Školní rok 2023/2024

Jan Novotný (I3C), Patrik Švehelka (I3C), Vendula Šteffková (I3B)

Spšei ostrava

Helpdesk – závěrečný projekt předmětů WEA a DAS

Dokumentace webové aplikace

Obsah

[Úvod 2](#_Toc168258524)

[I. Cíl práce 2](#_Toc168258525)

[Rozdělení úkolů pro jednotlivé členy týmu 3](#_Toc168258526)

[Teoretická část 4](#_Toc168258527)

[I. Návrh databáze 4](#_Toc168258528)

[II. Programování 4](#_Toc168258529)

[Praktická část 5](#_Toc168258530)

[I. Databáze 5](#_Toc168258531)

[i. E-R model 5](#_Toc168258532)

[ii. Popis tabulek 5](#_Toc168258533)

[II. Struktura aplikace 9](#_Toc168258534)

[III. Program 11](#_Toc168258535)

[iii. Popis vybraného PHP kódu 11](#_Toc168258536)

[iv. Stručný popis použité základní HTML struktury 12](#_Toc168258537)

[v. Základní rozbor zvoleného CSS layoutu 13](#_Toc168258538)

[Stručný manuál 14](#_Toc168258539)

[I. Přihlášení 14](#_Toc168258540)

[II. Žádost o vytvoření účtu 14](#_Toc168258541)

[III. Zákaznické rozhraní 15](#_Toc168258542)

[i. Zprávy 16](#_Toc168258543)

[ii. Odesílání ticketů 17](#_Toc168258544)

[iii. Zákaznický profil 17](#_Toc168258545)

[IV. Administrační rozhraní 18](#_Toc168258546)

[iv. Stránky typu ‚See all ...‘ 19](#_Toc168258547)

[v. Stránky typu ‚Create ...‘ 19](#_Toc168258548)

[vi. Vyhledávací pole 20](#_Toc168258549)

[Literatura 21](#_Toc168258550)

[Seznam obrázků 22](#_Toc168258551)

# Úvod

Všichni jsme již pravděpodobně setkali s nějakými technickými potíži, jako může být např. nestabilní připojení k internetu, účet, ke kterému se nelze přihlásit či předplatné, co nám stále „strhává“ peníze z účtu. Ještě větší problém však může nastat, když danou technickou obtíž chceme řešit, ale daná společnost nemá funkční telefonickou nebo jinou technickou podporu pro zákazníky, pokud se dá vůbec kontaktovat. S tímto problémem nám může pomoci Helpdesk. Jedná se o online webovou aplikaci, kam se zákazníci mohou obrátit se svými technickými problémy pomocí tickets a příslušní správci jim s problémy pomohou.

Projekt Helpdesk se zaměřuje na webovou aplikaci dělanou v jazyce PHP s připojenou MySQL databází. Zákazník si při prvotní návštěvě aplikace zažádá o vytvoření účtu, to mu můžou správci buďto povolit, nebo žádost smazat. Pokud dojde k vytvoření účtu, zákazník se může přihlásit a odesílat tzv. tickets. Jednoduše řečeno, tickets jsou zprávy nebo žádosti o pomoc s konkrétním problémem. Správci pak mohou na jednotlivé tickets odpovídat a řešit daný problém, nebo je mohou smazat. Celá aplikace je v anglickém jazyce a je stavěná tak, aby jednotlivá oddělení firmy měla přístup pouze k těm tickets, které jsou schopna řešit.

Toto téma jsme si vybrali proto, jelikož jsme tuto aplikaci dělali v rámci studentské zahraniční mobility Erasmus+ v Portugalsku. Aplikaci jsme již v té době měli téměř hotovou, a proto nám připadalo zbytečné začít dělat zcela novou aplikaci, a tak jsme pouze dodělali a upravili naši aplikaci tak, aby splňovala zadání.

**Klíčová slova:** databáze, programování, webová aplikace, helpdesk

## Cíl práce

Cílem naší práce je naprogramovat funkční webovou aplikaci v jazyce PHP a navrhnout a realizovat funkční MySQL databázi, se kterou bude webová aplikace pracovat. Mezi základní požadavky webové aplikace patří použití objektově orientovaného programování, užití jazyků PHP, HTML, CSS a databázového systému MySQL a umístění aplikace na veřejně dostupný webový server. Zároveň je naším úkolem mít alespoň tři tabulky v databázi, zajistit kompletní správu datových objektů prostřednictvím administračního rozhraní a zajistit přihlašování uživatelů.

# Rozdělení úkolů pro jednotlivé členy týmu

Jan Novotný:

* Návrh a tvorba databáze,
* zobrazení dat v databázi,
* tvorba filtrů pro zobrazení dat,
* realizace komunikace mezi zákazníky a správci.

Patrik Švehelka:

* Přihlašování uživatelů,
* zobrazení dat v databázi,
* ukládaní dat do databáze,
* úprava kódu na objektově orientované programování.

Vendula Šteffková:

* Tvorba stylů a úpravy vzhledu,
* tvorba search barů,
* tvorba dokumentace.

