Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačního inženýrství

Diplomová práce

Vývoj webové RPG aplikace

Bc. Jan Rajnošek

Čestné prohlášení

Prohlašuji, že svou diplomovou práci Vývoj webové RPG aplikace jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 29.3. 2023 _____

Poděkování

Rád bych touto cestou poděkoval Ing. Petru Hanzlíkovi, Ph.D. za trpělivost, ochotu, podporu a vedení mé diplomové práce.

Vývoj webové RPG aplikace

Abstrakt

Diplomová práce je zaměřena na vývoj webové RPG aplikace, která vychází z již zhotoveného prototypu z bakalářské práce autora. Teoretická část práce se zaměřuje na základní technologie týkající se webových stránek, databází, uživatelské zkušenosti, uživatelského rozhraní a bezpečnosti. Jsou zde vysvětleny základní pojmy webového prostředí, frontend, backend, databáze, jazyk SQL, bezpečnost a architektonický vzor MVC. Praktická část práce se zaměřuje na rozšíření existující webové RPG aplikace na základě analýzy funkčnosti. Za využití přístupů agilního vývoje jsou zde implementovány klíčové části RPG aplikace, včetně uživatelsky přívětivého mobile-first rozhraní, rozšíření herní mechaniky a dalších funkčních částí. Dále je provedeno a popsáno kvalitativní testování konečného prototypu. Výsledkem je plně hratelná webová RPG aplikace.

Klíčová slova: Webová aplikace, Databáze, Datový model, DBMS, MVC, RPG, PHP, SQL

RPG web game application development

Abstract

The diploma thesis focuses on the development of a web RPG application based on a prototype already made by the author in his bachelor's thesis. The theoretical part of the thesis focuses on basic technologies related to web pages, databases, user experience, user interface, and security. It explains the basic concepts of the web environment, frontend, backend, databases, SQL language, security, and the MVC architectural pattern. The practical part of the thesis focuses on expanding the existing web RPG application based on functionality analysis. Using agile development approaches, key parts of the RPG application are implemented, including a user-friendly mobile-first interface, expanded gameplay mechanics, and other functional parts. Additionally, qualitative testing of the final prototype is performed and described. The result is a fully playable web RPG application.

Keywords: Web application, Database, Data model, DBMS, MVC, RPG, PHP, SQL

Obsah

1. Úvod

Diplomová práce se zabývá vývojem webové RPG aplikace. Aplikace je stavěna na základě prototypu aplikace z předchozí bakalářské práce. Tento prototyp je dále rozvíjen, z hlediska komplexnosti a interaktivity. Databázový model je rozšířen a původní prototyp doplněn o uživatelské rozhraní založené na responsivním mobile-first css designu. Soubojový systém je refaktorován a jsou přidány další herní funkce.

Teoretická část práce je zaměřena na analýzu elementárních oblastí tvorby webových stránek. Popisuje jednotlivé technologie a vysvětluje, jakým způsobem fungují dohromady. Práce je nejdříve zaměřena na frontend a související technologie jako jsou HTML a CSS. Dále jsou popisovány prvky uživatelského rozhraní a uživatelské zkušenosti. Následuje backend a související technologie jako jsou PHP a SQL. Dále jsou zde části věnované bezpečnosti, architektonickým vzorům, agilním metodikám a krátké teorii her.

V praktické části práce je popsána tvorba webové aplikace za použití získaných vědomostí z teoretické části. V části analýzy řešení jsou stanoveny způsoby, jakými by webová aplikace mohla být vyvíjena. Dále v části implementace je popsána postupná tvorba jednotlivých částí aplikace a následně jsou zhodnocovány výsledky kvalitativního testování.

2. Cíl práce a metodika

2.1 Cíl práce

Tato diplomová práce se zaměřuje na problematiku vývoje webové hry za použití návrhového vzoru MVC a již namodelované databáze. Hlavním cílem této diplomové práce je zhotovení funkčního prototypu webové RPG hry.

Dílčími cíli jsou:

- Zhotovení administrátorského rozhraní pro kontrolu dat
- Real time interakce uživatelů v souboji
- Validní mobile first responsivní design
- Kvalitativní testování aplikace a vyhodnocení výsledků

2.2 Metodika

Tato práce vychází z existujícího datového modelu a základní webové aplikace, které vznikly v bakalářské práci autora. Na základě analýzy odborných informačních zdrojů souvisejících s tématem práce a syntézy takto získaných poznatků budou popsána pravidla tvorby responsivní webové aplikace a její funkčnosti. Bude vytvořen prototyp webové RPG hry, který bude kvalitativně testován a výsledky testování budou zhodnoceny. Získané poznatky z teoretické části budou využity při tvorbě robustní logiky aplikace, designu a základního herního prototypu. Při vývoji hry bude využito standardních metod a postupů softwarového inženýrství.

3. Teoretická část

Teoretická část práce je věnována aspektům moderní tvorby webových stránek a teorii technologií souvisejících s prací. Teoretická část začíná problematikou frontendu webových stránek, kde jsou popsány základní technologie pro tvorbu popředí stránky. Pojednává se zde o webových technologiích jako jsou HTML a CSS. Na tuto problematiku navazuje pohled na uživatelské rozhraní a uživatelskou zkušenost, kde jsou popsány základy těchto oblastí. Po části frontendu navazuje část věnovaná testování. Na další odvětví teorie navazuje problematika backendu. Pojednává se zde o různých technologiích, které jsou spolu související. Tato část začíná technologií PHP a dále popisuje API a REST API. Navazuje souvislost s databázovými technologiemi jako jsou SQL a SŘBD. Dále je nastíněna bezpečnost, kde se pojednává o základních prvcích bezpečnosti. Další téma je architektonický vzor MVC. Následuje téma agilních metodik a v závěru této teoretické části je pojednáváno o hrách.

3.1. Webová stránka

Pro vytvoření webové stránky je třeba mít určité znalosti a dovednosti. Mezi tyto znalosti patří jazyky HTML, CSS a JavaScript. Tyto jazyky jsou základním stavebním kamenem pro tvorbu webových stránek.

Pro tvorbu malé webové stránky není nutné, aby autor byl webový designer, spisovatel nebo programátor. Existují tzv. „frameworky“, které jsou nástrojem pro zjednodušení tvorby webové stránky. Zjednodušení je na takové úrovni, aby i laik dokázal vytvořit za desítky minut solidní a robustní webovou stránku. (Aniebiet, 2021)

Velké nebo střední projekty jsou odlišné od malých webových stránek hlavně mírou komplexity. Pro úspěšné vytvoření rozsáhlé a robustní webové stránky vznikají týmy desítek až stovek lidí, kteří mají různé zaměření a podílejí se na projektu různými aspekty. Je běžné, že tým lidí je rozdělen na frontend, backend, API, design, testování, analýzu a project management. Frontend je popředí stránky, které je prezentováno uživateli. Backend je pozadí stránky. Obsahuje skryté funkčnosti, skripty a kódy které umožnují systému fungovat dle představ. Mezi frontendem a backendem často stojí API (Application Programming Interface), které volá proměnné z backendu do frontendu, a zároveň funguje jako oddělovač, který umožňuje webové aplikaci, aby mohla být vyvíjena v nezávislosti popředí a pozadí stránky.

Designérský tým má více zaměření. Navrhuje vzhled webové stránky a dílčí části UI (User interface). Další zaměření je na použitelnost, přístupnost a celkovým dojmem, který člověk získá při používání webové aplikace. Součástí větších projektů je také testování. Testování je v moderním vývoji důležitý úkol, protože nejen objevuje chyby a nedostatky, ale také se na aplikaci dívá z pohledu uživatele. (Mishra, a další)

Celý proces bývá zastřešen agilní metodikou, která umožňuje rychlý a dynamický vývoj. Vývoj se odehráván v intervalech (sprintech), což jsou cykly procesů, které jsou schopny při vývoji reagovat na změnu požadavků. Tento přístup je v současnosti nejčastěji využíván. Je to z důvodu ověření správnosti projektu pomocí spolupráce se zákazníkem a následným rychlým vývojem, který reaguje na zpětnou vazbu.

Nyní byly popsány různé role, které se vyskytují ve vývoji webových aplikací. A z toho tedy vyplývá otázka: „Které technologie tvůrce webových stránek má umět?“.
Základem webových stránek je HTML, CSS, JavaScript a PHP. Webové stránky je možné stavět i na technologii ASP.NET za pomocí programovacího jazyka C#. Dále existuje Node.js, které je v dnešní době také velmi rozšířené. Ale všeobecně platí, že pro frontend je důležité znát client-side technologie, jako jsou HTML, CSS a JavaScript. A pro backend je důležité znát server-side technologie. V Současné době jsou nejvíce využívány jazyky PHP a JavaScript jako server-side technologie. Pro lepší práci s daty je vhodné znát i databázové technologie, které výborně spolupracují se server-side technologiemi. Následující kapitoly představí vybrané technologie související s webovými stránkami. (BURCHARD, 2013) (NIEDERST ROBBINS, 2012) (Noback, 2020)

3.2. Frontend

Frontend je část webové stránky, která je zaměřena na to, co je vidět. Zabývá se tvorbou uživatelského rozhraní a týká se všeho, co uživatel vidí a může s tím manipulovat v prohlížeči. Je to třeba text, obrázek, odkaz, formulář, nebo další prvky.

3.2.1. HTML

HTML (Hyper Text Markup Language) je standardní značkovací jazyk, který je používán pro tvorbu webových stránek. Značkovací jazyk HTML je poměrně jednoduchý na naučení. Skládá se z elementů, které jsou použity k označení různých částí obsahu stránky, jako jsou odstavce, nadpisy, seznamy a obrázky. Elementy jsou zapisovány v párových a nepárových značkách, například a je párová značka, která uzavírá odstavec. Příklad nepárové značky je <img>, která se používá pro vložení obrázku do HTML stránky. (Sharma, 2018)

Člověk, který tvoří svou malou webovou stránku musí zastávat více rolí a na stránku nahlížet z různých úhlů pohledů.

„If you are designing a small website on your own, you will need to wear many hats.“

„Consider that the day-to-day upkeep of your household requires you to be part-time chef, housecleaner, accountant, diplomat, gardener, and construction worker“ (NIEDERST ROBBINS, 2012)

Na stránku je nutné nahlížet z více pohledů. Z pohledu UX, grafiky, spisovatele, informačního architekta, a dalších pohledů.

HTML je systém pro identifikaci a popis jednotlivých komponentů v dokumentu jako jsou třeba: záhlaví, zápatí, paragraf, tlačítko, a další komponenty. Vytváření komponentů pomocí značek také produkuje strukturu v dokumentu. Jednotlivé komponenty lze editovat, stylovat a upravovat pomocí kaskádových stylů (CSS). (NIEDERST ROBBINS, 2012)

3.2.2. HTML5

HTML5 je verze HTML, která přináší po několika letech aktualizace pro vývoj moderního uživatelského rozhraní. S vývojem webového prostředí, bohužel HTML zůstávalo pozadu. Dříve HTML nemělo tolik funkcí a z toho důvodu bylo nutností stahovat různé pluginy (Flash, RealPlayer, …), aby vůbec uživatel mohl přehrát video. Se stahováním pluginů souviselo riziko, že uživatel stáhne malware nebo jiný nevyžádaný soubor. Z toho důvodu přichází HTML5, které přináší modernizaci a nové možnosti jako jsou přehrávání videa, audia, vektorová grafika, animace a drag and drop funkcionality. Největším rozdílem mezi HTML4 a HTML5 je způsob, jakým konsistentně zpracovává chyby. (Ghann, a další, 2011)

HTML je celkově považováno za budoucnost webových technologií a stále se vyvíjí a zlepšuje. Jeho používání je stále rozšiřováno a v současnosti převážná většina webových stránek používá technologie HTML5 s kombinací CSS a JavaScript pro vytváření bohatších a interaktivnějších zážitků pro uživatele. (Ghann, a další, 2011)

3.2.3. CSS

CSS (Cascading Style Sheets) je jazyk, který je používán ke stylování HTML dokumentu. CSS popisuje, jak by se HTML prvky měly chovat a jakým způsobem se zobrazují na webové stránce. (w3schools)

Vzhled webové stránky se nazývá prezentace (presentation). Všechno, co můžeme vidět na stránce, je prezentace. Barvy, písma, pozadí, obrázky, ohraničení, řádkování, rozvržení stránky a další.

Příklad CSS Selektoru prvku:

Obrázek 1: CSS Element Selektor (vlastní zpracování)

Na příkladu je vidět CSS Element Selektor „p“, který definuje vzhled pro všechny výchozí paragrafy na webové stránce. Ve složených závorkách je definována vlastnost elementu, tedy jeho chování. „Background-color: red;“ konkrétně říká, že pozadí bude červené. Celý tento selektor definuje všechny paragrafy a říká, že budou mít červené pozadí.

CSS poskytuje veškeré metody pro řízení toho, jak budou dokumenty prezentovány. Dokonce je možné určit i kontext, který kompletně mění chování stránky na různých zařízeních. CSS nemusí být definováno pouze na tradiční desktopové prohlížeče, ale může být specifikováno pro daná rozmezí šířky a výšky displeje. (NIEDERST ROBBINS, 2012) (w3schools)

CSS šablony jsou úžasným nástrojem pro automatizaci celé webové stránky. Je možné si nadefinovat prvek s konkrétní třídou a ten prvek volat vícekrát na různých místech webové

stránky. Pro CSS existuje rozšíření Sass. Sass (Syntactically Awesome Stylesheet) je CSS pre-processor který generuje CSS ze souboru SCSS. S novým zápisem vznikají další možnosti jako jsou proměnné, podmínky, funkce, nesting, mixins a další. Sass je nadstavba poskytující nový, kratší a přehlednější způsob zápisu. Tento nástroj je skvělý pro údržbu větších a komplexnějších stylů nebo šablon. (W3Schools)

3.2.4. Souvislost HTML a CSS

HTML a CSS jsou navzájem propojené a jsou elementární při tvorbě webových stránek. HTML je používáno k popisu struktury a obsahu stránky, zatímco CSS je používáno k popisu vzhledu a formátování stránky.

HTML elementy jsou označeny atributy, které umožňují nastavení vzhledu pomocí CSS pravidel. Například element <p> (odstavec) může mít atribut "class" nebo "id", který se použije jako selektor v CSS pravidlech pro stylizaci odstavců na stránce.

HTML by mohlo vypadat třeba takto:

```
<p class="uvod">Vítejte na mém webu!</p>
```

Obrázek 2: příklad HTML (vlastní zpracování)

CSS by mohlo vypadat třeba takto:

```
.uvod{
    font-size: 18px;
    font-weight: bold;
}
```

Obrázek 3: příklad kaskádového stylu úvodu (vlastní zpracování)

V tomto příkladu je použit atribut "class" v HTML pro označení typu odstavce a pravidlo CSS pro stylizaci tohoto odstavce pomocí selektorů třídy. Tento zápis umožňuje vytvořit jednotný vzhled pro všechny elementy s třídou „uvod“.

3.2.5. JavaScript

JavaScript je skriptovací jazyk, který je používán na zprostředkování interaktivity a chování webové stránky. Manipuluje elementy nebo i samotný prohlížeč. JavaScript je uznáván jako skriptovací jazyk, ale má lehké prvky programovacího jazyka. Konkrétně tedy funkce a objektově orientovanou strukturu. Je poměrně lehký na naučení a jednoduše interpretovatelný. JavaScript je uváděn jako nejdůležitější client-side skriptovací jazyk, ale lze jej použít i na server-side. Pro server-side prostředí, kde je využíván JavaScript se využívá Node.js. (Javatpoint, 2022)

3.3. User Interface (UI) - Uživatelské rozhraní

Uživatelské rozhraní je rozhraní mezi uživateli a softwarem, které umožňuje uživateli ovládat systém. Uživatelské rozhraní může zahrnovat vizuální prvky, jako jsou ikony, tlačítka, texty a grafika. Může podporovat interakční prvky, jako jsou gesta, kliknutí a přetahování. Dále také může být pouze příkazová řádka, která komunikuje s uživatelem pouze pomocí textových příkazů. Hlavním cílem UI je vytvořit uživatelsky přívětivé rozhraní, které umožňuje uživatelům snadné a efektivní ovládání systému, bez zbytečných obtíží. Mezi uživatelské rozhraní (UI) patří grafické uživatelské rozhraní (GUI) a příkazový řádek (CLI). (Unwin, a další, 2000) (Fellmann, a další, 2007)

3.3.1. Graphic User Interface (GUI)

GUI (Graphic User Interface) je typ uživatelského rozhraní, které umožňuje uživatelům ovládat počítač pomocí grafických prvků, jako jsou ikony, tlačítka a menu. GUI je běžné pro většinu operačních systémů. Umožňuje uživatelům snadno vybírat a spouštět programy, ovládat okna a manipulovat s vizuálními prvky pomocí myši nebo dotykové obrazovky. (Fellmann, a další, 2007) (Myers, 2003)

3.3.2. Command Line Interface (CLI)

CLI (Command Line Interface) je typ uživatelského rozhraní, které umožňuje uživatelům interagovat s počítačem pomocí textových příkazů. Jedná se o textovou konzoli, která zobrazuje uživateli textové pole, do kterého může zadávat příkazy pro spuštění programů, ovládání souborů a mnoho dalšího. CLI je běžně pro operační systémy jako jsou Linux a Unix, ale je také běžně využíváno ve vývojářském prostředí. (Fellmann, a další, 2007)

Z většiny případů je používáno GUI z důvodu jednoduchosti a přívětivosti pro nové uživatele, avšak CLI může být efektivnější a rychlejší pro pokročilé uživatele a programátory. (Fellmann, a další, 2007)

3.3.3. Mockup

Mockup je návrh nebo prototyp určitého produktu nebo služby, který je používán v oblasti UX (user experience). Prezentuje vzhled a funkčnosti produktu nebo služby, aniž by byl vytvořen plnohodnotný produkt nebo služba.

Mockupy se často používají v rané fázi vývoje, aby se zajistilo, že design a funkčnost odpovídají požadavkům a představám zákazníků nebo uživatelů. Jejich výhodou je, že umožňují rychlou a levnou kontrolu konceptu a poskytují představu o tom, jak by mohl výsledný produkt nebo služba vypadat a fungovat. (Stjepandić, a další, 2015)

3.3.4. Wireframe

„A wireframe diagram shows the structure of a web page using only outlines for each content type and widget“ (KRUG, 2010)

„The purpose of a wireframe diagram is to indicate how the screen real estate is divided and indicate where functionality and content such as navigation, search boxes, form elements, and so on, are placed, without any decoration or graphic design.“ (KRUG, 2010)

„They are usually annotated with instructions for how things should work so the development team knows what to build.“ (KRUG, 2010)

Wireframe (drátěný model) je návrh nebo schéma struktury webové stránky nebo aplikace. Je to nástroj, který slouží k prezentaci základní struktury stránky nebo aplikace, aniž by se zabýval detaily vzhledu a designu.

Rozdíl mezi mockupem a wireframem spočívá v tom, že mockup je návrh nebo prototyp produktu, ale wireframe je návrh nebo schéma struktury webové stránky nebo aplikace.

