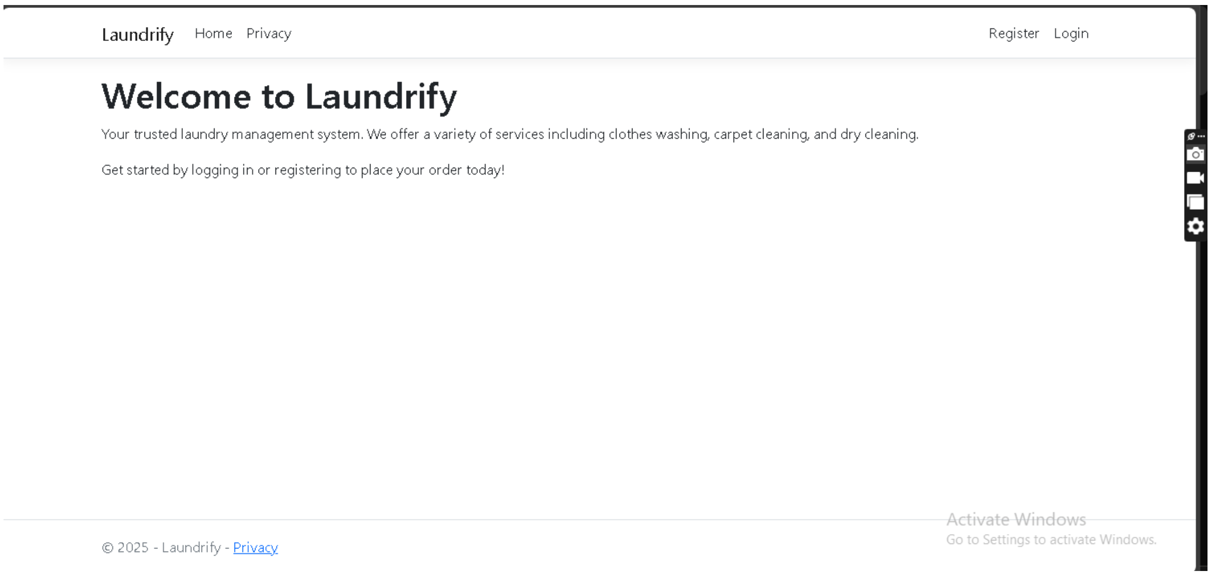
Laundrify - Laundry Management System Documentation

Introduction and Overview

Laundrify is built to make it easier for laundry service companies to handle their work. Laundry management is simple with the system, allowing both administrators and customers to check orders, available services and their own accounts with ease. Using ASP.NET Core Razor Pages, the program is built to function well on all devices, assessing data using the MVC model. The data is persisted using Entity Framework Core and stored in a SQL Server database for all system needs. Only through the custom authentication scheme can users access different sections of the application, according to their roles.



Welcome to Laundrify

Your trusted laundry management system. We offer a variety of services including clothes washing, carpet cleaning, and dry cleaning.

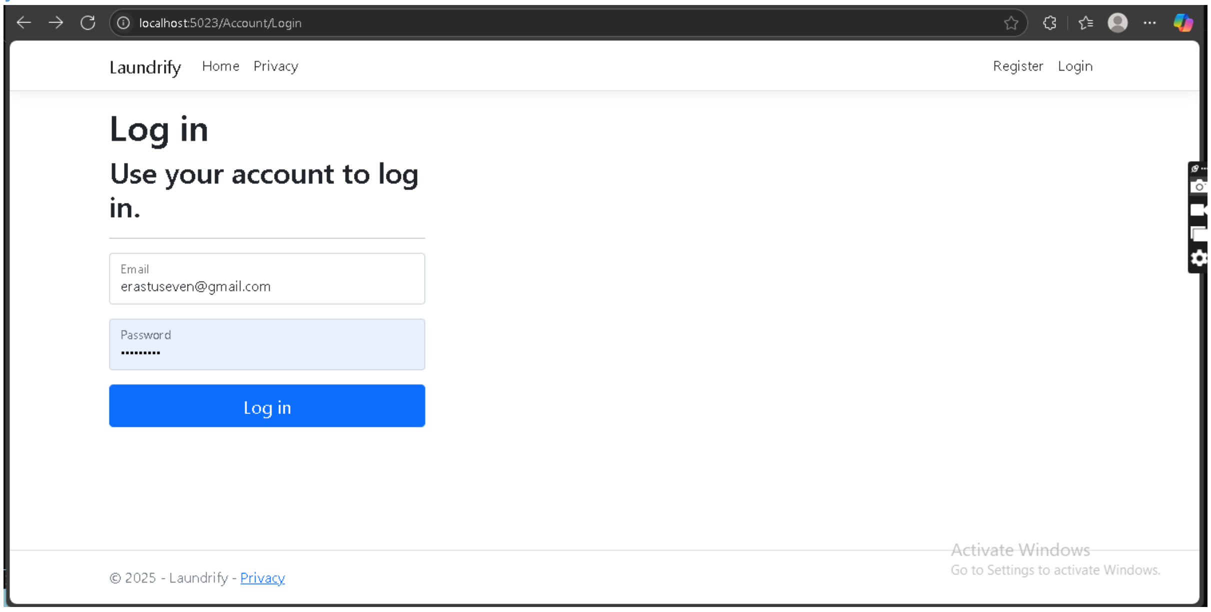
Get started by logging in or registering to place your order today!