# Teoretická část

## Návrh databáze

Pojmem databáze rozumíme systém informací s pevně danou strukturou. Databáze může být jak 3D (např. kartotéka u lékaře), tak i v elektronické podobě, které jsou dnes nejvíce rozšířené. V databázi se nacházejí tzv. tabulky, což je jedním ze základních databázových objektů a slouží k ukládaní dat (informací). Takovou tabulku si můžeme představit jako tabulku v programu Microsoft Excel, kde je pevně daný počet sloupců, které tvoří jednotlivé položky, tzv. atributy. Konkrétní informace se již zapisují do řádků. Díky tomu má každý záznam stejný počet položek stejného datového typu, a tak se nemůže stát, aby dva různé záznamy v tabulce měly jiný počet položek nebo aby jedna položka obsahovala dva různé datové typy[1]. Databázové systémy jsou mnohem komplexnější, než je tato definice a váže se k nim mnoho dalších věcí, ale pro účely tohoto projektu tato jednoduchá definice stačí.

Databáze je celá podstata našeho projektu, proto se musela nejprve udělat ona, a až po tom jsme mohli začít pracovat na tvorbě webu. První návrhy databáze probíhaly pouze na papír, viz obrázek 1, a to ve formě E-R modelu. Návrh databáze dělal Jan, později dělal i databázi samotnou. V první verzi databáze bylo pouze sedm tabulek, ale ve výsledné databázi je tabulek devět, viz obrázek 2, a to z toho důvodu, jelikož jsme neměli vyřešený způsob, jakým se bude na jednotlivé tickets odpovídat.

V jedné z prvních verzí databáze byly tabulky Users a Admins spojené do jedné tabulky. Typ účtu se dále rozlišoval pomocí atributu ‚type‘, ale od tohoto se později z důvodu přehlednosti upustilo.

## Programování

Programování je klíčovým pojmem v oboru informatiky. Jedná se o proces začínající návrhem řešení problému a končí spustitelným programem. Součástí tohoto procesu bývá analýza a pochopení problému, nalezení algoritmu (postupu) a vytvoření zdrojového kódu v cíleném programovacím jazyce[2]. V dnešní době existuje již mnoho programovacích jazyků a nástrojů, a každý se může hodit na něco jiného.

Celou webovou aplikaci jsme původně programovali strukturovaným programováním, ale kvůli zadání jsme byli nuceni přejít na objektově orientované programování. Jako primární programovací jazyk bylo použito PHP, avšak jsme využili i trochu JavaScript. Web jako takový je dělán pomocí HTML, vzhled následně pomocí CSS, u čehož jsme využili veřejně dostupnou sadu kaskádových stylů od Bootstrapu[3].

Ze začátku jsme se museli rozmýšlet, jak bude celá aplikace vypadat, a to jak frontend, tak backend. Backend byl po chvíli jasný – budeme ho dělat strukturovaně a budeme využívat funkce, které si vytvoříme. Frontend ale byl o něco horší. Nebyli jsme si zcela jistí tím, jak chceme, aby výsledná aplikace vypadala. Nakonec jsme se ale rozhodli ‚zrecyklovat‘ design jedné z Patrikových předešlých prací, jen bylo potřeba pozměnit barvy. Po obdržení zadání jsme ovšem museli aplikaci předělat, aby byla udělána objektově orientovaným programováním (oop).

# Praktická část

## Databáze

Jak již bylo řečeno, naše databáze se skládá z devíti tabulek. Databáze je dělána na MySQL a byla vytvořena v programu MySQL Workbench. Po ztráty přístupu k databázi firmy, u které jsme byli v rámci programu Erasmus+ jsme se přesunuli do prostředí phpMyAdmin.

### E-R model

Obsah obrázku text, snímek obrazovky, Písmo, číslo

Popis byl vytvořen automatickyE-R model nebo také E-R diagram je schéma obecné databáze, díky kterému by měl být kdokoliv schopný vytvořit databázi v jakémkoliv programu, viz obrázek 1.

Obrázek 1: E-R model

### Popis tabulek

Nyní bychom si popsali jednotlivé tabulky, k čemu slouží, jejich atributy a jejich relace.

#### Tabulka users

Slouží k uchovávání informací o zákaznících. Díky této tabulky jsou zákazníci schopni se přihlásit ke svému účtu a odesílat z něj tickets.

Primární klíč:

* userId – identifikátor zákazníka, celé číslo

Cizí klíče:

* žádné

Další atributy:

* userName – křestní jméno zákazníka, varchar
* userSurname – příjmení zákazníka, varchar
* userEmail – email zákazníka, varchar, unikátní
* userPasswd – heslo zákazníka zašifrováno pomocí sha256, varchar

#### Tabulka admins

Tabulka slouží, obdobně jako u tabulky users, k uchovávání informací, ale ne o zákaznících, nýbrž o správcích. Díky této tabulky se může správce přihlásit a spravovat tickets, zákaznické účty, zprávy, ale i oddělení a případně i podřízené správce.

Primární klíč:

* adminId – identifikátor správce, celé číslo

Cizí klíče:

* žádné

Další atributy:

* adminName – křestní jméno správce, varchar
* adminSurname – příjmení správce, varchar
* userEmail – email správce, varchar, unikátní
* userPasswd – heslo správce zašifrováno pomocí sha256, varchar

#### Tabulka requests

Tabulka requests je jediná tabulka, která není propojená s žádnou jinou tabulkou. Je to z toho důvodu, že to zkrátka není potřeba. Do tabulky se uloží informace o zákazníkovi, který si přeje vytvořit účet. Pokud některý ze správců schválí vytvoření účtu, konkrétní záznam se pomocí PHP kódu zkopíruje do tabulky users a poté se daný záznam z tabulky requests vymaže.

