el *payload*, aquest segment emmagatzema informació sobre l'aplicació. Aquestes dades es representen com a parelles de clau/valor. Dins d'aquest camp es poden distingir dos tipus de dades, la informació personalitzada que s'afegeix segons la implementació i dades de control del mateix estàndard anomenades *claims* [12]. Alguns dels valors més típics que ofereix JWT són *Issued At* (iat) i *Expiration Time* (exp), es fan servir noms curts per estalviar espai. Cal recordar que cap element va xifrat, tot el contingut és visible per a qualsevol que tingui accés al token. El que sí que permet controlar JWT és que les dades no es manipulin, fent servir una signatura criptogràfica [13]. En la figura 5, es pot veure el que seria un exemple de format del payload. En aquest cas es pot veure l'atribut *sub*, que permet relacionar el token i la persona que l'ha sol·licitat, també hi ha un camp *name* que conté informació personalitzada i per últim un paràmetre que indica quan s'ha emès el token.

```
{
  "sub": "1234567890",
  "name": "John Doe",
  "iat": 1516239022
}
```

Fig. 5: Format del *payload* d'un JWT

Finalment, trobem el que seria la signatura del token. Com ja s'ha comentat amb anterioritat, es fa servir un algoritme criptogràfic, que per defecte és l'HMAC-SHA256. La signatura es crea fent servir la capçalera, el *payload* i una clau secreta, que es recomana que sigui de 256 bits. A la Figura 6, es pot veure la funció que es fa servir i els diferents elements necessaris per formar el certificat. Cal destacar que el *header*, les dades i la clau, han d'estar codificats en base64.

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  954613746472950240c7f
) ☐ secret base64 encoded
```

Fig. 6: Format de la signatura d'un JWT

Un cop definides les tres parts del token, es codifiquen en base 64 [13] i s'agrupen utilitzant un punt per diferenciarles. Com a resultat s'obté un conjunt d'informació com el que es mostra en la Figura 7.

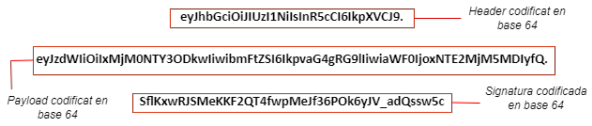

Fig. 7: Format d'un JWT codificat en Base64

6.2.2 API autenticadora

Per crear l'aplicació d'autenticació s'ha fet servir un entorn basat en Node.Js. Els principals motius de fer servir aquesta tecnologia basada en JavaScript són les característiques que ofereix i les llibreries que hi ha disponibles. En aquest cas s'han necessitat tres llibreries: JWT, Crypto i Cors. Altres peces de codi reutilitzable com Mongoose o Express no s'explicaran en aquest apartat, ja que han sigut comentades anteriorment. La primera llibreria permet treballar amb tokens JWT, inclou mètodes de creació i verificació indispensables per al funcionament del sistema. Crypto es fa servir per generar les claus que s'utilitzen a l'hora de signar amb HMAC-SHA256. Per últim Cors, aquesta llibreria afegeix unes capçaleres especials a les peticions, necessàries per a dur a terme la comunicació entre l'API i el *front-end* web [15].

Un cop definit el port amb el qual es treballa i realitzada la connexió a la base de dades, entren en joc les dues funcions principals de l'API.

Funció d'activació de tokens

Aquesta primera funció és l'encarregada d'activar els tokens, rep per paràmetre un codi corresponent a l'identificador d'un QR. Per comprendre més fàcilment com treballa aquest codi, s'inclou un diagrama de flux en l'apèndix que il·lustra tots els passos, relacions i resultats que poden sorgir.

El primer que es realitza és un càlcul de l'hora actual del sistema. Per obtenir l'hora local de forma correcta, primer s'ha de calcular la diferència horària i després afegir-la a la data UTC +0. Els valors calculats es faran servir posteriorment.

