**Complete the following test. You may use your textbook, labs and lecture notes.**

**All code submitted must be your original work. Code must be committed and pushed to your git repository with the solution name HOT2 to receive credit.**

**Please take absolutely nothing for granted. Ask questions if/as necessary.**

**HOT2**

For this assignment you will be building a Sales Order database application.

The application should have the following features:

* Show current products, ordered by product name (as shown).
* New products can be added.
* Existing products can be updated.
* Existing products can be deleted.

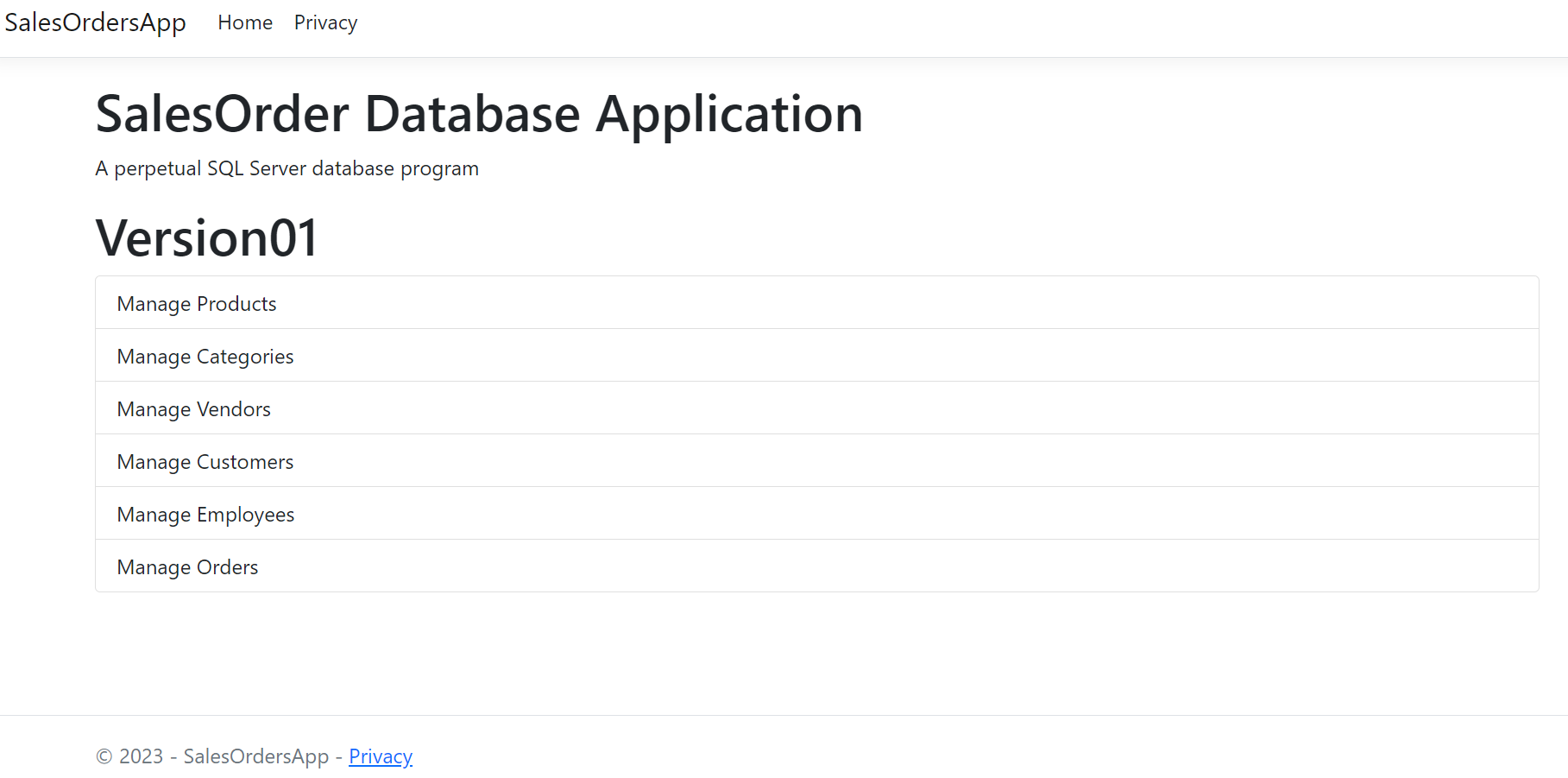
For this test, the database should have two tables:

* Products (Convert/add just the first five products from the attached 05-SOProducts.SQL file).
  + ProductID: int
  + ProductName: string
  + ProductDescShort: string (will be empty, or hold just “” for now)
  + ProductDescLong: string (will be empty, or hold just “” for now)
  + ProductImage: string (will be empty, or hold just “” for now)
  + ProductPrice: decimal (should default to 0.00)
  + ProductQty int (should default to 0)
* Categories (Convert/add all 6 categories from the attached 01-SOCategories.SQL file).
  + CategoryID : int
  + CategoryName : string

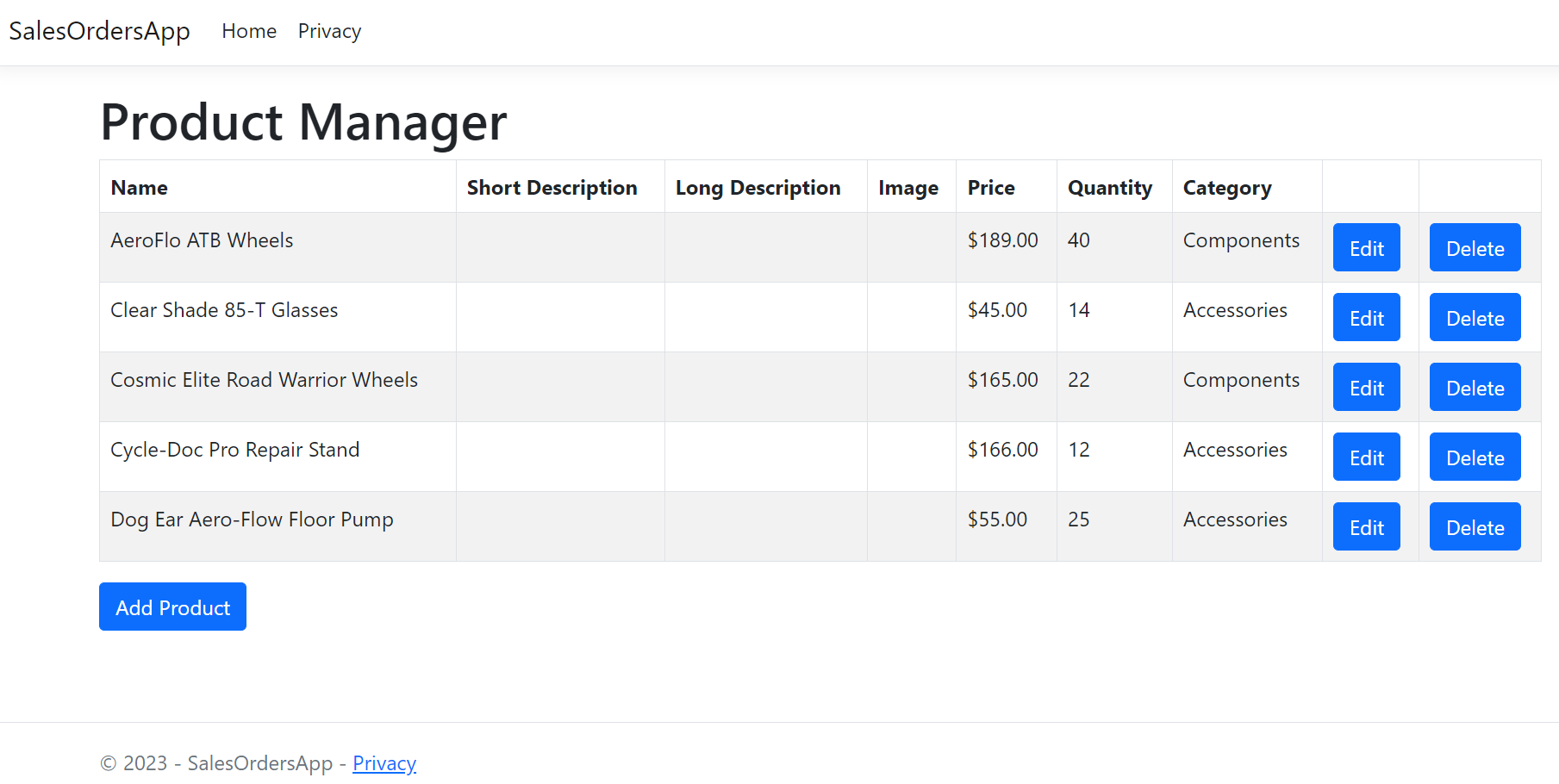
**Screen mockups are shown on the next 5 pages.**

