Manual de Usuario - Sistema de Torneos y Apuestas

Backend API v1.0

20 de febrero de 2026

Índice

1. Introducción

Este documento describe el uso completo del sistema de backend para torneos y apuestas. El sistema permite gestionar torneos deportivos con sesiones diarias, donde los participantes realizan selecciones (macho, hembra, alta, baja, runline) y acumulan puntos.

1.1. Flujo General del Sistema

El flujo del programa es el siguiente:

1. **Registro del Administrador**: El admin se registra en el sistema
2. **Creación del Tournament**: El admin define el tournament con sus reglas
3. **Creación de Sesiones**: Se crean las jornadas/días del tournament
4. **Creación de Eventos**: Se definen los partidos/carreras por sesión
5. **Configuración de Selecciones**: El admin define las opciones de apuesta
6. **Registro de Usuarios**: Los clientes se registran y recargan saldo
7. **Inscripción al Tournament**: Los usuarios se unen al tournament
8. **Envío de Predicciones**: Los participantes hacen sus selecciones por sesión
9. **Liquidación de Eventos**: El admin establece los resultados
10. **Distribución de Premios**: Se reparten los premios a los ganadores

1.2. Roles del Sistema

El sistema cuenta con dos roles:

- **admin**: Acceso completo a todas las funciones de gestión del sistema
- **user**: Usuario regular, puede participar en tournaments y gestionar su wallet

1.3. Usuario Administrador Inicial

Al ejecutar la aplicación por primera vez, se crea automáticamente un usuario administrador:

- **Email**: admin@betsystem.com
- **Password**: Admin123!

1.4. Formato de Respuestas

Todas las respuestas siguen el siguiente formato:

Respuesta Exitosa:

```
{
    "success": true,
    "message": "Mensaje descriptivo",
    "data": { ... }
}
```

Respuesta de Error:

```
{
    "success": false,
    "message": "Mensaje de error",
    "errors": "Detalle del error o null"
}
```

2. Autenticación

2.1. Registro de Usuario

Registra un nuevo usuario en el sistema.
Endpoint: `POST /api/v1/auth/register`

```
{
    "username": "jugador1",
    "email": "jugador1@example.com",
    "password": "SecurePass123"
}
```

Respuesta exitosa (201):

```
{
    "success": true,
    "message": "Usuario registrado exitosamente",
    "data": {
        "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
        "user": "jugador1"
    }
}
```

Error 400 - Datos inválidos:

```
{
    "success": false,
    "message": "Error de validación",
    "errors": "Key: 'RegisterRequest.Email' Error:Field validation for '
    Email' failed on the 'required' tag"
}
```

Error 409 - Usuario ya existe:

```
{
    "success": false,
    "message": "El usuario o email ya existe",
    "errors": null
}
```

2.2. Inicio de Sesión

Autentica a un usuario y devuelve un token JWT.
Endpoint: `POST /api/v1/auth/login`

```
{
    "email": "admin@betsystem.com",
    "password": "Admin123!"
}
```

Respuesta exitosa (200):

```
{
    "success": true,
    "message": "Bienvenido al sistema",
    "data": {
        "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
        "user": {
            "id": 1,
            "username": "admin",
```

```
            "email": "admin@betsystem.com",
            "role": "admin"
        }
    }
}
```

Nota: El token devuelto debe usarse en el header `Authorization: Bearer <token>` *para las rutas protegidas.*

Error 400 - Datos inválidos:

```
{
    "success": false,
    "message": "Datos de entrada inválidos",
    "errors": "Key: 'LoginRequest.Email' Error:Field validation for '
    Email' failed on the 'email' tag"
}
```

Error 401 - Credenciales incorrectas:

```
{
    "success": false,
    "message": "Credenciales incorrectas",
    "errors": null
}
```

Error 403 - Cuenta desactivada:

```
{
    "success": false,
    "message": "Cuenta de usuario desactivada",
    "errors": null
}
```

