Aplicació per a la gestió de clubs esportius amateur. Club Bàsquet Grup Barna.

William Gregorio Cruz Herrera

Resum— El present treball de fi de grau d'Enginyeria Informàtica, té com a objectiu principal desenvolupar una eina web per a gestionar les dades d'un club esportiu de bàsquet i oferir una interfície web que sigui intuïtiva i fàcil d'usar.
Per arribar assolir els objectius, el projecte passa per sis fases seguint el desenvolupament de l'enginyeria del software. A través de les fases s'elaborarà un anàlisi, disseny, implementació i proves de forma exhaustiva, és a dir, si hi ha alguna errada en alguna de les fases sempre es torna a la fase anterior per a corregir-la.
El resultat final serà un sistema web desenvolupat sobre la base principal d'un framework Laravel, essent aquesta la part de programació web i se sustentarà tota la informació en la base de dades MySQL. Aquest projecte pertany al programa d'Aprenentatge i Servei de la UAB.

Paraules clau— Aplicació web, PHP, SGBD, BD, Laravel, client servidor, Framework, HTTP, MVC, Kanban.

Abstract— The main aim of this final degree project in computer engineering is to develop a web tool able manage data from a basketball sport club and to provide a web interface both intuitive and easy to use.
To achieve these goals, the project will go through six phases following the development of the software engineering. These phases will be the analysis of the state of the art, the design of the application, the design of the interface, the implementation, the test, and finally the preparation of a memory and the presentation of the project. If an error is detected in any of the phases, it will be necessary to return to the previous phases and correct it.
The final result will be a web system developed over the main base of a framework Laravel, which will be the web programming part, and all the information will be held in the database MySQL. This project belongs to the UAB's learning and service program.

Index Terms— Web Aplication, PHP, DBMS, DB, Laravel, Server Client, Framework, HTTP, MVC, Kanban

1 Introducció

El C.B. Grup Barna és un equip de bàsquet del barri del Clot de Barcelona amb més de 50 anys d'història que té l'objectiu de promocionar l'esport del bàsquet tant masculí com femení i competir a totes les categories, siguin amateurs o professionals.

El club esportiu disposa d'un equip directiu, un equip tècnic així com jugadors i jugadores de totes les edats que competeixen. El club es manté econòmicament de les quotes que paguen els jugadors i de subvencions de la Federació Catalana de Basquet, l'Ajuntament de Barcelona i la Generalitat. Aquests ingressos li serveixen per a pagar l'equip tècnic, material i alguns sous d'alguns jugadors semi-professionals de la Lliga Catalana. Disposa de camps d'entrenament no propis, alguns cedits per l'Ajuntament com el pavelló de La Nau del Clot o el camp de la parròquia de Sant Ignasi, d'altres són patis d'escola com La Farigola, Provençals, etc. Els colors distintius de la camiseta són negre i vermell.

El club és present a les xarxes socials mitjançant la seva web (https://cbgrupbarna.com/), Twitter i Instagram (@cbgrupbarna). La digitalització del món esportiu, tant de la federació com a nivell de clubs motiva que clubs de la categoria del C.B. Grup Barna hagin d'adaptar-se a les noves tecnologies per a gestionar la informació que generen. Aquest Treball de Fi de Grau (TFG) pretén ser un primer pas a aquesta digitalització, i també poder disposar d'una memòria digital de la trajectòria del club esportiu.

2 Objectius

El principal objectiu del projecte és el disseny, implementació i test d'una aplicació web amb estructura client-servidor, de manera que els clients puguin connectar-se des de PC o des de mòbil. Per tant s'han de definir formularis i una interfície àgil i intuïtiva.
Podem establir que les principals funcionalitats de l'aplicació són els següents:

- Crear formularis que permetin introduir i actualitzar elements d'informació. En aquest formularis s'han d'incorporar tres botons: Confirmar, cancel·lar els canvis i un altre d'esborrat.
- En la part superior de la pantalla, s'ha de crear una opció de *Consulta* i al posar el ratolí a sobre ens haurà

- *E-mail de contacte: williamgregorio.cruz@e-campus.uab.cat*
- *Menció realitzada: Enginyeria del Software*
- *Treball tutoritzat per: Enric Martí Godia (Ciències de la Computació)*
- *Curs 2020/21*

de mostrar aquestes pestanyes: direcció, coordinació, equips i jugadors. Al fer clic en alguna d'aquestes pestanyes se'ns dirigirà a un formulari per a realitzar la consulta desitjada on l'usuari pugui especificar l'item o items de cerca.

El projecte s'implementa sobre l'arquitectura client servidor amb ús de navegadors.

Les eines de desenvolupament utilitzades són:

- **Xampp [1]:** Paquet de programari lliure, proporciona el sistema de gestió de base de dades *MySQL*, el servidor web Apache i els intèrprets per a llenguatges de scripts *PHP*.
- **Composer [2]:** Sistema de gestió de paquets per a programar en *PHP* el qual proveeix els formats estàndard necessaris per a gestionar dependències i llibreries de *PHP*.
- **Git Bash [3]:** Control de versions; la programació és en local o PC personal. Es farà servir la seva terminal per a anar pujant el treball al servidor de l'escola.
- **Laravel [4]:** *Framework* que es farà servir per el desenvolupament del *Back-End* i *Front-End* del TFG.
- **Visual Studio Code [5]:** Editor de codi que s'ha utilitzat en el desenvolupament del projecte.

Per a aconseguir els objectius és necessari establir una bona planificació per a l'execució del projecte. Per això s'utilitza la metodologia *Kanban* [6] per assolir l'acompliment d'aquest TFG.

Seguidament expliquem la planificació i la metodologia dutes a terme en el projecte.

2.1 Planificació

Per a dur a terme els objectius i la correcta finalització del projecte, en cada fase s'ha desenvolupat en les sis fases següents:

1. **Estat de l'art:** Cerca d'informació sobre l'entorn web similars al producte que se'ns proposa en els requeriments.
2. **Disseny Aplicació:** Anàlisi i disseny de l'aplicació que s'ha portat a terme mitjançant el document de requeriments v.1.0. A partir dels requeriments, s'ha fet un anàlisi exhaustiu sobre la infraestructura necessària, on s'ha realitzat i estudiat com s'aplica l'arquitectura model vista controlador. També s'ha buscat la tecnologia que ens ajuda a reaprofitar codi desenvolupat (*Laravel*). Aquesta fase es composa dels requisits del software i el disseny de la base de dades, amb el diagrama relacional que s'ha obtingut de les dades.
3. **Disseny d'interficie:** Es defineixen els elements gràfics, events, accions, etc, que permeten a l'usuari accedir amb facilitat als continguts, així com navegar i interactuar amb l'eina web amb eficiència.
4. **Implementació:** Previ al desenvolupament s'ha configurat l'entorn de treball amb les eines requerides per a la implementació d'aquest projecte. Un cop configurat, s'ha dut a terme el desenvolupament de l'estructura de l'aplicació mitjançant *framework Laravel* que permet desenvolupar el codi en *PHP* d'una forma simple i elegant. Amb el disseny de la base de dades, s'ha construït, tenint en compte que les dades estan en formats diversos com Excel, bloc de notes i bases de dades *MySQL*.
5. **Test:** En aquesta fase es comprovarà el bon funcionament de l'aplicació. S'han realizat proves unitàries, és a dir, s'ha fet test que ataquin a una part especifica del sistema (una funció o mètode).
6. **Memòria i presentació:** Preparació del present article i de la presentació oral (PowerPoint) i finalment la defensa del projecte davant del tribunal del projecte.

Per a veure més detall de les fases i tasques del projecte dutes a terme, es pot consultar en l'apartat de l'apèndix "*A1 Planificació i fases del projecte*". Es mostra el diagrama *WBS* del TFG, amb les 6 tasques. En cada tasca es desglossa un conjunt de sub-tasques desenvolupades.

2.2 Metodologia àgil

El procés de desenvolupament d'un projecte és una activitat molt complexa i difícil de controlar. Per tant, per a facilitar el desenvolupament del software d'aquest TFG s'ha utilitzat una metodologia àgil. La metodologia àgil més coneguda és *SCRUM* [7], però està orientada a treball en equip.

Durant la realització de l'aplicació s'ha utilitzat la metodologia *Kanban* [8]. *Kanban* és una solució per a treure el millor rendiment i gestionar els fluxes de treball, que permet visualitzar en un tauler i processar en cada moment les tasques a realitza. *Scrum* és un marc de treball molt prescriptiu, en comparació amb *Kanban*. *Scrum* requereix una planificació detallada i restrictiva, amb processos i rols predefinits, mentre que en *Kanban* no hi ha rols preestablerts, ofereix una enorme flexibilitat per a adaptar-se a qualsevol treball [9].

A la *taula 1* fem una comparació entre ambdues metodologia àgils.

