VYSOKÁ ŠKOLA POLYTECHNICKÁ JIHLAVA

Aplikovaná informatika

VIZUALIZACE DAT PRO SOFTWAROVÉ ŘÍZENÍ ELEKTROCHEMICKÝCH ANALYTICKÝCH PŘÍSTROJŮ

Bakalářská práce

Autor práce: Benjamin Hofrichter

Vedoucí práce: PaedDr. František Smrčka, Ph.D.

Jihlava 2026

Vysoká škola polytechnická Jihlava

Tolstého 16, 586 01 Jihlava

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Autor práce:	**Benjamin Hofrichter**
Studijní program:	Aplikovaná informatika
Garant studijního programu:	Ing. Lenka Kuklišová Pavelková, Ph.D.
Název práce:	**Vizualizace dat pro softwarové řízení elektrochemických analytických přístrojů**
Vedoucí práce:	PaedDr. František Smrčka, Ph.D.
Cíl práce:	Cílem této bakalářské práce je navrhnout a implementovat webovou aplikaci pro softwarové řízení naměřených dat z elektrochemických měřících přístrojů (např. pro analýzu baterií, palivových článků apod.), která umožní nejen efektivní vizualizaci naměřených dat v reálném čase, ale také interaktivní správu měření. Práce se také zaměří na výběr vhodné knihovny pro tvorbu grafů, která dokáže zpracovat velké množství dat, a její následnou integraci do software dodávaného firmou Kolibrik.net s využitím webových technologií. Pro vizualizaci bude použit jazyk TypeScript.

Abstrakt

Bakalářská práce se zabývá návrhem a implementací webové aplikace pro vizualizaci a správu dat z elektrochemických analytických přístrojů. Zaměřuje se na efektivní zobrazení rozsáhlých datových souborů vznikajících při měřeních baterií, palivových článků a dalších elektrochemických systémů, a to v reálném čase. Součástí práce je analýza specifik elektrochemických dat a porovnání moderních JavaScriptových knihoven určených pro vizualizaci velkých časových řad. Na základě této analýzy byla vybrána knihovna Apache ECharts, která byla následně integrována do webové aplikace vyvíjené ve spolupráci se společností Kolibrik.net. Výsledkem je funkční prototyp aplikace umožňující interaktivní práci s měřeními, jejich přehlednou vizualizaci a snadnou správu prostřednictvím moderních webových technologií.

Klíčová slova

vizualizace dat; elektrochemické měření; webová aplikace; real-time data; Apache ECharts; TypeScript; React

Abstract

The bachelor's thesis focuses on the design and implementation of a web application for visualization and management of data obtained from electrochemical analytical instruments. The work addresses efficient real-time visualization of large data sets generated during measurements of batteries, fuel cells, and other electrochemical systems. The thesis includes an analysis of the characteristics of electrochemical data and a comparison of modern JavaScript libraries suitable for visualizing large time series. Based on this analysis, the Apache ECharts library was selected and integrated into a web application developed in cooperation with the company Kolibrik.net. The result is a functional application prototype that enables interactive data handling, clear visualization of measurements, and efficient data management using modern web technologies.

Keywords

data visualization; electrochemical measurement; web application; real-time data; Apache ECharts; TypeScript; React

Prohlašuji, že předložená bakalářská práce je původní a zpracoval/a jsem ji samostatně. Prohlašuji, že citace použitých pramenů je úplná, že jsem v práci neporušil/a autorská práva (ve smyslu zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů, v platném znění, dále též „**AZ**“).

Byl/a jsem seznámen/a s tím, že na mou bakalářskou práci se plně vztahuje **AZ**, zejména § 60 (školní dílo).

Podle § 47b zákona o vysokých školách souhlasím se zveřejněním své práce podle Směrnice pro vedení, vypracování a zveřejňování závěrečných prací na VŠPJ, a to bez ohledu na výsledek obhajoby.

Beru na vědomí, že VŠPJ má právo na uzavření licenční smlouvy o užití mé bakalářské práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé bakalářské práce (prodej, zapůjčení apod.).

Jsem si vědom/a toho, že užít své bakalářské práce či poskytnout licenci k jejímu využití mohu jen se souhlasem VŠPJ, která má právo ode mě požadovat přiměřený příspěvek na úhradu nákladů, vynaložených vysokou školou na vytvoření díla (až do jejich skutečné výše), z výdělku dosaženého v souvislosti s užitím díla či poskytnutím licence.

V Jihlavě dne 14. dubna 2026

…….

Podpis studenta/ky

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

3D	Three-dimensional (trojrozměrný)
API	Application Programming Interface
BSD	Berkeley Software Distribution
CPU	Central Processing Unit
ČR	Česká republika
CSS	Cascading Style Sheets
DOM	Document Object Model
EIS	Electrochemical Impedance Spectroscopy
GPU	Graphics processing unit
HTML	Hyper Text Markup Language
JSON	JavaScript Object Notation
LOD	Level of Detail
LTTB	Largest Triangle Three Buckets
MIT	Massachusetts Institute of Technology
SVG	Scalable Vector Graphics
UI	User Interface
USD	United States Dollar
VŠPJ	Vysoká škola polytechnická Jihlava

Úvod

Elektrochemické analytické přístroje představují nezbytný nástroj při výzkumu a testování moderních zdrojů energie, mezi které patří zejména lithium-iontové baterie, palivové články či nové typy akumulátorů. Uvedené přístroje umožňují provádět řadu experimentů a testů, cílem měření je získaní přesné informace o chování elektrochemických systémů v závislosti na čase, proudu, napětí nebo dalších podmínkách. Díky nim lze sledovat kapacitu, životnost, účinnost či vnitřní odpor baterií, stejně jako jejich odezvu na dynamické změny zatížení. Výsledkem takových měření jsou rozsáhlé soubory dat, které je nutné efektivně zpracovat a interpretovat.

Společnost Kolibrik.net se dlouhodobě zaměřuje na vývoj softwarových a hardwarových řešení pro řízení a vyhodnocování elektrochemických procesů. Její produkty, jako jsou systémy pro měření elektrochemické impedance nebo cyklovače baterií, jsou využívány především národními i mezinárodními výzkumnými institucemi a vývojovými laboratořemi, kde slouží k získávání detailních informací o chování elektrochemických článků.

Téma bakalářské práce vychází z praktické spolupráce autora se společností Kolibrik.net. V rámci ní vznikl prototyp webové aplikace určené pro vizualizaci a správu dat z elektrochemických měření. Projekt se stal základem pro bakalářskou práci, které se v práci rozvíjí o teoretické zázemí, systematický návrh architektury a vyhodnocení z hlediska uživatelských i technických požadavků.

Cílem práce je navrhnout a implementovat webovou aplikaci, která umožní nejen efektivní vizualizaci naměřených dat z elektrochemických přístrojů v reálném čase, ale také interaktivní správu jednotlivých měření. Součástí je výběr vhodné knihovny pro vizualizaci velkých datových objemů a její integrace do softwarového prostředí vyvíjeného firmou Kolibrik.net. Výsledná aplikace má uživatelům poskytnout nástroj, který usnadní práci s experimentálními daty a zvýší jejich přehlednost.

Práce je rozdělena do několika kapitol. V úvodní části je nastíněno teoretické pozadí elektrochemických měření a typů dat, která se v této oblasti vyskytují. Následuje kapitola věnovaná problematice vizualizace dat v reálném čase. Poté jsou představeny webové technologie využité pro vývoj aplikace a proveden výběr vhodné knihovny pro tvorbu grafů. Další část práce se zaměřuje na analýzu a návrh řešení, implementaci a testování aplikace. Závěrečná kapitola shrnuje dosažené výsledky a zhodnocuje splnění cílů práce.

1 Teoretická část

Teoretická část shrnuje poznatky potřebné pro návrh a realizaci webové aplikace určené pro řízení a vizualizaci elektrochemických měření. Nejprve jsou popsány základní principy elektrochemických procesů, měřicí metody používané v systémech Kolibrik.net a charakter dat, která při těchto měřeních vznikají. Následuje přehled přístupů k vizualizaci rozsáhlých časových řad a požadavků spojených se zpracováním dat v reálném čase. Závěrečná část představuje webové technologie využité při vývoji nové aplikace Artemis, která navazuje na původní software Kolibrik.net a přináší modernější uživatelské rozhraní a efektivnější práci s daty. Uvedené poznatky tvoří teoretický základ pro praktickou část práce.

1.1 Elektrochemické měření a typy dat

Elektrochemická měření představují základní prostředek pro analýzu a charakterizaci bateriových článků, palivových článků či jiných elektrochemických systémů. Cílem měření je získat informace o vlastnostech elektrochemické soustavy, jejím aktuálním stavu a změnách, které probíhají v průběhu nabíjení, vybíjení nebo klidového režimu. Z naměřených dat lze určovat veličiny jako kapacita, účinnost, vnitřní odpor, impedance nebo životnost článku. (Bouzek, 1999) Uvedené veličiny jsou zásadní pro hodnocení chování a stability akumulátorů a tvoří podklad pro vývoj bezpečných a výkonných elektrochemických systémů.

V současnosti se metody elektrochemické analýzy uplatňují v širokém spektru výzkumných i praktických aplikací. Ve vědeckém prostředí slouží k diagnostice trakčních Li-ion baterií, kde umožňují sledovat změny impedance, kapacity či teplotního chování během dlouhodobého zatěžování. Elektrochemická impedanční spektroskopie je jednou z klíčových metod, protože poskytuje detailní informace o vnitřních procesech článku a umožňuje odhadovat jeho stav zdraví (Liu et al., 2023). Podobné postupy využívají také výzkumné a vývojové týmy zabývající se testováním akumulátorů pro vysokovýkonové systémy, například elektrické formule, drony či prototypy elektrických vozidel, kde je nezbytné sledovat odezvu článků na dynamické změny proudového zatížení. V průmyslových a experimentálních laboratořích pomáhají měření ověřovat nové elektrodové materiály, hodnotit chování článků v extrémních podmínkách nebo optimalizovat nabíjecí strategie (Kolibrik.net, 2025a).

1.1.1 Typy měření v systémech vyvíjených společností Kolibrik.net

Společnost Kolibrik.net vyvíjí měřicí a řídicí systémy určené pro výzkum a vývoj elektrochemických článků. Součástí těchto systémů je softwarová aplikace, která zajišťuje řízení experimentů, sledování průběhu měření a zpracování výsledků v reálném čase (Kolibrik.net, 2025a). Systémy kombinují několik typů elektrochemických metod, které poskytují komplexní pohled na chování a degradaci bateriových článků. Jedná se o následující typy:

a) Měření napětí a proudu

Základními měřenými veličinami jsou elektrické napětí (U) a proud (I), jejichž časové průběhy se sledují při nabíjení, vybíjení či v klidovém režimu. Z těchto dat lze odvozovat kapacitu článku, účinnost nabíjecích cyklů nebo změny vnitřního odporu. Naměřené průběhy se ukládají ve formě

časových řad s vysokou vzorkovací frekvencí, často až v jednotkách kilohertzů (Kolibrik.net, 2025a).

b) Elektrochemická impedance (EIS – Electrochemical Impedance Spectroscopy)

Elektrochemickou impedanční spektroskopii podporuje systém Kolibrik prostřednictvím modulu MegaEIS. EIS umožňuje sledovat dynamické procesy uvnitř elektrochemického článku v širokém frekvenčním rozsahu a poskytuje detailní informace o přenosu náboje, difuzi iontů či tvorbě povrchových vrstev (Kolibrik.net, 2025b). Díky své citlivosti na změny vnitřního stavu článku patří EIS mezi klíčové metody používané pro diagnostiku a odhad stavu zdraví baterií (Liu et al., 2023).

c) Chronoamperometrie a chronopotenciometrie

Aplikace umožňuje i tzv. chrono-metody, které sledují odezvu článku na skokovou změnu proudového nebo napěťového zatížení. Chronoamperometrie sleduje proud při konstantním napětí, chronopotenciometrie naopak napětí při konstantním proudu. Postupy se používají ke studiu reakčních rychlostí a kapacitních vlastností elektrochemických systémů. (Šimek, 1997)

d) Cyklické zatěžování a profilování

Systémy Kolibrik.net podporují řízené nabíjecí a vybíjecí profily („load cycling“), které umožňují analyzovat cyklickou životnost a sledovat pokles kapacity v čase. Během testů jsou zároveň měřeny teploty, napětí jednotlivých článků a další provozní parametry (Kolibrik.net, 2025a).

e) Měření teploty a dalších doplňkových veličin

Je umožňováno vícekanálové snímání teploty článků a modulů, případně i dalších doplňkových veličin, jako je tlak nebo koncentrace plynů. Hodnoty mají zásadní význam pro posouzení bezpečnosti a provozního stavu baterií. (Kolibrik.net, 2025a).

Všechna měření jsou zaznamenávána ve formě časových řad, které obsahují hodnoty napětí, proudu, teploty, případně komplexní složky impedance. Datové soubory mohou dosahovat velkého objemu – během dlouhodobých cyklovacích testů vznikají záznamy v řádu milionů datových bodů, zejména při vysoké vzorkovací frekvenci používané měřicími systémy Kolibrik.net (Kolibrik.net, 2025a).

Z hlediska vizualizace je proto nutné zajišťovat efektivní zpracování a agregaci dat, aby bylo možné:

- sledovat průběhy veličin v reálném čase,
- přepínat měřicí kanály a osové rozsahy,
- provádět detailní analýzy (zoom, kurzorové měření, filtrování).

Uvedený typ dat tvoří vstupní základ pro aplikaci vyvíjenou v rámci bakalářské práce, jejímž cílem je umožnit jejich interaktivní správu a vizualizaci prostřednictvím webového rozhraní.

Vizualizace elektrochemických měření představuje klíčovou součást práce s daty v oblasti výzkumu a testování akumulátorů. Během experimentů vznikají rozsáhlé a dynamicky se měnící časové řady, které je nutné zobrazovat průběžně a zároveň zajistit jejich srozumitelnost, přehlednost a rychlou odezvu uživatelského rozhraní. Současné pokročilé přístupy proto vyžadují kombinaci efektivního zpracování dat na straně serveru i klienta a volbu vhodných grafových knihoven umožňujících práci s velkými datasety.

