LAPORAN KEGIATAN PRAKTIK KERJA INDUSTRI PERANCANGAN APLIKASI SCOREBOARD BERBASIS WEBSITE MENGGUNAKAN FRAMEWORK LAREVEL

DI PT NUSANTARA ADIDAYA TEKNOLOGI



Disusun Oleh:

|  |  |
| --- | --- |
| **NAMA: MALIK ARIFIN HAKIM**  **NAMA: RAFI FITROTULLOH** | **NIPD : 222310063**  **NIPD : 222310072** |

**YAYASAN SETYA BHAKTI**

**SMK TARUNA BHAKTI**

**SMK PUSAT KEUNGGULAN**

**TERAKREDITASI : “A“ No : 555/BAN-SM/SK/2023**

**Izin No : 421.4/1150/DISDIK/2004 - NPSN : 20229232**

PROGRAM KEAHLIAN :

PENGEMBANGAN PERANGKAT LUNAK DAN GIM

JALAN PEKAPURAN CURUG CIMANGGIS DEPOK 16953

TELP. : (021) 8744810 FAKS. : (021) 87743374

**Website :** <http://www.smktarunabhakti.net> **/ E-mail :** [taruna@smktarunabhakti.net](mailto:taruna@smktarunabhakti.net)

**2025**

# PERSETUJUAN PENULISAN LAPORAN

|  |  |  |
| --- | --- | --- |
| Judul  Nama / NIPD | :  : | Laporan Kegiatan Praktik Kerja Industri Perancangan Aplikasi Scoreboard Berbasis Website Menggunakan Framework Larevel Di Pt Nusantara Adidaya Teknologi   1. Rafi Fitrotulloh / 222310072 2. Malik Arifin Hakim **/** 222310063 |

Diperiksa oleh :

|  |  |
| --- | --- |
| Pembimbing Industri  **Fenton Martin**  **NIP.** | Pembimbing Penulisan Laporan  **Miranda, S.Pd**  **NIP. 19990808202302062** |

Disetujui oleh :

|  |  |
| --- | --- |
| Waka Hubungan Industri  **Lia Debby Juwita, S.Pd**  **NIP. 19750830201007082** | Kepala Program PPLG  **Miranda, S.Pd**  **NIP. 19990808202302062** |

# PENGESAHAN PENULISAN LAPORAN

|  |  |  |
| --- | --- | --- |
| Judul  Nama / NIPD | :  : | Laporan Kegiatan Praktik Kerja Industri Perancangan Aplikasi Scoreboard Berbasis Website Menggunakan Framework Larevel Di Pt Nusantara Adidaya Teknologi   1. Rafi Fitrotulloh / 222310072 2. Malik Arifin Hakim **/** 222310063 |

Telah diuji oleh Tim Penguji

|  |  |  |
| --- | --- | --- |
| Penguji  NIP  Tanggal Ujian Sidang  Tanggal Lulus | :  :  :  : |  |

Disahkan oleh :

|  |  |
| --- | --- |
| Pimpinan Perusahaan  **Fenton Martin**  **NIP.** | Kepala SMK Taruna Bhakti  **Nursidik, S.T**  **NIP. 19790911200411201** |

# KATA PENGANTAR

Puji dan Syukur kami panjatkan kepada kehadirat Allah SWT yang telah melimpahkan rahmat dan karunia-Nya, sehingga saya dapat menyelesaikan Praktik Kerja Lapangan (PKL) dan dapat menyusun laporan Praktik Kerja Lapangan (PKL) sampai selesai dengan baik. Adapun tujuan dari penulisan laporan ini adalah untuk memenuhi syarat kelulusan program pendidikan jurusan Pengembangan Perangkat Lunak dan Gim selama 3 tahun. Pada dasarnya saya banyak memperoleh bimbingan dan bantuan dari banyak pihak. Oleh karena itu, melalui kesempatan ini saya ingin menyampaikan terima kasih kepada:

1. Bapak Nursidik, ST. selaku kepala sekolah SMK Taruna Bhakti Depok.
2. Ibu Lia Debby Juwita, S.Pd. selaku Waka Kesiswaan bagian Hubin.
3. Ibu Miranda, S.Pd. selaku kepala program dan pembimbing.
4. Bapak Alven Desnecmen selaku Direktur Utama PT Nusantara Adidaya Teknologi.
5. Bapak Fenton Martin selaku pembimbing selama masa Praktik Kerja Lapangan (PKL) di PT Nusantara Adidaya Teknologi.
6. Serta pegawai lain yang berada di PT Nusantara Adidaya Teknologi yang senantiasa memberi bimbingan, pengarahan, pengetahuan, pengalaman kepada saya selama kurang lebih empat bulan.

Dalam pembuatan laporan ini telah dilakukan dengan segala usaha semaksimal mungkin. Kami menyadari masih banyak kekurangan dalam penulisan laporan magang ini. Maka dari itu kami secara terbuka menerima saran dan kritik positif yang sifatnya membangun.

Depok, ………. 2025

Tim Penyusun

# DAFTAR ISI

[PERSETUJUAN PENULISAN LAPORAN i](#_Toc188520664)

[PENGESAHAN PENULISAN LAPORAN ii](#_Toc188520665)

[KATA PENGANTAR iii](#_Toc188520666)

[DAFTAR ISI iv](#_Toc188520667)

[DAFTAR GAMBAR vii](#_Toc188520668)

[DAFTAR TABEL ix](#_Toc188520669)

[DAFTAR LAMPIRAN x](#_Toc188520670)

[BAB I PENDAHULUAN 1](#_Toc188520671)

[1.1 Latar Belakang 1](#_Toc188520672)

[1.2 Tujuan Praktik Kerja Lapangan (PKL) 2](#_Toc188520673)

[1.3 Waktu Dan Tempat Pelaksaan 2](#_Toc188520674)

[1.4 Manfaat Praktik Kerja Lapangan (PKL) 2](#_Toc188520675)

[1.5 Dasar Pelaksaan Praktik Kerja Lapangan (PKL) 3](#_Toc188520676)

[1.6 Sejarah Perusahaan 3](#_Toc188520677)

[1.7 Struktur Organisasi Perusahaan 4](#_Toc188520678)

[1.8 Jenis Produksi Perusahaan 5](#_Toc188520679)

[BAB II LANDASAN TEORI 6](#_Toc188520680)

[2.1 Rekayasa Perangkat Lunak 6](#_Toc188520681)

[2.2 Produk RPL 6](#_Toc188520682)

[2.2.1 Sistem Informasi 6](#_Toc188520683)

[2.2.2 Aplikasi 7](#_Toc188520684)

[2.3 Pemodelan Perangkat Lunak 8](#_Toc188520685)

[2.3.1 ERD 8](#_Toc188520686)

[2.3.2 DFD 9](#_Toc188520687)

[2.3.3 UML 11](#_Toc188520688)

[2.3.4 Rancangan User Interface 13](#_Toc188520689)

[2.4 Metode Pengembangan Perangkat Lunak 13](#_Toc188520690)

[2.4.1 Waterfall 13](#_Toc188520691)

[2.4.2 RAD 14](#_Toc188520692)

[2.4.3 Spiral 15](#_Toc188520693)

[2.4.4 Scrum 15](#_Toc188520694)

[2.4.5 Agile 16](#_Toc188520695)

[2.4.6 Prototype 17](#_Toc188520696)

[2.5 *Tools* Pengembangan Perangkat Lunak 18](#_Toc188520697)

[2.5.1 Bahasa Pemrograman 18](#_Toc188520698)

[2.5.2 Platform 19](#_Toc188520699)

[2.5.3 Database 20](#_Toc188520700)

[2.5.4 IDE 20](#_Toc188520701)

[BAB III METODOLOGI PENELITIAN 21](#_Toc188520702)

[3.1 Metode Pengembangan 21](#_Toc188520703)

[3.1.1 Waterfall 21](#_Toc188520704)

[3.2 Jadwal Pengembangan Aplikasi 34](#_Toc188520705)

[BAB IV PEMBAHASAN 35](#_Toc188520706)

[4.1 Aplikasi 35](#_Toc188520707)

[4.2 Alat dan Bahan 35](#_Toc188520708)

[4.2.1 Alat 35](#_Toc188520709)

[4.2.2 Bahan 35](#_Toc188520710)

[4.3 Langkah Pembuatan Aplikasi 35](#_Toc188520711)

[4.4 Hasil Tampilan Aplikasi 35](#_Toc188520712)

[BAB V PENUTUP 36](#_Toc188520713)

[5.1 Simpulan 36](#_Toc188520714)

[5.2 Saran 36](#_Toc188520715)

[DAFTAR PUSTAKA 37](#_Toc188520716)

[CV CREW 38](#_Toc188520717)

[KODE PROGRAM 39](#_Toc188520718)

[LEMBAR BIMBINGAN 40](#_Toc188520719)

# DAFTAR GAMBAR

[Gambar 1. Struktur Organisasi Perusahaan 4](#_Toc188863487)

[Gambar 2. ERD 8](#_Toc188863488)

[Gambar 3. DFD 9](#_Toc188863489)

[Gambar 4. Metode *Waterfall* 14](#_Toc188863490)

[Gambar 5. Metode *Rapid Application Development* (RAD) 15](#_Toc188863491)

[Gambar 6. Metode Spiral 16](#_Toc188863492)

[Gambar 7. Scrum 16](#_Toc188863493)

[Gambar 8. Agile 17](#_Toc188863494)

[Gambar 9. *Prototype* 18](#_Toc188863495)

[Gambar 10. *Data Flow Diagram* (DFD) *Level* 0 *Scoreboard* 23](#_Toc188863496)

[Gambar 11. Flowchart *Scoreboard* 23](#_Toc188863497)

[Gambar 12. Usecase Diagram Scoreboard 24](#_Toc188863498)

[Gambar 13. Sequence Diagram *Scoreboard* 24](#_Toc188863499)

[Gambar 14. Class Diagram 25](#_Toc188863500)

[Gambar 15. *Entity Relationship Diagram* (ERD) 25](#_Toc188863501)

[Gambar 16. Mock up Halaman *Login* 26](#_Toc188863502)

[Gambar 17. Mock up Halaman *Home* 26](#_Toc188863503)

[Gambar 18. Mock up Halaman *Teams* 27](#_Toc188863504)

[Gambar 19. Mock up Tambah *Teams Dialog* 27](#_Toc188863505)

[Gambar 20. Mock up Import *Teams* 27](#_Toc188863506)

[Gambar 21. Mock up Tambah *Teams Dialog* 28](#_Toc188863507)

[Gambar 22. Mock up Edit *Teams Dialog* 28](#_Toc188863508)

[Gambar 23. Mock up Halaman *Players* 28](#_Toc188863509)

[Gambar 24. Mock up Tambah *Players Dialog* 29](#_Toc188863510)

[Gambar 25. Mock up Edit *Player Dialog* 29](#_Toc188863511)

[Gambar 26. Mock up *Export Player Dialog* 29](#_Toc188863512)

[Gambar 27. Mock up Halaman *Matches* 30](#_Toc188863513)

[Gambar 28. Mock up Tambah *Matches Dialog* 30](#_Toc188863514)

[Gambar 29. Mock up Edit *Match Dialog* 30](#_Toc188863515)

[Gambar 30. Mock up Halaman *Scoreboards* 31](#_Toc188863516)

[Gambar 31. Mock up Tambah *Scoreboard Dialog* 31](#_Toc188863517)

[Gambar 32. Mock up Edit *Scoreboard Dialog* 31](#_Toc188863518)

[Gambar 33. Mock up Halaman *Old Matches* 32](#_Toc188863519)

[Gambar 34. Mock up Halaman *Screen Settings* 32](#_Toc188863520)

[Gambar 35. *Display* *Line - up Team* 32](#_Toc188863521)

[Gambar 36. *Display Timer Match* 33](#_Toc188863522)

[Gambar 37. *Display* Pergantian Pemain 33](#_Toc188863523)

[Gambar 38. *Display* Goal 33](#_Toc188863524)

[Gambar 39. *Display* Pelanggaran Kartu Kuning 34](#_Toc188863525)

[Gambar 40. *Display* Pelanggaran Kartu Merah 34](#_Toc188863526)

# DAFTAR TABEL

[Tabel 1. Simbol ERD 9](#_Toc188520615)

[Tabel 2. Simbol DFD 10](#_Toc188520616)

[Tabel 3. Jadwal Pengembangan Aplikasi Scoreboard 34](#_Toc188520617)

# DAFTAR LAMPIRAN

# PENDAHULUAN

## Latar Belakang

Di era digital saat ini, teknologi telah menjadi elemen penting dalam berbagai bidang, termasuk industri olahraga. Transformasi digital dalam dunia olahraga tidak hanya meningkatkan efisiensi operasional tetapi juga memberikan pengalaman yang lebih baik bagi penonton, atlet, dan penyelenggara acara. Solusi berbasis teknologi seperti aplikasi *scoreboard* digital memainkan peran krusial dalam mendukung kebutuhan tersebut.

PT Nusantara Adidaya Teknologi (NAT), yang berfokus pada penyediaan solusi teknologi inovatif untuk mendukung transformasi digital bisnis, telah menjadi mitra penting dalam pengembangan solusi berbasis teknologi. Dengan visi menjadi penyedia layanan teknologi terkemuka di Indonesia, PT NAT menawarkan layanan seperti otomatisasi operasional bisnis dan pengembangan sistem berbasis IoT. Salah satu proyek unggulannya adalah pengembangan Aplikasi *Scoreboard* berbasis website untuk PT Spyros Indonesia.

PT Spyros Indonesia adalah perusahaan internasional yang mengkhususkan diri dalam solusi untuk pusat olahraga. Dengan pengalaman luas dalam konsultasi, distribusi, dan implementasi peralatan olahraga, PT Spyros memesan pengembangan aplikasi ini untuk meningkatkan efisiensi dan profesionalisme dalam penyelenggaraan pertandingan olahraga.

Aplikasi *Scoreboard* ini dirancang dengan fitur-fitur unggulan seperti live *scoreboard*, pengaturan waktu pertandingan, pengelolaan data tim dan pemain, serta riwayat pertandingan. Teknologi yang digunakan dalam pengembangannya meliputi Laravel untuk backend, Tailwind CSS untuk frontend, MySQL sebagai database, dan Redis untuk manajemen cache.

Melalui laporan ini, akan dijelaskan bagaimana proses pengembangan aplikasi ini dilakukan, mulai dari identifikasi kebutuhan klien hingga implementasi fitur-fitur yang mendukung pengalaman olahraga yang lebih baik dan profesional.

## Tujuan Praktik Kerja Lapangan (PKL)

Pelaksanaan Praktik Kerja Industri (PRAKERIN) diselenggarakan Sekolah Menengah Kejuruan dengan tujuan:

1. Memberikan pengalaman kerja langsung (*real*) untuk menanamkan (*internalize*) iklim kerja positif yang berorientasi pada peduli mutu proses dan hasil kerja.
2. Menanamkan etos kerja yang tinggi bagi peserta didik untuk memasuki dunia kerja menghadapi tuntutan pasar kerja global.
3. Memenuhui hal-hal yang belum dipenuhi di sekolah agar mencapai keutuhan standar kompetensi lulusan.
4. Mengaktualisasikan penyelenggaraan Model Pendidikan Sistem Ganda (PSG) antara SMK dan Institusi Pasangan (DUDI), memadukan secara sistematis dan sistemik program pendidikan di SMK dan program latihan di dunia kerja (DUDI).

## Waktu Dan Tempat Pelaksaan

|  |  |  |
| --- | --- | --- |
| Tempat Pelaksanaan | : | PT Nusantara Adidaya Teknologi |
| Alamat | : | Apartment Midtown Residence, Tower 1, Lantai 11, Unit 1114, Jl. TB Simatupang No.20, Cilandak Barat., Kec. Cilandak, Kota Jakarta Selatan, Daerah Khusus Ibukota Jakarta 12430 |
| Waktu Pelaksanaan | : | 04 September 2024 – 30 Desember 2024 |

## Manfaat Praktik Kerja Lapangan (PKL)

Adapun manfaat pelaksanaan Paktik Kerja Lapangan (PKL) adalah sebagai berikut:

1. Menghasilkan sumber daya manusia yang memiliki keahlian yang profesional, dengan keterampilan, pengetahuan, serta etika dan etos kerja yang sesuai dengan dunia kerja.
2. Mengasah keterampilan siswa/i SMK.
3. Menjalin hubungan kerja sama antara sekolah dan perusahaan terkait, baik dalam dunia usaha atau industri.
4. Memperkenalkan siswa/i kepada dunia kerja.
5. Meningkatkan efisiensi waktu dan tenaga dalam mendidik dan melatih tenaga kerja yang berkualitas.

## Dasar Pelaksaan Praktik Kerja Lapangan (PKL)

Praktik Kerja Lapangan (PKL) menjadi salah satu penyelenggaraan Pendidikan Sekolah Menengah Kejuruan (SMK) sesuai dengan ketentuan-ketentuan yang tercantum dalam:

1. Permendikbud No 50. Th. 2020 Tentang Praktik Kerja Lapangan bagi Peserta Didik
2. Undang-Undang Republik Indonesia Nomor 20 Tahun 2003 tentang Sistem Pendidikan Nasional
3. Peraturan Pemerintah Republik Indonesia Nomor 19 Tahun 2005 tentang Standar Nasional Pendidikan
4. Peraturan Menteri Ketenagakerjaan Republik Indonesia Nomor 6 Tahun 2020 tentang Penyelenggaraan Pemagangan di Dalam Negeri
5. Peraturan Direktur Jenderal Pendidikan Dasar dan Menengah Kemendikbud Nomor 07/D.D5/KK/2018 tentang Struktur Kurikulum Sekolah Menengah Kejuruan/Madrasah Aliyah Kejuruan

## Sejarah Perusahaan

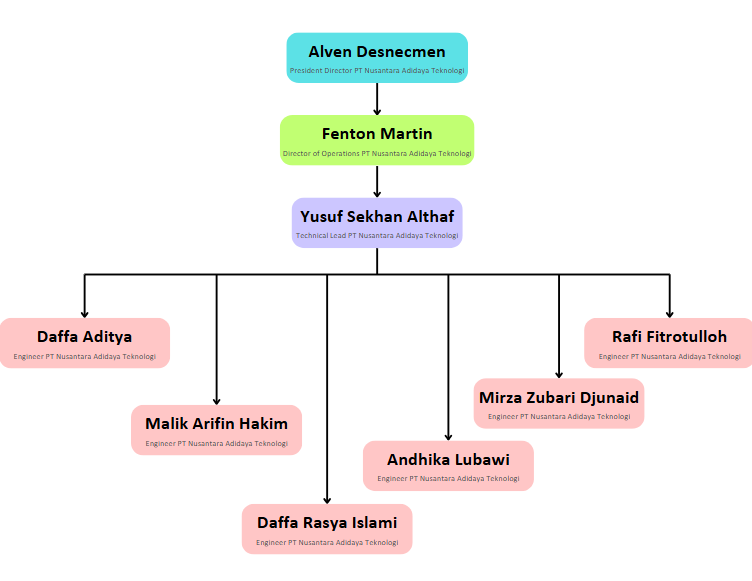
PT Nusantara Adidaya Teknologi didirikan untuk menjawab tantangan era digital di Indonesia. Dengan meningkatnya kebutuhan bisnis terhadap solusi berbasis teknologi yang inovatif, perusahaan ini hadir untuk menyediakan layanan yang dapat membantu meningkatkan efisiensi dan efektivitas operasional bisnis. Fokus utama perusahaan adalah memberikan solusi teknologi yang mendukung kebutuhan teknis bisnis lokal dan nasional.

