Guia de API GraphQL - Modulo de Guion (Agente 2 → Agente 3)

Documentacion tecnica para integracion del flujo de recomendacion basado en guiones del Agente 2.

Resumen del Flujo

El sistema implementa un flujo conversacional de ventas con las siguientes etapas:

1. **Entrada de Guion**: Agente 2 (Vision/Reconocimiento) detecta productos y genera un guion con barcodes
2. **Recomendacion**: Agente 3 (Ventas) procesa el guion, compara productos y recomienda el mejor
3. **Confirmacion**: Usuario aprueba o rechaza la recomendacion
4. **Datos de Envio**: Si aprueba, se solicita talla y direccion
5. **Checkout**: Con los datos completos, se redirige a la pantalla de pago

---

Mutations GraphQL

1. procesarGuionAgente2

Inicia el flujo de ventas procesando un guion del Agente 2.

URL: `POST /graphql`

Input: `GuionEntradaInput`

```
mutation ProcesarGuion {
  procesarGuionAgente2(
    guion: {
      sessionId: "sess-unique-001"
      productos: [
        {
          codigoBarras: "7501234567891"
          nombreDetectado: "Nike Air Max 90"
          prioridad: "alta"
          motivoSeleccion: "Zapatilla clasica popular"
        }
        {
          codigoBarras: "7501234567894"
          nombreDetectado: "Nike Court Vision Low"
          prioridad: "media"
          motivoSeleccion: "Alternativa economica"
        }
      ]
      preferencias: {
        estiloComunicacion: "cuencano"  # cuencano | juvenil | formal |
neutral
```

```
        presupuestoMaximo: 150
        urgencia: "media"  # alta | media | baja
        usoPrevisto: "Uso casual diario"
        buscaOfertas: true
      }
      contexto: {
        tipoEntrada: "texto"  # texto | voz | imagen | mixta
        intencionPrincipal: "compra_directa"  # compra_directa | comparar |
informacion
        necesitaRecomendacion: true
      }
      textoOriginalUsuario: "Quiero unas zapatillas comodas para caminar"
      resumenAnalisis: "Usuario busca zapatillas lifestyle para uso casual"
      confianzaProcesamiento: 0.92
    }
  ) {
    success
    mensaje
    productos {
      id
      productName
      finalPrice
      unitCost
      isOnSale
      discountPercent
      recommendationScore
      reason
    }
    mejorOpcionId
    reasoning
    siguientePaso
  }
}
```

Respuesta Exitosa:


```
{
  "data": {
    "procesarGuionAgente2": {
      "success": true,
      "mensaje": "Te recomiendo las Nike Air Max 90. Estan en oferta a
$104, te ahorras $26. Son perfectas para caminar por la ciudad. Te
interesan?",
      "productos": [
        {
          "id": "94d7c19b-856f-4f99-a6e6-553e0a1eac26",
          "productName": "Nike Air Max 90",
          "finalPrice": "104.0000",
          "unitCost": "130.00",
          "isOnSale": true,
          "discountPercent": "10.00",
          "recommendationScore": 85,
```

```
            "reason": "Producto de alta prioridad; Precio dentro del
presupuesto; En oferta"
          },
          {
            "id": "849b045e-05a9-4613-9de3-2e2ef9d67f28",
            "productName": "Nike Court Vision Low",
            "finalPrice": "45.0000",
            "unitCost": "75.00",
            "isOnSale": true,
            "discountPercent": "20.00",
            "recommendationScore": 65,
            "reason": "Precio economico; 20% descuento"
          }
        ],
        "mejorOpcionId": "94d7c19b-856f-4f99-a6e6-553e0a1eac26",
        "reasoning": "Recomendacion: Nike Air Max 90. Precio: $104.00 (antes
$130.00). Razones: Producto prioridad alta, dentro de presupuesto, en
oferta 10%",
        "siguientePaso": "confirmar_compra"
      }
    }
  }
```


Campos de Respuesta:

Campo	Tipo	Descripcion
`success`	Boolean	Indica si la operacion fue exitosa
`mensaje`	String	Mensaje persuasivo generado por el LLM para mostrar al usuario
`productos`	Array	Lista de productos del guion con sus scores de recomendacion
`mejorOpcionId`	UUID	ID del producto recomendado (mejor score)
`siguientePaso`	String	Indica la siguiente accion esperada: `confirmar_compra`

Almacenamiento de Sesion:

La mutation guarda automaticamente la sesion en Redis con:

- Key: `session:{sessionId}`
- TTL: 30 minutos
- Contenido: Productos del guion, seleccion actual, estilo de comunicacion

---

2. continuarConversacion

Continua el flujo de ventas procesando la respuesta del usuario.