1.2 Specifika vizualizace elektrochemických dat v reálném čase

Elektrochemická měření obvykle probíhají s vysokou vzorkovací frekvencí, často ve stovkách až tisících hertzů (Kolibrik.net, 2025a). Každý měřený kanál tak vytváří dlouhou časovou řadu, která může během experimentu obsahovat desítky tisíc až miliony vzorků. Kromě toho se měří různé typy signálů jako napětí, proud, teplota nebo impedanční odezvy, a jejich průběhy se mohou výrazně lišit (Zhang et al., 2020). Vizualizační systém proto musí zvládnout nejen velký objem dat, ale také jejich průběžné zobrazování v reálném čase a interakce ze strany uživatele

V prostředí webových aplikací se pro vykreslování grafů nejčastěji používají tři základní technologie a to SVG, Canvas a WebGL, na nich jsou postaveny běžně používané vizualizační knihovny. Každá z nich pracuje s daty jiným způsobem a má odlišné výkonové limity. SVG reprezentuje každý bod nebo čáru jako samostatný objekt v dokumentu, což je výhodné pro menší grafy, ale při tisících prvků se výkon rychle snižuje. Canvas vykresluje graf jako bitmapu a při každé změně překresluje celou scénu. Vykreslování probíhá převážně na procesoru (CPU) a při vhodné optimalizaci umožňuje plynulé zobrazení i rozsáhlejších datových řad. Tento přístup je rychlejší a běžně se používá pro rozsáhlejší časové řady, i když je omezen výkonem procesoru (Horak, 2018). WebGL je standard definovaný skupinou Khronos Group a je navržen tak, aby poskytoval nízkoúrovňový přístup ke grafickému hardware v prostředí webových prohlížečů (Khronos Group, 2024). Využívá grafickou kartu a umožňuje paralelní zpracování velkého množství bodů, což je výhodné zejména při práci se statisíci až miliony vzorků (Schütz et al., 2023). Je vhodný zejména pro velmi rozsáhlé datové sady nebo 3D vizualizace, ale jeho použití může být omezeno dostupností podpory na různých zařízeních.

Při vizualizaci elektrochemických měření je důležité, aby systém dokázal reagovat na průběžně přicházející data bez znatelného zpoždění. To vyžaduje techniky, které omezují množství výpočtů a aktualizují pouze nezbytné části grafu. V literatuře se často zmiňuje inkrementální vykreslování, kdy se nepřekresluje celý graf, ale jen jeho změněné segmenty, a práce s datovými buffery, které umožňují efektivní přidávání nových vzorků (Ortigossa et al., 2025).

Další skupinu tvoří metody redukce dat, které zachovávají tvar signálu, ale snižují počet bodů nutných k vykreslení. Patří sem decimace, kdy se vybírá pouze část vzorků, resampling, který převádí data na rovnoměrnější časovou osu, nebo modely úrovní detailu (LOD), které zobrazují různě podrobná data podle aktuálního přiblížení grafu (Liu et al., 2023). Tyto postupy se běžně používají v aplikacích pracujících s rozsáhlými časovými řadami, protože umožňují zobrazit dlouhé experimenty bez zahlcení grafu a bez ztráty čitelnosti.

Výkon vizualizace tedy závisí nejen na zvoleném vykreslovacím mechanismu, ale také na tom, jakým způsobem jsou data zpracovávána před samotným vykreslením. Kombinace vhodné technologie (SVG, Canvas nebo WebGL) a optimalizačních technik je klíčová pro to, aby bylo možné zobrazovat elektrochemická měření v reálném čase i při vysoké datové hustotě.

1.3 Porovnání vizualizačních knihoven

Pro účely bakalářské práce byla provedena systematická analýza několika dostupných JavaScriptových knihoven určených pro vizualizaci rozsáhlých datových souborů. Hlavním cílem bylo najít řešení, které spolehlivě pokryje požadavky vznikající při vizualizaci elektrochemických měření, a to zejména práci s časovými řadami obsahujícími stovky tisíc až miliony datových bodů, vykreslování průběhů v reálném čase a podporu více paralelních grafů. Současně bylo nezbytné,

aby knihovna byla snadno integrovatelná do React aplikace, která tvoří základ nového uživatelského rozhraní systému. Analýza byla provedena v rámci praxe u firmy Kolibirik.net v období únor 2025 – březen 2025.

Analýza se zaměřila na následující kritéria:

- výkon při vykreslování velkých datových sad,
- reakce na streaming dat a chování při postupném přidávání nových bodů,
- možnosti interakce, jako zoom, posun grafu, kurzorové měření nebo synchronizace více grafů,
- flexibilita a konfigurovatelnost, umožňující zobrazovat běžné časové řady i specializované grafy,
- možnosti integrace s Reactem,
- licenční podmínky, podmínkou byla použitelnost i v komerční distribuci,
- uživatelská přívětivost a náročnost implementace z pohledu vývojáře.

Následující podkapitoly shrnují zjištění o jednotlivých knihovnách, ke kterým došlo během zpracování analýzy, a jejich vyhodnocení.

1.3.1 LightningChart JS

LightningChart patří mezi nejvýkonnější knihovny dostupné na trhu. Díky plné GPU akceleraci poskytuje mimořádně rychlé vykreslování a je schopna pracovat i s desítkami milionů datových bodů bez výrazného omezení výkonu (Arction Ltd., 2024).

Nevýhodou je komplexnější implementace, vyšší nároky na porozumění API a především vysoká cena licence (od přibližně 3200 USD na vývojáře) (Arction Ltd., 2024). Z hlediska rozpočtu, požadavku na otevřenou integraci a dlouhodobé udržitelnosti proto LightningChart nepředstavuje vhodné řešení pro navrhované rozhraní.

1.3.2 Apache ECharts

Apache ECharts je výkonná open-source knihovna optimalizovaná pro velké datové objemy. Podporuje WebGL, více os, interaktivní animace, synchronizaci mezi grafy a pokročilé nástroje pro zoom či výběr dat (Apache Software Foundation, 2025).

Významnou výhodou je bezplatná licence Apache 2.0[1], vysoká přizpůsobitelnost a velmi dobrá integrace do React ekosystému. V interním testování vykazoval ECharts stabilní výkon i při vizualizaci datasetů obsahujících statisíce bodů a umožnil rychlé překreslování v reálném čase.

ECharts rovněž nabízí podporu pro zobrazování EIS dat, jako je vykreslování diagramů či interaktivní zvýraznění křivek, což je pro elektrochemická měření zásadní (Apache Software Foundation, 2025).

[1] Licence Apache 2.0 je open-source licence, která umožňuje volné použití, úpravy i komerční distribuci software bez povinnosti zveřejnit vlastní zdrojový kód (Apache Software Foundation, 2004).

1.3.3 Plotly.js

Plotly je knihovna zaměřená na vědeckou vizualizaci a nabízí širokou škálu grafů včetně 3D zobrazení, heatmap a analytických vizualizací. Je zdarma dostupná pod MIT licencí [2](Plotly Technologies, 2024).

Hlavní výhodou knihovny je velká variabilita typů grafů a možnost rychle vytvářet komplexní vizualizace bez nutnosti nízkoúrovňového programování. Nevýhodou je však náročnější konfigurace. Plotly používá rozsáhlý konfigurační objekt, který je často složitý, má mnoho vnořených struktur a některé funkce se chovají odlišně podle typu grafu. To vede k vyšší implementační zátěži, zejména pokud je potřeba kombinovat více interakcí nebo synchronizovat několik grafů současně.

Z hlediska výkonu Plotly spoléhá převážně na SVG a vykreslování pomocí Canvas, což znamená, že většina práce probíhá na CPU a každý grafický prvek (např. bod, úsečka, popisek) je reprezentován jako samostatný objekt. Při práci s kontinuálními časovými řadami je to limitující. Při větších datových sadách dochází k výraznějšímu zatížení CPU a k vyšší latenci při zoomování nebo posunu grafu.

V interním porovnání měl Plotly znatelně pomalejší odezvu při práci se statisícovými datasety a interakce nebyly tak plynulé jako u ECharts, který využívá optimalizované vykreslovací techniky.

Z těchto důvodů je Plotly vhodný pro analytické vizualizace a prezentační grafy, ale méně vhodný pro aplikace, které vyžadují plynulé real time vykreslování a práci s velkými objemy dat

1.3.4 D3.js

D3.js je nízkoúrovňová vizualizační knihovna nabízející téměř neomezené možnosti přizpůsobení grafů. Poskytuje plnou kontrolu nad vykreslováním. Použití je zdarma pod licenci BSD[3] (Bostock, 2024).

Nevýhodou je složitost při implementaci interaktivních vizualizací. D3 neobsahuje hotové komponenty grafů. Knihovna pracuje s nízkoúrovňovým modelem, kde je nutné ručně řešit škálování, osy, události myši, zoom, přepočty souřadnic, animace i samotné vykreslování prvků pomocí SVG nebo Canvasu. Každá interakce (zoom, pan, tooltip, výběr dat) vyžaduje vlastní logiku a propojení s DOM, což může po delší době vývoje a vyšší chybovosti.

Dalším omezením je výkon při práci s velkými objemy dat. D3.js vykresluje grafické prvky převážně na CPU, což je pro běžné vizualizace plně dostačující, ale při statisících bodů dochází k postupnému zpomalení a interakce, jako je zoom nebo posun, ztrácejí plynulost. Knihovna navíc neposkytuje vestavěné mechanismy pro efektivní práci se streamingovými daty, takže by bylo nutné implementovat řadu optimalizačních technik ručně. V kontextu požadavků projektu by použití D3.js znamenalo nepřiměřenou implementační zátěž a méně efektivní vývoj.

[2] MIT licence umožňuje software volně používat, upravovat a začleňovat i do komerčních aplikací. Uživatelé musí pouze zachovat původní copyrightové upozornění a text licence (MIT, 1988).

[3] BSD licence umožňuje volné použití a úpravy softwaru, včetně komerčního využití, při zachování původního licenčního oznámení (Bostock, 2024).

1.3.5 SciChart

SciChart je komerční vizualizační knihovna optimalizovaná pro vědecká a finanční data. Nabízí rychlé WebGL renderovací jádro navržené pro práci s milionovými datasety a pro aplikace vyžadující plynulé real time vykreslování (SciChart Ltd., 2024). Nabízí velmi vysoký výkon, nízkou latenci při zoomování a rychlou odezvu i při složitých interakcích, jako je synchronizace více grafů nebo práce s časovými řadami s velkým počtem datových bodů.

Nevýhodou je však placená licence, která se v dubnu 2025 pohybovala přibližně kolem 108 USD měsíčně, což by z dlouhodobého hlediska znamenalo vyšší náklady. SciChart má omezenější komunitu než u ECharts nebo Plotly, tj. méně dostupných příkladů, rozšíření a wrapperů pro moderní frameworky, takže při řešení specifických problémů je uživatel více odkázán na oficiální podporu.

SciChart sice poskytuje tzv. React wrappery (tedy komponenty umožňující použití knihovny v prostředí Reactu), jeho API je však více uzavřené a méně flexibilní, což může komplikovat integraci do projektů, které vyžadují vlastní interakce nebo nestandardní typy grafů.

Z hlediska výkonu patří SciChart mezi nejkvalitnější řešení, avšak v kombinaci s vysokými licenčními náklady a menší flexibilitou nepředstavuje optimální volbu pro projekt, který má být dlouhodobě udržitelný a komerčně provozovaný bez závislosti na drahém proprietárním řešení.

1.3.6 Highcharts

Highcharts nabízí širokou škálu grafů a silnou podporu business vizualizací. Je vhodný zejména pro dashboardy, reporty a prezentační grafy, kde je kladen důraz na vzhled a uživatelskou přívětivost. Knihovna však není optimalizována pro práci s velkými datasety ani pro vědecké či streamingové aplikace. Highcharts využívá převážně SVG rendering, což znamená, že každý grafický prvek je reprezentován jako samostatný DOM objekt. Při tisících až statisících bodů dochází k výraznému zatížení prohlížeče a interakce, jako zoom nebo posun, ztrácejí plynulost (Highcharts AS, 2024).

Další nevýhodou je komerční licencování. Highcharts je placené řešení s licencí poskytovanou na vyžádání, což zvyšuje dlouhodobé náklady. Použití Highcharts znamenalo trvalé licenční výdaje bez odpovídající přidané hodnoty oproti open source alternativám.

1.3.7 Chart.js

Chart.js je rozšířená knihovna pro základní vizualizace a menší datové objemy. Je často používaná kvůli jednoduchému API, nízké vstupní bariéře a dobré dostupnosti dokumentace, což z ní činí vhodné řešení pro rychlou tvorbu přehledných grafů. Knihovna je zdarma dostupná pod MIT licencí (Chart.js Team, 2024).

Pro potřeby vizualizace velkého množství datových sad však není vhodná. Chart.js je navržený především pro menší až středně velké datové sady a běžné webové vizualizace. Využívá vykreslování pomocí technologie v HTML5 Canvas, která je pro tyto účely dobře použitelné, avšak samotná knihovna není optimalizovaná pro práci s velmi rozsáhlými časovými řadami ani pro rychlé aktualizace v reálném čase. Při větším množství bodů dochází ke zvýšené zátěži

procesoru a interakce, jako je zoom nebo posun, ztrácejí plynulost. Knihovna také nenabízí pokročilé optimalizace pro streamingová data. (Chart.js Team, 2024).

1.3.8 Dygraphs

Dygraphs je velmi rychlá open source knihovna určená přímo pro vizualizaci časových řad. Nabízí interaktivní zoom, synchronizaci mezi více grafy, podporu dvou os a efektivní práci s daty, což z ní činí vhodné řešení pro aplikace zaměřené na průběhová měření (Dygraph, 2020).

Knihovna vychází z dřívější generace webových vizualizačních nástrojů, což se projevuje jednodušším API i vizuálním stylem. Ekosystém je omezený. Má méně dostupných rozšíření, integrací a komunitních zdrojů, takže některé funkce je nutné doplnit externími pluginy nebo vlastní implementací. Podle interních testů vykazuje Dygraphs dobrý výkon při práci s časovými řadami, ale horší vizuální kvalitu a menší možnosti přizpůsobení.

