

# LIBRERA SEARCH SOLUTION

Enables indexing content from PDFs making it searchable  
with the possibility to classify them by the Web UI

## Contents

Requirements.....	2
Overview.....	3
Scope.....	4
RQ-01   Automated Task Service.....	5
RQ-02   Indexer.....	8
RQ-03   Model Layer.....	9
RQ-04   Web API.....	10
RQ-05   Front end App.....	12
RQ-06   Database.....	16
Deployment.....	17
Software life cycle management.....	18
Annexes.....	19

Doc. Version	Date	Author	Description
1.0	2025-12-05	Name and surname	First version of the release

# Requirements

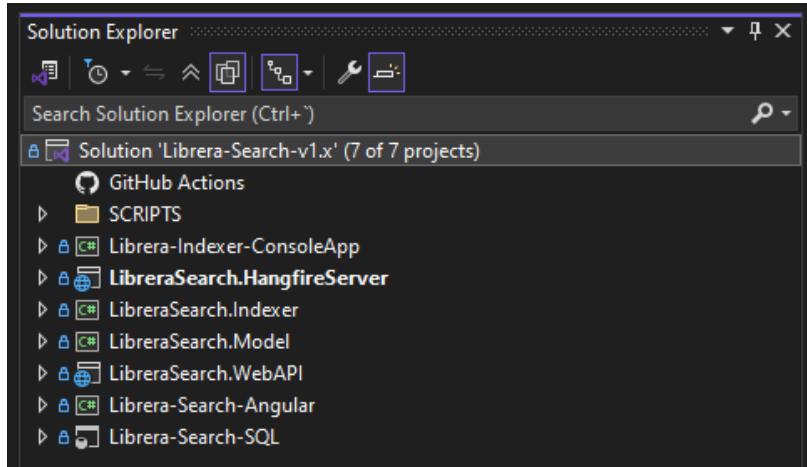
The next table shows the requirements for the solution described.

Requirement	Description
RQ-001   Automated Task Service	Hangfire offers the possibility to manage automated task services to process PDFs
RQ-002   Indexer	The indexer libraries that contains the logic to process the PDFs
RQ-003   Model Layer	The model layer with EF core to interact with database
RQ-004   Web API	The back-end to execute CRUD operations in server
RQ-005   Front end App	Angular App to offer UI to manage the records and associated records
RQ-006   Database	The data model, full text search and index services and the logic needed by Stored Procedures

# Overview

This document contains the technical details of the Librera Search v1.x solution built with Angular 16, Hangfire Server and .Net 8; enables to index content from PDFs making it searchable with the possibility to classify them by the Web UI. Implements token-based authentication for registered users and policy-based authorization in the endpoints.

Here the solution structure in Visual Studio 2022:



The application contains the next functionality:

- Each register has id, title ,authors ,series, ids, published, publisher, languages, tags, formats, path and indexed content
- We can create, retrieve, update, delete records, launch the index service and search PDF content linked to the records
- Includes two pages for finding registers by title or search indexed content
- JWT (JSON Web Token) format used for authentication
- Policy-based authorization in the endpoints

Token-based authentication is a popular approach to securing web applications, providing a stateless and scalable way to verify user identities. In this implementation, we'll explore how to integrate token-based authentication using Angular 16 as the frontend framework and .Net 8 WebAPI as the backend API.

Token-based authentication involves generating a unique token upon successful user authentication, which is then passed with each subsequent request to verify the user's identity. This approach eliminates the need for server-side session storage, making it ideal for modern web applications.

Key Components:

- .Net 8 WebAPI: Handles token generation and validation
- Angular 16: Manages user authentication and includes the token in requests

- **JWT (JSON Web Token):** The token format used for authentication

This implementation provides a secure and efficient way to authenticate users, allowing for seamless interaction between the frontend and backend.

In ASP.NET, policies are used to define requirements that must be met in order to give a user the authorization to access an endpoint.

Within Web API projects you can use policies to guard and protect endpoints from unauthorized users that don't meet the criteria. Instead of writing the authorization logic in the endpoint itself, you can refactor this into a policy that is reusable across multiple endpoints. By extracting the authorization logic from the endpoint, you can make your code more clear and keep your endpoints concise.

## Scope

This first version of the solutions grants a view of how could be easily implemented in .NET 8 a simple but effective tool to manage book records and make content searchable using open source elements like Hangfire and front-end technologies like Angular.

# RQ-01 | Automated Task Service

An easy way to perform background processing in .NET and .NET Core applications. No Windows Service or separate process required. Backed by persistent storage. Open and free for commercial use.

Here the Book Indexer Job definition in the dashboard:

The screenshot shows the Hangfire Dashboard for the 'BookIndexerJob'. The top navigation bar includes links for 'Hangfire Dashboard', 'Jobs (1)', 'Retries (0)', 'Recurring Jobs (1)', 'Servers (1)', and a 'Back to site' link. A message at the top states: 'You are using an unsupported command-line flag: --disable-web-security. Stability and security will suffer.' Below this, a sidebar lists job states: Enqueued (0/0), Scheduled (0), Processing (0), Succeeded (13), Failed (0), Deleted (2), and Awaiting (0). The main content area is titled 'BookIndexerJob' and contains a message: 'The job is finished. It will be removed automatically *in a day*'. A code snippet shows the C# code for the job:

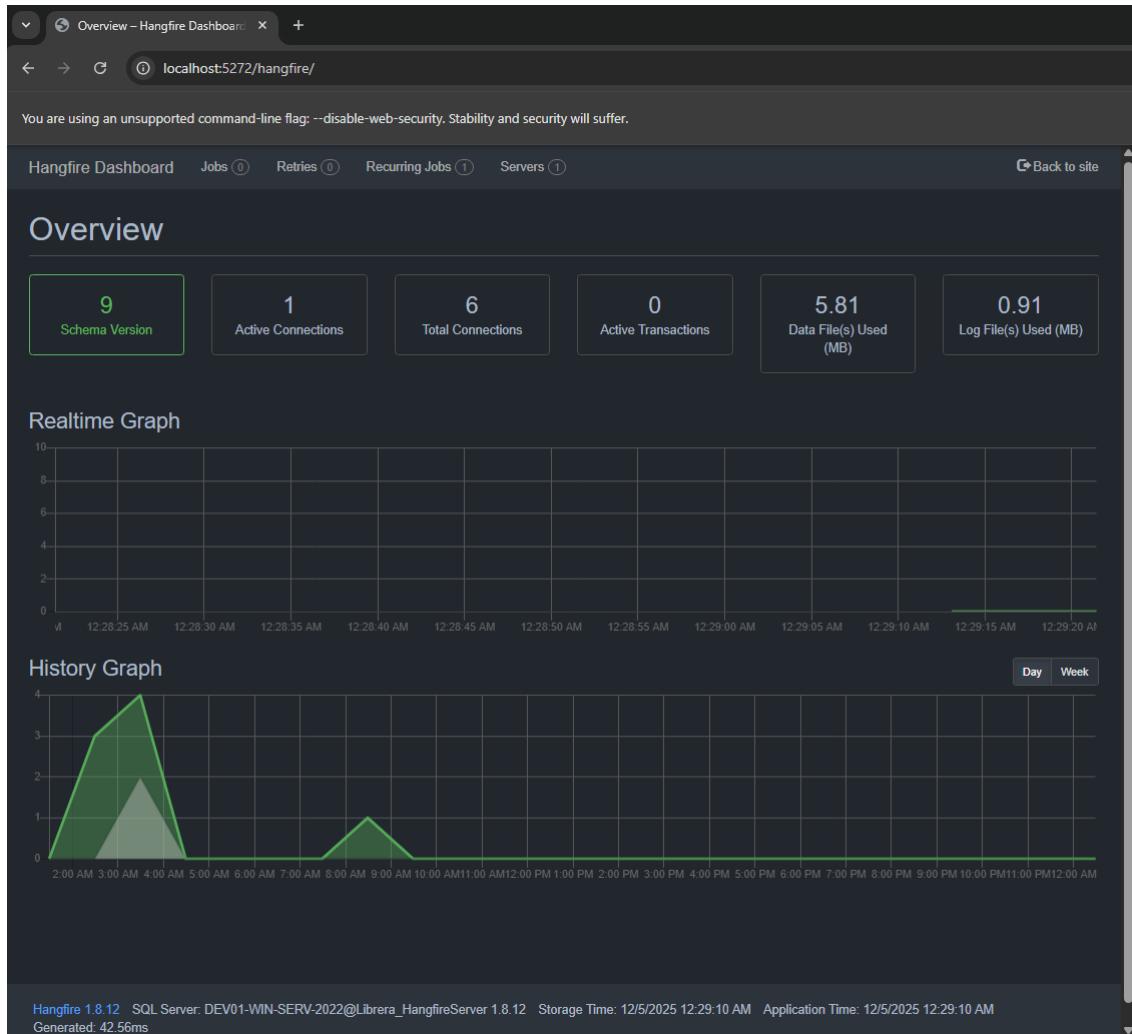
```
// Id: #15
using HangfireExample.Jobs;

var bookIndexerJob = Activate<BookIndexerJob>();
await bookIndexerJob.RunAsync();
```

The 'Parameters' section lists: CurrentCulture ("en-US"), CurrentUICulture ("en-US"), RecurringJobId ("BookIndexerJob"), and Time (1764923111). The 'State' section shows a history of events:

- Succeeded: 6 minutes ago (+39.010s)  
Latency: 9.034s  
Duration: 38.989s
- Processing: +6.970s  
Server: DEV01-WIN-SERV-10292  
Worker: 0955c009
- Enqueued: +2.053s  
Triggered by recurring job scheduler
- Created: 6 minutes ago

The next image image shows the process graph:



Here the recurring Job:

You are using an unsupported command-line flag: --disable-web-security. Stability and security will suffer.

Hangfire Dashboard    Jobs (0)    Retries (0)    Recurring Jobs (1)    Servers (0)    Back to site

## Recurring Jobs

Items per page: 10 20 50 100 500 1,000 5,000

<input type="checkbox"/>	<b>ID</b>	<b>Cron</b>	<b>Time zone</b>	<b>Job</b>	<b>Next execution</b>	<b>Last execution</b>	<b>Created</b>
<input type="checkbox"/>	BookIndexerJob	0 0 * * *	UTC	BookIndexerJob	in 15 hours	5 minutes ago	2 days ago

Total items: 1

Hangfire 1.8.12 SQL Server: DEV01-WIN-SERV-2022@Librera\_HangfireServer 1.8.12 Storage Time: 12/5/2025 12:30:13 AM Application Time: 12/5/2025 12:30:13 AM  
Generated: 203.49ms

Here the Hangfire service definition for the Job:

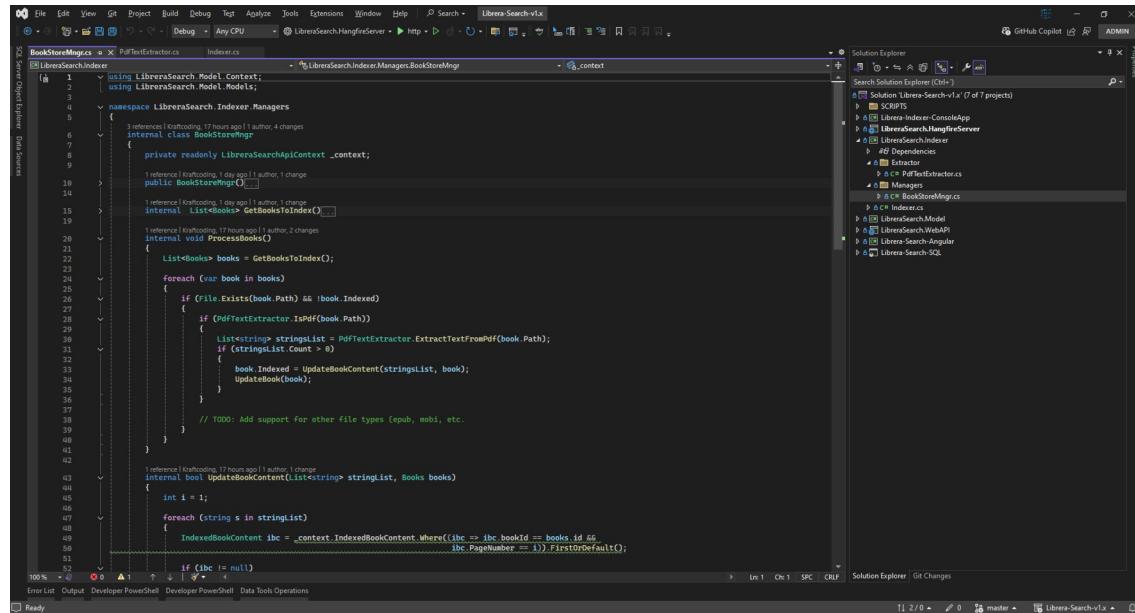
```

Program.cs
1 var hangfireConnectionString = builder.Configuration.GetConnectionString("DefaultConnection");
2
3 builder.Services.AddHangfire(configuration
4     .SetDataCompatibilityLevel(CompatibilityLevel.Version_170)
5     .UseSimpleAssemblyNameTypeSerializer()
6     .UseRecommendedSerializerSettings()
7     .UseSqlServerStorage(
8         hangfireConnectionString,
9         new SqlServerStorageOptions
10     {
11         CommandBatchMaxTimeout = TimeSpan.FromMinutes(5),
12         SlidingInvalidateByTime = TimeSpan.FromMinutes(5),
13         QueueLimit = 10000000,
14         UseRecommendedIsolationLevel = true,
15         DisableGlobalLocks = true,
16     }));
17
18 builder.Services.AddHangfireServer();
19
20 builder.Services.AddControllers();
21 builder.Services.AddEndpointsApiExplorer();
22 builder.Services.AddSwaggerGen();
23
24 //builder.Services.AddTransient<ITestRecurringJob, TestRecurringJob>();
25 //builder.Services.AddScoped<ITestJob, TestJob>();
26
27 builder.Services.AddTransient<IBookIndexerJob, BookIndexerJob>();
28
29 var app = builder.Build();
30
31 // Configures the HTTP request pipeline.
32 if (app.Environment.IsDevelopment())
33 {
34     app.UseSwagger();
35     app.UseSwaggerUI();
36 }
37
38 app.UseHttpsRedirection();
39 app.MapControllers();
40
41 // Change 'Back to site' link URL.
42 var options = new DashboardOptions { AppPath = "http://localhost:4200/" };
43 // Make 'Back to site' link working for subfolder applications
44 //var options = new DashboardOptions { AppPath = VirtualPathUtility.ToAbsolute("~/") };
45
46 app.UseHangfireDashboard("hangfire", options);
47 RecurringJob.AddOrUpdate<IBookIndexerJob>(() => app.Services.GetRequiredService<IBookIndexerJob>().RunAsync(), Cron.Daily);
48
49 amm.Run();
50
51 amm.Run();
52
53 amm.Run();
54
55 amm.Run();
56
57 amm.Run();
58
59 amm.Run();
60
61 amm.Run();
62
63 amm.Run();
64
65 amm.Run();
66
67 amm.Run();
68
69 amm.Run();
70
71 amm.Run();
72
73 amm.Run();
74
75 amm.Run();
76
77 amm.Run();
78
79 amm.Run();
80
81 amm.Run();
82
83 amm.Run();
84
85 amm.Run();
86
87 amm.Run();
88
89 amm.Run();
90
91 amm.Run();
92
93 amm.Run();
94
95 amm.Run();
96
97 amm.Run();
98
99 amm.Run();
100 amm.Run();
101
102 amm.Run();
103
104 amm.Run();
105
106 amm.Run();
107
108 amm.Run();
109
110 amm.Run();
111
112 amm.Run();
113
114 amm.Run();
115
116 amm.Run();
117
118 amm.Run();
119
120 amm.Run();
121
122 amm.Run();
123
124 amm.Run();
125
126 amm.Run();
127
128 amm.Run();
129
130 amm.Run();
131
132 amm.Run();
133
134 amm.Run();
135
136 amm.Run();
137
138 amm.Run();
139
140 amm.Run();
141
142 amm.Run();
143
144 amm.Run();
145
146 amm.Run();
147
148 amm.Run();
149
150 amm.Run();
151
152 amm.Run();
153
154 amm.Run();
155
156 amm.Run();
157
158 amm.Run();
159
160 amm.Run();
161
162 amm.Run();
163
164 amm.Run();
165
166 amm.Run();
167
168 amm.Run();
169
170 amm.Run();
171
172 amm.Run();
173
174 amm.Run();
175
176 amm.Run();
177
178 amm.Run();
179
180 amm.Run();
181
182 amm.Run();
183
184 amm.Run();
185
186 amm.Run();
187
188 amm.Run();
189
190 amm.Run();
191
192 amm.Run();
193
194 amm.Run();
195
196 amm.Run();
197
198 amm.Run();
199
200 amm.Run();
201
202 amm.Run();
203
204 amm.Run();
205
206 amm.Run();
207
208 amm.Run();
209
210 amm.Run();
211
212 amm.Run();
213
214 amm.Run();
215
216 amm.Run();
217
218 amm.Run();
219
220 amm.Run();
221
222 amm.Run();
223
224 amm.Run();
225
226 amm.Run();
227
228 amm.Run();
229
230 amm.Run();
231
232 amm.Run();
233
234 amm.Run();
235
236 amm.Run();
237
238 amm.Run();
239
240 amm.Run();
241
242 amm.Run();
243
244 amm.Run();
245
246 amm.Run();
247
248 amm.Run();
249
250 amm.Run();
251
252 amm.Run();
253
254 amm.Run();
255
256 amm.Run();
257
258 amm.Run();
259
260 amm.Run();
261
262 amm.Run();
263
264 amm.Run();
265
266 amm.Run();
267
268 amm.Run();
269
270 amm.Run();
271
272 amm.Run();
273
274 amm.Run();
275
276 amm.Run();
277
278 amm.Run();
279
280 amm.Run();
281
282 amm.Run();
283
284 amm.Run();
285
286 amm.Run();
287
288 amm.Run();
289
290 amm.Run();
291
292 amm.Run();
293
294 amm.Run();
295
296 amm.Run();
297
298 amm.Run();
299
300 amm.Run();
301
302 amm.Run();
303
304 amm.Run();
305
306 amm.Run();
307
308 amm.Run();
309
310 amm.Run();
311
312 amm.Run();
313
314 amm.Run();
315
316 amm.Run();
317
318 amm.Run();
319
320 amm.Run();
321
322 amm.Run();
323
324 amm.Run();
325
326 amm.Run();
327
328 amm.Run();
329
330 amm.Run();
331
332 amm.Run();
333
334 amm.Run();
335
336 amm.Run();
337
338 amm.Run();
339
340 amm.Run();
341
342 amm.Run();
343
344 amm.Run();
345
346 amm.Run();
347
348 amm.Run();
349
350 amm.Run();
351
352 amm.Run();
353
354 amm.Run();
355
356 amm.Run();
357
358 amm.Run();
359
360 amm.Run();
361
362 amm.Run();
363
364 amm.Run();
365
366 amm.Run();
367
368 amm.Run();
369
370 amm.Run();
371
372 amm.Run();
373
374 amm.Run();
375
376 amm.Run();
377
378 amm.Run();
379
380 amm.Run();
381
382 amm.Run();
383
384 amm.Run();
385
386 amm.Run();
387
388 amm.Run();
389
390 amm.Run();
391
392 amm.Run();
393
394 amm.Run();
395
396 amm.Run();
397
398 amm.Run();
399
400 amm.Run();
401
402 amm.Run();
403
404 amm.Run();
405
406 amm.Run();
407
408 amm.Run();
409
410 amm.Run();
411
412 amm.Run();
413
414 amm.Run();
415
416 amm.Run();
417
418 amm.Run();
419
420 amm.Run();
421
422 amm.Run();
423
424 amm.Run();
425
426 amm.Run();
427
428 amm.Run();
429
430 amm.Run();
431
432 amm.Run();
433
434 amm.Run();
435
436 amm.Run();
437
438 amm.Run();
439
440 amm.Run();
441
442 amm.Run();
443
444 amm.Run();
445
446 amm.Run();
447
448 amm.Run();
449
450 amm.Run();
451
452 amm.Run();
453
454 amm.Run();
455
456 amm.Run();
457
458 amm.Run();
459
460 amm.Run();
461
462 amm.Run();
463
464 amm.Run();
465
466 amm.Run();
467
468 amm.Run();
469
470 amm.Run();
471
472 amm.Run();
473
474 amm.Run();
475
476 amm.Run();
477
478 amm.Run();
479
480 amm.Run();
481
482 amm.Run();
483
484 amm.Run();
485
486 amm.Run();
487
488 amm.Run();
489
490 amm.Run();
491
492 amm.Run();
493
494 amm.Run();
495
496 amm.Run();
497
498 amm.Run();
499
500 amm.Run();
501
502 amm.Run();
503
504 amm.Run();
505
506 amm.Run();
507
508 amm.Run();
509
510 amm.Run();
511
512 amm.Run();
513
514 amm.Run();
515
516 amm.Run();
517
518 amm.Run();
519
