GYMNASIUM JANA KEPLERA

Parléřova 2/118, 169 00 Praha 6

Webové stránky pro Chytrou palici

Maturitní práce

Autor: Michal Vaňata

Třída: 4.A

Školní rok: 2020/2021

Předmět: Informatika

Vedoucí práce: Šimon Schierreich

Praha, 2021

GYMNASIUM JANA KEPLERA
Kabinet informatiky

ZADÁNÍ MATURITNÍ PRÁCE

Student:	**Michal Vaňata**
Třída:	**4.A**
Školní rok:	**2020/2021**
Platnost zadání:	**21. 10. 2021**
Vedoucí práce:	**Šimon Schierreich**
Název práce:	**Webové stránky pro Chytrou palici**

Pokyny pro vypracování:

Vytvořte webový portál pro školní literární soutěž Chytrá palice. Portál budou autorizovanému uživateli umožňovat dynamické přidávání a editaci obsahu a správu soutěže. Koncový uživatel pak bude mocu obash filtrovat, vyhledávat a komentovat.

Doporučená literatura:

[1] MARTIN, Robert C. Design Principles and Design Patterns. www.objectmentor.com, 2000. Dostupné z: https://fi.ort.edu.uy/innovaportal/file/2032/1/design_principles.pdf.

[2] FOWLER, Martin. Patterns of enterprise application architecture. Boston: Addison-Wesley Professional, 2003. ISBN 978-0321127426.

URL repozitáře:

https://github.com/Daybringer/chytra-palice

vedoucí práce *student*

V Praze dne 30. 10. 2020

Prohlášení

Prohlašuji, že jsem svou práci vypracoval samostatně a použil jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů. Nemám žádné námitky proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

V Praze dne 7. dubna 2021 Michal Vaňata

Poděkování

Schtěl bych poděkovat vedoucímu maturitní práce, Šimonu Schierreichovi za poskytnuté materiály, rady a konzultace.

Abstrakt

Abstrakt.

Klíčová slova

webová aplikace, CMS, správa soutěže

Abstract

Abstract.

Keywords

web application, CMS, contest management

Obsah

1. Teoretická část

Cílem projektu je vytvoření nové webové platformy pro školní literární soutěž Gymnázia Jana Keplera, Chytrou palici. Původní webová stránka sloužila pouze k nahrávání, filtraci a zobrazování prací zařazených do soutěže. Její vzhled byl zastaralý a některé její části byly nefunkční, a proto byl vypsán projekt na tvorbu nové. Vzhledem k tomu, že se stránka navrhovala od úplných základů, se naskytla možnost obohatit stávající a přidat další funkce.

Výsledná webová stránka by nakonec krom původních funkcí měla umožňovat přidávání a editace článků blogového typu, vytvoření soutěže, vyhlášení závěrečného pořadí, nahrávání obrázků a konverzi a následné stažení nahraných soutěžních prací. Jedním z dalších požadavků bylo umožnění komentování prací studenty.

2. Implementace

Vznik každé větší webové aplikace sestává z dvou částí: návrhu a programování. Návrhová část se v mém případě skládal ze sběru požadavků zadavetele, tvorby společného jazyka (*ubiquitous language*) a tvorby diagramů a datových modelů. Při tvorbě datových modelů jsem již uvažoval nad přibližnou podobou a rozložením API přístupových bodů. Vzhledem k tomu, že je to poprvé, co tvořím webovou stránku takového rozsahu, tak jsem se musel dovzdělat v oblasti návrhu REST API. Dále jsem si musel rozmyslet strukturu stránek a vytvořit přibližnou mapu stránek.

Po návrhové části přichází implementace webové stránky, která se skládá z tvorby frontendu a backendu.
Při dobře provedené návrhové části bývá mnohdy samotné programování procházkou růžovým sadem, v případě podcenění přípravy procházkou devíti kruhy Dantova Pekla.

