

[DOT & DOT] Diseño del Backend

Idea

El backend desarrollado actúa como árbitro central de las comunicaciones del ecosistema de aplicaciones. Esta aplicación se encarga de:

Responsabilidades externas:

1. Registrar y autorizar usuarios mediante sistema de verificación por email.
2. Implementar control de acceso basado en roles (ROLE_ADMIN, ROLE_REFeree, ROLE_USER).
3. Gestionar el almacenamiento, modificación y visualización de partidos en tiempo real.
4. Validar dispositivos y sesiones mediante tokens JWT con refresh tokens.
5. Proporcionar endpoints públicos para consulta de información sin autenticación.
6. Comunicación en tiempo real mediante WebSockets (STOMP) para actualización de partidos.

Responsabilidades internas:

1. Persistencia en base de datos MySQL con JPA/Hibernate.
2. Verificación de integridad referencial y validaciones a nivel de entidad.
3. Control de accesos mediante Spring Security con @PreAuthorize.
4. Gestión de imágenes con almacenamiento en servidor.
5. Servicio de correo electrónico para verificaciones y notificaciones.
6. Broadcasting de eventos de partidos mediante WebSocket Service.

Estructura del Sistema

El backend se organiza en los siguientes módulos principales:

1. **Admin** - Gestión completa de entidades (CRUD) con rol ROLE_ADMIN
2. **Referee** - Endpoints específicos para árbitros durante partidos
3. **Open/Public** - Información pública accesible sin autenticación
4. **Auth** - Autenticación, registro, verificación y gestión de roles
5. **WebSocket** - Comunicación bidireccional en tiempo real
6. **Images** - Servicio de almacenamiento y recuperación de imágenes
7. **Mail** - Envío de correos de verificación y contacto

Modelo de Datos: Partidos

El sistema implementa una estructura completa que digitaliza el acta real de voleibol:

| Nombre de la Competición: | | | | | | | | | | REAL FEDERACION ESPAÑOLA DE VOLEIBOL | | | | | | | | | | ACTA DE PARTIDO | | | | | | | | | |
|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|---------------------------------|--|--|--|--|--|--|--|--|--|
| Ciudad: _____ Fase: _____ Partido Nº: _____ | | | | | | | | | | Código: _____ Fecha: _____ Hora: _____ | | | | | | | | | | EQUIPOS: _____ | | | | | | | | | |
| División: _____ Masc: _____ Femen: _____ Categoría: Sen Juv Cad Int Ale | | | | | | | | | | EQUIPO A: _____ EQUIPO B: _____ | | | | | | | | | | EQUIPO A: _____ EQUIPO B: _____ | | | | | | | | | |
| SET 1 INICIO: EQUIPO A PUNTOS EQUIPO B FINAL: PUNTOS I II III IV V VI 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 1040 1041 1042 1043 1044 1045 1046 1047 1048 1049 1050 1051 1052 1053 1054 1055 1056 1057 1058 1059 1060 1061 1062 1063 1064 1065 1066 1067 1068 1069 1070 1071 1072 1073 1074 1075 1076 1077 1078 1079 1080 1081 1082 1083 1084 1085 1086 1087 1088 1089 1090 1091 1092 1093 1094 1095 1096 1097 1098 1099 1100 1101 1102 1103 1104 1105 1106 1107 1108 1109 1110 1111 1112 1113 1114 1115 1116 1117 1118 1119 1120 1121 1122 1123 1124 1125 1126 1127 1128 1129 1130 1131 1132 1133 1134 1135 1136 1137 1138 1139 1140 1141 1142 1143 1144 1145 1146 1147 1148 1149 1150 1151 1152 1153 1154 1155 1156 1157 1158 1159 1160 1161 1162 1163 1164 1165 1166 1167 1168 1169 1170 1171 1172 1173 1174 1175 1176 1177 1178 1179 1180 1181 1182 1183 1184 1185 1186 1187 1188 1189 1190 1191 1192 1193 1194 1195 1196 1197 1198 1199 1200 1201 1202 1203 1204 1205 1206 1207 1208 1209 1210 1211 1212 1213 1214 1215 1216 1217 1218 1219 1220 1221 1222 1223 1224 1225 1226 1227 1228 1229 1230 1231 1232 1233 1234 1235 1236 1237 1238 1239 1240 1241 1242 1243 1244 1245 1246 1247 1248 1249 1250 1251 1252 1253 1254 1255 1256 1257 1258 1259 1260 1261 1262 1263 1264 1265 1266 1267 1268 1269 1270 1271 1272 1273 1274 1275 1276 1277 1278 1279 1280 1281 1282 1283 1284 1285 1286 1287 1288 1289 1290 1291 1292 1293 1294 1295 1296 1297 1298 1299 1300 1301 1302 1303 1304 1305 1306 1307 1308 1309 1310 1311 1312 1313 1314 1315 1316 1317 1318 1319 1320 1321 1322 1323 1324 1325 1326 1327 1328 1329 1330 1331 1332 1333 1334 1335 1336 1337 1338 1339 1340 1341 1342 1343 1344 1345 1346 1347 1348 1349 1350 1351 1352 1353 1354 1355 1356 1357 1358 1359 1360 1361 1362 1363 1364 1365 1366 1367 1368 1369 1370 1371 1372 1373 1374 1375 1376 1377 1378 1379 1380 1381 1382 1383 1384 1385 1386 1387 1388 1389 1390 1391 1392 1393 1394 1395 1396 1397 1398 1399 1400 1401 1402 1403 1404 1405 1406 1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417 1418 1419 1420 1421 1422 1423 1424 1425 1426 1427 1428 1429 1430 1431 1432 1433 1434 1435 1436 1437 1438 1439 1440 1441 1442 1443 1444 1445 1446 1447 1448 1449 1450 1451 1452 1453 1454 1455 1456 1457 1458 1459 1460 1461 1462 1463 1464 1465 1466 1467 1468 1469 1470 1471 1472 1473 1474 1475 1476 1477 1478 1479 1480 1481 1482 1483 1484 1485 1486 1487 1488 1489 1490 1491 1492 1493 1494 1495 1496 1497 1498 1499 1500 1501 1502 1503 1504 1505 1506 1507 1508 1509 1510 1511 1512 1513 1514 1515 1516 1517 1518 1519 1520 1521 1522 1523 1524 1525 1526 1527 1528 1529 1530 1531 1532 1533 1534 1535 1536 1537 1538 1539 1540 1541 1542 1543 1544 1545 1546 1547 1548 1549 1550 1551 1552 1553 1554 1555 1556 1557 1558 1559 1560 1561 1562 1563 1564 1565 1566 1567 1568 1569 1570 1571 1572 1573 1574 1575 1576 1577 1578 1579 1580 1581 1582 1583 1584 1585 