2.3. Ver Mi Perfil

Obtiene los datos del usuario autenticado.
Endpoint: `GET /api/v1/me` (Protegido)
Headers: `Authorization: Bearer <token>`
Respuesta exitosa (200):

```
{
    "success": true,
    "message": "Perfil del usuario",
    "data": {
        "id": 1,
        "username": "jugador1",
        "email": "jugador1@example.com",
        "role": "user",
        "is_active": true
    }
}
```

Error 401 - No autorizado:

```
{
    "success": false,
    "message": "Se requiere token de autorización",
    "errors": null
}
```

3. Gestión de Torneos

3.1. Crear un Tournament

Crea un nuevo tournament con configuración de sesiones y selecciones. **(Solo Admin)**
Endpoint: `POST /api/v1/admin/tournaments`
Headers: `Authorization: Bearer <token>`

```
{
    "name": "Quiniela Liga BBVA 2026",
    "description": "Torneo de pronosticos de la liga espa\u00f1ola",
    "category": "Futbol",
    "start_date": "2026-02-23T00:00:00Z",
    "end_date": "2026-03-01T23:59:59Z",
    "entry_fee": 10.00,
    "entry_fee_tokens": 0,
    "prize_bonus": 10.00,
    "admin_fee_percent": 10.0,
    "settings": {
        "prize_distribution": [0.70, 0.20, 0.10],
        "selections_per_session": 5,
        "required_selection_types": ["macho", "hembra", "alta", "baja",
   "runline"],
        "total_sessions": 5,
        "horse_racing_points": []
    }
}
```

Explicación de Settings:

- `prize_distribution`: Porcentaje para 1ro (70 %), 2do (20 %), 3er (10 %)
- `selections_per_session`: Cantidad de selecciones por día/sesión
- `required_selection_types`: Tipos obligatorios en orden [macho, hembra, alta, baja, runline]
- `total_sessions`: Cantidad total de jornadas/días del tournament
- `horse_racing_points`: Solo para hipica [puntos_1ro, puntos_2do, puntos_3er]

Respuesta exitosa (201):

```
{
    "success": true,
    "message": "Torneo creado con exito",
    "data": {
        "id": 1,
        "name": "Quiniela Liga BBVA 2026",
        "slug": "quiniela-liga-bbva-2026",
        "status": "open",
        "entry_fee": 10.00,
        "prize_pool": 0,
        "settings": {
            "prize_distribution": [0.7, 0.2, 0.1],
            "selections_per_session": 5,
            "required_selection_types": ["macho", "hembra", "alta", "
   baja", "runline"],
            "total_sessions": 5
```

```
        }
    }
}
```

Error 400 - Datos inválidos:

```
{
    "success": false,
    "message": "Datos inválidos",
    "errors": "Key: 'CreateTournamentRequest.Name' Error:Field
   validation for 'Name' failed on the 'required' tag"
}
```

Error 401 - No autorizado:

```
{
    "success": false,
    "message": "Se requiere token de autorización",
    "errors": null
}
```

Error 403 - No es admin:

```
{
    "success": false,
    "message": "Acceso denegado. Se requiere rol de administrador",
    "errors": null
}
```

3.2. Listar Torneos

Obtiene todos los torneos disponibles.
Endpoint: `GET /api/v1/tournaments`
Respuesta exitosa (200):

```
{
    "success": true,
    "message": "Lista de torneos",
    "data": [
        {
            "id": 1,
            "name": "Quiniela Liga BBVA 2026",
            "category": "Futbol",
            "status": "open",
            "entry_fee": 10.00
        }
    ]
}
```

3.3. Ver Detalle de un Tournament

Obtiene los detalles completos de un tournament.
Endpoint: `GET /api/v1/tournaments/id/1`
Error 404 - Tournament no encontrado:

```
{
    "success": false,
    "message": "Torneo no encontrado por ID",
    "errors": null
}
```

3.4. Ver Leaderboard

Obtiene la tabla de clasificación del tournament.
Endpoint: `GET /api/v1/tournaments/id/1/leaderboard`
Respuesta exitosa (200):


```
{
    "success": true,
    "message": "Tabla de clasificacion",
    "data": [
        {
            "id": 1,
            "user_id": 5,
            "user": {
                "username": "jugador1"
            },
            "total_points": 45,
            "tournament_id": 1
        }
    ]
}
```


