Home Theater Feasibility Study

Definition and Motivation:

This is a non-profit open source project made to demonstrate software development, methodology usage and operational and organizational skill and knowledge.

The motivation is based on the rising annoyance of streaming services including higher prices for services while implementing ads into their high-price applications. With DVD's and movies lying around unused, implementing a private streaming service could solve the problem of having to pay for multiple streaming services.

Technical

Tech stack includes:

```
spring-boot-starter-web 3.5.5
Apache-2.0
spring-boot-starter-websocket 3.5.5
Apache-2.0
spring-boot-starter-thymeleaf 3.5.5
Apache-2.0
spring-boot-starter-security 3.5.5
Apache-2.0
spring-boot-starter-jdbc 3.5.5
Apache-2.0
sqlite-jdbc 3.42.0.0
Apache-2.0
```

We control all the dependencies and do not rely on third-party paid services.

We might need to implement media streaming protocols to support multiple devices, and or transcoding might be needed if devices support different formats, which can be CPU heavy.

Since this is a MVP (Minimal Viable Product) we will estimate the need for further multiple formats support in future iterations.

Network discovery:

Devices on the same internet must be able to discover the host PC, which can be implemented using the springboot framework by setting the server address to 0.0.0.0

This will result in a security problem, which forces us to implement a security or user login check, which springboot likewise includes with springboot security, where we can handle a custom loginpage, where the configuration annotated file for SecurityConfig can allow us to secure the rest of the application until the user has logged in.

Scalability:

The project can be modelled to be able to stream through server based interaction, once logged in the user could stream videos from a host PC on their own network anywhere in the world, but this would impose a larger security risk, which would need to be assessed.

Verdict:
For local network streaming the project is technically feasible, given that it does not include any new technologies, new hardware. carefully handling the security measures and making sure to test for SQL-injection on the main login page plus securing all endpoints from being viewed in any form as long as the user is not logged in.

Economical
Advantages:
Since the project is using a open source stack, there are no licensing fees.

The project is likewise locally installed on user host PC's and as long as the project is not scaled to being able to stream anywhere in the world, there are no cloud or PaaS(ervices) required.

The project is using SQLite for user data handling, which is free.

Development and assets are hosted on Github, which is free.

Since the development is of own interest, the time and effort price is free.

Potential costs:
If the project where to be scaled to allow users to stream from their host PC to anywhere in the world, that would include either using a On-Site services, which would cost me to run the server and or to use a PaaS to handle the streaming services through their servers.

Using a peer-to-peer (P2P) streaming system, with a secure relay via a central server, the users PC becomes the source of the stream, and the servers job is to mainly authenticate, connect peers securely and possibly relay streams if direct connections fail.
For example using Ant Media Server, which allows WebRTC and can act as a signaling server, and optionally relay.
This way the host PC would not need to port forward on the users PC, because the WebRTC NAT traversal handles it.
Or AWS / Azure which is scalable for many users, can provide TURN/STUN servers for NAT traversal, with security handled via standard cloud services.

*Ant Media: 1**

- Can be self-hosted on your server (one-time license + optional support).
- For enterprise-grade secure relaying, expect ~$195–$500/month depending on channels and load.

*AWS/Azure: 2**

- Cloud costs are mostly for bandwidth and relay servers (TURN).
- For 2,000 concurrent viewers, worst-case if all streams relay through the cloud, it could be expensive (~$0.08–$0.15 per GB).
- With mostly P2P connections, costs drop dramatically.

Cost Example:

- Using AWS at $0.15 per GB, a 1.5-hour movie streamed to 2,000 users with all streams relayed would cost around $600.
- If most connections succeed P2P or relay is disabled, the cost would drop to around $32–$60 for 2,000 simultaneous streams.

*Revenue Example (AdSense): *3*

- If each user sees 1 ad per movie, earnings would be around $30.
- Increasing to 4 ads/hour raises earnings to ~$60, enough to cover P2P server costs but leaving no profit.

Verdict:

If users only stream locally, the project has no cost to us, which is feasible.

If we scale to use AWS/Azure with P2P connections only, the project is feasible using 4 ads/hour to cover server costs, but there is no profit for scalability.
For larger user bases, we would need to rely on donations or sponsorships, which are variable and hard to estimate. Costs scale with users but remain measurable.

1* https://antmedia.io/

2*
https://aws.amazon.com/free/?trk=5814fef8-1ec5-4582-8e31-98f70b00afc2&sc_channel=ps&ef_id=CjwKCAjw3tzHBhBREiwAlMJoUuYc7VEhX5OD9QYTIuWsjY_VoaxT5EpquOtPXxyFF0hM2VyL_sdhCBoCI_gQAvD_BwE:G:s&s_kwcid=AL!4422!3!444204369308!e!!g!!aws!10287741720!106541669270&trk=8b3cfe80-3b72-45e5-af97-ee5567921658&sc_channel=ps&targetid=kwd-11698161&gad_campaignid=10287741720&gbraid=0AAAADjHtp9Ji66oMM9s7o7R2T1HLeSk9&gclid=CjwKCAjw3tzHBhBREiwAlMJoUuYc7VEhX5OD9QYTIuWsjY_VoaxT5EpquOtPXxyFF0hM2VyL_sdhCBoCI_gQAvD_BwE

3* https://adsense.google.com/start/#calculator

Legal

The project of course cannot distribute copyrighted content, that is why it relies on that the user owns the movies or videos legally.
Streaming within a private network for personal content using a third party application like this, is mostly legal.

Using the Springboot Framework and security including the Thymeleaf dependency imposes that we must include a Notice or Licences using the Apache License 2.0 license disclosure.
* 4

4* https://www.apache.org/licenses/LICENSE-2.0

If we scale the project and end up using external libraries that have other licenses, we should make sure to comply with the given licenses.

Verdict:
The project is legally feasible, provided the app itself does not provide or host copyrighted media, which it should not.

Operational
The users install the program on their PC, no server setup required, even if the project is scaled to use AWS or Azure to connect P2P.
The program is requires minimal maintenance, and version control should be automated, which would conclude that the user would not have to interact with the program to get the newest update.

The UI must be very user friendly, and or the instruction manual should be very comprehensive to minimize the problems that the user might have.
This includes handling errors or unsupported media gracefully to minimize bad user experience.

Verdict:
The project is operationally feasible,

Market Factors
The target audience are users that have personal movie collections, that want to reuse them in a modern way, Home media enthusiasts, small families or technically inclined users.

There is a large amount of competitors like Plex, Jellyfin and Emby, which are also all open source and freemioum media services

The marked is niche but there is a segment looking for completely local, privacy friendly streaming.

To compete with the larger competitors, we would have to implement the P2P feature, but for now, since the project is solely a personal project that is not a problem.

Verdict:
Moderate market potential, but given its a private project it is feasible.

Conclusion

Technical: Feasible with attention to streaming implementation and device compatibility.

Economic: Very low-cost and low-risk.

Legal: Feasible if users only stream their own content.

Operational: Feasible with good UI/UX design.

Market: Niche, but a valid target audience exists.

The project is overall feasible and is a “GO”.

Niklas Holt Läu 21/10/2025