1.3.9 Vega a Bokeh

Následující dvě knihovny jsou uvedeny společně, protože představují alternativní vizualizační nástroje, které se od hlavních výše zmíněných JavaScriptových knihoven liší svým zaměřením i způsobem použití. Vega využívá deklarativní přístup k definici vizualizací a Bokeh je primárně určen pro Python.

Obě knihovny byly do přehledu zařazeny proto, aby analýza pokryla i nástroje mimo hlavní JavaScriptový ekosystém a bylo zřejmé, že byly zvažovány i alternativní přístupy k vizualizaci, přestože se nakonec ukázaly jako nevhodné pro navrhovanou aplikaci.

Vega poskytuje flexibilní vizualizační framework založený na deklarativním popisu grafů. Vizualizace jsou definovány pomocí JSON specifikace, která popisuje datové transformace, vzhled i interaktivní chování. Tento přístup umožňuje vysokou míru kontroly nad výslednou vizualizací, avšak vytváření komplexnější interaktivity je časově náročné a vyžaduje detailní znalost deklarativního modelu (Vega Project, 2024). Pro navrhovanou aplikaci, která má být integrována do React prostředí a vyžaduje rychlou práci s rozsáhlými časovými řadami, se Vega nejeví jako optimální řešení.

Bokeh je navržen především pro Python. JavaScriptová část projektu (BokehJS) slouží primárně jako vykreslovací vrstva a nenabízí plnohodnotné API pro samostatný vývoj v JavaScriptu. Některé funkce dostupné v Python verzi nejsou v BokehJS implementovány nebo mají omezenou podporu, což omezuje možnosti interaktivity i přizpůsobení (Bokeh Team, 2024). Proto není Bokeh vhodný pro aplikaci, která má být vyvíjena přímo v JavaScriptu a integrována do React prostředí.

Vzhledem k odlišnému zaměření obou knihoven a jejich omezené vhodnosti pro webovou aplikaci zaměřenou na vizualizaci rozsáhlých časových řad nebyly Vega ani Bokeh zařazeny mezi hlavní kandidáty pro implementaci.

1.3.10 Výsledek porovnání a výběr knihovny

Následující text shrnuje výsledky provedené analýzy a porovnání jednotlivých vizualizačních knihoven. Cílem bylo posoudit jejich výkon, možnosti konfigurace, způsob vykreslování

i celkovou vhodnost pro použití v aplikaci zaměřené na vizualizaci rozsáhlých elektrochemických dat. Porovnání ukázalo výrazné rozdíly mezi jednotlivými přístupy a umožnilo identifikovat knihovnu, která nejlépe odpovídá požadavkům projektu.

Pro přehledné srovnání byly jednotlivé knihovny vyhodnoceny podle vybraných kritérií, která jsou shrnuta v následujících tabulkách 1 a 2. S ohledem na přehlednost byly výsledky rozděleny do dvou tabulek. Hodnocení je provedeno kvalitativně pomocí slovních kategorií (např. „nízký“, „střední“, „vysoký“) a vychází z dostupné dokumentace knihoven a z praktického testování v rámci projektu.

Kritérium „výkon při velkých datasetech“ reflektuje zejména schopnost knihovny pracovat s rozsáhlými časovými řadami bez výrazného zpomalení při interaktivních operacích, jako je zoom nebo posun v grafu. „Podpora real-time“ označuje schopnost průběžně aktualizovat data bez nutnosti kompletního překreslení grafu.

Tabulka 1: Porovnání vizualizačních knihoven, část 1

	Apache ECharts	**LightningChart JS**	**SciChart**	**Plotly.js**
Výkon	Vysoký (i pro big data)	Extrémní (miliony bodů)	Velmi vysoký (WebGL)	Střední (optimalizace SVG)
Licence	Open-source (Apache 2.0)	Komerční (Proprietární)	Komerční (Proprietární)	MIT (Open-source)
Cena	Zdarma (také komerčně)	Vysoká (~$3200+)	Vyšší (~$108/měsíc)	Zdarma / Placený support
Integrace do React	Výborná integrace i Real Time	Špičková podpora Real Time	Špičková podpora Real Time	Dobrá integrace
Výhody	Skvělý design „z krabice“, rozsáhlá galerie grafů.	Nejrychlejší vykreslování na trhu, GPU akcelerace.	Vysoká stabilita při RT datech, silná technická podpora.	Excelentní pro vědecké, 3D a statistické grafy.
Nevýhody	Rozsáhlá konfigurace vyžaduje delší učení.	Velmi vysoká cena, uzavřený ekosystém.	Nutnost placené licence i pro menší projekty.	Slabší výkon u velkých časových řad (SVG limit).

Zdroj: Vlastní zpracování

Tabulka 2: Porovnání vizualizačních knihoven, část 2

	Dygraphs	**D3.js**	**Highcharts**	**Chart.js**
Výkon	Vysoký (pro časové řady)	Plně v rukou vývojáře	Střední (SVG/Canvas)	Omezený (vhodné pro malá data)
Licence	MIT (Open-source)	BSD (Open-source)	Komerční (Placená)	MIT (Open-source)
Cena	Zdarma	Zdarma	~$590 (dle licence)	Zdarma
React / RT	Dobrá / Velmi rychlá	Vyžaduje vlastní „bridge“	Velmi dobrá	Výborná (snadný start)
Výhody	Bezkonkurenční v rychlosti zoomu časových os.	Neomezená kreativita, totální kontrola nad vizuálem.	Extrémně intuitivní API a špičková dokumentace.	Nejrychlejší implementace, moderní a čistý vzhled.
Nevýhody	Zastaralý vzhled, omezené typy grafů.	Extrémně strmá křivka učení, nemá hotové grafy.	Pro komerční sféru poměrně drahá licence.	Nevhodná pro real-time a miliony datových bodů.

Zdroj: Vlastní zpracování

Výsledky srovnání ukazují, že se jednotlivá řešení výrazně liší nejen výkonem, ale také způsobem vykreslování, možnostmi konfigurace a náročností integrace. Některé knihovny vynikají v dílčích oblastech, například LightningChart nabízí velmi vysoký výkon a Plotly širokou škálu typů grafů. Žádná z analyzovaných knihoven však neposkytla vyváženou kombinaci parametrů jako Apache ECharts.

Klíčovým kritériem pro výběr knihovny byla schopnost pracovat s velkými časovými řadami bez výrazného poklesu výkonu. Ukázalo se, že některé knihovny nedokáží při větších datových objemech udržet plynulé vykreslování, zejména při zoomování a posunu v grafu (např. Chart.js, D3.js, Highcharts). Naopak řešení využívající pokročilejší optimalizační techniky nebo GPU akceleraci dosahují v těchto scénářích lepších výsledků. ECharts v tomto ohledu poskytl stabilní výkon i při práci se statisícovými datovými sadami, a to bez nutnosti složité konfigurace.

Dalším důležitým aspektem byla podpora průběžné aktualizace dat. Elektrochemická měření generují nové datové body v krátkých intervalech, což klade nároky na plynulé vykreslování grafu. Některé knihovny (např. Plotly nebo Dygraphs) sice umožňují práci v reálném čase, ale při vyšší frekvenci aktualizací může docházet ke zpomalení nebo k nutnosti opakovaného překreslování grafu. ECharts oproti tomu umožňuje průběžné přidávání nových dat bez nutnosti překreslit celý graf, což snižuje výpočetní náročnost a umožňuje plynulé zobrazení i při dlouhodobém měření.

Významnou roli hrála také integrace do Reactu. Knihovny jako LightningChart nebo SciChart sice nabízejí vysoký výkon, ale jejich integrace je složitější a vyžaduje více vlastního kódu. ECharts naopak disponuje dostupnými wrappery, přehledným API a dobře zdokumentovaným způsobem konfigurace grafů, což zjednodušuje implementaci i následnou údržbu.

Licenční podmínky byly dalším faktorem, který ovlivnil výběr. Komerční knihovny s vysokými licenčními poplatky nejsou vhodné pro dlouhodobé použití v rámci vyvíjeného systému. ECharts díky licenci Apache 2.0 umožňuje akademické i komerční použití bez omezení, což z něj činí vhodné řešení i z hlediska budoucího rozvoje aplikace.

Na základě uvedených kritérií byla knihovna Apache ECharts vybrána jako nejvhodnější řešení. V porovnání s ostatními knihovnami nabízí vyvážený poměr mezi výkonem, flexibilitou, náročností implementace a dlouhodobou udržitelností.

1.4 Webové technologie pro vývoj aplikace

Vizualizace dat z elektrochemických měření klade vysoké nároky na použité technologie. Jak bylo uvedeno v předchozí kapitole, výsledná data mohou být velmi rozsáhlá a často je nutné je zobrazovat i průběžně během měření. Aplikace proto musí být schopna pracovat s dlouhými časovými řadami, efektivně aktualizovat zobrazení a zároveň poskytovat přehledné a flexibilní uživatelské rozhraní.

Tato kapitola představuje technologie použité při implementaci aplikace a důvody jejich výběru. TypeScript byl zvolen pro zvýšení přehlednosti a spolehlivosti kódu, React pro komponentový návrh uživatelského rozhraní a knihovna Apache ECharts pro vizualizaci dat. Pro tvorbu uživatelského rozhraní byly dále využity komponenty knihovny shadcn/ui.

Zvolená kombinace technologií umožňuje vytvořit aplikaci, která je schopná efektivně zobrazovat rozsáhlá data a zároveň poskytuje dostatečnou flexibilitu pro další rozvoj.

1.4.1 TypeScript

TypeScript je nadstavba jazyka JavaScript, která rozšiřuje jeho možnosti o statické typování. Použití TypeScriptu přináší několik výhod zejména ve vývojových projektech, kde je kladen důraz na stabilitu a udržovatelnost kódu (Microsoft, 2022).

Hlavní přínosy TypeScriptu:

- statické typování umožňuje odhalit chyby již při kompilaci, nikoli až za běhu,
- lepší dokumentace kódu díky typovým anotacím a rozhraním,
- snazší udržovatelnost při práci v týmu,
- kompatibilita s JavaScriptem, což umožňuje použití rozsáhlého ekosystému knihoven,
- podpora moderních jazykových konstrukcí.

V kontextu bakalářské práce TypeScript významně zjednodušil tvorbu datových struktur pro měření, práci s API a integraci grafových komponent. Umožnil také přehledné definování typů měřicích kanálů a časových řad, což usnadnilo jejich vizualizaci i další zpracování.

1.4.2 React

React je knihovna pro tvorbu uživatelských rozhraní založená na komponentovém přístupu. Její výhodou je schopnost efektivně aktualizovat uživatelské rozhraní díky využití virtuálního DOM (Meta, 2023).

Pro projekt byl React vhodnou volbou zejména díky:

- komponentové architektuře,
- virtuálnímu DOM, který zvyšuje výkon při častých změnách dat v reálném čase,
- silné komunitě a ekosystému,
- kompatibilitě s TypeScriptem,
- vhodnosti pro interaktivní vizualizace a řízení stavů.

Při vizualizaci elektrochemických dat je důležité, aby aplikace dokázala efektivně aktualizovat pouze ty části uživatelského rozhraní, které se mění. React poskytuje rychlou odezvu i při častém překreslování grafů, což je u real-time měření klíčové.

1.4.3 Apache ECharts

Apache ECharts je knihovna pro datovou vizualizaci, která je v projektu využita pro vykreslování časových řad, impedančních diagramů (Nyquist, Bode), vícekanálových měření i dalších datových struktur. Jedná se o open-source nástroj vyvíjený organizací Apache Software Foundation (Apache, 2025).

Při výběru vizualizační knihovny bylo zohledněno několik faktorů, zejména schopnost pracovat s velkým množstvím dat, podpora interaktivních funkcí a možnost integrace do webové aplikace.

Přestože moderní vizualizační nástroje mohou využívat GPU akceleraci pomocí WebGL, v této práci je využit výchozí Canvas renderer knihovny ECharts.

Canvas poskytuje dostatečný výkon pro zpracovávaná data a zároveň zajišťuje vysokou kompatibilitu napříč zařízeními, včetně mobilních. Vzhledem k tomu, že aplikace může být používána v různých prostředích, nebylo možné předem garantovat podporu pokročilejších technologií, jako je WebGL.

Z těchto důvodů byl ECharts vybrán jako vhodné řešení pro implementaci vizualizační části aplikace, zejména kvůli:

- schopnosti pracovat s velkým množstvím dat,
- bezplatné open-source licence Apache 2.0 vhodná i pro komerční použití,
- široké možnosti konfigurace grafů,
- podpoře interaktivních funkcí (zoom, posun, výběr, tooltipy),
- možnosti synchronizace více grafů,
- integraci do Reactu pomocí dostupných wrapperů.

Při práci s rozsáhlými datovými řadami byla využita také metoda vzorkování dat (sampling), konkrétně algoritmus *Largest Triangle Three Buckets* (LTTB), který umožňuje snížit počet zobrazovaných bodů při zachování tvaru signálu. Tento přístup zlepšuje výkon vykreslování a zároveň zachovává čitelnost grafu.

Pro vykreslování grafů je využit výchozí Canvas renderer. Ten zajišťuje vysokou kompatibilitu napříč prohlížeči a zařízeními a poskytuje dostatečný výkon pro běžné vizualizační úlohy. Oproti tomu technologie WebGL je závislá na podpoře grafického hardwaru a může vykazovat odlišné chování v různých prostředích. Canvas je vhodnější volbou pro aplikace, u nichž nelze předem určit cílové zařízení, a kde je kladen důraz na stabilitu a dostupnost.

Ve srovnání s dalšími knihovnami poskytl ECharts nejlepší poměr výkonu, flexibility a licenčních podmínek. Jeho schopnost pracovat s průběžně aktualizovanými daty byla zásadní při implementaci vizualizace v reálném čase.

1.4.4 shadcn/ui

Shadcn/ui je komponentová knihovna pro React, která staví na frameworku TailwindCSS. TailwindCSS je systém pro tvorbu vzhledu webových aplikací, který využívá jednoduché utility-třídy místo tradičních ručně psaných CSS stylů (Tailwind Labs, 2024). Shadcn/ui nad tímto základem poskytuje předpřipravené komponenty, jako jsou tlačítka, formuláře, tabulky nebo modální okna, které mají jednotný moderní vzhled a dají se snadno graficky upravovat (Shadcn, 2024).