El següent pas és verificar que la petició que arriba té com a paràmetre el codi que es requereix per a dur a terme tot el procés, en cas negatiu, s'envia un codi d'error 403. Si la petició disposa d'un identificador vàlid, es continua amb el còmput de les dades, sinó retorna un codi d'error 403. Un cop s'ha verificat que el valor és correcte, es comprova si el QR ha caducat, en cas afirmatiu s'actualitza el *flag active* i tramet un codi d'error 403. En cas negatiu, es procedeix amb l'activació del QR. Per activar el codi, primer és necessari actualitzar els *flags used* i *active*. Un cop aquestes característiques tenen valor true, reencamina l'usuari al *front-end*. En concret a una pantalla de càrrega, on de forma transparent, es procedeix a remetre una nova petició d'activació de codi a l'API. Aquesta nova petició arribarà un altre cop al segment de codi encarregat de l'activació dels QRs, però aquest cop no es realitzarà una petició nova, sinó que s'actualitzarà el token afegint el primer accés, es crearà un token JWT i es trametrà al client. La implementació s'ha dut a terme d'aquesta forma perquè, per a emmagatzemar el token, es requereix que l'usuari faci la petició estant al front-end de l'aplicació. Per això en el primer accés, es redirigeix al client a la pantalla de càrrega, on s'executa el codi necessari per dur a terme la petició final que provoqui la creació del token. Un cop creada i enviada aquesta peça d'informació, es procedeix a mostrar els continguts que demana l'usuari.

Un cop vist el recorregut d'accions i peticions que es duen a terme en el primer accés d'un codi QR, falta explorar un darrer cas que tracta aquest mòdul. S'ha contemplat la possibilitat que l'usuari esborri el token o faci servir un altre dispositiu per a l'accés. En aquest cas, es passen totes les validacions explicades anteriorment, amb la diferència que ara no es finalitza en la creació d'un token nou. El procediment és el següent, primer s'ha de verificar que el token encara es troba actiu i no s'ha esgotat la duració. En cas que la duració hagi arribat al seu màxim, s'envia un codi d'error 403 a l'usuari i s'actualitza el *flag* de *active* a *false*. En el cas contrari, s'identifica si la petició s'està trametent des de el *front-end* o si l'envia l'usuari des del cercador. En el primer cas, s'actualitza el *flag active* i es procedeix a crear un token nou amb la duració actualitzada. En el segon, es redirigeix al client cap al *front-end*. Com es pot veure, s'aplica la lògica exposada anteriorment. A l'apèndix es pot trobar un diagrama de flux que il·lustra el funcionament explicat sobre la funció d'activació de tokens.

En l'apartat de *Control i emmagatzematge dels tokens*, s'explica el funcionament de la part de codi del *front-end* i quina lògica se segueix per tractar les peticions.

Funció de verificació de tokens

Aquesta funció s'encarrega de verificar que els tokens existents no han sigut manipulats i que no incompleixen cap de les característiques assignades, com per exemple la duració. Un cop es rep la petició, on el token s'envia en el *body* i l'identificador del QR com a paràmetre, es verifica que existeix la informació a la base de dades. En cas negatiu, tramet un codi d'error 403, en cas positiu es procedeix amb la validació del token contingut en el *body*. La validació es duu a terme fent servir un mètode de la

llibreria JWT, aquesta funció rep com a paràmetres el token i la *key*. Si l'autenticació s'ha realitzat satisfactòriament, es reencamina al client cap als continguts. En cas negatiu, respon un codi d'error 403 i s'actualitza el *flag active*.

6.2.3 Control i emmagatzematge dels tokens

Aquest codi se situa al *front-end* de la web, concretament a la pantalla de càrrega. La idea és que aquesta funció s'encarregui de sol·licitar i gestionar els tokens de l'usuari. Com s'ha pogut veure en apartats anteriors, per a que el client pugui guardar aquestes peces d'informació, la sol·licitud s'ha de fer des del *front-end*, aquest mòdul és l'encarregat de fer les peticions i realitzar l'emmagatzematge. Un cop s' ha fet la redirecció a la pantalla de càrrega, comença l'execució amb la primera comprovació, s'analitza si existeix algun token guardat a *local storage*. En cas negatiu, vol dir que el client necessita sol·licitar a l'API l'activació del QR perquè no disposa de cap element que l'autentiqui per poder visualitzar els continguts. Abans de crear la petició HTTP, es comprova que el codi passat com a paràmetre no estigui buit, un cop verificat això es pot prosseguir amb la sol·licitud, emmagatzematge en *local storage* i redirecció al contingut. L'altre cas que es pot donar és que el client disposi d'un token, llavors el procediment canvia. El que fa el codi és generar una petició de verificació a l'API, s'envia l'element perquè el servidor pugui processar-lo i retornar un resultat. Si l'autenticació s'ha resolt correctament, es reencamina al client cap al contingut, en cas contrari respon un codi d'error 403.