KANBAN	SCRUM
Flux constant. S'alliberen entregues amb base aesdeveniments o necessi-tats.	Sprints regulars de duració limitada. (2 – 4 setmanes).
Entrega continua, abordant un petit nombre de tasques de manera fluida i simultània.	Entrega al final de cada Sprint amb l'aprovació del product Owner.
Els canvis es poden produir en qualsevol moment.	No hi ha canvis durant el Sprint.
No hi ha rols prescrits.	Hi ha rols definits com Product Owner, Scrum master, Development Team.

Taula 1: Comparació metodologia àgils kanban i scrum.

Per la seva flexibilitat, hem decidit utilitzar *Trello* per a planificar i organitzar les tasques. *Trello* ens ha ajudat en el seguiment i control de tot el projecte (*Apèndix - A.1 figura 15*).

A continuació expliquem en la secció 3 l'esta t de l'art, en la secció 4 exposarem el disseny, en la secció 5 expliquem

la implementació, en la secció 6 exposarem els resultats i en la secció 7 expliquem conclusió i millores.

3 Estat de l'art

S'han buscat productes i aplicacions en gestió esportiva, similars a la que ens proposem fer.
El primer que hem trobat és "gestiondeportiva" [10]. És una eina anomenada "web del club" que ofereix l'autogestió d'un club esportiu tant per a dispositius mòbils com a ordinadors. La persona encarregada en l'administració esportiva no ha de tenir coneixements informàtics previs, doncs és una aplicació molt intuïtiva i fàcil d'utilitzar. Aquesta aplicació permet gestionar plantilles a partir d'un formulari de (partits i resultats), jugadors, actualització de les classificacions dels equips, entrenaments (*screenshot*), calendaris. També disposa de formularis d'inscripció per a les noves altes.
Una altre eina del sector es *SportEasy* [11]. Es tracta d'una *app* per a dispositius mòbils i ordinadors dedicada a la gestió d'equips esportius. Aquesta *app* és aprofitada per aquells clubs amb baix pressupost, de manera que li permet crear calendari d'esdeveniments, generar estadístiques de participació, rendiment dels jugadors, assistència, etc. Aquesta *app* pot ser gestionada pel pare del jugador/a i vulgui gestionar la seva carrera des de zero.
Es té la possibilitat de fer servir l'aplicació per a diferents esports com futbol, bàsquet, tenis, etc. A més cal remarcar que es tracta d'aplicacions de pagament. Per la creació d'algún mòdul en l'aplicació del club esportiu, s'ha de posar en contacte amb l'empresa que ofereix el servei de gestió esportiu.
La principal diferència respecte la nostra proposta, és que la nostra aplicació no serà gestionada per empreses externes. Cal destacar que en fer un projecte des de zero a partir dels requeriments recollits ens permet desenvolupar un producte a mida i cenyint-nos a les necessitats que requereix el club, tal com s'explica en l'apartat d'objectius d'aquest projecte.

4 Disseny

El procés del disseny va des de la recollida de requeriments fins a l'anàlisi posterior. El disseny d'aquest projecte es divideix en dues parts:

1. **Disseny de l'aplicació:** Es realitza l'estructura interna o lògica (*Back-End*) del servidor web mitjançant la utilització d'un *framework* (*Laravel*). Per altra banda, el disseny de la base de dades per a respondre les necessitats del client, com: emmagatzemar, recuperar i suprimir dades.
2. **Disseny de la interfície:** En aquest apartat s'han fet diferents esbossos i prototips per a visualitzar la interfície web com: l'inici de sessió, formularis de registres, formularis de consulta d'informació *personal/equip/categoria*, formularis *inserció/actualització* d'elements, etc.

Seguidament expliquem el disseny de l'aplicació i disseny de la interfície.

4.1 Disseny de l'aplicació

En aquest projecte s'ha realitzat un codi formal i comprensible, així com definir una estructura fàcil d'entendre per a les persones o programadors que vulguin seguir amb millores o manteniment del projecte.
Per a aconseguir un codi estructurat s'ha aplicat *MVC* (*Model – View – Controller* o Model – Vista – controlador) [12]. L'arquitectura *MVC*, permet dividir la lògica del disseny, les dades i la interfície d'usuari en diferents mòduls fent el projecte escalable.
A continuació s'explica el flux de treball en el nostre esquema *MVC*, (*figura 1*).

1. El client realitza una sol·licitud al nostre servidor web (per exemple, s'apreta un botó o accedeix a un apartat de la web). Aquesta sol·licitud arriba al controlador.
2. El controlador es comunica tant amb el model com amb la vista. Al model es sol·licita dades o demana d'actualitzar les dades. A la vista es sol·licita una sortida corresponent.
3. Una vegada rebuda la sortida, el controlador envia la vista o fitxer *HTTP* (*response*) amb el contingut.
4. Finalment l'usuari visualitza el contingut.

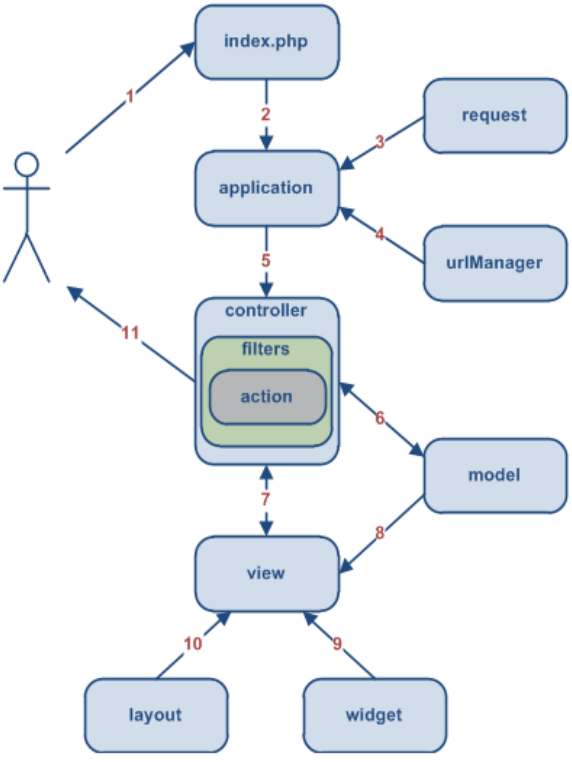

Figura 1: Esquema amb model-vista-controlador

Seguidament expliquem l'arquitectura del sistema, disseny de la base de dades, requeriments funcionals i plataforma de desenvolupament.

4.1.1 Arquitectura del sistema

Durant l'anàlisi de disseny de l'aplicació, es va pensar en programar la pàgina web des de zero, però es va trobar inviable pel temps que disposàvem, i també els possibles problemes que implica definir l'estructura de desenvolupament. Per tant, varem buscar una eina que ens oferis

una arquitectura amb funcions ja desenvolupades, de manera que ens ajudés a estalviar temps i així poder reutilitzar codi, és a dir, funcions o mètodes que són comuns en les pàgines web i que no sigués necessari programar des de zero.
En aquest projecte s'utilitza un *framework* per a reduir la quantitat de codi a desenvolupar. Tots els *frameworks* existents porten un conjunt d'utilitats per a:

- Definir l'arquitectura de desenvolupament *MVC* (Model-Vista-Controlador).
- Autenticació d'usuaris, nivells control d'accés, sessions, *cookies*, etc.
- Estructura de directoris i arxius modulars.
- Correcte gestió de formularis i validació de dades.

Hem estudiat els *frameworks Laravel* i *Django* que més ens han cridat l'atenció, doncs es pot construir una aplicació en poc temps. El llenguatge de programació és diferent en ambdues tecnologies: en *Laravel* és *PHP* i *Django* és *Python*. Després d'estudiar les característiques de cada *framework* l'opció triada és *Laravel*, que dins del món *PHP* és un dels *frameworks* més utilitzats. *Laravel* permet desenvolupar amb una qualitat elevada i a més té una gran comunitat a Internet. La idea principal d'aquest *framework* és tenir un codi elegant i simple per així poder evitar tenir problemes d'implementacions [13].
Els avantatges d'utilitzar *Laravel*, són que es pot desenvolupar i fer un disseny web a les necessitats de cada empresa, el manteniment és senzill comparat amb altres *frameworks* i disposa de múltiples funcionalitats. *Laravel* és compatible amb la majoria de base de dades, en temes de seguretat el *framework* és molt segur perquè es poden evitar atacs d'injeccions *SQL* i és escalable, entre altres avantatges.

4.1.2 Disseny base de dades

Un dels aspectes importants d'aquest treball ha consistit en el desenvolupament d'una base de dades per a donar suport a les necessitats del document de requeriments.
Basant-nos en el document de requeriments v.1.0, per a representar les dades s'utilitza el model d'entitat-relació (*E/R*), que ens permet modelar les dades d'una manera optima, així com organitzar la informació i tenir molt clares les restriccions que tindrà en la base de dades.

