|  |  |  |
| --- | --- | --- |
| Highlow Assessment | | |
|  |  |  |
|  | | |

# Functional Features

|  |  |
| --- | --- |
|  | 1. Real time price updates (chart and cards) for selected assets    1. Real-time chart update    2. Real-time asset card update    3. Coloring indication (Green / Red 2. Trade Assets view    1. Carousel display for asset cards    2. Trade Assets Filtering (filter by value or currency - including search input with autocomplete feature, or by bookmarked assets) 3. Bookmark Assets (add to favorites for watching) 4. Live Trading 5. User management    1. Login    2. Logout    3. Forgot password    4. Account management |
|  | 1. Notification on selected asset |

# Visual UI

|  |  |  |
| --- | --- | --- |
|  | 1. Interactive chart that shows real time changes to prices and interact with trading actions 2. Trading form (High / Low Action on the chart) 3. Filter controls (autocomplete, search input, filter by bookmarked) 4. Notification toaster 5. Timer countdown control 6. Transitions / contextual component animation 7. Tab controls 8. Carousel 9. Card UI | |
|  | 1. Custom dropdown menus | |
|  | 1. Sliding panels 2. Loading indicator | |
|  | 1. Logo, color theme, icons, and typography | |
|  |  |

# UI Components

Suggested UI components structure check the folder structure in the ReactJS app and wireframe image (Hanasoft - highlow UI components structure.drawio.svg) included in [**Hanasoft**](https://github.com/MostafaAbdElkareem/Hanasoft) github repo under Docs folder

# Quality-assurance

|  |  |
| --- | --- |
|  | 1. Stick to approved design and collaborate with UX/UI designer to understand the visual elements in the design and user interactions, UX designer should hand over UI guidelines or prototype to Devs and QA walk through different screen   **For developers**   * is to run unit testing on each component, preferable to automate tests to reduce the testing time and ensure covering all the cases * Consider automate/manually testing components against different browsers and devices if required * Consider applying accessibility guidelines while development * to test the UI and functionality after merging new features or updating existing ones |
|  |  |
|  | 1. **For QA**  * To make sure design is implemented exactly as in the approved screens or prototypes (pixel perfect concept) by running visual regression tests * To run complex test cases scenarios and make sure the developed features meet the requirement * Utilising automation tools to run test cases * Utlise tools for testing cross browser compatibility and targeted accessibility guidelines level (if required) |
|  | 1. Per each sprint make sure all tasks are done and meet the requirements and report if any issues and wrong actions was taken by running acceptance test |
|  | 1. Run performance test and report result to Devs to take actions to optimize it 2. UX designer to get involved periodically to review developed features and give feedback on its look and feel |

# Data modeling

|  |  |
| --- | --- |
|  | Check attached diagram in the Docs folder |

# Apis

# Development life cycle

Software Development lifecycle steps are defined as a framework with 7 steps that starts from

* **Planning** (all aspects related to the product going live from cost, timeline and scheduled releases, resources, and licenses)
* **Requirements** (collaboration between different stakeholders from business, SMEs, project managers, QAs and real end-users to gather and shape the features of the product
* **UX design and Prototyping** (transferring ideas and requirement into wireframes then visual design throughout different stages or applying processes like (user centered design or design thinking) to get approval on one design that match users needs
* **Development** (backend/front-end) implementation of accepted design through selecting suitable platforms and libraries and infrastructure
* **Testing** : ensure that accepted requirements functionality, design are developed as expected and also change request that might have during the development phase was applied and meeting business and user needs
* **Deployment** run final release into real world through different levels if needed like staging then production through automated / manual deployment steps (can utilize automated CICD tools like Jenkins, Travis and TeamCity
* **Operation and support**