7 TESTS DE FUNCIONAMENT I SEGURETAT

Al llarg del desenvolupament s'han realitzat una sèrie de proves, que tenen com a objectiu comprovar que els diferents mòduls del sistema autenticador funcionen de forma correcta i proporcionen un nivell de seguretat adequat a la situació.

Primer, es va posar el focus en el funcionament. Mentre es duia a terme la codificació d'aquesta part, es van tenir en compte diferents casos possibles que podien succeir durant l'execució de l'aplicació. Per poder fer l'estudi sobre el funcionament, van ser seleccionades tres característiques fonamentals que alteren l'estat del sistema. Un cop seleccionades les variables de prova, es comencen a recrear els diferents casos d'estudi. En la Taula 1, es poden veure els casos que es van testejar i els valors que prenen les variables en cada cas.

TAULA 1: CASOS D'ESTUDI

Token nou	Token en LS	Accés vàlid	Resposta
No	No	No	Error 403
No	No	Si	Cas impossible
No	Si	No	Error 403
No	Si	Si	Mostra contingut
Si	No	No	Error 403
Si	No	Si	Mostra contingut
Si	Si	No	Error 403
Si	Si	Si	Mostra contingut

Un cop executats tots els casos representats a la taula anterior, es va procedir a resoldre els errors trobats i passar a testejar possibles atacs sobre els tokens JWT. La idea principal de l'estudi, no és buscar esquerdes de seguretat que permetin vulnerar el sistema des de qualsevol punt, sinó que se centren els esforços només a treballar amb els JWT, ja que són l'element principal del procés d'autenticació. Les proves de seguretat realitzades sobre els tokens, es fonamentaven en l'alteració del contingut de l'element o inclús en la generació de peces d'informació noves.

A continuació, s'exposaran alguns dels atacs que s'han intentat dur a terme sobre el sistema. En primer lloc, es va intentar aprofitar una de les vulnerabilitats més bàsiques que poden existir pel que fa al tractament de tokens JWT, que el servidor no verifiqui la signatura. Si l'API no vàlida la signatura, o cosa que és equivalent, a la capçalera el camp *alg* pren el valor *none*, els tokens no són sotmesos a cap mena de control i poden ser totalment alterats [16]. En la implementació duta a terme, abans de permetre cap accés, sempre es verifica la signatura criptogràfica. Si s'intenta realitzar aquest atac, l'API respon amb un 403, encara que de forma interna l'error que apareix en el servidor és: *Verification failed: JsonWebTokenError: invalid token*.

Una altra opció per fer passar un token il·legítim com a vàlid podria ser, alterant la capçalera i modificant el camp d'algoritme, afegit el valor *none*. Un cop modificat el *header*, s'hauria d'eliminar l'última part d'aquesta peça d'informació, concretament la que fa referència a la signatura. Si no es fan les comprovacions necessàries, aquest token podria passar com a legítim. En aquesta implementació, l'API respon amb un 403, encara que de forma interna l'error que apareix en el servidor és: *Verification failed: JsonWebTokenError: jwt malformed.*

Un altre mètode, que podria permetre alterar de forma maliciosa els tokens, és trencar la clau de la signatura. Els mateixos desenvolupadors d'aquesta eina recomanen fer servir claus de 256 bits, per així maximitzar la seguretat de la criptografia. En cas d'utilitzar claus més petites o amb formats vulnerables, es pateix el risc que es descobreixi la clau i això permeti als atacants modificar tokens i fer-los passar per legítims. Existeixen softwares com *JWT_Cracker*, que permet dur a terme atacs de força bruta, o *JWT_Tool* que executa atacs de diccionari. El sistema implementat empra claus de 256 bits per evitar aquesta mena de vulnerabilitats.