© 2025 - Laundrify - Privacy

Introduction and Overview

Laundrify uses a one-of-a-kind authentication process that keeps data secure and manages user roles effectively. The system offers two main types of users called Administrators and Clients. Once you have a role assigned, you'll be given access to certain parts of the system. Login is took care of by the AuthService which handles registration, login and session activity. The system checks the user data which stores the passwords as secure hash values. When the authentication is correct, the system logs the user and sends them to the right dashboard for their role.

```
public class AuthService
{
    private readonly ApplicationDbContext _context;
    private readonly IHttpContextAccessor _httpContextAccessor;

    public AuthService(ApplicationDbContext context, IHttpContextAccessor
httpContextAccessor)
    {
        _context = context;
        _httpContextAccessor = httpContextAccessor;
    }

    public async Task<bool> LoginAsync(string email, string password)
    {
        var user = await _context.Users.FirstOrDefaultAsync(u => u.Email == email);
        if (user == null) return false;

        // Verify password hash
        return VerifyPasswordHash(password, user.PasswordHash);
    }
}
```


localhost:5023/Account/Login

Laundrify Home Privacy Register Login

Log in

Use your account to log in.

Email: erastuseven@gmail.com

Password: •••••••••

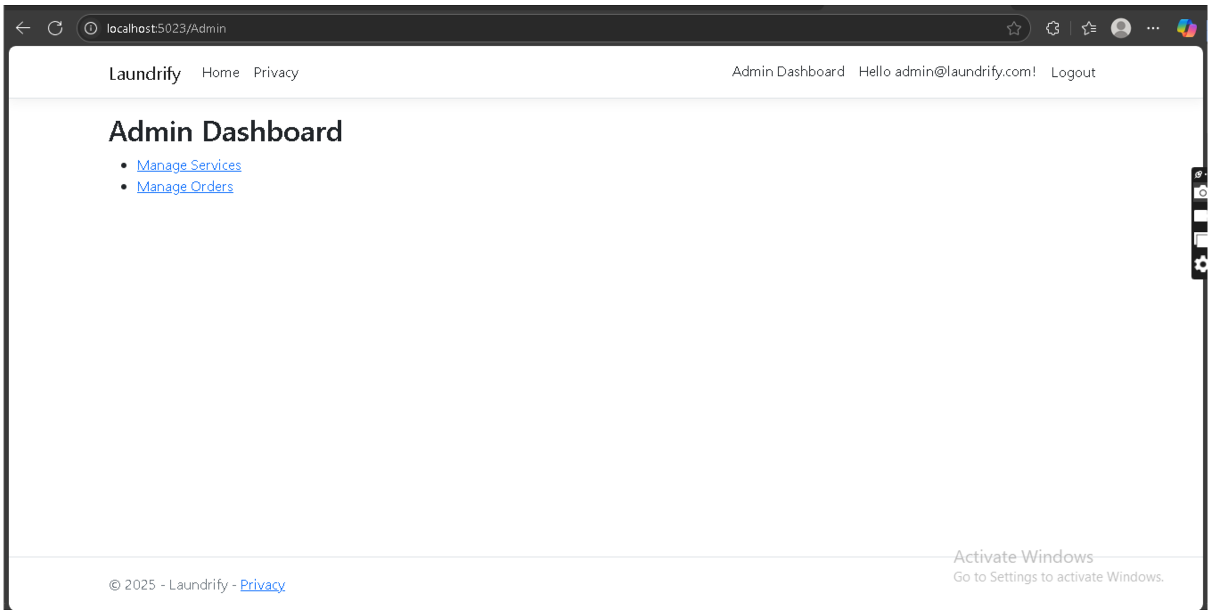
Log in

© 2025 - Laundrify - Privacy

Admin Dashboard and Order Management

The Admin Dashboard is where local managers review and manage all aspects of the laundry service. The main dashboard gives a quick look at total orders, orders yet to be processed, total clients and the available services. This information is organized on attractive cards and each card provides relevant statistics and details. The table in Task Orders is fully responsive, displaying what matters most about a client, the kind of service, the count, whether it’s pending and important dates. Via the dropdown menu, an order’s status may be changed and the new status is updated directly in the database without delay.