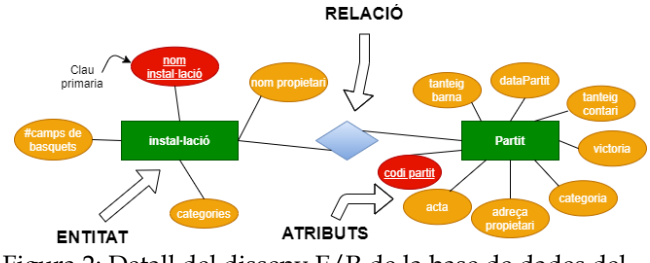

Figura 2: Detall del disseny E/R de la base de dades del C.B. Grup Barna.

En l'apèndix (*A3 Disseny de la base de dades*) es mostra el disseny E/R complet on es tenen en compte els requeriments de dades. El disseny E/R està representat per 18 entitats que inclou atributs i claus primàries per a cada entitat. També es defineix un total de 21 interrelacions entre entitats i les seves connectivitat.
A partir del model d'E/R es realitza la conversió a una estructura del model relacional (disseny lògic), la qual es presenta les principals taules, claus primàries, foranies, etc (*figura* 3).

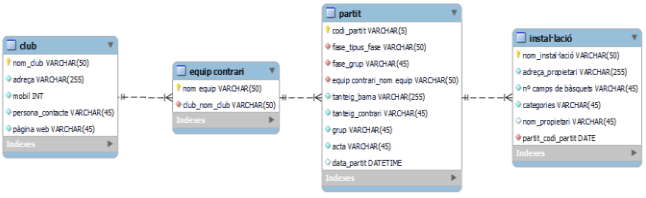

Figura 3: Detall del disseny relacional de la base de dades del C.B. Grup Barna.

Per veure amb més detall el disseny relacional realitzat, es pot consultar en l'apartat de l'apèndix "*A2 Disseny de la base de dades*".

4.1.3 Requeriments funcionals.

Com s'ha esmentat anteriorment, la idea és configurar un web server amb llenguatge *PHP* amb base de dades *MySQL*, de manera que els clients puguin connectar-se amb un *PC* o qualsevol dispositiu per a realitzar consultes utilitzant formularis. Tractem amb cinc tipus d'usuaris (*figura 4*):

- **Administrador (A):** Gestiona el servidor, dona d'alta i baixa usuaris, accés total a tot.
- **Equip Directiu (ED):** Plena gestió (inserció, actualització, esborrat i accés) a tota la informació del club.
- **Coordinador (C):** Gestiona un conjunt de categories. Usualment hi ha un de pilota petita (categories Pre-Mini, Mini, Pre-Infantil), un de pilota gran de categories inferiors (Infantil, Cadet, Junior) i un coordinador de sèniors. Aquests coordinadors han de tenir plena gestió (inserció, esborrat, actualització i accés) a tota la informació dels equips que coordinen.
- **Equip Tècnic (ET):** Entrenadors (primer, segon, tercer) i preparadors físics que tenen plena gestió a la informació dels equips amb que treballen.
- **Jugadors (J):** Poden accedir a la informació que genera el seu equip (entrenaments, partits), no a dades personals de jugadors. Es poden donar d'alta com a jugadors posant les seves dades personals i el coordinador valida la inscripció.

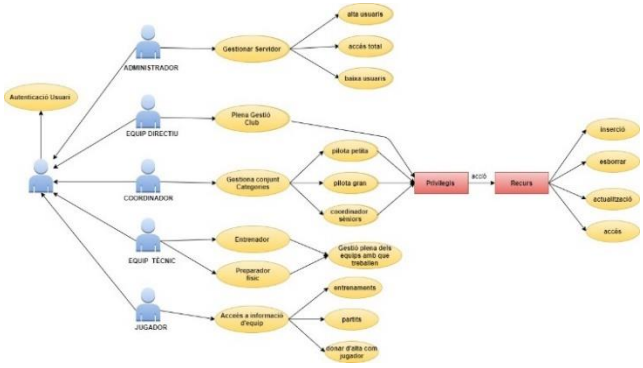

Figura 4: Diagrama d'usuaris del C.B Grup Barna amb restriccions d'accés.

A continuació es detallen els requeriments funcionals:

1. Construir formularis per a introduir i actualitzar elements d'informació, incorporant tres botons: Confirmar, cancel·lar els canvis i esborrat, amb símbols fàcilment identificables.
2. En la inserció, actualització o esborrat de qualsevol element d'informació es verificarà si les dades ja estan entrades prèviament. L'esborrat a qualsevol element d'informació estarà restringida si perilla la integritat referencial de la Base de Dades.
3. Si una persona té diferents rols en el club s'especificarà en el formulari, podent també aquesta informació ser actualitzada.
4. En la inserció-actualització de competicions, partits i tornejos s'ha de detectar que no hi hagi solapament de calendaris per part dels equips participants.
5. La interfície ha de ser clara i intuïtiva a l'hora de definir el que es vol buscar, definint formularis d'informació com jugadors, entrenadors, competicions, partits, etc. (per exemple per pestanyes) i volcant la informació de forma clara. Quan en el formulari aparegui una informació relacionada a un altra formulari es mostrarà com a seleccionable per a que sigui triada per l'usuari, proporcionant en lo possible una experiència de navegació dins la informació.
6. Quan calgui, en un formulari que mostri els elements d'una taula (per exemple jugadors de l'equip) s'ha de poder seleccionar un, varis o tots els jugadors sobre els que fer la consulta o estadística posterior i els atributs de cerca.

4.1.4 Plataforma de desenvolupament

Durant el desenvolupament de l'aplicació s'ha treballat en un *PC* personal, on s'han utilitzat les eines necessàries (secció 1.1) per a portar a terme aquest projecte.
Per altra banda s'ha tingut en compte que les eines utilizades siguin de programari lliure, és a dir, que no tinguin restriccions d'ús i siguin gratuïtes.
Un cop s'ha finalitzat la pàgina web el resultat final de l'aplicació s'ha validat en el servidor web de l'escola d'Enginyeria. En aquest servidor s'ha realitzat el mateix procediment d'instal·lacions d'eines compatible amb la Plataforma. També s'han fet les proves necessàries per a posar-ho en marxa per a l'usuari final.
Les característiques de la plataforma de desenvolupament es mostren a la *taula 2*.

HP Laptop 15-da0082ns	
Sistema Operatiu	Windows 10 Home
CPU i Cores	Intel® Core (TM) i7-8550U
RAM	16.0 GB
Tipus de Sistema	64 bits

Taula 2: Característiques del computador de desenvolupament.

El procés de desenvolupament de la pàgina web no requereix d'una potent màquina per a executar tota l'arquitectura de la pàgina web, amb un ordinador amb característiques similars a la *taula 2* és suficient pel desenvolupament d'aquest projecte. Les dades de la taula 2 són característiques on s'ha portat a terme el desenvolupament d'aquest projecte (depèn del nombre d'usuaris).

4.2 Disseny de la interfície

Fet el procés d'anàlisis dels diferents requeriments, cal pensar com serà la interfície d'usuari. Per tant, es defineix un esbós o prototip (*sketch*) visual per a fer-nos una idea de com serà la pàgina web.
Per a realitzar el prototip s'ha utilitzat l'eina de disseny *justinmind* [14]. *Justinmind* és una eina de creació de prototips i té la capacitat per a reproduir versions realistes d'un producte.
En l'accés principal de la web hi ha un petit canvi en la capçalera on s'han afegit dos camps:

- **Idioma:** L'usuari pot escollir un idioma (català, castellà o anglès). Aquest camp és opcional.
- **Inici sessió:** En aquest camp el membre del club s'identifica amb un número d'usuari (codi) i contrasenya per a iniciar sessió. Quan l'usuari s'identifica, s'inicia un procés de validació en el sistema que ha de ser capaç d'identificar el tipus d'usuari i redirigir-hi al perfil corresponent (*figura 6*).

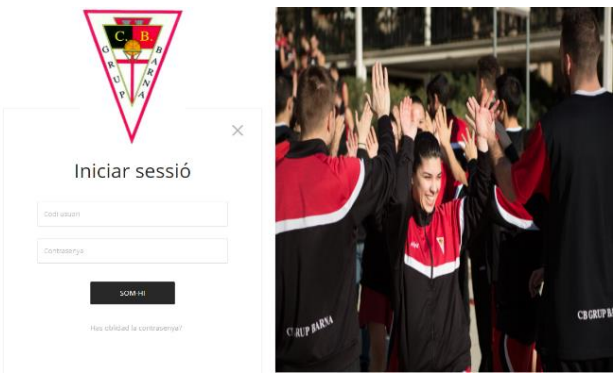

Figura 5: Prototip inici de sessió.

En la *figura 6* es mostra l'esbós d'una de les seccions de la pàgina web: Quan l'administrador inicia la sessió desseguida acceixà al panell principal d'administració i al pressionar el botó "*Accedir*" podrà gestionar l'alta i la baixa dels usuaris: jugador, coordinador, equip tècnic i equip directiu. L'equip directiu té la mateixa interfície que l'administrador, tenint la plena gestió (inserció, actualització, esborrat i accés) a tota la informació del club.

Figura 6: Prototip de la interfície de l'administrador.

5 IMPLEMENTACIÓ

La implementació és la fase més crítica del projecte, doncs es posa en marxa i s'executen totes les accions previstes en la planificació. El projecte de sistema de gestió d'informació Grup Barna es desenvolupa mitjançant l'ús de *framework Laravel* tant per a la part del *Front-End* i *Back-End* de l'aplicació web.

Per a la implementació d'aquesta part es requereix les instal·lacions dels programaris de la secció 1.1 que són necessàries per desenvolupar el projecte. Els passos previs a les instal·lacions de les eines s'expliquen a l'apèndix (*A.3 Instal·lació de Laravel via composer*).

La implementació es divideix en dues parts que es desenvolupen de forma paral·lela:

- **Desenvolupament de la base de dades**: A partir del disseny relacional, es procedeix a muntar la base de dades utilitzant tecnologies de migracions que ens ofereix *Laravel*.
- **Desenvolupament web:** Es defineix la creació de l'estructura interna del projecte utilitzant *frameworks* com *Laravel* tant pel *Back End* i *Front End*.

El *framework Laravel* ens ofereix un entorn de desenvolupament funcional, així com l'ús de línies de comandes que facilita la creació de fitxers amb un esquema preestablert per a la codificació d'interfícies webs que són comunes en totes les pàgines web, com per exemple: l'inici de sessió, formularis de registre d'alta d'usuaris, formularis de consultes, etc. *Laravel*, proporciona també un conjunt d'eines que ajuden en el procés de desenvolupament de l'aplicació perquè siguin més fàcil i ràpid.

Seguidament expliquem les dues parts de la implementació.

5.1 Desenvolupament de la base de dades.

Farem ús del mecanisme de migracions que ens proporciona *Laravel*, on es fa un control de versions de possibles canvis en l'estructura de la BD. Amb aquest mecanisme es podren realitzar canvis en un futur sobre la infraestructura, com per exemple afegir una columna en alguna de les taules, sense alterar la integritat de les dades. Amb les migracions es dissenya una estructura utilitzant *PHP* i programació orientada a objectes sense la necessitat d'escriure codi *SQL* [15].

En crear un esquema es defineixen els elements que formen una taula com els atributs d'una entitat, el tipus de les dades (string, int, boolean, etc.), relació que té amb altra/es entitat/s i finalment les limitacions de cada atribut.

Seguint l'exemple de la *figura 7*, es realitza el mateix procediment per a construir les taules amb una interficie orientada a objecte mitjançant aquests mètodes:

- *$table->string('nom_equip', 100)*, permet crear una columna de tipus varchar (cadena de màxim 100 caràcters). Amb "*$table -> primary (['nom_equip'])*" es defineix la clau primaria.
- *$table->foreign('temporada_any') -> $references('any') -> (temporada)* permet definir una referència sobre la columna "*temporada_any*" amb la columna "*any*" de la taula temporada.

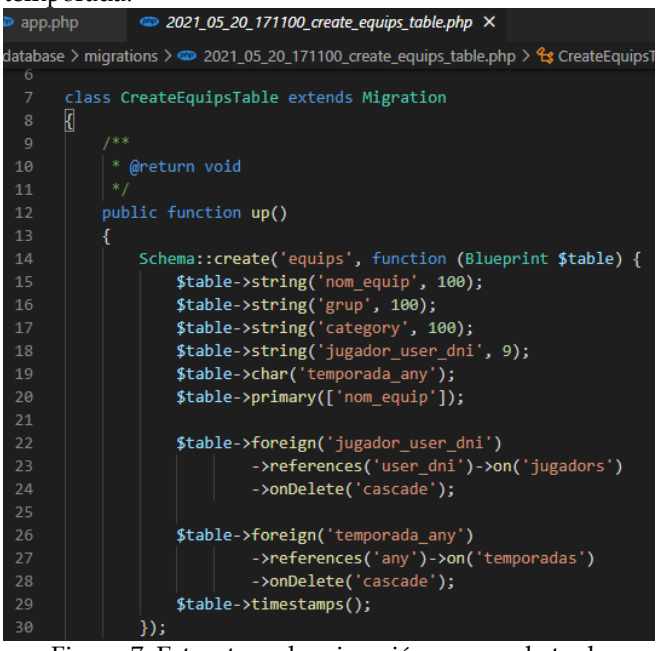

Figura 7: Estructura de migració per crear la taula d'equips.

S'han generat un total de 28 migracions que contenen tota l'estructura de la base de dades. Les migracions, poden ser executades en qualsevol sistema gestor de base de dades (*SGBD*), sempre que sigui suportat per *Laravel*.

Després de tenir les migracions generades mitjançant la consola de comandes de *Visual Studio Code* s'ha d'executar la següent comanda: *"php artisan migrate"*, per a poder crear les taules en *SGBD*, és a dir, a *phpMyadmin* [1]. Per veure amb més detall el diagrama relacional generat a *phpmyadmin*, es pot consultar a l'apèndix "*A2 Disseny de la base de dades figura 22*".

Finalment, el procés d'emplenar les taules creades amb dades fictícies es realitza de la següent manera:

1. S'utilitza l'eina de model *factories* [4] que treballa amb components *fakers* [15]. Es una forma d'omplir la base de dades amb dades de proves generades automàticament amb *Laravel*.
2. Per a inserir un nou registre en la base de dades s'ha de crear sentències *SQL* "insert into" per a poder introduir les en la base de dades (*figura 8*).

Figura 8: Sentencia *SQL* per a inserir un nou usuari en la BD.

El model *factories* amb *faker* és la millor opció per a generar les dades fictícies per a realitzar les proves que requereix el projecte, doncs ho fa de manera automàtica amb l'execució d'una comanda, per així omplir les taules en la base de dades.

5.2 Desenvolupament Back-End

Pel desenvolupament del *Back-End* cal crear un repositori en *Bitbucket* [16] on es guarda el projecte, el que dóna la

possibilitat de tornar enrere en cas de cometre un error.
En instal·lar *Laravel* es genera una estructura de desenvolupament de l'aplicació (*apèndix A.5 figura 24*). En aquesta estructura s'aniran afegint els fitxers que tindran les funcions necessàries per a posar en funcionament el gestor d'informació del club de bàsquet. D'altra banda, s'ha de tenir en compte l'estudi de la infraestructura que ens ha generat *Laravel*, és a dir, el codi i fitxers del *framework* per a veure les connexions que hi ha entre els directoris més rellevants de la infraestructura.
Durant el desenvolupament del *Back-End* s'han tingut presents les característiques que ens aporta *Laravel*, que són conceptes molt bàsics per a entendre l'organització en el desenvolupament i la metodologia que utilitza *Laravel* internament són els següents [4]:

- **Eloquent:** Forma de mapejar les dades que es troben en la base de dades emmagatzemats en un llenguatge de *script SQL* a objectes de *PHP*.
- **Routing:** Sistema de rutes que s'encarrega de conduir el flux de sol·licituds i respostes.
- **Middlewares:** Proporciona un mecanisme convenient per a inspeccionar i filtrar les sol·licituds *HTTP* que ingressen a la seva aplicació.
- **Blade:** Sistema de plantilles per a crear vistes en *Laravel*, amb el que es pot crear plantilles i seccions que es pot reutilitzar en diferents vistes (*Views*).

Un cop es té la base de dades plena amb dades fictícies, podem realitzar les diferents consultes *SQL* pel desenvolupament *Back-End* del gestor d'informació del club. Per això aplicarem les "*bones pràctiques de la programació*" [17] per a poder tenir un codi organitzat, seguint el paradigma *MVC* (Model-Vista-Controlador) i visualitzar les diferents interfícies esmentades la secció 4.2 de disseny d'interfície.

