TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

INTELIGENTNÍ MĚŘICÍ PRACOVIŠTĚ SE SYSTÉMEM DÁLKOVÉ SPRÁVY A ZPRACOVÁNÍ DAT

Diplomová práce

Studijní program: N2612 – Elektrotechnika a informatika
Studijní obor: 1802T007 – Informační technologie

Autor práce: **Bc. Lukáš Sieber**
Vedoucí práce: Ing. Petr Pfeifer

Liberec 2015

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

AN INTELLIGENT MEASUREMENT WORKPLACE WITH REMOTE CONTROL AND DATA PROCESSING FRAMEWORK

Diploma thesis

Study programme: N2612 – Electrical Engineering and Informatics
Study branch: 1802T007 – Information Technology

Author: **Bc. Lukáš Sieber**
Supervisor: Ing. Petr Pfeifer

Liberec 2015

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: **2014/2015**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Lukáš Sieber**
Osobní číslo:	**M12000223**
Studijní program:	**N2612 Elektrotechnika a informatika**
Studijní obor:	**Informační technologie**
Název tématu:	**Inteligentní měřicí pracoviště se systémem dálkové správy a zpracování dat**
Zadávající katedra:	**Ústav informačních technologií a elektroniky**

Zásady pro vypracování:

1. Nastudujte systémy a metody dálkového ovládání laboratorních přístrojů a zpracování dat.
2. Navrhněte a realizujte struktury Server/Klient a systém, které umožní lokální i dálkovou správu přístrojů a měřicích úloh přístrojů v učebně AP9 a SCPI-kompatibilních přístrojů pomocí HTTP/HTTPS.
3. Implementujte rovněž podporu přístrojů s velmi velkou hloubkou záznamu.
4. Pro ověření správné funkce systému rovněž navrhněte a realizujte jednoduchou úlohu analýzy sériového kanálu, případně jednoduché digitální filtrace na této Vámi navržené platformě.
5. Na vyvinuté platformě navrhněte a realizujte systém a aplikaci pro tvorbu šablon úloh do cvičení vč. ukládání výsledků a zpracování protokolů.
6. Vytvořte vzorovou úlohu do laboratorních cvičení.

Rozsah grafických prací: **Dle potřeby dokumentace**

Rozsah pracovní zprávy: **cca 40 až 50 stran**

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

[1] **Dokumentace k osciloskopům GW Instek série GDS2000**

[2] **Dokumentace k osciloskopu Agilent série 9000**

[3] **Dokument Inteligentní učebna a měřicí pracoviště, TUL, ESF 2012/2013**

[4] **Minimalizované webové servery (řešení MS, Borland, nginx, apod), a PHP**

[5] **Architektury Klient/Server**

[6] **Microsoft Visual C++ (PRATA, Stephen. C++ primer plus. 5th ed. Indianapolis: SAMS, 2005, 1202 s. ISBN 0-672-32697-3, VIRIUS, Miroslav. Pasti a propasti jazyka C++. 2. aktualiz. a rozš. vyd. Brno: CP Books, 2005, 375 s. ISBN 80-251-0509-1).**

Vedoucí diplomové práce: **Ing. Petr Pfeifer**
Ústav informačních technologií a elektroniky

Datum zadání diplomové práce: **12. září 2014**

Termín odevzdání diplomové práce: **15. května 2015**

V. Kopecký

prof. Ing. Václav Kopecký, CSc.
děkan

Technická univerzita v Liberci - 14 -

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
vedoucí ústavu

V Liberci dne 12. září 2014

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu TUL.

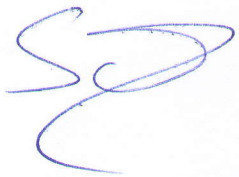
Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Diplomovou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum: 15.5.2015

Podpis:



Poděkování

Na tomto místě bych rád poděkoval vedoucímu mé diplomové práce, Ing. Petru Pfeiferovi, MSc, MBA, Ph.D., za všechny rady, bohaté zkušenosti a poznatky, které mně pomohly k úspěšné realizaci této práce. Dále bych chtěl poděkovat své rodině za její podporu a možnost studovat na vysoké škole. Nakonec všem svým blízkým přátelům, kteří mně při studiu velice pomohli a motivovali.

Abstrakt

Práce se zabývá modernizací výuky a pracoviště v laboratorní učebně AP9. V této učebně jsou moderní počítače a přístroje, se kterými je možné vzdáleně komunikovat. Tato práce má za cíl vytvoření aplikace umožňující komunikaci s přístroji. V úvodu práce byly zjištěny potřebné funkce aplikace a zvoleny technologie, kterými se bude aplikace vyvíjet. Zvolen byl webový server Apache, skriptovací jazyk PHP a databáze MySQL. Dále HTML verze 5, CSS, Javascript a několik pomocných knihoven. Systém server-klient je na straně lokálních PC na studentských pracovištích podporován jednoduchou lokální a univerzální aplikací.

Mezi hlavní funkce aplikace patří správa laboratorních cvičení, která jsou velice variabilní a dovolují vytvoření libovolného cvičení. Mezi další funkce patří správa jednotlivých skupin studentů, správa učeben včetně umístěných počítačů a přístrojů. Dále také správa předmětů a termínů jednotlivých cvičení. Aplikace dokáže komunikovat s přístroji na studentských pracovištích včetně přístrojů s velkou hloubkou záznamu. Výsledkem je komplexní aplikace umožňující moderní výuku na bázi inteligentních pracovišť v učebně AP9.

Klíčová slova

Vzdálená měřicí úloha; inteligentní pracoviště; SCPI kompatibilní přístroje; webová aplikace; učebna AP9

Abstract

This work and document deals with modernization of our classes and intelligent workplace in our laboratory AP9. There are many modern computers and equipment in this room, capable of far remote and intelligent communication. This work aims at creation of a new application for communication with or among the equipment. First, the required technologies were identified. Apache webserver, PHP scripting language and MySQL database, HTML5, CSS, JavaScript and a set of libraries were chosen. The global server-client scheme is locally supported by a small and universal application at the side of each PC present at all student workplaces.

The management of the laboratory classes is one of the main functions of the developed application. There were developed also management of each class, student group, other rooms and equipment functions as well. The developed application also communicates with measuring equipment at student workplaces, including the ones with very deep memories. The result is a very complex and versatile application for modern teaching on the base of intelligent workplaces in AP9 laboratory.

Keywords

Remote measuring task; intelligent workplace; SCPI compatible devices; web application; classroom AP9

Obsah

Seznam obrázků

Seznam ukázek kódu

Seznam tabulek

Seznam zkratek

- API (Application Programming Interface) – rozhraní pro programování aplikací
- ASCII (American Standard Code for Information Interchange) – tabulka definující znaky anglické abecedy a dalších znaků v informatice
- CSS (Cascading Style Sheet) – jazyk pro popis grafického zobrazení prvků na stránce
- DOM (Document Object Model) – objektově orientovaná reprezentace HTML dokumentu
- HTML (HypeText Markup Language) – značkovací jazyk pro tvorbu webových stránek
- HTTP (HyperText Transfer Protocol) – internetový protokol k výměně hypertextových dokumentů
- HTTPS (HypeText Transfer Protocol Secure) – zabezpečený HTTP protokol
- IIS (Internet Information Services) – webový server od společnosti Microsoft
- PDF (Portable Document Format) – přenosný formát dokumentů od firmy Adobe
- PHP (Hypertext PreProcesor) – skriptovací programovací jazyk pro generování dynamických webových stránek
- RGB (Red Green Blu) – červená zelená modrá; aditivní způsob míchání barev
- RLE (Run-Length Encoding) – bezeztrátová komprese kódující posloupnosti stejných hodnot do dvojic počet-hodnota
- SCPI (Standard Command for Programmable Instruments) – standard syntaxe a příkazů k ovládání programovatelných a měřicích přístrojů
- SMTP (Simple Mail Transfer Protocol) – protokol pro přenos elektronické pošty
- SQL (Structured Query Language) – standardizovaný strukturovaný dotazovací jazyk pro práci s relačními databázemi
- USB (Universal Seriál Bus) – univerzální sériová sběrnice; moderní způsob připojení periférií k počítači
- WYSIWYG (What You See Is What You Get) – editor textu, kde je přesně vidět, jak se text ve výsledku zobrazí
- XHTML (eXtensibile HyperText Markup Language) – rozšířitelný značkovací jazyk
- XML (eXtensibile Markup Languagu) – obecný značkovací jazyk

1 Úvod

Dnešní moderní přístroje umožňují díky svým pokročilým funkcím výrazné ulehčení každodenní práce. Jedním z příkladů může být tzv. chytrý mobilní telefon, bez kterého si již skoro nikdo nedokáže představit svůj den. Moderní technologie se začínají objevovat také na školních pracovištích a tato pracoviště se tak musejí vyvíjet spolu s těmito technologiemi.

Cílem této práce je vytvořit aplikaci pro zmodernizování výuky v učebně AP9. V této učebně jsou dostupné nové moderní přístroje, které je možné dálkově ovládat. Současný koncept výuky a samotných cvičení je již letitý až zastaralý, a vzešel tak požadavek na jeho modernizaci. Tato práce tak usnadňuje a zkvalitňuje práci vyučujícího, ale rovněž převede ruční vyplňování laboratorních protokolů do inteligentní digitální aplikace, kterou bude student ovládat na počítači. V aplikaci tak bude dostupné digitalizované každé laboratorní cvičení.

Na začátku práce bude rozebráno zadání a z něj vytvořen seznam důležitých funkcí, které by měla výsledná aplikace implementovat. Mezi tyto funkce patří správa studentů, správa učeben společně s počítači a k nim připojeným přístrojům, správa předmětů a termínů a nakonec samotná správa laboratorních cvičení. Systém bude také podporovat dálkovou správu a konfiguraci přístrojů přes jednoduché rozhraní na počítači vedoucího cvičení. Dále jsou v této práci ukázána pro porovnání některá podobná řešení, která ovšem nesplňují požadovanou koncepci pro učebnu AP9.

Jelikož bude aplikace tvořená jako webová stránka, budou ukázány a diskutovány možné technologie pro vývoj této aplikace, spolu s představením všech přístrojů v učebně. Jako vhodné technologie pro výslednou aplikaci byly zvoleny webový server Apache, skriptovací jazyk PHP a databázový server

MySQL. Dále HTML verze 5, CSS a JavaScript. Bude využito také několik knihoven, které ulehčí vývoj a přinesou komfort v používání dané aplikace. Například šablonovací systém Smarty, JavaScriptová knihovna jQuery, framework Bootstrap a další.

2 Rozbor zadání, požadavky na aplikaci a existující řešení

Zadání práce jasně definuje, co je cílem této diplomové práce. V prvé řadě je nutné se seznámit s možnostmi dálkového ovládání laboratorních přístrojů a zpracování dat. Dále je potřeba navrhnout systém pro ovládání SCPI kompatibilních přístrojů a přístrojů současně používaných v učebně AP9, spolu s přístroji s velkou hloubkou záznamu. Následně realizovat aplikaci pro tvorbu šablon laboratorních cvičení v učebně AP9 s vygenerováním výsledného protokolu daného cvičení. Tuto aplikaci je nutné poté otestovat vzorovou úlohou.

2.1 Požadavky na aplikaci

Dle zadání a jeho rozboru bylo nutné vytvořit seznam funkcí, které musí aplikace splňovat. Důležitou funkcí bude komunikace s přístroji připojenými ke studentským stanicím. Komunikace musí být navržena tak, aby zvládla přenos také velkých dat. Další důležitou funkcí bude vytváření laboratorních cvičení, které budou studenti zpracovávat. Tato cvičení musí být variabilní a dovolit budoucí rozvoj aplikace také na jiné předměty. Je tedy nutné navrhnout vytváření laboratorních cvičení, která budou rozšiřitelná o nové prvky.

Aby se mohl student do aplikace přihlásit, je v systému nutná evidence studentů. O každém studentovi se budou evidovat pouze základní údaje. Aby bylo možné komunikovat s přístroji na studentských stanicích, musí aplikace obsahovat v prvé řadě správu učeben. Následně se do každé učebny zařadí jednotlivé studentské počítače a ke každému všechny připojené SCPI kompatibilní přístroje. U učebny, počítače a přístroje bude nutné uchovávat základní informace potřebné pro běh samotné aplikace.

Aby bylo možné přiřadit cvičení konkrétním studentům, je nutné implementovat do aplikace také správu jednotlivých předmětů společně s jednotlivými termíny cvičení v konkrétní učebně. Díky tomu bude možné studenty zařadit přímo k určitému termínu daného předmětu. Jednotlivá cvičení je poté možné přiřadit konkrétnímu předmětu a tím také všem studentům daného předmětu.

2.2 Podobná nebo existující řešení

V současnosti existuje několik podobných řešení. Většina řešení se ale zaměřuje úzce na jednu oblast a v jednu chvíli je možné měřit pouze danou úlohu. Nejsou tedy variabilní s možností najednou měřit více úloh. V následujících odstavcích bude několik řešení představeno.

Na Matematicko-fyzikální fakultě Univerzity Karlovy v Praze je dostupných několik měřicích úloh. Tyto úlohy jsou vytvořeny pomocí experimentálního systému ISES a softwarové stavebnice iSES Remote Lab SDK [1]. Oboje bylo vytvořeno právě na této fakultě. Pomocí těchto nástrojů byly vytvořeny jednotlivé měřicí úlohy. Každá úloha má striktně danou svou funkci a nelze na ní měřit cokoliv jiného. Měření bylo dostupné skrze webové stránky a byla použita webová kamera pro snímání určitých prvků.

