Memòria pràctica 3

DAT QT2019

Alba Mendez

22 de desembre de 2019

1 Introducció

L'objectiu d'aquesta pràctica és programar en Haskell un sistema de preguntes i respostes, tipus fòrum. Es dona una aplicació d'exemple, i el codi de l'aplicació que l'estudiant ha de desenvolupar. L'aplicació també té un petit sistema de permisos, on els usuaris poden ser *webmaster* o *leader* d'un tema de discussió, i això els permet realitzar accions adicionals.

L'estructura bàsica, el model i les rutes ja venen fetes, i la tasca és fer les plantilles HTML així com el codi que gestiona les peticions GET i POST de cada ruta. Igual que en la pràctica anterior, es dona també un entorn de desenvolupament per a compilar i desplegar l'aplicació fàcilment; no obstant, hem preferit desenvolupar-lo en local.

2 Funcions d'utilitat

Abans de començar amb el desenvolupament, afegirem unes funcions al mòdul `Found`.

Veiem que al final del mòdul es defineix la funció `isAdmin user`. Definirem també una funció similar `isLeader theme user` que evalua si l'usuari té privilegis d'administració sobre un tema:

```
isLeader :: Theme -> UserId -> Bool
isLeader t u = isAdmin u || u == tLeader t
```

Nota: Per practicitat, hem fet que l'usuari webmaster també pugui administrar els temes.

Durant el desenvolupament, ens trobarem que en consultar l'objecte corresponent a una ruta, se'ns retorna un `Maybe`. Per extreure l'objecte del `Maybe` hem de gestionar la possibiltat de que no existeixi, i la millor manera de fer-ho és disparant un error 404.

Per tant, definirem una funció `liftChecked` que funciona com a substitut de `liftIO` però a més, extreu el valor del `Maybe` com hem comentat a dalt. La definició és molt senzilla:

```
liftChecked :: MonadHandler m => IO (Maybe a) -> m a
liftChecked x = liftIO x >>= maybe notFound pure
```

Llavors en comptes de `liftIO $ getQuestion qid db` (que retornaria `Maybe`) podem escriure `liftChecked $ getQuestion qid db` (que ens retorna l'objecte directament).

Per últim, en el mòdul `Model` definirem una funció per esborrar completament una pregunta, incloent les seves respostes:

```
deleteFullQuestion :: QuestionId -> ForumDb -> IO ()
deleteFullQuestion qid conn = do
    answers <- getAnswerList qid conn
    forM_ answers $ \ (aid, _) -> deleteAnswer aid conn
    deleteQuestion qid conn
```

3 Especificar la categoria

També, ens adonem que el formulari per crear un nou tema (el codi del qual ja ens ve donat) no permet especificar categoria, fixant-la sempre a una cadena buida:

```
themeForm :: AForm (HandlerFor Forum) Theme
themeForm =
    Theme <$> freq (checkM checkUserExists textField)
                   (withPlaceholder "Introduiu el nom de l'usuari
                     ↪  responsable" "Nom del responsable")
                   Nothing
          <*> pure ""
          <*> freq textField (withPlaceholder "Introduiu el títol del
            ↪  tema" "Titol") Nothing
          <*> freq textareaField (withPlaceholder "Introduiu la
            ↪  descripció del tema" "Descripció") Nothing
```

Per arreglar-ho, canviem `pure ""` per:
`freq textField (withPlaceholder "Introduiu la categoria" "Categoria") Nothing`
tal i com es fa en els altres camps.

4 Redisseny de la plantilla general

També canviarem algunes coses en el fitxer `default-layout.html`, que conté la plantilla base. En concret:

- Es substitueix Bootstrap 3 per Bootstrap 4 amb les icones de Font Awesome.
- Es fa servir una navbar de Bootstrap per a la barra superior.
- El cos de la pàgina ja no està dins un `container-fluid`, per tant és responsabilitat de la pàgina encapsular el contingut dins containers. Això dona més flexibilitat.

