

EXAMEN: Sistemes i Tecnologies Web

Juny 2024

Niu:

Nom:

Permutació A

Survey

Després de pujar el codi a moixero.uab.cat ompliu el següent requadre.

Entrega Electrònica

The SHA1 checksum of the received file is:

.....

Time stamp is:

.....

Guardeu-vos una còpia de l'arxiu que entregueu.

Guia de correcció:

La nota serà...	Quan...	Guia de puntuació
De 0 a 5 punts	Quan no funciona tot i hi ha errors de concepte <i>greus</i> en algun dels següents elements clau: callbacks, clausures, classes, herència, promises, i els diversos elements de vue (reactivitat, directives, events, props, components, ...).	Es parteix d'un 5 i es resta 1 punt per cada error de concepte.
De 5 a 7 punts	Quan no s'aconsegueix fer funcionar tot l'examen i no hi ha errors de concepte <i>greus</i> .	Es parteix d'un 7 i es resten 0.25 punts per cada error.
De 7 a 10 punts	L'examen passa tots els tests i feu entrega electrònica.	Teniu un 10. Us l'heu guanyat.

Instruccions

Seguiu les següents instruccions per a arrencar la màquina del laboratori i importar l'esquelet del projecte.

- Arrenqueu la màquina si no l'heu arrencada abans i seleccioneu la partició de Linux (user: examen i passwd: examen).
- Obriu una consola: Applications → Terminal.
- Descarregueu-vos l'esquelet del projecte executant la comanda:

```
wget https://moixero.uab.cat/ExamenSTW.7z
```

- Descomprimiu el fitxer.

```
7z x ExamenSTW.7z
```

- L'esquelet el teniu dins el directori **stw**. Feu l'examen (podeu fer servir el mateix terminal que ja teniu obert, i obrir el directori **stw** amb el Visual Studio Code).
- Per executar l'aplicació utilitzeu la comanda:

```
node exam.js
```

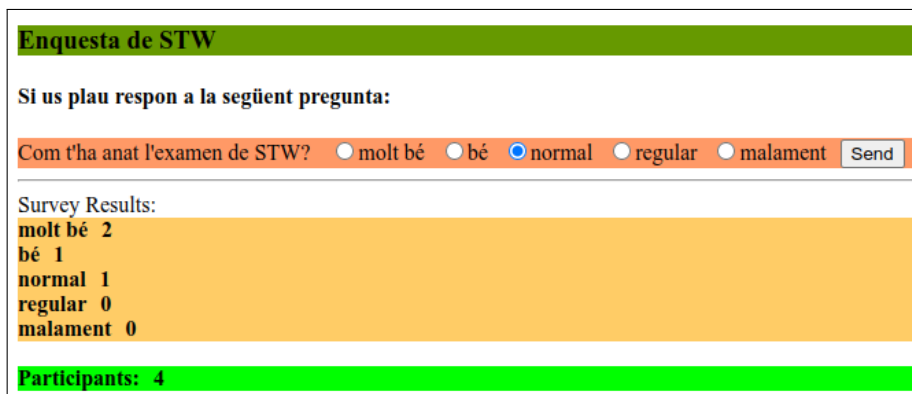
- Nota: Si utilitzeu Chrome, veureu que s'obre en mode incògnit i no hi ha disponible l'addon de Vue. Obriu un nou Chrome (Ctrl+N). La nova finestra ja no és incògnit i el addon de Vue es pot utilitzar.
- Un cop hagueu acabat de desenvolupar el projecte, haureu d'entregar el vostre codi electrònicament. **Si us funciona tota l'aplicació, aviseu-nos abans d'entregar electrònicament.**
- Per entregar electrònicament, creeu un zip de la següent manera:

```
cd stw  
7z a sol.zip exam.js public/app.js
```

- Comproveu el contingut de l'arxiu que entregareu (obriu l'arxiu i mireu el contingut dels arxius que hi ha a dintre).
- Un cop sapigueu segur que voleu entregar aquest arxiu, pugeu-lo a: <https://moixero.uab.cat/>.
- Anoteu els dos valors (el checksum i el timestamp) a l'examen en paper i entregueu l'examen en paper.
- **Quan acabeu no sortiu de la sessió i no pareu la màquina!!**

Context

Desenvoluparem una aplicació web per a gestionar enquestes online (Figura 1). Concretament, desenvoluparem una enquesta d'una sola pregunta amb cinc possible respostes. Es deixarà un temps per rebre respostes. En acabar aquest temps, el backend farà un recompte de les diferents respostes rebudes i mostrarà el resultat.



