UD5. Activity 1

MULTIMEDIA CONTENT IMPLEMENTATION

Toni Barceló García

IES JOAN RAMIS I RAMIS | DAW

Tabla de contenido

UD5. Activity 1 (Part 1)

In this activity we are going to work with indexedDB to manage the users of our blog. Until now we only had blogger users, but we are going to add another user type: an administrator.

To do this we must add a checkbox on the registration page to mark whether the user to be registered is a standard user or an administrator.

In addition, when registering we must check the values entered by users so that the fields are not empty, that the format of an email is correct and that the password contains at least 8 characters, containing lowercase and uppercase letters, a number and a special character. The password will not be stored in plain text, but we will use an encryption system that uses md5. There are several libraries to use this function, such as CryptoJS.

When a user registers, they will automatically be logged in as well. To do this we must save in our database which user is logged in.

- If the logged in user is a standard user then we redirect to the main page and display the user name and avatar according to the initial design.
- If the logged in user is an administrator, we redirect to a new page with the list of created users with all their information (except the password).

Remember that when the homepage is loaded, it is necessary to check if there is a logged in user or not and redirect us to the corresponding page. Likewise, if we access the url of the administrator user, if there is no user of this type logged in, the page should redirect us to the homepage.

In both cases there will be an option to **log out**.

Actions that users can perform:

Administrator. On the administrator page, where we have the list of users, we will be able to do the following: edit user's data, reset the password and delete a user's account. Remember to ask for confirmation to perform these actions that cannot be undone.

Standard user and administrator. On the main page, in addition to displaying the user name and avatar, there will be an option to edit the user's profile and other user preferences. There, the user will be able to edit their personal data, change the password, change the avatar, delete their user account and change the theme of the page (light or dark).

- If a user deletes the account, he/she automatically has to log out.
- If the colour of the theme is changed, the change must be applied.

We will separate the deliverables for this task into **two parts**:

- In the first one, the entire user registration and login function will be done with the respective redirections: main page for standard users with the display of name and avatar; new page with the list of created users for administrators.
- In the next part we will add the actions of both pages: editing users and profiles and logging out.

Source Github project

If you clone the project, you will have all the activities I have done so far sorted by folders in “activities” root directory.

Source project:

https://github.com/abarcelogarcia/abarcelogarcia.github.io

Root folder for this activity: UD5. Activity 1 (Part 1)

activities/UD5A_1_IndexedDB (Part 1)

github-pages (latest version)

https://abarcelogarcia.github.io/

The website structure

Directory	Files	Concept
root	index.html	main website file.
css	avatar_effect.css	css file to apply avatar’s effect without js
	bootstrap_custom.css	css file genered by sass
	bootstrap_custom.css.map	Css map file genered by sass
	bootstrap_custom.scss	sass file code
js	common.js	Common JS file
	form_validation.js	form validation functions
	index_admin.js	functions for admin page
	index.js	functions for home page
	crypto.js	functions for encrypt the password
fonts	BAHNSCHRIFT.TTF	typography chosen in the guide style.
img	*.png, *.jpg	images directory for the web.
node_modeules	*	Bootstrap sass modules
views	post.html	Post file

Anotations:

The name of the indexedDB database is **blogginDB.**

In the database there are 2 Objectstorage.

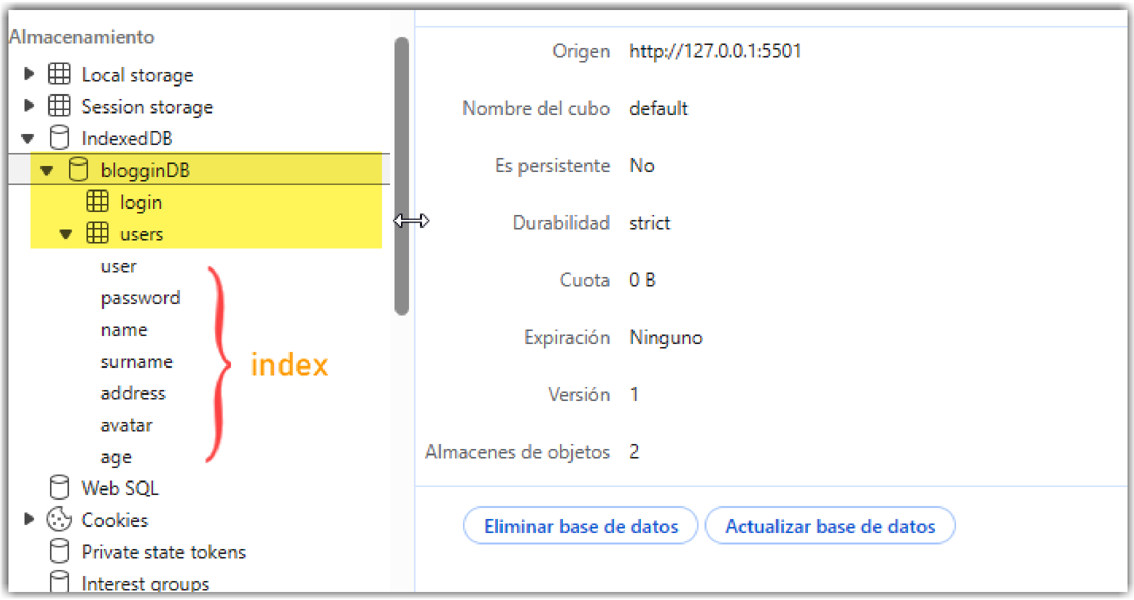
- **Users**: stores the users.
- **Login**: stores the user who has logged in.

