REPORTE TCNICO - API CONSULTA TARJETAS DE CRDITO

Este documento acompaa el resumen ejecutivo y est pensado para que el equipo de desarrollo tenga todo lo necesario para aplicar los fixes.

ERR-001 DEADLOCKS POR MEZCLAR ASYNC/SYNC

? Criticidad: Crtica
? Ubicacin: CardService.cs (lnea 68), CardDetailService.cs (lneas 64, 255, 326), BinesProductInfoService.cs (lneas 53-55), ErrorHandlerMiddleware.cs (lnea 73)
? Explicacin: Las llamadas usan .Result, .Wait() o GetAwaiter().GetResult() dentro de flujos async. Cuando aumenta la concurrencia se agota el thread pool y aparecen deadlocks.

Cdigo con problema

```
Task.WhenAll(taskFranCardData, taskPrivCardData).ConfigureAwait(false).GetAwaiter().GetRes
var cardData = _validateTokenService.ValidateCardToken(cardToken).Result;
Task.Run(() => persisteLog.AddLog(traceIdentifier!.GUID, error, error.Message)).Wait();
```

Cdigo propuesto

```
await Task.WhenAll(taskFranCardData, taskPrivCardData);
var cardData = await _validateTokenService.ValidateCardToken(cardToken);
await persisteLog.AddLog(traceIdentifier!.GUID, error, error.Message);
```

ERR-002 PARALELISMO FALSO EN VALIDATETOKENSERVICE

? Criticidad: Crtica
? Ubicacin: ValidateTokenService.cs (lneas 85-135)
? Explicacin: Parallel.ForEachAsync invoca lgica sncrona dentro de Task.Run. En picos de uso replica el problema de ERR-001 y no devuelve resultados parciales hasta que termina el lote.

```
flowchart LR
    A[Parallel.ForEachAsync] --> B[Task.Run sncrono]
    B --> C[ProccessCardToken bloqueante]
    C --> D[Thread pool agotado]
```

Cdigo con problema

```
    await Parallel.ForEachAsync(data, options, async (card, token) =>
    {
        await Task.Run(() =>
        {
            ProccessCardToken(customer, customerToken, cardsToken, baseUrl, headers, card);
        }, CancellationToken.None);
    });
```

Cdigo propuesto

```
    var tasks = data.Select(card => ProccessCardTokenAsync(
        customer, customerToken, baseUrl, headers, card));

    var results = await Task.WhenAll(tasks);
    var cardsToken = results.Where(card => card is not null).ToList();
```

ERR-003 HTTPCLIENT SIN LMITES NI RESILIENCIA

? Criticidad: Crtica
? Ubicacin: DependencyInjectionHandler.cs (configuracin del cliente), RestService.cs (CreateClient, PostServiceAsync)
? Explicacin: Cada request crea un cliente nuevo sin timeout, reutiliza un handler propio y carece de circuit breaker. Cuando un proveedor tarda minutos respondemos con timeouts generales.

Cdigo con problema

```
    services.AddHttpClient<IRestService, RestService>()
        .AddTransientHttpErrorPolicy(policyBuilder =>
            policyBuilder.WaitAndRetryAsync(Backoff.DecorrelatedJitterBackoffV2(TimeSpan.FromSec

    HttpClientHandler l_objHttpClientHandler = new HttpClientHandler();
    using var httpClient = CreateClient();
```

Cdigo propuesto

```
    services.AddHttpClient<IRestService, RestService>(client =>
    {
```

```
        client.Timeout = TimeSpan.FromSeconds(10);
    })
    .ConfigurePrimaryHttpMessageHandler(() => new HttpClientHandler
    {
        MaxConnectionsPerServer = 50,
    })
    .AddTransientHttpErrorPolicy(builder =>
        builder.WaitAndRetryAsync(Backoff.DecorrelatedJitterBackoffV2(TimeSpan.FromSeconds(1),
    .AddTransientHttpErrorPolicy(builder =>
        builder.CircuitBreakerAsync(5, TimeSpan.FromSeconds(30)))
    .AddPolicyHandler(Policy.TimeoutAsync<HttpResponseMessage>(TimeSpan.FromSeconds(5)))

    public async Task<T> PostServiceAsync<T>(string baseUrl, object parameters, IDictionary<strin
    {
        _createClient.DefaultRequestHeaders.Clear();
        _createClient.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("a
        AddHeadersForReq(headers!, _createClient);

        HttpContent jsonObject = new StringContent(
            JsonConvert.SerializeObject(parameters),
            Encoding.UTF8,
            "application/json");

        HttpResponseMessage res = await _createClient.PostAsync(baseUrl, jsonObject);
        var data = await res.Content.ReadAsStringAsync();
        return JsonConvert.DeserializeObject<T>(data)!;
    }
```