El codi de la plantilla queda així:

```
<!DOCTYPE html>
<html><head>
  <link rel="stylesheet" type="text/css"
    ↪  href="https://stackpath.bootstrapcdn.com/
    ↪  bootstrap/4.4.1/css/bootstrap.min.css">
  <link rel="stylesheet" type="text/css"
    ↪  href="https://stackpath.bootstrapcdn.com/
    ↪  font-awesome/4.7.0/css/font-awesome.min.css">
  <title>Forum: #{pcTitle page}</title>
  ^{pcHead page}
</head><body>
<nav class="navbar navbar-dark bg-primary">
  <a class="navbar-brand mb-0 h1 mr-auto" href="@{HomeR}">DatForum</a>

  $maybe{ user <- mbuser }
    <span class="navbar-text mx-3"><i class="fa fa-user"></i> Usuari:
      ↪  <strong>#{user}</strong></span>
    <a class="btn btn-primary my-1" href="@{AuthR LogoutR}"><i
      ↪  class="fa fa-sign-out"></i> Tanca sessió</a>
  $nothing
    <a class="btn btn-primary my-1" href="@{AuthR LoginR}"><i
      ↪  class="fa fa-sign-in"></i> Login</a>
  $end
</nav>

$maybe{ msg <- mbmsg }
<div class="container">
  <div class="row"><div class="col-sm-12">
    <div class="message error">#{msg}</div>
  </div></div>
```

```
</div>
$end

^{pcBody page}
</body></html>
```

Posteriorment es pot veure l'aspecte de la plantilla nova, per exemple en la figura 1. No obstant, però, la funcionalitat continua sent la mateixa.

5 Vista *home*

La lògica (tant de GET com de POST) de la vista principal ja ens ve feta, només faltaria fer la plantilla per a la vista. La llista de temes quedaria així:

```
<div class="container my-4">
<h1 class="my-3">Tauler de temes</h2>

<div class="row row-cols-1 row-cols-md-2">
  $forall{ p <- themes }
  <div class="col mb-4">
    <div class="card">
      <div class="card-body">
        <h5 class="card-title">#{tTitle (snd p)}</h5>
        <h6 class="card-subtitle mb-3 text-muted">#{tCategory (snd p)}
          ↪  <span class="mx-3"><i class="fa fa-user"></i> #{tLeader
          ↪  (snd p)}</span></h6>
        <p class="card-text" style="white-space:
          ↪  pre-line">#{tDescription (snd p)}</p>
        <a href="@{ThemeR (fst p)}" class="btn btn-primary">Visita</a>
      </div>
    </div>
  </div>
  $end
</div>
</div>
```

Hem fet servir els cards, un component de Bootstrap 4, per mostrar cada tema de discussió. Cal observar que, en el paràgraf on es mostra el text, s'estableix la propietat CSS `white-space` a `pre-line` per tal que el navegador respecti els salts de línia del text. Tot i així, els espais consecutius sí es col · lapsaran.

Per altra banda, el formulari quedaria així:

```
$if{ maybe False isAdmin mbuser }
<div class="container my-4">
<h1 class="my-3"><i class="fa fa-plus-circle"></i> Nou tema</h1>

<form role="form" method="POST" action="@{HomeR}">
  <div class="row">
    <div class="col-sm-12">
      ^{tformw}
    </div>
```

```
    </div>
    <div class="row">
      <div class="col-sm-12">
        <button type="submit" class="btn btn-success">Afegeix</button>
      </div>
    </div>
  </form>

  </div>
  $end
```

Hem corregit la condició perquè només es mostri el formulari si l'usuari està autenticat i és webmaster.

Ara comprovem que el formulari funciona i es poden afegir temes.
El disseny resultant es pot veure a la figura 1.

