To create a full application using **.NET MAUI Blazor Hybrid** with Stripe for payments, I will break the process into steps, covering both the website and the mobile app. This setup will enable customers to place orders and pay online while allowing drivers to manage and deliver orders.

**1. Project Setup**

**Install Tools**

1. Install **Visual Studio 2022** with the .NET MAUI workload.
2. Ensure **Stripe API Key** is set up from [Stripe Dashboard](https://dashboard.stripe.com/).

**2. Create a Backend API**

**Backend Project**

1. Create a new ASP.NET Core Web API project.
2. Install required NuGet packages:
   * Stripe.net for Stripe integration.
   * Microsoft.EntityFrameworkCore and Microsoft.EntityFrameworkCore.SqlServer for database.
3. Set up database with **Entity Framework Core**.

**Stripe Configuration**

1. Add Stripe to appsettings.json:
2. {
3. "Stripe": {
4. "SecretKey": "your\_secret\_key",
5. "PublishableKey": "your\_publishable\_key"
6. },
7. "ConnectionStrings": {
8. "DefaultConnection": "YourDatabaseConnectionString"
9. }
10. }
11. Configure Stripe in Program.cs:
12. var stripeSettings = builder.Configuration.GetSection("Stripe");
13. StripeConfiguration.ApiKey = stripeSettings["SecretKey"];

**Database Models**

Define database entities for menu items and orders:

**MenuItem.cs**

public class MenuItem

{

public int Id { get; set; }

public string Name { get; set; }

public string Description { get; set; }

public decimal Price { get; set; }

}

**Order.cs**

public class Order

{

public int Id { get; set; }

public string CustomerName { get; set; }

public string CustomerEmail { get; set; }

public string Address { get; set; }

public decimal TotalAmount { get; set; }

public string Status { get; set; } = "Pending";

}

**Create APIs**

**Menu Controller**

[ApiController]

[Route("api/menu")]

public class MenuController : ControllerBase

{

private readonly AppDbContext \_context;

public MenuController(AppDbContext context)

{

\_context = context;

}

[HttpGet]

public async Task<IActionResult> GetMenuItems()

{

var items = await \_context.MenuItems.ToListAsync();

return Ok(items);

}

}

**Order Controller**

Handle payment and order placement with Stripe.

[ApiController]

[Route("api/orders")]

public class OrdersController : ControllerBase

{

private readonly AppDbContext \_context;

public OrdersController(AppDbContext context)

{

\_context = context;

}

[HttpPost]

public async Task<IActionResult> CreateOrder([FromBody] Order order)

{

// Create Stripe Payment Intent

var options = new PaymentIntentCreateOptions

{

Amount = (long)(order.TotalAmount \* 100), // Convert to cents

Currency = "usd",

ReceiptEmail = order.CustomerEmail,

};

var service = new PaymentIntentService();

var intent = service.Create(options);

// Save order to database

order.Status = "Processing";

\_context.Orders.Add(order);

await \_context.SaveChangesAsync();

return Ok(new { OrderId = order.Id, ClientSecret = intent.ClientSecret });

}

}

**3. Build the Website**

**Set Up Blazor Project**

1. Create a **.NET MAUI Blazor Hybrid** project.
2. Add an HTTP client in MauiProgram.cs to communicate with the API:
3. builder.Services.AddHttpClient("ApiClient", client =>
4. {
5. client.BaseAddress = new Uri("https://your-api-url.com/");
6. });

**Pages**

**Menu Page**

Fetch menu items and allow users to add them to the cart.

@inject HttpClient Http

<PageTitle>Menu</PageTitle>

<h3>Menu</h3>

<ul>

@foreach (var item in MenuItems)

{

<li>

<h4>@item.Name - @item.Price</h4>

<button @onclick="() => AddToCart(item)">Add to Cart</button>

</li>

}

</ul>

@code {

private List<MenuItem> MenuItems = new();

private List<MenuItem> Cart = new();

protected override async Task OnInitializedAsync()

{

MenuItems = await Http.GetFromJsonAsync<List<MenuItem>>("api/menu");

}

private void AddToCart(MenuItem item)

{

Cart.Add(item);

}

}

**Checkout Page**

Collect user details and handle Stripe payment.

@inject HttpClient Http

<PageTitle>Checkout</PageTitle>

<h3>Checkout</h3>

<form>

<input placeholder="Name" @bind="CustomerName" />

<input placeholder="Email" @bind="CustomerEmail" />

<input placeholder="Address" @bind="Address" />

<button @onclick="PlaceOrder">Pay</button>

</form>

@code {

private string CustomerName;

private string CustomerEmail;

private string Address;

private decimal TotalAmount => Cart.Sum(i => i.Price);

private async Task PlaceOrder()

{

var order = new

{

CustomerName,

CustomerEmail,

Address,

TotalAmount

};

var response = await Http.PostAsJsonAsync("api/orders", order);

var result = await response.Content.ReadFromJsonAsync<dynamic>();

var clientSecret = result.ClientSecret;

// Use Stripe.js to handle payment in the browser

}

}

**4. Mobile App for Drivers**

**Order List**

Fetch orders assigned to the driver.

@inject HttpClient Http

<PageTitle>Orders</PageTitle>

<h3>Orders</h3>

<ul>

@foreach (var order in Orders)

{

<li>@order.Address - @order.TotalAmount</li>

}

</ul>

@code {

private List<Order> Orders = new();

protected override async Task OnInitializedAsync()

{

Orders = await Http.GetFromJsonAsync<List<Order>>("api/orders");

}

}

**5. Deploy and Test**

**Website**

1. Publish the Blazor app and deploy to **Azure Static Web Apps** or **Vercel**.
2. Ensure Stripe keys are correctly configured in production.

**Mobile App**

1. Build the MAUI app for Android and iOS.
2. Test payment flows using Stripe’s **test cards**.

This provides a scalable and unified application for your client! Let me know if you'd like more details on any step.