

ZÁVĚREČNÁ STUDIJNÍ PRÁCE

dokumentace

Mobilní aplikace pro ovládání funkcí počítače

Jiří Gebauer



FOSS-Deck

Obor: 18-20-M/01 INFORMAČNÍ TECHNOLOGIE
se zaměřením na počítačové sítě a programování

Třída: IT4

Školní rok: 2025/2026

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým účelům na Střední průmyslové a umělecké škole v Opavě, Praskova 399/8.

V Opavě 6. 1. 2026

podpis autora práce

ABSTRAKT

Tato práce se zabývá návrhem a realizací systému pro vzdálené ovládání vybraných funkcí osobního počítače pomocí mobilního zařízení. Řešení je založeno na klient-server architektuře, kde serverová část běží na osobním počítači a klientská část je realizována jako webová aplikace optimalizovaná pro mobilní zařízení. Komunikace mezi klientem a serverem probíhá prostřednictvím protokolu WebSocket. Serverová aplikace napsaná v Rustu zajišťuje ovládání systémových funkcí operačního systému, ovládání hlasitosti zvuku, ovládání multimediálního přehrávání a dalších vybraných akcí. Klientská aplikace vytvořená pomocí frameworku Tauri poskytuje uživatelské rozhraní ve formě konfigurovatelné sady ovládacích prvků, které umožňují jednoduché a intuitivní ovládání počítače na dálku. Součástí řešení je také mechanismus párování zařízení a autentizace, který zajišťuje základní úroveň zabezpečení komunikace.

Klíčová slova

klient, server, WebSocket, Rust, Tauri, párování, autentizace

ABSTRACT

This thesis deals with the design and implementation of a system for remote control of selected personal computer functions using a mobile device. The solution is based on a client-server architecture, where the server part runs on a personal computer and the client part is implemented as a web application optimized for mobile devices. Communication between the client and server takes place via the WebSocket protocol. The server application, written in Rust, provides control of operating system functions, audio volume control, multimedia playback control, and other selected actions. The client application, created using the Tauri framework, provides a user interface in the form of a configurable set of controls that enable simple and intuitive remote control of the computer. The solution also includes a device pairing and authentication mechanism that provides a basic level of communication security.

Keywords

client, server, WebSocket, Rust, Tauri, pairing, authentication

OBSAH

ÚVOD.....	5
1 TEORETICKÁ A METODICKÁ VÝCHODISKA.....	6
1.1 KLIENT-SERVER ARCHITEKTURA	6
1.2 WEBSOCKET KOMUNIKACE	6
1.3 BEZPEČNOST A AUTENTIZACE	6
2 VYUŽITÉ TECHNOLOGIE	7
3 ZPŮSOBY ŘEŠENÍ A POUŽITÉ POSTUPY.....	8
3.1 ZAČÁTEK PROJEKTU	8
3.2 PRVNÍ BUILD PRO TELEFON.....	8
3.3 PŘIDÁNÍ PÁROVÁNÍ	8
3.4 ÚPRAVA UI	9
3.5 PŘIDÁNÍ DALŠÍCH FUNKCÍ	10
3.6 BUILD APLIKACÍ	10
4 VÝSLEDKY ŘEŠENÍ, VÝSTUPY, UŽIVATELSKÝ MANUÁL.....	11
4.1 VÝSLEDEK.....	11
4.2 NESPLNĚNÉ CÍLE	11
4.3 MANUÁL	11
ZÁVĚR	12
SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ	13

ÚVOD

Cílem této práce bylo vytvořit mobilní aplikaci jako alternativu k zařízení Stream Deck od firmy Elgato. K tomuto tématu jsem se rozhodl z toho důvodu, že oficiální mobilní aplikace tohoto zařízení má některé funkce dostupné pouze po zaplacení a zároveň jsem nenašel jinou funkční alternativu, která by byla volně dostupná a splňovala mé požadavky.

Navržené řešení se skládá ze dvou aplikací. První z nich je serverová aplikace, která běží na počítači a vykonává jednotlivé funkce na základě příkazů zaslaných z mobilního telefonu. Druhou částí je klientská aplikace, která běží na mobilním zařízení a slouží jako uživatelské rozhraní. Obě aplikace spolu komunikují v rámci stejné lokální sítě.

Serverovou část projektu jsem se rozhodl implementovat v programovacím jazyce Rust, jelikož jsem s tímto jazykem měl již předchozí zkušenosti a považuji jej za programovací jazyk s dobrým výhledem do budoucna. Mobilní aplikace byla vytvořena pomocí frameworku Tauri, jehož backend je rovněž napsán v jazyce Rust. Frontend mobilní aplikace je realizován pomocí technologií HTML a JavaScript, se kterými mám rovněž předchozí zkušenosti.