L'aplicació permet crear formularis per a introduir i actualitzar elements d'informació. Per a assolir aquest objectiu s'ha creat una aplicació basada en *CRUD* (*Create*, *Read*, *Update*, *Delete*) que són les operacions bàsiques que es realitza sobre un conjunt de dades [18].
Per a implementar el disseny de les interfícies basat en *CRUD* i el model *MVC* (depèn dels permisos d'accés que s'atorgui a cada usuari), els passos a tenir en compte pel desenvolupament per a cadascuna són el següents:

1. Generar el controlador *userController* (*User* en el disseny relacional és l'entitat Persona). Executem la següent comanda en la consola de *Visual Studio Code* per a crear un nou controlador per a l'aplicació *PHP CRUD*:
   php artisan make:controller userController –resource
2. Accedir al fitxer que ha generat la comanda a la següent ruta "*App/Http/Controller/UserController*" (*figura 9*). En aquest fitxer es defineixen els següents mètodes/funcions per a portar a terme les funcionalitats per a crear, esborrar o modificar un usuari i el resultat de la consulta es mostrarà en una vista:
   a) **index():** mostra una llista dels deu primers usuaris de la BD. D'aquesta manera s'evita problemes de "Memoria Caché".
   b) **create():** Aquesta funció crida a la vista que tindrà el formulari amb els camps a omplir per l'usuari, que posteriorment serà validada per un coordinador del club.
   c) **store():** Amb les dades recollides en el formulari (en la funció *create*), crea un usuari en la base de dades en *phpMyadmin*.
   d) **edit():** Ens permet actualitzar les dades d'un usuari mitjançant l'ús d'un formulari definit en la vista.
   e) **show():** Mostra les dades específiques d'un usuari.
   f) **destroy ():** Elimina un usuari específic al polsar un botó definit en la vista.

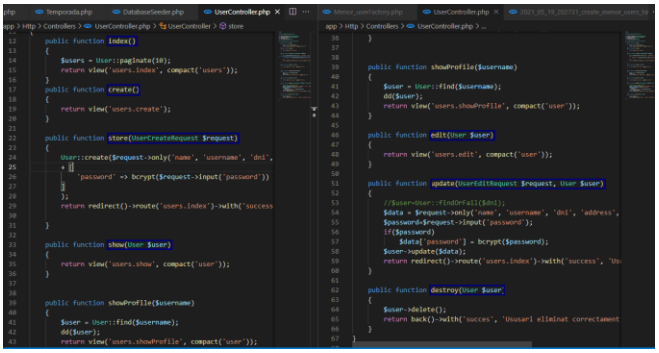

Figura 9: codi del controlador d'usuari.

3. En crear les migracions amb la mateixa comanda, es crea també el model. El fitxer del model s'ubica en la ruta *"App/Http/Models/Users.php"*. Aquests fitxers declaren una matriu $*fillable* que conté tots els camps de la taula *User* que es poden completar mitjançant assignació massiva (enviament d'una matriu al model per a crear un nou registre a la base de dades).
4. En el directori "*route/web.php*" es defineixen totes les rutes que s'utilitzen en el projecte.
5. Finalment es creen les vistes (***Views***). En la ruta *resouces/views* es troben una sèrie de plantilles com: *CSS*, *HTML*, *JS*, i fonts i icones que fan possible veure una interfície agradable per a accedir a un formulari web.

Pel control de rols i permisos descarreguem una migració de *Laravel-Permision* que ve amb una estructura predefinida que s'adapta a les nostres necessitats. En el moment de definir o crear les vistes de formularis seguint els passos del model *MVC* es tenen en compte els permisos que s'ha assignat a un rol. En *phpMyadmin* podem trobar l'associació de taules creades i sincronitzades amb diversos permisos amb el rol d'usuari corresponent (*figura 10*). Per exemple, el rol de Jugador tindrà a *Laravel-Permision* un conjunt de permisos assignats com veure la informació personal, entrenaments, partits.

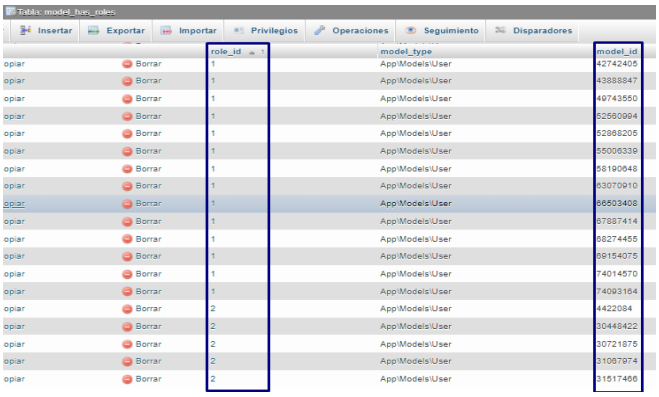

Figura 10: Associació de rols i permís mitjançant el DNI de l'usuari.

5.3 Desenvolupament Frond-End

Finalment, per a la part del *Front-End*, hem fet servir el *framework Bootstrap 4 Admin Dashboard* [18]. Aquest *framework* ens ofereix una plantilla que s'utilitza com a base pel panel d'administració del nostre projecte. A més *Bootstrap* proporciona una sèrie de components (botons, formularis, icones, barra de navegació, etc.), complements (validador de *jQuery*, notificació *Bootstrap*, entre altres.) implementades que es poden fer servir en qualsevol projecte web.

En *Front-End* s'ha implementat vàries interfícies com:

- Formulari "*Login*" per a iniciar sessió. En accedir, l'usuari ha d'introduir el correu electrònic o el nom d'usuari (*username*) i la contrasenya per acreditar-se. En cas d'introduir erròniament les credencials sortirà un missatge "Les credencials introduïdes són incorrectes".
- Formulari de registre on un nou membre del club ha de donar-se d'alta per primer cop.
- En la *figura 11*, es mostra el panell principal que l'usuari veu en iniciar sessió. En polsar el botó "*accedir*" l'usuari tindrà accés al formulari de consulta que li correspongui segons la política de rols i permisos establert.
- Per a llistar els usuaris existents a la base de dades, s'utilitza un formulari que disposa de tres botons per a les accions d'afegir, esborrar i actualitzar les dades d'un usuari.

Figura11: Panell principal d'accés a l'aplicació.

6 RESULTATS

Per a testejar el codi desenvolupat, s'han generat test unitaris (*unit tets*). Aquesta eina ens permet detectar a temps si el nostre codi està generant errors i si compleix amb la lògica esperada.

El que es pretén amb el *unit test* és buscar la part de codi que no funciona com s'espera. Amb el *unit test* s'ataca a la part de la lògica que es troba en el controlador. En els controladors es realitza una prova amb un test funcional on s'aïlla la part que ens interessa testejar. Consisteix a extreure la part que ens interessa en una nova classe.

S'han realitzat una sèrie de test en l'aplicació, en el qual es van recollir diverses errades a l'hora d'editar, eliminar, actualitzar, assignar rol a un usuari. Per a solucionar les errades, s'ha utilitzat la classe "*Request*"[4] implementada prèviament en *Laravel*. Ens ha ajudat a mantenir un control sobre les dades que es reben a través de les peticions que es realitzen, és a dir, es verifica el tipus de dada, la longitud, únic (*Unique*), etc.

En la *figura 12*, es mostra la classe "*LoginControllerTest*" per a test. En aquest test es realitza una prova de pàgina d'inici de sessió, és a dir, es realitza un test quan s'envia una sol·licitud de *GET* a la ruta login que ens ha de retornar la vista *auth.login*. En el mateix test, s'ha creat una funció "*testLoginFalse*" per a poder validar les credencials dels diferents membres del club.

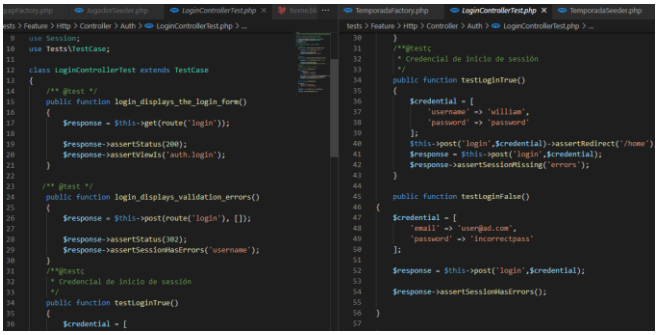

Figura 12: Codi de testing per a verificar l'accés a la web.

Un cop resoltes les errades, en executar la comanda *$ vendor/bin/phpunit* s'executen tots els tests definits en el projecte. En la *figura 13* es mostra el resultat del test de l'aplicació, on el color verd indica que tots els tests i assercions s'han complert correctament, també indica el temps que tarda a executar-se el test i la memòria consumida.

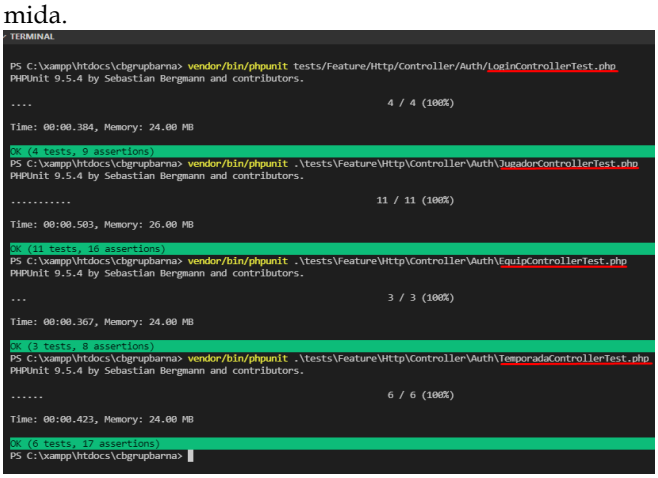

Figura 13: Resultats del testing per a verificar l'accés a la web.

