**Refactoring Approach**

Here's a revised and improved version of the provided code. The aim is to make it easy to read, understand, and extend with more mail types in the future.

**Identified Shortcomings:**

* **Code Duplication**

Similar code for sending emails is duplicated in email sending methods.

* **Difficult to Read**

Due to use of excessive comments and poor naming conventions it is difficult to read.

* **Hardcoded values**

Use of hard code strings like (host and from email), making it less flexible.

* **Poor Separation of Concerns**

Email methods are performing multiple things like doing business logic to select customers and sending emails.

* **Unconfigurable Email Contents**

Strings like email subjects and bodies are difficult to configure.

* **Poor Error Handling**

Error handling is basic and does not provide useful information for debugging.

* **Difficult to extend**

It is difficult to add more email types in the existing code because it contains multiple kind of operations.

**Refactoring Plan:**

* **Extract Email Sending Logic**

Move the email sending logic from methods (EmailWork and DoEmailWork2) to a separate method.

* **Separate Filtering and Content Creation**

Separate customer filtering logics and email content creating logic for better separation of concerns, flexibility and readability.

* **Separate Template Creation**

Separate content creation logic from the same file to separate compile able (razor page) files for handling complex email templates.

* **Use of Design Pattern**

Use design patterns to add more flexibility for adding more email types and for configurable email content.

* **Use Configuration**

Replace hardcoded values with configurable options.

* **Improve Error Handling**

Enhance error handling to provide more useful information.

* **Make Code Asynchronous**

Use asynchronous operations for sending emails.

* **Use Dependency Injection**

For better testability and flexibility, use dependency injection for dependencies like “SmtpEmailService”.

* **Enhance Folder Structure**

Maintain folder structure to keep logical things together for example (email logic related classes in EmailSenders folder).

* **Easy to Read**

Proper use of naming conventions and less use of comments to make code easy to read and understand.