

Títol:

Estudi potencialitat API FamilySearch

Autor: Sergi Porras Pagès

Data: 22 de setembre del 2016

Director: Enric Mayol Sarroca

Departament del director: ESSI

Titulació: Enginyeria en Informàtica (2003)

Centre: Facultat Informàtica de Barcelona (FIB)

Universitat: Universitat Politècnica de Catalunya (UPC)
BarcelonaTech

Continguts

1	Introducció al projecte	13
1.1	Introducció	13
1.2	Motivació i context	14
1.3	Objectius	15
1.4	Planificació	16
1.4.1	Planificació Febrer del 2014 - Juliol 2014	16
1.4.2	Planificació Febrer del 2016 - Setembre 2016	16
2	La genealogia	21
2.1	Què és la genealogia?	21
2.2	El paper de la genealogia en el transcurs de la història	22
2.3	Les lleis reguladores	23
2.4	Els codis ètics en la genealogia	24
2.5	El procés de recerca genealògica	25
2.6	Conclusió	26
3	L'organització FamilySearch	27
3.1	Què és FamilySearch?	27
3.2	La història de FamilySearch	28
3.3	L'església mormona i la família	29
3.4	Serveis per organitzacions amb arxius genealògics	31
3.4.1	Captura d'imatges	31
3.4.2	Conversió de formats digital	32
3.4.3	Indexació en línia	32
3.4.4	Accés en línia	32

3.4.5	Preservació dels registres i fitxers físics	32
3.4.6	Conclusions sobre els serveis professionals	33
3.5	Serveis per particulars i aficionats a la genealogia	33
3.5.1	Eines genealògiques	34
3.6	Recerca genealògica	37
3.6.1	Cerca de registres	37
3.6.2	Cerca de genealogies	38
3.6.3	Cerca per catàleg	39
3.6.4	Cerca en llibres d'història genealògica	39
3.6.5	Wiki de FamilySearch	39
3.7	Projectes d'indexació	39
3.8	Conclusió sobre les eines per particulars	40
4	Introducció a l'API de FamilySearch	41
4.1	El portal de desenvolupadors	41
4.2	L'arquitectura de l'Interfície de Programació d'Aplicacions (API) . .	42
4.2.1	Què és una API?	42
4.2.2	L'arquitectura REST	42
4.3	Formats de dades utilitzats pel Sistema	44
4.3.1	El format de dades GEDCOM i GEDCOM X	45
4.3.2	Format de dades FamilySearch	47
4.3.3	Format de dades Atom (o Atòmic)	47
4.3.4	Codificacions dels formats de dades	47
4.4	Evolució temporal de l'API	49
4.4.1	Evolucions al llarg del temps	49
4.4.2	Tree Foundation	49
4.5	Serveis extres oferts per l'API	51
4.5.1	Caching	52
4.5.2	Throttling	52
4.5.3	Sincronització	52
4.5.4	Internacionalització	53
4.5.5	Transferència de registres en grans quantitats	53
4.6	Eines de desenvolupament	54

CONTINGUTS	5
4.6.1 Els SDK de FamilySearch	54
4.6.2 Les aplicacions d'exemple	55
4.6.3 Altres eines intressants	55
4.7 Procés de certificació	55
4.8 Diferents entorns de treball	56
5 Estudi en profunditat de l'API de FamilySearch	57
5.1 Els recursos de FamilySearch	57
5.2 Enllaços hypermedia, la navegació entre recursos	57
5.3 L'arbre genealògic de FamilySearch	59
5.4 Recursos principals del bloc Persones	61
5.4.1 El recurs Persona (Person)	62
5.4.2 El recurs Gènere (Gender)	63
5.4.3 Els recursos Nom, Forma del Nom i Part del Nom (Name, NameForm, NamePart)	63
5.4.4 El recurs Esdeveniment (Fact)	65
5.4.5 El recurs Data (Date)	67
5.4.6 El recurs Referència de localització (PlaceReference)	67
5.4.7 El recurs Descripció de Localització (PlaceDescription)	68
5.4.8 El recurs Camps Bàsics (DisplayProperties)	69
5.4.9 El recurs Vista de Família (FamilyView)	70
5.5 Recursos principals del bloc Relacions Familiars	70
5.5.1 El recurs Relació (Relationship)	71
5.5.2 El recurs Relació Pares i Fill (ChildAndParentsRelationship)	72
5.6 Recursos principals del bloc Discussions	73
5.6.1 El recurs Referència a la Discussió (DiscussionsReference) . .	73
5.6.2 El recurs Discussió (Discussion)	74
5.6.3 El recurs Comentari (Comment)	75
5.7 Recursos principals del bloc Memòries	75
5.7.1 El recurs Referència a la Memòria d'una Persona (Person Memory Reference)	76
5.7.2 El recurs Persones en una Memòria (MemoryPersona)	76
5.7.3 El recurs Memòria (Memory)	77

5.8 Recursos principals del bloc Fonts de Dades	77
5.8.1 El recurs Col·lecció (Collection)	78
5.8.2 El recurs Contingut de la Col·lecció (CollectionContent) . . .	79
5.8.3 El recurs Atribució (Attribution)	80
5.8.4 El recurs Font de Dades (SourceDescription)	80
5.8.5 El recurs Referència a la Font de Dades (SourceReference) . .	82
5.9 Altres recursos interessants	83
5.9.1 El recurs Subjecte (Subject)	83
5.9.2 El recurs Conclusió (Conclusion)	84
5.9.3 El recurs Enllaços Hypermedia (Hypermedia Enabled Data) .	85
5.9.4 El recurs Enllaç (Link)	85
5.9.5 El recurs Dades Extensibles (Extensible Data)	86
5.9.6 El recurs Nota (Note)	86
5.9.7 El recurs Referència al Recurs (ResourceReference)	87
5.9.8 El recurs Usuari (User)	87
5.9.9 El recurs Canvi (Change)	88
5.9.10 El recurs Agent (Agent)	90
5.10 Camins d'accés a l'arbre familiar i operacions de cerca	91
5.10.1 L'accés directe a les dades	91
5.10.2 L'accés indirecte a les dades	92
5.10.3 Lectura de l'usuari identificat	93
5.10.4 Lectura de la persona relacionada a l'usuari identificat . .	93
5.10.5 Cerca de Persones a l'arbre familiar	94
5.10.6 Cerca de localitzacions	96
6 Valoració final sobre la potencialitat de l'API	99
6.1 Introducció	99
6.2 Distribució geogràfica de les dades	99
6.3 Dades contemporànies	100
6.4 Recursos i funcionalitats	101
6.5 Naturalesa de l'API	101
6.6 Utilització de l'API en el marc d'un PFC	102
6.7 Conclusió	102

7 Llista de propostes de projecte	105
7.1 Introducció	105
7.2 Comparació sobre la popularitat de noms	106
7.3 Portal de cerca localitzat al Català	106
7.4 Geolocalització d'un cognom en diferents nivells	107
7.5 La recerca genealògica i l'heràldica	107
7.6 Projectes d'indexació	108
7.7 La història de l'església mormona a través de FamilySearch	108
7.8 Diversitat geogràfica d'un cognom: Els nostres avantpassats	109
7.9 Les col·leccions de dades de FamilySearch	110
7.10 FamilySearch i la segona guerra mundial: Natalitat i Defuncions	110
7.11 FamilySearch i la segona guerra mundial: Increment en els casaments	111
7.12 FamilySearch i la segona guerra mundial: La llista de Schindler	111
7.13 La gran recessió o altres esdeveniments històrics	111
7.14 Comparacions amb els amics o seguidors de Facebook i Twitter	112
7.15 Comparació de dades genealògiques reals amb FamilySearch	113
7.16 Estudi de profunditat dels arbres familiars	114
7.17 Algoritme de marcatge de duplicats	114
8 Estudi tècnic de l'aplicació web	115
8.1 Decisió del tipus d'aplicació a implementar	115
8.2 Tecnologies i patrons utilitzats per l'aplicació web	116
8.2.1 El model: Model Vista Controlador	116
8.2.2 Les tres capes del disseny web (Front-end)	118
8.2.3 La tecnologia HTML 5	119
8.2.4 La tecnologia CSS	120
8.2.5 La tecnologia de plantilles Mustache	121
8.2.6 Les tecnologies Javascript i jQuery	122
8.2.7 La tecnologia Bootstrap	123
8.2.8 El SDK Javascript oficial de FamilySearch	125
8.2.9 La tecnologia Node.js	127
8.2.10 La tecnologia Express o Express js	127
8.2.11 Recapitulació de les tecnologies pel desenvolupament web	129

8.3	Tecnologies específiques pel desenvolupament de l'aplicació	129
8.3.1	La tecnologia GitHub	129
8.3.2	Node.js i Express en l'àmbit local	130
8.3.3	Tecnologia NPM	131
8.3.4	Paquet Nodemon	131
9	Javascript SDK Oficial de FamilySearch	133
9.1	Introducció al Javascript SDK	133
9.2	Emmascarant les crides REST a l'API de FamilySearch	134
9.3	Implementació basada en Promeses	134
9.4	Implementació pensada per la programació orientada a objectes	135
9.5	Model de dades quasi idèntic a FamilySearch	135
9.6	Captura d'errors	135
9.7	Intentar de nou peticions GET fallides	136
9.8	Gestió automàtica del Throttling	136
9.9	Autentificació mitjançant un pop-up	137
9.10	Autentificació automàtica	137
9.11	Emmagatzematge del Token en una cookie	137
9.12	Disparador automàtic de la funció d'expiració	137
9.13	Utilitzable des de diferents plataformes o capes	138
9.14	Suport en el desenvolupament	138
9.15	Conclusió sobre el SDK	138
10	Introducció a l'aplicació web	139
10.1	Accés a l'aplicació web i codi de l'aplicació	139
10.2	Requisits de l'aplicació web	140
10.2.1	Requisits funcionals	140
10.2.2	Requisits no funcionals	141
10.3	Estructura de l'aplicació web	142
10.3.1	Introducció a l'estructura de l'aplicació	142
10.3.2	Home o pàgina principal	143
10.3.3	Rerefons	143
10.3.4	Propostes de projecte	144

10.3.5 Detalls específics d'una proposta de projecte	145
10.3.6 Identificació amb FamilySearch	145
10.3.7 Exemples implementats	146
10.3.8 Funcionalitats de cerca, expansió geogràfica d'un cognom i evolució d'esdeveniments	146
10.4 Fitxers de l'aplicació web i la seva funcionalitat	146
10.5 Funcionament general de l'aplicació web	149
10.5.1 Creació del servidor	151
10.5.2 Accés a una pàgina del domini web	151
10.5.3 Interacció amb l'API de FamilySearch	151
10.5.4 Interacció amb elements del HTML	152
10.5.5 Conclusió	152
10.6 Detalls específics de la implementació	152
10.6.1 Introducció als detalls d'implementació	152
10.6.2 Estructura general d'una pàgina HTML:MustacheI	153
10.6.3 El fitxer header.html: Configuració de la pàgina	153
10.6.4 El fitxer navbar.html: La barra de navegació adaptativa	154
10.6.5 El fitxer pageTitle.html: Mustache II	155
10.6.6 El fitxer javascripts.html: Càrrega dels controladors	156
10.6.7 El fitxer index.html: El grid de Bootstrap I	156
10.6.8 El fitxer background.html: El grid de Bootstrap II	158
10.6.9 Els fitxers future-proposals.html i implemented-proposals.html: El grid de Bootstrap III i el component Thumbnail	158
10.6.10 El fitxer package.json: Complements de l'aplicació	159
10.6.11 El fitxer app.js: Funcionament del servidor	160
10.6.12 Configuració del motor d'impressió i carpetes	161
10.6.13 Configuració dels complements	161
10.6.14 Funcions de redirecció	162
10.6.15 Processament de peticions POST	163
10.6.16 Validació d'identificació	163
10.6.17 Configuració del servidor	163
10.6.18 El fitxer gaTagging.js: Enviant esdeveniments	164
10.6.19 El fitxer style.css: Configurant elements i classes pròpies	164

10.7 Certificant l'aplicació amb FamilySearch	165
10.7.1 Certificat d'autentificació	166
10.7.2 Certificat de lectura	166
10.7.3 Procés de certificació	167
10.8 Google Analytics	167
10.9 Optimització d'imatges	169
10.10 Hosting de l'aplicació	170
10.10.1 Fàcil configuració	170
10.10.2 Fàcil desplegament al núvol	171
10.10.3 Entorn de proves local	171
10.10.4 Decent versió gratuita	171
10.10.5 Fàcil escalatge de l'aplicació	172
11 Funcionalitats d'exemple amb FamilySearch	173
11.1 Introducció a les funcionalitats implementades	173
11.2 Identificació i desconexió a l'API de FamilySearch	173
11.2.1 Descripció de les funcionalitats	173
11.2.2 Detalls d'implementació	174
11.2.3 Aspectes d'usabilitat considerats	178
11.3 Cerca de persones a l'arbre familiar	179
11.3.1 Descripció de la funcionalitat	179
11.3.2 Recomenacions d'utilització	179
11.3.3 Detalls d'implementació	180
11.3.4 Aspectes d'usabilitat considerats	185
11.3.5 Principal interès d'ús	189
11.4 Evolució geogràfica d'un cognom	189
11.4.1 Descripció de la funcionalitat	189
11.4.2 Recomanacions d'utilització	191
11.4.3 Detalls d'implementació	192
11.4.4 Aspectes d'usabilitat considerats	198
11.4.5 Principal interès d'ús	202
11.5 Evolució temporal d'esdeveniments	202
11.5.1 Descripció de la funcionalitat	202

11.5.2 Recomanacions d'utilització	203
11.5.3 Detalls d'implementació	203
11.5.4 Aspectes d'usabilitat considerats	205
11.5.5 Principal interés d'ús	207

Secció 1

Introducció al projecte

1.1 Introducció

Aquest projecte neix de les conversacions amb Enric Mayol mentre explorava diferents opcions sobre quin projecte final de carrera realitzar. L'Enric em va introduir l'organització de FamilySearch i l'existència de la seva API, encarregada de gestionar l'accés a les dades d'índole genealògic.

FamilySearch és una organització sense ànim de lucre destinada a connectar famílies a través de generacions. La seva visió com a col·lectiu és el d'ajudar a les persones a crear un vincle amb els seus avantpassats, com a eina per poder comprendre millor qui són, crear un sentiment de família i teixir el pont entre passat i futur.

Com s'ha esmentat, les dades emmagatzemades per l'organització i accessibles a través de l'API són principalment de caràcter genealògic. En concret, es disposa d'una col·lecció de persones de les quals se'n coneix informació personal, esdeveniments rellevants en el transcurs de la seva vida, com podrien ser per exemple dades sobre el seu naixement i les seves relacions amb altres persones, en altres paraules, el seu arbre genealògic.

El projecte gira entorn aquesta API i ha estat dividit en tres grans blocs o seccions.

Per començar, realitzar un estudi profund de l'API. Això significa comprendre quines són les petites peces d'informació realment disponibles i com estan relacionades entre elles. D'aquesta forma, també s'ajudarà als futurs estudiants interessats a utilitzar aquesta API, a comprendre-la i poder començar a utilitzar-la, amb molta més facilitat.

En segon lloc, i com un dels tres blocs principals, utilitzant el coneixement adquirit durant l'estudi de l'API, així com les oportunitats i limitacions imposades per la plataforma, conegeudes durant la implementació dels exemples, plantejar un conjunt de propostes de projecte que puguin servir a futurs estudiants com a suport

i inspiració.

Finalment, l'últim bloc del projecte consisteix en implementar una aplicació que interactuï amb l'API de FamilySearch a través de diferents exemples. L'objectiu dels exemples és el de facilitar l'observació i comprensió del potencial de l'API, exposar-ne la informació emmagatzemada i oferir idees sobre com encarar-ne l'explotació.

1.2 Motivació i context

Durant el transcurs de la carrera són moltes i diverses les vessants de la informàtica que ens van ser introduïdes. D'aquestes, sempre vaig sentir més afinitat per aquelles que requerien allunyar-se un pèl dels detalls més tècnics i s'enfocaven en un exercici d'abstracció i conceptualització centrat en la comprensió d'un problema quotidià, el disseny d'una solució i finalment, la seva execució.

Entenc doncs, que donada la situació, no és d'estranyar que sentís una empatia més elevada per camps com la mineria de dades o l'enginyeria del software que no pas l'algorítmica o els sistemes operatius. Al mateix temps, sempre m'he considerat una persona dispersa a qui li agrada conèixer una mica de molts temes diferents i per tot això, buscava un projecte que em permetés trencar en certa forma amb aquests aspectes més tècnics de la informàtica i explorar un camp desconegut al qual es poguessin aplicar els coneixements adquirits durant la carrera.

El projecte proposat per l'Enric complia doncs, en bona mesura, amb tot allò que jo buscava. La genealogia representava un camp que desconeixia per complet i el projecte en si era un full de ruta obert. Només la realització de la primera part d'aquest ens permetria comprendre com de profundes i completes podrien ser les propostes que el projecte originaria o quins serien els exemples a implementar.

En aquest aspecte, el projecte resultava especialment atractiu, doncs la riquesa final d'aquest vindria donada per la capacitat de traduir els objectius d'un camp d'estudi com la genealogia, mitjançant el grup de dades disponible a través de FamilySearch, en propostes que fossin capaces de satisfer preguntes o inquietuds latents en la societat. A la vegada, aquest mateix aspecte convertia el projecte en aterridor, doncs no seria fins ben entrat en aquest, que comprendríem les opcions disponibles i fins a quin nivell podríem aprofundir en l'elaboració de propostes i exemples.

Un últim aspecte que em va ajudar a decidir-me per aquest projecte va ser la influència de la feina d'interí que estava realitzava en aquell moment com a dissenyador d'experiència d'usuari. El projecte m'oferia la possibilitat d'implementar una pàgina web com a apartat tècnic i aquesta em permetia posar a prova els coneixements d'usabilitat i experiència d'usuari adquirits durant els darrers mesos. De la mateixa forma, s'obria la porta a desenvolupar les habilitats necessàries per programar una pàgina web, una part de la informàtica que no havia explorat durant la carrera i em feia certa gràcia.

1.3 Objectius

Tot i la incertesa de quins seran els detalls finals del projecte en el moment de començar, l'abast dels objectius principals sí que es troba ben marcat i definit. Com hem comentat, els objectius principals del projecte giren al voltant de tres grans blocs:

- Estudi de l'API de FamilySearch
 - Estudi de les peces d'informació accessibles i utilitzables.
 - Relacions entre les diferents peces d'informació.
 - Fer transparent, als futurs estudiants, el potencial real de les dades emmagatzemades, els principals obstacles a tenir en compte si es vol utilitzar aquesta API i la viabilitat d'utilitzar FamilySearch a l'hora de realitzar un projecte final de carrera.
- Bateria d'idees relacionades amb l'API de FamilySearch tenint en compte les restriccions del sistema i la informació disponible per servir com a futurs projectes o com a font d'inspiració.
- Evaluació i implementació d'exemples que es comuniquin o nodeixin de l'API de FamilySearch, n'exposin la informació disponible i ofereixin una idea bàsica de les oportunitats i complicacions que aquesta comporta. Aquest objectiu es divideix en diversos apartats:
 - Estudiar les diferents opcions disponibles per la implementació.
 - Escollir el tipus d'aplicació a desenvolupar i estudiar les tecnologies necessàries per desenvolupar l'aplicació.
 - Definició de l'abast i esquelet de l'aplicació, així com dels exemples que seran desenvolupats i definició dels requisits funcionals i no funcionals de l'aplicació.
 - Implementació de l'aplicació.
 - Procés de certificació i proves del sistema.

Aquest projecte també presenta un seguit d'objectius secundaris o més aviat, objectius personals, que en certa forma m'agradaria deixar plasmats en la memòria.

- Adquisició del coneixement necessari sobre el funcionament d'una pàgina web, quins són els seus components principals i com interactuen.
- Estudi d'algunes de les tecnologies més usades en el mercat actualment.
- Aplicació bàsica dels coneixements d'usabilitat i experiència d'usuari adquirits durant l'etapa d'interí.
- Aprenentatge del llenguatge de maquetació de text LaTeX per tal de formatar articles i documents de caràcter tècnic.

1.4 Planificació

L'objectiu d'aquest apartat de la memòria és presentar les diferents planificacions que s'han portat a terme per encarar el projecte en cada una de les convocatòries matriculades.

1.4.1 Planificació Febrer del 2014 - Juliol 2014

Aquest projecte va ser matriculat per primera vegada al Febrer del 2014 amb la intenció de presentar-lo com a principis de juliol del mateix any. La idea inicial era aprofitar el mes de gener per avançar feina i disposar així d'un total de sis mesos per realitzar el projecte.

Al març del 2014 vaig començar a treballar a jornada completa i el projecte va deixar d'avançar a la velocitat esperada. Es van començar a patir forts endarreriments sobre la planificació original fins al punt que el projecte va quedar completament aturat. La figura 1.1 mostra en línies generals la planificació que s'hagués volgut portar a terme en cas de normalitat i ressaltat en vermell la part que es va veure interrompuda.

FASE DEL PROJECTE / MES	Gener 2016	Febrer 2016	Març 2016	Abril 2016	Maig 2016	Juny 2016	Juliol 2016
Estudi de la API							
Generació d'idees per futurs projectes							
Estudi de les tecnologies necessàries							
Implementació de l'aplicació							
Redacció de la memòria							
Preparació de la presentació							

Figura 1.1: Planificació original *Febrer 2014 - Juliol 2014*.

La falta de temps per realitzar un projecte acceptable, conjuntament, a la poca capacitat de maniobra de les que es va disposar entre els mesos de Març i Juliol, va provocar que es descartés la possibilitat de presentar el projecte durant la convocatòria prevista inicialment.

1.4.2 Planificació Febrer del 2016 - Setembre 2016

A conseqüència de l'extinció del pla d'enginyeries 2003, el projecte es torna a matricular al Febrer del 2016, tenint en consideració que s'hauria de començar pràcticament de 0.

Donada la diferència de temps entre la primera inscripció i la segona, l'API de FamilySearch s'havia vist sotmesa a grans canvis i la major part del material estudiat i coneixements tècnics adquirits fa dos anys, quedaven completament antiquats.

A pesar de matricular el projecte a mitjans de Febrer es coneixia que aquest no podria ser començat amb agilitat fins a principis d'abril a causa d'una situació excepcional en l'àmbit laboral. Gràcies a la disponibilitat d'una pròrroga extraordinària, que permetia estendre el període d'entrega fins a finals de setembre, la finestra de temps disponible per completar el projecte rondava els cinc o sis mesos.

Cal tenir en compte que la disponibilitat horària en el dia a dia de cara a treballar en el projecte era molt reduïda i en conseqüència, realitzar una bona planificació era essencial si es volien evitar els mateixos problemes que van provocar l'abandonament del projecte en el seu primer intent.

Tant en la figura 1.2, com en les seccions que segueixen a continuació, expliquem com va ser planificada i executada la feina entre els mesos d'abril i setembre.

Segona quinzena de Març

En aquest petit període de temps es va realitzar el primer estudi superficial sobre l'API de FamilySearch amb la finalitat d'observar quins canvis s'havien produït durant els darrers dos anys i com aquests podien afectar o modificar la proposta inicial inscrita del projecte.

L'objectiu d'aquesta repassada ràpida era la de proporcionar una visió global sobre certes limitacions que podrien afectar el desenvolupament del projecte i ens permetés elaborar una planificació coherent de com afrontar i estructurar la feina a realitzar.

Primera quinzena d'Abril

Tot i que l'estudi sobre la informació disponible a través de l'API es troava en els seus inicis es vaaprofitar aquesta quinzena per decidir quina mena d'aplicació volíem implementar. Aquesta decisió obriria pas a la recerca i estudi sobre quines tecnologies serien més adients de cara a la implementació dels exemples i les comunicacions amb l'API de FamilySearch.

També s'aprofitaria aquesta quinzena per familiaritzant-nos amb la diferent documentació disponible sobre l'API de FamilySearch i plantejar-ne l'ordre d'estudi.

Segona quinzena d'Abril - Finals de Maig

Aquest període inicial del projecte resultava crucial de cara a incorporar les eines necessàries al nostre coneixement que ens permetrien completar un dels objectius principals del projecte durant els mesos següents.

SECCIÓ 1. INTRODUCCIÓ AL PROJECTE

26/09/2016																								
19/09/2016																								
12/09/2016																								
05/09/2016																								
29/08/2016																								
22/08/2016																								
15/08/2016																								
08/08/2016																								
01/08/2016																								
25/07/2016																								
18/07/2016																								
11/07/2016																								
04/07/2016																								
27/06/2016																								
20/06/2016																								
13/06/2016																								
06/06/2016																								
30/05/2016																								
23/05/2016																								
16/05/2016																								
09/05/2016																								
02/05/2016																								
25/04/2016																								
18/04/2016																								
11/04/2016																								
04/04/2016																								
28/03/2016																								
21/03/2016																								
FASE DEL PROJECTE / SETMANA	1	2																						
Primer estudi superficial de l'API	1	2																						
Decisió tipus d'aplicació a implementar i definició de l'estructura		1	2																					
Decisió i estudi de les tecnologies a utilitzar per l'aplicació web			1	2	3	4	5	6							6,5	7	7,5	8						
Propostes de projecte per futurs estudiants i estudi de l'API				1	2	3	4	4,5	5	5,5	6						6,5	7	7,5	8				
Implementació de l'aplicació web					1	2	3	4	5	6	7	8	9				9,5	10	10,5	11	11,5			
Redacció de la memòria final																	1	2	3	4	5	6	7	
Preparació de la defensa del projecte																								

Figura 1.2: Planificació final Febrer 2016 - Setembre 2016.

Així doncs, l'objectiu era el de detallar i estudiar tots els aspectes referents a la part tècnica de l'aplicació. Escol·lir de forma correcta era indispensable si volíem evitar tancar-nos portes abans de començar o assegurar-nos de què utilitzàvem eines eficients que oferien un bon balanç entre esforç i qualitat. Els punts coberts durant aquesta fase van ser:

- Esbrinar els components principals que conformen una pàgina web avui en dia i com interactuen.
- Conèixer les diferents tecnologies disponibles per cada un d'aquests components.
- Escol·lir del grup de tecnologies estudiat les més adients per fer front als objectius del projecte i estudiar-les a fons.

El segon objectiu d'aquesta fase consistia en estudiar a fons l'API de FamilySearch per obtenir una idea més concreta quina mena de projectes es podrien arribar a realitzar i quins no. Aquest exercici ens ajudaria a comprendre quins exemples tindria més sentit implementar per tal de demostrar la potencialitat i abast de l'API.

Mesos de juny i juliol

Gairebé tot l'esforç durant aquests dos mesos es concentraria en la implementació de l'aplicació web. Es va decidir prioritzar aquesta tasca per sobre de la generació d'idees i l'estudi final de l'API per dos motius.

Gairebé tot l'esforç durant aquests dos mesos es concentraria en la implementació de l'aplicació web. Es va decidir prioritzar aquesta tasca per sobre de la generació d'idees i l'estudi final de l'API per dos motius.

A pesar dels avantatges que oferia desenvolupar l'aplicació en primer lloc, com a contrapartida, també significava tirar endavant una part important del projecte amb el risc de no haver arribat a conèixer la totalitat de l'abast de l'API.

Mes d'agost

L'objectiu del mes d'agost era fer front a tota aquella part del projecte que havia quedat oblidada fins aquest moment. En concret:

- Redactar la memòria del projecte.
- Detallar les diferents propostes de projecte pels futurs estudiants.
- Implementar algunes parts dels continguts estàtics de l'aplicació web.

Mes de setembre

El mes de setembre s'utilitzaria com a marge de maniobra per acabar de tancar aquelles tasques del projecte que poguessin estar sotmeses a petits retards. Segurament, l'acabat de redacció de la memòria i petits retocs en l'aplicació web.

També s'aprofitaria la part final del més, un cop el projecte estigués entregat, per preparar la defensa.

Secció 2

La genealogia

Resultaria estrany realitzar un projecte que parla o tracta la genealogia en tots els seus apartats i no realitzar una petita introducció que exposi en què consisteix aquesta ciència.

No representa un objectiu del projecte comprendre l'estat actual de la genealogia en el món contemporani, ni el de crear un dibuix detallat de quines lleis en regulen les seves activitats. No obstant això, sí que es creu que donar una petita visió general dels problemes i preguntes que aquesta ciència pretén abordar pot ajudar a lectors del projecte o futurs estudiants a comprendre millor les limitacions i oportunitats d'aquest sector.

2.1 Què és la genealogia?

La Genealogia (del grec: '*genea*', '*generació*'; i, '*logos*', '*coneixement*') és també coneguda pel nom d'història familiar. Aquesta ciència consisteix en l'estudi de les famílies, el seguiment dels seus llinatges, tant ascendents com descendents i l'estudi de la història de les persones.

Els genealogistes, o persones dedicades a la genealogia, tant en l'àmbit privat com personal, utilitzen com a recurs d'investigació arxius històrics rics en dades. Exemples d'aquests recursos poden ser les partides de naixement, documents de defunció, registres d'emigració o altres documents informatius del mateix caire. L'objectiu d'aquests documents és obtenir informació sobre una persona o família per així poder demostrar relacions de parentesc i llinatge o bé, fets empírics relatius a la vida d'un individu en concret.

Un altre recurs que es veu cada cop més utilitzat és l'anàlisi genètic, mètode que té una rebuda, demanda i interès més elevat en l'àmbit personal, que no pas en el científic. La finalitat principal d'aquest mètode és la d'esbrinar relacions familiars passades i presents de l'individu a través de l'anàlisi dels seus gens.

Els motius pels quals una persona pot estar interessada a endinsar-se en el món

de la genealogia són diversos. Un exemple podria ser el desig de situar la seva família en un marc més ampli dins de la història o bé, el sentiment de responsabilitat de cara a preservar la història familiar per les futures generacions.

Els aficionats a la genealogia, que la practiquen com a hobby, generalment investiguen la seva ascendència o la d'una persona propera. Per altra banda, els professionals, acostumen a encarregar-se de realitzar recerques genealògiques per tercers, estudiar i ensenyantar mètodes de recerca o mantenir les seves pròpies bases de dades.

Cal entendre que la genealogia no tracta només de recopilar informació sobre el moment històric en què una persona va néixer, viure o morir, sinó també el de recollir informació sobre l'estil de vida que aquella persona va portar, les seves biografies o quins van ser els esdeveniments i motivacions que van conduir i marcar la seva existència. En altres paraules, podríem dir que una part de les preguntes que la genealogia pretén respondre és la de com van viure o quin caràcter van mostrar els nostres avantpassats al viure durant el transcurs d'esdeveniments històrics, com per exemple, la segona guerra mundial.

Voldríem tancar aquesta secció indicant que si l'interès per la genealogia, ha anat en augment en els últims temps, és en gran part gràcies a la digitalització de documents, fet que ha permès que genealogistes amateurs disposin d'un ventall d'eines molt superior al que van disposar els seus avantpassats i per tant, que les possibilitats de mantenir un arbre familiar o realitzar recerca genealògica quedin a l'abast de tothom.

2.2 El paper de la genealogia en el transcurs de la història

Com s'ha comentat en l'apartat anterior, avui en dia la genealogia és una ciència que busca en gran mesura respondre preguntes de caràcter personal, no obstant això, aquest no va ser sempre el seu objectiu principal.

Històricament, en les cultures occidentals, les persones estaven interessades a mantenir-se ben informades sobre la seva ascendència de cara a fer latents les seves connexions amb nobles i governants. Generalment, la intenció era protegir la seva situació privilegiada o escapar de la precarietat. En aquesta època, el terme genealogia compartia significat amb el d'heràldica, terme usat avui en dia per la ciència que estudia els escuts d'armes. Així doncs, fins a finals del segle XIX, la genealogia deixava la seva marca en la història com a eina utilitzada principalment per aquells amb drets de poder o riquesa adquirits a través de l'erència.

Aquest exemple, que bé ens podria semblar distant en el temps, no és l'única mostra dels impactes històrics relacionats amb aquesta ciència i com veurem a continuació, existeixen altres exemples molt més propers.

No cal tornar gaires anys enrere per veure com durant l'època de l'alemanya nazi ser capaç de demostrar l'afiliació a la "raça suprema" era necessari per sobreviure o inclòs poder casar-se de forma legal. Per aquest motiu, no ens ha d'estranyar que avui en dia, Alemanya, segueixi sense fer públics la major part dels registres

genealògics del segle XX, doncs els fets històrics han portat a percebre la història familiar com un atac, o amenaça, a la privacitat i seguretat de les persones. Les conseqüències d'aquesta època de la història són conegudes per tothom i un no pot evitar entreveure certes relacions amb el camp de la genealogia.

Per situar un exemple que ens ocupa si pot ser encara més de ple, podem veure el valor de la memòria històrica i dels sentiments d'unió amb els nostres avantpassats arran de la gran quantitat de publicacions i missatges personals, recordant als seus avantpassats i als temps que els va tocar viure, en relació al vuitantè aniversari de l'esclat de la guerra civil espanyola. De fet, Catalunya és un altre clar exemple contemporani, conjuntament amb Alemanya, de com la memòria històrica pot ser present en la cultura, vida i sentiments de bona part d'una nació. Tant en l'àmbit personal, com col·lectiu.

Així doncs, podem concloure que la genealogia, no tant com a ciència sinó com eina, va desenvolupar, desenvolupa i probablement, seguirà desenvolupant, un paper important en la història de la humanitat. No hem d'oblidar que els problemes racials segueixen molt presents en l'actualitat de les nostres societats, Estats Units, n'ha estat últimament un clar exemple, i que és la raça sinó una característica més de les nostres característiques de naixement, o en altres paraules, de les nostres dades genealògiques.

2.3 Les lleis reguladores

Les seccions anteriors han introduït i descrit les ocupacions principals de la genealogia en els àmbits professional i amateur, així com el paper d'aquesta en la història. També s'ha esmentat que moltes de les dades amb les quals aquesta ciència interactua són de caràcter personal i per tant, sensibles a un ús impropri si no són regulades i protegides sota certes circumstàncies.

És per aquest motiu que bona part de les dades públiques enregistrades per l'estat, sobretot aquelles que afecten a persones que encara són活ives, es troben regulades sota un conjunt de lleis i legislacions. Aquestes lleis varien de nació en nació i per tant, no existeix un estàndard de quina informació és accessible pel domini públic, quina no i sota quines circumstàncies aquesta informació pot ser accedida.

En el cas de l'estat espanyol són dues les principals lleis que regulen l'accés a les dades genealògiques. La llei orgànica de protecció de dades (LOPD) i la legislació consolidada: Llei 20/2011, del 21 de juliol del Registre Civil.

El registre civil espanyol conté informació detallada d'una persona relacionada amb el seu naixement, relacions d'ascendència i descendència, nom i cognoms, emancipació, declaracions de concurs o suspensió de pagaments, nacionalitat, etcètera, etcètera. Com podem veure, aquest registre conté tota mena d'informació sensible i al mateix temps, de gran valor de cara a estudis genealògics.

Els habitants d'Espanya poden demanar accés a l'entrada d'una persona al registre civil mitjançant la presentació d'una sol·licitud digital, escrita o presencial. Per

aconseguir aquesta informació caldrà proporcionar tan dades personals pròpies com el motiu pel qual es vol poder accedir a la partida en concret. Motius recurrents són l'estudi genealògic, gestions administratives o simplement la recaptació d'informació.

Per altra banda, accedir al gruix de la informació no és fàcil i els genealogistes porten xocant amb portes tancades des de fa molts anys. Relacionat amb aquest aspecte, durant l'any 2011 es va aprovar una nova llei del Registre Civil que tenia com a objectiu racionalitzar l'estructura del registre i desjudicialitzar-lo. Aquesta llei havia d'entrar en vigor a partir del 2014, data que va ser posposada fins al juliol del 2015 i recentment ha tornat a ser ajornada fins al 30 de juny del 2017.

La part que farà referència sobre si el nou registre contemplarà l'accés al públic de cara a la recerca genealògica encara està a l'aire, però sembla que hi ha certa esperança gràcies a la inclusió del següent apartat en l'article 80:

4. Amb caràcter excepcional i amb finalitats d'investigació familiar, històrica o científica, es podrà autoritzar l'accés a la informació regstral en els termes que reglamentàriament s'estableixin.

Així doncs, sembla que un futur no molt llunyà, aquesta informació podria passar a ser explorable en grans escala de cara a estudis familiars, històrics o científics. Això si, sempre respectant la LOPD.

2.4 Els codis ètics en la genealogia

El fet que els genealogistes tinguin accés i treballin amb informació pública, però simultàniament, personal, provoca que la professió es vegi envoltada de codis ètics que tractin de protegir la integritat d'aquesta, el sentiment de professionalitat i la moralitat d'aquells que interactuen amb les dades.

Els codis morals giren al voltant de dos eixos. La protecció de la informació referent a les persones活es i les bones pràctiques de cara a la manipulació i tractament de les dades.

El primer eix, tal com hem indicat, fa referència a protegir a aquelles persones que encara són活es. En concret, es tracta d'evitar, en la mesura que sigui possible, publicar informació de caràcter personal que pogués resultar compromesa per un individu en concret.

El codi moral que hi ha en el rerefons és que a cap persona li agradaria trobar informació personal publicada al núvol, on tothom la pot accedir, pel simple fet que es tracta d'informació 'pública'. Per tant, cal respectar la privacitat de les persones i no publicar fets o dades compromeses independentment de l'opinió personal del genealogista.

Com ja s'ha mencionat amb anterioritat, la genealogia tracta en gran mesura d'estudiar els nostres avantpassats i la història de tots aquells que van existir abans

que nosaltres, per tant, el xafardeig i la tafaneria no tenen cabuda dins dels codis ètics i morals de la professió.

La segona branca ètica es correspon a un seguit de bones praxis de cara a la utilització d'informació genealògica. Tot i que no es tracta d'un manual oficial, la següent llista de 'regles' serveix per descriure i fer-nos una idea amb un alt grau de fiabilitat, del que significa el concepte de bones praxis en aquesta ciència:

- El genealogista mantindrà les fonts de referència i les citarà quan utilitzi dades fetes públiques per un altre individual o col·lectiu.
- El genealogista no compartirà ni utilitzarà informació no contrastada o amb altes probabilitats de ser errònia.
- El genealogista transmetrà seguretat i confiança a aquells que facin ús dels seus serveis.
- El genealogista donarà suport a aquelles iniciatives que preservin els fitxers públics i l'accés a aquests.
- Es tractarà amb cordialitat i respecte al personal de les facilitats d'investigació.
- El genealogista ajudarà en la mesura que sigui possible als altres genealogistes i organitzacions dedicades a la genealogia.
- El genealogista compartirà els resultats dels seus estudis i investigacions.
- No està permesa la invenció ni exageració de la informació.
- El genealogista complirà amb les lleis en rigor dels conjunts de dades que utilitzarà en els seus estudis.

2.5 El procés de recerca genealògica

A tota persona que li interessi realitzar recerca genealògica sobre la seva pròpia família o la d'una persona propera, se li suggereix el següent procés com a mètode de treball.

El procés de recerca es desenvolupa en cicles i cada cicle consta de cinc fases. A continuació es detallen una a una:

1. **Identificar el que es coneix:** Identificar i revisar tota la informació inicial que es coneix. Pel final de la fase s'haurien de tenir recopilats, ordenats i documentats tots els esdevenimentals relacionats amb la família o persona a estudiar disponibles.
2. **Decidir que es vol aprendre:** L'objectiu d'aquesta fase és identificar sobre quin individu es vol obtenir informació, que es vol aprendre d'aquesta persona i si és possible el temps i llocs aproximats en els quals aquesta persona va viure.
3. **Seleccionar els arxius a consultar:** Aquesta fase resulta la més complexa de tot el procés. L'objectiu, ordenar de més a menys útils les diferents fonts de dades a consultar i quins arxius resulten més interessants.

4. **Obtenir i consultar els arxius:** Durant aquesta fase es consultaran les fonts de dades seleccionades a l'apartat anterior. Al final d'aquesta, hauríem de tenir anotat tot allò que s'ha descobert i còpies dels documents que suporten els descobriments, ja sigui en format de fotocòpies, notes o qualsevol altra mena de suport físic o digital.
5. **Utilitzar la informació:** Finalment, en aquesta fase, tocarà avaluar la informació descoberta, transportar la nova informació als formularis adients, organitzar la informació i compartir els resultats. Un cop finalitzades totes aquestes tasques s'estarà preparat per tornar a iniciar la roda del procés i seguir així amb la recerca genealògica.

La figura 2.1 mostra les cinc fases d'aquest procés cíclic.

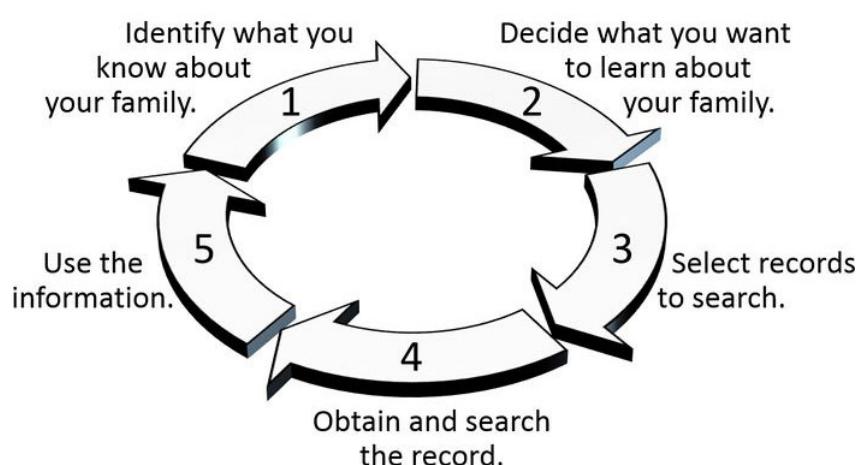


Figura 2.1: Procés de recerca genealògica.

2.6 Conclusió

Donem d'aquesta forma per conclosa la breu introducció a la genealogia. Com s'ha pogut observar, es tracta d'una ciència d'especial vocació personal, amb regulacions ambigües i rodejada de codis ètics i morals.

Tot i les dificultats que tant genealogistes amateurs com professionals poden haver de fer front, aquesta ciència és capaç de desemascarar i ajudar a comprendre les vivències dels nostres avantpassats i crear així un enllaç entre passat, present i futur.

Secció 3

L'organització FamilySearch

En aquest apartat de la memòria s'introduirà l'organització de FamilySearch. En concret, s'explicaran els objectius i motivacions sota les que va néixer l'organització, la seva història i el conjunt de funcionalitats i serveis que ofereixen a través del seu portal web i centres d'investigació.

3.1 Què és FamilySearch?

Si haguéssim de resumir l'organització en una sola paraula, aquesta seria família. Com s'indicava en la introducció, FamilySearch és una organització sense ànim de lucre destinada connectar famílies a través de diferents generacions. Des de l'organització es creu que les famílies són un condicionador de la felicitat i un dels factors que donen sentit a la vida.

La seva visió com a empresa, amb les seves pròpies paraules, és:

Aprendre dels nostres avantpassats, a través dels llaços familiars, ens ajudà a comprendre millor qui som, enllaçar el present amb el passat i construir els ponts cap al futur.

Aquesta visió és perseguida mitjançant un equip de professionals i voluntaris que treballen de cara a preservar i compartir el que s'ha convertit en la col·lecció més gran del món d'arxius genealògics. Aquest fet no és fruit de la casualitat, sinó dels més de cent anys que aquesta organització porta recol·lectant, preservant i compartint arxius genealògics des de tots els indrets del globus terraquí.

FamilySearch organitza i tracta els recursos disponibles tenint present que la finalitat de les dades és poder ser consumides per tercers. Per això, FamilySearch apostà per un model de dades eficaç, on aquelles persones interessades a explorar el seu passat, puguin descobrir amb certa facilitat quin són els seus orígens.

Algunes de les característiques o dades que ajuden a comprendre l'envergadura i èxit d'aquesta organització se citen a continuació:

- Servei gratuït.
- Més de 4 bilions de persones diferents emmagatzemades en el sistema. Per tenir una idea de la magnitud d'aquest nombre, actualment s'estima que en el món viuen 7,4 bilions de persones.
- 4.765 centres d'investigació distribuïts arreu del món.
- Servei d'ajuda 24/7.

3.2 La història de FamilySearch

FamilySearch, en els seus orígens coneguda com la Societat Genealògica de Utah, va néixer de les mans de l'església de Jesucrist dels Sants dels Darrers Dies l'any 1894. Aquesta església també és coneguda avui en dia pel nom de l'església mormona.

L'organització porta acumulats a les seves espatlles més de cent anys de recerca i preservació d'arxius històrics i genealògics. Durant aquest recorregut, FamilySearch s'ha associat amb més de 10.000 arxius i 200.000 voluntaris de tota mena d'indrets.

Mitjançant l'esforç col·lectiu d'aquestes organitzacions s'ha aconseguit preservar tant índexs com imatges d'arxiu de gran qualitat i posar tots aquests recursos a la disposició de milions de persones de forma gratuïta.

Durant aquest trajecte de més de 100 anys iniciat l'any 1894, l'organització ha tingut l'oportunitat de celebrar grans èxits. Entre ells, destaquen els següents:

- **1938, Microfilm:** La societat genealògica de Utah és pionera en començar a utilitzar microfilm per filmar i emmagatzemar informació relativa a arxius genealògics arreu del món.
- **1942, Family group record archive:** Es crea, de forma manual, una indexació de les genealogies compartides fins aquell moment.
- **1963, Baül d'arxius a les Granite Mountains:** Es completa la creació d'un baül d'arxius amb tecnologia punta situat a les muntanyes pròximes a Salt Lake City, Utah, indret on resideix, avui en dia, la seu de l'organització. Es tracta d'una instal·lació climatitzada que ha estat utilitzada des de la seva creació per preservar còpies de microfilm i arxius digitals de més de cent països diferents.
- **1970, Primers centres d'història familiar:** S'introduceixen els primers centres d'història familiar. Aquests centres formen part d'una ramificació de llibreries que ofereixen accés gratuït a la informació continguda per més de 2,4 milions d'arxius en microfilm. Com hem esmentat en l'apartat anterior, avui en dia existeixen 4.765 d'aquests centres.
- **1985, L'estàndard GEDCOM:** En aquest any FamilySearch introduceix l'estàndard de Comunicació de Dades Genealògiques (GEDCOM). L'estàndard GEDCOM consisteix en un conjunt d'especificacions i regles sobre com s'ha

d'estructurar la informació genealògica de cara a compartir-la amb facilitat a través del núvol.

- **1995, Arbres genealògics digitalitzats:** Es dóna l'oportunitat als genealogistes de digitalitzar els arbres genealògics a la seva disposició i habilitar-ne l'accés a altres usuaris.
- **1998, Digitalització dimatges:** FamilySearch comença a utilitzar tecnologies d'imatge digital per tal de capturar noves fonts de dades i transformar els milions de continguts, emmagatzemats fins ara en microfilm, en imatges digitals. La tecnologia també permet crear índexs de fàcil utilització que relacionen persones amb els continguts digitals.
- **1999, Nova pàgina web:** La pàgina web FamilySearch.org arriba al núvol. En la seva fase inicial, aquesta oferia la possibilitat de cercar informació en els registres històrics de forma relativament simple.
- **2012, Noves tecnologies digitals:** S'incorpora a les tecnologies utilitzades per FamilySearch la tecnologia dCamX, utilitzada per la digitalització de documents i la creació de sales de lectura digital, responsables de facilitar les tasques de comunicació i alliberació de coneixement entre els diferents centres.

Un dels altres grans èxits de l'organització, del que malauroadament no es coneix la data exacte de creació, va ser **l'obertura de la FamilySearch API**.

Aquesta va suposar que aplicacions externes poguessin connectar-se a les bases de dades de FamilySearch i utilitzar-ne la informació d'una forma regulada i eficient. Resulta prou evident que sense l'existència d'aquesta API, aquest projecte mai hagués pogut tenir lloc.

3.3 L'església mormona i la família

Ja que el principal beneficiari de FamilySearch és l'església mormona, creiem que és interessant estudiar de forma breu els orígens d'aquesta.

L'església de Jesucrist dels Sants dels Darrers Dies és una església que considera que la religió hauria de tornar als seus inicis apostòlics. Va ser fundada pel nord-americà Joseph Smith el 6 d'abril del 1830, a l'oest de Nova York.

Considerada actualment com la quarta comunitat cristiana més gran als Estats Units, troba situada la seva seu en l'actualitat a Salt Lake City, Utah. No obstant això, durant els inicis de l'església, Smith tenia la intenció de crear la Nova Jerusalem a prop de Nova York, en una ciutat que anomenaria *Zion*.

L'església, amb origen als voltants de Nova York, es va desplaçar cap a Kirtland, Ohio, des d'on va començar a expandir-se per Jackson County, Missouri, terra en què Smith volia situar la seu del col·lectiu en un futur pròxim.

Joseph va veure contrariats els seus plans, quan l'any 1833, els colons van expulsar brutalment al col·lectiu de Missouri. Com que no disposaven dels recursos

militars necessaris per recuperar el territori per la força, es van veure obligats a anar desplaçant-se al llarg de diferents localitzacions, sempre per culpa de conflictes amb els natius de les terres, fins a establir-se a Nauvoo, Illinois.

Després de la mort de Smith, per tal d'evitar els conflictes armats amb els residents d'Illinois, el col·lectiu es va desplaçar cap a Nebraska i més endavant, durant l'any 1847, a les terres que serien conegudes com a Utah.

Durant aquesta època, l'església es va veure sotmesa a grans pressions i crítiques a causa de la seva tolerància per la poligàmia. Les tensions entre el col·lectiu i el govern d'Estats Units anirien en augment, fins que l'any 1890, el congrés va disgregar l'església i es va apoderar de molts dels seus béns.

Arribats aquest punt, l'església fundada per Smith, va decidir deixar de donar suport als matrimonis plurals, però sense desfer les famílies que ja es troaven unides sota aquestes condicions.

Durant el segle XX, l'església va créixer substancialment i es va veure sotmesa a un procés d'internacionalització, en gran mesura, gràcies a la feina dels missioners enviats a diferents indrets del món.

Durant aquest període el col·lectiu es va convertir en un ferm defensor de les famílies nuclears, és a dir, de les famílies que consisteixen en dos progenitors i la seva descendència. L'església també va oposar-se en aquesta època a l'esmena pels drets igualitaris entre homes i dones, els casaments entre persones del mateix sexe i l'eutanàsia.

Hem volgut redactar aquests paràgrafs previs sobre els orígens de l'església mormònica per tal de poder presentar, amb cert rigor històric, com l'església va veure canviat i evolucionat el concepte de família al llarg del temps.

També queda latent, d'aquesta forma, com la història del col·lectiu es va veure marcada pel rebuig i el desterrament de moltes terres, fins al punt que van haver de recórrer a l'ús de missioners, repartits arreu del món, per tal de sobreviure com a religió.

Així doncs, creiem que per aquest projecte no esdevé necessari entrar en més detall pel que fa a les doctrines i pràctiques de l'església, ni enumerar quines són les principals diferencies entre l'església mormona i les altres corrents del cristianisme. Per altra banda, sí que volem realitzar una reflexió final sobre la posició actual de l'església mormona respecte a la família.

Pels mormons, les famílies representen els lligams que uneixen a les persones en relacions personals i les connecten tant amb les passades com amb les futures generacions. Creuen que cap èxit en la vida, pot compensar el fracàs en l'àmbit familiar.

Segons el seu punt de vista, construir nuclis familiars units i forts és el remei a molts dels fracassos que tenim les persones com a societat i creuen que la família inspira a l'individu a pensar més enllà de l'interès propi o la gratificació immediata i l'anima a entregar-se per altres persones, comunitats i a déu.

Forma part també de la cultura mormona la pràctica o deure d'acumular i preservar, tant les històries dels seus avantpassats com les pròpies, en benefici d'aquells que encara estan per arribar, enllaçant, d'aquesta forma, generacions disconnectades d'una altra forma.

Entenen la naturalesa real de la família, com un algú que transcendeix l'aquí i l'ara i que permet a les persones extreure forces d'aquells que van viure abans que nosaltres.

Concloïen, si ajuntem les dues variables que van marcar l'esdevenir de l'església mormona fins als temps contemporanis, és a dir, la seva semi forçada internacionalització i la importància del nucli familiar en la seva cultura, no hauríem de mostrarnos sorpresos pel fet que el col·lectiu s'hagi convertit en un dels referents mundials en el camp de la genealogia.

3.4 Serveis per organitzacions amb arxius genealògics

Com ja s'ha comentat en seccions anteriors, FamilySearch no és només una organització dedicada a posar a disposició del públic registres genealògics, sinó que també pretenen ajudar a altres organitzacions a digitalitzar i publicar els seus documents al nivell de forma econòmica, ja sigui mitjançant la plataforma FamilySearch o la construcció de noves eines pròpies al nivell.

Sigui com sigui, FamilySearch ofereix cinc serveis a altres organitzacions genealògiques:

3.4.1 Captura d'imatges

Obtenir imatges de qualitat és normalment el procés més costós per aquelles organitzacions que volen digitalitzar els seus registres, tant en el sentit econòmic, com en base als recursos humans necessaris.

El microfilm, que fins fa poc era l'estàndard en la indústria, comença a cedir pas al món digital i tant si l'objectiu de les organitzacions és digitalitzar el seu contingut mitjançant medis propis o utilitzant la tecnologia disponible en els centres de recerca de FamilySearch, aquests ofereixen la seva ajuda a les organitzacions que la sol·licitin.

La tecnologia dCamX, utilitzada per FamilySearch, es caracteritza per la creació d'imatges d'alta qualitat, de forma eficaç, al mateix temps que es capturen mètodes de la imatge. Aquest aspecte, conjuntament amb un procés de publicació posterior fàcil i ràpid, fan que aquesta tecnologia estigui cridada a ser el nou estàndard a la indústria.

3.4.2 Conversió de formats digital

Aquest servei està pensat per aquelles empreses o organitzacions que ja disposen d'una elevada quantitat de material en format de microfilm.

La tecnologia digital ha canviat dràsticament com els registres són capturats, emmagatzemats i fets accessibles. Aquesta tecnologia segueix progressant i per tant resulta indispensable començar a adaptar-se al més aviat possible.

FamilySearch posa a disposició de les organitzacions genealògiques la possibilitat d'utilitzar els mateixos processos i software que utilitzen ells per digitalitzar la seva col·lecció de més de 2,4 milions de microfilms. FamilySearch, també ofereix la possibilitat d'emmagatzemar els fitxers digitals d'aquestes organitzacions en els seus servidors, un cop convertits, si així ho prefereixen.

3.4.3 Indexació en línia

Un cop un registre ha estat digitalitzat en forma d'imatge, la informació principal necessita ser extreta i transcrita per tal de poder produir índexs sobre els quals clients o usuaris puguin cercar.

L'aplicació d'indexació en línia, creada per FamilySearch, permet, mitjançant la cadena de voluntaris, crear índexs de forma ràpida i precisa. Els arxius de les organitzacions, que així ho sol·licitin, podran disposar d'accés a aquesta cadena de voluntaris per digitalitzar els seus índexs o accés a les eines d'indexació auxiliars que permeten la creació de projectes propis.

3.4.4 Accés en línia

Si un document o registre no esdevé fàcilment accessible, resulta de poc valor pels usuaris. FamilySearch ofereix dos serveis diferents dependent de si les organitzacions desitgen fer públic l'accés a les seves dades a través de FamilySearch.org o no.