Visi perusahaan adalah menjadi penyedia layanan teknologi terkemuka di Indonesia yang memberikan hasil nyata bagi klien. Untuk mencapai visi tersebut, misi perusahaan meliputi membantu bisnis memanfaatkan teknologi modern untuk meningkatkan operasional mereka serta menyediakan layanan yang mudah diakses, efektif, dan sesuai dengan kebutuhan pasar lokal.

PT Nusantara Adidaya Teknologi didirikan pada bulan Mei 2023 dan memiliki kantor administrasi yang berlokasi di Cyber2 Tower, Jakarta Selatan, Indonesia. Operasional perusahaan dimulai dengan fokus pada pengembangan solusi otomatisasi operasional bisnis. Seiring waktu, layanan perusahaan diperluas untuk mencakup solusi berbasis *Internet of Things* (IoT) yang mendukung transformasi digital yang lebih luas, serta layanan dukungan teknis IT untuk memenuhi kebutuhan bisnis.

Dalam kontribusinya kepada industri, perusahaan memberikan pelatihan dan dukungan kepada klien dalam memahami tren teknologi terkini serta menjadi mitra strategis bagi berbagai sektor bisnis, dari *UMKM* hingga korporasi besar. Untuk menjaga relevansi, PT Nusantara Adidaya Teknologi terus berinvestasi dalam teknologi terkini dan mengembangkan solusi berbasis data yang dapat diadaptasi dengan cepat sesuai perubahan pasar.

## Struktur Organisasi Perusahaan

****

Gambar . Struktur Organisasi Perusahaan

## Jenis Produksi Perusahaan

Sebagai penyedia solusi terhadap kebutuhan konsumen atau klien, PT Nusantara Adidaya Teknologi memiliki solusi berupa produk yang akan membantu sebuah perusahaan untuk menjalankan bisnisnya khususnya pada bidang IT atau digital dan IoT.

1. Layanan IT *Support*:

Menyediakan dukungan teknis meliputi pemeliharaan perangkat keras, perangkat lunak, dan infrastruktur IT untuk memastikan kelancaran operasional bisnis klien. Layanan untuk mengidentifikasi dan memperbaiki masalah teknis yang dapat mengganggu operasional bisnis, baik dari segi perangkat keras maupun perangkat lunak.

1. Solusi *Internet of Things* (IoT):

Pengembangan sistem berbasis IoT yang memungkinkan perusahaan untuk mengintegrasikan perangkat pintar dalam operasi mereka, seperti otomatisasi industri, pemantauan real-time, dan pengelolaan inventaris. Solusi IoT yang membantu bisnis meningkatkan efisiensi operasional, misalnya melalui pengelolaan energi yang lebih baik, pemantauan kondisi perangkat, atau pengelolaan sumber daya secara otomatis.

1. Pelatihan dan Dukungan Strategis:

Program pelatihan untuk membantu klien memahami tren terbaru dalam teknologi dan penerapan IoT dalam operasional mereka. Layanan konsultasi untuk membantu klien merancang dan mengimplementasikan solusi teknologi yang tepat untuk mendukung transformasi digital dan meningkatkan efisiensi operasional.

Dengan fokus pada dukungan IT dan solusi berbasis IoT, PT Nusantara Adidaya Teknologi dapat membantu perusahaan dalam mengoptimalkan operasional dan meraih efisiensi melalui penerapan teknologi terkini.

# LANDASAN TEORI

## Rekayasa Perangkat Lunak

Rekayasa Perangkat Lunak (RPL) adalah disiplin ilmu yang menerapkan pendekatan terstruktur dan sistematis dalam pengembangan perangkat lunak berkualitas tinggi. Proses ini mencakup analisis kebutuhan, desain, implementasi, pengujian, dan pemeliharaan perangkat lunak, dengan tujuan memastikan perangkat lunak berfungsi optimal dan memenuhi kebutuhan pengguna. RPL juga mengintegrasikan berbagai bidang ilmu, seperti ilmu komputer, ekonomi, komunikasi, dan manajemen, untuk menghasilkan solusi perangkat lunak yang efisien dan efektif.

## Produk RPL

Rekayasa Perangkat Lunak (RPL) adalah disiplin ilmu yang mencakup seluruh aspek produksi perangkat lunak, mulai dari analisis kebutuhan pengguna, desain, pengkodean, pengujian, hingga pemeliharaan sistem setelah diimplementasikan. RPL tidak hanya berfokus pada pembuatan program komputer, tetapi juga melibatkan manajemen proyek, penentuan personel, anggaran biaya, metode, jadwal, kualitas, dan pelatihan pengguna. Produk perangkat lunak dapat dikategorikan menjadi dua jenis utama:

### Sistem Informasi

Sistem Informasi adalah serangkaian prosedur yang digunakan untuk mengumpulkan, menyimpan, memproses, dan menganalisis data dalam jumlah besar guna menemukan wawasan dan tren berharga, memungkinkan organisasi mengelola data lebih baik, membuat keputusan tepat, dan meningkatkan kinerja.

Menurut para ahli, Sistem Informasi melibatkan kombinasi manusia, teknologi (perangkat keras dan lunak), media, prosedur, dan pengendalian untuk mengatur komunikasi, memproses transaksi, membantu manajemen, dan menyediakan dasar pengambilan keputusan.

Manfaatnya meliputi efisiensi operasional melalui otomatisasi proses bisnis, pengurangan waktu pengerjaan tugas, pengambilan keputusan lebih baik, peningkatan responsivitas pasar, pengurangan biaya, dan integrasi data antar departemen.

Di era digital, Sistem Informasi berperan penting dalam memberikan keunggulan kompetitif bagi perusahaan yang mampu beradaptasi dengan teknologi informasi, mengoptimalkan proses bisnis, dan memberikan layanan lebih baik kepada pelanggan.

### Aplikasi

Aplikasi adalah perangkat lunak yang dirancang untuk menjalankan fungsi atau tugas tertentu sesuai kebutuhan pengguna. Aplikasi dapat berupa perangkat lunak desktop, web, atau mobile yang dikembangkan untuk berbagai tujuan, seperti produktivitas, hiburan, pendidikan, atau manajemen.

Dalam konteks pengembangan perangkat lunak modern, aplikasi sering kali dibangun menggunakan teknologi terkini untuk memastikan performa yang optimal, antarmuka pengguna yang intuitif, serta kemudahan integrasi dengan sistem lain. Salah satu contoh pengembangan aplikasi adalah aplikasi *Scoreboard* berbasis website, yang bertujuan untuk menyajikan informasi pertandingan olahraga secara *real-time* dan terstruktur.

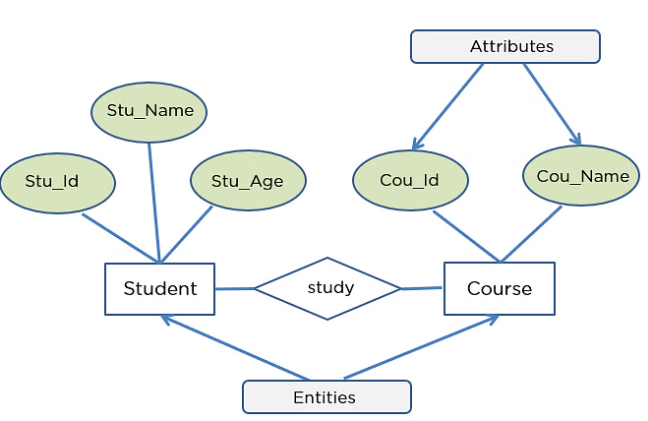
Aplikasi ini menawarkan berbagai fitur, seperti pengaturan waktu pertandingan, tampilan skor langsung, dan pengelolaan data tim dan pemain. Teknologi yang digunakan dalam pengembangan aplikasi mencakup Laravel sebagai framework *backend*, Tailwind CSS untuk desain antarmuka, MySQL sebagai basis data, serta Redis untuk manajemen cache. Dengan fitur dan teknologi tersebut, aplikasi ini dapat memberikan solusi yang efisien dan profesional bagi kebutuhan pengelolaan acara olahraga.

## Pemodelan Perangkat Lunak

Pemodelan Perangkat Lunak adalah disiplin ilmu untuk mempelajari bentuk-bentuk pemodelan perangkat lunak yang digunakan sebagai bagian dari tahapan pengembangan perangkat lunak secara terstruktur dan berorientasi objek. Pemodelan dalam rekayasa perangkat lunak merupakan suatu hal yg dilakukan di tahapan awal.

### ERD

*Entity Relationship Diagram* (ERD) atau diagram hubungan entitas adalah sebuah diagram yang digunakan untuk perancangan suatu database dan menunjukan relasi atau hubungan antar objek atau entitas beserta atribut-atributnya secara detail. Dengan menggunakan ERD, sistem database yang sedang dibentuk dapat digambarkan dengan lebih terstruktur dan terlihat rapi.



Gambar . ERD

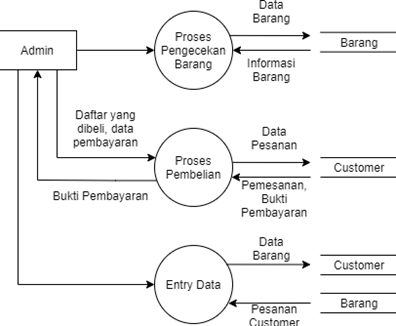
ERD ditemukan oleh Peter Chen dalam buku *Entity Relational Model-Toward a Unified of Data* pada tahun 1976. Chen mencoba merumuskan dasar-dasar model dan setelah itu dikembangkan dan dimodifikasi oleh Chen dan banyak pakar lainnya.

Tabel . Simbol ERD

|  |  |
| --- | --- |
| Simbol | Keterangan |
|  | Entitas merupakan kumpulan dari objek yang  dapat diidentifikasikan secara unik. |
|  | Relasi merupakan hubungan yang terjadi antara salah satu lebih entitas. Jenis hubungan  antara lain, one to one, one to many dan many to many. |
|  | Atribut merupakan karakteristik dari entitas atau relasi yang merupkan penjelasan. |
|  | Hubungan antara entitas dengan atributnya dan hubungan entitas dengan himpunan relasinya. |

### DFD

Data Flow Diagram (DFD) adalah alat yang digunakan untuk menggambarkan aliran data dalam suatu sistem informasi. DFD memperlihatkan input dan output dari setiap proses dalam sistem, tanpa mengontrol alur keputusan atau pengulangan.



Gambar . DFD

DFD pertama kali dipopulerkan pada 1970 oleh Larry Constantine dan Ed Yourdon, dan banyak digunakan dalam pengembangan perangkat lunak. Fungsi utama DFD meliputi penyampaian rancangan sistem dengan cara visual yang mudah dimengerti, menggambarkan sistem sebagai jaringan fungsional yang terhubung, serta perancangan model untuk melihat detail dari sistem tersebut.

Tabel . Simbol DFD

|  |  |
| --- | --- |
| **Simbol** | **Keterangan** |
|  | Proses menunjukan transformasi dari masukan menjadi keluaran. |
|  | Entitas Eksternal dimana entitas tersebut berkomunikasi dengan sistem. |
|  | Penyimpanan menunjukan penyimpanan dalam sebuah database. |
|  | Aliran menggambarkan aliran data yang masuk ke proses atau keluar dari suatu proses. |

Dengan menggunakan DFD, pengembang dapat memahami aliran data dalam sistem secara visual, memudahkan identifikasi masalah dan perancangan sistem yang lebih efisien. DFD memiliki beberapa level yang menunjukkan tingkat detail dari sistem yang dianalisis:

#### Level 0 (Diagram Konteks)

Memberikan gambaran umum dari sistem secara keseluruhan, menunjukkan entitas eksternal yang berinteraksi dengan sistem dan aliran data utama yang terjadi.

#### Level 1

Menguraikan proses utama yang ada dalam sistem, memperlihatkan bagaimana data mengalir antar proses dan entitas eksternal

#### Level 2 dan seterusnya

Memberikan detail lebih lanjut dari proses yang ada pada Level 1, menggambarkan sub-proses dan aliran data yang lebih spesifik.

### UML

UML (*Unified Modeling Language*) adalah bahasa pemodelan visual yang digunakan untuk merancang dan mendokumentasikan sistem, terutama yang berorientasi objek. UML menyediakan berbagai diagram, seperti diagram use case, diagram kelas, dan diagram urutan, yang membantu pengembang dan pemangku kepentingan untuk berkomunikasi mengenai desain sistem. Dengan menggunakan UML, kompleksitas sistem dapat dipahami dengan lebih mudah, meningkatkan kolaborasi dalam pengembangan dan pemeliharaan sistem.

#### Notasi UML

Notasi UML terdiri dari berbagai bentuk grafis yang memiliki makna tertentu. UML *Syntax* mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML diambil dari tiga metodologi sebelumnya:

1. Grady Booch OOD (*Object-Oriented Design*)
2. Jim Rumbaugh OMT (*Object Modeling Technique*)
3. Ivar Jacobson OOSE (*Object-Oriented Software Engineering*)

#### Konsep Dasar UML

UML memiliki konsep dasar untuk penggunaannya. Konsep dasar dalam UML terdiri dari beberapa bagian:

1. *Structural Classification*: Menyusun struktur dasar dari objek-objek yang terlibat dalam sistem.
2. *Dynamic Behavior*: Menggambarkan bagaimana objek berinteraksi dan berubah seiring waktu.
3. *Model Management*: Mengatur cara model UML digunakan dan dimodifikasi dalam pengembangan perangkat lunak.

#### Benda atau Objek dalam UML

UML memiliki benda atau objek di dalamnya. Secara garis besar, terdapat lima kategori utama benda atau objek dalam UML yaitu:

1. Kelas (*Classes*): Sekelompok objek dengan atribut dan operasi yang memiliki hubungan semantik.
2. *Interface*: Tampilan yang menghubungkan dan melayani setiap kelas dalam pemodelan.
3. Kolaborasi: Interaksi antara beberapa kelas atau elemen yang bekerja bersama untuk mencapai tujuan tertentu.
4. *Nodes*: Elemen yang ada selama eksekusi sistem, menggambarkan komponen fisik atau virtual yang menjalankan sistem.
5. Things: Ada empat kategori "*things*" dalam UML yaitu *Structural Things*, *Behavioral Things*, *Grouping Things* dan *Annotational Things*.

#### Relasi dalam UML

Relasi menggambarkan hubungan antara elemen-elemen dalam sistem:

1. Ketergantungan (*Dependency*): Ketika perubahan pada satu elemen independen memengaruhi elemen yang bergantung padanya.
2. Asosiasi: Hubungan antar objek dalam sistem, termasuk agregasi (hubungan antara objek dengan bagian-bagiannya).
3. Generalisasi: Hubungan di mana objek anak berbagi perilaku dan struktur data dengan objek induknya.

#### Diagram UML

Diagram adalah representasi grafis yang menggambarkan aspek tertentu dari sistem. Beberapa jenis diagram yang sering digunakan dalam UML adalah:

1. *Use Case Diagram*: Diagram ini menggambarkan interaksi antara aktor (misalnya pengguna atau sistem lain) dan use case (fungsi atau layanan yang ditawarkan oleh sistem).
2. *Class Diagram*: Diagram ini menggambarkan struktur statis dari sistem. *Class diagram* menunjukkan kelas-kelas yang ada dalam sistem, atribut-atribut mereka, operasi yang mereka miliki, dan hubungan antara kelas-kelas tersebut.

### Rancangan User Interface

*User interface* (UI) adalah aspek penting dalam teknologi yang mengacu pada tampilan dan tata letak sistem yang digunakan untuk berinteraksi dengan pengguna. UI mencakup elemen visual seperti tombol, ikon, menu, dan tata letak yang dirancang untuk memberikan pengalaman interaksi yang intuitif, efisien, dan estetis.

Konsep UI tidak hanya berfokus pada keindahan visual, tetapi juga pada fungsi dan kemudahan penggunaan, sehingga memungkinkan pengguna untuk memahami dan menggunakan sistem dengan mudah. Desain UI yang baik mempertimbangkan kebutuhan pengguna dan berusaha menciptakan pengalaman yang nyaman dan memuaskan.

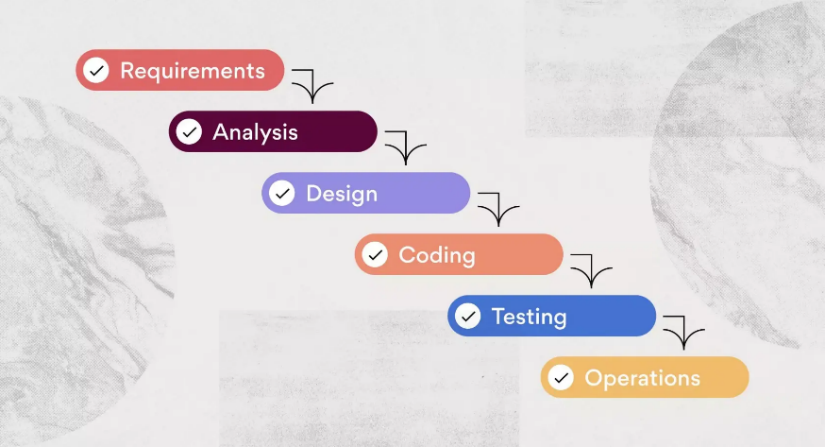
## Metode Pengembangan Perangkat Lunak

Metode perangkat lunak adalah kerangka kerja yang digunakan untuk menstrukturkan, merencanakan, dan mengendalikan proses pengembangan suatu sistem informasi. Berikut contoh metode pengembangan perangkat lunak yaitu:

### Waterfall

Metode *waterfall* adalah metodologi manajemen proyek yang dapat membantu Anda dan tim Anda mencapai tujuan bersama, satu tugas atau tonggak penting pada satu waktu. Dengan memprioritaskan tugas dan ketergantungan, metode waterfall membantu menjaga proyek Anda tetap pada jalurnya. Metode *waterfall* adalah proses desain berurutan yang digunakan dalam pengembangan perangkat lunak dan pengembangan produk di mana kemajuan proyek mengalir terus ke bawah melalui beberapa fase, seperti air terjun.

Model *waterfall* terstruktur di sekitar urutan langkah yang kaku yang bergerak dari konsepsi, inisiasi, analisis, desain, konstruksi, pengujian, implementasi, dan pemeliharaan. Metode *waterfall* mengharuskan setiap fase proyek diselesaikan sepenuhnya sebelum fase berikutnya dimulai. Ini membuatnya lebih mudah untuk selaras dengan anggaran tetap, tenggat waktu, dan persyaratan. Dengan mengintegrasikan dokumentasi komprehensif dan perencanaan awal yang luas, metode *waterfall* meminimalkan risiko dan cenderung selaras dengan pendekatan manajemen proyek tradisional yang bergantung pada catatan terperinci dan jalur yang jelas dan telah ditentukan sebelumnya untuk diikuti.