Užitečné nástroje pro tvorbu mockup nebo wireframe jsou Draw.io nebo Figma. Mockup a wireframe také může být jednoduše vytvořen ručně, pomocí náčrtu na papír. (KRUG, 2010) (KRUG, 2010)

3.3.5. Low/High Fidelity prototyp

Low fidelity prototyp (LFP) je málo detailní prototyp produktu. Je to jednoduchá a rychlá verze prototypu, která slouží k prezentaci základních funkcí a konceptů. LFP se často používá v rané fázi vývoje, kdy je potřeba rychle ověřit několik nápadů a získat zpětnou vazbu od uživatelů. (Stjepandić, a další, 2015)

High fidelity prototyp (HFP) je naopak pokročilejší verze prototypu, která je více podobná finálnímu produktu nebo službě. HFP obsahuje více detailů a funkcí a poskytuje uživatelům realističtější představu o tom, jak by mohl výsledný produkt nebo služba vypadat a fungovat. Často se používá v pozdější fázi vývoje, kdy je potřeba ověřit finální koncept a získat podrobnější zpětnou vazbu od uživatelů. (Stjepandić, a další, 2015)

3.4. User Experience (UX) – Uživatelská zkušenost

User experience (UX) je celkový dojem uživatele produktu nebo služby. UX se zabývá použitelností a přístupností. Dobrý UX design činí produkt nebo službu snadným a příjemným na použití. Dále pomáhá vytvořit pozitivní vztah mezi uživatelem a produktem nebo službou a jedná se o interdisciplinární oblast, která zahrnuje prvky designu, psychologie a informatiky. (Michelle, 2022) (Garrett, 2011)

3.4.1. Přístupnost

Přístupnost je schopnost věcí, míst nebo služeb být využity co nejširším okruhem lidí, včetně lidí s různými druhy omezení nebo bariérami. Může se týkat například přístupu ke vzdělání, zaměstnání, zdravotní péči, kulturním a společenským akcím nebo informacím na internetu.

V rámci webových stránek je přístupný web takový, který neklade žádné překážky jakémukoliv uživateli, který jeho služby využívá. Přístupnost se týká také designu webových stránek a aplikací. Je důležité vytvořit webové stránky tak, aby byly použitelné pro co největší

počet lidí, včetně lidí s omezeným zrakem, sluchem nebo pohybem. Cílem je umožnit všem lidem plné a rovnocenné využívání webových stránek a aplikací bez ohledu na to, zda používají klasické počítače, chytré telefony, nebo jakákoliv jiná zařízení. Využívání by nemělo být omezeno i v případě inputových zařízení, jako je třeba hlasové ovládání, dotyková obrazovka nebo myš. (KRUG, 2010) (Scott, a další, 2009) (Zitkus, a další, 2016)

3.4.2. Použitelnost

Použitelnost se týká schopnosti věcí, míst nebo služeb být využívány co nejefektivněji a nejpříjemněji. V této práci se použitelnost týká hlavně designu webových stránek a aplikací, kdy je cílem vytvořit pro uživatele snadno fungující prvky, které jsou jednoduché na ovládání a používání. Webová stránka by měla být vytvořena tak, aby její prvky byly snadné na ovládání a aby její prvky mohl použít každý. Když je něco použitelné, tak to znamená, že daný prvek je pro uživatele intuitivní, srozumitelný na ovládání, rychlý a spolehlivý. Souvisí s tím i příjemné a atraktivní grafické provedení, přehlednost, čistota a jednoduchost informací a také možnost přizpůsobení ovládání. Informace by měly být zobrazovány dle potřeb a preferencí jednotlivých uživatelů. (KRUG, 2010)

Dobrá použitelnost je důležitá pro zvýšení efektivity a produktivity uživatelů, a tím se zvyšuje schopnost webu nebo aplikace konkurovat ostatním. (Scott, a další, 2009) (Zitkus, a další, 2016) (KRUG, 2010)

3.4.3. Test Přístupnosti

Test přístupnosti (Accessibility Testing) se zaměřuje na ověření, zda jsou webové stránky, aplikace nebo jiné digitální produkty použitelné pro uživatele s různými omezeními, jako jsou například zraková nebo sluchová postižení. Cílem testování přístupnosti je zajistit, že všichni uživatelé, bez ohledu na své individuální potřeby a schopnosti, mohou plně využít digitální produkty. (Zitkus, a další, 2016)

K otestování přístupnosti na webové stránce lze použít crawlera. Crawler (pavouk) je program, který automaticky prochází internetové stránky a sbírá o nich data. Crawler začíná výchozí stránkou a postupně analyzuje celou stránku, dále postupuje po jednotlivých odkazech na stránce a následně analyzuje i ostatní stránky. Tímto způsobem lze hledat specifické chyby,

které by mohly omezovat přístupnost pro uživatele s postižením. Takto lze dohledat nedostatečně popsané obrázky, nečitelný text nebo chybějící alternativní způsoby navigace.

3.4.4. Test Použitelnosti

Test použitelnosti (Usability Testing) je test, při kterém je zkoumáno, jak snadné je pro běžného uživatele používání dané webové stránky nebo aplikace. Cílem je zjistit, zda jsou na webové stránce nebo aplikaci přítomny nějaké bariéry, které by mohly bránit uživatelům v plnohodnotném používání stránky. Tento proces může zahrnovat řadu různých technik, jako je například sledování pohybů očí uživatelů, testování s různými typy zařízení nebo zapojení skupiny uživatelů do testování. (Zitkus, a další, 2016)

3.4.5. Souvislost přístupnosti a použitelnosti

Testy přístupnosti a použitelnosti jsou součástí uživatelského testování, které je využíváno k ověření funkčnosti a uživatelského zážitku digitálního produktu. Uživatelské testování se zaměřuje na to, jak uživatelé produkt skutečně používají a jaký mají s ním zážitek. Existuje organizace WebAIM (Web Accessibility in Mind), která se zabývá zlepšováním přístupnosti a použitelnosti webových stránek pro všechny uživatele, včetně lidí s různými druhy omezení. Přístupnost a použitelnost jsou dva související koncepty. Přístupnost se zabývá schopností dané věci, aby byla použitelná pro co největší okruh lidí, včetně lidí s různými druhy omezení. A použitelnost se týká schopností dané věci, aby byla použita co nejefektivněji, a aby její používání bylo snadné a příjemné pro uživatele. Jsou to tedy koncepty navzájem propojené a doplňující se. (KRUG, 2010) (Scott, a další, 2009) (Zitkus, a další, 2016) (KRUG, 2010)

3.4.6. Kvalitativní testování

Kvalitativní testování se zaměřuje na aspekty produktu z pohledu uživatelské zkušenosti, použitelnosti, spolehlivosti a další. Kvalitativní přístup zahrnuje sběr subjektivních dat, které jsou získány prostřednictvím dotazníků, rozhovorů a pozorováním uživatelů. Pomocí kvalitativního testování lze odhalit potenciální problémy a lze vylepšit celkovou kvalitu produktu. (Flick, 1999)

5 second test

Jedna z metod kvalitativního testování je 5 second test. Jedná se o pětisekundový test, který je rychlou metodou testování, kde jsou uživatelé požádáni, aby se podívali na krátkou chvíli (5 sekund) na webovou stránku a poté odpověděli na pár otázek. Pokládané otázky se často týkají prvních dojmů, jako co si uživatel myslí, že webová stránka nabízí, jaké služby a jaké má uživatel pocity z vizuální podoby stránky. Metoda 5 second test může velmi pomoci k odhalení základních nedostatků webového designu a může napomoci k vylepšení prvního dojmu. (Gronier, 2016)

Think aloud

Metoda think aloud je používána při kvalitativním testování k získání rychlé zpětné vazby od uživatele při interakci s webovou stránkou, produktem nebo službou. Při této metodě je uživatel požádán, aby během testování nahlas popisoval své myšlenkové pochody a dojmy během používání webové stránky, produktu nebo služby. Tímto způsobem je možné získat ihned zpětnou vazbu během testování a následně tím lépe porozumět potřebám, frustracím a preferencím uživatele. Na základě těchto poznatků jsme schopni eliminovat nedostatky, bariéry a zlepšit uživatelský zážitek a přívětivost. (Donker, a další, 2002)

Blackbox

Metoda blackbox (černá skříňka) je využívána na testování chování softwarového produktu bez vnitřní znalosti kódu. Při této metodě jsou testeři požádání, aby testovali softwarový produkt z pohledu uživatele a zjišťuje se, zda produkt splňuje všechny požadavky a očekávání uživatele a jestli je produkt připraven, nebo jestli je nutné provést změny. (Ajunwa, 2020) (Hustinx, a další, 2009)

3.4.7. Kvantitativní testování

Kvantitativní testování je založeno na sběru měřitelných dat a na statistické analýze. Důraz je kladen hlavně na kvantitativní metriky jako jsou čas odezvy, rychlost, úspěšnost, počet chyb, a další. Kvantitativní testování je často prováděno pomocí automatizovaných testů. (Apuke, 2017)

A/B testování

Metoda A/B testování je využívána k porovnání dvou různých variant jedné webové stránky. Při testování jsou testeři rozděleni do dvou skupin a každé skupině je zobrazena jedna varianta produktu. Po testování jsou zkoumány statistické údaje o chování uživatelů v každé skupině a zkoumá se, která varianta měla lepší výkon. Tato metoda může pomoci ke zlepšení klíčových metrik stránky.

Eye-Tracking

Převážná většina informací z vnějšího světa je zpracovávána pomocí visuálního kontaktu, pomocí našich očí. Díky tomuto faktu jsme pomocí Eye-tracking technologií schopni zjistit velké množství informací jen z pohybu očí individuální osoby. Jsme schopni analyzovat a zaznamenávat pohyb očí uživatele při testování webové stránky. Pomocí této metody je možné získat data o tom, které části produktu jsou pro uživatele nejzajímavější, které jsou přehlížené a jaké je jejich pořadí. Technologie eye-tracking umožňuje vygenerovat teplotní mapu (heat map), která zobrazuje nejčastěji prohlížené části produktu. Pomocí této metody jsme schopni zlepšit uživatelské zkušenosti, orientaci, efektivitu designu a další. (Punde, a další, 2017)

Time on Task Measurements (TOT)

Metoda Time on Task Measurements (TOT) je používána k měření doby, kterou uživatelé tráví na zadané úloze, a k posouzení úspěšnosti dokončení této úlohy. Tato metoda umožňuje zjistit, kolik času uživatelé potřebují na splnění úkolu, zda je úkol dostatečně snadný nebo příliš složitý a jakou úroveň úspěšnosti dosahují. Pomocí TOT jsou vývojáři a designéři schopni zjistit jakým způsobem uživatelé využívají aplikaci a na základě zpětné vazby jsou schopni aplikaci dále upravit nebo rozvinout tak, aby byla zlepšena uživatelská přívětivost a produktivita. (Kovanovic, a další, 2016)

3.5. Backend

Backend je část aplikace, která se stará o zpracování dat, logiku a zprostředkovává data z databází a dalších externích zdrojů. Obvykle se skládá z více serverových technologií, jako jsou server-side programovací jazyky, databáze a aplikační rozhraní (API).

3.5.1. PHP

PHP (Hypertext Preprocessor) je velmi silný nástroj na skriptování server-side. Je nezávislý na platformě a podporuje mnoho knihoven pro různé účely. Velmi dobře spolupracuje s většinou databázových systémů (MySQL, Oracle, MariaDB) a podporuje mnoho internetových protokolů (HTTP, SMTP, FTP, IMAP). Pomocí PHP je možné dynamicky vygenerovat obsah webové stránky. Webová stránka může vypadat úplně jinak pro různé uživatele. Nepřihlášený člověk vidí formulář pro přihlášení. Přihlášený uživatel vidí obsah. Přihlášený administrátor vidí prostředí pro editaci obsahu.

PHP je často používáno ke zpracování formulářů, k ukládání a získávání informací z databází, pro vytváření cookies a mnoho další. PHP obsahuje proměnné, které mohou být využity k uložení dat v paměti počítače. Proměnné jsou v PHP označovány dolarovým znaménkem ($). Název proměnné může obsahovat čísla, písmenka, podtržítka, ale musí začínat písmenem nebo podtržítkem.

Příklad:
$jmeno = „Jan“;
$vek = 25;

Proměnné mohou být různými datovými typy, např.: celé číslo, řetězec, pole, nebo boolean. Můžete se také setkat s konstantami, které také patří mezi proměnné, ale jejich hodnota je fixní. Hodnota konstanty nemůže být změněna po prvním deklarování.

Nevýhoda PHP je, že většina akcí, jako odeslání formuláře, nebo přechod na jinou sekci aplikace, zapříčiní nutné obnovení stránky. Tento problém se ale dá obejít pomocí různých technologií, jako je třeba AJAX, Fetch API, nebo přepisování HTML kódu na stránce pomocí JavaScriptu. Pokud se stránka načte pouze jednou a dále se mění bez obnovení stránky, tak se

tato stránka nazývá SPA (Single Page Application). (BURCHARD, 2013) (NIEDERST ROBBINS, 2012) (Mach, 2003)

Výhody SPA jsou převážně v rychlosti a UX. A protože stránka je single page, tak je snažší takovou stránku vyvíjet, protože pozornost může být věnována samotnému obsahu aplikace, místo tvoření mnoho stránek s různým obsahem. (BURCHARD, 2013) (NIEDERST ROBBINS, 2012) (Mach, 2003)

3.5.2. PHP 8

Verze PHP 8 byla vydána v listopadu 2020 a přinesla mnoho nových funkcí a vylepšení oproti předchozí verzi. Nově umožňuje specifikovat datový typ proměnné a specifikovat argumenty funkcí pomocí jejich jména, místo toho, abyste si museli zapamatovat pořadí argumentů. Dále obsahuje nový kompilátor, který zlepšuje výkon PHP aplikací. Jednodušší zápis konstruktorů a mnohé další. (PHP: Hypertext Preprocessor, 2020)

3.5.3. JavaScript na backendu

I přesto, že JavaScript je tradičně vnímán jako jazyk pro vývoj webových aplikací na straně klienta, tak se díky technologii Node.js stal populární technologií i pro vývoj na straně serveru. Node.js je open source, cross-platform, back-endový JavaScriptový runtime, který umožňuje vývojářům psát serverové aplikace pomocí JavaScriptu. Node.js má také velké množství balíčků a modulů, které jsou k dispozici v jeho ekosystému, což usnadňuje vývojářům vývoj aplikací pomocí různých knihoven a frameworků. Díky těmto funkcím se Node.js stal velmi oblíbenou volbou pro vývojáře, kteří chtějí psát serverové aplikace pomocí JavaScriptu. (Kinsta, 2023) (Node.js)

3.5.4. API

API (Application Programming Interface) je sada pravidel a standardů, které umožňují jednomu počítačovému programu přistupovat k funkcím a datům druhého programu. API se používají k propojení různých systémů a aplikací a umožňují jim vzájemně spolupracovat.

API obsahuje definice vstupních a výstupních bodů pro dané funkce a metody, které mohou být volány z jiných programů. Tyto funkce a metody jsou obvykle poskytovány ve formě knihovny nebo modulu, který je možné importovat do jiného programu. Programy, které chtějí

používat funkce a metody definované v API, se musí k API připojit a poskytnout příslušné přihlašovací údaje pro autentizaci. (Bojinov, 2018) (Richardson, a další, 2007)

3.5.4.1. REST API

REST (Representational State Transfer) je softwarový architektonický styl, který stanovuje sadu omezení pro vytváření webových služeb. Webové služby, které se řídí architektonickým stylem REST nebo RESTful, poskytují spolupráci mezi počítačovými systémy na internetu.

REST API dodržují architektonický styl a omezení REST. Jsou navržena tak, aby poskytovala standardní způsob, jakým systémy přistupují a manipulují s daty a funkcemi. REST API používají základní protokol HTTP a podporují různé typy HTTP požadavků, jako například GET, POST, PUT, DELETE a podobně. REST API obvykle vracejí data ve standardním formátu, jako je například JSON nebo XML, což je velmi dobré, protože to ulehčuje práci s daty.

REST API je zpravidla používáno pro komunikaci mezi webovými službami nebo webovými aplikacemi. Hlavní výhodou REST API je, že je velmi jednoduché a snadno se implementuje. Navíc může být používáno na mnoha platformách, protože běží na běžném HTTP protokolu. (Bojinov, 2018) (Richardson, a další, 2007) (Mishra, a další)

3.5.5. Frontend a Backend

Frontend i backend se oba vztahují k tvorbě webových stránek. Frontendové soubory jsou ty, které jsou přenášeny uživateli přímo a jsou viditelné v prohlížeči. Tyto soubory zahrnují HTML, CSS a JavaScript. Backendové soubory jsou ty, které jsou umístěny na serveru a poskytují data pro frontendové soubory. Tyto soubory zahrnují různé výpočty a skripty pro zpracování dat na straně serveru. Často bývají napsány v jazycích PHP nebo JavaScript. Oba typy souborů jsou důležité pro tvorbu webových stránek a spolupracují spolu. Backendové soubory zajišťují, aby webová stránka mohla fungovat správně, zatímco frontendové soubory reprezentují spíše design, rozhraní a interakce s uživateli. Pro vývoj webových stránek jsou důležité obě strany. Je důležité, aby vývojáři měli alespoň základní znalosti z obou oblastí a dokázali si porozumět. (KRUG, 2010) (Laurenčík, 2019) (Noback, 2020)

3.5.6. Client side a Server side

Client side a server side jsou rozdílné v tom, kde se vykonávají kódy v rámci webové aplikace. Client side se týká kódu, který běží na počítači klienta. Například prohlížeč nebo mobilní zařízení. Tyto skripty mohou být třeba v jazycích jako je JavaScript nebo HTML5, a mohou být použity k vykonání akcí jako je validace formulářů, přechod mezi stránkami nebo interakce s uživatelem.

Server side se naopak týká kódu běžícího na serveru. Tyto skripty mohou být v jazycích PHP, Python, nebo v dnešní době i JavaScript. Kód na server side se spíše používá pro zpracování formulářů, komunikace s databází a vyhledávání v ní, nebo generování dynamického obsahu.

Obecně lze říci, že server side skripty se zabývají zpracováním obsahu v pozadí a také komunikací s databází a dalšími částmi systému. Zatímco client side skripty se používají pro interakci s uživatelem a zpracovávají úkony které uživatel požaduje na stránkách.

3.5.7. WebSocket

WebSocket je protokol pro přenos dat mezi webovým prohlížečem a serverem v reálném čase. Protokol WebSocket umožňuje obousměrnou komunikaci mezi klientem a serverem pomocí jediného TCP spojení. V tradiční webové aplikaci je používána metoda HTTP Request-Response pro komunikaci se serverem, která je vhodná pro odesílání žádostí ze strany klienta a zpracování odpovědí ze strany serveru. Nicméně, tato metoda není vhodná pro situace, kdy je potřeba provádět v reálném čase interaktivní komunikaci mezi klientem a serverem.

WebSocket technologie, oproti HTTP Request-Response umožňuje klientovi zasílat dotazy na server a následně na ně okamžitě reagovat v reálném čase. Umožňuje zasílání dat ze serveru na klienta bez nutnosti, aby klient posílal dotazy na sever. Tento princip je velmi užitečný a nutný pro aplikace, které vyžadují okamžitou aktualizaci dat. Na příklad online chat, hry nebo stream. (Fette, a další, 2011) (MDN)

3.6. Databázové technologie a SQL

SQL (Structured Query Language) je jazyk pro práci s relačními databázemi. Umožňuje vytvářet tabulky, vkládat do nich data, upravovat je a vyhledávat. Relační databáze ukládají data v tabulkách, které jsou mezi sebou propojené.

Při návrhu databází se často využívá ERD (Entity-Relationship diagram). Jedná se o vizuální nástroj zobrazující datový model, který popisuje vztahy mezi entitami. Model se skládá z entit (tabulek) a jejich vztahů (čáry mezi nimi). (Badia, 2004)

Pro uvedení do kontextu k databázovým technologiím si pojďme označit základní pojmy.