Enquesta de STW

Si us plau respon a la següent pregunta:

Com t'ha anat l'examen de STW? ☐ molt bé ☐ bé ☒ normal ☐ regular ☐ malament

Survey Results:

molt bé 2
bé 1
normal 1
regular 0
malament 0

Participants: 4

Figura 1: En l'enquesta de STW hi han participat quatre estudiants. A dos els ha anat molt bé, a un bé i a un altre normal.

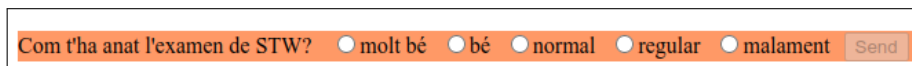
Podeu arrencar aquest portal executant `node exam.js`.

Exercici frontend

Caldrà que programeu dos components. **No** s'ha de modificar el component `RootComponent`. A continuació us indiquem els components a implementar:

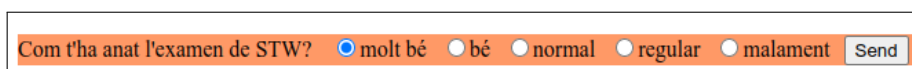
1. OneQuestionForm

Aquest component mostra una pregunta, cinc possibles respostes i un botó per enviar la resposta (Figures 2, 3).



Com t'ha anat l'examen de STW? ☐ molt bé ☐ bé ☐ normal ☐ regular ☐ malament

Figura 2: Formulari d'una sola pregunta amb cinc possibles respostes i un botó de submit deshabilitat.



Com t'ha anat l'examen de STW? ☒ molt bé ☐ bé ☐ normal ☐ regular ☐ malament

Figura 3: Un cop seleccionada una resposta s'habilita el botó 'send'.

Especificacions

- Aquest component rep dos paràmetres: la pregunta a mostrar i la llista de possibles respostes(['molt bé', 'bé', ..., 'malament']).
- El botó per enviar la resposta està deshabilitat (Figura 2) fins que se selecciona una resposta (Figura 3).
- Per simular fàcilment que som diferents usuaris responent l'enquesta, podem votar tants cops com vulguem des del mateix formulari, per tant, un cop hem habilitat el botó de 'send' ja no el deshabilitarem.
- Podeu utilitzar el següent template per fer el component:

```
<div>
<span class="question">Here goes the question</span>
<span><input type="radio" v-model="vote" value="0"><label>molt bé</label></span>
```

```

<span><input type="radio" v-model="vote" value="1"><label>bé</label></span>
<span><input type="radio" v-model="vote" value="2"><label>normal</label></span>
<button>Send</button>
</div>

```

- Genereu dinàmicament els tags `` de les opcions de resposta (radio buttons) amb la directiva `v-for` (`v-for="(element,index) in elements"`).
- El comportament per defecte dels *radio buttons* és de *single selection*. Només se'n pot seleccionar un (no heu de fer res).
- L'atribut `value` del *radio button* conté el vot (el id de resposta) que enviarem al servidor: **[0..4]** a on cada id correspon a una opció de vot: **0**: molt bé, **1**: bé, **2**: normal, **3**: regular, **4**: malament.
- Per guardar-vos el vot (**[0..4]**) de l'usuari en el model utilitzeu la directiva `v-model`.
- En clicar el botó 'send' cal fer la crida a l'endpoint del servidor: `/vote/id_vot`. Per exemple, si votem l'opció 'molt bé' (id=0), la crida a l'endpoint del server serà: `/vote/0`. Aquesta petició la farem amb el mètode POST. La podeu generar amb la funció: `fetch("http://localhost:3001/vote/0", {method: "POST"})`. Noteu que no esperem rebre cap resposta.