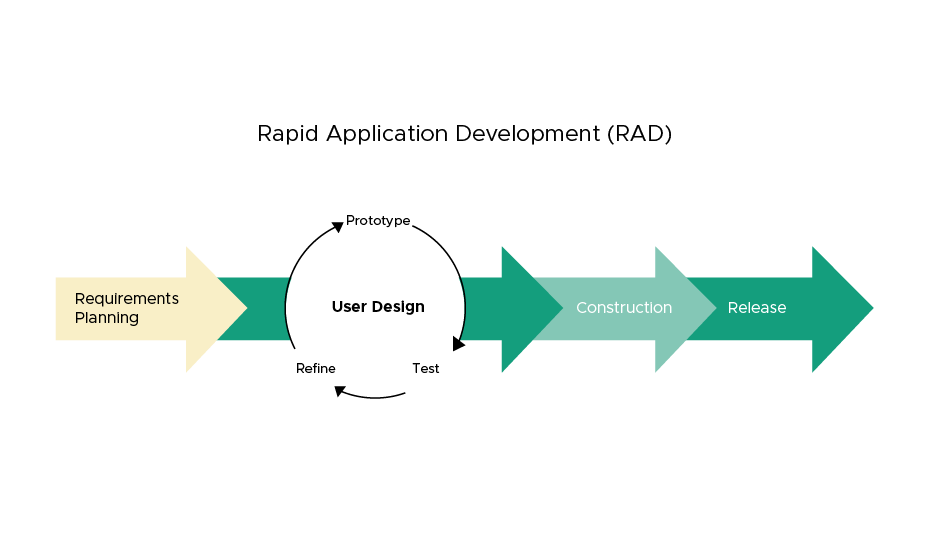


Gambar . Metode *Waterfall*

### RAD

Metode *Rapid Application Development* (RAD) adalah metodologi pengembangan perangkat lunak yang menekankan pada pengembangan prototipe secara cepat dan iteratif, dengan fokus utama pada umpan balik pengguna dan perangkat lunak yang berfungsi daripada perencanaan dan dokumentasi yang mendetail. Tujuan utama RAD adalah mempercepat proses pengembangan aplikasi dan memfasilitasi adaptasi terhadap perubahan kebutuhan bisnis. Metode ini sering digunakan untuk proyek-proyek yang sensitif terhadap waktu dan membutuhkan respons yang cepat terhadap perubahan.

RAD umumnya terdiri dari empat fase: perencanaan persyaratan, desain pengguna, konstruksi cepat (pengembangan prototipe), dan transisi (implementasi dan rilis). Keuntungan utama RAD meliputi waktu pengembangan yang lebih singkat, peningkatan efisiensi melalui penggunaan kembali komponen, umpan balik pengguna yang konstan, dan integrasi sistem yang lebih awal. Namun, RAD juga memiliki beberapa kekurangan, seperti membutuhkan tim yang sangat terampil, kurang cocok untuk proyek skala besar atau yang sangat kompleks, dan membutuhkan keterlibatan pengguna yang intensif.



Gambar . Metode *Rapid Application Development* (RAD)

### Spiral

Model spiral dalam pengembangan perangkat lunak adalah pilihan terbaik untuk proyek-proyek berisiko, ekstensif, dan rumit. Representasi visual proses ini menyerupai jaring spiral dengan banyak putaran, di mana jumlah putaran bergantung pada jenis dan kebutuhan proyek. Setiap putaran dalam model spiral memiliki empat fase utama proses pengembangan perangkat lunak. Model ini pertama kali diperkenalkan oleh insinyur perangkat lunak Amerika, Barry Boehm, dalam tulisannya "*Spiral Model of Software Development and Enhancement*" pada tahun 1986.

Metode spiral dalam proses pengembangan perangkat lunak menggabungkan model pengembangan iteratif dan elemen-elemen spesifik dari model waterfall. Model spiral dimulai dengan pengumpulan sejumlah kecil persyaratan dan melalui fase pengembangan kecil. Berbeda dengan model *waterfall*, model spiral memungkinkan penambahan fungsionalitas pada setiap fase spiral yang meningkat.

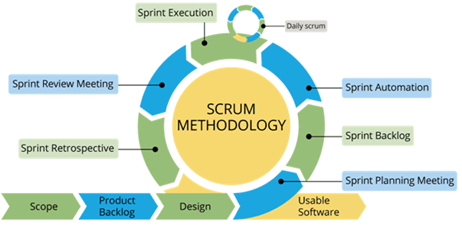


Gambar . Metode Spiral

### Scrum

Scrum adalah kerangka kerja (*framework*) Agile yang populer dalam pengembangan perangkat lunak, menekankan pada iterasi dan inkremental. Berfokus pada kolaborasi tim, transparansi (*transparency*), inspeksi (*inspection*), dan adaptasi (*adaptation*), Scrum dibangun atas tiga pilar dan lima nilai, termasuk komitmen (*commitment*) dan fokus (*focus*). Proyek dibagi menjadi siklus pendek yang disebut sprint, menghasilkan increment produk fungsional.

Proses Scrum melibatkan peran *Product Owner* (visi produk dan *Product Backlog*), *Scrum Master* (memfasilitasi dan menghilangkan hambatan), dan *Development Team* (pengembang produk). Acara (*events*) Scrum meliputi *Sprint Planning*, *Daily Scrum*, *Sprint Review* (*increment* untuk *stakeholder*), dan *Sprint Retrospective* (evaluasi sprint). Dengan iterasi dan kolaborasi, Scrum memungkinkan respons fleksibel terhadap perubahan dan menghasilkan produk bernilai berkelanjutan.

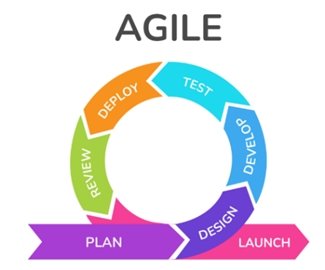


Gambar . Scrum

### Agile

Dalam pengembangan perangkat lunak, *Agile* adalah metodologi berbasis proses iteratif dengan aturan dan solusi yang disepakati, serta kolaborasi tim terstruktur. Menekankan adaptasi terhadap perubahan dan pengambilan keputusan cepat, *Agile* ideal untuk proyek jangka pendek. Proses *Agile development* berlandaskan *Agile* *Manifesto* yang digagas oleh tokoh-tokoh pengembangan perangkat lunak, bertujuan menghasilkan sistem aplikasi bernilai tinggi dan berfungsi (*high value & working app system*).

Prinsip *Agile Manifesto* memprioritaskan kepuasan pelanggan melalui pengiriman perangkat lunak berfungsi berkelanjutan, menerima perubahan persyaratan bahkan di tahap akhir, pengiriman sering dalam waktu singkat, kolaborasi erat bisnis dan pengembang, tim termotivasi dengan dukungan memadai, komunikasi tatap muka, perangkat lunak berfungsi sebagai metrik kemajuan, proses berkelanjutan untuk pengembangan, perhatian pada keunggulan teknis dan desain, kesederhanaan, arsitektur, persyaratan, dan desain terbaik dari tim yang mengatur diri sendiri, serta refleksi berkala tim untuk efektivitas.

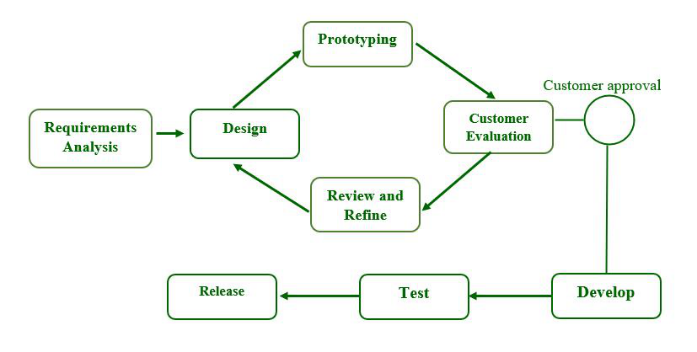


Gambar . Agile

### Prototype

Metode *prototyping* dalam pengembangan perangkat lunak adalah proses pembuatan model kerja awal (*prototype*) sebelum perangkat lunak final dikembangkan. *Prototype* dapat berupa representasi visual, model fungsional sebagian, atau simulasi interaktif. Tujuannya adalah memvalidasi persyaratan pengguna, mengidentifikasi masalah desain atau fungsionalitas, dan memastikan kesesuaian perangkat lunak dengan kebutuhan. Prosesnya iteratif, dengan *prototype* dievaluasi pengguna dan direvisi pengembang berdasarkan umpan balik.

Terdapat jenis *prototyping* seperti *throwaway* (*prototype* dibuang setelah evaluasi), *evolutionary* (*prototype* ditingkatkan menjadi perangkat lunak *final*), *incremental* (*prototype* dibuat per bagian lalu diintegrasikan), dan *extreme* (khusus web, fokus pada presentasi dan layanan data). Keuntungannya meliputi peningkatan keterlibatan pengguna, pengurangan risiko kesalahan desain dan pengembangan, serta pemahaman persyaratan yang lebih baik. Kekurangannya meliputi potensi ekspektasi tidak realistis dan penundaan proyek jika iterasi berlarut-larut.



Gambar . *Prototype*

## *Tools* Pengembangan Perangkat Lunak

### Bahasa Pemrograman

Dalam pengembangan perangkat lunak, pemilihan bahasa pemrograman yang tepat sangat penting karena dapat mempengaruhi performa, efisiensi pengembangan, dan kompatibilitas dengan platform yang dituju. Bahasa pemrograman yang digunakan harus mampu mendukung kebutuhan aplikasi, baik dari sisi fungsionalitas maupun kemudahan pemeliharaan kode. Pada pengembangan *Scoreboard*, bahasa pemrograman yang digunakan dipilih berdasarkan kemampuannya untuk mendukung pengembangan aplikasi *real-time* yaitu:

1. PHP

PHP (*Hypertext Preprocessor*) adalah bahasa pemrograman *server-side* yang digunakan untuk mengembangkan aplikasi web dinamis. PHP dirancang untuk mempermudah pembuatan halaman web interaktif yang dapat berinteraksi dengan pengguna dan basis data. Dengan PHP, pengembang dapat membuat situs web yang mampu memproses data dari formulir, menghasilkan konten yang berbeda berdasarkan input pengguna, serta mengelola sesi dan *cookies*. Pengembangan *Scoreboard* menggunakan bahasa pemrograman PHP dengan framework Laravel 10.

1. CSS

CSS (*Cascading Style Sheets*) adalah bahasa yang digunakan untuk mengatur tampilan elemen yang ditulis dengan bahasa markup seperti HTML, memisahkan konten dari tampilan visualnya. Diciptakan oleh W3C pada tahun 1996 untuk mengatasi masalah format halaman di HTML yang sebelumnya dilakukan dengan tag HTML, yang membuat kode menjadi panjang dan sulit dikelola. Dengan CSS, pengaturan tampilan seperti *font*, warna latar belakang, dan tata letak dapat dipisahkan ke *file* tersendiri, sehingga kode HTML lebih bersih dan mudah dimaintain. CSS bekerja dengan sintaks sederhana yang terdiri dari selector (memilih elemen HTML) dan declaration block (berisi properti dan nilainya). Ada tiga cara menerapkan CSS: internal (di dalam tag <style>, cocok untuk preview tapi memperlambat *loading* dan tidak bisa dipakai di halaman lain), eksternal (di *file* .css terpisah, paling mudah dan bisa dipakai di banyak halaman, tapi juga memperlambat *loading*), dan *inline* (langsung di dalam tag HTML, cocok untuk perubahan cepat pada satu elemen atau jika tidak ada akses ke *file* CSS, tapi tidak efisien untuk pengaturan tampilan kompleks).

### Platform

Platform dalam pengembangan perangkat lunak adalah lingkungan atau ekosistem tempat perangkat lunak dikembangkan, dijalankan, dan dikelola. Platform dapat berupa perangkat keras (*hardware*), sistem operasi, atau kerangka kerja perangkat lunak yang menyediakan layanan, alat, dan pustaka yang mendukung pengembangan dan eksekusi aplikasi. Platform yang tepat membantu memastikan kompatibilitas, efisiensi, dan skalabilitas perangkat lunak. *Platform* yang digunakan untuk mengembangkan aplikasi *Scoreboard* adalah Laptop dengan OS Windows.

### Database

MySQL adalah sistem manajemen database *open-source* yang populer, menggunakan bahasa SQL. Digunakan oleh lebih dari 66 juta pengguna di seluruh dunia, MySQL tersedia dalam dua lisensi: gratis (GNU GPL) dan berbayar (Shareware). Fungsinya adalah mengelola informasi di database pada sisi server menggunakan SQL. Kelebihannya antara lain gratis, keamanan terjamin, mendukung banyak pengguna (*multi-user*) dan bahasa pemrograman lain (seperti PHP dan JavaScript), tidak membutuhkan RAM besar, mendukung berbagai tipe data, struktur tabel fleksibel, dan telah menjadi standar industri. Kekurangannya adalah kurang cocok untuk aplikasi game dan mobile, kurang optimal untuk database berukuran besar, dan dukungan teknis terbatas untuk versi komunitas (berbayar untuk versi *Enterprise*). Cara kerjanya adalah dengan membuat database yang dapat memodifikasi, menyimpan, dan menentukan keterkaitan antar tabel; kemudian perangkat pengguna membuat permintaan dengan perintah SQL; dan terakhir *server* menerima, menjalankan perintah, dan menampilkan hasilnya.

### IDE

*Integrated Development Environment* (IDE) adalah *software* di dalam lingkungan pengembangan yang berfungsi untuk memfasilitasi berbagai tool pemrograman dalam satu aplikasi. IDE membantu *developer* mengembangkan *software* dengan lebih mudah.

IDE dapat membantu developer untuk meningkatkan produktivitas kerja sebagai kode editor. *Software Editor* yang digunakan dalam pembuatan Scoreboard adalah *Visual Studio Code*.

# METODOLOGI PENELITIAN

## Metode Pengembangan

Metode pengembangan adalah pendekatan sistematis dalam pembuatan perangkat lunak untuk memastikan setiap tahap dilakukan secara terstruktur dan memenuhi kebutuhan. Salah satu metode yang digunakan adalah *Waterfall*, yang mengalir secara berurutan dari analisis kebutuhan, perancangan, implementasi, pengujian, hingga pemeliharaan. Dalam penelitian ini, saya menggunakan metode pengembangan *Waterfall* untuk memastikan proses berjalan terorganisasi dengan baik.

### Waterfall

Metode *Waterfall* yang digunakan dalam pembuatan aplikasi web *Scoreboard* berasaskan *software engineering* yang menganut pendekatan berurutan. Metode ini memiliki tahapan yang terstruktur, di mana pada setiap fase harus terpenuhi persyaratan sepenuhnya sebelum masuk ke fase berikutnya. Ada setiap tahap memiliki *output* yang kemudian menjadi *input* bagi tahap selanjutnya, sehingga alur kerja terorganisir dan sistematis. Proses Pengembangan metode *Waterfall* dalam pembuatan aplikasi web *Scoreboard* adalah sebagai berikut:

1. *Analysis* (Analisis)

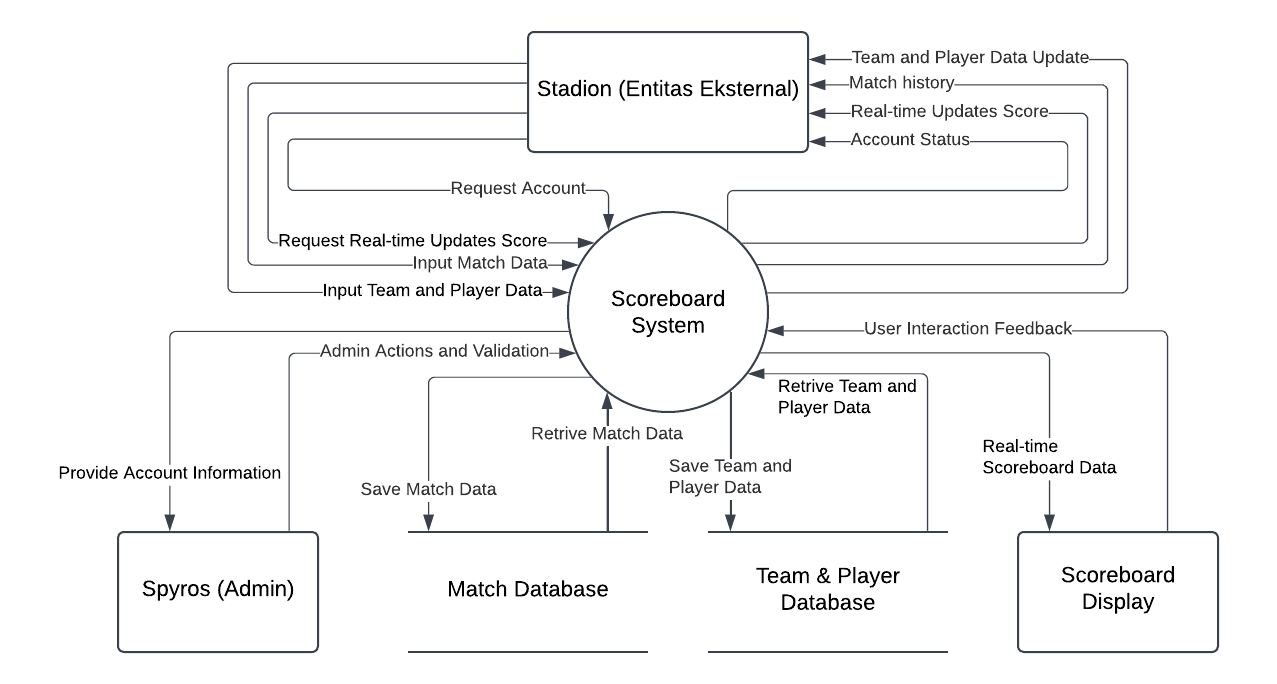
Pada tahap analisis, kami mendefinisikan kebutuhan scoreboard secara detail. Ini mencakup fungsi-fungsi yang harus dimiliki scoreboard, beberapa kebutuhan pada stadion dan juga *design* spada aplikasi *scoreboard*.