```
    public class ManageOrdersModel : PageModel
    {
      private readonly ApplicationDbContext _db;
      private readonly ILogger<ManageOrdersModel> _logger;

      public ManageOrdersModel(ApplicationDbContext db,
ILogger<ManageOrdersModel> logger)
      {
        _db = db;
        _logger = logger;
      }

      public async Task<IActionResult> OnPostUpdateStatusAsync(int orderId, string
status)
      {
        try
        {
          var order = await _db.Orders.FindAsync(orderId);
          if (order == null) return NotFound();

          order.Status = status;
          await _db.SaveChangesAsync();
          return new JsonResult(new { success = true });
        }
        catch (Exception ex)
        {
          _logger.LogError(ex, "Error updating order status");
          return new JsonResult(new { success = false, message = ex.Message });
        }
      }
    }
```


localhost:5023/Admin

Laundrify Home Privacy Admin Dashboard Hello admin@laundrify.com! Logout

Admin Dashboard

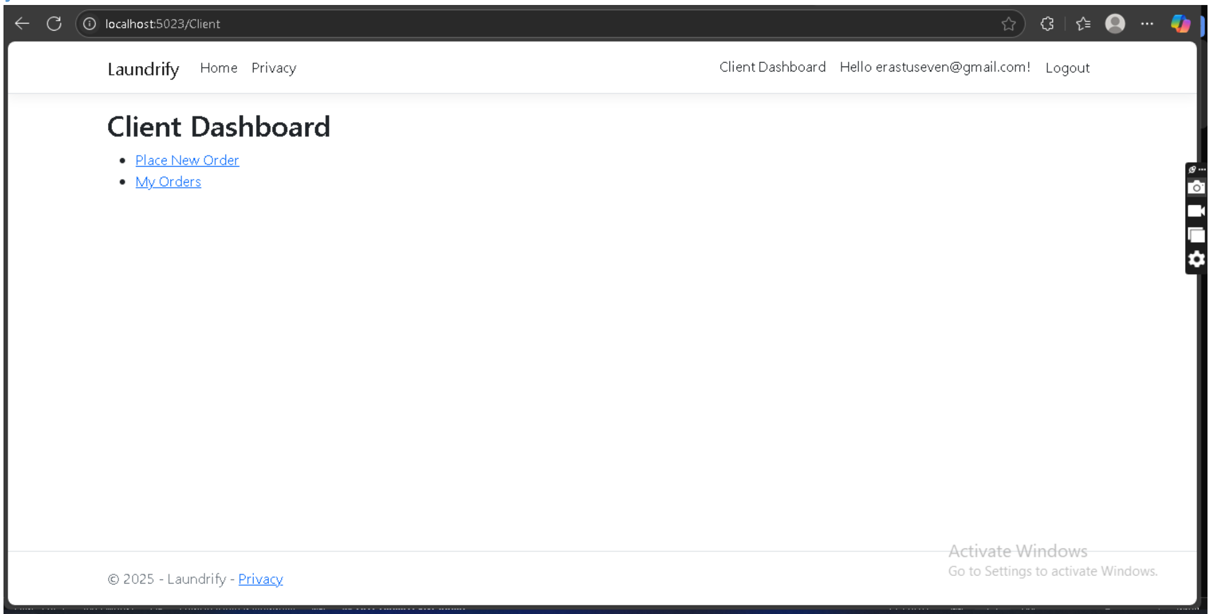
- Manage Services
- Manage Orders

Activate Windows
Go to Settings to activate Windows.

© 2025 - Laundrify - Privacy

Client Interface and Order Placement

On the client interface, people can easily place and follow their laundry orders. Steps in the process make it easy for users to pick their services, decide on how much they want and choose when they would like to drop off and pick up the order. The software gives users real-time information and feedback to correctly submit their orders. All orders appear with their status and clicking on any order shows the details, the amount provided and the dates the order was made. You also get notified about key updates and changes in the system.

