Combined Technical Document: Notification System, Help Center, and Enhancements to Queries Feature in MVC Project

1. Notification System

Methodology

The email notification system is event-driven, sending emails based on specific query actions:

1. **Query Creation:**
   - If the query is unresolved (ResolvedBy and DateResolved are empty), an email is sent to notify the recipient that the query is submitted but still unresolved.
   - If the query is resolved (ResolvedBy and DateResolved are provided), an email is sent confirming the query's resolution.
2. **Query Update:**
   - When an unresolved query is updated with ResolvedBy and DateResolved, an email is triggered to notify the user that the query has been resolved.

Implementation Steps

1. **Configure Email Settings in appsettings.json:**
   - Define SMTP settings: server address, port, sender email, and credentials.
2. **Create Email Service Class (EmailService.cs):**
   - Implement an asynchronous method (SendEmailAsync) to handle email sending.
3. **Integrate Email Notifications in Query Creation:**
   - In the Create action of QueriesController, check if the query is unresolved or resolved and send the appropriate email.
4. **Integrate Email Notifications in Query Update:**

- In the Edit action of QueriesController, send an email if the query's resolution status changes.

5. **Dependency Injection for Email Service:**
   - Register the EmailService in Program.cs for dependency injection.

---

2. Help Center

The Help Center offers users access to resources like the User Manual and FAQs for better support and guidance.

Features

1. **Navigation Dropdown:**
   - Accessible from the navigation bar with links to the User Manual and FAQs.
2. **User Manual:**
   - PDF user guide rendered using PDF.js with navigation controls (Previous and Next).
3. **FAQ:**
   - Accordion-style list of common questions and answers with a search function for easy navigation.

Implementation Details

1. **Navigation Menu:**
   - Dropdown in the navbar includes options for **User Manual** and **FAQ**.
2. **User Manual Page:**
   - PDF rendered using pdf.js, dynamically loaded from /manual/UserManual.pdf.
3. **FAQ Page:**
   - FAQs are displayed in a collapsible accordion format, with a search feature for filtering questions.

---

3. Enhancements to Queries Feature

Recent enhancements improve the user experience by adding search functionality and file previewing.

Overview

Changes include:

- Search functionality to filter logged queries.
- File preview option in the UI.
- Modifications in QueriesController and Razor views (AllPorts.cshtml and others).

Modified Files

1. **Controllers:**
   - QueriesController.cs
2. **Views:**
   - AllPorts.cshtml
   - Three other port-related views

Changes in QueriesController.cs

- **Updated AllPorts Action Method:**
  - Implemented search functionality to filter queries based on keywords.
  - Improved query efficiency by materializing data before applying filters.
  - Integrated pagination to display queries in pages.

Code Example:

```
public async Task<IActionResult> AllPorts(int? page, string searchTerm)
{
  ViewData["SearchTerm"] = searchTerm;
  var query = _context.Queries
              .Include(q => q.Clients)
              .Include(q => q.Departments)
              .Include(q => q.Port)
              .Where(q => q.Date.Year == DateTime.Now.Year);
```

```
    var materializedQuery = await query.ToListAsync();


    if (!string.IsNullOrEmpty(searchTerm))
    {
        materializedQuery = materializedQuery
            .Where(q => q.Description.Contains(searchTerm,
StringComparison.OrdinalIgnoreCase) ||
                        q.QueryNumber.ToString().Contains(searchTerm,
StringComparison.OrdinalIgnoreCase) ||
                        q.Departments?.Department.Contains(searchTerm,
StringComparison.OrdinalIgnoreCase) == true ||
                        q.Clients?.ClientName.Contains(searchTerm,
StringComparison.OrdinalIgnoreCase) == true ||
                        q.Port?.FPTPort.Contains(searchTerm,
StringComparison.OrdinalIgnoreCase) == true)
            .ToList();
    }


    ViewBag.QueryYear = materializedQuery.Count;
    ViewBag.Created = TempData["Create"] as string;
    ViewBag.Updated = TempData["Update"] as string;


    var pageSize = 6;
    var pageNumber = page ?? 1;


    var pagedQueries = materializedQuery
        .OrderByDescending(q => q.CreatedOn)
        .ToPagedList(pageNumber, pageSize);
    var viewModel = new QueryViewModel
    {
        Queries = pagedQueries,
        AllQueries = materializedQuery.AsQueryable()
    };

    return View(viewModel);
}
```

Changes in AllPorts.cshtml

1. **Search Functionality:**
    - Integrated a search bar to filter queries.
2. <form method="get" asp-action="AllPorts" asp-controller="Queries">
3.   <div class="input-group mb-3">
4.   <input type="text" name="searchTerm" class="form-control" placeholder="Search queries..." value="@ViewData["SearchTerm"]">
5.   <button class="btn btn-outline-primary" type="submit">
6.   <i class="fas fa-search"></i> Search
7.   </button>
8.   </div>
9. </form>
10. **File Viewing Modal:**
    - Added a modal to preview files without downloading.
11. <div id="fileViewer">
12.   <button onclick="closeViewer()" class="btn btn-danger" style="position: absolute; top: 5px; right: 5px;">Close</button>
13.   <iframe id="fileIframe" frameborder="0"></iframe>
14. </div>
15. **View File Button:**
    - Added a button for viewing files directly.

16. <a href="javascript:void(0);" class="btn btn-outline-success" title="View File" onclick="openFile('@Url.Action("ViewFile", "Queries", new { fileName = item.FileName })')">
17.   <i class="fas fa-eye"></i>
18. </a>
19. **JavaScript for File Preview:**
20. function openFile(url) {
21.   document.getElementById("fileViewer").style.display = "block";
22.   document.getElementById("fileIframe").src = url;
23. }
24.
25. function closeViewer() {
26.   document.getElementById("fileViewer").style.display = "none";
27.   document.getElementById("fileIframe").src = '';
28. }

Changes in Other Port Views

The same enhancements were applied to three additional views:

- Search functionality.
- File preview option.

Conclusion

These updates significantly improve the user experience:

1. Search functionality makes query filtering easier and faster.
2. File previewing allows users to view documents directly within the browser.
3. The changes ensure a consistent, user-friendly experience across port-related views.

This update enhances efficiency and usability for users managing queries in the system.