1. *Design* (Perancangan)

Tahap perancangan menerjemahkan kebutuhan yang telah dianalisis ke dalam desain sistem. Digunakan beberapa diagram untuk memvisualisasikan sistem:

1. *Data Flow Diagram* (DFD)

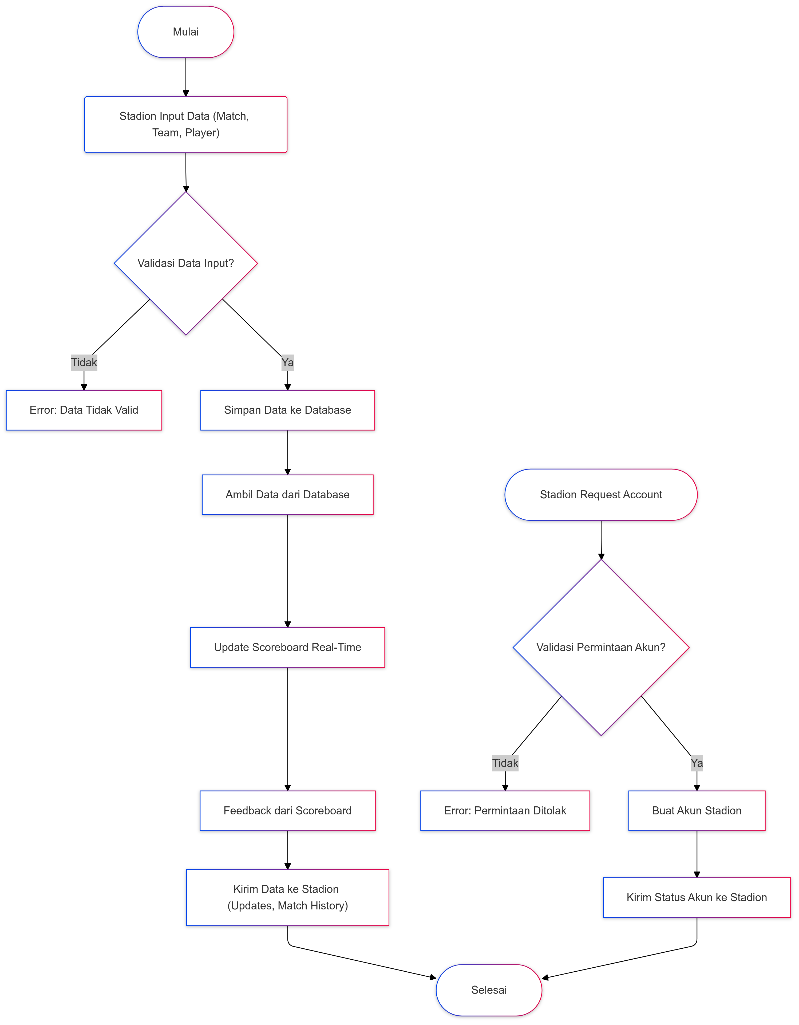
DFD menggambarkan aliran data utama antara sistem dan entitas eksternal.



Gambar . *Data Flow Diagram* (DFD) *Level* 0 *Scoreboard*

1. Flowchart

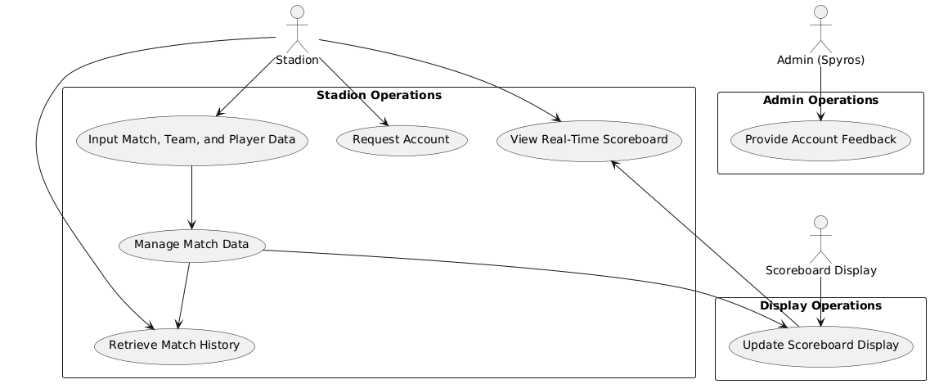
Flowchart memvisualisasikan alur proses dalam sistem, seperti input skor dan pembaruan tampilan.



Gambar . Flowchart *Scoreboard*

1. UML
   1. Usecase Diagram

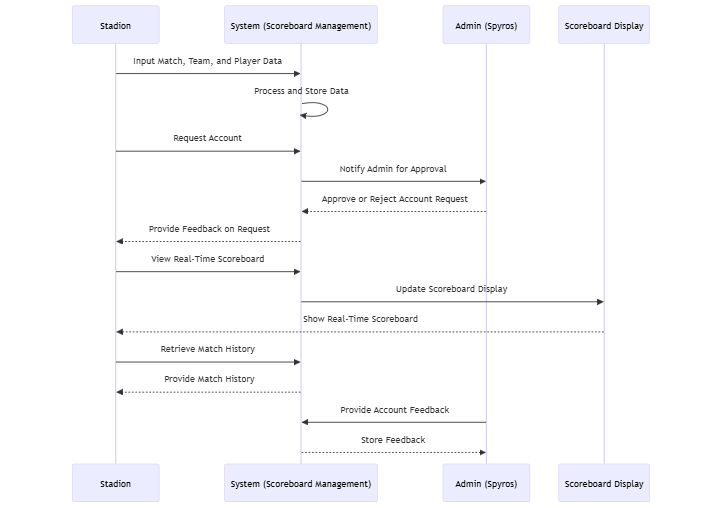
Use Case diagram untuk menggambarkan interaksi pengguna dengan sistem.



Gambar . Usecase Diagram Scoreboard

* 1. Sequence Diagram

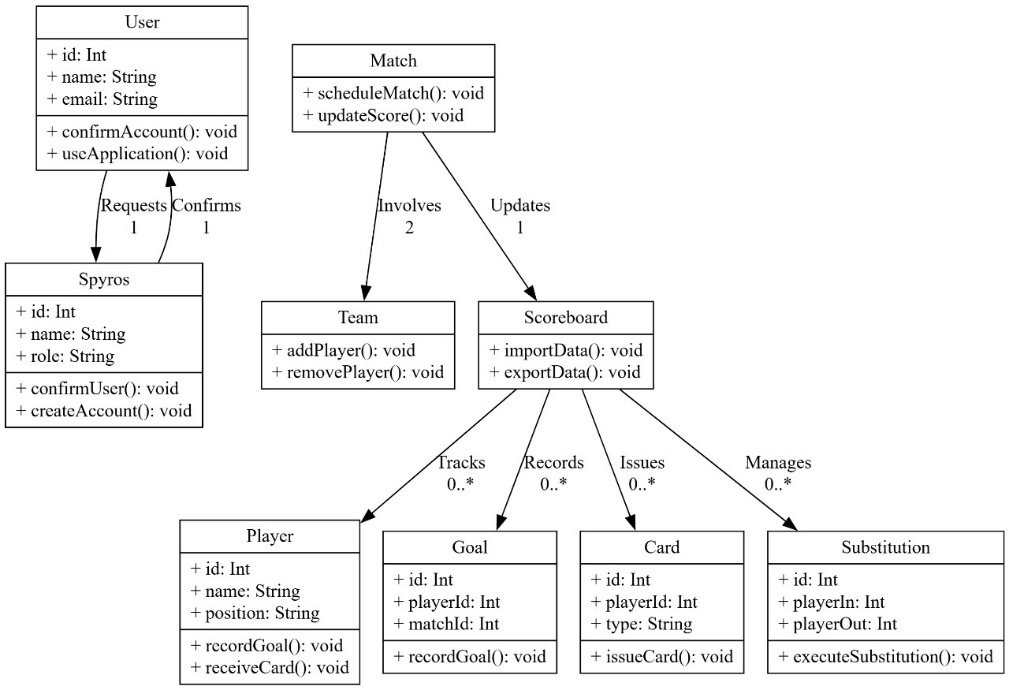
Sequence Diagram untuk urutan interaksi antar objek.



Gambar . Sequence Diagram *Scoreboard*

* 1. Class Diagram

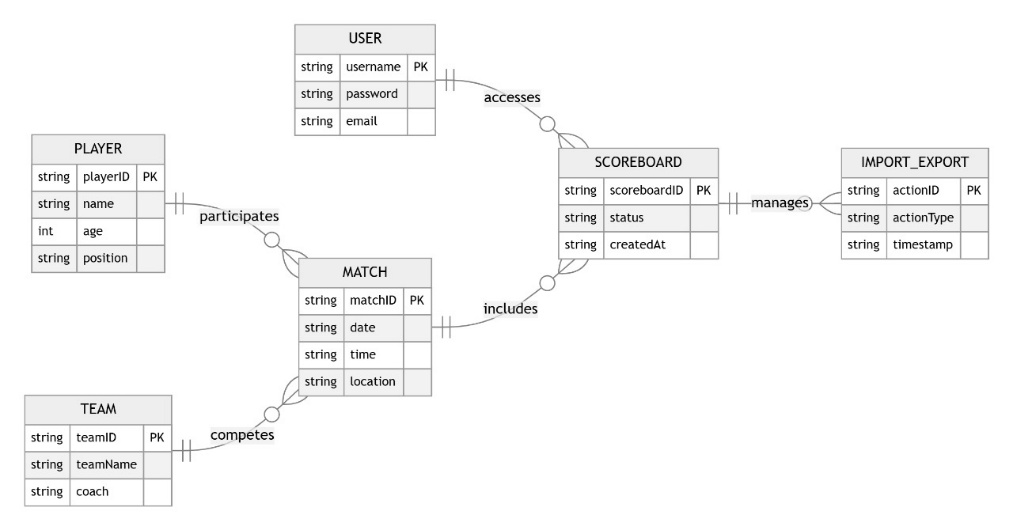
Class Diagram untuk struktur kelas dan relasinya.



Gambar . Class Diagram

1. *Entity Relationship Diagram* (ERD)

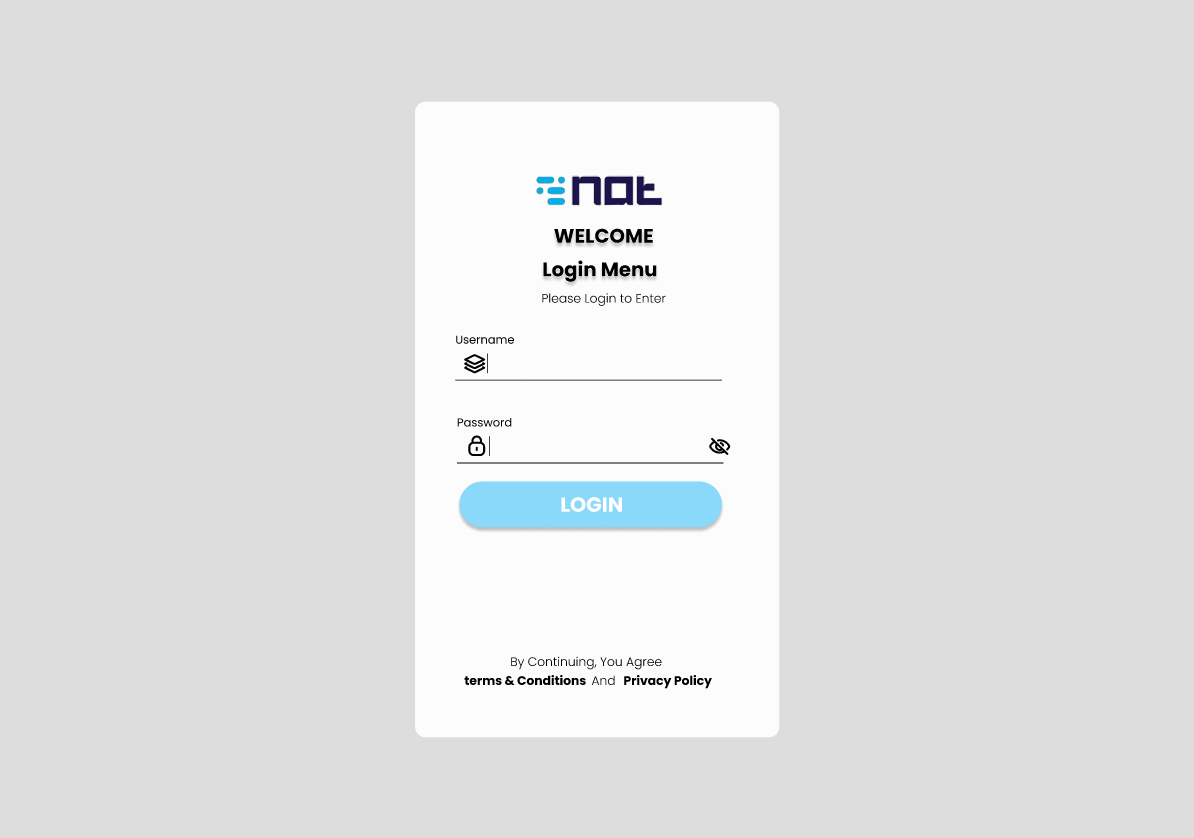
ERD memodelkan struktur basis data yang akan digunakan.



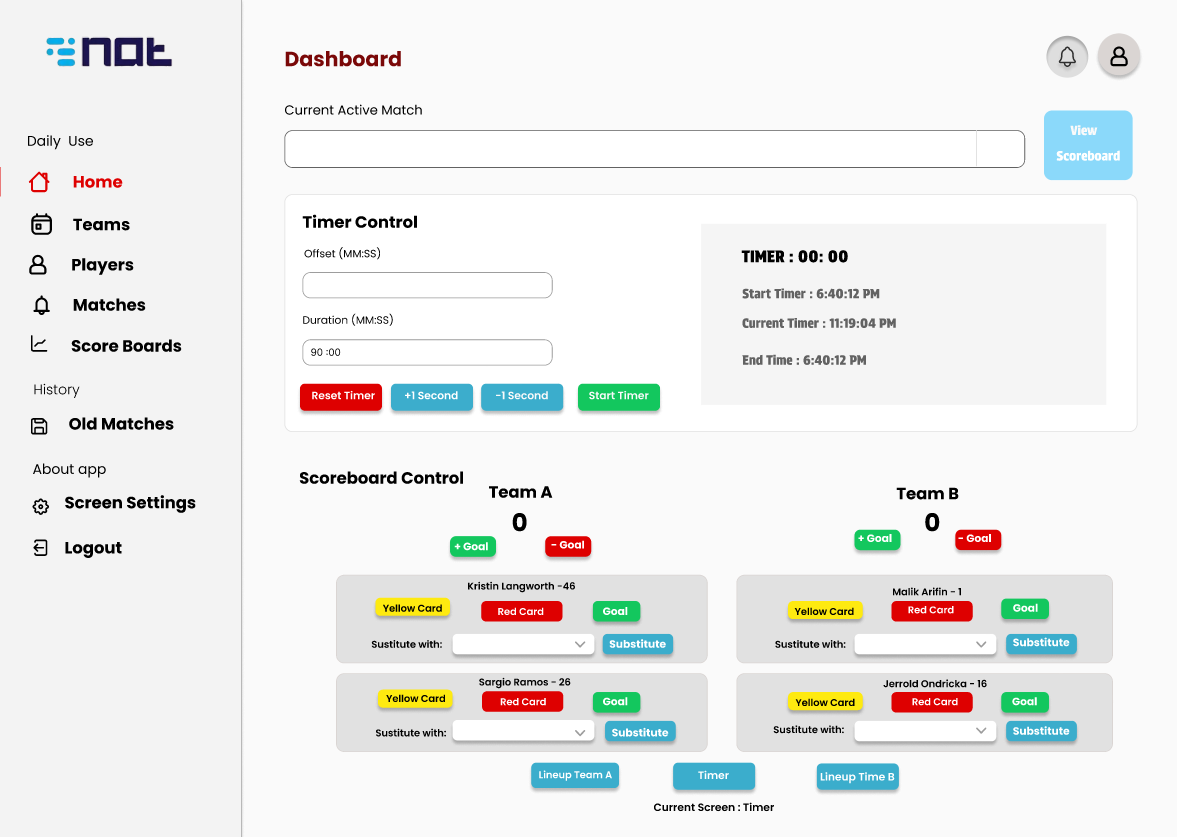
Gambar . *Entity Relationship Diagram* (ERD)

1. Mockup

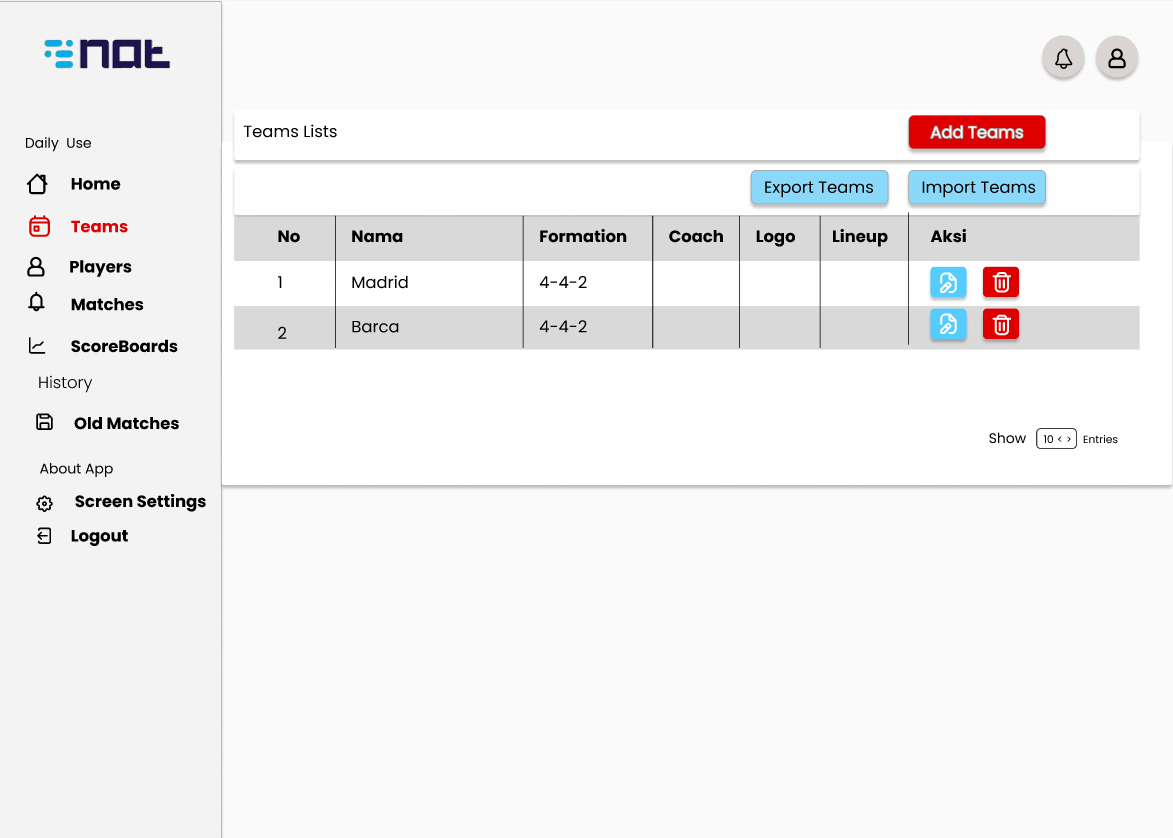
Mockup dibuat untuk merepresentasikan tampilan visual scoreboard.