1586 1587 1588 1589 1590 1591 1592 1593 1594 1595 1596 1597 1598 1599 1600 1601 1602 1603 1604 1605 1606 1607 1608 1609 1610 1611 1612 1613 1614 1615 1616 1617 1618 1619 1620 1621 1622 1623 1624 1625 1626 1627 1628 1629 1630 1631 1632 1633 1634 1635 1636 1637 1638 1639 1640 1641 1642 1643 1644 1645 1646 1647 1648 1649 1650 1651 1652 1653 1654 1655 1656 1657 1658 1659 1660 1661 1662 1663 1664 1665 1666 1667 1668 1669 1670 1671 1672 1673 1674 1675 1676 1677 1678 1679 1680 1681 1682 1683 1684 1685 1686 1687 1688 1689 1690 1691 1692 1693 1694 1695 1696 1697 1698 1699 1700 1701 1702 1703 1704 1705 1706 1707 1708 1709 1710 1711 1712 1713 1714 1715 1716 1717 1718 1719 1720 1721 1722 1723 1724 1725 1726 1727 1728 1729 1730 1731 1732 1733 1734 1735 1736 1737 1738 1739 1740 1741 1742 1743 1744 1745 1746 1747 1748 1749 1750 1751 1752 1753 1754 1755 1756 1757 1758 1759 1760 1761 1762 1763 1764 1765 1766 1767 1768 1769 1770 1771 1772 1773 1774 1775 1776 1777 1778 1779 1780 1781 1782 1783 1784 1785 1786 1787 1788 1789 1790 1791 1792 1793 1794 1795 1796 1797 1798 1799 1800 1801 1802 1803 1804 1805 1806 1807 1808 1809 1810 1811 1812 1813 1814 1815 1816 1817 1818 1819 1820 1821 1822 1823 1824 1825 1826 1827 1828 1829 1830 1831 1832 1833 1834 1835 1836 1837 1838 1839 1840 1841 1842 1843 1844 1845 1846 1847 1848 1849 1850 1851 1852 1853 1854 1855 1856 1857 1858 1859 1860 1861 1862 1863 1864 1865 1866 1867 1868 1869 1870 1871 1872 1873 1874 1875 1876 1877 1878 1879 1880 1881 1882 1883 1884 1885 1886 1887 1888 1889 1890 1891 1892 1893 1894 1895 1896 1897 1898 1899 1900 1901 1902 1903 1904 1905 1906 1907 1908 1909 1910 1911 1912 1913 1914 1915 1916 1917 1918 1919 1920 1921 1922 1923 1924 1925 1926 1927 1928 1929 1930 1931 1932 1933 1934 1935 1936 1937 1938 1939 1940 1941 1942 1943 1944 1945 1946 1947 1948 1949 1950 1951 1952 1953 1954 1955 1956 1957 1958 1959 1960 1961 1962 1963 1964 1965 1966 1967 1968 1969 1970 1971 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024 2025 2026 2027 2028 2029 2030 2031 2032 2033 2034 2035 2036 2037 2038 2039 2040 2041 2042 2043 2044 2045 2046 2047 2048 2049 2050 2051 2052 2053 2054 2055 2056 2057 2058 2059 2060 2061 2062 2063 2064 2065 2066 2067 2068 2069 2070 2071 2072 2073 2074 2075 2076 2077 2078 2079 2080 2081 2082 2083 2084 2085 2086 2087 2088 2089 2090 2091 2092 2093 2094 2095 2096 2097 2098 2099 2100 2101 2102 2103 2104 2105 2106 2107 2108 2109 2110 2111 2112 2113 2114 2115 2116 2117 2118 2119 2120 2121 2122 2123 2124 2125 2126 2127 2128 2129 2130 2131 2132 2133 2134 2135 2136 2137 2138 2139 2140 2141 2142 2143 2144 2145 2146 2147 2148 2149 2150 2151 2152 2153 2154 2155 2156 2157 2158 2159 2160 2161 2162 2163 2164 2165 2166 2167 2168 2169 2170 2171 2172 2173 2174 2175 2176 2177 2178 2179 2180 2181 2182 2183 2184 2185 2186 2187 2188 2189 2190 2191 2192 2193 2194 2195 2196 2197 2198 2199 2200 2201 2202 2203 2204 2205 2206 2207 2208 2209 2210 2211 2212 2213 2214 2215 2216 2217 2218 2219 2220 2221 2222 2223 2224 2225 2226 2227 2228 2229 2230 2231 2232 2233 2234 2235 2236 2237 2238 2239 2240 2241 2242 2243 2244 2245 2246 2247 2248 2249 2250 2251 2252 2253 2254 2255 2256 2257 2258 2259 2260 2261 2262 2263 2264 2265 2266 2267 2268 2269 2270 2271 2272 2273 2274 2275 2276 2277 2278 2279 2280 2281 2282 2283 2284 2285 2286 2287 2288 2289 2290 2291 2292 2293 2294 2295 2296 2297 2298 2299 2300 2301 2302 2303 2304 2305 2306 2307 2308 2309 2310 2311 2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