3.5. Actualizar Estado del Tournament

Finaliza el tournament y distribuye los premios. **(Solo Admin)**
Endpoint: `PATCH /api/v1/admin/tournaments/1/status`


```
{
    "status": "finished"
}
```


Nota: Al cambiar a "finished", el sistema automáticamente calcula los ganadores y distribuye los premios según prize_distribution.
Estados válidos: open, closed, finished
Respuesta exitosa (200):


```
{
    "success": true,
    "message": "Estado actualizado y premios procesados (si aplica)",
    "data": {
        "id": 1,
        "status": "finished",
        "prize_pool": 100.00,
        "prize_bonus": 10.00
    }
}
```


Error 400 - Estado inválido:


```
{
    "success": false,
    "message": "Datos inválidos",
    "errors": "Key: 'UpdateStatusRequest.Status' Error:Field validation
   for 'Status' failed on the 'oneof' tag"
}
```


Error 404 - Tournament no encontrado:

```
{
    "success": false,
    "message": "Torneo no encontrado",
    "errors": null
}
```

Error 403 - No es admin:

```
{
    "success": false,
    "message": "Acceso denegado. Se requiere rol de administrador",
    "errors": null
}
```

4. Gestión de Sesiones

4.1. Crear una Sesión

Crea una jornada/día dentro de un tournament. **(Solo Admin)**
Endpoint: `POST /api/v1/admin/sessions`

```
{
    "tournament_id": 1,
    "session_number": 1,
    "start_time": "2026-02-23T00:00:00Z",
    "end_time": "2026-02-23T18:00:00Z",
    "description": "Jornada 1 - Lunes"
}
```

Explicación:

- `session_number`: 1 = Lunes, 2 = Martes, etc.
- `start_time`: Cuándo abre la sesión para hacer selecciones
- `end_time`: Hora límite (antes de que inicien los partidos)

Respuesta exitosa (201):

```
{
    "success": true,
    "message": "Sesion creada correctamente",
    "data": {
        "id": 1,
        "tournament_id": 1,
        "session_number": 1,
        "start_time": "2026-02-23T00:00:00Z",
        "end_time": "2026-02-23T18:00:00Z",
        "status": "open"
    }
}
```

Error 400 - Datos inválidos:

```
{
    "success": false,
    "message": "Datos inválidos",
    "errors": "Key: 'CreateSessionRequest.SessionNumber' Error:Field
    validation for 'SessionNumber' failed on the 'min' tag"
}
```

Error 404 - Tournament no encontrado:

```
{
    "success": false,
    "message": "Torneo no encontrado",
    "errors": null
}
```

Error 409 - Sesión ya existe:

```
{
    "success": false,
    "message": "Ya existe la sesión #1 para este torneo",
    "errors": null
}
```

4.2. Listar Sesiones de un Tournament

Obtiene todas las sesiones de un tournament.
Endpoint: `GET /api/v1/tournaments/1/sessions`
Respuesta exitosa (200):


```
{
    "success": true,
    "message": "Sesiones del torneo",
    "data": [
        {
            "id": 1,
            "session_number": 1,
            "start_time": "2026-02-23T00:00:00Z",
            "end_time": "2026-02-23T18:00:00Z",
            "status": "open"
        }
    ]
}
```


4.3. Ver Detalle de Sesión

Obtiene los eventos y selecciones de una sesión.
Endpoint: `GET /api/v1/session-events/1`
Error 404 - Sesión no encontrada:


```
{
    "success": false,
    "message": "Sesión no encontrada",
    "errors": null
}
```


4.4. Cerrar una Sesión

Cierra una sesión para que no se puedan hacer más predicciones. **(Solo Admin)**
Endpoint: `PATCH /api/v1/admin/sessions/1/status`


```
{
    "status": "closed"
}
```


Estados válidos: open, closed, settled
Respuesta exitosa (200):


```
{
    "success": true,
    "message": "Estado de sesión actualizado",
    "data": {
        "id": 1,
        "status": "closed"
    }
}
```