Na Fakultě jaderné a fyzikálně inženýrské Českého vysokého učení technického v Praze bylo dostupné vzdálené měření VA charakteristik LED diod [2]. Měření bylo, stejně jako na Univerzitě Karlově, dostupné na webových stránkách s využitím webkamery a bylo postavené pouze pro tento účel. V průběhu tvorby této práce ale bylo měření nefunkční.

Gymnázium J. Vrchlického v Klatovech má také svou vzdálenou laboratoř dostupnou na svých webových stránkách [3]. Je zde několik unikátních úloh, každá s využitím webové kamery a navržena pouze pro svůj účel. Úlohy opět nejsou lehce modifikovatelné pro použití na další úlohy.

Na Technické univerzitě v Liberci je dostupný e-learningový portál, postavený na systému Moodle. V portálu je možné vytvářet modifikovatelná cvičení, která je možné vyplnit skrze webový prohlížeč. Portál ovšem nedisponuje žádnou možností fyzického měření na přístrojích ani správou a zpracováním úloh z měření.

Z uvedených řešení je zřejmé, že ani jedno není variabilní s možností využití pro více účelů. Tato práce je zaměřená na vytvoření právě takového systému, tedy jednoduše rozšířitelného a přístrojově nezávislého, který umožní praktické měření, zpracování a dálkové ovládání.

3 Teoretická část

V této části práce bude popsána obecná koncepce komunikace, možnosti technologií pro tvorbu webových stránek, návrh grafického rozhraní webových aplikací a vysvětleny a popsány SCPI kompatibilní přístroje.

3.1 Druhy komunikace mezi stanicemi

Mezi počítači (stanicemi) existují dva základní druhy komunikace. Komunikace klient / klient (peer-to-peer), kde každá stanice může fungovat jako klient a jako server současně. Všechny stanice jsou si zde rovnocenné. Následně jeden klient navazuje komunikaci s ostatními dle potřeby. Opakem komunikace klient / klient je komunikace klient / server. V tomto typu komunikace se klientská stanice pro poskytnutí služby připojuje na server. V případě webového prostředí jsou úlohy jednotlivých stanic jasně definovány. Server je stanice, kde je umístěna samotná aplikace a kde běží webový server. Klientem je stanice s webovým prohlížečem.

3.2 Technologie pro provoz a tvorbu webových stránek

Ve 21. století se vývoj webových stránek rapidně změnil. Za posledních deset let se na poli webových technologií objevilo několik nových jazyků a spousta knihoven, které vývojářům ulehčí vývoj webových stránek a aplikací. Stávající technologie mezitím udělaly výrazný krok vpřed a vývojáři tak mají široké možnosti.

3.2.1 Webový backend

Do kategorie webového backendu se řadí vše, co ve výsledku neuvidí konečný uživatel, který si webové stránky prohlíží. Patří sem tedy jak webový server, který komunikuje pomocí **HTTP** nebo **HTTPS** požadavků s webovým prohlížečem, tak také různé skriptovací jazyky, které formují a generují webové stránky do statické podoby a díky serveru je odešlou prohlížeči. V neposlední

řadě sem patří rovněž databáze, do kterých se všechna data ukládají a následně se také čtou.

Webových serverů, které zasílají čtenářům webových stránek obsah do prohlížeče, je celá řada. Mezi nejvíce používané webové servery patří například **Apache** od Apache Foundation, který následuje **IIS** od společnosti Microsoft, nebo server **nginx**, který je vyvíjen společností NGINX, Inc. Tyto tři webové servery zaujímaly v dubnu 2015 [4] celých 82 % trhu. Apache si drží svou první příčku již od roku 1996, posledních pár let jeho tržní podíl klesá ve prospěch ISS a nginx.

K tvorbě dynamického obsahu se na straně webového serveru používají skriptovací jazyky. Webový server v tomto případě předá zpracování skriptovacímu jazyku, který výslednou zprávu zpřístupní zpět serveru. V dubnu 2015 byl jako nejpoužívanější jazyk s asi 82 % tržního podílu **PHP,** který vyvíjí skupina PHP Group [5]. Druhý nejpoužívanější jazyk je **ASP.NET** od společnosti Microsoft s podílem okolo 17 %. S minoritním podílem jsou to dále například **Java**, **Ruby** nebo **Python**. V posledních letech se dostává do popředí rovněž **JavaScript**, který je obecně spjatý spíše se skriptováním na straně klienta. Rychlost interpretace JavaScriptového kódu rapidně vzrůstá, proto se tento jazyk začíná velmi často používat také na serverové části. Aby se JavaScript mohl použít na straně serveru, musí zde být instalován jeho interpret. Nejčastěji používaný je **Node.js**. Na webovém serveru může být dostupných více skriptovacích jazyků najednou.

K ukládání dat pro dynamické stránky se nejčastěji používá databáze. U webových stránek jsou to ve většině případů relační databázové systémy. Ve větší míře se používá pouze několik databází [6]. Z komerčně dostupných databázových systémů je nutné zmínit **MSSQL** od Microsoftu a databázi

Oracle. Z volně dostupných databázových systémů je pak nejoblíbenější **MySQL**, **PostgreSQL**, **SQLite** a **MariaDB**, což je vlastně odnož MySQL.

Všechny tyto technologie musí samozřejmě pracovat v rámci operačního systému. Pro webové servery (a servery obecně) se ve větší míře používá **Linux** [7], poloviční podíl oproti tomu má serverové řešení **Windows.** Windows je nutný pro běh webového serveru IIS a skriptovacího jazyka ASP.NET. Pro tyto systémy většinou existují různé kompletní balíčky technologií. Pro systém linux je to například **LAMP**, tedy Linux, Apache, MySQL a PHP. Pro Windows je to obdoba **WAMP**. Vše lze instalovat i jednotlivě a zvolit různě kompatibilní varianty.

3.2.2 Webový frontend

Webový frontend je skupina technologií, se kterou se pracuje na straně klienta, tedy typicky ve webovém prohlížeči. Jde primárně o technologie pro popis struktury stránky, technologie pro grafické úpravy a dále například možné dynamické změny pomocí skriptovacího jazyka.

Pro popis struktury webové stránky se většinou používá značkovací jazyk **HTML**, v menší míře **XHTML**, který je postaven na bázi XML [8]. HTML bylo v roce 2014 standardizováno ve verzi 5. Tato verze byla standardizována po patnáctileté odmlce. HTML verze 4.01 bylo standardizováno v roce 1999. HTML ve verzi 4.01 je i v současnosti používáno na drtivé většině stránek. HTML ve verzi 5 reflektuje aktuální moderní dobu a přidává podporu nových sémantických prvků a moderních technologií. Oproti tomu XHTML ve verzi 1.1, vydáno v roce 2001, již nedostane svého následovníka.

K finální grafické úpravě dokumentu se používá jazyk kaskádových stylů **CSS**. Tento jazyk slouží pro popis způsobu zobrazení prvků webových stránek v jazyce HTML nebo XHTML. CSS je aktuálně standardizované ve verzi 2.1. V roce 2015 by měla být standardizována očekávaná verze 3, jež přinese

moderní možnosti pro úpravu vzhledu. Tato verze byla vyvíjena společně s HTML verzí 5. Mnohé vlastnosti ale v moderních prohlížečích fungují zcela bez problému již v současnosti.

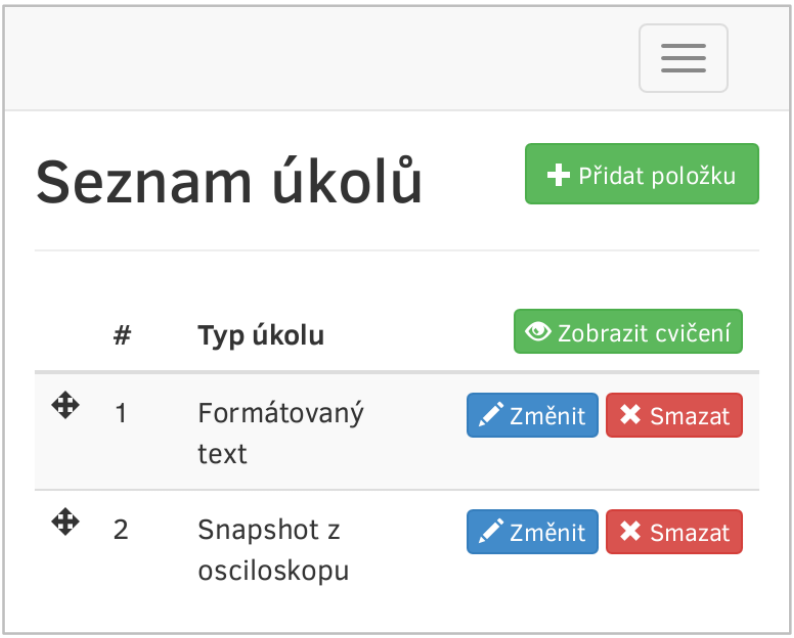
Pro skriptování na straně klienta, tedy ve webovém prohlížeči, se k dynamickému zobrazení stránek využívá nejvíce **JavaScript** [9]. JavaScript je objektově orientovaný skriptovací jazyk. Tímto jazykem lze ovládat jednotlivé HTML prvky stránky a měnit CSS vlastnosti těchto prvků. Interpretace samotného JavaScriptu se v prohlížeči zahájí až po stažení HTML souboru webové stránky. Standardizovaná verze JavaScriptu je označena jako EC**MAScript,** normovaný neziskovou společností ECMA International. Poslední verzí ECMAScriptu je verze 5.1 vydana v roce 2011. Aktuálně je v přípravě verze 6, která je stejně jako CSS 3 vyvíjena současně s HTML 5. Tato verze přinese mnoho moderních možností a podporu tříd a modulů. Jako další se používá technologie Flash od firmy Adobe, případně Silverlight od Microsoftu, které řeší jak grafickou podobu stránek, tak rovněž funkční stránku. Od těchto technologií se nicméně v posledních letech upouští a vše přechází pouze na JavaScript a HTML 5. V mobilních zařízeních totiž není ve většině případů Flash ani Silverlight dostupný.

V současnosti existuje mnoho frameworků, které řeší buď komplexně všechny klientské části, nebo pouze některé. Příkladem mohou být frameworky **Bootstrap** a **Foundation**, které se starají o sémantiku stránky a grafickou reprezentaci stránky. Dále také obsahují JavaScriptové funkce pomáhající s dynamičností stránek. Další knihovnou, zejména pro snadnější práci s HTML stránkou, je **jQuery**. Tato knihovna ale obsahuje mnohem více funkcí, například asynchronní požadavky.

3.3 Návrh grafického rozhraní

V dnešní moderní době je důležité držet se nových technologií. Zejména v posledních letech se vyrábějí nové typy zařízení, skrze které si prohlížíme obsah na internetu. Nezanedbatelné procento návštěv webových stránek je skrze mobilní telefony nebo tablety. A procento rok od roku stoupá. Při návrhu grafického prostředí je tedy důležité s těmito zařízeními počítat a stránku jim uzpůsobit. Pro mobilní telefony nebo tablety je nutné vytvořit ovládací prvky dostatečně velké, aby se daly ovládat prsty nebo stylusem. Rozložení prvků bude muset být trochu rozdílné. Obrázek 1 zobrazuje stránku pro klasický desktopový počítač. Obrázek 2 pak ukazuje stránku na mobilním telefonu. Při porovnání obrázků je vidět, že velikost textu i velikost tlačítek je na tomto malém zařízení větší. Dále si lze všimnout seskupeného menu do jedné položky. Po stisku se nám samotné menu zobrazí překryté přes ostatní obsah.

Obrázek 1 - Příklad rozložení stránky pro desktop



Obrázek 2 - Příklad rozložení aplikace pro mobilní telefon

3.4 SCPI kompatibilní přístroje

SCPI (Standard Commands for Programmable Instruments) je standard [10], definující syntaxi a samotné příkazy používané k ovládání programovatelných a měřicích přístrojů. Tento standard specifikuje společnou syntaxi, strukturu příkazů a datových formátů. Standard ale již nespecifikuje, jakými kanály mají být příkazy zasílány. Příkazy se tak mohou zasílat pomocí RS-232, USB, ethernetu nebo dalšími kanály. Příkazy jsou dvojího typu, nastavovací (nastavení napětí na zdroji) nebo dotazovací (získání aktuální hodnoty na čidle). Dotazy jsou identifikovány otazníkem na konci příkazu, u těchto příkazů se očekává odpověď přístroje. Některé příkazy mohou být použity jak pro nastavení, tak pro čtení. Stejný příkaz bez otazníku se tak použije pro nastavení určité hodnoty, s otazníkem pro čtení aktuální nebo dříve uložené hodnoty. Existují také kombinované dotazy, které spustí na přístroji nějakou operaci a následně vrátí výsledek operace. Jednotlivé příkazy se slučují do stromové hierarchie. Například veškeré příkazy pro měření hodnot začínají **MEASure**, následuje měření napětí **VOLTage** a poté stejnosměrná složka **DC**.

Jednotlivé části se oddělují znakem dvojtečky. Výsledný příkaz by pak vypadal následovně: **MEASure:VOLTage:DC?**. Některé příkazy dovolují zaslání hodnoty možného nastavení. Tato hodnota se zasílá za příkazem a od příkazu se oddělí mezerou, více hodnot se od sebe odděluje čárkou. Například příkaz **SYSTem:COMMunicate:SERial:BAUD 2400** nastaví rychlost sériové komunikace na 2400 bit/s. Pokud chceme zaslat do přístroje více příkazů najednou, je nutné je oddělit znakem středníku a na začátek každého příkazu, kromě toho úplně prvního, dodat dvojtečku. Více příkazů v jednom požadavku, jak bylo popsáno výše, může vypadat jako v následující ukázce: **MEASure:VOLTage:DC?;:MEASure:VOLTage:AC?**. Příkazy se píší velkými a malými písmeny, kde část příkazu malými písmeny je nepovinná a je možné ji vynechat. Příkaz **DISPlay:OUTPut?** je tedy ekvivalentní s příkazem **DISP:OUTP?**.