Per acabar serà comentada una última vulnerabilitat, que en la implementació actual no funcionaria, però que igualment és interessant exposar-la i entendre el funcionament. Es tracta d'un atac que consisteix en canviar l'algoritme de signatura de RS256 (clau publica/privada) a HS256(clau única). En realitzar aquesta modificació el servidor autenticador es podria confondre i permetre signar el token fent servir la clau pública, en comptes de la privada. Com s'ha indicat anteriorment, el sistema actual impossibilita aquesta vulnerabilitat pel fet que es fa ús des d'un principi l'algoritme de clau única HS256 [17].

8 PÀGINA DE TEST

La pàgina de test és una web que es fa servir per simular el reencaminament dels usuaris a un *front-end* final. Un cop es crea el token o es valida l'accés, els usuaris són enviats al recurs que sol·liciten. En aquest cas es fa servir una pàgina de test per simular el que seria el destí. Com es trac-

ta d'una aplicació web de test, que només es fa servir per comprovar si el sistema autentica les peticions, es va optar per fer servir un *template open source* [20]. Com es pot veure en la Figura 8, aquest simula el que seria la pàgina web d'una galeria d'art, donant així una imatge similar de la possible aplicació real del sistema. El recurs original és totalment funcional i disposa d'un gran nombre de seccions i característiques, però per aquesta implementació ha sigut modificat, ja que no es necessiten totes les funcionalitats que ofereix. La pàgina de test està limitada a mostrar una única plana amb diferents continguts, no es pot fer servir la navegació ni tampoc accedir als continguts.

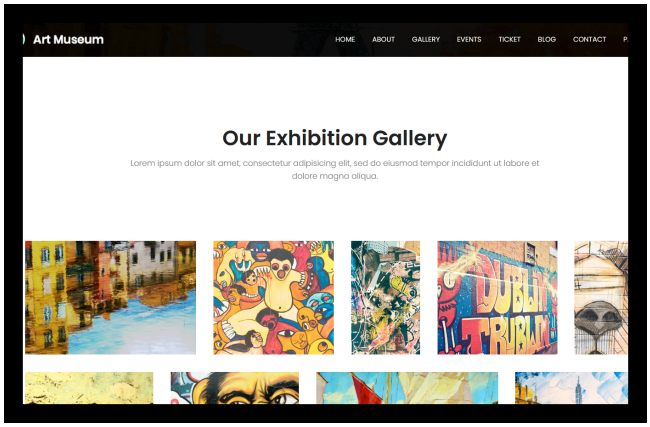

Fig. 8: Imatge de la pàgina de test utilitzada

9 Conclusions

Un cop desenvolupat i implementat tot el projecte, es pot afirmar que s'han complit els objectius proposats en els terminis establerts, a més d'haver obtingut uns bons resultats. Durant aquests mesos no ha sorgit cap inconvenient i no ha sigut necessari modificar la planificació, en gran part això és gràcies a que s'ha executat correctament la metodologia suggerida al principi del projecte. També han sigut de gran ajuda les diferents reunions realitzades amb la tutora, que han permès orientar el projecte i resoldre dubtes.

Pel que fa al desenvolupament, la primera part duta a terme (servidor web i *front-end*), no ha suposat un repte molt complicat, ha servit per refrescar alguns conceptes bàsics sobre la creació d'aplicacions web. En canvi, la codificació del mòdul autenticador i l'ús de tokens criptogràfics sí que ha suposat un desafiament on he après sobre tecnologies noves i conceptes que no havia vist abans. Realment, durant el desenvolupament de tot aquest treball, he assimilat molts coneixements i he posat en pràctica eines vistes durant la carrera. Cosa que m'ha acabat de formar com a enginyer informàtic i em permetrà afrontar nous reptes en el futur.

Com a possibles millores del projecte, crec que seria bo implementar codis QR temporals que no siguin unipersonals, és a dir, que un mateix QR el pugui fer servir un conjunt limitat de persones. Penso que seria una eina útil en certes circumstàncies.

Per finalitzar, crec que el resultat d'aquest projecte ha sigut tot un èxit i estic molt agraït amb tot el que he après.

Agraïments

M'agradaria agrair especialment a la meva tutora Marta Prim Sabrià, per tot el suport i el seguiment que ha fet durant el desenvolupament del projecte, sense la seva ajuda la qualitat d'aquest treball seria molt inferior.