Na rozdíl od běžných UI knihoven, jako je například Material UI, nejsou komponenty shadcn/ui dodávány jako uzavřená knihovna. Generují se přímo do projektu jako plnohodnotný zdrojový kód. Vývojář může každou komponentu přímo upravit podle potřeby – měnit její strukturu, vzhled i chování, bez nutnosti obcházet omezení knihovny. To poskytuje mnohem větší kontrolu nad výsledným rozhraním a umožňuje přesně přizpůsobit UI specifickým požadavkům aplikace (Shadcn, 2024).

Tento přístup je vhodný zejména pro aplikace, u nichž je důležitá dlouhodobá udržitelnost, snadná modifikace a možnost přizpůsobení rozhraní konkrétním požadavkům, jako je aplikace pro vizualizaci elektrochemických měření. Mezi hlavní výhody patří:

- minimální overhead – komponenty se generují do projektu přímo jako zdrojový kód,
- vysoká možnost úprav vzhledu,
- výborná integrace s TailwindCSS,
- vizuálně konzistentní a snadno upravitelné rozhraní,
- nízká závislost na externích knihovnách.

Výhodou je možnost snadno upravovat komponenty přímo v kódu, což je vhodné pro dlouhodobě rozvíjené firemní aplikace.

1.4.5 Celkové zhodnocení zvoleného technologického řešení

Zvolená kombinace technologií TypeScript, React, Apache ECharts a shadcn/ui vytváří vhodný základ pro vývoj webové aplikace určené k vizualizaci elektrochemických měření. Jednotlivé technologie řeší různé části aplikace a společně umožňují vytvořit stabilní a dobře rozšiřitelné řešení.

TypeScript přispívá k vyšší spolehlivosti kódu díky statickému typování a usnadňuje práci s datovými strukturami. React umožňuje návrh aplikace pomocí komponent, což zjednodušuje organizaci kódu a podporuje opakované použití jednotlivých částí uživatelského rozhraní.

Knihovna Apache ECharts zajišťuje vizualizaci dat a umožňuje efektivně pracovat s rozsáhlými datovými řadami i jejich průběžnou aktualizací. V kombinaci s výchozím Canvas rendererem poskytuje dostatečný výkon a zároveň zajišťuje dobrou kompatibilitu napříč zařízeními.

Komponenty knihovny shadcn/ui přispívají k jednotnému vzhledu aplikace a umožňují snadnou úpravu uživatelského rozhraní podle konkrétních požadavků.

Navržené technologické řešení tak umožňuje spolehlivé zpracování a vizualizaci dat v reálném čase a zároveň poskytuje flexibilní základ pro další rozvoj aplikace.

2 Praktická část

Praktická část se zaměřuje na návrh a implementaci nové webové aplikace určené pro řízení a vizualizaci elektrochemických měření v systémech Kolibrik.net. Aplikace Artemis vznikla jako náhrada původního softwaru, který již nevyhovoval požadavkům na moderní uživatelské rozhraní, práci s rozsáhlými daty a vizualizaci v reálném čase. Následující kapitoly popisují architekturu řešení, použitou technologii, návrh uživatelského rozhraní a implementaci klíčových funkcí, včetně vizualizace dat.

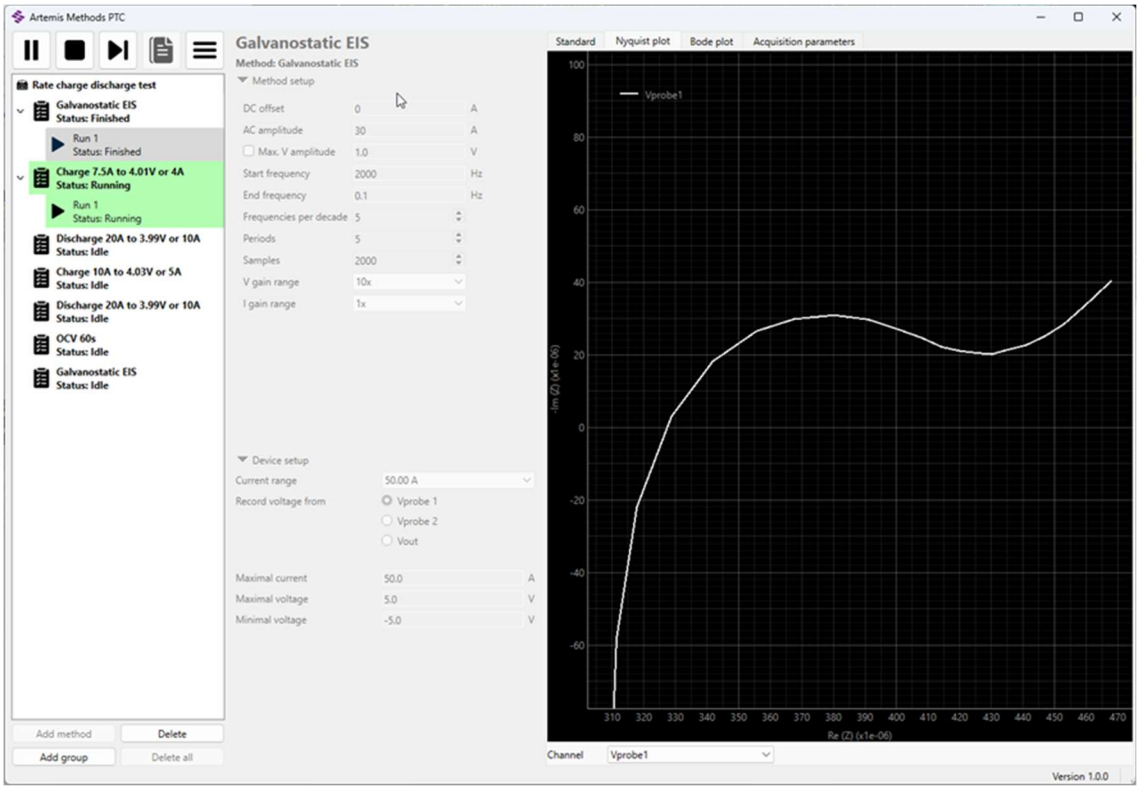
2.1 Analýza a návrh řešení

Kapitola se zaměřuje na analýzu původního softwarového řešení používaného v systémech Kolibrik.net a na identifikaci jeho omezení z hlediska výkonu, uživatelského rozhraní a práce s daty. Na základě této analýzy je následně navržen koncept nové aplikace, která tato omezení odstraňuje a lépe odpovídá požadavkům na moderní webové technologie a vizualizaci dat v reálném čase.

2.1.1 Analýza stávající aplikace

Původní desktopová aplikace společnosti Kolibrik.net slouží jako hlavní nástroj pro řízení měřicích zařízení a práci s daty, která tato zařízení generují. V praxi se používá při testování bateriových článků, palivových článků nebo jiných elektrochemických systémů, kde je potřeba přesně řídit proudové a napěťové profily a zároveň sledovat odezvu měřeného vzorku (Kolibrik.net, 2026). Aplikace je dlouhodobě využívána v praxi a její rozhraní odráží postupný vývoj funkcí i požadavků uživatelů.

Rozhraní lze rozdělit do tří hlavních částí, jak je patrné na obrázku 1. V levém panelu se nachází seznam dostupných měřicích metod a jednotlivých kroků experimentu. Tento panel funguje jako navigace a umožňuje rychle přepínat mezi různými částmi měření. Uživatel zde vidí, které kroky jsou připravené, které probíhají a které již byly dokončeny.



Obrázek 1: Původní aplikace

Zdroj: Kolibrik.net, 2026

Střední část rozhraní slouží k nastavování parametrů vybrané metody. Každá metoda má vlastní sadu vstupních polí, přepínačů a voleb, které určují, jak bude měření probíhat, například počáteční hodnoty signálu, limity měření nebo počet cyklů. Jedná se o klasický konfigurační formulář, který se dynamicky mění podle toho, jaký typ měření je zvolen. Rozhraní je však poměrně statické a jednotlivé formuláře jsou pevně zabudované do aplikace, což komplikuje jejich úpravy nebo rozšiřování.

Pravá část aplikace je určena pro zobrazení dat. Podle typu měření se zde vykreslují různé grafy – někdy časové průběhy, jindy dvourozměrné grafy nebo jiné vizualizace. Grafy se aktualizují v reálném čase podle toho, jak zařízení posílá data.

Přestože aplikace plní svou funkci, postupem času se začaly projevovat limity její architektury. Jde o desktopové řešení s pevně daným rozložením prvků, které není snadné upravovat ani přizpůsobovat. Jednotlivé části rozhraní se mezi sebou liší nejen vzhledem, ale i způsobem práce s daty. To komplikuje údržbu i přidávání nových funkcí.

Dalším problémem je výkon při práci s většími objemy dat. Původní vizualizační modul byl optimalizován pro menší objemy dat, avšak při práci s rozsáhlými datovými sadami začalo docházet k omezením ve výkonu grafického zobrazení. Uživatel tak může mít problém například s přibližováním grafu nebo s plynulým posunem v datech.

K omezením patří také menší flexibilita uživatelského rozhraní. Komponenty jsou pevně zabudované do aplikace a jejich úprava vyžaduje zásah do kódu na více místech. To znesnadňuje přizpůsobení aplikace specifickým požadavkům zákazníků nebo integraci nových typů vizualizací.

Tyto faktory vedly k rozhodnutí vytvořit nové řešení založené na moderní webové architektuře. Cílem je nabídnout flexibilnější uživatelské rozhraní, lepší práci s velkými objemy dat a jednodušší rozšiřitelnost do budoucna

2.1.2 Analýza vstupních dat

Vstupními daty jsou údaje z elektrochemických měření generovaná měřicími systémy Kolibrik.net, které jsou zaznamenávána během experimentu s vysokou vzorkovací frekvencí. Datové soubory mohou obsahovat velké množství záznamů. Základním typem dat jsou časové řady, kde každý datový bod představuje časovou značku a jednu nebo více měřených hodnot, například napětí, proud nebo další doplňkové veličiny. Typicky jde o několik paralelních signálů, například napětí na různých sondách (Vout, Vprobe1, Vprobe2), proud, další doplňkové hodnoty poskytované hardwarem (Kolibrik.net, 2026).

Z hlediska softwarového návrhu mají všechna data podobnou základní strukturu. Jedná se sekvence numerických hodnot, které jsou zobrazovány pomocí grafických vizualizací. Rozdíl mezi jednotlivými typy měření spočívá především v interpretaci os grafu a ve způsobu, jakým jsou jednotlivé datové body vykreslovány.

Jedná se o dvojice nebo n-tice čísel, které je potřeba vykreslit v grafu. Rozdíl spočívá pouze v tom, jaké hodnoty se použijí na osách. Bez ohledu na typ měření mají všechna vstupní data několik společných charakteristik:

- jsou to sekvence číselných hodnot, často velmi dlouhé,
- mohou obsahovat více paralelních kanálů,
- je potřeba je zobrazovat v reálném čase,
- musí být možné s nimi interaktivně pracovat (přiblížení, posun, filtrování),
- mohou mít různé rozsahy hodnot a různé typy os.

Mnoho experimentů generuje více datových řad najednou. Aplikace musí být schopna:

- zobrazit více signálů v jednom grafu,
- umožnit jejich zapínání a vypínání,
- pracovat s různými rozsahy hodnot.

Kromě časových řad se v aplikaci mohou vyskytovat i další typy dat, například data organizovaná podle jiné nezávislé proměnné (např. frekvence). Z pohledu vizualizace však mají tato data podobnou strukturu – jedná se o dvojice nebo n-tice numerických hodnot, které lze zobrazit pomocí grafu. Data nejsou organizována primárně podle času, ale podle frekvence signálu. Každý záznam obsahuje frekvenci měření a odpovídající hodnoty impedance, které jsou následně zobrazovány pomocí specializovaných grafů k analýze vztahů mezi jednotlivými veličinami v závislosti na frekvenci měření. Původní rozhraní to řeší pomocí jednoduché legendy a barevného rozlišení křivek. U rozsáhlejších dat však tato forma přestává být přehledná, což je jeden z důvodů, proč je potřeba modernější vizualizační komponenta.

Z hlediska návrhu aplikace je tedy klíčové vytvořit vizualizační komponentu, která dokáže pracovat s velkými objemy dat a zároveň je zobrazovat přehledně a jednotným způsobem.

2.1.3 Požadavky na novou aplikaci

Požadavky na novou aplikaci vycházely z analýzy původního desktopového řešení a z charakteru dat, se kterými uživatelé pracují. Nešlo o jednorázově definované zadání, požadavky se postupně zpřesňovaly během vývoje, při konzultacích s vývojovým týmem i při diskusích nad konkrétními problémy, které se objevovaly při používání staré aplikace. Iterativní přístup vývoje umožnil postupně zpřesňovat požadavky a přizpůsobovat návrh aplikace aktuálním potřebám.

Jedním z klíčových požadavků byla schopnost efektivně pracovat s velkým množstvím dat. Měřicí zařízení generují dlouhé datové řady s vysokou vzorkovací frekvencí, takže i krátký experiment může obsahovat statisíce až miliony záznamů. Původní desktopová aplikace měla při takovém objemu dat problémy s výkonem. Načítání grafů se zpomalovalo, interakce nebyla plynulá a při delších experimentech bylo obtížné s daty pracovat. Nová aplikace proto musí umožnit plynulé vykreslování rozsáhlých datasetů, rychlou odezvu při přibližování nebo posunu v grafu a efektivní práci s pamětí.

Dalším požadavkem byla podpora průběžně přicházejících dat. Měření mohou trvat od několika minut až po celé dny a aplikace musí být schopna zobrazovat data, která se postupně doplňují. Z toho důsledků musí aplikace umět efektivně přidávat nové body a zároveň zachovat možnost pracovat s historickými daty. Tento požadavek je zásadní zejména v prostředí webového prohlížeče, kde je výkon přirozeně omezenější než u desktopových aplikací.

