UNIT 3

Ionic

Final exercise

Client-side Web Development
2nd course – DAW
IES San Vicente 2024/2025
Author: Arturo Bernal Mayordomo

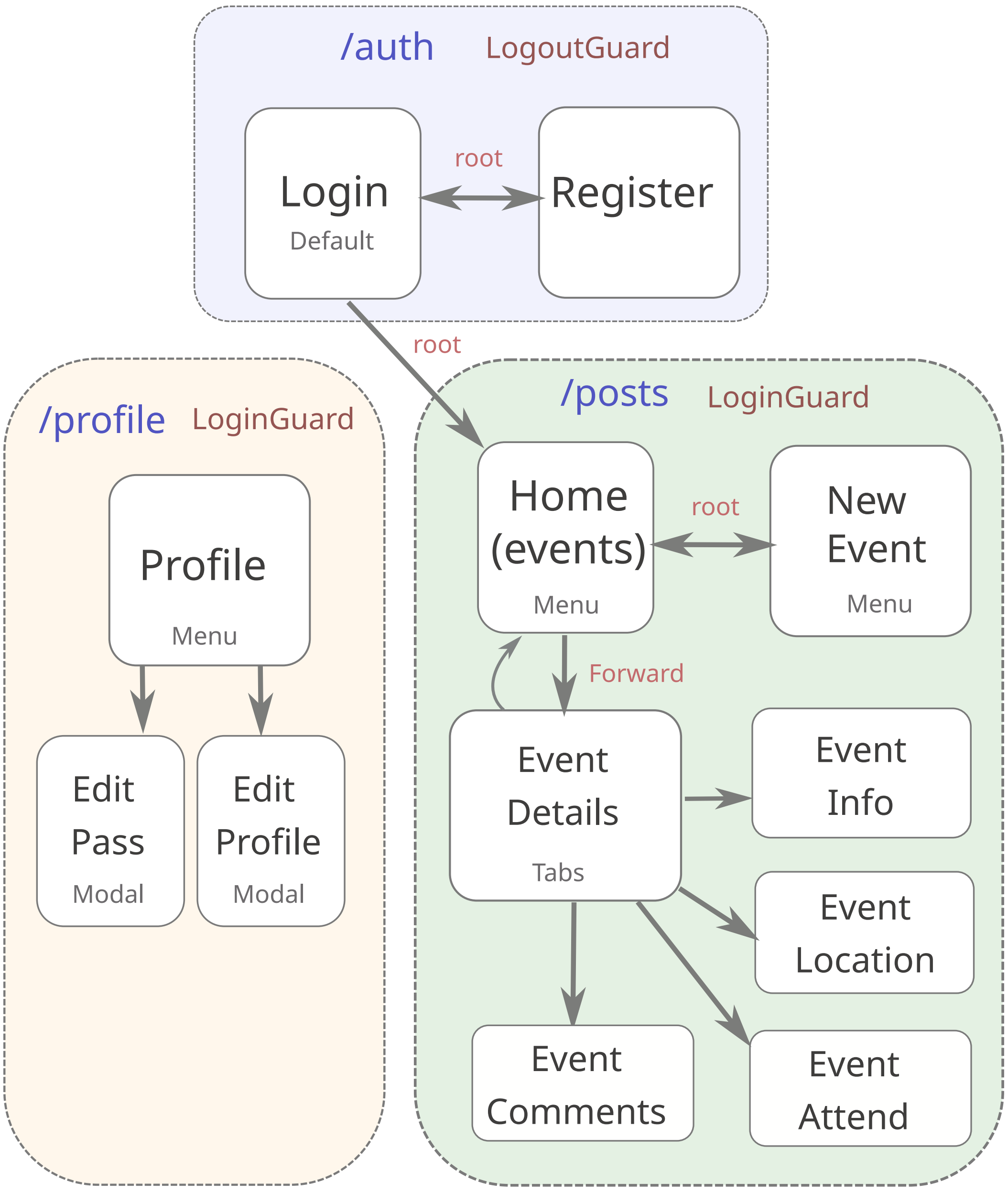
Index

Project structure

In this project, you'll have to create the pages/sections described below (some of them can be placed inside modal windows for example). This structure serves as a guide but it could be a little different as long as it's intuitive and works well.

The app must be configured as **zoneless**. Use signals (or sometimes ChangeDetectorRef) to detect changes in asynchronous operations.

Create also separate route files for routes with different prefixes: /**auth**, /**profile**, /**events**, like you did in the previous unit project. See products example.



Project sections

This project will connect to the same services and have more or less the same functionality as the previous unit (Angular project). Code from services, guards, interceptors, pipes, validators, etc can be used directly in this project with very little changes (like using Preferences plugin instead of localStorage, for example).

Login

In this section, you'll have a form where the user must enter **email** and **password**. There will be also a button to log in with **Google** and another to do the same using the **Facebook** SDK (use Capacitor plugin). This page won't have a menu button, and there will be a button to go to the **Register** page.