520 amm.Run();
521
522 amm.Run();
523
524 amm.Run();
525
526 amm.Run();
527
528 amm.Run();
529
530 amm.Run();
531
532 amm.Run();
533
534 amm.Run();
535
536 amm.Run();
537
538 amm.Run();
539
540 amm.Run();
541
542 amm.Run();
543
544 amm.Run();
545
546 amm.Run();
547
548 amm.Run();
549
550 amm.Run();
551
552 amm.Run();
553
554 amm.Run();
555
556 amm.Run();
557
558 amm.Run();
559
560 amm.Run();
561
562 amm.Run();
563
564 amm.Run();
565
566 amm.Run();
567
568 amm.Run();
569
570 amm.Run();
571
572 amm.Run();
573
574 amm.Run();
575
576 amm.Run();
577
578 amm.Run();
579
580 amm.Run();
581
582 amm.Run();
583
584 amm.Run();
585
586 amm.Run();
587
588 amm.Run();
589
590 amm.Run();
591
592 amm.Run();
593
594 amm.Run();
595
596 amm.Run();
597
598 amm.Run();
599
600 amm.Run();
601
602 amm.Run();
603
604 amm.Run();
605
606 amm.Run();
607
608 amm.Run();
609
610 amm.Run();
611
612 amm.Run();
613
614 amm.Run();
615
616 amm.Run();
617
618 amm.Run();
619
620 amm.Run();
621
622 amm.Run();
623
624 amm.Run();
625
626 amm.Run();
627
628 amm.Run();
629
630 amm.Run();
631
632 amm.Run();
633
634 amm.Run();
635
636 amm.Run();
637
638 amm.Run();
639
640 amm.Run();
641
642 amm.Run();
643
644 amm.Run();
645
646 amm.Run();
647
648 amm.Run();
649
650 amm.Run();
651
652 amm.Run();
653
654 amm.Run();
655
656 amm.Run();
657
658 amm.Run();
659
660 amm.Run();
661
662 amm.Run();
663
664 amm.Run();
665
666 amm.Run();
667
668 amm.Run();
669
670 amm.Run();
671
672 amm.Run();
673
674 amm.Run();
675
676 amm.Run();
677
678 amm.Run();
679
680 amm.Run();
681
682 amm.Run();
683
684 amm.Run();
685
686 amm.Run();
687
688 amm.Run();
689
690 amm.Run();
691
692 amm.Run();
693
694 amm.Run();
695
696 amm.Run();
697
698 amm.Run();
699
700 amm.Run();
701
702 amm.Run();
703
704 amm.Run();
705
706 amm.Run();
707
708 amm.Run();
709
710 amm.Run();
711
712 amm.Run();
713
714 amm.Run();
715
716 amm.Run();
717
718 amm.Run();
719
720 amm.Run();
721
722 amm.Run();
723
724 amm.Run();
725
726 amm.Run();
727
728 amm.Run();
729
730 amm.Run();
731
732 amm.Run();
733
734 amm.Run();
735
736 amm.Run();
737
738 amm.Run();
739
740 amm.Run();
741
742 amm.Run();
743
744 amm.Run();
745
746 amm.Run();
747
748 amm.Run();
749
750 amm.Run();
751
752 amm.Run();
753
754 amm.Run();
755
756 amm.Run();
757
758 amm.Run();
759
760 amm.Run();
761
762 amm.Run();
763
764 amm.Run();
765
766 amm.Run();
767
768 amm.Run();
769
770 amm.Run();
771
772 amm.Run();
773
774 amm.Run();
775
776 amm.Run();
777
778 amm.Run();
779
780 amm.Run();
781
782 amm.Run();
783
784 amm.Run();
785
786 amm.Run();
787
788 amm.Run();
789
790 amm.Run();
791
792 amm.Run();
793
794 amm.Run();
795
796 amm.Run();
797
798 amm.Run();
799
800 amm.Run();
801
802 amm.Run();
803
804 amm.Run();
805
806 amm.Run();
807
808 amm.Run();
809
810 amm.Run();
811
812 amm.Run();
813
814 amm.Run();
815
816 amm.Run();
817
818 amm.Run();
819
820 amm.Run();
821
822 amm.Run();
823
824 amm.Run();
825
826 amm.Run();
827
828 amm.Run();
829
830 amm.Run();
831
832 amm.Run();
833
834 amm.Run();
835
836 amm.Run();
837
838 amm.Run();
839
840 amm.Run();
841
842 amm.Run();
843
844 amm.Run();
845
846 amm.Run();
847
848 amm.Run();
849
850 amm.Run();
851
852 amm.Run();
853
854 amm.Run();
855
856 amm.Run();
857
858 amm.Run();
859
860 amm.Run();
861
862 amm.Run();
863
864 amm.Run();
865
866 amm.Run();
867
868 amm.Run();
869
870 amm.Run();
871
872 amm.Run();
873
874 amm.Run();
875
876 amm.Run();
877
878 amm.Run();
879
880 amm.Run();
881
882 amm.Run();
883
884 amm.Run();
885
886 amm.Run();
887
888 amm.Run();
889
890 amm.Run();
891
892 amm.Run();
893
894 amm.Run();
895
896 amm.Run();
897
898 amm.Run();
899
900 amm.Run();
901
902 amm.Run();
903
904 amm.Run();
905
906 amm.Run();
907
908 amm.Run();
909
910 amm.Run();
911
912 amm.Run();
913
914 amm.Run();
915
916 amm.Run();
917
918 amm.Run();
919
920 amm.Run();
921
922 amm.Run();
923
924 amm.Run();
925
926 amm.Run();
927
928 amm.Run();
929
930 amm.Run();
931
932 amm.Run();
933
934 amm.Run();
935
936 amm.Run();
937
938 amm.Run();
939
940 amm.Run();
941
942 amm.Run();
943
944 amm.Run();
945
946 amm.Run();
947
948 amm.Run();
949
950 amm.Run();
951
952 amm.Run();
953
954 amm.Run();
955
956 amm.Run();
957
958 amm.Run();
959
960 amm.Run();
961
962 amm.Run();
963
964 amm.Run();
965
966 amm.Run();
967
968 amm.Run();
969
970 amm.Run();
971
972 amm.Run();
973
974 amm.Run();
975
976 amm.Run();
977
978 amm.Run();
979
980 amm.Run();
981
982 amm.Run();
983
984 amm.Run();
985
986 amm.Run();
987
988 amm.Run();
989
990 amm.Run();
991
992 amm.Run();
993
994 amm.Run();
995
996 amm.Run();
997
998 amm.Run();
999
1000 amm.Run();
1001
1002 amm.Run();
1003
1004 amm.Run();
1005
1006 amm.Run();
1007
1008 amm.Run();
1009
1010 amm.Run();
1011
1012 amm.Run();
1013
1014 amm.Run();
1015
1016 amm.Run();
1017
1018 amm.Run();
1019
1020 amm.Run();
1021
1022 amm.Run();
1023
1024 amm.Run();
1025
1026 amm.Run();
1027
1028 amm.Run();
1029
1030 amm.Run();
1031
1032 amm.Run();
1033
1034 amm.Run();
1035
1036 amm.Run();
1037
1038 amm.Run();
1039
1040 amm.Run();
1041
1042 amm.Run();
1043
1044 amm.Run();
1045
1046 amm.Run();
1047
1048 amm.Run();
1049
1050 amm.Run();
1051
1052 amm.Run();
1053
1054 amm.Run();
1055
1056 amm.Run();
1057
1058 amm.Run();
1059
1060 amm.Run();
1061
1062 amm.Run();
1063
1064 amm.Run();
1065
1066 amm.Run();
1067
1068 amm.Run();
1069
1070 amm.Run();
1071
1072 amm.Run();
1073
1074 amm.Run();
1075
1076 amm.Run();
1077
1078 amm.Run();
1079
1080 amm.Run();
1081
1082 amm.Run();
1083
1084 amm.Run();
1085
1086 amm.Run();
1087
1088 amm.Run();
1089
1090 amm.Run();
1091
1092 amm.Run();
1093
1094 amm.Run();
1095
1096 amm.Run();
1097
1098 amm.Run();
1099
1100 amm.Run();
1101
1102 amm.Run();
1103
1104 amm.Run();
1105
1106 amm.Run();
1107
1108 amm.Run();
1109
1110 amm.Run();
1111
1112 amm.Run();
1113
1114 amm.Run();
1115
1116 amm.Run();
1117
1118 amm.Run();
1119
1120 amm.Run();
1121
1122 amm.Run();
1123
1124 amm.Run();
1125
1126 amm.Run();
1127
1128 amm.Run();
1129
1130 amm.Run();
1131
1132 amm.Run();
1133
1134 amm.Run();
1135
1136 amm.Run();
1137
1138 amm.Run();
1139
1140 amm.Run();
1141
1142 amm.Run();
1143
1144 amm.Run();
1145
1146 amm.Run();
1147
1148 amm.Run();
1149
1150 amm.Run();
1151
1152 amm.Run();
1153
1154 amm.Run();
1155
1156 amm.Run();
1157