Referències

[1] G. David Maayan. "5 Authentication Methods that Can Prevent the Next Breach." ID R&D. https://www.idrnd.ai/5-authentication-methods-that-can-prevent-the-next-breach/ (accessed Mar. 2, 2022)

[2] C. Roca. "La metodología Kanban, esencial para mejorar el flujo de trabajo de tu proyecto." The Power Business School. https://www.thepowermba.com/es/blog/metodologia-kanban/ (accessed Feb. 23, 2022)

[3] JavaScript Kit "Accessing objects of a window via another" JavaScript Kit. http://www.javascriptkit.com/javatutors/window4.shtml (accessed May. 30, 2022)

[4] S. Singh. "3D Flip Button." Free Frontend. https://freefrontend.com/css-buttons/ (accessed Mar. 20, 2022)

[5] A. Iker. "Input group :focus-within." Csshint. https://csshint.com/css-input-text/ (accessed Mar. 19, 2022)

[6] I. Viktoria Maciohsek. "Hover underline animation" 30Secondsofcode. https://www.30secondsofcode.org/css/s/hover-underline-animation (accessed Mar. 23, 2022)

[7] W3Schools. "How TO - Alerts" W3Schools. https://www.w3schools.com/howto/howto_js_alert.asp (accessed Mar. 23, 2022)

[8] R. Fadhil. "How to Generate Unique ID in JavaScript" DEV. https://dev.to/rahmanfadhil/how-to-generate-unique-id-in-javascript-1b13 (accessed Mar. 29, 2022)

[9] P. Katiyar. "Generate random alpha-numeric string in JavaScript" GeeksforGeeks. https://www.geeksforgeeks.org/generate-random-alpha-numeric-string-in-javascript/ (accessed Mar. 29, 2022)

[10] B. Regenspan. "Why does reddit use base36 for article id" StackOverflow. https://stackoverflow.com/questions/1712937/why-does-reddit-use-base36-for-article-id/1712957#1712957 (accessed Mar. 29, 2022)

[11] M. Jones, J. Bradley & N. Sakimura. "JSON Web Token (JWT)" Datatracker. https://datatracker.ietf.org/doc/html/rfc7519 (accessed Apr. 9, 2022)

[12] Ionos. ”JSON Web Token (JWT): an introduction” Ionos. https://www.ionos.com/digitalguide/websites/web-development/json-web-token-jwt/ (accessed Apr. 9, 2022)

[13] C. molin. ”Why Base64 is used in JWTs?” Stack Overflow. https://stackoverflow.com/questions/58341833/why-base64-is-used-in-jwts (accessed Jun. 19, 2022)

[14] J. Bradley, B. Campbell, M. B. Jones & C. Mortimore. ”JSON Web Token Claims” Iana. https://www.iana.org/assignments/jwt/jwt.xhtml (accessed Apr. 9, 2022)

[15] D. Parwal. ”How to deal with CORS error in express Node.js Project ?” GeeksForGeeks. https://www.geeksforgeeks.org/how-to-deal-with-cors-error-in-express-node-js-project/ (accessed Apr. 20, 2022)

[16] N. Tariq. ”Attacking JSON Web Tokens (JWTs)” InfoSecWriteUps. https://infosecwriteups.com/attacking-json-web-tokens-jwts-d1d51a1e17cb (accessed May. 5, 2022)

[17] A. Singh. ”Attacks on JSON Web Token (JWT)” InfoSecWriteUps. https://infosecwriteups.com/attacks-on-json-web-token-jwt-278a49a1ad2e (accessed May. 5, 2022)

[18] Ben. ”403 Forbidden HTML Templates” Free Frontend. https://freefrontend.com/403-forbidden-html-templates/ (accessed May. 16, 2022)

[19] J. Marron. ”81 CSS Spinners” Free Frontend. https://freefrontend.com/css-spinners/ (accessed May. 16, 2022)

[20] Colorlib ”Art Museum – Free HTML5 Museum Website Template” Theme Wagon. https://themewagon.com/themes/free-museum-website-template/ (accessed May. 30, 2022)

APÈNDIX

A.1 Planificació diagrama de Gantt

La planificació del projecte il·lustrada amb un diagrama de Gantt es troba a continuació, en la Figura 9. Es poden visualitzar diferents requadres que relacionen tasques amb setmanes, aquestes agrupacions són les fases descrites en l'apartat de planificació. Cal destacar que, la fase de la setmana vint, és una etapa extra que s'inicia un cop s'ha finalitzat el projecte i s'han entregat tots els documents. Aquesta setmana es dedica exclusivament a desenvolupar i entregar el pòster del projecte.

