

Repair System Web Application

Client: Anonymous

Business Size: Corporation

Industry: Retail

Country: Worldwide

Technology: Bootstrap, HTML, CSS, Javascript, JQuery

Objective: To repair and restore the system web application to full operational functionality by identifying existing faults, resolving technical issues, and ensuring stable, secure, and efficient performance that meets user and business requirements.

The Brief

This luxury retail brand has stores in many parts of the world, offering one of the most premium shopping experiences. Captivate is a web project to manage some operational processes between those stores.

Challenges

During my role on this project, I worked on the Repair System web application. The project was using Telerik UI for ASP.NET Core (Kendo UI) on the UI layer. However, due to the Kendo version being significantly outdated, making it difficult to maintain, causing bugs in modern browsers, being impossible to upgrade, and incurring an annual licensing cost. It was decided to completely remove Kendo UI from the project.

In addition, the jQuery version bundled with Kendo was also very outdated and could not be upgraded, which led to various issues across the application. For these reasons, eliminating the Kendo dependency entirely became a critical technical challenge.

I worked on this project for a total of three months.

During the first month, I migrated approximately 30 tables that were using Kendo Grid to Bootstrap Table.

I was then asked to remove all Kendo UI elements, references, files, and backend objects from the project. Within this scope, a two-month delivery timeline was defined, and I began the implementation.

As a first step, I carried out a comprehensive analysis to identify all areas where Kendo was used throughout the application. This analysis revealed nearly 700 Kendo UI components on the UI side, including:

Textbox, Dropdown, Window, Dialog, Checkbox, ListView, Button, ButtonGroup, TabStrip, TextArea, Switch, TimePicker, RadioButton, RadioGroup, NumericTextBox, MaskedTextBox, ListBox, ImageEditor, Editor, DropDownList, DateTimePicker, DatePicker, and Grid components.

In addition, the JavaScript logic layer relied heavily on numerous Kendo-specific methods that provided easy access to UI components. As a result, almost every page in the application required modification, making this one of the most challenging aspects of the project.

On the backend side, approximately 15 endpoints were using Kendo's proprietary filtering and request structures. These endpoints needed to be refactored so that Kendo specific objects were replaced with alternative implementations without breaking existing functionality.

Since Bootstrap could not fully replicate some of Kendo's advanced features such as editor, image editor, dropdowns, form validation, date pickers, dialogs, and popups. I integrated alternative JavaScript libraries, including:

- Tempus Dominus – Date pickers
- Select2 – Dropdowns
- BootboxJS – Dialogs & Popups
- jQuery Validation – Form validation
- DimBoxJS – Image editor
- Jodit Editor – Rich text editor

I prepared technical documentation for my own use that outlined all Kendo usage areas and detailed a clear plan for how each Kendo component would be removed and replaced. Based on this plan, I proceeded to remove Kendo incrementally, component by component, testing the application after each change.

Once all UI components and their associated JavaScript logic were replaced, I completed the following steps:

- Removed all Kendo-related UI JavaScript and CSS files from the project
- Upgraded jQuery to the latest stable version
- Refactored backend endpoints that were dependent on Kendo objects
- Completely removed all remaining Kendo references from the backend

Throughout this process, I also identified and fixed numerous existing bugs in the application. After completing all changes, I thoroughly tested the application multiple times and resolved all remaining issues, ensuring the system was stable and production ready.

Solution and implementation

Frontend:

- HTML
- CSS
- jQuery
- Bootstrap
- Kendo UI (for analysis and removal)
- Tempus Dominus – Date pickers
- Select2 – Dropdowns
- BootboxJS – Dialogs & Popups
- jQuery Validation – Form validation
- DimBoxJS – Image editor
- Jodit Editor – Rich text editor

Backend / CI/CD:

- .NET 8
- C#
- Git

- SourceTree

I independently tracked all issues in the application, identified and resolved them, and tested each fix. Visual Studio 2022 and MS Teams were used for development and communication. Finally, the QA environment was used to verify and compare all changes made in the application.

Results

My development work in the Repair System web application:

- Replaced all Kendo UI elements with Bootstrap and alternative libraries, ensuring consistent UI across the application
- Refactored backend endpoints to remove Kendo-specific objects without breaking existing functionality
- Fixed numerous existing bugs and resolved issues caused by outdated jQuery and Kendo dependencies
- Upgraded jQuery to the latest stable version and removed all Kendo-related files
- Tested the application thoroughly, including multiple iterations and different user scenarios, ensuring full stability and functionality

All identified technical challenges and bugs were addressed promptly and effectively, resulting in a fully modernized and stable application.

My contribution

For the Repair System web application, I started by analyzing the current implementation to identify all areas dependent on Kendo UI and the outdated jQuery. This assessment revealed nearly 700 UI components and several backend endpoints that relied on Kendo-specific structures.

I created a dedicated branch to perform migration. First, I replaced approximately 30 Kendo Grid tables with Bootstrap Table equivalents. Then, I systematically removed all Kendo UI elements, including Textboxes, Dropdowns, Windows, Dialogs, Checkboxes, Lists, Buttons, Editors, and other components. For each component, I implemented appropriate replacements using alternative libraries: Tempus Dominus for

date pickers, Select2 for dropdowns, BootboxJS for dialogs and popups, jQuery Validation for form validation, DimBoxJS for image editing, and Jodit Editor for rich text editing.

At the same time, I refactored backend endpoints to remove Kendo-specific logic without affecting existing functionality. I also upgraded jQuery to the latest stable version and removed all Kendo-related scripts and CSS files from the project.

After each change, I performed iterative testing. Every page was thoroughly checked for functionality and UI consistency. This careful approach allowed me to fix existing bugs and address any issues introduced during the migration.

By the end of the project, the application was fully modernized, free of Kendo dependencies, and worked smoothly across all pages. The work required a detailed and methodical approach to ensure the system was stable, maintainable, and fully functional.

Skills and expertise

- My skills and expertise include:
- HTML
- CSS
- Javascript
- JQuery
- (Kendo) Telerik UI for ASP.Net Core
- Bootstrap 5
- Tempus Dominus for Datepickers
- Select2 for dropdowns
- BootboxJS for dialogs and popups
- JQuery Validation for form validation
- DimBoxJS for image editor
- Jodit Editor
- .Net 8
- C#
- Git

- Collaboration and communication

Lessons learned

During this project, I reinforced the importance of thorough analysis and planning before making changes in a complex web application. I learned that a systematic approach to front-end updates is essential to maintain both functionality and UI consistency across the application.

I had the opportunity to collaborate effectively while also learning Kendo UI. Additionally, it was very valuable to learn and apply Bootstrap and other JavaScript libraries, including Tempus Dominus for date pickers, Select2 for dropdowns, BootboxJS for dialogs and popups, jQuery Validation for form validation, DimBoxJS for image editing, and Jodit Editor for rich text editing.

I also had the chance to refresh my knowledge of HTML, CSS, and jQuery, which highlighted the importance of keeping foundational skills up to date. Overall, I learned that careful testing, iterative improvements, and clear documentation are key to delivering high-quality and stable applications, and I genuinely enjoyed working on this project.