1158 amm.Run();
1159
1160 amm.Run();
1161
1162 amm.Run();
1163
1164 amm.Run();
1165
1166 amm.Run();
1167
1168 amm.Run();
1169
1170 amm.Run();
1171
1172 amm.Run();
1173
1174 amm.Run();
1175
1176 amm.Run();
1177
1178 amm.Run();
1179
1180 amm.Run();
1181
1182 amm.Run();
1183
1184 amm.Run();
1185
1186 amm.Run();
1187
1188 amm.Run();
1189
1190 amm.Run();
1191
1192 amm.Run();
1193
1194 amm.Run();
1195
1196 amm.Run();
1197
1198 amm.Run();
1199
1200 amm.Run();
1201
1202 amm.Run();
1203
1204 amm.Run();
1205
1206 amm.Run();
1207
1208 amm.Run();
1209
1210 amm.Run();
1211
1212 amm.Run();
1213
1214 amm.Run();
1215
1216 amm.Run();
1217
1218 amm.Run();
1219
1220 amm.Run();
1221
1222 amm.Run();
1223
1224 amm.Run();
1225
1226 amm.Run();
1227
1228 amm.Run();
1229
1230 amm.Run();
1231
1232 amm.Run();
1233
1234 amm.Run();
1235
1236 amm.Run();
1237
1238 amm.Run();
1239
1240 amm.Run();
1241
1242 amm.Run();
1243
1244 amm.Run();
1245
1246 amm.Run();
1247
1248 amm.Run();
1249
1250 amm.Run();
1251
1252 amm.Run();
1253
1254 amm.Run();
1255
1256 amm.Run();
1257
1258 amm.Run();
1259
1260 amm.Run();
1261
1262 amm.Run();
1263
1264 amm.Run();
1265
1266 amm.Run();
1267
1268 amm.Run();
1269
1270 amm.Run();
1271
1272 amm.Run();
1273
1274 amm.Run();
1275
1276 amm.Run();
1277
1278 amm.Run();
1279
1280 amm.Run();
1281
1282 amm.Run();
1283
1284 amm.Run();
1285
1286 amm.Run();
1287
1288 amm.Run();
1289
1290 amm.Run();
1291
1292 amm.Run();
1293
1294 amm.Run();
1295
1296 amm.Run();
1297
1298 amm.Run();
1299
1300 amm.Run();
1301
1302 amm.Run();
1303
1304 amm.Run();
1305
1306 amm.Run();
1307
1308 amm.Run();
1309
1310 amm.Run();
1311
1312 amm.Run();
1313
1314 amm.Run();
1315
1316 amm.Run();
1317
1318 amm.Run();
1319
1320 amm.Run();
1321
1322 amm.Run();
1323
1324 amm.Run();
1325
1326 amm.Run();
1327
1328 amm.Run();
1329
1330 amm.Run();
1331
1332 amm.Run();
1333
1334 amm.Run();
1335
1336 amm.Run();
1337
1338 amm.Run();
1339
1340 amm.Run();
1341
1342 amm.Run();
1343
1344 amm.Run();
1345
1346 amm.Run();
1347
1348 amm.Run();
1349
1350 amm.Run();
1351
1352 amm.Run();
1353
1354 amm.Run();
1355
1356 amm.Run();
1357
1358 amm.Run();
1359
1360 amm.Run();
1361
1362 amm.Run();
1363
1364 amm.Run();
1365
1366 amm.Run();
1367
1368 amm.Run();
1369
1370 amm.Run();
1371
1372 amm.Run();
1373
1374 amm.Run();
1375
1376 amm.Run();
1377
1378 amm.Run();
1379
1380 amm.Run();
1381
1382 amm.Run();
1383
1384 amm.Run();
1385
1386 amm.Run();
1387
1388 amm.Run();
1389
1390 amm.Run();
1391
1392 amm.Run();
1393
1394 amm.Run();
1395
1396 amm.Run();
1397
1398 amm.Run();
1399
1400 amm.Run();
1401
1402 amm.Run();
1403
1404 amm.Run();
1405
1406 amm.Run();
1407
1408 amm.Run();
1409
1410 amm.Run();
1411
1412 amm.Run();
1413
1414 amm.Run();
1415
1416 amm.Run();
1417
1418 amm.Run();
1419
1420 amm.Run();
1421
1422 amm.Run();
1423
1424 amm.Run();
1425
1426 amm.Run();
1427
1428 amm.Run();
1429
1430 amm.Run();
1431
1432 amm.Run();
1433
1434 amm.Run();
1435
1436 amm.Run();
1437
1438 amm.Run();
1439
1440 amm.Run();
1441
1442 amm.Run();
1443
1444 amm.Run();
1445
1446 amm.Run();
1447
1448 amm.Run();
1449
1450 amm.Run();
1451
1452 amm.Run();
1453
1454 amm.Run();
1455
1456 amm.Run();
1457
1458 amm.Run();
1459
1460 amm.Run();
1461
1462 amm.Run();
1463
1464 amm.Run();
1465
1466 amm.Run();
1467
1468 amm.Run();
1469
1470 amm.Run();
1471
1472 amm.Run();
1473
1474 amm.Run();
1475
1476 amm.Run();
1477
1478 amm.Run();
1479
1480 amm.Run();
1481
1482 amm.Run();
1483
1484 amm.Run();
1485
1486 amm.Run();
1487
1488 amm.Run();
1489
1490 amm.Run();
1491
1492 amm.Run();
1493
1494 amm.Run();
1495
1496 amm.Run();
1497
1498 amm.Run();
1499
1500 amm.Run();
1501
1502 amm.Run();
1503
1504 amm.Run();
1505
1506 amm.Run();
1507
1508 amm.Run();
1509
1510 amm.Run();
1511
1512 amm.Run();
1513
1514 amm.Run();
1515
1516 amm.Run();
1517
1518 amm.Run();
1519
1520 amm.Run();
1521
1522 amm.Run();
1523
1524 amm.Run();
1525
1526 amm.Run();
1527
1528 amm.Run();
1529
1530 amm.Run();
1531
1532 amm.Run();
1533
1534 amm.Run();
1535
1536 amm.Run();
1537
1538 amm.Run();
1539
1540 amm.Run();
1541
1542 amm.Run();
1543
1544 amm.Run();
1545
1546 amm.Run();
1547
1548 amm.Run();
1549
1550 amm.Run();
1551
1552 amm.Run();
1553
1554 amm.Run();
1555
1556 amm.Run();
1557
1558 amm.Run();
1559
1560 amm.Run();
1561
1562 amm.Run();
1563
1564 amm.Run();
1565
1566 amm.Run();
1567
1568 amm.Run();
1569
1570 amm.Run();
1571
1572 amm.Run();
1573
1574 amm.Run();
1575
1576 amm.Run();
1577
1578 amm.Run();
1579
1580 amm.Run();
1581
1582 amm.Run();
1583
1584 amm.Run();
1585
1586 amm.Run();
1587
1588 amm.Run();
1589
1590 amm.Run();
1591
1592 amm.Run();
1593
1594 amm.Run();
1595
1596 amm.Run();
1597
1598 amm.Run();
1599
1600 amm.Run();
1601
1602 amm.Run();
1603
1604 amm.Run();
1605
1606 amm.Run();
1607
1608 amm.Run();
1609
1610 amm.Run();
1611
1612 amm.Run();
1613
1614 amm.Run();
1615
1616 amm.Run();
1617
1618 amm.Run();
1619
1620 amm.Run();
1621
1622 amm.Run();
1623
1624 amm.Run();
1625
1626 amm.Run();
1627
1628 amm.Run();
1629
1630 amm.Run();
1631
1632 amm.Run();
1633
1634 amm.Run();
1635
1636 amm.Run();
1637
1638 amm.Run();
1639
1640 amm.Run();
1641
1642 amm.Run();
1643
1644 amm.Run();
1645
1646 amm.Run();
1647
1648 amm.Run();
1649
1650 amm.Run();
1651
1652 amm.Run();
1653
1654 amm.Run();
1655
1656 amm.Run();
1657
1658 amm.Run();
1659
1660 amm.Run();
1661
1662 amm.Run();
1663
1664 amm.Run();
1665
1666 amm.Run();
1667
1668 amm.Run();
1669
1670 amm.Run();
1671
1672 amm.Run();
1673
1674 amm.Run();
1675
1676 amm.Run();
1677
1678 amm.Run();
1679
1680 amm.Run();
1681
1682 amm.Run();
1683
1684 amm.Run();
1685
1686 amm.Run();
1687
1688 amm.Run();
1689
1690 amm.Run();
1691
1692 amm.Run();
1693
1694 amm.Run();
1695
1696 amm.Run();
1697
1698 amm.Run();
1699
1700 amm.Run();
1701
1702 amm.Run();
1703
1704 amm.Run();
1705
1706 amm.Run();
1707
1708 amm.Run();
1709
1710 amm.Run();
1711
1712 amm.Run();
1713
1714 amm.Run();
1715
1716 amm.Run();
1717
1718 amm.Run();
1719
1720 amm.Run();
1721
1722 amm.Run();
1723
1724 amm.Run();
1725
1726 amm.Run();
1727
1728 amm.Run();
1729
1730 amm.Run();
1731
1732 amm.Run();
1733
1734 amm.Run();
1735
1736 amm.Run();
1737
1738 amm.Run();
1739
1740 amm.Run();
1741
1742 amm.Run();
1743
1744 amm.Run();
1745
1746 amm.Run();
1747
1748 amm.Run();
1749

```

## RQ-02 | Indexer

The indexer is implemented by a library type project on which we have the PDF extractor and the c# method that the Hangfire service will call by a recurring job definition.



```
1  using LibreraSearch.Model.Context;
2  using LibreraSearch.Model.Models;
3
4  namespace LibreraSearch.Indexer.Managers
5  {
6      internal class BookStoreMngr
7      {
8          private readonly LibreraSearchDbContext _context;
9
10         public BookStoreMngr()
11         {
12             _context = new LibreraSearchDbContext();
13         }
14
15         internal void GetBooksToIndex()
16         {
17             foreach (var book in books)
18             {
19                 if (File.Exists(book.Path) && !book.Indexed)
20                 {
21                     if (PdfTextExtractor.IsPdf(book.Path))
22                     {
23                         List<string> stringList = PdfTextExtractor.ExtractTextFromPdf(book.Path);
24                         if (stringList.Count > 0)
25                         {
26                             book.Indexed = UpdateBookContent(stringList, book);
27                         }
28                     }
29                 }
30             }
31         }
32
33         internal void ProcessBooks()
34         {
35             List<Book> books = GetBooksToIndex();
36
37             foreach (var book in books)
38             {
39                 if (File.Exists(book.Path) && !book.Indexed)
40                 {
41                     if (PdfTextExtractor.IsPdf(book.Path))
42                     {
43                         List<string> stringList = PdfTextExtractor.ExtractTextFromPdf(book.Path);
44                         if (stringList.Count > 0)
45                         {
46                             book.Indexed = UpdateBookContent(stringList, book);
47                         }
48                     }
49                 }
50             }
51         }
52
53         internal void UpdateBookContent(List<string> stringList, Books books)
54         {
55             int i = 1;
56
57             foreach (string s in stringList)
58             {
59                 IndexedBookContent ibc = _context.IndexedBookContent.Where(ibc => ibc.bookId == books.id &&
60                                         ibc.PageNumber == i).FirstOrDefault();
61
62                 if (ibc != null)
63                 {
64                     ibc.Content += s;
65                     ibc.PageNumber++;
66                 }
67             }
68         }
69     }
70 }
```

The indexer extraccts content form PDFs and stores it in DB.

