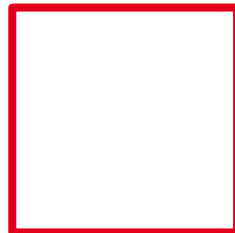
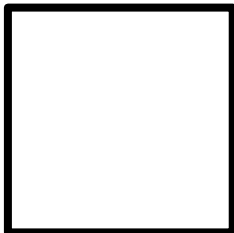


White Text in Red Box





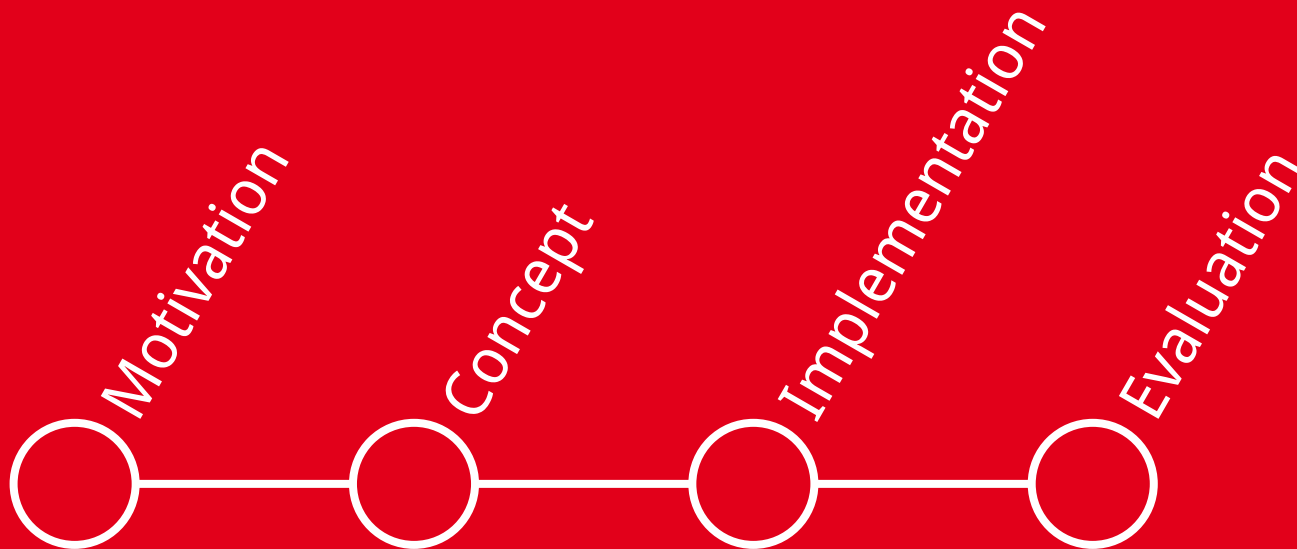
Hello

Torben Reetz
7th Semester SSD

A Recommender Framework for Skills Management

@SinnerSchrader

Agenda

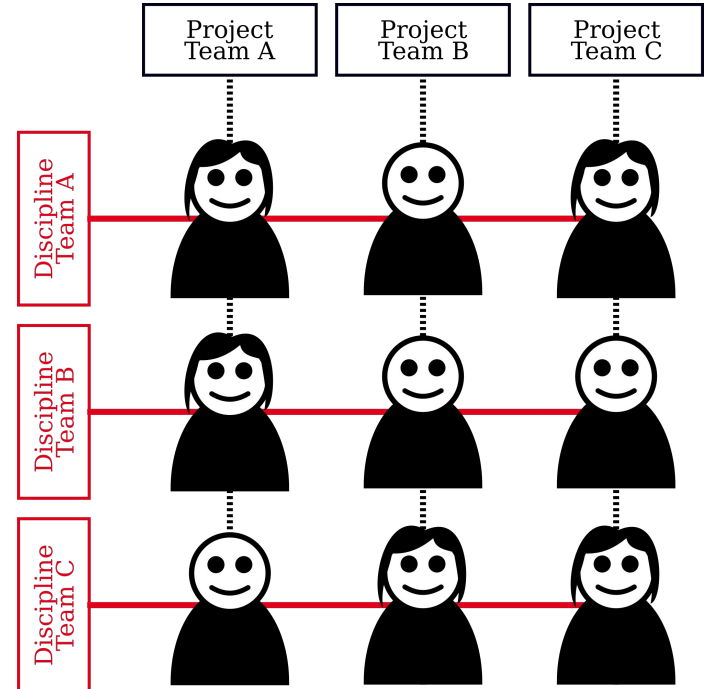


Motivation



- Hamburg based
- Full service web agency
- 459 full-time employees
- Revenue > 51M Euro (15/16)

- Domain specific teams
- Project teams
- Definitions:
 - Project Manager → Project team
 - Supervisor → Domain specific team



- Employees leave their teams
- Workload changes
- Shift in disciplines

Project managers frequently look for new team members.

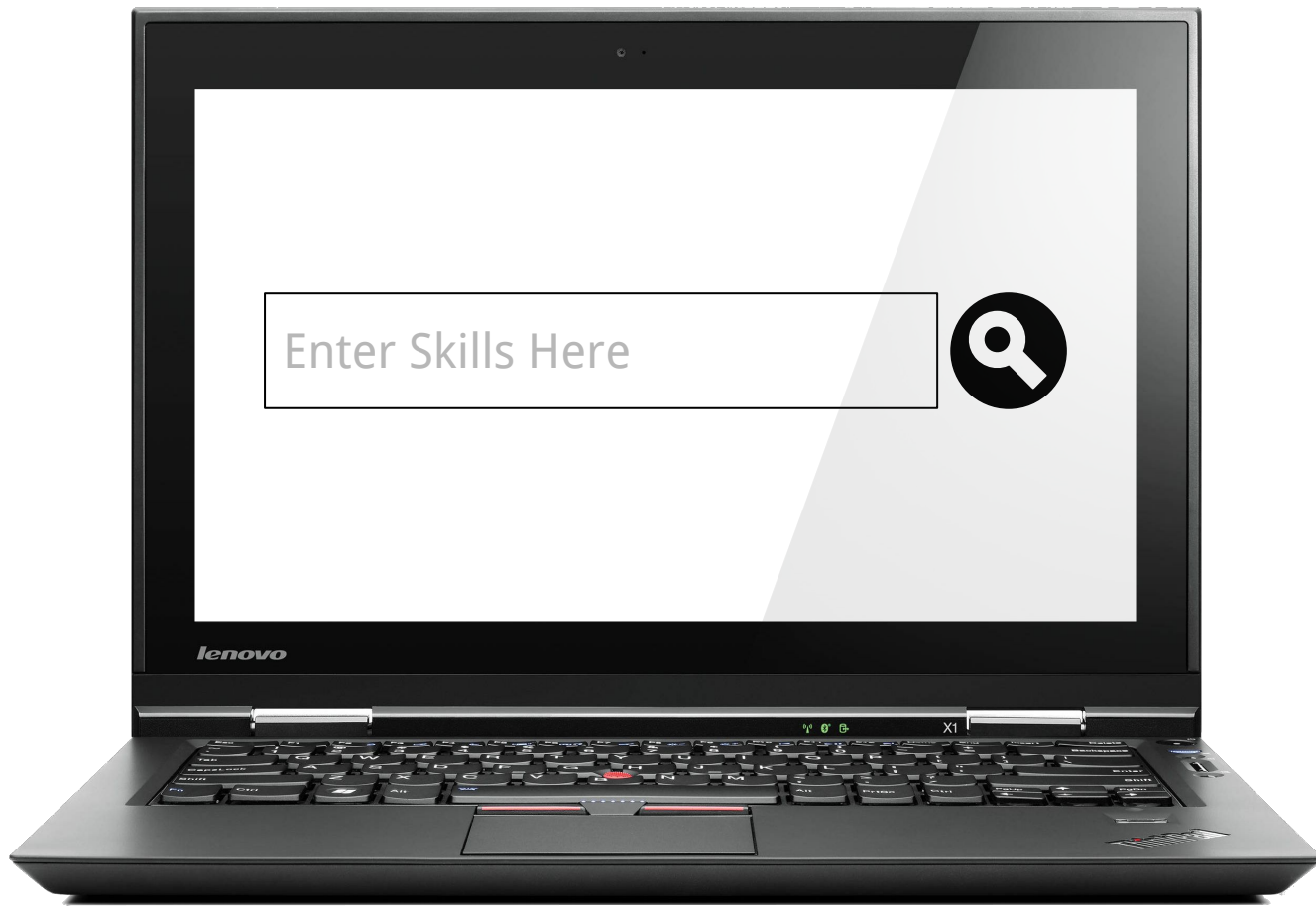
- Different experience and knowledge
- Different disciplines
- Different project setups

Employees search for people that can help to solve a specific problem.

- People search for other people that have specific skills
- Create a central source of information
- Focus: Motivation and Cooperation
- This thesis: Backend only
 - Visual concept / Frontend: Strecker

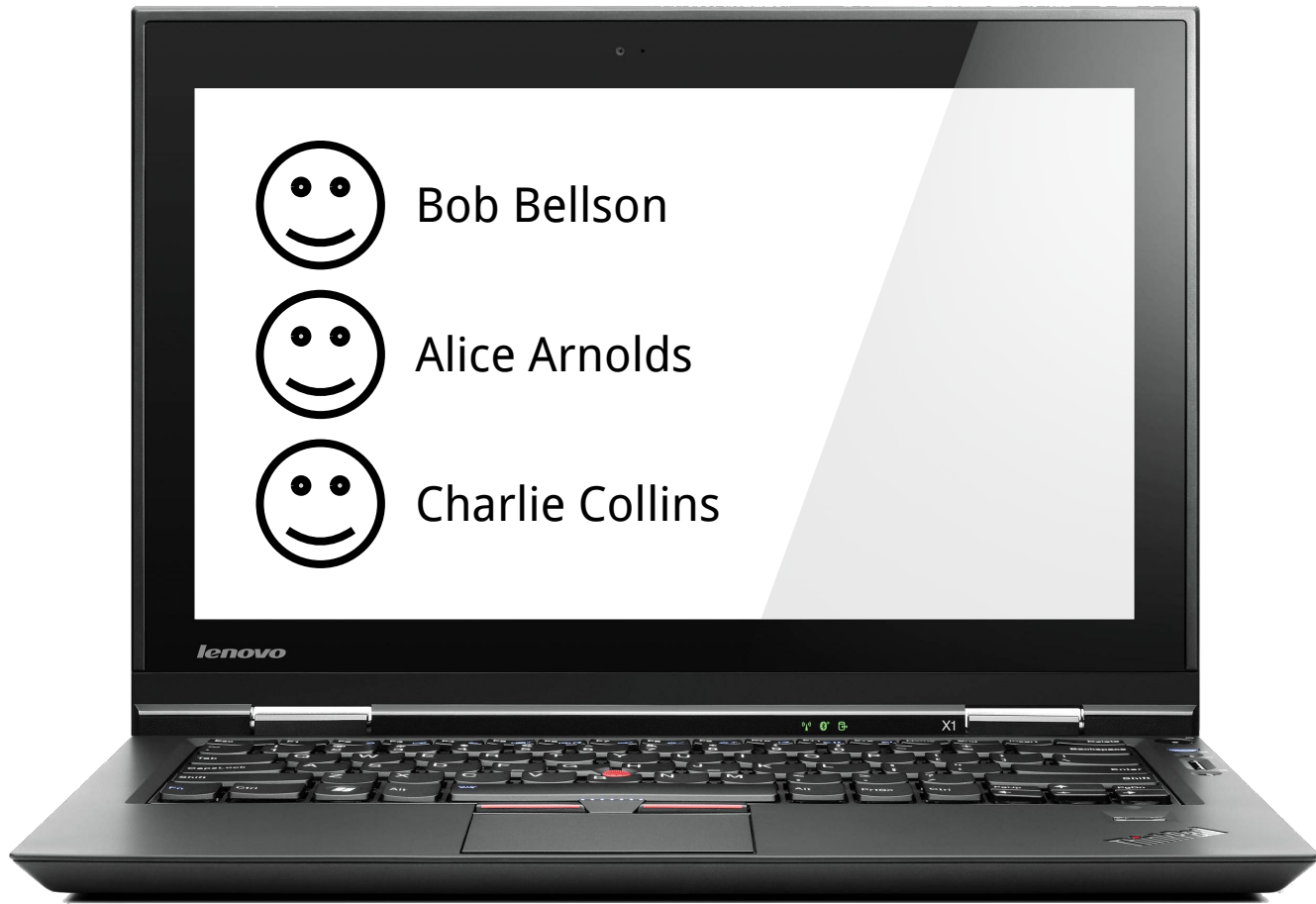
Concept













- Person Search
- Recommending skills to search
- Recommending similar profiles



Person Search

- Skill: ability of a specific person
- Levels
 - Skill level: knowledge
 - Will level: motivation
 - Four Step Scale (0-3)
- Fitness: Measurement of how well a person fits into a searched skill set