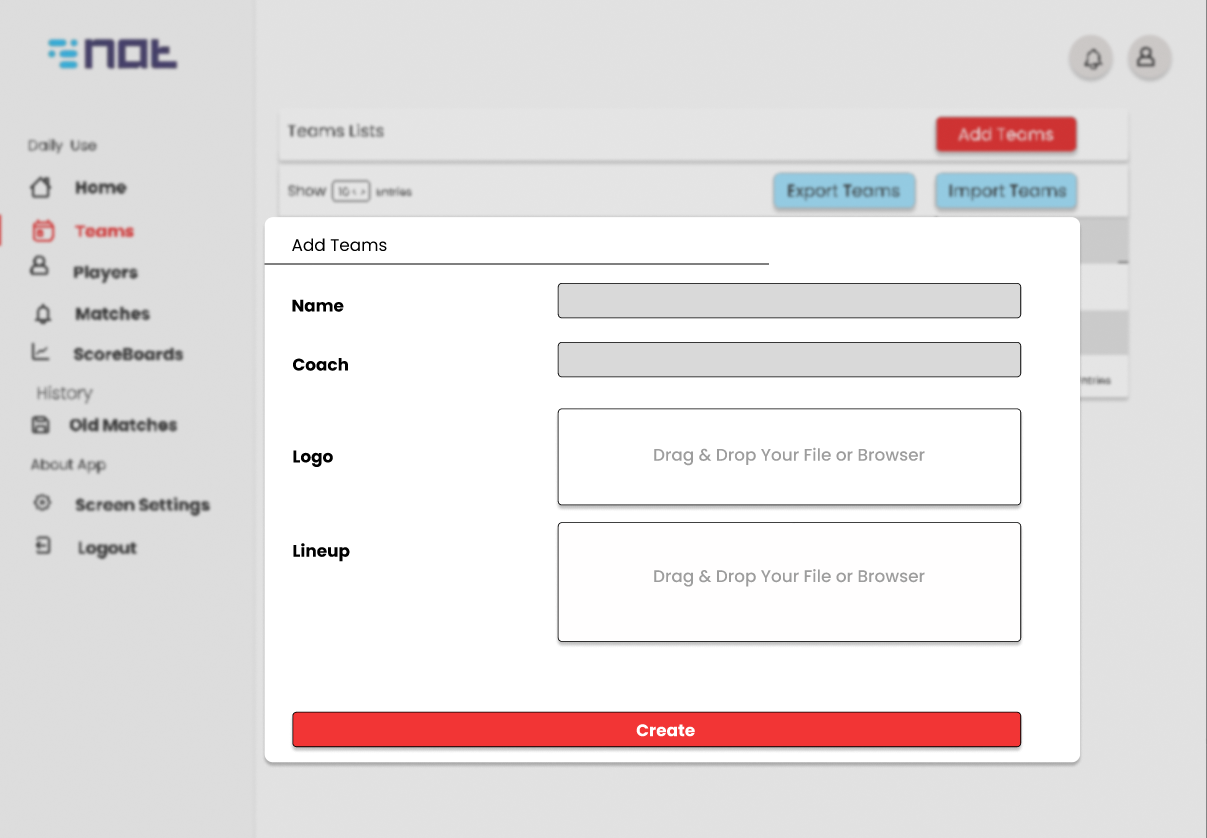
Gambar . Mock up Halaman *Login*



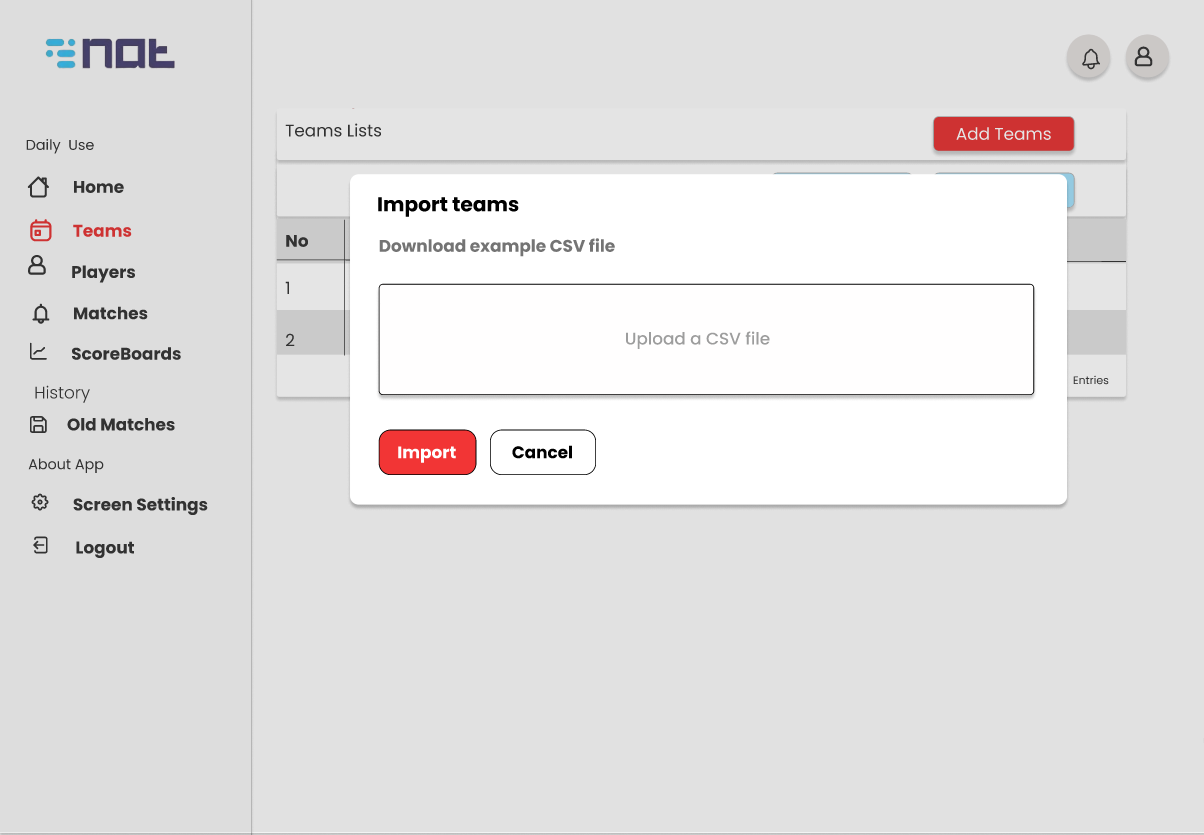
Gambar . Mock up Halaman *Home*



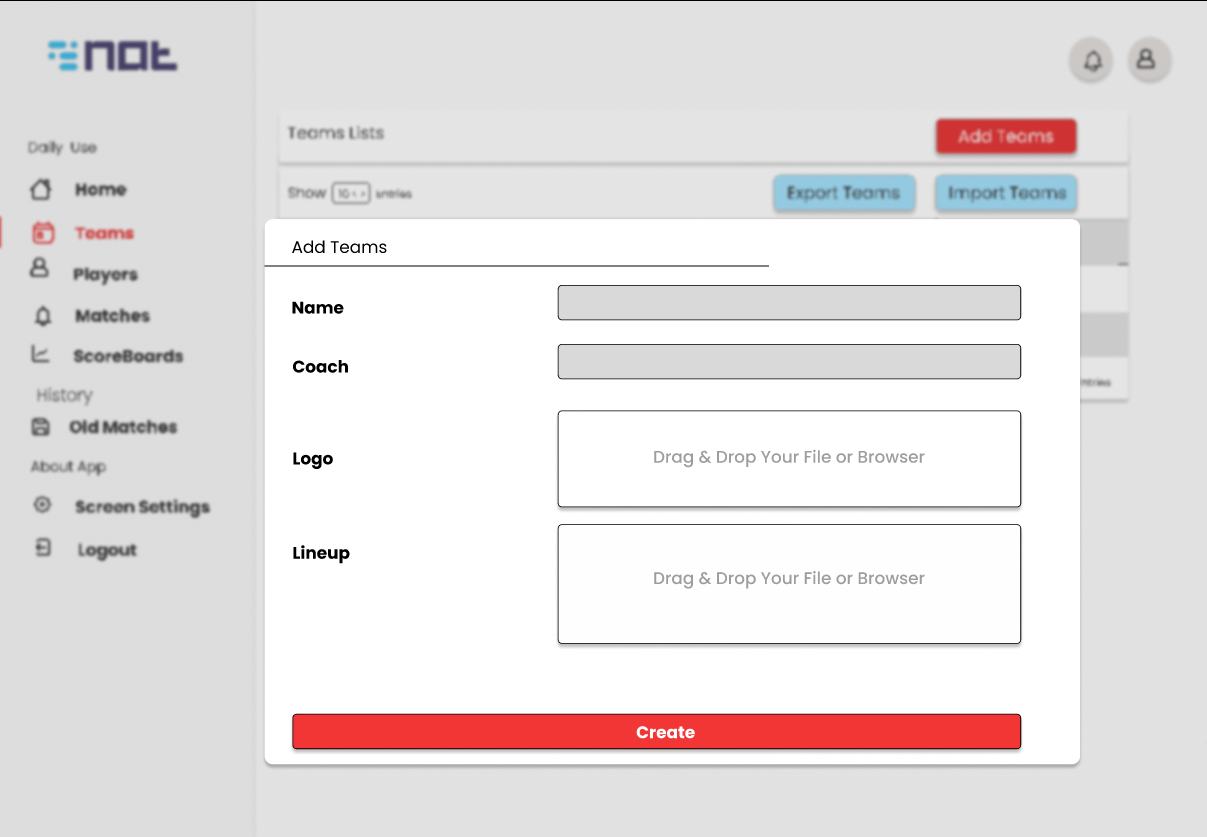
Gambar . Mock up Halaman *Teams*



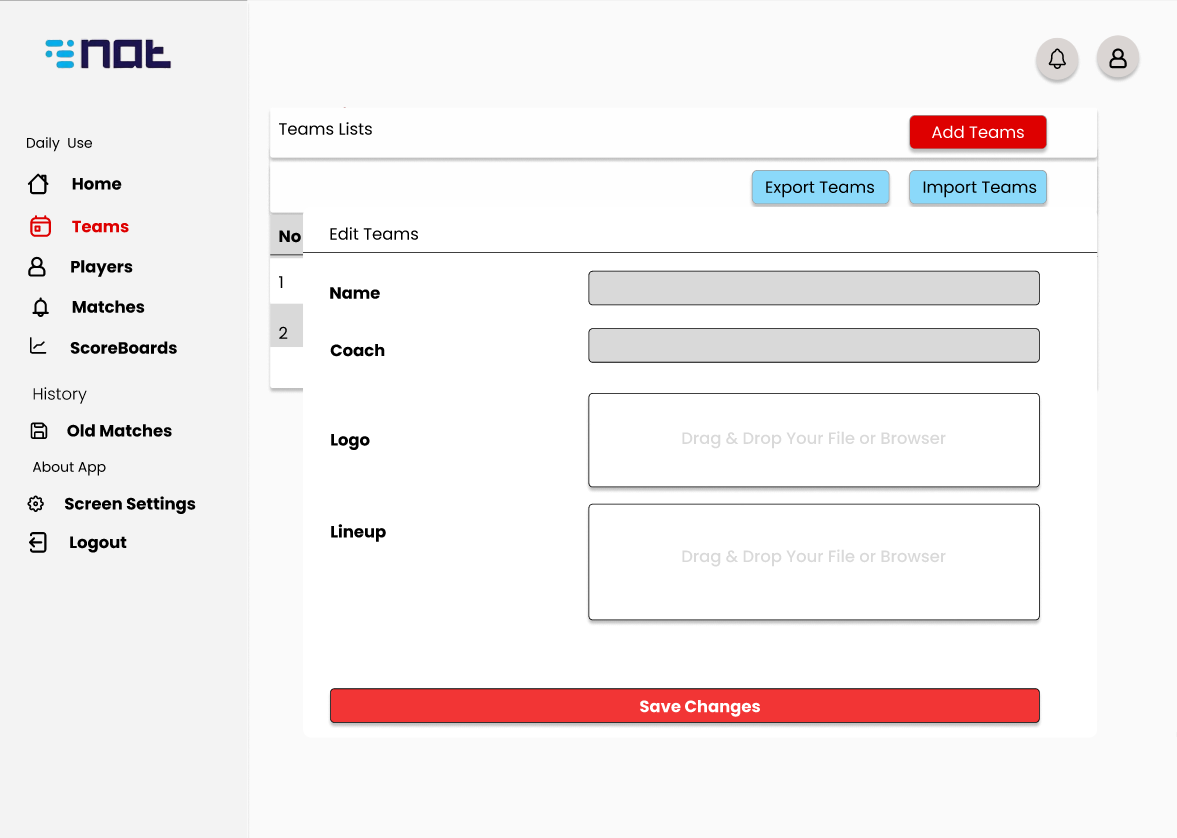
Gambar . Mock up Tambah *Teams Dialog*



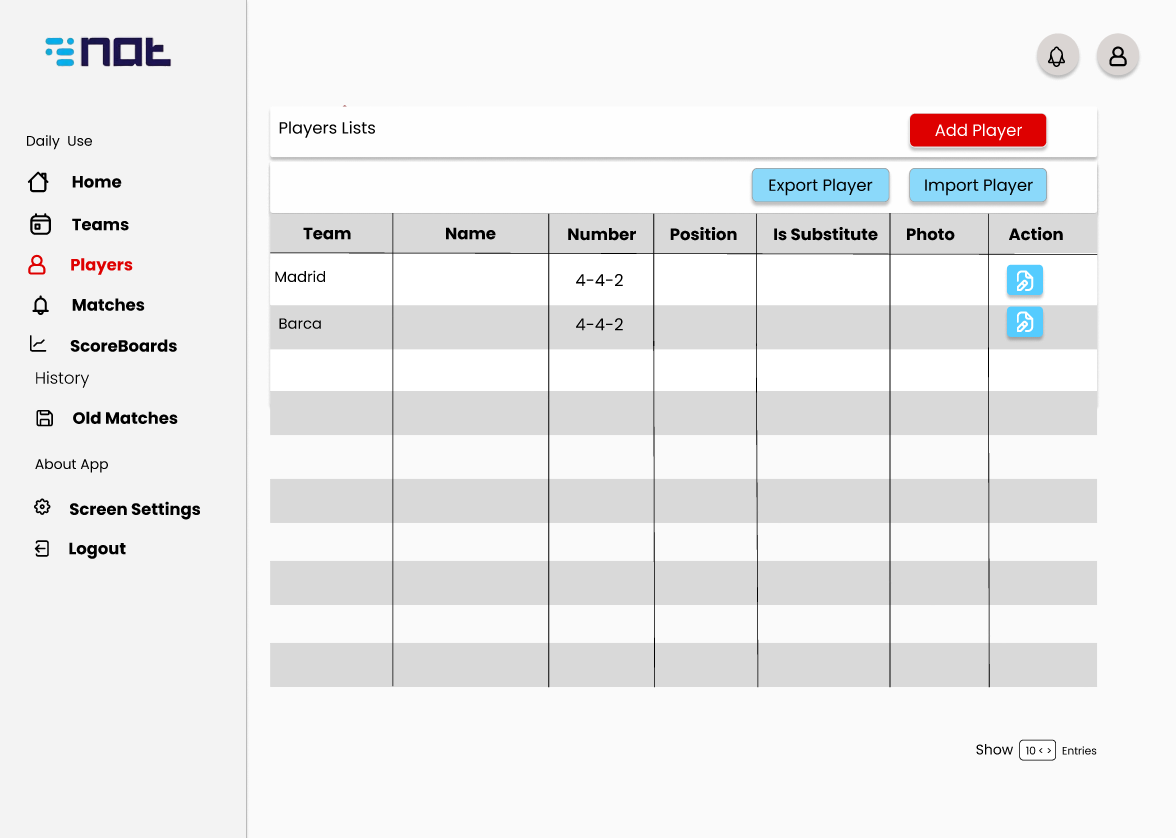
Gambar . Mock up Import *Teams*



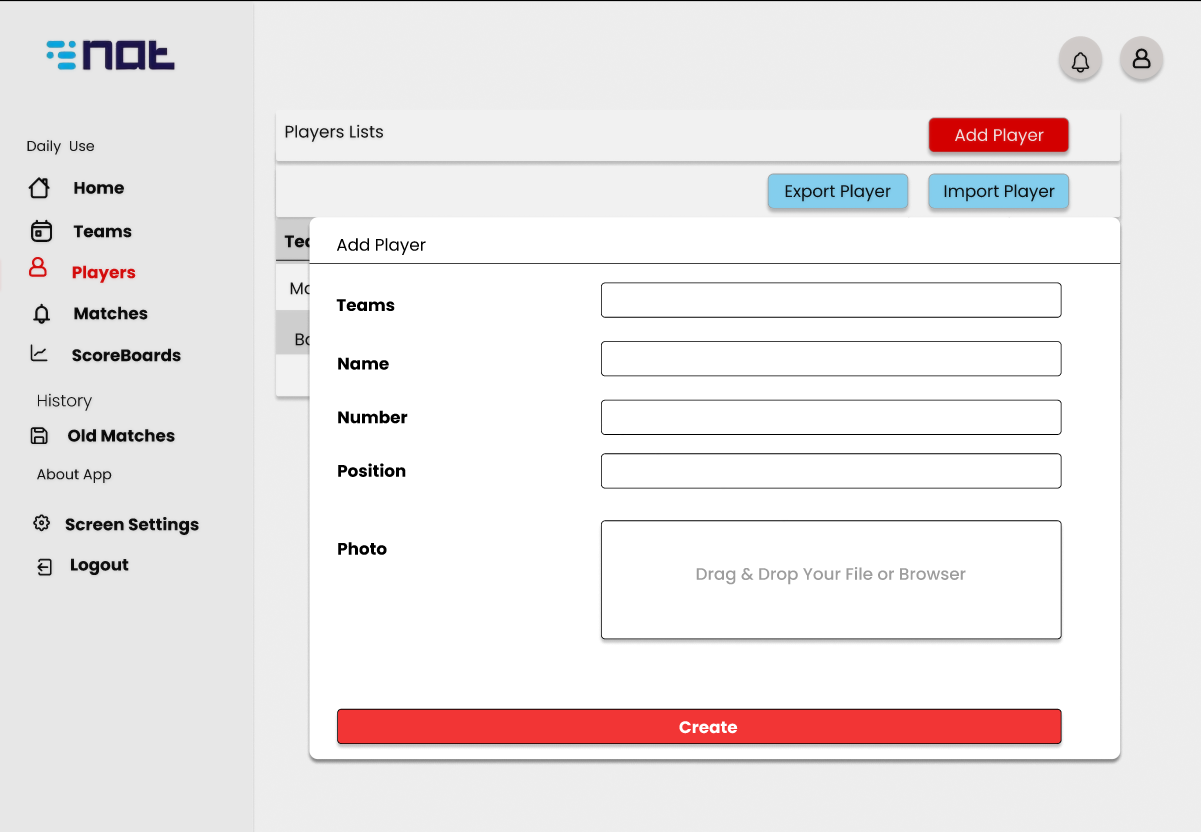
Gambar . Mock up Tambah *Teams Dialog*



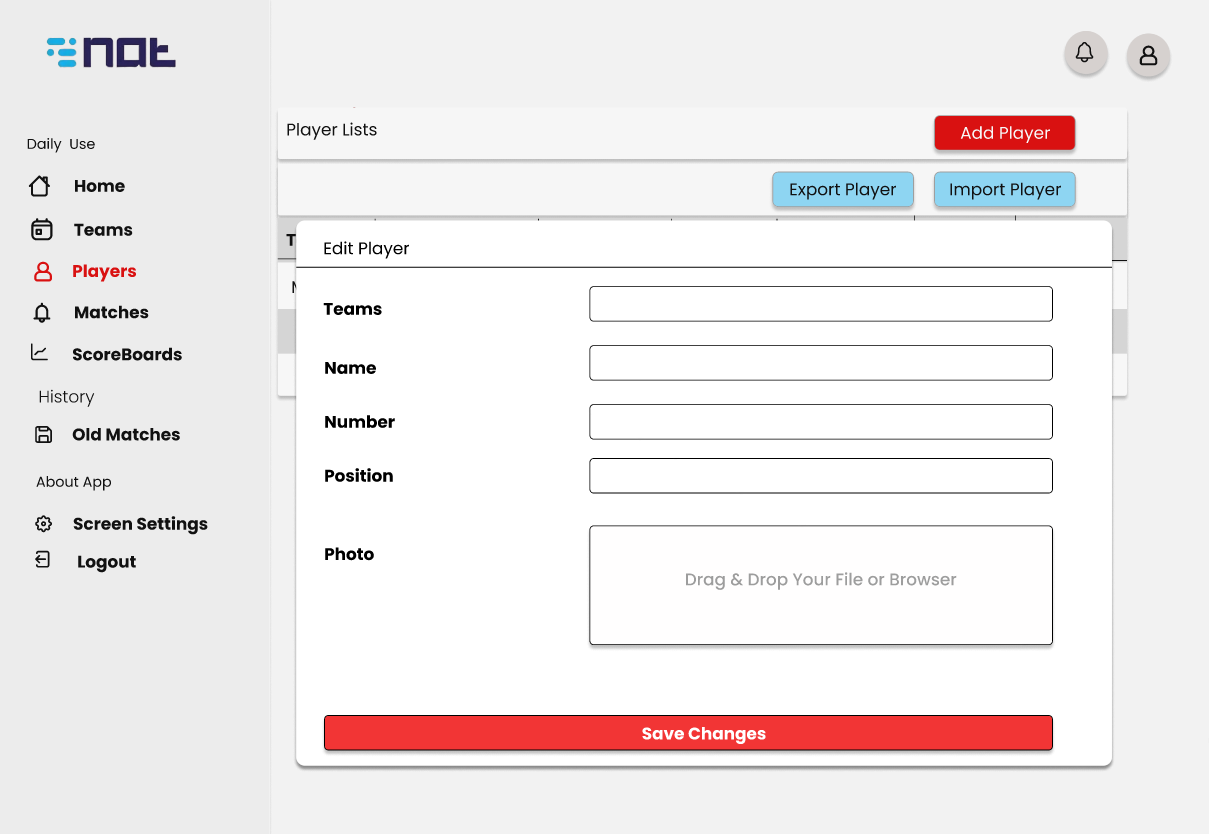