Primární klíč:

* requestId – identifikátor zákazníka žádajícího o účet, celé číslo

Cizí klíče:

* žádné

Další atributy:

* reqName – křestní jméno zákazníka žádajícího o účet, varchar
* reqSurname – příjmení zákazníka žádajícího o účet, varchar
* reqEmail – email zákazníka žádajícího o účet, varchar, unikátní
* reqPasswd – heslo zákazníka žádajícího o účet zašifrováno pomocí sha256, varchar

#### Tabulka tickets

Do tabulky tickets se ukládají informace o zákazníkovi, který danou ticket odeslal, díky čemuž je tabulka propojená s tabulkou users. Následně se zde ukládají informace jako název problému, popis problému, o jaký druh problému se jedná a datum. Dále se zde nachází atribut status, který, jak z jeho názvu vypovídá, nám sděluje aktuální status ticketu. Všechny nově příchozí tickets mají automaticky nastavený status Waiting, pokud na ně správce zareaguje, tak se jim nastaví status Pending, a pokud již jsou vyřešené, nastaví se jim status resolved. S tímto souvisí i atribut resolver, což je ve zkratce identifikátor správce, který konkrétní ticket řešil, což zajišťuje spojení tabulky s tabulkou admins. Dále se zde také nachází atribut ticketTypeId, což je atribut propojující tabulky tickets a ticket\_types, a díky kterému můžeme přiřadit ticketům jejich konkrétní typy.

Primární klíč:

* ticketId – identifikátor ticket, celé číslo

Cizí klíče:

* resolver – identifikátor správce, který na danou ticket odpověděl, celé číslo, nulový
* userId – identifikátor zákazníka, který danou ticket odeslal, celé číslo
* ticketTypeId – identifikátor typu ticket, celé číslo

Další atributy:

* title – nadpis ticket, varchar
* status – status ticket, enum (‚Waiting‘, ‚Pending‘, ‚Resolved‘)
* ticketDesc – popis konkrétního problému, longtext
* ticketDate – datum, kdy byla ticket vytvořena, datum

#### Tabulka ticket\_types

Tabulka ticket\_types slouží jako propojovací tabulka. Spojuje tabulky tickets a departments, díky čemuž můžeme konkrétním oddělením přiřadit jejich specifické typy tickets, které jsou schopna řešit (byla by hloupost, kdyby např. zaměstnanec marketingového oddělení měl řešit ticket zabývající se špatným připojením k internetu).

Primární klíč:

* ticketTypeId – identifikátor typu ticket, celé číslo

Cizí klíče:

* departmentId – identifikátor oddělení, ke kterému se daný typ ticket vztahuje, celé číslo

Další atributy:

* ticketTypeName – název typu ticket, varchar

#### Tabulka departments

V tabulce departments se nachází pouze identifikátor oddělení a jeho název. Nebylo totiž potřeba více atributů. Nová oddělení lze samozřejmě vytvářet podle potřeb firmy.

Primární klíč:

* departmentId – identifikátor oddělení, celé číslo

Cizí klíče:

* žádné

Další atributy:

* departmentName – název konkrétního oddělení, varchar

#### Tabulka department\_lists

Tabulka department\_lists slouží k propojení tabulek departments a admins. Cílem této relace je vyřešit vztah M:N mezi zmíněnými tabulkami, a to proto, že zaměstnanec může pracovat na vícero odděleních a zároveň každé oddělení může mít vícero zaměstnanců.

Primární klíč:

* departmentId – identifikátor oddělení, celé číslo
* adminId – identifikátor správce, který pracuje na konkrétním oddělení, celé číslo

Cizí klíče:

* departmentId – identifikátor oddělení, celé číslo
* adminId – identifikátor správce, který pracuje na konkrétním oddělení, celé číslo

Další atributy:

* žádné

#### Tabulka messages

Tato tabulka slouží k ukládaní všech odeslaných zpráv, a to jak od zákazníka, tak od správce. Každá zpráva se musí vztahovat ke konkrétní konverzaci. To kdo zprávu odeslal (správce nebo zákazník) se určí pomocí programu a následně se identifikátor konkrétního uživatele uloží do příslušného atributu (senderUserId nebo senderAdminId).

Primární klíč:

* msgId – identifikátor zprávy, celé číslo

Cizí klíče:

* senderUserId – identifikátor zákazníka, co zprávu odeslal, celé číslo, nulový
* senderAdminId – identifikátor správce, co zprávu odeslal, celé číslo, nulový
* conversationId – identifikátor konverzace, ke které se zpráva vztahuje, celé číslo

Další atributy:

* msgContent – obsah zprávy, longtext
* userReplied – informace o tom, zdali již zákazník odpověděl, tinyint, nulový
* adminReplied – informace o tom, zdali již správce odpověděl, tinyint, nulový

#### Tabulka conversation

Do této tabulky jsou ukládány informace o tom, jaký zákazník a správce spolu komunikují ohledně konkrétní ticket. Dalo by se tedy říct, že tabulka slouží jako propojovací tabulka.

Primární klíč:

* convoId – identifikátor konverzace, celé číslo

Cizí klíče:

* userId – identifikátor zákazníka, ke kterému se konverzace vztahuje, celé číslo
* adminId – identifikátor správce, ke kterému se konverzace vztahuje, celé číslo
* ticketId – identifikátor ticket, ke které se konverzace vztahuje, celé číslo

Další atributy:

* žádné

## Struktura aplikace

Aplikace je uložená v adresáři helpdesk. Zde se nacházejí další dva adresáře – src a img. V adresáři img se nachází logo firmy, u které jsme byli v rámci projektu Erasmus+. Toto logo je použito v navigaci na všech stránkách, kde se navigace nachází. Dále pak v adresáři src se nachází celý zdrojový kód (19 souborů s příponou \*.php) a další čtyři adresáře – css, js, classes a sql.