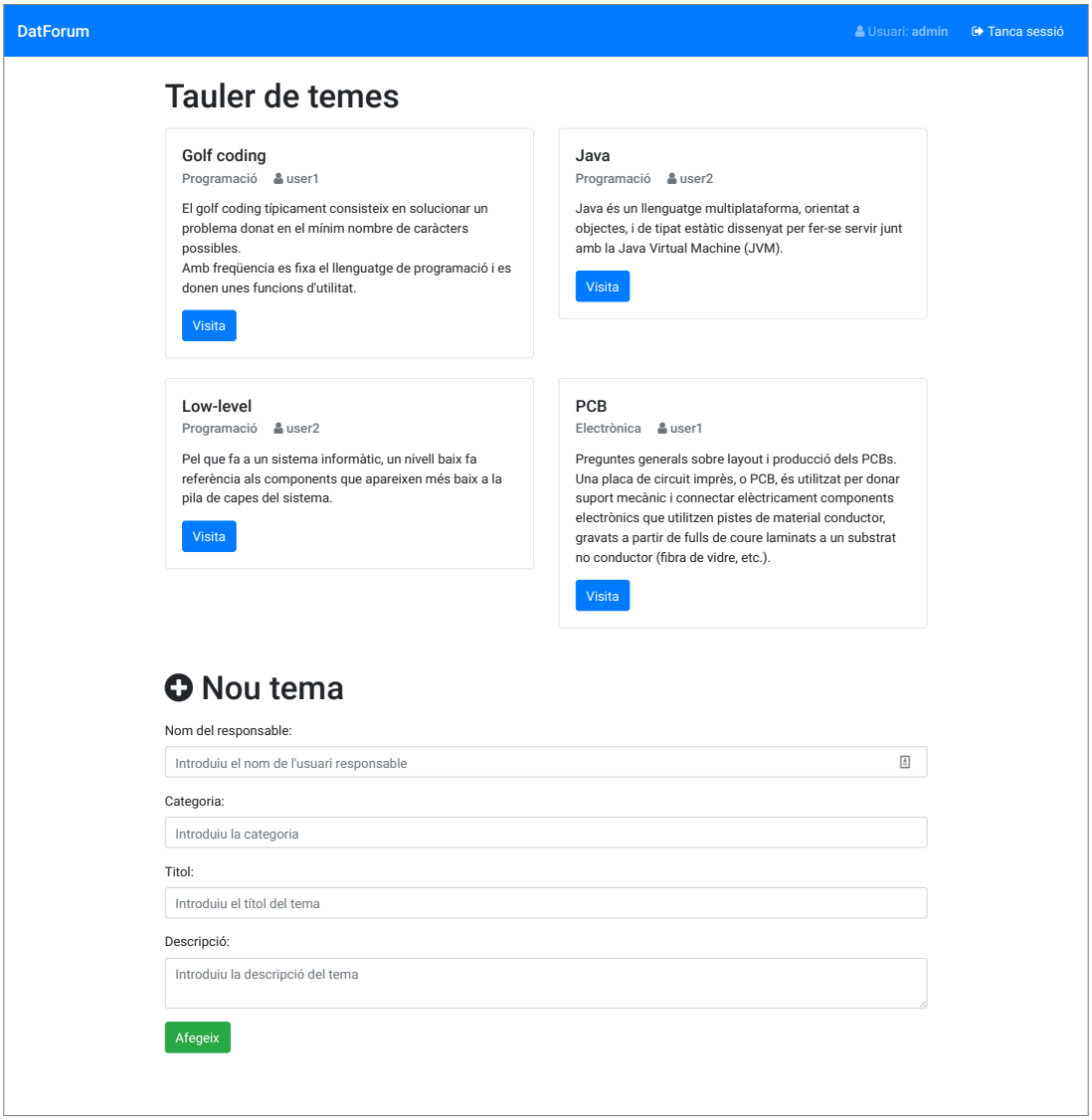

Figura 1: Vista principal (*home*) per al webmaster.

6 Vista *question*

Ara seguirem amb la vista *question*. Farem aquesta primer, ja que és menys complexa que la vista de tema, que té dos formularis.

Mètode GET

Creem manualment una pregunta amb dues respostes a la base de dades. Llavors definim el formulari per afegir una resposta, qué només consta d'un camp (el text):

```
answerForm :: AForm (HandlerFor Forum) Text
answerForm = freq textareaField (withPlaceholder "Introduiu la text de
  ↪  la resposta" "Text") Nothing
```

Noteu que en aquest cas el formulari no retorna un `Answer` directament; per fer-ho hauriem de saber l'usuari autenticat, la data actual i l'ID de la pregunta. És més senzill construir aquest `Answer` directament en el moment d'afegir la resposta a la base de dades.

A continuació escrivim el codi per al mètode GET de la ruta:

```
getQuestionR :: ThemeId -> QuestionId -> HandlerFor Forum Html
getQuestionR tid qid = do
    -- Get model info
    db <- getsSite forumDb
    (theme, question) <- getThemeQuestion tid qid db
    answers <- liftIO $ getAnswerList qid db
    mbuser <- maybeAuthId
    aformw <- generateAFormPost answerForm
    -- Return HTML content
    defaultLayout $ $(widgetTemplFile
      ↪  "src/forum/templates/question.html")
```

El codi és molt similar al de la home, es limita a renderitzar el formulari i la vista. Però en comptes d'obtenir la llista de temes, obté la llista de respostes de la pregunta, i la pregunta i tema que es visita. Això últim es fa mitjançant la funció `getThemeQuestion`:

```
getThemeQuestion tid qid db = do
    question <- liftChecked $ getQuestion qid db
    unless (qTheme question == tid) notFound
    theme <- liftChecked $ getTheme tid db
    pure (theme, question)
```

Aquesta funció obté la pregunta i el tema de la base de dades, però a més comprova que la pregunta correspon al tema. Si això no és així, llavors la URL és incorrecta i cal retornar un 404.