GameDetails (Detalles del Partido)

```
@Entity
public class GameDetails {
    private long id;
    private Game game;
    private Category category;        // INFANTIL, CADETE, JUVENIL, SENIOR
    private Division division;        // PRIMERA, SEGUNDA, TERCERA
    private Competition competition;  // Competición (Liga regular, Copa, etc.)
    private City city;                // Ciudad donde se juega
    private Date date;                // Fecha del partido
    private Date timeStart;           // Hora de inicio real
}
```

GameSet (Set del Partido)

```
@Entity
public class GameSet {
    private long id;
    private String uniqueCode;        // Referencia al partido
    private int setNumber;            // Número del set (1-5)
    private int pointsLocal;          // Puntos equipo local
    private int pointsVisit;          // Puntos equipo visitante
    private Date timeStart;           // Hora de inicio del set
    private Date timeEnd;             // Hora de fin del set
    private String localAlignment;    // Alineación equipo local (JSON)
    private String visitAlignment;    // Alineación equipo visitante (JSON)
}
```

GameInitialSituation (Situación Inicial)

```
@Entity
public class GameInitialSituation {
    private long id;
    private Game game;
    private Team localTeam;           // Equipo local
    private Team visitTeam;           // Equipo visitante
    private Team servingTeam;         // Equipo que saca
    private Team receivingTeam;       // Equipo que recibe
    private Team leftTeam;            // Equipo situado a la izquierda
    private Team rightTeam;           // Equipo situado a la derecha
}
```

GameSanctions (Sanciones del Partido)

```
@Entity
public class GameSanctions {
    private long id;
    private Game game;
    private SanctionType type;           // YELLOW, RED, EXPULSION, DISQUALIFICATION
    private Long teamId;                 // Equipo sancionado
    private String marcador;             // Marcador en el momento de la sanción
    private Date timestamp;              // Momento de la sanción
}
```

GameResult (Resultado Final)

```
@Entity
public class GameResult {
    private long id;
    private Game game;
    private Team winner;                 // Equipo ganador
    private Team loser;                 // Equipo perdedor
    private Date timeStart;             // Hora inicio del partido
    private Date timeEnd;               // Hora fin del partido
    private Long duration;              // Duración en milisegundos
    private int setsWonByLocal;         // Sets ganados por local
    private int setsWonByVisit;        // Sets ganados por visitante
}
```

Modelo de Datos: Personas y Equipos

Person (Persona - Clase Base)

```
@Entity
public class Person {
    private long id;
    private String dni;           // DNI único (12 caracteres)
    private String name;         // Nombre (20 caracteres)
    private String surnames;     // Apellidos (60 caracteres)
    private Date birthDate;      // Fecha de nacimiento
    private String address;      // Dirección (100 caracteres)
    private String phone;        // Teléfono (25 caracteres)
    private String email;        // Email (70 caracteres)
    private Boolean dniVerified; // DNI verificado
    private Boolean tutored;     // Es tutelado (menor)
    private Person tutor;       // Tutor (si es menor)

    // Relaciones especializadas
    private Coach coach;        // OneToOne si es entrenador
    private Player player;      // OneToOne si es jugador
    private Referee referee;    // OneToOne si es árbitro
}
```

Player (Jugador)

```
@Entity
public class Player {
    private long id;
    private String dni;          // FK a Person
    private Integer shirtNumber; // Número de camiseta
    private Team team;           // Equipo al que pertenece
    private Category category;   // Categoría del jugador
}
```

Coach (Entrenador)

```
@Entity
public class Coach {
    private long id;
    private String dni;           // FK a Person
    private String licenseNumber; // Número de licencia
    private String licenseLevel;  // Nivel de licencia
    private Team team;           // Equipo al que pertenece
}
```

Referee (Árbitro)

```
@Entity
public class Referee {
    private long id;
    private String dni;           // FK a Person
    private String licenseNumber; // Número de licencia
    private String licenseLevel;  // Nivel de licencia
    private City city;           // Ciudad de residencia
}
```

Team (Equipo)

```
@Entity
public class Team {
    private long id;
    private String name;          // Nombre del equipo
    private Person dniCaptain;    // Capitán del equipo
    private Club idClub;          // Club al que pertenece
    private Category category;    // Categoría del equipo
}
```