V adresáři css se nachází čtyři soubory kaskádových stylů a jejich úkolem je úprava vzhledu stránky tak, aby vypadala přesně dle našich představ. Tyto soubory jsou admin\_style.css, styles.css, searchbar.css a daterange.css. Soubory admin\_style.css a style.css jsou si do jisté míry podobné, a to v tom, že mají velký dopad na celkový vzhled aplikace, každý soubor však upravuje její jinou část. Soubor admin\_style.css upravuje vzhled administračního rozhraní, zatímco soubor style.css upravuje vzhled rozhraní zákazníků. Dále pak soubory daterange.css a searchbar.css upravují vzhled filtrů v administračním rozhraní. Jak již z jejich názvů vyplývá, soubor searchbar.css upravuje vzhled vyhledávacího pole ve filtrech se seznamem a soubor daterange.css upravuje vzhled filtru, pro výběr konkrétních dat.

V adresáři js se nachází pět souborů obsahující kód v jazyce JavaScript. Soubor admin\_script.js dělá několik věcí, např. to, že se správce před smazání ticketu zeptá, zdali si je jistý, že chce ticket smazat. Soubor script.js pracuje, obdobně jako to je u kaskádových stylů, na straně zákazníka, a to s navbarem – soubor se stará pouze o drobné věci ohledně funkčnosti navbaru. Soubor jquery.js dělá jen jedu věc, a tou je kontrola, zdali správce vybral oddělení při vytváření nového administračního účtu. Soubor calendar.js obsahuje zdrojový kód pro filtr pomocí data. Nakonec soubor passwordEye.js, v tom se nachází skript pro zobrazení hesla na stránce Login (index.php).

Adresář sql toho moc neobsahuje, obsahuje přesněji pouze jeden soubor, a to je export celé naší databáze.

Adresář classes obsahuje 13 souborů s přípono \*.php. Jedná se o soubory s třídama, které jsme si vytvořili, a které v našem projektu používáme. Souborů je tolik, jelikož obecně se doporučuje dávat do každého souboru jednu třídu, tím pádem používáme v našem projektu 13 vlastoručně vytvořených tříd. Bylo by zdlouhavé je zde všechny rozebírat, ale ve zkratce každá třída zajišťuje chod jiné části aplikace, např. připojení k databázi, komunikaci mezi správci a zákazníky nebo správu dat o odděleních.

Nyní si stručně popíšeme jednotlivé soubory zdrojového kódu (abecedně):

* admin\_header.php – header s navigací pro administrační rozhraní,
* admin\_messages.php - zde si správce bude moct prohlédnout všechny své přijaté zprávy od zákazníků, na které ještě nezareagoval
* admin\_page.php – stránka, kam bude správce po přihlášení přesměrován; obsahem stránky jsou všechny aktuálně nevyřízené žádosti o účet,
* admin\_tickets.php – stránka, kde jsou zobrazeny všechny tickets; správce si je může filtrovat pomocí několika filtrů, může na ně odpovídat (pokud mají status Waiting) anebo je mazat,
* create\_admin.php – na této stránce může správce vytvářet nové účty administračního rozhraní,
* create\_department.php – zde správce může vytvářet nová oddělení,
* create\_tickets.php – díky tomuto souboru správce má možnost vytvářet nové typy tickets pro konkrétní oddělení,
* departments.php – na této stránce je správce schopen si zobrazit veškerá existující oddělení, je schopen jim změnit typy ticketů, které se k oddělení vztahují nebo i změnit konkrétní správce, kteří na oddělení pracují,
* footer.php – tento soubor je automaticky připojován ke všem souborům, kde je připojen i soubor header.php nebo admin\_header.php a obsahem tohoto souboru je pata (footer) stránky,
* header.php – funguje obdobně jako soubor admin\_header.php až na to, že je tento soubor použit na rozhraní pro zákazníky,
* home.php - stránka, kam bude zákazník po přihlášení přesměrován; obsahem stránky jsou všechny tickety odeslané přihlášeným zákazníkem,
* index.php – stránka, kde se zákazníci nebo správci mohou přihlásit a odkud můžou být zákazníci přesměrováni na stránku register.php,
* logout.php – stránka, přes kterou budou správci i zákazníci přesměrováni na index.php poté, co se odhlásí z účtu,
* register.php – stránka, kde si zákazníci mohou zažádat o vytvoření účtu,
* searchbar.php – stránka, kam budou správci přesměrováni poté, co se pokusí něco vyhledat ve vyhledávacím poli v navigaci, a kde se jim zobrazí výsledky jejich vyhledávání,
* tck\_types.php – na této stránce jsou správci schopni si zobrazit, upravit, nebo smazat typy ticketů,
* user\_messages.php – zde si zákazník bude moct prohlédnout všechny své přijaté zprávy od správců, na které ještě nezareagoval,
* user\_send.php – na této stránce se nachází formulář pro odesílání ticketů, dostupný pouze pro zákazníky,
* users.php – stránka, kde si správci mohou zobrazit všechny stávající účty zákazníků.