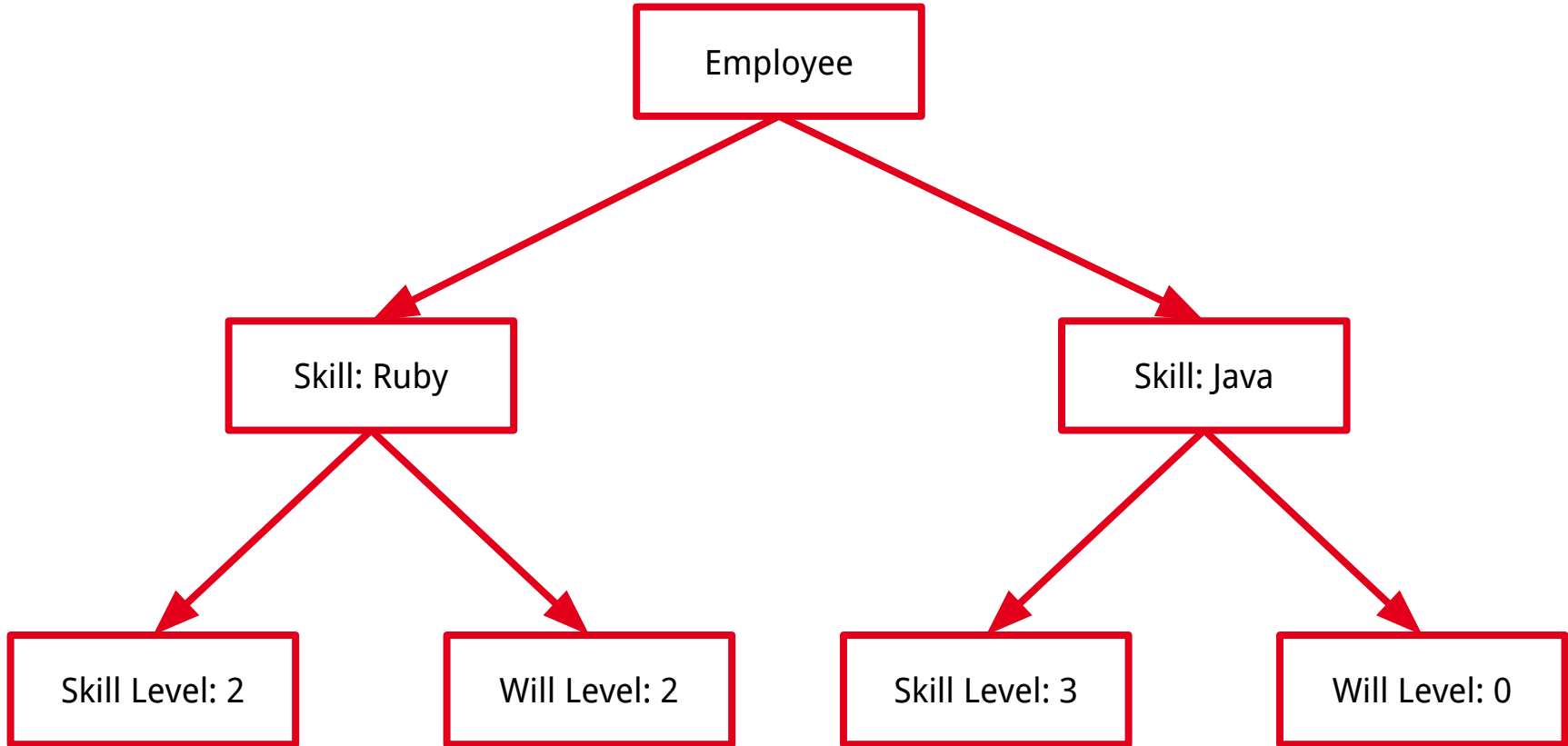
Four Step Scale

	Skill Level	Will Level
0	Novice	Uninterested
1	Basic Knowledge	Indifferent
2	Advanced Knowledge	Somewhat Interested
3	Expert	Highly Interested

Four Step Scale

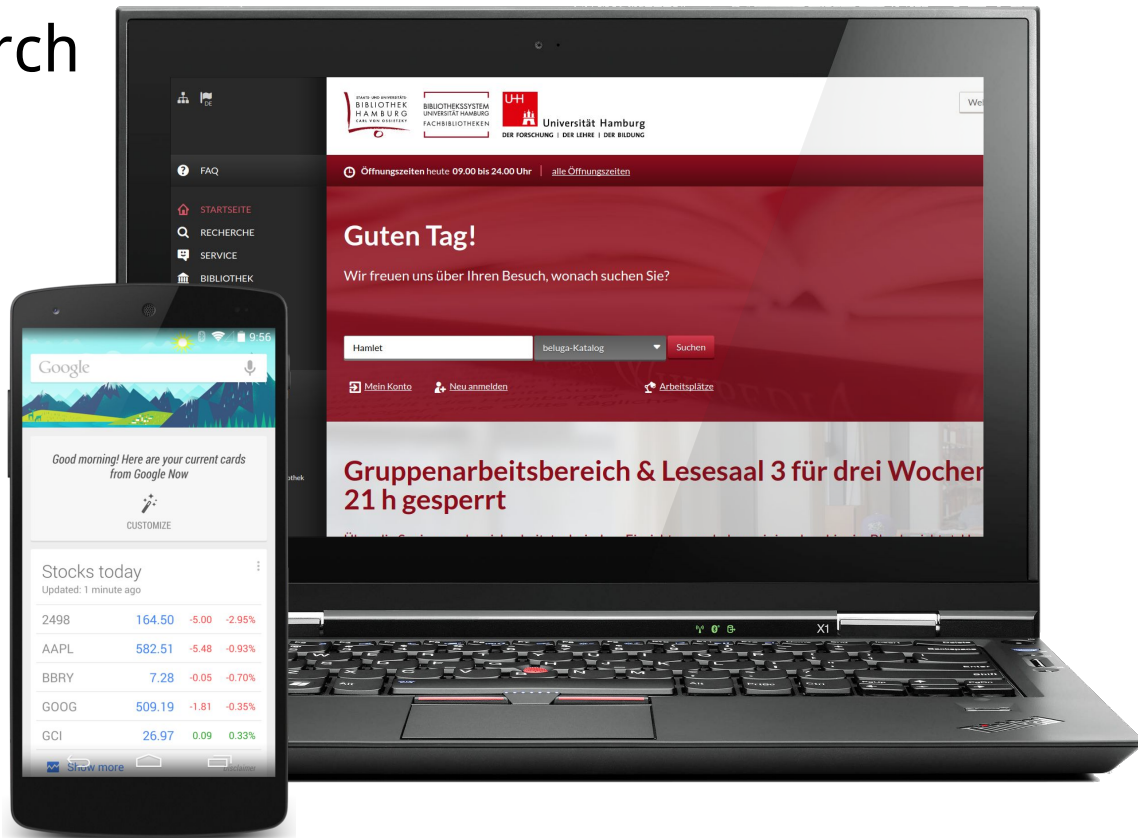
	Skill Level	Will Level
0	Novice	Uninterested
1	Basic Knowledge	Indifferent
2	Advanced Knowledge	Somewhat Interested
3	Expert	Highly Interested

- Skill Level = 0 → Person has little knowledge
- No knowledge → Skill not present



- User enters skills to look for
- Systems presents list of results
 - People that have all skills
 - Best match on first Position
- IR System

- Google
- Facebook Graph Search
- Siri/Alexa
- grep
- ...



“Information retrieval (IR) is finding material [...] that satisfies an information need from within large collections.”

Employees

“Information retrieval (IR) is finding material [...] that satisfies an information need from within large collections.”

Employees

“Information retrieval (IR) is finding material [...] that satisfies an information need from within large collections.”

Search Query

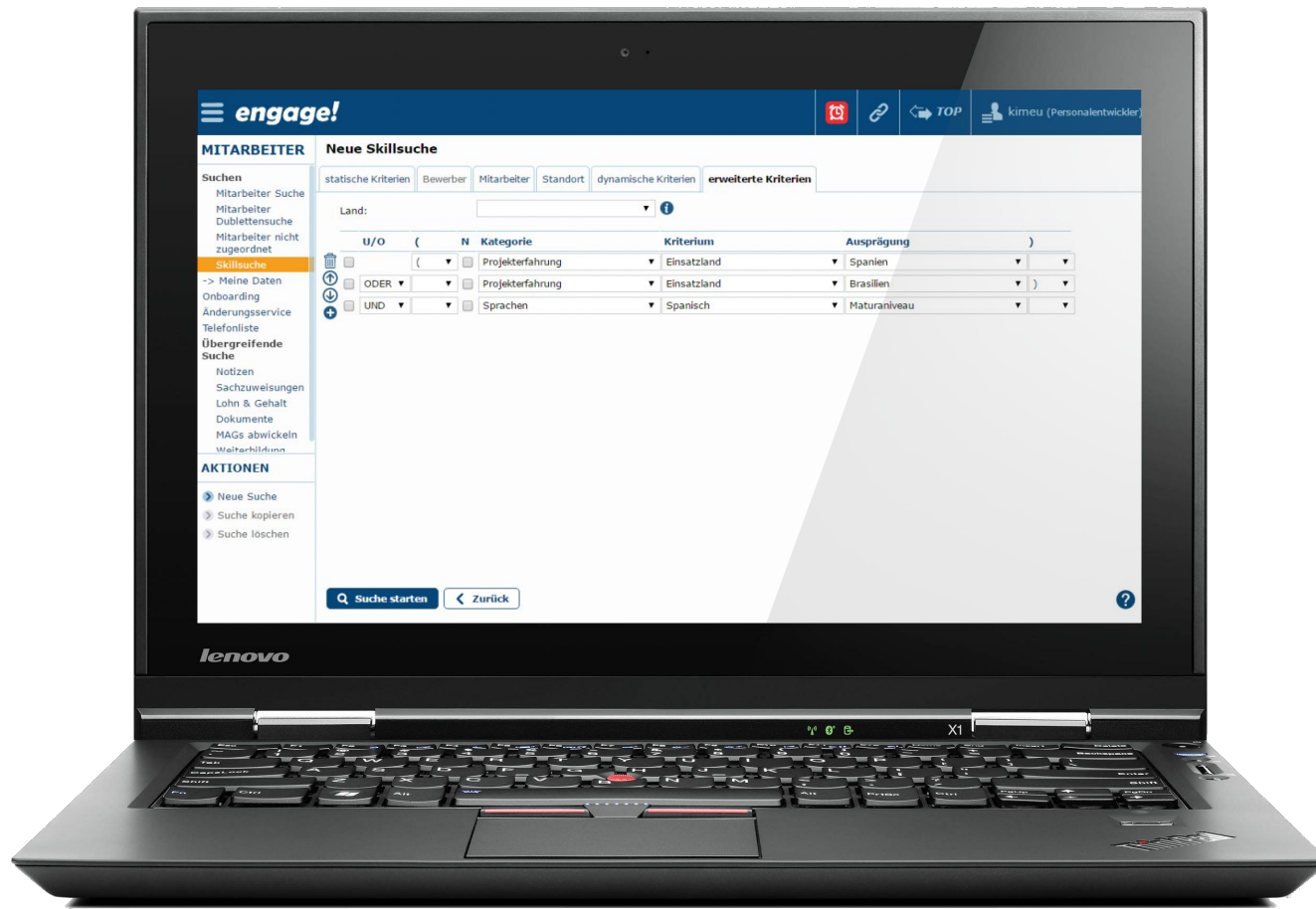
Employees

“Information retrieval (IR) is finding material [...] that satisfies an information need from within large collections.”

Search Query

Staff

- Boolean IR systems
 - Boolean operators
 - Example: Jira Query Language
 - “priority in (Blocker, Critical) AND project in (ProjA, ProjB, ProjC)”
- Ranked IR systems
 - Items ranked → Best match first
 - Example: Google





- Boolean Systems
- Complex queries
- Bloated interfaces
- No ranking

- Ranked IR system
- No complex queries
- Best match first
 - Fitness Score



**How to compose
the most
efficient team**

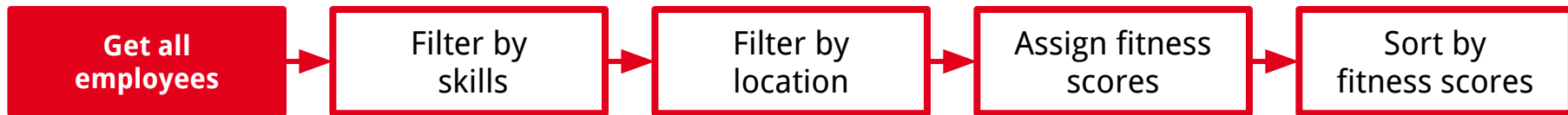
**How to measure
motivation**

- Adaption of Spoonamore et al.
- Weighted mean of factors
 - Average skill level in searched skills
 - Average will level in searched skills
 - Specialization (skill levels)
 - Specialization (will levels)
- Weighting parameters configurable



- $w_{as} = w_{aw} = w_{ss} = w_{sw} = 0.25$
- Notation: skill level | will level
- Java and Ruby
- In Hamburg

Example (Java and Ruby in HH)



Person	Location	Java	Ruby	C++
Alice	Hamburg	2 1	2 2	3 3
Bob	Hamburg	2 3	0 3	0 1
Charlie	Hamburg	3 3	2 1	1 2
Donald	Hamburg	3 3	-	2 2
Erika	Frankfurt	1 1	2 3	3 1

Get all
employees

**Filter by
skills**

Filter by
location

Assign fitness
scores

Sort by
fitness scores

Person	Location	Java	Ruby	C++
Alice	Hamburg	2 1	2 2	3 3
Bob	Hamburg	2 3	0 3	0 1
Charlie	Hamburg	3 3	2 1	1 2
Donald	Hamburg	3 3	-	2 2
Erika	Frankfurt	1 1	2 3	3 1

Example (Java and Ruby in HH)

Get all
employees

**Filter by
skills**

Filter by
location

Assign fitness
scores

Sort by
fitness scores

Person	Location	Java	Ruby	C++
Alice	Hamburg	2 1	2 2	3 3
Bob	Hamburg	2 3	0 3	0 1
Charlie	Hamburg	3 3	2 1	1 2
Donald	Hamburg	3 3	-	2 2
Erika	Frankfurt	1 1	2 3	3 1

Example (Java and Ruby in HH)

Get all
employees

Filter by
skills

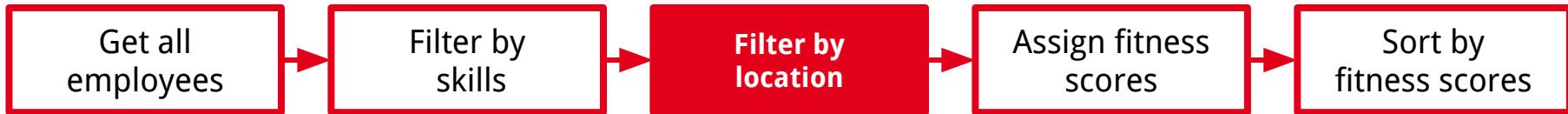
**Filter by
location**

Assign fitness
scores

Sort by
fitness scores

Person	Location	Java	Ruby	C++
Alice	Hamburg	2 1	2 2	3 3
Bob	Hamburg	2 3	0 3	0 1
Charlie	Hamburg	3 3	2 1	1 2
Donald	Hamburg	3 3	-	2 2
Erika	Frankfurt	1 1	2 3	3 1

Example (Java and Ruby in HH)



Person	Location	Java	Ruby	C++
Alice	Hamburg	2 1	2 2	3 3
Bob	Hamburg	2 3	0 3	0 1
Charlie	Hamburg	3 3	2 1	1 2
Donald	Hamburg	3 3	-	2 2
Erika	Frankfurt	1 1	2 3	3 1

Example (Java and Ruby in HH)



Person	Location	Java	Ruby	C++	f
Alice	Hamburg	2 1	2 2	3 3	0.44
Bob	Hamburg	2 3	0 3	0 1	0.71
Charlie	Hamburg	3 3	2 1	1 2	0.69
Donald	Hamburg	3 3	-	2 2	
Erika	Frankfurt	1 1	2 3	3 1	