SCPI dále obsahuje množinu povinných obecných příkazů (common commands) podle standardu **IEEE 488.2** (označovaným jako **GPIB**). Jedná se rovněž o speciální jedno- nebo více-vodičové příkazy, které slouží k ovládání přístrojů na nižších vrstvách, proto se mohou jejich efekty lišit od příkazů jinak běžně používaných. Tyto příkazy slouží například k identifikaci stavu a k ovládání samotného přístroje. Jsou uvozeny znakem hvězdičky a mají vždy pouze 3 znaky. V případě, že je na konci příkazu také otazník, je přístroj povinen odpovědět. Například pro nulování přístroje (ukončení probíhajících operací) a nastavení přístroje do klidového stavu se použije příkaz ***RST**. Tento příkaz se může lišit od podobného příkazu **:SYSTem:PRESet**. Pro získání specifikace přístroje se použije příkaz ***IDN?**.

3.4.1 Přístroje v učebně AP9

Učebna AP9 prošla v předešlých letech modernizací studentských pracovišť [11] [12]. Kromě změny koncepce cvičení byl obměněn přístrojový park každého studentského pracoviště. V této práci nás zajímají hlavně SCPI

kompatibilní přístroje. V učebně jsou využívány dva druhy osciloskopů. Vzhledem k neúměrně vysokým nákladům na pořízení, údržbu a servis špičkových typů přístrojů byl ke každé studentské stanici pořízen levnější osciloskop **GW Instek GDS-2072A**. Tento osciloskop je efektivně využíván při každém cvičení pro jednoho až dva studenty. Investičně náročný a vlastnostmi špičkový osciloskop **Agilent DSO9254A** je v učebně pouze jediný a využívá se pouze v úlohách, na které levnější osciloskop již nestačí. Práci na tomto osciloskopu prezentuje především vyučující. Studenti si následně úlohu mohou ve větších skupinkách vyzkoušet.

Osciloskop **GW Instek GDS-2072A** [13] [14] [15] je instalován na každém pracovišti studenta. Tento osciloskop pracuje na dedikovaném uzavřeném operačním systému. Disponuje dvěma vstupními kanály, které mají šířku pásma 70 MHz a vzorkování až 2 GSa/s (2 giga vzorků za sekundu) v reálném čase. Osciloskop disponuje 8″ LCD TFT displejem s rozlišením 800 x 600 bodů. Přístroje je možné propojit s počítačem pomocí USB. Některé přístroje jsou rozšířené o moduly LAN, GPIB, VGA nebo generátoru signálu.

Jediný osciloskop **Agilent DSO9254A** [16] [17] je profesionální osciloskop a je významným prvkem přístrojového parku. Přístroj se využívá zejména při úlohách měření na moderních rozhraních a sběrnicích, jako jsou USB, PCIe nebo SATA, kde se přenáší velké množství dat nebo se přenáší na vysokých přenosových rychlostech. Osciloskop pracuje na operačním systému Windows 7 a obsahuje výkonný hardware pro měření analogových signálů. Obsahuje 4 vstupní kanály. Osciloskop je dále možné rozšířit o další moduly. Základní šířka pásma je 2.5 GHz programově rozšířitelná na 4 GHz s maximálním vzorkováním 10 GSa/s v reálném čase pro všechny 4 kanály nebo 20 GSa/s pro kanály 1 a 3. Tento osciloskop má velmi dlouho paměť vzorků 500 Mpts na každý kanál, případně extrémně dlouhou paměť 1 Gpts při využití kanálů 1 a 3.

4 Praktická část

Nyní se dostáváme k samotné realizaci aplikace. Pro vytváření a testování aplikace byl k dispozici testovací počítač. Tento počítač disponoval procesorem se 2 jádry a taktovací frekvencí 3.5 GHz, dále obsahoval 3 GB operační paměti a 120 GB plotnový disk. Na počítači byly nainstalovány 64bitové Windows 7 a byl připojen na gigabitovou školní síť. K počítači byl pro testování připojen osciloskop GW Instek GDS-2072A. Stejný osciloskop je připojen ke každému studentskému PC na měřicím pracovišti v učebně AP9, kde jsou aktuálně instalovány stanice s procesory Intel i5. Vzhledem k návrhu celého systému na výkonu studentských PC stanic skoro nezáleží.

4.1 Výběr technologií

Před začátkem samotné realizace aplikace bylo nutné si na základě požadavků zvolit vhodné technologie. Vzhledem k tomu, že na dostupném testovacím stroji běžely Windows 7, ale pro samotné ostré nasazení se počítá spíše s linuxovým serverem, musely být technologie kompatibilní s oběma operačními systémy.

4.1.1 Webový backend

Pro webový backend v podobě webového serveru byl zvolen Apache. Tento server je nejpoužívanější a obsahuje mnoho modulů, kterými je možné rozšířit funkčnost. K tomuto serveru existuje velice přehledná a rozsáhlá dokumentace, nastavení tak není žádný problém. U tohoto serveru se bude v aplikaci využívat například modul mod_rewrite pro optimalizované tvary webové adresy.

Jako skriptovací jazyk pro dynamické stránky bylo zvoleno PHP, které je nejpoužívanějším jazykem a má plnou podporu v Apache serveru. Vzhledem k tomu, že se v budoucnu počítá s linuxovým serverem, nemohlo být zvoleno

například ASP.NET. PHP svým rozsahem funkcí a modulů umožňuje vytvářet všechny možné úkony. Spolu s podrobnou dokumentací a nepřeberným množstvím nastavení je to pro naši aplikaci ta nejlepší volba.

Jak již bylo uvedeno v předchozí kapitole, rovněž typů databází je dnes dostupné velké množství. Databáze MySQL má v jazyce PHP plnou podporu a proto byla vybrána pro tuto aplikaci. Databáze zvládne všechny pro aplikaci potřebné úkony. V jazyce PHP je podpora takzvaných „prepared statemens“, což jsou předpřipravené dotazy na databázi, kterým se následně pouze předají data. Díky tomu je komunikace s databází ošetřena proti nechtěnému napadnutí. Tento modul se v aplikaci využije pro zajištění co největšího zabezpečení dat.

4.1.2 Webový frontend

U webového frontendu není v zásadě příliš z čeho vybírat. Pro strukturu samotných stránek se využívá nejnovější pátá verze HTML. Pro vzhledovou úpravu stránek bude použito CSS ve verzi 3. Tato verze sice není ještě oficiálně vydána, podpora prvků této verze je ale již v moderních prohlížečích zakomponována a není tak důvod odkládat nasazení této verze do stránek. Pro zajištění dynamické části řešení na straně klienta se využije JavaScript. Aplikace by si měla vystačit se základními funkcemi tohoto jazyka, případně lze použít některé nové prvky z připravovaného standardu, které jsou již v prohlížečích implementovány.

4.1.3 Ostatní technologie a použité knihovny

Při vývoji aplikace se nebude spoléhat pouze na dříve uvedené technologie. Pro aplikaci se využijí také volně dostupné knihovny, které usnadní vývoj, vylepší komfort používání, a díky tomu se studentům i vyučujícím usnadní práce.

Pro oddělení aplikační vrstvy (PHP skripty spolu s požadavky na databázi) a prezenční vrstvy aplikace (HTML struktura stránky) se využije šablonovací systém Smarty ve verzi 3. Toto oddělení přispívá k lepší čitelnosti kódu. PHP skripty předávají všechna potřebná data do šablon pomocí proměnných. Systém Smarty umožňuje v šablonách použití řídících struktur, cyklů, vestavěných funkcí pro práci s řetězci a dalších funkcí. Ukázka šablony v systému Smarty je zobrazena níže. K vygenerování protokolu ve formátu PDF o vypracování cvičení se použije knihovna TCPDF. Knihovna umožňuje snadný převod HTML spolu s CSS do PDF dokumentu.

```
<table class="table table-striped table-hover">
    {foreach $studentArray as $student}
        {if $student@iteration == 1}
            <thead>
                <tr>
                    <th width="60">#</th>
                    <th width="200">Osobní číslo</th>
                    <th>Příjmení, jméno</th>
                    <th>Přihl. jméno</th>
                    <th width="200"></th>
                </tr>
            </thead>
        {/if}
        <tr>
            <td>{$student@iteration|string_format:'%d'}</td>
            <td>{$student.IDnumber|escape:'htmlall'}</td>
            <td>{$student.name|escape:'htmlall'}</td>
            <td>{$student.login|escape:'htmlall'}</td>
            <td align="right"></td>
        </tr>
    {foreachelse}
        <tr><td>Nenalezeny žádné záznamy.</td></tr>
    {/foreach}
</table>
```

Ukázka kódu 1 - Ukázka šablony v systému Smarty

Hlavní knihovnou, zejména pro práci s HTML stránkou a pro asynchronní načítání dat, bude **jQuery**. Tato knihovna umožňuje snadnou práci s HTML dokumentem, změnu CSS parametrů každého prvku, snadnou správu JavaScriptových událostí a mnoho dalšího.

Aby se při návrhu splnily všechny skutečnosti uvedené v kapitole 3.3, byl jako frontendový Framework zvolen Bootstrap ve verzi 3. Bootstrap je framework, ve kterém jsou obsaženy základní komponenty uživatelského rozhraní. Mezi ně patří například hlavní menu, tlačítka, formuláře, tabulky, ikonky a další prvky, zabaleny do pěkného vzhledu s nepřeberným množstvím nastavení. V tomto frameworku jsou vyřešena všechna důležitá pravidla pro správné zobrazení stránky na různých typech zařízení a typografie. Tvůrce stránek tak pouze dává jednotlivé komponenty dohromady pro vytvoření komplexní stránky, která je optimalizovaná pro všechna zařízení.

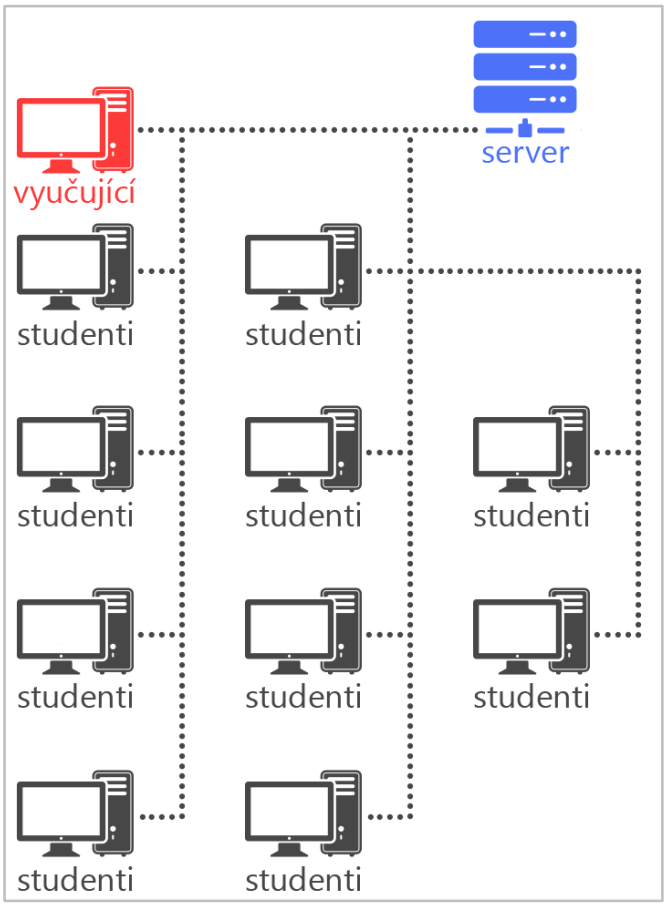
Obrázek 3 - Přihlášení do aplikace

Jako doplněk tohoto frameworku budou použity knihovny **Bootstrap Dialog**, **Boostrap Select** a **Bootsrap Datepicker**. První knihovna zajišťuje snadné vytvoření dialogových oken v Bootstrap stylu bez nutnosti psaní několika řádků HTML kódu. Druhá knihovna převede standardní formulářový prvek select na vylepšenou variantu tohoto prvku. Díky tomu lze například v tomto prvku vyhledávat a lépe vybírat více možností. Třetí knihovna dovoluje snadnější zadávání dat pomocí zobrazeného klikacího kalendáře. Poslední knihovnou je **TinyMCE**, která v aplikaci zpřístupňuje WYSIWYG

editor pro psaní formátovaných textů. Knihovna umožňuje jak základní formátování (tučný text, podtržený text, velikost písma, zarovnání a další), tak i složitější (tabulky, nadpisy, odrážky, odkazy a další).

4.2 Komunikace klient / server

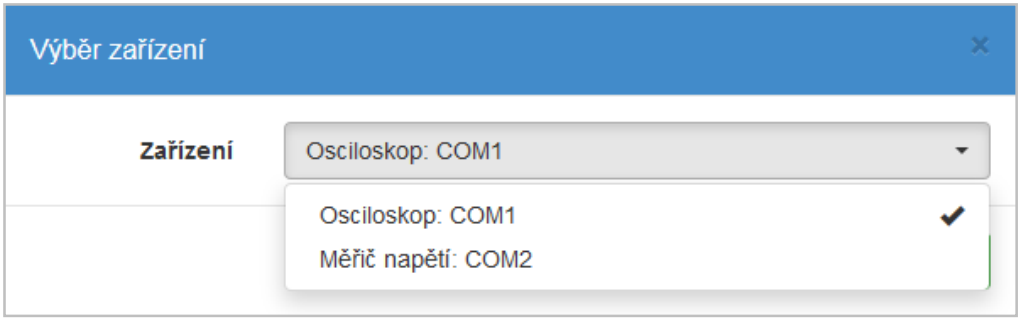
V kapitole 3.1 byly popsány druhy komunikace. V této aplikaci bude použita komunikace klient / server. Serverem je stanice s aplikací a databází. Klientem je studentská stanice, ke které jsou připojeny přístroje, případně stanice vyučujícího. V učebně AP9, jak již bylo dříve zmíněno, je dostupných 10 studentských stanic a jedna stanice vyučujícího. Máme zde dva typy komunikace mezi klientem a serverem. Komunikaci skrze webový prohlížeč a komunikaci skrze program, zajišťující přístup k připojeným zařízením na studentské stanici.