URL: `POST /graphql`

Parametros:

Parametro	Tipo	Requerido	Descripcion
sessionId	String	Si	ID de sesion devuelto en procesarGuionAgente2
respuestaUsuario	String	Si	Texto de respuesta del usuario

```
mutation ContinuarConversacion {
  continuarConversacion(
    sessionId: "sess-unique-001"
    respuestaUsuario: "Si me interesan"
  ) {
    success
    mensaje
    mejorOpcionId
    siguientePaso
  }
}
```

Respuesta Exitosa - Caso Aprobacion:

```
{
  "data": {
    "continuarConversacion": {
      "success": true,
      "mensaje": "Excelente! Que talla necesitas y a que direccion te los
enviamos?",
      "mejorOpcionId": "94d7c19b-856f-4f99-a6e6-553e0a1eac26",
      "siguientePaso": "solicitar_datos_envio"
    }
  }
}
```

Respuesta Exitosa - Caso Rechazo (con alternativas disponibles):

```
mutation ContinuarConversacion {
  continuarConversacion(
    sessionId: "sess-unique-001"
    respuestaUsuario: "No me gustan"
  ) {
    success
    mensaje
    mejorOpcionId
    siguientePaso
  }
}
```

```
{
  "data": {
    "continuarConversacion": {
      "success": true,
      "mensaje": "Entiendo. Entonces mira esta opcion: Nike Court Vision
Low a $45.00 (en oferta). Te interesa?",
      "mejorOpcionId": "849b045e-05a9-4613-9de3-2e2ef9d67f28",
      "siguientePaso": "confirmar_compra"
    }
  }
}
```


Respuesta - Caso Rechazo (sin alternativas):


```
{
  "data": {
    "continuarConversacion": {
      "success": true,
      "mensaje": "Entiendo. No tengo mas opciones de las que te mostre.
Quieres que hagamos una nueva busqueda?",
      "mejorOpcionId": "00000000-0000-0000-0000-000000000000",
      "siguientePaso": "nueva_conversacion"
    }
  }
}
```


Respuesta - Caso Datos de Envio:

```
mutation ContinuarConversacion {
  continuarConversacion(
    sessionId: "sess-unique-001"
    respuestaUsuario: "Talla 42, direccion Av. Americas 123 y 6 de
Diciembre"
  ) {
    success
    mensaje
    mejorOpcionId
    siguientePaso
  }
}
```


```
{
  "data": {
    "continuarConversacion": {
      "success": true,
      "mensaje": "Perfecto! Recibi talla 42 y direccion Av. Americas. Ahora
```

```
  te llevo a completar la compra.",
        "mejorOpcionId": "94d7c19b-856f-4f99-a6e6-553e0a1eac26",
        "siguientePaso": "ir_a_checkout"
      }
    }
  }
```

Valores de siguientePaso:

Valor	Descripcion	Accion del Frontend
`confirmar_compra`	Esperando confirmacion del usuario	Mostrar mensaje y opciones Si/No
`solicitar_datos_envio`	Usuario aprobo, se necesita talla y direccion	Mostrar formulario de datos de envio
`ir_a_checkout`	Datos completos, listo para pago	Redirigir a pantalla de checkout
`nueva_conversacion`	Sin alternativas o sesion expirada	Ofrecer iniciar nueva busqueda

Flujos de Uso

Flujo 1: Compra Directa (Usuario Aprueba a la Primera)

```
Paso 1: Frontend llama procesarGuionAgente2
        ↓
Paso 2: Recibe mensaje persuasivo + siguientePaso: "confirmar_compra"
        ↓
Paso 3: Muestra mensaje al usuario con botones "Si, me interesa" / "No,
gracias"
        ↓
Paso 4: Usuario presiona "Si, me interesa"
        ↓
Paso 5: Frontend llama continuarConversacion con respuestaUsuario: "Si me
interesa"
        ↓
Paso 6: Recibe siguientePaso: "solicitar_datos_envio"
        ↓
Paso 7: Muestra formulario de talla y direccion
        ↓
Paso 8: Usuario completa datos
        ↓
Paso 9: Frontend llama continuarConversacion con datos
        ↓
Paso 10: Recibe siguientePaso: "ir_a_checkout"
        ↓
Paso 11: Redirige a /checkout?session=sess-unique-001
```