En cas de voler per públics els registres, FamilySearch s'ofereix a penjar i mantenir els registres de forma econòmica. En cas de voler mantenir els registres en un àmbit privat, l'organització posa a disposició dels interessats les eines i experiència necessàries per crear un espai propi al núvol.

3.4.5 Preservació dels registres i fitxers físics

FamilySearch ofereix l'opció a les organitzacions de custodiar còpies de seguretat dels seus fitxers, en el baül de tecnologia punta situat a las Granite Mountains. En l'actualitat, còpies d'arxius de microfilm i digitals, provenents de més de cent països diferents, es troben guardades en aquest baül per precaució.

El fet de disposar de còpies de seguretat pels fitxers genealògics, suposa la salvació de registres en cas de terratrèmols, incendis, inundacions, tornados, guerres i actes

humans no controlables en les seus oficials dels arxius.

Les mesures de seguretat que FamilySearch utilitza pels seus registres i que a la vegada, queden a disposició d'altres organitzacions, són:

- Processos complets i automatitzats de comprovació, validació i actualització dels registres per garantir la màxima protecció possible.
- Migració gradual i eficient de registres cap a noves tecnologies quan els formats previs quedin obsolets, garantint així la seva accessibilitat a llarg termini.
- Processos de conversió i preservació de dades que compleixen amb les regulacions sobre els Sistemes de Registres d'Informació Oberts (OAIS).
- Col·leccions d'informació emmagatzemada al núvol i distribuïdes en diferents clústers arreu del món per garantir una alta capacitat d'emmagatzematge, escalabilitat i protecció contra els desastres.
- Utilització de les últimes tecnologies en el tractament de dades d'alta densitat.
- Procés ràpid i eficient de cara a processar l'arribada de nous registres al sistema. El sistema actual és capaç de processar més de vint terabytes al dia, mesura que incrementa, al mateix temps que la tecnologia avança.
- Servidors configurats en clústers virtuals per garantir una escalabilitat infinita.
- Facilitat amb control climàtic, prevenció de focs, fonts d'energia auxiliars per casos d'emergència i replicació d'arxius digitals.

3.4.6 Conclusions sobre els serveis professionals

En els apartats anteriors s'ha pogut observar com FamilySearch està clarament interessada a posar les seves tecnologies a disposició d'altres organitzacions genealògiques.

Aquesta estratègia de col·laboració els permet incorporar a les seves bases de dades registres d'informació, d'altre forma inaccessible i garantir la persistència de les dades davant d'esdeveniments no controlables, d'informació gestionada per altres organitzacions.

3.5 Serveis per particulars i aficionats a la genealogia

A part dels serveis per organitzacions, FamilySearch disposa d'un ampli ventall d'eines i funcionalitats pels genealogistes amateurs o aficionats. Aquestes eines es poden dividir en dos grans blocs. El bloc encarregat de gestionar la informació genealògica dels usuaris i el bloc de cerca sobre les bases de dades de FamilySearch.

Les eines de gestió, sobre la informació genealògica, permeten als usuaris digitalitzar la seva informació d'una forma fàcil, estructurada i accessible que compleix amb els estàndards del sector.

Per altra banda, les eines de cerca, constitueixen el principal atractiu de l'organització per particulars i aficionats a la genealogia. Aquest conjunt d'eines han estat pensades per l'exploració de les dades emmagatzemades en l'ampli catàleg de FamilySearch.

Finalment, FamilySearch també permet, a aquelles persones interessades, formar part d'un encadenat de voluntaris lligat a l'organització. Aquest grup de voluntaris s'encarrega d'ajudar, coordinar i arbitrar la indexació de registres genealògics.

3.5.1 Eines genealògiques

El primer bloc d'eines al que feiem referència en la introducció, eren el conjunt d'eines encarregades de la manipulació de la informació genealògica. Les eines principals giren al voltant de la creació d'arbres genealògics, amb opcions per adjuntar fonts d'informació, arxius multimèdia, documents i altres tipus de documents.

Actualment, existeixen moltes eines que satisfan aquesta mateixa necessitat, tant en línia, com aplicacions per escriptori i mòbil; tanmateix, no són moltes les que ofereixen la possibilitat de completar la informació coneguda pels usuaris amb registres i col·leccions penjades al núvol.

Dins d'aquest grup més reduït, destaquen Ancestry.com i FamilySearch.org, on només la darrera és d'accés gratuït. El conjunt de funcionalitats, que destaquen per part de FamilySearch, es detallen breument a continuació.

Arbre familiar

Dins de l'estudi de la genealogia, els arbres genealògics o arbres familiars solen ser l'element més conegut o visual per representar els lligams familiars i l'història familiar d'un individu.

Els arbres familiars a FamilySearch són creats mitjançant les relacions de parentesc entre diferents persones. Aquest fet és important, ja que FamilySearch demana crear aquestes persones per tal d'incorporar-les a les seves bases de dades i fer-les públiques a menys que s'especifiqui que volen ser utilitzades només privadament.

La peculiaritat de FamilySearch és que quan es crea una nova persona, es busca en el sistema, persones que podrien encaixar amb la persona proporcionada. D'aquesta forma, es redueix la creació de persones duplicades i inclús, a vegades, la informació proporcionada és complementada amb la ja existent en l'arbre familiar de FamilySearch.

Un exemple d'arbre familiar de FamilySearch pot ser visualitzat en la figura 5.2.

Fan chart

El Fan chart, o gràfic en forma de ventall, ofereix la possibilitat de visualitzar l'arbre familiar d'una persona, de forma més compacta. Aquest format ofereix una visió de 360 graus només sobre l'ascendència de la persona i de la seva parella.

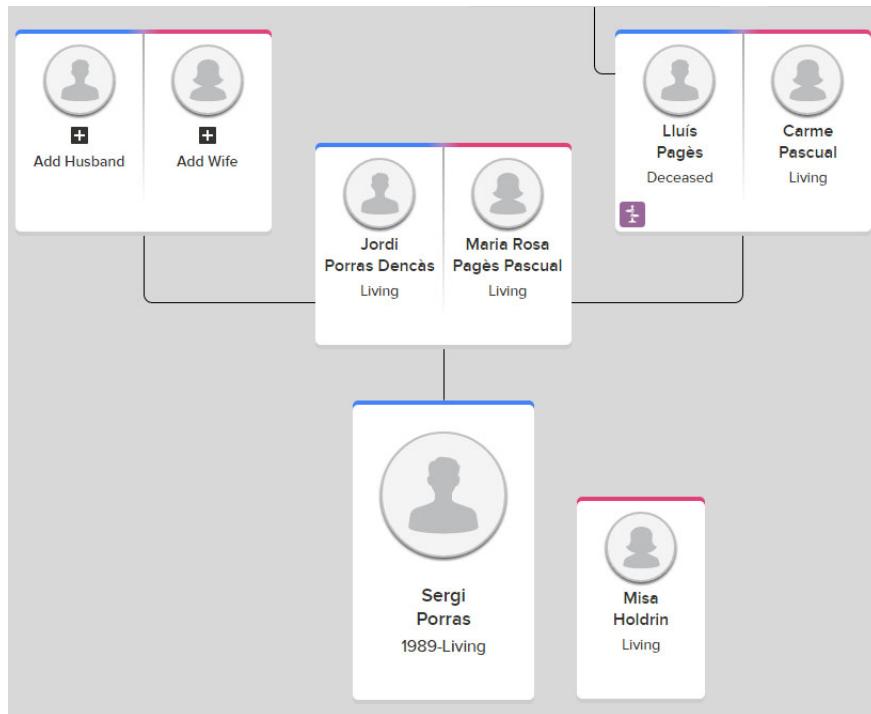


Figura 3.1: Exemple d'arbre familiar.

Oferim un exemple d'aquesta visualització en la figura 3.2.

Detalls personals

Una altra característica de les eines genealògiques és la capacitat de complementar la informació genealògica d'un usuari amb informació personal o informació genealògica més detallada.

El conjunt d'informació extra que pot ser introduïda, s'exposa a continuació:

- **Esbós de la vida d'una persona:** Resum, d'estructura narrativa, que detalla com va viure la persona i quines van ser les diferents etapes de la seva vida.
- **Informació vital:** Gran part d'aquesta informació és la que ja s'ha introduït en el moment de crear la persona a l'arbre genealògic. Aquesta secció permet editar-la i afegir esdeveniments importants en la vida d'una persona, més enllà de la informació bàsica de naixement o defunció.
- **Fonts d'informació:** Permet introduir informació relacionada a les fonts d'informació que contrasten i validen la informació introduïda sobre la persona.
- **Discussions:** Les discussions són conversacions que poden obrir els usuaris per tal de discutir informació relativa a la persona.



Figura 3.2: Exemple de Fan Chart.

- **Notes:** Les notes permeten anotar detalls concrets o informació extra que no té sentit dins de cap altre apartat.

Memòries

Les memòries són en gran part contingut multimèdia que els usuaris poden penjar i relacionar amb les persones de l'arbre familiar per tal d'aportar informació personal extra sobre aquestes.

Si recordem la definició de genealogia, aquesta no tractava només sobre l'estudi dels llinatges familiars, sinó també sobre l'estudi de com van viure aquestes persones. Aquest conjunt d'arxius multimèdia, permeten complementar la informació genealògica coneguda amb informació que ajudi a comprendre qui van ser aquestes persones o que va caracteritzar la seva vida.

FamilySearch suporta quatre tipus d'artefactes diferents. A continuació s'explica en més detall en que consisteix cada una d'elles:

- **Fotografies:** Les fotografies representen la ciència o art encarregat de crear imatges perdurables en el temps de persones, moments o esdeveniments. Aquests artefactes permeten adjuntar imatges d'una persona en el seu arbre familiar.
- **Documents:** Els documents soLEN ser fitxers PDF que proporcionen informació extra sobre aspectes concrets de la persona. Per exemple, un testament.

- **Notes de veu:** Les notes de veu són arxius d'àudio que els usuaris relacionin amb la vida d'una persona en qüestió. Un exemple podria ser, per exemple, un missatge de contestador o gravació auditiva.
- **Històries:** Les històries són peces narratives que els usuaris poden escriure per relatar les vivències d'una persona o documents escrits per la mateixa persona representada a l'arbre familiar.

Family Booklet

El Family Booklet és en un document que FamilySearch crea sota petició que recopila tota la informació disponible relacionada amb la família de la persona que el sol·licita. Aquesta informació s'utilitza per crear un petit llibret que pot ser encarregat en format físic o descarregat digitalment.

3.6 Recerca genealògica

El segon bloc d'eines que FamilySearch posa a disposició dels usuaris i aficionats a la genealogia són les eines necessàries per cercar en el seu immens catàleg de registres, col·leccions, genealogies i llibres genealògics.

Existeixen diferents funcionalitats de cerca segons el tipus d'informació que es vol cercar. A continuació detallarem la informació principal de cada una.

3.6.1 Cerca de registres

La cerca de registres és sens dubte la principal funcionalitat de cerca que queda a disposició dels usuaris. Aquesta permet cercar avantpassats ja difunts en els registres històrics de FamilySearch.

La cerca pot ser delimitada mitjançant diferents paràmetres de cerca i refinada posteriorment mitjançant l'ús de filtres. Els principals paràmetres amb els quals es pot configurar la cerca són:

- Nom i cognoms
- Cercar pel lloc i data aproximada o exacte d'esdeveniments particulars.
- Nom i cognoms dels relatius més propers com poden ser per exemple els pares o la parella.
- Restringir per localització del registre o persona. Pot ser una restricció de continent, país, província, ciutat, etcètera.
- Restringir per tipus de registre. Per exemple, obtenir només registres de naixement, baptisme, casament, servei militar, etcètera.
- Restringir per número de lot.

- Restringir per número de microfilm.
- Nivell d'exactitud desitjat sobre els camps introduïts. Es pot decidir entre un nivell més lax o un que intenti satisfer totes les condicions que han estat introduïdes.
- Restricció per col·lecció. Una col·lecció és una font de dades en concret. Per exemple, el cens de Nova York del 1905.

La figura 3.3 mostre el formulari de cerca bàsic.

Figura 3.3: Exemple del cercador de registres.

3.6.2 Cerca de genealogies

La cerca de genealogies consisteix en la cerca sobre els arbres de família pujats i creats a FamilySearch per altres usuaris o provinents de registres oficials. La fiabilitat de les línies familiars creades per usuaris, varia d'arbre en arbre i els usuaris estan més que convidats a comprovar la veritat de les genealogies resultants de la cerca.

La cerca de genealogies es pot concretar mitjançant els següents paràmetres:

- Nom i cognoms
- Cercar pel lloc i data aproximada o exacte d'esdeveniments particulars.

- Nom i cognoms dels relatius més propers com poden ser els pares o la parella.
- Restringir els arbres genealògics retornats a aquells que han estat creats per usuaris o importats de diferents sistemes i arxius oficials.

3.6.3 Cerca per catàleg

Els catàlegs són materials genealògics com poden ser llibres, materials en línia, pel·lícules de microfilm, microfiche i altres publicacions, com per exemple, revistes. Molts d'aquests recursos poden ser agafats en préstec en els centres d'història de FamilySearch.

3.6.4 Cerca en llibres d'història genealògica

La col·lecció de llibres d'història genealògica consisteix en més de 200.000 publicacions digitalitzades provinents de les més importants llibreries d'història familiar existents. La col·lecció inclou, evidentment, històries de família, revistes genealògiques, guies d'iniciació a la recerca genealògica, diccionaris geogràfics, històries medievals i arbres genealògics.

3.6.5 Wiki de FamilySearch

La wiki de FamilySearch és una petita enciclopèdia que pretén assistir, sobretot als nou vinguts en el món de la recerca genealògica, a comprendre com conduir i enfocar la recerca depenent de les preguntes o informació que estiguin intentant respondre o trobar.

De la wiki cal destacar la informació disponible per cada regió, país o província. Per cada un d'aquests nivells, es disposa d'informació sobre quins poden ser els punts o registres d'entrada més interessants segons la informació que s'estigui cercant.

3.7 Projectes d'indexació

La secció d'indexació posa a disposició dels usuaris tota la informació necessària per convertir-se en voluntaris i començar a transcriure informació continguda en registres genealògics digitalitzats.

El procés d'indexació es realitza a través d'un programa creat per FamilySearch que pot ser descarregat des de la mateixa web i s'encarrega de gestionar els registres que l'usuari pot indexar segons els projectes en els quals aquest es trobi inscrit.

Al mateix temps, FamilySearch està a punt de treure un nou servei que permetrà la indexació en línia des del mateix navegador.

Procés d'indexació

El cicle de vida d'un registre consisteix en les següents fases:

1. Digitalització del registre: En aquesta fase els arxius i registres es digitalitzen.
2. Agrupació dels registres en grups de 20–50 camps diferents dels que cal extreure informació.
3. Dos voluntaris diferents transcriuen els valors dels camps digitalitzats.
4. Si la informació introduïda pels dos voluntaris no coincideix, un àrbitre evalua les dues entrades i pren una decisió.
5. Si les dues extraccions coincideixen, o un cop han estat validades per un àrbitre, les dades s'envien a una base de dades i són preparades per la seva publicació.

Transcriure dades, sobretot aquelles que provenen de documents antics, no és simple i en conseqüència, el programa d'indexació permet introduir com a resposta que part dels camps del registre no poden ser transcrits o que no s'entén el que posa.

De la mateixa forma, FamilySearch proporciona exemples d'escriptura antiga, en diferents idiomes, per ajudar als usuaris a desxifrar els camps més complicats.

Què pot indexar un voluntari?

Els voluntaris poden escollir un o varis dels diferents projectes que actualment es troben en el procés d'indexació i participar en ells. L'usuari pot escollir amb completa llibertat si vol treballar en projectes d'un país o llengua específica i la quantitat de temps que vol dedicar-hi.

3.8 Conclusió sobre les eines per particulars

Tot sembla indicar que FamilySearch està enfocant una part del seu portal web en oferir eines atractives pels usuaris amb l'objectiu de capturar informació genealògica, d'altre forma, difícilment accessible.

Al mateix temps, l'organització està explorant mètodes alternatius i diferents per tal d'aconseguir ampliar la seva col·lecció de registres digitals de la forma més ràpida i eficaç possible. Per aconseguir-ho, està implicant als seus usuaris en el procés d'indexació, principal coll d'ampolla en la digitalització de documents.

Volem recordar, en aquesta conclusió, que molts cops la informació genealògica no passa a estar disponible al domini públic fins molts anys després de la defunció dels seus individus i en molts altres casos, mai arriba a veure la llum.

És per això, que capturar aquesta informació de primera mà, a través dels usuaris, redueix l'espera necessària per aconseguir les dades i redueix les possibilitats d'informació incorrecta sobre aquestes.

Secció 4

Introducció a l'API de FamilySearch

4.1 El portal de desenvolupadors

Tota la informació disponible per tal de poder començar a familiaritzar-se amb l'API de FamilySearch, pot ser trobada en el portal de desenvolupadors.

Aquest apartat de la web està format per diferents seccions, malauradament, l'estructura no acaba de resultar del tot clara per una persona que vulgui iniciar-se per primer cop en l'ús d'aquesta API.

Si ens enfoquem més en la documentació disponible, que no pas en l'estructura proposada per l'organització, podem veure que la informació es podria distribuir, en certa forma, en els següents grups:

- **Requisits tècnics:** Conjunt d'informació necessària per comprendre l'estructura de l'API, els formats de dades que maneja i els passos necessaris per començar a interactuar amb aquesta.
- **Recursos disponibles i rutes d'accés:** Informació detallada sobre cada recurs accessible a través de l'API. En concret, disposa dels detalls de com accedir al recurs, les operacions que es poden realitzar sobre ell, la informació que conté i quines són les connexions amb altres recursos.
- **Evolució i canvis produïts a l'API:** Informació semi ordenada de com l'API s'ha vist evolucionada al llarg del temps i un recull dels canvis produïts sobre els recursos, procés de certificació, material de documentació i eines de desenvolupament.
- **Serveis extres oferts per l'API:** Aquest recull d'articles conceptualitza característiques de l'API com poden ser els recursos d'*emmagatzematge*, *localització* o *throttling*.
- **Eines de desenvolupament:** Recull d'entorns de desenvolupament i eines

extres que poden facilitar la feina del desenvolupador.

- **Certificació:** Recull la informació necessària per gestionar els diferents processos de certificació i informació sobre les regulacions a les quals s'ha de fer front en cas de voler certificar l'aplicació.

4.2 L'arquitectura de l'API

4.2.1 Què és una API?

Abans d'entrar en detall en com funciona una API, estaria bé definir, amb una mica més de precisió, en què consisteix exactament.

Una API, de l'anglès ‘Application Programming Interface’ o Interfície de programació d'aplicacions en català, representa el conjunt de subrutines, funcions i procediments que ofereix una biblioteca per tal de ser utilitzada en el software de tercers com una capa d'abstracció. Aquest conjunt de subrutines, funcions i procediments, acostumen a oferir accés a certs serveis o conjunts de dades d'un particular, a tercers, de forma controlada.

4.2.2 L'arquitectura REST

L'arquitectura sobre la qual està creada l'API de FamilySearch és una arquitectura REST. Les sigles provenen de l'anglès i representen el concepte: Representational State Transfer (REST).

Les arquitectures REST es caracteritzen per estar orientades els recursos més que a les accions que es poden realitzar sobre ells i com a peculiaritat, es caracteritzen per sis regles o restriccions.

Aquestes són: interfície uniforme, sense estat, client-servidor, emmagatzemables, sistema per capes, i codi sota petició. Aquests sis conceptes són doncs els que defineixen les bases de es arquitectures REST.

És diu que una arquitectura REST és orientada als recursos perquè estan construïdes al voltant d'objectes i les relacions entre aquests, en comptes d'accions. Per exemple, a l'API de FamilySearch, es parla de persones i esdeveniments, en comptes de llegir persones o crear esdeveniments i en canvi, aquestes operacions, passen a formar part dels objectes *Persona* i *Esdeveniment*.

L'intercanvi de dades es produeix mitjançant l'ús de diferents representacions. Aquestes, expliquen com els recursos són tractats per l'API i de quina forma han de ser realitzades les comunicacions entre el servidor i el client. Els formats més freqüents són JSON i XML. FamilySearch ofereix suport per ambdós formats.

Per posar un exemple reduït que il·lustri el que estem explicant, podem descriure de la següent forma, el que podria ser una operació contra l'API de FamilySearch,

especificant quin element seria considerat el Recurs, quin el Servei i quin la Representació:

- **Recurs:** Persona (informació relacionada amb una persona en concret)
- **Servei:** Obtenir informació de la persona (GET)
- **Representació:** Nom, cognoms, esdeveniments relacionats amb la vida de la persona, etcètera, en format Llengutge de Marcatge Extensible (XML) o Notació d'Objectes Javascript (JSON).

Com també s'ha comentat, l'arquitectura REST es caracteritza per la implementació de sis restriccions imposades sobre el sistema. A continuació s'exposa amb més detall, cada una d'elles.

Interfície uniforme (uniform interface)

Aquesta restricció s'encarrega de definir la interfície de comunicació entre el client i el servidor.

En una arquitectura REST, s'utilitzen els protocols de comunicació HTTP i HTTPS de forma conjunta amb els Identificadors de Recurs Uniforme (URI), per aconseguir accés als diferents recursos i operacions proporcionades per l'API.

Els verbs permesos pels protocols de comunicació web són els coneguts: get, put, post, delete, options and head.

Per exemple, per fer la petició de lectura sobre el recurs d'una Persona a l'API de FamilySearch, executaríem la següent crida HTTP o HTTPS mitjançant el verb i URI especificats a continuació:

GET /platform/genealogies/persons/2:2:PPPJ-MYZ7.

Sense estat (stateless)

Aquesta restricció implica que el servidor no emmagatzema la informació del client. Això implica que cada petició d'aquest cap a l'API, ha de contenir tota la informació necessària perquè el servidor l'identifiqui i es defineixin les regles de comunicació adequades per processar la petició.

Hi ha exemples d'operacions a l'API de FamilySearch, com per exemple el procés d'identificació Oauth V2, que no són realment RESTful, doncs aquestes sí que guarden informació del client durant les diferents parts de la comunicació.

Client-servidor (client-server)

Per comprendre aquesta restricció, cal comprendre primer en què consisteix un sistema disconnectat.

En el cas de les arquitectures REST, un sistema disconnectat implica que el client mai tindrà accés directe a les bases de dades que emmagatzemen la informació, i que per tant, sempre haurà d'accendir a les dades mitjançant l'intermediari. En aquest cas, l'API.

Els protocols de comunicació descrits prèviament (HTTP i HTTPS), i la interfície de comunicació, són els encarregats de gestionar les comunicacions entre client i servidor.

Emmagatzematge en el client (cacheable)

Aquesta restricció fa referència a si les respostes retornades, des del servidor, al client, poden ser emmagatzemades per aquest i durant quant de temps les pot guardar. Existeixen tres nivells de configuració diferents:

- **Implícit:** Si és el client el que decideix quant de temps guardarà les dades o informació retornada, es tracte d'emmagatzematge implícit.
- **Explícit:** Si és el servidor el que mana i posa les regles, parlem d'emmagatzematge explícit.
- **Negociat:** Quan el client i el servidor negocien i arriben a un acord, es tractea d'emmagatzematge negociat.

Sistema per capes (layered system)

Aquest principi, o restricció, es basa en el fet que el client no pot assumir que tindrà connexió directa amb el servidor. És a dir, poden existir diferents intermediaris en forma de hardware i software entre client i servidor.

Això, facilita l'escalabilitat i persistència del sistema gràcies al fet que el client no s'ha de preocupar de comunicar-se amb elements o tecnologies específiques. D'aquesta forma, el servidor es pot veure subjectes a canvis de forma transparent pels clients.

Codi sota petició (code on demand)

Restricció que regula com, de forma excepcional, el servidor pot proporcionar accés al client sobre certes parts de la lògica del funcionament. Alguns exemples poden ser els *Java Applets* o blocs de codi *JavaScript*.

4.3 Formats de dades utilitzats pel Sistema

FamilySearch utilitza tres formats de dades diferents per representar la informació emmagatzemada en les seves bases de dades i dos formats extres per codificar aquesta informació i enviar-la a través del núvol.

Els conjunts de dades utilitzats per representar els recursos són els que segueixen:

- Les dades genealògiques es representen mitjançant el format GEDCOM X.
- Els recursos o objectes específics del model de FamilySearch, es representen mitjançant una extensió del model de dades GEDCOM X.
- El format de dades Atom, o atòmic, s'utilitza per proporcionar un format simple per les meta-dades.

4.3.1 El format de dades GEDCOM i GEDCOM X

El terme GEDCOM, és un acrònim de l'anglès Genealogical Data Communications.

El format GEDCOM consisteix en un conjunt de regles d'aplicació per tal de representar informació genealògica. Aquest format de dades, creat per FamilySearch l'any 1984, s'ha convertit en l'estàndard de la indústria.

Per simplificar-ho, podríem entedre un fitxer en format GEDCOM, com un fitxer de text que emmagatzema informació genealògica d'una persona i les metadades necessàries per poder enllaçar als diferents fitxers de la mateixa persona.

Tot i que l'última versió, datada del 1996, segueix sent molt utilitzada, FamilySearch va proposar durant l'any 2012, canviar aquest estàndard per la seva nova versió anomenada GEDCOM X.

El format de dades per la Comunicació de Dades Genealògiques Extesa (GEDCOM X), representava un nou projecte de codi obert i es diferenciava del seu antecessor en la implementació d'un sistema que facilitava la inclusió d'arbres genealògics i fonts de dades als recursos ja existents.

Al mateix temps, el nou estàndard també donava suport a l'intercanvi i enllaçament de dades a través del núvol i es creava així la primera versió de l'API de FamilySearch.

En la taula 4.1, s'ofereix un petit exemple de com la part bàsica del recurs persona és codificat sota el format de dades GEDCOM X.

Taula 4.1: Codificació GEDCOM X del recurs Persona

Nom	Descripció	Format de dades	Restriccions
Private	Indica si la instància de la persona ha estat designada com a privada o pública.	Booleà	OPCIONAL. Una descripció de com les aplicacions han de tractar les dades de caràcter privat.
Gender	El gènere de la persona	Gènere (GEDCOMX)	OPCIONAL.

Names	Els noms de la persona	Llista de Noms (GEDCOMX)	OPCIONAL. Si més d'un nom és introduït s'assumeix que aquests han estat introduïts en ordre de preferència amb el més preferit introduït primer.
Facts	Esdeveniments relacionats amb la vida d'una persona.	Llista d'Esdeveniments (GEDCOMX)	OPCIONAL.

Així doncs, podem veure com la instància del recurs Persona conté un camp booleà, que indica si aquesta pot ser utilitzada de forma pública o només en l'àmbit privat i tres camps que es troben codificats sota els estàndards del format GEDCOMX.

Per exemple, el format de dades <http://gedcomx.org/v1/Gender>, representaria un recurs amb l'estructura que s'exposa a la taula 4.2 i els valors possibles per l'enumeració de gènere, s'indiquen en la taula 4.3.

Taula 4.2: Codificació GEDCOM X del recurs Gènere

Nom	Descripció	Format de dades	Restriccions
type	Valor que indica el gènere de la persona.	Enumeració	REQUERIT. El gènere ha de ser especificat i es recomana utilitzar un valor dels acceptats per l'enumeració és recomanat.

Taula 4.3: Valors enumeració type

Gènere URI (Gedcom X)	Descripció
http://gedcomx.org/Male	Gènere Masculí
http://gedcomx.org/Female	Gènere Femení
http://gedcomx.org/Unknown	Gènere no especificat

Com que l'objectiu del projecte no és estudiar la codificació GEDCOM o GEDCOM X, sinó comprendre quina informació es troba realment disponible a través de

l'API de FamilySearch, no entrarem més en detall en aquests formats.

L'objectiu d'aquest apartat era explicar quin és l'estàndard de representació de dades genealògiques utilitzat per l'API. Per qualsevol informació extra que es vulgui consultar, s'adjunta a la bibliografia del projecte, l'enllaç a la documentació del model conceptual.

4.3.2 Format de dades FamilySearch

El format de dades FamilySearch, defineix el format d'aquells objectes específics relacionats amb la plataforma de dades pròpia de l'organització. Això implica que aquestes estructures no formen part de cap estàndard i manquen de sentit fora del context pel qual han estat definides.

L'estructura dels objectes específics de FamilySearch ha estat creada com una extensió de l'especificació GEDCOM X. Per tant, segueixen una estructura molt similar.

4.3.3 Format de dades Atom (o Atòmic)

Els formats de dades Atom, o atòmic, és utilitzat per proporcionar un format pel contingut web i les metadades.

Aquest format és utilitzat, entre altres llocs, en les col·leccions ordenades de resultats, com podrien, per exemple, les respostes a la funció de cerca de persones o l'obtenció del historial de canvis d'una persona.

4.3.4 Codificacions dels formats de dades

Els formats de dades que s'han exposat en els apartats anteriors no són més que unes convencions que marquen l'estructura a seguir per representar els diferents objectes o recursos utilitzats per l'API de FamilySearch.

Tanmateix, aquestes estructures han de ser codificades per tal de poder ser transmeses a través del núvol i en concret, FamilySearch, proporciona suport a les dues codificacions més comunes i utilitzades per aquesta finalitat. Els llenguatges XML i JSON.

El llenguatge XML

El XML, és un llenguatge de marcatge que defineix un conjunt de regles a seguir per tal de codificar, documents i informació, en un format lleigible i processable, tant per éssers humans, com màquines.

El llenguatge va ser definit pel Consorci World Wide Web i tracta d'emfatitzar la simplicitat, generalitat i usabilitat del model, per l'ús a través d'Internet.

Una versió reduïda de la representació en XML del recurs ‘Nota’, amb camps: subjecte, text i atribució, quedaría representat de la forma següent:

Codi 4.1: Representació bàsica en XML d’una Nota

```
<Note xmlns='...'>
    <subject>...</subject>
    <text>...</text>
    <attribution id='...'>
        <contributor resourceId='...' resource='...'/>
        <modified>...</modified>
        <changeMessage>...</changeMessage>
        <creator resourceId='...' resource='...'/>
        <created>...</created>
    </attribution>
</Note>
```

Com es pot observar, cada camp, objecte o peça d’informació, es troba envoltada per dues etiquetes que en marquen l’inici i final. El text situat a l’interior d’aquestes etiquetes, indica de quin camp es tracta. Per exemple, pel camp subjecte, tenim les etiquetes *<subject>* i *</subject>*.

El llenguatge JSON

El llenguatge JSON, és un estàndard de format obert que, de la mateixa forma que el llenguatge XML, pretén crear codificacions llegibles tant per éssers humans com màquines i al mateix temps, poder transmetre aquestes dades a través del núvol de forma ordenada.

Aquest format es basa en el concepte ‘clau -valor’. És a dir, cada camp d’un objecte a representar està format per una clau i un valor associat a aquesta clau.

El llenguatge JSON deriva del JavaScript. Al principi, només aquesta plataforma incorporava funcions per codificar i descodificar aquest llenguatge de marcatge.

Durant els últims anys, el format JSON s’ha vist convertit en l’estàndard de la indústria per l’intercanvi de dades a través d’Internet i en conseqüència, ha provo- cat que molts altres llenguatges de programació hagin incorporat les seves pròpies funcions de codificació i descodificació.

Una versió reduïda de la representació en JSON del recurs Nota, amb camps: subjecte, text i atribució, es representaria de la forma següent:

Codi 4.2: Representació bàsica en JSON d’una Nota

```
{
    'lang': '...',
    'subject': '...',
    'text': '...',
    'attribution': {
        'contributor': { },
        'modified': '...',
```

```

    'changeMessage': '...',
    'creator': { },
    'created': '...',
    'id': '...'
},
{
  'id': '...'
}

```

4.4 Evolució temporal de l'API

4.4.1 Evolucions al llarg del temps

L'api de FamilySearch s'ha vist subjecte a diversos canvis i modificacions al llarg del temps, algunes, amb més repercussió i implicacions que altres. La llista de canvis és prou llarga com per no intentar reproduir-la en la memòria. De totes maneres, aquesta és adjuntada a la bibliografia del projecte.

L'últim canvi introduït a l'API i de fet, probablement, dels més importants que s'han produït fins ara, fa referència a la nova estructura del back-end anomenada Tree Foundation.

A falta de documentació oficial al respecte, es va consultar un dels Webinair passats, oberts al públic per part de FamilySearch, per tal de comprendre millor la magnitud d'aquests canvis i quines implicacions tenia. En el següent apartat, tractarem més a fons en què consisteix Tree Foundation.

Les primeres fases d'aquest darrer canvi van arribar a producció a mitjans del mes de juliol, moment en què la implementació dels exemples pel projecte a través del SDK oficial de JavaScript estava complerta. Aquests canvis, van repercutir en gran mesura sobre el SDK oficial, trencant-ne moltes de les funcionalitats.

Al final, els principals problemes que va causar pel projecte van ser interrupcions permanents en l'entorn de proves Sandbox i la inhabilitat d'accedir als recursos Font de Dades i Discussió relacionats amb una persona, a causa de la no adaptació del SDK oficial.

4.4.2 Tree Foundation

Com s'ha comentat en l'apartat anterior, Tree Foundation és la nova versió del back-end que FamilySearch està implementant. A continuació, es detallen en més profunditat les implicacions d'aquest canvi, ja que pot tenir un gran impacte en com les principals dades emmagatzemades per l'API hauran de ser accedites en un futur no molt llunyà.

El projecte de Tree Foundation consisteix a canviar per complet el motor que fa funcionar el back-end de FamilySearch.

El motor actual, anomenat de forma interna, Conclusion Tree i que fins ara, era

l'estructura amb la que l'API es comunicava, caurà en desús en benefici de la nova estructura implementada en el projecte Tree Foundation.

El canvi d'arquitectura té com a raó de ser poder augmentar l'escalabilitat del sistema i fer front a l'elevada demanda de dades sota la que l'API es veu sotmesa. En la imatge 4.1 es pot veure com es passa d'un sistema unificat, a un sistema distribuït.

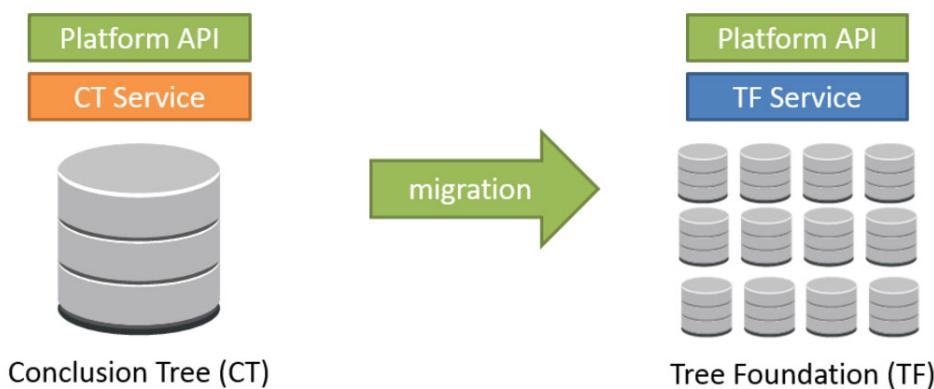


Figura 4.1: Arquitectura *Conclusion Tree* vs *Tree Foundation*

Els beneficis esperats del canvi són un increment en l'escalabilitat del sistema, una millora eventual del rendiment, tot i que aquest no és l'objectiu principal durant les primeres fases d'implementació i finalment, una millor correlació entre el funcionament del sistema i com l'API accedeix a les dades.

Addicionalment, el projecte canviarà part de la informació inclosa en els recursos accessibles a través de l'API.

Un dels canvis que l'evolució de Tree Foundation pretén integrar i que avui en dia, encara no es troba a producció, és integrar dins del recurs d'una persona tota la informació que s'acostuma a consultar i que ara no contenia. Per exemple, les relacions familiars o les fonts de dades que verifiquen que la informació introduïda és correcta.

Aquest canvi, pretén facilitar la navegació per les dades, evitant haver d'accedir a múltiples recursos per tal d'aconseguir tota la informació desitjada. D'aquesta forma, s'aconseguirà alliberar part la càrrega que suporta el sistema i millorar-ne, per tant, l'eficiència i disponibilitat. La figura 4.2 representa de forma visual el que aquest canvi implica en el recurs Persona.

Quan Tree Foundation va arribar a producció, només els recursos Source i Discussion es trobaven incorporats dins del recurs Persona i els antics enllaços, a aquesta informació quedaven obsolets. Més endavant, apuntant a mitjans d'Agost, s'espera que es realitzi el mateix canvi per les relacions familiars. La figura 4.3 mostra els canvis als quals ens estem referint.

Tot i que el canvi és molt prometedor, cal tenir en compte que té el potencial de

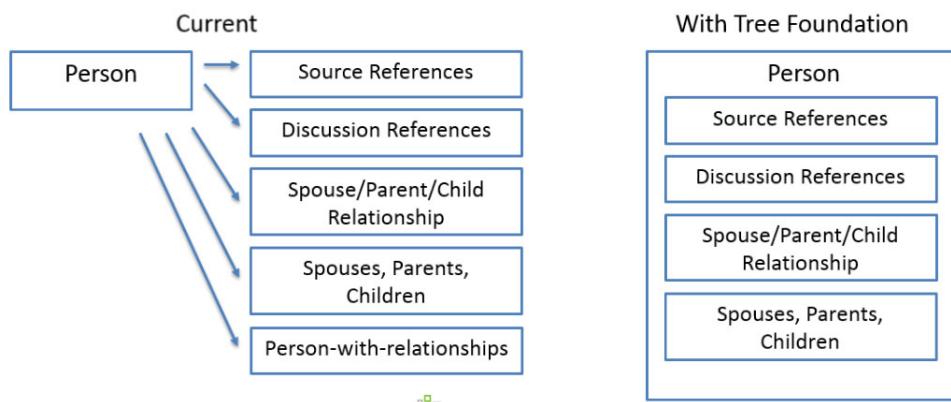


Figura 4.2: Relocalització d'informació sobre alguns recursos.

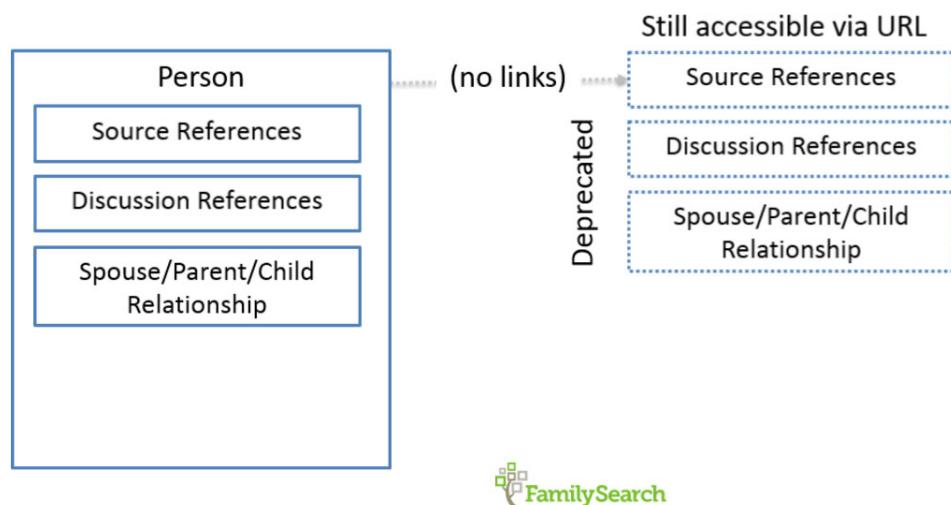


Figura 4.3: Localització dels recursos Discussió i Fonts de dades en la fase 1.

trencar part dels SDKs oficials i moltes de les aplicacions que tractin amb la versió antiga de l'API, de la mateixa forma, que ja ho han fet les primeres versions de Tree Foundation arribades a producció.

Per tant, caldrà estar al compte de quan aquests canvis arriben a producció i de quan els SDKs oficials estaran preparats, per ser utilitzats sense error, davant d'aquesta nova versió del back-end de FamilySearch.

4.5 Serveis extres oferts per l'API

L'API de FamilySearch no es caracteritza només per ser el centre d'informació obert amb més registres genealògics, sinó que a més a més, l'API ha estat dissenyada i pensada perquè pugui ser utilitzada de forma simultània per moltes persones, des

de diferents indrets del món, d'una forma còmode i personalitzada.

Tot i que evidentment cap sistema és infal·lible, i de fet, com ja hem comentat, FamilySearch està realitzat una actualització del seu sistema del back-end per tal de garantir una millor escalabilitat del sistema, el sistema actual ja presenta una sèrie de serveis dignes de menció.

4.5.1 Caching

El fet de voler oferir un servei de *caching* decent ha influenciat bona part del disseny de l'API. Davant la pregunta evident de, perquè és important el *caching* a FamilySearch?

- El fet que el client pugui aplicar tècniques d'emmagatzematge, evita la repetició de certes peticions al sistema.
- L'existència de reverse proxies permet, a aquests servidors, respondre a peticions dels clients sense necessitat d'accendir als servidors i bases de dades si detecten que la informació demana, ja es troba disponible en aquests. Per tant, ajuden a reduir la càrrega del sistema.
- Davant noves peticions, els servidors poden indicar als proxies que les seves dades encara són vàlides i que per tant, no fa falta que realitzin una nova petició i n'esperin la resposta.

4.5.2 Throttling

La funcionalitat de *throttling* s'encarrega de posar un límit al nombre de peticions, en un cert interval de temps, que un usuari pot realitzar. Aquests límits són creats en l'àmbit d'usuari i per tant, utilitzar més d'una sessió simultània, no permet saltar-se aquesta limitació.

Les polítiques de *throttling* permeses són calculades mitjançant el temps de processat, respecte al temps real que ha transcorregut. Diferents recursos, disposen de diferents polítiques. En cas d'excedir aquests límits, la capçalera de resposta indica quant de temps cal esperar entre peticions, per poder tornar a interactuar amb l'API.

Aquest servei, que evidentment, suposa una limitació pels usuaris, evita que certs usuaris de forma intencionada o involuntària, bloquegin tots els recursos disponibles per part dels servidors i el servei quedí inutilitzable per altres persones.

4.5.3 Sincronització

Un dels reptes més comuns, per aquelles aplicacions que emmagatzemen dades d'un tercer, és el de saber si aquestes dades han estat modificades des de l'últim cop que van ser consultades o extretes.

El paràmetre ETag, de la capçalera de les peticions contra l'API, ha estat modificat per part de FamilySearch, per obtenir la versió del recurs consultat. Per tant, si es realitza una petició de només aquestes capçaleres, a canvi d'un petit esforç de l'API, es podria comparar si els recursos desitjats han estat modificats o no sense causar un gran impacte.

Un cop es coneix que un recurs ha estat modificat, es poden descobrir els canvis realitzats mitjançant la comparació dels dos objectes o llegint l'historial de canvis. Evidentment, sempre es podria realitzar una sobre escriptura completa del recurs.

4.5.4 Internacionaltizació

Un dels serveis que més m'ha sorprès trobar-me a l'API de FamilySearch és el de suport a la internacionalització.

La internacionalització consisteix a adaptar una peça de software o contingut, a diferents idiomes, sense necessitat de realitzar canvis en el codi. L'API de FamilySearch permet realitzar les peticions de forma que certs conjunts de dades retornades, com poden ser per exemple, el nom de països o persones, estiguin localitzades en l'idioma especificat.

La internacionalització permet, per exemple, que un usuari que es troba a Espanya vegi el terme 'living' (viu), en el seu idioma natiu. Un altre recurs que moltes pàgines web localitzen són les dates. És ben sabut que l'estrucció americana d'una data, per exemple, difereix d'una data europea en la posició dels termes any, mes i dia.

El conjunt de dades que FamilySearch permet localitzar són:

- Les propietats de visualització del recurs Persona.
- La informació relativa a les localitzacions.
- Certs aspectes del vocabulari controlat, o comú, de l'API.
- Dates.

4.5.5 Transferència de registres en grans quantitats

Existeix la possibilitat per les organitzacions que així ho desitgin, de signar un acord amb FamilySearch, que els hi permeti mantenir la seva pròpia còpia de dades oficials de FamilySearch, relativa a persones difuntes.

Per realitzar aquestes transferències, que mouen alts volums de dades, FamilySearch disposa d'una API especial que permet a les organitzacions gestionar, mantenir i actualitzar, les dades emmagatzemades en els seus sistemes a través de diferents sessions.

Aquesta opció pot permetre a altres organitzacions ordenar i emmagatzemar les dades amb una estructura diferent i habilitar, d'aquesta forma, la possibilitat de

realitzar diferents tipus d'estudi. Per exemple, estudis de mineria de dades.

4.6 Eines de desenvolupament

FamilySearch posa a disposició dels desenvolupadors un seguit d'eines destinades a facilitar les seves implementacions contra l'API.

Són dos els recursos principals que permeten apropar la utilització de l'API a qualsevol desenvolupador. Els SDKs i les aplicacions d'exemple.

4.6.1 Els SDK de FamilySearch

Els Kit de Desenvolupament Software (SDK), són un conjunt d'eines que faciliten la creació d'aplicacions per un software, hardware, framework, sistema de computació, videojoc, sistema operatiu o plataforma de desenvolupament determinada.

En el cas de FamilySearch, es tracta d'eines de desenvolupament destinades a un llenguatge de programació específic.

Però, perquè esdevé útil la utilització d'un SDK? En el cas de FamilySearch un SDK ens ofereix la possibilitat de comunicar-nos amb l'API sense haver de preocuparnos, de forma directa, en com les URIs han de ser codificades o en parcejar els JSON/XML de les respostes. Taques que poden esdevenir tedioses i repetitives.

Per tant, els SDK permeten als desenvolupadors centrar-se en el desenvolupament de funcionalitats i no en la gestió de comunicacions amb l'API.

FamilySearch disposa en l'actualitat de sis SDKs diferents. Tots ells faciliten la interacció amb l'API, encara que no tots permeten les mateixes funcionalitats. A més a més, alguns d'aquests SDK són oficials, significant que són desenvolupats i mantinguts per l'organització FamilySearch, mentre que altres, són creats i gestionats per la comunitat.

Els diferents SDK disponibles es llisten a continuació:

- **Java SDK:** El Java SDK és un SDK oficial que serveix per crear aplicacions d'escriptori amb el llenguatge Java.
- **PHP SDK:** El SDK de PHP és un SDK oficial per crear aplicacions web. Desafortunadament, no es troba del tot actualitzat i l'última versió d'aquest no integra masses de les funcionalitats que es poden realitzar contra l'API.
- **C Sharp SDK:** El SDK de C Sharp és un altre dels SDK oficials i serveix per crear aplicacions amb la tecnologia .NET.
- **Javascript SDK:** El SDK de Javascript és l'últim SDK oficial i està orientat sobretot a la creació d'aplicacions web. És el SDK que s'ha utilitzat en aquest projecte ja que cobreix gran part de les funcionalitats de l'API i en facilita l'ús.

- **Python SDK:** El SDK de Python resultaria molt atractiu de cara a tota mena d'aplicacions, però per desgràcia, encara es troba en fase de desenvolupament i sense dates concretes de finalització.
- **Ruby SDK:** La perla de Ruby és l'últim SDK disponible per FamilySearch i es tracta d'un SDK no oficial.

4.6.2 Les aplicacions d'exemple

Les aplicacions d'exemple consisteixen en aplicacions de codi obert que permeten entendre, de forma més pràctica, com funcionen els SDK.

Solen cobrir un conjunt d'operacions bàsiques, com poden ser, per exemple, llegir l'usuari connectat, realitzar una cerca bàsica o la lectura d'una persona concreta de l'arbre familiar. En definitiva, representen un bon punt de partida per aquells que vulguin familiaritzar-se amb un SDK concret i comprendre'n les bases que els permeten l'elaboració de funcionalitats més complexes en el futur.

De la mateixa forma que els SDK, FamilySearch disposa de sis aplicacions d'exemple, aquest cop, sent només oficials les aplicacions que utilitzen els SDK de Javascript i Java.

4.6.3 Altres eines interessants

L'últim recurs per desenvolupadors que FamilySearch ofereix, és un llistat d'eines que poden ser interessants o esdevenir útils, de cara a la realització de proves amb l'API.

Entre aquestes, destaquen diferents intermediaris que serveixen per simular peticions HTTP i HTTPS contra l'API de FamilySearch i una utilitat, no oficial, que en teoria permet copiar dades de producció als entorns de desenvolupament.

4.7 Procés de certificació

Per tal de poder connectar les aplicacions desenvolupades per tercers al conjunt de dades oficial de FamilySearch, aquestes aplicacions han de ser sotmeses a un procés de certificació.

Les aplicacions poden ser certificades per l'ús comercial o per ús limitat. Les Apps que volen ser certificades només per ús limitat, requereixen un anàlisis tècnic sobre el seu funcionament, mentre que de les aplicacions d'ús comercial també són analitzades des d'un punt de vista de negoci i marketing. Com a contrapartida, les aplicacions d'ús limitat no poden aparèixer a la galeria d'aplicacions.

Hi ha tres tipus de certificacions principals:

- **Certificació de lectura:** L'aplicació ha de ser certificada per la lectura de

tots aquells recursos als quals accedeix. També cal que compleixi amb certs estàndards del procés d'identificació.

- **Certificació d'escriptura:** En cas que l'aplicació realitzi operacions d'escriptura, aquestes també hauran de ser revisades per part de FamilySearch.
- **Certificació per transferència d'arxius en grans quantitats:** Per aquestes organitzacions que tinguin permís per accedir als protocols de transferència de dades en grans quantitats, cal certificar i revisar, conjuntament amb FamilySearch, les operacions realitzades contra aqueta API.

Es veuran més detalls sobre el procés de certificació en la part pràctica de la memòria del projecte.

Un cop una aplicació ha estat certificada, en cas que aquesta modifiqui les operacions de lectura o escriptura que realitza, caldrà tornar a certificar-la abans de desplegar els canvis a producció.

Existeix un procés encarregat de controlar que es compleixen els estàndards marcats per FamilySearch i en cas de no complir-los, pot significant el retirament dels drets d'accés a producció.

Per acabar aquesta secció, comentar que mantenir una relació formar amb FamilySearch, proporciona certs beneficis com aparèixer en les seves galeries d'aplicacions i poder utilitzar el logotip d'aplicació certificada, entre altres petits avantatges.

4.8 Diferents entorns de treball

Es disposa de molt poca informació sobre els diferents entorns als quals es pot accedir relativament a l'API de FamilySearch. A força d'investigar diferents codis i possibilitats, s'ha arribat a la conclusió de què existeixen els següents entorns de treball:

- **Sandbox:** Entorn de treball únic per cada usuari. Utilitzable durant el desenvolupament de noves aplicacions i que replica les funcionalitats de l'API sense tenir accés a les dades de producció. Es pot entendre com un entorn de proves.
- **Staging:** Entorn de treball que s'utilitza per validar que una aplicació, que s'ha desenvolupat en un sandbox, està realment preparada per accedir a les dades de producció.
- **Beta:** Entorn de treball sobre el qual l'API va desplegant noves versions. Quan es vol realitzar un canvi en les funcionalitats o estructura de l'API, s'utilitza aquest entorn perquè els desenvolupadors puguin testejar el funcionament de les seves aplicacions abans de desplegar la nova versió a producció.
- **Producció:** Entorn de treball que pot accedir a les dades oficials de FamilySearch. Totes les aplicacions aspiren a connectar-se a aquest entorn de treball.

Secció 5

Estudi en profunditat de l'API de FamilySearch

5.1 Els recursos de FamilySearch

L'API de FamilySearch organitza tota la informació disponible en objectes que són anomenats recursos.

Cada recurs emmagatzema informació relativa al concepte que representen. Per exemple, el recurs Persona conté informació sobre si aquesta està viva o morta, el seu nom i tota mena d'informació i esdeveniments relacionats amb la seva vida.

En la taula 4.1 de la secció anterior de la memòria, observarem ja com a exemple alguns dels paràmetres continguts pel recurs Persona de l'API de FamilySearch.

El recursos també es caracteritzen per emmagatzemar informació relativa a quines operacions es poden realitzar sobre ells. Per exemple, el recurs Persona conté informació sobre com accedir a les operacions llegir una persona, editar una persona, esborrar una persona, buscar possibles duplicats de la persona, etcètera.

Finalment, els recursos també destaquen per contenir els enllaços o més ben dit, les URIs, als recursos relacionats. Parlarem més en detall d'aquestes URI en el següent apartat de la memòria.

5.2 Enllaços hypermedia, la navegació entre recursos

Com hem comentat en l'apartat anterior, els recursos emmagatzemen informació sobre les URI dels recursos relacionats. Aquestes URI són anomenades enllaços hypermedia i permeten la navegació entre els diferents recursos del model de dades.

Per posar un exemple, imaginem el recurs Persona. Aquest conté enllaços hypermedia que apunten als recursos que conformen el concepte Parella, Pares, Fills, Fonts d'informació, Ascendència, Descendència i Memòries. Cada un d'aquests recursos

apuntats conté informació específica i d'interès respecte a la persona consultada.

La figura 5.1 reflecteix com el recurs Persona és accessible a través de l'Arbre Familiar de FamilySearch i com les diferents relacions i recursos, associats a una persona, són connectats a través dels enllaços hypermedia d'aquest.

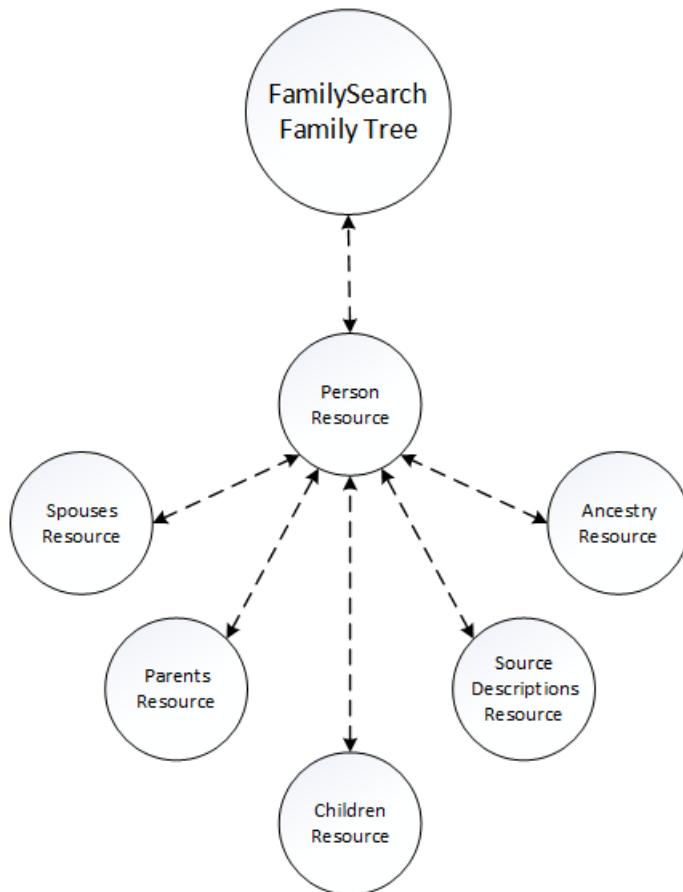


Figura 5.1: Enllaços hypermedia del recurs Persona.

Volemaprofitar aquest apartat de la memòria per recordar, que en el moment en què la nova versió del back-end, Family Tree, es trobi desplegada per complet a producció, el recurs Persona no contindrà més els enllaços hypermedia a tots aquests recursos, sinó que les relacions de parella o paternals, es trobaran incloses dins del mateix recurs Persona.