When creating the database, in the users' storage I create an index for each form field so that in future updates it will be easier to search for a specific field.

The admin page is *index_admin.html*

COMMON

On all pages I make two jobs. First, I open or create the indexedDB database and check if there is any user created in the *login* storage.



Origen	http://127.0.0.1:5501
Nombre del cubo	default
Es persistente	No
Durabilidad	strict
Cuota	0 B
Expiración	Ninguno
Versión	1
Almacenes de objetos	2

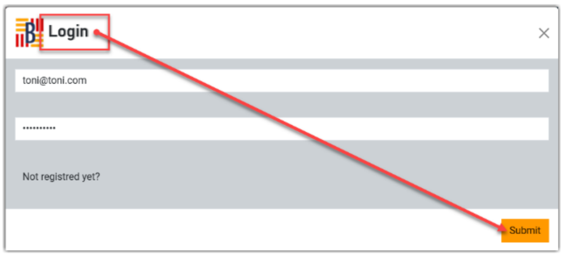
For users control, I check each time the page loads with an *addeventlistener*. I run a function that after creating/opening the db, checks if you are logged in. If you are not logged in (there is no record) it redirects to the home page. If there is a record, depending on whether it is admin or not, it does one or another action. To distinguish it according to the page it loads, I have added a parameter if it is an admin page or not.

There is also a logout function that removes registers from the login storage.

Registration/login

Index.js

On the homepage you will find the login/register form in a modal. In this modal you can either log in or register. If the registration data is not displayed, it is treated as a login, but as a registration. For this I have added my own attribute (**action**) which I pass as a parameter to the function.



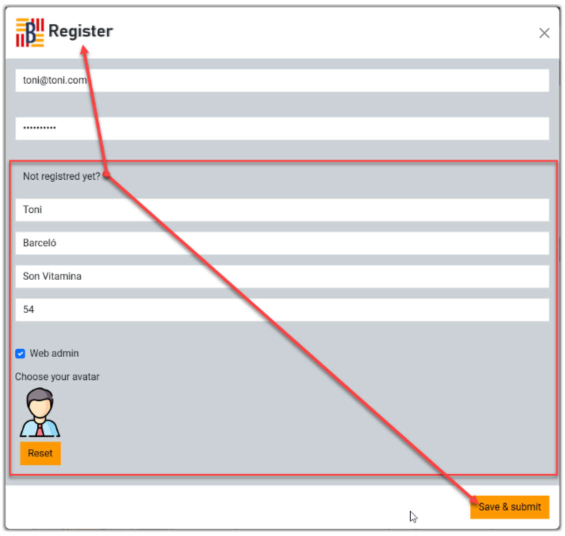
Login

toni@toni.com

....

Not registred yet?

Submit



Register

toni@toni.com

....

Not registred yet?