1 TEORETICKÁ A METODICKÁ VÝCHODISKA

1.1 Klient-server architektura

Klient-server architektura je jedním ze základních konceptů moderních síťových aplikací. Server poskytuje služby a zdroje, zatímco klient o tyto služby žádá. Tento model umožňuje centralizovanou správu logiky aplikace a zároveň jednoduché připojení více klientů.

V projektu FOSS-Deck je serverová část zodpovědná za komunikaci s operačním systémem počítače a vykonávání příkazů, zatímco klientská část slouží jako uživatelské rozhraní.

1.2 WebSocket komunikace

WebSocket je aplikační protokol umožňující trvalé obousměrné spojení mezi klientem a serverem. Na rozdíl od klasického HTTP dotazování umožňuje serveru aktivně zasílat data klientovi bez nutnosti opakovaných požadavků.

Tato vlastnost je klíčová pro aplikace vyžadující okamžitou odezvu, například při synchronizaci stavu hlasitosti nebo přehrávání médií.

1.3 Bezpečnost a autentizace

Při návrhu vzdáleného ovládání počítače je nutné klást důraz na zabezpečení. Neoprávněný přístup by mohl vést k zneužití systému. Z tohoto důvodu FOSS-Deck využívá mechanismus párování zařízení pomocí jednorázového kódu a následné autentizace pomocí tokenů.

2 VYUŽITÉ TECHNOLOGIE

Serverová část aplikace je implementována v programovacím jazyce Rust, který je známý svou bezpečností, vysokým výkonem a moderním přístupem ke správě paměti. Pro síťovou komunikaci je využit framework Warp a asynchronní běhové prostředí Tokio.

Klientská část je realizována jako webová aplikace pomocí HTML5, CSS3 a JavaScriptu. Důraz byl kladen na responzivní design a použitelnost na mobilních zařízeních. Pro ukládání lokálních dat je využito rozhraní LocalStorage.

3 ZPŮSOBY ŘEŠENÍ A POUŽITÉ POSTUPY

3.1 Začátek projektu

Nejdříve jsem vytvořil projekt pro počítač skrze RustRover a poté pomocí npm vytvořil šablonu pro Tauri projekt, přičemž jsem se rozhodl využít šablonu s JavaScriptem. První verze programu byl pouze otevřený port na počítači, na který se po zadání správné IP adresy připojil telefon, jenž pak mohl ovládat zvuk počítače.