2. ResultsViewer

Quan l'enquesta ha finalitzat, aquest component mostra el recompte de vots (Figures 4a i 4b). Mentre l'enquesta estigui activa, oberta per rebre vots, indiquem aquest estat a l'usuari (Figura 4c).

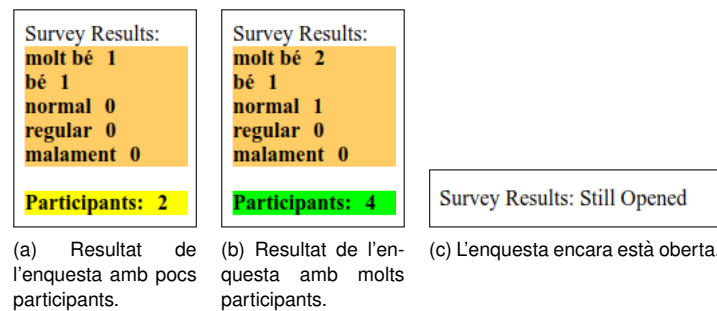


Figura 4: Les tres diferents casuístiques del component ResultsViewer.

- Aquest component rep com a paràmetre la llista de possibles respostes(['molt bé', 'bé', ..., 'malament']).
- Per obtenir el recompte de vots, quan **creem** o **muntem** el component, fem una crida a l'endpoint del servidor: `/results`. Per fer la petició a l'endpoint i esperar la resposta podeu utilitzar la funció: `fetch('http://localhost:3001/results').then(res=>res.json()).then(json=>{.....})`
- El servidor ens torna un `json` amb el següent format: `{participants: 30, responses: [5,10,12,1,2]}` a on l'atribut `participants` indica el nombre total de vots rebuts i l'atribut `responses` és un array a on els índex coincideixen amb el id de les opcions de vot: **0**: molt bé, **1**: bé, **2**: normal, etc. i el valor dels índex correspon al nombre de vots d'aquesta opció. En l'exemple anterior, a cinc persones els ha anat molt bé, a deu bé, a dotze normal, a una regular i a dues malament. El servidor ja està programat per tornar-vos uns resultats dummy d'una enquesta de 30 participants amb el següent recompte de vots: molt bé: 5, bé: 10, normal: 12, regular: 1, malament:2.
- Si a l'enquesta han participat menys de tres persones el nombre de participants es mostra en groc (*yellow*) (Figura 4a), sinó es mostra en verd (*lime*) (Figura 4b).
- Podeu utilitzar el següents templates per fer el component:
 - 1: En el cas que l'enquesta encara estigui oberta:

```
<div>
  Survey Results:
  <span>Still Opened</span>
</div>
```

2: En el cas que l'enquesta ja estigui tancada:

```
<div>
  <div>
    <div class="results" ><span>Molt bé: </span><span>5</span></div>
    <div class="results" ><span>bé: </span><span>10</span></div>
    <div class="results" ><span>normal: </span><span>12</span></div>
  </div>
  <br />
  <div class="participants" style="background-color: lime">
    <span>Participants:</span><span>30</span>
  </div>
</div>
```

- Utilitzeu la directiva `v-for` per a generar dinàmicament el tag `div` per cada opció de vot:

```
<div class="results" ><span>Molt bé: </span><span>5</span></div>
```

Notes

- No es pot fer servir `async/await`.

Exercici backend

Context

Desenvoluparem un servidor que reculli els vots de diferents usuaris. Un cop tancada l'enquesta, el servidor calcularà el resultat de la votació i l'encapsularà en un `json: {participants: 30, responses: [5,10,12,1,2]}` a on l'atribut `participants` indica el nombre de vots rebuts i l'atribut `responses` és un array a on els índex coincideixen amb el id de les opcions de vot: **0**: molt bé, **1**: bé, **2**: normal, etc. i el valor dels índex correspon al nombre de vots d'aquesta opció. Per a emmagatzemar els vots rebuts implementarem la classe `SurveyServer`. A part, per a que el `SurveyServer` no es col·lapsi per la rebuda massiva de vots, el que farem serà que el servidor web en rebre un vot l'encapsularà en una promesa que resoldrà al vot passat un temps aleatori. Per a fer- ho implementarem la funció `getDelayedPromiseGenerator`.