Gambar . Mock up Edit *Teams Dialog*



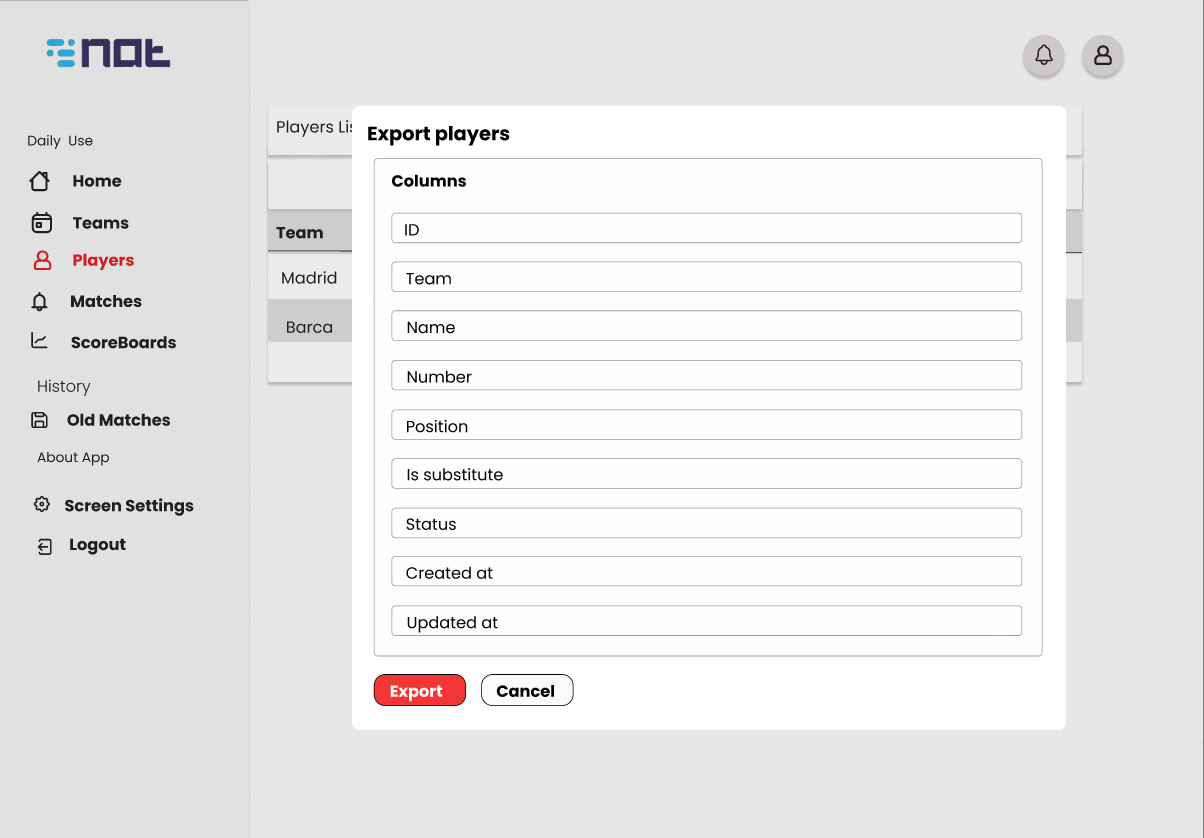
Gambar . Mock up Halaman *Players*



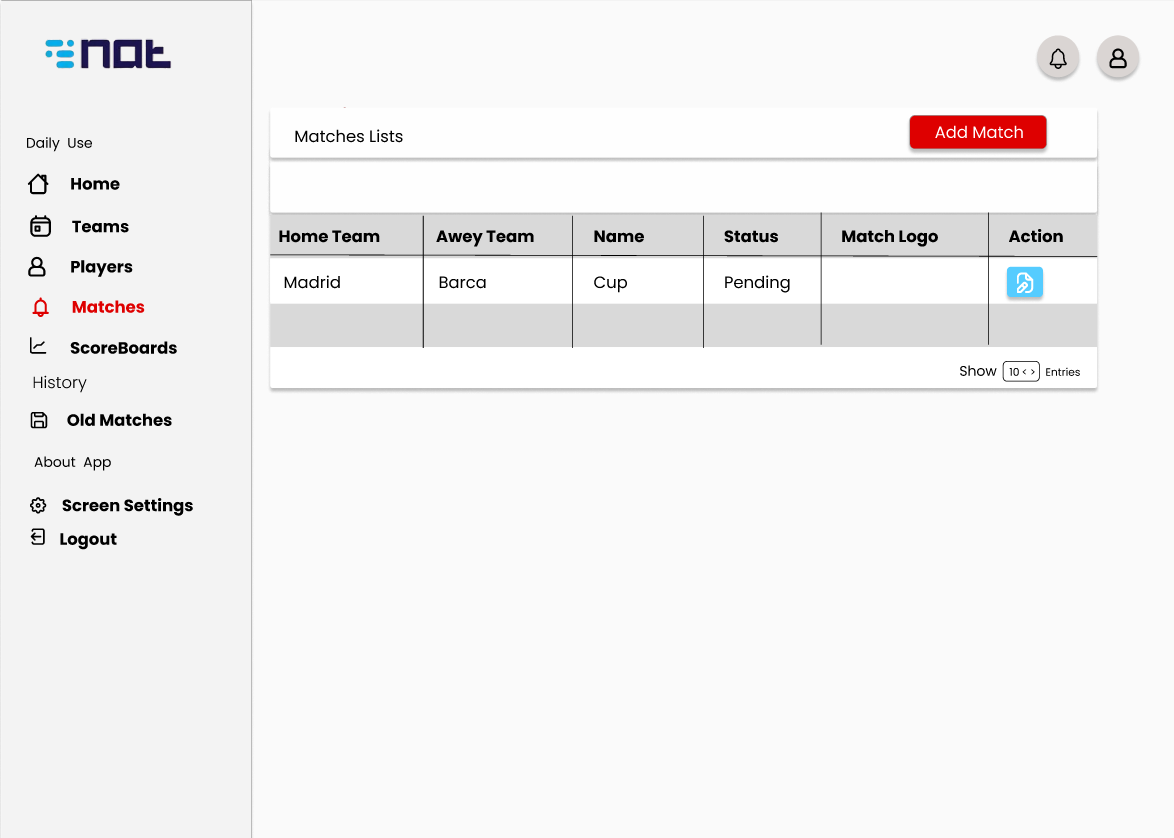
Gambar . Mock up Tambah *Players Dialog*



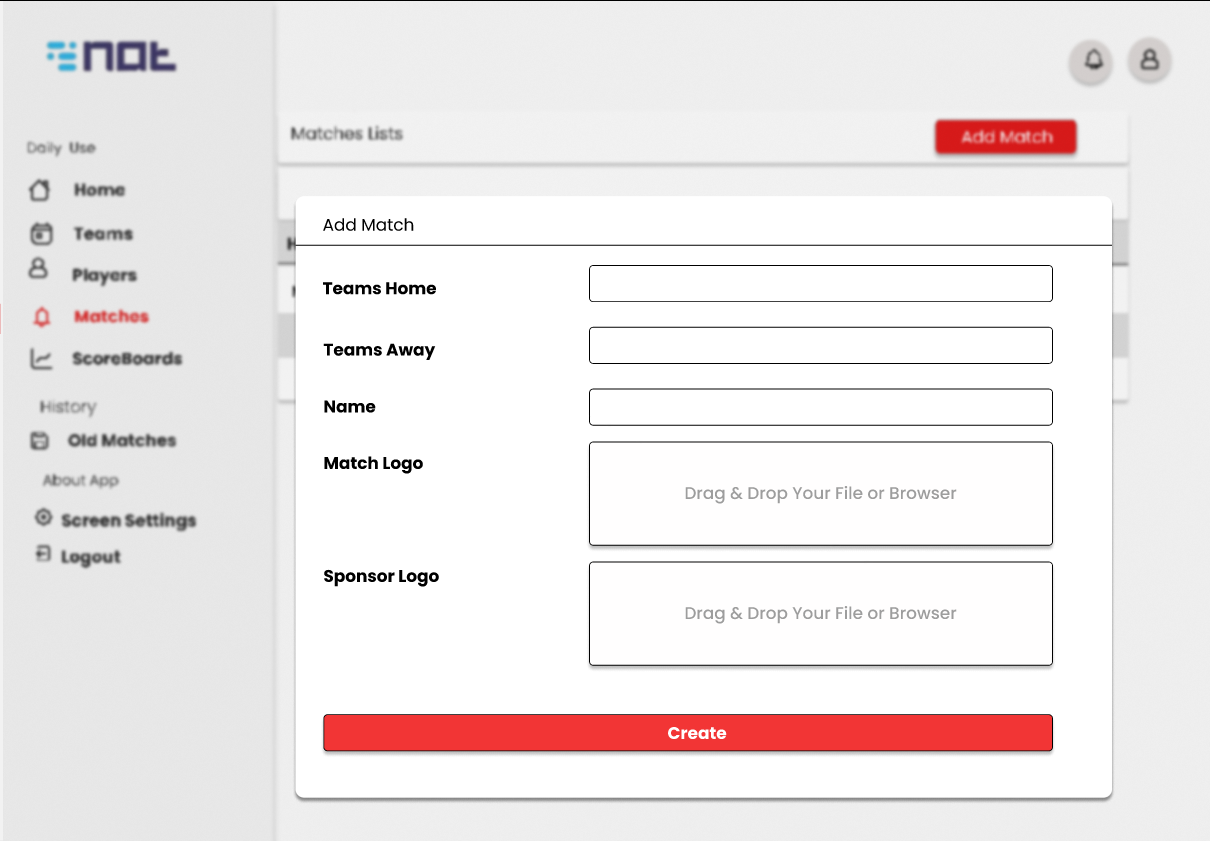
Gambar . Mock up Edit *Player Dialog*



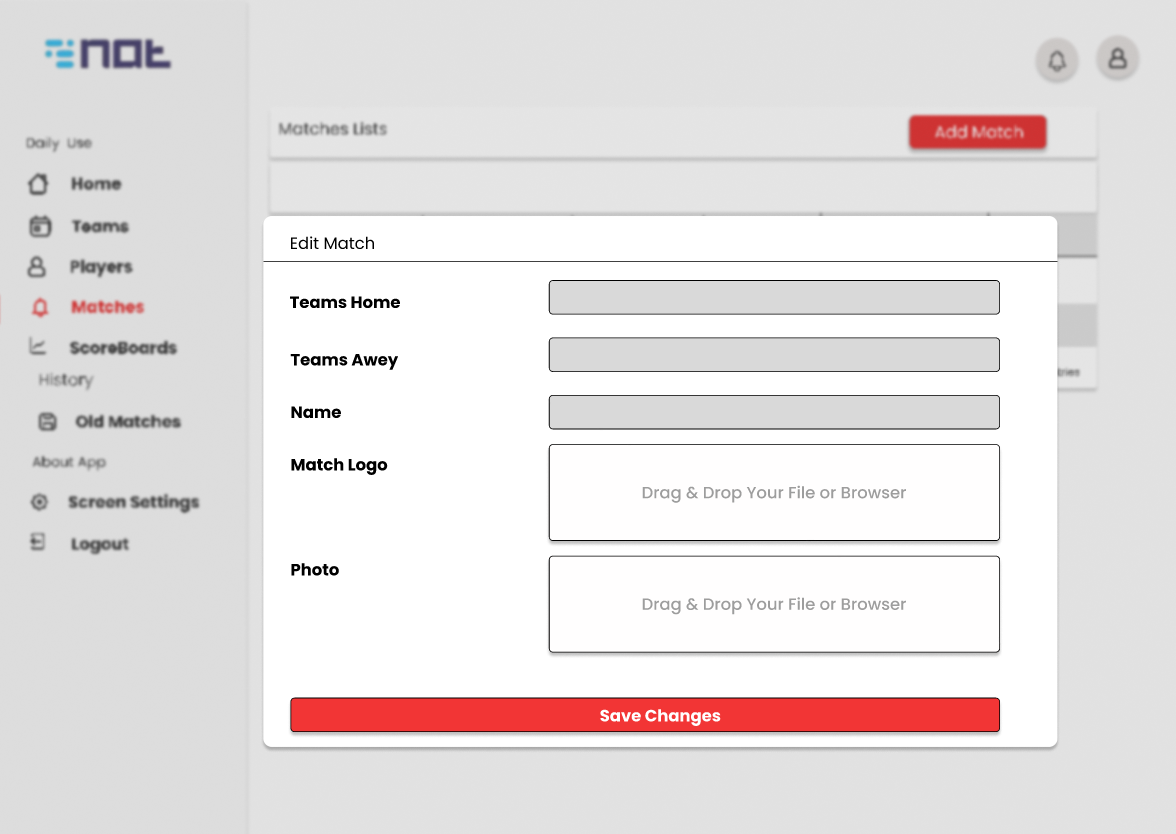
Gambar . Mock up *Export Player Dialog*



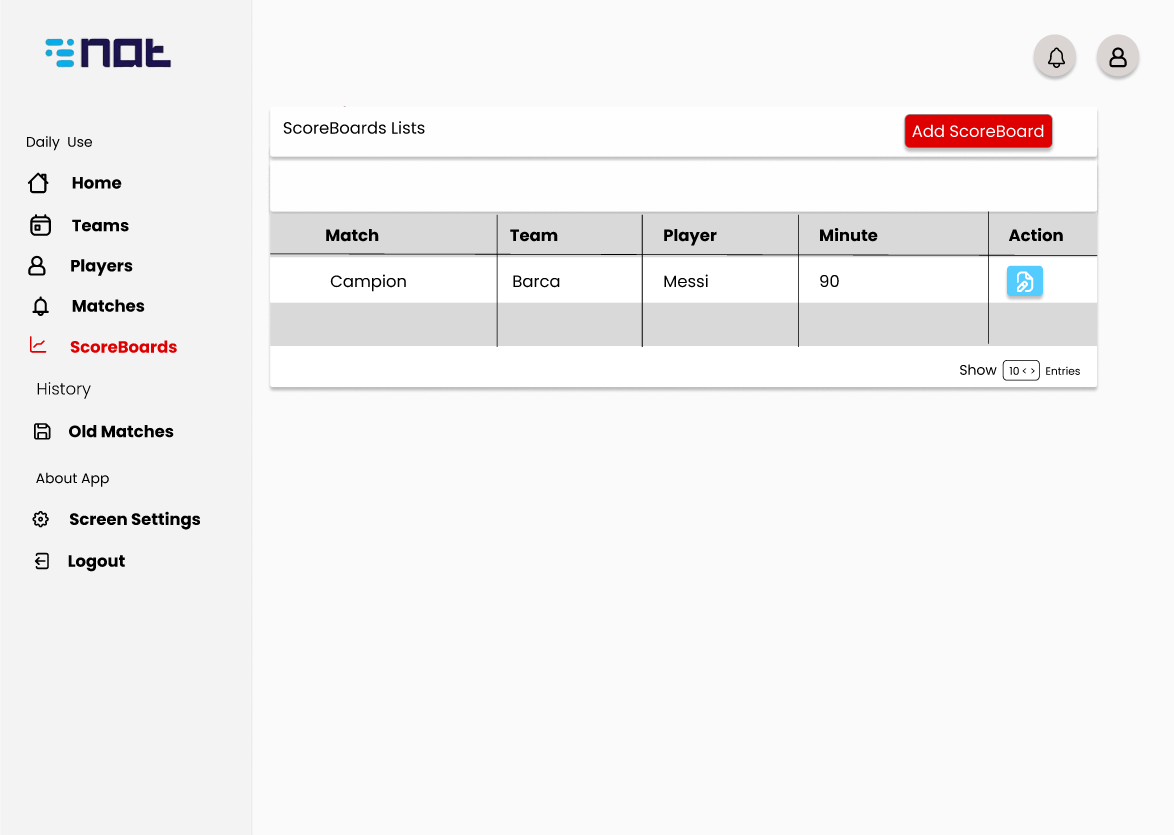
Gambar . Mock up Halaman *Matches*



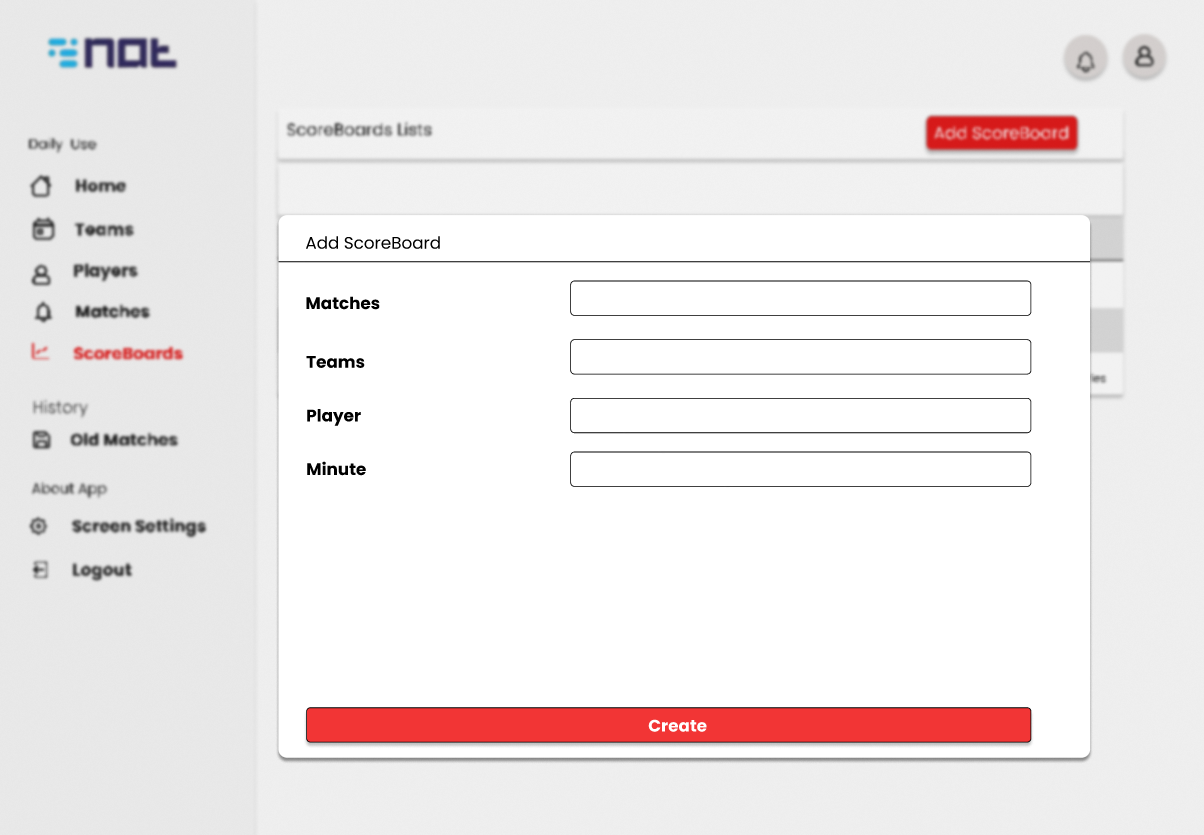
Gambar . Mock up Tambah *Matches Dialog*