## Program

### Popis vybraného PHP kódu

Aplikace je psána v jazyce PHP pomocí objektově orientovaného programování. Navíc je propojena s databází. Pro ošetření SQL injekcí (SQL injections) nepíšeme SQL dotazy „typicky“, ale na místa, kde by se tradičně napsala proměnná, je napsán místo ní otazník. O řádek níže je vždy uveden příkaz $stmt->bind\_param(). v parametrech funkce bind->param() je vždy nejprve napsán počet parametrů a jejich datový typ a za tím jsou vypsány právě ty proměnné, které bychom napsali přímo do SQL dotazu. Tímto se tyto proměnné „doplní“ do SQL dotazu (tzv. se svážou) a my tak můžeme bezpečně odeslat (nebo vypsat) data z databáze.

Pro popis vybraného kódu jsme vybrali úryvek kódu ze souboru TicketManager.php. Jedná se o soubor třídy TicketManager a jeho cílem je definice této třídy a jejích metod. Jelikož třída je na 83 řádků, rozhodli jsme se popsat pouze začátek její definice a dvě její metody.

<?php

- začátek PHP kódu

class TicketManager {

- začátek definice třídy

private $conn;

- deklarace soukromé proměnné conn

public function \_\_construct($db) {

- definice konstruktoru třídy s parametrem db

$this->conn = $db; }

- nastavení hodnoty proměnné db do proměnné conn

public function getTicketsByUserId($userId) {

- definice funkce (metody) getTicketsByUserId s parametrem userId (identifikátor zákazníka)

$stmt = $this->conn->prepare("SELECT \* FROM tickets WHERE userId = ?");

- příprava SQL dotazu pro výpis všech ticketů od zákazníka, jehož identifikátor je parametrem funkce

$stmt->bind\_param('i', $userId);

- svázání identifikátoru uživatele s parametrem v SQL dotazu

$stmt->execute();

- vykonání připraveného SQL dotazu

return $stmt->get\_result(); }

- vrácení výsledku SQL dotazu, zdali byl proveden v pořádku, nebo jestli došlo k chybě

...

- pokračování kódu

public function sendTicket($title, $type, $description, $date, $userId) {

- definice funkce (metody) sendTicket s parametry title (název/nadpis ticketu), type (typ ticketu), description (popis ticketu), date (datum vytvoření ticketu) a userId (identifikátor zákazníka)

$stmt = $this->conn->prepare("INSERT INTO `tickets` (`title`, `status`, `ticketDesc`, `ticketDate`, `userId`, `ticketTypeId`) VALUES (?, 'Waiting', ?, ?, ?, ?)");

- příprava SQL dotazu pro vložení dat z parametrů do tabulky tickets

$stmt->bind\_param("sssii", $title, $description, $date, $userId, $type);

- svázání parametrů SQL příkazu s parametry metody

return $stmt->execute(); }

- vrázení výsledku, zdali byl SQL dotaz proveden v pořádku, nebo jestli došlo k chybě

...

- pokračování PHP kódu

}

- konec definice třídy

?>

- konec PHP kódu

### Stručný popis použité základní HTML struktury

Jako každá webová stránka, i všechny stránky našeho projektu začínaly tagy <!DOCTYPE html> a <html>. V tomto tagu se vždy nacházely tagy <head> a <body>, ale s tím, že obsah tagů se na každé stránce měnil. Avšak na všech stránkách byl tag <head> velice podobný, v jeho obsahu se měnil pouze tag <title> a styly, které jsme ke stránkám připojovali. V tagu <head> se pak vyskytují nepárové tagy <meta> a <link> a párový tag <title>. Tag <meta> slouží k definování metadat o souboru a tag <link> slouží k připojení CSS souborů. Účel tagu <title> je definování titulku nebo názvu stránky. Tento text se také zobrazuje na záložce stránky v poli panelů v prohlížeči. Tagy <meta> a <title> jsou povinné a bez nich bychom se neobešli.

S tagem <body> už to tak jednoduché nebylo, jelikož každá stránka má zcela jiný kontent. Některé části si ale podobné byly. Napříhlad tagy <header> a <footer> celé jejich obsahy jsou na všech stránkách stejné, a to proto, že jsme udělali pro header a footer zvlášť soubory a na každou stránku je akorát připojíme. Je to tak nejen jednodušší, ale i přehlednější, jelikož člověk při každé úpravě navigace nemusí upravovat úplně všechny soubory, ale jen jeden.

Jednotlivé prvky máme na stránkách přehledně a prakticky rozmístěny do tagů <div>. Jedná se asi o nejčastěji se vyskytující tag na našich stránkách. Dále si stránky pro přehlednost dělíme i do částí pomocí tagu <section>. Zobrazovaný text/data z databáze vypisujeme do odstavců (tag <p>) a pro všechny formuláře používáme tag <form>. Ve formulářích používáme párové tagy jako jsou <textarea>, <select> nebo <option> a nepárové tagy jako jsou <input> nebo <button>. Hodně se v našem kódu objevovaly i tagy <script> pro kód jazyku JavaScript.

Další často používané tagy jsou <h1>, což je tag používaný pro nadpisy první úrovně. Jako další je tag <span>, který je použit většinou v tazích <p> pro změnu barvy textu, popř. pro zvýraznění textu.

### Základní rozbor zvoleného CSS layoutu

Co se našeho CSS layoutu týče, nechali jsme se inspirovat Bootstrapem[3]. Nedělali jsme to však pomocí Bootstrapu jako takového, jen jsme se částečně inspirovali jeho designem. Velkou část CSS, ne-li téměř všechno, jsme si dělali sami.