Velký důraz je kladen také na interaktivitu vizualizace. Uživatelé potřebují s grafy aktivně pracovat, přibližovat konkrétní úseky, posouvat se v datech, skrývat nebo zobrazovat jednotlivé datové kanály a rychle se orientovat v tom, co se během měření děje. Vizualizace tedy nemůže být statická a zobrazovat jen aktuální stav. Je to nástroj, který uživateli pomáhá pochopit průběh měření a rychle reagovat na případné odchylky.

Významným požadavkem byla i flexibilita uživatelského rozhraní. Původní desktopová aplikace obsahovala pevně definované komponenty, které bylo obtížné upravovat nebo rozšiřovat. Nové řešení má být navrženo tak, aby jednotlivé části rozhraní bylo možné snadno měnit, doplňovat nebo přizpůsobovat konkrétním potřebám. To zahrnuje možnost přidávat nové typy grafů, nové panely nebo nové způsoby konfigurace měření bez zásahu do jádra aplikace.

Zásadní změnou oproti původnímu řešení je přechod na webovou architekturu. Původní aplikace byla „all in one“ a běžela pouze na jednom počítači, který musel být fyzicky připojen k měřicímu zařízení. Nová aplikace komunikuje s backendem přes API, což umožňuje vzdálený přístup a práci z více zařízení současně. Uživatel si může nastavit port, přes který bude aplikace dostupná, a sledovat průběh měření například z notebooku, tabletu nebo mobilního telefonu. Tím se výrazně zvyšuje komfort práce a odpadá nutnost být fyzicky u měřicího zařízení.

S webovým řešením souvisí i požadavek na responzivní uživatelské rozhraní. Aplikace musí být použitelná na různých velikostech obrazovky a přizpůsobovat rozložení panelů dostupnému prostoru. To je důležité zejména při vzdáleném přístupu, kde uživatelé často pracují na různých typech zařízení.

Na základě těchto požadavků byla navržena nová aplikace zaměřená na vizualizaci dat, která umožňuje efektivní práci s rozsáhlými datovými řadami, podporuje průběžné zobrazování dat během měření, nabízí moderní interaktivní grafy a poskytuje flexibilní prostředí pro další rozvoj.

Díky webové architektuře navíc umožňuje uživatelům vzdáleně sledovat průběh experimentu a pracovat s daty odkudkoli.

2.2 Implementace aplikace

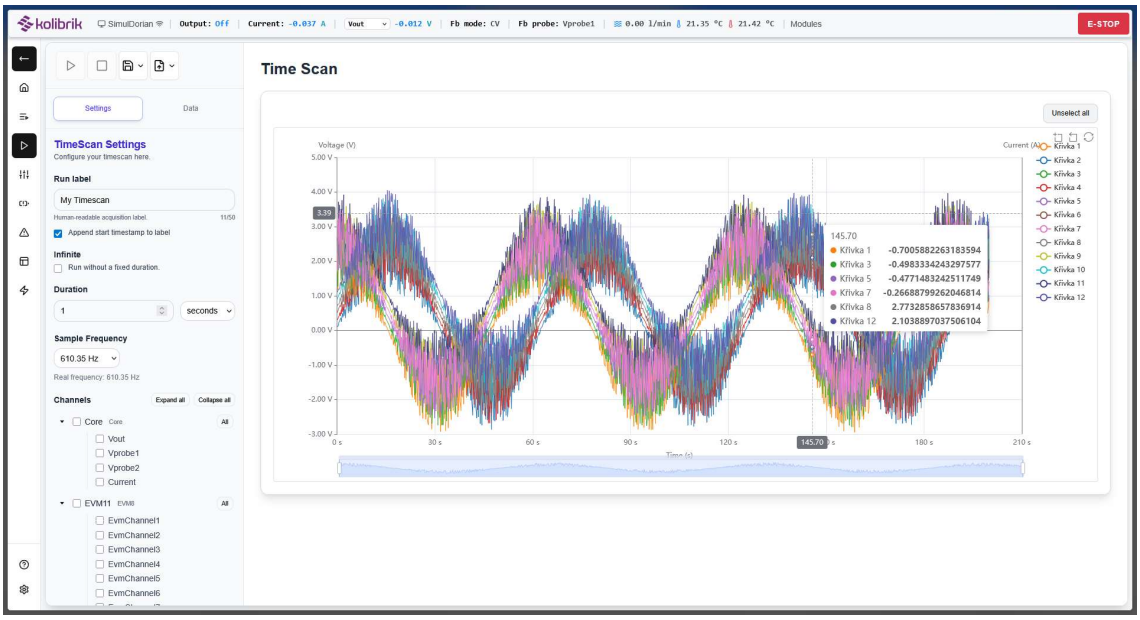
2.2.1 Architektura aplikace

Bakalářská práce se zaměřuje pouze na frontendovou část systému, tedy na vizualizaci dat. Backendová část, která zajišťuje komunikaci s měřicím zařízením a poskytuje data prostřednictvím API, není součástí řešení. Aplikace pracuje s daty získanými z existujícího rozhraní systému společnosti Kolibrik.

Aplikace je realizována jako webová klientská aplikace, což přináší několik výhod oproti původnímu desktopovému řešení. Webová architektura umožňuje oddělit uživatelské rozhraní od fyzického zařízení a zpřístupnit aplikaci vzdáleně. Uživatel tak není vázán na jeden počítač, ale může sledovat průběh měření z libovolného zařízení, které má přístup k síti. Tento přístup zároveň usnadňuje aktualizace a údržbu, protože není nutné distribuovat nové verze aplikace na jednotlivé počítače.

Z hlediska implementace je aplikace navržena jako komponentový systém. Jednotlivé části rozhraní jsou rozděleny do samostatných částí, které spolu komunikují prostřednictvím jasně definovaných rozhraní. Tento způsob návrhu přináší řadu výhod. Umožňuje například jednodušší identifikaci a izolaci chyb, protože problém lze obvykle omezit na konkrétní komponentu. Zároveň podporuje flexibilitu při budoucím rozšiřování systému. Nové funkce lze přidávat formou dalších komponent, aniž by bylo nutné zasahovat do stávajícího jádra aplikace. Komponenty lze navíc snadno znovu použít nebo rozšířit, což je výhodné při postupném doplňování funkcí.

Uživatelské rozhraní (obrázek 2) je tvořeno několika hlavními celky, které odpovídají způsobu práce s aplikací. Základ tvoří panel se stavem měření, který zobrazuje aktuální hodnoty a informace o průběhu experimentu. Další částí je komponenta pro vizualizaci dat, která zajišťuje vykreslování grafů a práci s rozsáhlými datovými řadami. Součástí rozhraní jsou také konfigurační panely, které umožňují nastavovat parametry měření nebo vybírat datové kanály.



Obrázek 2: Time Scan

Zdroj: vlastní zpracování

Klíčovou částí celé architektury je vizualizační komponenta dat, která zajišťuje vykreslování grafů obsahujících rozsáhlé datové řady a zároveň poskytuje interaktivní prvky, jako je přiblížení, posun nebo výběr jednotlivých signálů. Pro implementaci byla zvolena knihovna Apache ECharts, která nabízí dobrý výkon při práci s rozsáhlými datovými sadami a zároveň poskytuje široké možnosti konfigurace grafů. Díky tomu je možné zobrazovat různé typy dat jednotným způsobem, bez nutnosti vytvářet specializované grafické moduly pro jednotlivé měřicí režimy.

Navržená architektura vytváří flexibilní základ pro další rozvoj aplikace. Komponentový přístup umožňuje postupně doplňovat nové funkce, upravovat vzhled rozhraní nebo přidávat nové typy vizualizací, aniž by bylo nutné zasahovat do jádra aplikace. Webová architektura zároveň umožňuje vzdálený přístup a snadnější integraci s dalšími systémy. Celkově tak nové řešení odstraňuje limity původní desktopové aplikace a poskytuje moderní prostředí pro práci s daty.

2.2.2 Použité technologie

Při implementaci aplikace byly využity technologie popsané v teoretické části. Následující text shrnuje praktické využití jednotlivých technologií a jejich přínos pro výslednou podobu aplikace.

React tvoří základ uživatelského rozhraní. Komponentový přístup umožnil rozdělit aplikaci na menší části, které lze samostatně vyvíjet a udržovat. V praxi to znamená, že dashboard, grafová komponenta, konfigurační panely nebo tabulkové výpisy existují jako oddělené celky, které mezi sebou sdílejí pouze nezbytná data. React byl využit také pro navigaci mezi jednotlivými částmi aplikace. Routing umožňuje načítat pouze ty komponenty, které jsou v daný moment potřeba, což zlepšuje odezvu rozhraní a snižuje zátěž při práci s většími daty. Oproti původní desktopové aplikaci, kde se při změně pohledu načítalo celé rozhraní znovu, je tento přístup výrazně efektivnější. Ukázka části kódu je na následujícím obrázku 3.

```
24  const routes = [
25    {
26      path: "/",
27      element: <App />,
28      errorElement: <ErrorPage />,
29      children: [
30        {
31          index: true,
32          element: <EmptyPage title="Select devices" />,
33        },
34        {
35          path: "select-machine",
36          element: <EmptyPage title="Select devices" />,
37        },
38        {
39          path: "mega-eis/machines/:machineId",
40          children: [
41            {
42              path: "dashboard",
43              element: <MeDashboard />,
44            },
45            {
46              path: "experiments",
47              element: <PtcExperiments />,
48            },
49            {
50              path: "time-scan",
51              element: <MeTimeScan />,
52            },
```

Obrázek 3: Příklad částí kódu z react router

Zdroj: Vlastní zpracování

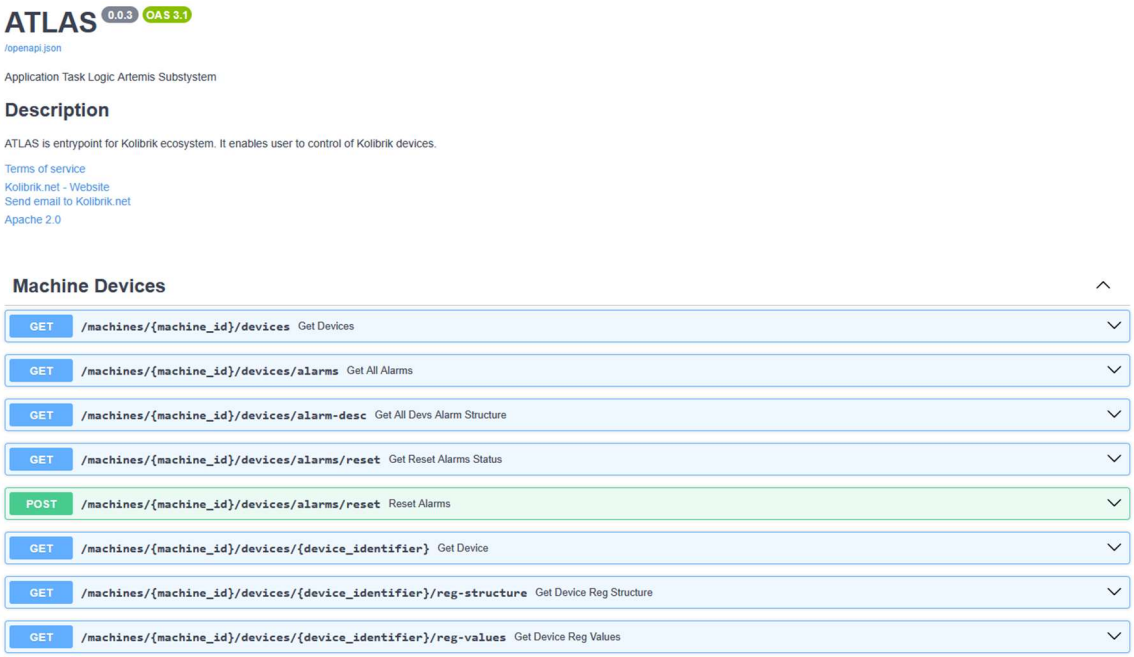
TypeScript byl použit pro implementaci celé aplikace, zejména pro nastavení prostředí pro práci s daty. Měřicí zařízení poskytuje různé typy datových struktur, které je potřeba správně interpretovat a zobrazovat.

Pro vizualizaci dat byla použita knihovna Apache ECharts. V teoretické části je popsán důvod její volby, v implementaci se ukázala jako vhodná zejména díky možnosti pracovat s více datovými řadami současně a díky podpoře interaktivních funkcí. Grafová komponenta využívá ECharts pro vykreslování časových průběhů, dvourozměrných grafů i vícekanálových měření. Knihovna umožňuje přibližování, posun v grafu, skrývání jednotlivých kanálů a další interakce, které jsou při práci s rozsáhlými daty nezbytné. Konfigurace grafů je dynamická – mění se podle toho, jaký typ dat aplikace zobrazuje.

2.2.3 Komunikace s API

Frontendová aplikace získává veškerá data prostřednictvím REST API poskytovaného měřicím systémem Kolibrik. API představuje jediný komunikační kanál mezi uživatelským rozhraním

a měřicím hardwarem. Aplikace nepracuje přímo s fyzickým zařízením, ale pouze s daty, která backend zpřístupňuje ve formátu JSON. Tento přístup umožňuje provozovat aplikaci na libovolném zařízení v síti a zpřístupnit ji vzdáleně prostřednictvím portu, který si uživatel nastaví. Na obrázku 4 je ukázka výstupu z API interního systému společnosti Kolibrik.net.