Obrázek 4 – Rozložení studentských stanic v učebně AP9

4.2.1 Komunikace skrze webové prostředí

Komunikací pomocí webového prohlížeče se ovládá celá aplikace – od přihlášení, přes nastavení učeben, počítačů a přístrojů v něm, správu cvičení až po samotné vyplňování cvičení studenty. Ve webovém prohlížeči tedy vyučující a student komunikuje se serverem. V případě, že je potřeba načíst určitá data z nějakého zařízení na klientské stanici, uloží se vše do databáze a na řadu se dostává program na klientské stanici.



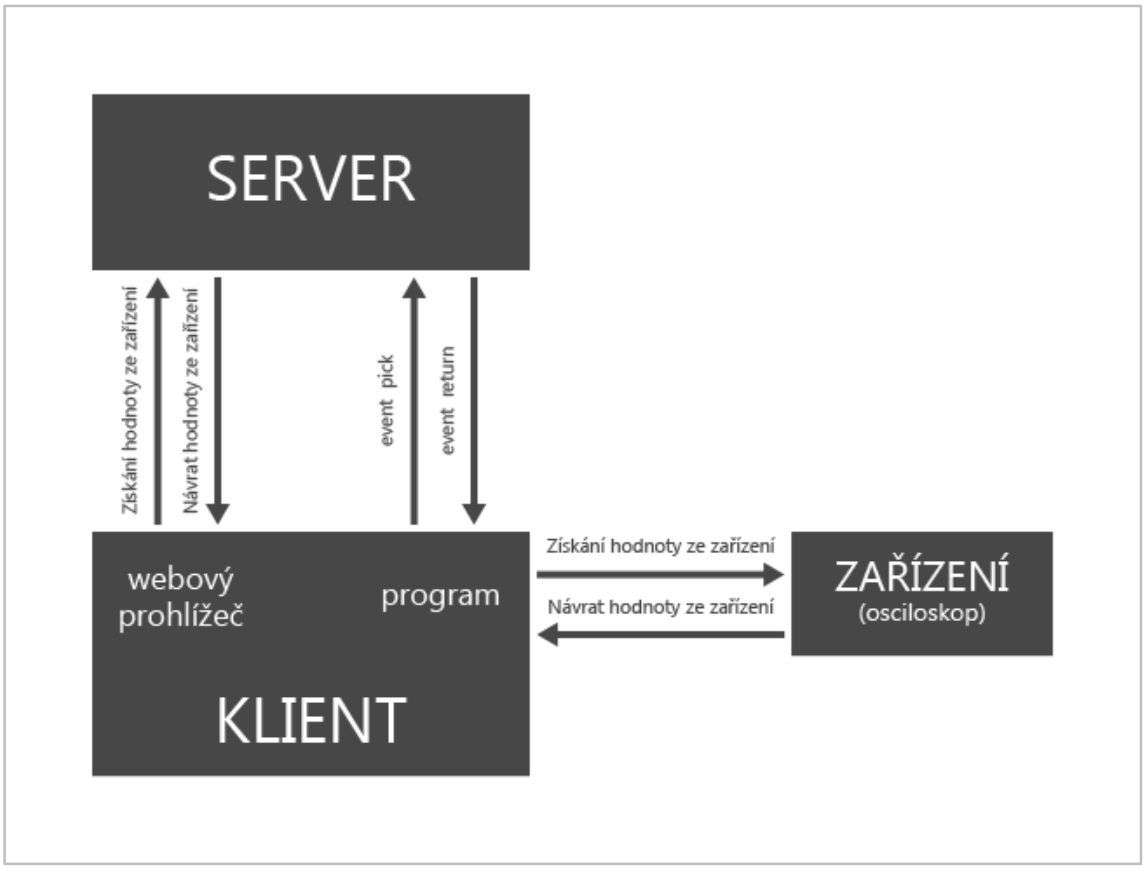
Obrázek 5 - Výběr zařízení, ze kterého se mají vyčítat data

4.2.2 Komunikace pro ovládání přístrojů

Program na klientské stanici má na starost dva základní úkoly. Prvním z nich je zjistit, zdali není potřeba získat data z nějakého zařízení (event pick) a druhým je vykonat příkaz a odeslat získaná data zpět (event return).

Veškerá komunikace programu se serverem probíhá pomocí HTTP, případně HTTPS protokolu. Vzhledem k tomu, že je ke komunikaci používán pouze požadavek typu GET, přicházejí zde určitá omezení. GET požadavek byl zvolen pro rychlejší zpracování a méně zasílaných dat v HTTP požadavku. GET požadavek znamená, že veškerá data jsou předávána v adrese požadavku. Apache webový server má omezení na délku výsledné adresy, v určitých případech tak nelze data vrátit na jeden požadavek, ale je těchto požadavků nutné provést více.

Vyzvednutí události k vykonání (event pick) je realizováno na adrese **{IP-SERVERU}/api/event/pick/** (případně lze použít kratší variantu dostupnou na adrese **{IP-SERVERU}/p**). Tento požadavek program na klientské stanici provádí každých 100ms. V případě, že pro tento počítač není žádná událost k vykonání, server vrátí programu hodnotu NULL. V opačném případě server vrátí požadavek, který je v pořadí první k vyřízení, a to ve formátu **{EVENT-ID};{PORT};{COMMAND}**, kde EVENT-ID je hexadecimální číslo identifikující událost a potřebné k odeslání získaných dat na server, PORT je port, na kterém je připojené zařízení, a COMMAND je příkaz, který se má do zařízení zaslat. Hodnota COMMAND může být sada příkazů oddělená středníkem. V jednom požadavku lze tedy zaslat více příkazů. Výstup může vypadat například takto: **2e;COM1;DISPLAY:OUTPNG**.



Obrázek 6 - Komunikace pro získání hodnoty ze zařízení

V případě, že program dostane událost k vyřízení, zašle příkaz na daný virtuální nebo fyzický port (příslušné zařízení) a počká na vrácení dat. Získaná data mohou být dvou typů; klasický textový výstup, například hodnota napětí na sondě, nebo binární, například snímek aktuálního stavu displeje osciloskopu ve formátu PNG. Data program následně odešle na server ke zpracování.

Odeslání dat na server (event return) je realizováno na adrese **{IP-SERVERU}/api/event/return/{EVENR-ID}** (případně lze použít zkrácenou variantu dostupnou na adrese **{IP-SERVERU}/r/{EVENT-ID}**). V případě klasického textu se použije GET parametr **d** se získanou hodnotou. Výsledná adresa pak vypadá například **{IP-SERVERU}/r/3a/?d=239V**. U binárních dat je to trochu složitější. Vzhledem k omezení délky adresy a s přihlédnutím k maximální velikosti standardního ethernetového rámce (tedy maximálně 1500 bytů dat), jsou binární data v základním režimu zasílána rozdělena na více částí.

Jelikož není možné v adrese přenášet binární data, je nutné binární data zakódovat. Data se tedy dle možností rozdělí do jednotlivých částí a následně se zakódují pomocí kódování Base64. Toto kódování převede každé 3 bajty binárních dat na 4 bajty ASCII znaků. Výsledná data jsou tedy přibližně o 33 % větší a skládají se pouze ze znaků anglické abecedy (malá a velká písmena), čísel, znaku plus (‚+‘) a lomítka (‚/‘). V případě, že počet bajtů dat není dělitelný 3, přidá se na konec příslušný počet rovnítek (‚=‘). Data se zasílají pomocí GET parametru **b**. Aby se zaručilo doručení všech dat a ve správném pořadí, bylo nutné navrhnout určitou kontrolu těchto dat. Za data v kódování Base64 se přidá ještě identifikátor části, což je číslo od nuly do devíti. Toto číslo se s každou další částí zvětšuje, po devítce následuje opět nula. Server toto číslo následně vrátí programu pro kontrolu, že data došla v pořádku. V případě, kdy server zjistí, že mezi předchozí a následující částí, která dorazila, chybí jedna nebo více částí (například přijde-li na server část nula a následně až část dva),

označí událost jako nevyřízenou a navrátí programu hodnotu NULL. Program poté celou operaci musí provést znovu. Od vyzvednutí události (event pick), po získání dat ze zařízení a následně k navrácení získaných hodnot. V případě poslední části se na konec připojí ještě znak vykřičník (‚!'), aby server mohl událost uzavřít a vrátit data studentovi. Výsledná adresa pro navrácení dat pak vypadá například **{IP-ADRESA}/r/3b?b=MjM5Vg==0!**. V příkladu je zaslán řetězec „239V" zakódovaný pomocí Base64, označen jako část 0 a zároveň jako poslední část.

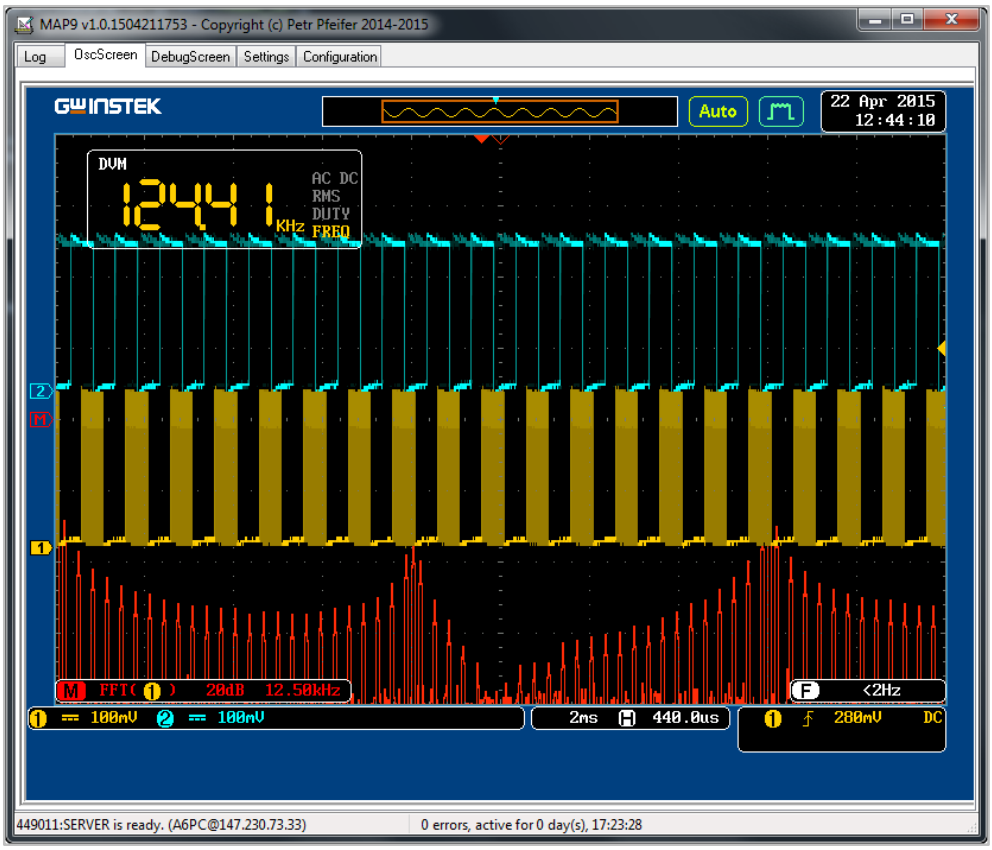
V aplikaci je také dostupná možnost pro podporu přístrojů s velkou hloubkou záznamu. Osciloskop **Agilent DSO9254A** podporuje hloubku záznamu až 1 Gpts na jeden kanál (pouze pro 1 a 3 kanál). Tato data tak mají maximální velikost 4 GB. Data se neukládají do databáze, nicméně jsou pro ně vytvořeny fyzické soubory na disku. Tyto soubory je pak možné dále analyzovat díky externímu programu a do aplikace dodat pouze výsledek, například v HTML formátu.

4.2.3 Program pro komunikaci se zařízením

Samotný program pro komunikaci se zařízeními nebyl součástí této diplomové práce. Byl ale vyvíjen v průběhu tvorby diplomové práce, aby na tuto práci navazoval a vše bylo plně funkční. Program byl vyvíjen vedoucím diplomové práce Petrem Pfeiferem.

Samotný program je koncipován tak, že se každých 100ms (respektive 100ms od poslední odpovědi ze serveru) zašle dotaz serveru, zda pro tento počítač není dostupný ve frontě příkaz k vykonání (event pick). V případě, že se vrátí hodnota **NULL**, zeptá se program znovu za zmíněných 100ms. Pokud dostane od serveru odpověď s příkazem, tento příkaz se zpracuje (zašle na příslušné zařízení) a jakmile je obdržena odpověď, zašlou se data zpět na server (event return).

Program také z připojeného osciloskopu automaticky stahuje a převádí interní formát snímků obrazovky na obrázek formátu PNG každých několik sekund, aby student, požadující tento obrázek z osciloskopu, nemusel čekat na přenos, zpracování a navrácení dat obrázku až 3 sekundy. Tolik totiž trvá navrácení obrázku a samotné dekódování. Obrázek se z osciloskopu vrací ve formátu RGB 565 zakódovaný pomocí bezeztrátové komprese RLE.