Plantilla

En primer lloc mostrem la informació i text de la pregunta:

```
<h1 class="mt-3">#{qTitle question}</h2>
<h5 class="text-muted mb-3">
  <i class="fa fa-question-circle"></i>
  Preguntat a <a href="@{ThemeR tid}">#{tTitle theme}</a>
  per #{qUser question} el #{formatPosted (qPosted question)}
</h5>
<p class="lead" style="white-space: pre-line">#{qText question}</p>
```

On la funció `formatPosted` l'hem definit així:

```
formatPosted :: UTCTime -> String
formatPosted = formatTime defaultTimeLocale "%Y-%m-%d %H:%M"
```

A continuació (abans de tancar el container) mostrem la llista de respostes:

```
<h2 class="mt-4">#{show (length answers)} respostes</h2>
<div>
  $forall{ p <- answers }
  <div class="my-4">
    <div class="card">
      <div class="card-body">
        <h6 class="card-subtitle mb-3 text-muted">
          <span><i class="fa fa-clock-o"></i> #{formatPosted (aPosted
            ↪ (snd p))}</span>
          <span class="mx-3"><i class="fa fa-user"></i> #{aUser (snd
            ↪ p)}</span>
        </h6>
        <p class="lead mb-0" style="white-space: pre-line">#{aText
          ↪ (snd p)}</p>
      </div>
    </div>
  </div>
  $end
</div>
```

Com que s'han de poder esborrar preguntes, ficarem el codi anterior dins un `<form>` amb un botó de submit (que només es mostra si hi ha respostes i l'usuari és leader):

```
<form role="form" method="POST" action="@{QuestionR tid qid}">
  <!-- ... -->

  $if{ null answers }
  $elseif{ maybe False (isLeader theme) mbuser}
  <div class="row">
    <div class="col-sm-12">
      <button type="submit" class="btn btn-danger"
        ↪  name="delete">Elimina respostes</button>
    </div>
  </div>
  $end
</form>
</div>
```

...i en cas de que l'usuari sigui un leader, afegirem checkboxes al prinicpi de cada resposta per seleccionar-les:

```
<h6 class="card-subtitle mb-3 text-muted">
  $if{ maybe False (isLeader theme) mbuser }<input type="checkbox"
    ↪  name="aid" value="#{fst p}"/>$end
  <span><i class="fa fa-clock-o"></i> #{formatPosted (aPosted (snd
    ↪  p))}</span>
  <span class="mx-3"><i class="fa fa-user"></i> #{aUser (snd
    ↪  p)}</span>
</h6>
```

Per últim, en un nou contenidor mostrem el formulari:

```
$if{ isJust mbuser }
<div class="container my-4">
<h2 class="my-3"><i class="fa fa-plus-circle"></i> Afegeix una
  ↪  resposta</h2>

<form role="form" method="POST" action="@{QuestionR tid qid}">
  <div class="row">
    <div class="col-sm-12">
      ^{aformw}
    </div>
  </div>
  <div class="row">
```

Figura 2: Vista d'una pregunta (*question*) per al leader.

```
    <div class="col-sm-12">
      <button type="submit" class="btn btn-success"
        ↪  name="add">Crea</button>
    </div>
  </div>
</form>
</div>
$end
```

El codi és pràcticament idèntic al de la secció anterior, però en aquest cas la condició és simplement que l'usuari estigui autenticat i la ruta del formulari es diferent.

A més, s'ha afegit `name` als botons de cadascun dels formularis, per poder saber quin d'ells s'ha enviat.

El disseny resultant es pot veure a les figures 2, 3 i 4.

Figura 3: Vista d'una pregunta (*question*) per a un usuari autenticat.

Figura 4: Vista d'una pregunta (*question*) per a un usuari no autenticat.