Get user's geolocation (use Ionic Native's plugin) at the beginning and send it with the login info like you did in the previous unit project.

If the user is already logged in, he/she will be redirected to **Home (events) page**. That's what the **LogoutGuard** should do automatically.

Register

Here the user will have to post a name, an email, a password (repeat it) and a photo (from the gallery or camera). Geolocation information (lat, lng) will also be sent to the server.

The avatar must have a size of 200x200 and the option to edit the photo set to true (so the user can crop the image).

Home (events)

In this page, you'll get the events from the server and display them using **cards** (ion-card). Also, put a fab button to go to the **New event** page.

In the header, add a toolbar with a search bar to filter events. Use a debounce time of 600ms and call the server with the search parameter (page 1).

Implement "pull to refresh" to reload events. Send the search string when reloading events.

Implement "infinite scroll" to load more events (next page) when the user is near the last one. Disable it when there are no more events (pages) to load.

The event card must show the same information as in the previous project, except the attend button (see event details), and the same functionality (delete button, etc.) but you can use a different approach like a popup menu, action sheet etc, to show the actions to the user, for example. You must be able to navigate to the events details page or the user's (creator) profile page from this card. Edit an event is optional.

Implement the card inside a separate component (**event-card**).

New event

It will have a form where you'll insert the necessary data to create a new event (title, description, image, price, date, address and coordinates). Use the same form validators as in the Angular project.

You can display a map here or open a modal window with a map to choose a location. See the Capacitor examples (Navigation) and the Angular integrations with OpenLayers and Geoapify.

The photo must have a width of 1024px and height of 768, and the option to edit it set to true.

Event detail

This page will have 4 tabs that will show information about the selected event, and a button in the header to go back to the Home page.

Event info

Here, event's information will be displayed. Use the same card as in the Home page (event-card component), with the same functionality. Deleting a event from here will redirect to the Home page.

Event location

Show a map with a marker where the event takes place. Put also a FAB button which will open the Navigation system in the phone, so you can go there by car. In the header, below the empty toolbar, add another toolbar with a label showing the event's address there.

Event Attend

Here we will show a button or other mechanism to confirm or cancel we are attending the event. We'll also show how many people attend the event and the list of users that are attending it (you should be able to go to a user profile in the list from here). When you confirm or cancel your attendance, reload the list (not the page).

Event comments

This tab will only show when the user is attending the event.

Show comments for the current event. Add a button to create a new comment. For adding a new comment use a modal page, a popup or an alert. If you choose alert, keep in mind that alert inputs don't have validation, so you should check in this case that the text is not empty before sending to the server.

Profile

In this page a user information will be displayed (name, email, photo and a small map with the user location). It can be the logged user's profile or any user profile (check if there's an id).

If the current profile is yours (logged user), you'll have an option to change your password, name and email, and your profile image (use buttons for gallery/camera). To change name/email (Edit profile) and password (Edit pass) you must use **modal windows** or at least **popups**, because you need to validate the forms.

General considerations

- The menu will be disabled if the user is not logged in (login, register pages), and will be enabled once the user is logged in. Similar to what we did in the Angular project but using the disabled attribute.
- Every time you want to test new code in your phone, update Capacitor project first:
    - **ionic build && npx cap sync**
- Use ionic components instead of normal HTML tags (when possible)
    - Use Ionic icons (do not install other icons font)
- When you delete any event that is yours, ask the user for a confirmation (alert, action sheet, etc) first!.

Sharing data between tabs

Like in the products example, the easiest way to share the event info between the **Event detail** page and its children is storing it in a signal inside the event-detail component, and injecting the component inside the child page's components. See the products example for more details.

Project marks

- Login page **1p**
- Register **0,5p**
- Home (Events page) **2p**
- New event **1,5p**
- Event detail (info, location, attend and comments) **2p**
- Profile and edit profile **2p**
- Code, use of Ionic (including capacitor) components, usability. **1p**

Optional (up to 2 extra points)

- Add the possibility to edit an event (if it's yours) reusing the new event component **0,5p**
- Implement a button in the Home page next to the search bar. When you click that button show an alert to choose how to order the events like in Angular's project (default: distance). When the user selects one, call the server with the new parameter. **0,5p**.
- In users profiles, put a link or button to see the events created by this user, and another to see the events this user is attending. **0,5p**
- Implement Push notifications (**0,5p**). A user will receive a push notification when someone has commented in an event he/she has created:
  - In the login, send the firebase token in an optional property called **firebaseToken**.
  - When login out in the application, also call the /**auth**/**logout** service (GET) in the server. It will remove the firebase token from the database, so the push notifications won't arrive to the phone.
  - When a push notification is received and the app is open, show a Toast message with the title and message (body).
  - When a push notification is received and the app is closed or in background, redirect to the comments page of that event. In the notification data, there will be a property called id that contains the event's id.