Error 400 - Estado inválido:


```
{
    "success": false,
    "message": "Datos inválidos",
```

```
    "errors": "Key: 'UpdateSessionStatusRequest.Status' Error:Field
    validation for 'Status' failed on the 'oneof' tag"
}
```

5. Gestión de Eventos

5.1. Crear un Evento

Crea un partido/carrera dentro de una sesión. **(Solo Admin)**
Endpoint: `POST /api/v1/admin/events`

```
{
    "tournament_id": 1,
    "session_id": 1,
    "name": "Real Madrid vs Barcelona",
    "order": 1,
    "start_time": "2026-02-23T19:00:00Z"
}
```

Respuesta exitosa (201):

```
{
    "success": true,
    "message": "Evento creado con exito",
    "data": {
        "id": 1,
        "tournament_id": 1,
        "session_id": 1,
        "name": "Real Madrid vs Barcelona",
        "start_time": "2026-02-23T19:00:00Z",
        "status": "scheduled"
    }
}
```

Error 400 - Datos inválidos:

```
{
    "success": false,
    "message": "Datos inválidos",
    "errors": "Key: 'CreateEventRequest.Name' Error:Field validation for
    'Name' failed on the 'required' tag"
}
```

5.2. Asignar Competidores

Asocia equipos/caballos a un evento. **(Solo Admin)**
Endpoint: `POST /api/v1/admin/events/id/1/competitors`

```
{
    "competitors": [
        {"name": "Real Madrid", "assigned_number": 1},
        {"name": "Barcelona", "assigned_number": 2}
    ]
}
```

Respuesta exitosa (200):

```
{
    "success": true,
    "message": "Competidores asignados correctamente",
    "data": null
}
```

Error 400 - Datos inválidos:

```
{
    "success": false,
    "message": "Datos de competidores inválidos",
    "errors": "Key: 'SetCompetitorsRequest.Competitors' Error:Field
   validation for 'Competitors' failed on the 'min' tag"
}
```

5.3. Ver Eventos del Tournament

Obtiene todos los eventos con sus selecciones disponibles.
Endpoint: `GET /api/v1/tournaments/1/events`
Respuesta exitosa (200):

```
{
    "success": true,
    "message": "Eventos y selecciones del torneo",
    "data": [
        {
            "id": 1,
            "name": "Real Madrid vs Barcelona",
            "start_time": "2026-02-23T19:00:00Z",
            "competitors": [...],
            "pickable_selections": [...]
        }
    ]
}
```

6. Gestión de Selecciones

6.1. Crear una Selección

Crea una opción de apuesta para un evento. **(Solo Admin)**
Endpoint: `POST /api/v1/admin/events/selections`
Ejemplo - Macho (Favorito):

```
{
    "event_id": 1,
    "description": "Gana Real Madrid",
    "selection_type": "macho",
    "odds": -120,
    "competitor_id": 1,
    "points_for_win": 10
}
```

Respuesta exitosa (201):

```
{
    "success": true,
    "message": "Seleccion creada correctamente",
    "data": {
        "id": 1,
        "event_id": 1,
        "description": "Gana Real Madrid",
        "selection_type": "macho",
        "odds": -120,
        "points_for_win": 10,
        "status": "pending"
    }
}
```

Error 400 - Datos inválidos:

```
{
    "success": false,
    "message": "Datos inválidos",
    "errors": "Key: 'CreateSelectionRequest.Description' Error:Field
   validation for 'Description' failed on the 'required' tag"
}
```

Error 404 - Evento no encontrado:

```
{
    "success": false,
    "message": "El evento no existe",
    "errors": null
}
```

6.2. Ver Selecciones de un Evento

Obtiene todas las opciones de apuesta de un evento.
Endpoint: `GET /api/v1/events/id/1/selections`
Respuesta exitosa (200):

```
{
    "success": true,
    "message": "Opciones disponibles",
```

```
    "data": [
        {
            "id": 1,
            "description": "Gana Real Madrid",
            "selection_type": "macho",
            "points_for_win": 10
        }
    ]
}
```