- Entita je označení pro tabulku.
- Každá tabulka obsahuje řádky a sloupce. Sloupce se nazývají atributy a řádky záznamy. Každá tabulka obsahuje specifické záznamy a slouží k ukládání specifických typů dat.
- V relační databázi se tabulky spojují pomocí klíčů. Klíče jsou jedinečné identifikátory, který identifikuje každý záznam jednotlivých řádků v tabulce. Tento klíč se nazývá primární klíč. Dále existuje cizí klíč, který se používá k propojení řádků v jedné tabulce s řádky v jiné tabulce.
- Relace (Relation) označuje množinu dat, které jsou uloženy v tabulce.

Velmi důležité je podotknout, že relace označuje tabulku jako celek, tedy množinu dat reprezentujících určitý typ informací. Zatím co záznam je pouze řádek v tabulce, který obsahuje specifické informace o entitě.

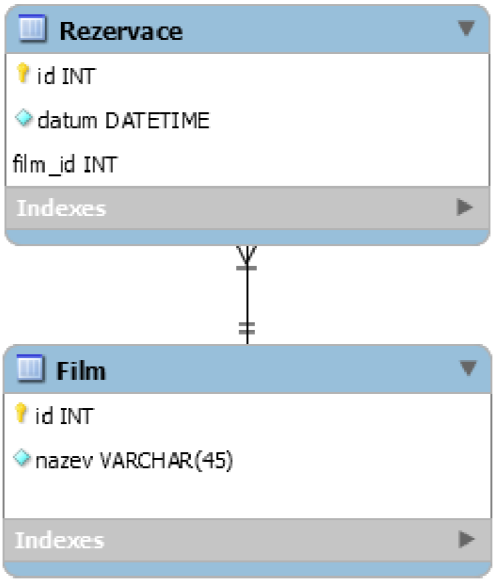
- Kardinalita označuje vztah mezi dvěma entitami v databázovém modelu a určuje, kolik záznamů v jedné relaci může odpovídat jednomu záznamu v jiné relaci. Typy spojení jsou:

1. mezi tabulkami není spojitost
2. 1:1 označuje, že každý záznam v jedné relaci odpovídá pouze jednomu záznamu v druhé relaci a naopak.
3. 1:n označuje, že jeden záznam v jedné relaci odpovídá více záznamům v druhé relaci.
4. m:n označuje, že více záznamů v jedné relaci může odpovídat více záznamům v druhé relaci a naopak. V tomto případě vzniká vazební tabulka.

- Multiplicita označuje počet vztahů, které mohou být mezi dvěma entitami.

- Optionality (volitelnost) se v SQL používá pro označení povinnosti atributu v databázové tabulce. Atribut může být nepovinný (optional) nebo povinný (mandatory). Povinnost sloupce je často využívána u cizích klíčů, kde cizí klíč může nebo nesmí být nulový (NULL – NOT NULL). (Vostrovský, 2004) (M., a další, 2007) (Kozlovský, a další, 2006)

Nyní když máme definované základní pojmy, pojďme si uvést příklad:



Obrázek 4: Příklad klíčů v relační databázi (vlastní zpracování)

Tento příklad popisuje zjednodušený model lístků do kina. Koupi lístku člověk vytvoří rezervaci na určitý datum a na konkrétní film. Kardinalita mezi tabulkami Rezervace a Film je 1:n, což znamená, že n rezervací může mít právě jeden film a jeden film může mít n rezervací.

V tabulce Rezervace je primární klíč „id“ a cizí klíč „film_id“. Cizí klíč v tabulce Rezervace je primární klíč v tabulce Film.

Datové typy v databázových systémech mohou být odlišné, avšak tyto základní typy jsou k dispozici ve většině SQL databází.

- INTEGER je celé číslo, např.: 1, 2, -5, -6
- FLOAT nebo DOUBLE je číslo s desetinnou čárkou, např.: 1.5, 3.14
- VARCHAR nebo CHAR je textový řetězec s určitou délkou

- BOOLEAN je datový typ představující hodnotu true nebo false, pravdu nebo nepravdu (1 nebo 0)
- DATE nebo TIMESTAMP je datum a čas, podle standardu ISO 8601 je formát pro datum definován jako YYYY-MM-DD a pro čas jako HH:MM:SS

(Vostrovský, 2004) (M., a další, 2007) (Kozlovský, a další, 2006) (Moved)

Každý systém řízení báze dat může definovat své vlastní specifické datové typy, ale tyto základní datové typy by měly být k dispozici ve většině SQL databází. Při návrhu databáze je důležité pečlivě zvážit, jaké datové typy použít pro jednotlivé sloupce v tabulkách, aby bylo zajištěno efektivní ukládání a rychlý přístup k datům.

V MySQL databázi existuje konvence pro pojmenování tabulek, která obsahuje několik pravidel. První pravidlo je používání malých písmen a nahrazování mezer podtržítky. Dalším pravidlem je vyhýbat se číslům v názvech tabulek a používat srozumitelné názvy, které jednoznačně popisují obsah tabulky. Dále je důležité, aby název tabulky vystihoval její účel a byl přehledný. Pro omezení délky názvu a zachování přehlednosti je žádoucí, aby název tabulky neobsahoval více než 64 znaků. Je také doporučeno vyhnout se použití předpon v názvech tabulek. (Ananda , 2015)

S relačními databázemi dále souvisí kaskády(cascade) a triggery. Kaskády jsou akce, které se automaticky provádějí při změně záznamu (např. při výmazu, aktualizaci nebo novém záznamu). Akce se provádí na úrovni databáze a mohou mít za následek další změny v jiných tabulkách, které jsou s první tabulkou propojeny. Kaskády jsou použity k udržování integrity dat v relační databázi. Tabulka obsahující cizí klíč je považována za nadřazenou, zatím co tabulka s vlastním klíčem je považována za podřízenou. (Ranga, 2019)

Trigger(spouštěč) je speciální typ uložené procedury v relačních databázích, který automaticky vykoná akci při určité události. Například při změně údajů v tabulce, nebo při vytvoření nového záznamu. V relačních databázích je běžné používat trigger při vložení nebo aktualizování řádku. Triggery mohou provádět jednu nebo více akcí, a dokonce volat další triggery. Triggery se často používají k udržování integrity dat a k automatizaci určitých úloh v databázi. (Vystavěl, 2021)

3.6.1. Systém řízení báze dat (SŘBD)

Systém řízení báze dat (SŘBD), anglicky database management systém (DBMS) je software, který umožňuje přístup a správu relačních databází. Pojem SŘBD se často zaměňuje za pojem databázový systém. SŘBD umožňuje vytvářet, upravovat, manipulovat a mazat data v relačních databázích. Také poskytuje nástroje pro dotazování, indexování a správu integrity dat. SŘBD je rozhraní mezi aplikačními programy a uloženými daty. Používá jazyk SQL k práci s daty v relační databázi.

3.6.2. Normalizace Databáze

Normalizace databáze je postup, který mění strukturu dat tak, aby využívaly výhody relačního modelu dat. Normalizovaná data jsou efektivnější v ukládání, vyhledávání, třídění, zpracovávání, a celkově se s nimi mnohem lépe pracuje.

0NF – Nenormalizovaná databáze.

1NF – Každý atribut obsahuje pouze atomické hodnoty.

2NF – Každý neklíčový atribut tabulky musí být plně funkčně závislý pouze na primárním klíči tabulky, tedy musí být závislý pouze na celém primárním klíči a ne na jeho částech.

3NF – Každý neklíčový atribut tabulky musí být nezávislý na jiných neklíčových atributech, tedy musí být závislý pouze na primárním klíči nebo jiných klíčových atributech tabulky.

BCNF (Boyce-Coddova normální forma) – Každá neprimární složka je závislá na primárním klíči.

4NF – Relace popisuje pouze příčinnou souvislost mezi klíčem a atributy.

5NF – Relaci již není možno bezeztrátově rozložit.

(Vostrovský, 2004) (Merunka, a další, 1998)

3.6.3. Databázová integrita

V databázi existuje integritní omezení. Je to soubory pravidel, který slouží k zajištění kvality a konsistence dat v databázi. Dá se to považovat za určitou prevenci bezpečnosti dat, kde struktura dat je uspořádaná tak, aby nenastaly chyby. Existuje entitní, referenční a doménová integrita. (Vostrovský, 2004)

Entitní integrita spočívá v tom, že každá tabulka musí mít primární klíč, který reprezentuje unikátní hodnotu v každém řádku. (Vostrovský, 2004)

Referenční integrita spočívá ve správnosti vztahů tabulek. Dá se definovat cizím klíčem v entitě a tím ukazuje vztah závislosti dvojice entit. (Vostrovský, 2004)

Doménová integrita se dá považovat jako kontrola, zda data spadají do množiny přípustných hodnot. (Vostrovský, 2004)

3.6.4. Relační a objektový model

Relační model využívá tabulky, kde každá tabulka reprezentuje konkrétní entitu a sloupce v tabulce reprezentují atributy této entity. Relační databáze se skládají z tabulek a vztahů mezi nimi, které jsou definovány pomocí klíčů. Tyto klíče umožňují propojování dat z různých tabulek a zajišťují integritu dat v databázi. Relační databáze používají SQL pro manipulaci s daty a dotazy na databázi.

Objektový model oproti relačním modelům reprezentuje data jako objekty s atributy a metodami. Tyto objekty mohou a nemusí mít hierarchii. V objektovém modelu existuje navíc dědění, které umožňuje potomkům rodičovských objektů, aby dědili jejich atributy a metody. Propojení objektů vzniká pomocí dílčích vazeb. Objektové databáze jsou tedy založeny na objektech a jejich vztazích, které jsou reprezentovány pomocí objektových vazeb.

Hlavním rozdílem mezi relačním a objektovým modelem je tedy způsob, jakým jsou data organizována. Relační databáze používají tabulky a klíče pro reprezentaci dat a vztahů mezi nimi, čímž vzniká struktura dat. Zatímco objektové databáze používají objekty s atributy a metodami pro reprezentaci dat a vztahů mezi nimi. Oba modely mají své výhody a nevýhody

a hodí se pro jiné účely. Závisí tedy na konkrétním použití a případě potřeby aplikace. (Akhtar, 2022)

3.7. Bezpečnost

Bezpečnost dat má hlavní tři zásady, kterými se zabývá. Jsou to: důvěrnost, integrita a dostupnost.

- Důvěrnost dat – zajišťuje, že data jsou přístupná pouze autorizovaným uživatelům. Informace jsou chráněny proti neoprávněnému přístupu a zneužití.
- Integrita dat – znamená, že data jsou kompletní a neporušená, tedy nebyla změněna nebo poškozena neoprávněnou osobou nebo procesem.
- Dostupnost dat – znamená, že jsou data vždy k dispozici. Jsou přístupná také pro ukládání, čtení a zpracování.

Tyto tři zásady se také označují jako CIA (Confidentiality, Integrity, Availability). Dále je důležité, aby součástí bezpečnosti byla odpovědnost, která zajišťuje správu a ochranu informací v organizaci. Je důležité si k těmto zásadám také definovat, co jsou to hrozby, rizika a útoky.

Hrozba – potenciální příčina nežádoucího jevu nebo incidentu, který může vést k poškození systému. *„A threat is defined as some entity that may be capable of attacking or affecting the organization's infrastructure.“* (Tipton, a další, 2007)

Riziko – Riziko je možnost, že k dané hrozbě dojde a ona tato poškození skutečně způsobí, v rámci hodnocení se riziko vyčísluje.

Útok – je akce nebo série akcí zaměřené na poškození, zničení nebo získání dat. Typické metody útoků pro získání dat jsou například phishing, man in the middle nebo social engineering. (Tipton, a další, 2007) (Easttom, 20232)

3.8. Architektonický vzor MVC

MVC (Model-View-Controller) je architektonický vzor pro tvorbu uživatelských rozhraní a aplikací. MVC rozděluje aplikaci na tři základní složky: Model, View a Controller. Model představuje datovou vrstvu aplikace, View představuje výstupní rozhraní uživatele a Controller slouží jako propojovací část mezi modelem a view. Všechny tři složky spolupracují tak, aby uživatelské akce ve view byly přeposílány controlleru, který komunikuje s modelem a na základě hodnot z modelu se aktualizuje view. Tímto způsobem se oddělují vrstvy dat a logiky od uživatelského rozhraní aplikace. Díky rozdělení složek lze vyvíjet frontend a backend nezávisle na sobě, lze snadno upravovat jednotlivé části a zároveň držet spojitost. MVC architektura se často používá pro vývoj webových aplikací, ale dříve vznikla pro vývoj desktopových aplikací. Avšak MVC architektura nemusí být použita pouze pro webové aplikace, ale lze jí použít i pro desktopové nebo mobilní aplikace. (Javatpoint, 2022) (GeeksforGeeks, 2022)

3.9. Agilní metodiky

Agilní metodiky jsou souborem přístupů a praktik pro vývoj software, které se soustředí na rychlou dodávku software a na to, aby byl vývojový proces co nejvíce přizpůsoben změnám v požadavcích zákazníka a prostředí, ve kterém se projekt odehrává. Agilní metodiky se většinou zaměřují na menší týmy a menší projekty, které vyžadují rychlou a častou iteraci a zpětnou vazbu.

Agilní metodiky se vyvíjely jako alternativa ke klasickým vodopádovým metodikám vývoje software. Klasický vodopádový přístup se zaměřuje na plánování a dokonalou specifikaci požadavků v počáteční fázi projektu, což může vést k problémům, když se během vývoje objeví nové požadavky nebo se objeví technické problémy.

Agilní metodiky přizpůsobují vývojový proces tak, aby byl co nejvíce pružný a schopný rychle reagovat na změny a nové požadavky zákazníka. Toto se často provádí pomocí krátkých iterací vývoje, kdy jsou vytvářeny funkční kusy software, které jsou průběžně testovány a upravovány na základě zpětné vazby. V agilním přístupu je kladen důraz na komunikaci a spolupráci, jak mezi členy týmů, tak se zákazníky. Agilní metodiky jsou stále více populární

a mezi nejznámější patří Scrum a XP (Extrémní Programování). (Hema, a další, 2020) (Beck, 2002)

3.9.1. Scrum

Scrum je agilní metodika, která se zaměřuje na flexibilní vývoj software v iteracích. Klade důraz na lepší spolupráci v týmu a se zákazníkem, za účelem zvýšení transparentnosti a efektivity vývoje pomocí krátkých sprintů (iterací). Během sprintu jsou vymezeny úlohy a jejich priority. Tým pracuje na prioritizovaných úlohách a průběžně se snaží doručit funkční kusy software. Součástí scrum je pravidelné setkání a retrospektivní zhodnocení práce. Tímto způsobem se tým průběžně hodnotí a zlepšuje tak svou výkonnost, práci a spolupráci. (Hema, a další, 2020)

3.9.2. Extrémní Programování (XP)

Extrémní programování (XP) je agilní metodika, která se zaměřuje na kvalitu kódu a zákaznickou spokojenost. Je zde také kladen důraz na spolupráci, komunikaci, testování, refaktorování kódu a iterativní vývoj. (Beck, 2002)

3.10. RPG Hry

RPG hry, v češtině „Hra na hrdiny“, jsou žánr her, který vychází ze stolní hry Dungeons and Dragons (D&D). Role-play aspekt představuje hráče, který se vžije do postavy a podle té role se i tak ve hře chová. S postupným vývojem her se tento role-play aspekt stále udržel, ale žánr se rozvinul o další aspekty jako jsou atributy postav, úrovňové systémy, předměty, kouzla a další. Při hraní her je uživatel postaven do určité situace a role. Tyto situace a role mohou hráči nabídnout originální zážitek a případně ho něco naučit. Stejně tak jako v reálném životě, lidé často čelí různým situacím a jsou zastánci různých rolí v těchto situacích.

Existuje rčení "Život jako hra". Pomocí her dokážeme simulovat různé okolnosti a tím hráče lze lépe připravit a naučit ho reagovat na obdobné faktory. Hry jsou výborným nástrojem pro rozvoj jednotlivců v různých oborech.

3.10.1. Výhody a nevýhody hraní her

V dnešní době je pro mnoho lidí běžné, že vlastní alespoň jedno mobilní zařízení, jako jsou chytré telefony či tablety, a tyto zařízení jsou pro mnoho lidí neodmyslitelnou součástí každodenního života. Jednou z možností, jak tato zařízení využít, jsou počítačové hry. Hraní her může mít pozitivní vliv na psychické zdraví jedinců. V roce 2010 na vědeckém setkání společnosti yank Pain Society bylo poukázáno na fakt, že videohry jsou účinným způsobem snížení úzkosti nebo bolesti způsobené chronickým onemocněním. Tento pozitivní účinek her na psychické zdraví je pravděpodobně způsoben tím, že hraní her dokáže odvést pozornost od každodenních stresorů a pomoci jednotlivcům zrelaxovat se. (Vidyapeeth Maharashtra, 2021)

Další obrovskou výhodou hraní video her je zlepšení nebo vyvinutí vlastností jedince. Hraní video her může mít pozitivní dopad na kognitivní funkce jednotlivců. Jedním z hlavních výhod hraní her je zlepšení celkového kognitivního myšlení. To zahrnuje různé aspekty jako koncentraci, pozornost, komunikaci, rychlost myšlení, schopnost pochopení informací, kreativita a lepší učení se. Další výhodou hraní her je zlepšení vůdčích schopností, spolupráce a disciplíny. Některé hry vyžadují od hráčů, aby převzali vedení a rozhodovali o dalších krocích týmu. Hraní her také může pomoci zlepšit schopnost spolupráce, protože mnoho her vyžaduje, aby hráči pracovali společně k dosažení společného cíle. (Granic, a další, 2014)

Počítačové hry bohužel mohou být dvousečný meč. Nadměrné nebo nutkavé používání počítačových her může narušovat každodenní život. Člověk se stává závislým, když začne přerušovat sociální kontakt s ostatními jedinci a místo toho upřednostňuje spíše dosahování herních úspěchů než úspěchů v reálném životě. Závislost způsobuje náhlé změny nálady, abstinenční příznaky, ztrátu soustředění, relaps, změnu tolerance uspokojení z počítačové hry a konfliktnost jedince. Závislost na počítačových hrách je stejná, jako každá jiná závislost. Nadměrné hraní počítačových her může mít negativní dopad hlavně na psychické zdraví jednotlivce. Pokud je narušeno psychické zdraví z důvodu závislosti, tak v některých případech se zdravotní problémy mohou promítnout i na fyzické zdraví, jak v podobě psychosomatiky, tak v podobě opravdových fyzických problémů. Existuje případ z roku 1980, kde jedinec, který nadměrně hrál video hry a stal se závislým, měl problémy s kůží, klouby a svaly z důsledku přemáhání těla na stiskávání tlačítek a joysticku na herních strojích. (Weinstein, 2010)

4. Vlastní práce

Tato část práce je zaměřena na aplikování nabytých znalostí z teoretické části. Práce vychází z již existující databáze a ze základu webové aplikace, které vznikly v bakalářské práci autora. Díky již zhotovenému základu bude práce zaměřena na komplexnější rozšíření a lepší propracování aplikace.