2.1 Před vývojem

2.1.1 Sběr požadavků a tvorba slovníku

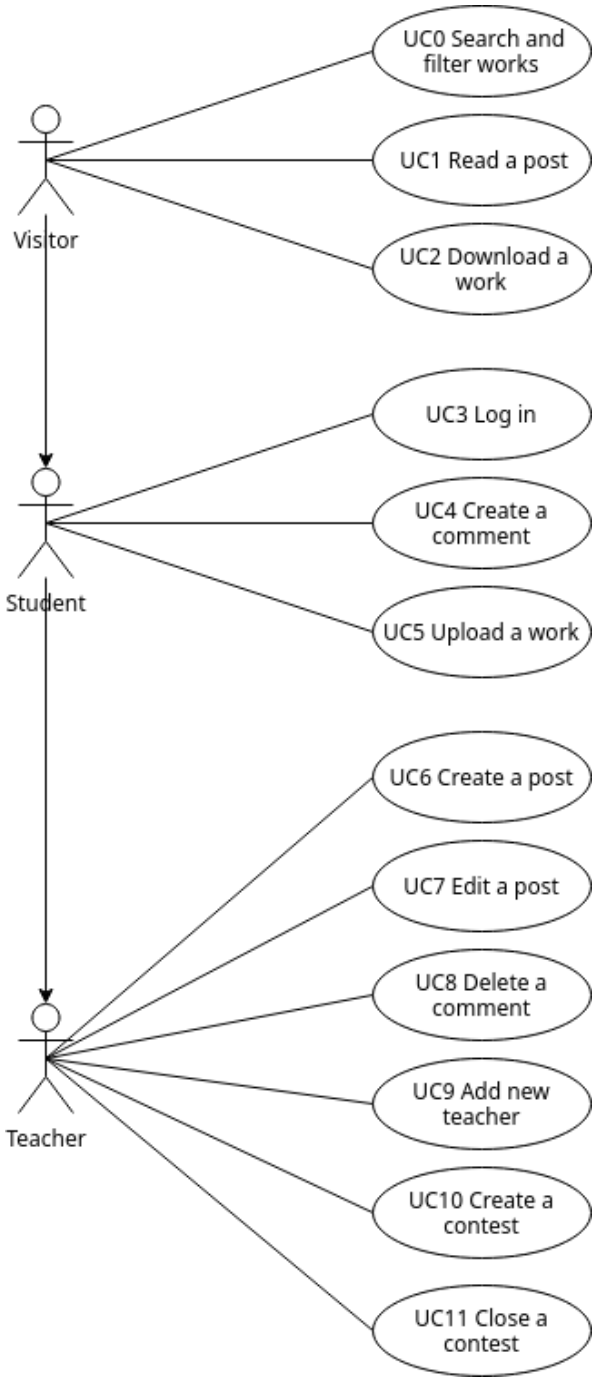
První krokem při tvorbě Chytré palice byly konzultace se zadavetelem projektu, při kterých jsem si musel ujasnit, co od stránky očekává, jaké má připomínky k podobě a funkcionalitě starých stránek a jak by si představoval přibližnou podobu. Po konzultacích by měla již být vytvořená tabulka požadavků (v 2.1), s kterou bude nadále pracovat (např. UCD).
Dále by jsme měli mít vytvořený tzv. *ubiquitous language*, což si můžete představit jako vytvoření slovníčku termínu, které budeme společně se zákazníkem používat. Jde o to, aby mezi zákazníkem a vývojářem nedocházelo k miskomunikacím. Inspirací při tvorbě designu a rozložení stránky mi měli být již zažité webové platformy A2larm a A2.

2.1.2 Tvorba diagramů

Dalším krokem je vytvoření diagramů.
Ty tvoříme především pro ujasnění struktury programu a dat v něm.
Pro tvorbu diagramů jsem použil webový nástroj DrawIO dostupný na https://app.diagrams.net

