Storyboard PSE Demo Video 2

Shot #01

Thema

Intro

Dauer

10/10

Inhalt /Beschreibung

Intro mit Logo von eonum, Projekt PSE mit allen Namen

Off-Stimme

-

Shot #02

Thema

KLV Abschluss

Dauer

40/50

Inhalt /Beschreibung

Man sieht die Frontpage von Medcodesearch. Man geht auf den Reiter ‚Gesetze und Reglemente', wo man dann eine Suchanfrage startet. Es wird ein Resultat angeklickt, welches dann durchgescrollt wird. Man geht via Breadcrumbs eine Ebene zurück und klickt ein anderen Eintrag an. Man kann dann noch die ‚ähnliche Code' anschauen und durchklicken.

Off-Stimme

Medcodesearch wurde nun mit der Krankenkassen Leistungsverordnung als durchsuchbare Quelle erweitert. Hierfür werden die jährlich aktualisierten PDF Dokumente mit einem Crawler aus dem Web geholt, geparst und in die Datenbank gespiesen. Via Elasticsearch kann man nun den gesamten Inhalt via Frontend durchsuchen. Die Darstellung der Suchresultate wurde gemäss Kundenwunsch angepasst. Es wurde noch Breadcrumbs hinzugefügt mit denen die Navigation im KLV einfacher wird und verwandte KLV-Massnahmen werden automatisch verlinkt.

Shot #03

Thema

KLV Abschluss

Dauer

40/90

Inhalt /Beschreibung

Man sieht das originale KLV Dokument, welches eine einigermassen klare Struktur hat. Datenbank-Schema aufzeigen, welches die Struktur modelliert.

Off-Stimme

Es ist uns gelungen, im KLV-Anhang eine Struktur zu finden, so dass wir diesen Anhang als strukturierte Quelle integrieren konnten. Die Integration als strukturierte Quelle bedeutete einen grösseren Programmieraufwand, bietet jedoch deutlich mehr Vorteile beim Verknüpfen verschiedener Datenbankeinträgen. Nach dem erfolgreichen Abschluss der KLV Integration, einigten wir uns mit eonum darauf, im zweiten Teil des Praktikums ein Proof of Concept zur Integration von unstrukturierten Quellen zu erarbeiten.

Shot #04

Thema

Proof of Concept - Einleitung

Dauer

40/130

Inhalt /Beschreibung

Man sieht das MKB, welches keine wirkliche Struktur hat. Vielleicht Auflistung machen zB. Abschnitt, Beispiel, Beispiel, Liste, Tabelle, Liste, Abschnitt, Highlight, wo keine erkennbare Struktur vorhanden ist.

Off-Stimme

Das neue Ziel war, herauszufinden, ob es möglich ist ganze PDFs in Elasticsearch zu indexieren ohne grosse Verarbeitung, wie wir es nun mit dem KLV gemacht haben. Auf Wunsch von eonum sollen wir es mit dem Dokument ‚Medizinischen Kodierhandbuchs' zeigen, welches ein sehr unstrukturiertes Format hat. Da das Medizinische Kodierhandbuch das Komplexeste der zu integrierenden Dokumente ist, sollte der Proof of Concept auch für weniger komplizierte Dokumente einwandfrei funtionieren.

Shot #05

Thema

Proof of Concept - Einleitung

Dauer

30/160

Inhalt /Beschreibung

Folie mit „Es funktioniert". Ein PDF-Icon mit Titel darunter und Seitenanzahl wird vom Raketask in viele Einzelseiten zerlegt. AM

Ende wird solch eine Einzelseite rechts aus dem Bild geschoben.

Off-Stimme

In den letzten zwei Wochen, in denen wir noch am Proof of Concept gearbeitet haben, konnten wir beweisen, dass dies geht. Man braucht hierfür nicht mehr ein Parser, welcher alle Edge-Cases berücksichtigt und eine komplizierte Implementierung voraussetzt. Die einzige Verarbeitung die es nun noch braucht, um die Dokumente durchsuchbar zu machen sind folgende: Ein Dokument kann via Rake-Task in einzelne Seiten zerstückelt werden.

