Daniel Nkurunziza

Citipost Technical Development Task

Summary

A user-friendly application that can be used by warehouse personnel built in Visual Studio using C# Windows Forms App (Template). The application imports data from excel files, assigns suppliers automatically, generates carrier labels and generates summary reports outlining key information.

Purpose of files

- Data Folder: to store the excel and docx files used in the application
- Model Folder: manages the getters, setters and structure for suppliers, subscribers, labels and summary report
- Reports Folder: to store the generated labels and assigned excel file
- Services Folder: manages the logic for data import, processing, storage, label generation and correct formatting

Challenges and Solutions

- Faced errors with *InitialiseComponent* in both MainForm and ReportFrom forms. The method was being referenced in the designer subclass of both forms, initially the method was deleted which led to build fail. Instead, the method name was modified in both the subclasses of the two forms (MainForm.Designer.cs and ReportForm.Designer.cs).
- The application threw EPPlus licensing errors when it was run which halted the excel files processing. GitHub copilot had to be used to analyse the issue in real time and online forums that showed the application needed appropriate license configuration (NonCommercial use).
- Worksheet position out of range error occurred when attempting to access the excel worksheet. I added error handling and included checks before accessing the worksheet as well as integrity validation to prevent index out of range exceptions.
- File not found errors were occurring during runtime. The file paths had to be verified and file existence checks, a method to return to the root directory was implemented to find the files in the *Data* folder rather than *bin/Debug* directory. Error messages were added so users are guided when files are missing in the directory.
- Carrier labels were initially printed as one label per page instead of following the INTL template. Graphics had to be modified and the label generation logic to follow the INTL template as well as a helper method to wrap the text as long texts would go through the label margins initially.

Daniel Nkurunziza

System Requirements Needed

- .NET Framework
- Visual Studio
- EPPlus Library
- System.Drawing.Printing for graphics control
- Excel and word documents provided

Future Improvements

- Implementing a web-based solution for better accessibility remotely
- Using RDLC for different export formats like pdf or word
- Implementing an error logging system
- User authentication and roles for access control
- Integrating a SQL server database to store data

Assumptions Made

- The Pageant Routing Guide remains static
- Excel formatting and structure remains consistent
- The application is designed for single user desktop deployments
- More information can be added on the label and can be customized

Limitations

- Layout needs to be manually changed using graphics and layout changes can only be seen after running the application
- Application supports only the INTL template format
- Depends on excel file formats specifically
- Manual file management is needed to run application successfully