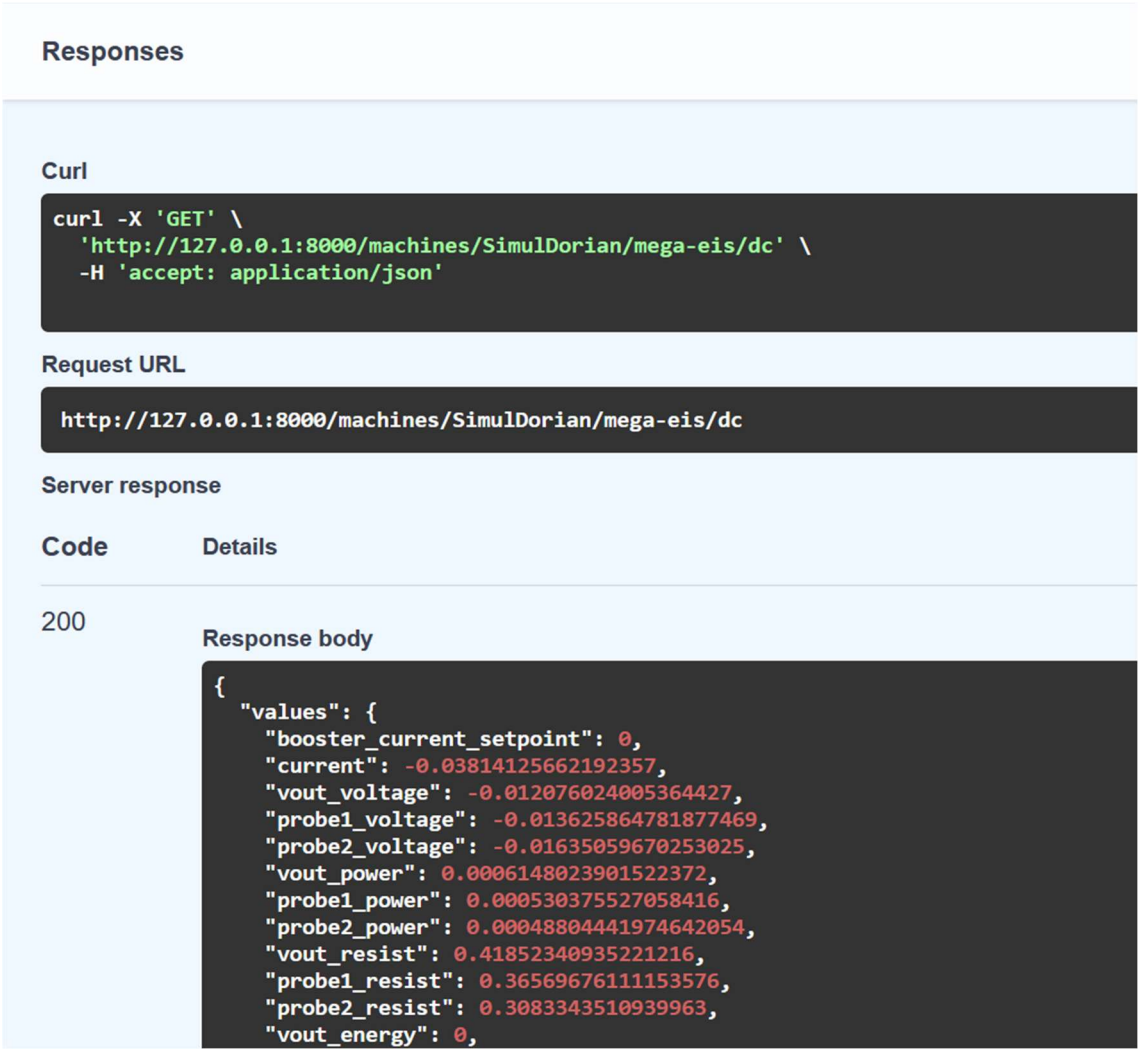


Obrázek 4: Rest API

Zdroj: interní systém společnosti Kolibrik.net (neveřejné)

Komunikace probíhá pomocí opakovaných HTTP požadavků. Data jsou načítána v pravidelných intervalech, jejichž délka se liší podle typu požadavku. Pro rychle se měnící hodnoty se používá interval 1 s, pro méně dynamické části rozhraní 2 s nebo 5 s. Tento způsob periodického dotazování (polling) nahrazuje streamování dat a umožňuje průběžně aktualizovat stav aplikace bez nutnosti udržovat trvalé spojení.

API poskytuje data ve formě strukturovaných objektů, které obsahují aktuální hodnoty měřených kanálů i doplňující informace o stavu zařízení (ukázka na obrázku 5). Typickým příkladem je endpoint /mega-eis/dc, který vrací objekt s hodnotami napětí, proudu, výkonu nebo energie. Aplikace tato data ukládá do stavového systému a postupně je doplňuje o nové záznamy. Díky tomu je možné vytvářet časové řady, které se následně zobrazují ve vizualizační komponentě.



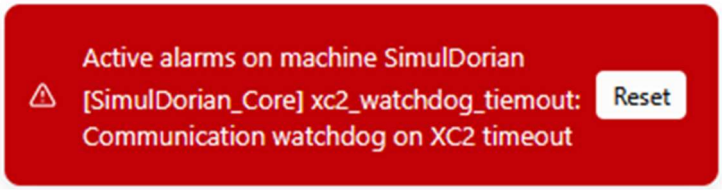
Obrázek 5: API data

Zdroj: interní systém společnosti Kolibrik.net (neveřejné)

Při práci s API je kladen důraz na minimalizaci zbytečných překreslení uživatelského rozhraní. Aplikace využívá knihovnu TanStack Query, která zajišťuje automatické obnovování dat, jejich ukládání do mezipaměti a správu stavu dříve načtených hodnot. Nové datové body jsou přidávány přímo do existujících struktur, aniž by bylo nutné znovu načítat celý dataset. Tento přístup je vhodný zejména pro dlouhá měření, u nichž počet záznamů postupně narůstá.

Součástí komunikace je také načítání metadat, například seznamu dostupných zařízení, jejich identifikátorů nebo alarmových stavů. Tato metadata se načítají při inicializaci aplikace a určují, jaké části rozhraní se zobrazí. Díky tomu je možné dynamicky přizpůsobit strukturu aplikace podle toho, jaké zařízení je připojeno a jaké informace poskytuje.

Aplikace Artemis zpracovává i chybové stavy, které mohou nastat při komunikaci s API. Pokud server vrátí chybu nebo není dostupný, zobrazí se uživateli odpovídající informace (ukázka na obrázku 6) a aplikace se pokusí o opětovné připojení. Tento mechanismus je důležitý zejména při vzdáleném přístupu, kde může docházet k dočasným výpadkům spojení.



Obrázek 6: Ukázka chybové hlášky
Zdroj: Vlastní zpracování

Komunikace s API tak tvoří základ celé aplikace. Umožňuje průběžné načítání dat, jejich ukládání a následnou vizualizaci. Díky periodickému dotazování, jednotnému zpracování odpovědí a využití stavového systému je možné zobrazovat data v reálném čase a zároveň zachovat plynulou odezvu uživatelského rozhraní i při dlouhodobém měření.

2.2.4 Implementace vizualizace dat

Vizualizace dat tvoří jednu z klíčových částí aplikace, protože uživatelům umožňuje sledovat průběh měření a pracovat s daty v reálném čase. Při návrhu vizualizačního modulu bylo nutné řešit nejen samotné vykreslování grafů, ale také výkon, práci s rozsáhlými datovými řadami a zachování interaktivity při průběžných aktualizacích.

Aplikace využívá knihovnu Apache ECharts, která umožňuje vykreslování velkého množství datových bodů a nabízí široké možnosti konfigurace. Vstupní data jsou předávána ve formě dvojic hodnot [x, y], kde x představuje nezávislou proměnnou (typicky čas) a y hodnotu měřeného kanálu. Pokud měření obsahuje více kanálů, graf je tvořen několika datovými řadami vykreslovanými současně. ECharts umožňuje definovat i více os, což je užitečné v situacích, kdy mají jednotlivé kanály výrazně odlišné rozsahy hodnot.

Konfigurace grafu je založena na objektu, který určuje uspořádání a chování jednotlivých prvků vizualizace. Tento objekt obsahuje zejména:

- osy grafu (xAxis, yAxis),
- datové řady (series),
- legendu umožňující zapínat a vypínat jednotlivé křivky,
- interaktivní prvky (dataZoom) pro přiblížení a posun v grafu.

Konfigurace zahrnuje také možnost exportu grafu jako obrázku, což je užitečné při dokumentaci měření. Interaktivní funkce jsou dostupné přímo v rámci knihovny. Uživatel může graf přibližovat kolečkem myši, výběrem oblasti nebo pomocí ovládacích prvků. ECharts podporuje i postupné vracení přiblížení zpět a reset do výchozího stavu.

Při implementaci bylo nutné řešit výkon při práci s rozsáhlými datovými soubory. Pokud se do grafu předá příliš mnoho bodů najednou, dochází ke zpomalení vykreslování a zhoršení odezvy uživatelského rozhraní. Proto byly použity optimalizační techniky, které umožňují zachovat přesnost grafu a zároveň snížit zátěž na prohlížeč. Jednou z těchto technik je omezení počtu vykreslovaných bodů. Pokud například datová řada obsahuje několik tisíc bodů, aplikace zobrazí pouze každý druhý nebo třetí bod, aniž by to mělo vliv na interpretaci výsledků. Při přiblížení grafu se však vykreslí všechny body, aby byla zachována maximální přesnost. Knihovna ECharts navíc nabízí vlastní optimalizační mechanismy, jako je sampling, které dále snižují zátěž při vykreslování.

Další optimalizací je zjednodušení grafických prvků. Pokud se v grafu nachází mnoho bodů, není nutné vykreslovat každý bod jako samostatný symbol. ECharts umožňuje vypnout vykreslování symbolů a zobrazovat pouze spojnice, což výrazně zlepšuje výkon. Symboly se zobrazují pouze tehdy, když uživatel najede myší na konkrétní místo v grafu.

Aplikace podporuje také vizualizaci dat v reálném čase. Nové datové body jsou přidávány do existujících datových řad a graf je aktualizován pouze v rozsahu změn. Tento přístup zabraňuje tomu, aby se graf při každé aktualizaci překresloval celý, což by vedlo ke ztrátě interaktivity. Při nevhodné implementaci docházelo k resetování grafu, ztrátě přiblížení nebo vypnutí vybraných datových řad. Tyto problémy se objevily zejména při práci s živými daty, kdy se graf aktualizoval příliš agresivně.

Řešením bylo oddělit aktualizaci dat od aktualizace konfigurace grafu. Při přidání nových dat se mění pouze datové řady, zatímco ostatní části konfigurace – například aktuální zoom, výběr kanálů nebo pozice v grafu – zůstávají zachovány. Tím se zabránilo resetování grafu a uživatel může pokračovat v interakci i během průběžné aktualizace dat. Aktualizace grafu je navržena tak, aby byla co nejméně invazivní a nezasahovala do probíhajících uživatelských akcí, například výběru oblasti pro přiblížení.

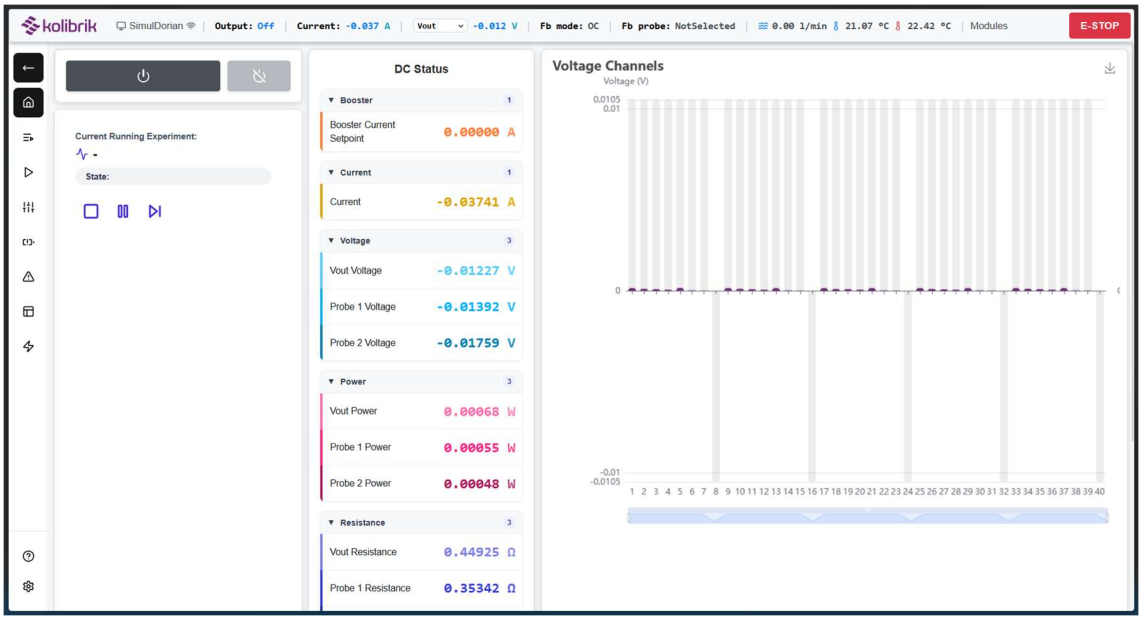
Implementace vizualizace dat tak kombinuje několik přístupů: optimalizaci počtu vykreslovaných bodů, postupnou aktualizaci datových řad, zachování stavu grafu a využití výkonných funkcí knihovny ECharts. Výsledkem je řešení, které umožňuje plynulé zobrazení rozsáhlých datových souborů i jejich průběžnou aktualizaci v reálném čase, aniž by docházelo ke ztrátě interaktivity nebo zpomalení aplikace.

2.2.5 Struktura aplikace

Aplikace je navržena jako komponentově orientovaný frontend, kde je uživatelské rozhraní rozděleno do samostatných částí. Každá část je implementována jako nezávislá komponenta s jasně definovanou odpovědností. Tento přístup umožňuje oddělit logiku jednotlivých částí systému, usnadňuje údržbu kódu a umožňuje opakované použití komponent v různých částech aplikace. Komponentová architektura zároveň podporuje postupné rozšiřování aplikace bez nutnosti zásahů do jejího jádra.

Základní strukturu aplikace tvoří několik hlavních komponent, které společně zajišťují zobrazování dat, interakci s uživatelem a konfiguraci systému.

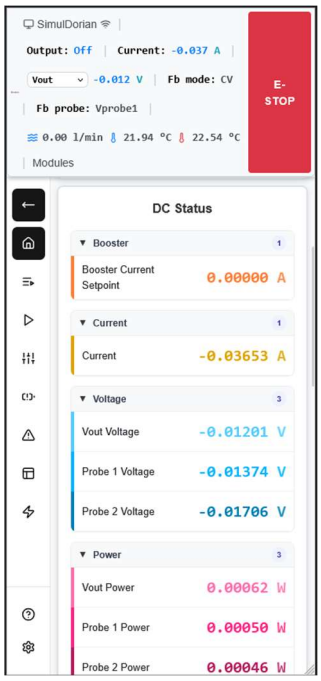
Dashboard představuje výchozí pracovní obrazovku aplikace (obrázek 7). Uživatel zde získává rychlý přehled o aktuálním stavu měření a základních parametrech experimentu. Součástí dashboardu jsou grafické indikátory zobrazující vybrané veličiny, například napětí, proud nebo stav výstupu. Tyto informace jsou aktualizovány v pravidelných intervalech a poskytují okamžitý přehled bez nutnosti otevírat detailní grafy.