## **RQ-03** | Model Layer

The model layer is managed by EF Core and LinQ is used to launch the queries.

# RQ-04 | Web API

The Web API permits to the front-end application to execute CRUD operations my RESTful end-points. Those end-points are securitized by policy-based authorization.

Here some images.

Web API Auth. service setup:

```
1  using LibreraSearch.Models.Context;
2  using Microsoft.AspNetCore.Authentication.JwtBearer;
3  using Microsoft.EntityFrameworkCore;
4  using Microsoft.IdentityModel.Tokens;
5  using System.Text;
6
7  var builder = WebApplication.CreateBuilder(args);
8
9  // Add services to the container.
10
11 builder.Services.AddControllers();
12 // Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle
13 builder.Services.AddEndpointsApiExplorer();
14 builder.Services.AddSwaggerGen();
15
16 // DB Context
17 builder.Services.AddDbContext<LibreraSearchDbContext>(o => o.UseSqlServer(builder.Configuration.GetConnectionString("LibreraSearchDBConnString")));
18
19 // This code ensures that only users who possess valid JWT tokens issued by "YourIssuerName" & containing the "UserId" claim can access restricted s
20
21 builder.Services.AddAuthentication(JwtBearerDefaults.AuthenticationScheme)
22     .AddJwtBearer(options =>
23         {
24             options.TokenValidationParameters = new TokenValidationParameters
25             {
26                 ValidateIssuer = true,
27                 ValidateIssuerName = "YourIssuerName",
28                 ValidateAudience = true,
29                 ValidateLifetime = true,
30                 IssuerSigningKey = SymmetricSecurityKey(Encoding.UTF8.GetBytes(builder.Configuration["Tokens:Key"]))
31             };
32         });
33
34
35         builder.Services.AddAuthorization(
36             options => options.AddPolicy("Authenticated",
37                 policy => policy.RequireClaim("Email")));
38
39
40         var app = builder.Build();
41
42         // Configure the HTTP request pipeline.
43         if (app.Environment.IsDevelopment())
44         {
45             app.UseSwagger();
46             app.UseSwaggerUI();
47         }
48
49         app.UseHttpsRedirection();
50
51         app.UseAuthorization();
52
53     }]
```

Web API Tokens configuration:

```
Schema: https://www.schemastore.org/appsettings.json
1  {
2      "AllowedHosts": "*",
3      "ConnectionStrings": {
4          "LibreraSearchDBConnString": "Data Source=DEV01-WIN-SERV-2022"
5      },
6      "Logging": {
7          "LogLevel": {
8              "Default": "Information",
9              "Microsoft.AspNetCore": "Warning"
10         }
11     },
12     "Tokens": {
13         "Key": "ThisIsTheMostSecretStringEver123456789%",
14         "Issuer": "localhost",
15         "Audience": "http://localhost:4200/"
16     }
17 }
```

Web API securitized end-points:

The screenshot shows a Visual Studio interface with the following details:

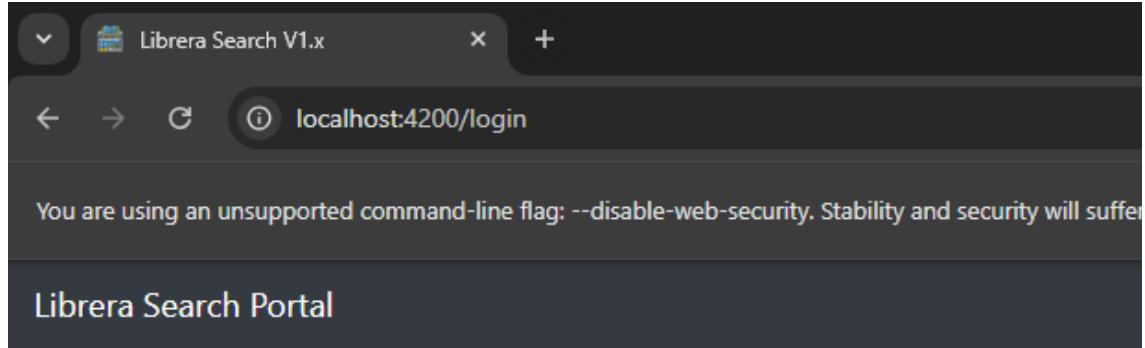
- Code Editor:** Displays the `LibraSearchController.cs` file under the `LibraSearch.WebAPI` project. The code is a .NET Core API controller for managing books.
- Solution Explorer:** Shows the solution structure for "Libra-Search-v1.x" containing seven projects:
  - ASP.NET Core Library (Web API)
  - GitHub Actions
  - Libra-Indexer-ConsoleApp
  - Libra-Search-HanlderServer
  - Libra-Search-Model
  - Libra-Search-WEB
  - LibraSearch.WebAPI
- Status Bar:** Shows "Ready" status, file paths like "Libra-Search-v1.x", and other system information.

## RQ-05 | Front end App

The front end app is built in Angular 16.

Here some images.

Login form:



Login:

email	password	Submit
-------	----------	--------

Search by title list and details:

The screenshot shows a web-based application titled "Librera Search V1.x" running on "localhost:4200/LibreraSearch". The interface includes a navigation bar with links for "Librera Search Portal", "Librera Search", "Add", "Text Content Search", "Hangfire Dashboard", and "Log out". Below the navigation is a search bar with a placeholder "Search by title" and a "Search" button. The main content area is divided into two sections: "Librera Search List" and "Librera Search". The "Librera Search List" section displays a single item: "Kali inside Proxmox (Guest VM)" with the subtitle "Linux Encrypted Filesystem with dm-crypt". A red "Remove All" button is located below this item. The "Librera Search" section provides detailed information about the item, including its modified date (2025-12-05T00:00:00), title (Kali inside Proxmox (Guest VM)), author (Kali), series (1), ID (isbn:0001), publication date (Nov 10, 2025), publisher (Linux), language (Eng), tags (cybersecurity), formats (pdf), path (E:\TEST\PDF\Kali inside Proxmox (Guest VM).pdf), and indexed status (false). An "Edit" button is located at the bottom right of the detailed view.