Use case diagram

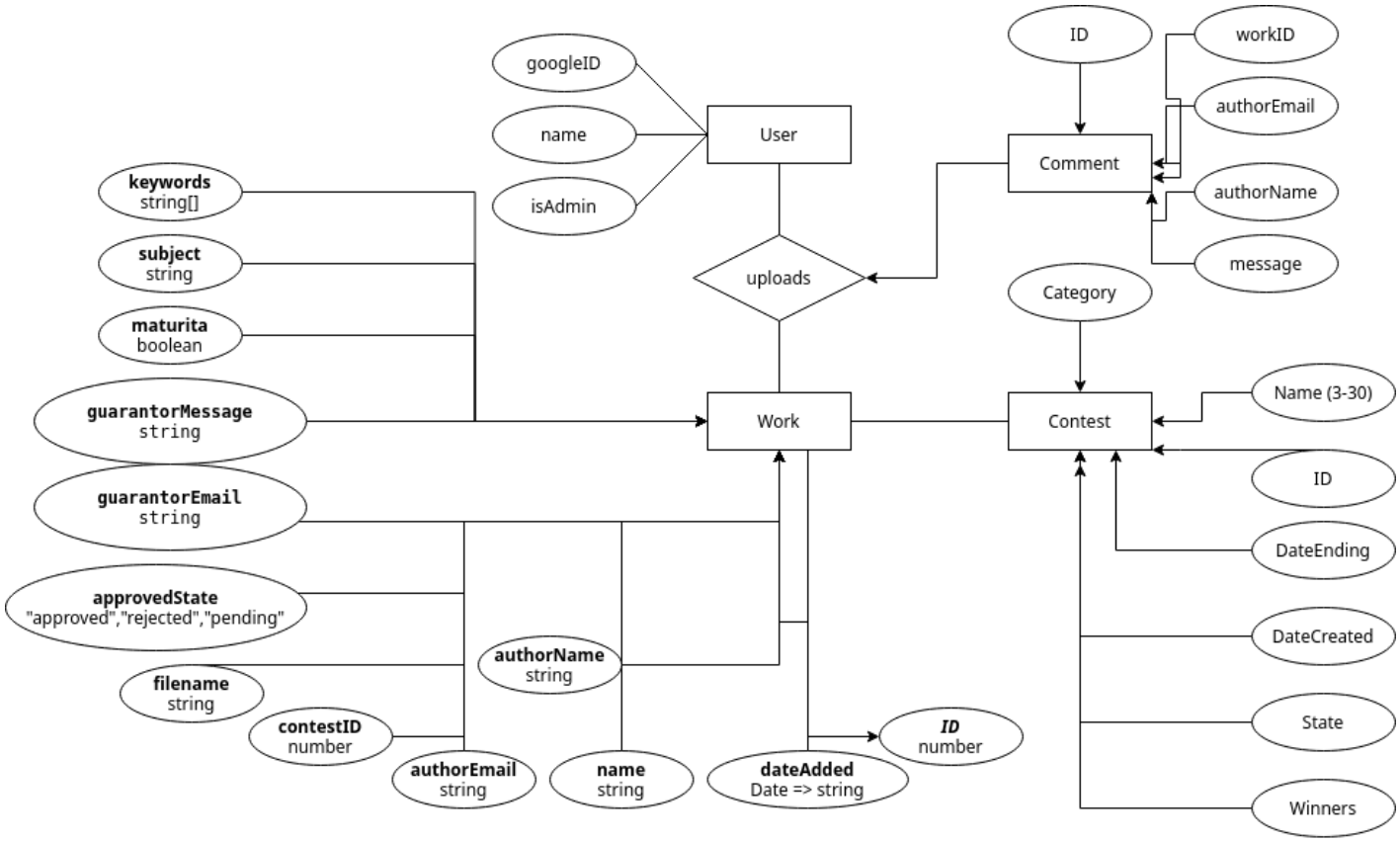
Use case diagram neboli diagram užití je typem UML diagramu, který se používá jako náhled na programovaný systém. Zobrazují se v něm vztahy mezi tzv. aktory čili uživateli systému a službami. Může mít jak textovou podobu, která bývá podrobnější a každý případ užití je podrobně popsán, tak i podobu diagramovou.
Je u něj důležitá jednoduchost, neboť se primárně používá pro komunikace mezi vývojářem a zákazníkem. To znamená, že by neměl obsahovat technikálie a měl by používat *ubiquitous language*.
Při jeho vytváření využíváme tabulku požadavků, abychom si byli jistí, že jsme splnili všechny funkční požadavky.