Mètode POST

Per a implementar el mètode POST, en primer lloc com és habitual obtindrem la informació de la base de dades i l'usuari que fa la petició:

```
postQuestionR :: ThemeId -> QuestionId -> HandlerFor Forum Html
postQuestionR tid qid = do
    user <- requireAuthId
    db <- getsSite forumDb
    (theme, question) <- getThemeQuestion tid qid db
    answers <- liftIO $ getAnswerList qid db
```

A continuació mirarem quin dels botons s'ha premut per saber el formulari:

```
    deleteForm <- isJust <$> lookupPostParam "delete"
    addForm <- isJust <$> lookupPostParam "add"
    if deleteForm
      then do
        -- formulari 1
      else if addForm then do
        -- formulari 2
      else
        invalidArgs ["delete","add"]
```

Llavors, en cas que sigui el primer formulari: s'extreuen les IDs dels checkboxes que s'han premut, s'eliminen de la base de dades i es redirigeix l'usuari a la pàgina actual:

```
checkBoxes <- lookupPostParams "aid"
let aids = catMaybes ((readMaybe . T.unpack) <$> checkBoxes)
forM_ aids $ \ aid ->
    liftIO $ deleteAnswer aid db
redirectRoute (QuestionR tid qid) []
```

En el cas del segon formulari (answerForm), l'executem per validar els camps introduits. Si tot està bé, es crea la resposta en la base de dades i es redirigeix. Si no, es renderitza la pàgina amb els errors:

```
(aformr, aformw) <- runAFormPost answerForm
case aformr of
    FormSuccess text -> do
        time <- liftIO $ getCurrentTime
        let answer = Answer qid user time text
        liftIO $ addAnswer answer db
        redirectRoute (QuestionR tid qid) []
```

```
        _ -> do
            let mbuser = Just user
            defaultLayout $(widgetTemplFile
              ↪  "src/forum/templates/question.html")
```

Es comprova que tots els formularis funcionen correctament.

7 Vista *theme*

Mètode GET

Primer creem manualment quatre temes a la base de dades. Llavors hem de definir els dos formularis que es renderitzaran a la vista: el de modificar el tema, i el d'afegir una pregunta.

Respecte al de modificar el tema, és més senzill modificar `themeForm` (veure secció 3) perquè accepti un `Maybe Theme` i el faci servir per als valors per defecte dels camps. Per fer-ho, només hem de fer que aquests valors per defecte siguin `(camp <$> theme)` en comptes de `Nothing`:

```
themeForm :: Maybe Theme -> AForm (HandlerFor Forum) Theme
themeForm theme =
    Theme <$> maybe
                (freq (checkM checkUserExists textField)
                (withPlaceholder "Introduiu el nom de l'usuari
                  ↪  responsable" "Nom del responsable")
                Nothing)
              pure (tLeader <$> theme)
          <*> freq textField (withPlaceholder "Introduiu la categoria"
            ↪  "Categoria") (tCategory <$> theme)
          <*> freq textField (withPlaceholder "Introduiu el títol del
            ↪  tema" "Titol") (tTitle <$> theme)
          <*> freq textareaField (withPlaceholder "Introduiu la
            ↪  descripció del tema" "Descripció") (tDescription <$>
            ↪  theme)
```

Observeu que per al camp del leader el que es fa és mostrar-lo només si `theme` no està present, en cas contrari no es deixa modificar-lo (es fa servir el valor existent).

Llavors, reemplacem tots els usos de `themeForm` per `themeForm Nothing`, i per al formulari de modificació farem servir `themeForm $ Just theme`.

Respecte al formulari d'afegir una pregunta, aquest té dos camps que retornarem dins una tupla:

```
questionForm :: AForm (HandlerFor Forum) (Text, Text)
questionForm =
    (,) <$> freq textField (withPlaceholder "Introduiu l'assumpte de
      ↪  la pregunta" "Assumpte") Nothing
        <*> freq textareaField (withPlaceholder "Introduiu el text de
          ↪  la pregunta" "Text") Nothing
```

Ja podem escriure el mètode GET, que és molt similar a l'anterior, excepte que renderitzem tots dos formularis:

```
getThemeR :: ThemeId -> HandlerFor Forum Html
getThemeR tid = do
    -- Get model info
    db <- getsSite forumDb
    theme <- liftChecked $ getTheme tid db
    questions <- liftIO $ getQuestionList tid db
    mbuser <- maybeAuthId
    tformw <- generateAFormPost (themeForm $ Just theme)
    qformw <- generateAFormPost questionForm
    -- Return HTML content
    defaultLayout $ $(widgetTemplFile
      ↪ "src/forum/templates/theme.html")
```