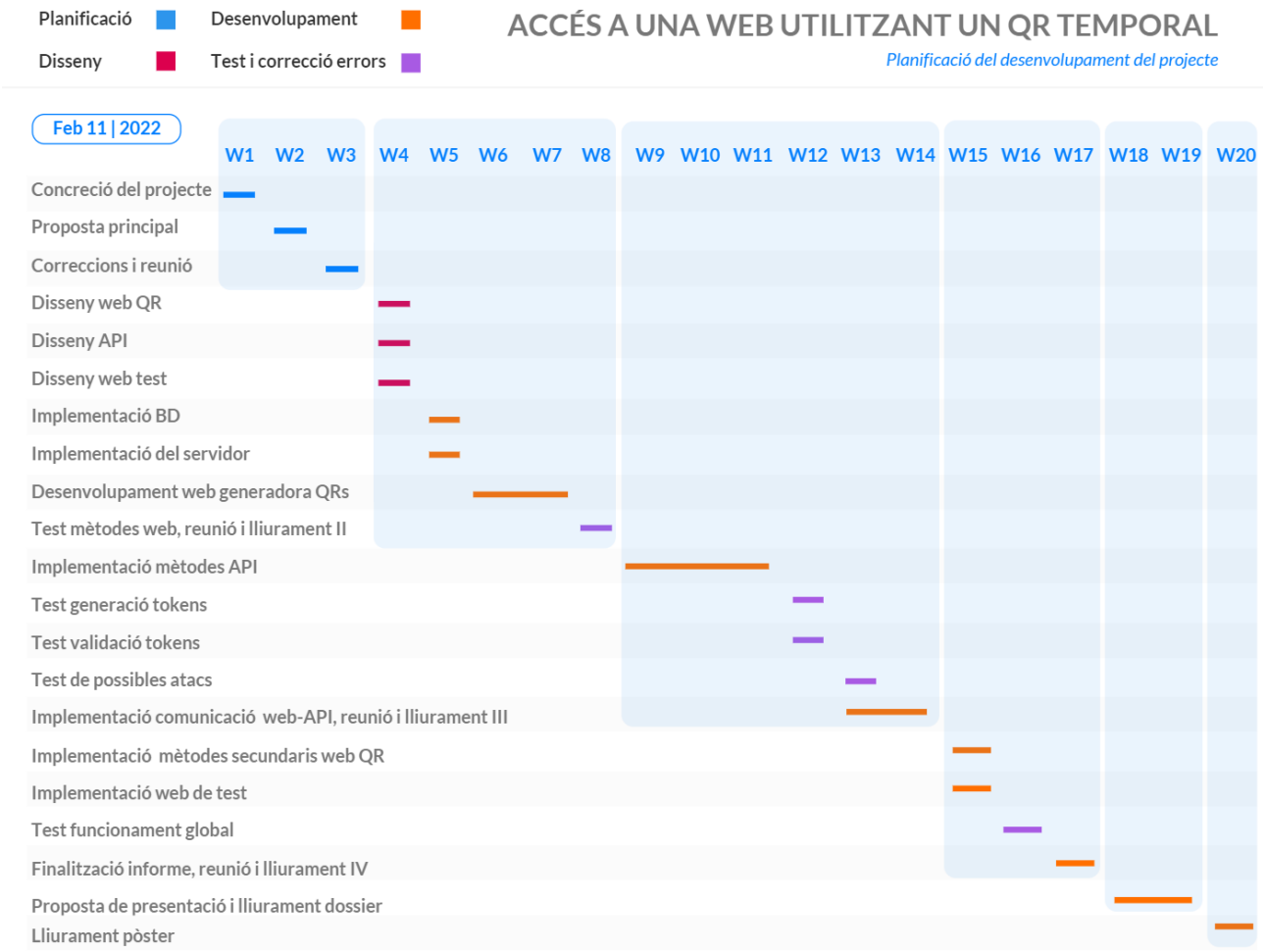

Fig. 9: Diagrama de Gantt amb la planificació del projecte

A.2 Diagrama de flux funció d'activació de tokens

A continuació, es troba la Figura 10 que mostra un diagrama de flux amb la lògica utilitzada per al desenvolupament de la funció d'activació de tokens. Aquesta figura ajuda a entendre, de forma gràfica, el procés que se segueix en els diferents casos possibles.