De totes maneres, el concepte dels enllaços hypermedia seguirà sent vàlid de cara a les relacions del recurs Persona amb la resta de recursos vinculats i per tota la resta de relacions entre els diferents recursos de l'API de FamilySearch.

Els enllaços hypermedia cobren especial interès a l'hora de crear aplicacions robustes que es vegin el menys afectades possible per canvis en la localització o crida dels recursos.

El primer gran avantatge d'utilitzar aquests enllaços és que no cal implementar

en el codi, de forma específica, les URI d'accés a cada recurs. D'aquesta forma, s'aconsegueix evitar que en cas de canvis en les URI, sigui necessari realitzar modificacions en el codi de les nostres aplicacions.

El segon, és que si es coneix un sol punt d'entrada al sistema, les aplicacions ja no requeriran més informació per tal de navegar entre els diferents recursos de la resposta. D'aquesta forma, podríem descriure els enllaços hypermedia com una espècie d'índexs que permeten explorar i navegar a través del conjunt de recursos que conformen les respostes de l'API de FamilySearch.

5.3 L'arbre genealògic de FamilySearch

El model de dades que conforma l'arbre genealògic de FamilySearch, consta de molts recursos i enumeracions diferents. No citem en la memòria el nombre total de recursos diferents, ja que alguns dels objectes documentats de forma oficial es troben en desús, mentre que alguns objectes nous, encara no han rebut la documentació pertinent.

El conjunt d'objectes o recursos connectats, conformen el que ha estat enomenat l'arbre familiar de FamilySearch (Family Tree). Aquest arbre, pot ser subdividit en cinc grans blocs:

- **El bloc de persones:** Aquest bloc representa al conjunt de recursos que emmagatzemen la informació personal de les diferents persones representades a l'arbre familiar.
- **El bloc de les relacions familiars:** Aquest bloc està format per aquells recursos que contenen informació sobre les diferents relacions familiars entre les persones emmagatzemades en el sistema.
- **El bloc de col·leccions:** Aquest bloc recull els recursos que guarden la informació relativa a les fonts de dades i documents digitalitzats, que certifiquen la veritat de les dades.
- **El bloc de discussions:** El bloc de discussions conté aquells recursos que contenen la informació relativa a les converses o discussions creades, per part dels usuaris, al voltant de les persones de l'arbre familiar.
- **El bloc de memòries:** Aquest últim bloc està format per aquells recursos que emmagatzemen la informació relativa a les memòries descrites en la secció tres d'aquesta memòria.

La imatge 5.2 mostra aquests cinc grans blocs que conformen l'arbre familiar de FamilySearch i com es troben relacionats entre ells.

Els blocs principals de l'arbre familiar són, sense cap mena de dubte, els que contenen la informació relativa a les persones i a les relacions familiars que les lliguen.

L'objectiu de la resta de blocs que conformen l'arbre familiar és el de proporcionar suport i informació extra més detallada sobre les persones, relacions familiars,

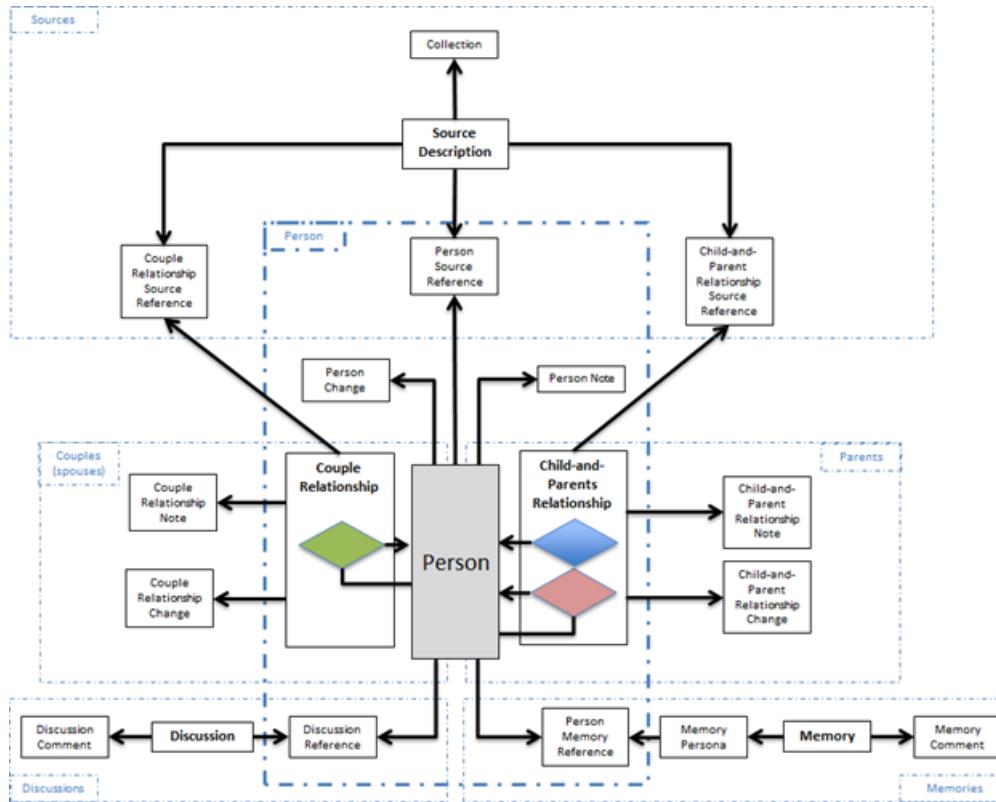


Figura 5.2: Estructura de l'arbre familiar de FamilySearch

veracitat de les dades i investigacions realitzades sobre aquestes persones i línies genealògiques.

En els següents apartats de la memòria s'estudiarà el conjunt de recursos principals que conformen cada bloc de l'arbre familiar i quines són les diferents peces d'informació accessibles a través de cada un d'aquests recursos.

No es representarà en aquest projecte el conjunt total d'operacions diferents que pot ser realitzat sobre els diferents recursos, ja que les possibilitats de configuració d'aquestes són molt elevades i no té gaire sentit duplicar tota la documentació oficial respecte aquest punt.

Tanmateix, sí que volem mencionar que gairebé tots els recursos que formen part del model de dades de FamilySearch, poden ser sotmesos als següents grups d'operacions:

- **Lectura:** Tots els recursos són, evidentment, llegibles. Cada un dels recursos conté diferents opcions de lectura i generalment, també solen ser personalitzables mitjançant la inclusió de diferents paràmetres.
- **Actualització:** Sempre que es disposi dels permisos adequats sobre les dades, els usuaris també poden realitzar operacions d'actualització per corregir errors, modificar la informació existent o bé afegir nova informació.

- **Esborrat:** Si es disposa dels permisos suficients, els usuaris poden esborrar informació incorrecta o duplicada dels diferents recursos, o el recurs sencer, mitjançant les operacions d'esborrat.
- **Creació:** En cas de voler afegir noves peces d'informació a les bases de dades de FamilySearch, siguin noves persones o informació específica relacionada a alguna persona que ja es troba en el sistema, això és realitzable a través de les operacions de creació de cada recurs.

Dit això, passem doncs a analitzar els principals recursos de cada bloc de l'arbre familiar i les peces d'informació més destacables que els conformen.

5.4 Recursos principals del bloc Persones

El centre d'aquest bloc de recursos és el recurs Persona. Aquest recurs, emmagatzema els detalls que identifiquen i caracteritzen a cada una de les persones de l'arbre així com la informació bàsica sobre els seus relatius més propers.

D'aquest recurs principal, pengen els enllaços cap als recursos que permeten obtenir informació sobre els relatius de la persona, els estudis realitzats sobre aquesta, l'historial de canvis, les fonts d'informació i les memòries afegides pels usuaris.

La imatge [ref] ofereix una visió de l'esquema que acabem de descriure i permet veure com el recurs Persona s'enllaça amb la resta de blocs que conformen l'arbre familiar 5.3.

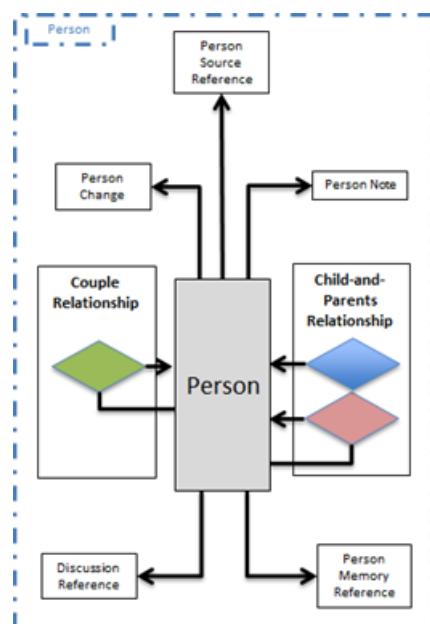


Figura 5.3: El bloc de l'arbre familiar relatiu a les Persones.

Aquest bloc de recursos és el més gran de tots i pràcticament, emmagatzema tota

la informació rellevant dels individus accessibles a través de l'API. En els següents subapartats s'exposaran els diferents recursos que conformen aquest bloc i quines són les peces d'informació utilitzables que aquests contenen.

Podreu observar, en les taules que representen l'estructura dels recursos, que a vegades, per la columna que marca el format de dades d'un paràmetre, aquest es troba especificat entre els caràcters '[' i ']'. Aquesta terminologia s'utilitza per indicar que aquest paràmetre és en realitat un recurs o objecte de dades diferent inclòs dins del recurs estudiat.

També s'observarà que sovint, els recursos exposats, hereten dades d'altres recursos i en els casos que aquests siguin rellevants, se n'exposarà l'estructura a l'apartat 'Altres recursos interessants', més endavant en la memòria.

5.4.1 El recurs Persona (Person)

El recurs Persona és el primer objecte amb què cal familiaritzar-se per tal de comprendre la potencialitat emmascarada d'aquesta API.

Cada instància, fa referència a una persona diferent de l'arbre familiar i generalment, representa el punt d'entrada per tal d'accedir a tota la informació disponible sobre un individu, ja sigui perquè aquesta es troba inclosa en el recurs o esdevé accessible a través dels enllaços hypermedia.

Els enllaços hypermedia del recurs Persona permeten accedir a la informació relativa als seus avantpassats, descendents, artefactes, historial de canvis, parelles, discussions, notes i fonts de dades.

Les dades pròpies pel recurs Persona poden ser observades a la taula 5.1. Cal recordar que aquest recurs també hereta els paràmetres dels recursos Subjecte, Conclusió, Enllaços Hypermedia i Dades Extensibles que poden ser trobats a la secció 'Altres recursos interessants'.

Taula 5.1: Codificació GEDCOM X del recurs Persona

Paràmetre	Format de Dades	Descripció
principal	boolean	Booleà per indicar si questa persona és la principal extreta d'un registre.
private	boolean	Indicador de si l'accés a la informació d'aquesta persona és visible pel domini públic o només privat.
living	boolean	Indicador sobre si la persona està viva o morta.
gender	[Gender]	El gènere de la persona consultada.
names	[array of Name]	Els diferents noms pels que és coneix a la persona.
facts	[array of Facts]	Els esdeveniments relacionats amb la vida de la persona.
fields	[array of Field]	Camps que s'utilitzen com evidència.

display	[DisplayProperties]	Recopilació d'informació bàsica d'accés ràpid sobre la persona per que resulti accessible sense necessitat de navegar pels diferents recursos.
---------	---------------------	--

5.4.2 El recurs Gènere (Gender)

El recurs Gènere s'utilitza per especificar el gènere d'una persona en concret. Aquest recurs conté els paràmetres propis mostrats a la taula 5.2 i hereta els camps de dades dels recursos , Conclusió, Enllaços Hypermedia i Dades Extensibles que poden ser trobats a la secció ‘Altres recursos interessants’.

Taula 5.2: Codificació GEDCOM X del recurs Persona

Paràmetre	Format de Dades	Descripció
type	[genderType]	El gènere de la persona relacionada al recurs. Se'n descriuen els possibles valors a l'enumeració genderType.
fields	[array of Field]	Camps que s'utilitzen com evidència.

L'enumeració genderType

L'enumeració genderType segueix l'estructura de definició GEDCOMX. Com a tal, els valors possibles per l'enumeració segueixen la pauta:
<http://gedcomx.org/> + ‘genderType’

La següent taula mostra els tres possibles valors de l'enumeració genderType.

Taula 5.3: Valors possibles per l'enumeració genderType

Male	Female	Unknown
------	--------	---------

5.4.3 Els recursos Nom, Forma del Nom i Part del Nom (Name, NameForm, NamePart)

Aquest conjunt de recursos s'utilitza per representar la informació relativa als noms d'una persona. Contenen informació sobre si un nom és el preferit de cara a ser utilitzat com a nom principal, en quin moment la persona va adoptar aquest nom

i diferents formes de representació.

Resulta útil poder accedir a diferents noms d'una mateixa persona, per poder així alternar, per exemple, entre el seu mot i el nom en el moment de naixement o defunció.

El recurs Nom està format pels paràmetres mostrats a la taula 5.4 i hereta també els paràmetres dels recursos Conclusió, Enllaços Hypermedia i Dades Extensibles que poden ser trobats a la secció ‘Altres recursos interessants’.

Per altre banda, els recursos Forma del Nom i Part del Nom, contenen els paràmetres mostrats a les taules 5.6 i 5.7 respectivament i hereten els paràmetres del recurs Dades Extensibles descrit en l'apartat ‘Altres recursos interessants’.

Taula 5.4: Paràmetres del recurs Nom

Paràmetre	Format de Dades	Descripció
type	[nameType]	El tipus de nom. Se'n descriuen els possibles valors a l'enumeració nameType.
preferred	Boolean	Indicador sobre si és el nom preferit de cara a la impressió de dades.
date	[Date]	El primer dia en què el nom va ser aplicat o adoptat.

L'enumeració nameType

L'enumeració nameType segueix l'estruatura de definició GEDCOMX. Com a tal, els valors possibles per l'enumeració segueixen la pauta:

<http://gedcomx.org/> + ‘nameType’

La següent taula mostra els possibles valors de l'enumeració nameType.

Taula 5.5: Valors possibles per l'enumeració nameType

BirthName	DeathName	MarriedName	AlsoKnownAs
Nickname	AdoptiveName	FormalName	ReligiousName

Taula 5.6: Paràmetres del recurs Forma del Nom

Paràmetre	Format de Dades	Descripció
lang	string	El llenguatge utilitzat. Representat segons els codis internacionals.

fullText	string	El nom en la seva forma completa.
parts	[array of NamePart]	Les diferents parts del nom.
fields	[array of Field]	Camps utilitats com evidència.

Taula 5.7: Paràmetres del recurs Part del Nom

Paràmetre	Format de Dades	Descripció
value	string	El valor com a text de la part del nom.
type	[namePartType]	Indica a quina part del nom fa referència aquesta instància del recurs. Se'n descriuen els possibles valors a l'enumeració namePartType.
fields	[array of Field]	Camps utilitats com evidència.
qualifiers	[array of Qualifier]	Els qualificadors associats a aquesta part del nom. No tenen per què ser valors numèrics. En aquests casos poden ser considerats més aviat com a etiquetes.

L'enumeració namePartType

L'enumeració namePartType segueix l'estructura de definició GEDCOMX. Com a tal, els valors possibles per l'enumeració segueixen la pauta:

<http://gedcomx.org/+ 'namePartType'>

La següent taula mostra els possibles valors de l'enumeració namePartType.

Taula 5.8: Valors possibles per l'enumeració namePartType

Prefix	Suffix	Given (principal)	Surname (cognom)

5.4.4 El recurs Esdeveniment (Fact)

Informació sobre un esdeveniment relacionat a la vida d'una persona o relació familiar. Conté informació sobre el tipus d'esdeveniment del qual es tracta, així com de la data i localització on va succeir.

Aquest recurs està format pels paràmetres mostrats a la taula [ref] i els paràmetres heretats dels recursos Conclusió, Enllaços Hypermedia i Dades Extensibles que poden ser trobats a la secció 'Altres recursos interessants'.

Taula 5.9: Paràmetres del recurs Esdeveniment

Paràmetre	Format de Dades	Descripció
primary	boolean	Indicador de si aquest esdeveniment és el principal esdeveniment extret del registre on es troava contingut.
type	[factType]	El tipus d'esdeveniment. Se'n descriuen els possibles valors a l'enumeració factType.
date	[Date]	La data en la que va succeir aquest esdeveniment.
place	[PlaceReference]	El lloc on va succeir aquest esdeveniment.
value	string	El valor aportat per l'usuari.
qualifier	[array of Qualifier]	Els qualificadors associats a aquesta part del nom. No tenen per què ser valors numèrics. En aquests casos poden ser considerats més aviat com a etiquetes.
fields	[array of Field]	Els camps utilitzats com evidència.

L'enumeració factType

L'enumeració factType segueix l'estructura de definició GEDCOMX. Com a tal, els possibles valors per l'enumeració segueixen la pauta:

<http://gedcomx.org/> + 'factType'

La següent taula mostra els possibles valors de l'enumeració factType.

Taula 5.10: Valors possibles per l'enumeració factType

Adoption	AdultChristening	Amnesty	Apprenticeship
Arrest	Baptism	BarMitzvah	BatMitzvah
Birth	Blessing	Burial	Caste
Census	Christening	Circumcision	Clan
Confirmation	Cremation	Court	Death
Education	EducationEnrollment	Emigration	Ethnicity
Excommunication	FirstCommunion	Funeral	GenderChange
Graduation	Immigration	Imprisonment	Inquest
LandTransaction	Language	Living	MaritalStatus
Medical	MilitaryAward	MilitaryDischarge	MilitaryDraft-Registration
MilitaryInduction	MilitaryService	Mission	MoveFrom
MoveTo	MultipleBirth	NationalId	Nationality
Naturalization	NumberOfMarriages	Obituary	Occupation
Ordination	Pardon	PhysicalDescription	Probate
Property	Religion	Residence	Retirement
Stillbirth	TaxAssessment	Will	Visit

Yahrzeit	Annulment	CommonLaw-Marriage	CivilUnion
Divorce	DivorceFiling	DomesticPartnership	Engagement
Marriage	MarriageBanns	MarriageContract	MarriageLicense
MarriageNotice	NumberOfChildren	Separation	AdoptiveParent
BiologicalParent	FosterParent	GuardianParent	StepParent
SociologicalParent			

5.4.5 El recurs Data (Date)

Aquest recurs conté la informació sobre les dates relacionades a alguna dada genealògica i diferents formats d'aquesta.

En concret, emmagatzema la informació pròpia que es mostra a la taula 5.11 i hereta els paràmetres del recurs Dades Extensibles que pot ser trobar a la secció ‘Altres recursos interessants’.

Taula 5.11: Paràmetres del recurs Data

Paràmetre	Format de Dades	Descripció
original	string	Data original tal com va ser introduïda per l'usuari.
formal	string	El valor estandarditzat o normalitzat de la data.
normalized	[array of TextValue]	Llista de valors formalitzats per la data. Proporcionat principalment per facilitat d'ús a l'hora de mostrar les dates.
fields	[array of Field]	Camps utilitzat com evidència.

5.4.6 El recurs Referència de localització (PlaceReference)

Aquest recurs conté informació sobre localitzacions concretes vinculades a alguna dada genealògica.

Aquest recurs conté les dades pròpies que es mostren a la taula 5.12 i hereta també els paràmetres del recurs Dades Extensibles que pot ser trobat a la secció ‘Altres recursos interessants’.

Taula 5.12: Paràmetres del recurs Referència de localització

Paràmetre	Format de Dades	Descripció

description	[PlaceDescription]	Descripció del lloc al què fa referència el recurs.
original	string	El valor original tal com va ser introduït per l'usuari.
normalized	[array of TValue]	La llista de valors normalitzats sobre la localització.
fields	[array of Field]	Els camps utilitzats com evidència.

5.4.7 El recurs Descripció de Localització (PlaceDescription)

Aquest recurs descriu els detalls d'una localització . Pretén representar la fotografia d'un lloc en un moment específic de la història.

A part dels paràmetres que seran descrits a continuació a la taula 5.13, aquest recurs també hereta tots els paràmetres dels recursos Subjecte, Conclusió, Enllaços Hypermedia i Dades Extensibles que poden ser trobats a la secció ‘Altres recursos interessants’.

Taula 5.13: Paràmetres del recurs Descripció de localització

Paràmetre	Format de Dades	Descripció
type	string	Identificador que expressa si la localització es tracta d'una adreça ciutat estat país etcètera.
names	[array of TValue]	Una llista ordenada de valors estandarditzats o normalitzats que descriuen la localització.
temporal-Description	[Date]	Descripció del moment temporal en que aquest nom aplica per la localització.
latitude	double	Desviació cap al nord o sud respecte a l'equador.
longitude	double	Desviació cap a l'oest o est respecte a l'equador.
spatial-Description	[ResourceReference]	Referència a la descripció geoespatial de la localització.
place	[ResourceReference]	Referència a la localització a la qual es fa referència.
jurisdiction	[ResourceReference]	Referència a la jurisdicció de la localització.
display	[PlaceDisplay-Properties]	Propietats de visualització de la localització. Format per un conjunt de strings que permeten pintar el nom complet curt i el tipus de localització.

5.4.8 El recurs Camps Bàsics (DisplayProperties)

Aquest recurs conté el conjunt de propietats bàsiques d'una persona recopilades en un sol recurs. L'objectiu principal és facilitar l'accés a les dades més comunes per tal d'incrementar l'eficiència a l'hora de mostrar informació als usuaris i reduir així el nombre d'interaccions i connexions necessàries amb l'API.

Com a extra, totes les propietats són localitzades amb l'idioma del local utilitzat.

Les dades pròpies per aquest recurs s'indiquen a la taula 5.14 i el recurs també hereta les dades del recurs Dades Extensibles que pot ser trobat a la secció 'Altres recursos interessants'.

Taula 5.14: Paràmetres del recurs Camps Bàsics

Paràmetre	Format de Dades	Descripció
ascendancy-Number	string	El nombre de generació en l'ascendència familiar en relació a les altres persones incloses a la consulta.
birthDate	string	El text mostrable per la data de naixement de la persona.
birthPlace	string	El text mostrable pel lloc de naixement de la persona.
deathDate	string	El text mostrable per la data de defunció de la persona.
deathPlace	string	El text mostrable pel lloc de defunció de la persona.
descendancy-Number	string	El nombre de generació en la descendència familiar en relació a les altres persones incloses a la consulta.
families-AsChild	[array of FamilyView]	Els arbres genealògics en les que aquesta persona té el rol de fill.
families-AsParent	[array of FamilyView]	Els arbres genealògics en les que aquesta persona té el rol de pare o mare.
gender	string	El text mostrable pel gènere de la persona.
lifespan	string	El text mostrable del període de temps en què la persona va viure.
marriage-Date	string	El text mostrable per la data de casament de la persona.
marriage-Place	string	El text mostrable pel lloc de casament de la persona.
name	string	El nom mostrable de la persona.

5.4.9 El recurs Vista de Família (FamilyView)

Aquest recurs conté informació bàsica sobre les relacions entre pares i fills. El recurs Relacions conté la informació canònica respecte a aquestes relacions i és el recurs que ha de ser utilitzat si es vol extreure més informació d'aquestes.

Tanmateix, si només es desitja la informació bàsica de la relació, aquest recurs pot resultar molt convenient.

Aquest recurs conté les dades pròpies que es mostren a la taula 5.15 i hereta també les dels recursos Enllaços Hypermedia i Dades Extensibles que poden ser trobades a la secció ‘Altres recursos interessants’.

Taula 5.15: Paràmetres del recurs Vista de Família

Paràmetre	Format de Dades	Descripció
parent1	[ResourceReference]	Referència al primer pare de la família. La posició del pare 1 no és indicadora del rol d'aquest en la relació.
parent2	[ResourceReference]	Referència al segon pare de la família. La posició del pare 2 no és indicadora del rol d'aquest en la relació.
children	[array of ResourceReference]	Referències als fills de la relació parental.

5.5 Recursos principals del bloc Relacions Familiars

Aquest bloc està format principalment per dos recursos que permeten la representació de relacions parentals i les relacions de parella.

S'utilitza el recurs Relació per representar les relacions de parella i el recurs Relació Pares i Fill per representar la relació entre dos pares i un fill, on un dels pares pot no ser especificat. Destaquem aquest fet, perquè antigament FamilySearch només suportava el conjunt de famílies nuclears. És a dir, aquelles formades per una parella completa i la seva descendència, si és que aquesta existia.

Així doncs, els recursos Relació i Relació Pares i Fill, proporcionen informació sobre les persones que en formen part i els esdeveniments associats a aquestes relacions. Per exemple, informació sobre el matrimoni.

Els dos recursos, que representen les dues relacions familiars de màxima proximitat, poden relacionar-se mitjançant enllaços hypermedia, amb els recursos Nota i Historial de Canvis. Aquests recursos contenen informació extra afegida pels usuaris i un historial complet de com la relació s'ha vist modificada al llarg del temps així com el motiu d'aquests canvis. La figura 5.4 mostra l'esquema específic de l'estruatura d'aquest bloc.

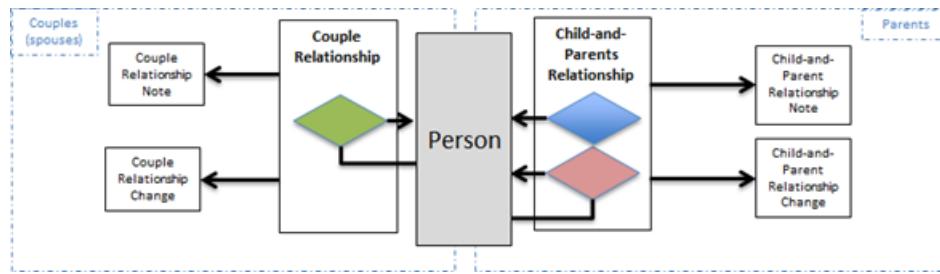


Figura 5.4: El bloc de l'arbre familiar relatiu a les relacions

Podreu observar també, a les taules que representen l'estrucció dels recursos, que a vegades, per la columna que marca el format de dades d'un paràmetre, aquest es troba especificat entre els caràcters '[' i ']'. Aquesta terminologia s'utilitza per indicar que aquest paràmetre és en realitat un recurs o objecte de dades diferent inclòs dins del recurs estudiat.

També s'observarà que sovint, els recursos exposats, hereten dades d'altres recursos i en els casos que aquests siguin rellevants, se n'exposarà l'estrucció a l'apartat 'Altres recursos interessants', més endavant en la memòria.

5.5.1 El recurs Relació (Relationship)

Aquest recurs s'utilitza en l'actualitat per representar només les relacions de parella. En el passat, també va ser utilitzat per representar relacions entre pares i fills, mitjançant l'ús del paràmetre *type* i l'enumeració *relationshipType*. Tanmateix, aquest ha caigut en el desús des de la incorporació del recurs Relacions Pares i Fill.

Aquest recurs emmagatzema informació sobre les persones que conformen la relació i els esdeveniments relacionats a aquesta. Les dades pròpies del recurs es mostren a la taula 5.16 i també hereta les dels recursos Subjecte, Conclusió, Enllaços Hypermedia i Dades Extensibles que poden ser trobats a l'apartat 'Altres recursos interessants'.

Taula 5.16: Paràmetres del recurs Relació

Paràmetre	Format de Dades	Descripció
type	[relationshipType]	Descriu el tipus de relació.
person1	[ResourceReference]	Referència al recurs de la primera persona que participa en la relació.
person2	[ResourceReference]	Referència al recurs de la segona persona que participa en la relació.
facts	[array of Fact]	Els fets relacionats amb la relació.
fields	[array of Fields]	Llista de camps utilitzats com evidència.

L'enumeració relationshipType

L'enumeració relationshipType segueix l'estructura de definició GEDCOMX. Com a tal, els valors possibles per l'enumeració segueixen la pauta:
<http://gedcomx.org/+relationshipType>

La següent taula mostra els possibles valors per l'enumeració relationshipType, però com ja hem comentat, l'ús del valor ParentChild, ja no s'utilitza en favor del nou recurs Relació Pares i Fill.

Taula 5.17: Valors possibles per l'enumeració relationshipType

Couple	ParentChild
--------	-------------

5.5.2 El recurs Relació Pares i Fill (ChildAndParentsRelationship)

Com ja hem comentat, aquest recurs s'utilitza per representar les relacions entre dos pares i un fill. Existeix també la possibilitat de deixar un pare sense especificar, permetent així, la introducció de famílies monoparentals en el sistema.

Aquest recurs conté informació sobre les persones que conformen els rols de pare, mare i fill, així com el conjunt d'esdeveniments relacionats amb el pare i la mare.

Els paràmetres del recurs són descrits sa la taula 5.18 i recordar que també hereta els paràmetres dels recursos Subjecte, Conclusió, Enllaços Hypermedia i Dades Extensibles que poden ser trobats a l'apartat ‘Altres recursos interessants’.

Taula 5.18: Paràmetres del recurs Relació Pares i Fill

Paràmetre	Format de Dades	Descripció
father	[ResourceReference]	Referència al recurs del pare.
mother	[ResourceReference]	Referència al recurs de la mare.
child	[ResourceReference]	Referència al recurs del fill.
fatherFacts	[array of Fact]	Els esdeveniments relacionats amb el pare.
motherFacts	[array of Fact]	Els esdeveniments relacionats amb la mare.

5.6 Recursos principals del bloc Discussions

El recurs principal d'aquest bloc de l'arbre familiar rep el nom de Discussió, que es troba principalment composta, per comentaris creats mitjançant el recurs Comentari.

Les discussions a FamilySearch són tòpics de discussió introduïts pels usuaris i relacionades a una persona en concret. Aquestes discussions estan formades per diferents comentaris i destaca la utilització d'un recurs intermedi per fer de pont entre les discussions i els recursos de les persones les quals fan referència accessible a través dels enllaços hypermedia.

El contingut d'aquestes discussions és divers, però generalment són utilitzades per discutir entre diferents usuaris sobre les dades relatives a una persona, l'origen de les fonts de dades i similars.

La figura 5.5 mostra com es troben relacionats els recursos que conformen el bloc Discussions.

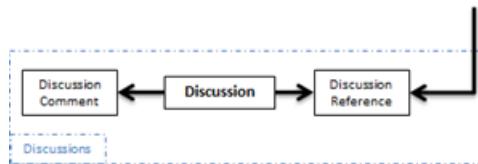


Figura 5.5: El bloc de l'arbre familiar relatiu a les discussions

Podreu observar també, a les taules que representen l'estrucció dels recursos, que a vegades, per la columna que marca el format de dades d'un paràmetre, aquest es troba especificat entre els caràcters '[' i ']'. Aquesta terminologia s'utilitza per indicar que aquest paràmetre és en realitat un recurs o objecte de dades diferent inclòs dins del recurs estudiat.

També s'observarà que sovint, els recursos exposats, hereten dades d'altres recursos i en els casos que aquests siguin rellevants, se n'exposarà l'estrucció a l'apartat 'Altres recursos interessants', més endavant en la memòria.

5.6.1 El recurs Referència a la Discussió (DiscussionsReference)

Aquest recurs s'utilitza principalment perquè el recurs Persona pugui accedir al conjunt de discussions que tracten sobre ella i viceversa. En concret, existirà un enllaç per cada una de les discussions que reverenciïn a la mateixa persona.

Les dades contingudes per aquest recurs poden ser trobades a la taula 5.19 i també hereta els paràmetres dels recursos Enllaços Hypermedia i Dades Extensibles que poden ser trobats a l'apartat 'Altres recursos interessants'.

Taula 5.19: Paràmetres del recurs Referència a la Discussió

Paràmetre	Format de Dades	Descripció
resourceId	string	El ID de la Discussió a la qual es fa referència.
resource	anyURI	Direcció URI del recurs al què és fa referència.
attribution	[Attribution]	Metadades de l'atribució per aquest recurs.

5.6.2 El recurs Discussió (Discussion)

El recurs Discussió és utilitzat per representar el tòpic sobre el qual tractarà la conversació entre els diferents usuaris de FamilySearch i emmagatzemar el conjunt de comentaris introduïts per aquests.

El recurs Discussió, esdevé un objecte bastant simple, contenint només la informació necessària perquè altres usuaris puguin comprendre el tòpic de discussió, les metadades de la creació i accedir al conjunt de comentaris.

Aquest recurs està format pels paràmetres mostrats a la taula 5.20 i també hereta els paràmetres dels recursos Enllaços Hypermedia i Dades Extensibles que poden ser trobats a l'apartat ‘Altres recursos interessants’.

Taula 5.20: Paràmetres del recurs Discussió

Paràmetre	Format de Dades	Descripció
title	string	Títol o breu descripció del tòpic a discutir.
details	string	Text detallat introduït per l'usuari per tal d'iniciar la discussió.
created	dateTime	La data i moment en què la discussió va ser creada.
contributor	[ResourceReference]	Referència al recurs del contribuïdor que va crear la discussió.
modified	dateTime	Data i moment en què l'últim comentari o canvi va ser introduït a la discussió.
numberOfComments	int	Nombre de comentaris associats a la discussió.
comments	[array of Comment]	Vector amb els diferents comentaris que conformen la discussió.

5.6.3 El recurs Comentari (Comment)

El recurs Comentari conté els missatges introduïts pels diferents usuaris com a resposta a una discussió. Principalment, conté informació sobre la data de creació, l'usuari que l'ha enviat i el text en qüestió propi del comentari.

Les dades pròpies d'aquest recurs es mostren a la taula 5.21 i també hereta els paràmetres dels recursos Enllaços Hypermedia i Dades Extensibles que poden ser trobats a l'apartat 'Altres recursos interessants'.

Taula 5.21: Paràmetres del recurs Comentari

Paràmetre	Format de Dades	Descripció
text	string	Text del comentari.
created	dateTime	Data i moment de creació del comentari.
contributor	[ResourceReference]	Referència al recurs de l'usuari que va enviar el comentari.

5.7 Recursos principals del bloc Memòries

El bloc Memòries és relativament nou a l'API de FamilySearch i la documentació al respecte és pràcticament inexistent. Aquest bloc conté les memòries penjades al núvol pels usuaris i són relacionades amb les persones de l'arbre familiar.

Recordem, que el concepte memòries ha estat descrit a la secció tres de la memòria, 'L'organització Familysearch' i consisteixen principalment en contingut fotogràfic, històries, documents diversos i fitxers d'àudio.

El recurs principal d'aquest bloc és el recurs Memòria, que conté tota la informació específica d'aquesta. Les memòries també contenen comentaris, en un estil molt similar a les discussions i també disposen de recursos intermedis per relacionar les memòries amb les persones a les quals fan referència. Com sempre, moltes d'aquestes connexions són implementades a través d'enllaços hypermedia.

De moment, resulta necessari crear una instància del recurs Memòria per cada contingut que es vulgui pujar al sistema, però la possibilitat de suportar més d'un artefacte amb el mateix recurs memòria ha estat estudiat i en algun moment o altre serà implementat. Un exemple de cas d'ús podria ser la necessitat de pujar les dues cares d'una fotografia.

La imatge 5.6 ofereix un esquema de forma visual de com es relacionen els recursos vinculats al bloc Memòries.

Desgraciadament, la documentació referent a aquest apartat per part de Fam-

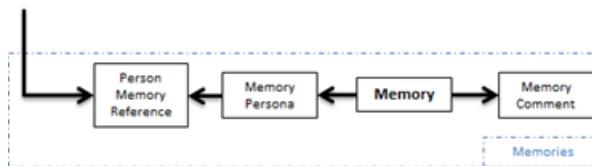


Figura 5.6: El bloc de l'arbre familiar relatiu a les memòries

ilySearch és molt pobre i el contingut de cada un dels recursos ha estat creat per inferència mitjançant el collage de petites peces d'informació i les similituds que presentaven amb altres recursos similars de l'API. És possible que els recursos presentats a continuació no descriguin amb total precisió la realitat exacta.

Podreu observar també, a les taules que representen l'estructura dels recursos, que a vegades, per la columna que marca el format de dades d'un paràmetre, aquest es troba especificat entre els caràcters '[' i ']'. Aquesta terminologia s'utilitza per indicar que aquest paràmetre és en realitat un recurs o objecte de dades diferent inclòs dins del recurs estudiat.

També s'observarà que sovint, els recursos exposats, hereten dades d'altres recursos i en els casos que aquests siguin rellevants, se n'exposarà l'estructura a l'apartat 'Altres recursos interessants', més endavant en la memòria.

5.7.1 El recurs Referència a la Memòria d'una Persona (Person Memory Reference)

Aquest recurs és utilitzat com a pont entre el recurs Persona i el contingut específic de les memòries. Existirà un enllaç hypermedia diferent per cada una de les memòries a les quals el recurs Persona hagi de tenir accés i viceversa.

Les dades contingudes per aquest recurs poden ser trobades a la taula 5.22 i també hereta els paràmetres dels recursos Enllaços Hypermedia i Dades Extensibles que poden ser trobats a l'apartat 'Altres recursos interessants'.

Taula 5.22: Paràmetres del recurs Referència a la Memòria d'una Persona

Paràmetre	Format de Dades	Descripció
resourceId	string	El ID de la memòria relacionada a una persona de l'arbre familiar.
resource	anyURI	Direcció URI del recurs al qual es fa referència.
attribution	[Attribution]	Metadades de l'atribució per aquest recurs.

5.7.2 El recurs Persones en una Memòria (MemoryPersona)

Aquest recurs s'utilitza com a pont per accedir a totes les persones que han estat marcades com a relacionades en una memòria. Per exemple, si una fotografia conté la imatge de diverses persones i es vol relacionar la memòria amb totes elles, aquest recurs ho fa possible sense la necessitat d'haver de pujar la imatge per cada usuari.

La taula 5.23 mostra els paràmetres d'aquest recurs que també hereta els paràme-

tres dels recursos Enllaços Hypermedia i Dades Extensibles que poden ser trobats a l'apartat ‘Altres recursos interessants’.

Taula 5.23: Paràmetres del recurs Persones en una memòria

Paràmetre	Format de Dades	Descripció
persons	[array of ResourceReference]	Serveix per enllaçar les diferents persones relacionades amb una memòria en concret.
source-Descriptions	[ResourceReference]	Informació sobre la memòria a la qual es fa referència.

5.7.3 El recurs Memòria (Memory)

El recurs Memòria, com el seu nom indica, és el recurs principal utilitzat per emmagatzemar la informació relacionada amb un artefacte pujat per un usuari. Per fer-ho, s'utilitza el recurs Descripció de la Font de Dades, que s'especificarà més endavant a l'apartat de recursos relacionats amb el bloc de Fonts de Dades.

Aquest recurs també inclou instàncies dels comentaris que han estat associats a les memòries.

Els paràmetres d'aquest recurs es mostren a la taula 5.24 i també hereta els paràmetres dels recursos Enllaços Hypermedia i Dades Extensibles que poden ser trobats a l'apartat ‘Altres recursos interessants’.

Taula 5.24: Paràmetres del recurs Memòria

Paràmetre	Format de Dades	Descripció
source-Descriptions	[SourceDescription]	Informació de la memòria. Tipus de document dimensions contribuïdor descripció etcètera.
comments	[array of Comment]	Vector de comentaris.

5.8 Recursos principals del bloc Fonts de Dades

Aquest bloc de l'arbre familiar destaca pels recursos Col·lecció i Font de Dades, que s'utilitzen per certificar la veritat de les dades referents a les persones que es poden trobar a l'arbre familiar o les relacions que les uneixen.

Aquest bloc és realment important, ja que és l'encarregat de garantir la integritat

de les dades i per tant, d'assegurar que la informació continguda a l'arbre familiar de FamilySearch sigui usable i interessant pels usuaris de l'aplicació.

Així doncs, les fonts de dades són utilitzades per demostrar que un esdeveniment en concret va succeir o que les dades de la persona citada són les mateixes que aquelles redactades en els documents oficials.

Les fonts de dades s'agrupen i emmagatzemen en col·leccions i cada persona o relació, disposa d'un recurs que actua com intermediari per relacionar-los amb les fonts de dades. Com sempre, els enllaços explícits entre els diferents recursos es realitzen mitjançant els enllaços hypermedia. La imatge 5.7 ofereix una vista de com s'estructuren i relacionen els diferents recursos del bloc Font de Dades.

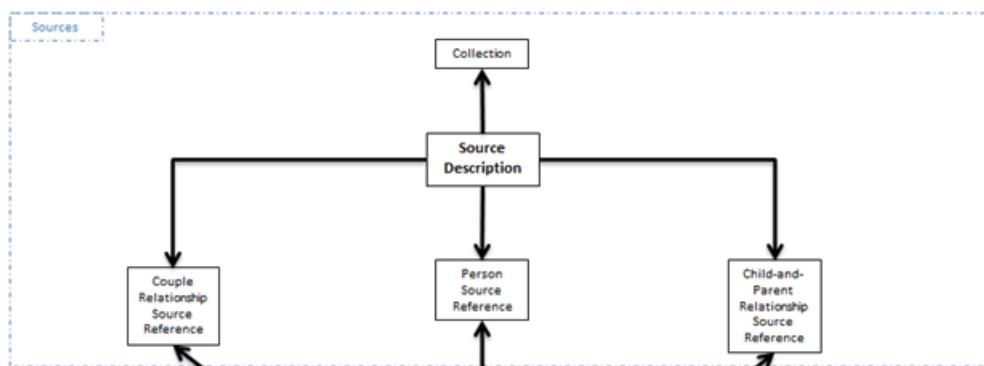


Figura 5.7: El bloc de l'arbre familiar relatiu a les fonts de dades

Podreu observar també, a les taules que representen l'estrucció dels recursos, que a vegades, per la columna que marca el format de dades d'un paràmetre, aquest es troba especificat entre els caràcters '[' i ']'. Aquesta terminologia s'utilitza per indicar que aquest paràmetre és en realitat un recurs o objecte de dades diferent inclòs dins del recurs estudiat.

També s'observarà que sovint, els recursos exposats, hereten dades d'altres recursos i en els casos que aquests siguin rellevants, se n'exposarà l'estrucció a l'apartat 'Altres recursos interessants', més endavant en la memòria.

5.8.1 El recurs Col·lecció (Collection)

Aquest recurs representa una col·lecció o agregació, de fonts de dades de caràcter genealògic.

Aquestes dades poden fer referència tant a les dades internes de FamilySearch, com als registres bolcats per tercers a les bases de dades. Per tal d'ajudar al lector a fer-se una idea, una col·lecció podria ser per exemple el conjunt de fonts de dades que conformen l'arbre familiar.

La taula 5.25 mostra els paràmetres que aquest recurs incorpora i també hereta els paràmetres dels recursos Enllaços Hypermedia i Dades Extensibles que poden ser

trobats a l'apartat 'Altres recursos interessants'.

Taula 5.25: Paràmetres del recurs Col·lecció

Paràmetre	Format de Dades	Descripció
lang	string	El llenguatge en què es troba la descripció de la col·lecció.
identifiers	[array of Identifier]	Llista d'identificadors de les fonts de dades.
title	string	Títol de la col·lecció.
size	int	Les dimensions de la col·lecció en referència al nombre d'elements que conté.
content	[array of Collection-Content]	Descripcions del contingut de la col·lecció.
attribution	[Attribution]	Metadades d'atribució referents a la col·lecció.

5.8.2 El recurs Contingut de la Col·lecció (CollectionContent)

Aquest recurs conté informació específica sobre la col·lecció a la que fa referència. Serveix per comprendre millor l'estat actual de migració de la col·lecció a les bases de dades de FamilySearch, així com informació sobre el contingut que aquesta aporta.

La taula 5.26 mostra els paràmetres inclosos en el recurs, que a la vegada hereta els paràmetres dels recursos Enllaços Hypermedia i Dades Extensibles que poden ser trobats a l'apartat 'Altres recursos interessants'.

Taula 5.26: Paràmetres del recurs Contingut de la Col·lecció

Paràmetre	Format de Dades	Descripció
completeness	float	Percentatge de dades ja migrades a FamilySearch.
count	int	El nombre total d'elements que poden ser extrets de la col·lecció relacionats amb el resourceType definit en el següent paràmetre.
resourceType	string	El tipus de recurs sobre el qual la col·lecció aporta informació.

5.8.3 El recurs Atribució (Attribution)

El recurs Atribució s'utilitza en més llocs que les fonts de dades, però com que juga un paper especialment important en aquest, hem decidit incloure'l dins d'aquesta secció.

Aquest recurs conté principalment paràmetres que permeten descriure qui, quan i perquè, va ser realitzada una contribució o modificació sobre les dades.

Els paràmetres d'aquest recurs són descrits a la taula 5.27, que també hereta els paràmetres del recurs Dades Extensibles que pot ser trobat a l'apartat 'Altres recursos interessants'.

Taula 5.27: Paràmetres del recurs Atribució

Paràmetre	Format de Dades	Descripció
contributor	[ResourceReference]	Referència al recurs del contribuïdor.
modified	dateTime	El moment en què la dada va ser modificar per últim cop.
change-Message	string	El missatge que descriu el canvi realitzat i introduït pel contribuïdor.
creator	[ResourceReference]	Referència al creador de les dades atribuïdes. Pot ser diferent al contribuïdor si les dades s'han vist modificades des de la seva creació.
created	datetime	El moment en què van ser creades les dades a les quals es fa referència.

5.8.4 El recurs Font de Dades (SourceDescription)

Aquest recurs defineix una font de dades i n'emmaigatzema tota la informació que la caracteritza. Les fonts de dades formen part d'una col·lecció i es troben relacionades a ella mitjançant enllaços hypermedia.

La millor forma d'explicar aquest recurs és comprendre'n els paràmetres i aquests s'exposen a la taula 5.28. El recurs Font de Dades també hereta els paràmetres dels recursos Enllaços Hypermedia i Dades Extensibles que poden ser trobats a l'apartat 'Altres recursos interessants'.

Taula 5.28: Paràmetres del recurs Font de Dades

Paràmetre	Format de Dades	Descripció
about	anyURI	La URI (si existeix) de l'actual font de dades.

lang	string	L'idioma o llengua en què es troba la font de dades.
mediaType	string	Pista sobre el format del document que la font de dades descriu.
version	string	Versió del recurs.
sortKey	string	Clau per identificar la posició relativa de la font de dades dins de la col·lecció.
resource-Type	[resourceType]	El tipus de recurs descrit.
citations	[array of SourceCitation]	Les citacions bibliogràfiques de la font de dades.
mediator	[ResourceReference]	Referència a l'entitat que fa de mediadora entre FamilySearch i les dades.
sources	[array of SourceReference]	Referències a altres fonts de dades relacionades. Generalment aplicable quan una font de dades forma part de la categoria secundària d'una altra font de dades.
analysis	[ResourceReference]	Referència al document d'anàlisis realitzat per descriure la font de dades.
component-Of	[SourceReference]	Referència a la font de dades de la que forma part el recurs (si existeix).
titles	[array of TValue]	Llista de títols per la font de dades.
titleLabel	[TextValue]	Títol preferit per la descripció de la font de dades.
notes	[array of Note]	Notes relacionades amb la font de dades.
attribution	[Attribution]	Les metadades d'atribució per la font de dades.
descriptions	[array of TValue]	Descripcions de la font de dades de forma redactada.
identifiers	[array of Identifier]	La llista d'identificadors de la font de dades.
created	dateTime	La data i moment en què la font de dades va ser creada.
modified	dateTime	La data i moment en què la font de dades va ser modificada per últim cop.
coverage	[array of Coverage]	Declaracions de l'abast cobert per la font de dades en termes de tipus de registre localització geogràfica i època .
rights	array of string	Els drets sobre les dades contingudes en la Font de Dades.
fields	[array of Field]	Els camps utilitzats com evidència aplicables al recurs descrit.
Repository	[ResourceReference]	Referència a l'agent que descriu on pot ser trobada la font de dades.
descriptor	[ResourceReference]	Referència a una sèrie de descriptors que indiquen quines dades es poden esperar de la font de dades.
replacedBy	string	La URI per la que aquest recurs s'ha reemplaçat.

replaces	array of string	La llista de recursos als quals aquesta font de dades substitueix.
statuses	array of string	La llista d'estats que pot tenir aquesta font de dades.

L'enumeració resourceType

L'enumeració resourceType segueix l'estructura de definició GEDCOMX. Com a tal, els valors possibles per l'enumeració segueixen la pauta:

<http://gedcomx.org/> + 'resourceType'

La següent taula mostra els possibles valors per l'enumeració resourceType.

Taula 5.29: Valors possibles per l'enumeració resourceType

Record	Collection	DigitalArtifact
PhysicalArtifact	Person	

5.8.5 El recurs Referència a la Font de Dades (SourceReference)

Aquest recurs s'utilitza per fer de pont entre els diferents recursos de l'arbre familiar que contenen informació genealògica i les fonts de dades que l'han proporcionat.

Com s'ha pogut veure en la imatge [ref] que obria el bloc de Fonts de Dades, el recurs Referència a la Font de Dades aplica tant a les relacions familiars com a les persones. De la mateixa forma, també es veu utilitzat aquest recurs per referenciar unes Fonts de Dades, amb algunes altres.

A part dels paràmetres que conformen aquest recurs i que es mostren a la taula 5.30, també hereta els paràmetres dels recursos Enllaços Hypermedia i Dades Extensibles que poden ser trobats a l'apartat 'Altres recursos interessants'.

Taula 5.30: Recurs Referència a la Font de Dades

Paràmetre	Format de Dades	Descripció
description	string	Descripció de la font de dades referenciada.
descriptionId	string	Identificador de la font de dades.
attribution	[Attribution]	Les metadades d'atribució per la font de dades.
qualifiers	[array of Qualifier]	Els qualificadors associats a aquesta part del nom. No tenen per què ser valors numèrics. En aquests casos poden ser considerats més aviat com a etiquetes.

5.9 Altres recursos interessants

Com haureu pogut observar després de la lectura de les seccions anteriors, no tots els objectes o recursos que han anat apareixent com a paràmetres dels diferents recursos estudiants, han estat descrits en profunditat. Les raons són diverses segons el recurs en qüestió, però alguns motius comuns són els següents:

- És un recurs orientat a la manipulació de les dades més que a proporcionar informació genealògica a l'usuari i per tant, no acaba de prendre valor per l'estudi realitzat en els apartats anteriors.
- Es tracta d'un recurs simple del qual no s'aportaria més informació mostrant-ne l'estructura, que simplement disposant de la descripció del camp.
- És un recurs utilitzat en cassos molt específics i per tant, no aplicable en un àmbit general.

Això no obstant, sí que hi ha altres recursos que s'han vist representats a les seccions anteriors de forma transversal, com per exemple en les imatges que unen els diferents recursos d'un bloc, i que volem explicar. Exemples d'aquests recursos són les notes i els històrics de canvis.

També existeix un conjunt de recursos que hem anat mencionant en els apartats anteriors, on els recursos estudiats, heretaven els paràmetres d'aquests. Aquests recursos incorporats de forma global a molts altres, també seran estudiats en aquest apartat de la memòria.

A part d'aquests dos grups de recursos, també volem descriure'n d'altres que considerem que sota certes circumstàncies podrien resultar interessants i encara que no s'han vist relacionats de forma directa amb els blocs de dades anteriors, volem deixar-ne constància en el projecte.

5.9.1 El recurs Subjecte (Subject)

El recurs Subjecte fa referència al concepte abstracte de subjecte genealògic, entenent-lo com una entitat única que o bé pot fer referència a una persona o a una localització concreta sobre el globus terraquí.

Aquesta entitat única, agrega i emmagatzema les diferents evidències i fonts de dades particulars del subjecte, és a dir, el col·lectiu d'informació que fan que aquest subjecte sigui diferent dels altres.

La taula 5.31 mostra al que ens referim quan parlem d'aquesta agregació d'informació.

Taula 5.31: Paràmetres del recurs Subjecte

Paràmetre	Format de Dades	Descripció
extracted	boolean	Indica si aquesta persona o localització està marcada com 'extreta'. S'utilitza aquesta denominació quan la informació d'una persona o localització prové de la mateixa font de dades.
evidence	[array of Evidence-Reference]	Referències a les evidències relacionades al subjecte. Cada evidència conté un enllaç al recurs que emmagatzema la informació.
media	[array of Source-Reference]	Referències als arxius multimèdia associats al subjecte. Representen les memòries relacionades a la persona.
identifiers	[array of Identifier]	Llista d'identificadors relacionats amb la persona o localització.

5.9.2 El recurs Conclusió (Conclusion)

En l'àmbit de l'API de FamilySearch, el terme conclusió fa referència al valor que pren una dada genealògica després de ser contrastada amb una font de dades. És doncs, l'avaluació del valor que certa dada genealògica, o conjunt de dades genealògiques, han rebut.

Aquest recurs conté els paràmetres que s'expressen a la taula 5.32:

Taula 5.32: Paràmetres del recurs Conclusió

Paràmetre	Format de Dades	Descripció
confidence	[confidenceLevel]	El nivell de confiança que el contribuïdor té sobre la veritat de les dades. Alt mitjà o baix.
sortKey	string	Una clau per facilitar l'ordenació de cara la impressió de les dades.
lang	string	El llenguatge en què està redactada la conclusió.
attribution	[Attribution]	Meta dades per l'atribució de les dades.
sources	[array of Source-Reference]	Referències a les fonts de dades que donen suport a la conclusió.
analysis	[ResourceReference]	Referència al document d'anàlisis explicant l'anàlisi que s'ha portat a terme per tal arribar a la conclusió.
notes	[array of Note]	Notes sobre la persona.

5.9.3 El recurs Enllaços Hypermedia (Hypermedia Enabled Data)

Aquest recurs es troba en pràcticament tots els altres recursos de l'API de FamilySearch. Són els responsables de poder navegar, a través dels diferents recursos que es troben enllaçats en el model de dades de FamilySearch, sense necessitat de codificar noves URIs de forma manual, o des d'una aplicació.

Aquest recurs consisteix bàsicament en un conjunt d'enllaços que apunten als diferents recursos. La taula 5.33 en mostra l'estructura en detall.

Taula 5.33: Paràmetres del recurs Enllaços Hypermedia

Paràmetre	Format de Dades	Descripció
links	[array of Links]	Llista d'enllaços hypermedia als recursos relacionats al recurs consultat. Aquests enllaços són els que ens permeten navegar entre recursos sense haver de posar-los al codi de forma explícita.

5.9.4 El recurs Enllaç (Link)

A diferència del que es pugui creure, en el context de l'API de FamilySearch, Link no fa referència al famós protagonista de la saga de videojocs 'Zelda', sinó a un enllaç hypermedia com el descrit en els comentaris d'aquesta secció de la memòria.

Com ja s'ha comentat nombroses vegades, aquests enllaços existeixen per facilitar la navegació entre recursos. L'estructura del recurs Enllaç pot veure's en detall a la taula 5.34.

Taula 5.34: Paràmetres del recurs Enllaç

Paràmetre	Format de Dades	Descripció
template	string	Una plantilla URI que segueix l'estàndard RFC 6570 utilitzat per referenciar a un rang de URIs.
allow	string	Metadades dels mètodes permisibles a l'hora de navegar entre un recurs i un altre.
count	int	El nombre de recursos a la pàgina referenciada si és que aquest enllaç fa referència a una pàgina de recursos.
accept	string	Metadades dels formats multimèdia acceptables.
type	string	Metadades dels formats multimèdia disponibles de l'enllaç que és referència.
hreflang	string	L'idioma en què es troba el recurs referenciat.

title	string	Títol descriptiu de l'enllaç.
results	int	El nombre total de resultats continguts a la pàgina enllaçada si és que el recurs enllaça a una pàgina de resultats. Per exemple la cerca de persones a l'arbre familiar.
rel	string	La relació d'enllaç. És el paràmetre utilitzat per comprendre de quina relació es tracta.
offset	int	Els resultats a saltar o resultat pel qual començarà la pàgina enllaçada si aquest enllaç fa referència a una pàgina de resultats.
hreflang	anyURI	La direcció URI de l'enllaç.

5.9.5 El recurs Dades Extensibles (Extensible Data)

Aquest recurs és probablement el més simple de tots. S'encarrega de dotar a tots els altres recursos de l'API de FamilySearch, amb un identificador local únic que permeti referenciar-lo a les bases de dades.

La taula 5.35 mostra l'estruatura d'aquest recurs.

Taula 5.35: Paràmetres del recurs Dades Extensibles

Paràmetre	Format de Dades	Descripció
id	string	Un identificador local per les dades.

5.9.6 El recurs Nota (Note)

Aquest recurs pot ser vinculat a persones, relacions i fonts de dades. És un recurs utilitzat per realitzar alguna anotació específica sobre les dades a les quals ha estat relacionat.

El recurs Nota, també hereta els paràmetres dels recursos Enllaços Hypermedia i Dades Extensibles, que poden trobar-se en aquest mateix apartat de la memòria i té per paràmetres propis els que es mostren a la taula 5.36.

Taula 5.36: Paràmetres del recurs Nota

Paràmetre	Format de Dades	Descripció
lang	string	La llengua en què està escrita la Nota.

subject	string	El subjecte de la nota. Breu títol que descriu els continguts.
text	string	El text de la nota.
attribution	[Attribution]	Meta dades d'atribució de la Nota.

5.9.7 El recurs Referència al Recurs (ResourceReference)

Un altre recurs que ha sortit molt fins ara, com a recurs contingut dins dels altres recursos, és el de Referència al Recurs. Aquest objecte, s'utilitza per enllaçar mitjançant un contingut semàntic, representat per l'identificador del camp del recurs apuntat i la seva URI, un recurs amb un altre.

El seu format és molt simple i consta només dels dos paràmetres exposats a la taula 5.37.

Taula 5.37: Paràmetres del recurs Referència al Recurs

Paràmetre	Format de Dades	Descripció
resourceId	string	Identificador del recurs referenciat. Utilitzat com una extensió de l'atribut quan no es pot resoldre el recurs.
resourceId	anyURI	La URI del recurs apuntat.

5.9.8 El recurs Usuari (User)

El recurs Usuari conté tota la informació relacionada a l'usuari que es troba identificat amb l'aplicació FamilySearch.

Aquesta informació pot resultar d'interès pels frontals de les aplicacions en cas de desitjar realitzar alguna mena de personalització del contingut o inclòs realitzar temes d'analítica web. També pot ser utilitzada per accedir a l'arbre genealògic de l'usuari o al recurs de la seva persona a l'arbre familiar.

Aquest recurs, com molts altres que ja hem exposat, també hereta els paràmetres dels recursos Enllaços Hypermedia i Dades Extensibles, que poden trobar-se en aquest mateix apartat de la memòria i té, per paràmetres propis, els que es mostren a la taula 5.38.

Taula 5.38: Paràmetres del recurs Usuari

Paràmetre	Format de Dades	Descripció
alternateEmail	string	Adreça de correu alternativa.
birthDate	string	Data de naixement.
contactName	string	Nom de contacte.
country	string	País d'origen.
displayName	string	Nom a mostrar.
email	string	Correu electrònic de l'usuari.
familyName	string	Cognom de l'usuari.
fullName	string	Nom complet de l'usuari.
gender	string	Gènere de l'usuari.
givenName	string	Nom de l'usuari.
helperAccessPin	string	Pin d'accés en cas de necessitat d'ajuda.
mailingAddress	string	Direcció postal.
mobilePhoneNumber	string	Número de telèfon mòbil.
personId	string	Identificador de l'usuari.
phoneNumber	string	Número de telèfon.
preferredLanguage	string	Llenguatge preferit.
treeUserId	string	Identificador de l'usuari a l'arbre de FamilySearch.

5.9.9 El recurs Canvi (Change)

Un altre recurs transversal, per alguns dels recursos més importants de l'API de FamilySearch, és el recurs Canvi.

Quan un usuari realitza alguna modificació de qualsevol mena, ja sigui sobre la informació d'una persona o sobre una relació de parella o parental, aquest canvi queda enregistrat per diversos motius.

El primer, poder veure com les dades s'han anat modificat al llarg del temps i veure'n la progressió. El segon, poder recuperar un estat anterior en cas d'error o problema en el sistema.

El recurs Canvi està format pels paràmetres mostrats a la taula 5.39.

Taula 5.39: Paràmetres del recurs Canvi

Paràmetre	Format de Dades	Descripció
-----------	-----------------	------------

object-Modifier	[changeObject-Modifier]	Un modificador opcional per l'objecte al qual afecta l'operació de canvi. Per exemple en cas que el canvi sigui sobre un Esdeveniment es podria utilitzar aquest paràmetre per indicar que era un esdeveniment sobre una relació de parella o parental.
operation	[changeOperation]	L'operació que s'ha realitzat.
reason	string	La raó del canvi.
objecType	[changeObjecType]	El tipus d'objecte sobre el qual s'aplica l'operació.
original	[ResourceReference]	El subjecte o recurs tal com es troava just abans de realitzar el canvi.
parent	[ResourceReference]	El canvi del recurs pare que al efectuar-se ha causat la pèrdua o introducció d'aquest canvi.
removed	[ResourceReference]	El subjecte o recurs que representa els valors esborrats que existien abans del canvi.
resulting	[ResourceReference]	El subjecte o recurs tal com ha quedat després del canvi.

L'enumeració changeObjectModifier

L'enumeració changeObjectModifier segueix l'estruccura de definició GEDCOMX. Com a tal, els valors possibles per l'enumeració segueixen la pauta:
<http://gedcomx.org/+ 'changeObjectModifier'>

La taula 5.40 mostra els possible valors de l'enumeració changeOperation.

Taula 5.40: Valors possibles per l'enumeració changeObjectModifier

Person	Couple	ChildAndParentsRelationship
--------	--------	-----------------------------

L'enumeració changeOperation

L'enumeració changeOperation segueix l'estruccura de definició GEDCOMX. Com a tal, els valors possibles per l'enumeració segueixen la pauta:
<http://gedcomx.org/+ 'changeOperation'>

La taula 5.41 mostra els possibles valors de l'enumeració changeOperation.

Taula 5.41: Valors possibles per l'enumeració changeOperation

Create	Read	Update	Delete
Merge	Unmerge	Restore	

L'enumeració changeObjectType

L'enumeració changeObjectType segueix l'estructura de definició GEDCOMX. Com a tal, els valors possibles per l'enumeració segueixen la pauta:
<http://gedcomx.org/> + 'changeObjectType'

La taula 5.42 mostra els possibles valors de l'enumeració changeObjectType.

Taula 5.42: Valors possibles per l'enumeració changeObjectType

Person	Couple	ChildAndParents-Relationship	Man
Woman	Father	Mother	Child
SourceReference	DiscussionReference	EvidenceReference	Affiliation
Annulment	Burial	Death	Naturalization
BarMitzvah	Christening	Divorce	Occupation
BatMitzvah	Cremation	Marriage	Religion
Birth	CommonLawMarriage	MilitaryService	Residence
Stillbirth	Fact	Caste	Clan
NationalId	Ethnicity	Name	MarriedName
Nationality	Gender	BirthName	Nickname
PhysicalDescription	Note	AlsoKnownAs	DiedBeforeEight
TribeName	LivingStatus	Confirmation	Sealing
BirthOrder	TitleOfNobility	Initiatory	NotAMatch
LifeSketch	Baptism	Endowment	

5.9.10 El recurs Agent (Agent)

El recurs agent representa a una persona, organització o col·lectiu. En la recerca genealògica, un Agent, generalment pren el rol de contribuïdor.

La gran majoria dels paràmetres de contribuïdors, que ens hem trobat fins ara en els diferents recursos, apunten a una instància de recurs Agent.

Aquest recurs conté la informació específica que detallarem a la taula 5.43 així com els paràmetres heretats dels recursos Enllaços Hypermedia i Dades Extensibles.

Taula 5.43: Paràmetres del recurs Agent

Paràmetre	Format de Dades	Descripció
accounts	[array of Online-Account]	Els diferents comptes d'usuari que formen part de l'organització. Només la d'un usuari si es tracta d'un individu particular.

5.10. CAMINS D'ACCÉS A L'ARBRE FAMILIAR I OPERACIONS DE CERCA91

addresses	[array of Address]	Les adreces postals que pertanyen a la persona o membres de l'organització.
emails	[array of ResourceReference]	Els correus electrònics que pertanyen a l'individu o col·lectiu.
homepage	[ResourceReference]	La pàgina principal de l'organització. (Fora de FamilySearch)
identifiers	Identifier	Llista d'identificadors del recurs.
names	TextValue	La llista dels noms pels quals es coneix a l'agent.
openid	[ResourceReference]	L'identificador públic de la persona o organització.
phones	[ResourceReference]	Els números de telèfon que pertanyen a l'individu o col·lectiu.

5.10 Camins d'accés a l'arbre familiar i operacions de cerca

L'accés a les dades contingudes per l'API de FamilySearch es pot realitzar de maneres diferents segons la informació inicial coneguda en el moment d'iniciar la cerca.

Tot procés de cerca es podria dividir en dues fases principals, on la segona, realment podria ser dividida en diverses opcions diferents. Aquestes dues fases consisteixen en:

1. Accedir al sistema de FamilySearch mitjançant un usuari i contrasenya.
2. Accedir a les dades de forma directa o indirecta segons la informació coneuguda.

La imatge 5.8 ofereix una vista preliminar de les diferents opcions disponibles per tal d'accendir a les dades de FamilySearch relacionades amb les persones. En els següents apartats, s'exposarà el comportament dels mètodes directes i indirectes d'accés a les dades.

5.10.1 L'accés directe a les dades

L'accés directe a les dades es pot realitzar si es coneix l'identificador del recurs que vol ser consultat.

Per exemple, si es coneix el PID de la persona a consultar, es pot codificar des de les aplicacions el diferent conjunt d'URIs per manipular els recursos i evitar així la necessitat de passar pel sistema. D'aquesta forma, es pot doncs accedir a la informació de la persona, memòries, discussions, fills, pares, parelles, avantpassats i

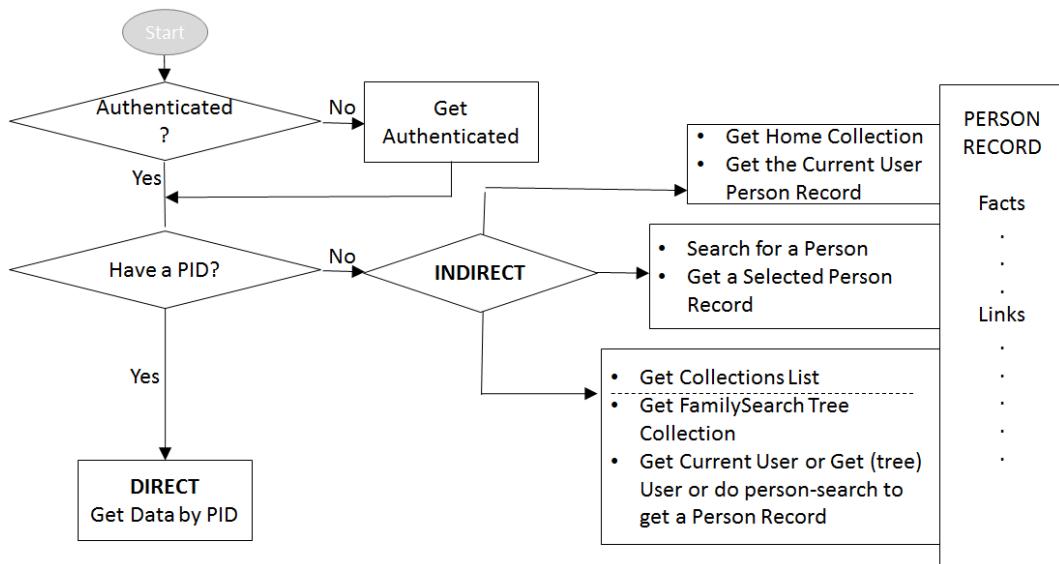


Figura 5.8: Sistemes d'accés a les dades de l'arbre familiar

descendència; sempre i quan, es coneagi tota la informació necessària per codificar les diferents URIs d'accés.

Un exemple d'ús, podria ser per exemple, donat un identificador de persona conegut, accedir directament a les seves memòries mitjançant la codificació de la URI i evitar, així, haver de passar per l'arbre familiar.

5.10.2 L'accés indirecte a les dades

A pesar que poder accedir directament a les dades, pot semblar un procés ideal, el més normal és que ens trobem en la situació de necessitar realitzar un accés indirecte a les dades. En altres paraules, accedir en primera instància als recursos del sistema i obtenir així els diferents enllaços que ens condueixin cap a les operacions desitjades.

Per descobrir l'URI del recurs exacte que vol ser consultat, existeixen principalment tres opcions diferents:

- Llegir la col·lecció Arrel, que representa la col·lecció que conté totes les dades de FamilySearch i seguir-la per tal d'obtenir l'usuari identificat. Un cop es disposa de la persona relacionada a l'usuari, es pot accedir a qualsevol altre recurs mitjançant els enllaços hypermedia de la resposta i simular, de forma dinàmica, el mateix que podríem haver realitzat mitjançant l'accés directe.
- Llegir la llista de col·leccions disponibles per tal d'accendir a l'arbre familiar o accedir a ell de forma directa. Un cop es disposa de l'arbre familiar es pot llegir l'usuari actual o la persona de l'usuari i d'aquí procedir com es desitgi. Una alternativa és accedir mitjançant els enllaços o plantilles URL, als recursos

5.10. CAMINS D'ACCÉS A L'ARBRE FAMILIAR I OPERACIONS DE CERCA93

generals de Memòries, Discussions, Relacions, etcètera i consultar la informació general d'aquests recursos mitjançant els seus identificadors retornats.

- La tercera opció, i probablement la més comuna, passa per la realització d'una cerca contra la base de dades de FamilySearch per tal d'obtenir una Persona o Localització. Un cop s'obté el resultat de la cerca, es pot navegar a qualsevol de les peces d'informació relacionades als recursos consultats, mitjançant les URI i enllaços hypermedia de les respuestes.

El procés descrit en el tercer punt de les opcions de cerca indirecta, serà el més comú de cara a accedir a la informació, ja que les altres opcions no permeten gaire filtratge de les dades, a menys que coneuem la col·lecció específica sobre la qual volem realitzar una investigació genealògica o només vulguem consultar la informació de l'arbre familiar, de l'usuari identificat.

Com hem comentat doncs, són quatre les principals opcions d'entrada que permeten obtenir persones concretes de l'arbre de forma indirecta i començar a navegar, des de la resposta d'aquestes, per les dades genealògiques.

En els següents apartats, s'explicarà en més detall com poden ser configurades cada una d'aquestes operacions.

5.10.3 Lectura de l'usuari identificat

Aquesta operació pot ser realitzada de forma directe, accedint a la URI:

/platform/users/current

La crida retorna la informació específica del recurs Usuari descrita en els apartats anteriors d'aquesta memòria. Des d'aquest, es pot accedir a la persona de l'arbre familiar que representa a l'usuari i des d'aquesta, a qualsevol altre peça d'informació genealògica relacionada amb la persona.

5.10.4 Lectura de la persona relacionada a l'usuari identificat

Aquesta operació és realitzable a través de la URI:

/platform/tree/current-person

L'operació en qüestió retorna el recurs Persona de l'usuari identificat, amb tota la informació descrita en els apartats anteriors de la memòria. Des d'aquest, es pot navegar a qualsevol altre recurs o dada genealògica relacionada.

Aquesta forma d'accés representa un pas menys que l'exposada en l'apartat anterior, si sabem d'entrada que volem accedir a l'arbre familiar. En cas que també volguéssim alguna dada del recurs Usuari, caldria entrar per la funcionalitat descrita en l'apartat anterior.

5.10.5 Cerca de Persones a l'arbre familiar

L'operació cerca de persones, a l'arbre familiar, és sense cap dubte la més interessant de totes les funcionalitats que ofereix l'API.

Aquesta operació es realitza mitjançant una petició a la URI /platform/tree/search, que a més a més, pot ser configurada i personalitzada mitjançant la inclusió de diferents paràmetres.

Un cop finalitzada la petició, aquesta retorna el conjunt de persones de l'arbre familiar, que compleixen amb les condicions imposades en la cerca. L'usuari, pot començar a navegar per la resposta, accedint a aquelles persones que li resultin de més interès i a tots els altres recursos disponibles relacionats amb aquestes.

Es pot entendre la funcionalitat de cerca a l'arbre familiar, com una porta a totes les dades emmagatzemades per FamilySearch.

La cerca pot ser controlada mitjançant tres paràmetres principals, un dels quals, accepta molts paràmetres secundaris. Els paràmetres principals són descrits a la taula 5.44.

Taula 5.44: Variables principals per la cerca de persones

Variable	Descripció
start	Indica l'índex del primer resultat desitjat de cara a la pàgina de resultats que serà retornada. Les persones són retornades en blocs i el nombre de persones en cada bloc depèn del paràmetre count. Així doncs aquest paràmetre ens permet fixar per quina pàgina de la resposta volem explorar els resultats.
count	El nombre de resultats de cerca desitjats per pàgina.
q	Aquest paràmetre és el que descriu les condicions de cerca. El nom del paràmetre i el valor desitjat per aquest són separats mitjançant dos punts mentre que els diferents paràmetres de cerca són separats per un espai en blanc. També es pot indicar que no busquem els valors exactes d'un paràmetre afegint el caràcter '~' al final d'una parella clau:valor. D'aquesta forma s'inclouran a la resposta persones amb valors similars pel camp senyalat i no només aquelles amb coincidències exactes.

El paràmetre q , descrit en la taula 5.44, accepta com a paràmetres vàlids els camps exposats a la taula 5.45. L'etiqueta {relation}, dels camps d'aquesta taula, pot ser reemplaçada per qualsevol dels següents valors: father, mother, spouse (pare, mare, parella).

Taula 5.45: Paràmetres acceptats per la variable q

5.10. CAMINS D'ACCÉS A L'ARBRE FAMILIAR I OPERACIONS DE CERCA95

Variable	Descripció
name	El nom complet de la persona cercada.
givenName	El nom de la persona cercada.
surname	El cognom o nom de família de la persona cercada.
gender	El gènere de la persona cercada.
birthDate	La data en què la persona va néixer. S'accepten rangs de dates mitjançant el caràcter '-' entremig.
birthPlace	La localització en què la persona va néixer.
christeningDate	La data en què la persona va ser batejada. S'accepten rangs de dates mitjançant el caràcter '-' entremig.
christeningPlace	La localització en què la persona va ser batejada.
deathDate	La data en què la persona va morir. S'accepten rangs de dates mitjançant el caràcter '-' entremig.
deathPlace	La localització en què la persona va morir.
burialDate	La data en què la persona va ser enterrada. S'accepten rangs de dates mitjançant el caràcter '-' entremig.
burialPlace	La localització en què la persona va ser enterrada.
marriageDate	La data en què la persona es va casar. S'accepten rangs de dates mitjançant el caràcter '-' entremig.
marriagePlace	La localització en què la persona es va casar.
{relation}Name	El nom complet de la {relació} de la persona cercada.
{relation}GivenName	El nom de la {relació} de la persona cercada.
{relation}Surname	El cognom de la {relació} de la persona cercada.
{relation}BirthDate	La data de naixement de la {relació} de la persona cercada. S'accepten rangs de dates mitjançant el caràcter '-' entremig.
{relation}BirthPlace	El lloc de naixement de la {relació} de la persona cercada.
{relation}DeathDate	La data de defunció de la {relació} de la persona cercada. S'accepten rangs de dates mitjançant el caràcter '-' entremig.
{relation}DeathPlace	El lloc de defunció de la {relació} de la persona cercada.
{relation}MarriageDate	La data de casament de la {relació} de la persona cercada. S'accepten rangs de dates mitjançant el caràcter '-' entremig.
{relation}MarriagePlace	El lloc de casament de la {relació} de la persona cercada.

Cerca de persones duplicades

Una característica interessant de les respostes en la cerca de persones és que per cada persona de la resposta, es pot accedir al conjunt de persones de l'arbre familiar,

que amb alta probabilitat, poden representar a la mateixa persona consultada. En altres paraules, persones que podrien tractar-se de possibles duplicats.

Aquesta funcionalitat de cerca de persones duplicades, també es troba accessible mitjançant una URI pròpia, si es coneix l'identificador personal de la persona sobre la qual es vol buscar els possibles duplicats.

5.10.6 Cerca de localitzacions

La cerca de localitzacions és la segona opció de cerca massiva que permet l'API de FamilySearch.

Aquesta operació cobra especial interès quan es necessita consultar informació extra sobre una localització, més enllà de la informació bàsica relacionada a certs recursos de l'arbre familiar, o es vol obtenir informació sobre totes les localitzacions que compleixen amb certs criteris. Per exemple, totes les localitzacions dins d'una jurisdicció específica.

Així doncs, la cerca de localitzacions permet relacionar o interpretar, el nom d'una localització, mitjançant una descripció estandarditzada a la URI:

/platform/places/search

De la mateixa forma que en la cerca de persones, la cerca de localitzacions pot ser configurada mitjançant tres paràmetres principals, on un d'aquests accepta diversos paràmetres secundaris. Els paràmetres principals són descrits a la taula 5.46.

Taula 5.46: Variables principals per la cerca de localitzacions

Variable	Descripció
start	Indica l'índex del primer resultat desitjat de cara a la pàgina de resultats que serà retornada. Les localitzacions són retornades en blocs i el nombre de localitzacions en cada bloc depèn del paràmetre count. Així doncs aquest paràmetre ens permet fixar per quina pàgina de la resposta volem explorar els resultats.
count	El nombre de resultats de cerca desitjats per pàgina.
q	Aquest paràmetre és el que descriu les condicions de cerca. El nom del paràmetre i el valor desitjat per aquest són separats mitjançant dos punts mentre que els diferents paràmetres de cerca són separats per un espai en blanc. També es pot indicar que no busquem els valors exactes d'un paràmetre afegint el caràcter '~' al final d'una parella clau:valor. D'aquesta forma s'inclouran a la resposta persones amb valors similars pel camp senyalat i no només aquelles amb coincidències exactes.

En aquesta operació, el paràmetre *q*, accepta com a vàlids el conjunt de paràme-

5.10. CAMINS D'ACCÉS A L'ARBRE FAMILIAR I OPERACIONS DE CERCA97

tres especificats a la taula 5.47.

Taula 5.47: Paràmetres acceptats per la variable q

Variable	Descripció
name	El nom de la localització a cercar. Aquest paràmetre és subjecte a interpretació per part del sistema i ordena els resultats segons criteri propi. Permet la utilització dels caràcters '?' i '*' de les expressions regulars però no en primera posició. En cas de voler utilitzar els caràcters cal escapar-los.
partialName	El nom parcial d'una localització. Aquest paràmetre s'utilitza per cercar la part parcial introduïda en el conjunt de localitzacions. Generalment s'utilitza en els casos en què es coneix el tipus de localització que es vol cercar. En cas de ser introduït el paràmetre nom és ignorat.
date	La data o rang de dates. Suporta els operadors '+' i '-' al principi de la data o interval per incloure només resultats que es trobin per davant o darrere de l'any indicat.
typeId	L'identificador d'una localització. Permet utilitzar els operadors '+' i '-' per obtenir només resultats que incloguin o excloguin els resultats de l'identificador especificat.
typeGroupId	El tipus de localització. Per exemple país ciutat etcètera. Permet l'ús dels operadors '+' i '-' per incloure o excloure les localitzacions que compleixin els requisits de cerca.
parentId	Paràmetre per indicar si es coneix la jurisdicció del pare de la localització. Permet l'ús dels caràcters '+' i '-' per modificar el set de resultats retornat.
latitude	La latitud del centroide que es vol cercar.
longitude	La longitud del centroide que es vol cercar.
distance	La distància màxima al centroide a la que es vol cercar. Les unitats poden ser especificades en quilòmetres o milles.

Secció 6

Valoració final sobre la potencialitat de l'API

6.1 Introducció

Aquesta secció de la memòria pretén cobrir, des del meu punt de vista personal basat tant en el coneixement adquirit mitjançant l'estudi teòric de l'API, com en les petites pinzellades tècniques que s'han pogut aprendre durant la implementació dels exemples, el potencial que emmascara aquesta API de cara a generar propostes de projecte per futurs estudiants.

En conseqüència, a pesar de la localització dins de la memòria d'aquest apartat, aquest és redactat després d'adquirir el coneixement pràctic bàsic, gràcies a la implementació dels exemples, que seran exposats en les següents seccions de la memòria.

Per comprendre la potencialitat global d'aquesta API, cal dividir-ne l'estudi en diferents blocs o peces. Intentar emetre un judici de valor global, sense sintetitzar-ne primer diferents oportunitats i complicacions de conceptes més específics, no aconseguiria transmetre la profunditat i complexitat de l'abast d'aquest pregunta.

6.2 Distribució geogràfica de les dades

Per comprendre la potencialitat global d'aquesta API, cal dividir-ne l'estudi en diferents blocs o peces. Intentar emetre un judici de valor global, sense sintetitzar-ne primer diferents oportunitats i complicacions de conceptes més específics, no aconseguiria transmetre la profunditat i complexitat de l'abast d'aquest pregunta.

FamilySearch, pot presumir de tenir una de les bases de dades d'informació genealògica oberta al públic més gran del món, si no la més gran, amb més de quatre bilions de registres.

A pesar que el nombre de 4 bilions pugui semblar molt elevat, si el comparem amb

els més de 33 bilions de persones que han nascut aproximadament des del 1200 fins a l'any 2011 (agafant així, una dada de referència pública que s'ajusti més o menys al període de temps sobre el que FamilySearch disposa d'informació), ens adonem del fet que disposem d'una mostra acceptable, però lluny de suposar una representació real.

Cal també tenir en compte que els 4 bilions de registres emmagatzemats a FamilySearch no es troben repartits de forma proporcional sobre les diferents regions o països, sinó que l'organització, de forma evident, disposa més dades en aquells indrets en què històricament ha tingut més presència o facilitat d'accés a dades.

D'aquesta forma, els Estats Units d'Amèrica, amb 991 col·leccions de dades diferents, ofereix la informació de 2,5 bilions de registres daten entre els anys 1500 i 2015. En altres paraules, un 62% del volum total de dades.

De forma paral·lela, i per oferir una escala diferent, Espanya, amb 45 col·leccions, ofereix la informació de 24 milions de registres compresos entre els anys 1251 i 2013. Per tant, resulta fàcil observar que la dispersió de les dades i volum difereix molt segons la regió que vol ser consultada.

Cal doncs, tenir en compte aquesta limitació de cara a proposar o realitzar certa mena de projectes, com poden ser per exemple, els estudis estadístics d'aspectes demogràfics. Com a recomanació, s'encoratja als estudiants a utilitzar aquelles regions, com els Estats Units, més plegades de registres, de cara a funcionalitats generals.

6.3 Dades contemporànies

Un dels inconvenients de les dades genealògiques és que aquestes generalment es troben subjectes a lleis de protecció, durant períodes de temps prolongats, abans de poder fer-se públiques. És més, en casos especials com els que hem esmentat en les primeres seccions de la memòria, aquestes poden inclús no arribar mai al domini públic.

Aquest aspecte implica que el valor percebut, del conjunt de dades disponible a través de FamilySearch, sigui més elevat en projectes enfocats al passat, que no pas estudis més contemporanis.

La ironia en aquest punt de la memòria és que a menys que les legislacions canviïn per complet, l'affirmació de què sempre es disposarà de menys dades contemporànies, seguirà sent aplicable independentment dels anys que passin.

Un altre fet que pot impactar a la quantitat de registres contemporanis disponibles és la quantitat d'afiliacions a l'Església de Jesucrist dels Sants dels Darrers Dies i és que cal no oblidar, que aquesta organització, representa al principal benefactor de FamilySearch i per tant, principal origen de font de dades.

6.4 Recursos i funcionalitats

El conjunt de recursos utilitzables i les funcionalitats creades al seu voltant, esdevenen un dels punts més favorables de l'API.

El conjunt de paràmetres accessibles relacionats a una persona, o qualsevol altre recurs, resulta immens. Des dels esdeveniments principals relacionats a la seva vida d'una persona, fins a petits detalls com els diferents noms que la persona va rebre al llarg de la seva vida. La informació es troba molt ben estructurada i tots elements relacionats, resulten fàcilment accessibles.

La robustesa de les dades tampoc és cap broma, mitjançant el sistema de canvis es pot desfer qualsevol ús malintencionat o involuntari sobre el conjunt de dades. Clarament, FamilySearch es pren molt seriosament poder garantir la qualitat de les dades.

Per si tot el conjunt de recursos accessible no fos suficient, FamilySearch posa a la disposició dels usuaris una sèrie de funcions de conveniència que permeten, entre altres exemples, cercar persones duplicades, accedir a les ascendències i descendències d'una persona de forma reglada i estructurada i delimitar les cerques per més paràmetres dels que un es podria imaginar.

Tot plegat, el conjunt de recursos, granularitat de la informació i les funcionalitats de fàcil accés, converteixen l'API de FamilySearch en un poderós aliat de cara a la recerca genealògica.

6.5 Naturalesa de l'API

Un dels primers xocs que em vaig emportar quan vaig començar a estudiar més a fons la potencialitat de l'API, va ser perquè fins aquell moment no havia tingut en compte el motiu pel qual aquesta havia estat concebuda.

L'API de FamilySearch neix per ajudar a individus particulars a realitzar recerca genealògica, sobre els seus avantpassats o els d'un tercer. Cal tenir molt present aquesta definició, ja que limita o debilita en gran mesura, els possibles projectes a realitzar.

FamilySearch va dissenyar l'API perquè un usuari pogués realitzar una cerca, de la forma més específica possible, després de la recopilació prèvia d'informació sobre la persona cercada. Per tant, no està pensada per accedir a un gran volum de registres de forma simultània, accedir a les dades per un nivell de granularitat inferior a la del concepte 'persona', ni realitzar moltes peticions consecutives contra la plataforma.

En conseqüència tota aspiració de realitzar projectes de mineria de dades o estudis de baixa granularitat, queden completament descartats, o si més no, condicionats en gran mesura de cara a la implementació tècnica o automatització de tasques.

6.6 Utilització de l'API en el marc d'un PFC

Un últim concepte a destacar és com encaixa la utilització d'aquesta API en el marc d'un projecte final de carrera. És a dir, si deixem de banda la discussió de com és de potent aquesta, resulta factible utilitzar-la de cara a un projecte final de carrera?

Crec que resulta de vital importància respondre a aquesta pregunta de forma independent a la de la potencialitat, per no vincular dos conceptes diferents.

Un projecte final de carrera es desenvolupa generalment en el període de temps equivalent al d'un quadrimestre. Com ja s'ha esmentat en altres seccions de la memòria i encara tornarà a aparèixer més endavant, per tal d'aconseguir accés a les dades de producció de FamilySearch, cal certificar l'aplicació.

Aquest procés, com aquest projecte n'és una clara mostra, pot resultar complicat i ple de complicacions, i encara podria esdevenir més complex si es volgués realitzar una aplicació amb drets d'escriptura a l'arbre genealògic o comercialitzable.

El que volem indicar en aquest apartat és que si la planificació del projecte no és bona, i inclús així, s'incore en un cert risc, existeix la possibilitat de no disposar del temps suficient per implementar, certificar i extreure les conclusions necessàries, sobre les dades de producció.

El fet que existeixi en l'actualitat, la possibilitat de certificar les aplicacions per ús personal i no només comercial, augmenta en gran mesura les possibilitats dels estudiants a aventurar-se en projectes de recerca genealògica.

En certa forma, esdevé probable que la utilització d'aquesta API condicioni l'estructura dels projectes de la mateixa forma que ho ha fet amb aquest. Forçant un clar esforç inicial per acabar la implementació tan aviat com es pugui, per tal de poder experimentar amb les dades de producció, o en el nostre cas, esbrinar de què es tractava exactament aquest procés de certificació.

Això no obstant, s'espera que l'estudi realitzat en aquest projecte representi una facilitació i acceleració considerable de la corba d'aprenentatge pels futurs estudiants, el que els permetria accedir abans a producció.

En resum, si, l'API de FamilySearch és un recurs utilitzable de cara a la realització de projectes finals de carrera, però cal tenir en compte les seves peculiaritats de cara a la planificació i pot resultar més atractiu a aquelles persones que pretenguin tenir clar, el projecte que volen realitzar en profunditat, abans de matricular-lo o inclús començar-lo amb un quadrimestre d'antelació, al que serà matriculat.

6.7 Conclusió

Considerades les limitacions geogràfiques, temporals, estructurals i temporals (en l'àmbit de temps del que es disposa per realitzar un projecte) podria semblar que no queden moltes opcions possibles, de cara a plantejar propostes de projecte, més

enllà de la recerca genealògica bàsica. Malgrat això, no és la meva opinió exacta que aquest en sigui el cas.

Si bé és cert, que cal pensar en propostes de projecte que encaixin dins del marc delimitat per aquestes restriccions, les eines posades a disposició dels usuaris i la quantitat d'informació disponible, permeten la utilització de vies secundàries per tal d'assolir diferents objectius i ajudar així a respondre certes preguntes que aquest projecte no ha tingut temps d'explorar.

En la següent secció de la memòria es podran observar un conjunt de propostes de projecte que encaixen dins del marc descrit en aquesta secció i que pretenen oferir resposta, a certes preguntes més específiques, sobre la potencialitat o possibilitats d'ús d'aquesta.

Secció 7

Llista de propostes de projecte

7.1 Introducció

Aquesta secció recopila un conjunt d'idees que poden servir com a projectes finals de carrera per estudiants de la Facultat d'Informàtica de Barcelona.

L'objectiu de cada una de les propostes no és la de representar un enunci tancat, sinó oferir pistes sobre diferents implementacions possibles que puguin inspirar als estudiants a modificar-les, combinar-les, reduir-les, ampliar-les o crear-ne de noves.

Com es podrà veure, moltes d'aquestes propostes giren al voltant d'estudis històrics o la validació de les dades emmagatzemades en els sistemes de FamilySearch respecte a la realitat. El motiu, és que més enllà de la funcionalitat de cerca, la principal preocupació sobre aquestes dades és amb quin grau d'exactitud representen la realitat que les envolta.

A més a més, recordar que l'API de FamilySearch es troba sempre en constant evolució, i que per tant, és una bona idea revisar la viabilitat de cada proposta abans de decidir, amb total certesa, el projecte que es vol realitzar.

Així doncs, les propostes que s'ofereixen a continuació se centren bastant en la utilització del conjunt de dades que creiem més complet de cara a realitzar projectes finals de carrera amb ell. És per aquest motiu, que moltes de les propostes giren al voltant de les dades dels Estats Units, que recordem, representa el 62% del total de registres accessibles a través de l'API.

La major part de propostes restants, se centren a respondre algunes preguntes específiques sobre l'API que aquest projecte no ha pogut estudiar. Aquestes, prenenen esbrinar el grau de fidelitat dels diferents blocs d'informació emmagatzemats per FamilySearch, respecte a la realitat coneguda.

Finalment, algunes de les idees proposades giren al voltant de la creació de funcionalitats i per tant, l'èxit d'aquestes no depèn tant del conjunt de dades accessible.

7.2 Comparació sobre la popularitat de noms

L'objectiu d'aquesta funcionalitat és comparar la popularitat d'un o més noms, en un període concret del temps i amb la possibilitat de fixar la regió demogràfica a consultar.

La idea principal és que donada la introducció d'un o més noms, el sistema cerqui el nombre d'instàncies de persones nascudes amb el nom especificat, en els deu anys anteriors o posteriors a la data indicada, per la regió geogràfica especificada.

D'aquesta forma, es podria comparar quin dels dos noms ha estat més popular, segons les dades de FamilySearch, any a any.

Al mateix temps, esdevé interessant permetre la cerca d'un sol nom per observar si certs esdeveniments històrics han pogut influenciar el nombre de nadons amb un cert nom. Per exemple, suposa l'elecció d'Obama com a president dels Estats Units, un increment en el nombre de persones nascudes amb aquest nom durant els següents anys?

La segona raó de ser de l'eina és ajudar a decidir, per exemple, el nom dels fills d'una persona, comparant, d'aquesta forma, la popularitat actual dels noms que s'estiguin avaluant.

A continuació llistem diferents possibilitats d'extensió:

- Ampliar la cerca a diferents països, on per cada país, el nom introduït serà localitzat. Per exemple, si l'usuari introduceix Alexander, la cerca a Espanya fos realitzada amb el nom d'Alejandro o Alex. En cas de no trobar aquesta base de dades, sempre es podria crear una taula manual d'exemple, amb unes quantes llengües i noms i utilitzar-la.
- Comparar instàncies de noms a Catalunya extrets de les dades de FamilySearch, amb la comparació real extreta del institut nacional d'estadística.

7.3 Portal de cerca localitzat al Català

En aquesta memòria ja hem parlat de la funcionalitat de localització habilitada per part de FamilySearch i encara que aquesta suporta la localització a la llengua espanyola, no ho fa per la catalana.

La idea d'aquesta funcionalitat és oferir a l'usuari un portal de cerca en català sobre les dades de FamilySearch, que no només faciliti la comprensió de la cerca a aquelles persones que vulguin utilitzar el català, sinó que també localitzi, en la mesura que sigui possible, la resposta retornada per l'API.

Per exemple, es podria localitzar la informació relativa a l'estat actual d'una persona: living o deceased, que podria ser mostrada com a viva o difunta. De la mateixa forma, es pot localitzar tot el contingut de la resposta, des del nom dels camps d'informació, al contingut d'aquests en algunes situacions.

Alguns exemples de possibles extensions pel projecte són:

- Restringir la cerca del portal a només Catalunya, amb la possibilitat de desactivar la funció. A més a més, oferir ajuda de refinació a la cerca, de cara a introduir les diferents províncies o ciutats, evitant que l'usuari introduceixi valors invàlids.
- Posar-se en contacte amb l'organització FamilySearch per convertir la localització realitzada a la llengua Catalana, en una localització oficial acceptada pel sistema.

7.4 Geolocalització d'un cognom en diferents nivells

Aquesta funcionalitat pretén ampliar l'exemple programat en aquest projecte final de carrera, evolució geogràfica d'un cognom.

L'exemple implementat només permet la visualització d'instàncies d'un cognom al nivell de país. Aquesta nova funcionalitat hauria de permetre, com a mínim, les visualitzacions en l'àmbit de continent i en l'àmbit d'estat o comunitat autònoma. De totes maneres, com més nivells de profunditat diferents poguessin ser utilitzats, millor.

L'usuari haurà de ser capaç de navegar per aquests diferents nivells de forma interactiva mitjançant el mapa (existeixen mapes interactius a disposició dels desenvolupadors) o controls específics.

La dificultat d'aquesta eina, a diferència de la implementada com a exemple, és que o bé farien faltar llençar més peticions contra l'API o realitzar una petició més gran i explorar després els registres d'un a un. Una alternativa, podria ser realitzar les peticions a l'API, després que l'usuari indiqués que vol canviar el nivell mostrat i anar emmagatzemant la informació de forma local.

7.5 La recerca genealògica i l'heràldica

Aquesta funcionalitat vol relacionar les dues ciències que sempre han caminat de la mà, la genealogia i l'heràldica.

El primer objectiu del projecte implicaria identificar quina font de dades podria ser utilitzada per obtenir els diferents escuts d'armes. Probablement, no existeixi cap API en línia de la qual es puguin obtenir i caldrà realitzar una extracció automàtica d'alguna pàgina web o aplicació d'escriptori.

Un cop es disposin dels diferents escuts d'armes, es proposa realitzar una implementació de cerca simple per veure, conjuntament amb els detalls d'una persona, l'escut d'armes del nom de família.

Un cop es disposin dels diferents escuts d'armes, es proposa realitzar una implementació de cerca simple per veure, conjuntament amb els detalls d'una persona,

l'escut d'armes del nom de família.

Una possibilitat d'extensió, que podria ser bastant atractiva, consisteix a implementar una API que serveixi els diferents escuts d'armes i que la nostra aplicació l'utilitzi per complementar les dades de les persones trobades a l'API de FamilySearch.

7.6 Projectes d'indexació

Tot i que aquesta proposta no pretén interactuar directament amb l'API de FamilySearch, volíem realitzar com a mínim una proposta que estigués relacionada amb el procés d'indexació.

Existeixen dos processos d'indexació diferents, els que es realitzen sobre fitxers amb un format específic i els que es basen en la transcripció d'imatges a través del software de FamilySearch. Aquesta proposta de projecte, és en realitat dividida, en dues diferents.

La primera, aconseguir accés a algun registre genealògic local o posar-se amb contacte amb alguna organització que vulgui pujar el contingut de registres amb un format específic, al núvol. Sobre aquest registre, implementar un sistema d'automatització que transcrigui les dades i les prepari per ser enviades a FamilySearch.

La segona possibilitat, és realitzar un programa que interactuï amb les imatges digitalitzades, llegeixi les seccions de la imatge sobre les que s'ha d'extreure la informació i intenti informar a l'usuari, que no omplir automàticament, sobre el contingut dels camps.

La gràcia d'aquest projecte és que es podria realitzar sense preocupar-nos per la certificació de l'aplicació, ja que per indexar registres un només s'ha de declarar com a voluntari i començar a experimentar.

Aquestes dues propostes esdevenen, amb una alta probabilitat, bastant complexes, per aquest motiu, es prega a l'estudiant que realitzi un bon estudi previ sobre l'abast i viabilitat del que vol realitzar abans d'inscriure el projecte.

7.7 La història de l'església mormona a través de FamilySearch

En una de les primeres seccions de la memòria, s'ha exposat per sobre la història de l'església mormona. Com s'ha pogut observar, aquesta ha estat marcada per nombroses expulsions de diferents territoris i conflictes.

L'objectiu d'aquesta funcionalitat seria intentar traçar una relació entre els diferents emplaçaments o seus principals del col·lectiu, en els diferents moments del temps i les dades emmagatzemades per FamilySearch.

7.8. DIVERSITAT GEOGRÀFICA D'UN COGNOM: ELS NOSTRES AVANTPASSATS109

És a dir, destaca FamilySearch per tenir dades sobretot d'aquelles zones en les quals s'ha trobat especialment present en comparació a la resta de localitzacions? Com d'evident és aquesta relació, si és que existeix? Podem deduir aleshores que la mostra de les dades no és representativa sota cap circumstància?

Aquestes són algunes de les preguntes que aquest projecte podria intentar respondre.

7.8 Diversitat geogràfica d'un cognom: Els nostres avantpassats

Aquesta funcionalitat ha estat inspirada per una campanya publicitària creada per l'agència de viatges Momondo, anomenada, *The DNA Journey*. Recomanem la visualització del vídeo¹ per comprendre millor les motivacions darrere d'aquesta funcionalitat.

Totes les persones creiem que som bastant autòctones del lloc on hem nascut, però els estudis d'ADN sobre els nostres gens ens poden deixar molt sorpresos i demostrar que gairebé tothom té una diversitat ètnica i geogràfica important en els seus gens.

Evidentment, no tothom vol pagar per un test d'ADN i la funcionalitat que proposem, tot i que evidentment, no podrà comparar-se amb aquesta opció, pretén intentar informar a les persones que el seu propi cognom apareix en tota mena de països.

L'objectiu d'aquesta funcionalitat, de forma similar a l'exemple implementat sobre l'evolució geogràfica d'un cognom, és trobar una forma eficient d'ofrir a l'usuari un llistat del nombre total o percentatge d'instàncies d'un cognom, per cada país i en diferents èpoques. Podria ser una bona idea, restringir la cerca als primers deu o vint països més grans de cada continent.

El repte del projecte, no és només la consulta de les dades, sinó trobar una forma intel·ligent d'extrapolar l'influència d'un cognom en una regió determinada, en comptes de basar-nos en el total d'instàncies com ha fet l'exemple implementat, el que provoca que el resultat estigui condicionat pel nombre de registres disponibles en cada país.

Queda per tant, a decisió de l'usuari, com realitzar una interpretació de la influència d'un cognom en cada regió i evidentment, sempre de forma aproximada. L'objectiu final és fer palpable la diversitat geogràfica dels nostres possibles avantpassats o relatius.

Aquesta proposta pot ser complicada degut al control sobre el temps d'execució. En cas de no veure viable realitzar una proposta que extregui les dades en temps real, sempre es pot realitzar un estudi específic sobre alguns cognoms d'interès o persones conegudes i mostrar-ne els resultats.

¹<https://www.youtube.com/watch?v=tyaEQEmt5ls>

7.9 Les col·leccions de dades de FamilySearch

L'objectiu d'aquesta aplicació és llegir les diferents col·leccions o fonts de dades de FamilySearch i llistar-ne, de quines regions i sobre quins períodes de temps, contenen registres genealògics.

L'objectiu, és oferir als usuaris un portal d'informació que permeti, mitjançant la introducció d'un país, període de temps o ambdues condicions, visualitzar les diferents col·leccions disponibles i la informació específica d'aquestes.

Per exemple, donada la introducció del país Espanya i el període 1500–2000, la funcionalitat hauria de llistar totes les col·leccions que contenen dades que compleixen aquestes condicions, quants registres totals suposen aquestes, quants d'aquests estan accessibles a través de FamilySearch, quans pendents d'indexar, etcètera, etcètera.

Aquesta funcionalitat pretén respondre a un dels problemes principals de l'API i és la falta d'informació sobre la informació emmagatzemada. D'aquesta forma, mitjançant un petit anàlisi previ, podríem esbrinar si la informació que desitgem és probable que existeixi o no, sense perdre el temps en l'exploració manual de registres.

7.10 FamilySearch i la segona guerra mundial: Natalitat i Defuncions

Durant el transcurs del temps han succeït un gran nombre d'esdeveniments que han afectat a la població mundial de diferents formes. Un dels conflictes que ha causat més repertori, ha estat la segona guerra mundial.

L'objectiu d'aquesta proposta de projecte és observar l'impacte que va tenir aquest esdeveniment en l'índex de natalitat i defuncions, dels diferents països implicats, a través dels anys del conflicte. Es recomana ampliar la finestra de temps estudiat més enllà dels anys del conflicte per observar quins eren els valors normals, previs i posteriors, a l'esdeveniment.

L'estudiant haurà de trobar la forma d'escalar les dades de cada país segons el volum de registres disponibles.

Una altra tasca que pot realitzar l'estudiant, és comparar els valors obtinguts a través de FamilySearch amb les dades oficials de la segona guerra mundial i respondre preguntes de l'estil: Es corresponen els països amb un increment de defuncions més elevat amb els que van patir més durant la segona guerra mundial?

7.11 FamilySearch i la segona guerra mundial: Increment en els casaments

Un estudi realitzat per Randal S. Olson², demostra que quan la segona guerra mundial va esclatar als Estats Units, es va registrar un increment enorme del nombre de casaments al llarg del país., demostra que quan la segona guerra mundial va esclatar als Estats Units, es va registrar un increment enorme del nombre de casaments al llarg del país.

L'objectiu d'aquesta funcionalitat és intentar replicar aquest estudi pels estats units i estendre'l als països més implicats en la segona guerra mundial, per veure si l'afecte va ser el mateix en diferents regions del globus terraqui.

7.12 FamilySearch i la segona guerra mundial: La llista de Schindler

Aquesta proposta pretén realitzar un estudi sobre un dels col·lectius que es va veure més afectat durant la segona guerra mundial, els jueus.

Aquesta proposta de projecte ofereix a l'estudiant parcejar els cognoms coneguts d'aquelles persones que van formar la llista de Schindler, i realitzar un estudi d'aquests sobre les dades de FamilySearch.

El projecte pot intentar respondre preguntes com: Van disminuir en gran quantitat el nombre de registres amb els cognoms indicats? Van realitzar aquestes persones un procés d'emigració a diferents indrets del món? Quina mena de documents registrats han quedat d'aquestes persones?

La proposta que ens ocupa se'm va acudir quan vaig descobrir, en les fases prèvies del projecte, el certificat d'emigració de Wladyslaw Szpilman, també conegit, com el pianista de Varsòvia. La imatge 7.1 mostra el registre físic trobat a FamilySearch.

7.13 La gran recessió o altres esdeveniments històrics

L'exemple específic que es proposa per aquesta funcionalitat, s'aprofita del fet que es disposa d'un major nombre de registres pels Estats Units que no per la resta de països. Malgrat això, aquesta pot intentar ser reproduïble mitjançant qualsevol altre esdeveniment històric.

L'objectiu de l'exemple proposat és estudiar els impactes de la gran recessió en la població dels Estats Units a través de les dades de FamilySearch i respondre a la pregunta de si la realitat observada és similar als fets reals.

Es suggereix a l'estudiant que estudiï el nombre de morts i emigracions al voltant

²<http://www.randalolson.com/2015/06/15/144-years-of-marriage-and-divorce-in-1-chart/>



Figura 7.1: Registre d'emigració de *Wladyslaw Szpilman*

d'aquest període com a punt de partida.

També volem aprofitar aquesta proposta per suggerir als estudiants que qualsevol fet històric, amb un cert impacte, pot intentar ser estudiat a través d'aquesta API. Per aquest motiu, es convida als estudiants a plantejar els seus propis casos d'estudi.

7.14 Comparacions amb els amics o seguidors de Facebook i Twitter

L'objectiu d'aquesta funcionalitat és proporcionar a l'usuari, mitjançant les API de FamilySearch, Facebook o Twitter, un conjunt d'eines per tal de comparar les seves dades amb les dels seus amics.

El nombre de comparacions o eines de cerca a FamilySearch que podrien ser utilitzades és il·limitada, però per donar alguna idea als estudiants del que tenim al cap, en citem algunes a continuació:

- Quin dels nostres amics o seguidors té el nom o cognom més popular en el país on viuen o arreu del món?
- Quin dels nostres amics és més probable que tingui ascendència als estats units (per instàncies del cognom en el país)?
- Quina diversitat cultural representen les teves amistats?

7.15. COMPARACIÓ DE DADES GENEALÒGIQUES REALS AMB FAMILYSEARCH113

- Mostrar l'esperança de vida mitjana de les persones enregistrades a FamilySearch amb el mateix nom que algun dels nostres amics i comparacions amb aquestes dades.

Les possibilitats són realment il·limitades i a més a més, l'aplicació podria permetre accions com, seleccionar només els amics que ens interessin per certes consultes, realitzar una cerca sobre tots els nostres amics per localitzar de forma fàcil als que ens interessin, etcètera.

7.15 Comparació de dades genealògiques reals amb FamilySearch

Aquesta proposta de projecte pretén validar, en certa forma, com de bé representen les dades de FamilySearch la realitat d'un país o països a través de diferents èpoques, o per una època determinada.

Aquesta proposta podria ser dividida en molts projectes diferents, un per cada país, concepte o època que l'estudiant vulgui explorar. Per ajudar a comprendre als estudiants al que ens estem referint, a continuació citem una sèrie d'exemples:

- Comparació de mortalitat per infants: Global, per continents, països, èpoques, etcètera.
- Mitja de fills per família: Global, per continents, països, èpoques, etcètera.
- Edat mitjana de totes les persones enregistrades, en un moment i localització determinades? Evolució d'aquest indicador al llarg del temps.
- Proporció d'homes i dones: Global, per continents, països, èpoques, etcètera.
- Esperança de vida per les dones i homes: Global, per continents, països, èpoques, etcètera.

Aquesta proposta no pretén que l'estudiant abordi tots els conceptes diferents que es pugui imaginar, però si en aquells que cregui que poden tenir un valor més elevat de cara a comparar regions i èpoques.

Segons els factors a estudiar, la complexitat de com hauran de ser processades les dades canvia, i per tant, caldrà tenir-ho en compte a l'hora de definir l'abast del projecte.

Es recomana també als estudiants que cerquin estudis estadístics sobre la població del món, per poder inspirar-se de cara al plantejament de propostes de projecte. A nosaltres ens va ajudar la presentació a les conferències Ted de Kim Preshoff³.

³<https://www.youtube.com/watch?v=RLmKfXwWQtE>

7.16 Estudi de profunditat dels arbres familiars

Aquesta funcionalitat es basa en l'estudi de les genealogies disponibles a través de FamilySearch.

Es planteja a l'usuari un estudi de la profunditat d'aquestes, quantes generacions diferents estan emmagatzemades de mitjana, com de completes soLEN ESTAR les dues bandes de l'arbre familiar, etcètera.

Un segon conjunt de dades que l'estudi pot intentar respondre és quins són els períodes de temps en els que era més probable mantenir un arbre genealògic. S'està perdent la tradició? Ha anat en augment durant els últims anys? Quina mena d'informació és més probable que es trobi disponible?

Aquests són alguns dels exemples que plantegem als futurs estudiants.

7.17 Algoritme de marcatge de duplicats

FamilySearch té una funció que permet, donada una persona, aconseguir les persones de l'arbre que tenen una alta probabilitat de ser un duplicat.

L'objectiu d'aquesta aplicació seria realitzar un algoritme, que donada una persona, estudie les persones marcades com a candidates a ser un duplicat i avalués si aquestes marques tenen pinta de ser correctes o no.

L'estudiant podria cercar i comparar segons la diferent informació disponible de cada persona i emetre una conclusió final de diferents nivells, com per exemple:

- Insuficient informació per conoure.
- Duplicat descartat per inconsistència en els naixements.
- Duplicat real amb coincidències de dates de naixement.
- Etcètera.

Un altre aspecte que el projecte podria intentar atacar és comparar la fiabilitat d'aquest algoritme amb la identificació de duplicats per part de FamilySearch.

Aquest projecte pot resultar bastant complex, i a priori, es desconeix la precisió o condicions sota les quals FamilySearch marca a una persona com a duplicada. Per tant, es recomana realitzar un bon estudi previ d'aquests conceptes abans d'embarcar-se en el projecte.

Secció 8

Estudi tècnic de l'aplicació web

En aquesta secció, s'introduirà quina aplicació ha decidit implementar-se per implementar els exemples d'interacció amb l'API de FamilySearch, les diferents tecnologies que han estat estudiades i utilitzades pel desenvolupament, en tots els àmbits.

8.1 Decisió del tipus d'aplicació a implementar

Per poder començar a estudiar el conjunt de tecnologies que el projecte requeriria, primer necessitavem saber quina mena d'aplicació seria implementada.

Des del principi teníem bastant clar, que de disposar de l'oportunitat, intentaríem implementar una pàgina web. Després de realitzar l'estudi inicial de l'API i de les diferents opcions de desenvolupament possibles, crear una pàgina web semblava l'opció més flexible i factible i per tant, vam decidir tirar per aquest camí.

La pàgina web representava l'opció més flexible i factible, ja que els protocols per integrar-se amb APIs es troben bastant desenvolupats i a més a més, oferia l'oportunitat d'utilitzar els SDK oficials, així com diverses eines de desenvolupament que facilitarien les tasques de creació i interacció amb usuaris.

Per tots aquests motius, a part de la motivació personal d'assolir les habilitats necessàries per desenvolupar una aplicació web, escollir aquesta opció tenia tot el sentit del món.

Així doncs, un cop decidit que s'implementaria una pàgina web, calia fer un reconeixement de les diferents tecnologies disponibles en el mercat i escollir-ne les més adequades, que poguessin treballar de forma conjunta.

Les tecnologies estudiades poden ser dividides en tres grans blocs: Les tecnologies per la creació d'aplicacions web, tecnologies de desenvolupament i les tecnologies de desplegament.