Obrázek 7 - Ukázka programu pro komunikaci se zařízeními

4.3 Přihlášení do aplikace a její rozhraní

Přihlášení do aplikace se provede zadáním přihlašovacího jména a hesla. Obrázek 3 zobrazuje, jak vypadá přihlašovací formulář. Pod formulářem je zelená informace o počítači, ze kterého se student aktuálně přihlašuje. V případě, že počítač není v aplikaci zaveden, bude student o této skutečnosti informován červeným textem.

Hesla jsou uložena v databázi a hashována pomocí algoritmu **bcrypt**. Tento algoritmus je v současnosti nejvhodnější pro hashování hesel dostupný v jazyce PHP [18]. Tento algoritmus je, oproti například **md5** nebo **sha1**, pomalý. Dva zmíněné algoritmy jsou naopak velice rychlé. Na moderních počítačových clusterech s mnoha výkonnými grafickými kartami je tak otázka sekund, než takové heslo prolomí. Existují také před počítané tabulky (rainbow tables) hesel. Algoritmus se používá ve verzi **$2y**, která je nejvíce bezpečná. Parametr **cost** tohoto algoritmu je zvolen na hodnotu 12. Tento parametr určuje sílu výsledného hesla a s rostoucím číslem se zvyšuje také náročnost na výpočet. V budoucnu je možné tento parametr navyšovat podle toho, jak se bude zvyšovat výkon počítačů. Jelikož je možné aplikaci provozovat také po nezabezpečeném spojení, před vstupem do algoritmu bcrypt se k heslu přidá sůl (náhodný řetězec) a zahashuje pomocí algoritmu md5. JavaScript při přihlášení získá z databáze sůl a následně heslo zahashuje pomocí této funkce, přenáší se tak pouze otisk a samotné heslo by se muselo teprve dešifrovat. Je ovšem silně doporučeno používat aplikaci na zašifrovaném spojení.

```
$salt = getRandomString(10);
$blowfishSalt = bin2hex(openssl_random_pseudo_bytes(22));
$passwordHash = md5(md5($password).md5($salt));
$passwordHash = crypt($passwordHash, '$2y$12$'.$blowfishSalt);
```

Ukázka kódu 2 - Ukázka hashování hesel v aplikaci

Po přihlášení se student nebo učitel dostane do samotného rozhraní aplikace, které je přehledně děleno. V horní části je dostupné hlavní menu, které drží svou pozici i při rolování stránky dolů. V levé části menu jsou jednotlivé sekce aplikace, v pravé informace o přihlášeném uživateli spolu s informací o počítači a odkaz k odhlášení uživatele. Pod menu se nachází samotný obsah stránky, který se automaticky uzpůsobuje šířce stránky. Každá stránka má svůj nadpis a samotná data pod tímto nadpisem. Na pravé straně jsou vždy umístěna funkční tlačítka k různým akcím v dané sekci.

4.4 Rozhraní pro vyučující

Rozhraní pro vyučující je důležitým prvkem celé aplikace. V tomto rozhraní se aplikace nastavuje, spravují se zde všichni studenti, předměty a termíny, učebny a jejich hardwarové vybavení, a hlavně se spravují všechna cvičení pro studenty. V následujících kapitolách bude každá sekce aplikace podrobně popsána a budou vysvětleny důležité elementy pro správný chod celé aplikace.

4.4.1 Správa studentů

Správa studentů je jedním z hlavních pilířů aplikace. Aby mohl student vypracovávat cvičení, musí se do systému vložit. Každý student může být v aplikaci veden pouze jednou. Rozhodující je pro to osobní číslo, které je v rámci aplikace unikátní. O každém studentovi si aplikace uchovává jeho osobní číslo, jméno a příjmení, e-mail, přihlašovací jméno a heslo v zašifrované podobě. Na stránce se seznamem uživatelů se zobrazí na každou stránku 20 studentů. V seznamu lze vyhledávat pomocí fultextového vyhledávání dle jména nebo příjmení.

4.4.2 Správa učeben, počítačů a přístrojů

Další nedílnou součástí aplikace je správa učeben, v nich obsažených počítačů a přístrojů k nim připojených. U každé učebny se eviduje označení této učebny (například AP9), číslo dveří (například A-AP9) a poznámka. V seznamu učeben vidíme také počet počítačů, které jsou v dané učebně dostupné. Po kliknutí na toto tlačítko se vyučující dostane na seznam počítačů v dané učebně.

O každém počítači se eviduje označení počítače, poznámka a jeho IP adresa. IP adresa je důležitá pro následné přihlášení studenta, aby mohl na zařízení připojená k jeho počítači zasílat příkazy. Ze seznamu počítačů se lze v horní části dostat na stránku, kde jsou zobrazeny všechny obrazovky osciloskopů ze všech aktivních počítačů v dané učebně. Vyučující tak má

detailní přehled, jak studenti na jednotlivých pracovištích pracují a zda nějaký student nemá problém se správným nastavením osciloskopu. Zároveň lze na této stránce ukončit všechny běžící programy na počítačích dané učebny. Toto se může provést třeba na konci dne, kdy už v učebně žádná hodina nebude, aby programy zbytečně nekomunikovaly se serverem. V seznamu všech počítačů v učebně je u každého z nich vidět počet aktuálně přiřazených přístrojů. Po kliknutí na příslušné tlačítko se vyučující dostane na seznam těchto přístrojů.

Obrázek 8 - Ukázka stránky s osciloskopy v dané učebně

U každého přístroje se eviduje jeho název, port, který musí být v rámci počítače unikátní, dále možnost stahování snapshotu z daného zařízení, unikátní inventární číslo, unikátní sériové číslo a poznámka k přístroji. Na základě portu a možnosti stahování obrázků je možné na tyto přístroje zasílat

příkazy a získávat z nich informace. Na každé zařízení z tohoto seznamu je možné zaslat předem definovanou konfiguraci. Lze tak například studentovi nastavit osciloskop přímo pro dané cvičení, aby vše nemusel dělat ručně. Správa těchto konfigurací bude popsána v kapitolách dále.

4.4.3 Správa předmětů a termínů

Správa předmětů a termínů je další nedílnou součástí aplikace. Aby bylo možné vytvořit cvičení daného předmětu, musí se do systému prvně vložit. O každém předmětu se eviduje jeho název, zkratka, školní rok, ve kterém je předmět dostupný, zimní nebo letní semestr, jeden nebo více vyučujících, kteří mají následně právo pro daný předmět spravovat cvičení, a poznámka ke cvičení. V seznamu je k dispozici pro každý předmět počet přiřazených termínů. Po kliknutí na tlačítko se dostaneme na seznam těchto termínů.

Každý termín je ve výsledku reálné cvičení daného předmětu. Každý termín se nastaví na přesný den v týdnu, přesný blok a dále také na učebnu. Tato kombinace musí být v systému unikátní, stejně jako je to ve skutečnosti. Ke každému termínu je možné přidat také poznámku. U každého termínu daného předmětu je k dispozici aktuální počet studentů, kteří jsou k termínu přiřazeni. Po kliknutí na příslušné tlačítko se dostaneme na jejich seznam.

V seznamu studentů na termínu jsou zobrazeny pouze základní informace o studentovi, přesněji pouze osobní číslo a studentovo jméno. Přidání studenta nebo studentů probíhá skrze dialogové okno. Následně se vybere jeden nebo více studentů, kteří mají být k termínu přiřazeni. Dostupní jsou pouze ti studenti, kteří doposud nejsou přiřazeni na žádný termín tohoto předmětu. V seznamu těchto studentů se dá vyhledávat pro rychlejší vybrání studentů.

4.4.4 Správa laboratorních úloh a cvičení

Správa laboratorních úloh a cvičení je hlavní funkcí celé aplikace. Vyučující, který má přiřazen nějaký předmět, zde může spravovat cvičení na každou hodinu v průběhu celého semestru. Pro přidání nového cvičení je nutné zadat předmět, dostupnost cvičení od určitého data, počet týdnů, po který je cvičení možné vypracovávat studenty, název samotného cvičení a popis cvičení ve WYSIWYG editoru, který se následně zobrazí studentovi při řešení. Dostupnost cvičení se zadává na začátek (pondělí) určitého týdne. Dále je možné cvičení ponechat neaktivním. Například pokud chceme cvičení nejprve připravit a zobrazit ho teprve poté, až bude celé hotové. Jakmile je nové cvičení přidané, je možné do tohoto cvičení vkládat jednotlivé úkoly.

V seznamu úkolů je tak dostupný přehled jednotlivých prvků cvičení seřazených přesně tak, jak se následně studentovi zobrazí. Přidávání nových cvičení probíhá přes dialogové okno. V tomto okně se vybere typ nové položky z dostupných šablon a následně je nutné vyplnit všechny potřebné informace. Aplikace nedovolí položku uložit, pokud nebude vše správně nastaveno. Po přidání této položky se zobrazí na úplném konci seznamu.

Obrázek 9 - Ukázka všech úkolů jednoho cvičení

Na začátku každého řádku s položkami je možné za pomocí myši měnit pořadí jednotlivých prvků. Stačí levé tlačítko myši podržet na ikonce všesměrové šipky a táhnout myší nahoru nebo dolů, v závislosti kam chceme danou položku přetáhnout. Pokud jsou všechny položky cvičení přidány, nastaveny a zařazeny ve správném pořadí, může si vyučující dané cvičení zobrazit. Toto zobrazení je naprosto identické, jak ho uvidí sám student, je pouze nefunkční a nelze tak pracovat se všemi součástmi stejně, jako bude moci student. Vyučující tak má přesnou představu o tom, jak se studentovi cvičení zobrazí a může případně před samotným zpřístupněním cvičení doladit veškeré detaily.

U každého cvičení lze také zobrazit seznam všech studentů, kteří mohou dané cvičení řešit. V tomto seznamu jsou vidět pouze základní informace o studentovi, konkrétně jméno a osobní číslo. Dále je v seznamu vidět poslední aktivita daného studenta u tohoto cvičení. Vyučující tak má přehled o tom, který student se cvičení věnuje a který ne. V případě, že student již cvičení uzavřel, může si vyučující zobrazit výsledný protokol ve formátu PDF se všemi vyplněnými výsledky.

4.4.5 Další funkce

V aplikaci jsou dostupné také další funkce. Mezi ně patří správa vyučujících. U vyučujícího se eviduje jméno, příjmení, přihlašovací jméno, heslo, e-mail a typ účtu. Typ účtu je administrátor nebo vyučující. Hlavní rozdíl je v tom, že administrátor může vykonávat některé operace, které vyučující vykonávat nemůže. Například je to právě správa vyučujících, nastavení aplikace, správa předmětů a termínů, správa učeben a počítačů a správa studentů. Vyučující má v těchto případech pouze práva pro čtení těchto dat.

V předešlém odstavci bylo zmíněno nastavení aplikace. To je dostupné pouze pro administrátory. V aplikaci lze nastavit rok startu aplikace, aby se při

vytváření předmětů neukazovaly roky, které již nejsou pro aplikaci relevantní. Dále e-mail, který slouží jako odesílatel všech e-mailů z aplikace a nakonec odkaz, kde je umístěna aktuální verze programu pro komunikaci mezi serverem a zařízeními na počítači studenta.

V rámci správy přístrojů byly zmíněny také předem definované konfigurace, které lze do jednotlivých zařízení nahrát. Správa těchto konfigurací je velice jednoduchá. Ke každé konfiguraci se ukládá pouze název konfigurace a samotná konfigurace, což je sada příkazů oddělených středníkem, které se odešlou na příslušné zařízení.

Další sekcí v rámci rozhraní pro vyučující je seznam aktivních programů. To je seznam zařízení, která aktuálně komunikují se serverem. Vyučující má díky tomu přehled, které počítače se serverem komunikují a které nikoliv. Je tak možné rychle zjistit, kde je problém a na problematickém počítači problém s programem vyřešit. Poslední funkcí v rozhraní je změna hesla vyučujícího. Při změně hesla je potřeba zadat heslo současné a dvakrát heslo nové.

4.5 Rozhraní pro studenty

V následujících kapitolách je popsáno rozhraní pro studenty. Studenti zde mají základní přehled o svých předmětech, vypracovávají zde samotná cvičení a mohou si zobrazit výsledky již hotových cvičení. Po přihlášení se student dostane na seznam všech svých předmětů. V seznamu je přehledně vidět název předmětu, zkratka předmětu, ve kterém školním roce a semestru je předmět vyučován a v jakém termínu a ve které učebně je cvičení dostupné. U každého předmětu je dále odkaz na seznam aktivních cvičení pro daný předmět.

Uživatel dále v menu vidí, z jakého počítače je aktuálně připojen. Informace je dostupná po najetí myší na jméno aktuálně přihlášeného studenta.

Po kliknutí na toto jméno se student dostane ke změně svého hesla. Při změně musí student zadat také své současné heslo pro kontrolu, že heslo mění právě on.

4.5.1 Cvičení

Nejdůležitější částí studentského rozhraní jsou cvičení. Z menu se student dostane na seznam všech svých aktivních cvičení. V seznamu je zobrazeno, od kdy je dané cvičení dostupné a do kdy je nutné cvičení vypracovat. Dále název předmětu a název samotného cvičení. V případě, že student cvičení již vypracovával, je zobrazen čas poslední aktivity. Následně se student přesune na vypracování cvičení.