Obrázek 2.1: UCD

Entity-relationship diagram

Entity-relationship diagram je vizuální reprezentací objektů (entit) a vztahů mezi entitami. ER diagramy se nejčastěji používají při analýze informačních systémů a u návrhů systémů při jejich budování. Při správném vytvoření je dokonce možné použít jeho variantu pro vygenerování databáze.



Obrázek 2.2: ERD

2.2 Vývoj

Jakmile jsme hotovi s přípravou, tak se můžeme pustit do vývoje.
V mém případě jsem si jako první připravil frontendovou část webové aplikace, k němuž jsem později připojil server. Nakonec bych zvolil asi cestu opačnou, ale o tom více v závěru. Ideálně by celý proces vývoje mělo doprovázet psaní testů, aby jsme si byli jisti, že jsme u psaní splnily určitě nároky.

Při volbě technologií jsem zohledňoval především svojí předchozí zkušenost s ní. Až poté jsem uvažoval nad vhodností knihovny či nástroje.

2.2.1 Frontend

Vzhledem k rozsáhlosti projektu byla nasnadě volba některého javascriptího frameworku, abych si ulehčil práci s opakujícími se prvky a s reaktivitou Zde se v podstatě nabízejí 4 možnosti, a to React, VueJS, Svelte a Angular. Z těchto frameworků jsem měl dosavadní zkušenost jen s Vue a jelikož jsem chtěl ihned začít tvořit prototyp a mezi ostatními frameworky není takový rozdíl, tak jsem zvolil jej. U Vue existují v tuto dobu dvě paralelní verze: 2 a 3. Verze 3 nabízí řadu zlepšení v reaktivitě dat a hlavně bezproblémové používání Typescriptu. Velkou nevýhodou je ale nezralost ekosystému knihoven, což nakonec rozhodlo o mém použití VueJS verze 2.
Při vytvoření prototypu bylo nutné simulovat volání na server (REST API), pro tento účel jsem využil mockAPI.

Abych nemusel věnovat přespřílišnou pozornost stylům a tvorbě UI, tak jsem využil předpřipravené komponenty z Buefy.

K voláním na server jsem použil nejpoužívanější knihovnu,axios, který jsem za pomocí repository design patternu a Vuex (Vue store) implementoval tak, aby byla možná jednoduchá úprava konfigurací a přístupových bodů.

2.2.2 Server

Pro jazyk použitý na serveru jsem měl dvě podmínky, a to aby byl typovaný a umožňoval snazší objektové programování. Vzhledem k předchozí znalosti Javascriptu jsem zvolil Typescript. Dále jsem musel najít framework a databázové ORM, které by jej nativně podporovali. NestJS a TypeORM s Postgresql byly mou volbou.

Pro konverzi nahraných soutěžní prací z otevřených formátů (.doc, .docx, .dot) do pdf jsem využil funkci konverze libreoffice.

Název požadavku	Popis požadavku	Fn
Filtrování	práce a soutěže by měli být filtrovatelné na základě vlastností	ano
Uspořádávání	práce a soutěže by měli být uspořadatelné na základě vlastností	ano
Zobrazení práce	návštěvník si může práci zobrazit skrze prohlížeč	ano
Stažení práce	návštěvník si může práci stáhnout ve formát PDF či EPUB	ano
přihlášení	studenti a učitelé by měli být schopní přihlásit se přes školní mail	ano
Multimédia	umožnit přidání multimédií (obrázků) ke článku	ano
Práce	přidání soutěžní práce	ano
Soutěž	vytvoření a správa soutěže	ano
Adminy	zpřístupnit některé funkci jen adminům	ano
Migrace prací	přesunout staré práci na nový web	ne
Docker	před nahráním práce na školní web vytvořit docker image	ne
Reference na práci	umožnit vygenerování reference na práci při psaní článku	ano
Nahrání souboru	umožnit studentům nahrát práci ve formátech PDF, DOC(X), ODT	ano
Podcast	možnost dynamicky přidat odkaz na podcast	ano
Nahrání souboru	umožnit studentům nahrát práci ve formátech PDF, DOC(X), ODT	ano
Komentování	komentování soutěžní prací studenty a učiteli	ano
Nejčtenější	zobrazení nejčtenější práce	ano