#	Person	Location	Java	Ruby	C++	f
3	Alice	Hamburg	2 1	2 2	3 3	0.44
1	Bob	Hamburg	2 3	0 3	0 1	0.71
2	Charlie	Hamburg	3 3	2 1	1 2	0.69
	Donald	Hamburg	3 3	-	2 2	
	Erika	Frankfurt	1 1	2 3	3 1	

Example (Java and Ruby in HH)

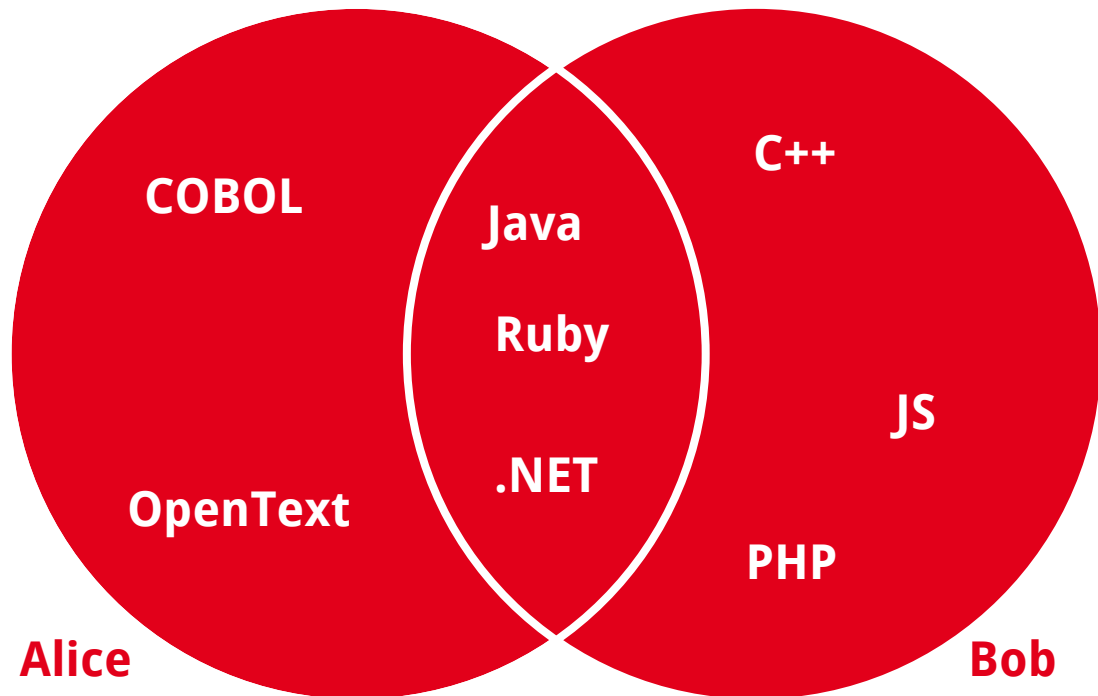


#	Person	Location	Java	Ruby	C++	f
1	Bob	Hamburg	2 3	0 3	0 1	0.71
2	Charlie	Hamburg	3 3	2 1	1 2	0.69
3	Alice	Hamburg	2 1	2 2	3 3	0.44

Recommending Similar Users

- Recommender System
 - Content-based filtering
- User = Set of Skills
- Jaccard Similarity Coefficient (JSC)

- Size of intersection: 3
- Size of union: 8
- JSE: $3 / 8$ (37.5%)



Recommending Skills to Search

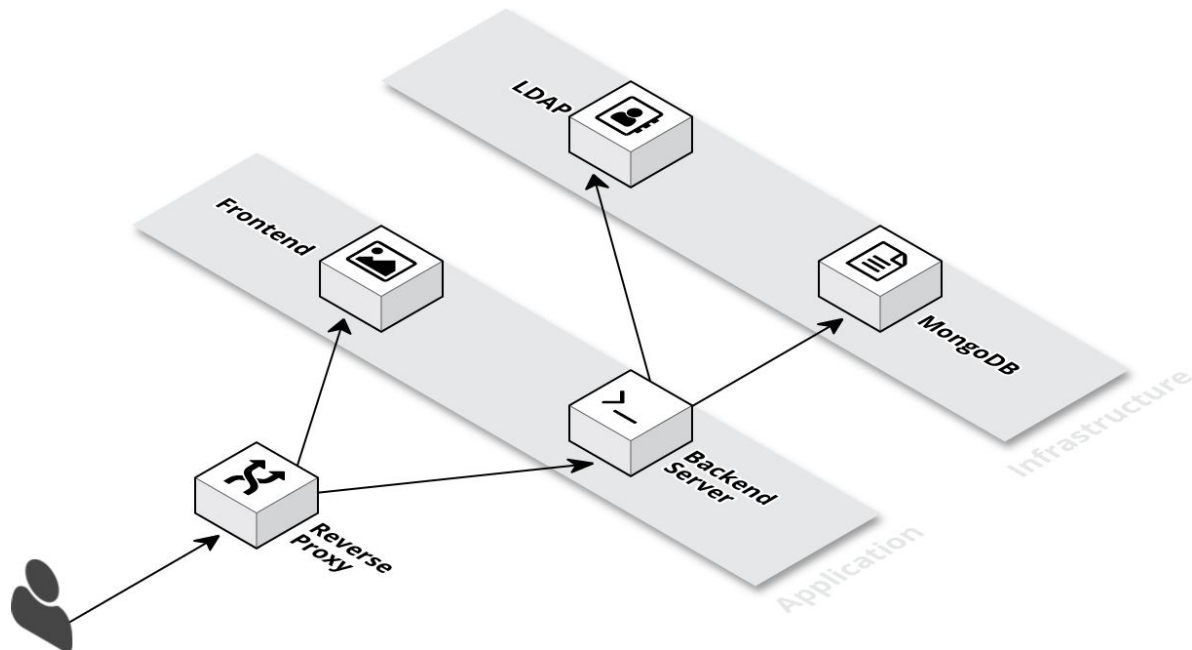
- Skills: name and icon
- No user context
- Item-based (collaborative) filtering
 - Examine what skills other users' searched for
 - Recommend those skills

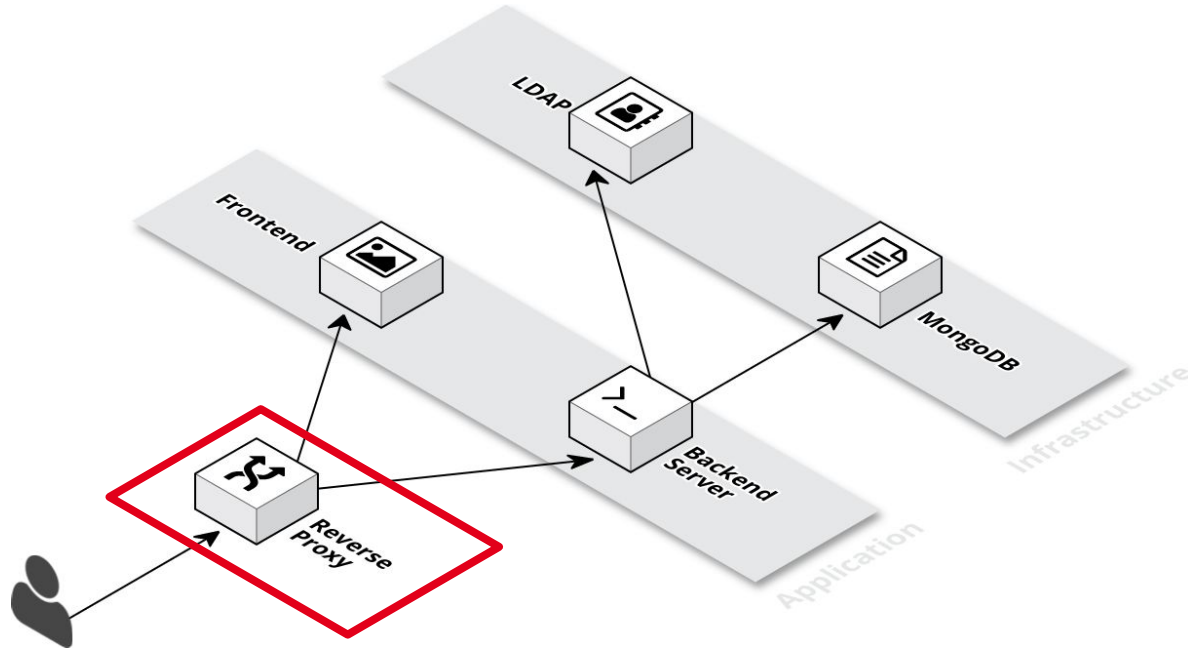
- Markov Chain
- States: Single Skills
- Transitions: Count of joint searches
- Generate Recommendations:
 - Get all suggestions from entered items
 - Aggregate suggestions (add counts)
 - Remove entered items
 - Sort by search count
 - Return first n items

Implementation

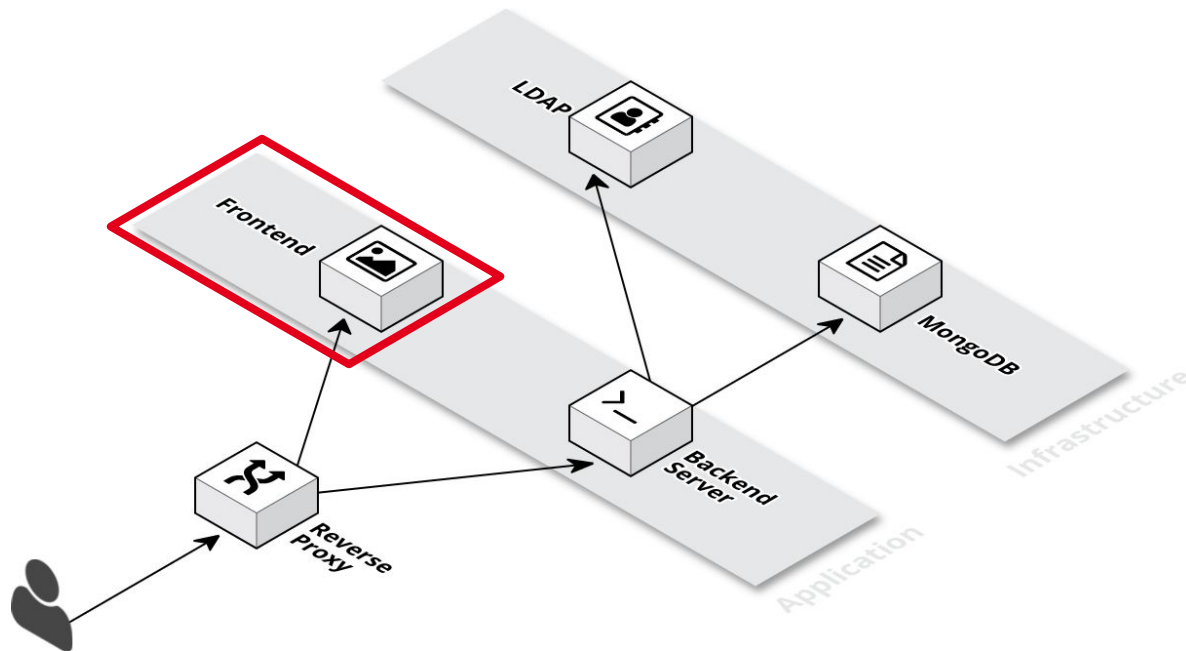


Application Structure

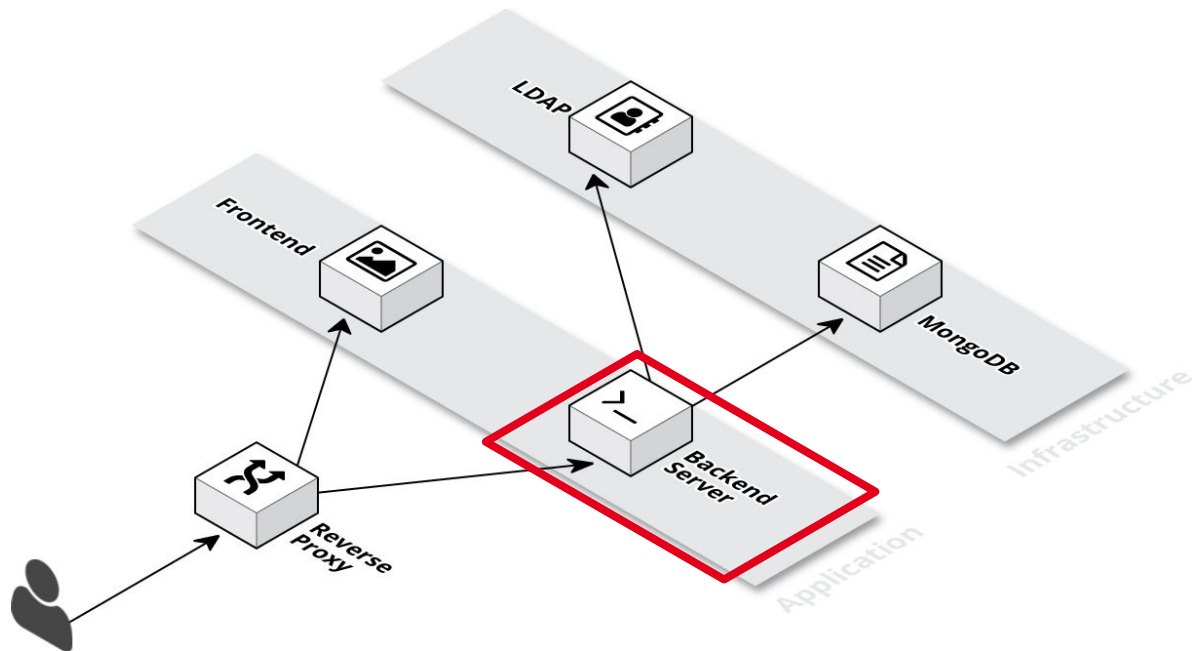




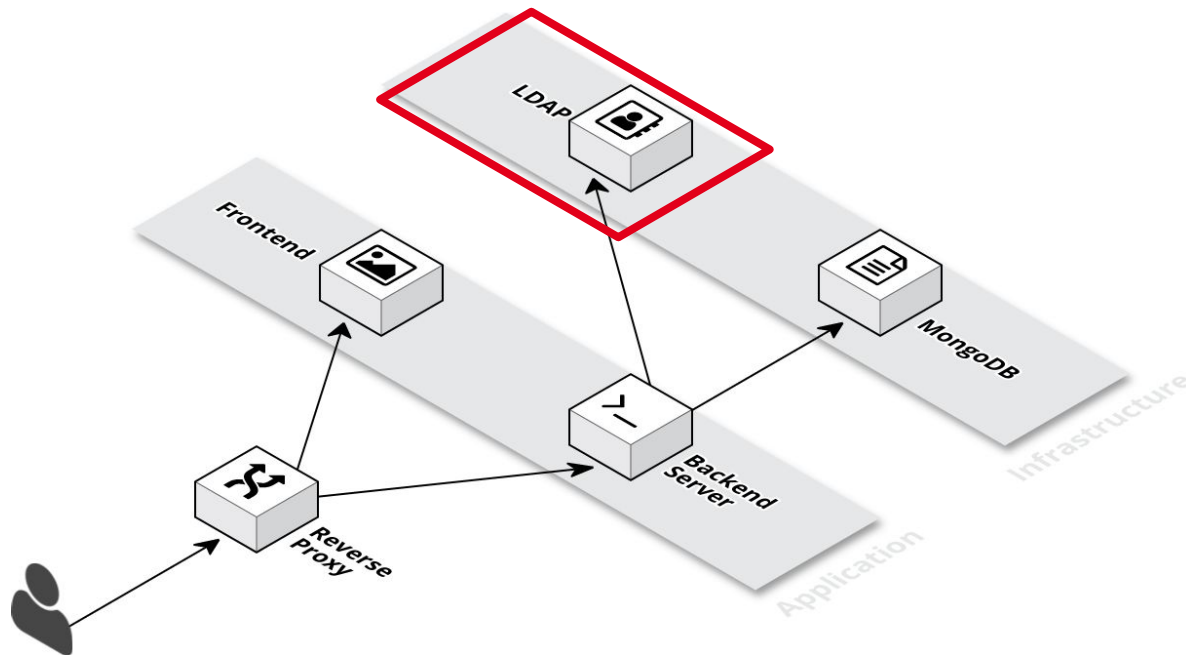
- Serve static files
- Forward API calls
- SSL endpoint