Rozhodli jsme se použít pouze světlý barevný motiv, a to proto, že sám o sobě vypadá elegantně. Jako primární barvy jsme se rozhodli použít bílou, černou, světle šedou a specifický odstín oranžové (#F08963). Tento odstín jsme se rozhodli použít z toho důvodu, že je použit i v logu firmy, u které jsme byli v rámci programu Erasmus+. Ovšem použili jsme i další barvy, např. červenou na mazací tlačítka atd.

V administračním rozhraní jsme se rozhodli zvolit jednoduchý, ale elegantní design s navigací v horní části stránek. V této navigaci se nachází logo firmy Techbase, dropdown seznamy, vyhledávací pole a ikonu profilu. Celá navigace je sjednocena jemným stínem. Stránku profilu jsme nahradili pouze rozbalovacím polem, které se rozbalí po kliknutí na ikonu profilu. Ve spodní části se vyskytuje bílý footer, který obsahuje copyright. Toto rozložení je velice časté u webových stránek. Skoro by se až dalo říci, že se jedná o standard – header/navigace v horní části stránky a footer ve spodní.

Na všech stránkách, kde vypisujeme data z databáze, jsme se rozhodli data vypisovat do přehledných „karet“. Každá karta se skládá z bílého pozadí a šedých polí, kde se zobrazují jednotlivé informace. K tomu jsou pro přehlednost všechny konkrétní data zvýrazněna oranžovou barvou. Na konci karty se nachází vždy mazací tlačítko, které kartu vymaže. Dále se tam pak mohou vyskytovat i tlačítka pro editaci karty nebo pro odeslání zprávy zákazníkovi. Krom toho se na některých stránkách vyskytují filtry pro filtrování karet. Na stránkách s formuláři pro odeslání dat je pro změnu pouze jedna karta, a ta má šedé pozadí a bílá pole. Samozřejmě nesmí chybět oranžové tlačítko pro odeslání formuláře.

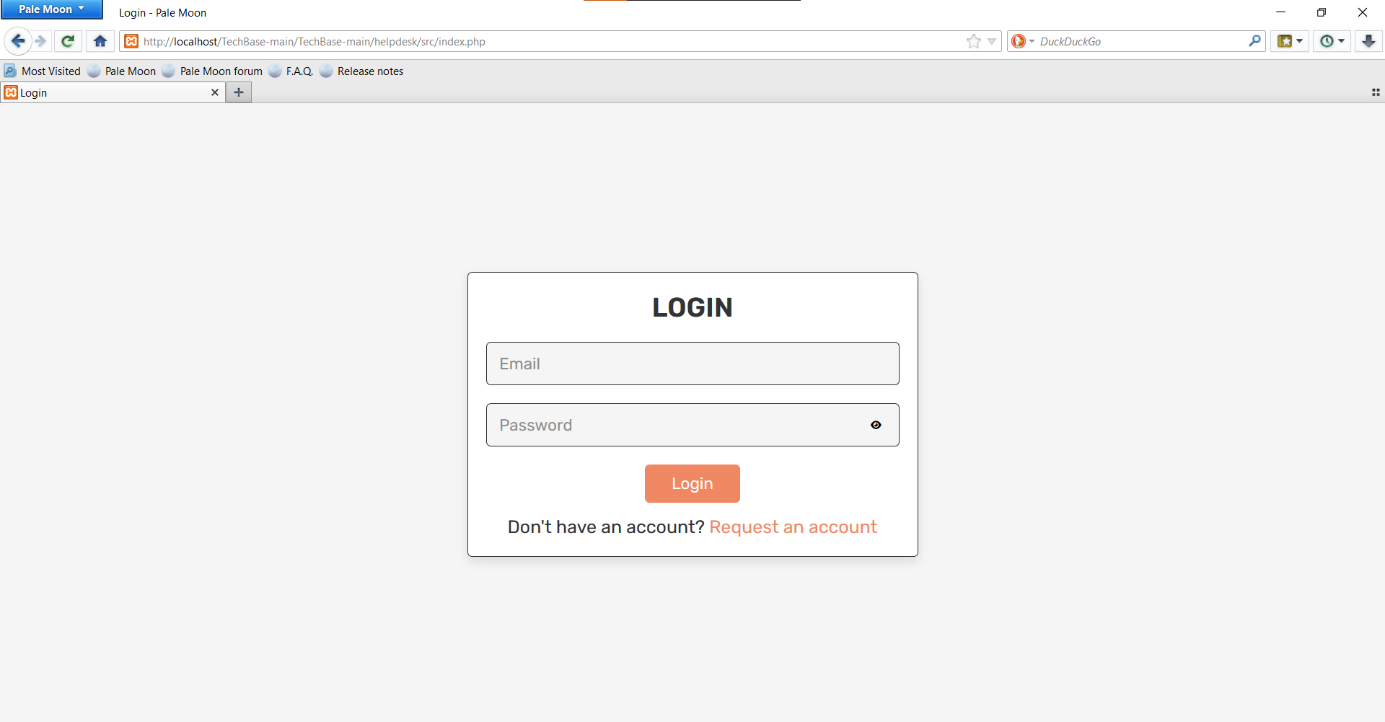
Rozhraní pro zákazníky vypadá velice podobně, jen v navigaci se nenacházejí dropdowny a vyhledávací pole, ale pouze logo firmy Techbase, tři odkazy na jiné stránky a opět ikona profilu. Karty i formuláře se v rozhraní pro zákazníky zobrazují stejně jako v administračním rozhraní. Ve spodní části se opět vyskytuje footer s copyrightem.