Shot #06

Thema

Proof of Concept - Umsetzung

Dauer

45/205

Inhalt /Beschreibung

Die Einzelseite kommt von links ins Bild. Es wird nun der Text extrahiert in ein Text-Icon. Die Einzelseite wird auch in ein Base64 umgewandelt. Dann kommt noch ein französisches und italienisches, bei welchem dasselbe passiert. Wird dann in DB geschoben.

Off-Stimme

Man muss via Gems den Text aus den einzelnen PDF-Seiten herauslesen, sowie die PDF-Seite in Base64 umwandeln. Den herausgelesenen Text und der Base64-Code wird in die Datenbank geschrieben. Bei mehrsprachigen Dokumenten muss man diese Verarbeitung jeweils für die erste Seite aller PDFs machen, dann für die zweite Seite etc. also eine parallele Verarbeitung. Pro Quelldokument braucht es ein Model, welche alle Sprachen beinhaltet. (Man könnte auch ein Model machen welches verschiedene Quelldokumente enthält, doch dies würde die Durchsuchung verlangsamen).

Shot #07

Thema

PoC - Umsetzung

Dauer

45/250

Inhalt /Beschreibung

Flowchart machen mit API-GET request, welcher nach Text sucht. Extrahierte Texte in DB werden durchsucht. API-Response ist nun das Textschnipsel. Es wird auf den Textschnipsel geklickt, es gibt eine neue API-GET Request, welches nun Base64 anfordert von dieser Seite und es nun umgewandelt im Frontend anzeigt.

Off-Stimme

Elasticsearch kann nun den extrahierten Text in der Datenbank indexieren und durchsuchbar machen. Wenn man nun via Frontend nach etwas sucht, geht die Suchanfrage via Elasticsearch die extrahierten Texte durchsuchen. Elastic Search gibt eine ID zurück, wo diese Übereinstimmung in der Datenbank zu finden ist. Um nun das Formatierungsproblem des unstrukturierten Inhalts zu lösen, kann man als Backend-Response das Base64 zurückgeben. Dies kann man dann im Frontend wieder in das originale PDF zurückkonvertieren mithilfe einer Javascript PDF-Viewer Library.

Shot #08

Thema

PoC Umsetzung

Dauer

20/270

Inhalt /Beschreibung

Flowchart mit PDF Funktionen. Suchbegriff-Highlighting kann gechained werden. Previous/Next Page, welches auf vorherige/nächste Base64 in der DB zeigt.

Off-Stimme

Mithilfe dieser Library ist sogar die Durchsuchung des PDFs möglich, was das Highlighten der Suchbegriffe im Original-PDFs ermöglicht. Wenn man nun weiterlesen möchte auf der nächsten Seite, kann man einfach ‚weiterblättern'. Dabei kommt macht es ein Backend-Request, der das Base64 der nächsten Seite anfordert und dann wieder anzeigt.

Shot #09

Thema

PoC Umsetzung

Dauer

20/290

Inhalt /Beschreibung

Folie mit mehreren ‚Dokumenten'. Vor und Nachteile Liste: Ein Model pro neues Dokument. Anpassungen im Controller. Eine Methode mehr im Rake-Task. Vorteil: Effizient, schnell, einfach. Nachteile: Verknüpfbarkeit schwieriger, Daten unstrukturiert in DB, nur durchsuchbar.

Off-Stimme

Bei der Umsetzung haben wir stetig beachtet, dass das ganze skalierbar sein muss. Mithilfe dieses Proof of Concept sollte es nun möglich sein, neue unstrukturierte Quellen mit verhältnissmässig wenig Aufwand bei Medcodesearch hinzuzufügen. Dabei

muss lediglich ein Model pro neues Dokument erstellt werden, im Controller eine Methode angepasst werden und eine neue Rake-Task Methode geschrieben werden. Wir haben nun ein Weg aufgezeigt, wie man schnell mit wenig Aufwand neue Quellen hinzufügen und durchsuchbar machen kann.