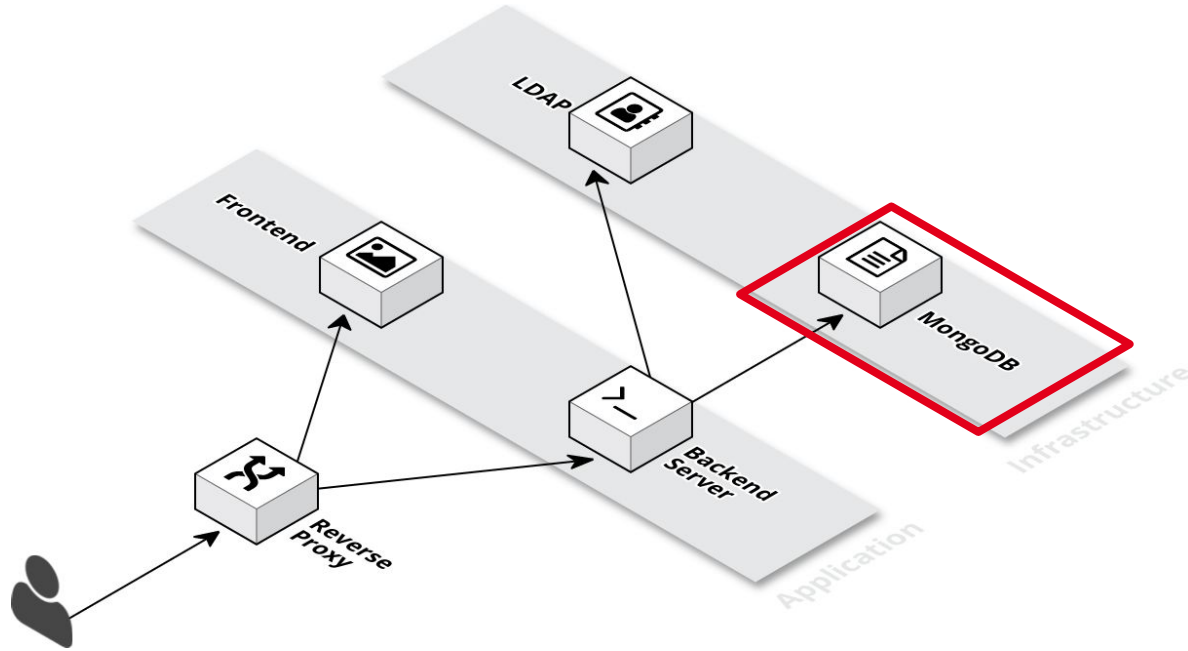
- Graphical Interface
- HTML, CSS, JS, Assets
- Executes API calls (AJAX)



- Main Application
- REST API



- Personal Data
- Authentication
- Used for most internal services



- Application data
- NoSQL













Evaluation



- Scalability
- Response times (1s)
- Search algorithm
 - Are results distributed uniformly?
 - Do algorithm's and managers' ratings correlate?

- Scalability ✓
- Response times (1s) (✓)
- Search algorithm
 - Are results distributed uniformly?
 - Do algorithm's and managers' ratings correlate?

- Scalability ✓
- Response times (1s) (✓)
- Search algorithm
 - Are results distributed uniformly? ✓
 - Do algorithm's and managers' ratings correlate?

- Online Survey (Google Forms)
- 41 participants (8% of the staff)
- 9 Questions

SkillWill

Alice

Alice hat die Fähigkeiten Java (4/4), AEM (3/4), Ruby (1/2) und .NET(3/4). Gesucht wird nach Java und AEM.

Wie gut passt Alice deiner Meinung nach auf die Suche?

	1	2	3	4	5	
passt gar nicht	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	passt perfekt

BACK

NEXT

Page 4 of 13

SkillWill

Faktor 1: Durchschnitt der Skills

Du suchst mit dem Tool nach bestimmten Fähigkeiten. In die Sortierung der gefundenen Personen wird der durchschnittliche Skill der jeweiligen Person in den gesuchten Fähigkeiten einbezogen.

Wie wichtig ist dir dieser Faktor?

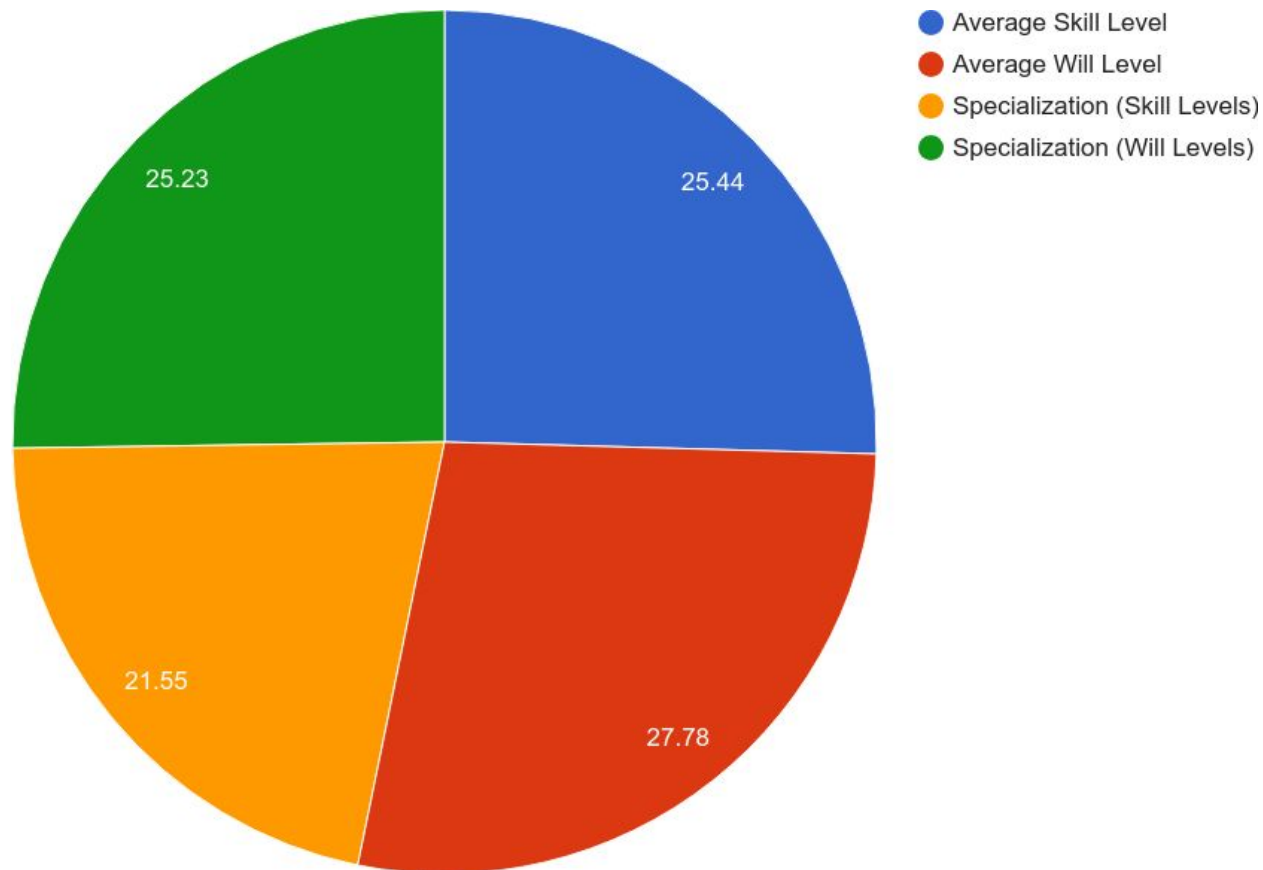
	1	2	3	4	5	
nicht wichtig	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	sehr wichtig

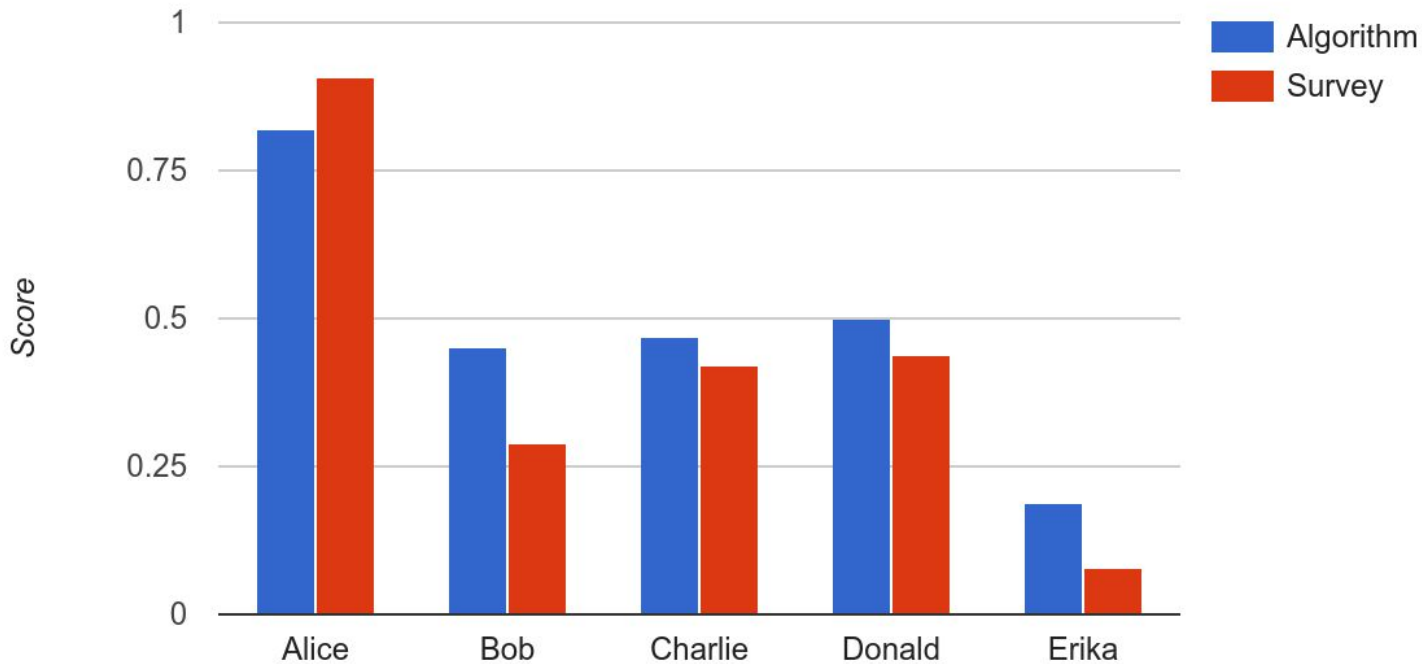
BACK

NEXT

Page 10 of 13

Evaluation (Survey)





- Two-tailed heteroscedastic T-Test
- $p \geq 0.1 \rightarrow$ No significant deviation
- $p \geq 0.05 \rightarrow$ Two data rows deviate significantly

- Scalability ✓
- Response times (1s) (✓)
- Search algorithm
 - Are results distributed uniformly? ✓
 - Do algorithm's and managers' ratings correlate?

- Scalability ✓
- Response times (1s) (✓)
- Search algorithm
 - Are results distributed uniformly? ✓
 - Do algorithm's and managers' ratings correlate? ✓



Universität Hamburg

DER FORSCHUNG | DER LEHRE | DER BILDUNG

FAKULTÄT

FÜR MATHEMATIK, INFORMATIK
UND NATURWISSENSCHAFTEN

That's it



Demo



Demo



<https://demo.torben.xyz>



Thanks



github.com/t0rbn/BSc
github.com/sinnerschrader*

Contact

Github: @t0rbn

Mail: 3reetz@informatik.uni-hamburg.de

More: <http://torben.xyz>

Image Sources

Thinkpad X1: <https://www.bhphotovideo.com>

Google Now on Nexus 5: <https://www.androidcentral.com/google-now>

Screenshot engage!: <https://www.infonika.com/hr-software/talent-management>

Application Architectures created with <https://cloudcraft.co/>

Octocat: <https://github.com/logo>

Java Logo: https://upload.wikimedia.org/wikipedia/en/thumb/3/30/Java_programming_language_logo.svg/412px-Java_programming_language_logo.svg.png

OpenJDK Logo: https://upload.wikimedia.org/wikipedia/commons/thumb/1/18/OpenJDK_logo.svg/2000px-OpenJDK_logo.svg.png

Swagger Logo: <https://avatars2.githubusercontent.com/u/7658037?v=3&s=400>

Junit Logo: <http://junit.org/junit4/images/junit5-banner.png>

Maven Logo: <https://maven.apache.org/images/maven-logo-black-on-white.png>

Git Logo: <https://git-scm.com/images/logos/logomark-orange@2x.png>

Gitlab Logo: https://upload.wikimedia.org/wikipedia/commons/thumb/1/18/GitLab_Logo.svg/2000px-GitLab_Logo.svg.png

Flapdoodle Logo: <https://avatars0.githubusercontent.com/u/1661811?v=3&s=280>

Unboundid Logo: https://media.licdn.com/mpr/mpr/shrink_200_200/AAEAAQAAAAAAAAUAAAAJGEyYmI1MDImLWFkMjktNGU3ZS1hOTk2LWM0MDJjZTQ5MzA2Mw.png

Jacoco Logo: <https://cdn.liviu tudor.com/wp-content/uploads/2016/02/Jacoco.png>

Spring Logo: https://upload.wikimedia.org/wikipedia/en/2/20/Pivotal_Java_Spring_Logo.png



Q&A

Bonus Slides

Requirements

- Person search
 - Enter skills → find best matching person
- User profiles
 - Skills, personal data, direct contact
 - Enter own skills (knowledge, motivation)
 - Login
- Management of registered Skills
 - Pool of predefined skills
 - Add new skills
 - Rename skills
 - Delete skills

- Different devices
 - Primary: Desktops
 - Mobile devices optional
- Browser support
 - Chrome, Firefox, Safari
 - No support for IE/Edge
- Response Times (RAIL)
 - 100ms to acknowledge input
 - 1s to finish rendering results
- Scalability
 - Increased number of users should not be a problem
 - Enlarge storage and computing resources

Related Work

- Ivanovska et. al: Algorithms for Effective Team Building
- Canós-Darós: An algorithm to identify the most motivated employees
 - General Motivation \leftrightarrow Task specific
 - Asking employees to rate their motivation generates suitable data
- Spoonamore et. al: Matching Sailors to Positions Based on Skill

- Multiple Factors
 - Rating
 - Pay grade
 - NECs
- Basic principle: weighted mean of factors
 - $S = \alpha \text{ ratingscore} + \beta \text{ paygradescore} + \gamma \text{ NECscore}$

Fitness Score Algorithm

$$V = \{x \in \mathbb{N}_0^+ \mid 0 \leq x \leq 3\}$$

$$S = \{Java, Ruby, C++ , \dots\}$$

$$E = \{x \in S \mid \text{employee has skill } x\}$$

$$Q = \{x \in S \mid \text{user searches for skill } x\}$$

$$v_s : E \mapsto V$$

$$v_w : E \mapsto V$$

Fitness Score (Factors)

$$a_s = \left(\sum_{x \in E \cap Q} v_s(x) \right) \cdot \frac{1}{|E \cap Q|}$$

$$a_w = \left(\sum_{x \in E \cap Q} v_w(x) \right) \cdot \frac{1}{|E \cap Q|}$$

$$s_s = \frac{\max(V) + a_s - \left(\left(\sum_{x \in E \setminus Q} v_s(x) \right) \cdot \frac{1}{|E \setminus Q|} \right)}{2\max(V)}$$

$$s_w = \frac{\max(V) + a_w - \left(\left(\sum_{x \in E \setminus Q} v_w(x) \right) \cdot \frac{1}{|E \setminus Q|} \right)}{2\max(V)}$$

$$f = \frac{w_{as} \cdot a_s}{\max(V)} + \frac{w_{aw} \cdot a_w}{\max(V)} + w_{ss} \cdot s_s + w_{sw} \cdot s_w$$

- Estimated Factors ≈ 0.25
- Setting all to 0.25
 - Simplifies algorithm
 - No drastic effect on accuracy

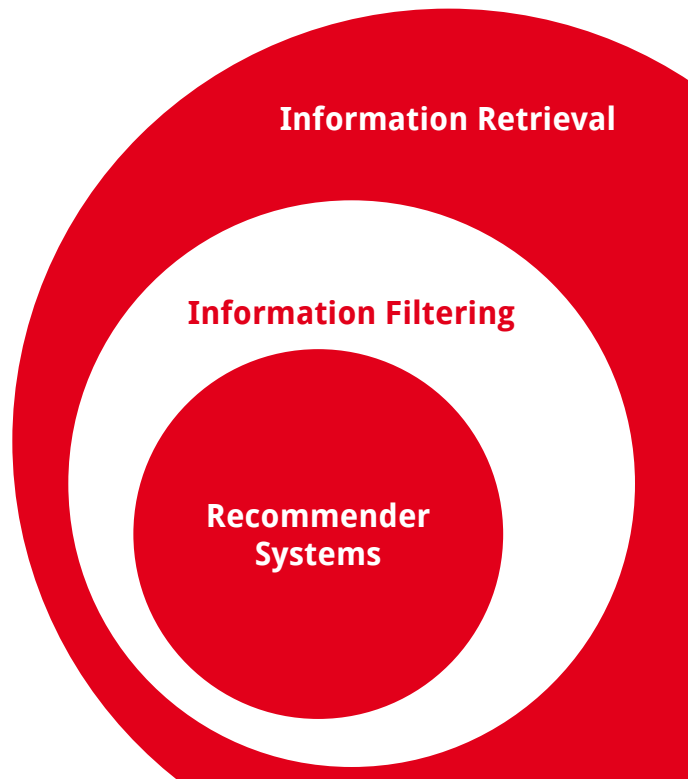
$$f = \frac{w_{as} \cdot a_s}{\max(V)} + \frac{w_{aw} \cdot a_w}{\max(V)} + w_{ss} \cdot s_s + w_{sw} \cdot s_w$$

$$\Rightarrow f = \frac{a_s + a_w}{4\max(V)} + \frac{s_s + s_w}{4}$$

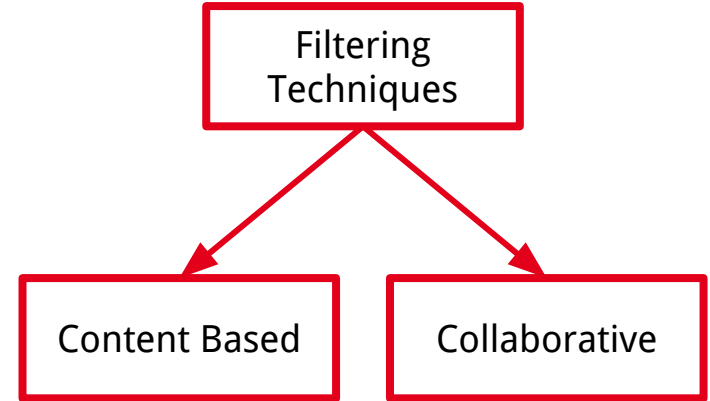
Recommender Systems

“information filtering systems that deal with the problem of information overload by filtering vital information fragment out of large amount of [...] information [sic]”

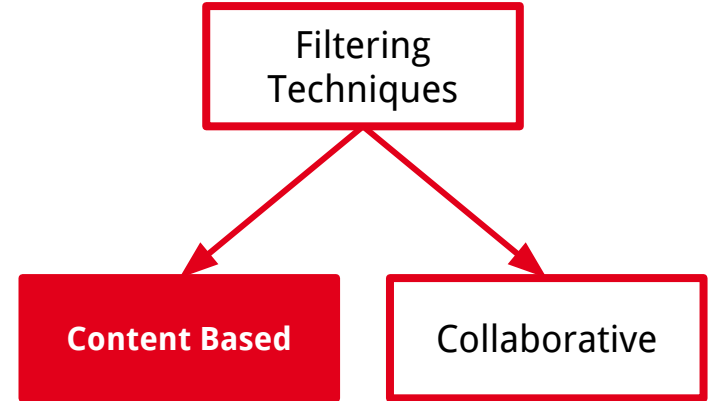
- Subset of IR systems
- Both find relevant information
- Recommender Systems → “zero query” IR
 - IR: user actively searches
 - Recommender: system proactively recommends
 - Person search is not a recommender system



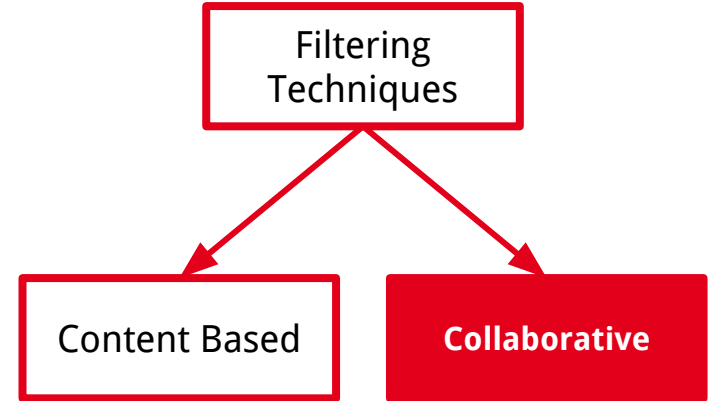
- Isinkaye et al.
- Techniques to find the items to suggest
- Hybrid filtering: combine multiple techniques



- Examine content
- Find items that are similar to the ones the user interacted with



- Users behave similarly
- Find “neighbours”
- Recommend items neighbours interacted with
- Subclasses:
 - Model based
 - Memory based
 - Item based
 - User based



Example: Amazon (Collaborative)





Recommending Skills to Search (Example)

Recommending Skills to Search (Example)



- Entered: Java, PHP
- Number to recommend (n): 1

	Java	PHP	CSS	COBOL
Java	-	7	3	1
PHP	7	-	9	5
CSS	3	9	-	8
COBOL	1	5	8	-

Recommending Skills to Search (Example)

Get all
suggestions

Aggregate

Remove Entered
Items

Sort by Count

Return first n

- PHP (7)
- CSS (3)
- COBOL (1)
- Java (7)
- CSS (9)
- COBOL (5)

	Java	PHP	CSS	COBOL
Java	-	7	3	1
PHP	7	-	9	5
CSS	3	9	-	8
COBOL	1	5	8	-

Recommending Skills to Search (Example)



- PHP (7)
- Java (7)
- CSS (12)
- COBOL (6)

	Java	PHP	CSS	COBOL
Java	-	7	3	1
PHP	7	-	9	5
CSS	3	9	-	8
COBOL	1	5	8	-

Recommending Skills to Search (Example)



- CSS (12)
- COBOL (6)

	Java	PHP	CSS	COBOL
Java	-	7	3	1
PHP	7	-	9	5
CSS	3	9	-	8
COBOL	1	5	8	-

Recommending Skills to Search (Example)



- CSS (12)
- COBOL (6)

	Java	PHP	CSS	COBOL
Java	-	7	3	1
PHP	7	-	9	5
CSS	3	9	-	8
COBOL	1	5	8	-



- CSS (12)

	Java	PHP	CSS	COBOL
Java	-	7	3	1
PHP	7	-	9	5
CSS	3	9	-	8
COBOL	1	5	8	-

API Endpoints

URL	Method	Feature
/login	POST	User Login
/logout	POST	User Logout

- Login to edit user's skills
- No login needed to search

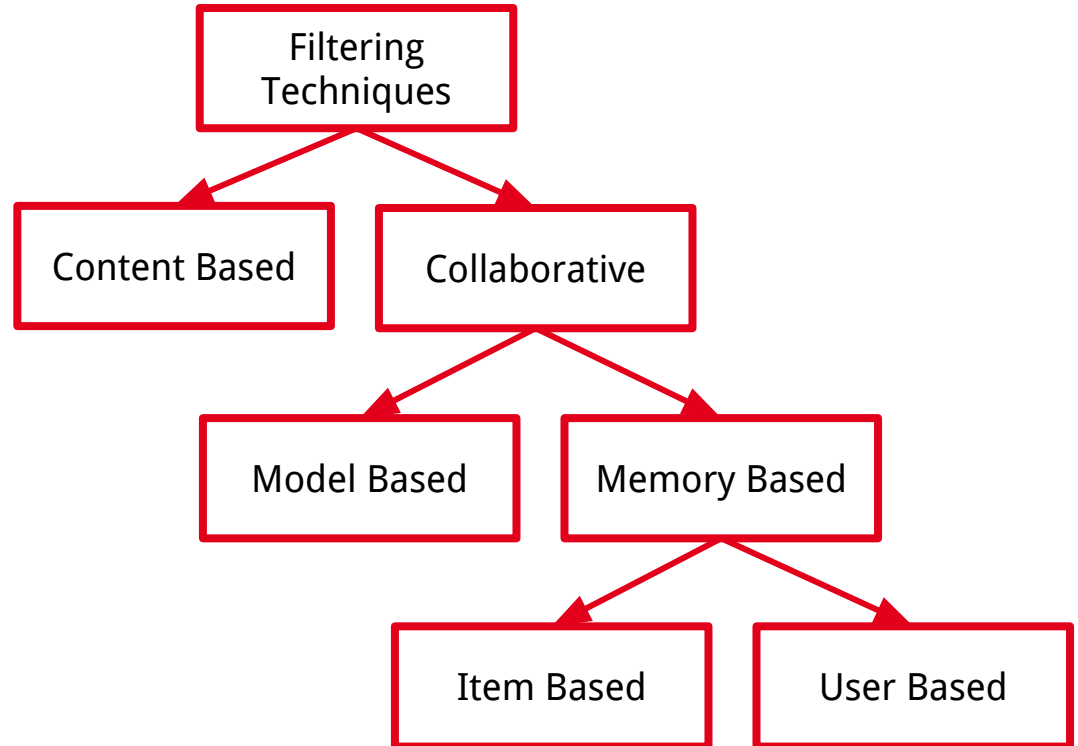
URL	Method	Feature
/users	GET	Main person search function
/users/{user}	GET	Get specific user's details
/users/{user}/skills	POST	Add/Update user's skills
/users/{user}/skills	DELETE	Remove skill from user's profile
/users/{user}/similar	GET	Recommend similar users

- New users added on their first login
- No removing of users

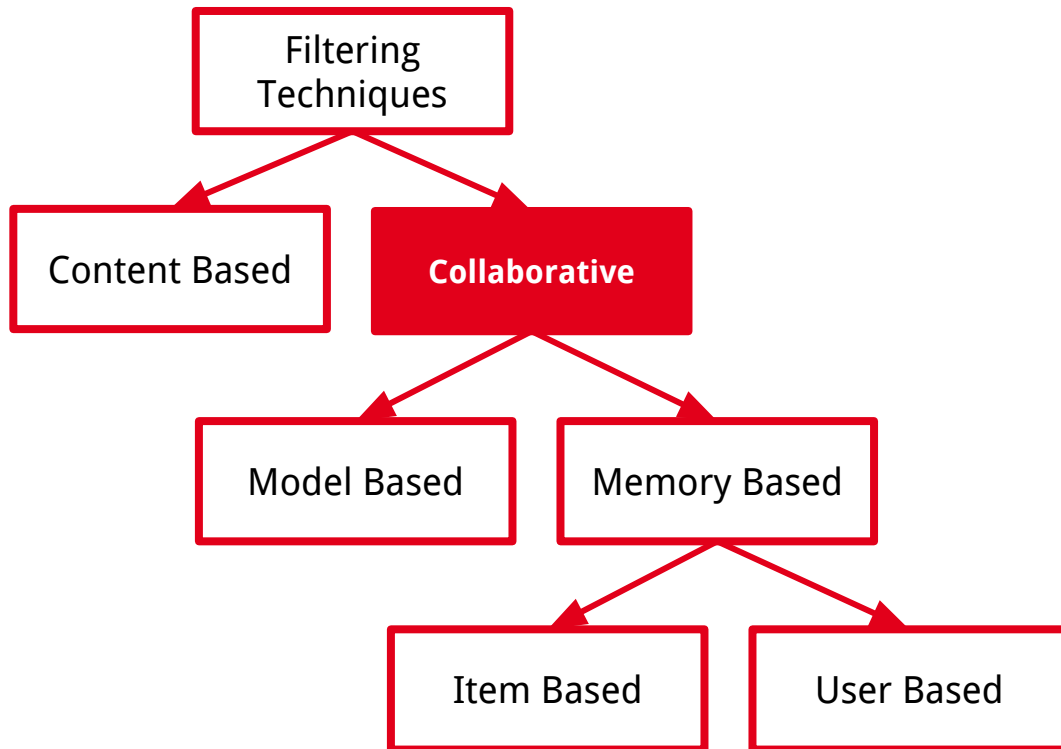
URL	Method	Feature
/skills	GET	Get all skills/text autocomplete
/skills	POST	Create new Skill
/skills/next	GET	Recommend next skill to enter
skills/{skill}	PUT	Edit skill (rename)
skills/{skill}	DELETE	Delete skill

Concept: Collaborative Filtering Techniques

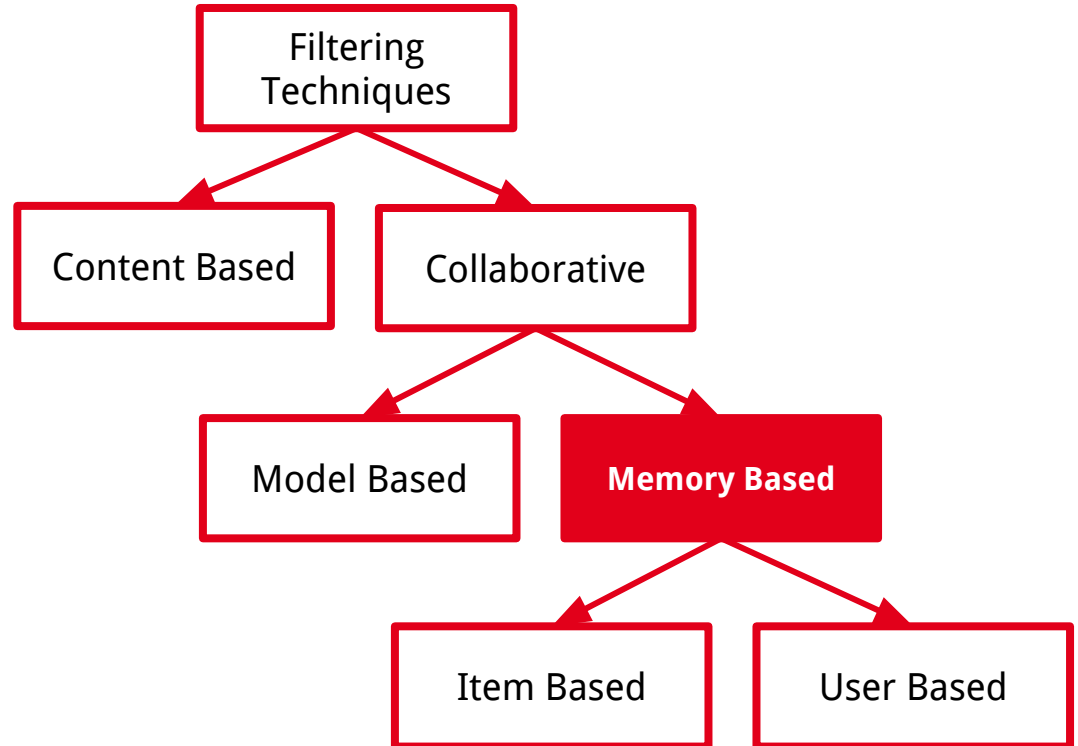
- Isinkaye et al.
- Techniques to find the items to suggest
- Hybrid filtering: combine multiple techniques



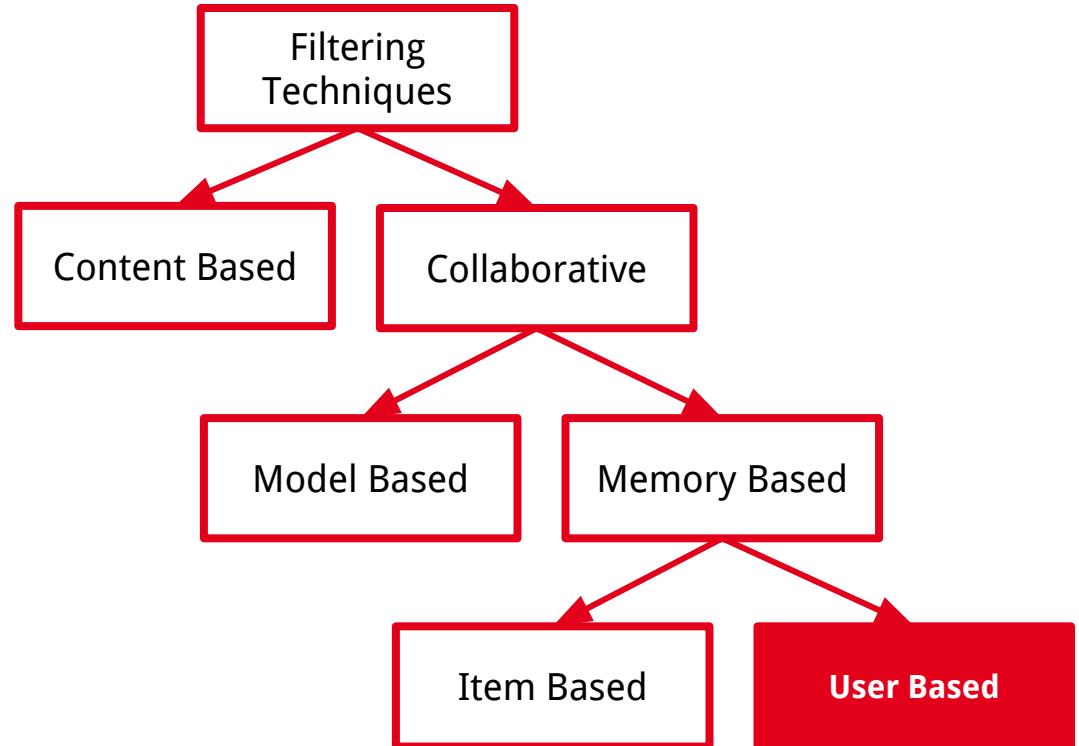
- People behave similarly
- Find neighbours



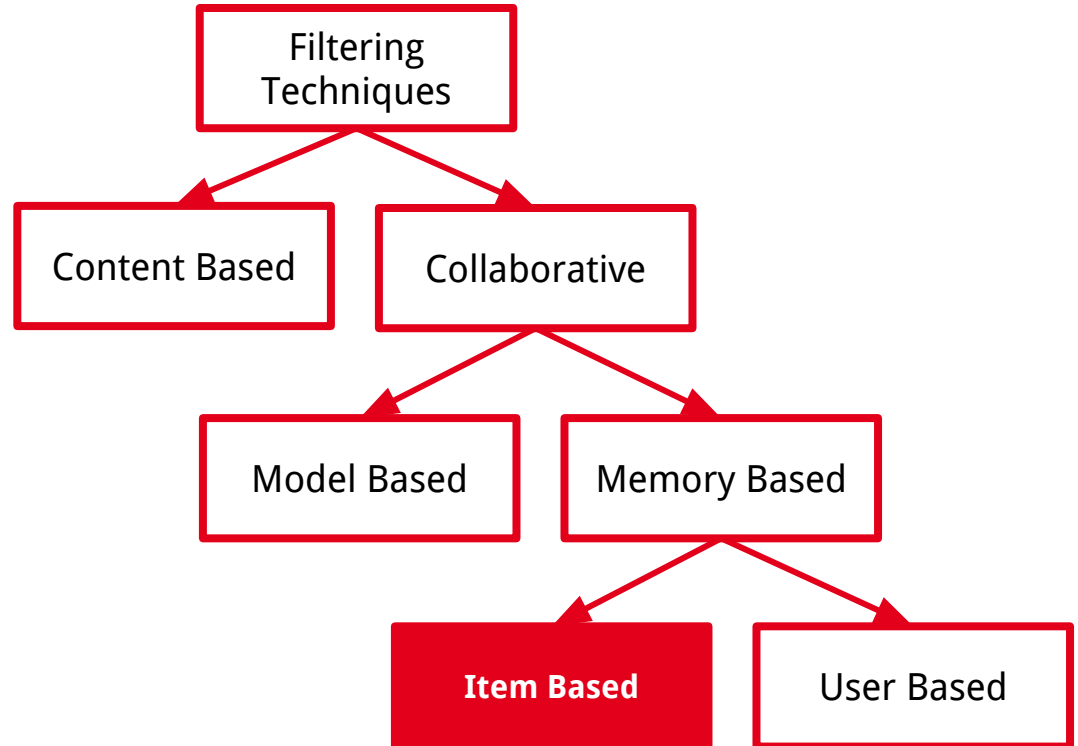
- Subset of collaborative filtering
- Operates directly on saved interaction history



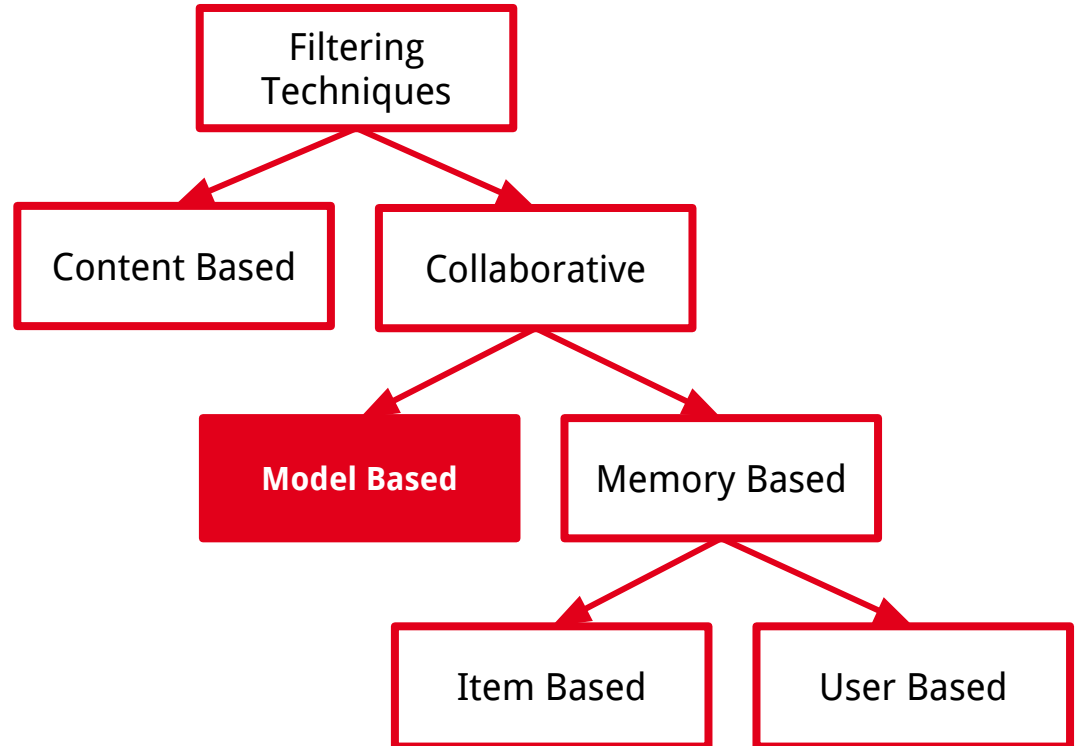
- Subset of memory based filtering
- Save interaction history per user
- Find groups of users that have similar interaction histories



- Subset of memory based filtering
- Save interaction history per item
- Find items that are similar to each other

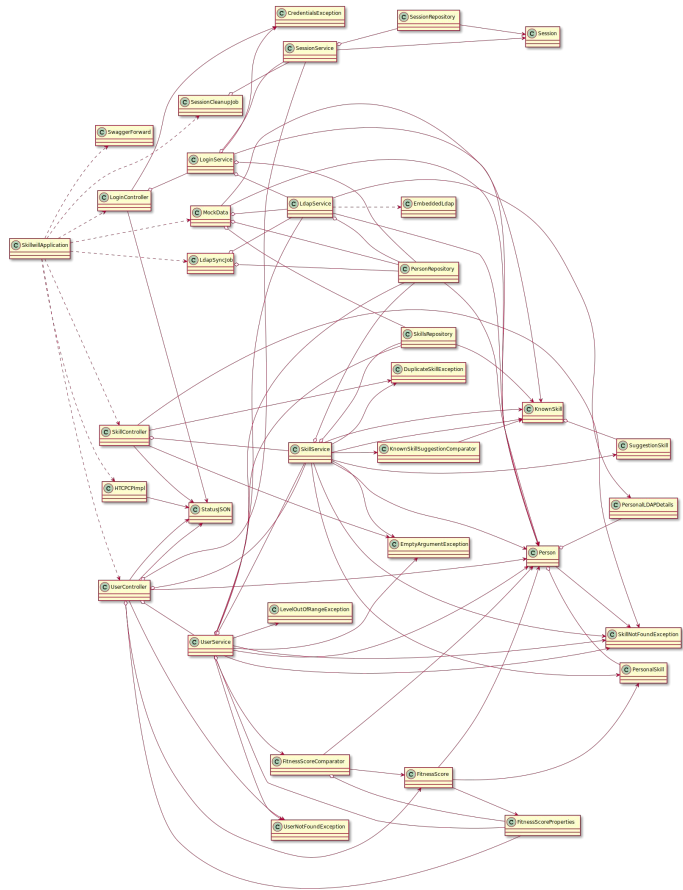


- Subset of collaborative filtering
- Model used to create suggestions
- Interactions to learn the model