3.2 První build pro telefon

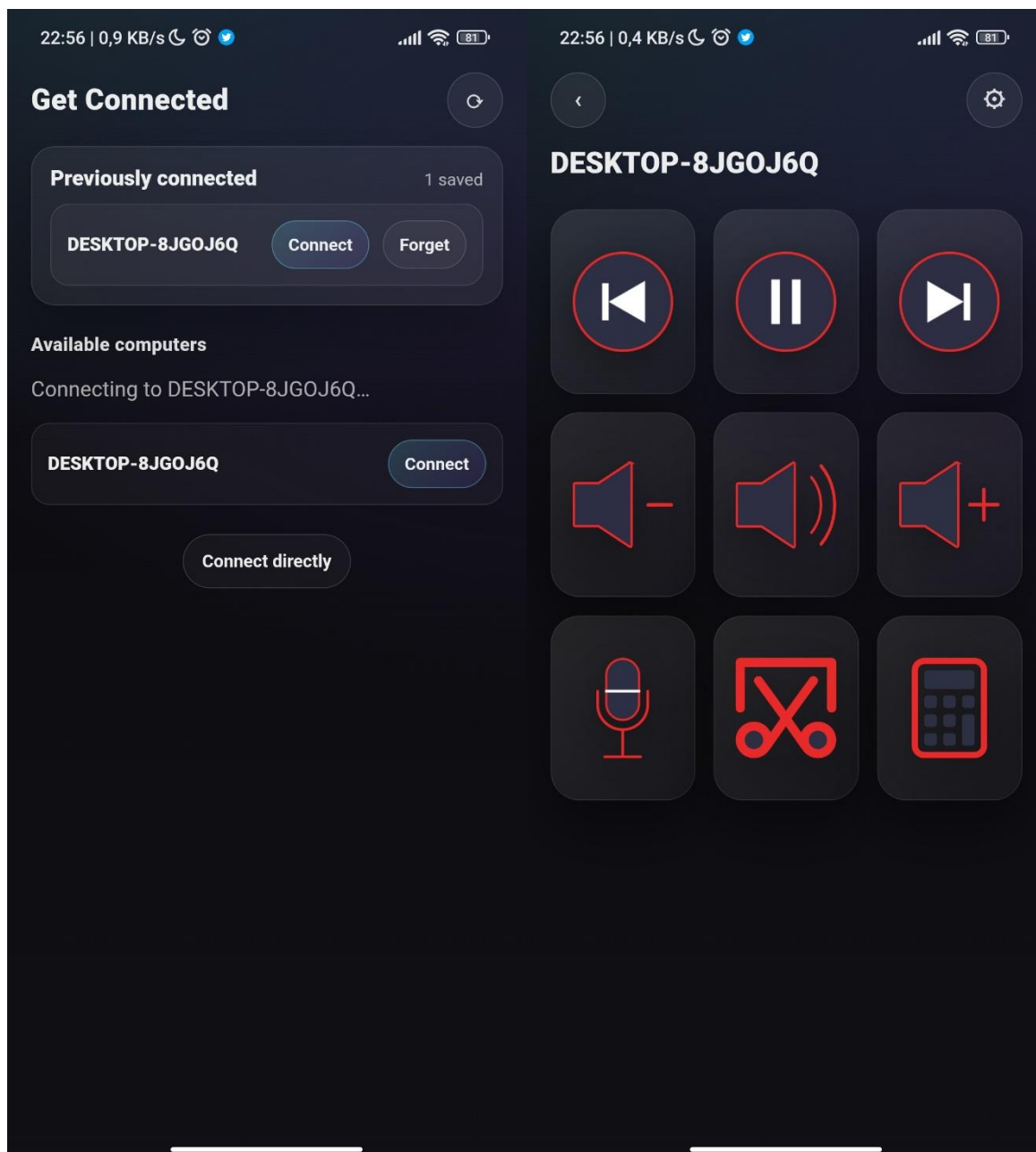
Chtěl jsem přidat možnost skenovat zařízení v okolí, a jelikož jsem do té doby používal pouze emulátor, který se mohl pouze připojit přes lokální adresu 10.0.2.2, tak jsem musel buildnout aplikaci na telefon. Tauri to naštěstí udělalo jednoduché, kdy stačilo pouze na telefonu, který kabelem připojíte k počítači, povolit ladění přes USB a automaticky to začalo instalovat přímo do telefonu. Mohl jsem tak přidat funkci do aplikace, která se umí vyhledat počítače, jenž mají zapnutou viditelnost. V GUI počítače je možné buď zapnout pouze server, takže se na něj dá připojit pouze napřímo, nebo se k tomu dá zapnout i viditelnost, takže se počítač bude zobrazovat v mobilní aplikaci.

3.3 Přidání párování

Jelikož jsem chtěl zabezpečit připojení, tak aby se nemohl jen tak někdo připojit, tak jsem přidal nutnost zadat párovací kód, který se vygeneruje na obrazovce počítače. Později jsem přidal automatickou autentizaci, kdy telefony které se už jednou spárovaly budou pomocí unikátního tokenu automaticky spárovány bez nutnosti znovu zadat párovací kód. Zapamatované telefony je možné odebrat v GUI pomocí tlačítka „Revoke“

3.4 Úprava UI

Jelikož je hlavně důležitý vzhled mobilní aplikace, tak jsem ji upravil, aby to již nebyla pouze bílá stránka se třemi tlačítky a logem pro debuggování. Aplikace má dvě stránky, první ve které se páruje s počítačem a druhá ve které je již spárovaná a ukazuje tlačítka na ovládání počítače.