Obrázek 10 - Ukázka seznamu cvičení dostupných studentovi

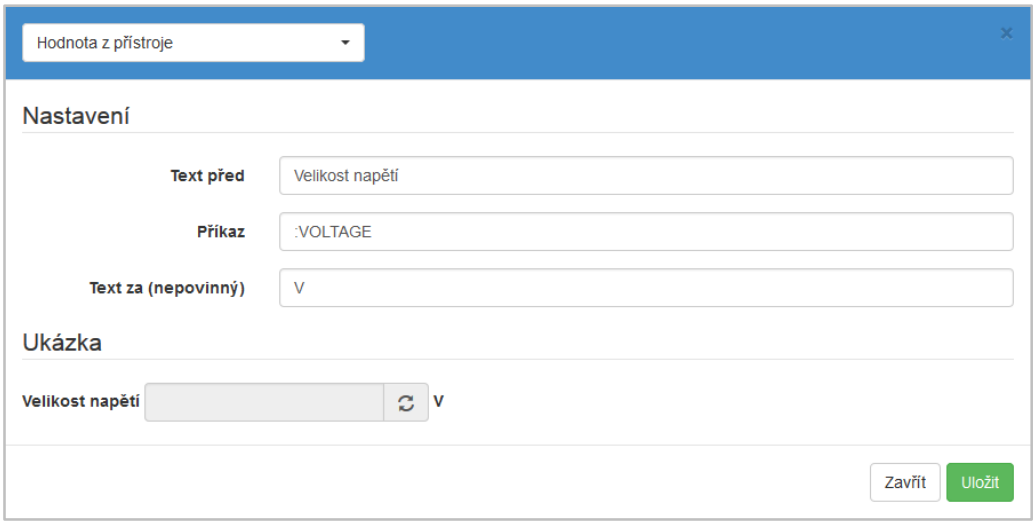
Na stránce se samotným cvičením se studentovi zobrazí po sobě jdoucí jednotlivé prvky, které musí student postupně vyplnit. V horní části stránky si může student stáhnout aktuální verzi programu pro komunikaci se zařízeními. Na konci celé stránky je možnost cvičení uložit. V případě, že student nestihl vypracovat cvičení v dané hodině, veškerá data se uloží a student se ke cvičení může vrátit třeba následující hodinu. Pokud má student celé cvičení vypracované, může toto cvičení natrvalo uzavřít pomocí tlačítka, které nalezne na konci stránky. Pokud jsou vyplněna všechna potřebná data, aplikace je uloží a následně studenta přesměruje na seznam již vypracovaných cvičení, kde si může prohlédnout výsledný protokol o vypracování cvičení. Kdykoliv má tak

své výsledky k dispozici. Tento protokol je studentovi automaticky odeslán také na jeho e-mail.

4.6 Vytváření šablon pro cvičení

Vzhledem k tomu, že je aplikace primárně navržena pro cvičení v rámci předmětu, je nutné tato cvičení také vytvářet. Jednotlivá cvičení se skládají z několika po sobě následujících prvků, šablon. Každou tuto šablonu lze dle typu nastavit a student tak ve výsledku dostane kompletní cvičení k vypracování. Tyto šablony je nutno nejdříve vytvořit.

V aplikaci je navržen a vytvořen systém pro tvorbu a správu šablon cvičení. Tento systém dovoluje budoucí rozvoj aplikace s ohledem na rozšíření také na jiné předměty. Kdykoliv je díky tomu možné rozšířit cvičení o nové šablony prvků. Aby bylo možné tyto nové šablony vytvářet, jsou v systému vytvořeny nástroje, které napomohou v základních operacích.



Obrázek 11 - Přidání nového prvku do cvičení

4.6.1 Základní popis

Základem tvorby šablon je abstraktní třída **Task**. Jak tato třída vypadá, je zobrazeno v **příloze B**. Je to třída, pomocí které budou jednotlivé šablony rozšířeny, a určuje, co vše je potřeba ke správné funkci. Včetně funkcí

povinných pro nové šablony obsahuje ještě další metody potřebné ke správné funkci aplikace. Mezi tyto metody patří **teacherTemplateShow,** přejímající parametry **$template**, **$eventType** a **$settings**, dále **studentTemplateShow,** přejímající parametry **$template**, **$settings** a **$data,** a poslední metodou je **documentTemplateShow,** přejímající parametry **$template**, **$settings** a **$data**. Tyto metody jsou privátní a volány pouze z šablon. Význam přijímaných parametrů bude vysvětlen v následujícím odstavci. Další metoda je veřejná a finální (šablona ji tedy nemůže přepsat) s názvem **studentTemplateDisplay**, díky které se dá zjistit, zda má být studentská šablona zobrazena či nikoliv. Nastavení zobrazení se provádí u každé šablony zvlášť a bude to vysvětleno dále v textu. Další metodou je **getClass**. Tato metoda nám vrací název třídy. V ukázce z následujícího odstavce by to byl název **TaskOscilloscope**. Dále je zde dostupná statická metoda **registerTask(Task $task)**. Touto metodou naši novou šablonu zaregistrujeme. Poslední metodou je **getRegistredTask**, která vrací seznam všech registrovaných šablon.

Aby bylo možné vytvořit novou šablonu, musíme nejprve vytvořit novou třídu, například **TaskOscilloscope**, která bude rozšiřovat třídu **Task.** Tuto třídu následně uložíme do adresáře **/app/class/task/item/** pod názvem **Class.TaskOscilloscope.php**. Aby tato nově vytvořená šablona fungovala správně, musíme implementovat několik metod. V prvé řadě je to metoda **getTaskName()**, která vrací název šablony. V tomto případě „Snapshot z osciloskopu“.

Další metodou je **teacherTemplate** s parametry **$template**, **$eventType** a **$settings**. Tato metoda slouží k zobrazení šablony pro nastavení nového nebo editaci stávajícího prvku cvičení. První parametr je proměnná s šablonou, druhý je typ požadavku (*insert* nebo *update*, tedy přidání nové položky nebo editace stávající) a v posledním parametru je dostupné aktuální nastavení prvku, sloužící při editaci prvku. V metodě je možné s daty různě pracovat

nebo přiřazovat pomocné proměnné do šablony. Na konci metody je nutné zavolat metodu **teacherTemplateShow** a předat jí všechny parametry. Bez toho by nedošlo k zobrazení příslušné šablony.

Metoda **studentTemplate** s parametry **$template**, **$settings** a **$data** slouží k zobrazení studentské šablony. Tedy té šablony, do které bude student vyplňovat své hodnoty. Parametry **$template** a **$settings** obsahují stejná data jako v předešlé metodě. V parametru **$data** jsou předána aktuální vyplněná data v případě, že se student vrátí k vyplňování cvičení třeba příští hodinu. Opět je v metodě možné pracovat s daty a šablonou. Stejně jako u předešlé metody je na jejím konci nutné zavolat metodu pro zobrazení šablony. V tomto případě je to metoda **studentTemplateShow**, které se předají všechny přijaté parametry. Tím dojde k zobrazení samotné šablony v cvičení. Tato metoda je nicméně volána pouze v případě, že je proměnná **$studentTemplateDisplay** nastavena na hodnotu **true**. Tato hodnota je defaultně nastavena, ve třídě ji ale můžeme nastavit na hodnotu **false** a tato šablona se studentovi nezobrazí. Toho se dá využít v případě, že šablona slouží pouze pro dokument. Například pokud chceme v dokumentu ukončit stránku.

Poslední metodou je **documentTemplate** s parametry **$template**, **$settings** a **$data**. Parametry jsou stejné jako v předešlé metodě. Tato metoda slouží k zobrazení šablony pro vygenerovaný PDF výstup ze cvičení. V metodě je možné pracovat s přijatými daty, stejně jako ve dvou předchozích metodách. Na konci této metody, na rozdíl od předešlých metod, navrátíme jako výstup obsah metody **documentTemplateShow**, které předáme přijaté parametry**.**

V ukázce kódu pod tímto odstavcem je nastíněno, jak se nové šablony vytvářejí a jak se pracuje s jednotlivými metodami. Lze si všimnout, že za samotnou třídou je volána statická metoda **registerTask** třídy **Task** a předána

instance nové šablony. Bez toho by aplikace nevěděla, že je daná šablona dostupná.

```php
class TaskOscilloscope extends Task {
    /**
    * navrácení názvu šablony
    * @return string
    */
    public function getTaskName() {
        return "Snapshot z osciloskopu";
    }

    /**
    * metoda pro zobrazení šablony s nastavením
    * @param Smarty $template šablona
    * @param string $eventType insert|update - typ požadavku
    * @param array $settings nastavení šablony
    * @return void
    */
    public function teacherTemplate($template, $eventType, $settings) {
        // možné operace s daty

        // zavolání metody $this->teacherTemplateShow pro zobrazení šablony
        $this->teacherTemplateShow($template, $eventType, $settings);
    }

    /**
    * metoda pro zobrazení šablony studenta
    * @param Smarty $template šablona
    * @param array $settings nastavení šablony
    * @param array $data uložená data studenta
    * @return void
    */
    public function studentTemplate($template, $settings, $data) {
        // možné operace s daty

        // zavolání metody $this->studentTemplateShow pro zobrazení šablony
        $this->studentTemplateShow($template, $settings, $data);
    }

    /**
    * metoda pro zobrazení šablony v dokumentu
    * @param Smarty $template šablona
    * @param array $settings nastavení šablony
    * @param array $data uložená data studenta
    * @return string
    */
    public function documentTemplate($template, $settings, $data) {
        // možné operace s daty

        // navrácení obsahu metody $this->documentTemplateShow pro zobrazení šablony
        return $this->documentTemplateShow($template, $settings, $data);
    }
}

// zaregistrování šablony do aplikace
Task::registerTask(new TaskOscilloscope());
```

Ukázka kódu 3 - Ukázka třídy TaskOscilloscope

Kromě této třídy je nutné vytvořit jednotlivé stránky šablon, které jsou umístěny ve složce **/app/template/task/**. Stránky každé šablony se umístí do složky s názvem dané šablony. V našem případě složka TaskOscilloscope. V této složce budou následně šablony stránek v souborech **teacher.tpl**,

student.tpl a **document.tpl**. Popis jednotlivých šablon je dostupný v následujících kapitolách.

4.6.2 Šablony pro vyučující

Šablona pro vyučující je umístěna v souboru **teacher.tpl**, slouží při vytváření cvičení a obsahuje nastavení dané šablony. Toto nastavení se následně promítne do studentské šablony a do šablony pro dokument. Do šablony jsou předány 2 proměnné, a to **$eventType** a **$settings**. Dle toho je možné šablonu upravit a při editaci vyplnit aktuální nastavení. V šabloně je možné psát jakýkoliv HTML kód s využitím Bootstrap prvků, používat vlastní CSS styly a JavaScriptové funkce.

Aby bylo možné tuto šablonu uložit, je potřeba implementovat JavaScriptovou funkci **task.getData(dialog, eventType)**. Tuto funkci aplikace automaticky zavolá při stisknutí tlačítka k uložení. Aplikace funkci předá v parametru **dialog** kontext dialogového okna pro práci s DOM a dále parametr **eventType** s typem požadavku (*insert* nebo *update*). Funkce vrátí aplikaci objekt s aktuálním nastavením, případně hodnotu **null,** pokud šablonu nelze uložit, například pokud nejsou vyplněna všechna potřebná pole.

```
task.getData = function(dialog, eventType) {
    if(isEmpty($('#pictureText', dialog).val())) {
        $('#pictureText', dialog).addClass("inputError").focus();
        return null;
    }
    $('#pictureText', dialog).addClass("inputError");
    var settings = {
        pictureText: $('#pictureText', dialog).val(),
    };
    return settings;
}
```

Ukázka kódu 4 - Ukázka funkce task.getData(dialog, eventType)

4.6.3 Šablony pro studenty

Studentská šablona je dostupná v souboru **student.tpl** a slouží k vyplňování dat ze strany studenta. Do šablony jsou propsány proměnné

$settings, ve které je nastavení dané šablony, a **$data**, která obsahuje vložená data ze strany studenta. Stejně jako u šablony pro vyučující, i zde je možné psát jakýkoliv HTML kód s využitím Bootstrap prvků, vytvářet vlastní CSS styly a psát JavaScriptové funkce.

Aby mohla aplikace získat data ze studentské šablony, je nutné implementovat JavaScriptovou funkci **task.getData()**. Na rozdíl od stejné funkce v šabloně pro vyučující, této funkci nejsou předány žádné proměnné. Je to z toho důvodu, že studentská šablona je zobrazena v rámu a tím je izolována od samotné stránky. Kontextem je tedy celá studentská šablona. Tato funkce vrací objekt s vyplněnými daty, případně hodnotu **null**, pokud nejsou všechna data korektně vyplněna.

4.6.4 Šablony pro dokument

Šablona pro dokument je v souboru **document.tpl**. Tato šablona se zobrazí ve vygenerovaném PDF s obsahem výsledků celého cvičení. Do šablony jsou, stejně jako u studentské šablony, propsány proměnné **$settings** a **$data**. V této šabloně je možné použít jakýkoliv HTML kód, tentokrát bez využití Bootstrap prvků. Pro úpravu vzhledu je možné použít vlastní CSS styly. JavaScript není možné použít.

4.6.5 JavaScriptové API pro šablony

V aplikaci je dostupné JavaScriptové API, které jednotlivé šablony mohou využít. Většina funkcí je společných pro šablony vyučujícího i studenta. U studentské šablony jsou ale navíc dostupné další funkce. Toto API pokrývá základní práci s událostmi, zařízeními a některé potřebné funkce tak, aby bylo vytváření šablon co nejsnazší.

Funkce **file.uploadImage(elementId, callback(data), multiple)** slouží k nahrání obrázků v šabloně. Jako parametr **elementId** se funkci předá identifikátor políčka s jedním nebo více obrázky (políčko typu file). Dalším

parametrem je **callback**, což je funkce, která se zavolá po nahrání všech obrázků, a té se předá, v závislosti na posledním parametru, Base64 hodnota jednoho obrázku, případně pole Base64 hodnot všech nahrávaných obrázků. Jak bylo již nastíněno, poslední parametr **multiple** určuje, zda se nahrává jeden nebo více obrázků.