Toni

Barceló

Son Vitamina

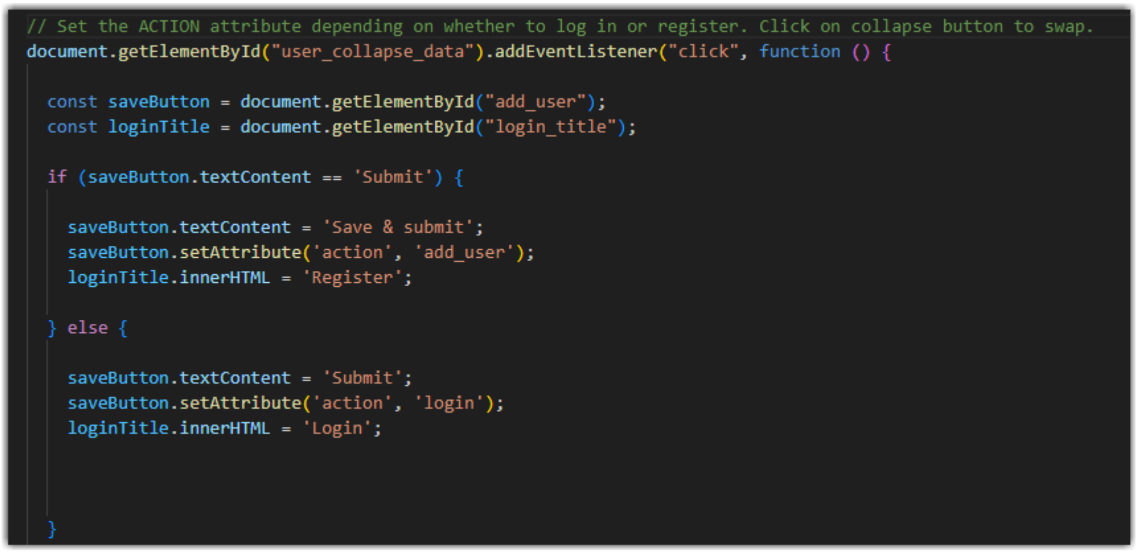
54

☑ Web admin

Choose your avatar

Reset

Save & submit



```
// Set the ACTION attribute depending on whether to log in or register. Click on collapse button to swap.
document.getElementById("user_collapse_data").addEventListener("click", function () {

  const saveButton = document.getElementById("add_user");
  const loginTitle = document.getElementById("login_title");

  if (saveButton.textContent == 'Submit') {

    saveButton.textContent = 'Save & submit';
    saveButton.setAttribute('action', 'add_user');
    loginTitle.innerHTML = 'Register';

  } else {

    saveButton.textContent = 'Submit';
    saveButton.setAttribute('action', 'login');
    loginTitle.innerHTML = 'Login';



  }
```

After pressing the button, first of all validate email and password fields, then if it is a new register (***action = add_user***) I execute the method that adds the record from the database (*addUser(db)*)and at the end I execute the *login(db)* method that register the login in *ObjectStore login* (*setLogin()*)and redirects him/her to the corresponding web. If it is a login (***action = login***), I execute the *login(db)* method directly.

New register
validateForm(this.getAttribute('action')) -> addUser(db) -> login(db) -> setLogin(user, admin, avatar)

Login
validateForm(this.getAttribute('action')) -> login(db) -> setLogin(user, admin, avatar)

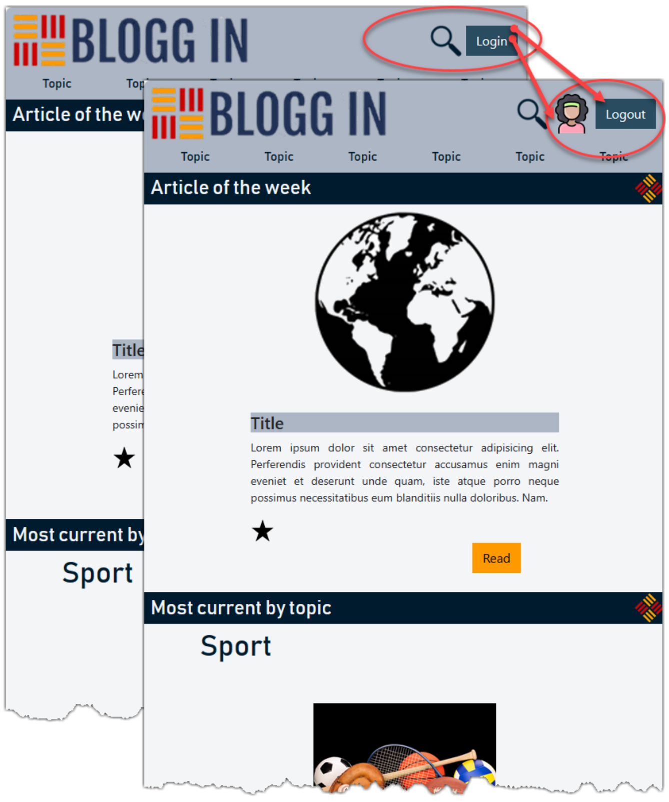
Logged

User

When a user (non-admin) is logged in, it redirects you to the home page, shows their resgistered avatar and reformulates the login button to have logout functionality.