Flujo 2: Usuario Rechaza y Selecciona Alternativa

```
Paso 1: Frontend llama procesarGuionAgente2 (guion con 2+ productos)
        ↓
Paso 2: Recibe recomendacion del producto 1 (mejor score)
        ↓
Paso 3: Muestra mensaje al usuario
        ↓
Paso 4: Usuario presiona "No me gustan"
        ↓
Paso 5: Frontend llama continuarConversacion con respuestaUsuario: "No me
gustan"
        ↓
Paso 6: Backend detecta rechazo y recomienda producto 2 (segunda opcion del
guion)
        ↓
Paso 7: Recibe nuevo mensaje + siguientePaso: "confirmar_compra"
        ↓
Paso 8: Muestra nueva recomendacion
        ↓
Paso 9: Usuario presiona "Si, esas si"
        ↓
Paso 10: Frontend llama continuarConversacion
        ↓
Paso 11: Recibe siguientePaso: "solicitar_datos_envio"
        ↓
Paso 12: Continua flujo normal (datos envio → checkout)
```

Flujo 3: Usuario Rechaza Todo

```
Paso 1: Frontend llama procesarGuionAgente2 (guion con 1 o 2 productos)
        ↓
Paso 2: Recibe recomendacion del producto 1
        ↓
Paso 3: Usuario rechaza
        ↓
Paso 4: Frontend llama continuarConversacion
        ↓
Paso 5: Backend recomienda producto 2 (si existe)
        ↓
Paso 6: Usuario vuelve a rechazar
        ↓
Paso 7: Frontend llama continuarConversacion
        ↓
Paso 8: Recibe siguientePaso: "nueva_conversacion"
        ↓
Paso 9: Muestra mensaje "No tengo mas opciones" + boton "Nueva busqueda"
        ↓
Paso 10: Usuario presiona "Nueva busqueda"
        ↓
Paso 11: Frontend redirige a pantalla de busqueda o inicia nuevo flujo
```

Manejo de Sesiones

Almacenamiento

Las sesiones se almacenan en Redis con las siguientes caracteristicas:

- **Key**: `session:{session_id}`
- **TTL**: 30 minutos (1800 segundos)
- **Formato**: JSON serializado del estado de la conversacion

Estructura de la Sesion

```
{
  "session_id": "sess-unique-001",
  "user_query": "Quiero zapatillas",
  "search_results": [
    {
      "id": "uuid-producto-1",
      "name": "Nike Air Max 90",
      "price": 104.00,
      "barcode": "7501234567891",
      "is_on_sale": true
    },
    {
      "id": "uuid-producto-2",
      "name": "Nike Court Vision Low",
      "price": 45.00,
      "barcode": "7501234567894",
      "is_on_sale": true
    }
  ],
  "selected_products": ["uuid-producto-1"],
  "conversation_stage": "esperando_confirmacion",
  "metadata": {
    "estilo": "cuencano",
    "producto_recomendado": "Nike Air Max 90",
    "precio": 104.00
  },
  "created_at": "2026-02-05T15:30:00Z"
}
```

Manejo de Expiracion

Si una sesion expira (pasan 30 minutos sin actividad):

```
{
  "data": {
    "continuarConversacion": {
      "success": false,
```

```
      "mensaje": "La sesion expiro. Por favor, inicia una nueva
conversacion.",
      "siguientePaso": "nueva_conversacion"
    }
  }
}
```

Accion del Frontend: Mostrar mensaje de sesion expirada y boton para reiniciar.