Club

```
@Entity
public class Club {
    private long id;
    private String name;          // Nombre del club
    private City city;           // Ciudad del club
}
```

Modelo de Datos: Definiciones y Lugares

City (Ciudad)

```
@Entity
public class City {
    private long id;
    private String name;           // Nombre de la ciudad
    private String region;        // Región/Provincia
    private String cpRange;       // Rango de códigos postales
}
```

Gameplace (Cancha/Pabellón)

```
@Entity
public class Gameplace {
    private long id;
    private String name;           // Nombre del pabellón
    private City city;            // Ciudad donde está ubicado
    private String address;       // Dirección completa
    private Integer courtsNumber; // Número de pistas disponibles
}
```

League (Liga)

```
@Entity
public class League {
    private long id;
    private String name;           // Nombre de la liga
    private Competition competition; // Competición a la que pertenece
    private Category category;     // Categoría
    private Division division;     // División
    private Season season;         // Temporada
}
```

Competition (Competición)

```
@Entity
public class Competition {
    private long id;
    private String name;           // Nombre de la competición
    private String description;    // Descripción
    private Season season;         // Temporada
}
```

Season (Temporada)

```
@Entity
public class Season {
    private long id;
    private String name;           // Ej: "National 2024-2025"
    private Date startDate;        // Fecha de inicio
    private Date endDate;          // Fecha de finalización
    private Boolean active;         // Temporada activa
}
```


Enumeraciones Implementadas

```
public enum Category {
    PRE_BENJAMIN(0, "Pre-Benjamin", 6, 7),
    BENJAMIN(1, "Benjamin", 8, 9),
    ALEVIN(2, "Alevin", 10, 11),
    INFANTILE(3, "Infantile", 12, 13),
    CADET(4, "Cadet", 14, 15),
    YOUTH(5, "Youth", 16, 17),
    JUNIOR(6, "Junior", 18, 19),
    ABSOLUTE(7, "Absolute", 20, 39),
    SENIOR(8, "Senior", 40, 99);
}

public enum Division {
    MIXED('X', "Mixed"),
    FEMININE('F', "Feminine"),
    MASCULINE('M', "Masculine");
}

public enum SanctionType {
    INDIVIDUAL('I', "Individual"),
    COACH('C', "Coach"),
    ASSISTANT_COACH('A', "Assistant Coach"),
    SUPPORT('T', "Support"),
    IMPROPER('S', "Improper solicitude"),
    DELAY('D', "Delay");
}
```

Sistema de Autenticación y Autorización

Spring Security + JWT

El sistema utiliza Spring Security con tokens JWT (JSON Web Tokens) para autenticación y autorización.

User (Usuario del Sistema)

```
@Entity
@Table(name = "users")
public class User {
    private Long id;
    private String username;           // Nombre de usuario único
    private String email;             // Email único
    private String password;          // Contraseña cifrada con BCrypt
    private Boolean enabled;          // Usuario habilitado
    private Boolean verified;         // Email verificado
    private Set<Role> roles;          // Roles asignados
    private Instant createdAt;        // Fecha de creación
    private Instant updatedAt;        // Última actualización
}
```

Role (Roles del Sistema)

```
@Entity
@Table(name = "roles")
public class Role {
    private Long id;
    private String name;              // Nombre del rol
    private Instant createdAt;
    private Instant updatedAt;
}
```

Roles Implementados

- **ROLE_ADMIN:** Control total sobre la plataforma. CRUD completo de todas las entidades.
- **ROLE_REFEREE:** Gestión de partidos asignados, registro de puntos y sanciones en tiempo real.
- **ROLE_USER:** Acceso a perfil personal, consulta de información pública, favoritos.