Les tecnologies per la creació d'aplicacions web, representen aquell conjunt de llenguatges, arquitectures i frameworks, que serien utilitzats de cara a la construc-

ció de la pàgina web. Amb altres paraules, el conjunt d'eines i llenguatges que s'utilitzaria per implementar el servidor, les comunicacions entre el servidor i l'API de FamilySearch i el frontal o visual de l'aplicació.

Les tecnologies de desenvolupament, representen el conjunt de tecnologies i eines específiques que han estat utilitzades per assistir i facilitar la creació de l'aplicació web.

Finalment, les tecnologies de desplegament, fan referència a l'allotjament web escollit i les tecnologies necessàries per poder completar el desplegament de l'aplicació al núvol.

8.2 Tecnologies i patrons utilitzats per l'aplicació web

En aquest apartat de la memòria es descriuran el conjunt de patrons de disseny web i tecnologies, que han estat utilitzades per tal que l'aplicació web funcioni i compleixi amb tots els requisits que ens havíem marcat per ella.

8.2.1 El model: Model Vista Controlador

A l'hora de crear l'aplicació web, volíem crear-la mitjançant una estructura comprensible i eficient, on cada tecnologia realitzes el seu rol principal i deixés aquelles tasques per les quals no havia estat concebuda, a altres tecnologies.

En el món del desenvolupament web, sembla que predomina molt una arquitectura de tres capes, que emula bastant bé el model vista controlador. Però, en què consisteix exactament aquest model? El model vista controlador, també conegut com a MVC, és un patró d'arquitectura pensat per la implementació d'applicacions que disposen d'una interfície d'usuari.

Com bé indica el nom, el model és compost principalment per tres elements. El Model, la Vista i el Controlador. A continuació, descrivim amb més profunditat el rol de cada un d'aquests components.

El Model és el principal encarregat de gestionar i manipular les dades amb les quals treballa el sistema. El Model, també s'encarrega de crear la lògica i regles, sobre les que l'aplicació funciona. D'aquesta forma, aquest component serà l'encarregat de gestionar les connexions amb les bases de dades, en cas que es requereixi la utilització d'aquestes, i crearà els blocs de dades a retornar de forma que puguin ser compresos pel Controlador.

La Vista o Vistes, representen les representacions visuals de la informació. En altres paraules, la interfície que l'usuari veurà i amb la que podrà interactuar. Mitjançant les diferents interaccions possibles amb aquesta, l'usuari és capaç d'indicar a l'aplicació quines són les accions que vol que realitzi.

Finalment, el Controlador s'encarrega de recollir els diferents inputs enviats per l'usuari, validar-ne l'estat i comunicar-se amb la capa del Model per obtenir les dades

o recursos (entesos com a fitxers servits per la capa Model), demanats per l'usuari. En cas de necessitat, el Controlador també és l'encarregat de modificar la vista o l'estat d'una vista, per reflectir en tot moment l'estat de l'aplicació a la interfície d'usuari.

La gràcia d'aquest model és que l'usuari només disposa d'accés i permisos d'interacció amb la capa de les vistes. Aquestes, comuniquen les accions realitzades per l'usuari al controlador, que a la vegada, s'encarrega de gestionar les comunicacions amb el model i quan aquest retorna dades, transmetre-les a la vista.

D'aquesta forma, l'ús de la capa model és completament transparent per l'usuari. La figura 8.1, mostra un exemple de com podria funcionar el model MVC en un cas ideal. En la nostra aplicació web, intentarem seguir aquest model en la mesura que sigui possible.

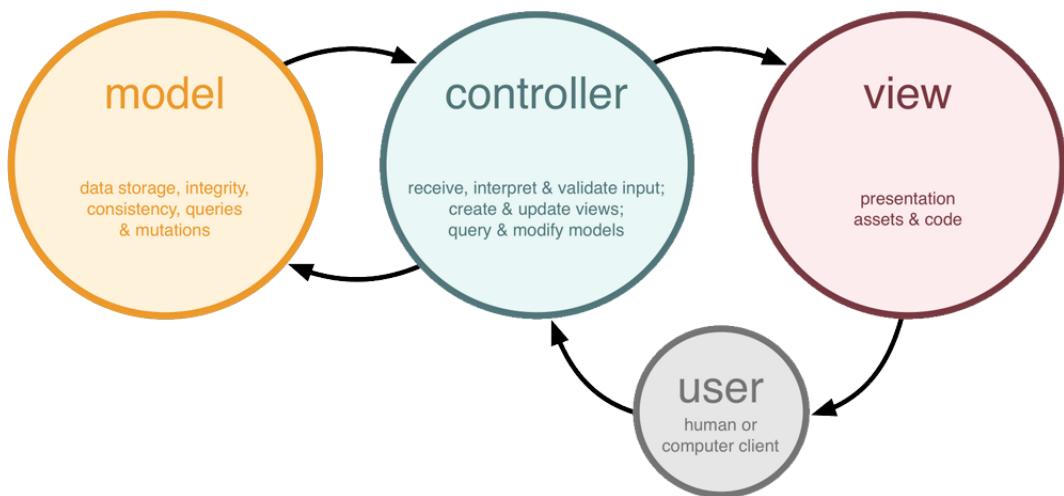


Figura 8.1: Exemple de funcionament del model MVC.

Un últim aspecte important, de cara a la terminologia d'aquesta secció de la memòria, és entendre com relacionem cada una de les capes del patró MVC, amb els elements que conformen una aplicació web.

Les pàgines web, s'acostumen a poder dividir en dos grans conceptes o elements principals: el front-end i el back-end.

El front-end, fa referència a la capa de presentació o amb altres paraules, el navegador de l'usuari i per tant, aquest element serà associat a la capa Vista del model MVC.

Per altra banda, el back-end és el component que acostuma a realitzar l'accés a les dades i tracta la informació que ha de ser servida al front-end. Per tant, en el nostre cas el back-end jugarà el paper de la capa Model en la nostra aplicació web.

Finalment, el paper del Controlador, sol ser representat en una aplicació web com la part del codi del client (navegador), que l'usuari no veu, ni interactua directament amb ella. En alguns sectors, aquesta part d'una pàgina web es coneix pel nom del

'back-end del front-end'.

La figura 8.2, mostra els tres elements descrits de les aplicacions web i el paper que jugaran, dins del model MVC, cada un d'ells. Durant els següents apartats d'aquesta secció de la memòria, anirem emplenant cada una de les caixes, amb les diferents tecnologies que seran utilitzades.

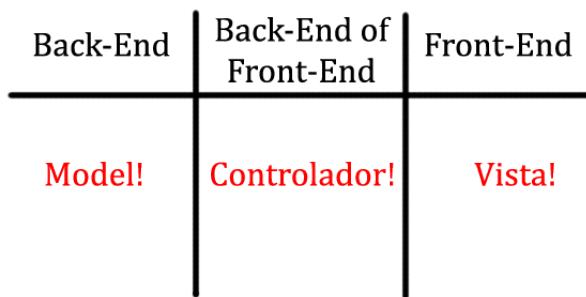


Figura 8.2: Elements de l'aplicació web i el patró MVC

8.2.2 Les tres capes del disseny web (Front-end)

Abans d'entrar en les tecnologies que utilitzarem per crear l'aplicació web, volem destacar una peculiaritat del front-end. En general, el front-end és dividit en tres capes, també conegudes com les capes d'Estructura, Estil i Comportaments.

La capa d'estructures és generalment coneguda com la capa del contingut. Aquesta, representa l'estructura sobre la qual el contingut de la pàgina web serà pintat. Es podria entendre també, com la creació de les diferents caixes, sobre les que més endavant es pintarà el contingut.

La capa d'estil, defineix com les diferents estructures han de ser situades en el navegador de l'usuari, així com l'estil dels diferents elements que seran pintats en aquestes estructures. Per exemple, controla el color de la font o les imatges de fons.

Finalment, la capa de comportament, s'encarrega de respondre a les diferents accions realitzades per l'usuari i a modificar l'estat de les capes estructures i estil. Aquesta capa del front-end és el que hem anomenat en l'apartat anterior, 'el back end del front end', i com ja hem comentat, jugarà el paper de Controlador en el model MVC.

Ara bé, però perquè és important diferenciar aquestes tres capes i perquè les hem reflectit en la memòria? Les raons són diverses:

- **Recursos compartits:** Moltes vegades, certs aspectes de les capes d'estil i comportament, podran ser reutilitzats en diverses pàgines de l'aplicació web i per tant, no té cap sentit haver de duplicar el contingut d'aquests a cada pàgina de l'aplicació, en cas de no voler diferenciar les capes.
- **Descàrregues més ràpides:** Un cop un d'aquests recursos compartits ha

estat descarregat, aquest s'emmagatzema en el navegador de l'usuari i ja no cal tornar-lo a descarregar. En cas no voler utilitzar aquesta arquitectura per capes, cada pàgina hauria de contenir tot el codi necessari per funcionar.

- **Treball en equip:** Permet que diferents persones treballin a la vegada sobre la mateixa pàgina de l'aplicació web, en apartats diferents, sense superposició.
- **User-friendly:** Permet crear aplicacions més fàcils d'interpretar pels motors d'optimització de cerca, així com una accessibilitat i compatibilitat amb diferents navegadors, més elevada.

8.2.3 La tecnologia HTML 5

La primera tecnologia que calia estudiar, si es volia implementar una pàgina web, era la tecnologia HTML i més concretament, el HTML 5.

Aquesta tecnologia és utilitzada per crear l'estructura de les aplicacions web i configurar quina informació s'ensenyarà a cada lloc del navegador. La tecnologia HTML, és considerada un llenguatge d'etiquetatge.

El concepte llenguatge d'etiquetatge, significa que els diferents blocs de contingut, s'envolten per etiquetes d'obertura i clausura, que atorguen un significat concret al contingut situat a l'interior.

Per exemple, per indicar la creació d'un bloc de contingut (`<div>`), que en el seu interior, conté tres paràgrafs (`<p>`), es podria utilitzar una estructura com la que segueix:

Codi 8.1: Bloc de contingunts HTML amb tres paràgrafs

```
<div>
  <p> ... </p>
  <p> ... </p>
  <p> ... </p>
</div>
```

El llenguatge HTML disposa d'un ampli ventall d'etiquetes diferents, que permeten introduir imatges, crear enllaços web, ressaltar el text en negreta o cursiva, etcètera, etcètera.

A més a més, aquest llenguatge crea el que s'anomenen estructures jeràrquiques, on un element pot tenir el rol de pare o fill d'un altre. Per exemple, en l'exemple de codi mostrat, el contenidor `<div>` és el pare de tots els paràgrafs. El llenguatge HTML, permet crear tants nivells de jerarquia diferents com es desitgin.

Així doncs, el llenguatge HTML és utilitzat per crear la capa d'estructures explícada a l'apartat: 'Les tres capes del disseny web', i per tant, podem dir que aquesta tecnologia és utilitzada únicament, a la capa del front-end de la nostra aplicació web.

8.2.4 La tecnologia CSS

La tecnologia CSS, també coneguda com a generació de fulls d'estil en cascada, és un llenguatge utilitzat per explicitar com ha de ser l'aspecte i forma dels diversos elements que apareixen en l'estructura d'una pàgina web, o el que és el mateix, en el seu HTML.

El CSS va néixer per poder separar el contingut d'un document, de la presentació d'aquest. Aquest llenguatge permet, entre moltes altres coses, decidir la font i estil de cada element de la pàgina web, l'alignació del text, les separacions entre els diferents components del HTML, els colors o imatges de fons, l'estil dels enllaços web, les transicions entre els diferents estats d'un component HTML, etcètera.

Una de les principals característiques del CSS és que en tractar-se d'una fulla d'estils en cascada, un element pot tenir definides diferents regles que marquen el valor d'un cert atribut, i el llenguatge CSS és capaç de sabers quina ha d'aplicar segons la posició de cada regla dins de l'estructura HTML.

A menys que s'especifiqui de forma contrària, la regla que s'aplica per personalitzar l'atribut d'un element HTML, és la que aplica directament sobre l'element, o aquella heretada del seu pare més proper en l'estructura HTML, que té aquell atribut estilitzat.

D'aquesta forma, s'aconsegueix dotar d'un comportament genèric a certs components HTML i personalitzar només aquells als que volem dotar d'atributs diferents.

En resumen, la tecnologia CSS permet controlar tots els atributs que fan referència a l'aparença de les diferents estructures i components HTML. Això ho aconsegueix mitjançant tres classes d'etiquetatge diferents:

- **Etiquetes HTML:** Aquestes etiquetes s'utilitzen per aplicar regles a tots els elements HTML que encaixin amb una etiqueta en concret. Per exemple, si s'utilitza l'etiqueta 'div', es podria decidir des d'aquesta regla la tipografia a ser utilitzada per tots els contenidors del HTML.
- **Etiquetes de classe:** Aquestes etiquetes, creades per l'usuari mitjançant la concatenació del caràcter '.', amb un nom qualsevol a decisió de l'usuari (per exemple: '.color-blue'), permeten assignar regles d'estil a classes concretes creades per usuaris. Aquestes classes, poden ser incloses dins dels components HTML, proporcionant així l'estil desitjat només a aquell conjunt de components marcats per l'etiqueta de classe especificada.
- **Etiquetes identificadores:** Aquestes etiquetes són similars a les de classe, però en comptes d'utilitzar el caràcter '.', abans del nom que vol ser introduït, s'utilitza el caràcter '#'. La diferència principal entre les etiquetes de classe i identificadores, és semàntica. En principi, en un HTML, hauria d'existir com a màxim una instància de cada etiqueta identificadora, mentre que les etiquetes de classe poden ser utilitzades en múltiples elements HTML.

Per exemple, podríem incloure l'etiqueta identificadora *first* i l'etiqueta de classe

notícia a un element HTML, de la següent forma:

```
<div id='first' class='noticia'> ... </div>
```

El llenguatge CSS també ofereix la possibilitat de definir regles més complexes, com per exemple, definir un conjunt d'atributs per aquells elements que tinguin un pare específic a l'estructura HTML o que es trobin afectats per esdeveniments especials, com podria ser per exemple, el 'mouseover' (element HTML enfocat pel cursor del ratolí).

Tanmateix, les bases són les mateixes tant pels casos simples com pels casos més complexos i creiem que el text introduït en aquesta secció, ja hauria de ser suficient per entendre de què s'encarrega aquesta tecnologia, fer-se una idea de com funciona i entendre el codi creat en els fitxers HTML i CSS de l'aplicació web implementada.

La tecnologia CSS es veu utilitzada únicament a la capa del front-end.

8.2.5 La tecnologia de plantilles Mustache

Abans d'entrar en els detalls d'aquesta tecnologia, cal comprendre que és un llenyatge de plantilles. Els llenguatges de plantilles existeixen principalment per satisfer dos propòsits.

En primer lloc, reutilitzar i minimitzar la quantitat de codi HTML necessari. Els llenguatges de plantilles permeten la incorporació de documents HTML, dins d'altres documents HTML. D'aquesta forma, s'evita la creació de codi redundant o replicat.

Per exemple, si quasi totes les pàgines d'un domini web tenen la mateixa barra de navegació, seria poc eficient haver de replicar l'estructura HTML d'aquesta en totes i cada una de les pàgines. De forma contrària, s'utilitzen aquestes plantilles per definir-la un cop i utilitzar-la a tot arreu.

Aquests llenguatges també permeten crear bucles de codi HTML i iterar sobre ells. Imaginem que es volen crear 100 paràgrafs de text en un document HTML, en comptes de crear-los de forma manual, podem crear un bucle amb un llenguatge de plantilles, definir el contingut de només una de les iteracions i deixar que el codi s'encarregui de crear totes les línies de codi necessàries.

El segon propòsit dels llenguatges de plantilla, consisteix a introduir contingut dinàmic, als fitxers HTML, a través dels paràmetres servits pel servidor. D'aquesta forma, s'aconsegueix dotar als documents HTML de contingut personalitzat, abans d'enviar la pàgina cap al client o navegador.

Per la creació de la pàgina web, es van estudiar tres llenguatges de plantilles diferents: Mustache, Twig i EJS.

Mustache era el més simple dels tres i el que dotava al frontal de menys lògica. En el costat oposat, estava Twig, que permetia realitzar tota mena d'operacions en el frontal, fins al punt de permetre la creació de variables dins del HTML. EJS, es trobava en un punt intermedi i mai va acabar de resultar una opció a valorar.

Una de les altres diferències principals entre Mustache i Twig, és que Mustache pot ser utilitzat en quasi qualsevol llenguatge de programació, mentre que Twig és específic del llenguatge PHP. Al final, es va decidir utilitzar un servidor Node.js (com s'explicarà més endavant en aquesta secció de la memòria), fet que descartava la possibilitat d'utilitzar el llenguatge Twig i feia de Mustache l'opció escollida.

El fet que Mustache sigui un llenguatge de plantilles sense gaire lògica, no significa que sigui menys complet que els altres. Segueix podent complir amb les dues funcionalitats principals descrites en aquest apartat i només implica que les dades que volen ser utilitzades en el HTML, han de venir estructurades des del servidor.

Les tres operacions principals de Mustache són:

- **Utilitzar un paràmetre del servidor en el HTML:** Per invocar en el HTML un paràmetre del servidor, només cal utilitzar el nom del paràmetre envoltant dels caràcters '{{ i }}'. Per exemple, {{parametreServer1}}.
- **Invocar el HTML d'un altre fitxer:** Això s'aconsegueix mitjançant la inclusió de la següent etiqueta en el codi HTML del fitxer desitjat: {{> navbar}}. El codi anterior, importaria per exemple, el contingut del fitxer navbar.html, en el document HTML actual.
- **Iteracions sobre blocs de codi:** Imaginem que el servidor retorna un vector de països, si volguéssim pintar el nom de cada país un en un paràgraf diferent, podríem definir el bucle Mustache de la següent forma: {{#countries}} <p> {{name}} </p> {{/countries}}. On {{#countries}} ... {{/countries}} representa el bloc de codi HTML a replicar a cada iteració i {{name}}, el paràmetre a imprimir de cada país.

Resulta doncs, bastant palpable, la utilitat que poden arribar a ser aquests llenuguatges de plantilles i perquè s'ha decidit utilitzar-ne un, en el desenvolupament de l'aplicació web.

Aquesta tecnologia pot esdevenir confusa de cara a comprendre en quin lloc de l'aplicació web treballa. Es podria pensar que és una tecnologia del front-end, ja que manipula el HTML, però en realitat, aplica al back-end, modificant el contingut del HTML, abans de servir-lo al client.

8.2.6 Les tecnologies Javascript i jQuery

Tot i que aquestes dues tecnologies no són exactament el mateix, volem presentar-les de forma conjunta, ja que són utilitzades d'aquesta forma en el món de les aplicacions web.

Javascript és un llenguatge dinàmic d'alt nivell que s'ha convertit, conjuntament amb el HTML i el CSS, en un dels tres pilars de la programació web. Gairebé totes les pàgines l'utilitzen i és suportat per tots els navegadors moderns. Javascript és un llenguatge de programació molt flexible i permet diferents estils de programació, com pot ser la programació orientada a objectes, o estils de programació imperatius

i funcionals.

Cal comprendre que Javascript també és utilitzat fora del món web, però aquest representa el seu mercat principal. Tampoc s'ha de confondre aquest llenguatge amb el llenguatge de programació Java. A pesar de les similituds entre els dos, es tracta de dos llenguatges de programació diferents, amb desenvolupaments separats.

Per altra banda, jQuery és una llibreria de Javascript plena de funcionalitats dedicada a la manipulació de documents HTML, CSS i gestió de les comunicacions entre aquests i el servidor de l'aplicació web. Aquesta llibreria, s'ha convertit en un estàndard de la programació web i resulta indispensable de cara a crear aplicacions web funcionals i interactives.

jQuery destaca sobretot per la possibilitat d'executar blocs de codi diferents quan l'usuari interactua de forma específica amb elements HTML, que disposen de certes etiquetes de classe o identificadores, de la mateixa forma que els fitxers CSS utilitzen aquestes etiquetes per definir l'aparença dels components HTML de la web.

Javascript i jQuery són utilitzats de forma conjunta en la capa anomenada ‘backend del front-end’ i seran, per tant, els encarregats de realitzar la funció de Controlador en l'arquitectura MVC de la nostra aplicació web.

No volem entrar més en detall en com s'utilitzen aquestes tecnologies, ja que les possibilitats són realment il·limitades i intentar fer un petit manual d'ús, seria impossible. No obstant això, per tal que s'entengui una mica més la funcionalitat d'aquestes, imaginem-nos una pàgina web que disposa d'un botó amb identificador submit, que un cop pressionat, canvia el valor d'un camp de text del HTML de la pàgina web.

En aquest exemple, s'utilitzaria jQuery per detectar que el botó submit del HTML ha estat pressionat, Javascript per definir una nova variable de text i jQuery per imprimir el contingut d'aquesta nova variable al camp de text del HTML. El codi podria tenir un aspecte similar al que segueix:

Codi 8.2: Example of jQuery and Javascript

```
$('#submit').click(function () {
    var newText = 'next text to display';
    $('#textField').text(newText);
});
```

Tot i ser un exemple molt bàsic, creiem que pot ajudar a comprendre perquè s'utilitzen aquestes tecnologies i com interactuen amb els elements de la pàgina web.

8.2.7 La tecnologia Bootstrap

Bootstrap és el marc de treball o ‘framework’, més popular pels llenguatges HTML, CSS i Javascript de cara a desenvolupar aplicacions web de disseny adaptatiu i aplicacions web orientades a dispositius mòbils.

Bootstrap proporciona un conjunt d'estils CSS i codis Javascript que faciliten el

desenvolupament de certes funcionalitats per les aplicacions web.

La funcionalitat principal que va portant-se a decidir utilitzar aquesta tecnologia és la funcionalitat del grid o quadrícula. A l'hora de programar una pàgina web que vol ser pintada de forma apropiada tant en dispositius mòbils com escriptoris, un dels punts més complicats és el d'assegurar que tots els components definits es comportaran com s'espera d'ells en cada dispositiu i que ocuparan posicions diferents segons la grandària del dispositiu en què es mostren.

El grid de Bootstrap parteix de la base que tot el contingut web se situarà dins de contenidors que poden ocupar o bé tot l'ample disponible del dispositiu o bé una amplada màxima. L'objectiu del grid és partir aquests contenidors en un seguit de files i columnes, on cada casella resultant pot contenir un bloc de codi diferent.

La imatge 8.3 mostra un exemple de com un contenidor podria estar dividit en quatre files, on cada fila esta formada per un nombre de columnes diferent.

.col-md-1											
.col-md-8								.col-md-4			
.col-md-4				.col-md-4				.col-md-4			
.col-md-6						.col-md-6					

Figura 8.3: Exemple d'un contenidor bootstrap amb 4 files i diferent nombre de files

La gràcia de dividir, cada fila en columnes és que en el moment que la grandària de la fila definida, superi la del dispositiu que les vol pintar, les columnes interiors de la fila s'apilen unes sobre les altres de forma automàtica, adaptant d'aquesta forma el contingut per un dispositiu mòbil sense necessitat de realitzar canvis en el codi o implementar un codi específic per aquests dispositius.

La imatge 8.4 mostra com una configuració web per escriptori es reorganitzaria per un dispositiu mòbil.

A part del grid, que representa la característica principal per la qual es va decidir utilitzar Bootstrap, aquesta entorn de treball també ofereix estils, o classes CSS, per taules, botons, formularis, imatges, tipografies, icones, barres de navegació, alertes, barres de progrés, contingut expansible i més funcionalitats.

A pesar que la majoria de classes oferides per Bootstrap, han de ser retocades mitjançant CSS propi per tal d'encaixar i adaptar els diferents elements a la nostra aplicació web, representen un molt bon punt de partida, que estalvia temps als desenvolupadors i a la vegada, n'assegura la correcta visualització a través dels diferents dispositius.

Resumint, el marc de treball Bootstrap proporciona principalment fitxers de codi CSS i Javascript que serveixen per complementar els nostres propis fitxers de codi.

Aplicació Desktop



Aplicació Mòbil

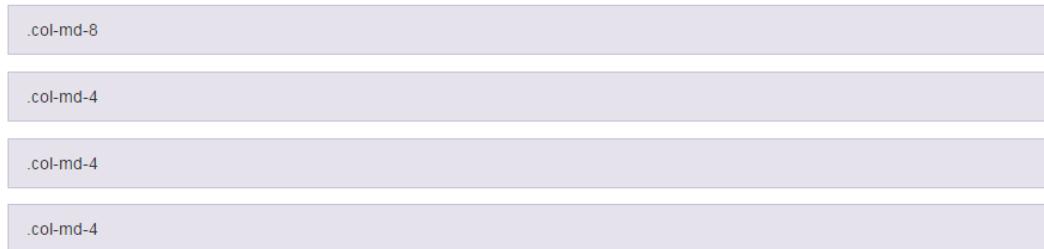


Figura 8.4: Exemple del mateix grid en pantalles Escriptori i Mòbil

De la mateixa forma que les tecnologies CSS i Javascript, anteriorment descrites, aquest entorn de treball actua tant en el front-end del nostre sistema, com en el ‘back-end del front-end’. Per tant, exerceix a la vegada, un efecte a les capes Vista i Controlador de la nostra aplicació web.

8.2.8 El SDK Javascript oficial de FamilySearch

Una de les tecnologies principals sobre les quals feia falta prendre una decisió al més aviat possible, era la que marcaria com s'haurien de realitzar les comunicacions entre l'aplicació web i l'API de FamilySearch.

La primera opció de la qual disposàvem era realitzar una implementació directa contra l'API. Aquesta aproximació tenia tots els números de ser la més flexible de totes, ja que permetria realitzar les peticions a l'API de forma completament personalitzada.

Per contrapartida, s'haurien de tractar les respostes XML o JSON de l'API de forma manual, el que acostuma a resultar una tasca repetitiva i ineficient. A part, la codificació de les diferents URI per tal d'accedir als recursos de l'API, també hauria de ser realitzada de forma manual, amb la inseguretat afegida, de que qualsevol lleuger canvi en l'estructura de l'API, implicaria haver de canviar el codi de la nostra aplicació.

Després d'estudiar les funcionalitats i documentació disponible en el portal de desenvolupadors de FamilySearch, la idea d'utilitzar un SDK per la implementació de l'aplicació web, va anar cobrant força.

Els avantatges principals d'utilitzar un SDK eren el processat automàtic de les respostes de l'API i la utilització de funcions de conveniència, que simplificaven certes tasques i permetien accedir a la informació de forma fàcil i transparent. El desavantatge principal, una reducció en la flexibilitat a l'hora de demanar i navegar a través dels recursos de FamilySearch i el fet d'haver d'estudiar el funcionament d'una nova tecnologia.

Al cap de valorar-ho bastant, es va decidir realitzar la implementació dels exemples d'interacció amb l'API, mitjançant un SDK.

El benefici de tenir un cert grau de robustesa sobre els petits canvis als quals l'API es pogués veure sotmesa, més el fet de no haver de realitzar el tractament de dades de forma manual, compensava l'esforç d'estudiar el funcionament del SDK. A part, en primera instància, per tal de demostrar el propòsit i potencial general de l'API, no feia falta ser capaç de controlar totes les interaccions al més mínim detall.

Pel simple fet d'haver decidit implementar una aplicació web, eren tres els diferents SDK que podien ser utilitzats. El SDK de Python va quedar descartat des del començament, ja que no es troava acabat i tampoc és un SDK oficial. Recordar, que els SDK oficials són desenvolupats per la mateixa organització de FamilySearch i per tant, més robusts de cara a possibles canvis de l'API.

La disputa doncs, quedava entre la utilització del SDK de Javascript o el de PHP. Després d'un estudi 'superficial' dels dos, ens vam acabar decantant pel Javascript SDK.

A pesar que la preferència inicial era utilitzar el SDK de PHP, ja que en el passat havia tocat una mica el llenguatge i tenia una idea aproximada de com es podia crear una aplicació web amb PHP, les funcionalitats d'aquest estaven molt per darrere de les que oferia el SDK de Javascript i l'extensa documentació del segon, també garantia una corba d'aprenentatge fàcil i ràpida.

Tot plegat, va fer que el SDK escollit per implementar els exemples fos el que va ser creat amb el llenguatge Javascript. Els detalls més tècnics d'aquest, així com el seu funcionament bàsic, seran explicats en la següent secció de la memòria, just abans de presentar l'aplicació web i els exemples implementats.

Finalment, esmentar que el SDK de Javascript, ens obria les portes a escollir si el volíem implementar a la capa del Model o a la capa del Controlador de l'arquitectura MVC.

La implementació més robusta i segura seria implementar-lo en la part del servidor, per tant, en el back-end. No obstant això, a causa del baix grau de dificultat de les operacions que es volien realitzar i la inexistent necessitat d'emmagatzemar informació en bases de dades, es va decidir implementar-lo a la capa del controlador.

D'aquesta forma, es reduïa al mínim la necessitat d'utilitzar la tecnologia Ajax, el que reduïa el nombre de tecnologies a estudiar i facilitava la impressió dels resultats provinents de les peticions a l'API al front-end de l'aplicació.

Al final del projecte i veient-ho amb la perspectiva del coneixement obtingut,

probablement, l'opció d'implementar el SDK al back-end de l'aplicació, no hauria esdevingut gaire més complicada. Tanmateix, cal recordar que el coneixement inicial a l'hora d'implementar una aplicació web, era pràcticament nul.

8.2.9 La tecnologia Node.js

Una de les poques tecnologies que ens quedaven per escollir, era la que faria funcionar el back-end o servidor de l'aplicació.

Aquest component de l'aplicació web generalment s'encarrega de gestionar les peticions de navegació i servir el contingut demanat als navegadors o usuaris. També s'encarrega de gestionar l'accés a les bases de dades (en cas que existeixin) i processar-ne la informació, per tal de servir-la d'una forma que la capa Controlador pugui comprendre i utilitzar.

Com que havíem escollit utilitzar el SDK de Javascript per realitzar les connexions a l'API de FamilySearch, es va decidir que era bona idea implementar el back-end amb una tecnologia que pogués fer funcionar el SDK, en cas que al final no volguéssim implementar la lògica de la connexió a la capa del controlador.

Després de buscar, l'única tecnologia que complia amb les condicions que buscàvem, era Node.js. Node.js ha estat implementat com un model basat en esdeveniments sense interrupcions, el que el converteix en un sistema en temps d'execució lleuger i eficient.

El paràgraf anterior, ve a significar que Node.js funciona mitjançant esdeveniments i de forma asíncrona, el que permet que moltes connexions siguin tractades de forma simultània. En el moment que una petició conclou, es dispara un esdeveniment de finalització, que s'encarrega d'activar una rutina que decidirà quina és la següent acció a realitzar. Aquest aspecte, converteix a Node.js, com una tecnologia molt escalable.

A més a més, el fet de tractar-se d'un software de codi obert que ha rebut una gran acceptació, ha propiciat que molts desenvolupadors s'hagin dedicat a crear paquets de software que n'amplien les funcionalitats inicials. Totes aquestes extensions, poden ser trobades al repositori de paquets, de l'ecosistema npm, el més gran del món en quant a llibreries de codi obert.

Comentarem més sobre aquesta tecnologia i com funciona, en els apartats específics d'implementació que apareixeran més endavant en aquesta memòria.

Aquesta tecnologia realitza el paper del Model en l'arquitectura MVC de la nostra aplicació o el que és el mateix, en el back-end d'aquesta.

8.2.10 La tecnologia Express o Express js

De la mateixa forma que descrivíem a Bootstrap com un marc de treball per les capes vista i controlador, Express és un marc de treball pel servidor de l'aplicació i

en el nostre cas concret, per la tecnologia Node.js.

Així doncs, Express és un marc de treball minimalist i flexible pensat per crear aplicacions web i aplicacions mòbil simples, mitjançant Node.js. Express també està pensat per la creació d'APIs robustes.

Diem que Express és minimalist, perquè mitjançant l'aplicació d'una fina capa d'eines, destinades al desenvolupament web i situada per sobre de la tecnologia Node.js, s'aconsegueix una gran potencialitat sense la necessitat d'emmascarar o alterar les funcionalitats principals i simples per les quals Node.js destaca.

Molts altres marcs de treball o frameworks populars que han desitjat ampliar les possibilitats de Node.js, han nascut a partir de la base d'Express.

Podem justificar la utilització d'aquest entorn de treball per la nostra aplicació, ja que els requisits que tenim pel back-end, després de decidir implementar la interacció amb l'API de FamilySearch a la capa del controlador, són relativament simples.

Les funcionalitats o eines principals que proporciona Express són:

- **Eines d'encaminament:** L'encaminament recull el conjunt d'accions i processos que tenen lloc des que el servidor web rep la petició d'accés a una URL, fins que respon a la petició amb els recursos adequats.
- **Eines middleware:** El concepte middleware serveix per referir-nos a aquelles peces de software que fan de pont entre dues parts o components d'un sistema. En el marc de treball en el qual ens estem referint, serveix per capturar les peticions al servidor per part de l'usuari i abans de servir-ne la resposta, realitzar algunes accions o comprovacions. Un exemple típic podria ser el de comprovar que un usuari té els permisos suficients per veure la pàgina demanada.
- **Eines per la utilització de llenguatges plantilla:** Aquest conjunt d'eines són les que ens permeten configurar el servidor Node.js per comprendre i poder utilitzar, per exemple, el llenguatge de plantilles Mustache que hem descrit en aquest mateix apartat de la memòria.
- **Eines per gestionar errors:** Eines específiques per gestionar els errors com a middleware.
- **Eines de depuració:** Eines destinades a ajudar als desenvolupadors durant els moments de desenvolupament a detectar errors mitjançant, en gran part, la consola.
- **Eines de connexió a bases de dades:** Conjunt d'eines per integrar de forma fàcil diferents bases de dades amb el servidor.

Donada la naturalesa de la nostra aplicació, les funcionalitats que ens interessen més són les tres primeres. Aquesta tecnologia, en tractar-se d'un framework de Node.js, s'utilitza, evidentment, en la part del back-end de la nostra aplicació.

8.2.11 Recapitulació de les tecnologies pel desenvolupament web

Els apartats anteriors d'aquest apartat de la memòria s'han utilitzat per explicar i justificar les diferents tecnologies que formen part de la nostra aplicació web.

En total, es tracta de nou tecnologies que han hagut de ser estudiades per separat i après a utilitzar de forma conjunta. L'única tecnologia sobre la qual es disposava en certa forma, d'un petit coneixement previ, era la tecnologia Javascript.

Es va dedicar una part important del temps destinat al projecte a aprendre a utilitzar tot aquest conjunt de tecnologies, però creiem que el resultat ha estat bastant satisfactori.

El mapa de tecnologies i el paper que juga cada una d'aquestes en l'arquitectura de capes proposada, es pot veure representada a la figura 8.5.

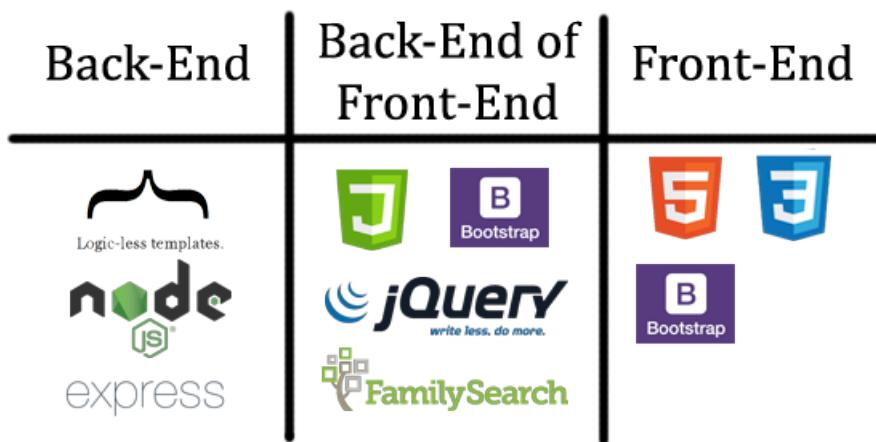


Figura 8.5: Mapa final de tecnologies en cada capa de l'aplicació

8.3 Technologies específiques pel desenvolupament de l'aplicació

Apart de les tecnologies que permeten la creació i funcionament de l'aplicació web, hi ha un altre conjunt de tecnologies o eines que s'han utilitzat per facilitar el desenvolupament d'aquesta.

8.3.1 La tecnologia GitHub

A causa de les complicacions resultants de compaginar una jornada laboral a temps complet amb l'elaboració del projecte, el desenvolupament d'aquest s'ha realitzat des de diverses localitzacions diferents. En concret, han estat tres els principals ordinadors o localitzacions des de les que s'ha treballat: Casa, Portàtil i oficina de l'empresa.

Per tal de facilitar la sincronització d'arxius entre les diferents estacions de treball, es va decidir utilitzar la tecnologia GitHub.

GitHub és un repositori Git hostejat al núvol. Ofereix totes les funcionalitats pròpies de Git, destinades al control de revisions distribuïdes i control del codi font, afegint-ne algunes de noves.

Un cop creat el repositori on tot el projecte queda emmagatzemat, els servidors de GitHub s'encarreguen de què aquest sigui disponible des de tot arreu. D'aquesta forma, cada estació de treball diferent pot descarregar-se el codi per primera vegada, mitjançant la instrucció:

```
git clone https://github.com/sinh15/pfc-family-search.git pfc-family-search
```

És important evitar treballar en dues estacions de treball diferents, a menys que entre sessions de treball s'actualitzin els canvis en el repositori emmagatzemat al núvol, per tal d'evitar superposicions i conflictes entre les versions dels arxius.

GitHub disposa d'eines per fer front a aquestes situacions, ja que en realitat és una eina pensada per treballar en equip sobre els mateixos arxius de codi. No obstant això, com que aquest projecte ha estat realitzat per una sola persona, s'ha prescindit de la utilització de branques i tots els canvis s'han aplicat sobre la branca mestra del projecte.

Un cop s'acaba una sessió de feina, independentment de l'estació de treball, es poden pujar els canvis al núvol mitjançant tres simples instruccions:

```
git add .
git commit -m "sessió de treball X finalitzada"
git push origin master
```

De la mateixa forma, abans de començar a treballar des de qualsevol estació de treball, podem recuperar l'últim estat del projecte mitjançant la instrucció:

```
git pull origin master
```

A part dels beneficis que aquesta eina ens proporciona de cara a treballar de forma distribuïda i en diferents entorns, serveix al mateix temps, de 'backup' o reserva, en cas que alguna de les oficines de treball quedí malmesa o es vulgui recuperar una versió antiga d'algun fitxer del codi.

8.3.2 Node.js i Express en l'àmbit local

Durant el desenvolupament, per tal de facilitar les proves sobre l'aplicació i no haver de realitzar un desplegament al núvol per cada canvi que es vulgui comprovar, aquesta s'ha programat en un entorn local.

Aquest fet implica que el nostre sistema operatiu feia de servidor per l'aplicació web i aquesta només resultava accessible a través de la URL ‘<http://localhost:8080>’.

Per tant, el nostre sistema local necessitava ser capaç d'emular les tecnologies que formarien part del servidor. A recordar, Node.js i Express.

No entrarem a descriure la tecnologia Node.js ni Express, ja que ja ho hem fet en apartats anteriors. El que sí que volíem fer, era exposar que aquestes tecnologies havien estat instal·lades en l'àmbit local.

8.3.3 Tecnologia NPM

Com s'ha indicat en un apartat anterior, NPM és un contingut de paquets orientats a la plataforma Node.js. Aquesta tecnologia pot ser instal·lada en l'àmbit local i d'aquesta forma, descarregar extensions per les aplicacions Node.js que volem provar.

Per instal·lar un conjunt de paquets, podem fer-ho introduint la instrucció: `npm install`, que s'encarrega d'instal·lar en l'àmbit local, totes aquelles dependències que hagin estat declarades en el servidor de l'aplicació web, o introduir la instrucció: `npm install 'package name'`, per instal·lar un paquet en concret.

8.3.4 Paquet Nodemon

Un paquet que volem destacar, instal·lat a través del repositori NPM, però que només ha estat utilitzat en l'entorn de desenvolupament local i no desplegat al núvol, s'anomena Nodemon.

Nodemon observa els canvis realitzats en els fitxers de codi de la nostra aplicació i en cas que algun canvií, aquest reinicia el servidor que fa córrer l'aplicació de forma automàtica, per tant, sense necessitat de reiniciar-lo de forma manual i poder veure així els canvis a l'entorn local de forma immediata.

Pot semblar un paquet que aporta poca funcionalitat, però quan et trobes implementat una pàgina web, generalment realitzes molts petits canvis en el codi dels quals vols observar-ne l'afecte abans de continuar programant. Per fer-ho, faria falta reiniciar manualment el servidor, però aquest paquet, ens estalvia aquesta tasca.

El fet de ser un paquet que ha estalviat bastant de temps a l'hora de realitzar petites proves i observar-ne els canvis, volíem que trobes la seva representació en la memòria.

Secció 9

Javascript SDK Oficial de FamilySearch

9.1 Introducció al Javascript SDK

Abans de procedir a explicar com funciona l'aplicació web desenvolupada, volíem presentar per sobre en què consisteix el SDK oficial Javascript de FamilySearch.

No explicarem la informació accessible a través d'aquest, perquè en realitat, es tracta gairebé de la mateixa que la presentada en la secció cinc d'aquesta memòria. No obstant això, creiem que pot resultar interessant presentar les principals funcionalitats que aquest SDK ofereix i exposar els diferents motius, pels que aquestes funcionalitats, compensen l'esforç d'estudiar-lo.

L'objectiu del SDK, per resumir-ho d'una forma simple, és facilitar el consum dels recursos accessibles a través de l'API de FamilySearch.

Per aconseguir-ho, el SDK envolta cada funció que realitza una crida a l'API amb una funció pròpia i afegeix funcions de conveniència a la resposta per tal de navegar pels resultats de forma còmoda. De totes maneres, el SDK no emmascara la resposta retornada per l'API i en conseqüència, els recursos JSON o XML retornats per aquesta, són accessibles en cas de voler accedir a una peça d'informació concreta que no hagi estat emmascara en una funció de conveniència.

En cas que el SDK no estigués preparat per fer front a alguna de les peticions que l'API ofereix, aquest implementa un conjunt de funcions de plomeria que permeten a l'usuari realitzar amb comoditat les típiques crides GET, POST, DEL, etcètera, que es realitzarien contra l'API en cas d'una integració directa, mitjançant el SDK.

Per tant, el SDK ens ofereix una cobertura quasi completa de l'API oficial amb funcions de conveniència i en cas que alguna consulta no fos suportada, sempre es podria realitzar mitjançant la via tradicional a través del SDK.

A continuació, s'expliquen les principals funcionalitats d'aquest SDK i en què consisteixen.

9.2 Emmascarant les crides REST a l'API de FamilySearch

Qualsevol crida que es pretengui realitzar contra l'API de FamilySearch, es realitzarà en el SDK mitjançant una funció Javascript asíncrona que l'emmascara. Això ens permet no haver de preocupar-nos per les URI dels recursos als quals volem accedir o haver d'afegir les capçaleres correctes a cada petició, ja que és el mateix SDK el que s'encarrega de fer-ho i mantenir el conjunt d'URIs als diferents recursos i operacions actualitzat.

Per exemple, si volem accedir a la persona amb identificador: 'KW7S-VQJ', ho podríem fer mitjançant la següent funció.

Codi 9.1: Exempla crida emmascarada a l'API de FamilySearch

```
client.getPerson('KW7S-VQJ', {persons:true}).then(function(response) {
    ...
});
```

La variable *client* representa una instància del SDK, l'operació, *getPerson*, l'operació del SDK que volem invocar, l'identificador *KW7S-VQJ* i el JSON *{persons:true}*, són els paràmetres a passar a la funció i la variable *response*, és l'objecte que emmagatzemarà la resposta de l'API.

9.3 Implementació basada en Promeses

Les promeses són els objectes retornats pels blocs de codi Javascript asíncrons, com per exemple, en la funció de l'apartat anterior, la promesa retornada seria la variable *response*. Una promesa es troba sempre en un dels següents tres estats:

- Pendent: Estat inicial de la promesa.
- Satisfeta: L'estat de la promesa representa una operació finalitzada amb èxit.
- Rebutjada: L'estat de la promesa representa una operació fallida.

Tan bon punt una promesa rep l'estat de satisfeta o rebutjada, ja no pot tornar a canviar d'estat.

La gràcia de les promeses és que tan bon punt són resoltes, permeten executar una part del codi definida amb anterioritat (el que seria el cos de la funció de l'apartat anterior) i mentre aquestes no són satisfetes o rebutjades, la resta del codi pot seguir executant-se. Amb altres paraules, el codi no roman bloquejat mentre espera la resposta de la funció contra l'API.

9.4 Implementació pensada per la programació orientada a objectes

Les funcions de crida del SDK, retornen promeses, que no deixen de ser objectes Javascript. Aquests objectes, disposen de funcions de conveniència que permeten accedir als resultats de la resposta sense haver de navegar per objectes de format XML o JSON.

Per exemple, en la crida de fa dos apartats, podríem obtenir el nom de la Persona cercada, a través de l'objecte response, mitjançant la navegació al recurs de la persona i seguidament, demanant-ne el nom. El següent bloc de codi mostra aquest exemple.

Codi 9.2: Example navigació per l'objecte *response*

```
client.getPerson('KW7S-VQJ', {persons:true}).then(function(response) {
    console.log(response.getPrimaryPerson().getDisplayName());
});
```

9.5 Model de dades quasi idèntic a FamilySearch

Una altra característica destacable del SDK és que implementa un model de dades equivalent al de l'API de FamilySearch, però en aquest cas, pensat per ser navegat mitjançant els estàndards dels llenguatges de programació orientada a objectes, en comptes dels enllaços hypermedia.

A la figura 9.1 podem observar el model de dades proposat pel SDK i com cada un dels objectes es troba relacionat amb els altres.

9.6 Captura d'errors

Els errors de connexió amb l'API són fàcilment tractables gràcies a la naturalesa de les crides asíncrones implementades pel SDK. Només cal afegir la clàusula '.catch' al codi de la petició, tal com es mostra en el següent bloc de codi. La variable *error* és un altre objecte Javascript que conté informació sobre l'error.

Codi 9.3: Tractament d'errors amb l'objecte *error*

```
client.getPerson('KW7S-VQJ', {persons:true}).then(function(response) {
    // Tractar les dades retornades per l'API
})
.catch(function(error) {
    // Codi per gestionar l'error
});
```

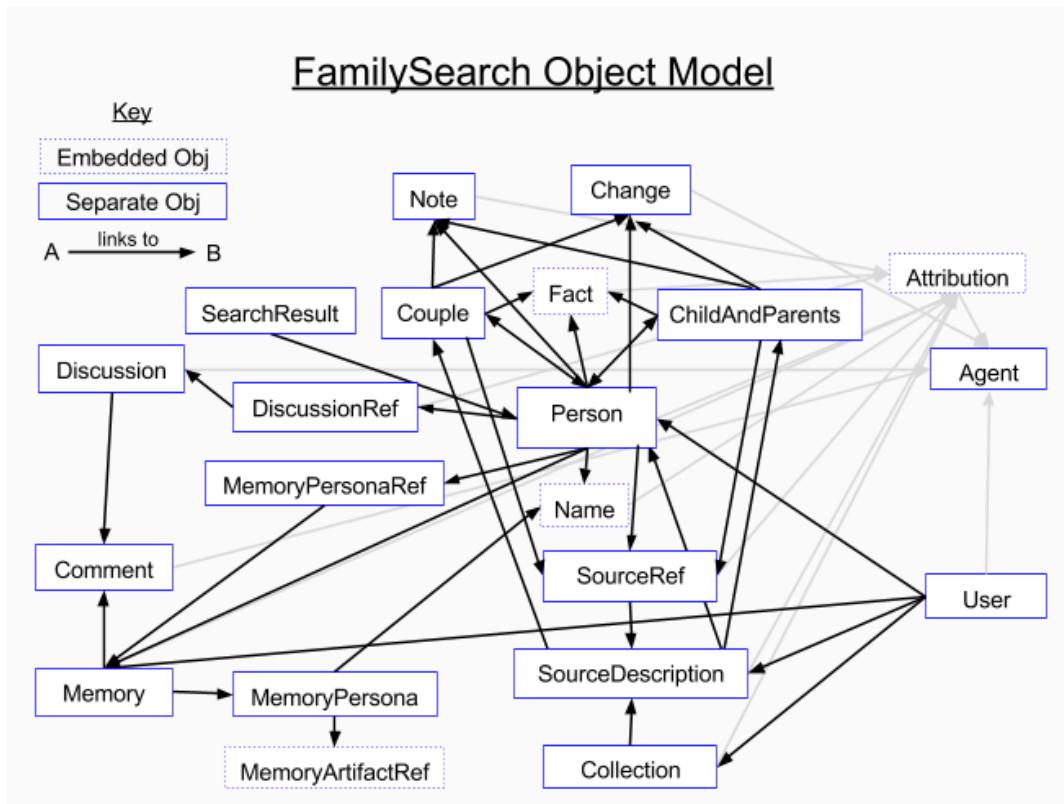


Figura 9.1: Model de dades del Javascript SDK de FamilySearch

9.7 Intentar de nou peticions GET fallides

Els errors de connexió amb l'API són fàcilment tractables gràcies a la naturalesa de les crides asíncròniques implementades pel SDK. Només cal afegir la clàusula ‘.catch’ al codi de la petició, tal com es mostra en el següent bloc de codi. La variable error és un altre objecte Javascript que conté informació sobre l'error.

9.8 Gestió automàtica del Throttling

Com s'exposava en la quarta secció de la memòria, quan es parlava d'algunes de les funcionalitats extres que l'API de FamilySearch oferia, va ser presentada la funcionalitat de *Throttling*, que evitava que un usuari realitzes ‘masses’ peticions, durant un cert període de temps.

En cas que l'usuari del SDK sigui bloquejat durant un període de temps per aquesta causa, el mateix SDK, s'encarregarà de gestionar la duració del bloqueig, evitant rellançar la crida fins que aquesta pugui tornar a ser llençada.

9.9 Autentificació mitjançant un pop-up

Mentre es realitzi des d'un navegador, l'autentificació d'una aplicació amb FamilySearch, pot ser gestionada mitjançant un pop-up a la pàgina oficial de l'organització.

D'aquesta forma, no ens hem de preocupar de crear una pàgina de redirecció específica pel protocol OAuth. L'únic requisit, és registrar un URL vàlid, en el mateix domini i port que l'aplicació web, que serà cridada un cop es finalitzi el procés d'identificació a la banda de FamilySearch.

9.10 Autentificació automàtica

Existeix l'opció d'activar el protocol d'identificació de forma automàtica, en cas que un usuari intenti realitzar una operació contra l'API sense identificar-se primer, aquesta acció serà demanada abans d'executar la crida. Pot esdevenir útil, ja que les connexions expiren al cap d'un temps d'inactivitat.

9.11 Emmagatzematge del Token en una cookie

Existeix l'opció d'emmagatzemar el token d'identificació (retornat per l'operació d'identificació amb FamilySearch) en una cookie.

Resulta útil de cara a utilitzar el SDK des de la capa del controlador, ja que d'aquesta forma no resulta necessari crear una instància del client a cada pàgina diferent de l'aplicació web o implementar una aplicació web basada en una sola pàgina.

Malauradament, viola una de les condicions per la certificació de les aplicacions, així que no s'acaba d'entendre perquè apareix en un SDK oficial, mes enllà d'ajuda durant el desenvolupament.

Per tal d'intentar superar aquesta restricció, en el nostre projecte, hem emmagatzemat el valor del Token en l'espai local del navegador.

9.12 Disparador automàtic de la funció d'expiració

Per tal de mantenir la coherència entre l'aplicació web i l'estat de la identificació, recordem que aquesta pot expirar per inactivitat, existeix la possibilitat d'implementar una funció que s'executi quan el token expira.

D'aquesta forma, podem garantir que en tot moment existeix una concordança d'estat entre el client i el servidor i que per tant, no corre el risc que l'usuari pugui realitzar operacions no permeses en expirar el token d'identificació.

9.13 Utilitzable des de diferents plataformes o capes

El SDK és utilitzable tant des del client com el servidor. En altres paraules, pot córrer tant mitjançant la tecnologia Javascript en la capa del controlador, com mitjançant Node.js a la capa del back-end.

9.14 Suport en el desenvolupament

Una de les característiques principals del SDK és que es troba relativament ben documentat. Inclòs m'atreviria a dir, que de cara a obtenir una idea general de com funciona l'API de FamilySearch i quina informació conté, realitza una millor feina que la documentació oficial de l'API. Encara que sigui de forma involuntària, és un punt a destacar.

Al mateix temps, existeixen diferents grups d'ajuda per aquells desenvolupadors que desitgin utilitzar l'API de FamilySearch i que són visitats de forma regular pels desenvolupadors dels SDK oficials i l'API de FamilySearch.

Des d'aquesta memòria, es recomana passar pel grup de Google 'FamilySearch Developer Network', ja que resulta l'emplaçament ideal per qualsevol dubte que puguem tenir i un centre d'informació ideal sobre el funcionament de l'API i l'estat en què es troba a cada moment.

9.15 Conclusió sobre el SDK

Com s'ha pogut observar en les seccions anteriors, són molts els beneficis resultats d'utilitzar un SDK oficial, en comptes de realitzar una implementació directa contra l'API. Per altra banda, el preu a pagar és pràcticament nul, més enllà d'haver d'estudiar el funcionament del SDK, però resulta un esforç rendible.

Secció 10

Introducció a l'aplicació web

En aquesta secció de la memòria s'introduiran tots aquells aspectes generals referents a l'aplicació web. En concret, es tractaran els següents punts:

- Accés i codi de l'aplicació web.
- Requisits funcionals i no funcionals de l'aplicació web.
- Estructura de l'aplicació web.
- Estructura de l'aplicació web.
- Funcionament general de l'aplicació web.
- Detalls específics de la implementació.
- Certificació de l'aplicació.
- Google Analytics.
- Optimització d'imatges.
- Hosting de l'aplicació web.

10.1 Accés a l'aplicació web i codi de l'aplicació

L'aplicació web es troba desplegada al núvol sota l'URL:

<https://pfc-family-search.herokuapp.com/>

Per accedir a la zona específica d'exemples, la que s'encarrega de mostrar els diferents exemples d'interacció amb l'API, fa falta utilitzar el següent usuari i contrasenya:

- **Usuari:** tum000145207
- **Contrasenya:** 1234pass

Per altra banda, el codi de les aplicacions web sol ser extens en nombre de línies i mostrar-lo en aquesta memòria resulta impossible. El codi realitzat ocupa un total de [nombre de línies] repartides un [x%] en HTML, un [y%] en Javascript i jQuery i un [z%] en css.

Tot el codi de l'aplicació pot ser trobat en el repositori GitHub accessible a través del següent URL:

<https://github.com/sinh15/pfc-family-search>

L'estructura del codi serà presentada més endavant, en aquesta mateixa secció de la memòria, però principalment, el servidor està compost pel fitxer *app.js*, els fitxers HTML es troben a la carpeta *views* i els fitxers Javascript i jQuery, a la carpeta *assets*.

10.2 Requisits de l'aplicació web

Aquesta llista pretén oferir un tast dels requisits o manaments que s'han tingut en compte durant el desenvolupament de l'aplicació web.