7. Gestión de Billeteras

7.1. Recargar Saldo

Permite a un usuario depositar dinero en su wallet.
Endpoint: `POST /api/v1/wallet/deposit` (Protegido)

```
{
    "amount": 100.00,
    "method": "pago_movil",
    "reference_number": "PM123456",
    "bank_name": "Banco de Venezuela"
}
```

Métodos disponibles: "pago_movil", "zelle", "binance", "transferencia", "paypal"
Respuesta exitosa (201):

```
{
    "success": true,
    "message": "Deposito realizado con exito",
    "data": {
        "id": 1,
        "user_id": 5,
        "amount": 100.00,
        "type": "in",
        "method": "pago_movil",
        "status": "pending"
    }
}
```

Error 400 - Monto inválido:

```
{
    "success": false,
    "message": "Datos inválidos",
    "errors": "Key: 'DepositRequest.Amount' Error:Field validation for '
    Amount' failed on the 'gt' tag"
}
```

7.2. Consultar Saldo

Obtiene el balance actual del usuario.
Endpoint: `GET /api/v1/wallet/balance` (Protegido)
Respuesta exitosa (200):

```
{
    "success": true,
    "message": "Balance consultado",
    "data": {
        "user_id": 5,
        "balance": 90.00,
        "frozen_balance": 0,
        "bonus_balance": 0,
        "token_balance": 0,
        "currency": "USD"
    }
}
```

7.3. Historial de Transacciones

Obtiene el historial de transacciones del usuario.
Endpoint: `GET /api/v1/wallet/history` (Protegido)

7.4. Estadísticas del Usuario

Obtiene estadísticas de participación del usuario.
Endpoint: `GET /api/v1/wallet/statistics` (Protegido)

8. Participación en Torneos

8.1. Inscribirse en un Tournament

Un usuario se une a un tournament pagando la inscripción.
Endpoint: `POST /api/v1/tournaments/1/join` (Protegido)

```
{
    "pay_with_tokens": false
}
```

Nota: `pay_with_tokens: true` si desea pagar con tokens en lugar de dinero real.
Respuesta exitosa (201):

```
{
    "success": true,
    "message": "Inscripcion exitosa",
    "data": {
        "id": 1,
        "user_id": 5,
        "tournament_id": 1,
        "total_points": 0
    }
}
```

Error 403 - No inscrito en el tournament:

```
{
    "success": false,
    "message": "No estás inscrito en este torneo",
    "errors": null
}
```

Error 400 - Tournament cerrado:

```
{
    "success": false,
    "message": "El torneo no está abierto para inscripciones",
    "errors": null
}
```

Error 400 - Saldo insuficiente:

```
{
    "success": false,
    "message": "Saldo insuficiente",
    "errors": null
}
```

Error 409 - Ya inscrito:

```
{
    "success": false,
    "message": "Ya estás inscrito en este torneo",
    "errors": null
}
```

8.2. Enviar Predicciones (Por Sesión)

El participante envía sus selecciones para una sesión específica.
Endpoint: `POST /api/v1/tournaments/1/sessions/picks` (Protegido)

```
{
    "session_id": 1,
    "selection_ids": [1, 2, 3, 4, 5]
}
```

Respuesta exitosa (201):

```
{
    "success": true,
    "message": "Predicciones guardadas para la sesion",
    "data": [
        {
            "id": 1,
            "participant_id": 1,
            "selection_id": 1,
            "session_id": 1,
            "status": "pending",
            "awarded_points": 0
        }
    ]
}
```

Error 400 - Cantidad incorrecta de selecciones:

```
{
    "success": false,
    "message": "Debe hacer exactamente 5 selecciones por sesión",
    "errors": null
}
```

Error 400 - Sesión cerrada:

```
{
    "success": false,
    "message": "La sesión no está abierta para predicciones",
    "errors": null
}
```

Error 400 - Hora límite excedida:

```
{
    "success": false,
    "message": "Ya cerró la hora límite para hacer selecciones en esta
    sesión",
    "errors": null
}
```