Společnou funkcí pro obě šablony je funkce **message.error(text)**. Tato funkce slouží k zobrazení chybové hlášky s obsahem v parametru **text**. Dalšími společnými funkcemi jsou funkce na správu WYSIWYG editoru TinyMCE. Funkce **textarea.init()** slouží k zobrazení editoru. Aby bylo možné data z editoru dostat, je dostupná funkce **textarea.getContent(id[, parentElement])**. Této funkci se předá identifikátor editoru a případně nepovinný parametr kontextu, například pro šablony vyučujících by to byla proměnná **dialog**.

Další funkce, kterou lze využít, je funkce **control.randomString([length[, charSet]])**, sloužící k vygenerování náhodného řetězce o délce 10 znaků. V případě, že je funkci předán nepovinný parametr **length**, může se délka řetězce ovlivnit. Dalším nepovinným parametrem lze určit, jaká abeceda bude použita pro výsledný řetězec. Pro práci s Base64 jsou dostupné funkce **Base64.encode(string)** a **Base64.decode(string)**.

```
$('#myButton').on('click', function() {
    var callback = fucntion(data) {
        $('#myImmage').attr('src', '' + data);
    }
    event.add('COM1', 'DISPLAY', callback);
});
```

Ukázka kódu 5 - Ukázka načtení obrázku po kliknutí na tlačítko

Funkce **event.add(port, command, callback(data))** slouží pro vložení nového dotazu na přístroj. Prvním parametrem je **port**, na který se výsledný příkaz pošle. Dále **command**, což je příkaz, který se zašle a na který bude odpovězeno. Posledním parametrem je **callback(data)**, funkce, která se zavolá při ukončení dotazu na zařízení. Této funkci jsou předána výsledná data.

Důležitou funkcí je **device.get(callback(port))**. Slouží k zjištění, na jaké zařízení a jaký port se má daný příkaz zaslat. Tato funkce vyvolá dialogové okno s výběrem zařízení. Obrázek 5 ukazuje příklad tohoto dialogového okna. Po výběru zařízení funkce zavolá callback funkci a předá jí výsledný port. Tento port je pak následně možné předat například funkci **event.add**.

Poslední funkcí, dostupnou pouze pro studentskou šablonu, je **task.save()** sloužící k uložení aktuálních dat příslušné šablony. Je tak možné provádět automatické uložení dat při změně. Při zavolání této funkce se získají data skrze funkci **task.getData** a následně jsou uložena do databáze. Pokud se data nepodaří uložit, je vyvoláno chybové hlášení.

5 Instalace a zprovoznění aplikace

V následujících kapitolách bude popsána instalace a zprovoznění aplikace na serveru. Tento postup je popsán pro testovací počítač, jehož popis byl v předchozích kapitolách. V budoucnu se počítá s použitím výkonnějšího serveru s předinstalovaným systémem linux. Postup instalace a nastavení webového serveru tak mohou být mírně odlišné.

5.1 Instalace a nastavení webového serveru

Nejdříve se vytvoří adresářová struktura pro budoucí aplikaci. Na disku se vytvoří (C:/) složka **web**. Složka bude obsahovat vše potřebné pro aplikaci. V této složce se vytvoří následující 3 složky. První bude složka **install**, kam se uloží všechny instalační soubory aplikací. Další složkou je **prog**. Do této složky budou následně instalovány aplikace. Poslední složkou je **web**, kde jsou umístěny samotné soubory aplikace.

Apache (při testování ve verzi 2.2.25) se nainstaluje do složky C:/web/prog/Apache2/. Dále je nutné nastavit některé parametry. Nastavení je dostupné v souboru **httpd.conf**. Z důvodu zabezpečení je možné změnit **ServerTokens** na hodnotu **Prod**. Tato hodnota zajistí, že server při vracení HTTP požadavku nebude v hlavičce vracet verzi serveru Apache. Nastavení **ServerSignature** na hodnotu **Off** zajistí minimální identifikaci Apache serveru [19]. Následně je nutné nastavit virtuální hosty [20]. Nový virtuální host se nasměruje na složku, kde bude umístěna aplikace, přesněji na složku **public**.

Skriptovací jazyk PHP (při testování ve verzi 5.4.34) bude instalován do složky C:/web/prog/php/. Nastavení se provádí v souboru **php.ini** [21]. Jako první je to hodnota **short_open_tag**, která se nastaví na hodnotu **Off**. Nebude tak možné používat zkrácený zápis začátku PHP jazyka (<?), ale bude nutné použít jeho delší variantu (<?php). Je to zejména kvůli přehlednosti výsledného

kódu. Dále se nastaví z důvodu zabezpečení **expose_php** na hodnotu **Off**. To zajistí, že se ve vrácených požadavcích nebude ukazovat, že stránky generuje jazyk PHP, ani v jaké verzi. Potencionální útočník tak neví, jaká verze PHP je použita, a nemůže tak využít případnou chybu pouze této verze. Další parametr **session.cookie_httponly** se nastaví na hodnotu **On**. Díky tomu bude SESSION identifikátor dostupný pouze v HTTP požadavku. V případě provozu aplikace pomocí HTTPS je dobré nastavit také parametr **session.cookue_secure** na hodnotu **On**. To zajistí přenos SESSION identifikátoru pouze přes šifrované spojení. Dále je nutné zavést podporu jazyka PHP v **Apache** serveru. To se provede přidáním následujícího řádku do konfiguračního souboru serveru (soubor **httpd.conf**, viz předchozí odstavec): **LoadModule php5_module "C:/web/prog/php/php5apache2_2.dll"**. Musí se také nastavit cesta ke konfiguračnímu souboru PHP. Do nastavení serveru se vloží řádek: **PHPIniDir "C:/web/prog/php"**. Dále je potřeba povolit PHP rozšíření **fileinfo**, **gd2**, **mbstring**, **openssl** a **pdo_mysql**.

Databázový server MySQL (při testování ve verzi 5.5.40) se nainstaluje do složky C:/web/prog/mysql/. U databázového serveru MySQL toho není potřeba nastavit mnoho. Nastavení je dostupné v souboru **my.ini** [22]. Parametr **lower_case_table_names** nastavený na hodnotu **0** zajistí možnost použití velkých písmen v názvu tabulek. Například lze poté použít tabulku s názvem practiceData. Aby bylo možné ukládat v jednom dotazu větší data, je nutné nastavit parametr **max_allowed_packet** na hodnotu například 10 MB. V případě připojování více klientů je možné upravit maximální počet spojení pomocí parametru **max_connections**.

Na serveru bude potřeba také odesílat e-maily. V systému Windows je nutné tuto podporu doinstalovat. Jako vhodná varianta byla zvolena aplikace **sendmail** [23]. Tuto aplikaci stačí rozbalit. V tomto případě do složky **C:/web/prog/sendmail/**. Následně je nutné provázání s PHP. To se provede

nastavením parametru **sendmail_path** v nastavení PHP (soubor php.ini) na hodnotu "**C:/web/prog/sendmail/sendmail.exe -t**". K odesílání bude následně potřeba nastavení samotné aplikace sendmail. A také nějaký mailový server, na který se bude přistupovat a který bude ve výsledku e-maily odesílat. Nastavení je dostupné v souboru **sendmail.ini**. Zde se nastaví adresa SMTP serveru pod položkou **smtp_server**. Dále **smtp_port**, standardně 25 nebo 465 pro šifrovanou verzi. V případě šifrované verze se nastaví také **smtp_ssl**. Pokud server vyžaduje **SMTP autentifikace**, provede se nastavení hodnot **auth_username** a **auth_password**.

Výsledná adresářová struktura ve složce C:/web/ tedy bude následující:

- install
- prog
    - Apache2
    - php
    - mysql
    - sendmail
- web

5.2 Instalace a nastavení aplikace

Instalace aplikace je velice jednoduchá. Stačí nakopírovat zdrojové kódy do určené složky. Následně je potřeba provést nastavení práv složek. Složce *temp* a v ní vnořeným složkám je nutné nastavit práva pro zápis.

Následně se vytvoří databáze určená pro aplikaci spolu s novým uživatelem. Tomuto uživateli se nastaví práva pouze pro danou databázi, přesněji pouze práva **DELETE, INSERT, SELECT, UPDATE a DROP**. Tato práva stačí k plnohodnotnému chodu aplikace. Poté se do nové databáze importují šablony všech tabulek spolu se základními daty. Šablona databáze je dostupná na přiloženém CD.

Základní nastavení aplikace se nachází v souboru **config.php** v adresáři **/app/config/**. V tomto souboru je nastavení připojení k databázi MySQL. Zde se zadají informace o vytvořeném uživateli databáze z předchozího odstavce. V souboru se dále nachází také informativní zprávy, které se zobrazují při provedení různých akcí v aplikaci. Je tak možné některá hlášení aplikace v budoucnu upravit nebo upřesnit jejich význam.

Ostatní nastavení je k dispozici v samotné aplikaci. Nemusí se tak editovat žádný soubor. V databázi je vytvořen uživatel **admin** s heslem **heslo**, pod tímto uživatelem je tak možné se do aplikace přihlásit. Mezi základní nastavení patří odesílací e-mail, rok startu aplikace a odkaz, kde je umístěna aktuální verze programu pro komunikaci.

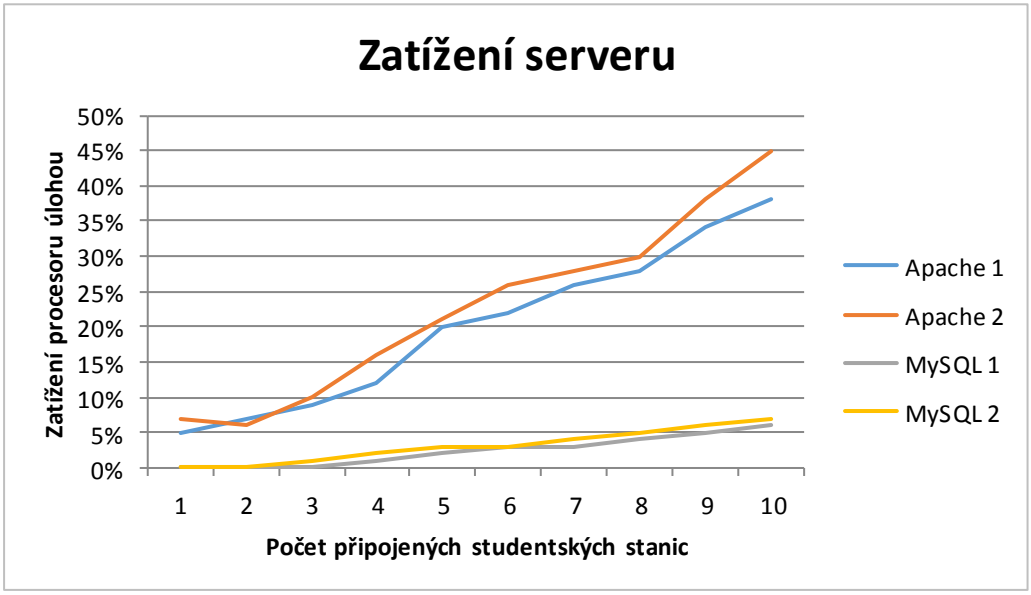
Jako poslední úkon je nastavení automatické úlohy. Je potřeba nastavit, aby se soubor **/app/cron/application-maintenance.php** spouštěl každý den. Spuštění se nastaví na čas, kdy již studenti nebudou měřit, ale server s aplikací pořád poběží. Tento soubor slouží k údržbě databáze, aby nedošlo k nechtěnému zaplnění daty z přístrojů, o které student žádá ve cvičení (snapshot z osciloskopu, případně další hodnoty stahované z přístrojů). V systému Windows k tomuto účelu slouží **Plánovač úloh**, v systémech linux je to **cron**.

6 Realizované experimenty, testování aplikace a výsledky

Aplikace byla v průběhu tvorby dostupná na testovacím počítači. Konfigurace testovacího počítače s operačním systémem Windows 7 byla popsána v kapitole 4. V průběhu tvorby aplikace bylo testováno jeho zatížení. Tabulka 1 s průměry z několika pozorování ukazuje, jak se zatížení serveru vyvíjelo v závislosti na počtu aktivních studentských stanic. Na každém počítači byl spuštěn program pro komunikaci s přístroji, který se dle kapitoly 4.2.3 každých max. 100 ms dotazuje serveru, zda není ve frontě dostupný příkaz k vyřízení. Vždy se měřilo vytížení procesu zajišťujícího chod Apache serveru (httpd.exe – 32 bitová verze) a MySQL databáze (mysqld.exe – 64 bitová verze). Prováděla se dvě měření. Jedno pouze s aktivním programem, bez žádných požadavků ve frontě k vyřízení. Ve druhém měření byla na počítači vyučujícího spuštěna aplikace na stránce, zobrazující obrazovky všech osciloskopů v učebně. Tyto snímky se každých 5 sekund obnovovaly. V následující tabulce a grafu lze pozorovat, že při druhém měření zatížení, kdy programy zasílaly serveru kopie obrazovek osciloskopu, se vytížení mírně zvýšilo. V budoucnu se počítá s využitím výkonnějšího serveru spolu s použitím operačního systému linux. Nový moderní server tak nebude tolik vytížen a zvládne obsloužit více studentských stanic. Vytížení sítě bylo zanedbatelné, nedosahovalo ani 1 % dostupné kapacity gigabitové sítě.