10.2.1 Requisits funcionals

- La web ha de permetre identificar-se amb FamilySearch mitjançant el sistema d'autentificació per pop-up.
- La web ha de permetre a l'usuari tancar la connexió amb FamilySearch mitjançant una funcionalitat de 'Sign Out'.
- L'aplicació ha de ser capaç de tancar automàticament la connexió amb FamilySearch si aquesta expira.
- La web ha d'incloure una secció que ofereixi un petit resum del rerefons que va originar el projecte.
- La web ha de disposar d'una secció en què s'enlacen i exposin les diferents propostes de projecte generades pels futurs estudiants.
- L'aplicació ha d'ofrir la possibilitat de cercar persones en l'arbre familiar de FamilySearch i observar-ne els detalls d'alguna en concret.
- L'aplicació ha de permetre a l'usuari observar l'evolució geogràfica d'un cognom donat un conjunt de països i període de temps.
- L'aplicació ha de permetre la visualització del nombre de naixements, casaments i defuncions enregistrades per un país al voltant d'un any concret.
- La secció d'exemples ha de ser només accessible si l'usuari es troba identificat a FamilySearch i ha rebut el token d'ús pertinent.

- En cas que el token expiri, l'usuari ha de ser redirigit a la pàgina principal en el moment d'expiració o en la seva següent interacció si aquest es troba dins de l'àrea d'exemples.
- L'aplicació ha d'emmagatzemar el token proporcionat per FamilySearch que rep l'usuari en un recurs que no sigui accessible ni modificable per tercers.
- No es permetrà a l'usuari llençar dues crides contra l'API de FamilySearch simultànies per la mateixa funcionalitat des de la mateixa pestanya del navegador.
- L'aplicació ha d'aportar la informació bàsica sobre l'origen del projecte i el seu rerefons. L'aplicació també ha d'enllaçar en algun lloc amb el codi font del projecte.

10.2.2 Requisits no funcionals

- L'aplicació web ha de funcionar i ser visualitzada de forma correcta en els principals navegadors web moderns.
- Els formularis de l'aplicació web que puguin generar errors han de proporcionar informació bàsica a l'usuari en el moment que el camp és abandonat o informació més detallada si envia el formulari amb errors.
- Els formularis de l'aplicació han de donar un feedback positiu en cas que els camps siguin omplerts de forma correcta.
- La web ha de ser relativament fàcil d'utilitzar, oferint les eines necessàries als usuaris i facilitant la navegació per les diferents seccions.
- Mentre l'aplicació web espera resposta de l'API de FamilySearch, s'ha de mostrar a l'usuari que l'aplicació es troba esperant resultats i el progrés realitzat fins al moment.
- L'aplicació ha de donar un feedback clar a l'usuari quan la interacció amb l'API de FamilySearch finalitza.
- L'aplicació ha de ser navegable de forma acceptable mitjançant dispositius mòbil. Les integracions amb l'API de FamilySearch també han de ser utilitzables, però no cal que la informació resultant es trobi completament adaptada a aquests dispositius.
- Les imatges de l'aplicació s'han de trobar optimitzades en la mesura que sigui possible per intentar que aquesta carregui el més ràpid possible en un entorn d'hostalatge gratuït.
- Les imatges principals de l'aplicació s'han de carregar de forma transparent quan l'aplicació és iniciada i la primera pàgina és carregada, per millorar la visualització de la web quan l'usuari navega entre les diferents seccions.
- La llengua utilitzada en el web serà l'anglès per tal d'ajudar i facilitar el procés de certificació.

- El projecte s'ha de trobar sota una certificació Creative Commons d'atribució no comercial.
- La informació sobre el codi font del projecte, la llicència i la facultat d'informàtica ha de trobar-se disponible en el peu de pàgina de les pàgines de la web.
- Es podrà monitorar la navegació dels usuaris pel web, així com les seves accions principals i errors generats.
- Es podrà obtenir la configuració dels sistemes amb els quals s'ha navecat per la web i veure si el comportament d'algun d'ells és més propici a la generació d'errors.
- Els fitxers Javascript s'han de trobar el més al final possible dels arxius HTML per facilitar la càrrega del contingut.
- Es reutilitzarà codi HTML i Javascript en la mesura que sigui possible per tal d'evitar la duplicació de contingut.

10.3 Estructura de l'aplicació web

10.3.1 Introducció a l'estructura de l'aplicació

L'aplicació web és relativament simple pel que respecta a la navegació i les diferents seccions que la conformen.

La figura 10.1 mostra l'arbre de continguts accessibles. Cal indicar, que aquesta figura no representa les úniques rutes de navegació existents entre les diferents seccions, sinó un breu mapa del contingut total disponible a través del web i d'on penja cada secció.

Més endavant, s'explicarà més en detall en què consisteix cada una de les pàgines o seccions de la nostra aplicació web, però primer volem presentar l'estructura general o esquelet, que segueixen gairebé totes les pàgines del nostre web.

La figura 10.2 mostra l'esquema bàsic sobre el qual les pàgines són construïdes. Aquest pot ser descrit o desglossat en les següents seccions:

1. **Barra de navegació:** Permet desplaçar-se per les diferents seccions principals.
2. **Capçalera de secció:** Conté el títol i subtítol de la pàgina sobre impressionat a una imatge relacionada.
3. **Contingut principal:** El contingut principal i únic d'aquesta pàgina.
4. **Footer:** Peu de pàgina. Inclou informació sobre el codi font del projecte, la llicència i la facultat d'informàtica de Barcelona.

Convidem des de la memòria, als usuaris de l'aplicació web, a redimensionar la finestra web per tal d'observar com els continguts de cada pàgina s'adapten al dispositiu que els mostra.

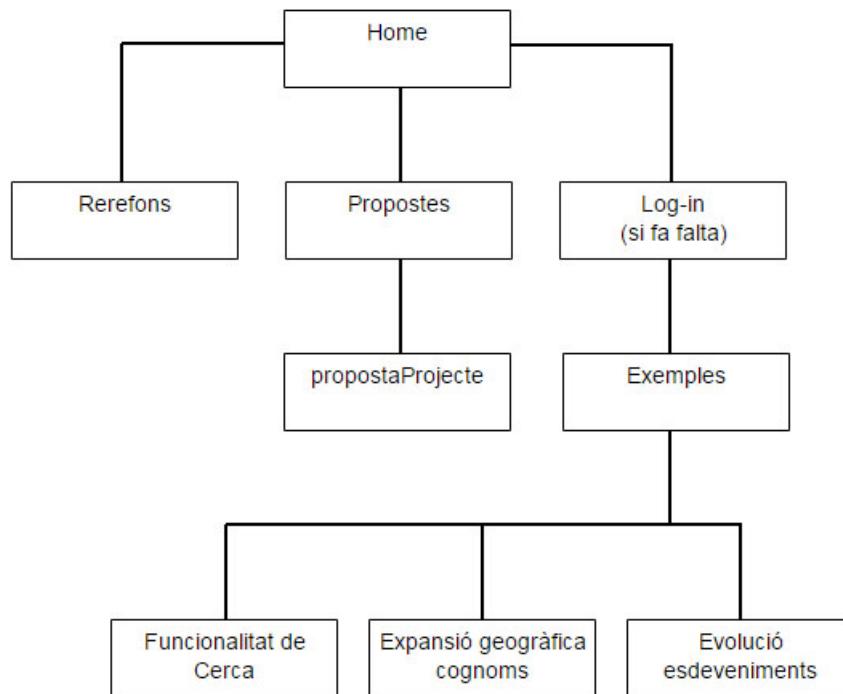


Figura 10.1: Diferents seccions de l'aplicació web

A continuació descriurem el contingut de cada secció o apartat que conforma la nostra aplicació web.

10.3.2 Home o pàgina principal

La home és la primera pàgina que veu l'usuari quan entra a l'aplicació web. Aquesta no té cap altre propòsit que el de donar la benvinguda i enllaçar als diferents continguts.

El bloc de contingut principal d'aquesta pàgina inclou enllaços i breus descripcions, a les tres seccions principals del projecte. A saber: rerefons, propostes de projecte i exemples d'implementació.

La principal diferència entre la versió per dispositius mòbils i la d'escriptori és que la primera fa desaparèixer el subtítol, simplifica el títol, transforma la barra de navegació en una usable per dispositius mòbil i transforma la secció de contingut principal apilant-ne el contingut de cada una de les tres columnes que la conformen.

10.3.3 Rerefons

La pàgina de rerefons conté la informació bàsica respecte a l'origen, context i motivacions, que ens van portar a realitzar aquest projecte. Consisteix en un breu

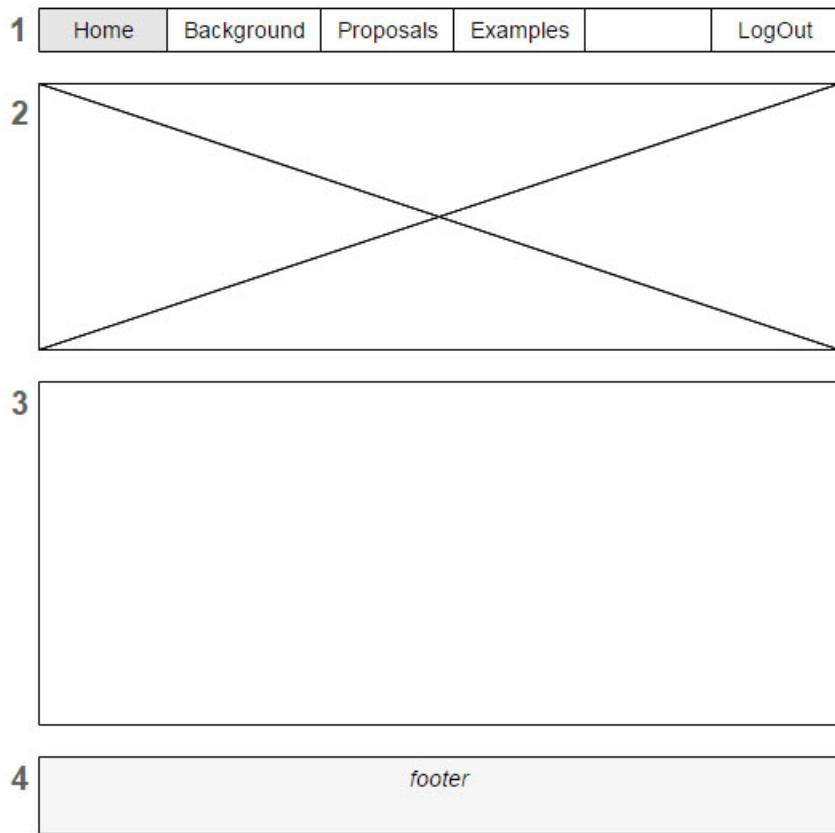


Figura 10.2: Esquema bàsic de les pàgines implementades en el domini web

resum, molt superficial, d'alguns dels apartats exposats en la primera secció de la memòria.

El bloc de contingut principal d'aquesta pàgina consisteix en dos grans blocs de text. El primer, descriu el rerefons del projecte, mentre que el segon ofereix una petita descripció del context i motivacions.

La principal diferència entre les versions d'escriptori i mòbil és que la segona presenta un títol més simple, la desaparició del subtítol i una barra de navegació adaptada a dispositius mòbils. L'estrucció del contingut roman igual, això si, adaptat a la grandària del dispositiu que el conté.

10.3.4 Propostes de projecte

L'apartat de propostes de projecte recull les diferents propostes que han estat generades per servir com a projectes finals de carrera pels estudiants d'informàtica.

El bloc de contingut principal per aquesta pàgina consisteix en dos blocs composts per petites caixes que contenen una imatge, un títol i una petita descripció de la

proposta que representen. Cada una d'aquestes caixes enllaça també amb una pàgina que conté alguns detalls de la proposta.

El primer bloc de caixes representa les propostes generades pels futurs estudiants, mentre que el segon bloc està format per les propostes relacionades amb els exemples implementats.

Les principals diferencies entre les versions d'escriptori i dispositius mòbils, és que la segona presenta un títol simplificat, la desaparició del subtítol, la barra de navegació adaptada i diferent nombre de caixes per fila segons el dispositiu utilitzat. Tres columnes per escriptoris, dues per tauletes gràfiques i una per mòbils.

10.3.5 Details específics d'una proposta de projecte

Com bé indica el nom de la secció, aquesta pàgina mostra els detalls específics de la proposta seleccionada des de la pàgina propostes.

Les diferents propostes són totes generades des del mateix document HTML. És en aquest cas el servidor, l'encarregat d'enviar un conjunt d'informació diferent segons la proposta que ha estat seleccionada. Veurem en més detall com aquest procés funciona en futurs apartats d'aquesta secció.

El bloc del contingut principal per cada una de les propostes, està conformat per una breu descripció del projecte i en alguns casos, certs exemples, preguntes o possibilitats d'extensió, que la proposta pot abordar. Les propostes també disposen d'una petita valoració numèrica, que representa una valoració de la dificultat d'execució de la proposta. Com més gran és el valor, més complexa.

La versió d'escriptori i mòbil no es diferencien en grans aspectes excepte en l'adaptació del contingut a la pantalla del dispositiu i els típics canvis esmentats en els apartats anteriors sobre el títol, subtítol i barra de navegació.

10.3.6 Identificació amb FamilySearch

Aquesta pàgina s'utilitza per assegurar que l'usuari no pot utilitzar els exemples sense identificar-se abans amb l'API de FamilySearch.

La pàgina apareix quan l'usuari intenta accedir a la pàgina d'exemples o la pàgina d'un exemple en concret, però encara no s'ha identificat amb FamilySearch. La pàgina permet dues accions simples, tornar enredadera (o a la home si s'ha accedit a la pàgina mitjançant la introducció directa de l'URL) o identificar-se amb FamilySearch.

El procés d'identificació s'inicia mitjançant el llançament d'un pop-up, que obre la pàgina de FamilySearch. Aquesta demana la introducció del nom d'usuari i contrasenya. Un cop aquesta informació ha estat verificada, el servidor redirigeix a l'usuari a la pàgina que havia demanat accedir.

La pàgina d'identificació tampoc pateix cap reestructuració especial del contingut quan es veu amb dispositius més petits. Simplement, s'adapta a la pantalla que la

mostra i reorganitza els botons que permeten tornar endarrere o obrir el procés d'identificació.

10.3.7 Exemples implementats

La pàgina d'exemples implementats permet a l'usuari descobrir les diferents eines que han estat implementades, per demostrar possibles interaccions amb l'API de FamilySearch i accedir a cada una d'elles.

El bloc de contingut principal, segueix un estil molt similar a la pàgina propostes de projecte, descrita tres apartats endarrere. Aquesta, mostra per cada un dels exemples implementats un títol, una breu descripció i permet a l'usuari navegar cap a les pàgines que contenen les implementacions concretes.

Les principals diferencies entre les versions d'escriptori i dispositius mòbils són exactament les mateixes que per la pàgina de propostes de projecte. És a dir, títol adaptat, eliminació del subtítol, adaptació de la barra de navegació i nombre de propostes per fila segons la grandària del dispositiu.

Realment, aquestes dues pàgines tenen un comportament tècnic idèntic i l'únic que les diferencia és el concepte semàntic que representen i en conseqüència, el contingut.

10.3.8 Funcionalitats de cerca, expansió geogràfica d'un cognom i evolució d'esdeveniments

Aquestes pàgines segueixen el mateix patró que la gran majoria de pàgines web de l'aplicació. Recordem que l'esquelet d'aquestes pàgines es mostrava en la figura 10.2 d'aquesta mateixa secció de la memòria.

La gran diferencia, d'aquestes pàgines amb la resta és que la part del contingut principal és relativament més complexa i different per cada una d'elles. És per aquest motiu, que el comportament exacte de cada una d'aquestes pàgines serà exposat per separat a la secció onze de la memòria.

Pel que fa a l'estructura en comú que comparteixen amb la resta de pàgines, les diferències entre la visualització entre dispositius mòbils i escriptori són les ja conegeudes: minimització del títol, desaparició del subtítol i adaptació de la barra de navegació.

10.4 Fitxers de l'aplicació web i la seva funcionalitat

En aquest apartat de la memòria volem presentar l'arbre d'arxius generats per tal de programar la pàgina web i exposar la funció que desenvolupa cada un d'ells en el marc de l'aplicació.

Recordem que tot el codi programat per tal de fer funcionar l'aplicació web pot ser trobat a l'URL:

<https://github.com/sinh15/pfc-family-search>

El conjunt de fitxers programats, representa un total de [línies codi] línies de codi, de les quals un XX% són codi HTML, un XX% codi Javascript i un X% codi CSS.

La taula 10.1 mostra la localització relativa de cada arxiu i en descriu breument la seva funcionalitat.

Taula 10.1: Fitxers de codi de l'aplicació web

Arxiu	Funció
app.js	Servidor Node.js. Aquest fitxer s'encarrega de gestionar totes les peticions HTTP i HTTPS i de configurar l'aplicació web. També manté la traçabilitat sobre l'estat actual de l'usuari i gestiona els ports de l'aplicació.
package.json	Conté metadades sobre l'aplicació web així com les dependències d'aquesta. S'utilitza sobretot per indicar al servidor Node.js quins paquets han de ser instal·lats per assegurar el correcte funcionament de l'aplicació.
Procfile	Fitxer utilitzat per indicar al proveïdor del servei d'hostalatge Heroku el tipus d'aplicació a configurar i el fitxer a executar per iniciar el servidor.
README.md	Breu fitxer que descriu el motiu de creació del projecte i l'ús de cada un dels fitxers que el conformen.
views/background.html	Fitxer HTML utilitzat per pintar la pàgina rerefons.
views/exemples.html	Fitxer HTML utilitzat per pintar la pàgina d'exemples.
views/facts.html	Fitxer HTML utilitzat per pintar la funcionalitat evolució d'esdeveniments.
views/index.html	Fitxer HTML utilitzat per pintar la pàgina inicial de l'aplicació.
views/login.html	Fitxer HTML utilitzat per pintar la pàgina d'identificació amb FamilySearch.
views/proposals.html	Fitxer HTML utilitzat per pintar la pàgina que recull el conjunt de propostes de projecte per futurs estudiants.
views/-proposalsTemplate.html	Fitxer HTML utilitzat per pintar els detalls d'una proposta en concret. El contingut és variable segons la proposta seleccionada per l'usuari.
views/search.html	Fitxer HTML utilitzat per pintar la funcionalitat de funcionalitat de cerca.
views/surnames.html	Fitxer HTML utilitzat per pintar la funcionalitat d'expansió geogràfica d'un cognom.

views/globals/-footer.html	Fitxer HTML encarregat de pintar el footer de la pàgina (per aquelles pàgines que el contenen).
views/globals/future-proposals.html	Fitxer HTML encarregat de pintar les propostes de projecte per futurs estudiants a la pàgina de propostes.
views/globals/-header.html	Fitxer HTML encarregat de pintar l'encapçalament de la web en totes les pàgines.
views/globals/-implemented-proposals.html	Fitxer HTML encarregat de pintar les caixes dels exemples implementats en les pàgines de recopilació de propostes i exemples.
views/globals/-javascripts.html	Fitxer HTML encarregat d'inserir els Javascripts comuns a totes les pàgines a cada pàgina.
views/globals/-navbar.html	Fitxer HTML encarregat de pintar la barra de navegació a cada pàgina en què vol ser mostrada.
views/globals/-pageTitle.html	Fitxer HTML encarregat de pintar la capçalera de secció de cada pàgina. Els continguts d'aquesta varien segons la pàgina seleccionada mitjançant els paràmetres enviats pel servidor.
views/globals/-searchMainPerson.html	Fitxer HTML encarregat de pintar el formulari de cerca pels camps de l'usuari principal a l'exemple de cerca.
views/globals/-searchMother.html	Fitxer HTML encarregat de pintar el formulari de cerca pels camps de la mare a l'exemple de cerca.
views/globals/-searchSpouse.html	Fitxer HTML encarregat de pintar el formulari de cerca pels camps de la parella a l'exemple de cerca.
assets/css/style.css	Fitxer CSS encarregat de modificar l'estil dels diferents elements HTML classes i identificadors.
assets/js/client.js	Fitxer Javascript que gestiona les funcions d'identificació tancament de sessió i establiment de la connexió amb l'API de Family-Search.
assets/js/cookies.js	Fitxer Javascript per gestionar les cookies per la banda del client (navegador). Fitxer de codi obert reutilitzat del web.
assets/js/-countryParameters.js	Fitxer Javascript per emmagatzemar la informació dels països continguts a cada continent. S'utilitza perquè el servidor pugui enviar al HTML els continguts i generar HTML dinàmic.
assets/js/facts.js	Fitxer Javascript encarregat de gestionar totes les interaccions que l'usuari pot realitzar en l'exemple evolució d'esdeveniments i les connexions amb l'API d'aquesta funcionalitat.
assets/js/-formValidation.js	Fitxer Javascript per escapar els camps dels formularis i validar-ne el contingut.
assets/js/gaTagging.js	Fitxer Javascript encarregat de configurar els esdeveniments de Google Analytics generals i llençar les crides de tots els esdeveniments.

assets/js/geosurnames.js	Fitxer Javascript encarregat de gestionar totes les interaccions que l'usuari pot realitzar en l'exemple d'evolució geogràfica d'un cognom i les connexions amb l'API de Familysearch d'aquesta funcionalitat.
assets/js/index.js	Utilitzat per llençar la càrrega d'imatges quan l'usuari aterra a la pàgina principal de l'aplicació.
assets/js/-jquery.preload.min.js	Fitxer Javascript encarregat d'executar la precàrrega dels documents que s'enviïn contra el complement. En la nostra aplicació l'utilitzem per carregar les imatges de les capçaleres de secció abans que l'usuari les demani. (Fitxer de codi obert programat per Ben Lin 2011)
assets/js/login.js	Fitxer Javascript encarregat de gestionar les interaccions d'usuari de la pàgina d'identificació i les connexions amb l'API de Familysearch realitzades des de la pàgina.
assets/js/pageTitles.js	Fitxer Javascript que emmagatzema els paràmetres de configuració de totes les capçaleres de secció. El servidor l'utilitza per obtenir els valors específics de cada pàgina i utilitzar-los per personalitzar els continguts del fitxer pageTitle.html
assets/js/-projectProposals.js	Fitxer Javascript que emmagatzema els detalls de les diferents propostes de projecte. El servidor l'utilitza per obtenir els valors específics de la proposta demandada i utilitzar-los per personalitzar els continguts del fitxer proposalsTemplate.html
assets/js/-proposalExamplesLinks.js	Fitxer Javascript utilitzat per redirigir de forma correcta les caixes de les pàgines de propostes i exemples.
assets/js/search.js	Fitxer Javascript encarregat de gestionar totes les interaccions que l'usuari pot realitzar en l'exemple de cerca i les connexions amb l'API de FamilySearch d'aquesta funcionalitat.

10.5 Funcionament general de l'aplicació web

L'objectiu d'aquest apartat és explicar com els diferents components de la web interactuen entre ells per tal de crear l'aplicació web desplegada al núvol.

Com a suport a les explicacions que s'oferiran, la figura 10.3 mostra un diagrama amb els components principals de l'aplicació, com interactuen entre ells i els principals formats de dades que intercanvien.

En la imatge es poden observar les capes Servidor, Controlador i Client, que conformen l'arquitectura en tres capes explicada en seccions anteriors. També es pot observar el Javascript SDK, del que es detalla com es comunica amb l'API de FamilySearch i a quina capa de l'aplicació es troba connectat.

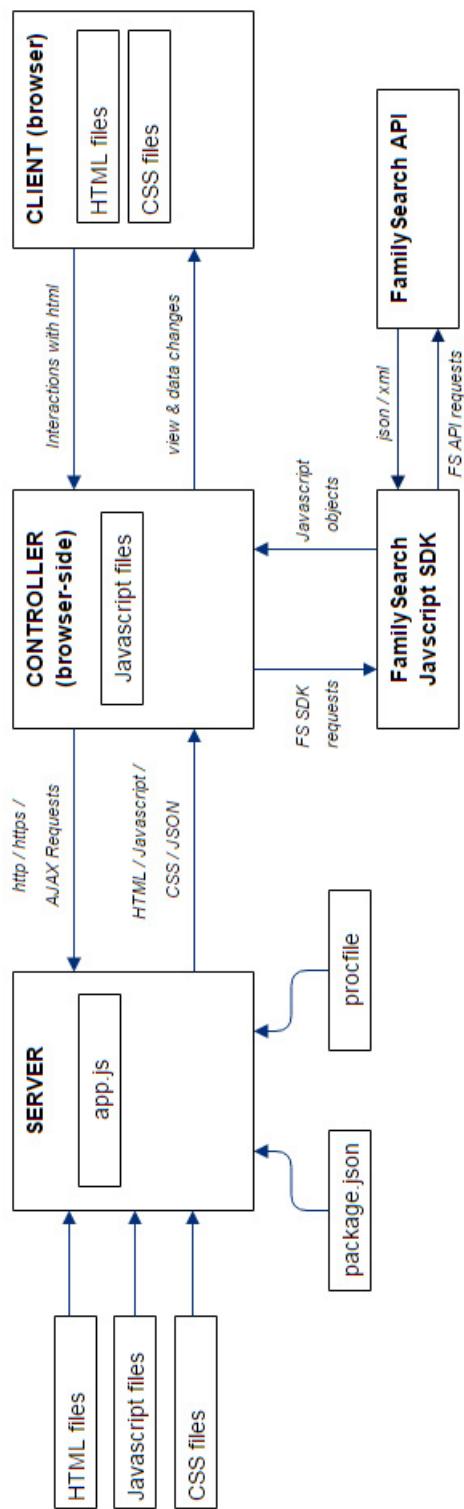


Figura 10.3: Comunicació entre els diferents components de l'aplicació web

10.5.1 Creació del servidor

L'aplicació comença a funcionar quan aquesta és executada en els servidors al núvol de Heroku, la plataforma d'hostalatge. La configuració del servidor es realitza mitjançant els fitxers *Procfile* i *package.json*, que s'encarreguen d'assegurar que tots els complements Javascript necessaris siguin instal·lats i s'executi el fitxer *app.js* per arrancar i configurar el servidor.

Quan la configuració del servidor acaba, aquest es troba preparat per començar a rebre peticions dels usuaris.

10.5.2 Accés a una pàgina del domini web

Quan un usuari demana carregar la pàgina inicial de la nostra aplicació, una petició HTTP o HTTPS és generada i enviada cap al servidor a través del navegador del client.

Quan el servidor la rep, l'avalua i en cas d'èxit, retorna al client els fitxers processats necessaris perquè el navegador pugui mostrar la pàgina demandada i carregar totes les funcionalitats o interaccions possibles d'aquesta al controlador.

10.5.3 Interacció amb l'API de FamilySearch

El moment en què l'usuari vol interactuar amb l'API de FamilySearch, aquest es veu forçat a interactuar primer amb un element del codi HTML per comunicar l'acció a realitzar. Per exemple, el botó de cerca de la funcionalitat d'evolució temporal d'esdeveniments.

Quan el controlador detecta que el botó de cerca ha estat pressionat, capture l'esdeveniment i avalua la petició. En cas que no es detecti cap problema ni error en els paràmetres introduïts, el controlador realitza una crida asíncrona al SDK de FamilySearch i n'espera la resposta.

De forma transparent a l'aplicació web, el SDK es comunica amb l'API de FamilySearch i si no hi ha cap problema en la comunicació i la petició és vàlida, aquesta retorna les dades demandades en format XML o JSON. Posteriorment, el SDK transforma la resposta de l'API en un objecte Javascript amb funcions de conveniència que facilitaran l'accés a les dades de la resposta i el retorna al controlador.

En el moment que el SDK retorna l'objecte, la promesa pendent de resolució que havia creat el controlador és resolta i l'objecte retornat pel SDK passa a ser accessible. Arribats a aquest punt, el controlador processa i transforma les dades contingudes a l'objecte de la forma desitjada i un cop finalitzades les operacions necessàries, modifica la vista del client introduint els canvis pertinents en aquesta.

10.5.4 Interacció amb elements del HTML

Quan els usuaris interactuen amb elements bàsics del HTML, per exemple, quan interactuen amb les caixes que permeten expandir o plegar seccions del formulari en les funcionalitats de cerca o evolució geogràfica de cognoms, la resposta a realitzar per part del controlador és bastant simple.

En el moment que el controlador detecta que s'ha interactuat amb algun dels elements que escolta, capture l'esdeveniment, avalua com s'ha de procedir segons el context de l'acció, en el cas de l'exemple, plegar o desplegar contingut d'un formulari i realitza de forma immediata els canvis a la vista del navegador.

10.5.5 Conclusió

Els casos d'ús que s'han cobert en els apartats anteriors són relativament simples, però són una mostra representativa del conjunt d'accions diferents a les quals l'aplicació pot haver de fer front.

Esperem que aquesta secció hagi servit per il·lustrar el funcionament general de la web i com els diferents components interactuen entre ells. La resta d'accions possibles en el web, requerint més o menys complexitat per part del controlador, segueixen una de les tres rutes descrites en els apartats anteriors, per tal de ser satisfetes.

10.6 Detalls específics de la implementació

10.6.1 Introducció als detalls d'implementació

L'objectiu d'aquesta secció és destacar alguns dels detalls d'implementació dels fitxers de l'aplicació.

En aquest apartat, no s'exposaran els detalls d'implementació de les funcionalitats integrades amb FamilySearch, doncs aquestes seran explicades, més en profunditat, en el seu propi apartat de la següent secció de la memòria.

En aquest apartat, no s'exposaran els detalls d'implementació de les funcionalitats integrades amb FamilySearch, doncs aquestes seran explicades, més en profunditat, en el seu propi apartat de la següent secció de la memòria.

Cal entendre que una pàgina web està formada per moltes línies curtes i altres de molt llargues. En conseqüència, mostrar el codi complet de cada fitxer, resulta impossible i inefficient, de cara a una memòria. A canvi, s'han seleccionat detalls d'implementació, que permeten explicar les bases sota les quals els diferents arxius funcionen i repliquen diverses vegades, al llarg del seu codi.

Instem als lectors de la memòria, a visitar també els fitxers de codi originals, que poden ser trobats a l'URL¹ ja mencionat diverses vegades, si així ho prefereixen.

¹<https://github.com/sinh15/pfc-family-search>

10.6.2 Estructura general d'una pàgina HTML:MustacheI

A excepció de la pàgina d'identificació amb FamilySearch, la resta de pàgines de l'aplicació web segueixen la mateixa estructura, tal com s'ha exposat en l'apartat Estructura de l'aplicació web.

Per evitar la duplicació de codi entre les diferents pàgines, els elements compartits han estat creats en fitxers separats i són carregats a cada pàgina segons si volem mostrar-los o no. Recordem que la tecnologia utilitzada, que fa això possible, és el llenguatge de plantilles Mustache.

L'esquelet del codi d'una pàgina qualsevol del nostre web segueix la forma mostrada en el bloc de codi que segueix.

Codi 10.1: Exemple d'inclusió de fitxers HTML amb Mustache

```
<!-- header includes -->
{{> header }}
{{> navbar }}
{{> pageTitle }}
<!-- specific page content -->
...
{{> javascripts }}
<!-- specific javascripts for this file -->
...
<!-- footer -->
{{> footer }}
```

Les etiquetes `{{> fileName}}`, s'utilitzen per indicar que es vol importar, en aquesta posició, el codi HTML del fitxer `fileName.html`.

Mitjançant aquesta estructura tenim el control absolut de quins components volem incloure a cada una de les pàgines. Per exemple, l'esquelet de la pàgina d'identificació, es diferencia de la resta en el fet que no s'inclouen els fitxers `navbar`, `pageTitle` ni `footer`.

També cal destacar que abans del `footer`, deixem un espai en blanc, per incloure fitxers Javascript que són utilitzats únicament per la pàgina carregada, d'aquesta forma s'evita carregar tots els controladors a totes les pàgines, si aquests no han de ser utilitzats.

10.6.3 El fitxer header.html: Configuració de la pàgina

Aquest fitxer s'encarrega de configurar la capçalera de les pàgines del nostre domini i obrir el cos del codi. També es declaren en aquesta secció els arxius CSS a carregar i les fonts a utilitzar.

Les línies de codi més interessants són les que configuren el ‘viewport’ del dispositiu, perquè aquest s’adapti de forma adequada a qualsevol pantalla i l’etiqueta necessària per indicar al navegador la codificació en què es troben els caràcters (línies 1 i 4 del següent codi).

Codi 10.2: Capçaleres incloses en el fitxer header.html

```
<meta charset='utf-8'>
<meta http-equiv='X-UA-Compatible' content='IE=edge'>
<title>PFC - Family Search API Study</title>
<meta name='viewport' content='width=device-width, initial-scale=1'>
<link href='/node_modules/.../bootstrap.min.css' rel='stylesheet'>
<link href='/assets/css/style.css' rel='stylesheet'>
<link href='fontNumber1' rel='stylesheet' type='text/css'>
<link href='fontNumber2' rel='stylesheet' type='text/css'>
```

10.6.4 El fitxer navbar.html: La barra de navegació adaptativa

Com el nom del fitxer indica, aquest s'encarrega de configurar la barra de navegació. Aquesta ha estat creada mitjançant la potencialitat d'un dels components de Bootstrap.

Consta principalment de dos blocs de codi HTML. El primer, s'utilitza per indicar la capçalera de la barra de navegació, que conté el botó per expandir-la en dispositius mòbils, la icona de l'aplicació que es mostra només en dispositius d'escriptori i el nom de l'aplicació web.

La capçalera adaptativa de Bootstrap, per la barra de navegació, s'invoca mitjançant la classe *navbar-header*.

Codi 10.3: Capçalera de la barra de navegació

```
<div class='navbar-header'>
    <button class='navbar-toggle collapsed'>...</button>
    <a href='/'><img src='/images/littleIco.png' /></a>
    <a class='navbar-brand navbar-link' href='/'>FamilySearch - PFC</a>
</div>
```

El segon bloc, s'encarrega de generar els enllaços a les diferents seccions de l'aplicació i configurar el botó dedicat al tancament de la connexió amb FamilySearch, si aquesta està oberta.

Cada enllaç és estilitzat i configurat mitjançant la classe *navbar-link*. Per decidir si un enllaç ha d'aparèixer per l'esquerra o dreta de la barra de navegació, es mira la inclusió de la classe *navbar-right* en el contingut dels enllaços.

Codi 10.4: Enllaços de la barra de navegació

```
<div class='collapse navbar-collapse' id='...>
    <ul class='nav navbar-nav'>
        <li><a class='navbar-link' href='/background'>Background</a></li>
        <li><a class='navbar-link' href='/proposals'>Proposals</a></li>
        <li><a class='navbar-link' href='/examples'>Examples</a></li>
    </ul>
    <ul class='nav navbar-nav navbar-right'>
        <li><a id='signOut' class='navbar-link' href>Sign Out</a>
    </ul>
</div>
```

10.6.5 El fitxer pageTitle.html: Mustache II

El fitxer *pageTitle.html* resulta un dels fitxers HTML més interessant de l'aplicació. Com s'ha comentat en el primer apartat d'aquesta secció, quasi totes les pàgines de l'aplicació contenen el bloc de codi que hem anomenat 'capçalera de secció'.

Aquesta capçalera consta d'una imatge de fons, amb un títol i subtítol superposats segons el dispositiu des del qual s'accedeix a la pàgina. Els paràmetres que marquen la imatge de fons, els texts, el color del ressaltat i si cal mostrar un botó de navegació o no, són enviats pel servidor i carregats de forma dinàmica segons la pàgina visitada.

Recordem que les etiquetes `{{nomEtiqueta}}` són referències a codi Mustache i representen paràmetres dinàmics emplenats pel servidor abans de servir el HTML. En el bloc de codi que es mostrarà més endavant, es podran veure importats els paràmetres: `backgroundImage`, `highlight`, `title`, `subtitleDesktop`, `subtitleTablet` i `button`.

Fixem-nos també en el fet que aquests paràmetres poder ser importants a qualsevol lloc del codi HTML. En alguns casos, s'importen com a classe d'un element HTML, indicant-ne l'estil i comportament esperat i altres, simplement, es tracta de contingut estàtic, com per exemple, el títol.

Destacar també, que pot resultat interessant fixar-se en com l'aplicació diferencia entre si s'ha de mostrar el subtítol per escriptori o tauleta gràfica, ja que la lògica és la mateixa per qualsevol altre component del domini web.

Els encarregats de gestionar aquest aspecte són les etiquetes de classe: `hidden-sm` i `hidden-xs`, en els subtítols d'escriptori i en contrapartida, l'etiqueta de classe: `visible-sm`, en els subtítols de la tauleta gràfica. Aquestes etiquetes poden ser llegides de forma semàntica com: Si la pantalla que mostra la pàgina web és petita (tauletes gràfiques) o ultra reduïda (mòbils), no mostris el subtítol per escriptoris i si el de tablet.

Així doncs, el següent bloc de codi representa la configuració de la capçalera de secció per dispositius de pantalla petita o superiors (fixem-nos en l'etiqueta de classe de la primera línia que només amaga la secció per dispositius `xs`).

Codi 10.5: Posicionament de paràmetres amb Mustache en el HTML

```
<div class='container-fluid {{backgroundImage}} hidden-xs'>
    <!-- main title --&gt;
    &lt;div class='row'&gt;&lt;h1 class='{{highlight}}'&gt;{{title}}&lt;/h1&gt;&lt;/div&gt;

    <!-- subtitle --&gt;
    &lt;div class='row'&gt;
        <!-- Subtitles: Desktop --&gt;
        &lt;div class='col-md-12 hidden-sm hidden-xs'&gt;
            &lt;h2 class='{{highlight}} text-italic'&gt;{{subtitleDesktop}}&lt;/h2&gt;
        &lt;/div&gt;
        <!-- Subtitles: Tablet --&gt;
        &lt;div class='col-md-12 visible-sm'&gt;
            &lt;h2 class='{{highlight}} text-italic'&gt;{{subtitleTablet}}&lt;/h2&gt;
        &lt;/div&gt;
    &lt;/div&gt;</pre>

```

```

</div>

<!-- display button if required --&gt;
{{#button}} ... {{/button}}
&lt;/div&gt;
</pre>

```

Al final del bloc de codi mostrat, en el fitxer original, s'inicia una capçalera similar per dispositius de pantalla extra reduïda. El motiu pel qual creem dues capçaleres diferents, és per mostrar una capçalera més petita en alçada i aprofitar així millor l'espai disponible en dispositius mòbils.

10.6.6 El fitxer javascripts.html: Càrrega dels controladors

Aquest fitxer s'encarrega de carregar tots els scripts comuns necessaris a les pàgines del nostre domini web.

El fitxer no representa cap misteri, excepte que l'ordre de declaració és important, per assegurar que no es carrega codi jQuery sense carregar abans el complement.

El fitxer carrega els complements jQuery, SDK de FamilySearch, fitxers Javascripts encarregats de manipular les galetes, la declaració de l'objecte FamilySearch i control de sessió, les interaccions pels components específics de Bootstrap i el snippet de codi de Google Analytics, que s'encarrega de monitorar el tràfic de l'aplicació web.

Les porques línies de codi interessants del fitxer són les encarregades de configurar el compte de Google Analytics i monitorar la pàgina visitada. Aquesta pàgina, també emmagatzemarà de forma automàtica, tota la informació relativa a la sessió d'usuari.

Codi 10.6: Càrrega general de controladors per totes les pàgines

```

<script src='https://ajax.googleapis.com/.../jquery.min.js'></script>
<script src='https://.../javascript-sdk.min.js'></script>
<script src='/assets/js/cookies.js'></script>
<script src='/assets/js/client.js'></script>
<script src='/node_modules/.../bootstrap.min.js'></script>
<script src='/assets/js/gaTagging.js'></script>
<script>
    (function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=function()
        {...} })(window,document,'script','https://www.google-
        analytics.com/analytics.js','ga');

    ga('create', 'UA-80847078-1', {
        'cookieDomain' : 'pfc-family-search.herokuapp.com'
    });
    ga('send', 'pageview');
</script>

```

10.6.7 El fitxer index.html: El grid de Bootstrap I

Aprofitarem aquest fitxer per explicar en profunditat el funcionament del grid de Bootstrap, introduït ja en la secció 'Estudi tècnic del projecte.'

El contingut principal del fitxer index.html, representa un conjunt d'enllaços als tres grans blocs de l'aplicació web, on cada un, es veu representat per una imatge, un títol i una descripció. En aplicacions d'escriptori, aquests tres blocs es mostren un al costat de l'altre, mentre que en aplicacions de pantalles més reduïdes, es mostren un sobre l'altre.

Per aconseguir-ho, es juga amb el concepte de files i columnes del grid de bootstrap. El grid, està format per contenidors o blocs d'espai, en els que es poden crear diferents fileres i cada una d'aquestes fileres, pot ser dividida en dotze columnnes.

Com que en la versió d'escriptori volem crear tres blocs idèntics, on cada un contindrà un enllaç a una secció de la pàgina web, a cada un dels blocs li corresponen $12/3 = 4$ columnnes. Per indicar-ho a bootstrap, s'utilitza la classe, *col-md-4*.

Així doncs, el codi extremadament simplificat d'aquesta secció podria ser representat de la següent forma:

Codi 10.7: Exemple bàsic de divisió d'una filera en tres columnnes

```
<div class='container'>
  <div class='row'>
    <div class='col-md-4'> ... </div>
    <div class='col-md-4'> ... </div>
    <div class='col-md-4'> ... </div>
  </div>
</div>
```

La configuració mostrada en el bloc de codi anterior representaria exactament el format que volem que la pàgina agafi per escriptoris, però perquè aquesta estructura funciona també en els dispositius mòbils apilant-ne els blocs de quatre columnnes un sobre l'altre?

Per comprendre-ho hem d'explicar primer, que Bootstrap, categoritza les aplicacions en quatre grandàries, segons l'amplada en píxels de la pantalla dels dispositius:

- **Dispositius extra reduïts (xs):** Fa referència als dispositius mòbils o pantalles amb amplitud inferior als 768 píxels.
- **Dispositius petits (sm):** Fa referència sobretot a tauletes gràfiques o pantalles amb amplitud superior o igual a 768 píxels, però inferior a 992 píxels.
- **Dispositius mitjans (md):** Fa referència a escriptoris o pantalles amb amplitud superior o igual als 992 píxels i inferior als 1200 píxels.
- **Dispositius grans (lg):** Fa referència a escriptoris grans o pantalles amb amplitud superior o igual als 1200 píxels.

Les lletres que s'han indicat entre parestèsies en el llistat anterior, són un codi que s'introduceix en la classe columna de Bootstrap (*col-xx-4*), per indicar sobre quina grandària mínima de dispositiu, s'ha d'aplicar la regla.

D'aquesta forma, la classe utilitzada en el bloc de codi d'exemple (*col-md-4*), indica que es vol declarar un bloc de quatre columnnes per dispositius mitjans o

superiors. Els dispositius que no compleixin aquesta regla, és a dir, els dispositius petits o extra reduïts, veuran transformada la classe de forma automàtica a *col-sm-12* o *col-xs-12* respectivament.

10.6.8 El fitxer background.html: El grid de Bootstrap II

Utilitzarem aquest fitxer per explicar una altra funcionalitat interessant del grid de Bootstrap.

A causa del fet que aquesta pàgina està formada únicament per text, volem intentar augmentar la llegibilitat d'aquesta. Existeixen diversos estudis que demostren que l'ésser humà llegeix, amb més comoditat, línies curtes que contenen entre 45 i 75 caràcters.

Per aquest motiu, en aquesta pàgina que tot el contingut principal és text, hem volgut reduir l'amplada màxima utilitzada i en comptes d'utilitzar les dotze columnes que el grid permet, hem decidit utilitzar-ne només vuit.

Per tal que el contingut no quedi descentrat, és a dir, ocupant vuit columnes des de l'esquerra de la fila i deixant-ne quatre en blanc a la dreta, utilitzem una classe especial de Bootstrap que permet deixar columnes en blanc entre diferents blocs de columnes. Aquesta classe, segueix la forma: *col-SIZE-offset-NUMBER* i es regeix per les mateixes regles que les explicades a l'apartat anterior.

Això significa que si declarem una regla per dispositius *md*, aquesta no aplicarà per dispositius de grandària inferior. Garantint, que el text segueixi ocupant tota l'amplada possible, en dispositius més petits.

Així doncs, si volem centrar un contingut de vuit columnes en una fila de dotze columnes, hem de deixar dues columnes en blanc a cada banda del text. La línia de codi que garanteix aquesta visualització és cita a continuació. Fixem-nos, en el fet que primer cal declarar el bloc de columnes i després declarar el *offset*.

Codi 10.8: Separació entre blocs de columnes d'una fila

```
<div class='row'>
    <div id='proposal-box-1' class='col-md-4 col-sm-6'> ... </div>
</div>
```

10.6.9 Els fitxers future-proposals.html i implemented-proposals.html: El grid de Bootstrap III i el component Thumbnail

Aquests dos fitxers són utilitzats per crear el contingut principal de la pàgina propostes i exemples de l'aplicació web. En concret, s'encarreguen de crear les caixes formades per una imatge, un títol i una descripció, que representen una proposta de projecte o un projecte implementat.

Aquestes caixes, originalment tres per fila, s'adapten a dos per fila en tauletes gràfiques i a un per fila en dispositius mòbils. El comportament s'aconsegueix de

forma similar a l'explicat en el fitxer index.html, però en aquest cas, considerant tres formats diferents en comptes de dos. Això es pot aconseguir mitjançant la línia de codi:

Codi 10.9: Multiples configuracions en un bloc de columnes

```
<div class='row'>
    <div id='proposal-box-1' class='col-md-4 col-sm-6'> ... </div>
</div>
```

D'aquesta forma, indiquem que en dispositius mitjans o més grans, volem que la capsa ocupa quatre de les dotze columnes, en dispositius petits, sis columnes i en dispositius extra reduïts, per omissió, dotze columnes.

Les caixes que conformen cada una de les propostes, han estat generades mitjançant el component *Thumbnail* de Bootstrap. Aquests es caracteritzen per utilitzar imatges que s'adapten a la grandària del contingut (*img*) i la possibilitat d'incloure un títol i descripció a cada caixa (*caption*). En el següent bloc de codi, és mostra l'esquelet principal d'una d'aquestes caixes.

Codi 10.10: Exemple de Bootstrap Thumbnail

```
<div class="thumbnail">
    <!-- image -->
    <img src='/images/thumbnails/search-min.png'>
    <!-- title + text -->
    <div class="caption">
        <h3> ... </h3>
        <p> ... </p>
    </div>
</div>
```

10.6.10 El fitxer package.json: Complements de l'aplicació

El fitxer *package.json*, s'utilitza per configurar l'aplicació web quan aquesta és desplegada al núvol. La part del codi més interessant és la que especifica les dependències de l'aplicació i que dictamina els components que seran instal·lats, per tal de garantir el correcte funcionament d'aquesta.

Codi 10.11: Dependències declarades en el fitxer pacakge.json

```
"dependencies": {
    "body-parser": "^1.15.2",
    "bootstrap": "3.3.6",
    "cookie-session": "2.0.0-alpha.1",
    "express": "4.14.0",
    "mustache-express": "1.2.2"
}
```

Els diferents paquets compleixen les següents funcionalitats:

- **Body-parser:** Utilitzat per capturar els paràmetres enviats des del frontal al servidor.

- **Bootstrap:** Com ja s'ha comentat, utilitzat per controlar l'estrucció de les pàgines i la utilització de components genèrics.
- **Cookie-session:** Utilitzat per crear galetes de sessió no editables i signades, per tal d'evitar atacs a la seguretat dels usuaris.
- **Express:** Framework sobre el que es programarà l'aplicació Node.js.
- **Mustache-express:** Càrrega del llenguatge de plantilles pel framework Express.

10.6.11 El fitxer app.js: Funcionament del servidor

El fitxer, *app.json*, s'encarrega de configurar el servidor i gestionar les peticions HTTP, HTTPS i AJAX provinents del client. És un dels fitxers principals de l'aplicació i probablement un dels més interessants.

El fitxer es divideix en diferents seccions:

- Creació de variables
- Configuració del motor d'impressió i carpetes
- Configuració dels complements
- Funcions de redirecció
- Processament de peticions *POST*
- Validació d'identificació
- Configuració del servidor

Creació de variables

En aquesta secció es declarà la utilització dels diferents complements i la creació de l'aplicació mitjançant el framework Express.

També es creen instàncies dels objectes amb funcions de conveniència emmascarats en els fitxers *projectProposals.js*, *pageTitles.js* i *countryParameters.js*.

Codi 10.12: Declaració de variables en el servidor

```
var express = require('express'),
    app = express(),
    path = require('path'),
    mustacheExpress = require('mustache-express'),
    cookieSession = require('cookie-session'),
    bodyParser = require('body-parser');

var projectProposals = require('./assets/js/projectProposals.js');
var projectProposalsIns = new projectProposals();

var pageTitles = require('./assets/js/pageTitles.js');
var pageTitlesIns = new pageTitles();
```

```
var countryParameters = require('./assets/js/countryParameters.js');
var countryParametersIns = new countryParameters();
```

10.6.12 Configuració del motor d'impressió i carpetes

Aquest punt esdevé important, ja que indica a l'aplicació quina mena de fitxers ha de renderitzar. En el nostre cas, calia indicar a l'aplicació que els fitxers a pintar en el client eren del format HTML i que s'utilitzava el motor de plantilles Mustache com a complement.

Codi 10.13: Declaració del motor d'impressió

```
app.set('views', path.join(__dirname, 'views'));
app.use('/assets', express.static(path.join(__dirname, 'assets')));
```

De cara a què el servidor sigui capaç de resoldre on es troben localitzats, els diferents recursos o fitxers, cal també indicar-ne la ruta de forma relativa al servidor. Així doncs, s'emmascaren les carpetes *views*, *assets*, *node_modules* i *images* de l'aplicació, de la següent forma.

Codi 10.14: Configuració de les rutes a les carpetes amb fitxers

```
app.set('views', path.join(__dirname, 'views'));
app.use('/assets', express.static(path.join(__dirname, 'assets')));
app.use('/node_modules', express.static(path.join(__dirname, 'node_modules')));
app.use('/images', express.static(path.join(__dirname, 'images')));
```

10.6.13 Configuració dels complements

En aquest apartat es configuren dos complements. L'encarregat de gestionar les galetes (*cookie-session*) i l'encarregat d'interpretar els paràmetres enviats al servidor des del client (*body-parser*).

El complement *cookie-session* no cal configurar-lo gaire, ja que els paràmetres per defecte ja marquen les galetes a crear com a segures (transmesa si és possible a través de https) i httpOnly (no modificable pel client). Per tant, només hem d'indicar-ne el nom i un parell de claus amb les quals vulguem firmar i xifrar el contingut d'aquestes.

Codi 10.15: Configuració del complement *cookie-session*

```
app.use(cookieSession({
  name: 'session',
  keys: ['misaholdrin', 'tommarvoloriddle']
}));
```

Per altra banda, el complement *bodyParser*, es preparat per rebre paràmetres de l'URL i descodificar paràmetres JSON.

Codi 10.16: Configuració del complement *bodyParser*

```
app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());
```

10.6.14 Funcions de redirecció

Aquesta secció del fitxer s'encarrega de decidir i configurar els fitxers HTML, que han de ser retornats, segons les peticions realitzades pels usuaris al servidor.

La resposta a totes les peticions GET dels clients són solucionades de forma similar. Primer es capturen, després es recullen tots els paràmetres necessaris per emplenar els paràmetres amb Mustache i s'envia el HTML processat cap al client.

Codi 10.17: Respota del servidor davant la petició del recurs ‘/’

```
app.get('/', function(req, res){
  var params = pageTitleIns.getTitle('index');
  res.render('index.html', {
    backgroundImage: params[0],
    highlight: params[1],
    title: params[2],
    titleMobile: params[3],
    subtitleDesktop: params[4],
    subtitleTablet: params[5],
    button: params[6],
    buttonHref: ''
  });
});
```

El codi anterior representa la resposta a la petició d'accés a la pàgina inicial de l'aplicació (/).

Un cop es capture la petició GET, denotada per l'URL ‘/’, es demana a l'objecte Javascript *pageTitleIns*, carregat a l'inici del servidor, els paràmetres a utilitzar per pintar el fitxer *pageTitle.html*. Un cop rebuts aquests paràmetres, es processa el fitxer, *index.html*, amb els paràmetres desitjats i s'envia al client.

Altres peticions GET més complexes, com per exemple, la de pintar els detalls d'una proposta de projecte específica, funcionen de forma similar però carregant més paràmetres a part dels de l'objecte *pageTitleIns*. Aquest cas, conté una excepcionalitat més i és que es capture part de l'URL, tal com s'indica en el següent bloc de codi, per utilitzar-la com a paràmetre.

Codi 10.18: Exemple d'utilització com a paràmetre, d'una part de l'URL

```
app.get('/proposals/:project', function(req, res) {
  var params = pageTitleIns.getTitle('index');
  var proposal = projectProposalsIns.getExample(req.params.project);
});
```

10.6.15 Processament de peticions POST

Les dues úniques peticions POST que l'aplicació està preparada per acceptar i respondre, són les crides AJAX que serveixen per indicar al servidor que l'usuari s'ha identificat o desconnectat de l'API de FamilySearch.

Quan es rep una d'aquestes peticions, s'actualitza la galeta de sessió amb el token retornat per l'API de FamilySearch o se n'esborra el contingut, segons la petició tractada. Un cop processada la informació, s'envia un paràmetre *redirect* cap al navegador, que indica a la capa del controlador, a quina pàgina s'ha de redirigir a l'usuari.

És mostra a continuació, com a exemple, el processament de la petició POST d'identificació.

Codi 10.19: Resposta a la petició AJAX d'identificació

```
app.post('/token/login', function(req, res) {
    req.session.logged = req.session.logged || req.body.token;
    res.end({'redirect' : '/examples'});
});
```

10.6.16 Validació d'identificació

La validació d'identificació és una funció que s'utilitza per comprovar si un usuari disposa dels permisos suficients per visitar les pàgines d'exemples implementats.

Quan el servidor rep la petició de mostrar la pàgina d'exemples o un exemple en concret, realitza una crida a aquesta funció, que resoldrà si la petició ha de ser processada o no. Aquesta funció (*isAuthenticated*), s'insereix com a *middleware* a la capçalera de la petició de la forma següent:

Codi 10.20: Inserció de *middleware* en una petició del client

```
app.get('/examples', isAuthenticated, function(req, res) { ... });
```

La funció, *isAuthenticated*, comprova si la galeta *session* està configurada o no. En cas afirmatiu, es processa la petició inicial de l'usuari i en cas contrari, es mostra la pàgina d'identificació.

Codi 10.21: Comprovació de la galeta *session*

```
function isAuthenticated(req, res, next) {
    if(req.session.isPopulated) next();
    else res.redirect('/login');
}
```

10.6.17 Configuració del servidor

La part de configuració del servidor consisteix, principalment, a indicar quin port ha de ser escoltat. La línia de codi utilitzada, permet alternar entre un valor per

defecte o l'especificat en els paràmetres de configuració de l'entorn que desplegui el servidor.

Codi 10.22: Configuració del servidor

```
var server = app.listen(process.env.PORT || 8080, process.env.IP,
    function () { ... });
```

10.6.18 El fitxer gaTagging.js: Enviant esdeveniments

Aquest fitxer, s'encarrega de configurar els esdeveniments generals que seran enviats a Google Analytics, però l'aspecte més interessant que conté, és la funció genèrica que s'invoca per enviar esdeveniments des de qualsevol punt de l'aplicació.

Els esdeveniments de Google Analytics estan formats per quatre paràmetres: *categoria*, *acció*, *etiqueta* i *valor*, on els paràmetres etiqueta i valor són opcionals. El sistema plantejat, ofereix la possibilitat que el paràmetre, *categoria*, sigui omplert automàticament, amb el path de la pàgina, que dispara l'esdeveniment.