7 CONCLUSIÓ I MILLORES

- S'ha implementat un gestor d'informació per a un club esportiu de bàsquet amb estructura client-servidor, en el que els clients del club poden accedir mitjançant un PC o des de mòbil, on tenen accés a formularis de manera àgil i intuïtiva a consultes d'informació com jugadors, entrenadors, competició, partits, etc.
- S'ha dissenyat i implementat la base de dades, definint l'estructura del nostre sistema gestor d'informació.
- S'ha dissenyat i implementat un Back-End i un Front-End, creant una sèrie de funcionalitats per a posar en marxa la pàgina web.
- En la part de desenvolupament de la base de dades, m'he trobat amb diversos problemes utilitzant l'eina *faker* and *seeders* de *Laravel*, per a generar dades fictícies. A l'hora d'omplir de dades en la base de dades, es donaven una sèrie de problemes d'incompatibilitat amb el tipus de dades definides en la migració. En els casos, que no s'ha trobat la solució, s'ha hagut de generar dades en les taules manualment (a *phpmyadmin*).
- En aquest projecte he après moltíssim a treballar de forma organitzada i a seguir una planificació establerta per a portar a terme cadascuna de les entregues del TFG. Cal destacar que treballar amb el framework Laravel ha estat molt satisfactori, disposa de diferents eines que són de gran ajuda a l'hora d'implementar qualsevol interficie web, doncs els recursos (manuals, foros, vídeo) que ofereix la comunitat de Laravel és molt ampli.

Hi ha diverses millores en el projecte:

- En el registre d'usuaris, en lloc de posar la data de naixement manualment, caldria afegir un desplegable on l'usuari pugui seleccionar la data de naixement. En aquest formulari també crec que és necessari definir desplegables per seleccionar la ciutat, provincia i país, en la part de domicili de l'usuari.
- Millorar la interficie principal d'accés, afegint algun tipus d'animacions tipus Ajax. Per exemple en el moment de situar el ratolí a sobre el requadre del tipus d'usuari (*figura 11*) que es pugui il·luminar.
- Desenvolupament d'aplicacions per a mòbils amb diferents sistemes operatius (*Android*, *iOS*, *Windows phone*), doncs els usuaris s'adaptarian millor a una aplicació mòbils que en pàgines web.

Com a opinió personal del treball final de grau, cal dir que no ha sigut fàcil i més si no s'obté coneixement previ sobre aquest tema. És un estudi que s'ha de fer de manera delicada i amb molta paciència, ja que es generen etapes de frustració per les dificultats que es presenten durant l'estudi. És un tema molt interessant a investigar i estudiar, però és dificultós.

AGRAÏMENTS

Agraeixo en primer lloc als meus pares Guillermina i Julio per mostrar-me el camí cap a la superació, doncs amb el seu sacrifici i esforç, heu aconseguit que pugui superar aquesta etapa de la meva vida. No podria sentir-me més a gust amb la confiança posada sobre la meva persona. Especialment i des de sempre he comptat amb el vostre suport incondicional. Als meus germans, per estar sempre al meu costat donant-me suport i animant-me en època dures d'exàmens. A la resta de la família, especialment a la meva segona mare Amparo amb la seva tendresa i consells m'ajudat a seguir endavant.

Al meu tutor Enric Martí, per haver-me guiat i tenir una gran paciència a explicar-me cadascun dels dubtes que he tingut en el TFG. Aquest projecte m'ha ajudat a posar en pràctica els coneixements adquirits al llarg de la carrera, on he après a utilitzar les eines necessàries per a poder dur a terme aquest treball.

Finalment, als meus companys de la universitat, per compartir grats moments d'alegries i anècdotes que mai oblidaré.

BIBLIOGRAFIA

[1] www.apachefriends.org/es/index.html, web oficial Composer amb informació sobre "A dependency Manager for PHP" (data últim accés: març 2021).

[2] getcomposer.org/, web oficial Composer amb informació sobre "A Dependence Manager for PHP" (data últim accés: març 2021).

[3] git-scm.com, web oficial Git amb informació sobre "git – distributed is the new centralized" (data últim accés: març 2021).

[4] laravel.com/, web oficial de Laravel amb informació sobre "The PHP Framework web artisans" (data últim accés: març 2021).

[5] code.visualstudio.com/, web oficial Visual Studio Code amb informació sobre "Code editiing redefined" (data últim accés: març 2021).

[6] kanbanize.com/es/recursos-de-kanban/software-kanban, web oficial de kanbanize amb informació sobre "Software Kaban: Explora las Oportunidades " (data últim accés: febrer 2021).

[7] M. Ahmad, J. Markulla, M. Oivo (2013). Kanban in software development: A systematic literatura review. In Procedings – 39th Euro micro Conference Series on Software Engineering and Advanced Applications IEEE Computer Society.

[8] www.atlassian.com/es/agile/scrum, web oficial Atlassian amb informació sobre "¿Qué es Scrum?" (data últim accés: març 2021).

[9] kanbanize.com/es/recursos-de-kanban/software-kanban, web oficial de kanbanize amb informació sobre "Software Kaban: Explora las Oportunidades " (data últim accés: febrer 2021).

[10] www.gestiondeportiva.com, web oficial de gestió Deportiva amb informació sobre el "mòduls de gestió administrativa del club esportiu que ofereix" (data últim accés: febrer 2021).

[11] www.sporteasy.net, web oficial de sport easy amb informació sobre el "Gestiona tu club deportivo de amateur" (data últim accés: febrer 2021).

[12] www.artima.com/articles/the-dci-architecture-a-new-vision-of-object-oriented-programming, web oficial d'artima amb informació sobre "Model Vista Controlador" (data últim accés: febrer 2021).

[13] laravel.com/, web oficial de Laravel amb informació sobre "The PHP Framework web artisans" (data últim accés: març 2021).

[14] justinmind.com, web oficial justinmind amb informació sobre "All-in-one prototyping tool for web and mobile apps" (data últim accés: abril 2021).

[15] richos.gitbooks.io/laravel-5/content/capitulos/chapter8.html, web oficial Gitbook amb informació sobre "model factories"

(data últim accés: maig 2021).
[16] bitbucket.org/cbgrupbarna/tfg-cbgrupbarna/src/master/, web oficial bitbucket amb informació sobre "tfg-cbgrupbarna" (data últim accés: maig 2021).
[17] https://blog.pleets.org/article/mvc-en-laravel web oficial pleets amb informació sobre "Buenas prácticas" (data últim accés: maig 2021).
[18] https://cosasdedevs.com/posts/crud-api-laravel-8-parte-1-modelos-creacion/, web oficial Wikiserver amb informació sobre "Laravel 8 CRUD Tutorial by Example" (data últim accés: juny 2021).

APÈNDIX

A1. PLANIFICACIÓ I FASE DEL PROJECTE

En la *figura 14*, es mostra el diagrama WBS del TFG "Aplicació per a la gestió de clubs esportius C.B. Grup Barna". Es composa de 6 tasques. En cada tasca es desglossa un conjunt de sub-tasques desenvolupades al llarg del projecte.

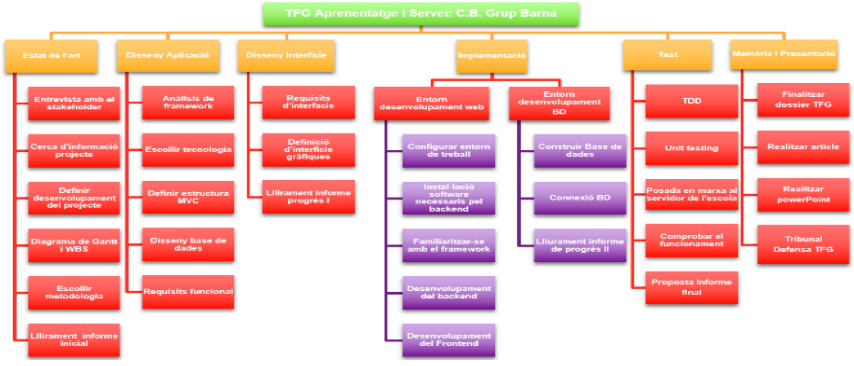

Figura 14: Diagrama WBS per visualitzar les fases del projecte.

En la *figura 15*, es mostra el funcionament de l'eina *Trello*. En la finestra "*Doing*" es veu la llista de tasques, tant de la fase d'Implementació com de la fase de Test que s'estàn executant i a l'espera d'acabar el dia 30 de maig. També s'observen les activitats etiquetades en color verd, que són les tasques finalitzades de les tres primeres fases. Les tasques que han sigut finalitzades no implica que no es pugui realitzar millores.