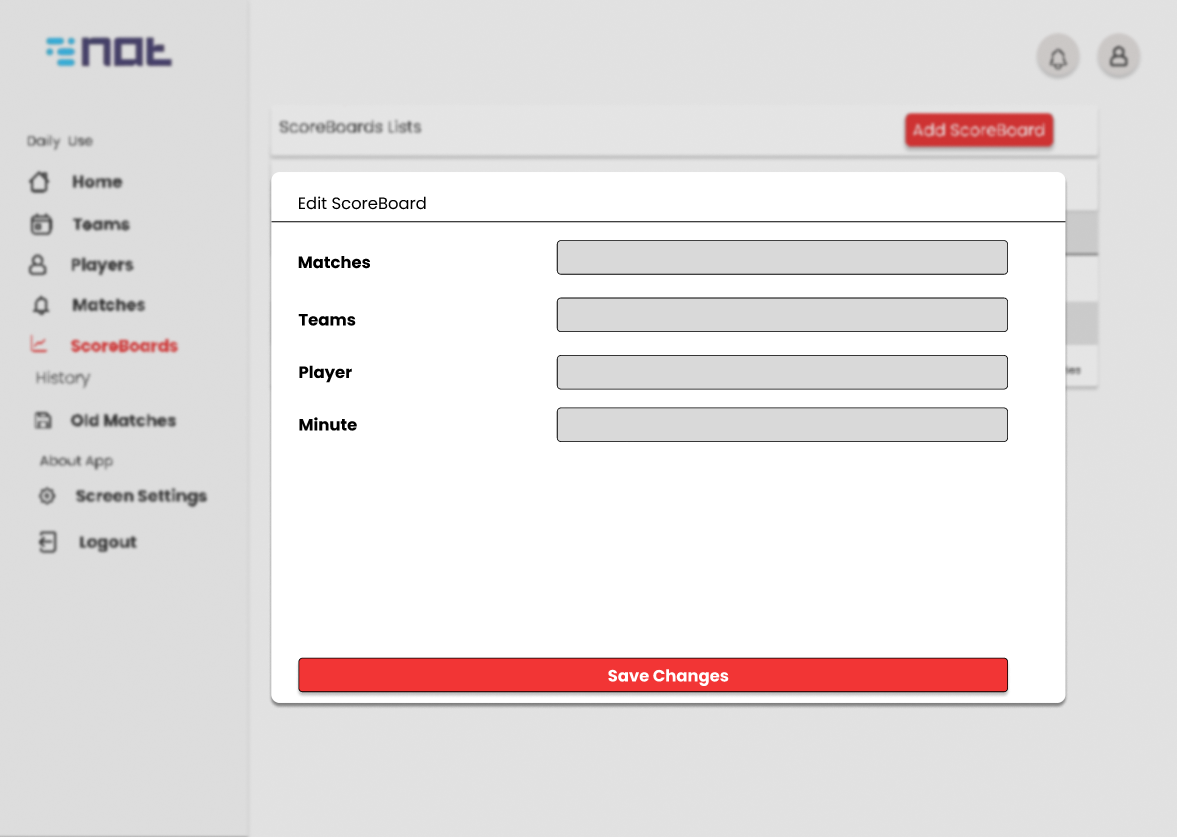
Gambar . Mock up Edit *Match Dialog*



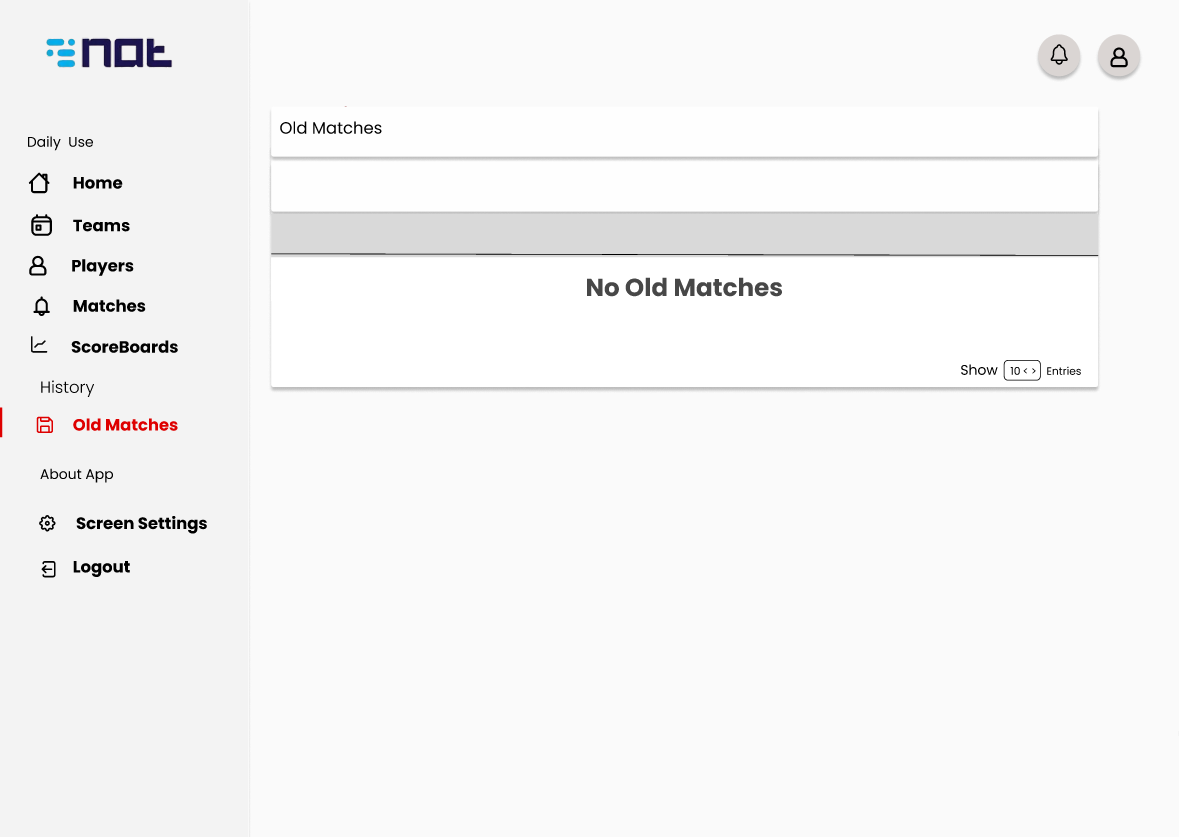
Gambar . Mock up Halaman *Scoreboards*



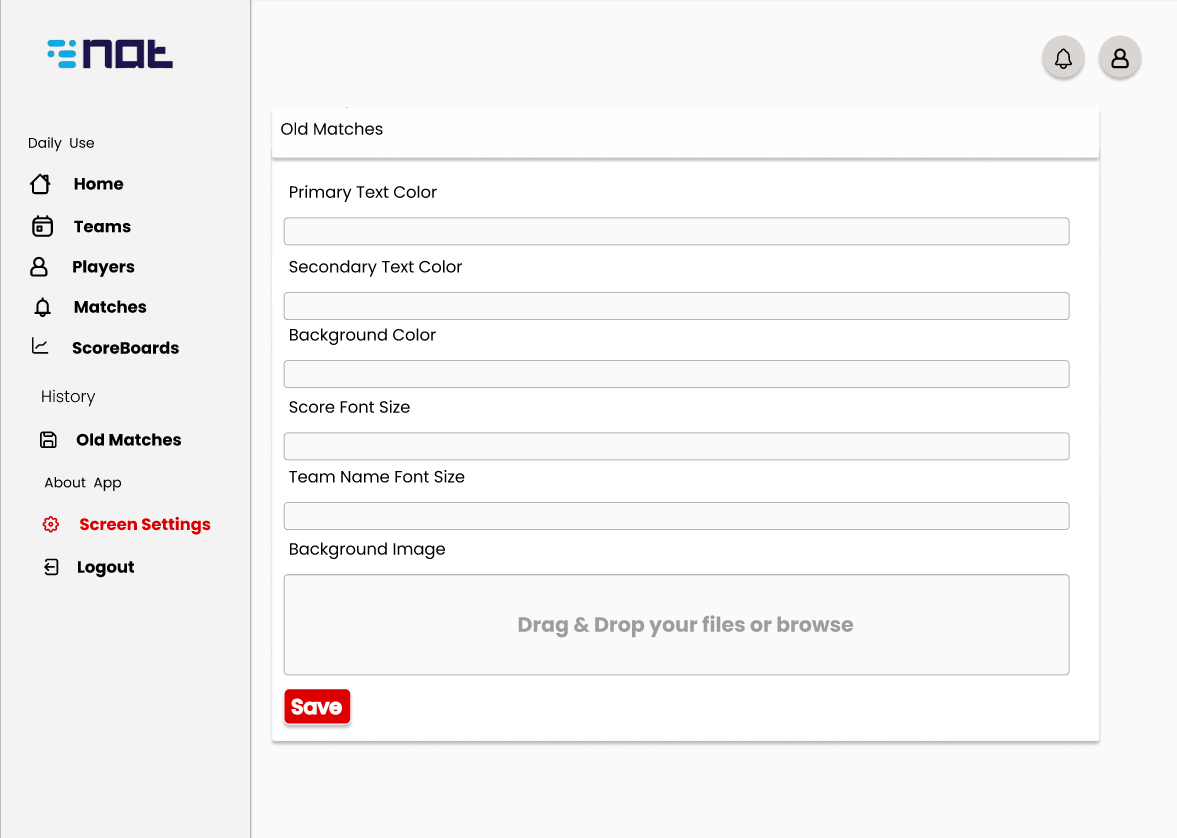
Gambar . Mock up Tambah *Scoreboard Dialog*



Gambar . Mock up Edit *Scoreboard Dialog*



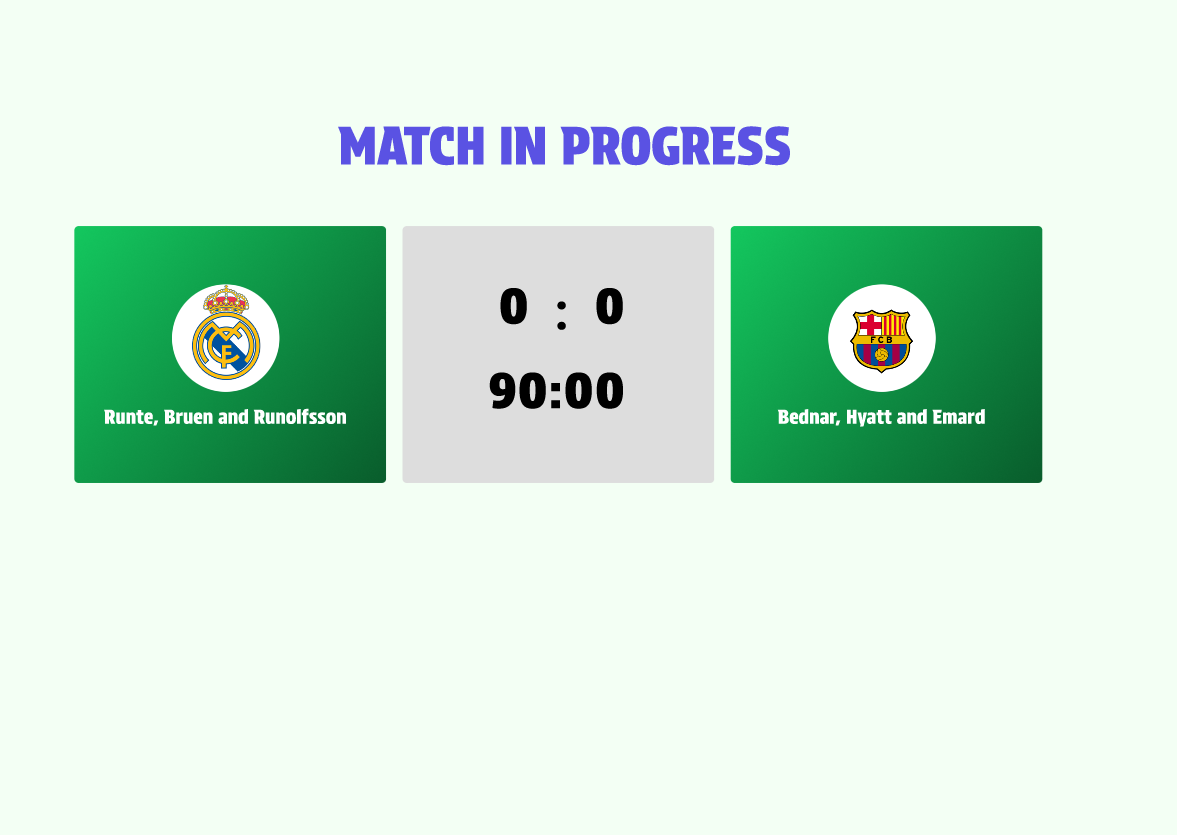
Gambar . Mock up Halaman *Old Matches*



Gambar . Mock up Halaman *Screen Settings*



Gambar . *Display* *Line - up Team*



Gambar . *Display Timer Match*



Gambar . *Display* Pergantian Pemain



Gambar . *Display* Goal



Gambar . *Display* Pelanggaran Kartu Kuning



Gambar . *Display* Pelanggaran Kartu Merah

1. *Coding* (Pengkodean)

Pada tahap pengkodean, tim pengembang mengimplementasikan desain yang telah dibuat ke dalam kode program. Bahasa pemrograman dan framework yang dipilih disesuaikan dengan kebutuhan proyek. Tahap ini fokus pada penulisan kode yang bersih, efisien, dan sesuai dengan standar yang telah ditetapkan.

1. *Testing* (Pengujian)

Setelah pengkodean selesai, *scoreboard* diuji secara menyeluruh. Tujuan pengujian adalah untuk menemukan dan memperbaiki bug atau kesalahan sebelum *scoreboard* diimplementasikan.

1. *Operation/Maintenance* (Operasi/Pemeliharaan)

Tahap terakhir adalah implementasi dan operasi *scoreboard*. Setelah diimplementasikan, *scoreboard* dipantau dan dipelihara secara rutin. Pemeliharaan ini mencakup perbaikan bug yang mungkin muncul, peningkatan fitur jika diperlukan, dan memastikan scoreboard tetap berfungsi dengan baik dan relevan dengan kebutuhan pengguna seiring waktu.

## Jadwal Pengembangan Aplikasi

Tabel . Jadwal Pengembangan Aplikasi Scoreboard



# PEMBAHASAN

## Aplikasi

## Alat dan Bahan

### Alat

Berikut adalah alat yang digunakan untuk pengembangan dan penggunaan aplikasi Scoreboard Spyros:

1. Laptop

Laptop digunakan sebagai perangkat utama dalam proses pengembangan aplikasi *Scoreboard* Spyros. Perangkat ini memungkinkan tim pengembang untuk menulis, menguji, dan menjalankan aplikasi secara efisien. Dengan spesifikasi yang memadai, laptop mendukung pemrosesan kode, desain UI/UX, serta pengelolaan versi kode dalam proyek.

1. *Integrated Development Environtment* (IDE)

IDE digunakan sebagai alat utama untuk menulis, mengedit, dan mengelola kode program dalam pengembangan aplikasi Scoreboard Spyros. IDE yang digunakan dalam proyek ini adalah Visual Studio Code, yang menawarkan berbagai fitur seperti debugging, integrasi dengan Git, serta dukungan ekstensi untuk meningkatkan produktivitas pengembang.

1. Perangkat Lunak Desain

Perangkat lunak desain digunakan untuk merancang antarmuka pengguna (*User Interface*) serta meningkatkan pengalaman pengguna (*User Experience*) dalam aplikasi *Scoreboard* Spyros. Dalam proyek ini, tim menggunakan Figma sebagai alat utama dalam proses desain. Figma memungkinkan kolaborasi *real-time*, *prototyping* interaktif, dan pengelolaan elemen desain secara terstruktur untuk memastikan tampilan aplikasi yang intuitif dan menarik.

1. Sistem Kontrol Versi

Sistem kontrol versi digunakan untuk mengelola perubahan kode secara terstruktur serta memungkinkan kolaborasi antar anggota tim dalam pengembangan aplikasi Scoreboard Spyros. Proyek ini menggunakan Git sebagai sistem kontrol versi, yang memungkinkan pengembang untuk melacak perubahan kode, mengelola cabang pengembangan, serta mengatasi konflik dalam pengembangan bersama. Untuk penyimpanan repositori dan kolaborasi secara online, tim menggunakan GitHub, yang menyediakan fitur seperti *pull request*, *issue tracking*, serta *continuous integration* untuk mendukung pengembangan aplikasi yang lebih efisien.

