Specifications: Implementation of a Notification, Sharing and Favorites System for a Symfony 6.4 Site

1. Context and Objectives

This specification aims to define the technical specifications for implementing a notification, page sharing and bookmarking system on a site already developed with the Symfony 6.4 framework , using Bootstrap and JavaScript. This system must meet the following needs: - Allow non-registered users to receive notifications, share pages and add items to bookmarks (for logged in users). - Offer registered users a personalized notification and bookmark management interface.

2. Requested Features

2.1. For Users without an Account

A°) Alert/Notification

Integrate a popup system on all pages of the site to allow users to register via their email address to receive notifications regarding specific sections (value changes, news, company changes, etc.).

Specifications:- Registration popup: - Detection of non-logged in users to display the popup. - Input field for the email address. - Drop-down list or checkboxes to select the sections for which to receive notifications. - Validation button to register.- Unsubscribe management: - Each notification sent must include an unsubscribe link specific to the relevant section. - Simple interface allowing selective unsubscription by section.- GDPR compliance:
- Compliance with personal data protection standards (validation of registration by double opt-in, possibility of deleting data on request, etc.).

B°) Sharing

Integrate a popup system on all pages allowing users to share the page in question via email, social networks (Facebook, X, LinkedIn, WhatsApp, etc.), and by phone.

Specifications: - Share popup: - Share buttons for each platform (email, Facebook, X, LinkedIn, WhatsApp, phone). - Simple and intuitive interface allowing to select the sharing

method and confirm the action. - Ability to customize the message sent when sharing via email or social networks.

C°) Favorites

Integrate a popup system on all pages allowing users to log in and add a page, product or company to their favorites. This action is only possible when the user is logged in and has a personal account.

Specifications: - Favorites popup: - Detection of user login status. - Invitation to log in to add to favorites if the user is not logged in. - Simple interface to add the item to favorites once logged in.

2.2. For Users with an Account

Creation of a notification/alert and favorites management interface, integrated into the back office. Allows the creation, modification, and deactivation of notifications for specific values that the user wishes to track.

Specifications: - Management interface: - Accessible from the user dashboard in the back office. - List of available sections and elements for which the user can enable or disable notifications. - Favorites management: adding, deleting, and organizing favorite items.

3. Technical Architecture

3.1. Managing Notifications, Sharing and Favorites

Database:
- Creation of tables dedicated to managing notifications/alerts, shares and favorites, including the necessary fields (user id, section, notification type, frequency, favorited item id, etc.).

Symfony Commands: - Creation of Symfony commands to trigger notification sending and favorites management based on defined time intervals (minute, hour, day, month, year). - Use of Cron services or a scheduled task for automatic execution of commands.

Controller and Templates :
- Creation of a dedicated Controller for managing notifications, sharing and favorites in the back office. - Development of HTML Templates for the user interface.

4. Expected Deliverables

Source Code:
- PHP files (Controller, Commands, etc.) needed to manage notifications, sharing and favorites. - Twig files for the user interface.

- SQL scripts to create the necessary tables in the database.

5. Constraints and Requirements

Compatibility:
- The system must be fully compatible with Symfony 6.4, Bootstrap, and JavaScript.

Security: - All features must comply with security best practices, especially for user data management.

Testing: - The service provider must provide unit and functional tests to ensure the proper functioning of the system.

6. Monitoring and Validation Procedures

Validation phases:
- Validation of user interface mockups before development. - Intermediate tests to validate the various features (registration, sharing, favorites management, sending emails, etc.). - Final validation after integration into the production environment.

Support and Maintenance: - The service provider must guarantee a post-delivery support period to correct any bugs and ensure system maintenance.