BIF2 - Web Frameworks - MEAN

What is the semester project?

- In addition to all the functionalities we've done during the lectures, we'll further extend the existing project.
- Goal is to create a puzzle game app (like the project in WebScripting), with Login, Register, Time, Scoring and HighScore, … components.
- This requires some restructuring and extending of the existing project.

Prerequisites

- First, you'll need to implement all previous exercises, to get a working foundation.
  - If you already completed all previous exercises, this prerequisite is already finished for you.
- Please get together in teams of 2 members. This project will be a team exercise.
- Make sure, that your project is working without errors.
- Check the "DB connection" to your InMemoryDB and create dummy data for testing.

What to do?

- Check the Puzzle-Game-Project requirements of WebScripting. This puzzle game will be the core functionality. If not already done, wrap the game logic in separate TypeScript classes.
- The part with the browser-detection will be omitted.
- Add a nav bar (either on top or on the left side) to cover login, signup, logo, FAQ, high score, profile, … and link the buttons to the corresponding pages.
- Implement the different parts of your Puzzle Game Website and make sure to test them.
- Fill parts without logic (FAQ) with realistic text. Make sure to search online for good examples.

Core Functionality

- Check the Puzzle-Game-Project requirements of WebScripting. This puzzle game will be the core functionality. If not already done, wrap the game logic in separate TypeScript classes.
- **DON‘T** use HTML-Tables as layout elements for the pictures. Use CSS or Angular components.
- The HTML site should **NOT** contain any static elements needed for the game (layout, pictures, ...). Only necessary buttons and labels. Make sure to load content dynamically via TypeScript logic.
- **DON‘T** use external npm packages or JavaScript libraries for Non-UI tasks, like login, signup, etc...
- **DON‘T** use the innerHTML property, use DOM methods for creating/removing of necessary elements.

Core Functionality

- Extend the components, to check if a user visiting the site is already logged in. (hint: create a service for this!)
- If a user is currently logged in, the signup button should not be displayed.
- If a user is currently logged in, the login button should change its caption to logout. When the logout button is clicked, the user will be logged out.
- If a user is currently logged in, display a button „Profile“ which links to the users profile.
  - The user profile shows information about the user, like username, additional information and the highest score ever achieved, after finishing a puzzle.
- Create a logo and always show it on the upper left.

Core Functionality

- Implement a timer, to check how long the user takes to solve a puzzle. Show the timer above the puzzle.
- The timer will start at 00:00:00 and measure the time a user needs to solve the puzzle.
- The time in seconds, a user needs to solve a puzzle, should be saved for the score calculation. The score is calculated via a simple formula:
  - 100 - secondsToSolveThePuzzle = score
  - If the score is < 0 it will be automatically 0 for this attempt.
- After solving the puzzle, the score for the logged in user will be calculated and saved in the DB.
- Only registered and logged in users get a score. Not logged in users will only see the timer.

Core Functionality

- It’s possible to play the puzzle game, no matter if the website visitor is currently logged in or not.
- When a logged in user solves a puzzle, the score is saved in the DB, if it’s better than the current score of this user.
- Create an FAQ page (get information about how this may look on the web) and fill it with realistic text samples. (hint: use Angular Material expansion panel for this: https://material.angular.io/components/expansion/overview)
- Create a HighScore Page and display the 10 usernames + scores with the highest scores of all users. Sort them, starting with the highest score (first place).

Upload to Moodle

- Upload a ZIP file with your group number and names of the group members into the moodle course (section **Präsenz - Projektabnahmen**) BEFORE the end of submission!!!
- Give the ZIP file a name following this schema: webframeworks-groupxx-ifxxbxxx-ifxxbxxx.zip (e.g. webframeworks-group01-if20b001-if20b002.zip)
- Make sure that everything (except the **node_modules** folder) is within the ZIP file.

Grading

- This project will be graded with 40pts.
  - Port, design & implementation of core functionality (puzzle game logic) -> 12 pts
  - Login (Authentication token, logout, …), signup -> 4 pts
  - Profile page, site navigation -> 8 pts
  - High Score page + InMemory DB attachment -> 8 pts
  - good UI styling (usage of material libraries, useful design & theming [colors]) -> 4pts
  - Logo, “realistic” FAQ -> 4pts
  - optional bonus points (see bonus slide) -> 5 pts

Bonus

- To get 5 bonus points you can implement the following bonus task. Please note that this exercise grading can not exceed the maximum of 40pts, but by implementing this bonus task, you can make up for already lost points.
- Create a second puzzle (or more) and before starting the puzzle game, let the user choose between your created puzzles. Make sure to create a puzzle overview component for this.