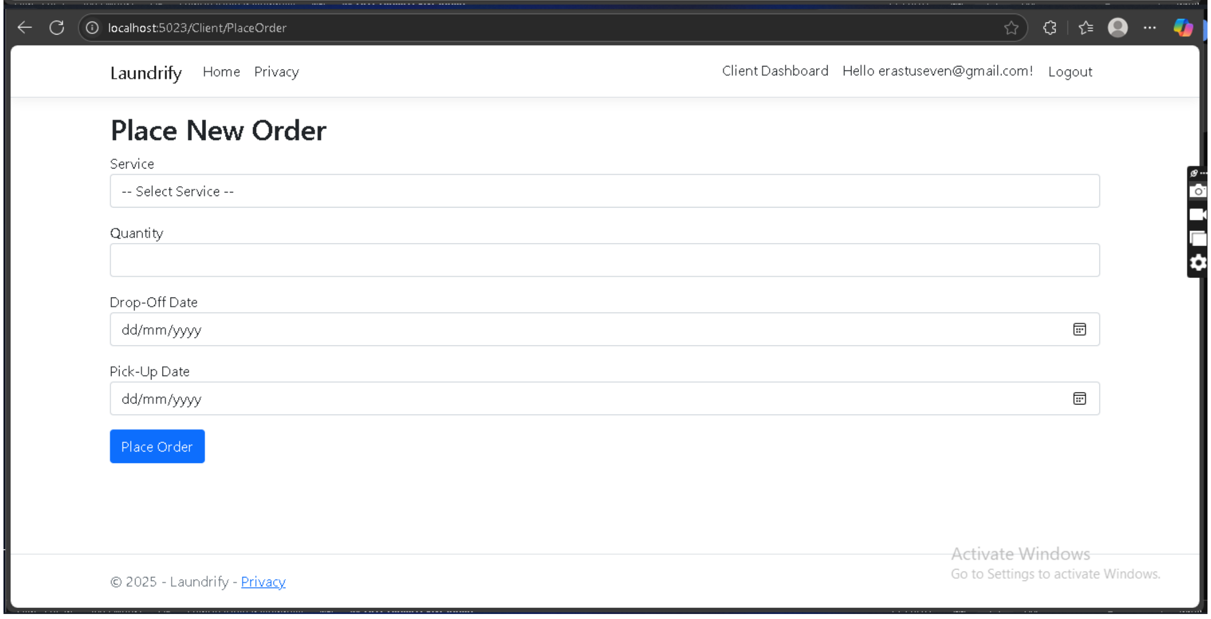
```
public class PlaceOrderModel : PageModel
{
    private readonly ApplicationDbContext _db;
    private readonly AuthService _authService;

    public PlaceOrderModel(ApplicationDbContext db, AuthService authService)
    {
        _db = db;
        _authService = authService;
    }

    public async Task<IActionResult> OnPostAsync()
    {
        if (!ModelState.IsValid) return Page();

        var currentUser = await _authService.GetCurrentUserAsync();
        if (currentUser == null) return RedirectToPage("/Account/Login");

        var order = new Order
        {
            ClientId = currentUser.Id,
            ServiceId = Input.ServiceId,
            Quantity = Input.Quantity,
            Status = "Pending",
            DropOffDate = Input.DropOffDate,
            PickUpDate = Input.PickUpDate
        };

        _db.Orders.Add(order);
        await _db.SaveChangesAsync();

        return RedirectToPage("/Client/Orders");
    }
}
```

localhost:5023/Client

Laundrify Home Privacy Client Dashboard Hello erastuseven@gmail.com! Logout

Client Dashboard

- Place New Order
- My Orders

© 2025 - Laundrify - Privacy


localhost:5023/Client/PlaceOrder

Laundrify Home Privacy Client Dashboard Hello erastuseven@gmail.com! Logout

Place New Order

Service: -- Select Service --

Quantity:

Drop-Off Date: dd/mm/yyyy

Pick-Up Date: dd/mm/yyyy

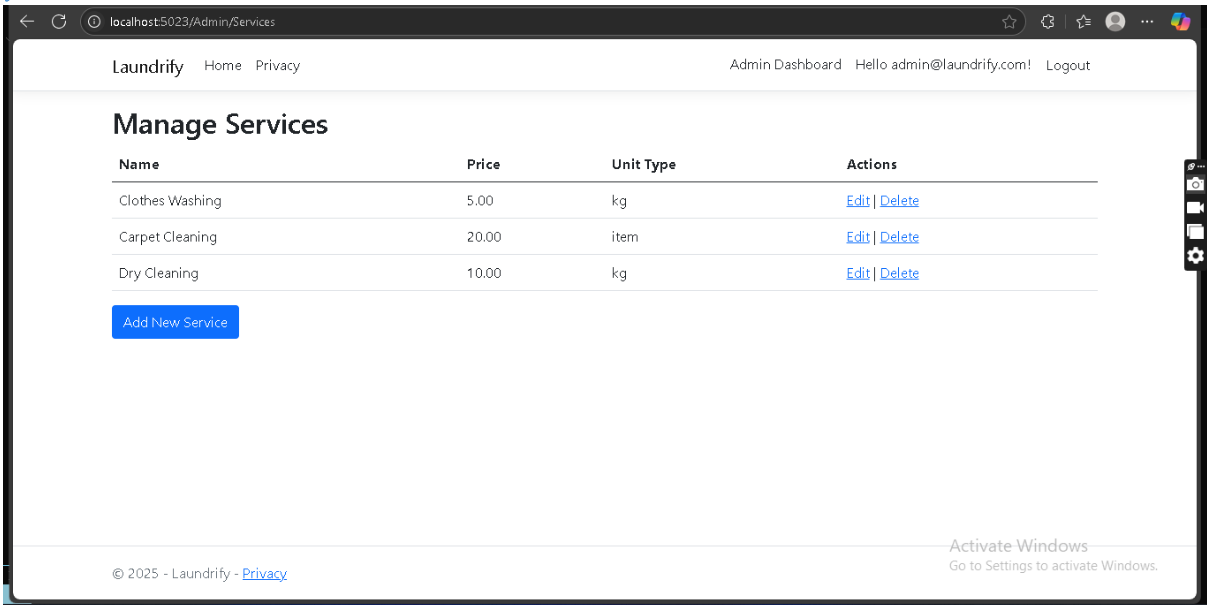
Place Order

© 2025 - Laundrify - Privacy

Service Management

With this module, administrators are able to monitor the list of services users can choose. You may have to make changes by introducing new services, changing some old ones and eliminating those that have become unnecessary. Each service has attributes such as its name, a description and its price set. In the interface, you can observe and decide anytime if certain services should be updated or erased. Whenever services are updated, the changes won't be applied to running orders, maintaining the integrity of the information in the application.

```
public class Service
{
    public int Id { get; set; }
    public string Name { get; set; }
    public string Description { get; set; }
    public decimal Price { get; set; }
    public bool IsActive { get; set; }
    public ICollection<Order> Orders { get; set; }
}
```


localhost:5023/Admin/Services
Laundrify
Home
Privacy
Admin Dashboard
Hello admin@laundrify.com!
Logout

Manage Services

Name	Price	Unit Type	Actions
Clothes Washing	5.00	kg	Edit | Delete
Carpet Cleaning	20.00	item	Edit | Delete
Dry Cleaning	10.00	kg	Edit | Delete

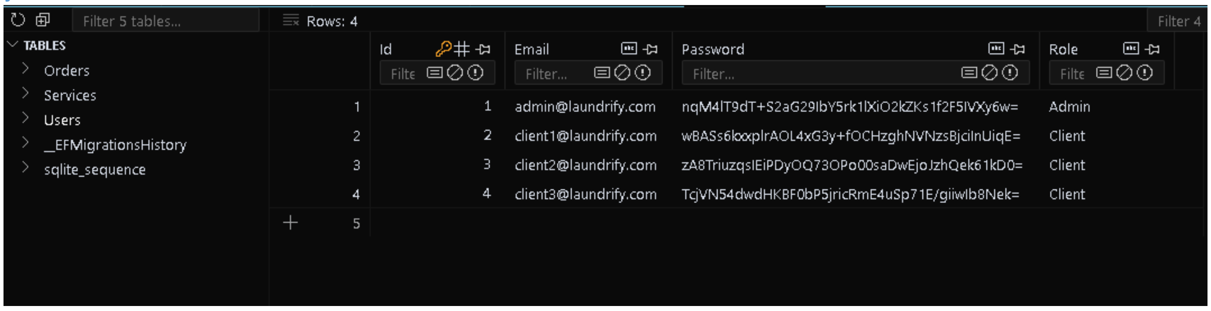
Add New Service
Activate Windows
Go to Settings to activate Windows.

© 2025 - Laundrify - Privacy

Database Structure and Data Models

All system information is organized and supported smoothly in Laundrify's database. The main components are Users, Orders and Services, all of which are related and subject to data integrity constraints. The system utilizes Entity Framework Core for database operations, designing the database through code first. Both administrators and clients are represented by the User model and role-based access control is provided with the help of a Role property. For every laundry order, the Order model includes who the client is, the service requested, along with current status and scheduled appointments. The Service model provides a list of services, along with the cost and what is included.

```
public class ApplicationDbContext : DbContext
{
    public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)
        : base(options)
    {
    }

    public DbSet<User> Users { get; set; }
    public DbSet<Order> Orders { get; set; }
    public DbSet<Service> Services { get; set; }

    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        base.OnModelCreating(modelBuilder);

        modelBuilder.Entity<User>()
            .HasIndex(u => u.Email)
            .IsUnique();

        modelBuilder.Entity<Order>()
            .HasOne(o => o.Client)
            .WithMany(u => u.Orders)
            .HasForeignKey(o => o.ClientId);

        modelBuilder.Entity<Order>()
            .HasOne(o => o.Service)
            .WithMany(s => s.Orders)
            .HasForeignKey(o => o.ServiceId);
    }
}
```