Error 400 - Tipo de selección incorrecto:

```
{
    "success": false,
    "message": "La selección #1 debe ser de tipo 'macho' (posición 1)",
    "errors": null
}
```

Error 403 - No inscrito:

```
{
    "success": false,
    "message": "No estás inscrito en este torneo",
    "errors": null
}
```

8.3. Ver Mis Predicciones de una Sesión

El usuario puede ver sus predicciones enviadas.
Endpoint: `GET /api/v1/my-sessions/1/picks` (Protegido)
Respuesta exitosa (200):


```
{
    "success": true,
    "message": "Tus predicciones en esta sesion",
    "data": [
        {
            "id": 1,
            "selection_id": 1,
            "session_id": 1,
            "status": "won",
            "awarded_points": 10,
            "selection": {
                "description": "Gana Real Madrid",
                "selection_type": "macho"
            }
        }
    ]
}
```


8.4. Ver Eventos del Tournament

El usuario ve la cartelera completa para hacer sus picks.
Endpoint: `GET /api/v1/tournaments/1/events`

9. Liquidación de Eventos

9.1. Establecer Resultados

El admin establece los resultados de un evento. **(Solo Admin)**
Endpoint: `POST /api/v1/admin/events/id/1/settle`
Para deportes de equipo:

```
{
    "results": [
        {"competitor_id": 1, "final_score": 3, "position": 0},
        {"competitor_id": 2, "final_score": 1, "position": 0}
    ]
}
```

Para carreras de caballos:

```
{
    "results": [
        {"competitor_id": 1, "final_score": 0, "position": 1},
        {"competitor_id": 2, "final_score": 0, "position": 2},
        {"competitor_id": 3, "final_score": 0, "position": 3}
    ]
}
```

Respuesta exitosa (200):

```
{
    "success": true,
    "message": "Evento liquidado y puntos asignados correctamente",
    "data": null
}
```

Error 400 - Evento ya liquidado:

```
{
    "success": false,
    "message": "Este evento ya ha sido liquidado",
    "errors": null
}
```

Error 404 - Evento no encontrado:

```
{
    "success": false,
    "message": "Evento no encontrado",
    "errors": null
}
```

9.2. Finalizar Tournament y Pagar Premios

El admin termina el tournament y se distribuyen los premios.
Endpoint: `PATCH /api/v1/admin/tournaments/1/status`

```
{
    "status": "finished"
}
```

Respuesta exitosa (200):

```
{
    "success": true,
    "message": "Estado actualizado y premios procesados (si aplica)",
    "data": {
        "id": 1,
        "status": "finished",
        "prize_pool": 100.00,
        "prize_bonus": 10.00
    }
}
```

10. Gestión de Usuarios (Admin)

10.1. Listar Usuarios

Obtiene todos los usuarios del sistema. **(Solo Admin)**
Endpoint: `GET /api/v1/admin/users`
Respuesta exitosa (200):

```
{
    "success": true,
    "message": "Lista de usuarios",
    "data": [
        {
            "id": 1,
            "username": "admin",
            "email": "admin@betsystem.com",
            "role": "admin",
            "is_active": true
        }
    ]
}
```

Error 403 - No es admin:

```
{
    "success": false,
    "message": "Acceso denegado. Se requiere rol de administrador",
    "errors": null
}
```

10.2. Ver Usuario por ID

Obtiene los detalles de un usuario específico. **(Solo Admin)**
Endpoint: `GET /api/v1/admin/users/1`
Error 404 - Usuario no encontrado:

```
{
    "success": false,
    "message": "Usuario no encontrado",
    "errors": null
}
```

10.3. Actualizar Rol de Usuario

Cambia el rol de un usuario. **(Solo Admin)**
Endpoint: `PATCH /api/v1/admin/users/1/role`

```
{
    "role": "admin"
}
```

Roles válidos: user, admin
Respuesta exitosa (200):

```
{
    "success": true,
    "message": "Rol actualizado correctamente",
```

```
    "data": {
        "id": 2,
        "username": "jugador1",
        "role": "admin"
    }
}
```