V této části jsou stanoveny funkční a nefunkční požadavky práce, které jsou následně analyzovány a na základě této analýzy je tvořena aplikace za pomocí agilních sprintů. Po analýze následuje část implementace, kde jsou popsány prvky práce a jakým způsobem bylo postupováno. Nejdříve je popsána databáze, její elementární prvky a postupující myšlenky a principy tvorby databáze. Dále je popsána základní bezpečnost aplikace a principy bezpečnosti na straně serveru a klienta. Práce přechází do části popředí stránky, kde je popsáno uživatelské rozhraní, design a jednotlivé části stránky. Po části uživatelské rozhraní je popsáno funkční prostředí jednotlivých částí stránek a následně jsou popisovány varianty soubojových systémů a jejich implementace. Ke konci této části jsou ukázky bojového systému a kódu v pozadí aplikace. V závěru je sekce popisující kvantitativní testování diplomové práce.

4.1. Funkční požadavky

Funkční požadavky této práce byly definovány:

- Registrace – umožnění uživateli vytvořit postavu. Dále zajistit bezpečné ukládání uživatelských dat a ověřování údajů při registraci. Vysoká priorita.
- Přihlášení – možnost uživatele se přihlásit a bezpečně ověřit identitu uživatele. Vysoká priorita.
- Přesměrování uživatele z menu do konkrétní sekce – Zajištění intuitivní navigace na stránce pro uživatele. Zobrazení relevantních informací v sekci, ke které uživatel přistoupil. Snadné možnost vrácení se na hlavní stránku. Vysoká priorita.
- Profilový systém – Možnost zobrazení profilu uživatele nebo jiného uživatele. Možnost editace ikonky postavy uživatele. Možnost interakce s jinými uživateli skrz profilový systém. Vysoká priorita.
- Oblékání předmětů postavě – Možnost uživatele oblékat oblečení a předměty postavě s vizuální zpětnou vazba užívaných předmětů. Střední priorita.

- Vkládání kouzel do akčních políček kouzelné knížky postavy – Možnost uživateli zobrazit naučená kouzla. Zajistit, aby naučená kouzla mohla být využita v aktivním políčku postavy. Střední priorita.
- Změna ikonky postavy – Možnost uživatele změnit profilovou ikonku postavy pro lepší reprezentaci postavy. Nízká priorita.
- Interakce s předměty – Uživatel je schopen prodat předměty postavy a nakupovat nové. Zajištění správného zobrazování předmětu včetně jeho ceny a kvantity. Detailní popis předmětu na rozkliknutí. Vysoká priorita.
- Soubojový systém – Možnost uživatele soupeřit v reálném čase s různými nepřáteli. Zajistit správné vizuální indikátory v souboji. Více možností volby útoků v průběhu souboje. Zobrazení výsledků souboje. Vysoká priorita.

4.2. Nefunkční požadavky

Jelikož diplomová práce vychází z již existující bakalářské práce, tak nová verze prototypu těchto stránek bude splňovat následující nefunkční požadavky:

- Responzivní webové stránky – Webová aplikace musí být navržena tak, aby byla použitelná na různých zařízení s různými velikostmi displeje.
- Mobile first – Webová aplikace by měla být navržena s ohledem na mobilní zařízení jako primární zdroj návštěvnosti stránky. Po mobile first přistupu lze stránku upravovat pro specifikace ostatních zařízení.
- Uživatelsky intuitivní – Webová aplikace by měla být snadno použitelná a přístupná pro uživatele. Používání srozumitelných ikon, snadné ovládání, přehledné rozvržení stránek a jednoduché navigační prvky.

4.3. Analýza řešení

Na základě analýzy řešení bylo uváženo použití vhodných frameworky pro tvorbu webových stránek. Pro frontend React, pro backend Nette nebo Symphony, pro převedení relační databáze do objektové databáze Doctrine. Dále bylo uváženo použít css framework Bootstrap pro efektivnější práci s vizualitou stránek. Avšak z důvodu, že práce vychází z již zhotoveného základu databáze a aplikace, tak nebyl zvolen žádný framework. Dále byly navrženy dvě varianty tvorby soubojového systému.

4.3.1. Problematika registrace a přihlášení

Funkcionalita registrace a přihlášení je základním prvkem většiny webových aplikací. Avšak prototyp této webové aplikace bude vytvořen tak, aby každý účet obsahoval pouze jednu postavu a dále není od uživatelů vyžadován email v registračním formuláři. V reálném produktu by byl email vyžadován v registračním formuláři a účet uživatele by byl upraven tak, aby mohl obsahovat více postav. Tento způsob by pomohl uživateli lépe organizovat své postavy a účet, zatím co z pohledu hry by bylo jednodušší sbírat statistiky ohledně účtů a jejich postav.

Registrace a přihlášení musí být ošetřeno jak na straně serveru, tak na straně klienta. Pro ošetření na straně serveru je velmi vhodné kontrolovat formulář pomocí PHP a zároveň využít princip z bezpečnosti týkající se předpřipravených příkazů pro zamezení SQL injection. V reálném produktu by také bylo důležité klást důraz na šifrování hesla alespoň pomocí bezpečnostního hashovacího algoritmu SHA-256. Na straně klienta musí být správně validován formulář a musí být povolena relevantní délka vstupu.

Po registraci jsou údaje vloženy do databáze a nyní je uživatel schopen se přihlásit. Formulář musí být ošetřen tak, aby přihlašující se člověk byl řádně autorizován a aby uživateli po přihlášení byla udělena správná session. Do reálného produktu by určitě měla být přidána unikátnost session (ID) a kontrola platnosti session.

4.3.2. Problematika profilového systému

Profilový systém je esenciální pro zobrazení informací o postavě uživatelům. Uživatel vidět a být schopen spravovat svá data. Musí být schopen intuitivně manipulovat s předměty a kouzly postavy. Dále by měl mít možnosti nastavení svého profilu a možnosti interakce profilu s jinými uživateli.

V tomto prototypu aplikace je profilový systém složen ze záložek: Character, Inventory, Spellbook a Icons. Pro reálný produkt nebo další iteraci vývoje je doporučeno provést kvalitativní testování, konkrétně metodu A/B testování pro zvolení nejlépe možného prostředí pro profilový systém. Je důležité zjistit, které prvky profilového systému jsou vyhovující a intuitivní pro uživatele a celkově zjistit které varianty jsou nejefektivnější.

4.3.3. Problematika Soubojového systému

První varianta soubojového systému představuje pouze hru hráče proti počítači. V této variantě je nutné pouze implementovat část databáze, která bude kontrolovat stav souboje. Jakákoliv akce hráče volá koncový bod serveru, komunikuje s databází a zjišťuje, zda akce je validní. Po úspěšné odezvě koncového bodu je aktualizován client side. Je důležité, aby všechny akce hráče byly ošetřeny na obou stranách, jak na straně serveru, tak na straně clienta.

Druhá varianta představuje hratelnost a komunikaci více hráčů. Pro řešení bude užitá WebSocket technologie, která poskytne komunikační kanál pro uživatele. Po vstoupení do souboje bude vytvořena serverová místnost, do které se připojí hráči, kteří se účastní souboje. Bude vytvořena hierarchie pořadí, která bude určovat hráče, který je právě na tahu. Jakákoliv akce bude promítnuta na server a odeslána všem ostatním účastníkům. Tímto jsou veškeré akce sjednoceny a je důležité, aby veškeré inputy byly ošetřeny jak na straně clienta, tak na straně serveru.

Z důvodu časového omezení tohoto projektu byla pro vývoj soubojového systému zvolena první varianta a technologie WebSocket nebyla použita.

4.4.Implementace

V této části práce je popisován krok po kroku jakým způsobem byly implementovány jednotlivé části aplikace za účelem dokončení dílčích požadavků a cílů. Nejdříve bude popsána problematika databáze a její řešení. Následuje zabezpečení databáze a webového prostředí. Poté se pojednává o jednotlivých stránkách, jejich interface, designu a funkčnostech. Dále je popsán soubojový systém a jeho řešení. Na závěr je aplikace kvalitativně testována.

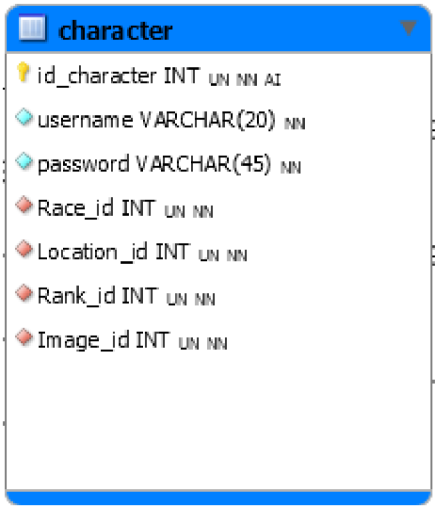
4.4.1. Architektura aplikace

Práce vychází z již zhotovené struktury webové aplikace z bakalářské práce. Na obrázku 33. viz. Příloha, je zobrazena architektura webové stránky, která byla využita jako základ pro další vývoj.

4.4.2. Databáze

Databáze vychází z předešlé bakalářské práce autora a je rozšířena a vylepšena o soubojové a interakční prvky. Pro uvedení do kontextu budou popsány pouze elementární prvky rozsáhlého databázového modelu, na které je tvořena návaznost a logika soubojového systému v aplikaci.

Základní myšlenka vychází z představy pohledu uživatele, který má postavu. S postavou může být zacházeno mnoha způsoby. Lze jí upravovat ikonku, oblékat předměty a také přiřazovat aktivní kouzla, která může používat v souboji. Uživatel je schopen nakupovat, prodávat a vytvářet předměty pro konkrétní potřeby úpravy postavy. Postavy jsou schopny interakce mezi sebou. Mohou se navzájem přidávat do přátel, vyzývat na souboje, vytvářet skupiny, obchodovat a psát si. Taková kombinace akcí může přinést uživateli zajímavou herní zkušenost s mnoha variacemi a způsoby hraní hry. Uživatelé jsou nabádáni k tvorbě vlastního herního stylu a zároveň se ve hře otevírá více způsobů, jak jí rozšířit a zdokonalit. Jeden z příkladů na rozšíření hry jsou úkoly, které vyžadují společné úsilí hráčů na poražení více nepřátel za předpokladu lepší odměny.



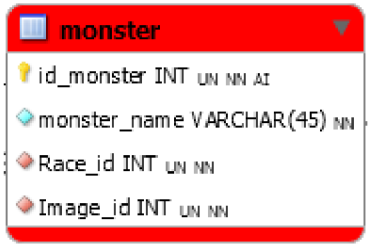
Obrázek 5: character – tabulka v databázi (vlastní zpracování)

Tabulka character je elementární tabulkou v databázi aplikace. Obsahuje ID (unikátní identifikátor) postavy, jméno a heslo. Heslo je samozřejmě zašifrováno. Dále postava obsahuje atributy skládající se z cizích klíčů. Jsou zde id rasy, lokace, hodnosti a obrázku. ID rasy reprezentuje rasu postavy, např.: člověk, elf, trpaslík a další. Lokace reprezentuje, v jaké oblasti se postava nachází. Hodnost je míra práv přiřazených k této postavě a obrázek je profilová ikonka postavy. Je důležité podotknout, že v reálném produktu by bylo velmi vhodné použít

i atribut s emailem, což ale při současném designu tabulky vede k omezení postavy na hráče. Tabulka postavy je dále vázána na tabulky předmětů, atributů, kouzel, přátelství, úkolů, profesí, klanů a soubojů. Mezi těmito tabulkami je často m:n vazba, a z tohoto důvodu vzniká vazební tabulka, která spojuje postavu a další tabulku.

Je vhodné dodat, že v praxi není vítané mazání postav a ostatních dat. Místo smazání dat je vhodné vytvořit atribut, který bude sloužit jako „měkké odstranění“ (soft delete).

Soft delete je pouze označení záznamu jako neaktivní nebo neplatný, aniž by byl skutečně odstraněn z databáze.



Obrázek 6: monster – tabulka v databázi (vlastní zpracování)

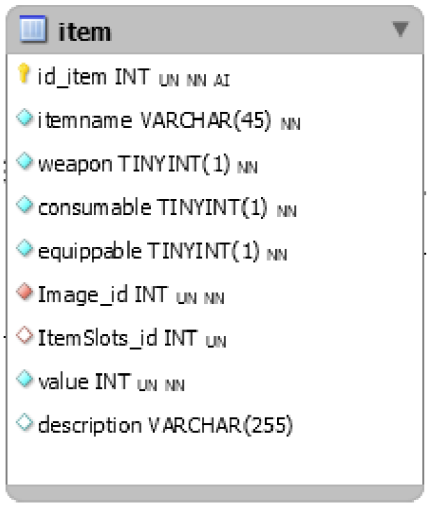
Tabulka monster představuje příšery, které hráč může potkat na svých výpravách. Příšera obsahuje atributy ID a jméno, a také cizí klíče rasy a obrázku. Tabulka příšery také obsahuje vazby na kouzla a políčka pro kouzla, kde existuje vazba m:n a tedy vznikají vazební tabulky, které spojují příšeru s danými tabulkami.

Je možné argumentovat, že tabulka postavy a příšery je velmi podobná, proč tedy není oddělená? Důvod je takový, že v relační databázi neexistuje dědění, ale dá se svým způsobem simulovat. Je tedy možné vytvořit všeobecnou tabulku postavy, která bude mít společné atributy ID, jméno, a dále cizí klíče rasy a obrázku. Postava by musela být rozšířena o další tabulky hráče a příšery pomocí vazby 1:1, které by měly doplňující informace. U hráče by to mohly být atributy heslo a email. A u příšery by pravděpodobně žádné doplňující informace nebyly. Zbytek vazebních tabulek by následně byl navázán na všeobecnou tabulku postavy a model by byl plně funkční. Bohužel autor vychází z již vytvořeného datového modelu a základu aplikace z předešlé práce. Z tohoto důvodu je navázáno na tento způsob řešení. Samozřejmě nově navržený způsob řešení má předpoklady k větší robustnosti a mohl by zamezit budoucí

nadbytečnosti v části databáze týkající se bojového systému, ale o tom až dále. Aktuální způsob řešení databáze nelze označit jako nevhodný.

Obrázek 7: stat – tabulka v databázi (vlastní zpracování)

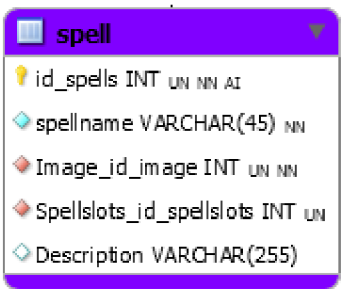
Stat je tabulka v databázi, která představuje atributy postav na příklad: síla, inteligence, obratnost, level, zkušenosti, násobnost sílou, a další. Tyto atributy mohou být přiřazeny jak hráči a příšeře, tak i kouzlu nebo předmětu. Tímto vzniká více variací a kombinací herních stylů v souboji. Tabulka obsahuje unikátní identifikátor a název statu(atributu). Jak zde bylo zmíněno, tak tabulka je propojena na hráče, monstra, kouzla a předměty ve vazbě n:m a tím vzniká více vazebních tabulek.



Obrázek 8: item – tabulka v databázi (vlastní zpracování)

Tabulka item slouží k uchování dat předmětu. Obsahuje základní atribut id, jako primární klíč a itemname, jako název předmětu. V tabulce existují tři atributy, které mají datový typ TINYINT o velikosti 1. Tento datový typ je náhražka za booleovskou proměnnou, která bohužel v MySql není. Tyto tři atributy slouží k nastavení funkcionality předmětu a tím je řešena logika předmětu. Pokud je hodnota atributu 0, tak neplatí a pokud je 1, tak platí. Weapon je atribut, který klasifikuje předmět jako zbraň. Atribut consumable zapříčiní, že předmět je možno použít, například: předmět „Lektvar zdraví“ přidá hráči 5 životů. Atribut equippable umožňuje předmětu, aby byl vložen do oblékacího políčka. S tímto atributem souvisí atribut

ItemSlots_id. ItemSlots_id je nepovinný cizí klíč spojující předmět s políčkem pro oblečení. Dále je zde hodnota předmětu, popis předmětu a cizí klíč pro obrázek.

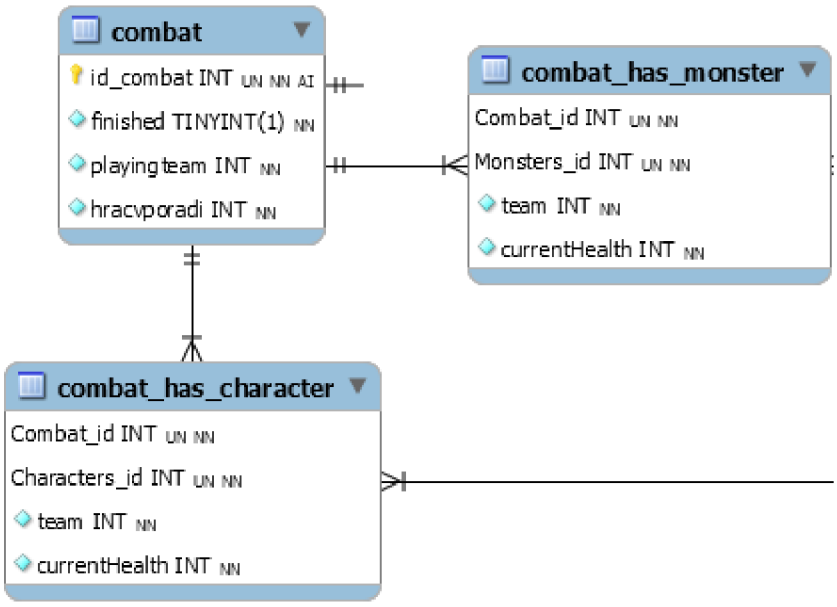


Obrázek 9: spell – tabulka v databázi (vlastní zpracování)

Tabulka spell představuje kouzlo. Kouzlo má id, název, popis a cizí klíče obrázku a políčka pro kouzlo. Kouzla mají podobný princip jako předměty. Stejně jako předměty, kouzla lze vkládat do políček, která mají svou roli v souboji. Kouzla mají vazbu n:m na tabulku stat, kde tato vazba určuje atributy kouzla.

Obrázek 10: combat – tabulka v databázi (vlastní zpracování)

Combat je tabulka představující stav souboje. Id souboje je unikátní klíč. Atribut finished představuje booleovskou proměnnou, kde 0 znamená stále probíhající souboj a 1 znamená dokončený souboj. V tabulce se také kontroluje, jaký hráč je v pořadí a jaký tým je aktuálně na řadě. Tato tabulka z části řeší logiku herního souboje.



Obrázek 11: Elementární tabulky pro soubojový systém (vlastní zpracování)

Tabulky combat_has_character a combat_has_monster jsou základní tabulky pro soubojovou strukturu a logiku. Jak již bylo zmíněno, na základě předešlé práce je struktura postavy hráče a monstra rozdělena. Kvůli tomuto faktu jsou tyto tabulky též rozděleny.

Již bylo zmíněno robustnější řešení, které by zapříčinilo sloučení těchto dvou tabulek. V robustnějším řešení by vznikla pouze tabulka „combat_has_character“, kde by tato tabulka představovala jakoukoliv postavu bez rozdílu, jestli je to hráč, nebo příšera. Avšak jediný rozdíl v aktuálním přístupu je struktura. Soubojový systém funguje na stejném principu, kde hráč nebo příšera mohou být ve stejném týmu, nebo v jiném. Pořadí je vypočítáváno až v aplikační vrstvě s atributy hrajícího hráče a hráče v pořadí. Tyto atributy jsou tedy jen čísla, ke kterým je důležité znát kontext.

Tabulky hráčů a monster v souboji také obsahují id, které určuje zařazení do souboje, id hráče nebo příšery, číslo týmu a také stav aktuálního zdraví. Zbytek hodnot je dopočítáváno v aplikační vrstvě a není potřeba aby hodnoty byly uloženy v databázi.