Endpoints de Autenticación

AuthController (/auth):

1. POST /auth/register

- Registro de nuevos usuarios
- Envío de email de verificación
- Respuesta: {status: 201, message: "User created"}

2. POST /auth/login

- Autenticación con username/email y password
- Respuesta: JWT token + refresh token
- Formato: {status: 200, message: {token, refreshToken, username, email, roles}}

3. GET /auth/verify?token={token}

- Verificación de email mediante token
- Activa la cuenta del usuario

4. GET /auth/forgot-password?email={email}

- Solicitud de reset de contraseña
- Envío de email con token de recuperación

5. POST /auth/reset-password

- Reseteo de contraseña con token
- Body: {token, newPassword}

6. GET /auth/validate

- Validación de sesión activa mediante JWT
- Requiere: Authorization: Bearer {token}

7. GET /auth/role/{role}

- Verificación de rol específico del usuario autenticado
- Requiere: Authorization: Bearer {token}

PermissionController (/permissions)

Gestión de permisos y roles (ROLE_ADMIN):

1. **GET /permissions/roles** - Listar todos los roles
2. **POST /permissions/roles** - Crear nuevo rol
3. **PUT /permissions/users/{userId}/roles** - Asignar/modificar roles de usuario

Seguridad Implementada

- **BCryptPasswordEncoder**: Cifrado de contraseñas con algoritmo BCrypt
- **JWT**: Tokens firmados con secret key, expiración configurable
- **Refresh Tokens**: Renovación de sesión sin re-autenticación
- **CORS**: Configuración permisiva con `@CrossOrigin("*")` en controladores
- **@PreAuthorize**: Anotaciones en métodos para control de acceso por rol

API REST: Módulo Admin

El módulo admin proporciona endpoints CRUD completos para todas las entidades. Requiere **ROLE_ADMIN**.

Estructura General de Controladores

Todos los controladores admin siguen el mismo patrón CRUD estándar:

- **GET** /admin/{resource} - Listar todos los elementos
- **GET** /admin/{resource}/{id} - Obtener elemento por ID
- **POST** /admin/{resource} - Crear nuevo elemento
- **PUT** /admin/{resource}/{id} - Actualizar elemento
- **DELETE** /admin/{resource}/{id} - Eliminar elemento

Controladores Implementados

Gestión de Personas

- **PersonController** (/admin/persons) - Personas del sistema
- **PlayerController** (/admin/players) - Jugadores
- **CoachController** (/admin/coaches) - Entrenadores
- **RefereeController** (/admin/referees) - Árbitros

Gestión de Clubes y Equipos

- **ClubController** (/admin/clubs) - Clubes
- **TeamController** (/admin/teams) - Equipos

Gestión de Ubicaciones

- **CitiesController** (/admin/cities) - Ciudades
- **GameplacesController** (/admin/gameplaces) - Pabellones/Canchas

Gestión de Competiciones

- **SeasonController** (/admin/seasons) - Temporadas
- **CompetitionController** (/admin/competitions) - Competiciones
- **LeagueController** (/admin/leagues) - Ligas

Gestión de Partidos

- **GameController** (/admin/games) - Partidos

Endpoints Especializados del GameController

Además de los endpoints CRUD estándar, GameController incluye:

GET /admin/games/code/{uniqueCode}

Obtiene un partido por su código único en lugar de por ID numérico.

- **Uso:** Búsqueda rápida cuando se conoce el código del partido (ej: "1A_PRE_AAA000")
- **Response:** Objeto Game completo con todas sus relaciones

GET /admin/games/refereeable

Lista partidos disponibles para asignar árbitro.

- **Filtros aplicados:** partidos no finalizados, no en curso, sin árbitro asignado
- **Uso:** Vista para asignación de árbitros a partidos pendientes

API REST: Módulo Referee

El módulo referee proporciona endpoints específicos para árbitros durante la gestión de partidos. Requiere **ROLE_REFEREE**.

RefereeGamesController (/referee/games)

GET /referee/games/{refereeld}

Obtiene todos los partidos asignados a un árbitro específico.

- **Auth:** ROLE_REFEREE
- **Response:** Lista de partidos asignados

RefereeMatchController (/referee/matches)

GET /referee/matches/{uniqueCode}

Obtiene los detalles completos de un partido por código único.