	Id	Email	Password	Role
1	1	admin@laundrify.com	nqM4lT9dT+S2aG29lbY5rk1lXiO2kZKs1f2F5IVXy6w=	Admin
2	2	client1@laundrify.com	wBASs6kxxplrAOL4xG3y+fOCHzghNVNzsBjcilnUiqE=	Client
3	3	client2@laundrify.com	zA8TriuzqsIEiPDyOQ73OPo00saDwEjoJzhQek61kD0=	Client
4	4	client3@laundrify.com	TcjVN54dwdHKBF0bP5jricRmE4uSp71E/giiwlb8Nek=	Client
5				

Security and Error Handling

The company works to secure your information and provide careful access control to its systems. Various security processes are included in the application such as using password hashing, preventing CSRF and carrying out role-based authorization. All important operations in a program require users to be properly authenticated and authorized. The application also includes error handling to offer users valuable feedback and ensure it remains stable. A try-catch block, log function and easy-to-understand error messages are used in the system to address all sorts of errors.

```
public class ErrorHandlingMiddleware
{
    private readonly RequestDelegate _next;
    private readonly ILogger<ErrorHandlingMiddleware> _logger;

    public ErrorHandlingMiddleware(RequestDelegate next, ILogger<ErrorHandlingMiddleware> logger)
    {
        _next = next;
        _logger = logger;
    }

    public async Task InvokeAsync(HttpContext context)
    {
        try
        {
            await _next(context);
        }
        catch (Exception ex)
        {
            _logger.LogError(ex, "An unhandled exception occurred");
            context.Response.StatusCode = 500;
            await context.Response.WriteAsync("An error occurred. Please try again later.");
        }
    }
}
```

User Interface and Experience

Laundrify is made so that everyone can easily manage their laundry tasks. Ensuring anyone can work with Gator App Builder easily on any device, Bootstrap 5 is used for responsive design. The interface allows simple and logical navigation, a clear order of elements and one style across the whole app. Because the interface is simple and intuitive, those using it need less training.

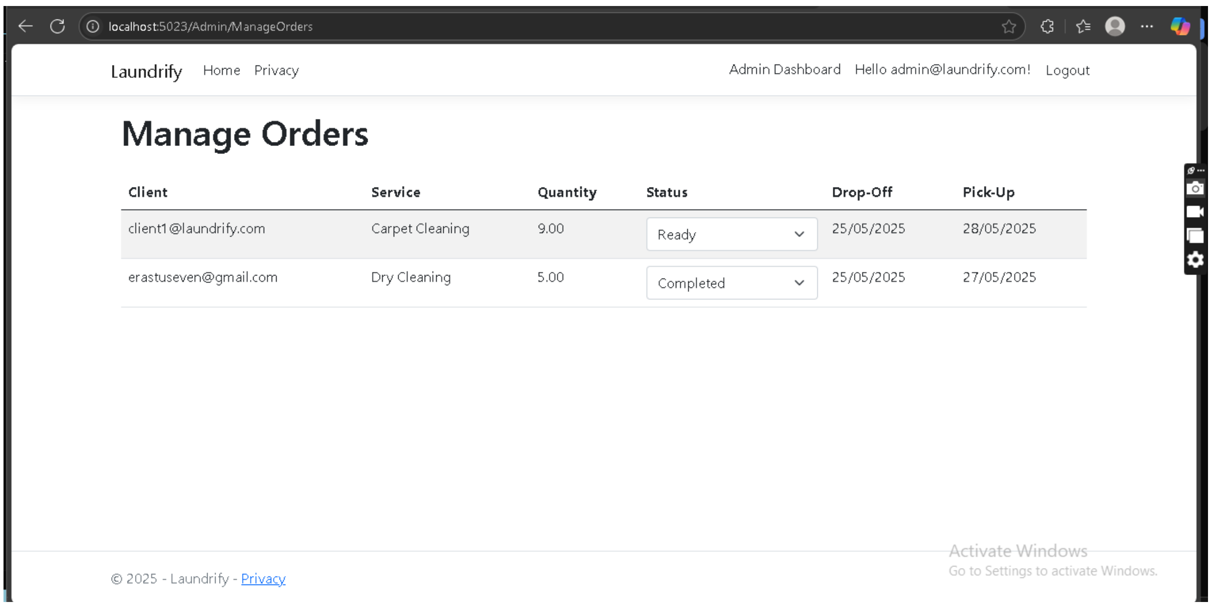
```
<div class="container">
  <div class="row">
    <div class="col-md-8 offset-md-2">
      <div class="card">
        <div class="card-body">
          <h5 class="card-title">Order Status</h5>
          <div class="alert alert-info">
            Your order is currently being processed.
          </div>
          <div class="progress">
            <div class="progress-bar" role="progressbar" style="width:
75%"></div>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>
```