Dále existuje tabulka, která zaznamenává průběh souboje a tímto způsobem je možné si zpětně přehrát souboj. Tento elementární tvar souboje může být rozšířen i o efekty a stavy postav. Avšak dle analýzy by tento způsob rozšiřování databáze měl smysl pouze v případě hráče proti počítači. Pokud by soubojový systém byl implementován na řešení více hráčů a byl by použit websocket, tak se stavy a efekty hráčů mohou ukládat v paměti hubu (serveru).

4.4.3. Zabezpečení webových stránek a aplikace

Zabezpečení webových stránek je komplexní téma, kde existuje mnoho metod, nástrojů a vrstev pro zajištění bezpečnosti. Je vhodné zabezpečit jak stranu klienta, tak stranu serveru.

V této práci byl kladen důraz hlavně na bezpečnost aplikace zejména na straně serveru. Veškeré formuláře a vstupy jsou ošetřeny předpřipravenými příkazy, které brání útokům typu SQL injection. Citlivé informace jsou šifrovány, oprávnění je extra ošetřeno a administrátorské rozhraní opravdu vidí jen oprávněný uživatel.

```php
function Select($dotaz,$parametr,$conn, $type="i"){
    $stmt = $conn->prepare($dotaz);
    $stmt->bind_param($type,$parametr);
    $stmt->execute();

    $result = $stmt->get_result();
    $results = array();
    while($row =$result->fetch_assoc()){
        array_push($results,$row);
    }
    return $results;
}
```

Obrázek 12: Příklad funkce ošetřeného select příkazu (vlastní zpracování)

Na tomto obrázku je funkce, která je využívána na dotazování databáze pomocí předpřipraveného příkazu. Funkce má 4 vstupy. První vstup očekává SQL dotaz v řetězcové podobě. Druhý vstup je proměnná parametru, který je očekáván v dotazu. Třetí vstup je proměnná s údaji na připojení do databáze. A poslední vstup je volitelný a určuje jaký datový typ druhého vstupu má být.

```php
function Insert($dotaz,$parametr,$conn){
    $stmt = $conn->prepare($dotaz);
    $stmt->bind_param("i",$parametr);
    $stmt->execute();
}
```

Obrázek 13: Příklad funkce vložení do databáze za pomocí předpřipraveného příkazu (vlastní zpracování)

Zde na obrázku je vidět funkce, která vkládá data do databáze. Funkce používá předpřipravený příkaz, který slouží jako ochrana před SQL injection. Funkce má 3 vstupy. První vstup je SQL dotaz v řetězcové podobě. Druhý vstup je proměnná vstupního parametru a třetí vstup jsou údaje pro připojení k databázi.

Další mezery v bezpečnosti, které v aplikaci hrozí jsou převážně v soubojovém systému na straně klienta. I když bylo zvolena varianta souboje hráče proti počítači, tak je uživatel schopen na své straně modifikovat html prvky a rozhraní a následně se pokusit o zaslání jiných údajů na koncový bod, nebo se snažit hrát, když není na řadě. Všechny tyto hrozby jsou ošetřeny pomocí databáze a zatím nebyla nalezena další mezera v bezpečnosti v soubojovém systému.

Poslední mezera v bezpečnosti, kterou se v této aplikaci budeme zabývat je problematika volání skriptů ve více vláknech. V praxi je velmi časté, že aplikace má jiné chování na lokálním serveru než na veřejném. Častokrát záleží na webhostingu a doméně, kde zabezpečení serveru může být nastaveno na jinou úroveň než lokální prostředí. Celkově je doporučeno pro vývoj webové stránky, aby vývojové prostředí bylo co nejvíce podobné reálnému nasazení. Je velmi běžné, že volání skriptů může vytvářet mnoho problémů na serveru. Skripty nemusí být řádně ukončeny a po zavolání a vykonání funkce stále běží v procesu, kde zbytečně zabírají výpočetní paměť serveru.

4.4.4. UI a design

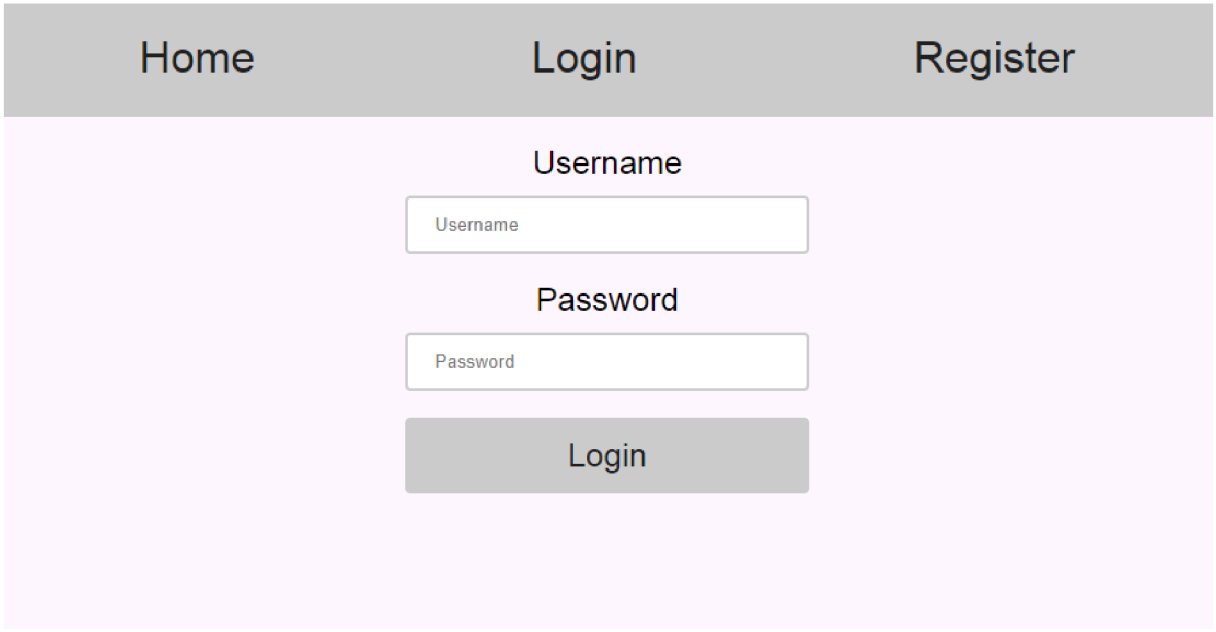
Struktura webové stránky aplikace je poměrně standardní. Obsahuje záhlaví, zápatí a hlavní obsah. Hlavní obsah je dynamický a při změně stránky se změní pouze hlavní obsah, tedy view, které uživatel zažádal.

```
$pages = $_GET["pages"] ?? "main";
$splitpage = (explode('?', $pages));
$pagename = $splitpage[0];

include_once("templates/header.php");
?>
    <div class="menu-content-wrap">
        <?php
        if (file_exists("templates/" . $splitpage[0] . ".php"))
            include_once("templates/" . $splitpage[0] . ".php");
        else
            include_once("templates/404.php");
        ?>
    </div>
<?php
include_once("templates/footer.php");
?>
```

Obrázek 14: Proměnná část hlavního obsahu (vlastní zpracování)

Na obrázku je vidět část kódu, který zpracovává parametry z části ULR adresy. Pokud je parametr prázdný nebo neexistuje, tak je předložena hlavní stránka. V URL se také získává název stránky a případně další parametry. Celkový obsah je zahrnut v záhlaví a zápatí stránky. Celou část kódu lze přirovnat k jednoduchému controlleru v MVC architektuře.

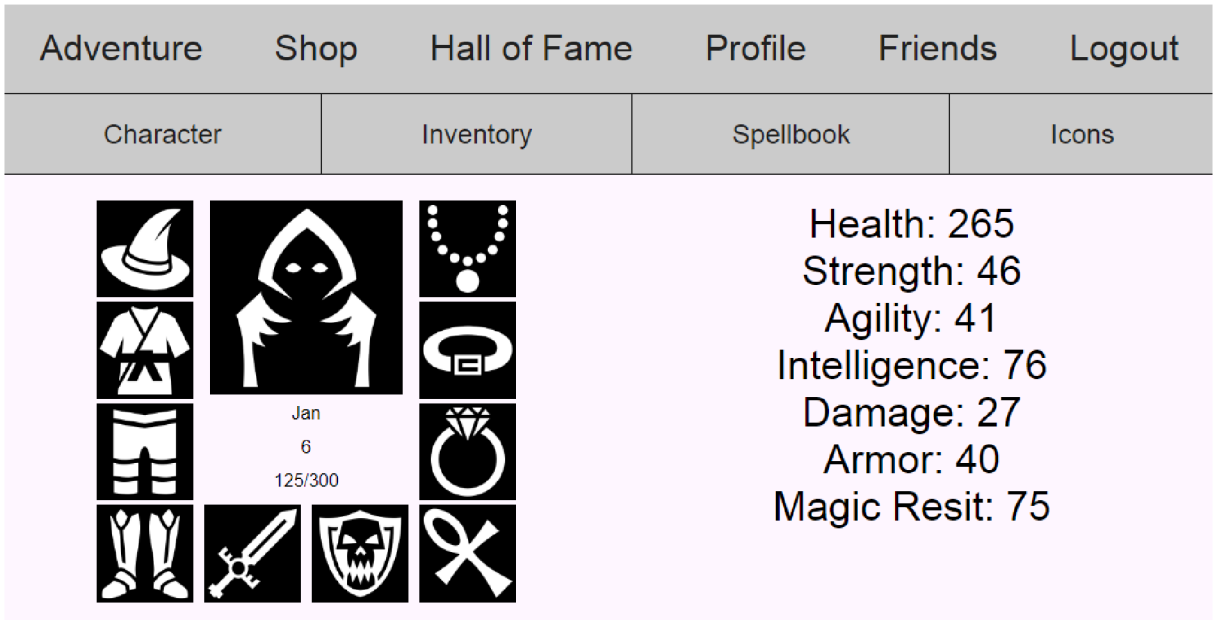
Výchozí stav stránky nabízí pouze malé množství možností. Uživatel má možnost se zaregistrovat nebo přihlásit, případně získat více informací o stránce v zápatí. Zápatí, záhlaví a menu jsou plně responzivní.



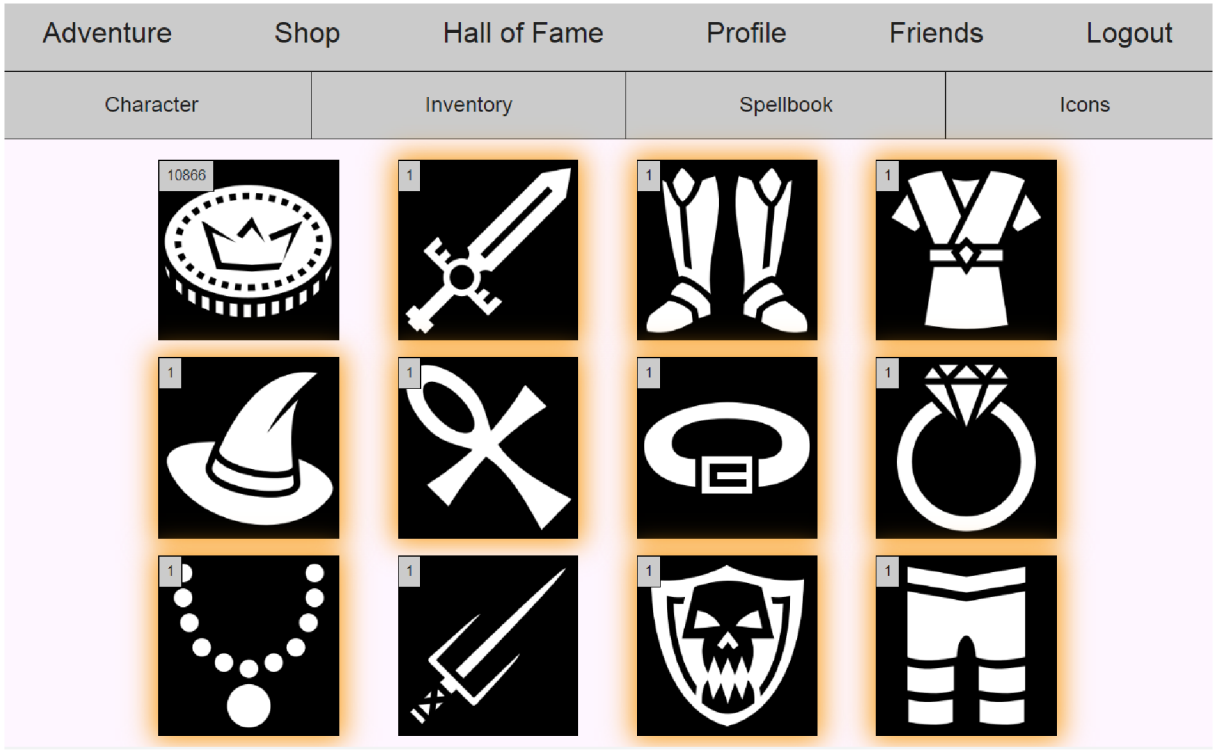
Obrázek 15: Přihlašovací formulář (vlastní zpracování)

Na obrázku je vidět záhlaví a hlavní obsah s přihlašovacím formulářem uvnitř.

Po úspěšném přihlášení je uživatel odkázán na profil své postavy. Profil postavy je složen z další vrstvy menu, které umožňuje uživateli více možností interakce s postavou.

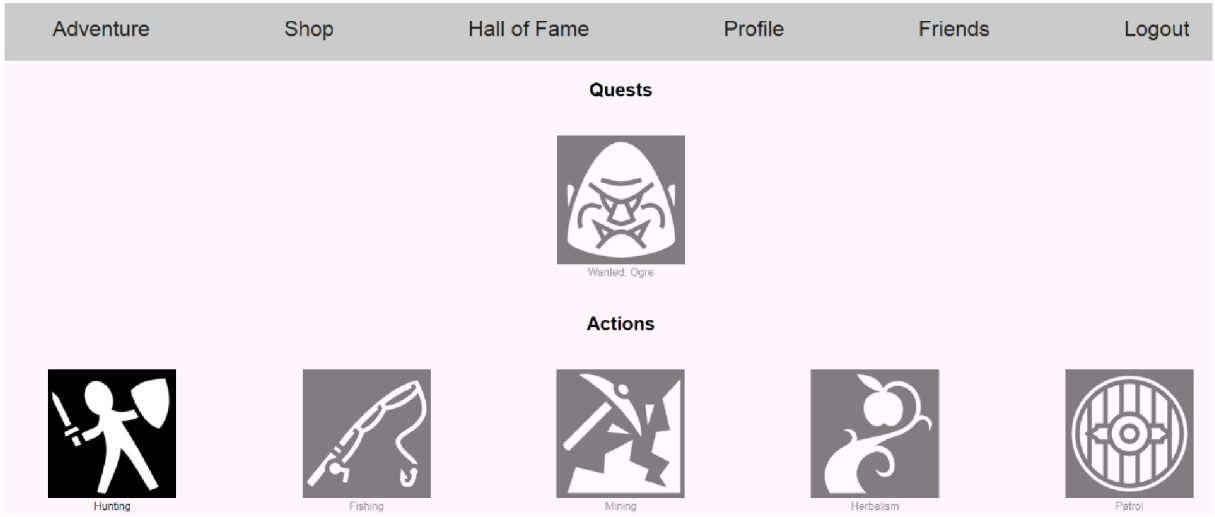


Obrázek 16: Profil postavy (vlastní zpracování)



Obrázek 17: Inventář postavy (vlastní zpracování)

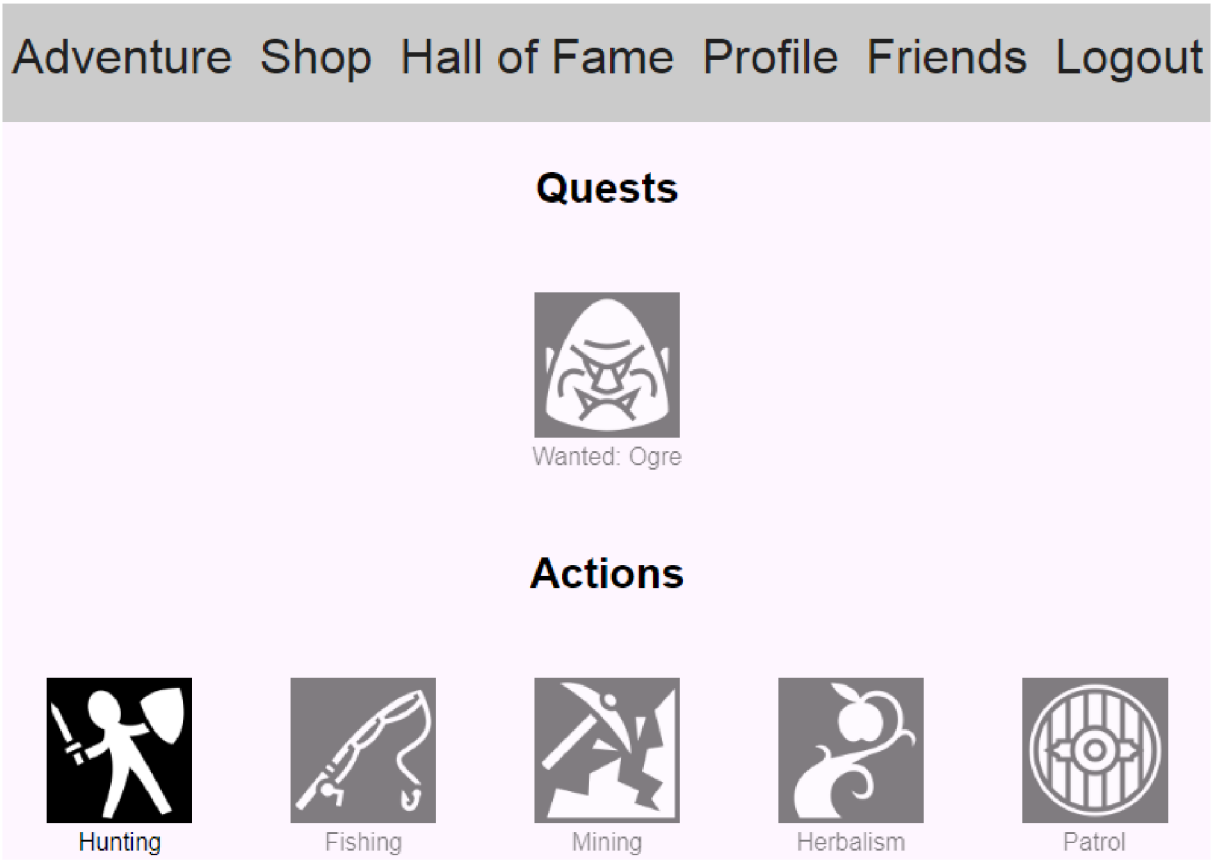
Na obrázku 17. je vidět inventář postavy. V inventáři má každá postava své předměty. Předměty mohou být různých typů, třeba poživatelné nebo s možností obléct je postavě. Oblečené předměty mají kolem ikonky oranžový indikátor.



