Full Stack Developer Assignment

This document contains descriptions and requirements of assignment for full stack web developer - JavaScript stack.

Goal

The goal of this assignment is to verify that the applicant is able to:

- Make a new structured frontend project
- Make responsive views
- Work with Web-socket
- Make a structured backend project
- Create well-structured restful APIs
- Design database and data models
- Use implementation and deployment tools
- Write clean code
- Use git and source controls
- Write short and clean documentation

Requirements

This section describes a sample task which applicant would be handling as part of our team. Clean code and the structured solution is the main factor to consider.

- Create a git repository (Gitlab) and add hamidreza.momeni@bitazza.com as Maintainer.
- A well-written readme.md is a must with the instruction on how to run your application.
- For web frontend, and admin console implementation you should use React js.
- To store data, you should use MySQL.
- For backend development, you should use Node Js as the programming language.
- All the communications between frontend and database should be done through restful APIs/ WSS which must be designed, build using Nodejs, Express, and Postman for documenting the APIs.

Assignment

This assignment consists of a web single page Exchange application, a simple admin console, and restful APIs for communication with the database.

1. Console: Create a responsive single page web with login functionality.
    a. Use JWT for authentication implementation and REST APIs
    b. Admin should be able to log-in and logout.
    c. Product list section: Logged-in admin can see list of products with basic details (show the details of your choice), please use the flowing wss API (https://api-doc.bitazza.com/#getproducts).
    d. Instruments list section: Logged-in admin can see list of Instruments with basic details (show the details of your choice), please use the flowing wss API
    (https://api-doc.bitazza.com/#getinstruments)
    e. Exchange available instruments section: Logged-in admin can select one or many instruments to be shown on the exchange. Admin can add or remove available instruments to this list.
    f. Rest APIs to be implemented to update the selected instrument list in DB.

2. Create a responsive single page admin console for the exchange web app.
    a. The Exchange app should be a single page public page. No authentication required.
    b. Users should be able to see the live market data change for the instruments selected by admin.
        - The instrument name can be obtained from the link below
        (https://api-doc.bitazza.com/#getinstruments)
        - Instrument live market data can be obtained from
        (https://api-doc.bitazza.com/#subscribelevel1)
        Please only show: BestBid, BestOffer, Volume, LasTradePX, LastTradedQty and TimeStamp (converted to BKK time)
        - Note that these values should get updated in real-time as the subscribelevel1 will send the response continuously.
    c. If the changed value is increased please show the value in Green, and if it decreased please show in Red.
    d. Please design the UI for this page by your choice. However, being mobile-friendly is mandatory.

3. Node Js API project & MySQL DB.
    a. Design the required MYSQL database
    b. Create necessary APIs for admin console and Exchange app using Nodejs, Express
    c. use Postman for API documentation.

API Endpoints
Websocket URL: wss://apexapi.bitazza.com/WSGateway/
API Documentation: https://api-doc.bitazza.com/

Bonus

The tasks in this section are not mandatory but providing them as your solution would an excellent plus point.

1. Use Docker Compose for the deployment of web frontend, admin console, APIs, and your database.
2. Deploy your solution on a free cloud web hosting platform of your choice.