Maintenance and Support

A laundry management system such as Laundrify needs to be constantly updated and supported by helpful people. The application was developed to reduce periods without service and to guarantee that both administrators and clients can use its services anytime. Updating .NET runtime and dependencies when updating your NuGet packages protects against security flaws and boosts the application’s speed. Official as well as third-party repositories for packages should be watched by the development team and all available patches should be quickly installed. The Laundrify system is built using Entity Framework Core and SQL Server, so backing up the database gets important regularly in case of hardware problems or accidents that erase data. Regularly, administrators need to review the database’s integrity, rebuild indexes and clean up completed and stored orders that have aged past a set period. They help users keep up with any changes by showing available data and managing data requests. Those using the system can reach out with their questions, report problems or get help without leaving the website. Customers send their requests to support by email or through tickets, depending on the company’s settings and the support staff promptly responds to handle them. The system is explained and supported using comprehensive manuals and tutorials so users can solve most problems themselves. The support team can assign tough problems to the development team which handles looking into and fixing them. Laundrify relies on ASP.NET Core to log important activities, errors and warnings to either a log file or external logging systems. Going through the logs, administrators can understand trends, find reasons for problems and design strategies for upcoming improvements. Teams working in a business environment should use centralized tools like Azure Application Insights or ELK Stack to study issues in their systems and their users’ actions.Security concerns should be reviewed by running regular checks to know if the systems are secure. Among other things, this means looking at access permissions, checking that admins are legitimate and making certain sensitive data is kept safe during and after transit. The infrastructure’s maintenance plan includes technical upgrades, monitoring databases, user assistance, reporting of actions, security testing and more. Putting these best practices in place, administrators give the system and all its users a reliable foundation for the future.

Conclusion

Laundrify provides a fully equipped system for running laundry services that serves both administrators and users in a convenient and simple way. The site is built using ASP.NET Core Razor Pages and Entity Framework Core which allows it to expand, be maintained and remain secure, helping companies of any size. Throughout this documentation, we have looked at Laundrify's main functions, including registering users, setting roles, organizing services, handling orders, the database design and security measures powered by Bootstrap 5. You will also find code snippets and advice on where to put your screenshots in each section, giving you both practical understanding and visual support. Because the system is modular, you can upgrade or add features as your company grows and you can see that the software is designed with privacy and legal requirements in mind. Getting the solution up and running is simple and tools provide in-depth guidelines for using it in real-world applications. To maintain success, the system relies on tools and processes that keep the system secure, reliable and satisfy users.