- **Auth:** ROLE_REFEREE
- **Response:** Objeto Game con todas sus relaciones

PUT /referee/matches/{uniqueCode}/start

Inicia un partido, creando el primer set automáticamente.

- **Auth:** ROLE_REFEREE
- **Lógica:**
 - Valida que el partido no esté ya en curso
 - Crea GameSet #1 con puntos en 0
 - Marca game.playing = true
 - Broadcast WebSocket: "STARTED"
- **Response:** Game actualizado

POST /referee/matches/{uniqueCode}/sets/{setId}/points

Actualiza los puntos de un set durante el partido.

- **Auth:** ROLE_REFEREE
- **Body:** {team: "local"|"visit", points: Integer}
- **Lógica:**
 - Incrementa puntos del equipo especificado
 - Valida reglas de finalización de set (25 puntos, diferencia de 2)
 - Finaliza set automáticamente si cumple condiciones
 - Crea siguiente set si es necesario
 - Finaliza partido automáticamente si un equipo gana 3 sets
 - Broadcast WebSocket: "POINT_UPDATE"
- **Response:** Set actualizado

POST /referee/matches/{uniqueCode}/sanctions

Registra una sanción durante el partido.

- **Auth:** ROLE_REFEREE
- **Body:** {type: SanctionType, teamId: Long, marcador: String}
- **Lógica:**
 - Crea GameSanctions con timestamp actual
 - Asocia al partido y equipo correspondiente
 - Broadcast WebSocket: "SANCTION_ADDED"
- **Response:** Sanción creada

PUT /referee/matches/{uniqueCode}/finish

Finaliza un partido manualmente.

- **Auth:** ROLE_REFEREE
- **Lógica:**
 - Finaliza el set activo
 - Calcula resultado final (ganador, perdedor, duración)
 - Crea GameResult con estadísticas
 - Marca game.finished = true
 - Broadcast WebSocket: "FINISHED"
- **Response:** Game finalizado con resultado

API REST: Módulo Open/Public

El módulo open proporciona endpoints públicos accesibles sin autenticación para consulta de información.

OpenGameController (/open/games)

GET /open/games

Lista todos los partidos disponibles para el público.

- **Auth:** No requiere
- **Response:** Lista de partidos con información básica

GET /open/games/outstanding

Obtiene el partido destacado (mayor relevancia).

- **Auth:** No requiere
- **Response:** Partido destacado con información completa

OpenClubController (/open/clubs)

GET /open/clubs

Lista todos los clubes registrados.

- **Auth:** No requiere
- **Response:** Lista de clubes

GET /open/clubs/{id}

Obtiene información detallada de un club específico.

- **Auth:** No requiere
- **Response:** Club con equipos y jugadores

OpenCompetitionController (`/open/competitions`)

GET /open/competitions

Lista todas las competiciones disponibles.

- **Auth:** No requiere
- **Response:** Lista de competiciones con ligas asociadas

GET /open/competitions/{id}

Obtiene información detallada de una competición.

- **Auth:** No requiere
- **Response:** Competición con ligas y partidos

OpenLeagueController (**/open/leagues**)

GET /open/leagues

Lista todas las ligas disponibles.

- **Auth:** No requiere
- **Response:** Lista de ligas

GET /open/leagues/{id}

Obtiene información detallada de una liga específica.

- **Auth:** No requiere
- **Response:** Liga con clasificación y partidos

WebSocket: Comunicación en Tiempo Real

Configuración WebSocket

Se utiliza STOMP (Simple Text Oriented Messaging Protocol) sobre WebSocket para comunicación bidireccional.

WebSocketConfig

```
@Configuration
@EnableWebSocketMessageBroker
public class WebSocketConfig implements WebSocketMessageBrokerConfigurer {

    @Override
    public void configureMessageBroker(MessageBrokerRegistry config) {
        config.enableSimpleBroker("/topic"); // Broadcast
        config.setApplicationDestinationPrefixes("/app");
    }

    @Override
    public void registerStompEndpoints(StompEndpointRegistry registry) {
        registry.addEndpoint("/ws")
            .setAllowedOriginPatterns("*")
            .withSockJS();
    }
}
```

RefereeWebSocketController

Endpoint: /app/games/{uniqueCode}/subscribe

Cliente se suscribe a actualizaciones de un partido específico.

