Technical Test – Full Stack Developer: WordPress Plugin Development

Objective: Develop a WordPress plugin that allows users to display offers from an external API. The plugin will create a Gutenberg block, named "Offers Block", which users can add to their posts or pages. The block will have a field for entering the API URL, and once set, it will display the offers on the frontend.

Requirements:

1. Plugin Setup:

- The plugin should be a standard WordPress plugin with proper file structure.
- It should follow WordPress coding standards and best practices.

2. Object-Oriented Programming:

- The codebase should be structured using Object-Oriented Programming (OOP) principles.
- Make use of classes and methods appropriately to organize the functionality.

3. Gutenberg Block Creation:

- Create a custom Gutenberg block using modern JavaScript (ES6+).
- The block should have a user-editable field for inputting the API URL.
- Implement block validation to ensure the URL is valid.

4. Advanced Custom Fields (ACF) Integration (Optional):

• If comfortable, use ACF to create the field for the API URL. This is optional and can be done without ACF as well.

5. SCSS Skills:

- Style the Gutenberg block and the frontend display using SCSS.
- Demonstrate the use of SCSS features like variables, nesting, and mixins.

6. JavaScript Skills:

- Use JavaScript for dynamic elements of the block.
- Show proficiency in modern JavaScript features

7. API Consumption:

- Identify the required data (based on design below) and fetch it from the provided API: https://api.jsonbin.io/v3/b/65b7a4281f5677401f27c75b
- For images such as logo, deposit icons please use the <u>dark</u> version.
- Handle API responses and errors gracefully.
- Parse and display the offers data on the frontend (based on designs below).

8. Plugin Functionality:

- Once the API URL is added in the Gutenberg block, the plugin should fetch and display the offers.
- Ensure the display is visually appealing and user-friendly.
- Implement a carousel for deposit methods (or use a 3rd party script if preferred)

9. **Documentation:**

- 1. Provide clear documentation on how to install and use the plugin.
- 2. Include inline comments in the code for clarity.

10. Bonus (Optiona – Backend Developmentl):

- Implement caching for API responses to optimize performance.
- Implement a local caching mechanism for images

Submission Guidelines:

- Submit the complete plugin code, including all source files (zip file or you can use Git).
- Provide a README file with installation and usage instructions.
- Include any additional notes or explanations regarding the implementation.
- The designs don't need to be pixel perfect

Design that needs to be achieved on Desktop & Mobile:



