



## **ARlebnispfade Oberberg**

Synchronisierung  
mehrerer Clients

# Inhaltsverzeichnis

- Feedback aus dem 3. Audit
- Architekturdiagramme
- Tunneling Lösungsansatz
- Custom Lösungsansatz
- Development Dokumentation
- Vergleich & Bewertung
- Zukunftsideen
- Weitere Lösungsansätze
- Exkurs: Leitfragen
- Prozessassessment & Fazit
- Arbeitsmatrix
- Poster

Entwicklungsprojekt  
Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

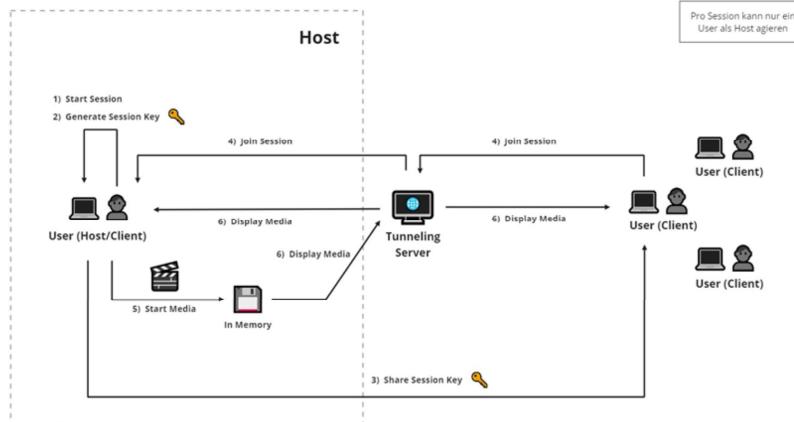


## Feedback aus dem 3. Audit

- Überlegung, ob beide Lösungsansätze (Cloud und Custom) weiter verfolgt werden sollen oder sich auf einen festgelegt werden soll
- Nach zwei Tagen wurde die Weiterentwicklung der Cloud Lösung beendet
- Stattdessen wurde ein neuer Lösungsansatz mit Tunneling entwickelt
- Zu weiteren Ansätzen wie bspw. HotSpot wurde recherchiert

Entwicklungsprojekt  
Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

# Tunneling: Architekturdiagramm & Prototyp

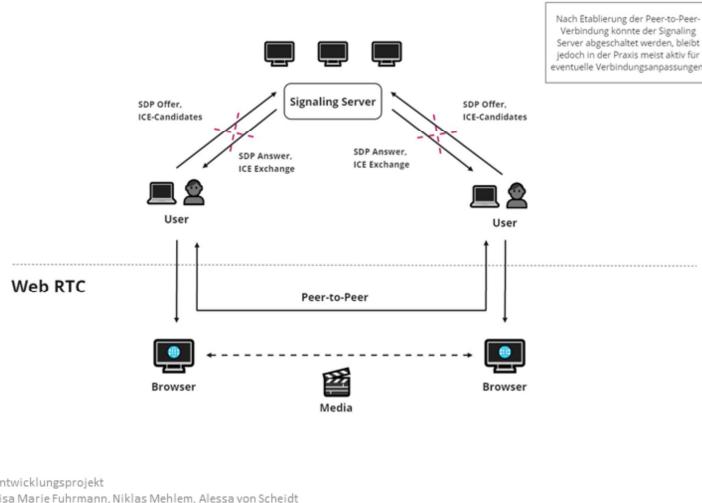


Entwicklungsprojekt  
Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

Bei der Tunneling Lösung agiert ein User als Host und startet lokal eine neue Session, für die ein Session Key generiert wird. Dieser wird an die anderen User weitergegeben, damit diese über den Tunneling Server der Session beitreten können. Der Host sendet einen Befehl das Video zu starten in den Memory Cache, danach lesen alle Clients den Befehl aus dem Cache um das Video zu starten.

Hier findet ihr den Code: <https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/tree/main/Main-Prototype/Tunneling-Solution> und hier das Architekturdiagramm: [https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Architekturdiagramm\\_Tunneling\\_Lösung\\_v1.0.jpg](https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Architekturdiagramm_Tunneling_Lösung_v1.0.jpg)

# Custom: Architekturdiagramm & Prototyp



Unsere individuelle Lösung nutzt WebRTC zur Herstellung direkter Peer-to-Peer-Verbindungen und WebSockets für den Signaling-Prozess. Ein Signaling-Server orchestriert den initialen Austausch von Verbindungsdetails wie SDP-Angebote und ICE-Kandidaten, um die Verbindungen zwischen den Nutzern aufzubauen. Nach der Etablierung ermöglicht WebRTC einen direkten Mediendatenaustausch ohne Zwischenstation.

Hier findet ihr den Code: <https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/tree/main/Main-Prototype/Custom-Solution>  
Und das Architekturdiagramm: [https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Architekturdiagramm\\_Custom\\_Lösung\\_v2.0.jpg](https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Architekturdiagramm_Custom_Lösung_v2.0.jpg)

# Development Dokumentation

## Development Dokumentation

Entwicklungsprojekt Zeitplan

Prozessassessment und Fazit

Es soll eine synchronisierte Wiedergabe von Medien über verschiedene Endgeräte hinweg ermöglicht werden. Die Synchronisation erfolgt einfach, anonym und ad hoc, ohne auf eine zentrale Unit (Server) zugreifen zu müssen. Dabei wird auch das Konzept für die Erstellung und Verwaltung einer Gruppe erarbeitet.

### Oracle/Cloud Ansatz

#### Entwicklungsleitfaden

1. Oracle Projekt in AWS übertragen
2. Erfolgreich Verbindung herstellen
3. Synchronisation umsetzen (POC)
4. Mediübertragung / Wiedergabe
5. Gruppenbildung und Konfiguration
6. Benutzeroberfläche
7. Systemtests

#### Ressourcen

- <https://technology.ams.nl/frontend/serverless-distributed-multi-user-browser-session-synchronization/>
- <https://technology.ams.nl/cloud/implementing-serverless-multi-client-session-synchronization-with-oracle-cloud-infrastructure/>
- <https://www.youtube.com/watch?v=7u1p5dieh0>

#### Technologien und Tools

- AWS (Lambda, API Gateway, VPC, ElasticCache)

#### Entwicklungsprojekt

Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

### Tunneling Ansatz

#### Entwicklungsleitfaden

1. Cloud Project für Tunneling umschreiben ✓
2. Erfolgreich Verbindung herstellen ✓
3. Synchronisation umsetzen (POC) ✓
4. Gruppenbildung und Konfiguration ✓
5. Mediübertragung / Wiedergabe ✓
6. Benutzeroberfläche
7. Systemtests

#### Ressourcen

- <https://technology.ams.nl/frontend/serverless-distributed-multi-user-browser-session-synchronization/>
- <https://technology.ams.nl/cloud/implementing-serverless-multi-client-session-synchronization-with-oracle-cloud-infrastructure/>

#### Technologien und Tools

- HTML / CSS
- Node.js

### Custom Ansatz

#### Entwicklungsleitfaden

1. Signaling Server aufsetzen ✓
2. P2P-Verbindung in Räumen über Sockets ✓
3. Synchrone Übertragung eines Streams ✓
4. Synchrone Übertragung einer YouTube Videos ✓
5. Synchrone Übertragung einer lokalen Datei ✓
6. Datenbank für Medien ✗
7. P2P-Verbindung auf Mesh ausweiten ✓
8. optional: Signaling Server /Sockets umgehen ✓
9. Testing und Debugging ✓
10. Gruppenverwaltung ✗
11. Deployment ✗

#### Ressourcen

- <https://webrtc.org/hl/de>
- <https://firebase.google.com/docs?hl=de>
- <https://blog.printf.net/articles/2013/05/17/webrtc-without-a->

Hier findet ihr unsere Development Dokumentation: <https://little-cashew-cf8.notion.site/Development-Dokumentation-30b439ae92bc4f7cb817bca335e9f40c?pvs=4>

# Vergleich

Merkmal	WebRTC/Sockets	Tunneling
Fähigkeit zur Echtzeitübertragung	Beide Technologien ermöglichen die Übertragung von Daten in Echtzeit, wobei WebRTC sich auf Audio-, Video- und direkte Datenkommunikation konzentriert, während Tunneling eine breite Palette von Netzwerkdaten über sichere Verbindungen unterstützt.	
Grundlegende Verschlüsselungsfunktionen	Sowohl WebRTC als auch Tunneling bieten Mechanismen zur Verschlüsselung und Sicherung der Datenübertragung, um Vertraulichkeit und Integrität der übertragenen Informationen zu gewährleisten.	
Konfiguration und Einrichtung erforderlich	Beide Technologien erfordern eine anfängliche Konfiguration und Einrichtung, wie z.B. die Einrichtung von Signalisierung bei WebRTC oder die Konfiguration von VPN-/SSH-Tunneln, um die Kommunikation zu ermöglichen.	
Unterstützung über Plattformen hinweg	WebRTC und Tunneling werden von einer Vielzahl von Plattformen unterstützt. WebRTC ist in modernen Webbrowsern integriert, während Tunneling-Technologien auf den meisten Betriebssystemen nativ oder über Zusatzsoftware verfügbar sind.	
Anpassung an spezifische Anwendungsfälle	Beide Technologien lassen sich an spezifische Anwendungsfälle anpassen, sei es für direkte Echtzeitkommunikation und Medienaustausch bei WebRTC oder für sichereren Fernzugriff und Netzwerkerbindungen bei Tunneling.	
Anwendungsbereich	Echtzeitkommunikation zwischen Browsern ohne Plugins, ermöglicht interaktive Anwendungen	Ermöglicht eine sichere, direkte Verbindung für Datenübertragung, Fernzugriff und Netzwerkerweitungen, nicht speziell für Echtzeit-Medienkommunikation entworfen.
Verbindungsstruktur	Peer-to-Peer, wodurch Bandbreite gespart und Serverlast reduziert wird, fordert die Skalierbarkeit und Effizienz durch direkten Austausch von Medien und Daten	Establiert eine verschlüsselte "Tunnel"-Verbindung zwischen zwei Punkten, was oft einen oder mehrere Server zur Datenweiterleitung nutzt.

Entwicklungsprojekt  
Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

In dieser Tabelle haben wir unsere Erkenntnisse zusammengetragen, die wir innerhalb des Projektes gesammelt haben und die unser Fazit beeinflussen.

Hier findet ihr unsere Vergleichstabelle:

[https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Vergleichstabelle\\_v1.0-1.jpg](https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Vergleichstabelle_v1.0-1.jpg)

# Vergleich

<b>NAT Traversal</b>	Nutzt ICE, das STUN und TURN einschließt, um die öffentliche IP-Adresse und Ports zu ermitteln und direkte Verbindungen zu ermöglichen oder Verkehr über TURN-Server zu leiten, wenn nötig	Oft nicht erforderlich oder durch die Einrichtung differenzierter Zugangspunkte (VPN-Endpunkte) gelöst, die nicht hinter NAT sind oder spezielle Konfigurationen nutzen
<b>Erweiterte Sicherheitsmerkmale</b>	Bietet Ende-zu-Ende-Verschlüsselung für alle Daten. Sicherheitsmerkmale sind integraler Bestandteil der Technologie	Sicherheit wird durch Verschlüsselungsprotokolle innerhalb des Tunnels gewährleistet, wie z.B. diejenigen, die in VPNs verwendet werden, die Sicherheit hängt stark von der Konfiguration und dem verwendeten Protokoll ab
<b>Implementierungskomplexität</b>	Erfordert die Implementierung von Signallierungsmechanismen für den Verbindungsaufbau und die Handhabung von NAT Traversal. Plattformübergreifende Kompatibilität und die Behandlung verschiedener Mediencodierungen können herausfordernd sein	In der Regel einfacher mit bestehenden Protokollen und Tools (z.B. SSH, VPN). Die Einrichtung von Tunneln ist oft weniger komplex als die Einrichtung einer WebRTC-Infrastruktur
<b>Anpassung an Netzwerkbedingungen</b>	Passt dynamisch z.B. Videoqualität und Bitrate an, basierend auf aktuellen Netzwerkbedingungen, um eine optimale Performance zu gewährleisten	Bietet in der Regel keine dynamische Anpassung der Übertragungsqualität an die Netzwerkbedingungen. Die Leistung kann durch die Bandbreite des schmalsten Pfades im Netzwerk begrenzt sein
<b>Abhängigkeit von zentralen Servern</b>	Minimiert durch direkte Peer-to-Peer-Verbindungen. Signalisierungs- und TURN-Server sind notwendig, aber nur für den Verbindungsaufbau bzw. als Fallback	Erfordert oft zentrale Server für die Einrichtung des Tunnels oder als Teil der Tunnelinfrastruktur, was zu zentralen Engpassen führen kann
<b>Skalierbarkeit und Effizienz</b>	Hoch, dank direkter Peer-to-Peer-Verbindungen und effizienter Nutzung der Bandbreite, geeignet für skalierbare Anwendungen (mit vielen Teilnehmern?)	Kann durch die Notwendigkeit zentraler Server und die begrenzte Bandbreite dieser Server eingeschränkt sein. Effizienz abhängig von der Netzwerktopologie und -konfiguration
<b>Plattform- und Geräteunterstützung</b>	Breite Unterstützung in modernen Webbrowsers; erfordert keine zusätzlichen Plugins oder Softwareinstallations auf Benutzergeräten	Kann zusätzliche Softwareinstallationen erfordern (z.B. VPN-Client) und ist möglicherweise nicht auf allen Geräten oder in allen Netzwerken kompatibel

Entwicklungsprojekt  
Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

Hier findet ihr die Tabelle: [https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Vergleichstabelle\\_v1.0-2.jpg](https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Vergleichstabelle_v1.0-2.jpg)

# Bewertungsmatrix – Entwickler & User

Kriterium	Aspekt	Multiplikator	Tunneling Lösung	Bewertung	Custom Lösung	Bewertung
Leistung	Geschwindigkeitsvorteile für typische Aufgaben		Keine spürbaren Verzögerungen	4	Direkter Datenaustausch minimiert Verzögerungen	5
	Ladezeiten für Inhalte		Keine spürbaren Verzögerungen	4	Peer-to-peer-Übertragungen bestimmen das Laden	5
	Hohe von Latenzzeiten		Keine spürbaren Verzögerungen	4	Konzipiert für minimale Latenz	5
	Stabilität der Anwendung		Abhängig vom Tunneling Server	3	Stabilität kann unter extremen Netzwerkverhältnissen variieren	4
Zwischenergebnis	Zusammenfassende Bewertung für dieses Kriterium	2	Zu viele Anfragen können (je nach Anzahl) die Leistung gepeinigt den Tunnel überlasten	2	ermöglicht nahezu sofortige Reaktionen	5
Leistung unter Stress	Performance mit steigender Nutzerzahl	2	Mehr bei höherer Last Verzögerungen führen	3.4	schnelle und effiziente Endkommunikation mit verbesserten Konsolidierungsprozessen	4.8
	Performance mit steigender Datenmenge	2	Tunneling Server muss für die geforderte Last ausgerichtet sein, sonst wird er zum Bottleneck	2	Bei zu hoher Last Traversing können bei sehr hohen Nutzerzahlen Probleme auftreten	3
Zwischenergebnis	Zusammenfassende Bewertung für dieses Kriterium	1.8	Kann unter hoher Last zu merklichen Verzögerungen führen	2	Verbesserte Leistung und Stabilität unter Last durch effiziente Zuverlässigkeit und Fehlerwiederherstellung abhängig von der	3
					Tunneling-Mechanismus	
Anpassbarkeit	Möglichkeit der Anpassung				Bietet von sich aus nur so viel Möglichkeiten zur Anpassung wie nötig, aber gleichzeitig kaum ein	3
	Höhe des Anpassungsaufwandes				Ahnlich wie bei der gewählten Tunneling Technologie	3
	Einschränkungen oder Probleme				Ein großer Vorteil entspringt ganz den Kapazitäten des Tunneling Servers	3
Zwischenergebnis	Zusammenfassende Bewertung für dieses Kriterium	1.4			Schräkt wenig ein, bietet aber zahlreiche Möglichkeiten außerhalb von gewöhnlichen Server-Konfigurationen	3
Nutzerinteraktion	Aufwand/ Dauer zum Durchführen einer Aktion				hoher Verzögerung aufgrund der Cache Lösung	4
Zwischenergebnis					nutzt hohe soziale Netzwerke	5
Gesamtbewertung	Mögliche Gesamtpunktzahl = 75				Untersetzt hochgradig interaktive Anwendungen	5
					und unterstützt sie nicht	6
						76 %

Die ehemals getrennten Bewertungsmatrizen aus der Perspektive des Entwicklers und des Nutzers wurden von uns in einer Matrix zusammen gefasst und einige Kriterien ausgelassen. Die beiden ursprünglichen Matrizen waren für fertig entwickelte Systeme konzipiert worden, sodass auch Kriterien zu prüfen waren, die zwar wichtig sind, die von uns zum jetzigen Stand aber noch nicht geprüft werden können, da die beiden Technologien nicht so weit entwickelt sind. Als Beispiel ist hier das ausgelassene Kriterium für die Benutzeroberfläche und das Design zu nennen. Daher wurden die beiden Matrizen von uns so zurecht geschnitten/gekürzt, dass mit dem aktuellen Stand der Technologien alle Kriterien geprüft werden können. Da die somit gewählten Kriterien bis auf die Nutzerinteraktion gleich waren wurden sie direkt in einer Matrix zusammengefasst.

Die Bewertung erfolgt folgendermaßen:

- Je Unterkategorie kann eine Anmerkung und Punkte von 1-5 vergeben werden (5 = höchste Punktzahl; 1 = niedrigste Punktzahl)
  - Im Zwischenergebnis wird aus den Unterkategorien eine Durchschnittsnote errechnet, die später beim Zusammenzählen mit dem Multiplikator des Kriteriums multipliziert wird
  - die Multiplikatoren reichen von 0-3 (0 = am niedrigsten; 2 am höchsten)
  - die Punkte der Zwischenergebnisse werden mit ihren Multiplikatoren multipliziert

und dann zusammen gerechnet; es sind maximal 75 Punkte zu erreichen  
- neben der Angabe der Punkte werden die errechnete Gesamtpunktzahl auch als Prozentsatz angegeben

Unsere vollständige Matrix findet ihr hier:

<https://github.com/lisafuhrmann/EPWS2324->

[FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix\\_PerspektiveEntwickler %26 User v1.0-1.jpg](FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix_PerspektiveEntwickler %26 User v1.0-1.jpg)

<https://github.com/lisafuhrmann/EPWS2324->

[FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix\\_PerspektiveEntwickler %26 User v1.0-2.jpg](FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix_PerspektiveEntwickler %26 User v1.0-2.jpg)

<https://github.com/lisafuhrmann/EPWS2324->

[FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix\\_PerspektiveEntwickler %26 User v1.0-3.jpg](FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix_PerspektiveEntwickler %26 User v1.0-3.jpg)

<https://github.com/lisafuhrmann/EPWS2324->

[FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix\\_PerspektiveEntwickler %26 User v1.0-4.jpg](FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix_PerspektiveEntwickler %26 User v1.0-4.jpg)

<https://github.com/lisafuhrmann/EPWS2324->

[FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix\\_PerspektiveEntwickler %26 User v1.0-5.jpg](FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix_PerspektiveEntwickler %26 User v1.0-5.jpg)

## Bewertungsmatrix – Anforderungen

Anforderung	Art		Multiplikator	Kriterien	Tunneling Lösung	Bewertung	Custom Lösung	Bewertung																													
	Funktional	muss																																			
Das System muss fähig sein QR-Codes zu scannen				Genaugheit des Scans																																	
				Geschwindigkeit des Scans																																	
				Kompatibilität auf verschiedenen Endgeräten		nicht implementiert																															
				Umgang mit beschädigten QR-Codes																																	
Zwischenergebnis			2	Zusammenfassende Bewertung für dieses Kriterium	Nicht implementiert	0		0																													
Das System muss fähig sein Videos und Audios abspielen zu können				Unterstützte Dateiformate				3																													
				Qualität der Wiedergabe				4																													
				Steuerung der Wiedergabe	Auf Endgeräten muss das Video pausiert und wieder gestartet werden, wenn die neue Position an die anderen Clients gesendet werden soll.	2	Ist in der Lage, jedoch aktuell nur Video implementiert	3																													
				Ladezeiten				5																													
Zwischenergebnis			2	Zusammenfassende Bewertung für dieses Kriterium	Auf Mobilen: Endgeräten müssen die Medien heruntergeladen werden. Da kein Headphones es vor dem Audio-Datenstrom nicht am korrekten Zeitpunkt abgespielt würden.	2		3																													
<table border="1"> <tr> <td>Zwischenergebnis</td> <td></td> <td></td> <td></td> <td>1</td> <td>Zusammenfassende Bewertung für dieses Kriterium</td> <td>nicht implementiert</td> <td>0</td> <td>0</td> </tr> <tr> <td>Gesamtbewertung</td> <td></td> <td></td> <td></td> <td></td> <td>Gesamtpunktzahl durch Anzahl Kriterien</td> <td>157,5</td> <td>54,6%</td> <td>157,5</td> <td>39,7%</td> </tr> <tr> <td>Gesamtbewertung - möglich</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>86/125</td> <td>68,8%</td> <td>62/85</td> <td>72,9%</td> </tr> </table>									Zwischenergebnis				1	Zusammenfassende Bewertung für dieses Kriterium	nicht implementiert	0	0	Gesamtbewertung					Gesamtpunktzahl durch Anzahl Kriterien	157,5	54,6%	157,5	39,7%	Gesamtbewertung - möglich						86/125	68,8%	62/85	72,9%
Zwischenergebnis				1	Zusammenfassende Bewertung für dieses Kriterium	nicht implementiert	0	0																													
Gesamtbewertung					Gesamtpunktzahl durch Anzahl Kriterien	157,5	54,6%	157,5	39,7%																												
Gesamtbewertung - möglich						86/125	68,8%	62/85	72,9%																												

Entwicklungsprojekt  
Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

Hier ist ein Ausschnitt der Bewertungsmatrix über die Erfüllung der Anforderungen abgebildet. Die Bewertung dieser Bewertungsmatrix erfolgt wie für die Bewertungsmatrix zur Perspektive Entwickler und Nutzer.

Ein Unterschied besteht in der Angabe des Gesamtergebnisses. Da nicht alle Anforderungen erfüllt werden konnten wurden diese in der Bewertung mit 0 Punkten gewertet.

Bei einer komplett ausgefüllten Matrix können bis zu 157,5 Punkte erreicht werden. Je nach Lösungsansatz wurde errechnet wie viele Punkte für alle bewerteten Kriterien erreicht werden können und wie viele davon tatsächlich erreicht wurden. Daraus ergibt sich der untere Prozentsatz.

Mit dem oberen Prozentsatz wird angezeigt, wie viel von der gesamten Matrix damit erreicht wird.

Hier findet ihr die vollständige Matrix:

<https://github.com/lisafuhrmann/EPWS2324->

[FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix\\_Erfüllung\\_der\\_Anforderungen\\_v2.0-1.jpg](FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix_Erfüllung_der_Anforderungen_v2.0-1.jpg)

<https://github.com/lisafuhrmann/EPWS2324->

[FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix\\_Erfüllung\\_der\\_Anforderungen\\_v2.0-2.jpg](FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix_Erfüllung_der_Anforderungen_v2.0-2.jpg)

[https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix\\_Erfüllung\\_der\\_Anforderungen\\_v2.0-3.jpg](https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix_Erfüllung_der_Anforderungen_v2.0-3.jpg)

[https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix\\_Erfüllung\\_der\\_Anforderungen\\_v2.0-4.jpg](https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix_Erfüllung_der_Anforderungen_v2.0-4.jpg)

[https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix\\_Erfüllung\\_der\\_Anforderungen\\_v2.0-5.jpg](https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix_Erfüllung_der_Anforderungen_v2.0-5.jpg)

[https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix\\_Erfüllung\\_der\\_Anforderungen\\_v2.0-6.jpg](https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix_Erfüllung_der_Anforderungen_v2.0-6.jpg)

[https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix\\_Erfüllung\\_der\\_Anforderungen\\_v2.0-7.jpg](https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Bewertungsmatrix_Erfüllung_der_Anforderungen_v2.0-7.jpg)

## ZukunftsIdeen

- Weiterentwicklung des Systems um auch das Darstellen von AR-Inhalten zu ermöglichen
- Ausbau des Frontends, Gestaltung der Benutzeroberfläche und entwickeln eines benutzerfreundlichen Designs und Designmuster
- Implementierung von einer passenderen Technologie für das Verwalten von Gruppen für die Custom Lösung
- Vollständiges Einbinden von QR-Codes

Entwicklungsprojekt  
Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

## Weitere Lösungsansätze

Merkmal	WebTorrent	HotSpot	Cloud
Fähigkeit zur Echtzeitübertragung	Nicht Ausgelegt für schnelle Echtzeit Synchronisierung und es ist damit wahrscheinlich mit wahrnehmbarer Latenz zu rechnen.	Nicht Ausgelegt für schnelle Echtzeit Synchronisierung und es ist damit wahrscheinlich mit wahrnehmbarer Latenz zu rechnen.	Nicht Ausgelegt für schnelle Echtzeit Synchronisierung und es ist damit wahrscheinlich mit wahrnehmbarer Latenz zu rechnen.
Grundlegende Verschlüsselungsfunktionen	Hat keine Verschlüsselung von sich aus, kann aber mit Protokollen wie TLS verschlüsselt werden, was aber auch etwas mehr Rechenleistung benötigt.	Hat keine Verschlüsselung von sich aus	Je nach Cloud Anbieter ist es mögliches Services einzubinden, ansonsten gibt es keine native Verschlüsselung
Konfiguration und Einrichtung erforderlich	Minimale Konfiguration erforderlich, mehr je nach speziellem Anwendungsfall.	Keine Einrichtung erforderlich	Je nach Cloud unterschiedlich. Im Falle von AWS durfte viel eingelesen und eingerichtet werden.
Unterstützung über Plattformen hinweg	WebTorrent ist eine JavaScript Bibliothek die in allen Browsern mit JavaScript Support läuft.	(Abhängig von der final verwendeten Technologie und Programmiersprache)	Die Cloud Technologie hat keinen Einfluss auf die Ausführung des Codes auf verschiedenen Plattformen
Anpassung an spezifische Anwendungsfälle	Torrents können verschiedene Anwendungsfälle wie streaming, file sharing, oder data synchronization angepasst werden.	Bietet nicht viel Anpassungsmöglichkeiten abseits der verwendeten Technologie für die Kontaktaufnahme	Je nach Cloud Anbieter kann es viele Service geben die einen spezielle Anpassungen ermöglichen.
Anwendungsbereich	Senden und empfangen von Dateien. WebTorrent hat Vorteile die es ermöglichen Videos während des Downloads bereits zu gucken.	Ursprünglich einfach die das Teilen einer Internetverbindung konzipiert.	Anbieten von Rechenleistung gegen Entgelt, so dass keine statischen Server gekauft oder gemietet werden müssen
Verbindungsstruktur	Peer-to-Peer, wodurch Bandbreite gespart und Serverlast reduziert wird, fordert die Skalierbarkeit und Effizienz durch direkten Austausch von Dateien.	Verbindung von Geräten über ein lokales Netzwerk	Geräte sind nicht miteinander verbunden und teilen sich lediglich den gleichen Cache in der Cloud.
NAT Traversal	Nutzt häufig Technologien wie UDP, NAT-PMP oder UPnP um eine Verbindung trotz der NAT Barriere herzustellen.	Je nach Technologie regelt diese bereits das NAT Traversal, da im selben lokalen Netzwerk gearbeitet wird.	Nicht Erforderlich, je nach Cloud Struktur ist es aber notwendig ein NAT in einer Private Cloud einzurichten.

Entwicklungsprojekt  
Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

Nach weiteren möglichen Lösungsansätzen haben wir recherchiert.

Hier findet ihr die Tabelle: [https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Weitere\\_Lösungsansätze\\_v1.0-1.jpg](https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Weitere_Lösungsansätze_v1.0-1.jpg)

## Weitere Lösungsansätze

Erweiterte Sicherheitsmerkmale	Abhängig von den Implementierten Methoden.	Abhängig von den Implementierten Methoden.	Abhängig von der gewählten Cloud und Services.
Implementierungskomplexität	-	-	Hoch da sich in die verschiedenen Funktionen und Services der Cloud eingelezen werden muss, so wie in Ihre generelle Bedienung.
Anpassung an Netzwerkbedingungen	Passt automatisch die Anzahl der Verbindungen und Bandbreite an um die beste Leistung unter den gegebenen Bedingungen zu erzielen.	Passt sich selbst nicht an.	Abhängig vom gewählten Cloud Anbieter und Services. Allerdings sind Clouds meistens darauf ausgelegt sich dynamisch den Anforderungen an ihre Leistung anzupassen.
Abhängigkeit von zentralen Servern	Keine, da WebTorrent keinen Torrent Client benötigt.	Keine da die Clients direkt über das lokale Netzwerk miteinander kommunizieren.	Abhängigkeit von der gewählten Cloud.
Skalierbarkeit und Effizienz	Hoch, dank direkter Peer-to-Peer-Verbindungen und effizienter Nutzung der Bandbreite, geeignet für skalierbare Anwendungen.	-	Hoch da die meisten Clouds auf Skalierbarkeit ausgebaut sind und sich ihre Leistung leicht anpassen lässt.
Plattform- und Geräteunterstützung	Breite Unterstützung in modernen Webbrowsern; erfordert keine zusätzlichen Plugins oder Softwareinstallationen auf Benutzergeräten	Abhängig von der gewählten Technologie, aber es ist davon auszugehen dass diese Gerät unabhängig funktionieren.	Bereite Unterstützung, da der Code direkt in der Cloud läuft.

Entwicklungsprojekt  
Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

Nach weiteren möglichen Lösungsansätzen haben wir recherchiert.

Hier findet ihr die Tabelle: [https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Weitere\\_Lösungsansätze\\_v1.0-2.jpg](https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Weitere_Lösungsansätze_v1.0-2.jpg)

# Exkurs: Leitfragen

## Architektur des Systems

- Welche Interaktionen bzw. Kommunikationen sind synchron und welche asynchron um wie ist das aus dem Problemszenario begründet?
  - Synchron: Die Steuerung der Medien Wiedergabe ist in beiden Ansätzen Synchron um der Anforderung eines gemeinsamen Erlebnisses gerecht zu werden.
  - Asynchron: Im Falle des Tunneling Ansatzes sind die Interaktionen mit dem SessionCache Asynchron.
- Welche Protokolle sind für die Realisierung geeignet und warum wurden diese Protokolle gewählt?
  - WebRTC (direktes P2P)
  - HTTPS & WebSocket (sichere und interaktive Kommunikation)
  - localtunnel (einfach Aufzusetzen und zu implementieren)
  - WebTorrent (benötigt keinen Server oder Torrent Client)
  - UPnP (ermöglicht Verbindung zwischen Geräten im selben lokalen Netzwerk)
- Welche Daten werden in den einzelnen Kommunikationsschritten ausgetauscht?
  - Nutzerdaten (Name / Rechte)
  - Steuerbefehle
  - Streaming-Informationen
- Werden Dienste im Web genutzt? Wenn ja: Wie ist deren Benutzung hinsichtlich des Datenschutzes zu rechtfertigen und wie ist die Abhängigkeit zu bewerten?
  - Lediglich die Cloud Lösung nutzte einen Dienst im Web (AWS). Die Abhängigkeit von AWS hätte den Vorteil gehabt, dass Nutzer die Anwendung hätten nutzen können, ohne sie auf ihrem Gerät haben zu müssen.
- Sind die Komponenten auf verteilten Systemen angesiedelt?

Entwicklungsprojekt  
Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

Unsere vollständige Ausarbeitung zu den Leitfragen findet ihr hier:  
<https://little-cashew-cf8.notion.site/Prozessassessment-und-Fazit-3ffc77b69e804096b420ca1e4c7f0969?pvs=74>

# Prozessassessment & Fazit

## Prozessassessment

### 1. Zieldefinition:

*Überprüfung, ob die Ziele des Projekts klar definiert und kommuniziert wurden*

Das Hauptziel des Projekts "ARlebnispfade Oberberg" liegt in der Entwicklung einer Anwendung, die eine synchrone Wiedergabe von Medieninhalten wie Videos und Audios auf den Geräten mehrerer Benutzer ermöglicht, ohne auf zentrale Server zurückzugreifen. Im Rahmen des Entwicklungsprojektes sollten für diesen Zweck verschiedene Synchronisierungsmethoden miteinander verglichen und bewertet werden. Hierfür wurden ursprünglich zwei Technologien evaluiert: WebRTC für direkte Peer-to-Peer-Kommunikation und Cloud Services für Serverless Kommunikation. Letzteres wurde später durch Tunneling als Methode zur Umgehung von Netzwerkbegrenzungen ersetzt. Die klare Kommunikation dieser technischen Ziele an Stakeholder wurde durch vereinfachte Darstellungen und regelmäßige Updates gewährleistet, um die Relevanz und den Fortschritt des Projekts transparent zu machen. Ein besonderer Fokus lag auf der Sicherstellung, dass die Anwendung einfach, anonym und ad hoc von den Nutzern verwendet werden kann, was die Grundlage für ein innovatives Lernerlebnis im Bereich der kulturellen Bildung bildet. Weiter wurden Bewertungsmatrizen erarbeitet, mit denen die gewählten Technologien im Hinblick auf Bewertungskriterien aus der Perspektive von Entwicklern und Nutzern und die Erfüllung der Anforderungen miteinander verglichen und bewertet werden können.

### 2. Technologieauswahl:

*Bewertung der Entscheidungsfindung bei der Auswahl von Technologien, insbesondere im Hinblick auf die serverless Architekturen*

Entwicklungsprojekt  
Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

Unsere vollständige Ausarbeitung zum Prozessassessment und unser Fazit findet ihr hier:

<https://little-cashew-cf8.notion.site/Prozessassessment-und-Fazit-3ffc77b69e804096b420ca1e4c7f0969?pvs=74>

# Arbeitsmatrix

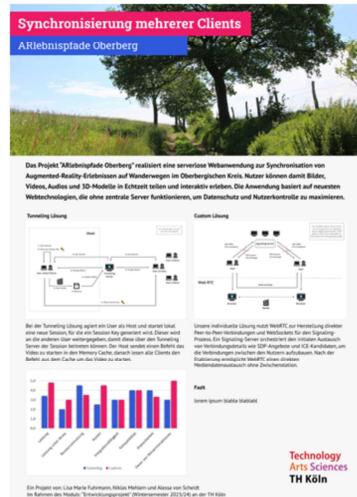
Aufgabenbereich	Lisa	Niklas	Alessa
Problemraum-Recherche	33.3%	33.3%	33.3%
Domänenanalyse	33.3%	33.3%	33.3%
Modellierung	30%	30%	40%
PoCs	40%	30%	30%
Prototypes	40%	40%	21%
Technologierecherche	35%	40%	25%
Bewertungsmatrix	25%	25%	50%
Poster	25%	25%	50%
Assessment und Fazit	33.3%	33.3%	33.3%
Gesamtanteil am Projekt	33.3%	33.3%	33.3%

Entwicklungsprojekt  
Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

Mit der Arbeitsteilung innerhalb des Teams waren wir alle zufrieden. Die Arbeitsmatrix wurde nur der Vollständigkeit halber angelegt.

Hier findet ihr die Arbeitsmatrix: [https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Arbeitsmatrix\\_v1.0.jpg](https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Artefakte/Arbeitsmatrix_v1.0.jpg)

# Poster



Entwicklungsprojekt  
Lisa Marie Fuhrmann, Niklas Mehlem, Alessa von Scheidt

Hier findet ihr das Poster:

[https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Audits/Audit\\_4/Poster\\_Entwicklungsprojekt\\_202324\\_WebDev\\_Fuhrmann%2CMehlem%26vonScheidt\\_Synchronisierung\\_mehrerer\\_Clients.pdf](https://github.com/lisafuhrmann/EPWS2324-FuhrmannMehlemVonScheidt/blob/main/Audits/Audit_4/Poster_Entwicklungsprojekt_202324_WebDev_Fuhrmann%2CMehlem%26vonScheidt_Synchronisierung_mehrerer_Clients.pdf)



**Vielen Dank für  
eure  
Aufmerksamkeit**

Unser Sonnenlicht: Lisa's Katze Ivy ☺