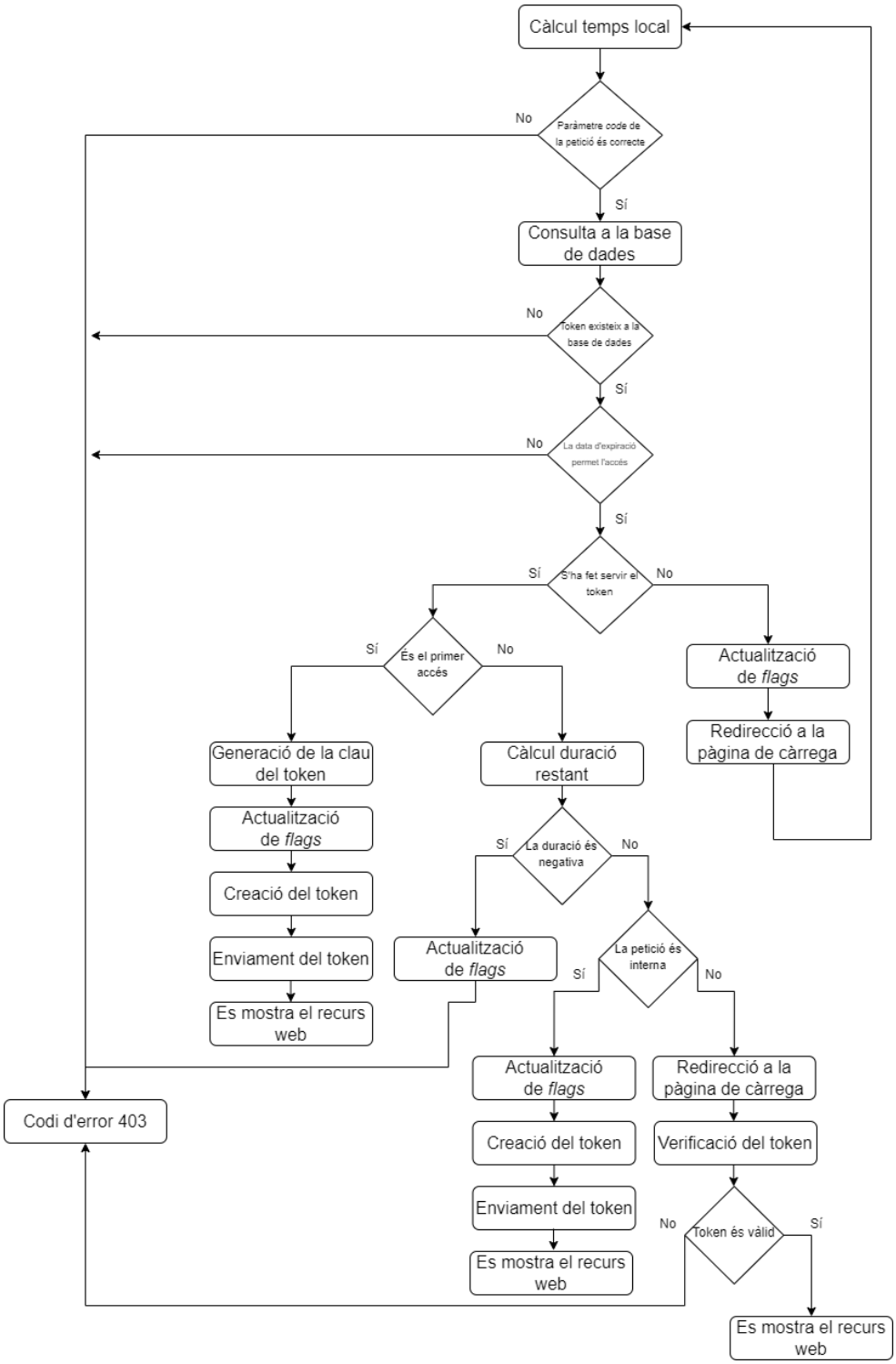

Fig. 10: Diagrama de flux funció d'activació de tokens.