Tipos de Input GraphQL

GuionEntradaInput

```
input GuionEntradaInput {
  sessionId: String!
  productos: [ProductoEnGuionInput!]!
  preferencias: PreferenciasUsuarioInput!
  contexto: ContextoBusquedaInput!
  textoOriginalUsuario: String!
  resumenAnalisis: String!
  confianzaProcesamiento: Float!
}
```

ProductoEnGuionInput

```
input ProductoEnGuionInput {
  codigoBarras: String!
  nombreDetectado: String!
  marca: String
  categoria: String
  prioridad: String  # alta | media | baja
  motivoSeleccion: String
}
```

PreferenciasUsuarioInput

```
input PreferenciasUsuarioInput {
  estiloComunicacion: String    # cuencano | juvenil | formal | neutral
  usoPrevisto: String
  nivelActividad: String        # alto | medio | bajo
  tallaPreferida: String
  colorPreferido: String
  presupuestoMaximo: Decimal
  buscaOfertas: Boolean
  urgencia: String              # alta | media | baja
```

```
  caracteristicasImportantes: [String!]
}
```

ContextoBusquedaInput

```
input ContextoBusquedaInput {
  tipoEntrada: String!          # texto | voz | imagen | mixta
  productoMencionadoExplicitamente: Boolean
  necesitaRecomendacion: Boolean!
  intencionPrincipal: String!   # compra_directa | comparar | informacion
  restriccionesAdicionales: [String!]
}
```

Tipos de Respuesta GraphQL

RecomendacionResponse

```
type RecomendacionResponse {
  success: Boolean!
  mensaje: String!
  productos: [ProductComparisonType!]!
  mejorOpcionId: UUID!
  reasoning: String!
  siguientePaso: String!
}
```

ProductComparisonType

```
type ProductComparisonType {
  id: UUID!
  productName: String!
  barcode: String
  brand: String
  category: String
  unitCost: Decimal!
  finalPrice: Decimal!
  savingsAmount: Decimal!
  isOnSale: Boolean!
  discountPercent: Decimal
  promotionDescription: String
  quantityAvailable: Int!
  recommendationScore: Float!
  reason: String!
}
```

Headers Requeridos

Todas las mutations requieren autenticacion JWT:

```
Authorization: Bearer {token_jwt}
Content-Type: application/json
```

Obtener token via REST:

```
POST /auth/login
{
  "username": "Cliente1",
  "password": "cliente123"
}
```

Ejemplo de Integracion Completa (React)