Error 400 - Rol inválido:

```
{
    "success": false,
    "message": "Datos inválidos",
    "errors": "Key: 'UpdateUserRoleRequest.Role' Error:Field validation for 'Role' failed on the 'oneof' tag"
}
```

Error 404 - Usuario no encontrado:

```
{
    "success": false,
    "message": "Usuario no encontrado",
    "errors": null
}
```

Error 403 - No puede degradarse a sí mismo:

```
{
    "success": false,
    "message": "No puede degradar su propio rol de administrador",
    "errors": null
}
```

10.4. Actualizar Estado de Usuario

Activa o desactiva un usuario. **(Solo Admin)**
Endpoint: `PATCH /api/v1/admin/users/1/status`

```
{
    "is_active": false
}
```

Error 403 - No puede desactivarse a sí mismo:

```
{
    "success": false,
    "message": "No puede desactivarse a sí mismo",
    "errors": null
}
```

11. Códigos de Estado HTTP

Código	Estado	Descripción
200	OK	La solicitud fue exitosa
201	Created	Recurso creado exitosamente
400	Bad Request	Datos inválidos o mal formados
401	Unauthorized	Token no proporcionado o inválido
403	Forbidden	No tiene permisos para esta acción
404	Not Found	Recurso no encontrado
409	Conflict	Conflicto (ej: ya inscrito)
500	Internal Server Error	Error del servidor

12. Endpoints Resumen

12.1. Rutas Públicas

Endpoint	Descripción
POST /api/v1/auth/register	Registrar usuario
POST /api/v1/auth/login	Iniciar sesión
GET /api/v1/tournaments	Listar torneos
GET /api/v1/tournaments/id/:id	Ver torneo
GET /api/v1/tournaments/s/:slug	Ver torneo por slug
GET /api/v1/tournaments/id/:id/leaderboard	Leaderboard
GET /api/v1/tournaments/id/:id/events	Eventos del torneo
GET /api/v1/tournaments/id/:id/sessions	Sesiones del torneo
GET /api/v1/session-events/:id	Detalle de sesión
GET /api/v1/events/id/:id	Ver evento
GET /api/v1/events/s/:slug	Ver evento por slug
GET /api/v1/events/id/:id/selections	Selecciones del evento

12.2. Rutas de Usuario (Requiere Auth)

Endpoint	Descripción
GET /api/v1/me	Mi perfil
POST /api/v1/tournaments/:id/join	Inscribirse a un torneo
POST /api/v1/tournaments/:id/sessions/picks	Enviar pronósticos
GET /api/v1/my-sessions/:session_id/picks	Ver mis pronósticos
GET /api/v1/wallet/balance	Consultar saldo
POST /api/v1/wallet/deposit	Recargar saldo
GET /api/v1/wallet/history	Historial transacciones
GET /api/v1/wallet/statistics	Estadísticas usuario

12.3. Rutas de Administrador (Requiere Auth + Rol Admin)

Endpoint	Descripción
GET /api/v1/admin/users/	Listar usuarios
GET /api/v1/admin/users/:id	Detalle de usuario
PATCH /api/v1/admin/users/:id/role	Cambiar rol
PATCH /api/v1/admin/users/:id/status	Banear/Activar
POST /api/v1/admin/tournaments/	Crear torneo
PATCH /api/v1/admin/tournaments/:id/status	Estado del torneo
POST /api/v1/admin/sessions/	Crear sesión
PATCH /api/v1/admin/sessions/:id/status	Estado de la sesión
POST /api/v1/admin/events/	Crear evento
POST /api/v1/admin/events/selections	Crear selección
POST /api/v1/admin/events/:id/competitors	Competidores
POST /api/v1/admin/events/:id/settle	Liquidar evento

13. Pruebas del Sistema

El proyecto incluye un conjunto de pruebas automatizadas que verifican el correcto funcionamiento del backend. Las pruebas utilizan una base de datos SQLite en memoria para ejecutar tests sin modificar la base de datos de producción.