- **Topic:** /topic/games/{uniqueCode}
- **Eventos emitidos:**
 - **STARTED** - Partido iniciado
 - **POINT_UPDATE** - Puntos actualizados
 - **SET_FINISHED** - Set finalizado
 - **SANCTION_ADDED** - Sanción registrada
 - **FINISHED** - Partido finalizado

LiveGameBroadcastService

Servicio que gestiona el broadcasting de eventos:

```
@Service
public class LiveGameBroadcastService {

    public void broadcastGameUpdate(String uniqueCode, Game game, String
eventType) {
        GameUpdateMessage message = new GameUpdateMessage();
        message.setEventType(eventType);
        message.setGame(game);
        message.setTimestamp(new Date());

        simpMessagingTemplate.convertAndSend(
            "/topic/games/" + uniqueCode,
            message
        );
    }
}
```

Flujo de Comunicación

1. Cliente se conecta: `new SockJS('/ws')`
2. Cliente se suscribe: `/topic/games/{uniqueCode}`
3. Árbitro actualiza puntos: `POST /referee/matches/{uniqueCode}/sets/{setId}/points`
4. Backend procesa y guarda en BD
5. Backend broadcast: `/topic/games/{uniqueCode}` con tipo "POINT_UPDATE"
6. Todos los clientes suscritos reciben actualización instantánea

Servicios Adicionales

ImageController (/images)

GET /images/{profilepic}

Obtiene la imagen de perfil del usuario autenticado.

- **Auth:** Requiere JWT
- **Response:** Imagen en bytes (image/jpeg)

POST /images/{profilepic}

Sube una nueva imagen de perfil.

- **Auth:** Requiere JWT
- **Body:** MultipartFile (imagen)
- **Response:** URL de la imagen guardada

ContactController (/mail/contact)

POST /mail/contact

Envía un email de contacto desde el formulario público.

- **Auth:** No requiere
- **Body:** {name, email, message}
- **Response:** Confirmación de envío

EmailService

Servicio interno para envío de correos electrónicos:

- Verificación de cuenta (con token)
- Recuperación de contraseña (con token)
- Notificaciones de sistema
- Formularios de contacto

HealthController (/health)

GET /health

Endpoint de health check para monitoreo.

- **Auth:** No requiere
- **Response:** OK{1}

Conclusión

El backend de DOT & DOT implementa una arquitectura REST robusta con comunicación en tiempo real mediante WebSockets, autenticación segura mediante JWT, y un modelo de datos completo que digitaliza el sistema de arbitraje de voleibol.

Tecnologías Utilizadas

- **Spring Boot 3.4.2** - Framework principal
- **Spring Security** - Autenticación y autorización
- **Spring Data JPA** - Persistencia con Hibernate
- **MySQL** - Base de datos relacional
- **JWT** - Tokens de autenticación
- **WebSocket + STOMP** - Comunicación en tiempo real
- **JavaMail** - Envío de correos
- **Lombok** - Reducción de boilerplate

Características Implementadas

1. Sistema completo de autenticación con verificación por email
2. Control de acceso basado en roles (Admin, Referee, User)
3. CRUD completo de todas las entidades del sistema
4. API de arbitraje con validación automática de reglas de voleibol
5. Comunicación en tiempo real mediante WebSockets
6. Endpoints públicos para consulta sin autenticación
7. Gestión de imágenes de perfil
8. Servicio de correo electrónico integrado
9. Broadcasting automático de eventos de partidos
10. Arquitectura en capas (Controller-Service-Repository)
11. Validaciones a nivel de entidad y controlador
12. Manejo centralizado de errores
13. Logging de actividades

Escalabilidad y Mantenimiento

La arquitectura implementada permite:

- Separación de responsabilidades por módulos
- Fácil extensión con nuevos endpoints
- Testing independiente por capas
- Migración a microservicios si es necesario
- Integración con servicios externos