Figura 15: Tauler Trello amb les columnes que s'utilitzarán durant el projecte.

La *figura 16*, conté tota la feina que s'ha dut a terme en la primera fase i lliurament del primer informe. Està relacionat amb la investigació de tota informació sobre el disseny i implementació de l'aplicació. Es crea una llista de tasques i objectius per a assolir. I es fa el diagrama de *Gantt* per a la planificació al llarg del projecte (tasques descrites en 6 fases).

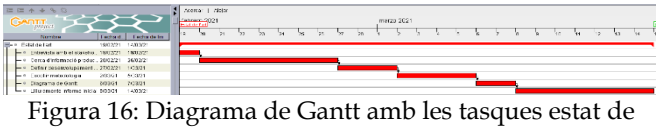
Figura 16: Diagrama de Gantt amb les tasques estat de l'art i disseny.

La *figura 17*, conté tota la feina que s'ha dut a terme en les fases de disseny de l'aplicació, disseny d'interfície i finalment lliurament del segon informe, que s'ha realitzat el 25 d'abril.

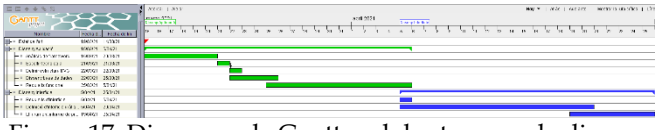
Figura 17: Diagrama de Gantt amb les tasques de disseny de l'aplicació i implementació.

La *figura 18*, conté totes les tasques que s'han dut a terme en la fase d'implementació, que s'ha dividit en dues parts: desenvolupament web i desenvolupament de la base de dades. A més, la fase d'implementació és la part essencial del projecte on s'ha realitzat el desenvolupament intern de la codificació de la pàgina web. Ha tingut una durada de sis setmanes.

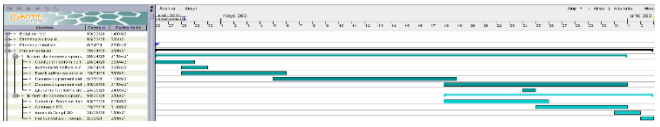
Figura 18: Diagrama de Gantt amb les tasques de la implementació del projecte.

En la *figura 19*, el diagrama de *Gantt* descriu les dues darreres fases del projecte. En el desenvolupament d'implementació de codi es realitza juntament amb la realització del test funcional per a verificar que la funcionalitat implementada sigui el correcte. La darrera fase es mostra el procés final a seguir.

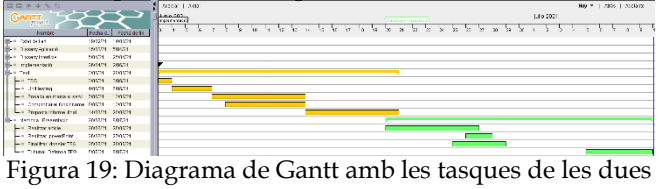
Figura 19: Diagrama de Gantt amb les tasques de les dues darreres fases del projecte.

A2. DISSENY DE LA BASE DE DADES

Basant-nos en el document de requeriments, és representen les dades utilitzant el model entitat relació (E/R). En la *figura 20* es mostra el disseny E/R, representat per 18 entitats que inclou atributs i claus primàries per a cada entitat. També es defineix un total de 21 interrelacions entre entitats i les seves connectivitat. Cal destacar que l'entitat Temporada és molt important, està molt lligada amb altres entitats com equips, categories, competicions i fase, doncs la temporada defineix moltes coses en un club (equips, jugadors, partits, etc).

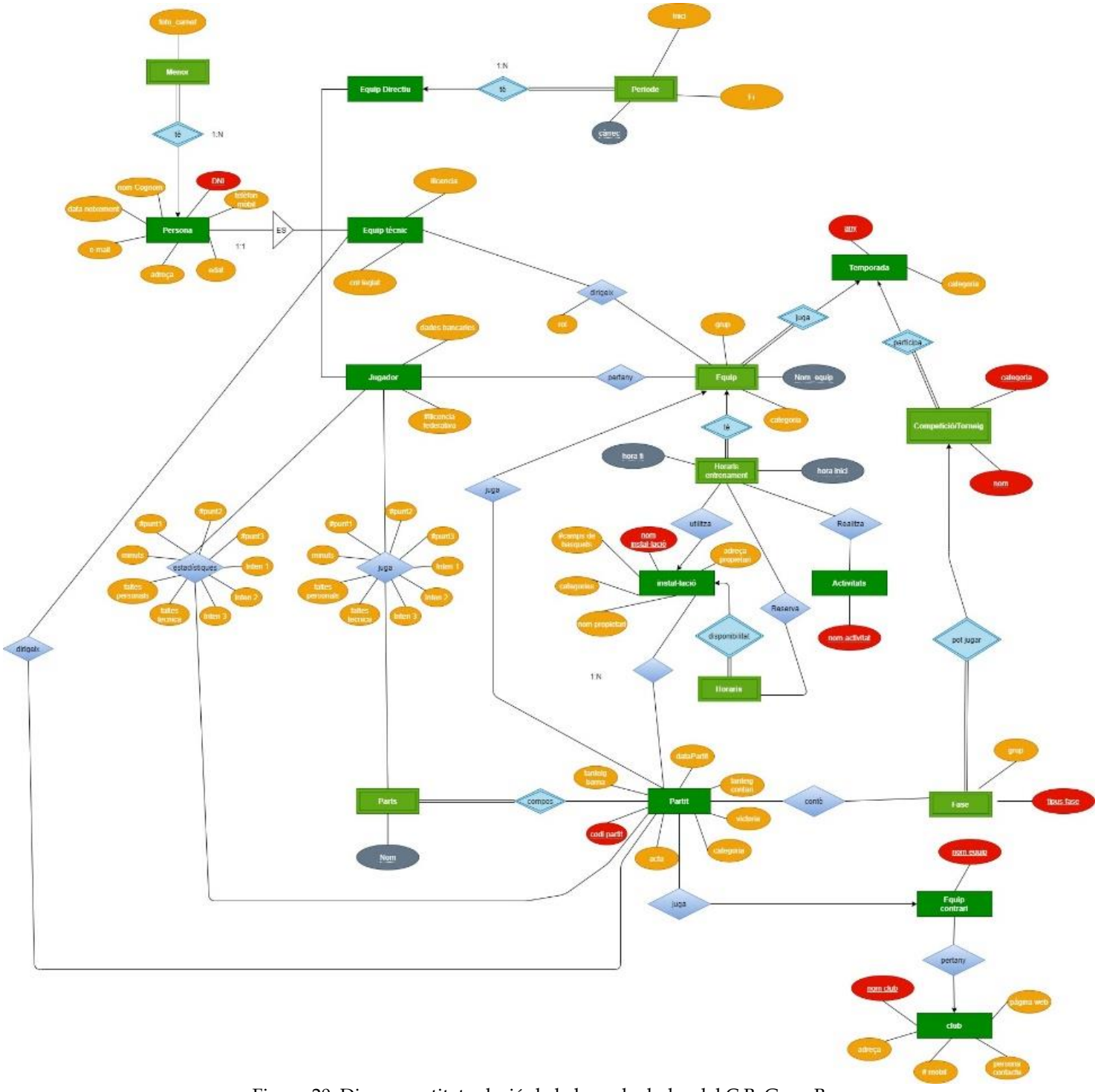

Figura 20: Disseny entitat relació de la base de dades del C.B. Grup Barna.

En la *figura 21* es mostra el diagrama relacional que presenta aspectes com: taules, claus primaries, foranies. És molt important remarcar, després d'executar les migracions (apartat *4.1 desenvolupament de la base de dades*) se n'han generat un total de 24 taules corresponent al disseny E/R (*figura 20*) més 5 taules generades a l'utilitzar *Laravel permision*.