Obrázek 7: Dashboard

Zdroj: Vlastní zpracování

Dashboard je navržen tak, aby byl přehledný i na menších zařízeních. Rozložení prvků se přizpůsobuje velikosti obrazovky, což umožňuje pohodlné použití aplikace i na mobilních telefonech nebo tabletech. Mobilní verze aplikace je zjednodušená a neobsahuje grafovou komponentu. Důvodem je omezený prostor na displeji a také výkonová náročnost vizualizace rozsáhlých datových řad. Ukázka rozhraní je na následujícím obrázku 8.

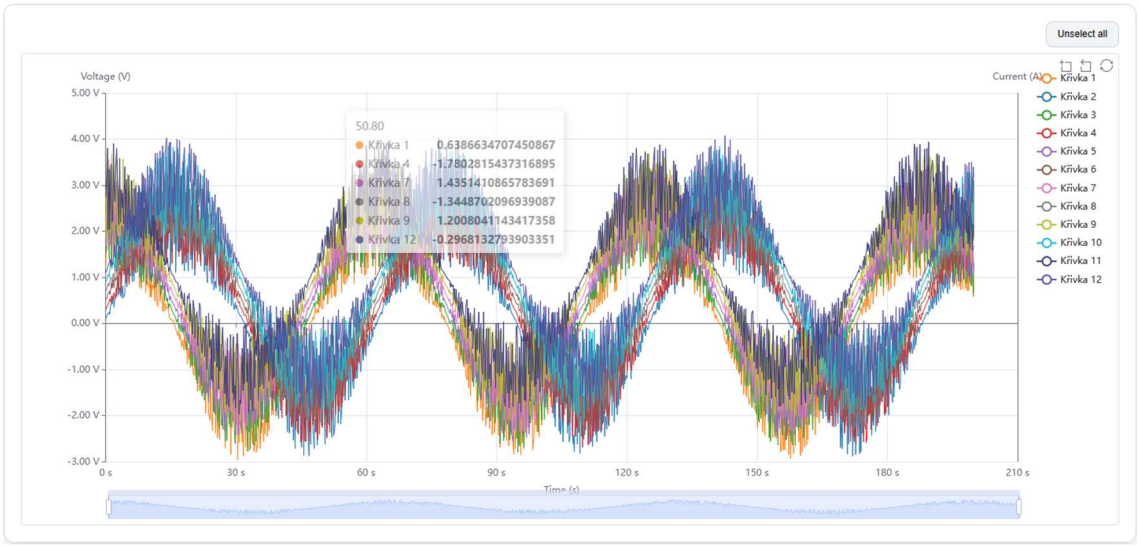


Obrázek 8: Mobilní náhled aplikace

Zdroj: Vlastní zpracování

Nejdůležitější částí aplikace je komponenta pro vizualizaci dat (obrázek 9). Tato komponenta zajišťuje vykreslování grafů obsahujících datové řady získané z měření a umožňuje zobrazovat

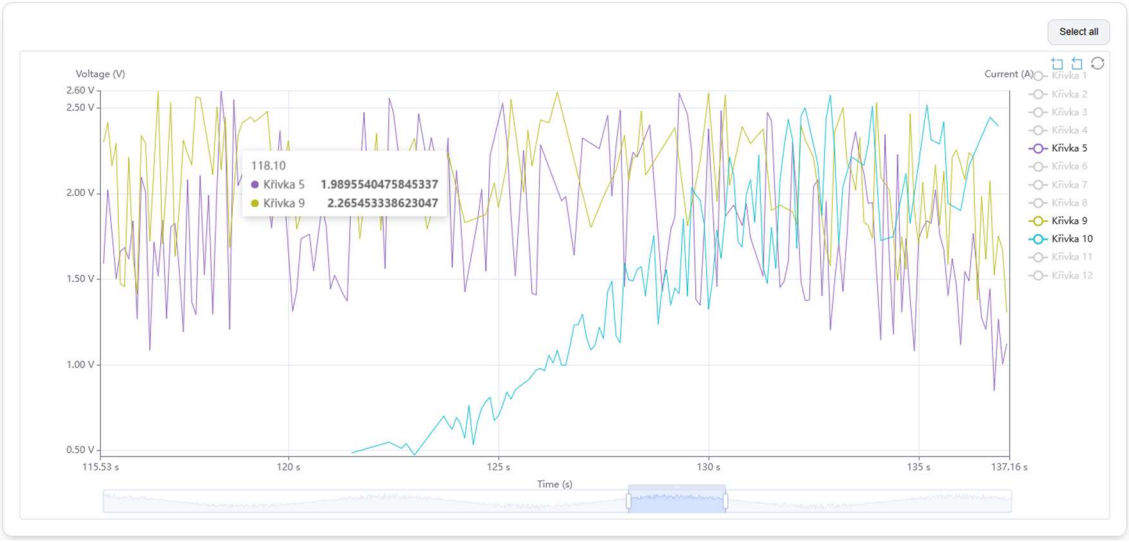
více kanálů současně. Graf poskytuje interaktivní funkce, jako je přiblížení, posun nebo zapnutí a vypnutí jednotlivých křivek pomocí legendy.



Obrázek 9: Komponenta Grafu

Zdroj: Vlastní zpracování

Komponenta využívá knihovnu Apache ECharts, která umožňuje efektivní vykreslování velkého množství datových bodů a nabízí široké možnosti konfigurace. Graf je definován pomocí konfiguračního objektu, který určuje strukturu a chování jednotlivých prvků vizualizace, například osy, datové řady, legendu nebo interaktivní prvky typu dataZoom. Díky tomu je možné přizpůsobit vzhled i chování grafu podle typu zobrazovaných dat. Na obrázku 10 je zobrazen příklad přiblížení.

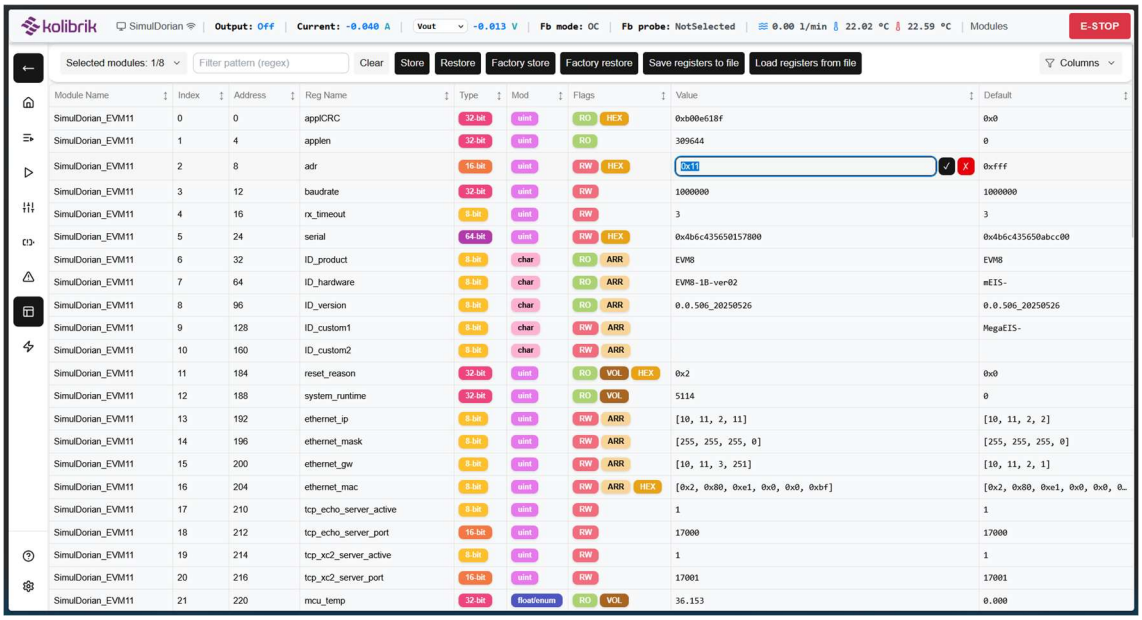


Obrázek 10: Komponenta Grafu – Přiblížení

Zdroj: Vlastní zpracování

Komponenta grafu je navržena tak, aby byla nezávislá na ostatních částech aplikace. Přijímá pouze datové řady a metadata, což umožňuje její opakované použití v různých částech systému.

Další část aplikace tvoří konfigurační panely (obrázek 11), které umožňují nastavovat parametry systému a pracovat s konfiguračními registry zařízení. Tyto panely poskytují uživateli možnost upravovat nastavení zařízení, měnit provozní režimy nebo zobrazovat diagnostické informace. Slouží jako rozhraní mezi uživatelem a hardwarem a umožňují nejen zobrazovat aktuální hodnoty, ale také je přímo upravovat.



Obrázek 11: Reg View

Zdroj: Vlastní zpracování

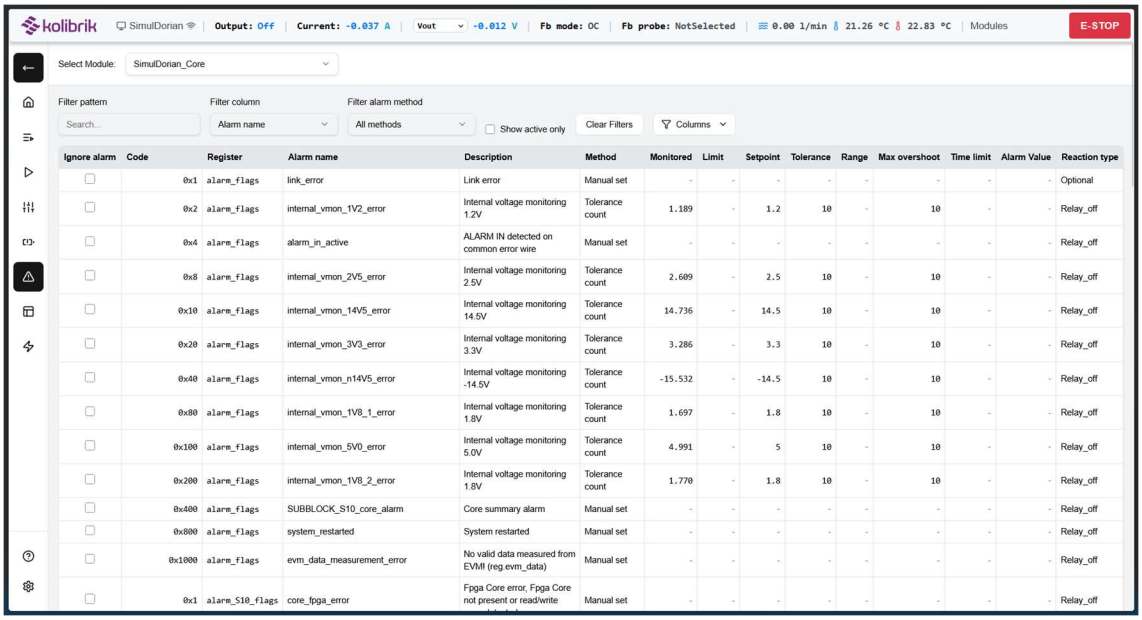
Na obrázku 11 je zobrazen tzv. „Reg View“, tedy přehled registrů zařízení. Každý řádek reprezentuje jeden registr, který obsahuje informace o konkrétním parametru systému, například komunikačním nastavení, identifikaci zařízení nebo provozních hodnotách. Tabulka obsahuje kromě názvu registru také jeho adresu, datový typ, přístupová práva (pro čtení nebo čtení i zápis) a aktuální hodnotu.

U registrů s povoleným zápisem má uživatel možnost hodnotu přímo upravit. Editace probíhá prostřednictvím vstupního pole, které se aktivuje po výběru konkrétního registru. Po zadání nové hodnoty je změna odeslána přes API do zařízení, kde dojde k aktualizaci konfigurace.

Kromě samotné editace má komponenta další funkce, jako je filtrování a vyhledávání registrů, načítání a ukládání konfigurace, zobrazení výchozích hodnot, indikace přístupových práv (read-only, read-write).

Mezi další konfigurační části patří například:

- diagnostické panely (obrázek 12),
- zobrazení alarmů a stavových hlášení,
- komponenty pro sledování aktuálního stavu zařízení.



Ignore alarm	Code	Register	Alarm name	Description	Method	Monitored	Limit	Setpoint	Tolerance	Range	Max overshoot	Time limit	Alarm Value	Reaction type
☐	0x1	alarm_flags	link_error	Link error	Manual set	-	-	-	-	-	-	-	-	Optional
☐	0x2	alarm_flags	internal_vmon_1V2_error	Internal voltage monitoring 1.2V	Tolerance count	1.189	-	1.2	10	-	10	-	-	Relay_off
☐	0x4	alarm_flags	alarm_in_active	ALARM IN detected on common error wire	Manual set	-	-	-	-	-	-	-	-	Relay_off
☐	0x8	alarm_flags	internal_vmon_2V5_error	Internal voltage monitoring 2.5V	Tolerance count	2.609	-	2.5	10	-	10	-	-	Relay_off
☐	0x10	alarm_flags	internal_vmon_14V5_error	Internal voltage monitoring 14.5V	Tolerance count	14.736	-	14.5	10	-	10	-	-	Relay_off
☐	0x20	alarm_flags	internal_vmon_3V3_error	Internal voltage monitoring 3.3V	Tolerance count	3.286	-	3.3	10	-	10	-	-	Relay_off
☐	0x40	alarm_flags	internal_vmon_n14V5_error	Internal voltage monitoring -14.5V	Tolerance count	-15.532	-	-14.5	10	-	10	-	-	Relay_off
☐	0x80	alarm_flags	internal_vmon_1V8_1_error	Internal voltage monitoring 1.8V	Tolerance count	1.697	-	1.8	10	-	10	-	-	Relay_off
☐	0x100	alarm_flags	internal_vmon_5V0_error	Internal voltage monitoring 5.0V	Tolerance count	4.991	-	5	10	-	10	-	-	Relay_off
☐	0x200	alarm_flags	internal_vmon_1V8_2_error	Internal voltage monitoring 1.8V	Tolerance count	1.770	-	1.8	10	-	10	-	-	Relay_off
☐	0x400	alarm_flags	SUBBLOCK_S10_core_alarm	Core summary alarm	Manual set	-	-	-	-	-	-	-	-	Relay_off
☐	0x800	alarm_flags	system_restarted	System restarted	Manual set	-	-	-	-	-	-	-	-	Relay_off
☐	0x1000	alarm_flags	evm_data_measurement_error	No valid data measured from EVMI (reg.evm_data)	Manual set	-	-	-	-	-	-	-	-	Relay_off
☐	0x1	alarm_S10_flags	core_fpga_error	Fpga Core error, Fpga Core not present or read/write	Manual set	-	-	-	-	-	-	-	-	Relay_off

Obrázek 12: Diagnostická tabulka – přehled alarmů

Zdroj: Vlastní zpracování

Součástí konfiguračních komponent jsou také diagnostické panely, které slouží ke sledování stavu systému a vyhodnocování chybových stavů. Tyto panely zobrazují seznam alarmů a diagnostických hlášení, které jsou generovány na základě aktuálního stavu zařízení.