Especificacions de la funció `getDelayedPromiseGenerator`

- Aquesta funció no rep cap paràmetre i retorna una funció "g" que rep com a paràmetre el vot emès per l'usuari i retorna una promesa:

```
getDelayedPromiseGenerator() : g
g(vot) : promise
```

La promise retornada per `g` resol amb el vot rebut com a paràmetre després d'un temps aleatori. Aquest temps aleatori depèn del temps que `g` ha assignat en una execució prèvia. Per exemple:

1. A t_0 ens entra un vot que hem encapsulat en la promesa P_0 que resol passats $delay_{t_0}$ segons a on: $delay_{t_0} = 0$
2. A t_1 ens entra un altre vot que encapsulem en la promesa P_1 que resol passats $delay_{t_1} = delay_{t_0} + Math.random() * 100$ segons.

- Per aquesta funcionalitat us caldran **clausures**.
- No utilitzeu variables globals.

Especificacions de la classe SurveyServer

- La classe **SurveyServer** implementa tres funcions públiques (**this.**):
 - **addResponse(promise) : void**
 - **getResults() : promise**
 - **toString() : void**
Aquesta funció printa el recompte i ja us la donem feta.
- Tots els atributs que us calguin dins la classe feu-los privats (**let**).

Especificacions de la funció SurveyServer::getResults

- Aquesta funció no rep cap paràmetre i retorna una promesa que encapsula el resultat de la votació. **Aquesta promesa es resol quan la votació estigui tancada.**

Especificacions de la funció SurveyServer::addResponse(p)

- Aquesta funció és la que afegeix el vot rebut al recompte. El recompte és el json:
`{participants: 0, responses: [0,0,0,0,0]}`
- Després d'afegir un vot, inicia un timeout de 5 segons de cortesia. Si durant aquests cinc segons no s'ha rebut cap votació es tanca la votació. Si pel contrari, es rep un vot, es cancel·la aquest timeout engegat i se'n comença un altre de nou de 5 segons. Per a fer-ho utilitzeu el següent codi:

En la classe **SurveyServer** definiu la propietat privada **timeout**:

```
let timeout = undefined.
```

En la funció **addResponse** afegiu el següent codi:

```
if(timeout != undefined){clearTimeout(timeout)}
timeout = setTimeout(()=>{
  timeout = undefined
  //Tanquem la votació si no queda cap promesa de vot per resoldre
}, 5000)
```

- Quan la funció **addResponse** tanca la votació, resol amb el recompte final la promesa que retorna **getResults**.
- **Atenció, la votació no es pot tancar per expiració del timeout de cortesia si encara queda alguna promesa que encapsula un vot per resoldre. En aquest cas es tanca el recompte quan la darrera promesa amb vot s'hagi resolt.**
- En l'esquelet veureu que teniu **dos tests** per provar la funció **addResponse** i **getResults**. Abans de programar el servidor web executeu els tests, cadascú per separat. No els executeu junts. Cada test us indica la sortida esperada. **Quan passeu els tests ensenyeu-li al professor.**

Especificacions del servidor web

- El servidor tindrà dos endpoints:

/vote/:option a on el paràmetre **option** és el vot de l'usuari (**0**: molt bé, **1**: bé, **2**: normal, **3**: regular, **4**: malament). En rebre aquesta petició s'ha d'encapsular el vot en una promesa mitjançant la funció "g" que retorna la crida a **getDelayedPromiseGenerator** i invocar a la funció **addResponse**.

/results En rebre aquesta petició s'ha de retornar en format **json** el recompte de vots quan s'hagi tancat la votació (`{participants: 30, responses: [5,10,12,1,2]}`).

Notes

- No es pot fer servir **async/await**.
- No es poden fer servir variables globals.
- No es poden fer servir class expressions (class syntactic sugar).

Pàgina d'esboranys. No la desgrapeu.
