

Last indexed: 27 August 2025 (c12f7a)

PadelFlow Overview

Core Application Architecture

Server Setup and Configuration

User Roles and Authentication

Tournament Management
Features

Real-time Features

Database Layer

SQLite Database Management

Database Extensions

Qt Framework Components

Image Format Support

Development Environment

IDE Configuration

Debugging Setup

Project Configuration

User Roles and Authentication

Relevant source files

Purpose and Scope

This document describes the user role hierarchy and authentication system in PadelFlow. It covers the four distinct user types, their permissions, the session-based authentication mechanism, and the middleware components that enforce access control throughout the application.

For information about the main server configuration and routing setup, see [Server Setup and Configuration](#). For details about tournament management workflows that different user roles can access, see [Tournament Management Features](#).

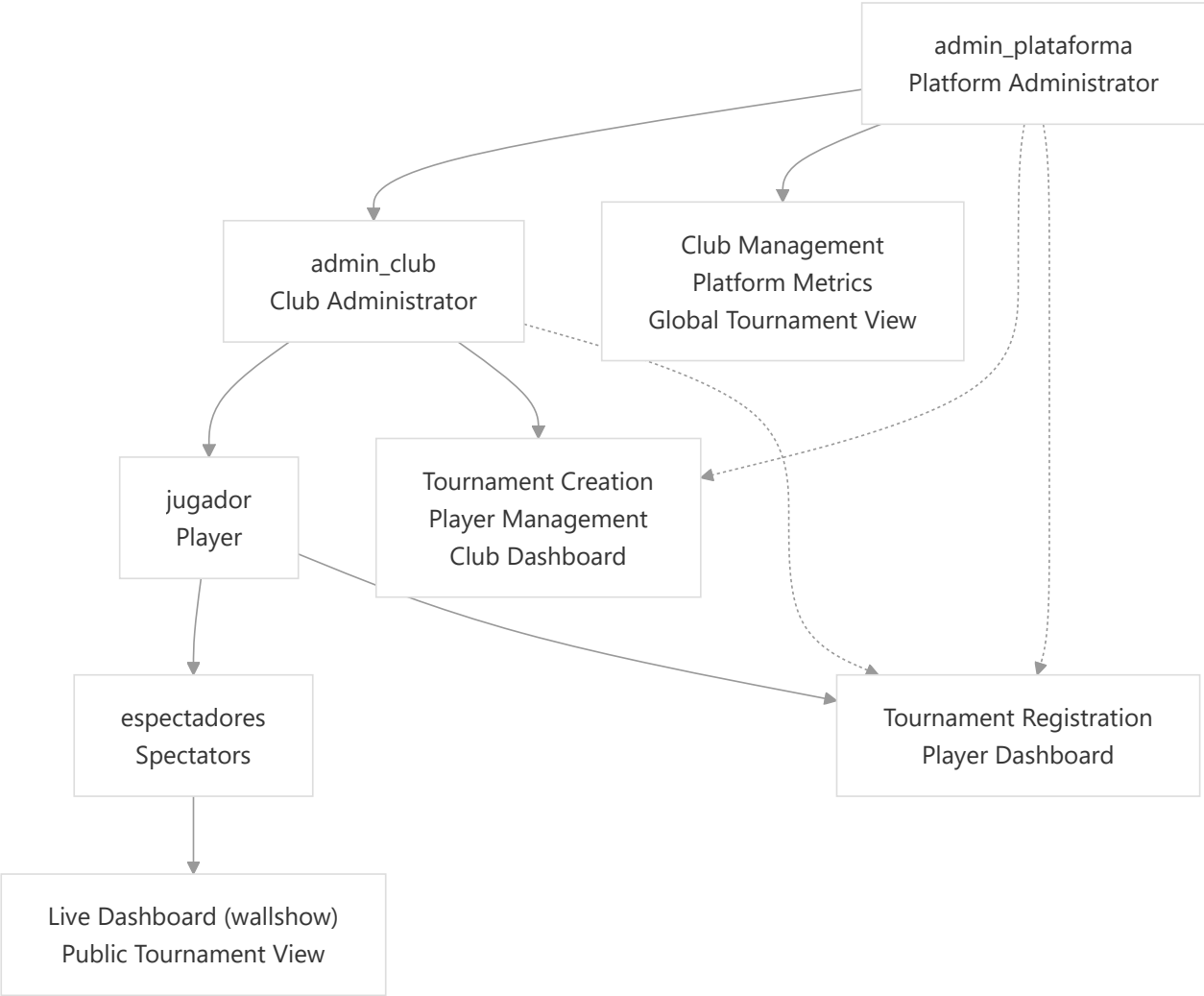
User Role Hierarchy

PadelFlow implements a four-tier user role system with clearly defined permissions and access levels:

Role	Spanish Name	Access Level	Primary Functions
Platform Administrator	admin_plataforma	System-wide	Manage clubs, view platform metrics, oversee all tournaments

Role	Spanish Name	Access Level	Primary Functions
Club Administrator	admin_club	Club-specific	Create tournaments, manage club players, add players to events
Player	jugador	Tournament participation	Register for tournaments, view tournament status
Spectator	espectadores	Public read-only	View live dashboard (wallshow) without registration

User Role Hierarchy Diagram

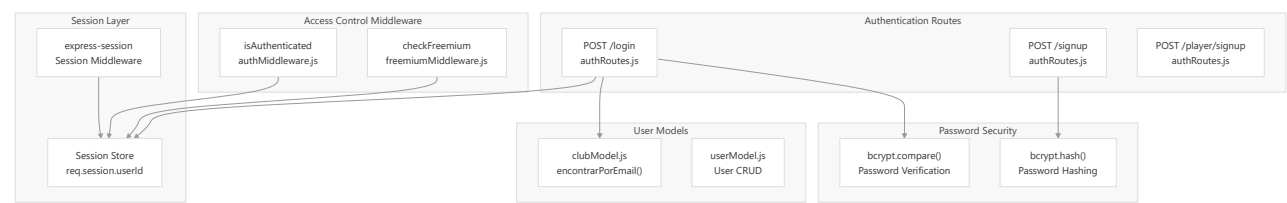


Sources: README.md | 6-11 README.md | 184-237

Authentication System Architecture

The authentication system is built on Express.js sessions with bcrypt password hashing. The architecture separates authentication logic into dedicated route handlers and middleware components.

Authentication Component Architecture



Sources: [app.js](#) | 46-52 | [README.md](#) | 102-125

Session Configuration

The application uses `express-session` for maintaining user authentication state across HTTP requests.

Session Middleware Setup

The session is configured in the main application file with the following characteristics:

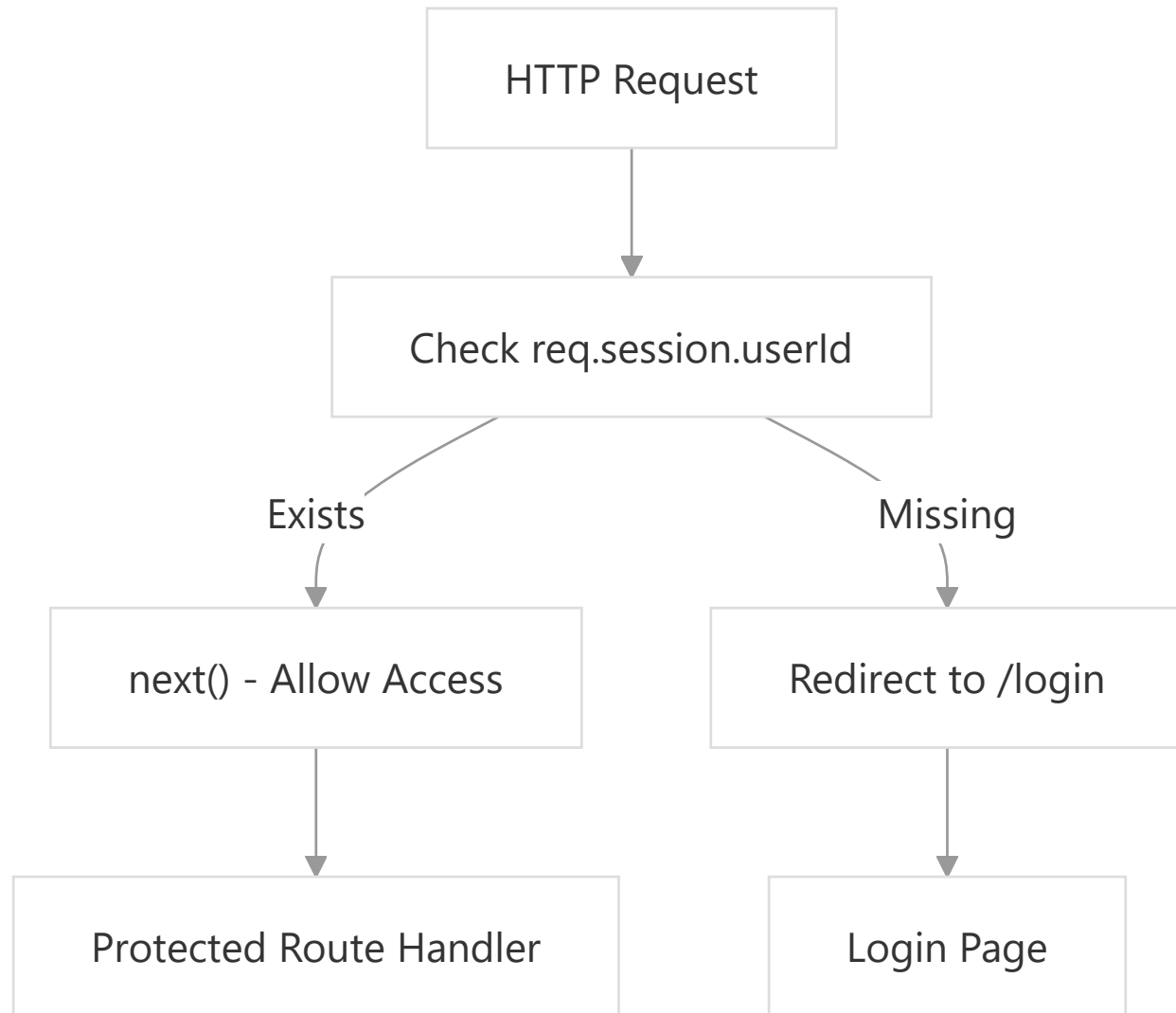
- **Secret:** Currently uses a hardcoded value `'tu_secreto'` (should be moved to environment variables)
- **Storage:** In-memory storage (default)
- **Security:** `secure: false` for development (should be `true` in production with HTTPS)
- **Session Data:** Stores `userId` to identify authenticated users

Sources:  app.js | 46-52

Authentication Middleware Components

isAuthenticated Middleware

The `isAuthenticated` function protects routes that require user login:



This middleware is applied to all tournament management routes and administrative functions.

Sources:  README.md | 116-119

checkFreemium Middleware

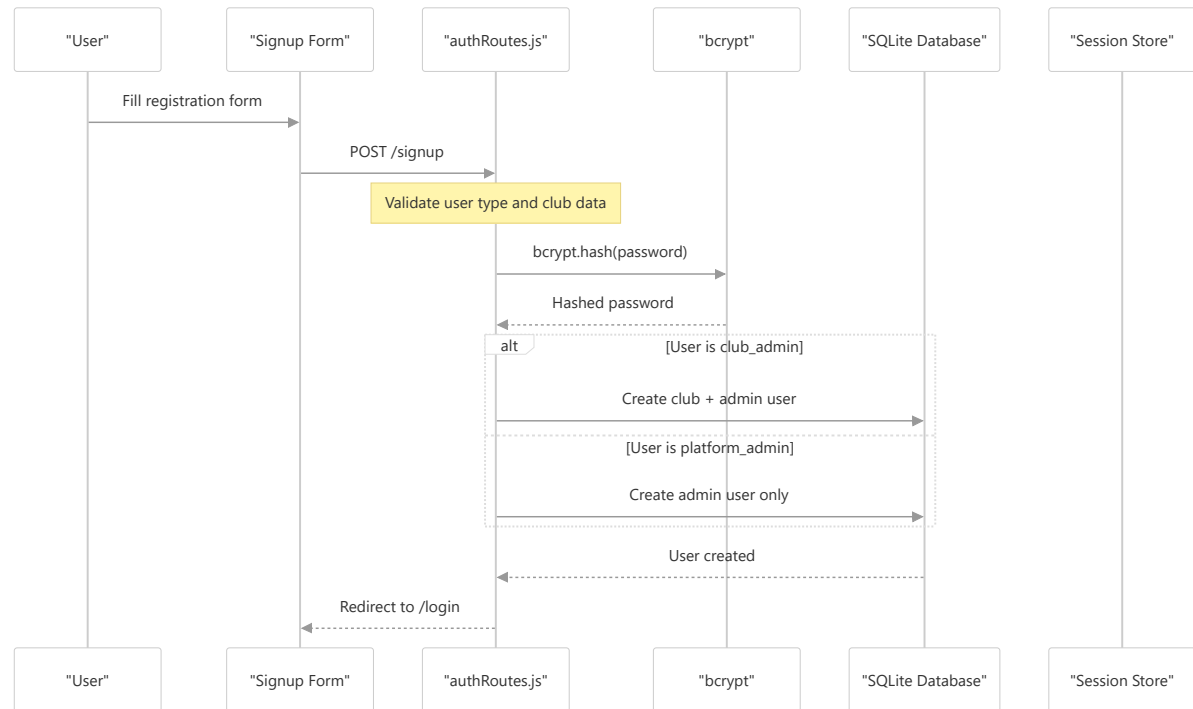
The `checkFreemium` middleware enforces subscription limits for club administrators:

- Retrieves the club ID from the user session
- Counts active tournaments for the club
- Prevents creation of more than 3 tournaments for freemium clubs
- Redirects with error message if limit exceeded

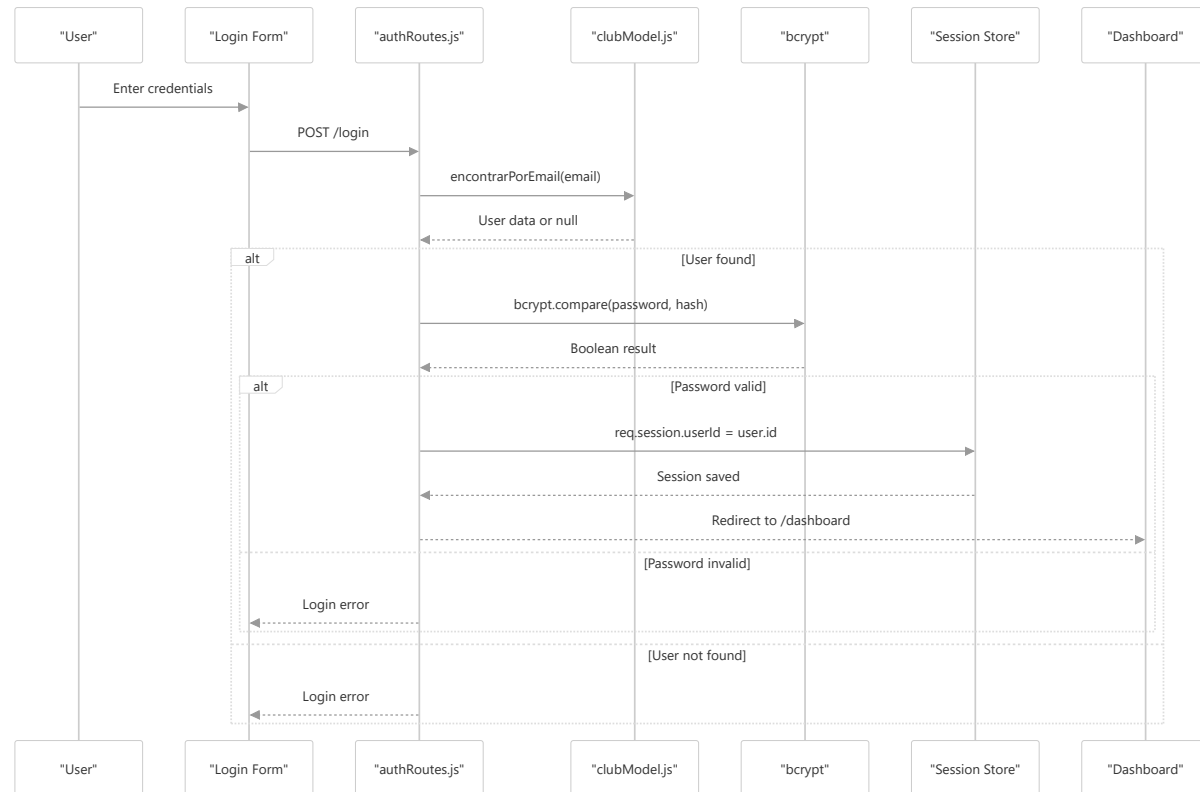
Sources:  README.md | 120-124

Registration and Login Flow

User Registration Process



Login Authentication Flow



Sources: README.md | 103-115

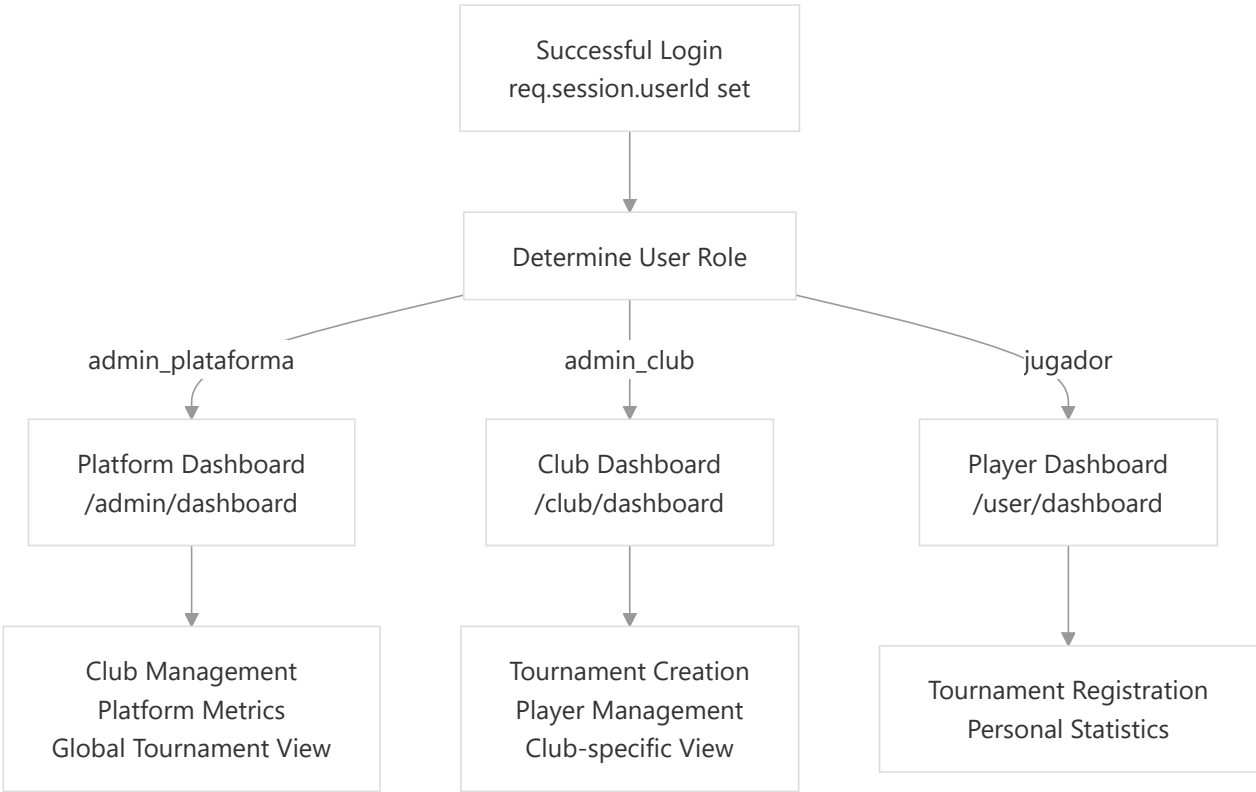
Role-based Access Control Implementation



Route Protection Strategy

Different route groups are protected with appropriate middleware based on the required access level:

Route Group	Middleware Applied	Accessible To
/admin/*	isAuthenticated	Platform Administrators
/club/*	isAuthenticated	Club Administrators
/tournaments/create	isAuthenticated , checkFreemium	Club Administrators (with limits)
Public routes	None	All users including spectators

Dashboard Routing by Role



Sources:  app.js | 62-66  README.md | 130-131

Authentication Route Structure

The authentication system is organized into specific route handlers:

Core Authentication Routes

- `GET /login` : Renders the login form
- `GET /signup` : Renders the registration form
- `POST /login` : Processes login credentials and establishes session
- `POST /signup` : Creates new users with role-specific logic
- `POST /player/signup` : Handles player registration by club administrators

Registration Type Handling

The signup process differentiates between user types:

- **Club Admin Registration:** Creates both a new club and admin user
- **Platform Admin Registration:** Creates admin user without club association
- **Player Registration:** Requires club association and is typically initiated by club administrators

Sources:  README.md | 102-115  app.js | 10-15