Stránka pro přihlášení a registraci/žádost o účet jsou jediné dvě stránky, které nemají header ani footer. Vzhledově jsou to velice jednoduché stránky, obě mají šedé pozadí, na kterém se nachází bílá karta s formulářem. V tomto formuláři se opět vyskytují světle šedá pole a oranžové tlačítko pro odeslání formuláře.

# Stručný manuál

Tento manuál slouží k vysvětlení používání aplikace.

## Přihlášení

Na této stránce se uživatel může přihlásit nebo se přepnout na stránku pro registraci. Při přihlašování musí uživatel napsat své přihlašovací informace do příslušných polí a buďto zmáčknout klávesu enter nebo kliknout na tlačítko Login. Pokud uživatel zadal všechny přihlašovací údaje správně, bude přihlášen.

Obrázek 2: Login

## Žádost o vytvoření účtu

Pokud se uživatel rozhodl, že se nechce přihlásit, ale že si chce zažádat o vytvoření účtu, pak musí kliknout na stránce Login na oranžový text „Request an account“. Poté bude přesměrován na stránku Request an account. Na této stránce musí zákazník vyplnit všechna pole a opět buď stisknout klávesu enter nebo zmáčknout tlačítko Send Request. Pak musí počkat, jestli bude jeho účet vytvořen, nebo jestli bude žádost zamítnuta.

## Obsah obrázku text, snímek obrazovky, software, Webová stránka Popis byl vytvořen automatickyZákaznické rozhraní

Obrázek 3: Request an account

Obsah obrázku text, snímek obrazovky, software, Webová stránka

Popis byl vytvořen automatickyJestliže se přihlásil zákazník, bude přesměrován na stránku home.php. Na této stránce se zákazníkovi zobrazí všechny jeho tickets. Pokud zákazník ještě neodeslal žádné tickets, zobrazí se mu pouze text „No tickets“. Na tuto stránku se lze také přepnout skrze navigaci, a to tak, že zákazník najede myší na text ‚My tickets‘ v navigaci a klikne na něj. Stejně se přepíná i na všechny ostatní stránky pomocí navigace.

Obrázek 4: My tickets

### Zprávy

Pokud zákazník odeslal alespoň jednu ticket, mohla mu přijít zpráva od správce. Zprávy se zobrazují na stránce New messages. Obsah obrázku text, snímek obrazovky, software, Webová stránka

Popis byl vytvořen automaticky

Obrázek 5: Your messages

Obsah obrázku text, snímek obrazovky, software, Webová stránka

Popis byl vytvořen automatickyZákazník po přečtení zprávy může buď zaškrtnout, že mu odpověď správce pomohla a ticket se smaže. Pokud mu odpověď správce nepomohla, klikne na tlačítko ‚No‘ a rozepíše více svůj problém.

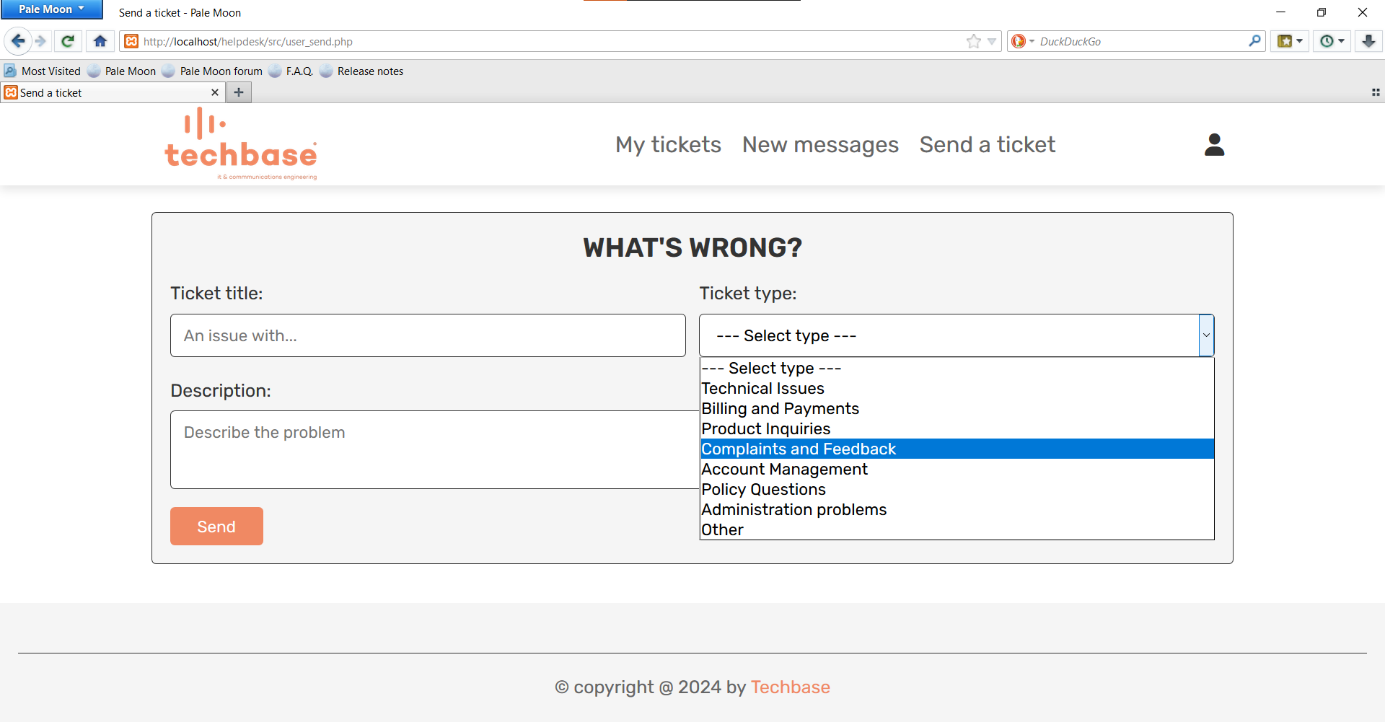
Obrázek 6: Odpověď správce zákazníkovi pomohla – ticket se smaže