### Bahan

1. *Framework*

Dalam pengembangan aplikasi *Scoreboard* Spyros, digunakan Laravel sebagai *framework* utama pada sisi *Back-End*, yang menyediakan berbagai fitur penting seperti manajemen API, autentikasi pengguna, dan interaksi dengan *database*. Untuk mempermudah pembuatan dan pengelolaan antarmuka administrasi, digunakan Filament sebagai admin panel berbasis Laravel. Sedangkan, pada sisi *Front-End*, digunakan Vite sebagai *build tool* yang berfungsi untuk mempercepat proses pengembangan dan optimasi performa aplikasi.

1. *Package*

Beberapa *package* yang digunakan dalam pengembangan aplikasi ini adalah filament/filament untuk pembuatan admin *panel*, sentry/sentry-laravel untuk *monitoring* dan *error tracking*, serta fakerphp/faker untuk pembuatan data *dummy* yang diperlukan dalam pengujian aplikasi. Di sisi *Front-End*, tailwindcss digunakan sebagai *framework* CSS untuk mempercepat pembuatan antarmuka pengguna yang responsif, sedangkan laravel-vite-plugin menghubungkan Laravel dengan Vite untuk memaksimalkan *build performance* aplikasi.

1. *Database*

Untuk penyimpanan dan pengelolaan data aplikasi, digunakan MySQL sebagai sistem manajemen basis data. Pemilihan MySQL didasarkan pada kemampuannya dalam menangani transaksi dan skalabilitas yang tinggi. Laravel Eloquent ORM digunakan sebagai alat bantu untuk mempermudah interaksi dengan *database*, memungkinkan pengelolaan data dengan cara yang lebih efisien dan terstruktur menggunakan model-model yang sesuai dengan kebutuhan aplikasi.

## Langkah Pembuatan Aplikasi

Dalam pengembangan aplikasi berbasis web ini, Framework Laravel Filament digunakan untuk pengelolaan backend dan frontend aplikasi ini.

1. Instalasi Laravel

Install Laravel sebagai framework utama untuk backend. Pastikan Laravel berhasil diinstal dan berjalan di localhost.

1. Penambahan Laravel Filament dan Package

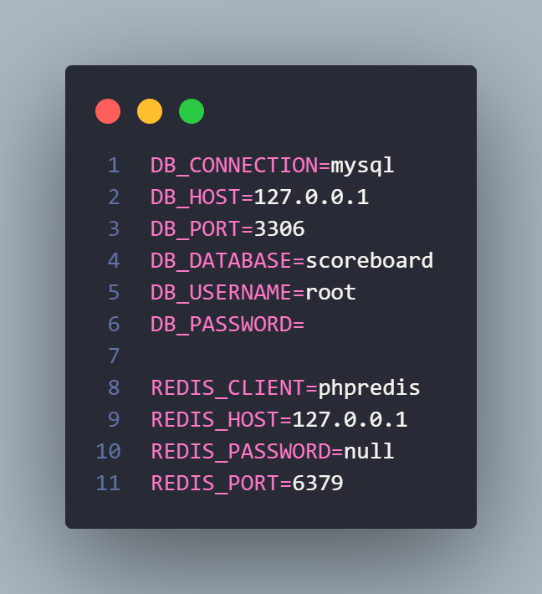
Install Laravel Filament dan package yang dibutuhkan.



Gambar . List *package* pada *file* composer.json

1. Konfigurasi Database dan Redis

Atur koneksi database dan redis pada file .env. Buat database yang diperlukan menggunakan MySql.



Gambar . Konfigurasi *database* dan redis pada *file* .env

1. Pembuatan Model

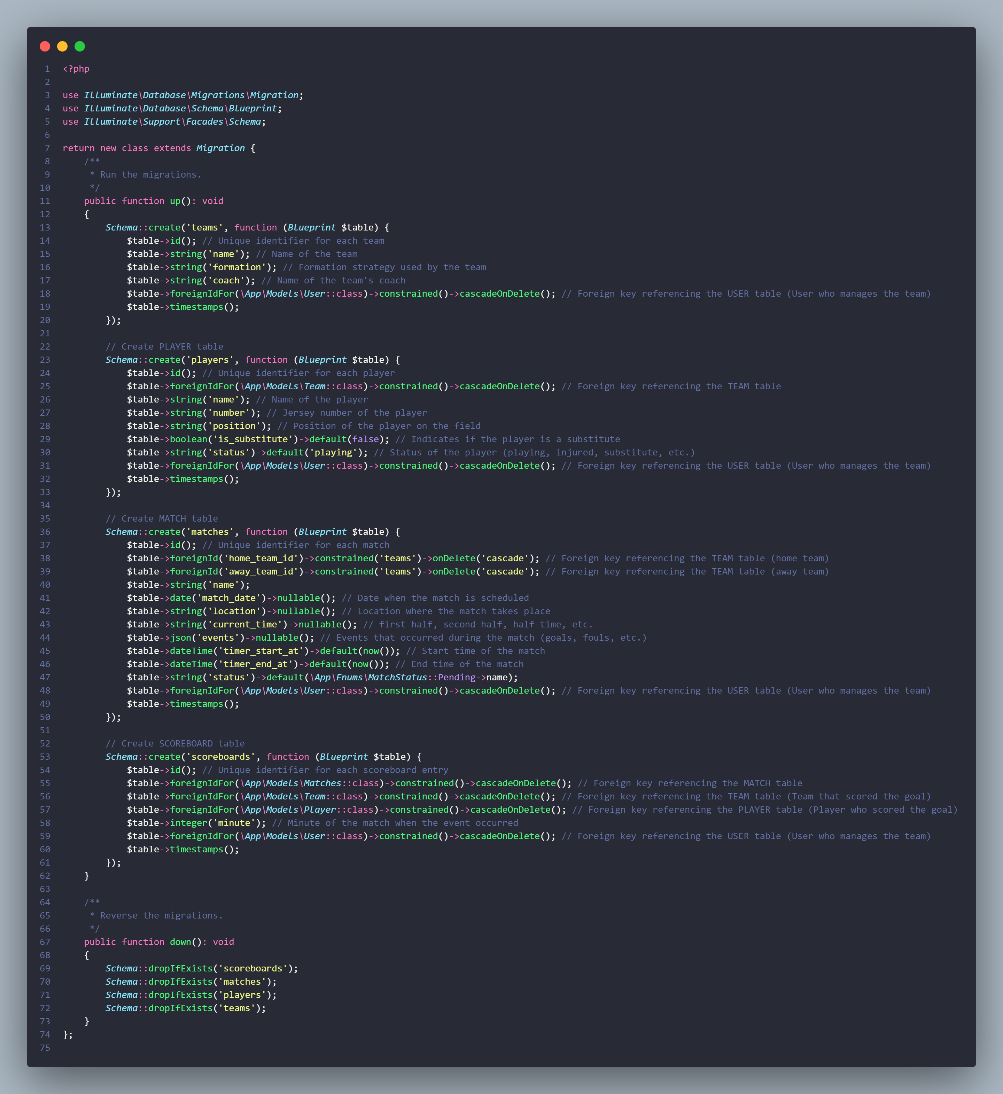
Buat model untuk entitas. Model ini akan digunakan untuk mewakili data tabel dalam bentuk objek.



Gambar . Model *Scoreboard*

1. Pembuatan Migrasi

Buat file migrasi untuk mendefinisikan struktur tabel dalam database. Jalankan migrasi untuk membuat tabel di database.



Gambar . *Migration Scoreboard*

1. Pembuatan Controller

Buat controller untuk menangani logika permintaan untuk menyimpan, memperbarui dan mengambil data dari database

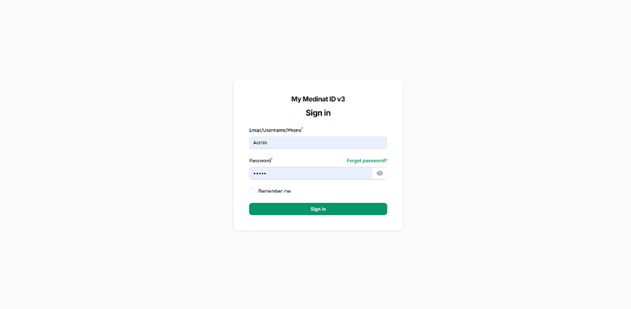


Gambar . *Controller clsss* Login

## Hasil Tampilan Aplikasi

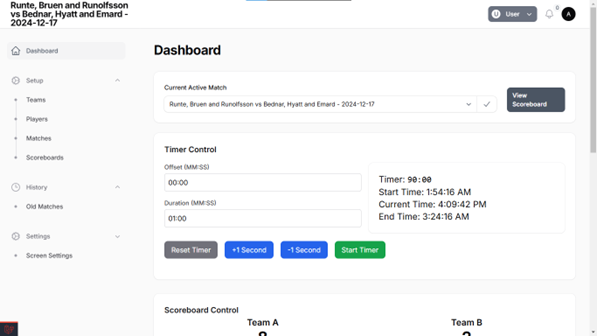
1. Tampilan *Login*

Halaman ini berfungsi sebagai pintu masuk ke aplikasi web. Pengguna harus memasukan kredensial berupa email dan kata sandi untuk mengakses fitur yang tersedia.



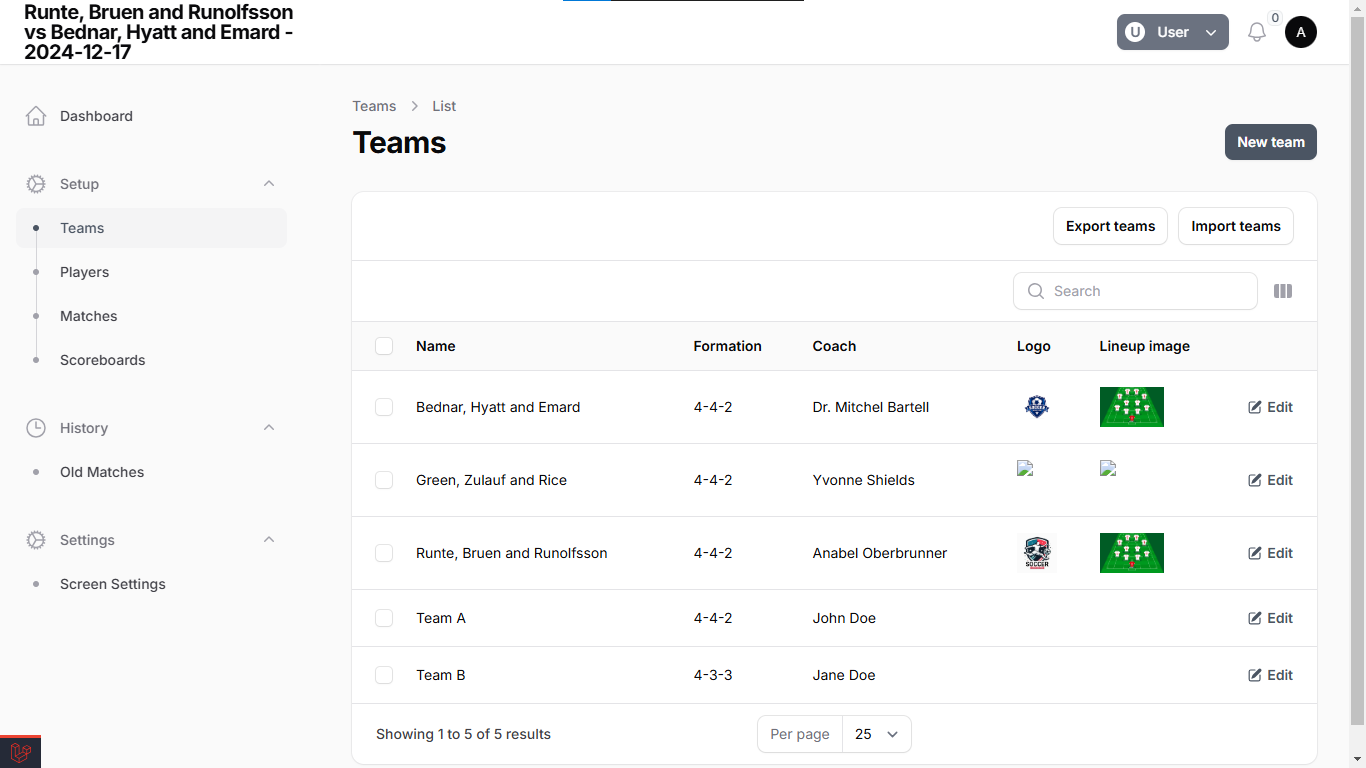
1. Tampilan *Dashboard*

Dashboard menampilkan rangkuman informasi utama mengenai



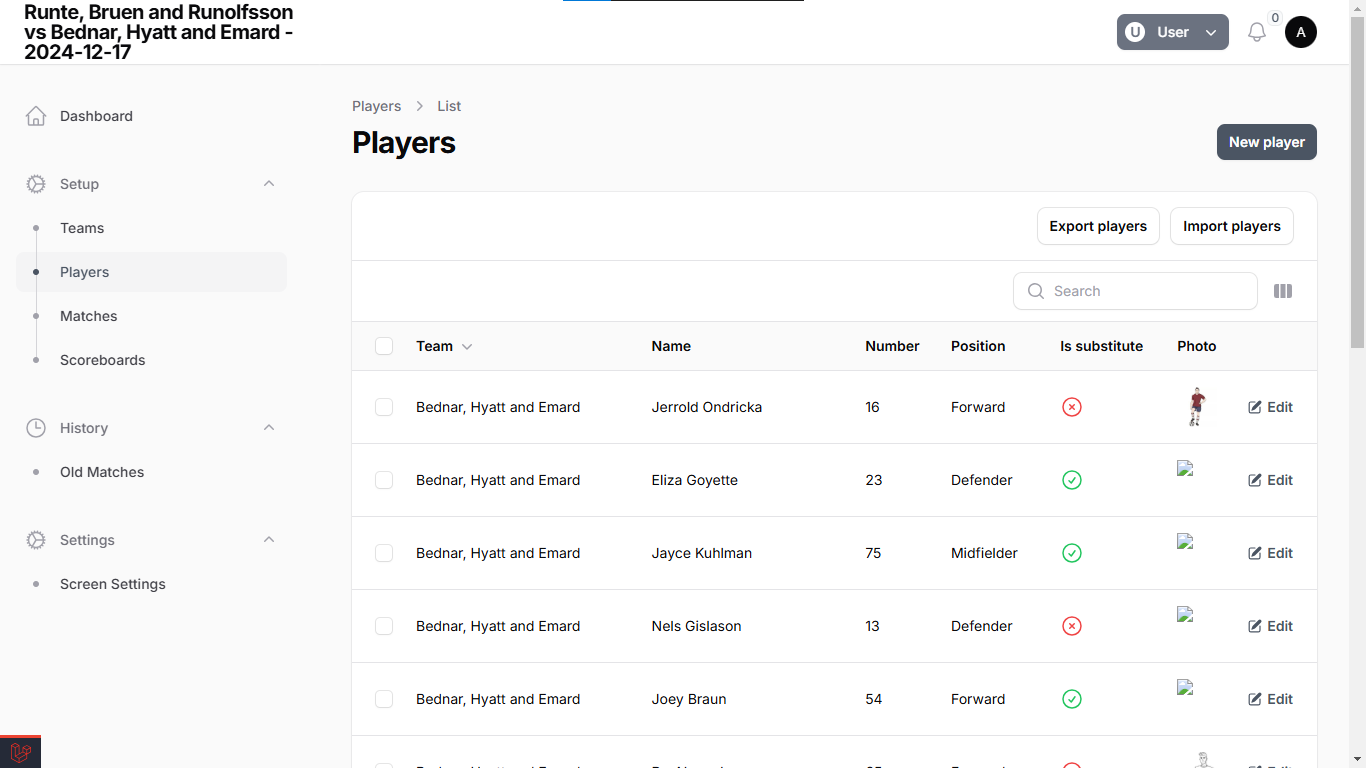
1. Tampilan *Teams*

Halaman ini menampilkan



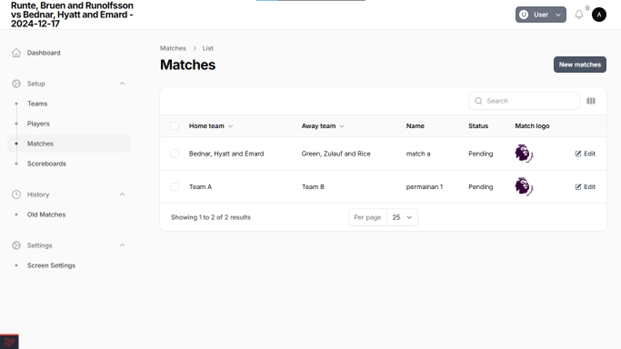
1. Tampilan Players

Halaman ini menampilkan

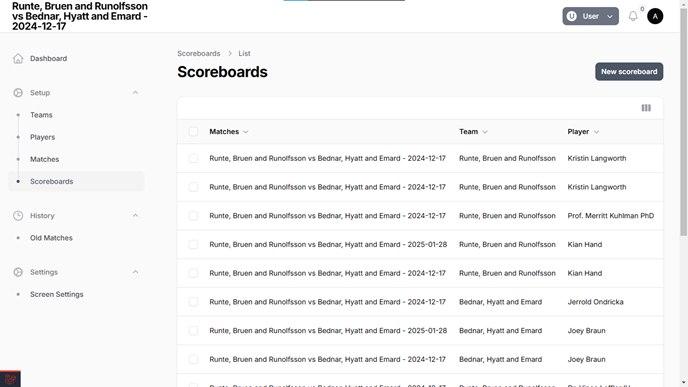


1. Tampilan Matches

Halaman ini menampilkan

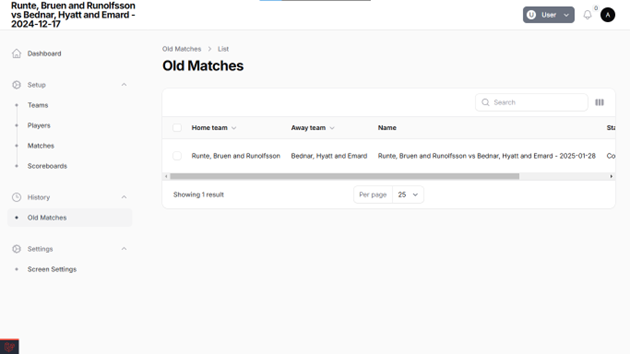


1. Tampilan History Scoreboards

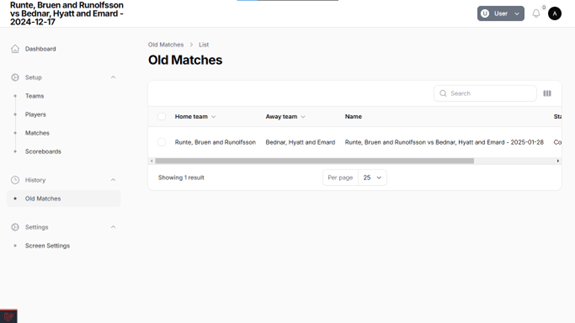


1. Tampilan Old Matches

Halaman ini berisi informasi *history match*



1. Tampilan Theme Settings



# PENUTUP

## Simpulan

## Saran

# DAFTAR PUSTAKA

# CV CREW

# KODE PROGRAM

# LEMBAR BIMBINGAN