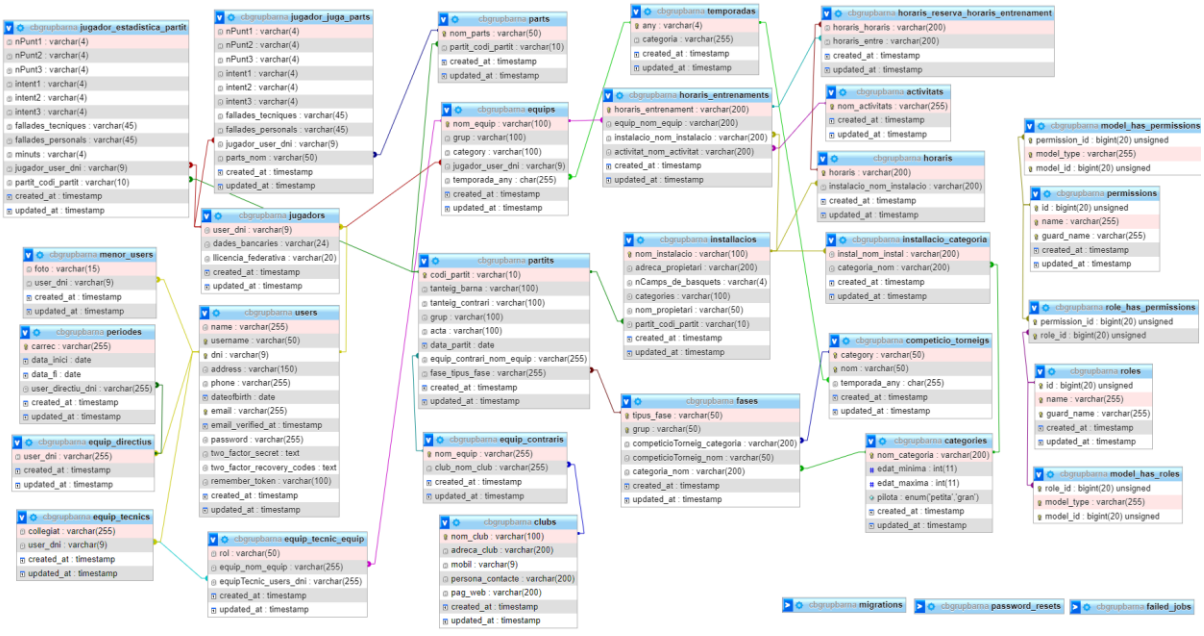

Figura 21: Diagrama relacional de la base de dades del C.B. Grup Barna.

A3. Instal·lació de laravel via composer

Per a instal·lar Laravel en un nou projecte d'aplicació és necessari complir els requisits fonamentals:

- ***PHP* 7.3 o superior**: es necessita instal·lar l'última versió del llenguatge *php*, doncs és uns dels requeriments previs de *Laravel 8*. La instal·lació de *XAMPP* conté l'última versió de *PHP* i mysql. En la pàgina oficial descarreguem l'instal·lador per a Windows, seguint els passos per a la seva instal·lació [1].
- **Composer:** És el gestor de dependència de *PHP*. En la pàgina oficial descarreguen i executem l'arxiu *Composer.exe*, que instal·la la versió més recent de composer [2].

a) Comanda d'instal·lació *Laravel*:

1. El projecte del TFG es realitza des de zero, pel que requereix instal·lar Laravel per a començar el seu desenvolupament. Ens situem en la següent ruta i executem la següent comanda:
   $ composer create-project --prefer-dist laravel/laravel cbgrupbarna

b) Comandes d'instal·lació de la interfície d'usuari (*Laravel UI*):

1. *Bootstrap* i *Vue scaffolding* proporcionats per *Laravel* es troben al paquet *Laravel/IU Composer*, que es pot instal·lar utilitzant composer:
   $ composer require laravel/ui
2. Un cop s'ha instal·lat el paquet *laravel/iu*, s'ha d'instal·lar el frontend *scaffolding*. S'executa la següent comanda d'artisan:
   $ php artisan ui bootstrap --auth

c) Comandes per a iniciar el procés d'instal·lació de *Laravel Fortify*:

1. Per a instal·lar *Fortify* s'utilitza l'administrador de paquets de composer:
   $ composer require laravel/fortify
2. A continuació, es publica els recursos utilitzant *vendor:publish*:
   $ php artisan vendor:publish --provider = ``Laravel/Fortify/Fortify/Fortify ServiceProvider"

A4. Connexió amb la base de dades

Laravel té suport per als motors de bases de dades més populars com: *MySQL*, *Postgresql*, *SQLite3*, *SQL Server*.
En aquest projecte s'ha utilitzat *MySQL*. En l'arrel del projecte s'ha de configurar el fitxer *.env*, on se li especifica el nom "cbgrupbarna", i el port "3306" de la base de dades per a realitzar la connexió amb MySQL (*figura 22*).

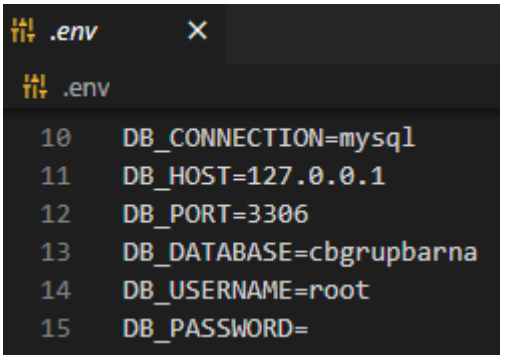

Figura 22: Fitxer de configuració "*.env*".

Configurat el fitxer "*.env*" i definides les migracions de taules seguint el disseny relacional, per a poder crear les taules en la base de dades *MySQL*, s'executa la següent comanda:

$ php artisan migrate

A la *figura 23* es mostra la interfície d'administració per a la base de dades *MySQL*. Es podem administrar les taules creades automàticament mitjançant les migracions definides en Laravel.

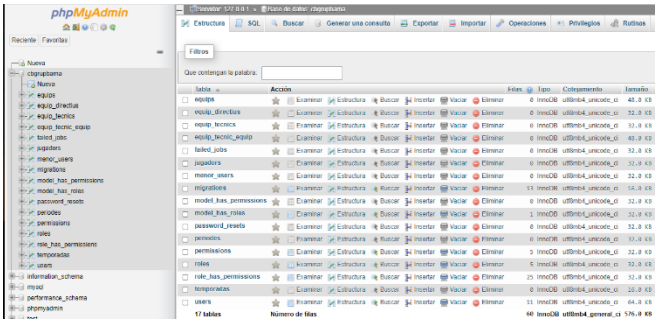

Figura 23: Panel d'administració de la base de dades a phpMyAdmin.

A5. DESENVOLUPAMENT WEB – ESTRUCTURA DEL DIRECTORI LARAVEL

En la figura 24, es mostra l'estructura *Laravel* del projecte "CBGRUPBARNA". Es veu una estructura de carpetes i fitxers per a organitzar el nostre codi. A continuació s'explica les carpetes que s'utilitzarà en aquest projecte:

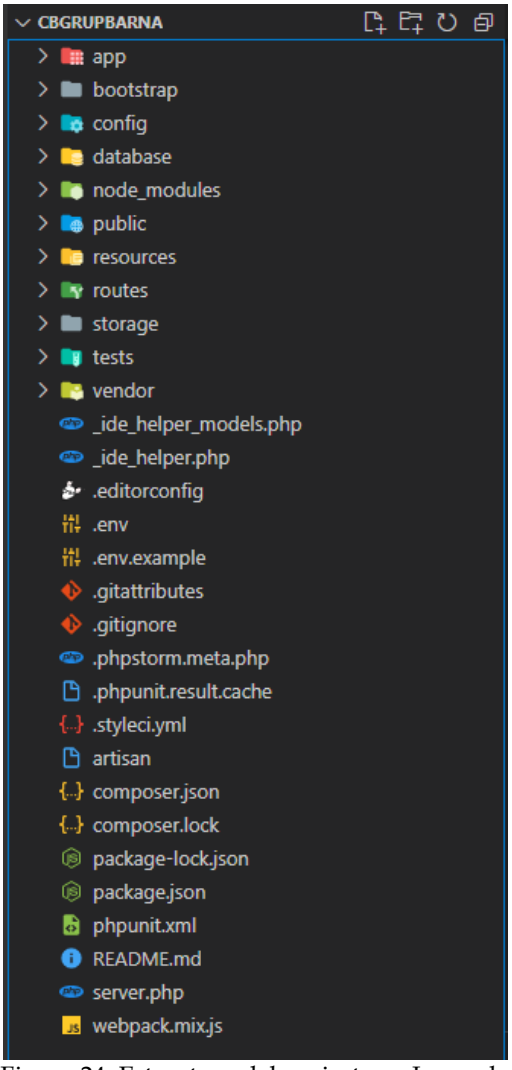

Figura 24: Estructura del projecte en Laravel.

- **APP:** Conté el codi principal de l'aplicació. Està subdividit en carpetes on s'especifiquen rutes, controladors, filtres i models de dades.
- **Config:** es troben tots els arxius de configuració de l'aplicació com base de dades, views, autenticació d'usuaris, permisos, mail, etc.
- **Database:** Aquesta carpeta inclou tot el relacionat amb la base de dades del nostre projecte. Dins podem trobar tres subcarpetes que són factories, migrations i seeds.
- **Públic:** Aquest directori és l'únic que ha de ser visible en el nostre servidor web. Totes les peticions i sol·licituds passen per aquest directori, doncs en ell es troba la pàgina principal de la nostra web o index.php (fitxer que inicia el procés d'execució del framework).
- **Test:** Aquest directori inclou un sistema que facilita tot el test de proves amb *PHPunit*.