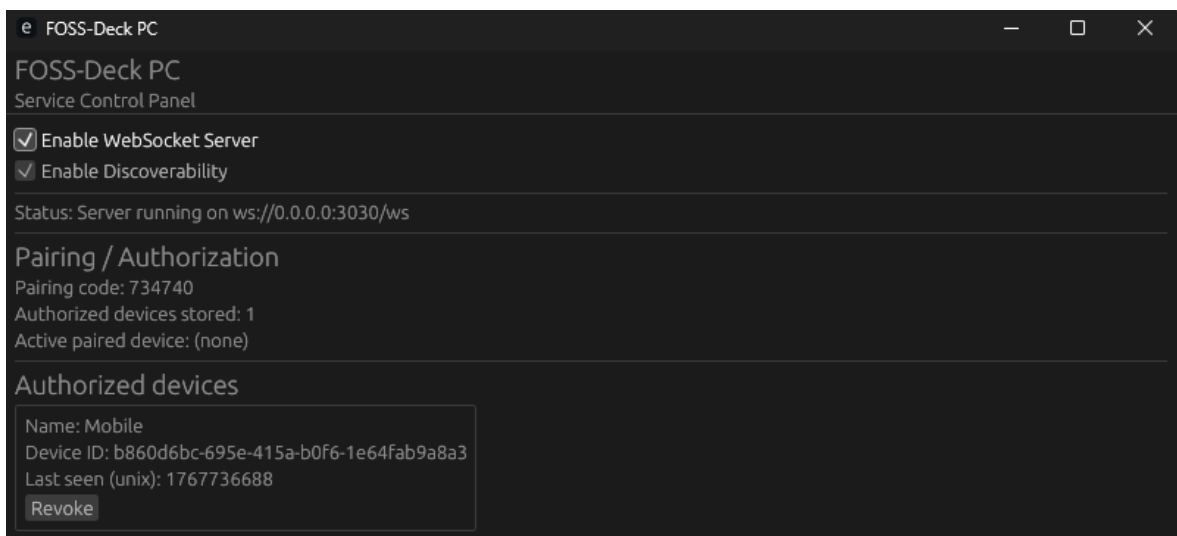


3.5 Přidání dalších funkcí

Doted' bylo možné ovládat pouze hlasitost zvuku, a tak jsem přidal další funkce, které bych osobně používal. Možnost přepínání multimédií, ovládání mikrofону, screenshot obrazovky a otevření kalkulačky.

3.6 Build aplikací

Na závěr již zbývalo pouze buildnout release verzi aplikací. To bylo u počítačové aplikace jednoduché, stačilo použít `cargo build --release` a ve složce `/target` bylo optimalizované `.exe`. U mobilní aplikace to bylo složitější, pomocí „keytool“ jsem si vytvořil unikátní podpis, jehož cestu jsem poté přidal do tauri configu, který pak veškeré buildy tvořil s již podepsaným `.apk`. Tyto spustitelné programy jsem poté releasnul na Github repositáři tohoto projektu.



4 VÝSLEDKY ŘEŠENÍ, VÝSTUPY, UŽIVATELSKÝ MANUÁL

4.1 Výsledek

Výsledkem práce je funkční aplikace umožňující vzdálené ovládání základních funkcí počítače. Byly splněny všechny hlavní cíle práce, zejména vytvoření stabilní komunikace mezi klientem a serverem a implementace uživatelsky přívětivého rozhraní.

4.2 Nesplněné cíle

Hlavní nesplněný cíl je, že jsem chtěl přidat možnost spouštět stisknutím tlačítka aplikace i řadu aplikací a popřípadě je i nějak nastavit, to by však vyžadovalo předělání jak aplikace momentálně funguje, anebo bych to mohl implementovat pouze v omezené verzi, kde by šlo spouštět pouze předdefinované aplikace.

Byl i optimistický cíl, kdy bych vytvořil i verzi pro linux, vzhledem k tomu, že i Stream Deck oficiálně podporuje pouze Windows, tak si však nemyslím, že bych toho byl schopen, maximálně v omezené verzi.

Úprava GUI na počítači, nebyla nikdy úplně v plánu, jelikož není nijak důležitá, avšak dala by se zlepšit.

4.3 Manuál

Spuštění a používání aplikace je jednoduché, z Githubu si stáhněte aplikaci pro windows a pro android a pokud máte zařízení na stejné síti, tak na počítači zapnete server a popřípadě i viditelnost, na telefonu pak buď zadáte IP adresu počítače, nebo použijete funkci skenování.

ZÁVĚR

Práci hodnotím obecně pozitivně, vyzkoušel jsem si více pracovat s Rustem a i tvořit mobilní aplikaci. Většinu cílů, které jsem si dal, jsem splnil a jako hlavní vylepšení, na kterém bych pracoval, je právě možnost práce s aplikacemi. Pro mé osobní využití, jsem však přidal vše co bych sám použil.

Kód je k nalezení zde: <https://github.com/007king700/FOSS-Deck>

SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ

- [1] *Tauri*. Online. Dostupné z: <https://v2.tauri.app/>. [cit. 2026-01-06].
- [2] *Stack overflow*. Online. [Stackoverflow.com/questions](https://stackoverflow.com/questions). [cit. 2026-01-06].
- [3] *ChatGPT*. Online. Dostupné z: <https://chatgpt.com/>. [cit. 2026-01-06].