Na obrázku 12 je zobrazen přehled alarmů, kde každý řádek odpovídá jednomu sledovanému stavu. Pro jednotlivé alarmy jsou uvedeny informace, jako je název, popis, metoda vyhodnocení, aktuální hodnota, nastavený limit a tolerance. Systém tak umožňuje nejen sledovat aktuální stav, ale také definovat podmínky, za kterých dojde k vyvolání alarmu. Uživatel může parametry upravovat, například měnit limity nebo toleranci sledovaných veličin. Tím lze přizpůsobit chování systému konkrétnímu typu měření nebo provozním podmínkám.

Diagnostické panely zároveň umožňují filtrování a vyhledávání jednotlivých alarmů, což usnadňuje orientaci v systému při větším počtu sledovaných parametrů. V případě překročení nastavených mezí může systém reagovat definovaným způsobem, například vypnutím relé.

Komponenty představují důležitou část aplikace, která umožňuje nejen vizualizaci dat, ale i kontrolu správného chodu zařízení a včasnou detekci chyb.

Celá aplikace je navržena modulárně. Jednotlivé komponenty spolu komunikují prostřednictvím sdíleného stavu aplikace, který zajišťuje konzistentní předávání dat mezi dashboardem, grafy a konfiguračními panely. Tento přístup umožňuje snadné rozšiřování aplikace a přidávání nových funkcí bez zásahu do stávající struktury. Pokud je například potřeba změnit způsob vizualizace dat nebo přidat nový typ grafu, lze tyto změny provést izolovaně, aniž by bylo nutné upravovat ostatní části systému.

2.3 Testování aplikace

Testování aplikace probíhalo průběžně během celého vývoje a jeho cílem bylo ověřit správnou funkčnost jednotlivých částí systému, stabilitu vizualizace a chování aplikace při práci s rozsáhlými datovými soubory i s daty přicházejícími v reálném čase. Pro účely testování byl

vytvořen simulovaný dataset obsahující dvanáct datových řad (přímek), z nichž každá měla přibližně padesát tisíc bodů. Celkově tak aplikace pracovala s více než šesti sty tisíci hodnotami, přičemž byla ověřena i funkčnost při zátěži blížící se jednomu milionu bodů. Data byla generována pomocí jednoduché lineární funkce, která vytvářela nový bod každou sekundu, což umožnilo simulovat dlouhodobé měření.

V rámci testování byla nejprve ověřena základní funkčnost aplikace. Bylo potřeba potvrdit, že data jsou správně načítána z API, zobrazují se v grafu a jednotlivé komponenty mezi sebou bez problémů spolupracují. Testovaly se běžné scénáře, které odpovídají reálnému používání aplikace: načtení dat z API a jejich zobrazení v grafu, vykreslení více datových řad současně, přepínání jejich viditelnosti pomocí legendy, funkčnost přiblížení a posunu v grafu a také správné zobrazování hodnot na dashboardu a v konfiguračních panelech. Tato část testování potvrdila, že aplikace reaguje konzistentně a jednotlivé komponenty fungují tak, jak bylo navrženo.

Zásadní část testování byla zaměřena na výkon při práci s velkým množstvím dat. Při prvních pokusech se ukázalo, že přímé vykreslení všech bodů vede ke zpomalení aplikace a v některých případech i k nestabilnímu chování. Interakce s grafem, zejména přiblížení a posun, byla při velkém množství dat znatelně pomalejší. Na základě těchto zjištění byla upravena implementace vizualizace tak, aby se při velkém oddálení zobrazoval pouze omezený počet bodů, zatímco při přiblížení se vykreslí všechny hodnoty pro zachování přesnosti. Současně byly využity optimalizační mechanismy knihovny ECharts, například sampling nebo vypnutí symbolů jednotlivých bodů. Po těchto úpravách byla aplikace schopna pracovat i s velmi rozsáhlými datovými sadami bez výrazného dopadu na výkon.

Aplikace byla testována také při práci s daty přicházejícími v reálném čase. V této oblasti bylo důležité ověřit, že nové datové body lze do grafu přidávat průběžně, aniž by docházelo k resetování grafu nebo ztrátě uživatelského nastavení. Při prvních testech se ukázalo, že nevhodná aktualizace dat vedla k tomu, že se graf po každém doplnění dat vrátil do výchozího stavu, což znamenalo ztrátu přiblížení i výběru aktivních datových řad. Tento problém byl odstraněn úpravou logiky aktualizace tak, aby se měnily pouze samotné datové řady, zatímco ostatní části konfigurace grafu zůstaly zachovány. Díky tomu bylo možné zachovat interaktivitu i při častých aktualizacích a graf se choval stabilně i při dlouhodobém měření.

Testování jako celek potvrdilo, že aplikace splňuje požadavky na vizualizaci dat, práci s rozsáhlými datovými soubory i zobrazení dat v reálném čase. Identifikované problémy vedly k úpravám implementace, které výrazně zlepšily výkon, stabilitu a uživatelskou přívětivost. Aplikace je díky tomu schopna pracovat s velkým množstvím dat a poskytuje plynulou a interaktivní vizualizaci i při dlouhodobém provozu.

Závěr

Cílem této bakalářské práce bylo navrhnout a implementovat webovou aplikaci pro správu a vizualizaci dat z elektrochemických měřicích přístrojů, která umožňuje zobrazování naměřených hodnot v reálném čase a poskytuje uživateli nástroje pro interaktivní práci s měřením. Součástí práce bylo také vyhodnocení dostupných vizualizačních knihoven, výběr vhodného řešení pro práci s rozsáhlými datovými sadami a jeho integrace do softwaru používaného společností Kolibrik.net.

Teoretická část práce se zaměřila na charakter dat vznikajících při elektrochemických měřeních a na požadavky, které tyto datové soubory kladou na jejich zpracování a vizualizaci. Byly identifikovány hlavní technické výzvy, jako je práce s dlouhými časovými řadami, vysoká frekvence aktualizací a potřeba plynulé interakce v prostředí webové aplikace. Na základě porovnání dostupných vizualizačních knihoven bylo vybráno řešení Apache ECharts, které nejlépe splnilo požadavky na výkon, flexibilitu, licenční podmínky i integraci do moderního webového stacku.

Praktická část práce se věnovala návrhu a implementaci webové aplikace postavené na technologiích TypeScript, React a Apache ECharts. Výsledná aplikace umožňuje načítání, správu a vizualizaci dat z různých typů elektrochemických měření, podporuje real-time přenos dat z backendu a nabízí interaktivní práci s grafy včetně zoomování, posunu, synchronizace více zobrazení a správy měřicích kanálů. Implementace potvrdila vhodnost zvolených technologií a ukázala, že je možné dosáhnout stabilního výkonu i při práci s rozsáhlými datovými sadami.

Přínosem bakalářské práce je ucelený přehled problematiky vizualizace elektrochemických měření a formulace technologických požadavků nezbytných při návrhu webových aplikací zaměřených na zpracování a prezentaci elektrochemických dat. Systematicky jsou shrnuty klíčové výzvy spojené se zpracováním rozsáhlých časových řad a identifikovány vhodné přístupy k jejich řešení, včetně využití akcelerovaného vykreslování a interaktivních nástrojů.

Výsledkem práce je funkční webová aplikace, která splňuje stanovené cíle a představuje použitelný základ pro další rozvoj v rámci softwaru Kolibrik.net. Aplikaci je možné dále rozšířit například o pokročilejší analytické nástroje, export dat, správu měřicích kampaní nebo hlubší integraci s dalšími moduly systému. Práce tak přináší nejen teoretický přehled problematiky vizualizace elektrochemických dat, ale i prakticky ověřené řešení připravené pro reálné nasazení.

Seznam použité literatury

APACHE SOFTWARE FOUNDATION. *Apache License, Version 2.0* [online]. Forest Hill (MD): Apache Software Foundation, 2004 [cit. 2026-04-04]. Dostupné z: https://www.apache.org/licenses/LICENSE-2.0

APACHE SOFTWARE FOUNDATION. *Apache ECharts Documentation*. [online]. 2025 [cit. 2025-01-20]. Dostupné z: https://echarts.apache.org/

ARCTION LTD. *LightningChart® JS – High-Performance JavaScript Charting Library*. [online]. 2024 [cit. 2025-01-20]. Dostupné z: https://www.arction.com/lightningchart-js/

BOKEH TEAM. *BokehJS Documentation*. [online]. 2024 [cit. 2025-01-20]. Dostupné z: https://docs.bokeh.org/

BOSTOCK, Mike. *D3.js Documentation*. [online]. 2024 [cit. 2025-01-20]. Dostupné z: https://d3js.org/

BOUZEK, Karel. *Elektrochemické inženýrství*. Praha: Vydavatelství VŠCHT, 1999, s. 119. ISBN 80-7080-368-1. Dostupné také z: https://ceskadigitalniknihovna.cz/uuid/uuid:d1145923-2632-11e5-a988-001b63bd97ba

CHART.JS TEAM. *Chart.js Documentation*. [online]. 2024 [cit. 2025-01-20]. Dostupné z: https://www.chartjs.org/

DYGRAPH. *Dygraphs: Open Source JavaScript Charting Library*. [online]. 2020 [cit. 2025-01-20]. Dostupné z: http://dygraphs.com/

HIGHCHARTS AS. *Highcharts JS API Reference*. [online]. 2024 [cit. 2025-01-20]. Dostupné z: https://www.highcharts.com/

Horák, T. Graph Performance: Rendering Large Graphs Efficiently. 2018 [cit. 2025-01-20]. Dostupné z: https://imld.de/cnt/uploads/Horak-2018-Graph-Performance.pdf

KOLIBRIK.NET. *Artemis – Kolibrik Measurement Software* [online]. [2026]. [cit. 2026-04-10]. Dostupné z: https://www.kolibrik.net/en/products/potentiostat-galvanostat/artemis-kolibrik-measurement-software

KOLIBRIK.NET. *Battery Cycler – What It's Used For*. [online]. 2025a [cit. 2025-01-20]. Dostupné z: https://www.kolibrik.net/en/products/battery-cycler

KOLIBRIK.NET. *MegaEIS System for Batteries*. [online]. 2025b [cit. 2025-01-20]. Dostupné z: https://www.kolibrik.net/en/solutions-products/megaeis-system-for-batteries

LIU, Y., ZHANG, Q., LI, H. et al. State-of-health estimation of lithium-ion batteries based on electrochemical impedance spectroscopy. *Protection and Control of Modern Power Systems*. 2023, 8:28. DOI: 10.1186/s41601-023-00314-w.

META PLATFORMS. *React Documentation*. [online]. 2023 [cit. 2025-01-20]. Dostupné z: https://react.dev/

MICROSOFT. *TypeScript Handbook*. [online]. 2022 [cit. 2025-01-20]. Dostupné z: https://www.typescriptlang.org/docs/

MIT. *MIT License* [online]. Cambridge (MA): Massachusetts Institute of Technology, 1988 [cit. 2026-04-04]. Dostupné z: https://opensource.org/licenses/MIT

ORTIGOSSA, E. S., DIAS, F., NASCIMENTO, D. a NONATO, L. G. *Time Series Information Visualization – A Review of Approaches and Tools*. [online]. arXiv:2507.14920. 2025 [cit. 2025-01-20]. Dostupné z: https://arxiv.org/abs/2507.14920

PLOTLY TECHNOLOGIES. *Plotly.js – Open-Source JavaScript Graphing Library*. [online]. 2024 [cit. 2025-01-20]. Dostupné z: https://plotly.com/javascript/

SCHÜTZ, M., HERZBERGER, L. a WIMMER, M. SimLOD: Simultaneous LOD Generation and Rendering. *arXiv:2310.03567*. 2023.

SCI CHART LTD. *SciChart JS Documentation*. [online]. 2024 [cit. 2025-01-20]. Dostupné z: https://www.scichart.com/javascript-chart-documentation/

SHADCN. *shadcn/ui Documentation*. [online]. 2024 [cit. 2025-01-20]. Dostupné z: https://ui.shadcn.com/

ŠIMEK, Lubomír a HRNČIŘÍK, Josef. *Fyzikální chemie I*. 2. vyd. Brno: VUTIUM, 1997. ISBN 80-214-0950-9.

TAILWIND LABS. *TailwindCSS Documentation*. [online]. 2024 [cit. 2025-01-20]. Dostupné z: https://tailwindcss.com/docs

VEGA PROJECT. *Vega Visualization Grammar*. [online]. 2024 [cit. 2025-01-20]. Dostupné z: https://vega.github.io/vega/

ZHANG, Y., TANG, Q., ZHANG, Y., WANG, J., STIMMING, U. a LEE, A. A. Identifying degradation patterns of lithium-ion batteries from impedance spectroscopy using machine learning. *Nature Communications*. 2020, 11, 1706. DOI: 10.1038/s41467-020-15235-7.