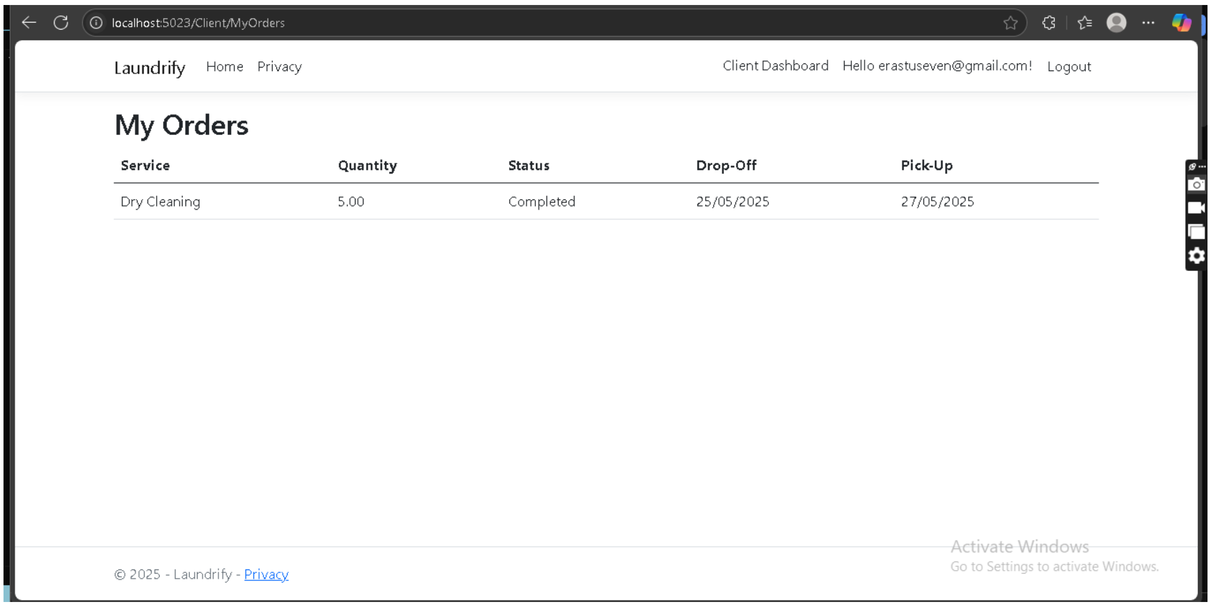


Manage Orders

Client	Service	Quantity	Status	Drop-Off	Pick-Up
client1@laundrify.com	Carpet Cleaning	9.00	Ready	25/05/2025	28/05/2025
erastuseven@gmail.com	Dry Cleaning	5.00	Completed	25/05/2025	27/05/2025

© 2025 - Laundrify - Privacy

Activate Windows
Go to Settings to activate Windows.



localhost:5023/Client/MyOrders

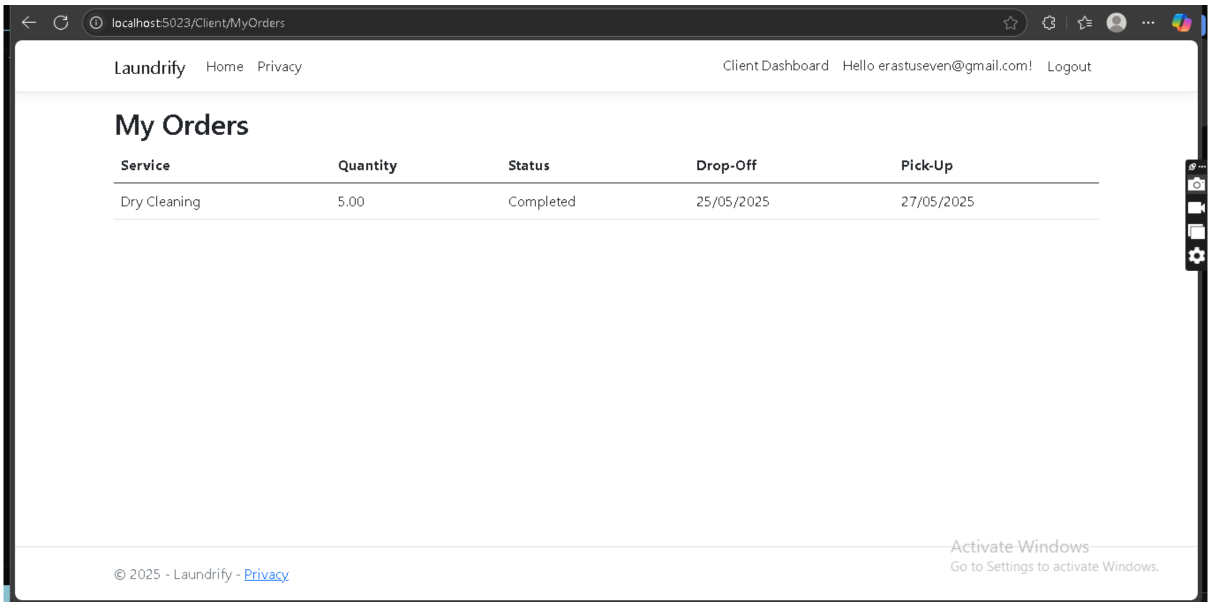
Laundrify Home Privacy Client Dashboard Hello erastuseven@gmail.com! Logout

My Orders

Service	Quantity	Status	Drop-Off	Pick-Up
Dry Cleaning	5.00	Completed	25/05/2025	27/05/2025

© 2025 - Laundrify - Privacy

Activate Windows
Go to Settings to activate Windows.



localhost:5023/Client/MyOrders

Laundrify Home Privacy Client Dashboard Hello erastuseven@gmail.com! Logout

My Orders

Service	Quantity	Status	Drop-Off	Pick-Up
Dry Cleaning	5.00	Completed	25/05/2025	27/05/2025

© 2025 - Laundrify - Privacy

Activate Windows
Go to Settings to activate Windows.