Plantilla

Comencem mostrant el nom del tema i la descripció:

```
<h1 class="my-3">#{tTitle theme}</h2>
<p class="lead" style="white-space: pre-line">#{tDescription
  ↪ theme}</p>
```

A continuació mostrem les preguntes en cards, com es fa en la home:

```
<div class="row row-cols-1 row-cols-md-2">
  $forall{ p <- questions }
  <div class="col mb-4">
    <div class="card">
      <div class="card-body">
        <h5 class="card-title">
          #{qTitle (snd p)}
        </h5>
        <h6 class="card-subtitle mb-3 text-muted">
          <span><i class="fa fa-clock-o"></i> #{formatPosted (qPosted
            ↪ (snd p))}</span>
          <span class="mx-3"><i class="fa fa-user"></i> #{qUser (snd
            ↪ p)}</span>
        </h6>
        <p class="card-text" style="white-space: pre-line">#{qText
          ↪ (snd p)}</p>
```

```
          <a href="@{QuestionR tid (fst p)}" class="btn
            ↪ btn-primary">Visita</a>
        </div>
      </div>
    </div>
    $end
  </div>
```

Igual que en la secció anterior, per tal de poder esborrar preguntes, movem aquest bloc dins un un `<form>` amb un botó que només es mostra si hi ha preguntes i l'usuari és leader:

```
<form role="form" method="POST" action="@{ThemeR tid}">
  <div class="row row-cols-1 row-cols-md-2">
    <!-- ... -->
  </div>

  $if{ null questions }
  <p>Encara no hi ha preguntes.</p>
  $elseif{ maybe False (isLeader theme) mbuser}
  <div class="row">
    <div class="col-sm-12">
      <button type="submit" class="btn btn-danger"
        ↪ name="delete">Elimina preguntes</button>
    </div>
  </div>
  $end
</form>
</div>
```

I afegim checkboxes a cada pregunta per seleccionar-la:

```
<h5 class="card-title">
  $if{ maybe False (isLeader theme) mbuser }<input type="checkbox"
    ↪ name="qid" value="#{fst p}"/>$end
  #{qTitle (snd p)}
</h5>
```

A continuació, renderitzem el formulari d'afegir una pregunta (si l'usuari està autenticat) i el de modificar el tema (si l'usuari és leader):

```
$if{ isJust mbuser }
<div class="container my-4">
```

```
<h2 class="my-3"><i class="fa fa-plus-circle"></i> Crea una nova
  ↪ pregunta</h2>

<form role="form" method="POST" action="@{ThemeR tid}">
  <div class="row">
    <div class="col-sm-12">
      ^{qformw}
    </div>
  </div>
  <div class="row">
    <div class="col-sm-12">
      <button type="submit" class="btn btn-success"
        ↪ name="add">Crea</button>
    </div>
  </div>
</form>

</div>
$end

$if{ maybe False (isLeader theme) mbuser }
<div class="container my-4">
<h2 class="my-3"><i class="fa fa-pencil"></i> Modifica el tema</h2>

<form role="form" method="POST" action="@{ThemeR tid}">
  <div class="row">
    <div class="col-sm-12">
      ^{tformw}
    </div>
  </div>
  <div class="row">
    <div class="col-sm-12">
      <button type="submit" class="btn btn-primary"
        ↪ name="modify">Modifica</button>
    </div>
  </div>
</form>

</div>
$end
```

Igual que en la secció anterior, s'ha afegit `name` als botons per diferenciar quins dels tres formularis s'ha enviat.