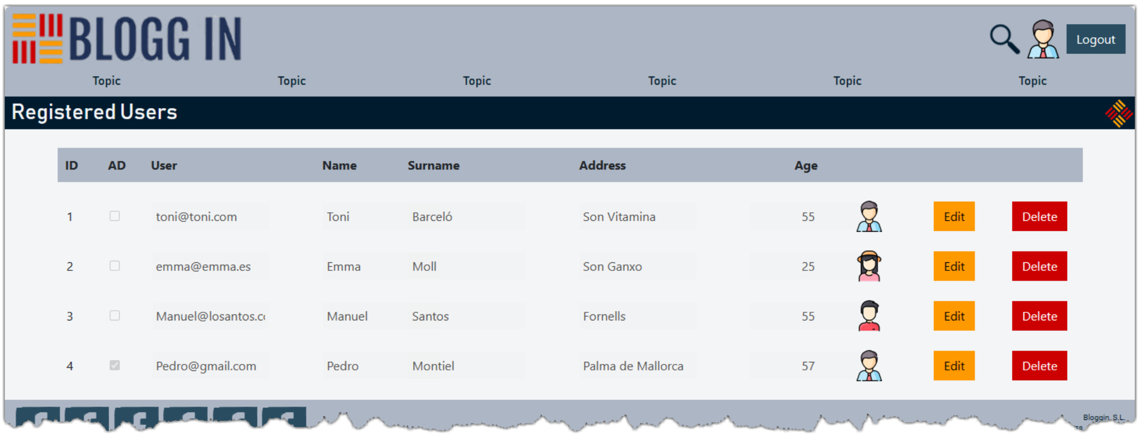
Index.html onLoad -> verifyUser("")->openCreateDb(onDbCompleted)

->setUser(db).


BLOGG IN
Login
Logout
Topic
Topic
Topic
Topic
Topic
Topic
Article of the week
Title
Lorem ipsum dolor sit amet consectetur adipisicing elit. Perferendis provident consectetur accusamus enim magni eveniet et deserunt unde quam, iste atque porro neque possimus necessitatibus eum blanditiis nulla doloribus. Nam.
Read
Most current by topic
Sport

Admin

When an admin user logs in, he is redirected to index_admin.html which, after verifying his identity, reads the db and displays a list of all records. It also adds his avatar and rephrases the login button to logout.



Registered Users

ID	AD	User	Name	Surname	Address	Age			
1	☐	toni@toni.com	Toni	Barceló	Son Vitamina	55		Edit	Delete
2	☐	emma@emma.es	Emma	Moll	Son Ganxo	25		Edit	Delete
3	☐	Manuel@losantos.c	Manuel	Santos	Fornells	55		Edit	Delete
4	☑	Pedro@gmail.com	Pedro	Montiel	Palma de Mallorca	57		Edit	Delete

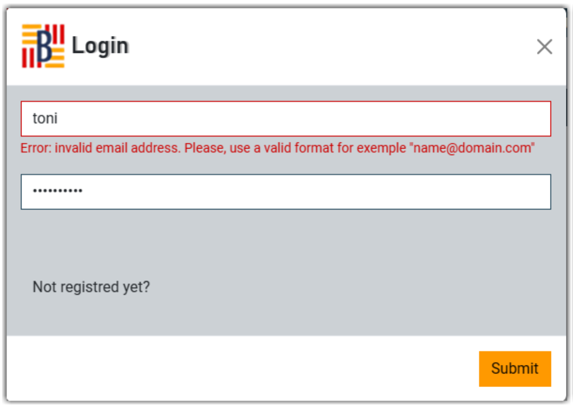
Index_admin onLoad -> verifyUser ("admin") -> readData()->

openCreateDb(onDbCompleted)->readUsers(db)

Validation Form

Form validator

For the validation of the form (user and password), I have the file form_validator.js that contains the methods and error messages when the email format is not met and if the password is insecure.



Login

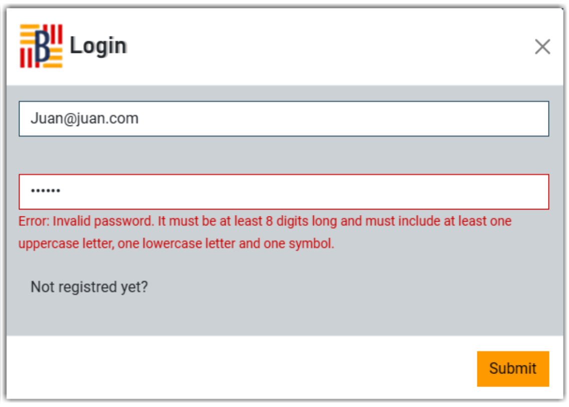
toni

Error: invalid email address. Please, use a valid format for exemple "name@domain.com"

••••••••••

Not registred yet?

Submit



Login

Juan@juan.com

••••••

Error: Invalid password. It must be at least 8 digits long and must include at least one uppercase letter, one lowercase letter and one symbol.

Not registred yet?

Submit

If the email and password are valid, continue with the registration or login.