Librera Search Portal | Librera Search | Add | Text Content Search | Hangfire Dashboard | Log out

Search by title

**Librera Search List**

Kali inside Proxmox (Guest VM)

Linux Encrypted Filesystem with dm-crypt

**Librera Search**

**Modified:** 2025-12-05T00:00:00

**Title:** Kali inside Proxmox (Guest VM)

**Authors:** Kali

**Series:** 1

**Ids:** isbn:0001

**Published:** Nov 10, 2025

**Publisher:** Linux

**Languages:** Eng

**Tags:** cybersecurity

**Formats:** pdf

**Path:** E:\TEST\PDF\Kali inside Proxmox (Guest VM).pdf

**Indexed:** false

Add form:

Librera Search V1.x

localhost:4200/add

Librera Search Portal Librera Search Add Text Content Search Hangfire Dashboard Log out

Modified

Title

Authors

Series  0

Ids

Published

Publisher

Languages

Tags

Formats

Path

Indexed

**Submit**

Search by content page details:

The screenshot shows a web browser window titled "Librera Search V1.x" with the URL "localhost:4200/LibreraTextSearch". The page header includes "Librera Search Portal", "Librera Search", "Add", "Text Content Search", "Hangfire Dashboard", and a "Log out" button. Below the header is a search bar with the word "store" and a "Search by Text Content" button. The main content area is titled "Librera Text Search List" and displays a single search result. The result includes the following fields and their values:

- Title:** Linux Encrypted Filesystem with dm-crypt
- Authors:** Joe Doe
- PageNumber:** 1

The detailed description of the result is as follows:

Linux Encrypted Filesystem with dm-crypt An encrypted filesystem will protect against bare-metal attacks against a hard drive. Anyone getting their hands on the drive would have to use brute force to guess the encryption key, a substantial hindrance to getting at your data. Windows and Mac OS X each provide its own standard cryptofs tools while Linux, of course, provides many tools to accomplish the task. The tool of choice these days, it seems, is dm-crypt . Invoked with the userspace cryptsetup utility, dm-crypt provides a fairly clean and easy-to-use cryptofs tool for Linux. Additionally, CentOS 5 includes an improved version of dm-crypt that supports LUKS . LUKS is an upcoming standard for an on-disk representation of information about encrypted volumes. Meta- data about encrypted data is stored in the partition header, and allows for compatibility between different systems and support for multiple user passwords. Besides that, GNOME and HAL have support for handling LUKS volumes, and can automatically prompt for a password if a removable medium with a LUKS volume is attached. If you do not require compatibility with older CentOS versions or systems that do not support LUKS, it is advised to use the LUKS scheme. The commands for setting up encrypted LUKS volumes are also described in the examples in this article. Here are Scripts to automate creation, un-mounting, and remounting of LUKS encrypted filesystems following the method described below.

1. Required Packages Before getting started, make sure all the requisite packages are installed:

- cryptsetup (cryptsetup-luks for CentOS-5)
- device-mapper
- util-linux

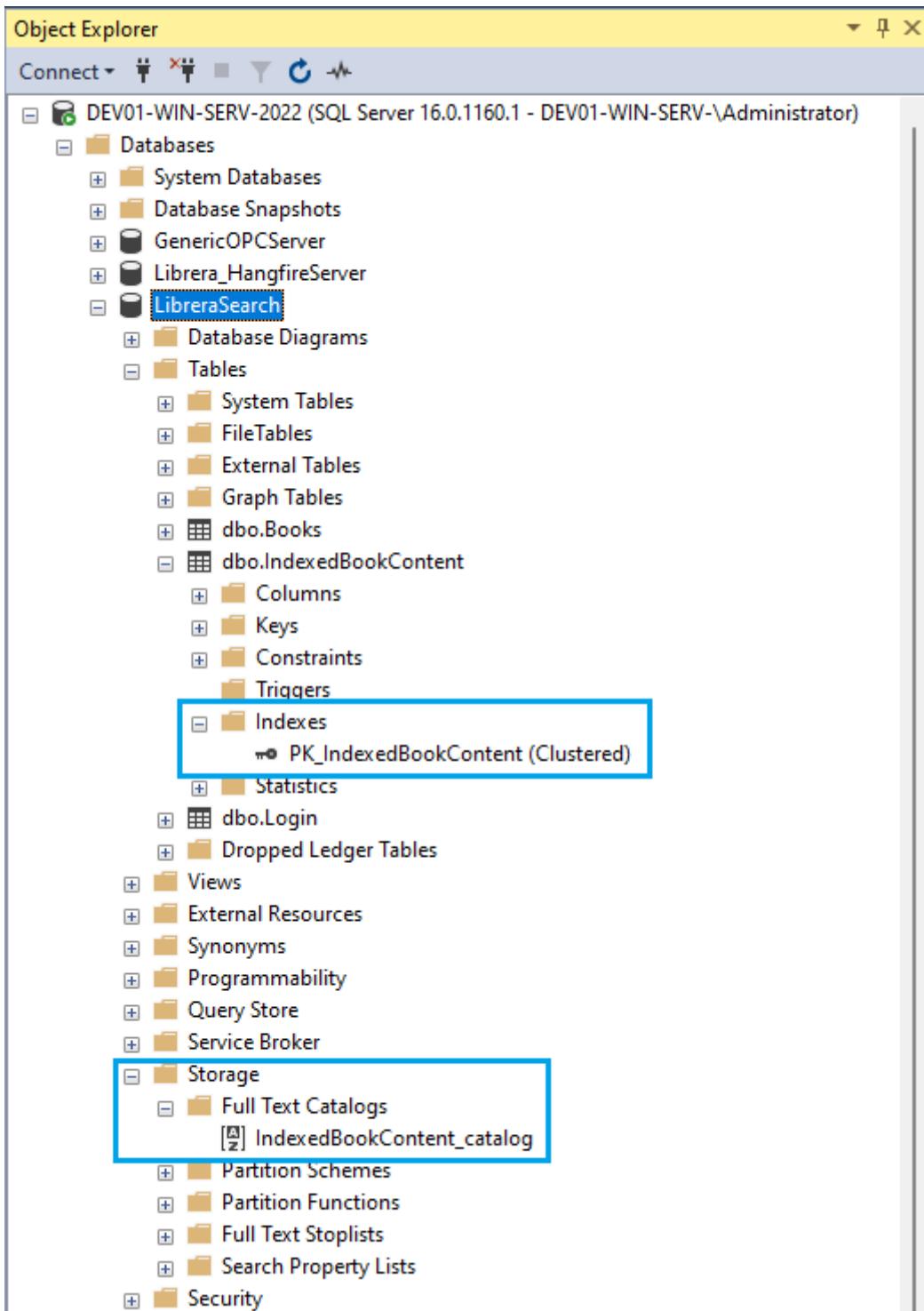
It's likely, however, that they're already present on your system, unless you performed a very minimal installation.

2. Initial FS Creation I typically encrypt files, not whole partitions, so I combine dm-crypt with the losetup loopback device maintenance tool. In the bare language of the Unix shell, here are the steps to create and mount an encrypted filesystem.

```
# Create an empty file sized to suit your needs. The one created # in this example will be a sparse file of 8GB, meaning that no # real blocks are written. Since we will force block allocation # lateron, it would not make much sense to do this now, since # the blocks will be rewritten anyway.
dd of=/path/to/secretfs bs=1G count=0 seek=8
# Lock down normal access to the file
chmod 600 /path/to/secretfs
# Associate a loopback device with the file
losetup /dev/loop0 /path/to/secretfs
```

## RQ-06 | Database

The database is hosted by M. SQL Server 2022, all the schemas and T-SQL logic is in the project. Full-Text Search feature is needed, as we see in the next image:



# Deployment

To do.

# Software life cycle management

To do.

## Annexes

A console App is added to test the indexer.

A script is added to launch Chrome avoiding CORS