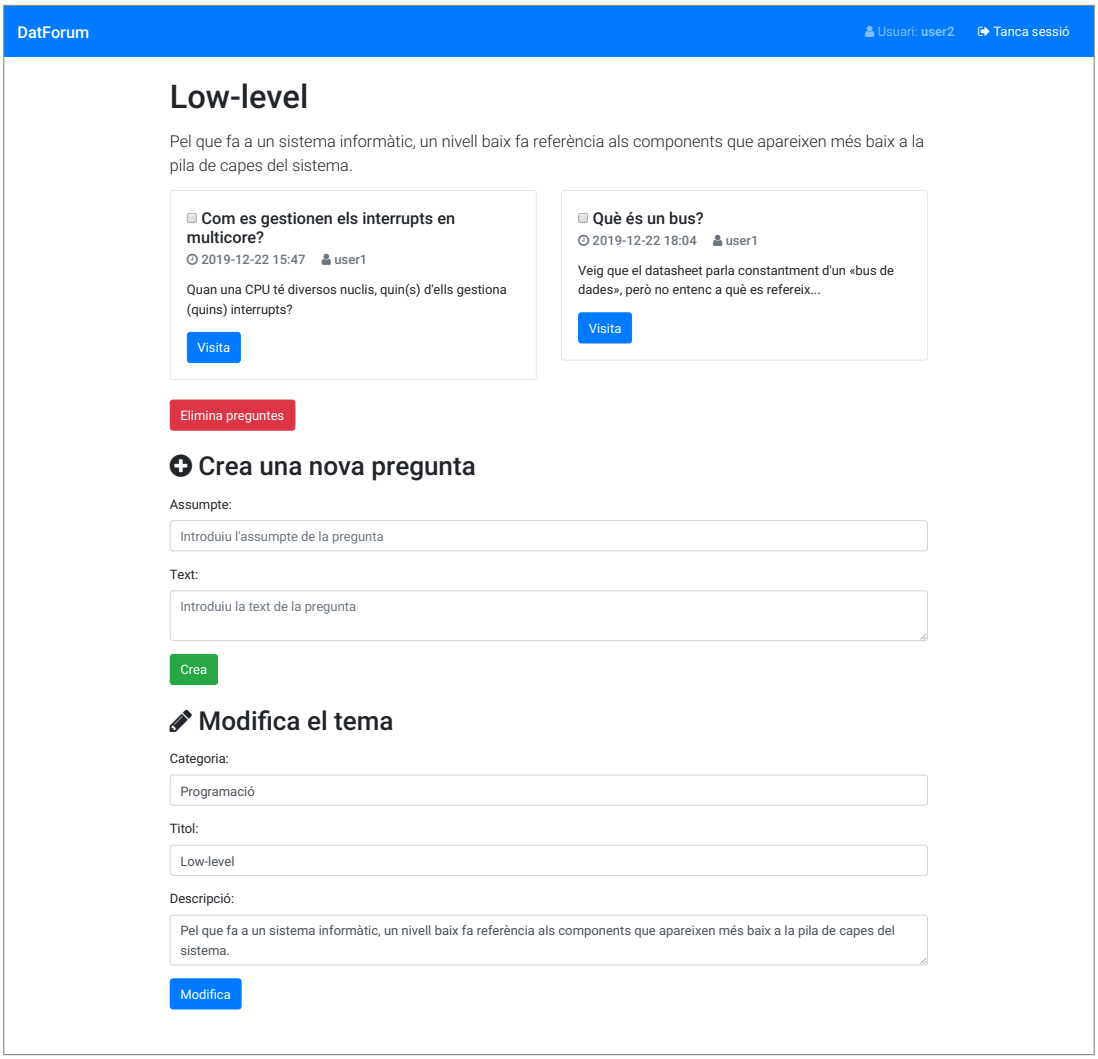

Figura 5: Vista d'un tema (*theme*) per al leader.

El disseny resultant es pot veure a la figura 5.

Mètode POST

L'estructura del mètode POST és molt similar a la de la secció anterior, obtindrem la informació i discriminarem entre els tres formularis possibles:

```
postThemeR :: ThemeId -> HandlerFor Forum Html
postThemeR tid = do
    user <- requireAuthId
    db <- getsSite forumDb
    theme <- liftChecked $ getTheme tid db
    questions <- liftIO $ getQuestionList tid db
    modifyForm <- isJust <$> lookupPostParam "modify"
    deleteForm <- isJust <$> lookupPostParam "delete"
    addForm <- isJust <$> lookupPostParam "add"
    if modifyForm
      then do
        -- formulari 1
      else if deleteForm then do
        -- formulari 2
      else if addForm then do
        -- formulari 3
      else
        invalidArgs ["modify","delete","add"]
```

Pel cas de modificació del tema, com hem fet abans, validem els camps del formulari. Si estan bé, es modifica el tema a la base de dades i es redirigeix. Sino es torna a renderitzar la pàgina, però abans cal renderitzar també l'altre formulari:

```
(tformr, tformw) <- runAFormPost (themeForm $ Just theme)
case tformr of
    FormSuccess newtheme -> do
        liftIO $ updateTheme tid newtheme db
        redirectRoute (ThemeR tid) []
    _ -> do
        qformw <- generateAFormPost questionForm
        let mbuser = Just user
        defaultLayout $(widgetTemplFile
          ↪  "src/forum/templates/theme.html")
```