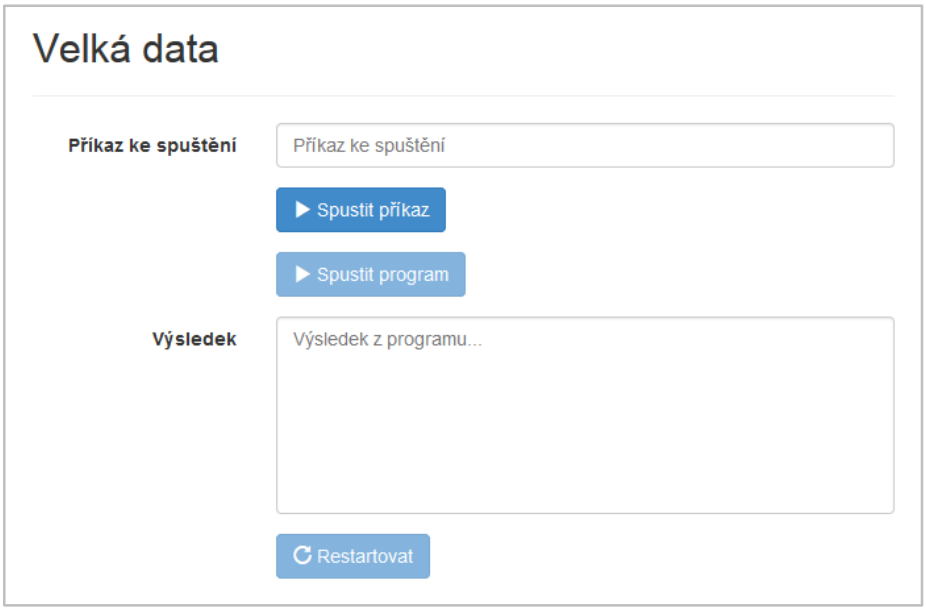
Tabulka 1 - Zatížení serveru v závislosti na počtu aktivních stanic

	1	2	3	4	5	6	7	8	9	10
Apache 1	5%	7%	9%	12%	20%	22%	26%	28%	34%	38%
Apache 2	7%	6%	10%	16%	21%	26%	28%	30%	38%	45%
MySQL 1	0%	0%	0%	1%	2%	3%	3%	4%	5%	6%
MySQL 2	0%	0%	1%	2%	3%	3%	4%	5%	6%	7%



Obrázek 12 - Graf závislosti počtu aktivních stanic na zatížení serveru

Dle zadání bylo úkolem také otestování dané aplikace při realizaci jednoduché úlohy analýzy sériového kanálu, případně jednoduché digitální filtrace. Stahování velkého množství dat, zejména z osciloskopu **Agilent DSO9254A,** je poměrně velice náročné. Stejně tak je ovšem často velmi náročné samotné zpracování takto velkých dat. Ke zpracování velkých dat není PHP, případně JavaScript, příliš vhodné. Pro zpracování je tak lepší využít externích programů napsaných například v jazyce **C/C++**, což je onen klíčový prvek podporovaný vyvinutou aplikací. Vzhledem k tomuto faktu je v systému dostupná speciální sekce, která je přímo určena pro tyto účely. Velká data se neukládají do databáze, nýbrž je pro ně vyčleněn speciální soubor. Do tohoto souboru se uloží všechna binární data přijatá z daného přístroje. Soubor je následně předán externímu programu, který všechna data rychle zpracuje nebo zanalyzuje a následně vygeneruje výstup této analýzy, například v HTML formátu nebo v dalších souborech. Aplikace tento výstup následně zobrazí. Uživatel tak vše ovládá plně dálkově pouze pomocí vyvinuté aplikace.



Obrázek 13 - Ukázka sekce pro zpracování velkých dat

Posledním bodem zadání bylo vytvoření vzorové úlohy v této aplikaci. Byla vybrána jedna úloha z předmětu PMN (Cvičení č. 9 – Základní měření na kitech PMN), který se v učebně vyučuje a pro který byla tato aplikace vytvořena. Tato úloha byla kompletně převedena do vytvořené aplikace a vyzkoušena funkčnost vypracování v učebně AP9.

Obrázek 14 - Ukázka vzorové úlohy z předmětu PMN

7 Závěr

V úvodu této práce byl stanoven cíl v podobě vytvoření aplikace pro zmodernizování výuky v učebně AP9. Tato aplikace měla nahradit papírové protokoly laboratorních cvičení a transformovat je do aplikace, kterou student bude ovládat skrze počítač na studentském pracovišti.

Stanovený cíl byl splněn a výsledkem práce je webová aplikace, kterou student i vyučující ovládají skrze dostupný počítač na pracovišti. Aplikace implementuje základní správu studentů. Dále aplikace obsahuje správu jednotlivých učeben a počítačů v těchto učebnách, spolu s veškerými přístroji na jednotlivých počítačích. Nechybí zde správa všech předmětů spolu s termíny jednotlivých cvičení. Hlavní komponentou aplikace je správa cvičení. V aplikaci byl vytvořen systém vytváření nových šablon cvičení. Do každé šablony cvičení lze za sebe vložit několik stejných nebo různých prvků, které je možné libovolně konfigurovat. Systém těchto prvků je navržen tak, že je kdykoliv možné do aplikace přidat prvek nový. K tomu jsou v systému implementovány prostředky a funkce pro snadnou tvorbu jednotlivých prvků.

V aplikaci je vyřešena komunikace s SCPI kompatibilními přístroji. Tato komunikace je implementována v podobě fronty příkazů pro každý počítač. Program na počítači se dotazuje serveru, jaké příkazy a na jaké zařízení má zaslat, a následně serveru odešle odpověď určeného přístroje. Tato komunikace umožňuje také přenos velkých dat z přístrojů, která mohou být dále analyzována pomocí externího programu.

V práci tak byly splněny všechny body zadání a výsledná aplikace byla otestována přímo v učebně AP9 na reálných přístrojích. Modernizace této učebny je tak dokončena.

Aplikaci je dále možné jednoduše rozšiřovat. Do aplikace je možné vytvořit nové prvky šablon cvičení a díky tomu se využití aplikace rozšíří na nové předměty. Dále je možné rozšiřovat současné funkce systému, například doplněním různých informací k předmětům a jednotlivým cvičením ve stylu e-learningu nebo podporu zařízení, která nejsou SCPI kompatibilní.

Seznam použité literatury

[1] J. Dvořák, P. Kuriščák a F. Lustig, „iSES - Internet School Experimental System: iSES Remote Lab SDK," [Online]. Available: http://www.ises.info/index.php/cs/systemises/sdkisesstudio.

[2] M. Křelína, „Měření VA charakteristik LED diod," 2014. [Online]. Available: http://praktikum.fjfi.cvut.cz/pluginfile.php/2998/mod_resource/content/1/manual-SS-uloha1-v1.pdf.

[3] Gymnázium J. Vrchlického, Klatovy, „Remote-LAB : Vzdálená laboratoř GymKT," [Online]. Available: http://remote-lab.fyzika.net/.

[4] Q-Success, „Usage Statistics and Market Share of Web Servers for Websites, April 2015," 2015. [Online]. Available: http://w3techs.com/technologies/overview/web_server/all.

[5] Q-Success, „Usage Statistics and Market Share of Server-side Programming Languages for Websites, April 2015," 2015. [Online]. Available: http://w3techs.com/technologies/overview/programming_language/all.

[6] solidIT consulting & software development gmbh, „DB-Engines Ranking - popularity ranking of database management systems," 2015. [Online]. Available: http://db-engines.com/en/ranking.

[7] Q-Success, „Usage Statistics and Market Share of Operating Systems for Websites, April 2015," 2015. [Online]. Available: http://w3techs.com/technologies/overview/operating_system/all.

[8] Q-Success, „Usage Statistics and Market Share of Markup Languages for Websites, April 2015," 2015. [Online]. Available: http://w3techs.com/technologies/overview/markup_language/all.

[9] Q-Success, „Usage Statistics of Client-side Programming Languages for Websites, April 2015," 2015. [Online]. Available: http://w3techs.com/technologies/overview/client_side_language/all.

[10] SCPI Consorcium, Standard Commands for Programmable Instruments (SCPI) - Volume 1: Syntax and Style, květen 1999. [Online]. Available: http://www.ivifoundation.org/docs/scpi-99.pdf.

[11] P. Pfeifer, *Inteligentní měřicí pracoviště - Popis řešení modernizace AP9, ESF*, Liberec: Technická univerzita v Libereci, FM ITE, 2012.

[12] P. Pfeifer, *Manuál sady přípravků do cvičení PMN (Pokročilé metody návrhu), ESF*, Liberec: Technická univerzita v Libereci, FM ITE, 2014.

[13] GW Instek, „GDS-2000A Specification," [Online]. Available: http://www.gwinstek.com/en-global/products/Oscilloscopes/Digital_Storage_Oscilloscopes/GDS-2000A.

[14] GW Instek, „GDS-2000A User Manual," [Online]. Available: http://www.gwinstek.com/en-global/products/Oscilloscopes/Digital_Storage_Oscilloscopes/GDS-2000A.

[15] GW Instek, „Programming manual for GDS-2000A," [Online]. Available: http://www.gwinstek.com/en-global/products/Oscilloscopes/Digital_Storage_Oscilloscopes/GDS-2000A.

[16] Agilent Technologies, Inc., „Agilent Infiniium 9000 Series Oscilloscopes - Data Sheet," Červenec 2014. [Online]. Available: http://cp.literature.agilent.com/litweb/pdf/5990-3746EN.pdf.

[17] Agilent Technologies, Inc., „Agilent Infiniium 9000 Series Oscilloscopes - Programmer's Reference," Leden 2014. [Online]. Available: http://www.keysight.com/upload/cmc_upload/All/9000_series_prog_ref.pdf.

[18] J. Bernardi, „Secure Password Hashing with PHP - The Dev Files," 22 8 2014. [Online]. Available: http://www.thedevfiles.com/2014/08/secure-password-hashing/.

[19] The Apache Software Foundation, „core - Apache HTTP Server Version 2.2," 2015. [Online]. Available: http://httpd.apache.org/docs/2.2/mod/core.html#serversignature.

[20] The Apache Software Foundation, „VirtualHost Examples - Apache HTTP Server Version 2.2," 2015. [Online]. Available:

http://httpd.apache.org/docs/2.2/vhosts/examples.html.

[21] The PHP Group, „PHP: List of php.ini directives - Manual,“ 2015. [Online]. Available: http://php.net/manual/en/ini.list.php.

[22] Oracle Corporation, "MySQL :: MySQL 5.5 Reference Manual :: 5.1.4 Server System Variables," 2015. [Online]. Available: https://dev.mysql.com/doc/refman/5.5/en/server-system-variables.html.

[23] J. Byron, "Sendmail," 2015, [Online]. Available: http://glob.com.au/sendmail/.

Seznam příloh

Příloha A – CD-ROM

Obsahem přiloženého CD-ROM jsou následující složky:

- diplomová práce – obsahuje PDF verzi diplomové práce
- zdrojové kódy – obsahuje zdrojové kódy celé aplikace
- databáze – obsahuje šablonu databáze

Příloha B – abstraktní třída pro správu šablon cvičení

```php
abstract class Task {
    static private $registredTask = array();

    protected $studentTemplateDisplay = true;

    protected abstract function getTaskName();
    protected abstract function teacherTemplate($template, $eventType, $settings);
    protected abstract function studentTemplate($template, $settings, $data);
    protected abstract function documentTemplate($template, $settings, $data);

    protected final function teacherTemplateShow($template, $eventType, $settings) {
        $template->assign("eventType", $eventType);
        $template->assign("settings", $settings);

        $template->display("task/".$this->getClass()."/teacher.tpl");
    }

    protected final function studentTemplateShow($template, $settings, $data) {
        $template->assign("settings", $settings);
        $template->assign("data", $data);

        $template->display("task/".$this->getClass()."/student.tpl");
    }

    protected final function documentTemplateShow($template, $settings, $data) {
        $template->assign("settings", $settings);
        $template->assign("data", $data);

        return $template->fetch("task/".$this->getClass()."/document.tpl");
    }

    public final function studentTemplateDisplay() {
        return $this->studentTemplateDisplay;
    }

    public final function getClass() {
        return get_class($this);
    }

    static public final function getRegistredTask() {
        return Task::$registredTask;
    }

    static public final function registerTask(Task $task) {
        $taskFind = false;

        foreach(Task::$registredTask as $_task) {
            if($_task->getClass() === $task->getClass()) {
                $taskFind = true;

                break;
            }
        }
        if(!$taskFind) {
            Task::$registredTask[] = $task;
        }
    }
}
```

Příloha C – JavaScriptové funkce obsluhující dotazy na zařízení

```
var event = {
    list: [],
    add: function(port, command, callback, fakeIp, bigData) {
        if(port === undefined || command === undefined || callback === undefined)
            return false;
        if(bigData === undefined || bigData !== true)
            var bigData = false;
        if(fakeIp === undefined || fakeIp === false)
            var fakeIp = undefined;
        $.ajax({
            url: apiUrl + "event/insert/",
            data: {
                port: port,
                command: command,
                fakeIp: fakeIp,
                bigData: bigData === true ? 1 : undefined,
            },
            type: "GET",
            success: function(data) {
                event.list.push({
                    id: data,
                    callback: callback,
                    checkCount: bigData ? -1 : 0,
                    fakeIp: fakeIp,
                });
            },
        });
    },
    foreach: function() {
        $.each(event.list, function(index, value) {
            $.ajax({
                url: apiUrl + "event/check/" + value.id + "/",
                data: {
                    fakeIp: value.fakeIp,
                },
                type: "GET",
                success: function(data) {
                    if(parseInt(data.return) == 1) {
                        value.callback(data.returnData);
                        event.remove(value.id);
                    } else {
                        if(value.checkCount >= 0) {
                            value.checkCount++;
                        }
                    }
                    if(value.checkCount > 60) {
                        event.remove(value.id);
                    }
                },
                dataType: "json",
            });
        });
    },
    remove: function(id) {
        var _eventList = [];
        $.each(event.list, function(index, value) {
            if(value.id != id) {
                _eventList.push(value);
            }
        });
        event.list = _eventList;
    },
};
setInterval(function() {
    event.foreach();
}, 1000);
```