Obrázek 18: Stránka dobrodružství (vlastní zpracování)

Stránka dobrodružství je důležitá pro uživatele, protože umožňuje uživateli zaúkolovat postavu a tímto dosáhnout pokroku ve hře. Zašedlé ikonky jsou momentálně nepřístupné pro postavu. V horní části jsou úkoly, které často vyžadují vyšší úroveň postavy a jsou značně

obtížnější. Políčko lovu pošle postavu na lov, kde potká náhodné nepřátelské monstrum, které musí porazit. Vítězství postavy nad příšerou přidává postavě pár bodů zkušeností a náhodný předmět. Soubojový systém je popsán níže v kapitole „Bojový systém“. Políčka rybaření, kopání a sbírání jsou akce, které zaúkolují postavu na určitý čas a postava za daný časový úsek sbírá suroviny, které může prodávat nebo z nich dále tvořit jiné předměty. Políčko hlídky zaúkoluje postavu v hlídkování. Za hlídkování postava dostane zaplaceno.

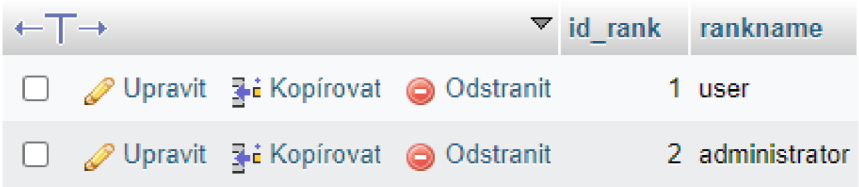


Obrázek 19: Stránka dobrodružství na mobilu (vlastní zpracování)

Na obrázku 19. je stránka dobrodružství, ale v mobilním zobrazení. Zde je vidět, že v porovnání se stránkou ve zobrazení na desktopové zařízení je stránka opravdu responsivní.

Obrázek 20: Administrátorské rozhraní (vlastní zpracování)

Administrátorské rozhraní je využíváno pro kontrolu dat. Pro nahlédnutí do administrátorského rozhraní je nutné, aby uživatel měl oprávnění v databázi. Bez oprávnění se uživateli administrátorské rozhraní v menu neukáže. Kdyby se do administrátorského menu chtěl kdokoliv chtěl dostat pomocí umělého odkazu, bude zamítnut a odkázán na domovskou stránku.



Obrázek 21: Hodnosti v databázi (vlastní zpracování)

Na obrázku 21. je část databáze z tabulky hodnosti, kde jsou vidět jednotlivé záznamy úrovní hodností. Tento princip může být rozšiřován a zároveň se stejný princip využívá v klánové hierarchii.

V administrátorském rozhraní lze všeobecně editovat, kontrolovat a mazat obrázky, rasy, postavy, předměty, kouzla, monstra a další. V záložce hráčů lze jednotlivým hráčům spravovat ikonku, jméno, rasu, úroveň, zkušenosti, atributy, předměty a kouzla s jejich atributy.

Dále v záložce předmětů lze vytvářet, editovat a mazat předměty. Je možné vytvářet nová políčka pro předměty a tím vzniká možnost rozšiřitelnosti hry o speciální předměty. Lze nahlédnout na jednotlivé předměty blíže s možností editace, kontroly a mazání jejich názvu,

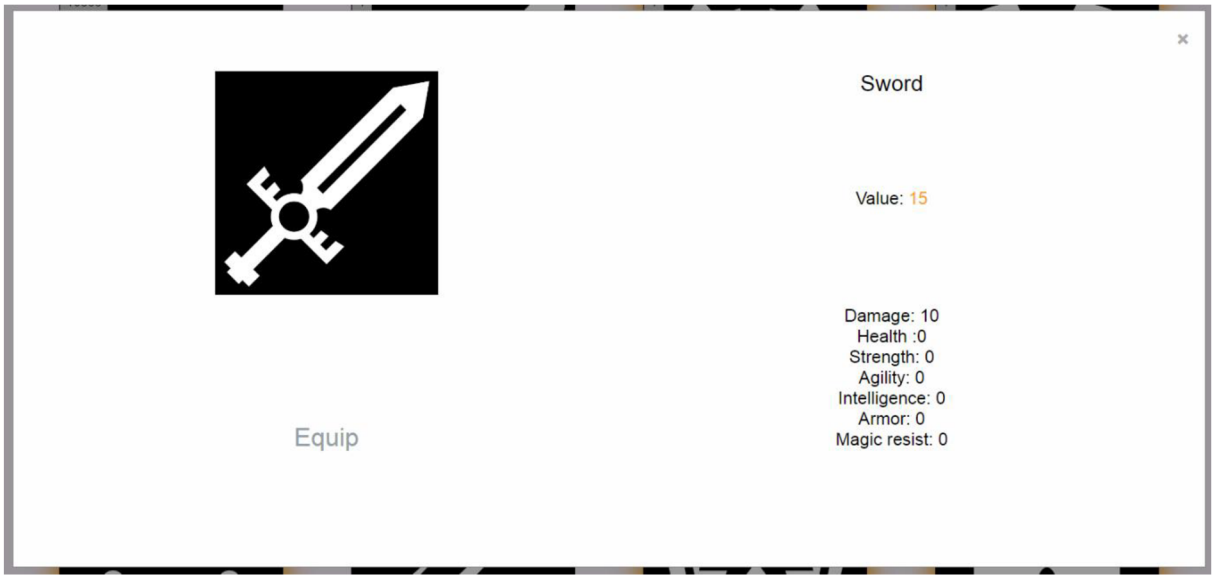
hodnoty, atributů, popisků a chování ve hře. Jednotlivé předměty mohou být též editovány o možnost, do jakého políčka mohou být oblečeny.

Záložka kouzel je velmi podobná záložce předmětů. Je zde možnost upravovat, editovat a mazat kouzla a políčka do kterých patří. Je zde také náhled na bližší informace k jednotlivému kouzlu. V náhledu je možnost mazání a změny obrázku kouzla, názvu, atribut a popisku. Záložka kouzel by dále mohla být rozšířena o návrh a editaci fungování kouzla v soubojovém systému. Avšak implementace takové vlastnosti je velmi náročná a z tohoto důvodu jsou kouzla striktně naskriptována v soubojovém systému.

Poslední záložka se týká příšer. Příšery jsou všechny postavy, které hráči mohou potkat na svých dobrodružství. Lze vytvářet, editovat a kontrolovat všechna monstra, stejně jako u hráčů. V náhledu na monstrum je velmi podobný systém jako u hráčů. Lze editovat, kontrolovat a mazat ikonky, jména, rasy, atributy a kouzla. V této aplikaci monstra nemají předměty, avšak nebylo by složité takovou vlastnosti přidat.

Funkčnosti rozhraní

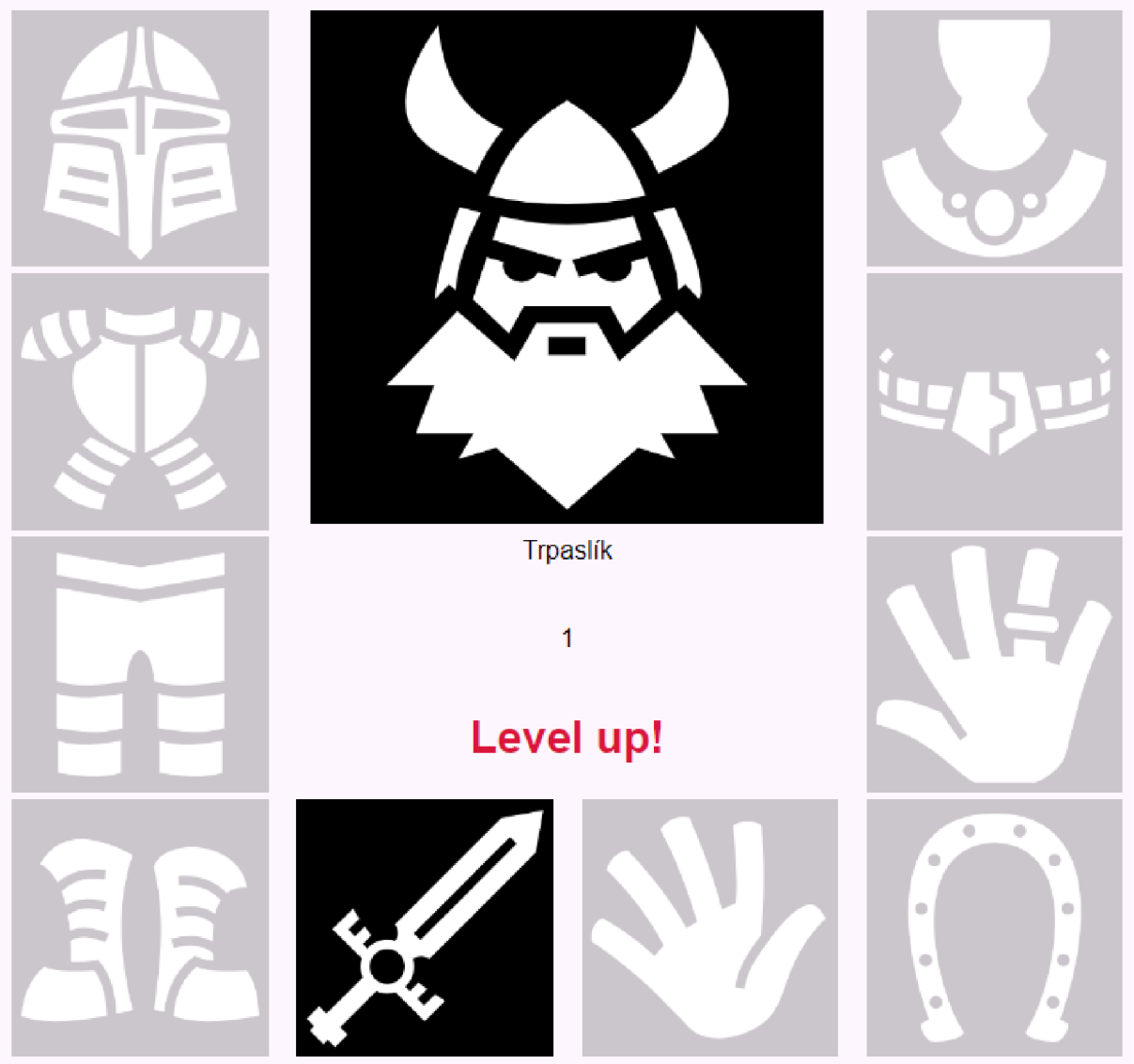
Nejdůležitější funkčností profilového rozhraní je manipulace s předměty. Po kliknutí na předmět je zobrazeno dialogové okno, který obsahuje informace o konkrétním předmětu. Detaily předmětu zahrnují ikonku, název, hodnotu, atributy a možnosti interakce s předmětem.



Obrázek 22: Dialogové okno předmětu meče (vlastní zpracování)

Tímto způsobem je velmi jednoduché pro uživatele pracovat s předměty. Předměty mohou být nasazeny nebo sundány postavě, za předpokladu, že mají políčko, do kterého je možné předmět vložit. Další typ předmětu je spotřební. Spotřební předměty mají v dialogovém okně možnost použití. Po použití předmětu je vykonána daná akce předmětu a předmět mizí z batůžku hráče. Tato funkcionalita je též přidána kouzlům hráče. Kouzla také mají ikonku, název a atributy, ale liší se v možnosti interakce. Jedno kouzlo nemůže být ve více políčkách, ale kouzla lze, na rozdíl od předmětů, dávat do různých políček a vytvářet tak unikátní herní strategie. Další důležitou částí pro uživatele ohledně předmětů je obchod, kde uživatelé mohou nakupovat a prodávat předměty.

Další vlastnosti profilového systému je možnost zvýšení úrovně postavy. Pro zvýšení úrovně postavy musí hráč porazit nepřátele a získat potřebný počet zkušeností na zvýšení úrovně. Zvýšení úrovně hráči poskytne výhodu v navýšení atributů jako jsou síla, obratnost, inteligence, životy a další. Každá úroveň přidává postavě 5 zdraví, 3 síly, inteligence a obratnosti a 1 bod do útoku.



Obrázek 23: Navýšení úrovně postavy (vlastní zpracování)

Záložka „Hall of Fame“, síň slávy, nabízí možnost uživateli prohlédnout si všechny existující hráče. Po kliknutí na hráče je uživatel přesměrován na daný profil. Pokud profil není uživatele, tak záložka nastavení profilu je změněna v záložku interakce. V záložce interakce je poskytnuta možnost uživateli, který nevlastní profil této postavy, aby postavu vyzval na souboj, přidal do přátel, napsal zprávu, otevřel s ní obchod nebo postavu pozval do skupiny.

Id	Img	Name	Level
1		Jan	7
2		Trpaslík	2
3		Void	1
4		Djinn	1
5		Gnome	1
6		Fialka	1
7		Johny	1
9		Jane	1
10		Jill	1

Obrázek 24: Záložka síň slávy (vlastní zpracování)

V situaci, kdy se větší počet jedinců ve skupině, dochází ke zvolení jednoho z členů, který se stává vedoucím. Tento jedinec poté stanovuje směr, kterým se skupina vydává, a může ji vést k lovu, boji, plnění úkolů a dalším aktivitám.

4.5. Bojový systém

Systém souboje v se skládá z několika prvků. Nejdříve je důležité stanovit požadavky, které soubojový systém musí mít. Dle analýzy lze soubojový systém vytvořit dvěma způsoby. První způsob zahrnuje pouze hratelnost hráče proti počítači, zatímco druhý způsob pokrývá hratelnost více hráčů. Každý způsob má jisté stanovisko k funkcionalitě a složitosti. Soubojový systém hráče proti počítači není nutně tak složitý, jako soubojový systém více hráčů, ať už ve spolupráci nebo v souboji proti sobě.

4.5.1. Hráč proti počítači

Pro tvorbu bojového systému hráče proti počítači v této práci potřebujeme pouze získat informace o hráči a jeho oponentech, kteří se účastní souboje. Dále zajistit hierarchii v souboji, tedy tahový systém. A poslední částí je zvládnutí soubojových mechanik.

Jinými slovy, je potřeba získat data z databáze které server zpracuje a ukáže konkrétní výstup uživateli. Uživatel je tedy dále schopný interakce a každá jeho akce způsobí volání koncového bodu, který znovu navrátí požadované hodnoty. V koncovém bodu je také aktualizována databáze, za předpokladu, že by se uživatel náhodou odpojil, znovu načetl stránku nebo se snažil obejít client side prostředí.

Informace o hráči je získána pomocí databáze, která je popsána dříve v této části. Avšak s relačním modelem je obtížné a výpočetně náročnější pracovat v interaktivní vrstvě. Tento fakt podporuje myšlenku mapování relační databáze na objektový model. Tímto tedy vznikají factory funkce, které mají výstup instance hráče obsahující všechny potřebné informace z databáze. Stejná funkce existuje ve vrstvě nepřátel, které může hráč potkat. Nyní když existuje způsob vytvoření instance postav, které korespondují s údaji z databáze, jsme schopni také vygenerovat kartičku postavy, která obsahuje dynamické hodnoty.

Na obrázku č. 34 (viz. příloha) je vidět kód, který dynamicky přiděluje hodnoty do šablony kartičky hráče. Většina elementů používá id jako jméno hráče bez diakritiky a mezer, aby se s těmito elementy dalo snadněji pracovat v client side.

Obrázek 25: Kartička hráče (vlastní zpracování)

Na tomto obrázku je vidět výsledek kódu z předešlého obrázku. Je zde šablona kartičky postavy, která je naplněna dynamickými hodnoty z databáze a jsou plně připraveny k funkcionalitě.

Nyní je důležité definovat hierarchii tahů hráčů v soubojovém systému.

```php
$playingorder = array();
for ($i = 0; $i < count($team1); $i++) {
    array_push($playingorder, $team1[$i]);
}
for ($i = 0; $i < count($team2); $i++) {
    array_push($playingorder, $team2[$i]);
}

$whoisplaying = "";

for ($i = 0; $i < count($playingorder); $i++) {
    if ($hracvporadi >= count($playingorder)) {
        $hracvporadi = 0;
    }
    if ($hracvporadi === $i) {
        $whoisplaying = $playingorder[$i]->getUsernameWithoutBlankSpaces();
    }
}
```

Obrázek 26: Tahový systém soubojového systému v PHP (vlastní zpracování)

Obrázek 26. zobrazuje kód, který určuje pořadí hráčů. Je důležité podotknout, že tento kód je zde hlavně z důvodu načtení stránky, aby pořadí hráčů bylo takové, jaké má být. Protože v části kódu herního pole je také schovaný input typu hidden, který ukládá pořadí hráčů. Po načtení stránky se hodnota automaticky vyplní na straně klienta a zároveň se při každé interakci zjišťuje, zda hrající hráč je v pořadí. Zde bohužel vzniká bezpečnostní mezera, která umožní uživateli znalci, aby mohl hodnotu na straně client side přepsat, avšak tahle mezera se znovu kontroluje na end pointu, kde se hráč v pořadí kontroluje s hodnotou v databázi. Z toho vyplývá, že pokud by hráč, který není na řadě obešel tuto mezeru na client side, tak by stále jeho akce neprošla přes serverové ošetření.

```
<div class="game-field">
    <input type="hidden" id="hracvporadi" value="<?= $hracvporadi; ?>">
    <div class="combat-top-section">
```

Obrázek 27: Skrytý vstup pro pořadí hrajícího hráče (vlastní zpracování)

Na obrázku 27. je vidět část kódu se schovaným inputem, který má dynamickou hodnotu z databáze a určuje pořadí hrajícího hráče.

```
const actions = {
    DEFAULT: "default",
    ATTACK: "attack",
    WAIT: "wait",
    DEFEND: "defend",
    SPELL1: "spell1",
    SPELL2: "spell2",
    SPELL3: "spell3",
    SPELL4: "spell4"
};

$(document).ready(function () {
    let playerid = $("#playerid").val();
    let playername = $("#playername").val();
    let monstername = $("#monstername").val();
    let monsterid = $("#monsterid").val();

    let whoisplaying = $("#whoisplaying").val();
    let whoisplayingid = $("input[id='playerid " + whoisplaying + "']").val();
    let hracvporadi = $("#hracvporadi").val();

    highlightPlayer(playername, playerid);
    highlightPlayerActions(playername);

    $(".action").click(function () {
        if (playerid === whoisplayingid) {
            if ($(this).is("#" + whoisplaying)) {
                action = $(this).attr("data-action");
                console.log(action);
                if (action === actions.ATTACK) {
                    HracUtok();
                }

                if (action === actions.WAIT) {
                    cekej();
                }
```

Obrázek 28: Javascript v souboji (vlastní zpracování)

Tento soubor se zabývá pouze problematikou hráče proti počítači a zároveň souboje pouze jeden na jednoho. V souboru je vidět Javascriptový kód, který v první části definuje akce na základě tlačítka, které bylo stisknuto. Dále je zde funkce, která je aktivní při načtení souboru. V této funkci se kontroluje, zda id hráče v session je to stejné jako id hráče, který je na tahu.

Pokud ano, zde předchází podmínka, zda jsou stisknuta tlačítka hráče, který je na tahu a zároveň je vlastníkem té postavy. Pokud jsou tyto podmínky splněny, tak se zavolá akce odpovídající stisknutému tlačítku. Pojďme si popsat funkci útoku hráče.