ERR-004 ESCRITURAS PESADAS EN MONGODB DENTRO DEL REQUEST

? Criticidad: Alta
? Ubicacin: CrudService.cs (AddOrUpdate)
? Explicacin: Se actualizan colecciones grandes con AddToSetEach dentro del request. El hilo queda bloqueado hasta que Mongo confirma.

Cdigo con problema

```
    result = await collection.UpdateOneAsync(
        Builders<TEntity>.Filter.Eq(i => i.Id, data.Id),
        Builders<TEntity>.Update
            .SetOnInsert(s => s.Id, data.Id)
            .SetOnInsert(s => s.CreateDateTime, DateTime.Now)
            .Set(s => s.IdCard, data.IdCard)
            .AddToSetEach(s => s.CardsNumber, data.CardsNumber)
            .AddToSetEach(s => s.CardsToken, data.CardsToken)
```

```
            .AddToSetEach(s => s.CardsType, data.CardsType),
        new UpdateOptions { IsUpsert = true });
```

Cdigo propuesto

```
    public async Task QueueCardUpdateAsync(CardData data)
    {
        await _eventBus.PublishAsync("card-data-updated", data);
    }

    public async Task HandleCardUpdateAsync(CardData data)
    {
        await _collection.UpdateOneAsync(
            Builders<CardData>.Filter.Eq(i => i.Id, data.Id),
            Builders<CardData>.Update
                .SetOnInsert(s => s.Id, data.Id)
                .SetOnInsert(s => s.CreateDateTime, DateTime.UtcNow)
                .Set(s => s.IdCard, data.IdCard)
                .AddToSetEach(s => s.CardsNumber, data.CardsNumber)
                .AddToSetEach(s => s.CardsToken, data.CardsToken)
                .AddToSetEach(s => s.CardsType, data.CardsType),
            new UpdateOptions { IsUpsert = true });
    }
```

ERR-005 CACHE DUPLICADO Y COSTOSO EN CARDDETAILSERVICE

? Criticidad: Media
? Ubicacin: CardDetailService.cs (lneas 40-58)
? Explicacin: Cada request reconstruye un cache en memoria y adems consulta Redis sin invalidacin clara.

Cdigo con problema

```
    var cacheKey = $"card-detail-{cardId}";
    if (!_memoryCache.TryGetValue(cacheKey, out CardDetail? detail))
    {
        detail = _redisCache.Get<CardDetail>(cacheKey);
        if (detail is null)
        {
            detail = GetCardDetailFromDb(cardId);
            _redisCache.Set(cacheKey, detail, TimeSpan.FromMinutes(5));
        }
        _memoryCache.Set(cacheKey, detail, TimeSpan.FromMinutes(1));
```

```
    }
    return detail;
```

Cdigo propuesto

```
    public async Task<CardDetail?> GetCardDetailAsync(Guid cardId)
    {
        var cacheKey = $"card-detail-{cardId}";
        return await _distributedCache.GetOrCreateAsync(cacheKey, async entry =>
        {
            entry.AbsoluteExpirationRelativeToNow = TimeSpan.FromMinutes(5);
            return await _cardRepository.GetCardDetailAsync(cardId);
        });
    }
```

ERR-006 LOGGING BLOQUEANTE EN MIDDLEWARE DE ERRORES

? Criticidad: Media
? Ubicacin: ErrorHandlerMiddleware.cs (lneas 68-90)
? Explicacin: El middleware atrapa excepciones pero registra el error con .Wait(),
  devolviendo la respuesta tarde y bloqueando el pipeline.

Cdigo con problema

```
    Task.Run(() =>
    {
        persisteLog.AddLog(traceIdentifier!.GUID, error, error.Message);
    }).Wait();
```

Cdigo propuesto

```
    await persisteLog.AddLog(traceIdentifier!.GUID, error, error.Message);
```

Si necesitan ms contexto o pairing para aplicar los cambios, avsenme y lo vemos juntos.