**See the scoring rubric on the last page.**

**Please ask if you have any questions. Good luck.**



**SCREEN 1 – OPENING SCREEN WHEN RUNNING PROGRAM**



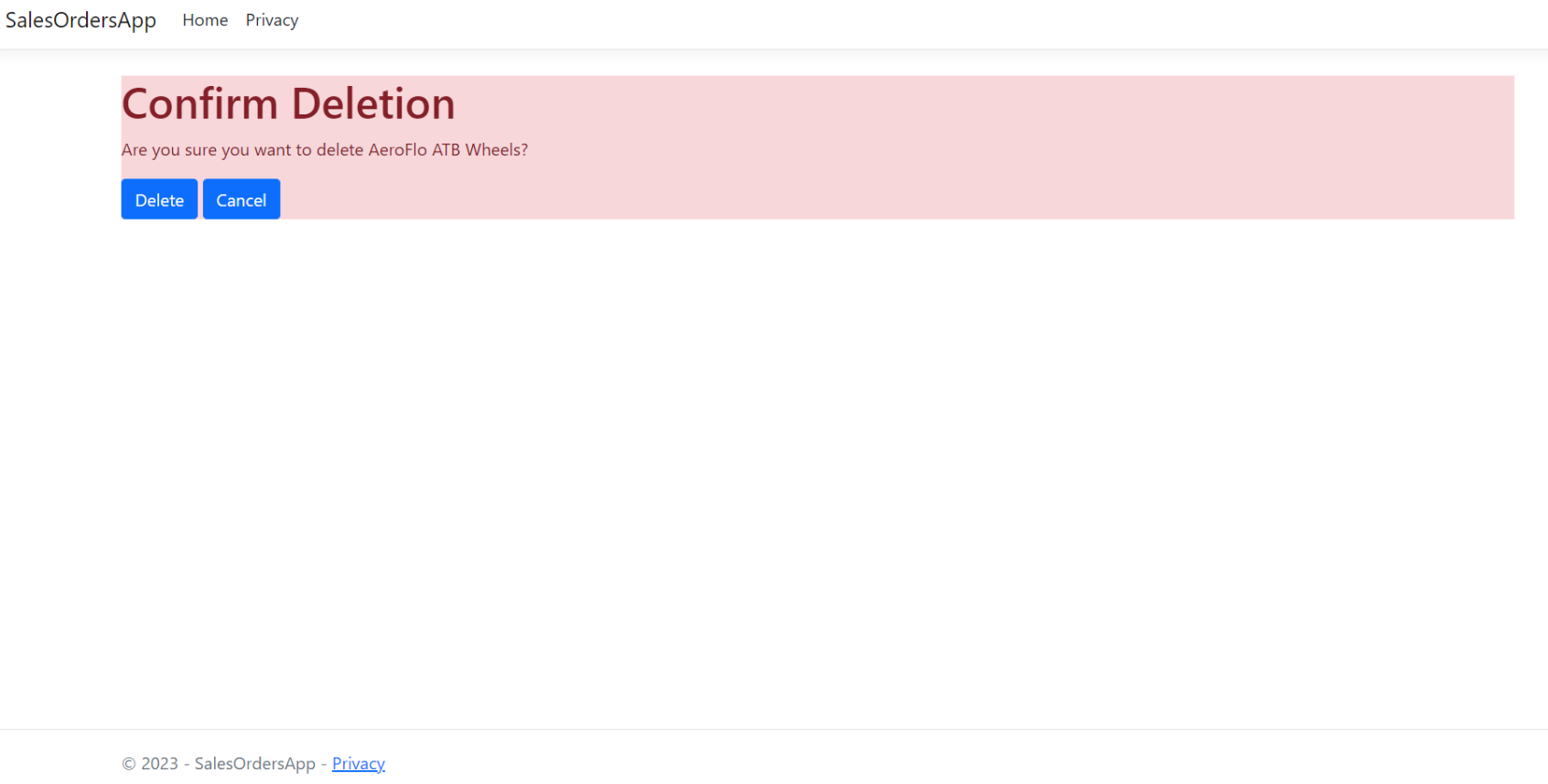
**SCREEN 2 - RESULTS AFTER CLICKING MANAGE PRODUCTS ON THE PREVIOUS SCREEN**



**SCREEN 3 – ADD PRODUCT SCREEN**



**SCREEN 4 – EDIT PRODUCT SCREEN FOR AeroFlo ATB Wheels**



**SCREEN 5 – DELETE SCREEN (After AeroFlo ATB Wheels Chosen To Delete)**

**Rubric**

1. Go into NuGet and add the latest version of the following packages (2 points):
   * **Microsoft.EntityFrameworkCore.SqlServer**
   * **Microsoft.EntityFrameworkCore.Tools**
2. Update appsettings.json to add the ConnectionString. Call the database SalesOrdersDB (6 points)
3. Update Program.cs to add enable Dependency Injection (6 points)
4. Create the Product Model file using the specs shown on Page 1 (6 points)

ProductName, ProductImage required. Include associated error messages.

ProductPrice range should be set to between 1 – 100000

ProductQty range should be set to between 1 - 1000

1. Create the SalesOrderContext file, including adding the DbSet() and Seed data (6 points)
2. Perform a successful Add-Migration, giving the migration the name Migration01-Products.

Take a screenshot proving the migration worked. Save this as part of your repo (5 points)

1. Perform a successful Update-Database, again taking a screenshot proving the update

worked. Save this as part of your repo (5 points)

1. Create a ProductController, including List, AddEdit, and Delete methods, and any

additional methods/code you may deem necessary. (10 points)

1. Create a new Product folder under the View menu and in this folder, create these views:
   * List.cshtml (5 points)
   * AddEdit.cshtml (5 points) As in class, have this handle both Adds and Edits
   * Delete.cshtml (5 points)
2. Add a new Categories Model file using the specs shown on Page 1 (6 points)

CategoryName required. Include associated error message.

Category name should appear in output. Category name input should be in dropdown list format.

1. Update the ProductContext file, including adding the DbSet() and Seed data (6 points)
2. Update the Product model to reflect the changes above, including updating the Seed data (6 points)
3. Perform a second successful Add-Migration, giving the migration the name Migration02-Categories.

Take a screenshot proving the migration worked. Save this as part of your repo (5 points)

1. Perform a successful Update-Database, again taking a screenshot proving the update

worked. Save this as part of your repo (5 points)

1. Add the code to make the URLs lowercase and add a “/” at the end of the URL. (5 points)
2. Add a slug as shown in the text. This slug should be the name of the product, and be added to the end of the URL. (6 points)