En el bloc de codi que es mostra a continuació es poden observar dues funcions. La funció, *getCategory*, és l'encarregada d'emplenar el camp *categoria* amb la pàgina i la funció, *sendEvent*, és la funció que s'invoca per disparar tots els esdeveniments de l'aplicació web, de forma centralitzada i cridar a la funció, *getCategory* en cas de necessitat.

Codi 10.23: Funcions que controlen l'enviament d'esdeveniments a GA

```
function getCategory() {
    var currentPage = location.pathname.split('?')[0].slice(1);
    return currentPage != '' ? currentPage : 'home'
}

function sendEvent(category, action, label, value) {
    category = category != '' ? category : getCategory();
    ...
    ga('send', 'event', category, action, label);
}
```

10.6.19 El fitxer style.css: Configurant elements i classes pròpies

El fitxer style.css ha estat utilitzat principalment per controlar l'aparença de certs elements mostrats a l'aplicació web.

Els dos elements que han estat més manipulats són la font associada a cada element HTML, pel fet que les fonts per defecte de Bootstrap són bastant simples i manquen de personalitat i la creació d'unes noves classes destinades a controlar la distància entre línies del grid de Bootstrap, que per defecte, mostra les diferents línies pràcticament enganxades i com a conseqüència, la pàgina web no 'respira'.

A continuació, mostrem un exemple d'assignació de font als elements HTML `<h1>..<h6>` i algun exemple dels espaiadors de classe, que hem creat, seguint la nomenclatura Bootstrap.

Codi 10.24: Assignant fonts a elements HTML i creació d'espaiadors

```
// Fonts for html <h1> .. <h6>
h1, h2, h3, h4, h5, h6 {
    font-family: 'Lora', serif;
    font-weight: 400;
    line-height: 1.8em;
}

// Spacers
.xs-buffer { margin-top:10px; }
.sm-buffer { margin-top:20px; }
.md-buffer { margin-top:30px; }
.xl-buffer { margin-top:40px; }
.xxl-buffer { margin-top:50px; }
.fff-buffer { margin-top: 60px; }
```

A part, també s'han creat diferents classes per tal de controlar aspectes, com la dissociació dels controls de les funcionalitats d'exemple, de la seva posició fixa, carregar els diferents imatges de fons en les capçaleres de secció, canvia l'estil dels enllaços URL, controlar la visibilitat inicial de certes zones del HTML que volem mantenir ocultes, estils de les taules, barres de progrés, camps del formulari, etcètera, etcètera.

Es recomana donar un cop d'ull al fitxer original: style.css, per tenir una idea més global de totes les petites configuracions que aquest realitza.

10.7 Certificant l'aplicació amb FamilySearch

Com s'havia comentat en la quarta secció d'aquesta memòria, en la que s'introduïa l'API de FamilySearch, per tal d'obtenir accés a les dades de producció cal certificar les aplicacions.

El procés de certificació, pot ser vist com una validació de l'aplicació per part de l'organització FamilySearch, per assegurar que no es realitza cap operació que pugui afectar al rendiment de l'API, la integritat de les dades o la seguretat del sistema.

En el cas de la nostra aplicació, el conjunt de passos a realitzar per tal de certificar l'aplicació web s'exposen a continuació:

1. Aplicar l'aplicació per certificació des de la secció 'les meves aplicacions' del portal de desenvolupadors de FamilySearch.
2. Completar, signar i retornar l'acord d'afiliació de producte, les regles de seguretat i la petició d'obtenció d'una clau d'accés a producció.
3. Registrar l'aplicació a FamilySearch i monitorar el procés de certificació.

4. L'aplicació serà avaluada de diferents formes segons els diferents certificats que es vulguin obtenir. Aquests certificats van relacionats amb les operacions que realitza l'aplicació contra l'API.

10.7.1 Certificat d'autentificació

El primer certificat que l'aplicació desenvolupada requereix és el relacionat amb el procés d'identificació dels usuaris a l'API de FamilySearch.

La llista de consideracions a tenir en compte per obtenir el certificat són:

- Cada usuari ha d'adquirir un token d'identificació propi per tal de llegir dades de l'arbre familiar de FamilySearch.
- Els tokens d'identificació han de ser protegits. En cas que es vulgui emmagatzemar un token en una galeta del navegador, aquesta ha de ser una galeta segura.
- El tràfic és xifrat mitjançant el protocol SSL des de l'usuari fins a l'API de FamilySearch.
- L'autentificació de l'usuari és completada mitjançant la crida directa als protocols d'identificació OAuth 2 de FamilySearch. No es permet als tercers emmagatzemar informació sobre els noms d'usuari, contrasenyes i identificadors de sessió. En una aplicació web, l'autentificació pot ser realitzar, mitjançant la crida a un pop-up, al protocol d'autentificació de FamilySearch.

Donat que l'aplicació utilitza els serveis oferts pel SDK oficial de Javascript, de cara a l'autentificació, aquesta operació no hauria de presentar molts problemes de cara a obtenir el certificat.

10.7.2 Certificat de lectura

El segon certificat que l'aplicació requereix va relacionat amb les operacions de lectura que realitza contra l'API de FamilySearch. Les principals restriccions a tenir en compte de cara a obtenir el certificat de lectura s'exposen a continuació:

- Procés d'autentificació certificat.
- Demostració de la correcta implementació de diferents funcions de lectura.
- Demostració de l'ús correcte de la cache de FamilySearch.
- Les aplicacions han de guiar a l'usuari a l'hora d'utilitzar les funcionalitats i ajudar-los a superar els possibles errors.
- Les aplicacions han de tenir en compte els estàndards del mercat a l'hora d'evitar atacs a la seguretat mitjançant injeccions de codi i altres vulnerabilitats.
- Complir amb les bones pràctiques de seguretat:

- Només està premés mostrar informació regulada de persones vives a les persones autentificades amb FamilySearch.
- La informació de persones difuntes no regulada pot ser mostrada a qualsevol usuari.
- Les dades locals emmagatzemades durant la sessió en el navegador, han de ser eliminades al final d'aquesta.
- Són permeses les tasques d'elevat temps de processat sempre i quan es compleixin certes regulacions.
- Les aplicacions poden guardar informació genealògica, de persones difuntes, obtinguda a través d'usuaris identificats amb FamilySearch.
- Les aplicacions poden guardar, però no fer pública, informació genealògica de persones vives.
- Les aplicacions poden emmagatzemar els identificadors de les persones, amb dades regulades, però no les dades regulades per se.
- Les aplicacions no poden emmagatzemar relacions concretes que indiquin que una persona va trobar-se, en un lloc concret, en una data concreta.

Un altre cop, pel fet d'utilitzar el SDK oficial de Javascript, molts dels punts principals necessaris per obtenir la certificació de lectura haurien d'estar coberts. Un exemple clar, és el de la correcta implementació de les funcions de lectura. Donat que aquestes no són realment controlades per nosaltres, no hi hauria d'haver cap problema.

Pel que fa a les bones pràctiques de seguretat, s'han utilitzat precaucions per evitar problemes com la injecció de codi a través dels formularis de l'aplicació i la nostra aplicació no emmagatzema cap mena d'informació de l'usuari, ni de les dades retornades, més enllà del token d'identificació.

10.7.3 Procés de certificació

10.8 Google Analytics

Google Analytics és un servei d'analítica web, proporcionat per Google, que s'encarrega de monitorar i reportar dades relatives al tràfic d'una aplicació web o aplicació mòbil.

Mitjançant una implementació relativament simple, és possible emmagatzemar informació relativa a les pàgines visitades, la navegació entre pàgines, els sistemes operatius utilitzats, informació sobre els diferents navegadors, resolucions de pantalla, dispositius mòbils utilitzats i interaccions bàsiques dels usuaris amb les diferents pàgines del web.

Gran part d'aquesta informació és capturada de forma automàtica, per Google Analytics, pel simple fet d'incloure el ‘snippet’ de codi de l'eina a les nostres pàgines del domini web. A canvi, es disposa de moltes variables que poden ser creuades per tal d'analitzar el rendiment del domini web i l'ús que li donen els usuaris.

De totes maneres, no s'espera que la nostra aplicació web disposi de grans quantitats de tràfic i la implementació de Google Analytics no ha vingut donada pel fet de poder analitzar el rendiment tècnic o d'usabilitat de l'aplicació, sinó per poder controlar el funcionament de la integració amb l'API de FamilySearch.

Durant el desenvolupament de l'aplicació, mentre interactuàvem amb l'entorn sandbox de l'API, ens varem adonar que durant diversos dies i inclòs a vegades, períodes de tres o quatre dies, l'entorn no funcionava i les peticions llençades contra l'API eren cancel·lades.

Per tal de poder monitorar en tot moment el funcionament de les interaccions dels usuaris amb l'API, s'ha utilitzat la funcionalitat de Google Analytics que permet enviar esdeveniments personalitzats, que o bé reflecteixen accions dels usuaris o condicions que s'han donat en l'aplicació web o mòbil.

En concret, s'han creat quatre nivells d'esdeveniments diferents per cada una de les funcionalitats principals que interactuen amb l'API de FamilySearch:

- *Formulari incorrecte*: En el cas que un usuari intenti llençar una petició contra l'API i aquesta no s'iniciï perquè el formulari contenia errors, es marca l'intent amb una etiqueta de formulari incorrecte.
- *Petició llençada*: Si la validació de tots els camps és correcta, s'envia un esdeveniment indicant que un intent de connexió amb l'API, s'ha llençat amb el SDK de FamilySearch i els paràmetres d'aquesta petició.
- *Petició rebutjada*: En cas que el SDK no pugui resoldre la petició per qualsevol motiu, s'enregistra un esdeveniment que indica el rebuig de la petició i n'especifica el motiu (Per exemple, timeout, no existeix el recurs, excés de peticions en un període limitat de temps, etcètera).
- *Petició retornada amb èxit*: Quan el SDK processa la petició i retorna resultats, s'envia un esdeveniment d'èxit.

Mitjançant l'existència d'aquests quatre nivells per cada una de les funcionalitats implementades, podrem conèixer l'estat de la connexió amb l'API de cada una d'elles, amb un esforç relativament baix.

La figura 10.4 mostra els diferents quatre nivells d'esdeveniments per la funcionalitat evolució d'esdeveniments. Com es pot veure en la imatge, Google Analytics captura quantes vegades s'ha donat un esdeveniment concret en el període de temps definit i quantes sessions l'han contingut en algun moment donat.

A continuació, també citem els diferents exemples mostrats en la figura [] i hi afegim algun comentari.

- **Formulari incorrecte:** *facts_error_formValidation*. Esdeveniment que s'envia quan la petició no s'ha arribat a processar perquè la configuració dels paràmetres no era correcta.
- **Petició llençada:** *facts_Deaths_1937_1947_unitedstates*. Com es pot veure, quan es llança una petició contra l'API, també capturem el tipus d'esdeveniment

Event Label	Total Events	Unique Events
	4 % of Total: 17.39% (23)	1 % of Total: 100.00% (1)
facts_Deaths_1937_1947_unitedstates	1 (33.33%)	1 (33.33%)
facts_error_formValidation	1 (33.33%)	1 (33.33%)
facts_successful	1 (33.33%)	1 (33.33%)
facts_error_gateway_timeout	1 (33.33%)	1 (33.33%)

Figura 10.4: Exemple d'esdeveniments de la funcionalitat evolució d'esdeveniments

cercat (defuncions), les dates per les quals s'ha cercat (1937-1947) i el país seleccionat (Estats Units).

- **Petició rebutjada:** *facts_error_gateway_timeout*. Aquest és un exemple dels esdeveniments que poden ser retornats quan l'API no ha pogut processar la petició.
- **Petició retornada amb èxit:** *facts_successful*. Esdeveniment que s'envia quan tot ha funcionat com s'esperava i l'API ha retornat resultats.

Els exemples d'esdeveniments anteriors, serveixen per demostrar el potencial que pot arribar a desencadenar una eina d'analítica web com Google Analytics.

A part dels esdeveniments relatius a les funcionalitats implementades, que interactuen amb l'API de FamilySearch, també s'han implementat esdeveniments per controlar els intents d'identificació contra FamilySearch satisfactoris, les peticions de desconexió, posicions dels exemples i propostes de projecte amb les que s'ha interactuat, interaccions amb la barra de navegació, posició de la persona seleccionada en la llista de resultats de la funcionalitat de cerca, etcètera.

Tot i que el conjunt d'informació que s'ha exposat fins ara relativa a Google Analytics podria ser considerada simple, no creiem que sigui l'objectiu de la memòria entrar en molt més detall en les possibilitats d'una eina d'analítica web com aquesta, doncs realitzar una proposta profunda i exhaustiva de les possibilitats de monitoratge d'una web com la implementada, bé podria ser un projecte propi.

10.9 Optimitizació dimatges

Un dels elements més importants quan es desenvolupa una pàgina web és que aquesta carregui de forma ràpida. La diferència entre una pàgina lenta i una de ràpida, s'acaba traduint generalment en una pàgina web sense usuaris o amb usuaris.

No era un objectiu del projecte fer una pàgina web el més optimitzada possible, ja que els coneixements tècnics necessaris requereixen temps per ser adquirits. No

obstant això, sí que es volia realitzar les optimitzacions més típiques, que també solen ser les que endarrereixen més la càrrega de les pàgines web.

Aquest apartat, cobra relativa importància, si tenim present que estem utilitzant un servei d'hostalatge gratuït i que per tant, la velocitat de resposta del servidor, no és de les millors del mercat.

La manca d'optimització en les imatges sol ser un dels factors que més afecta a l'hora de carregar una pàgina web. Els programes de disseny utilitzats per generar imatges de gran qualitat, solen emmagatzemar més informació que la perceptible per l'ull humà, en circumstàncies normals.

És per aquest motiu que optimitzar les imatges, els arxius més grans a descarregar de forma general en una web, esdevé un procés relativament comú.

Per optimitzar les imatges de la nostra aplicació web s'ha utilitzat l'aplicació Optimizilla. Aquesta eina, penjada al núvol, utilitza una combinació de tècniques d'optimització i compressió amb pèrdues, per tal de reduir al màxim el pes d'imatges JPG i PNG, sense reduir el nivell de qualitat perceptible per l'ull humà.

La utilització d'aquesta eina ha reduït, de forma aproximada, el 60-80% del pes de totes les imatges que s'utilitzen en l'aplicació web. Fet considerable, si tenim en compte que no s'ha reduït la qualitat perceptible d'aquestes.

10.10 Hosting de l'aplicació

Com ja s'ha indicat en el primer apartat d'aquesta secció de la memòria, l'aplicació es troba accessible a través del domini:

<https://pfc-family-search.herokuapp.com/>

Hem escollit la plataforma Heroku per desplegar l'aplicació, ja que oferia un ampli ventall d'eines i documentació, que resultaven ideals per un desenvolupador novici de la plataforma Node.js.

No obstant això, no van ser només aquestes facilitats i eines les que ens van portar a desplegar l'aplicació a Heroku, sinó també el fet que el seu pla de hosting gratuït s'ajustava en gran mesura al que el projecte requeria i en cas d'acabar necessitant més, sempre existia la possibilitat d'augmentar la capacitat de processat necessària.

Les característiques que més ens van atreure de la plataforma Heroku es presenten d'una en una en els següents apartats.

10.10.1 Fàcil configuració

Per una aplicació simple com la desenvolupada per aquest projecte, no fa falta canviar res en el codi de la nostra aplicació per tal de desplegar-la a Heroku. Només

hem d'incloure un nou fitxer, a la carpeta arrel del projecte i amb el nom *Procfile*, que indica el tipus d'aplicació i la comanda que ha ser utilitzada per tal d'iniciar-la.

En el nostre cas, el fitxer *Procfile* conte la següent línia de codi:

```
> web: node app.js
```

10.10.2 Fàcil desplegament al núvol

La plataforma Heroku es troba molt ben integrada amb Github, l'eina que hem utilitzat per mantenir sincronitzats els desenvolupaments en les diferents estacions de treball. Gràcies a aquesta integració, Heroku disposa d'una comanda que ens permet afegir un remot, a la carpeta del nostre projecte.

```
> heroku git:remote -a pfc-family-search
```

Un cop tenim el remot de Heroku configurat, de la mateixa forma que podem enviar el codi de la nostra estació de treball al núvol, el podem enviar a Heroku per tal de desplegar la nostra aplicació web. Això és tan fàcil com llençar la següent instrucció.

```
> git push heroku master
```

10.10.3 Entorn de proves local

A part de l'entorn de proves local que podem configurar mitjançant la instal·lació de Node.js en el nostre sistema, Heroku també ens permet simular un entorn de producció Heroku, en l'àmbit local. S'aconsegueix mitjançant la següent instrucció:

```
> heroku local web
```

Tot i que no aporta gaires diferències respecte a desplegar l'aplicació en local mitjançant Node.js de la forma convencional, pot esdevenir útil sota certes circumstàncies.

10.10.4 Decent versió gratuita

Durant tot el procés de desenvolupament, l'aplicació es troava sota el paquet de funcionalitats gratuït ofert per Heroku.

Aquest ofereix les següents funcionalitats:

- Desplegament des de repositoris GIT.
- Actualitzacions automàtiques.

- Auto reparació d'aplicacions.
- Logs del sistema.
- Nombre de processos diferents suportats: 2
- 1000 hores mensuals de *dyno* actius. L'aplicació s'adorm després de 30 minuts d'inactivitat.
- Dominis personalitzables.
- 512MB de RAM

10.10.5 Fàcil escalatge de l'aplicació

En cas que es desitgi millorar la capacitat de concorrència de l'aplicació, per poder rebre i processar més peticions en paral·lel i realitzar més processos diferents al mateix temps, aquesta és fàcilment escalable mitjançant la inclusió de nous *dynos*.

Els *dynos* són els contenidors que executen les comandes dels usuaris. Per la nostra aplicació web, bàsicament es necessiten *dynos* que processin el tràfic HTTP i HTTPS. Gràcies al fet que el nostre servidor és bastant simple (hem posat la lògica d'interacció amb l'API de FamilySearch a la capa del controlador) és probable que no faci falta augmentar el nombre de *dynos* inicial.

De totes maneres, aquest és fàcilment escalable mitjançant una simple comanda, sempre i quan el nostre pla contractat amb Heroku no sigui el gratuït. Si volem augmentar, per exemple, a 2 el nombre de *dynos* disponibles, executaríem la següent comanda:

```
> heroku ps:scale web=2
```

Secció 11

Funcionalitats d'exemple amb FamilySearch

11.1 Introducció a les funcionalitats implementades

En aquesta secció de la memòria seran presentats els diferents exemples implementats, en la nostra aplicació web, que interactuen amb l'API de FamilySearch.

Per cada una de les funcionalitat es proporcionarà una descripció adequada de la seva funció, detalls d'implementació, patrons d'usabilitat i recomanacions i exemples d'utilització si s'escau.

Recordar també en aquest apartat, que publicar tot el codi implementat de les diferents funcionalitats és impossible, ja que només el controlador d'alguna d'elles, arriba a les quasi mil línies de codi.

És per aquest motiu, que en els següents apartats, es mostraran només certes parts del codi, sempre simplificades, que creiem que poden resultar d'especial interès per tal de comprendre les bàsiques sota les quals funcionen les diferents funcionalitats. Com sempre, tot el codi font pot ser trobat a la pàgina de GitHub¹.

11.2 Identificació i desconexió a l'API de FamilySearch

11.2.1 Descripció de les funcionalitats

Les funcionalitats d'identificació i desconexió a l'API de FamilySearch, s'utilitzen exactament pel que el seu nom indica, per permetre als usuaris identificar-se o desconnectar-se de FamilySearch, des de la pàgina d'un tercer.

Els usuaris no disposen de permisos d'interacció amb l'API, a menys que s'identifiquin primer en el sistema. El motiu, és que un cop identificats, cada usuari rep un 'token',

¹<https://github.com/sinh15/pfc-family-search>



Figura 11.1: Pàgina d'identificació

que al ser adjuntat a les crides a l'API, fa que aquestes siguin acceptades.

En l'aplicació web implementada, la possibilitat d'identificar-se amb FamilySearch apareix quan l'usuari intenta accedir a la secció d'exemples o a un exemple concret. La pàgina en qüestió és relativament simple. L'usuari disposa de dues opcions, tornar enredadera o identificar-se amb FamilySearch.

La figura 11.1 mostra la pàgina d'identificació.

Un cop els usuaris es troben identificats, tenen l'opció de disconnectar-se, des de qualsevol pàgina del web, mitjançant un botó situat a la part dreta de la barra de navegació.

11.2.2 Details d'implementació

El client de FamilySearch

Pel fet que es va decidir interactuar amb l'API de FamilySearch, des de la capa Controlador de la pàgina web, ens trobem en la situació d'haver de crear una instància del client cada cop que l'usuari canvia de pàgina.

Això és degut al fet que quan el navegador canvia d'URL, també neteja el conjunt de variables globals declarades en els fitxers Javascript. De totes maneres, això no vol dir que l'usuari s'hagi d'identificar de nou, cada cop que canvia de pàgina.

El fitxer, client.js, és l'encarregat de crear les instàncies del client i gestionar la concordança d'estat, entre la connexió a FamilySearch i la informació emmagatzemada en el nostre servidor. Recordem, que el servidor és el que decideix si l'usuari pot accedir i utilitzar els exemples i que per tant, esdevé important mantenir aquesta

concordança d'estat en tot moment.

A pesar que els usuaris només necessiten estar identificats a les pàgines d'exemples, el fet de voler oferir la possibilitat de desconnexió des de qualsevol punt de l'aplicació, el fitxer *client.js* s'adjunta a totes les pàgines del domini web.

Les instàncies del client FamilySearch són declarades a cada pàgina mitjançant la funció, *new Familysearch()*, proporcionada pel Javascript SDK.

Codi 11.1: Creació d'una instància del client FamilySearch

```
var client = new FamilySearch({
  client_id: 'a02j00000E5DXqAAN',
  redirect_uri: document.location.protocol + '//' + document.
    location.host + '/',
  save_access_token: saveCookie,
  access_token: token,
  auto_expire: true,
  auto_signin: true,
  expire_callback: function(data) { ... },
  environment: 'sandbox'
});
```

Els paràmetres inclosos en la creació de la instància del client, compleixen les següents funcions:

- **client_id:** Número del client, que identifica l'aplicació, a la plataforma de desenvolupadors de FamilySearch. Diferent per cada aplicació.
- **redirect_uri:** URL de redirecció registrada a la plataforma de desenvolupadors de FamilySearch, per indicar el punt de retorn pel procés d'identificació.
- **save_access_token:** Variable que permet emmagatzemar el 'token' retornat per l'API, en el procés d'identificació, en una galeta del navegador. Aquesta permet crear instàncies del client a totes les pàgines del domini web, sense la necessitat d'anar demanant a l'usuari que s'identifiqui. En el nostre cas, es tracta d'un booleà que contindrà el valor false quasi sempre, ja que utilitzem un mètode alternatiu per emmagatzemar el 'token' d'accés, de forma que compleixi amb els requisits de certificació.
- **test_token:** En cas que l'usuari ja es trobi identificat amb FamilySearch, li passem el 'token' emmagatzemant de forma automàtica. Això ens permet no haver de demanar a l'usuari que s'identifiqui a cada una de les pàgines de l'aplicació web.
- **auto_expire:** Booleà que indica si volem que el sistema netegi el 'token' de forma automàtica en cas que aquest quedí invalidat.
- **auto_signin:** Booleà que indica si volem que es demani a l'usuari que s'identifiqui cada cop que intenta realitzar una operació contra l'API sense trobar-se connectat a FamilySearch. En la nostra aplicació, li passem el paràmetre true, de totes maneres, no s'hauria de poder donar el cas en què un usuari pogués llençar una crida contra l'API sense trobar-se identificat. L'utilitzem, simplement, com a mesura de seguretat.

- **expire_callback:** Funció a executar quan el ‘token’ expira. Conté quasi el mateix codi que la funció de desconexió implementada, així que veurem el seu comportament més endavant.
- **environment:** Entorn de l’API de FamilySearch en el que ens volem connectar: ‘production’, ‘sandbox’, ‘beta’, etcètera.

Fins aquí, tot és relativament simple, però on s’emmagatzema el ‘token’ per complir amb les regles de certificació i poder reutilitzar-lo? La resposta és a l’espai local del navegador. Els navegadors moderns permeten escriure en el que es coneix com l’espai local del navegador, el mateix lloc, on els navegadors emmagatzemen les imatges i recursos d’una aplicació web per tal de reutilitzar-los en futures crides, evitant així, haver de descarregar-los de nou i augmentar la velocitat de càrrega.

S’utilitza l’espai local del navegador per emmagatzemar el ‘token’, ja que és un espai inaccessible per tercers i dificulta, en gran mesura, el robament d’identitat.

Al principi del fitxer *client.js* es comprova si el navegador utilitzat per l’usuari suporta l’escriptura a l’espai local. En cas afirmatiu, es carrega el ‘token’ emmagatzema’t, si aquest existeix, en el paràmetre *token*. Altrament, es configura el paràmetre *save_access_token* amb el valor true.

Aquests dos paràmetres, són els mateixos que s’utilitzen en la creació del client, descrita en el bloc de codi anterior. A continuació, mostrem el bloc de codi que configura els paràmetres *save_access_token* i *token*.

Codi 11.2: Configuració dels paràmetres *access_test_token* i *token*

```
if (typeof(Storage) !== 'undefined') {
    token = localStorage.token ? localStorage.token : '';
} else {
    saveCookie = true;
}
```

La funció d’identificació amb FamilySearch

La funció d’identificació es realitza des de la pàgina *login.html* de l’aplicació i el controlador que la gestiona està inclosa en el fitxer *login.js*.

Quan l’usuari pressiona el botó ‘Access FamilySearch’, mostrat a la figura 11.1, el controlador executa la funció *getAccessToken()*, proporcionada pel SDK de FamilySearch. Aquesta funció, obre un pop-up que permet a l’usuari identificar-se i n’espera la resposta de forma asíncrona.

En cas que l’usuari s’identifiqui de forma correcta, s’emmagatzema el *testTokenValue* retornat pel SDK a l’espai local del navegador i és llença la funció *serverLogIn(testTokenValue)*, que s’encarregarà de mantenir la concordança d’estat entre un token vàlid de FamilySearch i el nostre servidor.

Codi 11.3: Petició al SDK de Javascript d’un token d’identificació

```
client.getAccessToken().then(function(testTokenValue) {
    localStorage.token = testTokenValue;
    serverLogIn(testTokenValue);
});
```

La funció *serverLogIn()* està inclosa en el fitxer *client.js*, que recordem, està inclòs a totes les pàgines del web. Aquesta funció, realitza una crida AJAX al servidor passant com a paràmetre el valor *testTokenValue* retornat per l'API. Un cop el servidor ha processat la petició, la funció definida en el paràmetre *success*, redirigeix l'aplicació a la pàgina indicada pel servidor.

Codi 11.4: Crida AJAX al servidor amb el token retornat pel SDK

```
function serverLogIn(apiToken) {
    $.ajax({
        type: 'POST',
        url: '/token/login',
        data: JSON.stringify({token: apiToken}),
        dataType: 'json',
        contentType: 'application/json; charset=UTF-8',
        success: function(data) { ... }
    });
}
```

El servidor, per la seva banda, en rebre la petició AJAX del client, crea una galeta de sessió segura i no modificable pel client amb el valor del paràmetre *testTokenValue* codificat. Mentre aquesta galeta existeixi, el servidor permetrà accedir a l'usuari a les pàgines d'exemples implementats.

Codi 11.5: Petició POST d'identificació processada pel servidor

```
app.post('/token/login', function(req, res) {
    req.session.logged = req.session.logged || req.body.token;
    res.end('{{ redirect : "/examples"}}');
});
```

La funció de desconnexió amb FamilySearch

Aquesta funció s'invoca de forma automàtica quan el 'token' expira o quan l'usuari utilitzà la funcionalitat de desconnexió situada a la barra de navegació.

La funció de desconnexió amb FamilySearch, realitza el procés invers realitzat per les funcions *getAccessToken()* i *serverLogIn()* descrites a l'apartat anterior. Aquesta, invalida el 'token' configurat en la instància del client, elimina el valor contingut a l'espai local del navegador i sincronitza l'estat amb el servidor mitjançant una crida AJAX com la de l'apartat anterior.

Codi 11.6: Cridar AJAX al servidor per indicar que l'usuari s'ha disconnectat

```
function serverLogOut() {
    client.invalidateAccessToken();
    localStorage.removeItem('token');
```

```

$.ajax({
  type: 'POST',
  url: '/token/logout',
  ...
});

```

Per la seva banda, el servidor elimina la galeta de sessió, simplement assignant-li el valor *null* i demana al controlador que redirigeixi a l'usuari cap a la pàgina principal de l'aplicació.

Codi 11.7: Petició POST de desconexió processada pel servidor

```

app.post('/token/logout', function(req, res) {
  req.session = null;
  res.end({'redirect' : '/'});
});

```

11.2.3 Aspectes d'usabilitat considerats

Les funcions de connexió i desconexió realitzen feina sense necessitat d'interacció per part de l'usuari i per tant, no s'han pogut tractar gaires aspectes d'usabilitat. No obstant això, sí que s'ha volgut informar a l'usuari de què s'està esperant la seva identificació amb FamilySearch, quan aquest prem el botó d'identificació.

Quan el pop up d'identificació amb FamilySearch és mostrat, el contingut de la pàgina *login.html* s'esvaeix i apareix la imatge animada, mostrada a la figura 11.2, que indica que s'està esperant una interacció per part de l'usuari. Quan l'usuari acaba el procés d'identificació, aquest és redirigit de forma automàtica a la pàgina d'exemples o a la pàgina principal en cas d'error.



Please complete the Family Search log-in...

Figura 11.2: Imatge animada mostrada mentre s'espera l'identificació de l'usuari

El segon aspecte d'usabilitat considerat, és que el botó de disconnectar-se només apareix, evidentment, si l'usuari es troba identificat.

11.3 Cerca de persones a l'arbre familiar

11.3.1 Descripció de la funcionalitat

La funcionalitat d'exemple de cerca, permet buscar instàncies de persones per tot l'arbre familiar de FamilySearch.

La funcionalitat ha estat habilitada per permetre la utilització de tots els paràmetres de cerca disponibles per l'API i el SDK i que han estat descrits a la secció cinc de la memòria. De totes maneres, per alguna raó desconeguda, sembla que la cerca d'esdeveniments relacionats a familiars de la persona cercada, no acaba de funcionar i de fet, FamilySearch no ofereix aquesta possibilitat de cerca des de la seva pàgina web. En conseqüència, hem decidit desactivar aquests camps fins a comprendre que està passant exactament.

La cerca es realitza amb el paràmetre *count*, que indica el nombre de resultats a retornar, sense especificar. En aquesta situació, l'API retorna quinze persones per defecte i un cop retornats els resultats, el controlador pinta la informació d'aquestes quinze persones en una taula. Aquesta taula recull, per cada persona, informació sobre el seu identificador, nom, data de naixement i data de defunció.

La taula de resultats permet la navegació entre totes les persones que complien les condicions de cerca, carregant-les en blocs de quinze persones. Cal tenir en compte, que per cada acció de paginació, el client ha de realitzar una crida a l'API de FamilySearch i que per tant, pot tardar alguns segons a recarregar-se amb nova informació.

Finalment, la taula de resultats permet la selecció d'una de les persones, per tal de desplegar tota la informació principal que es disposa sobre ella, els seus relatius més propers i informació s'obre l'ascendència i descendència de la persona.

11.3.2 Recomenacions d'utilització

En aquesta funcionalitat no s'ha restringit els camps de dates a només la introducció d'un any. Aquest camp, accepta diversos formats vàlids. Per exemple: '02-02-1807', '2/2/1807', '1807/02/02', '2 Febrero 1807', '2 February 1807', '2 1807', 'February 1807', 1807, etcètera.

No s'ha volgut restringir la possibilitat d'introducció de dates per dotar de més flexibilitat a l'eina, no obstant això, cal tenir en compte que és imprescindible especificar sempre l'any. Si intentem realitzar una cerca especificant el dia, el més, o el dia i el més, però sense la inclusió d'un any, la cerca no obtindrà resultats.

De forma similar, els camps relacionats amb localitzacions accepten una gran diversitat, pel que fa a nivells d'especificació i idiomes suportats, però per tal que la cerca produueixi resultats, cal sempre introduir com a mínim, el nivell de província o estat. De totes maneres, per l'exploració de la funcionalitat, es recomana cercar a escala de país.

11.3.3 Details d'implementació

Abans de comentar en detall alguns aspectes de la implementació, volem indicar que només el fitxer del controlador d'aquesta funcionalitat, està compost per quasi mil línies de codi i que evidentment, és impossible exposar totes les tasques que aquest realitza en aquesta secció.

És per aquest motiu, que només es destaquen les interaccions principals o més importants, de cara al funcionament de la interacció amb l'API i configuració de la pàgina.

El controlador que s'encarrega de gestionar totes les interaccions de l'usuari amb la funcionalitat es troba en el fitxer *search.js*.

Iniciant la cerca de personnes

Quan l'usuari prem el botó de cerca, la primera tasca del controlador és validar que els continguts introduïts en el formulari siguin correctes. En cas que el formulari no compleixi les condicions de cerca, es dispararà un conjunt d'errors informant l'usuari de la informació a corregir i no es llençarà cap crida contra l'API.

Per realitzar aquesta validació, el formulari escapa cada camp que pot ser introduït per l'usuari i comprova si s'ha de realitzar alguna verificació específica del contingut.

Escapar el camp d'un formulari significa codificar certs caràcters especials, com per exemple, '&', '<', '>', '/', etcètera, per tal d'assegurar el seu correcte transport pels URL i evitar atacs malintencionats, que pretenguin trencar el HTML i afectar el servidor, mitjançant injeccions de codi. El fitxer, *formValidation.js*, s'encarrega de realitzar aquestes transformacions i validacions.

Mostrem a continuació, el codi que escapa els caràcters no desitjats. Per escapar un paràmetre, només cal enviar-lo contra la funció *escapeHtml()*.

Codi 11.8: Funció *escapeHtml()* i la variable *entityMap*

```
var entityMap={‘&’: ‘&amp;’, ‘<’: ‘&lt;’, ... ‘/’: ‘&#x2F;’};

function escapeHtml(string) {
    return String(string).replace(/[\&<>"'\\"]/g, function (s) {
        return entityMap[s];
});
}
```

Si no es mostra cap error de validació, significa que el formulari és correcte i compleix amb les condicions de cerca.

Arribats a aquest punt, es prepara l'objecte *params*, que emmagatzema tots els paràmetres de cerca descrits a l'últim apartat de la secció cinc de la memòria, s'eliminen de la crida aquells que no han estat introduïts per l'usuari i es demana la

cerca dels primers quinze resultats mitjançant la crida a la funció *printPersonsToTable(0)*.

La funció printPersonsToTable()

Aquesta funció, acaba de configurar els paràmetres necessaris per poder realitzar la cerca, la llença i en tracte la resposta.

En concret, configura els paràmetres *inici* i *context* i tot seguit, llença la crida asíncrona cap al SDK. Aquests paràmetres, ja han estat descrits en seccions anteriors de la memòria, però recordem que s'encarreguen de delimitar el primer resultat a retornar i a quina cerca es fa referència, en cas de trobar-se demanant més blocs de resultats a l'API de FamilySearch.

Si no es produeix cap error, l'API retornarà els primers quinze resultats i aquests es pintaran sobre una taula. En cas contrari, l'aplicació ensenyarà l'error a l'usuari tot indicant-ne el motiu de fallida.

Codi 11.9: Llençament de peticions contra el SDK i captura d'errors

```
client.getPersonSearch(params).then(function(searchResponse) {
    // Get persons and display id, name, birth date, death date
    ...
})
.catch(function(e) {
    // Display error
    ...
});
```

L'estructura de codi anterior, encarregada de llençar una petició al SDK i capturar possibles errors, és l'estàndard que segueixen totes les crides realitzades a la nostra aplicació web, que interactuen amb el SDK de FamilySearch.

La llista de resultats, que s'obté com a resultat d'una cerca, és mostra a la figura 11.3.

La gràcia de l'operació *printPersonsToTable(pos)*, és que poden reutilitzar-la quan l'usuari vol navegar pels diferents blocs o pàgines de resultats, sense la necessitat de revalidar el formulari de cerca.

Quan el SDK retorna els resultats al controlador, aquest emmagatzema en variables globals, els paràmetres resultats totals, índex del primer resultat mostrat i context de la cerca.

Gràcies a aquestes variables, la funció es pot encarregar de configurar els paràmetres: posició d'inici i context, just abans de realitzar la cerca contra el SDK i per tant, pot ser reutilitzada amb facilitat, conjuntament a l'objecte *params*, per demanar més blocs de resultats d'una cerca realitzada.

Codi 11.10: Actualització i utilització dels paràmetres *count*, *start* i *context*

```
function printPersonsToTable(pos) {
```

Person search results

ID	NAME	BIRTH	DEATH
SGCV-KDB	Pat Smith	1 January 1800	2 January 1880
SGCV-KD5	Pat Smith	1 January 1800	2 January 1880
SGCV-KDN	Pat Smith	1 January 1800	2 January 1880
SGCV-KZD	Pat Smith	1 January 1800	2 January 1880
SGCV-KCT	Pat Smith	1 January 1800	2 January 1880
SGCV-KHW	Pat Smith	1 January 1800	2 January 1880

Figura 11.3: Exemple de resultats d'una cerca a l'arbre familiar

```
// Update start params
params.start = pos;
params.context = context;

// Search with the defined parameters
client.getPersonSearch(params).then(function(searchResponse) {
    // Get parameters
    count = searchResponse.getResultsCount();
    start = searchResponse.getIndex();
    context = searchResponse.getContext();
    ... // tractament dels resultats
});
}
```

Mostrant els detalls d'una persona

La selecció d'una persona, a la taula de resultats originada per la cerca, provoca una segona crida al SDK de FamilySearch en la que s'obté tota la informació disponible sobre la persona seleccionada i els seus familiars.

Com en totes les crides a l'API, en cas d'error, aquest és mostrat en una secció específica. En cas d'èxit, es pinta, en diferents taules, la informació bàsica de la persona, els diferents noms pels quals és coneguda, els esdeveniments relacionats amb la seva vida, informació bàsica sobre els seus pares, parelles i fills, informació de la seva ascendència i descendència, notes de FamilySearch, fonts de dades que verifiquen la informació mostrada i l'historial de canvis realitzat sobre la persona.

Es mostra en la figura 11.4, l'inici de la secció de taules resultant de demanar els detalls específics d'una persona.

La crida a l'API que gestiona aquesta informació és mostra en el següent bloc de codi.

Jaydon M. SMITH details

Basic Information ↑

Display information			
ID	Gender	Lifespan	Living
L5CP-VFN	Male	1807-1882	false
Birth Date	Birth Place	Death Date	Death Place
2 February 1807	Garden City, Rich, Utah	2 February 1882	Garden City, Rich, Utah

Names ↑

http://gedcomx.org/BirthName Preferred			
Full Text	Given Name	Surname	Lang
Jaydon M. SMITH	Jaydon M.	SMITH	

Figura 11.4: Exemples de l'inici dels detalls d'una persona

Codi 11.11: Crida al SDK per obtenir tota la informació d'una persona

```
client.getPersonWithRelationships(personID).then(function(
  personResponse) {
  var mainPerson = personResponse.getPrimaryPerson();
  personDisplayProperties(mainPerson);
  personDisplayNames(mainPerson.getNames());
  personDisplayFacts(mainPerson.getFacts());
  ...
  client.getAncestry( ... );
  client.getDescendancy( ... );
  ...
  mainPerson.getChanges( ... );
});
```

Com s'ha pogut observar en el bloc de codi anterior, per obtenir les diferents peces d'informació, s'ha de navegar per diferents nivells de la resposta. Alguna informació és accessible directament des de l'objecte inicial, mentre que alguns altres paràmetres cal anar a buscar-los a l'objecte específic de la persona. Per altra banda, la informació relativa a l'ascendència i descendència, cal demanar-la de forma explícita al SDK.

Les dades retornades per les crides al SDK, sobre l'ascendència i descendència, esdevenen interessants, ja que la navegació pels resultats es realitza mitjançant la nomenclatura ‘Ahnentafel’ i ‘Aboville’ respectivament.

La nomenclatura ‘Anhentafel’, atorga a la persona sobre la qual se cerca l'ascendència, el nombre 1. El seu pare, rep el número 2 i la mare, el 3. Les regles per calcular els nombres dels pares de qualsevol persona en l'ascendència són:

- **Pare:** nombre de la persona × 2

- **Mare:** nombre de la persona $\times 2 + 1$

Per altra banda, la nomenclatura ‘Aboville’, utilitza una estructura similar a la de les seccions i apartats d'aquesta memòria. Si s'atorga el nombre 1, a la persona sobre la qual se cerca la descendència, s'utilitza per les diferents generacions:

- **Fills:** 1.1 / 1.2 / 1.3 / etcètera.
- **Els fills del fill 1.1:** 1.1.1 / 1.1.2 / 1.1.3 / etcètera.
- **Els fills del fill 1.2 del fill 1.1:** 1.1.2.1 / 1.1.2.2 / etcètera.

Generació de taules dinàmiques amb informació sobre un recurs

Com s'ha especificat en l'apartat anterior, la informació referent als diversos recursos disponibles dels detalls d'una persona, s'imprimeix en diferents taules.

La idea de com representar aquestes taules neix de l'aplicació d'exemple del SDK de Javascript. Mitjançant l'adaptació del codi d'aquest, per ajustar-lo a les necessitats del nostre projecte, s'aconsegueix, mitjançant la combinació de Bootstrap, javascript i jQuery, la generació de taules dinàmiques.

Es requereixen taules dinàmiques, ja que el nombre d'entrades en recursos com l'historial de canvis o el nombre de fills d'una parella, no és estàndard i per tant, s'ha d'adaptar per cada persona en concret.

La generació de les taules es realitza mitjançant la creació de matrius de dades, on cada cel·la és composta per un vector de dos valors. El primer, indica el tipus del camp de la cel·la, mentre que el segon, el contingut.

Mostrem en el següent bloc de codi, les dues primeres fileres de la matriu de dades relativa a la informació bàsica d'una persona.

Codi 11.12: Exemple de les dues primeres files, d'una matriu de dades

```
var displayProperties = createPanelTable('Display information', [
  [
    ['th', 'ID'],
    ['th', 'Gender'],
    ['th', 'Lifespan'],
    ['th', 'Living']
  ],
  [
    ['td', person.getId()],
    ['td', person.getDisplayGender()],
    ['td', person.getDisplayLifeSpan()],
    ['td', person.isLiving()]
  ],
  ...
]);
```

Un cop s'han creat les diferents matrius de dades, s'utilitza la funció *createPanelTable(header, rows)*, per transformar-les en un bloc de codi HTML que pinta la

taules dins d'un panell de Bootstrap. Tot seguit, mostrem el codi que tradueix el bloc principal de files a HTML.

Codi 11.13: Transformació de files d'una matriu en HTML

```
for(var i = 0; i < rows.length; i++){
    var $row = $('<tr>').appendTo($table);
    for(var j = 0; j < rows[i].length; j++){
        $('<' +rows[i][j][0]+ '>').text(rows[i][j][1]).appendTo($row);
    }
}
```

11.3.4 Aspectes d'usabilitat considerats

Llargada del formulari

Aquesta funcionalitat pot arribar a presentar un formulari relativament llarg si considerem tots els camps, en teoria disponibles, de cara a la cerca de persones. Per aquest motiu, s'han dissenyat un parell de funcionalitats que pretenen facilitar la comprensió i utilització del formulari de cerca.

La primera d'elles, és que els blocs del formulari referents a la persona cercada i els seus relatius més propers, poden ser contrets i expandits, mitjançant un clic en la capçalera del bloc.

Per tal d'indicar que es pot interactuar amb aquestes capçaleres, s'ha afegit el signe ‘-’ o ‘+’, a l'esquerra del títol i una icona que també reflecteix l'estat actual, del bloc del formulari, a la dreta. A la vegada, un text en cursiva explica la funcionalitat en la mateixa barra i el color de la barra, canvia en passar el ratolí per sobre i la icona es transforma en una mà d'interacció.

La segona funcionalitat, té l'objectiu de facilitar la iniciació de la cerca. En moltes circumstàncies, els usuaris només voldran emplenar detalls bàsics de la persona principal a cercar i no pas dels seus relatius. En aquestes situacions, sobretot en dispositius mòbils, el botó de cercar quedarà molt allunyat de l'inici del formulari i per tant, pot resultar confús pels usuaris com iniciar la cerca.

Per evitar aquesta confusió, un cop se superen 780 píxels de la posició inicial del formulari de la persona principal i fins que s'arriba a la posició original del botó de cerca, una barra fixa apareix al final de la pàgina amb el botó de cercar i se sobreposa al contingut de la pàgina web. D'aquesta forma, el botó de cercar és accessible en tot moment pels usuaris, sense necessitat d'arribar al final del formulari.

La figura 11.5 mostra els dos estats possibles, de les capçaleres dels blocs del formulari i la barra mencionada, que apareix sobre impressionada al final de la pantalla amb el botó de cerca, quan es compleixen les condicions necessàries.

L'efecte de la barra sobre impressionada amb el botó de cerca, s'aconsegueix mitjançant jQuery i el canvi de classes CSS, en els elements HTML que contenen la barra de cerca. El jQuery s'encarrega d'espiar la posició de l'usuari cada cop que es

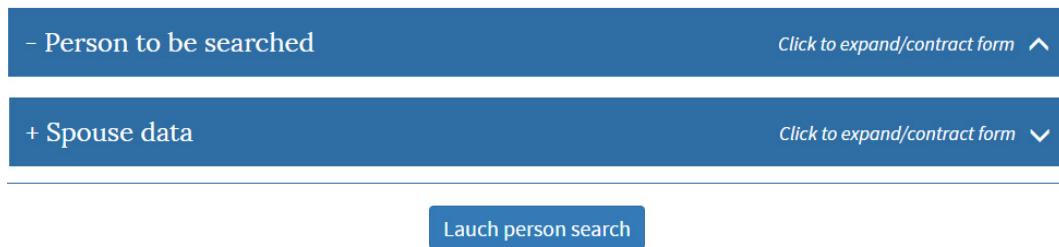


Figura 11.5: Blocs de formulari interactius i sobreposició del botó de cerca

desplaça verticalment per la pàgina web i les diferents classes CSS, fixen o alliberen, la posició del contingut de la barra de cerca.

A continuació, mostrem la classe CSS que converteix la barra de cerca, en una barra de posició fixa, al final de la pàgina web. Les propietats que ho fan possible són la propietat, *position:fixed* i *bottom:0px*.

Codi 11.14: Classe CSS per sobre impressionar el botó de cerca

```
.detached-bottom {
    position: fixed;
    bottom: 0px;
    height: 80px;
    width: 100%;
    margin-top: 0px;
    border-top: 1px solid #2e6da4;
    z-index: 9999;
    background-color: #fff;
    -webkit-transition: all 0.2s linear;
    -moz-transition: all 0.2s linear;
    -o-transition: all 0.2s linear;
    transition: all 0.2s linear;
}
```

Validació del formulari

Com s'ha exposat en la descripció de la funcionalitat, els paràmetres del formulari no estan sotmesos a validacions del contingut. No obstant això, si que es realitza una comprovació abans de llençar la cerca i és que com a mínim, s'ha d'emplenar un camp del formulari, a part del sexe de la persona principal.

En cas de no complir-la, un error es dispara a l'inici del formulari.

Animació de cerca mentre s'espera la resposta del SDK

De la mateixa forma que en l'operació d'identificació, mentre s'espera la resposta del SDK a la cerca de persones, s'informarà a l'usuari d'aquesta situació mitjançant una imatge animada que apareixerà a la zona dels resultats.

Aquest aspecte d'usabilitat s'aplica tant a la cerca inicial de persones, com en la petició d'obtenir els detalls d'una persona en concret.

La figura 11.6 mostra un exemple d'aquesta situació.

Person search results



Figura 11.6: Imatge animada mentre es processa la cerca de l'usuari

Missatges d'error del SDK

En cas que alguna operació del SDK no acabi de la forma prevista, s'ensenya un error indicant que el problema ha recaigut a la banda de FamilySearch i s'indica el missatge d'error retornat des de l'organització.

Per exemple, en aquesta funcionalitat, és normal que certes persones retornades no existeixin realment a l'arbre. L'estil utilitzant per representar l'error és el mateix que pels errors de validació de formulari.

La imatge 11.7 ensenya un d'aquests possibles missatges d'error.

Person search results

FamilySearch *Oops! It seems that the connection with the FamilySearch API didn't go as expected. Please try waiting a few minutes, logging in again or refreshing the page.*

- Not Found

Figura 11.7: Exemple de missatge d'error del SDK

Fletxes d'exploració pels resultats de cerca

S'ha inclòs, just per sobre de la taula de resultats de cerca, una barra de navegació que permet a l'usuari explorar els diferents blocs de persones retornats pel SDK. Recordem, que cada petició retorna un conjunt de quinze persones.

Aquesta barra de navegació consisteix de dues fletxes que permeten navegar endavant i enrere, pels blocs de resultats i un missatge de text que indica quins són els resultats mostrats a la taula, respecte al total de resultats disponibles.

S'ha dotat d'intel·ligència a les fletxes de navegació per assegurar que no s'accedeix a blocs no existents o invàlids.

La imatge 11.8 mostra l'aspecte visual d'aquesta funcionalitat.

Person search results

The screenshot shows a table with four columns: ID, NAME, BIRTH, and DEATH. There are two rows of data. The first row corresponds to the entry 'L5CP-VFN' under 'ID', 'Jaydon M. SMITH' under 'NAME', '2 February 1807' under 'BIRTH', and '2 February 1882' under 'DEATH'. The second row corresponds to the entry 'SGCZ-SKN' under 'ID', 'Pat Smith' under 'NAME', '1 January 1800' under 'BIRTH', and '2 January 1880' under 'DEATH'. Above the table, a navigation bar displays 'Displaying persons: 1-15 of 2162' with arrows for navigating through the results.

ID	NAME	BIRTH	DEATH
L5CP-VFN	Jaydon M. SMITH	2 February 1807	2 February 1882
SGCZ-SKN	Pat Smith	1 January 1800	2 January 1880

Figura 11.8: Fletxes de navegació pels resultats de la cerca a l'arbre familiar

Fletxes de retorn a l'inici dels detalls d'una persona

El conjunt de detalls obtingut sobre una persona, pot acabar significant una alta quantitat d'informació disponible.

Per facilitar el retorn cap a les zones d'inici de la funcionalitat, a l'inici de cada secció dels detalls d'una persona, s'ha inclòs una fletxa per retornar al capdamunt de la secció. La imatge 11.4, d'aquesta mateixa funcionalitat, mostra aquestes fletxes, en el context dels detalls d'una persona.

Navegació vertical animada

Aquesta funcionalitat, s'encarrega d'animar els canvis de posició vertical automàtics, implementats en la funcionalitat. Exemples d'aquests canvis són prémer el botó de cerca, seleccionar una persona de la llista de resultats, utilitzar una fletxa de les seccions dels detalls d'una persona o l'aparició d'un error.

Per evitar confondre a l'usuari amb un canvi sobtat de la pantalla, el trasllat a la nova secció de la pàgina es realitza de forma animada, en una transició d'un segon, independentment de la distància a desplaçar.

Representació de l'estat

Una de les característiques principals de les quals s'ha intentat dotar l'aplicació és la capacitat de representar, en tot moment, l'estat actual de les funcionalitats independentment del que hagi passat anteriorment.

Les diferents micro transicions d'estat, que succeeixen per aquesta funcionalitat i que no han quedat cobertes en les seccions anteriors, es llisten a continuació:

- Quan es prem el botó de cercar, el text d'aquest canvia a 'Searching now...' i passa a un estat de desactivació, que n'impedeix la utilització fins que l'estat actual és resolt. Quan la cerca finalitza o es produeix un error, l'estat del botó torna a la seva normalitat.
- Quan es realitza una nova cerca, els resultats de la cerca anterior (si aquesta

existia), són esborrats per evitar causar confusió.

- Els missatges d'error provinents del SDK o validacions del formulari, desapareixen quan es realitza un nou intent de cerca.
- Quan s'utilitzen les fletxes de navegació pels resultats de la cerca, la taula desapareix i tornar a aparèixer per indicar que els continguts han estat refrescats i s'actualitza el valor del bloc de persones mostrat.
- Quan es selecciona una persona de la taula de resultats, per mostrar-ne els detalls, la secció de detalls pren el títol de ‘Loading information...’, i el canvia a ‘Nom _ persona details’, quan el SDK retorna la informació demandada.
- En seleccionar una persona per mostrar-ne els detalls, quan una altra persona ja havia estat seleccionada amb anterioritat, els detalls de la primera persona seleccionada s'esborran per evitar causar confusió.

11.3.5 Principal interès d'ús

El principal interès d'ús d'aquesta funcionalitat és explorar l'arbre familiar de FamilySearch i observar la diferent informació disponible relacionada a les persones retornades.

11.4 Evolució geogràfica d'un cognom

11.4.1 Descripció de la funcionalitat

La funcionalitat evolució geogràfica d'un cognom permet explorar el nombre de persones nascudes, enregistrades a les bases de dades de FamilySearch, amb un cognom específic, per un conjunt de països seleccionats, en un any o interval de temps definit.

Aquesta funcionalitat explota la funció de conveniència `getResultsCount()`, inclosa en la resposta de cerca de persones del SDK de FamilySearch.

Com ja s'ha comentat en diversos llocs de la memòria, els resultats d'una cerca de persones, retornen el nombre de persones indicades en el paràmetre `count` o quinze per defecte. Com més persones es demanin, més tardarà el SDK a processar la petició.

Tanmateix, independentment del nombre de persones demanades, el SDK disposa d'accés al nombre total de registres que compleixen les condicions de cerca i posa aquest valor a disposició dels desenvolupadors, mitjançant la funció de conveniència esmentada.

D'aquesta forma, podem realitzar una cerca utilitzant el paràmetre `count`, configurat a zero i obtenir així, el nombre total de persones que compleixen les condicions de cerca, mitjançant la funció de conveniència i sense sacrificar velocitat de resposta.

Com que aquesta funcionalitat no pretén accedir als detalls de les persones en cap moment, no importa si aquestes no són retornades.

La cerca permet configurar el cognom, països i any o anys, pels que s'ha de realitzar la cerca. A continuació llistem els paràmetres introduïbles per l'usuari:

- **Cognom:** Qualsevol valor introduït serà donat per vàlid, sempre i quan no es deixa el camp en blanc.
- **Països:** Per seleccionar un país, només cal marcar el checkbox situat a l'esquerra del seu nom. Els països estan agrupats per continents i dins de cada continent, els països es troben ordenats en ordre alfàbetic.
- **Any de naixement:** Indica l'any pel qual es vol llençar la cerca o el primer any de l'interval que vol ser considerat. L'any ha de ser introduït en un format de quatre díigits. Per exemple, 2016.
- **Rang:** El *rang* és un paràmetre opcional. En cas d'omplir-lo, indica quin és l'últim any de l'interval que es vol ser considerat. També ha de ser introduït en format de quatre díigits. Conjuntament, al paràmetre *Any de naixement*, representa el període de temps comprés entre: *Any de naixement-Rang*.
- **Interval:** El paràmetre *interval*, només és necessari si s'ha especificat el camp *Rang*. Aquest paràmetre, serveix per indicar, cada quants anys volem que es realitzi una fotografia, del nombre de persones nascudes amb un cognom determinat, en el conjunt de països seleccionats. Per exemple, si el paràmetre *Any de naixement* és 1920, el paràmetre *rang* és 1940 i *l'interval* és 10, es realitzarà una foto de l'estat, pels anys: 1920, 1930 i 1940.