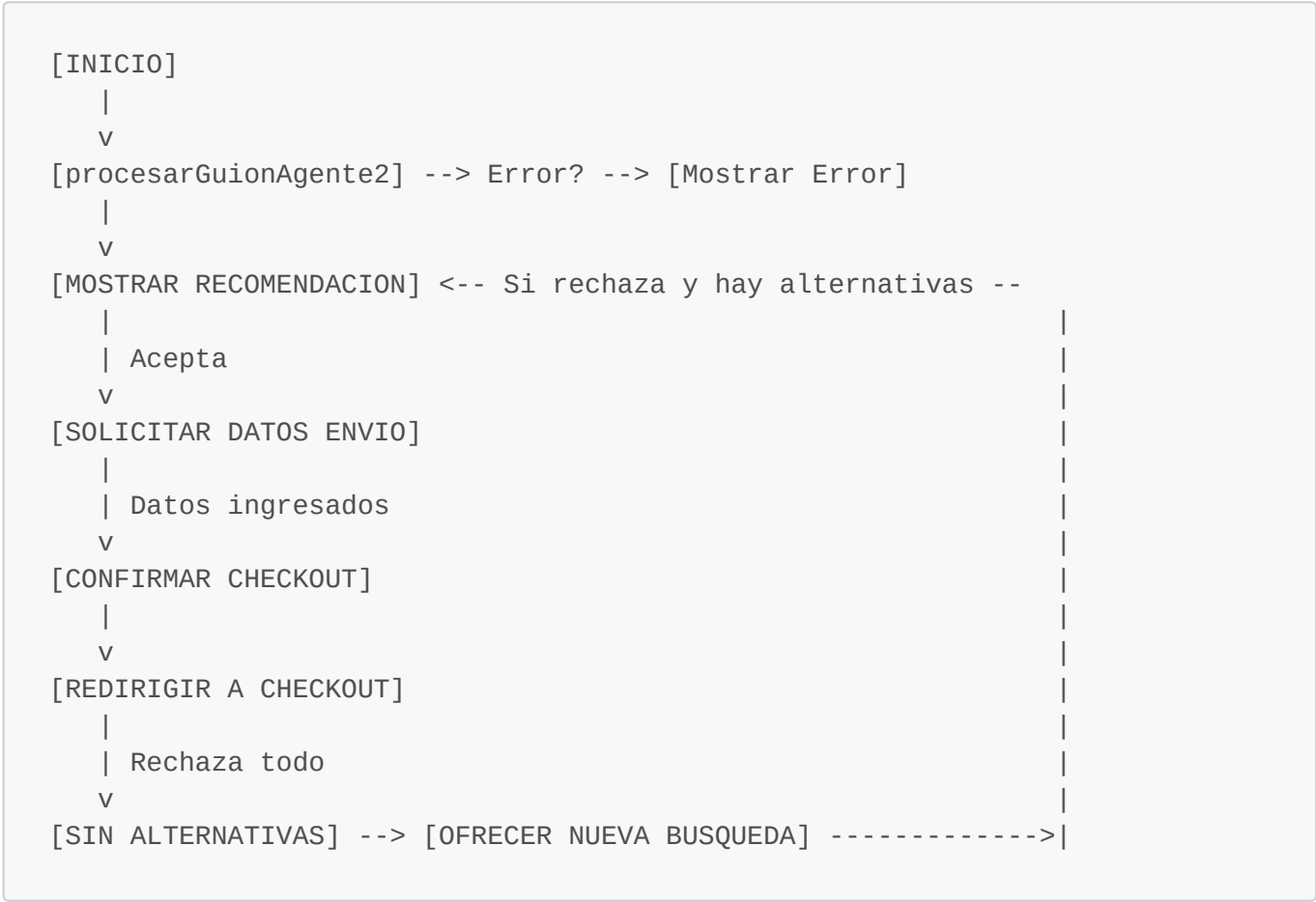
```
// 1. Iniciar conversacion
const iniciarConversacion = async (guion) => {
  const response = await fetch('/graphql', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
      'Authorization': `Bearer ${token}`
    },
    body: JSON.stringify({
      query: `
        mutation ProcesarGuion($guion: GuionEntradaInput!) {
          procesarGuionAgente2(guion: $guion) {
            success
            mensaje
            mejorOpcionId
            siguientePaso
          }
        }
      `,
      variables: { guion }
    })
  });

  const data = await response.json();

  if (data.data.procesarGuionAgente2.siguientePaso === 'confirmar_compra')
{
    // Mostrar mensaje y botones Si/No
    mostrarRecomendacion(data.data.procesarGuionAgente2.mensaje);
```

```
  }
};

// 2. Manejar respuesta del usuario
const manejarRespuestaUsuario = async (sessionId, respuesta) => {
  const response = await fetch('/graphql', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
      'Authorization': `Bearer ${token}`
    },
    body: JSON.stringify({
      query: `
        mutation Continuar($sessionId: String!, $respuesta: String!) {
          continuarConversacion(
            sessionId: $sessionId
            respuestaUsuario: $respuesta
          ) {
            success
            mensaje
            siguientePaso
          }
        }
      `,
      variables: { sessionId, respuesta }
    })
  });

  const data = await response.json();
  const { siguientePaso, mensaje } = data.data.continuarConversacion;

  switch (siguientePaso) {
    case 'confirmar_compra':
      // Usuario rechazo, se muestra nueva recomendacion
      mostrarRecomendacion(mensaje);
      break;

    case 'solicitar_datos_envio':
      // Usuario aprobo, mostrar formulario
      mostrarFormularioDatosEnvio(mensaje);
      break;

    case 'ir_a_checkout':
      // Datos completos, redirigir
      window.location.href = `/checkout?session=${sessionId}`;
      break;

    case 'nueva_conversacion':
      // Sin alternativas, ofrecer nueva busqueda
      mostrarOpcionNuevaBusqueda(mensaje);
      break;
  }
};
```

Consideraciones Importantes

1. **Persistencia de Session ID**: El frontend debe almacenar el `sessionId` entre llamadas a `continuarConversacion`.
2. **Manejo de Errores**: Siempre verificar el campo `success` antes de procesar la respuesta.
3. **Timeout de Sesion**: Si la sesion expira (30 minutos), el usuario debe iniciar un nuevo flujo con `procesarGuionAgente2`.
4. **Rechazos Multiples**: Si el guion solo tiene 1 producto y el usuario rechaza, se devuelve `nueva_conversacion`. Si tiene 2+ productos, se recomienda el segundo automaticamente.
5. **Estilos de Comunicacion**: El mensaje generado por el LLM varia segun el `estiloComunicacion` (cuencano, juvenil, formal, neutral).

Diagrama de Estados del Frontend



Documentacion generada: Febrero 2026