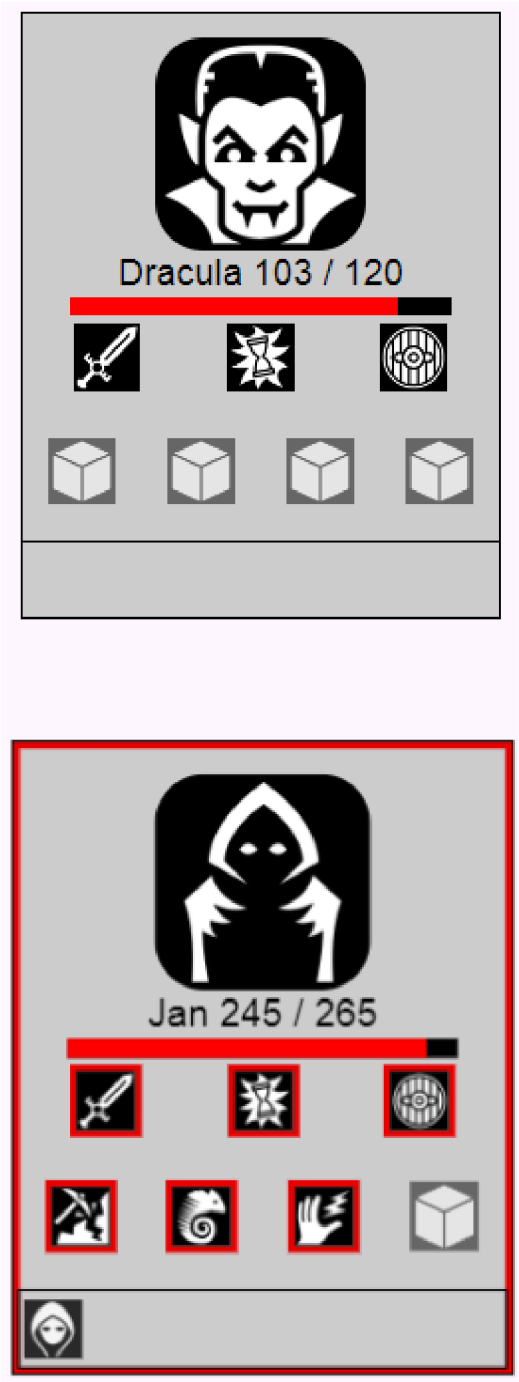
```
function HracUtok() {
    if (($("#playercurrenthp[data-targetname='" + playername + "']").text()) > 0) {
        let json = {"playerid": playerid, "targetid": monsterid, "action": action};
        console.log(json);
        $.ajax({
            type: "POST",
            url: "endpoints/hracutok.php",
            data: {
                json: JSON.stringify(json)
            },
            success: function (response) {
                let odpoved = JSON.parse(response);
                setTargetHP(monstername, odpoved.targetcurrenthp, odpoved.targetmaxhp);
                $(this).off("click");
                $(this).on("click");

            }
        })
    } else{
        console.log("Hráč je mrtvý");
        Konec(monstername);
    }
}
```

Obrázek 29: Javascriptová funkce útoku hráče (vlastní zpracování)

Funkce hráče se skládá pouze z podmínky a ajaxu, který zasílá hodnoty na endpoint, který přijímá výstupy. Podmínka zjišťuje, zda útočící hráč má více než 0 životů. Pokud ne, tak je označen jako mrtvý a vyhrává tedy monstrum. Pokud ano, může útočit. Json objekt se skládá ze 3 vlastností: id hráče, id terče a akce. Tyto 3 vlastnosti jsou elementární pro útok. Je důležité znát kdo na koho útočí a pomocí jaké akce. Práce s těmito proměnnými je zachycena na obrázku 34. viz. příloha.

Na horní části obrázku lze vidět čtení proměnných z javascriptu pomocí POST metody. Dále vzniká proměnná, která rozděluje json proměnnou na pole. Dále se instancuje hráč a nepřítel pomocí factory funkcí ze tříd a jako vstupní proměnnou mají id. Kód pokračuje selektováním souboje a příslušných tabulek k tomu, v tomto endpointu se konkrétně zjišťuje, jaké monstrum je proti hráči v souboji. Do instancí se aktualizují údaje z databáze a provádí se útok tak, že se přepočtou životy podle vzorce útok mínus brnění, kde útok nesmí být menší než

0. Vypočtená hodnota se aktualizuje v databázi jako momentální zdraví konkrétního oponenta a změní se pořadí hrajícího hráče. Poslední řádky kódu jen znovu posílají na javascriptový soubor hodnotu maximálního a aktuálního života zasaženého nepřítele. Je to z důvodu aktualizování života v reálném čase na client side, ale zároveň jsou všechny hodnoty chráněny v pozadí kódu a v databázi. Na předešlém obrázku javascriptu lze vidět, že po úspěšné odpovědi endpointu je také očekáván json, který obsahuje maximální a momentální stav zdraví zasaženého nepřítele a aktualizuje jeho hodnotu zdraví. Existují zde ještě 2 řádky kódu, které aktivují a deaktivují tlačítko, je to z důvodu, že jQuary v tomto případě užití není perfektní a kdyby tlačítko nebylo tímto způsobem vypnuto a zapnuto, tak v souboji dochází k rekurzi, což není žádoucí.



Obrázek 30: Vizuální výsledek po zavolání akce útoku (vlastní zpracování)

Na obrázku 30. je zobrazován vizuální stav uživatele po stisknutí útoku. Útok byl proveden přesně tak, jak byl popsán výše a následně stejným principem byl proveden útok nepřítelem a hráči bylo uděleno poškození. Tímto je z obrázku patrné, že hráč je znovu na tahu a může použít zvýrazněné akce.

4.5.2. Řešení více hráčů

Řešení více hráčů je o něco složitější než první řešení hráče proti počítači. V tomto řešení se počítá s možností více hráčů na obou stranách souboje. Znamená to, že je potřeba vytvořit systém, který zvládne interakce hráčů jak v souboji proti sobě, tak ve spolupráci.

Dle analýzy bylo pro řešení více hráčů zvoleno použití technologie WebSocket. Při započetí souboje bude vytvořen hub (místnost na serveru), kam se následně připojí všichni uživatelé účastnící se souboje. Každý hráč je schopen v reálném čase naslouchat a odpovídat tomu, co se děje v místnosti. V tomto případě je důležité nastavit omezení tak, aby vždy mohl táhnout pouze hráč, který je na tahu. Tyto omezení musí být nastaveny jak na straně serveru, tak na straně klienta.

Technologie websocket umožňuje jednotlivým uživatelům tvářit se jako end-to-end spojení, ale stále je zde server jako prostředník. Každá změna uživatele se odešle na server a server změnu zašle dál. Tento způsob je mnohem robustnější než předešlé řešení pro hráče proti počítači. Umožňuje neomezený počet hráčů a monster v souboji a je rozšiřitelnější. Avšak problematické může být zasílání správných vizuálních indikátorů pro client side vrstvu všech uživatelů.

4.6. Kvalitativní testování

Účelem tohoto testování bylo zjistit, zda aplikace funguje správně. Při testování byla snaha o identifikaci co nejvíce chyb a nedostatků v prototypu. Díky zpětné vazbě participantů je možné zlepšit kvalitu produktu při další iteraci.

Před testováním je participantům a potenciálním participantům zaslán malý dotazník. Dotazník obsahuje jen několik otázek, které mohou být užitečné při testování. Jedna otázka je otevřeného typu a zbytek otázek jsou single nebo multiple choice. Otázky jsou následující v daném pořadí:

- Do jaké věkové kategorie by ses zařadil/a?
- Kolik hodin denně používáš počítač?
- Používáš počítač spíše na hry, školu, práci nebo jiné?
- Hrál/a jsi někdy počítačové hry?
- Jaké hry jsi hrál/a?
- Byly některé hry, které jsi hrál/a prohlížečové?

Dotazník může být užitečný pro výběr kandidátů na testování a zároveň nám může poskytnout důležité informace ke konkrétnímu testování. Například participant, který již hrál prohlížečové hry může mít jiné frustrace než participant, který s takovými hrami nemá žádnou zkušenost.

Testování začíná uvedením participanta do problematiky a následné vysvětlení typu testování a jakým způsobem testování bude probíhat. Po uvedení do problematiky je participant požádán, aby nahlas hovořil a popisoval své myšlenkové pochody při interakci s webovou aplikací. Pomocí metody „Think aloud“ je testování efektivnější, protože vzniká možnost okamžité a skutečné zpětné vazby, která nám pomáhá porozumět potřebám a frustracím uživatele. Cílem tohoto testování je získat informace, které poukážou na nedostatky a zároveň pomohou v dalších iteracích vývoje webové aplikace.

V průběhu testování je jsou participantovi předloženy následující úkoly v tomto pořadí:

1. Zaregistruj se
2. Přihlas se

3. Změň profilovou ikonku své postavy
4. Najdi záložku předmětů/inventáře postavy
5. Vybav svou postavičku mečem
6. Zjisti, kde najdeš svou novou hodnotu útoku
7. Najdi záložku kouzelné knížky postavy
8. Přidej kouzlo své libovolné kouzlo do libovolného aktivního políčka.
9. Jdi na dobrodružství a ulov příšeru. Použij libovolnou kombinaci útoků a kouzel pro poražení příšery.

Po absolvování následujících úkolů byl vymezen extra čas pro participanta, aby měl možnost prozkoumat aplikaci dle své úvahy a v průběhu tohoto času byla částečně prováděna diskuse. Při diskusi byl participant tázán jednoduchých otázek zaměřených na konkrétní dojmy, frustrace a potřeby participanta za účelem získání více konkrétních informací, které by mohly ještě více přispět k následné analýze a eliminování nedostatků. Testování participanta trvalo 20-25 minut.

5. Výsledky a diskuse

V práci byl kladen důraz na zhotovení robustního prototypu webové aplikace RPG hry. Díky znalostem získaných z teoretické části bylo velmi jednoduché tyto znalosti aplikovat a rozvíjet aplikaci, která vycházel z již existujícího základu aplikace a datového modelu z předešlé bakalářské práce autora. Nejdříve bylo přepsáno jádro stránek a na to navazoval přístup mobile first pro tvorbu robustního responsivního css designu. Dále aplikace byla rozvíjena o funkčnosti jako jsou profilový systém, systém přátelství, obchodování a další. Po vytvoření základu těchto funkčností následovala tvorba administrátorského rozhraní, které sloužilo převážně pro kontrolu dat, ale bylo zde také možnost manipulovat a tvořit data. Díky tomuto základu byla tvorba soubojového systému velmi zjednodušena. Při vývoji soubojového systému byly nalezeny dva způsoby, jak dosáhnout cíle práce. Byl zvolen způsob pouze souboje hráče proti počítači z důvodu již zvolených technologií. Další způsob představoval použití technologie websocket. Nicméně soubojový systém byl zhotoven a tímto byly splněny první tři cíle práce. Po vytvoření základního prototypu RPG hry následovalo kvalitativní testování.

Testování se zúčastnilo 8 participantů, z toho 3 ženy a 5 mužů. Byli zde 4 účastníci, kteří se zúčastnili testování a již měli zkušenosti s prohlížečovými hrami. Věkové rozhraní bylo 20-25 let pro 6 účastníků a 26-30 pro zbylé 2 účastníky. 6 participantů používalo počítač více jak 5 hodin denně a 2 participanti používali počítač 4-5 hodin denně. Každý participant používal počítač na práci a pouze 6 z nich na hry.

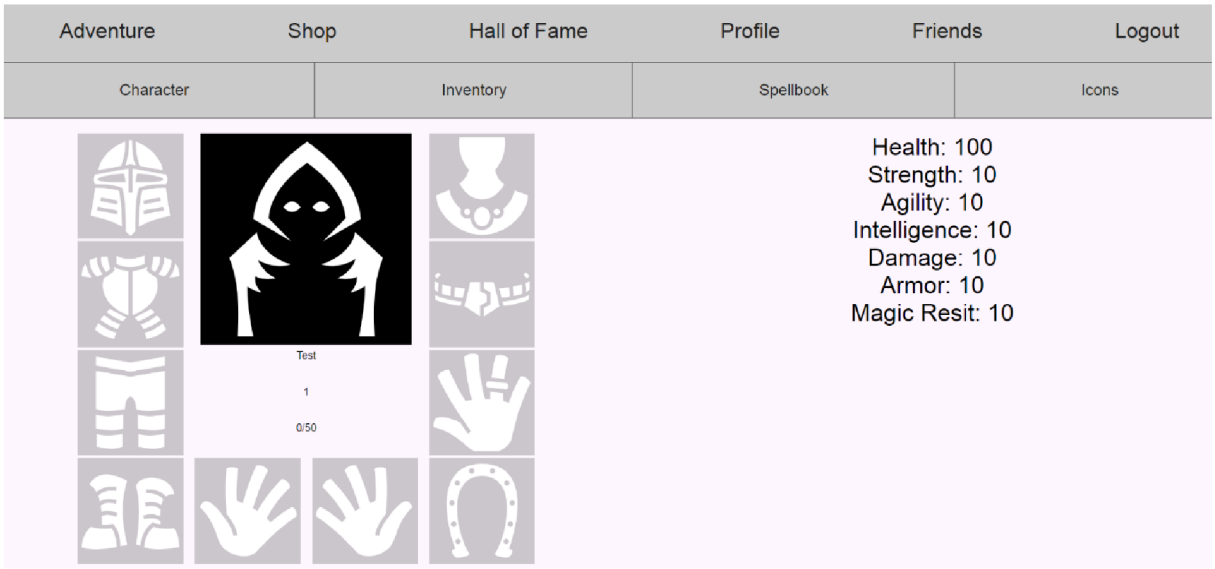
5.1. Výsledky kvalitativního testování

Z výsledků testování vychází, že nejčastější problémy byly v nedostatku vizuální zpětné vazby na různých místech. Tyto nedostatky jsou na záložkách profilu, kouzelné knížky, v obchodě a v soubojovém systému. Šest z osmi uživatelům chybělo více informací, ať už vizuálním nebo textovým způsobem na různých místech. A dva uživatelé, kteří s tímto konceptem neměli problém již měli zkušenosti s hraním prohlížečových her. Na základě testování jednotlivých úkolů bude popsány problémy v dané oblasti.

První úkol. Registrace proběhla úspěšně, až na 2 případy, kde uživatelé očekávali, že po registraci budou automaticky přihlášeni. Jeden z participantů špatně pochopil koncept a myslel si, že nejdříve bude provedena registrace a následně vytvořena postava.

Druhý úkol navazoval na první a jednalo se o přihlášení. Bylo jasné, že uživatelé chtějí mít lepší zpětnou vazbu od registrace. V tomto prototypu formulář neobsahoval políčko pro e-mail a pět z osmi uživatelů očekávalo, že ve formuláři zadají e-mail a do e-mailu dostanou zpětnou vazbu. Kromě těchto nejasností se jiné problémy s přihlášením neobjevily.

Třetí úkol obsahoval změnu profilového obrázku postavy. Tento úkol se ukázal velmi problematický, protože uživatelé si nebyli jisti, zda se nacházejí na hlavní stránce, nebo na profilu postavy. Profilový systém nebyl dostatečně intuitivní pro hráče, kteří nemají předešlé zkušenosti s prohlížečovými hry. Tento úkol vyžadoval více času pro orientaci uživatele a bylo často očekáváno, že tato funkce bude vycházet z prokliku na profilový obrázek hráče.



Obrázek 31: Záložka profilu hráče po přihlášení (vlastní zpracování)

Na obrázku se nachází profil postavy hráče. Z výsledku testování je zřejmé, že uživatelé očekávají více způsobů k dosažení změny profilového obrázku hráče.

Při čtvrtém úkolu, kde participant měl najít záložku předmětů své postavy, se neukázaly žádné problémy. Pouze v jednom případě byl tento úkol lehce problematický, a to z důvodu, že participant neměl zkušenosti s prohlížečovými hrami a zároveň zde byla malá jazyková bariéra.

Pátý úkol představoval vybavení postavy mečem. Všichni participanti uspěli při plnění tohoto úkolu, ale dva z nich očekávali lepší označení vybavení předmětu. Pravděpodobně piktogram, textové označení nebo jiné uspořádání v záložce předmětů by dle názoru participantů lépe sedělo jejich potřebám.

Šestý úkol pověřuje participanta, aby zpozoroval, jakou hodnotu poškození jeho postava má. Tento úkol byl jednoznačný a žádný účastník testování s tímto úkolem neměl problém.

Sedmý úkol říká participantovi, ať přejde na záložku kouzelné knížky postavy. Dále osmý úkol pověřuje participanta, aby přidal libovolné kouzlo do libovolného akčního políčka. Úkol najít záložku knížky byl v pořádku. Avšak osmý úkol se prokázal, stejně jako třetí úkol, jako velmi problémový. Záložka kouzelné knížky nebyla zřejmá. Uživatele očekávali více popisků, lepší rozdělení a jednoznačně neporozuměli funkci akčních políček kouzel. Dále čtyři účastníci očekávali drag and drop funkcionalitu.

Poslední úkol zahrnoval najít záložku dobrodružství a jít ulovit příšeru. V této sekci se ukázalo, že popisky v souboji jsou nejednoznačné. Participantům chybělo více informací a efektů v souboji. Systém souboje se zdál pochopitelný, ale v půlce případů byl očekáván textový záznam a ve zbytku případů byl vyžadován alespoň vizuální efekt po akci. Dále si participanti přáli, aby byl lépe označen výsledek souboje a aby bylo zřejmé, co ze souboje získali. Ve dvou případech bylo zmíněno, aby po souboji byl hráč odkázán na záložku dobrodružství místo svého profilu.

V části po splnění úkolů byla vyhrazena volná chvilka pro participanty, ať si aplikaci proklikají dle potřeby. Díky této části byl nalezen další nedostatek. Nedostatek se týká profilového systému, kde participanti měli problém přidat si další hráče do kolonky přátel. V záložce přátel bylo očekáváno okénko s vyhledáváním dle jména hráče nebo alespoň přesměrování na záložku s tabulkou hráčů, kde by taková funkce existovala. Dále se ukázalo, že přidávání hráčů přes profilový systém je velice neintuitivní a participanti by očekávali více způsobů, jak by mohli hráče přidat do své kolonky přátel.

Na základě výsledků testování by při další iteraci vývoje hry bude kladen důraz hlavně na změnu profilového systému. Přidání více způsobů, jak změnit profilovou ikonku. Přidání funkcionalit drag and drop v kouzelné knížce a změna záložky předmětů hráče. Dále bude kladen důraz na zlepšení vlastností profilového systému a systému přátel. Bude kladen důraz na celkové zlepšení zpětné vazby pro uživatele. A soubojový systém bude vylepšen o mnoho vlastností jako je textový záznam, vizuální efekty, lepší popisky po najetí kurzorem na element a lepší znázornění výsledků a odměn ze souboje.

6. Závěr

Výsledkem této diplomové práce je prototyp webové RPG hry. Při vývoji prototypu hry bylo zhotoveno uživatelské rozhraní za použití mobile first přístupu k vytvoření robustního responsivního designu. Dále bylo zhotoveno administrátorské rozhraní pro kontrolu dat a pro možnost přidání herního obsahu. Následně byl kladen důraz na rozvinutí soubojového systému, který umožňuje uživateli bojovat v reálném čase. Na závěr byl prototyp aplikace kvalitativně testován a výsledky byly vyhodnoceny.

V teoretické části jsou analyzovány prvky technologií souvisejících s vývojem webových stránek. Jsou zde popsány technologie související s frontendem jako jsou HTML a CSS. Dále na je rozvedeno uživatelské rozhraní, teorie prototypování a uživatelská zkušenost. Další oddíl v teoretické části je věnován backendu. V této části jsou popsány programovací jazyky PHP a JavaScript a dále jsou rozvedeny technologie API, WebSocket, databáze a jazyk SQL. Pojednává se zde i o architektonickém vzoru MVC. Závěr teoretické části byl věnován krátké teorii RPG her.