13.1. Ejecutar las Pruebas

Para ejecutar todas las pruebas del proyecto:

```
# Ejecutar todas las pruebas
go test ./tests/... -v

# Ejecutar solo pruebas de autenticación
go test ./tests/... -v -run TestRegister

# Ejecutar solo pruebas de autenticación de login
go test ./tests/... -v -run TestLogin
```

13.2. Tests de Autenticación

Los tests de autenticación verifican los siguientes escenarios:

1. **Registro de Usuario**
   - Registro exitoso con datos válidos
   - Registro con email ya existente
   - Registro con username duplicado
   - Registro con email inválido
   - Registro con contraseña muy corta

2. **Inicio de Sesión**
    - Login con credenciales correctas
    - Login con contraseña incorrecta
    - Login con usuario inexistente
3. **Protección de Rutas**
    - Rutas públicas accesibles sin autenticación
    - Rutas protegidas requieren token JWT
    - Rutas de admin requieren rol de administrador

13.3. Resultado Esperado

Al ejecutar las pruebas, debería ver una salida similar a:

```
=== RUN   TestRegister_Success
--- PASS: TestRegister_Success (1.08s)
=== RUN   TestRegister_EmailAlreadyExists
--- PASS: TestRegister_EmailAlreadyExists (2.15s)
=== RUN   TestLogin_Success
--- PASS: TestLogin_Success (2.23s)
=== RUN   TestLogin_InvalidCredentials
--- PASS: TestLogin_InvalidCredentials (2.15s)
=== RUN   TestPublicRoutes_Accessible
--- PASS: TestPublicRoutes_Accessible (0.01s)
=== RUN   TestProtectedRoutes_RequireAuth
--- PASS: TestProtectedRoutes_RequireAuth (0.01s)
=== RUN   TestRegister_InvalidEmail
--- PASS: TestRegister_InvalidEmail (0.00s)
=== RUN   TestRegister_ShortPassword
--- PASS: TestRegister_ShortPassword (0.01s)
=== RUN   TestRegister_DuplicateUsername
--- PASS: TestRegister_DuplicateUsername (2.23s)
=== RUN   TestLogin_NonExistentUser
--- PASS: TestLogin_NonExistentUser (0.01s)
=== RUN   TestAdminRoutes_RequireAdmin
--- PASS: TestAdminRoutes_RequireAdmin (1.09s)
PASS
ok   github.com/cesarbmathec/bets-backend/tests 11.348s
```

13.4. Agregar Nuevas Pruebas

Para agregar nuevas pruebas, cree archivos en el directorio `tests/` con el sufijo `_test.go`. El archivo `tests/helpers_test.go` contiene funciones helper para configurar la base de datos de pruebas y hacer requests HTTP.

```
package tests

import (
    "testing"
    "github.com/cesarbmathec/bets-backend/config"
    "github.com/cesarbmathec/bets-backend/models"
)

func TestNuevoEscenario(t *testing.T) {
    // Setup - crear base de datos en memoria
    SetupTestDB(t)
    router := SetupRouter()

    // Arrange - preparar datos de prueba

    // Act - ejecutar la operación
    w := MakeJSONRequest(router, "POST", "/api/v1/endpoint", data)

    // Assert - verificar resultados
    assert.Equal(t, http.StatusOK, w.Code)
}
```

14. Nuevas Funcionalidades

14.1. Login con Email o Username

El sistema permite autenticarse usando email O username:

```
{"email": "user@example.com", "password": "password123"}
{"username": "user1", "password": "password123"}
```

14.2. Campo Nickname

El modelo de usuario incluye un campo `nickname` que se muestra en las tablas y leaderboards del frontend.

14.3. Metodos de Pago/Retiro

El sistema permite registrar multiples metodos de pago para retiros:

- Pago Movil (telefono, banco, cuenta)
- Zelle (email, nombre)
- Binance/USDT (direccion, red, email)
- PayPal (email)
- Transferencia Bancaria (numero de cuenta, CLABE, SWIFT)

14.4. Rutas de Metodos de Pago

Endpoint	**Descripcion**
GET /api/v1/payment-methods	Listar metodos de pago
POST /api/v1/payment-methods	Agregar metodo de pago
DELETE /api/v1/payment-methods/:id	Eliminar metodo de pago