LeadQuizzes FRONTEND coding challenge

This assessment consist of 2 challenges:

- Angular 4 assessment
- Html 5 and Css 3 (SCSS) assessment

Note: The project codebase should be hosted on Bitbucket and after the assessment is finished, please add users: read access.

Angular 4 assessment

Task

`Make a single-page application in Angular (4). Show of your skills.`

Description

`Write a Quiz application that should contain these pages and functionalities:`

- Quizzes page – contains a list of quizzes (image#1). Here one can see:
    - Status of quiz in the top left corner that shows the published or unpublished status of the quiz (it shown on hover).
    - Title of the quiz on the bottom of the quiz box.
    - When hovering the box the title holder will scroll up and you will have 3 buttons there: Publish/Unpublish, Edit and Delete quiz (image#4).
        - When user clicks buttons “Publish/Unpublish” from that menu – he should get a popup asking him to confirm his action: “Are you sure that you want to [publish][unpublish] quiz “[quiz_name]”? Yes || No” . Change text “Publish” or “Unpublish” depending on the action, and also change the “[quiz_name]” to be the name of the quiz you are trying to publish/unpublish.
        - When user clicks button “Delete” from that menu – he should get a popup asking him to confirm his action: “Are you sure that you want to delete quiz “[quiz_name]”? Yes || No”. Change the “[quiz_name]” to be the name of the quiz you are trying to delete.
- Navigation – links:
    - “Quizzes” – shows the dashboard with list of quizzes (image#1),
    - “New Quiz” – shows the page for creating a new quiz (image#3).

- CRUD functionality for Quizzes (Create / Read / Update / Delete quiz).
- Add / Edit pages - make a real-time preview part on the right side of the screen (image#3) so when you type in the inputs on the left side it should show on the right side in that preview part. Left side is the part for quizzes Basic Info (id, name, title, description, button name, image URL).
    - On Add / Edit page there is a "Questions" button which will take you to the page where you can add questions and answers for that chosen Quiz (image#4). This part is more flexible and you do not need to make a live-preview part as for the Basic Info part above. Just make a nice way of adding multiple questions and answers for that quiz
- User should not be able to Edit the quiz (go to the page for Editing a quiz) if the Quiz is not Published. So, prevent the user from going to that route (use guard) and inform him that he can't access that route until he publishes that quiz.

Images

- Image#1: Quizzes page
- Image#2: Add / Edit quiz page basic info
- Image#3: Add / Edit quiz page questions and answers
- Image#4: Quiz Box when not hovered and when hovered. Here is the GIF as well.

Suggestions

- Components
- Routing and Guards
- Modules
- Models
- Services (services injected in services like maybe LogService, etc)
- Forms and Validation
- Frontend part (Bootstrap, Custom or other) - it does not need to be perfect, but keep it clean.
- Backend - use any type of API for storing data (Custom, MockBackend and MockConnection, store it in objects or cookies, etc), since this is more a frontend test than a backend test
- Cross-Component Communication / Passing and using data (Input, Output, ViewChild, Services, Emitters, Observables, etc)
- Structure/nesting of the components (decoupling)
- Shared folder for all components used everywhere.
- Environments files.
- Lifecycle Hooks [Optional]
- Tests [Bonus]
- etc.

Note

- Images above are only an example, it does not need to look exactly like that.

- Suggestions above are just some suggestions for the project. You can do the project any way you see fit and think of the best approach, structure and solutions for this project.
- The structure does not need to be exactly like on the images, feel free to make your structure, just be sure to make it clean and to provide the functionality that is needed.
- Particularly appreciated is clean code, good structure and comments. Code supported by the tests is a plus of course.

Html 5 and Css 3 (SCSS) assessment

1. When hovering the Quiz Box on Dashboard page the title holder needs to scroll up with a nice transition and there will be 3 buttons there: Publish/Unpublish, Edit and Delete quiz. Check out the **[GIF]** in order to see the transition, but you can also see how the box should look like when hovered and not hovered (image#4).
2. On Live Preview – content should always be centered - horizontally and vertically (Image#2 and Image#3).3. Depending on the content size – expand the centered Live Preview holder (Image#2 and Image#3).
3. All content should be responsive until a certain width that makes sense (768px).
4. On Dashboard you have 2 headers. Both headers should be fixed position. But the content part bellow should be scrollable (Image#1).
5. Put Loaders on certain actions and buttons.
6. Make interesting transitions to elements and actions. Left to your imagination.
7. Use SCSS.
8. **Bonus**: *Make you own custom form elements: Dropdown, Checkbox, Radio Box, Inputs, etc. You can put another button in the header near the "Create Quiz" button that takes you to a page that holds 1 form with those custom elements*.