V části vlastní práce jsou využity poznatky a znalosti získané z teoretické části k tvorbě robustního prototypu webové RPG hry. Na začátku jsou definovány požadavky a je analyzován postup k tvorbě aplikace. Další část práce pokračuje implementací, kde je popsána tvorba databáze, zabezpečení webu a popředí stránek. Dále jsou uvedeny jednotlivé části webové aplikace a jejich funkčnosti. Následuje problematika bojového systému, kde se pojednává o dvou způsobech řešení cíle práce. Na závěr je provedeno kvalitativní testování prototypu webové aplikace.

Během vývoje a testování bylo zjištěno, že neustálá iterace a vylepšování je klíčové pro dosažení kvalitního výsledku. Další vývoj by mohl být zaměřen na vylepšení uživatelského rozhraní a rozšíření funkčnosti aplikace. Díky uživatelskému testování bylo zjištěno, že pro uživatele je klíčové orientovat se v menu, a v profilové části stránky.

Výsledky této práce mohou pomoci při tvorbě podobných webových aplikací. Tato práce může sloužit jako zdroj informací a návod pro vývojáře, kteří se chtějí zabývat tvorbou webových RPG her.

7. Seznam použitých zdrojů

AJUNWA, Ifeoma. The “black box” at work [online]. 2020, 7(2). ISSN 2053-9517. Dostupné z: doi:10.1177/2053951720938093

Relational Database VS Object-Oriented Database (Key Differences) [online]. 2022. Dostupné z: https://databasetown.com/relational-database-vs-object-oriented-database-key-differences

PANDEY, Ananda Raj. MySQL naming / coding conventions: tips on mySQL database [online]. 2015 Dostupné z: https://anandarajpandey.com/2015/05/10/mysql-naming-coding-conventions-tips-on-mysql-database

NTUI, A. I. (2021). Key Steps To Building A Great Small Business Website. Zenodo. https://doi.org/10.5281/ZENODO.5493581

APUKE, Oberiri Destiny. Quantitative Research Methods: A Synopsis Approach. Kuwait Chapter of Arabian Journal of Business and Management Review [online]. 2017, 6(11), 40-47. ISSN 22248358. Dostupné z: doi:10.12816/0040336

BADIA, Antonio. Entity-Relationship modeling revisited. ACM SIGMOD Record [online]. 2004, 33(1), 77-82. ISSN 0163-5808. Dostupné z: doi:10.1145/974121.974135

BECK, Kent. Extrémní programování [online]. 2002 Dostupné z: doi:80-247-0300-9

BOJINOV, Valentin. RESTful web API design with node.js 10: Learn to create robust restful web services with node.js, mongodb, and express.js. Packt Publishing, 2018. ISBN 978-1788623322.

BURCHARD, Evan, 2013. The Web Game Developer's Cookbook: Using JavaScript and HTML5 to Develop Games (Game Design). Addison-Wesley Professional. ISBN 978-0321898388.

CONNOLLY, Thomas M., Carolyn E. BEGG a Jennifer WIDOM. Database systems: a practical approach to design, implementation, and management. 5th ed. London: Addison-Wesley, c2010. ISBN 978-0-32-152306-8.

DONKER, Afke a Panos MARKOPOULOS. A Comparison of Think-aloud, Questionnaires and Interviews for Testing Usability with Children. People and Computers XVI - Memorable Yet Invisible [online]. London: Springer London, 2002, 2002, 305-316. ISBN 978-1-85233-659-2. Dostupné z: doi:10.1007/978-1-4471-0105-5_18

EASTTOM, Chuck. Computer security fundamentals. Third edition. Indianapolis, Indiana: Pearson, [2016]. ISBN 07-897-5746-X.

FELLMANN, Tom a Manolya KAVAKLI. A command line interface versus a graphical user interface in coding VR systems [online]. 2007 Dostupné z:

https://www.researchgate.net/publication/234818436_A_command_line_interface_versus_a_graphical_user_interface_in_coding_VR_systems

FETTE a MELNIKOV. The WebSocket Protocol [online]. Dostupné z: https://www.rfc-editor.org/rfc/rfc6455

FLICK, Uwe. An introduction to qualitative research. Sage Publications, 1999. ISBN 9780761955870.

GALLAS, Milan. MS-SQL krok za krokem: Triggery (DML) [online]. Dostupné z: https://www.itnetwork.cz/ms-sql/ms-sql-trigger-dml

GARRETT, Jesse James. *The elements of user experience: user-centered design for the Web and beyond*. 2nd ed. Berkeley, CA: New Riders, c2011. Voices that matter. ISBN 03-216-8368-4.

MVC Framework Introduction [online]. Dostupné z: https://www.geeksforgeeks.org/mvc-framework-introduction/

GHANN, Patricia a Jun STEED HUANG. RESEARCH ON HTML5 DEVELOPMENT [online]. 2011. Dostupné z: https://www.researchgate.net/publication/271329059_RESEARCH_ON_HTML5_DEVELOPMENT

GRANIC, Isabela, Adam LOBEL a Rutger C. M. E. ENGELS. The benefits of playing video games. American Psychologist [online]. 2014, 69(1), 66-78. ISSN 1935-990X. Dostupné z: doi:10.1037/a0034857

GRONIER, Guillaume. Measuring the First Impression: Testing the Validity of the 5 Second Test [online]. 2016. Dostupné z: https://www.researchgate.net/publication/316275909_Measuring_the_First_Impression_Testing_the_Validity_of_the_5_Second_Test

HEMA, Valpadasu, Sravanthi THOTA, S NARESH KUMAR, Ch PADMAJA, C. Bala RAMA KRISHNA a K MAHENDER. Scrum: An Effective Software Development Agile Tool. IOP Conference Series: Materials Science and Engineering [online]. 2020, 981(2). ISSN 1757-8981. Dostupné z: doi:10.1088/1757-899X/981/2/022060

HUSTINX, Lesley a Thomas DENK. The 'Black Box' Problem in the Study of Participation. Journal of Civil Society [online]. 2009, 5(3), 209-226. ISSN 1744-8689. Dostupné z: doi:10.1080/17448680903351693

MVC Architecture in Java [online]. Dostupné z: https://www.javatpoint.com/mvc-architecture-in-java

What Is Node.js and Why You Should Use It [online]. Dostupné z: https://kinsta.com/knowledgebase/what-is-node-js/

KOVANOVIC, Vitomir, Dragan GAŠEVIĆ, Shane DAWSON, Srećko JOKSIMOVIC a Ryan BAKER. Does Time-on-task Estimation Matter? Implications on Validity of Learning

Analytics Findings. Journal of Learning Analytics [online]. 2016, 2(3), 81-110. ISSN 1929-7750. Dostupné z: doi:10.18608/jla.2015.23.6

KOZLOVSKÝ, Pavel a Zdeňka TELNAROVÁ. Relační databáze - jazyk SQL: učební text pro distanční formu vzdělávání. Orlová: Obchodní akademie Orlová, 2006 [i.e. 2007]. Informatika v ekonomice v distanční formě vzdělávání na středních školách. ISBN 978-80-87113-25-7.

NORMAN, Donald A. Design pro každý den [online]. 2010 [cit. 2023-03-20]. Dostupné z: doi:978-80-7363-314-1

KRUG, Steve. Rocket surgery made easy: the do-it-yourself guide to finding and fixing usability problems. Berkeley, CA: New riders, c[2010]. ISBN 978-0321657299.

RICHARDSON, Leonard a Sam RUBY. RESTful web services [online]. 2007. Dostupné z: doi:9780596529260

KRUG, Steve. Nenuťte uživatele přemýšlet!: praktický průvodce testováním a opravou chyb použitelnost [sic] webu. Brno: Computer Press, 2010. ISBN 978-80-251-2923-4.

LAURENČÍK, Marek. Tvorba www stránek v HTML a CSS. Praha: Grada Publishing, 2019. Průvodce (Grada). ISBN 978-80-271-2241-7.

WILLIAMS, H. E., & Tahaghoghi, S. (2006). Learning MySQL. O'Reilly Media.

MACH, Jakub. PHP pro úplné začátečníky. 2. přeprac. a rozš. vyd. Brno: Computer Press, 2003. ISBN 80-722-6834-1.

MASLAKOWSKI, Mark. Naučte se MySQL za 21 dní. Praha: Computer Press, 2001. Všechny cesty k informacím. ISBN 80-722-6448-6.

MDN. JavaScript [online]. 2023. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/JavaScript

MERUNKA, Vojtěch a Václav VOSTROVSKÝ. Databázové systémy. Praha: Česká zemědělská univerzita, 1998. ISBN 80-213-0388-3.

BARKER, Michelle. Optimizing A Vue App. Smashing Magazine [online]. 2022 [cit. 2023-03-20]. Dostupné z: https://www.smashingmagazine.com/2022/11/optimizing-vue-app/

MISHRA, Debani Prasad, Kshirod Kumar ROUT a Surender Reddy SALKUTI. Modern tools and current trends in web-development. Indonesian Journal of Electrical Engineering and Computer Science [online]. 2021, 24(2), 978-985. ISSN 2502-4760. Dostupné z: doi:10.11591/ijeecs.v24.i2.pp978-985

MOVED. A ISO 8601 Date, Time, and Duration Support [online]. Dostupné z: https://docs.oracle.com/en/database/oracle/oracle-database/21/adjsn/iso-8601-date-time-and-duration-support.html

MYERS, Brad A. Graphical User Interface Programming [online]. 2003. Dostupné z: https://www.researchgate.net/publication/2866174_Graphical_User_Interface_Programming

ROBBINS, J. N. (2012). Learning Web Design: A beginner's guide to HTML, CSS, JavaScript, and web graphics (4th ed.). O'Reilly Media.

NOBACK, M. (2020). Advanced web application architecture.

About | Node.js. [online]. Dostupné z: https://nodejs.org/en/about

NORMAN, Donald A. a Jakob NIELSEN. Usability testing 101

PHP: PHP 8.0.0 Release Announcement. PHP: Hypertext Preprocessor [online]. Copyright © 2001. Dostupné z: https://www.php.net/releases/8.0/en.php

PUNDE, Pramodini A., Mukti E. JADHAV a Ramesh R. MANZA. A study of eye tracking technology and its applications. 2017 1st International Conference on Intelligent Systems and Information Management (ICISIM) [online]. IEEE, 2017, 2017, 86-90. ISBN 978-1-5090-4264-7. Dostupné z: doi:10.1109/ICISIM.2017.8122153

RANGA, Babu. DELETE CASCADE and UPDATE CASCADE in SQL Server foreign key [online]. 2019. Dostupné z: https://www.sqlshack.com/delete-cascade-and-update-cascade-in-sql-server-foreign-key/

SCOTT, B., & Neil, T. (2009). Designing web interfaces: Principles and patterns for rich interactions. O'Reilly Media.

SHARMA, Aakanksha. Introduction to HTML (Hyper Text Markup Language) - A Review Paper [online]. 2018. Dostupné z: https://www.researchgate.net/publication/327861224_Introduction_to_HTML_Hyper_Text_Markup_Language_-_A_Review_Paper. Federation University Australia.

STJEPANDIC, J., Wognum, N., & Verhagen, W. J. C. (Eds.). (2015). Concurrent engineering in the 21st century: Foundations, developments and challenges (2015th ed.). Springer International Publishing.

VIDYAPEETH MAHARASHTRA, Tilak. A Study of advantages of playing video games for people [online]. 2021. Dostupné z: https://www.researchgate.net/publication/349686534_A_Study_of_advantages_of_playing_video_games_for_people

TIPTON, H. F., & Krause, M. (Eds.). (2009). Information Security Management Handbook, Volume 3. Auerbach Publications.

ANTONY, Unwin a Heike HOFMANN. GUI and Command-line - Conflict or Synergy? [online]. 2000. Dostupné z: https://www.researchgate.net/publication/2425912_GUI_and_Command-line_-_Conflict_or_Synergy

VOSTROVSKÝ, Václav. Vytváření databází v ORACLE. Praha: Česká zemědělská univerzita v Praze, Provozně ekonomická fakulta, 2004. ISBN 978-80-213-1191-6.

VYSTAVĚL, Radek. Databáze a SQL pro začátečníky. Ondřejov: Radek Vystavěl, 2021. ISBN 978-80-908144-0-0.

W3SCHOOLS. CSS Tutorial [online]. Dostupné z: https://www.w3schools.com/css/

W3SCHOOLS. Sass Tutorial. [online]. Dostupné z: https://www.w3schools.com/sass/

WALKER, Miriam, Leila TAKAYAMA a James A. LANDAY. High-Fidelity or Low-Fidelity, Paper or Computer? Choosing Attributes when Testing Web Prototypes. Proceedings of the Human Factors and Ergonomics Society Annual Meeting [online]. 2002, 46(5), 661-665. ISSN 2169-5067. Dostupné z: doi:10.1177/154193120204600513

WEINSTEIN, Aviv Malkiel. Computer and Video Game Addiction—A Comparison between Game Users and Non-Game Users. The American Journal of Drug and Alcohol Abuse [online]. 2010, 36(5), 268-276. ISSN 0095-2990. Dostupné z: doi:10.3109/00952990.2010.491879

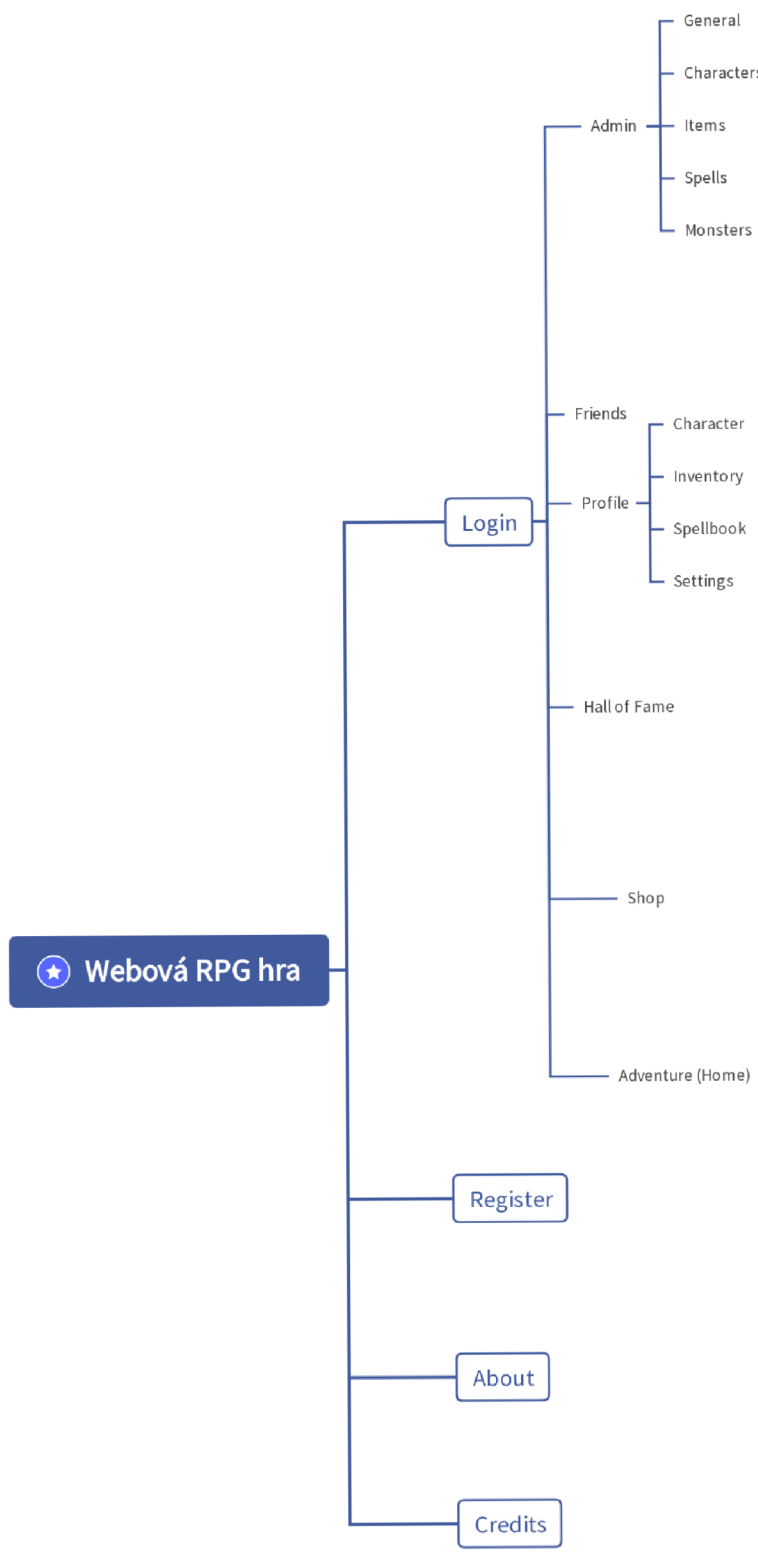
ZITKUS, Emilene, Aline C. BRIGATTO, Ana Lya M. FERRARI, Gabriel H. C. BONFIM, Idinei F. P. CARVALHO FILHO, Thaís D. REIS, Fausto O. MEDOLA a Luis C. PASCHOARELLI. Accessibility and Usability of Websites Intended for People with Disabilities: A Preliminary Study. Design, User Experience, and Usability: Novel User Experiences [online]. Cham: Springer International Publishing, 2016, 2016-06-22, 678-688. Lecture Notes in Computer Science. ISBN 978-3-319-40354-0. Dostupné z: doi:10.1007/978-3-319-40355-7_66

8. Seznam obrázků, tabulek, grafů a zkratek

Seznam obrázků

Seznam použitých zkratek

RPG, PHP, HTML, FE, BE, UI, UX, GUI, CLI, MVC, CSS, SCSS, Sass, LFP, HFP, SPA, SQL, ERD, XP, DBMS, SŘBD



Obrázek 32: Architektura aplikace (vlastní zpracování)

```php
$playerjson = json_decode($_POST["json"]);
$playerjsonencode = array("playerid"=>$playerjson->{'playerid'}, "targetid"=>$playerjson->{'targetid'}, "action"=>$playerjson->{'action'});

$target = MonsterFactory::createMonster($playerjson->{'targetid'});
$player = PlayerFactory::createPlayer($playerjson->{'playerid'});

$playerid = $player->getId();
$targetid = $target->getId();

$getcombatid = Select("SELECT * FROM combat WHERE id_player = ?",$playerid,$conn)[0];
$combatid = $getcombatid["id_combat"];

$gettargethp = Select("SELECT * FROM combat_has_monsters WHERE Monsters_id_monster = ?",$targetid,$conn)[0];
$hpzdb = $gettargethp["currentHealth"];
$target->setCurrentHealth($hpzdb);
$target->takeDamage($player->getTotalDamage() - $target->getTotalArmor());

$newtargethp = $target->getCurrentHealth();
if($target->getArmor() > $player->getTotalDamage()){
} else{
    $newtargethp = $target->getCurrentHealth();
}
$settargethp = mysqli_query($conn,"UPDATE `combat_has_monsters` SET `currentHealth` = $newtargethp
                            WHERE `combat_has_monsters`.`Combat_id_combat` = $combatid AND `combat_has_monsters`.`Monsters_id_monster` = $targetid");

//hraje další hráč
$dalsihracnatahu = mysqli_query($conn,"UPDATE `combat` SET `hracvporadi` = 0 WHERE `combat`.`id_combat` = $combatid");

$playerjsonsend = array("targetcurrenthp"=>$target->getCurrentHealth(),"targetmaxhp"=>$target->getTotalHealth());
echo json_encode($playerjsonsend);
```

Obrázek 33: Koncový bod útoku hráče (vlastní zpracování)

Obrázek 34: Dynamické hodnoty v kartičce hráče (vlastní zpracování)