Pel cas d'eliminació, de forma també similar a abans, s'obtenen els IDs dels *checkboxes* marcats, s'eliminen de la base de dades i es redirigeix:

```
checkBoxes <- lookupPostParams "qid"
let qids = catMaybes ((readMaybe . T.unpack) <$> checkBoxes)
forM_ qids $ \ qid ->
    liftIO $ deleteFullQuestion qid db
redirectRoute (ThemeR tid) []
```

Pel cas d'afegir una pregunta se segueix un procediment similar al primer cas:

```
(qformr, qformw) <- runAFormPost questionForm
case qformr of
    FormSuccess (title, text) -> do
        time <- liftIO $ getCurrentTime
        let question = Question tid user time title text
        qid <- liftIO $ addQuestion question db
        redirectRoute (QuestionR tid qid) []
    _ -> do
        tformw <- generateAFormPost (themeForm $ Just theme)
        let mbuser = Just user
        defaultLayout $(widgetTemplFile
          ↪  "src/forum/templates/theme.html")
```

Comprovem que tots els formularis funcionen correctament.

8 Comprovacions de seguretat

Cal observar que, a banda de comprovar que l'usuari estigui autenticat, els mètodes POST no verifiquen que l'usuari sigui webmaster o leader en les operacions que ho requereixen. Això és el que afegirem ara.

En primer lloc, al mòdul **Found**, definirem funcions d'utilitat que comproven els permisos de l'usuari, denegant la petició si no es compleixen:

```
requireAdmin :: MonadHandler m => UserId -> m ()
requireAdmin u =
    unless (isAdmin u) (permissionDenied "User is not admin")

requireLeader :: MonadHandler m => Theme -> UserId -> m ()
requireLeader t u =
    unless (isLeader t u) (permissionDenied "User is not leader")
```

Ara només cal que fem servir aquestes funcions en els handlers. En el de la home:

```
postHomeR = do
    user <- requireAuthId
    db <- getsSite forumDb
    requireAdmin user
    (tformr, tformw) <- runAFormPost themeForm
    case tformr of
        -- ...
```

En el de la vista de tema:

```
postThemeR tid = do
    user <- requireAuthId
    db <- getsSite forumDb
    -- ...
    if modifyForm
      then do
        requireLeader theme user
        (tformr, tformw) <- runAFormPost (themeForm $ Just theme)
        case tformr of
            -- ...
      else if deleteForm then do
        requireLeader theme user
        checkBoxes <- lookupPostParams "qid"
        -- ...
      else if addForm then do
```

```
        (qformr, qformw) <- runAFormPost questionForm
        case qformr of
            -- ...
      else
        invalidArgs ["modify","delete","add"]
```

En el de la vista de pregunta:

```
postQuestionR tid qid = do
    user <- requireAuthId
    db <- getsSite forumDb
    -- ...
    if deleteForm
      then do
        requireLeader theme user
        checkBoxes <- lookupPostParams "aid"
        -- ...
      else if addForm then do
        (aformr, aformw) <- runAFormPost answerForm
        case aformr of
            -- ...
      else
        invalidArgs ["delete","add"]
```

També cal que, quan s'eliminen preguntes, validem que pertanyen al tema visitat:

```
requireLeader theme user
checkBoxes <- lookupPostParams "qid"
let qids = catMaybes ((readMaybe . T.unpack) <$> checkBoxes)
let valid = fromList $ map fst questions
unless (fromList qids `isSubsetOf` valid) notFound
forM_ qids $ \ qid ->
    liftIO $ deleteFullQuestion qid db
redirectRoute (ThemeR tid) []
```

I el mateix quan s'eliminen respostes:

```
requireLeader theme user
checkBoxes <- lookupPostParams "aid"
let aids = catMaybes ((readMaybe . T.unpack) <$> checkBoxes)
let valid = fromList $ map fst answers
unless (fromList aids `isSubsetOf` valid) notFound
forM_ aids $ \ aid ->
    liftIO $ deleteAnswer aid db
```

```
    redirectRoute (QuestionR tid qid) []
```

Observeu que en aquest cas només comprovem que pertanyen a la pregunta visitada. Això és suficient, perquè la funció `getThemeQuestion` ja comprova que aquesta pregunta pertany al tema visitat.