Shot #10

Thema

PoC Demo

Dauer

60/350

Inhalt /Beschreibung

Man sieht den Frontend Prototypen. Man sucht ein Begriff und klickt das Snippet an. Es wird die PDF Seite nagezeigt und man hovert mit der maus über den gehighlighteten Begriff. Es wird die Sprache umgestellt und ein französischer Begriff gesucht. Man kann dann doch weiterblätter. Am Ende noch ein Suchsnippet anzeigen., welches ein komisches Snippet hat.

Off-Stimme

Damit der Proof of Concept nicht nur theoretisch in einem Dokument beschribene wurde, haben wir natürlich all dies code-technisch realisiert. Das Backend wurde im bestehenden Projekt in einem neuen Branche implementiert. Für das Frontend haben wird jedoch ein eigenes Projekt erstellt und darin ein Throwaway Prototyp gebaut. Die Umsetzung des Frontends ist an Medcodesearch angelehnt. Wenn man in der Suchleiste nun etwas eingibt, werden die Snippets aus dem Dokument mit der jeweiligen Seitennummer angezeigt. Sobald man darauf klickt, wird das PDF gerendert und der gesuchte Begriff wird gehighlighted. Die Suche ist auch in mehreren Sprachen möglich und weiterblättern im Dokument kann man auch.
Natürlich hat es auch hier noch Details, welche nicht komplett ausgearbeitet sind, da es sich ja um ein Proof of Concept und nicht um ein ausgereiftes Produkt handelt. Beispielsweise bei Tabellen im Querformat wird der Text etwas speziell extrahiert und ist nicht optimal durchsuchbar, wie hier zu sehen ist. (Evtl. Begriff suchen in Tabelle, welcher dann nicht gefunden wird). Doch auch dies sollte lösbar sein, indem man den extrahierten Text leicht verarbeitet.

Shot #11

Thema

Abschluss

Dauer

45/395

Inhalt /Beschreibung

Links/Rechts Parser/Komplettindexierung Vergleich. Vor und Nachteile aufschrieben auf Folie.

Off-Stimme

Es war spannend und lehrreich, mit verschiedenen Ansätzen unstrukturierte PDF Quellen in eine Suchmaschiene abzubilden. Wenn es reicht ein Dokument Seitenweise anzuzeigen und keine komplexe Verknüpfbarkeit nötig ist, ermöglicht unser Proof of Concept sehr einfach PDF-Quellen volltext zu durchsuchen und anzuzeigen. Sollen Daten für komplexere Verknüpfungen verfügbar sein, dann lohnt es sich pro Quelle ein Parser zu erstellen wie wir das für die KLV Quelle gemacht haben. Wie immer im Leben muss man auch in diesem Bereich abwägen zwischen Vor- und Nachteilen. Man will immer das beste Resultat mit dem geringsten Aufwand erreichen.

Shot #12

Thema

Retro

Dauer

45/395

Inhalt /Beschreibung

Alle Iterationen aufzeigen und Hauptpunkte pro Iteration aufzeigen

Off-Stimme

In der ersten Iteration ging es hauptsächlich um das Einarbeiten in die verschiedenen uns mehrheitlich unbekannten Technologien. Auch das Lesen und Verstehen des bestehenden Codes war sehr zeitaufwändig.
Während der zweiten Iteration waren wir mehrheitlich mit dem KLV integrieren beschäftigt. Hierfür wurde viel am Back- und Frontend gearbeitet. Der Parser und der Crawler wurde hier erstellt.
In der dritten Iteration arbeiteten wir an der Codequalität, verbesserten das Design im Frontend und starteten mit dem PoC.
In der vierten und kürzesten Iteration vollendeten wir den PoC und die dazugehörige Dokumentation.