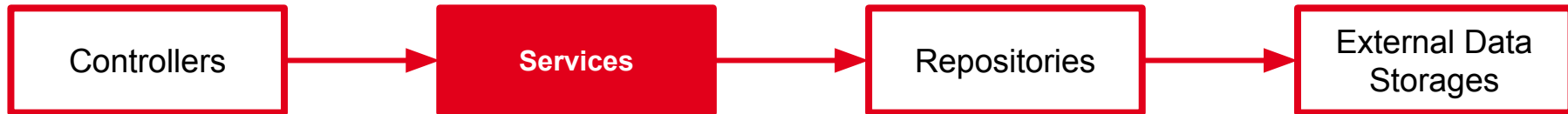
Class Structure

- Controllers
- Services
- Repositories
- Jobs
- Helpers
- Data Types





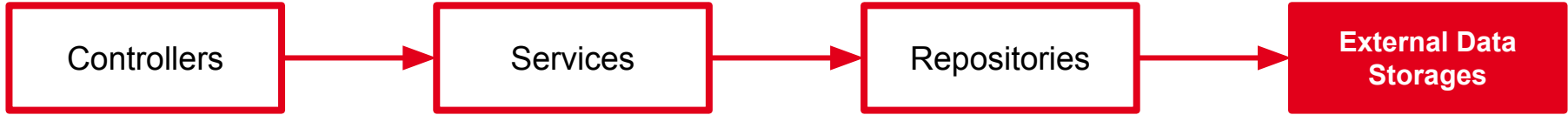
- Listen to API endpoints
- Use services to
 - Get data
 - Send data
 - Send commands
- Convert Objects to JSON



- Get, filter, transform, edit data
- Use repository objects to
 - Retrieve data from external sources
 - Write data to external sources
- Contain business logic



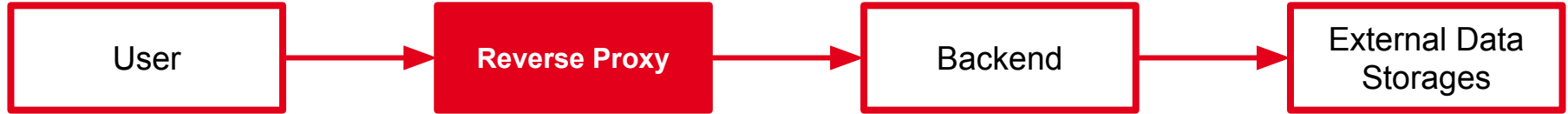
- Spring Data repository objects
- Provide Methods for CRUD operations
- Wrappers to simplify storage access



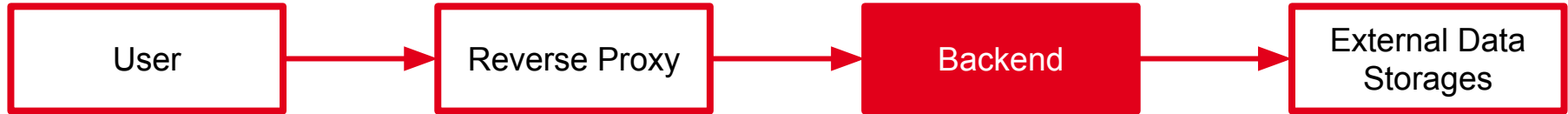
- MongoDB
- Skills
- Persons
- Sessions



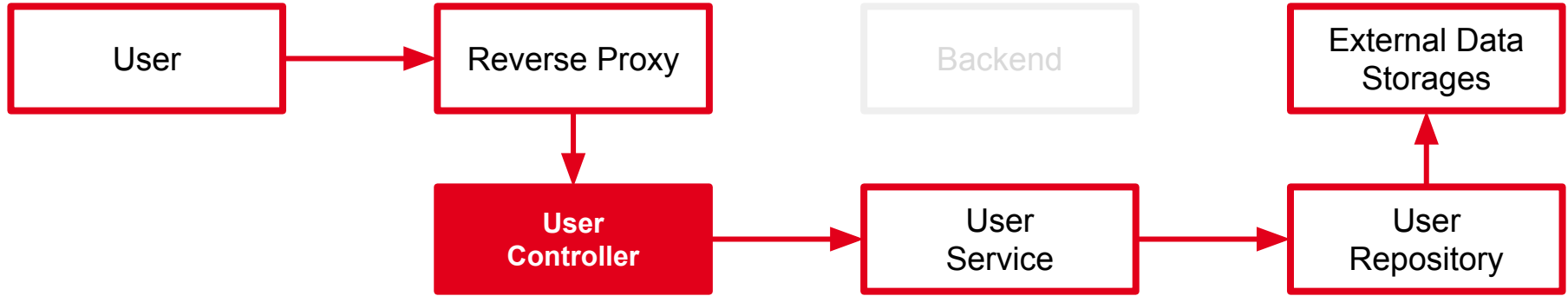
- Search for Java and Ruby in HH
- Browser Calls API
 - *api.some.tld/users?skills=Java,Ruby&location=Hamburg*



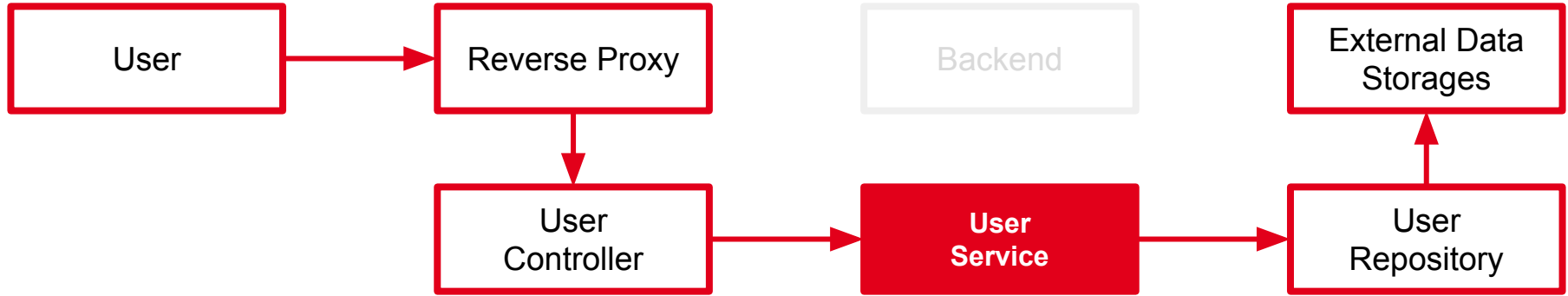
- Recognize API Call (Domain)
- Forward to Backend Server



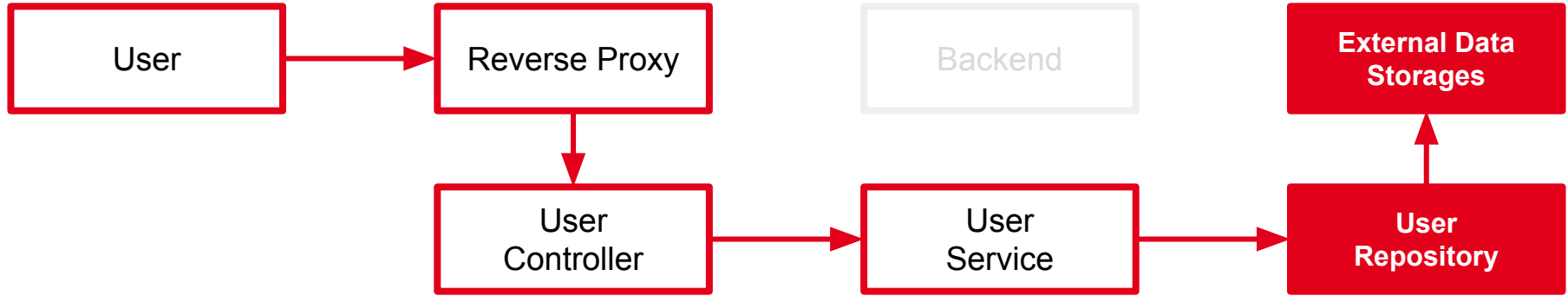
- Waits for HTTP Requests
- Dispatching to Controller



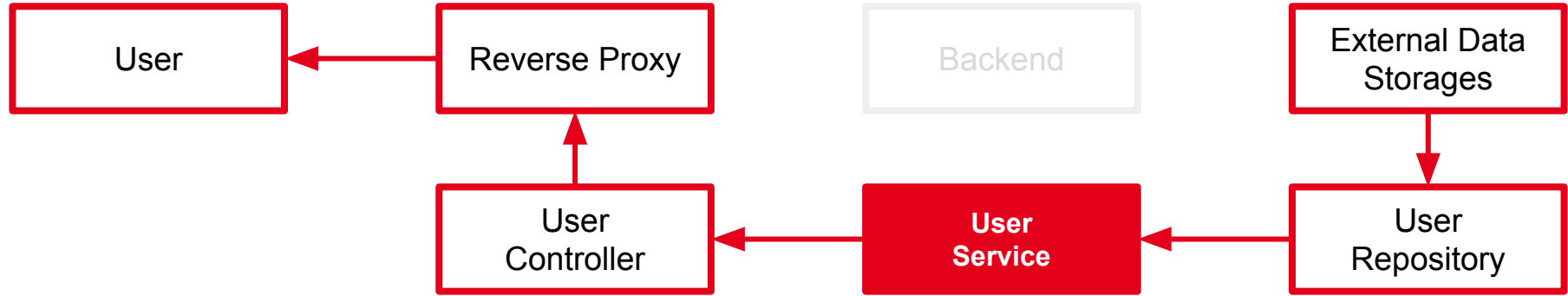
- Call method based on request URL and Parameters
 - /users/
 - ?skills=Java,Ruby&location=Hamburg
- Request matching persons from UserService



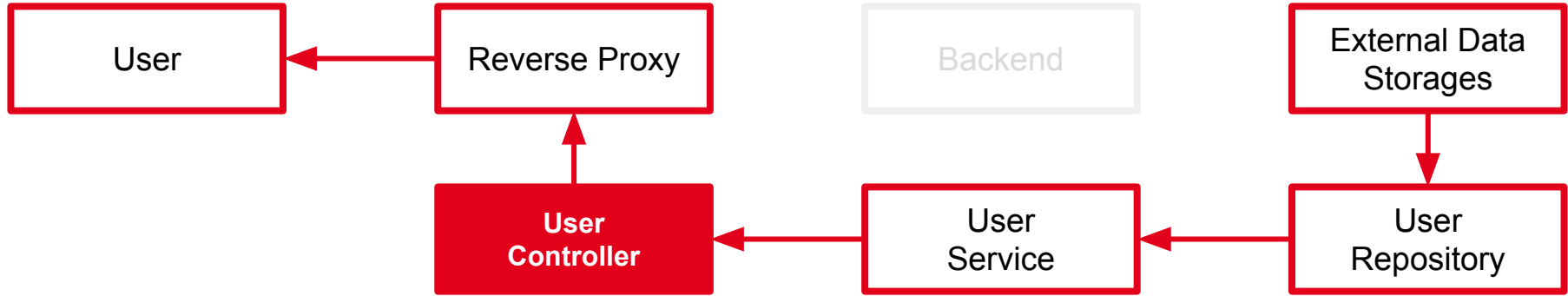
- Request needed users from Repository



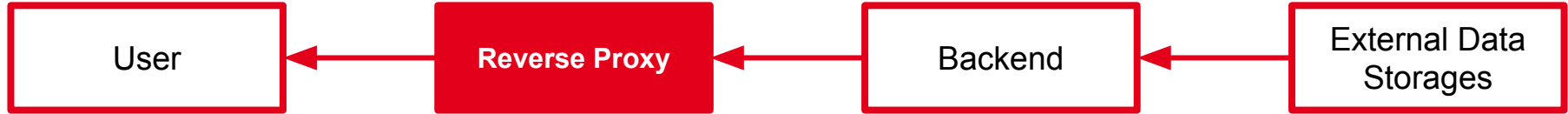
- Get data from MongoDB
- Return user objects to UserService



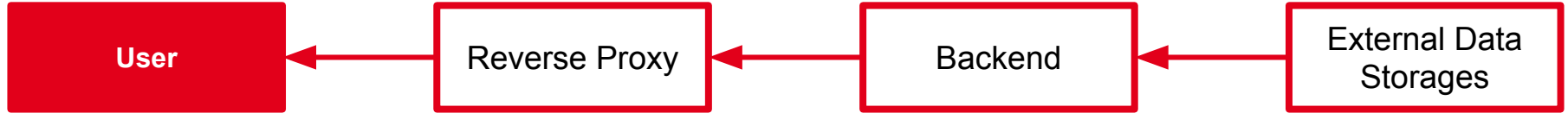
- Process retrieved user objects
- Apply search and fitness score algorithms



- Convert found user objects to JSON
- Return HTTP Response
 - In case of error, return corresponding HTTP Code



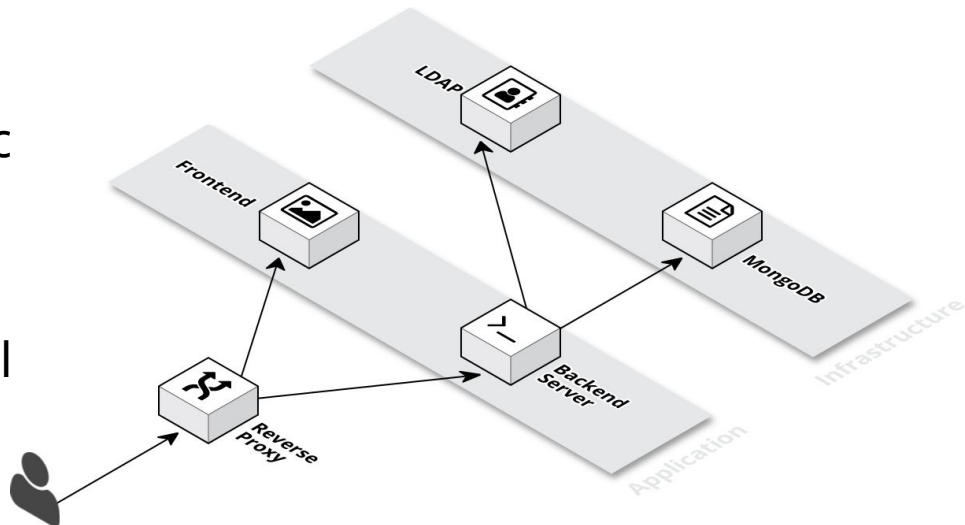
- Forward JSON response to client



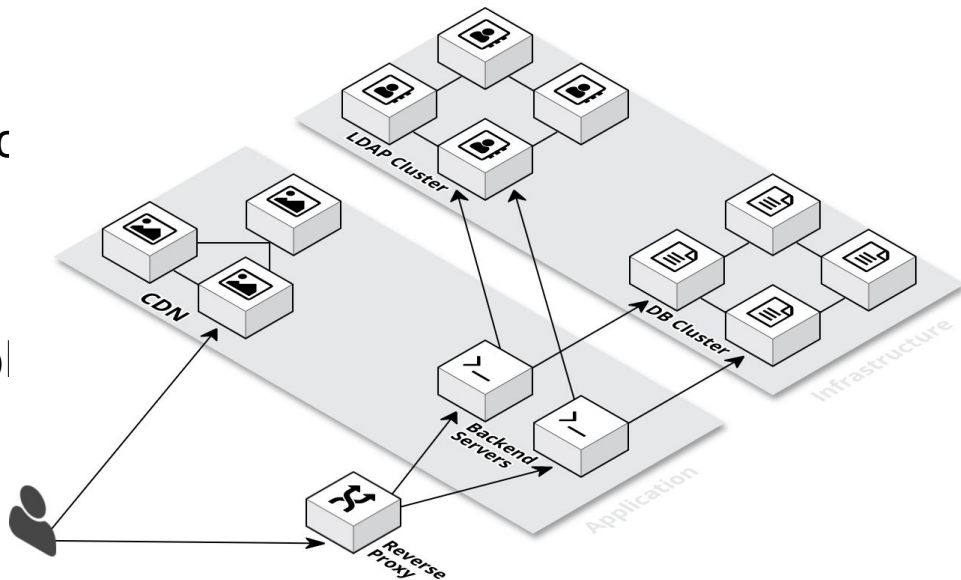
- Parse JSON response
- Render result list

Evaluation: Scalability

- MongoDB
 - Designed and shown to be sc
- LDAP
 - Six servers
 - Cluster is transparent to appl
- Frontend
 - CDN
- Backend
 - Stateless Application
 - Reverse proxy as load balancer
 - Tested



- MongoDB
 - Designed and shown to be scalable
- LDAP
 - Six servers
 - Cluster is transparent to application
- Frontend
 - CDN
- Backend
 - Stateless Application
 - Reverse proxy as load balancer
 - Tested



SINNERSCHRADER



Accenture Interactive

Part of Accenture Digital

**SINNERSCHRADER AG UND ACCENTURE
VEREINBAREN ZUSAMMENSCHLUSS; ACCENTURE
KÜNDIGT FREIWILLIGES ÖFFENTLICHES
ÜBERNAHMEANGEBOT FÜR SÄMTLICHE AKTIEN
DER SINNERSCHRADER AG AN**

AD-HOC NEWS

20. Februar 2017, 7:21

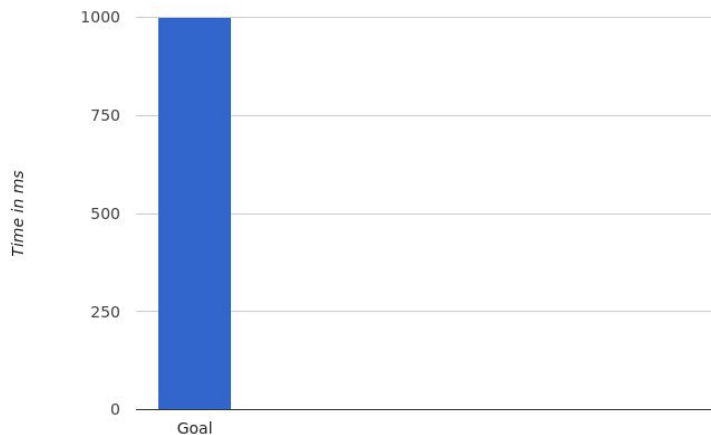
Die SinnerSchrader Aktiengesellschaft („SinnerSchrader“) hat heute nach entsprechenden Beschlüssen von Aufsichtsrat und Vorstand mit der Accenture Digital Holdings GmbH, einer 100-prozentigen... [weiter lesen](#)

Alle Finanznachrichten

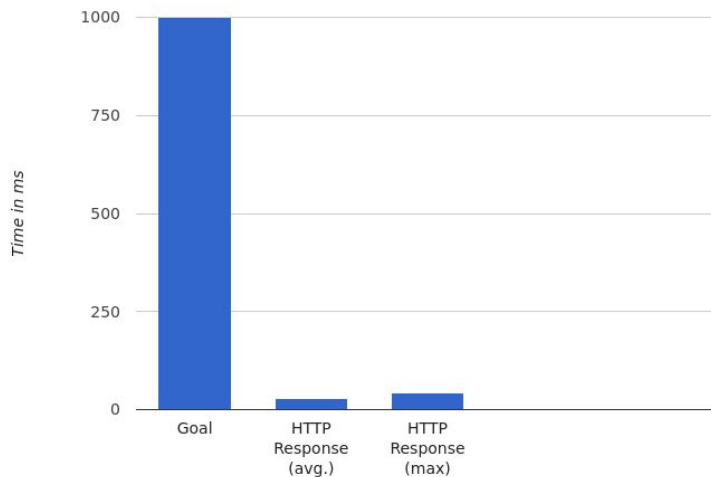


Evaluation: Response Times

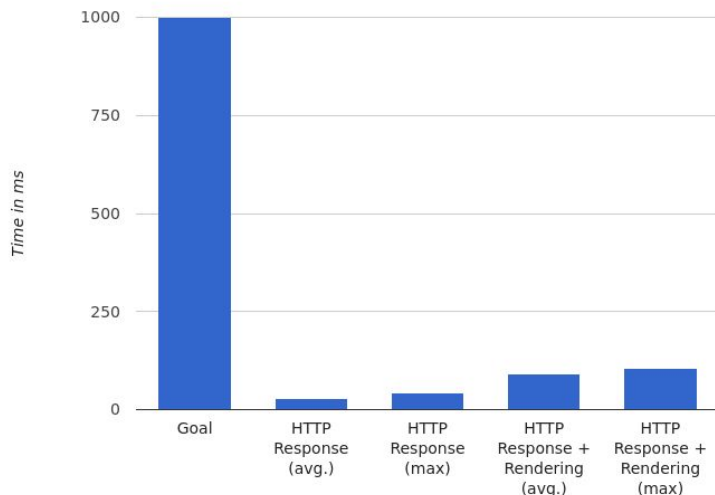
- Goal: 1s max to finish rendering



- Goal: 1s max to finish rendering
- HTTP Response: 33ms/44ms (40 samples)

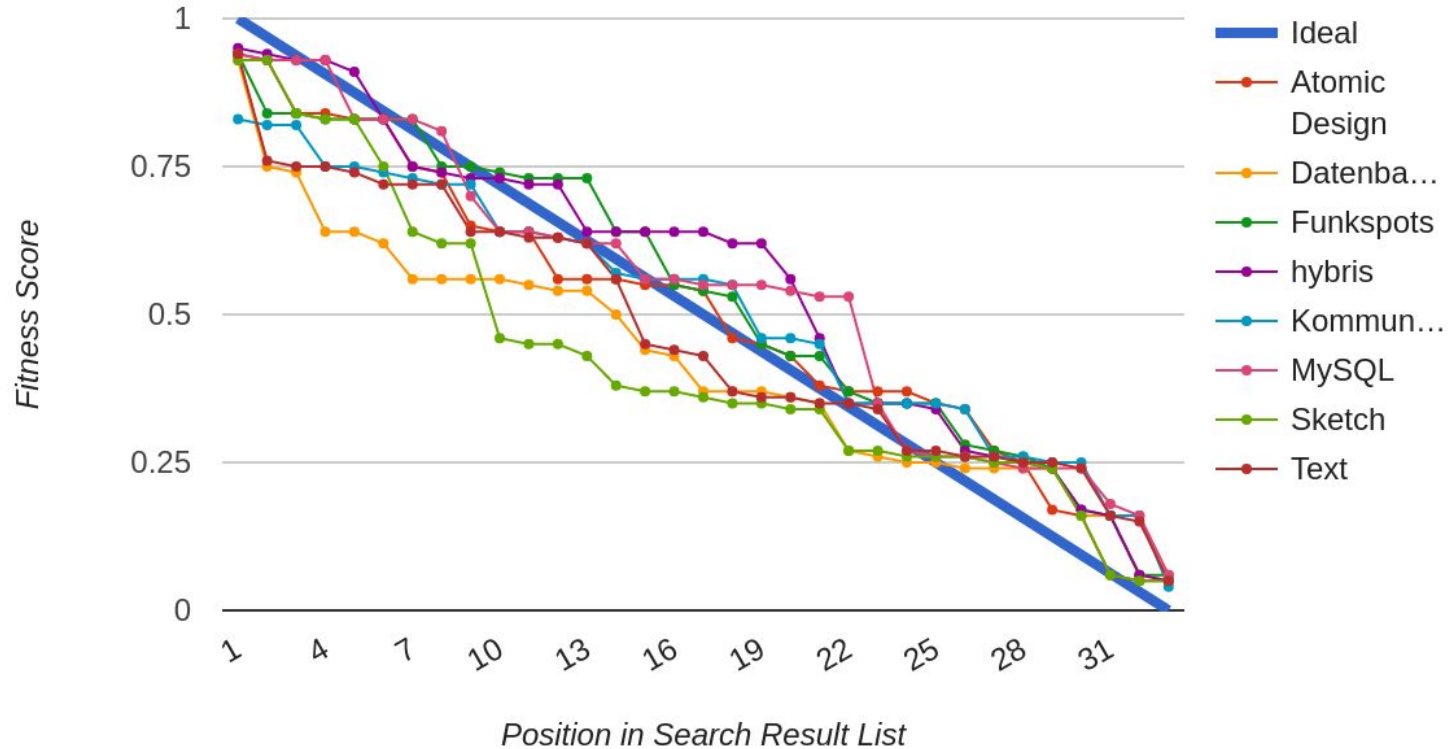


- Goal: 1s max to finish rendering
- HTTP Response: 33ms/44ms (40 samples)
- + Rendering: 90ms/106ms (16 samples)

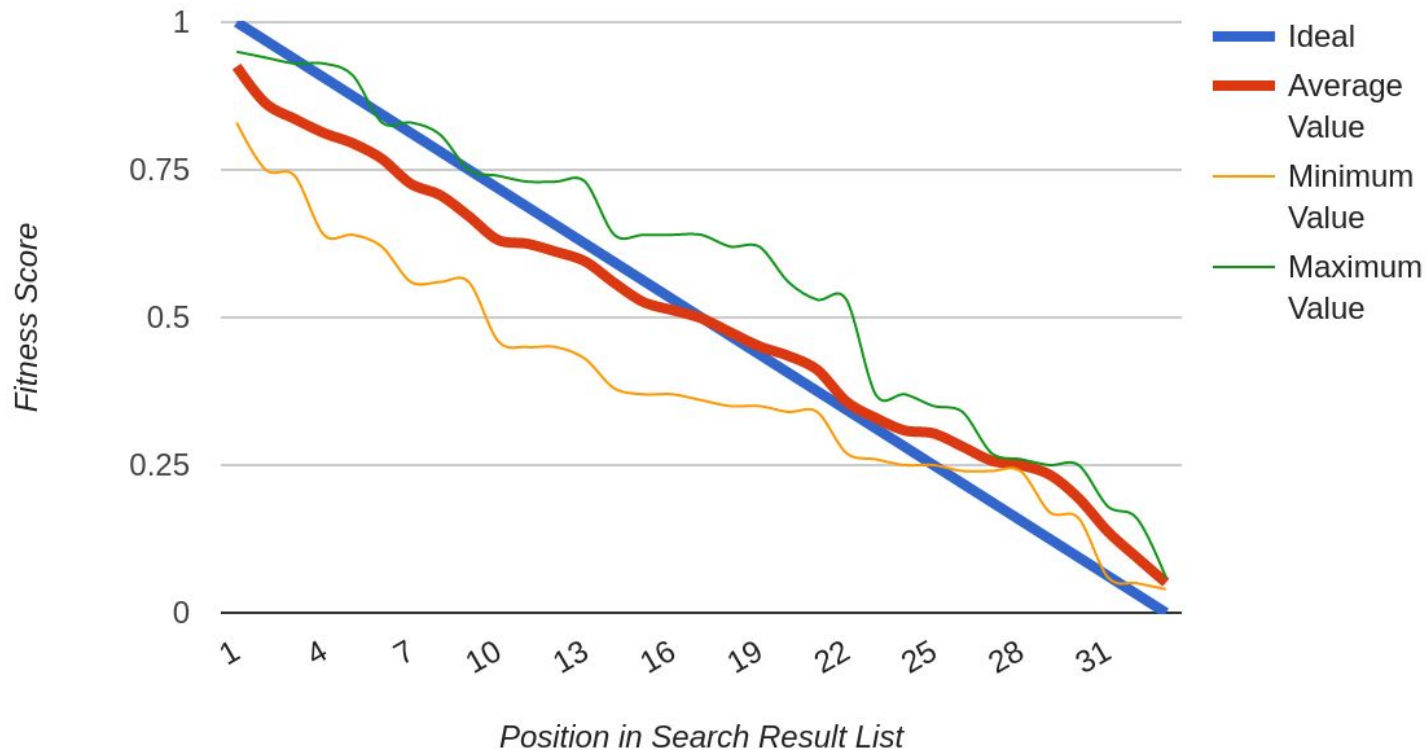


Evaluation: Uniform Distribution of Fitness Scores

- 100 users
- ≤ 17 skills per users
- Random levels
- Fitness of employees distributed uniformly
- Are fitness scores distributed uniformly?



Evaluation (Distribution of Fitness Scores)



- Average Deviation: 6%
- Maximum Deviation: 27%