Obsah obrázku text, snímek obrazovky, software, Webová stránka

Popis byl vytvořen automaticky

Obrázek 7: Odpověď správce zákazníkovi nepomohla – zákazník svůj problém rozepíše

### Odesílání ticketů

Pokud má zákazník problém, může se přepnout na stránku Send a ticket a svůj problém začít řešit se správci. Na této stránce se nachází formulář, který zákazník vyplní, popíše v něm svůj problém a odešle ho pomocí tlačítka ‚Send‘. V typech ticketů si zákazník může vybrat ze seznamu přesně ten, který popisuje jeho situaci.

Obrázek 8: Send a ticket

### Zákaznický profil

Kliknutím na ikonu postavy v pravém horním rohu se zákazníkovi otevře karta s jeho osobními informacemi a tlačítkem pro odhlášení. Stisknutím zmíněného tlačítka se zákazník může odhlásit.

## Obsah obrázku text, snímek obrazovky, software, Webová stránka Popis byl vytvořen automatickyAdministrační rozhraní

Obrázek 9: Profil, odhlášení

Administrační rozhraní je již komplexnější. Po přihlášení se správci ukáže stránka Requests. Na této stránce se mohou spravovat žádosti o vytvoření účtů. Obsah obrázku text, snímek obrazovky, software, Webová stránka

Popis byl vytvořen automaticky

Obrázek 10: Requests

V navigaci se nachází mnohonásobně víc stránek, proto jejich použití popíšeme velice stručně. Profil a odhlašování funguje u správce úplně stejně jako u zákazníka.

### Stránky typu ‚See all ...‘

Obsah obrázku text, snímek obrazovky, software, Písmo

Popis byl vytvořen automatickyNa stránkách, které začínají slovy ‚See all‘ si správce může zobrazit a spravovat vždy dané položky, např. všechny uživatelské účty, všechny oddělení nebo všechny tickety. Na některých těchto stránkách existují i filtry, díky kterým si správce může data vyfiltrovat, aby se mu v nich jednodušeji vyhledávalo. Jako příklad jsme zvolili stránku See all tickets.

Obrázek 11: See all users

### Stránky typu ‚Create ...‘

Obsah obrázku text, snímek obrazovky, software, Počítačová ikona

Popis byl vytvořen automatickyNa tšchto stránkách má správce možnost vytvářet nová data, např. nová oddělení, nového správce nebo nové typy ticketů. Na každé stránce se nachází formulář, který je nutné vyplnit. Poté stačí formulář odeslat tlačítkem ‚Create‘ nebo ‚Add‘. Pro příklad byla zvolena stránka Create an admin.

Obrázek 12: Create an admin

### Vyhledávací pole

Obsah obrázku text, snímek obrazovky, software, Webová stránka

Popis byl vytvořen automatickyV navigaci se navíc vyskytuje tzv. search bar neboli vyhledávací pole. Funguje stejně jako např. Google – napíšete, co chcete hledat a stisknete klávesu enter (nebo kliknete myší na tlačítko s ikonou lupy). Tímto se přesměrujete na stránku Search database, kde se zobrazí všechny nalezené shody.

Obrázek 13: Search database

Obsah obrázku text, snímek obrazovky, software, Počítačová ikona

Popis byl vytvořen automaticky

Obrázek 14: Search database (více výsledků)

Pokud chcete přejít stránku s výsledkem, stačí kliknout na modrý text vedle slova ‚Page:‘.

# Literatura

1. Databáze – Wikipedie: [https://cs.wikipedia.org/wiki/Databáze](https://cs.wikipedia.org/wiki/Datab%C3%A1ze)
2. Programování – Wikipedie: [https://cs.wikipedia.org/wiki/Programování](https://cs.wikipedia.org/wiki/Programov%C3%A1n%C3%AD)
3. Bootstrap: <https://getbootstrap.com/>

# Seznam obrázků

[Obrázek 1: E-R model 5](#_Toc168258480)

[Obrázek 2: Login 14](#_Toc168258481)

[Obrázek 3: Request an account 15](#_Toc168258482)

[Obrázek 4: My tickets 15](#_Toc168258483)

[Obrázek 5: Your messages 16](#_Toc168258484)

[Obrázek 6: Odpověď správce zákazníkovi pomohla – ticket se smaže 16](#_Toc168258485)

[Obrázek 7: Odpověď správce zákazníkovi nepomohla – zákazník svůj problém rozepíše 17](#_Toc168258486)

[Obrázek 8: Send a ticket 17](#_Toc168258487)

[Obrázek 9: Profil, odhlášení 18](#_Toc168258488)

[Obrázek 10: Requests 18](#_Toc168258489)

[Obrázek 11: See all users 19](#_Toc168258490)

[Obrázek 12: Create an admin 19](#_Toc168258491)

[Obrázek 13: Search database 20](#_Toc168258492)

[Obrázek 14: Search database (více výsledků) 20](#_Toc168258493)