Dit això, cal tenir en compte, que per cada combinació d'any i país, es realitzarà una crida diferent, al SDK, demanant el nombre d'instàncies de persones nascudes amb el cognom indicat.

S'utilitza aquesta aproximació, ja que l'alternativa consisteix a cercar el nombre de persones nascudes, amb el cognom seleccionat, pel rang de temps desitjat i navegar pels detalls de cada una de les persones de la resposta, per tal de geolocalitzar o cercar, el país de naixement d'aquesta. Aquesta aproximació, resulta bastant inviable si tenim en compte el volum de dades contingut a producció.

Com a últim detall tècnic de la funcionalitat, comentar que s'obté la fotografia cada cert període de temps, en comptes d'agafar totes les instàncies de persones nascudes entre els diferents períodes, a causa d'unes discrepàncies en la forma de realitzar la cerca entre el SDK i la pàgina oficial de FamilySearch.

Tot i que la documentació del SDK indica que permet la utilització de rangs de dates, en el paràmetre data de naixement, a la pràctica, no és així. La utilització d'aquesta opció, en el SDK, es tradueix en l'obtenció de resultats incorrectes.

En utilitzar la funcionalitat, interval de dates, el SDK no retorna resultats. Després d'indagar una mica, es va descobrir que el motiu pel qual el SDK i la web oficial de FamilySearch, tenien comportaments dispers, era per la utilització d'un paràmetre de cerca diferent de cara a la introducció de dates.

El SDK, realitza una crida a l'API de FamilySearch mitjançant el paràmetre especificat: *birthDate*, mentre que la web FamilySearch utilitza el paràmetre *birth_year*. La diferència, resideix en què el paràmetre *birthDate*, en realitat permet la inclusió del dia i mes de naixement en diferents formats, però no rangs de dates; mentre que el paràmetre *birth_year*, només considera els anys i per tant, si suporta la utilització d'intervals de temps.

Així doncs, sembla que el paràmetre *birth_year*, no és suportat pel SDK de forma natural, ni tampoc funciona si aquest és forçat de forma manual des del codi. Per tots aquests motius, es va decidir que la funcionalitat implementada pintes la foto cada cert nombre d'anys, en comptes de recopilar la informació de tots els anys entre els diferents intervals.

Retornant a la funcionalitat, un cop l'usuari llença una cerca, es mostra a l'usuari informació sobre la configuració de la cerca, la duració estimada i el percentatge completat fins al moment.

A mesura que el SDK va retornant dades, es pinten diferents gràfics, que il·lustren, de formes diferents, el nombre d'instàncies del cognom introduït trobades a cada país.

En concret s'han implementat tres tipus de gràfics diferents:

- **Mapa geogràfic:** Aquest mapa del globus terraquí desplegat, emplena, amb diferents tonalitats de color verd, el nombre d'instàncies del cognom cercat pels diferents països. La foscor del color de cada país, indica la quantitat d'instàncies trobades respecte als altres països.
- **Gràfic de barres:** Aquest gràfic, ordena els països cercats, de més a menys instàncies trobades pel cognom seleccionat. Es mostra la mateixa informació que en el mapa geogràfic, però en format de gràfic de barres ordenat.
- **Gràfic de línies:** Només disponible quan la cerca conté dades per més d'un any. Mostra la quantitat d'instàncies trobades, a cada país, per cada any de l'interval. Permet observar l'evolució temporal de cada país respecte als altres.

Un cop finalitza la cerca de dades, es pot navegar pels gràfics dels diferents anys, mitjançant uns controls de navegació específics, situats al capdamunt de la zona de resultats.

11.4.2 Recomanacions d'utilització

Volem recordar en aquesta secció que les crides al SDK de FamilySearch són asíncrones i que per tant, res ens impedeix realitzar deu, cent o mil crides simultànies al SDK i en conseqüència, a l'API de FamilySearch.

Tanmateix, si recordem la funcionalitat de 'throttling', mencionada introduïda anteriorment en aquesta memòria, aquesta impedeix l'abús de la connexió i en cas de fer-ho, les nostres peticions quedarien bloquejades.

Com que la funcionalitat evolució geogràfica d'un cognom, llença una cerca al

SDK per cada país i any de l'interval, s'ha imposat en el codi, de forma manual, una separació de dos segons entre les diferents crides. Això es tradueix, en què el temps total d'execució aproximat per una cerca, és de: $2 \times \text{nombre de països} \times \text{nombre d'anys}$ (segons).

Per aquest motiu, es recomana als usuaris mantenir un nombre baix d'anys i països, en les seves cerques de prova. Recordar també, en aquest punt, que no tots els països disposen del mateix nombre de registres i que per tant, alguns tipus de cerques, són candidates a no obtenir cap resultat.

11.4.3 Details d'implementació

Abans de comentar en detall alguns aspectes de la implementació, volem indicar que el fitxer del controlador d'aquesta funcionalitat està compost per cinc-centes línies de codi i que evidentment, resulta impossible exposar en aquesta memòria, tots els blocs de codi interessants.

És per aquest motiu, que només es destaquen les interaccions principals o més importants, de cara al funcionament de la interacció amb l'API i configuració de la pàgina.

El controlador que s'encarrega de gestionar totes les interaccions de l'usuari amb la funcionalitat es troba en el fitxer *geo-surnames.js*.

Validació del formulari de cerca

A diferència de la funcionalitat de cerca general, aquesta funcionalitat, sí que realitza una validació més exhaustiva dels valors introduïts per l'usuari a través del formulari, ja que una configuració incorrecta d'aquests, resultaria l'obtenció de zero resultats.

Han estat implementades dues menes de validacions diferents. La coneguda com a validació en línia i la validació del formulari quan es prem el botó de cerca.

La validació en línia s'activa quan l'usuari surt d'un camp del formulari i s'aprofita aquest moment, per mostrar el marc del camp, en vermell, si aquest conté algun error, o en verd, si el valor introduït és correcte. L'objectiu d'aquesta validació és facilitar a l'usuari la comprensió dels errors i corregir-los al més aviat possible; reduint així, la frustració en el moment de realitzar la cerca.

Per activar la validació en línia, s'utilitza la funció jQuery *onFocusOut()* en conjunció a les classes de Bootstrap *has-success* o *has-error*, que en ser afegides al camp d'un formulari, en pinten la bora de color verd o vermell de forma respectiva.

Codi 11.15: Activació de la validació en línia

```
$('.form-vali').focusout(function() {
    if(inlineValidation($(this).attr('id'))) {
        $(this).parent().removeClass('has-success');
        $(this).parent().addClass('has-error');
```

```

    }
else {
    $(this).parent().removeClass('has-error');
    $(this).parent().addClass('has-success');
}
});

```

Com es pot observar en el bloc de codi anterior, per tal de comprovar si un camp és vàlid o no, es crida la funció *inlineValidation(param)*, amb l'identificador del camp a comprovar. Aquesta funció es reutilitza tant per la validació en línia, com per la validació en el moment de cerca.

Les regles de validació per cada un dels camps del formulari s'especifiquen a continuació:

- **Cognom:** Si el paràmetre té longitud zero, es mostra un error.
- **Països:** Si no s'ha seleccionat com a mínim un país, es mostra un error.
- **Any de naixement:** Si la longitud és diferent de quatre o no és un número, es mostra un error.
- **Rang:** Si el camp no està buit i la longitud és diferent de quatre o no és un número, es mostra un error.
- **Interval:** Si els paràmetres *any de naixement* i *rang* són diferents, el paràmetre *rang* no és buit i *l'interval* no està especificat o no és un nombre, es mostra un error.

La imatge 11.9, mostra a la vegada, els errors en línia (marc dels camps del formulari en vermell o verd) i la caixa d'errors que informa l'usuari dels errors produïts, de forma més detallada, quan l'usuari prem el botó de cerca amb paràmetres incorrectes.

Search Form

Surname	Birth Year	Range (optional)	Interval
smith	1920	1520	aa

Figura 11.9: Exemples d'error en els formularis de cerca

Si totes les validacions són superades, quan el formulari és enviat cap al SDK, s'escapen els paràmetres llegits de la mateixa forma que s'ha explicat per la funcionalitat de cerca en l'arbre familiar.

Regulació de les crides asíncrones

En cas que es passi el procés de validació descrit en l'apartat anterior, significa que podem llençar la cerca contra el SDK i per extensió, contra l'API de FamilySearch.

Recordant el fet, que ja hem introduït en les recomanacions d'utilització d'aquesta funcionalitat, llençarem una crida al SDK per cada combinació de país i any i aquestes peticions, pel fet de ser asíncrones, no esperaran a obtenir una resposta abans de llançar la següent petició cap al SDK.

Per evitar que FamilySearch ens bloquegi per abús del servei, hem introduït una espera de dos segons entre crida i crida, mitjançant el paràmetre *apiDELAY* i la funcionalitat *setTimeout()* de Javascript. També s'han encapsulat les crides, en una funció, que conté la representació dels paràmetres any (*i*) i país (*k*), executats en aquella iteració.

Codi 11.16: Separació manual de les crides asíncrones al SDK

```
for (var i = 0; i < years.length; i++) {
    ...
    for(var k = 0; k < countries.length; k++) {
        (function(k, i) {
            setTimeout(function() {
                client.getPersonSearch(params).then(function(
                    searchResponse) { ... })
            }, apiDELAY*k+(i*countries.length*apiDELAY));
        } (k, i));
    }
}
```

El codi anterior, permet que totes les crides al SDK siguin configurades a la vegada i de forma quasi instantània, però que gràcies a la funcionalitat *setTimeout()*, s'aniran executant una per una a intervals de dos segons.

El fet que les crides siguin asíncrones, significa que els resultats retornats pel SDK es processaran en un moment incert i en conseqüència, les variables que emmagatzemem les dades retornades, necessiten ser globals per tal de poder ser accessibles des de qualsevol part del codi.

En concret, s'han definit dues variables globals que emmagatzemem la mateixa informació, però de forma diferent. Una pels gràfics del mapa geogràfic i gràfic de barres i una altra pel gràfic de línies.

El motiu principal pel qual s'utilitzen dues estructures diferents és que l'estruatura de dades a passar, a l'API de Google per cada un dels gràfics, és diferent i resulta més còmode disposar de dues estructures. Els paràmetres *i* i *k*, passats a cada crida, s'encarreguen de guardar les dades de cada iteració, al lloc que els hi correspon, de les variables globals.

El primer objecte global, la variable *geomapCountries* és una matriu de dades on cada fila conté les dades d'un any i cada casella de la fila està formada per un vector de dos elements. El codi del país i el nombre de persones que compleixen les

condicions de cerca per aquell país.

Per altra banda, la variable global *linechartRows* és una matriu on cada columna representa un país i cada fila un any de l'interval. Cada casella de la matriu conté només un valor, que indica el nombre de persones que compleixen els criteris de cerca pel país denotat en la columna i l'any denotat per la fila.

A continuació, es mostra el bloc de codi que llança la cerca al SDK i emmagatzema els resultats a les variables globals.

Codi 11.17: Crida al SDK i actualització de variables globals

```
client.getPersonSearch(params).then(function(searchResponse) {
    // Get instances of people with name in country [k]
    var total = searchResponse.getResultsCount();
    ...
    geomapCountries[i].push([countries[k].code, total]);
    linechartRows[i].push(total);
    ...
})
```

Impressió dels gràfics

A causa de la funcionalitat, resulta fàcil, que a la mínima que es realitza una cerca interessant, aquesta tardí més d'un minut en completar-se.

Per tal de transmetre la sensació de moviment i que el sistema està treballant, hem introduït a la funcionalitat la capacitat d'anar pintant els gràfics de cada any, a mesura que les dades van quedant disponibles. És a dir, que no cal esperar fins al final de la cerca per poder començar a veure resultats.

Per aconseguir aquest efecte, disposem de variables globals que comptabilitzen el nombre de cerques que ja han retornat del SDK i cada cop que les dades d'un any són completades, es pinta el mapa geogràfic i el gràfic de barres relatiu a l'any. Un cop es completen les dades de tots els anys, es pinta el gràfic de línies.

Els gràfics es pinten mitjançant la crida a l'API de GoogleCharts. Per pintar qualsevol gràfic sempre se segueix un procés similar:

1. Transformació de les dades a un format que Google utilitzarà després per crear el gràfic.
2. Creació del tipus de gràfic desitjat, mitjançant una crida a l'API de Google i selecció del contingut HTML en el qual volem inserir el gràfic.
3. Renderització del gràfic en el HTML.

Pel mapa geogràfic, gràcies a la forma en què s'han anat emmagatzemant les dades, el procés és relativament simple. Recordem que cada fila de la matriu *geomapCountries*, representa els valors d'un any i cada columna, un país diferent.

Codi 11.18: Creació del mapa geogràfic

```

function printGeomap(i) {
    var geomapData = google.visualization.arrayToDataTable(
        geomapCountries[i]);
    geomap = new google.visualization.GeoChart(document.getElementById(
        'geomap'));
    geomap.draw(geomapData, geomapOptions);
}

```

Per representar el gràfic de barres, el procés és molt similar al del mapa geogràfic, però primer realitzem un petit tractament de les dades per tal d'ordenar els països de més a menys instàncies del cognom trobades i obtenir així, una representació més clara.

Codi 11.19: Creació del gràfic de barres

```

function compareCountries(a, b) {
    if(parseInt(a[1]) < parseInt(b[1])) return 1;
    else if(parseInt(a[1]) > parseInt(b[1])) return -1;
    else return 0;
}

function printBarchart(i) {
    // Sort data
    var barchartCountries = geomapCountries[i];
    var first = barchartCountries.splice(0, 1);
    barchartCountries.sort(compareCountries);
    barchartCountries.unshift(first[0]);

    // transform and plot
    var barchartData = google.visualization.arrayToDataTable(
        barchartCountries);
    barchart = new google.charts.Bar(document.getElementById('barchart'));
    barchart.draw(barchartData, barchartOptions);
}

```

Finalment, el gràfic de línies també resulta bastant simple de pintar, ja que gràcies a la forma en què s'han emmagatzemat les dades, no fa falta realitzar cap tractament especial abans de pintar-les.

Codi 11.20: Creació del gràfic de línies

```

function printLinechart() {
    ...
    linechartData.addRows(linechartRows);
    linechart = new google.charts.Line(document.getElementById(
        'linechart'));
    linechart.draw(linechartData, linechartOptions);
    ...
}

```

Las figures 11.10, 11.11 i 11.12, mostren una visualització dels diferents gràfics disponibles.

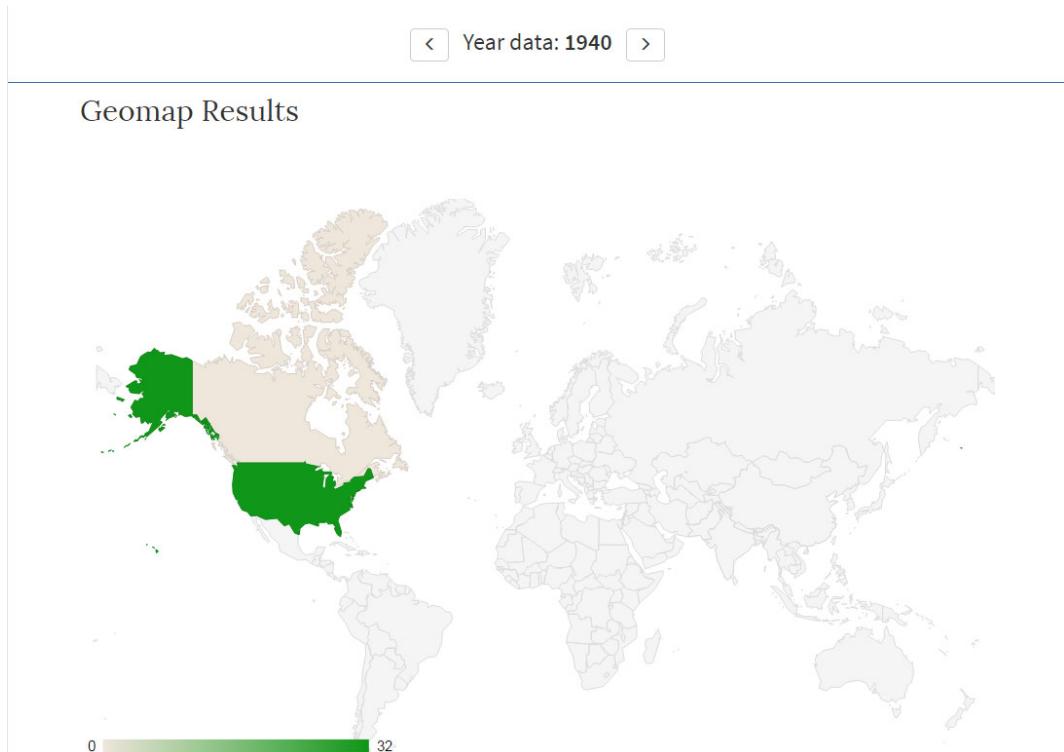


Figura 11.10: Mapa geogràfic

Creació de les caselles de selecció per cada continent

Intentar crear la secció de caselles de selecció (checkboxes), de forma manual, en el HTML, hauria estat una bogeria. No només per l'elevada quantitat de feina manual, sinó també per la poca impracticabilitat d'aquesta. La creació d'aquestes seccions, s'ha realitzat mitjançant la creació d'HTML dinàmic de Mustache.

El servidor, mitjançant Mustache, utilitza la informació continguda en el fitxer *countryParameters.js*, per crear un procés iteratiu que genera totes les caselles necessàries per a cada continent.

Per exemple, pel continent nord-americà, s'utilitza el següent bloc de codi. Recordem que les etiquetes `{{#northAmerica}}` i `{{/northAmerica}}`, indiquen que es vol replicar el HTML situat entre elles per cada element del vector *northAmerica*.

Codi 11.21: Caselles de selecció creades mitjançant Mustache

```

{{#northAmerica}}
  <div class='col-md-3 col-sm-6 col-xs-12'>
    <label class='checkbox-inline'>
      <input type='checkbox' id='{{code}}' ... > {{name}}
    </label>
  </div>
{{/northAmerica}}

```

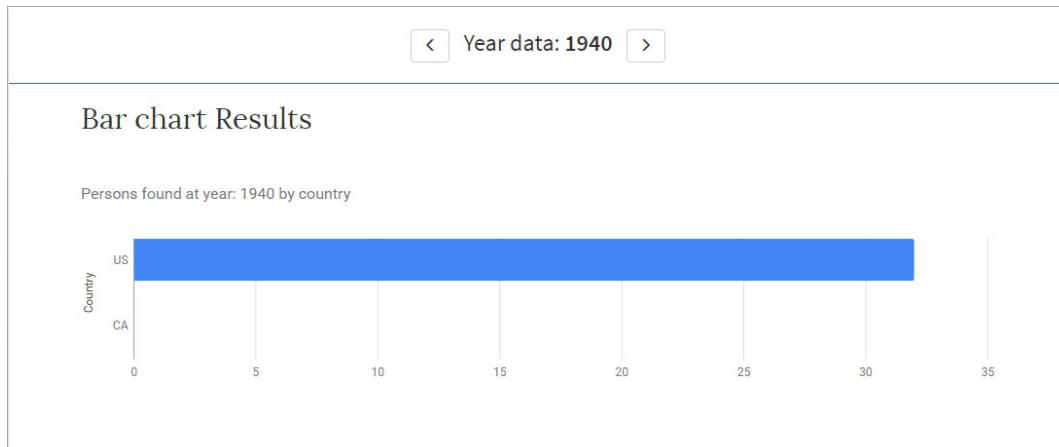


Figura 11.11: Gràfic de barres

D'aquesta forma, per cada país contingut en l'objecte *northAmercia*, es representa el nom del país (`name`) i s'emmagatzema el codi d'aquest en el camp ID (`code`), per tal de ser utilitzat com a paràmetre en la creació del mapa geogràfic.

La figura 11.13 mostra un exemple de com queden disposades les diferents caselles de selecció.

11.4.4 Aspectes d'usabilitat considerats

Aquesta funcionalitat aprofita molts conceptes d'usabilitat explícits en la funcionalitat de cerca a l'arbre familiar i que per tant, no repetirem en aquesta secció, però si anomenarem.

- La funcionalitat també implementa la interacció sobre certes capçaleres del formulari, per tal expandir o contraure parts d'aquest. També utilitza una barra sobreposada al final de la finestra, per facilitar l'accés al botó de cerca.
- Els missatges d'error que puguin aparèixer en prémer el botó de cerca o causats per la fallida del SDK, es representen de la mateixa forma que els apartats de la funcionalitat anterior.
- La navegació vertical animada, també forma part d'aquesta funcionalitat, quan l'usuari realitza interaccions que canviem la seva posició en la pàgina.

A continuació se citen alguns aspectes d'usabilitat propis d'aquesta funcionalitat.

Impressió dels gràfics

Aquesta funcionalitat, disposa d'una barra de navegació, que permeten canviar l'any sobre el qual els gràfics mostren informació. Aquesta barra, es troba a l'inici

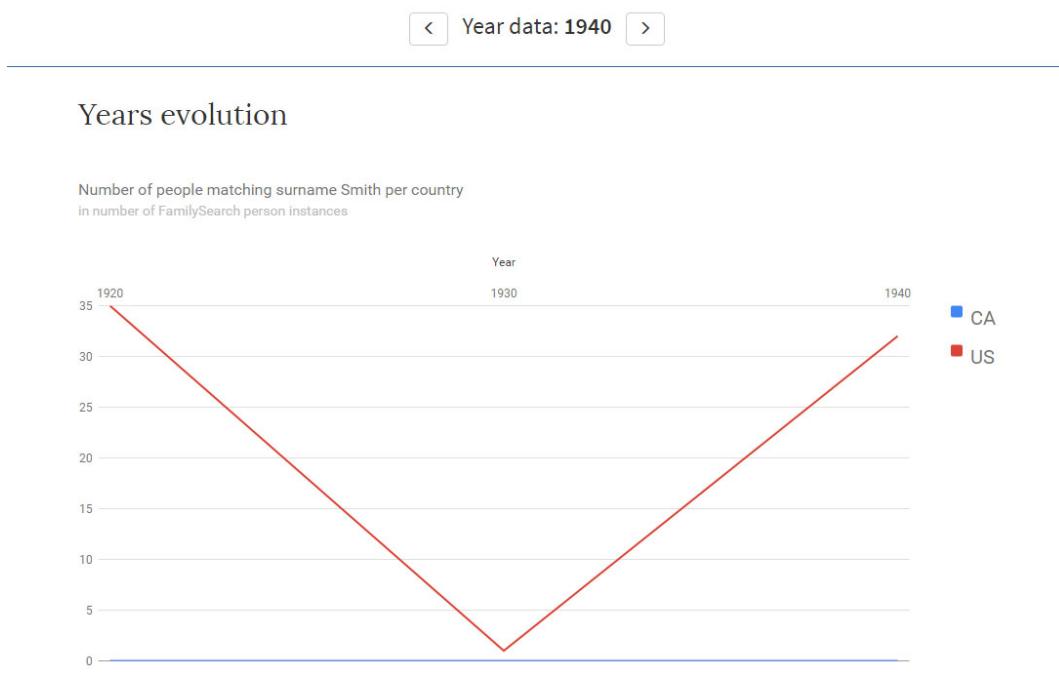


Figura 11.12: Gràfic de línies

de la zona de resultats i es manté fixa i sobreposada als altres elements del HTML, si s'arriba a certa profunditat en la pàgina web.

Aquesta barra de navegació, cobreix un doble objectiu. Primerament, informar a l'usuari sobre a quin any pertanyen les dades dels gràfics pintats i en segon lloc, permetre la navegació a través de les dades dels diferents anys, si aquestes existeixen.

La navegació a través dels gràfics de diferents anys, només es troba disponible, si s'han demanat dades per més d'un any. A més a més, s'han creat regles per impedir l'accés a gràfics que no existeixen, a través d'aquests controls.

Les anteriors imatges 11.10, 11.11 i 11.12, mostren aquesta barra de navegació.

Progressió de la cerca

Un dels aspectes d'usabilitat més interessants d'aquesta funcionalitat és la secció dedicada a mostrar la progressió de la cerca. Si ja havíem comentat que a mesura que es reben les dades dels diferents anys, aquestes són representades en els gràfics pertinents, aquesta secció de la pàgina compleix un rol similar, però més informatiu.

En el moment que es llança una cerca, l'usuari és transportat a aquesta secció. Durant el temps de cerca, aquesta secció mostra el cognom pel qual s'està realitzant la cerca, la duració estimada calculada mitjançant la fórmula: $númeroPaïsos \times númeroAnys \times apiDELAY$, la informació sobre el país i any dels quals s'està esperant la resposta per part del SDK i una barra del progrés actual, respecta el total estimat.

Search Form

Surname	Birth Year	Range (optional)	Interval
smith	1920	1940	10

- North American countries
Click to expand/contract form ▾

- Antigua and Barbuda
- Canada
- Dominican Republic
- Haiti
- Nicaragua
- Saint Vincent and the Grenadines
- Bahamas
- Costa Rica
- El Salvador
- Honduras
- Panama
- Trinidad and Tobago
- Barbados
- Cuba
- Grenada
- Jamaica
- Saint Kitts and Nevis
- United States
- Belize
- Dominica
- Guatemala
- Mexico
- Saint Lucia

Figura 11.13: Caselles de selecció creades amb Mustache

Una cosa que cal tenir en compte és que les crides al SDK són asíncrones i que per tant, no està garantit que el retorn d'aquestes segueixi el mateix ordre que l'ordre d'enviament. Això significa, que potser, el bloc HTML que mostra la progressió de la cerca, pot indicar que s'estan esperant dades que ja han arribat o viceversa. De totes maneres, s'espera que en la gran majoria dels casos, hi hagi una correlació entre l'ordre d'enviament i retorn de les peticions al SDK.

En qualsevol cas, el fet més important relatiu a la barra de progrés, és que aquesta arriba al 100% quan s'han processat totes les crides a l'API i que cada crida emmagatzemi el resultat a les caselles de la matriu que li pertoca. Recordem que això és aconseguit gràcies als paràmetres *i* i *k*, encapsulats en les crides.

Quan la cerca és completada, canvia l'estat dels components de la secció i s'indica que la cerca ha estat completada pel cognom especificat, s'elimina el temps estimat de finalització i és substituït per una indicació sobre la localització dels resultats i s'indica el nombre total de països i anys cercats. L'efecte de moviment en la barra de progressió, també és eliminat, per tal d'evitar confondre a l'usuari.

La imatge 11.14 mostra dos estats diferents de la secció progressió de la cerca.

Representació de l'estat

Una de les característiques principals amb les quals s'ha intentat dotar l'aplicació és la capacitat de representar, en tot moment, l'estat actual de les funcionalitats

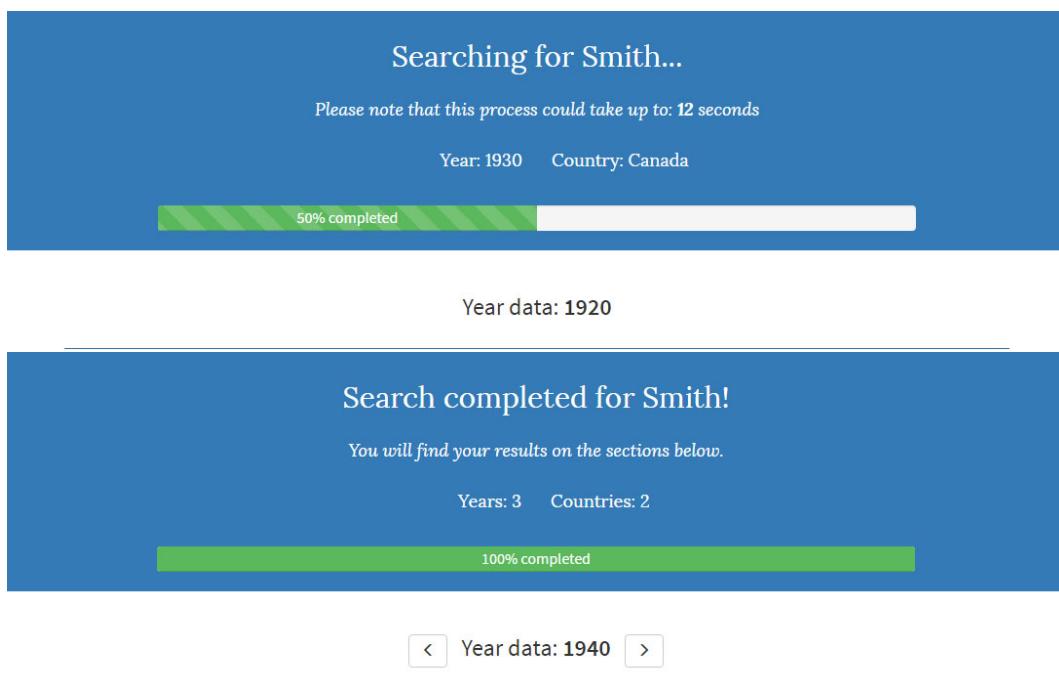


Figura 11.14: Exemples de la secció progressió de la cerca

independentment del que hagi passat anteriorment.

Les diferents micro transicions d'estat que succeeixen per aquesta funcionalitat i que poden no haver quedat cobertes en les seccions anteriors es llisten a continuació:

- Quan es prem el botó de cerca, el text d'aquest canvia a 'Searching now...' i passa a un estat de desactivació que n'impedeix la utilització fins que l'estat actual és resolt. Quan la cerca finalitza o un és produïx un error, l'estat del botó torna a la seva normalitat.
- Quan es realitza una nova cerca, els resultats de l'anterior s'amaguen per no causar confusió.
- Els missatges d'error provinents del SDK o errors de validació del formulari, desapareixen quan es llança una nova operació de cerca.
- Quan s'utilitzen les fletxes de navegació, de la barra de navegació, els gràfics apareixen i desapareixen per tal de fer palpable que aquests han estat refrescats.
- S'ha inclòs, dins de cada continent, la possibilitat de seleccionar o desseleccionar tots els checkboxes, mitjançant un botó.
- El gràfic de línies només es mostra si existeix més d'un any de dades, evitant d'aquesta forma la redundància amb el gràfic de barres, en el cas que només s'hagin consultat dades per un any concret.
- El gràfic de barres ordena els països de més instàncies a menys, per facilitar-ne la comprensió i obtenir una millora visual.

11.4.5 Principal interès d'ús

Aquesta funcionalitat, cobra un especial interès de cara a l'estudi de tendències d'emigració importants, donades per circumstàncies històriques.

També hem volgut implementar aquesta funcionalitat perquè els usuaris poguessin observar, com les instàncies d'un cognom en concret, poden ser trobades en infinitat de països diferents i que cada un de nosaltres, probablement, tinguem orígens completament dispersos.

La funcionalitat evolució geogràfica d'un cognom, pretenia ser només un petit test de la potencialitat dels estudis que es poden realitzar, mitjançant la combinació de dades genealògiques, amb agrupacions per país o comarca. Com s'ha pogut veure en la secció de propostes de projecte, aquesta proposta ha estat ampliada i diversificada, per donar pas a un parell de projectes, que creiem podem ser d'interès pels futurs estudiants.

11.5 Evolució temporal d'esdeveniments

11.5.1 Descripció de la funcionalitat

La funcionalitat evolució temporal d'esdeveniments, permet als usuaris explorar el nombre d'instàncies de naixements, casaments i defuncions, enregistrades a les bases de dades de FamilySearch, per un país, al llarg d'un període d'onze anys.

Aquesta funcionalitat, probablement la més simple de les tres implementades, doncs reutilitza molta part del codi de les funcionalitats anteriors, és també una de les més interessants de cara a les diferents utilitzacions que si li poden donar.

Aquesta funcionalitat, de la mateixa forma que l'evolució geogràfica d'un cognom, utilitza la funció de conveniència del SDK, `getResultsCount()`, per millorar l'eficiència en l'obtenció de resultats i per tant, no entrarem a enumerar els seus avantatges una altre vegada.

La funcionalitat, llançarà un total d'onze crides al SDK, una per cada any de l'interval a considerar.

L'evolució temporal d'esdeveniments, permet a l'usuari configurar els següents tres paràmetres:

- **Esdeveniment:** A seleccionar entre naixements, casaments o defuncions.
- **Localització:** Localització en la qual seran cercades les instàncies de l'esdeveniment seleccionat.
- **Any central:** Any central, del període d'onze anys, pel que es realitzarà la cerca. La funcionalitat genera l'interval +/- 5 anys, respecta l'any introduït. Així, per exemple, si introduïm l'any 1942, el rang d'anys utilitzat serà 1937-1947, ambdós inclosos.

Un cop el SDK retorna resultats per totes les consultes enviades, es pinta un gràfic de línies que mostra l'evolució del nombre d'instàncies de *esdeveniment*, trobades al llarg del període seleccionat.

11.5.2 Recomanacions d'utilització

Aquesta funcionalitat, no restringeix el nivell de localitzacions a utilitzar, de forma contrària a la funcionalitat d'expansió geogràfica d'un cognom i replica, de fet, el funcionament de les localitzacions que s'introduïen en la funcionalitat de cerca.

El paràmetre localització, per tant, accepta una gran diversitat de nivells d'especificació diferents, però per assegurar que la cerca produeix resultats, cal sempre introduir, com a mínim, el nivell de província o estat. De totes maneres, per l'exploració d'aquesta funcionalitat, es recomana introduir també el país o millor encara, cercar directament per un país.

Una segona consideració a tenir en compte és que les bases de dades de FamilySearch no contenen, per tots els països, localitzacions i anys, la mateixa quantitat de registres. Per tant, utilitzar la funcionalitat en regions i períodes de temps rics en registres, produirà, amb alta probabilitat, resultats més interessants.

11.5.3 Details d'implementació

Abans de destacar els details d'implementació d'aquesta funcionalitat, volem comentar, de la mateixa forma que per les dues funcionalitats anteriors, que només el controlador que gestiona la funcionalitat, està format per tres-centes línies de codi i en conseqüència, manca de sentit intentar representar totes les funcionalitats i aspectes d'aquesta en la memòria.

El controlador encarregat del funcionament de l'evolució temporal d'esdeveniments és el fitxer *facts.js*.

Validació del formulari de cerca

Aquesta funcionalitat, utilitza el mateix sistema de validació en línia i validació en el moment de cerca, explicada en detall per la funcionalitat anterior, evolució geogràfica d'un cognom.

De la mateixa forma que les dues funcionalitats anteriors, aquesta també es capa els valors obtinguts del formulari, abans d'enviar-los al SDK, per tal d'evitar injeccions de codi i atacs al sistema.

Les regles de validació per cada un dels camps del formulari es llisten a continuació:

- **Esdeveniment:** En principi no pot existir un estat en què cap dels esdeveniments es trobi seleccionat, però en cas de forcar-ho per edició del HTML, es

dispara un error en el cas que cap tipus d'esdeveniment estigui seleccionat.

- **Localització:** Qualsevol localització és acceptada sempre i quan el camp no es deixi en blanc. Suggerim des d'aquesta part de la memòria, la utilització d'un país com a paràmetre de proves i que es tinguin presents les consideracions d'utilització descrites en l'apartat anterior.
- **Any central:** Aquest camp és acceptat com a vàlid, si el valor introduït té longitud quatre i és un número.

Per observar el resultat visual de la validació en línia o validació en el moment de cerca, es pot observar la figura 11.9 de la funcionalitat anterior.

Regulació de les crides asíncrones

De la mateixa forma que la funcionalitat expansió geogràfica d'un cognom, aquesta funcionalitat llança múltiples crides asíncrones contra el SDK de FamilySearch.

En concret, aquesta funcionalitat, sempre llença onze crides que per evitar ser bloquejades per la funcionalitat de ‘throttling’ de l’API de FamilySearch, han estat serialitzades imposant una pausa de dos segons i mig entre crida i crida.

Aquesta serialització s’ha aconseguit, de forma anàloga a la descrita en detall a la funcionalitat anterior, mitjançant la funció `setTimeout()` de jQuery i el paràmetre `apiDELAY`, que s’encarrega d’indicar l’interval d’espera entre les diferents crides. També s’han encapsulat les crides en una funció, que emmagatzema com a paràmetre a quina de les onze crides correspon la iteració.

A continuació, mostrem el bloc de codi reduït, que representa la serialització de les crides a l’API.

Codi 11.22: Separació manual de les crides asíncrones al SDK

```
for(var i = 0; i < 11; i++) {
    ...
    (function(i) {
        setTimeout(function() {
            ...
            client.getPersonSearch(params).then(function(
                searchResponse) {
                    var total = searchResponse.getResultsCount();
                    linechartRows[i].push(String(firstYear+i));
                    linechartRows[i].push(total);
                    ...
                });
            }, apiDELAY*i);
        }(i));
}
```

Aquestes crides retornen al controlador una per una i encara que no podem garantir l’ordre de rebuda, s’espera que en la gran majoria dels casos, aquest es correspongui a l’ordre d’enviament. De totes maneres, gràcies al paràmetre *i*, em-

paquetat en cada una de les crides, podem emmagatzemar les dades retornades pel SDK al lloc que els hi correspon de la matriu *linechartRows*.

La variable global encarregada de guardar els resultats de les diferents crides al SDK és la mateixa que la utilitzada pel gràfic de línies de la funcionalitat expansió geogràfica d'un cognom, però que en aquest cas, consisteix en una matriu d'una sola columna, on la columna representa la localització cercada i cada fila, el valor per un any concret.

Explicar que utilitzem una matriu d'una sola columna, en comptes d'un vector, perquè aquest és el format que espera l'API de Google, si volem que el gràfic representi la quantitat d'instàncies trobades en l'eix vertical i els anys a l'eix horitzontal.

La línia de codi responsable de guardar els valors per cada crida a l'API, ha estat mostrada en el bloc de codi anterior i en concret, a les files vuit i nou. Recordem, que el paràmetre *i* fa referència a la iteració del bucle executada i per extensió, a quin any fan referència les dades respecta els onze anys de l'interval cercat.

Impressió del gràfic

La funcionalitat evolució temporal d'esdeveniments, utilitza una variable global per comptar quantes iteracions de les enviades al SDK, han estat ja retornades per aquest (*yearsConsulted*).

En el moment en què una petició retorna del SDK i aquesta variable, passa a tenir el valor onze, significa que ja s'han rebut totes les dades i que per tant, aquestes poden ser impresees. Les línies de codi que realitzen la gestió d'aquesta variable, dins de les crides al SDK, es mostren a continuació.

Codi 11.23: Gestió de la variable *yearsConsulted*, per controlar el final de la cerca

```
client.getPersonSearch(params).then(function(searchResponse) {
    yearsConsulted = yearsConsulted + 1;
    linechartRows[i].push(String(firstYear+i));
    linechartRows[i].push(total);
    if(yearsConsulted == 11) printLinechart();
});
```

Per altra banda, la funcionalitat encarrega d'imprimir el gràfic de línies en el HTML, segueix els mateixos detalls d'implementació, que els mostrats en la funcionalitat evolució geogràfica d'un cognom.

Un exemple del gràfic imprès per aquesta funcionalitat, pot ser vist en la figura 11.15.

11.5.4 Aspectes d'usabilitat considerats

Aquesta funcionalitat, com és normal, reutilitza molts dels aspectes d'usabilitat ja esmentats i explicats en detall per les funcionalitats anteriors. En conseqüència, molts d'aquests no seran explicats al detall, però sí que els anomenarem per tal de deixar-ne constància.

Yearly deaths evolution

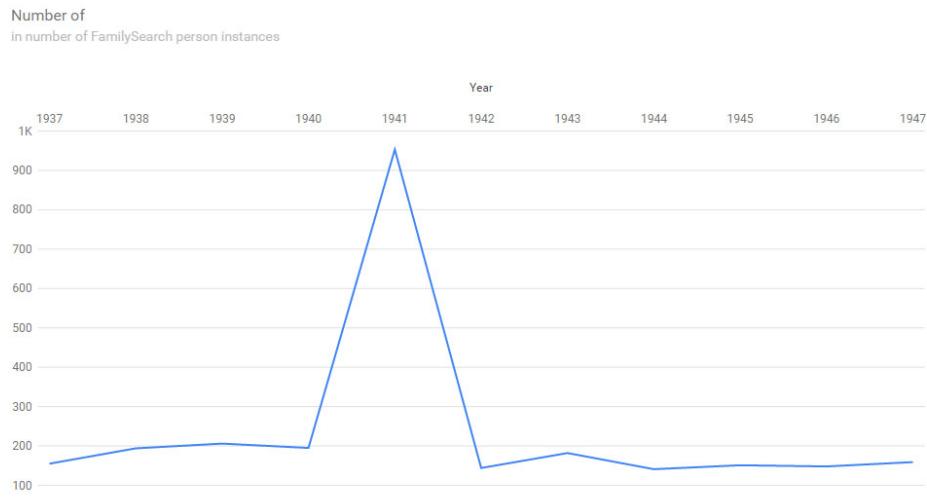


Figura 11.15: Gràfic de línies de la funcionalitat evolució d'esdeveniments

- Els missatges d'error, que puguin aparèixer en prémer el botó de cerca o causats per la fallida del SDK, es representen de la mateixa forma que l'explicada en la funcionalitat de cerca a l'arbre familiar.
- La navegació vertical animada també forma part d'aquesta funcionalitat, quan es donen situacions que alteren la posició de l'usuari en la pàgina. Per exemple, prémer el botó de cerca o l'aparició d'un error.

Progressió de la cerca

La progressió de la cerca és una secció del HTML molt similar al de la funcionalitat expansió geogràfica d'un cognom.

La finalitat d'aquesta secció és proporcionar a l'usuari un indicar del progrés de cerca realitzat fins al moment i el temps estimat per la finalització d'aquesta. L'objectiu, reduir la frustració de l'usuari i ensenyar de forma transparent, la feina ja realitzada.

No entrarem en els detalls més tècnics d'aquesta funcionalitat, pel fet que aquests ja han estat exposats en la funcionalitat anterior.

La informació que es mostra a l'usuari en aquesta funcionalitat és el tipus d'esdeveniment pel qual s'està realitzant la cerca, el país introduït i l'any central sobre el qual es realitza la cerca.

Aquesta funcionalitat, també mostra una barra de progrés que va completant-se a intervals de 9%, a mesura que el SDK va resolent les diferents peticions enviades

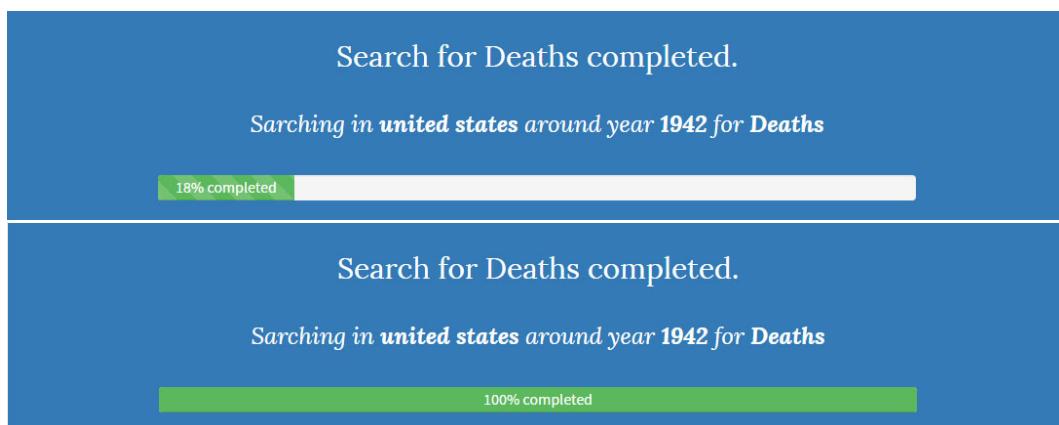


Figura 11.16: Exemples de la secció progressió de la cerca

des del client.

Un cop ha finalitzat la cerca i el gràfic de línies ha estat pintat, el nou estat s'indica en aquesta secció, mitjançant un canvi en el text i l'eliminació de l'efecte de moviment en la barra de progrés.

La figura 11.16 mostra els dos estats diferents d'aquesta secció de la funcionalitat.

Representació de l'estat

Les funcionalitats de l'aplicació es caracteritzen per intentar mantenir una representació de l'estat, fidel a cada moment, independentment del que hagi pogut passar amb anterioritat.

Les diferents micro transicions d'estat que succeeixen per aquesta funcionalitat i que poden no haver quedat cobertes en les seccions anteriors, es llisten a continuació:

- Quan es prem el botó de cerca, el text d'aquest canvia a 'Searching now...' i passa a un estat de desactivació, que n'impedeix la utilització fins que l'estat actual és resolt. Quan la cerca finalitza o un es produeix un error, l'estat del botó torna a la seva normalitat.
- Quan es realitza una nova cerca, els resultats de l'anterior s'amaguen per no causar confusió.
- Els missatges d'error provinents del SDK o validacions del formulari, desapareixen quan es llança una nova operació de cerca.

11.5.5 Principal interès d'ús

Aquesta funcionalitat, com ja hem comentat, a pesar de ser de les més simples que hem implementat, pot ser la que tingui un interès més elevat, un cop s'obtingui accés a les dades de producció.

La funcionalitat va ser programada per relacionar fets històrics, com la gran recessió del vint-i-nou o la segona guerra mundial, amb les taxes de natalitat i defuncions.

Un objectiu secundari d'aquesta funcionalitat era intentar comprendre si les bases de dades de FamilySearch representaven una fotografia de la realitat, donada l'existència d'un cert nombre de registres o com de desviades es trobaven.

Per exemple, com s'ha explicat en una proposta de projecte, és conegut que durant la segona guerra mundial el nombre de matrimonis es va disparar als Estats Units d'Amèrica. Aquesta funcionalitat, permetria realitzar una primera comprovació, sobre si aquestes relacions conegeudes són també observables en les dades de l'organització.

Bibliografia

- [1] Bootstrap. *Bootstrap, HTML CSS JS framework*. [Online; accessed 25-August-2016]. 2016. URL: <https://jquery.com/>.
- [2] Population Reference Bureau. *2014 World Population Highlights*. [Online; accessed 25-August-2016]. 2014. URL: <https://www.youtube.com/watch?v=h5zp1Krc9-0>.
- [3] Population Reference Bureau. *Distilled Demographics: How Many People Have Ever Lived on Earth?* [Online; accessed 25-August-2016]. 2011. URL: <https://www.youtube.com/watch?v=7WTctr5kviA>.
- [4] Cookie Session Collective. *Cookie Session Documentation*. [Online; accessed 25-August-2016]. 2016. URL: <https://github.com/expressjs/cookie-session/>.
- [5] Chris Coyier. *Design Considerations: Text on images*. [Online; accessed 25-August-2016]. 2014. URL: <https://css-tricks.com/design-considerations-text-images/>.
- [6] Chris Coyier. *Smooth Scrolling*. [Online; accessed 25-August-2016]. 2016. URL: <https://css-tricks.com/snippets/jquery/smooth-scrolling/>.
- [7] EJS. *Embedded Javascript*. [Online; accessed 25-August-2016]. 2016. URL: <http://www.embeddedjs.com/>.
- [8] ExpressJS. *ExpressJS, Node Framework*. [Online; accessed 25-August-2016]. 2016. URL: <https://expressjs.com/>.
- [9] David Úbeda Fuster. *Com afecta la legislació actual a la feina del genealogista?* [Online; accessed 25-August-2016]. 2010. URL: <http://upcommons.upc.edu/bitstream/handle/2099.1/9460/60418.pdf>.
- [10] GitHub. *GitHub help documentation*. [Online; accessed 25-August-2016]. 2016. URL: <https://help.github.com/>.
- [11] Google. *Barchart Documentation*. [Online; accessed 25-August-2016]. 2016. URL: <https://developers.google.com/chart/interactive/docs/gallery/barchart>.
- [12] Google. *Geomap Documentation*. [Online; accessed 25-August-2016]. 2016. URL: <https://developers.google.com/chart/interactive/docs/gallery/geomap>.

- [13] Google. *Google Analytics Documentation*. [Online; accessed 25-August-2016]. 2016. URL: <https://developers.google.com/analytics/>.
- [14] Google. *Linechart Documentation*. [Online; accessed 25-August-2016]. 2016. URL: <https://developers.google.com/chart/interactive/docs/gallery/linechart>.
- [15] Heroku. *Heroku Node.js Documentation*. [Online; accessed 25-August-2016]. 2016. URL: <https://devcenter.heroku.com/categories/nodejs>.
- [16] Church of Jesus Christ of Latter-Day Saints. *The Family*. [Online; accessed 25-August-2016]. 2016. URL: <https://discover.mormon.org/en-us/topics/the-family/>.
- [17] jQuery. *jQuery, Write less do more*. [Online; accessed 25-August-2016]. 2016. URL: <https://jquery.com/>.
- [18] Jennifer Kyrnin. *The three layers of web design*. [Online; accessed 25-August-2016]. 2016. URL: <http://webdesign.about.com/od/intermediatetutorials/a/aa010707.htm>.
- [19] Ben Lin. *jQuery Preload*. [Online; accessed 25-August-2016]. 2011. URL: <http://dreamerslab.com/blog/en/preload-images-with-jquery-preload-plugin/>.
- [20] Forbes Lindesay. *Promises overview*. [Online; accessed 25-August-2016]. 2016. URL: <https://www.promisejs.org/>.
- [21] Masylum. *Include Javascript Files to Node*. [Online; accessed 25-August-2016]. 2011. URL: <http://stackoverflow.com/questions/5797852/in-node-js-how-do-i-include-functions-from-my-other-files>.
- [22] Rhonda R. McClure. *Understanding Sources*. [Online; accessed 25-August-2016]. 2002. URL: <http://www.genealogy.com/articles/twigs/rhonda011702.html>.
- [23] Mustache. *Escaping HTML*. [Online; accessed 25-August-2016]. 2015. URL: <https://github.com/janl/mustache.js/blob/master/mustache.js#L60>.
- [24] Mustache. *Mustache, Logic-less template*. [Online; accessed 25-August-2016]. 2016. URL: <https://mustache.github.io/>.
- [25] NodeJS. *NodeJS, Javascript Engine*. [Online; accessed 25-August-2016]. 2016. URL: <https://nodejs.org/en/>.
- [26] NPM. *NPM, Package Manager for Javascript*. [Online; accessed 25-August-2016]. 2016. URL: <https://www.npmjs.com/>.
- [27] Optimizilla. *Optimizilla, Web optimizer*. [Online; accessed 25-August-2016]. 2016. URL: <http://optimizilla.com/>.
- [28] FamilySearch Organization. *FamilySearch developers portal and documentation*. [Online; accessed 25-August-2016]. 2016. URL: <https://familysearch.org/developers/>.

- [29] FamilySearch Organization. *FamilySearch Javascript SDK*. [Online; accessed 25-August-2016]. 2016. URL: <http://familysearch.github.io/familysearch-javascript-sdk/2.5/>.
- [30] FamilySearch Organization. *FamilySearch webpage*. [Online; accessed 25-August-2016]. 2015. URL: <https://familysearch.org>.
- [31] FamilySearch Organization. *Helper functions Sample App Javascript SDK*. [Online; accessed 25-August-2016]. 2015. URL: <https://github.com/FamilySearch/javascript-sdk-sample-app/blob/master/assets/helpers.js#L241>.
- [32] Antonio Alfaro de Prado. *El registro civil de España (1871—¿?)* [Online; accessed 25-August-2016]. 2015. URL: <http://www.genealogiahispana.com/archivos/el-registro-civil-de-espana-1871/>.
- [33] Chris Sevilleja. *Use ExpressJS to Get URL and POST Parameters*. [Online; accessed 25-August-2016]. 2015. URL: <https://scotch.io/tutorials/use-expressjs-to-get-url-and-post-parameters>.
- [34] TWIG. *Twig, Template Engine for PHP*. [Online; accessed 25-August-2016]. 2016. URL: <http://twig.sensiolabs.org/>.
- [35] w3schools. *HTML5 Local Storage*. [Online; accessed 25-August-2016]. 2016. URL: http://www.w3schools.com/html/html5_webstorage.asp.
- [36] FamilySearch Wiki. *Genealogical Ethics (National Institute) — FamilySearch Wiki*, [Online; accessed 25-August-2016]. 2015. URL: [https://familysearch.org/wiki/en/index.php?title=Genealogical_Ethics_\(National_Institute\)&oldid=2301878](https://familysearch.org/wiki/en/index.php?title=Genealogical_Ethics_(National_Institute)&oldid=2301878).
- [37] FamilySearch Wiki. *Research Process — FamilySearch Wiki*, [Online; accessed 25-August-2016]. 2016. URL: https://familysearch.org/wiki/en/index.php?title=Research_Process&oldid=2567582.
- [38] Wikipedia. *Cascading Style Sheets — Wikipedia, The Free Encyclopedia*. [Online; accessed 25-August-2016]. 2016. URL: https://en.wikipedia.org/w/index.php?title=Cascading_Style_Sheets&oldid=735986518.
- [39] Wikipedia. *GEDCOM — Wikipedia, The Free Encyclopedia*. [Online; accessed 25-August-2016]. 2016. URL: <https://en.wikipedia.org/w/index.php?title=GEDCOM&oldid=731965739>.
- [40] Wikipedia. *Genealogy — Wikipedia, The Free Encyclopedia*. [Online; accessed 25-August-2016]. 2016. URL: <https://en.wikipedia.org/w/index.php?title=Genealogy&oldid=735149060>.
- [41] Wikipedia. *HTML5 — Wikipedia, The Free Encyclopedia*. [Online; accessed 25-August-2016]. 2016. URL: <https://en.wikipedia.org/w/index.php?title=HTML5&oldid=735803895>.
- [42] Wikipedia. *JavaScript — Wikipedia, The Free Encyclopedia*. [Online; accessed 25-August-2016]. 2016. URL: <https://en.wikipedia.org/w/index.php?title=JavaScript&oldid=735960528>.

- [43] Wikipedia. *JSON* — Wikipedia, The Free Encyclopedia. [Online; accessed 25-August-2016]. 2016. URL: <https://en.wikipedia.org/w/index.php?title=JSON&oldid=735120383>.
- [44] Wikipedia. *Model-view-controller* — Wikipedia, The Free Encyclopedia. [Online; accessed 25-August-2016]. 2016. URL: <https://en.wikipedia.org/w/index.php?title=Model%E2%80%93view%E2%80%93controller&oldid=735649380>.
- [45] Wikipedia. *Representational state transfer* — Wikipedia, The Free Encyclopedia. [Online; accessed 25-August-2016]. 2016. URL: https://en.wikipedia.org/w/index.php?title=Representational_state_transfer&oldid=735975196.
- [46] Wikipedia. *The Church of Jesus Christ of Latter-day Saints* — Wikipedia, The Free Encyclopedia. [Online; accessed 25-August-2016]. 2016. URL: https://en.wikipedia.org/w/index.php?title=The_Church_of_Jesus_Christ_of_Latter-day_Saints&oldid=735587626.
- [47] Wikipedia. *XML* — Wikipedia, The Free Encyclopedia. [Online; accessed 25-August-2016]. 2016. URL: <https://en.wikipedia.org/w/index.php?title=XML&oldid=736122302>.
- [48] Wogan. *Sorting Javascript Objects*. [Online; accessed 25-August-2016]. 2016. URL: <http://stackoverflow.com/questions/1129216/sort-array-of-objects-by-string-property-value-in-javascript>.