1. Crossword Puzzle Web Application Development Requirements

Overview

Develop a sophisticated web-based application designed for creating, viewing, and exporting crossword puzzles. The application will be developed using ReactJS, TypeScript, and Bootstrap, with jsPDF for PDF generation functionality. This application aims to provide an intuitive and efficient user experience for crossword enthusiasts and creators.

Technical Specifications

Frontend Framework: ReactJS with TypeScript.
Styling Framework: Bootstrap for UI components.
PDF Generation: jsPDF.
Supported File Format: CSV for word and clue input.

Functional Requirements

User Input Interface:

- A text area for manual input and a file upload option for CSV files.
- The CSV file should follow the "word,clue" pair format.

Crossword Puzzle Generation:

- An algorithm to automatically place words into a crossword grid.
- Grid expansion and dynamic adjustment based on word length and placement.
- Generate as many pages as the amount of words require.

Puzzle Preview and Navigation:

- Carousel-based preview for viewing each generated crossword puzzle.
- Each puzzle slide will display the crossword grid.

PDF Export Functionality:

- Capability to export the crossword puzzles to a PDF file.
- Each puzzle's clues listed below the corresponding grid in the PDF.
- A dedicated section at the end of the PDF for all crossword answers.

Platform Compatibility:

- The application is intended for desktop use with no immediate requirement for mobile responsiveness.

Performance and Optimization

- Ensure efficient load times and responsiveness of the crossword generation and PDF export functionalities.
- Implement asynchronous processing where applicable to enhance user experience.

Additional Features

- Robust error handling, especially for CSV parsing and grid generation logic.

- Clear user instructions for input formats and operational guidance.
- Modular and well-documented code structure for maintainability and scalability.

Example images:

CROSSWORD **PUZZLE 4**

ACROSS

1. Spray art
5. Climb
9. Kind of year
10. Fondle
12. Balances
13. Financial House
14. Alarm buttons
16. Extra
19. Open
21. Tailor's machine
23. Back side
25. A deadly sin
26. Gold's friend
27. Type of anthem
28. Commuter, at times
29. Practice

DOWN

1. Col. Sanders feature
2. Knowledge of
3. Zookeeper to the animals
4. Youngster's PM home
6. Friend of Dorothy
7. Swelling
8. Spread out
11. Big cheese
15. Stick to it
17. Lord of the manor
18. Alleged
20. Cry out
21. Superficial
22. Put on hold
24. Corporate department
25. Capture

2. Word Search Puzzle Web Application Development Requirements

Project Setup

Creating a word search app generator using ReactJS, TypeScript, and CSS and jsPDF for PDF generation.

Designing the UI

- Create a form for users to input words, select puzzle size. The input words must generate as many pages as the amount of words require.
- Add checkboxes for direction selection (horizontal, vertical, diagonal, etc.).

Generating the Puzzle Grid

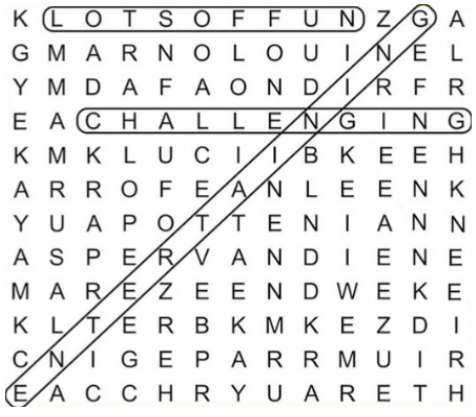
- Dynamically create a grid based on the user-selected size.
- Place the words in the grid using the algorithm.
- Fill the empty spaces with random letters.

Word Placement Algorithm

- Implement an algorithm to place words in the grid in various directions.

1. Horizontal Forward
2. Vertical Downward
3. Diagonal Downward
4. Diagonal Upward
5. Vertical Upward
6. Horizontal Backward
7. Diagonal Reverse

- Ensure the algorithm respects the selected directions and puzzle size.
- A word can overlay another
- There must be a toggle button to show and hide the answers in the the grid and when the answers are shown, they must be enclose like in the picture below.



PDF Generation

- Use jsPDF to convert the generated puzzles into downloadable PDF format.
- Ensure the PDF layout matches the puzzle size selected by the user.
- All the answers must be shown at the end of all pages.

Final result per page must be like the picture below:

3. Sudoku Puzzle Web Application Development Requirements.

Subject: Development Plan for Web-Based Sudoku Application.

We are initiating a project to develop a dynamic, web-based Sudoku puzzle application. This application will be built using ReactJS, TypeScript, CSS, and jsPDF for PDF generation. Our goal is to create a user-friendly, interactive Sudoku generator that caters to various skill levels, with a unique feature of generating downloadable puzzles in PDF format. Below is a detailed outline of the project requirements:

Project Setup

- Framework and Technologies: The application will be developed using ReactJS and TypeScript, utilizing CSS for styling and jsPDF for the generation of downloadable PDFs.
- Sudoku Puzzle Generator: A central aspect of the application, responsible for creating puzzles with adjustable difficulty levels.

Sudoku Puzzle Generator

- Algorithm Implementation: Develop a versatile algorithm capable of generating Sudoku puzzles. The algorithm should allow for the modification of difficulty levels by varying the number of starting clues.

Designing the UI

- User Interaction: Implement a user interface where players can generate puzzles at the click of a button.
- Puzzle Quantity Selection: Users should be able to select the number of puzzles they wish to generate, with a maximum limit of 100 pages per request.

Controlling Difficulty Levels

- Number of Clues (Given Numbers):
    - Easier Levels: Provide more numbers (e.g., 40-45 numbers) to simplify the puzzle-solving process.
    - Harder Levels: Offer fewer numbers (e.g., 17-25 numbers) to increase the challenge.
- Placement of Clues:
    - Easier Levels: Clues should be strategically placed to guide players logically toward the solution.
    - Harder Levels: Clues should be more scattered, requiring players to employ advanced solving techniques.

PDF Generation

- Functionality: Integrate jsPDF to enable the conversion of generated puzzles into a PDF format.
- Layout Consistency: Ensure the PDF layout is coherent with the puzzle size selected by the user, maintaining readability and quality.