Tabulka 2.1: Tabulka požadavků

3. Technická dokumentace

3.1 Instalace a spuštění

LibreOffice

Pokud nemáte nainstalovaný LibreOffice, tak si jej nainstalujte
https://www.libreoffice.org/get-help/install-howto

NodeJS

https://nodejs.org/en/download/

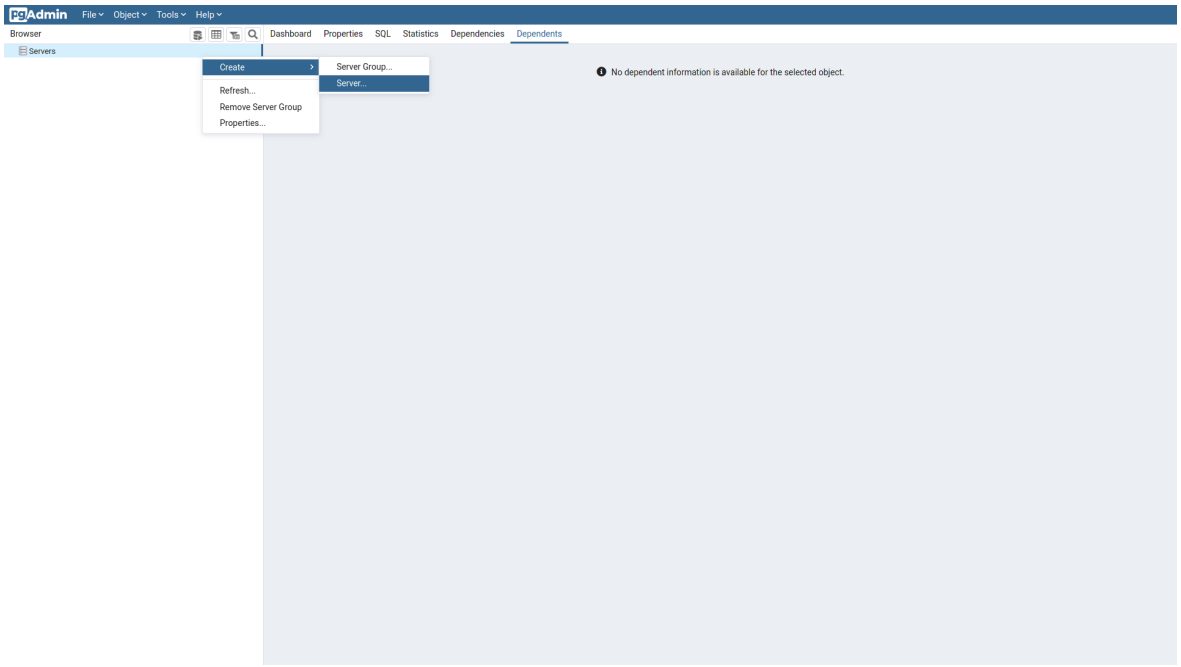
Yarn

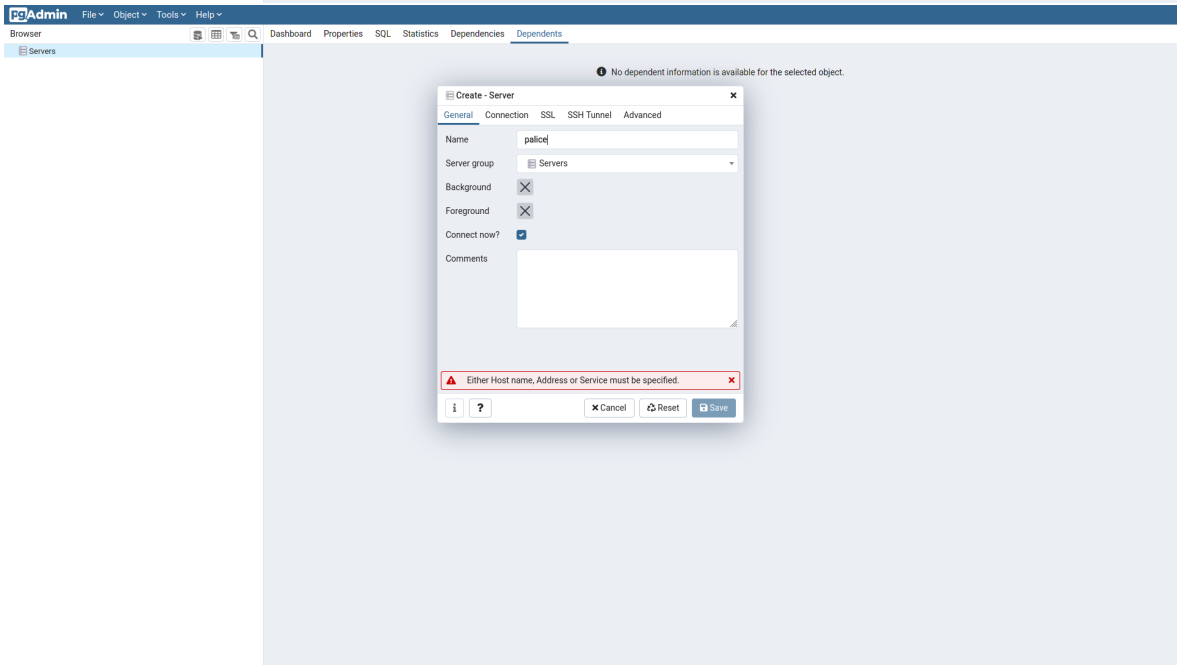
```
npm install --global yarn
```

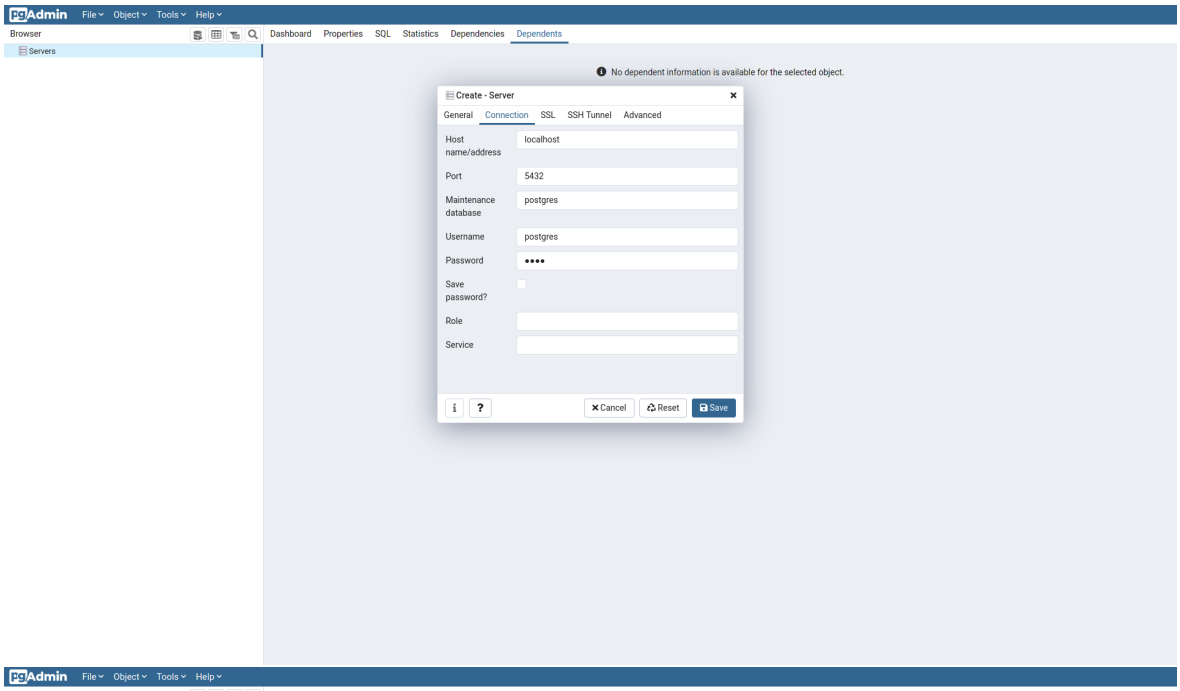
PostgreSQL a pgAdmin

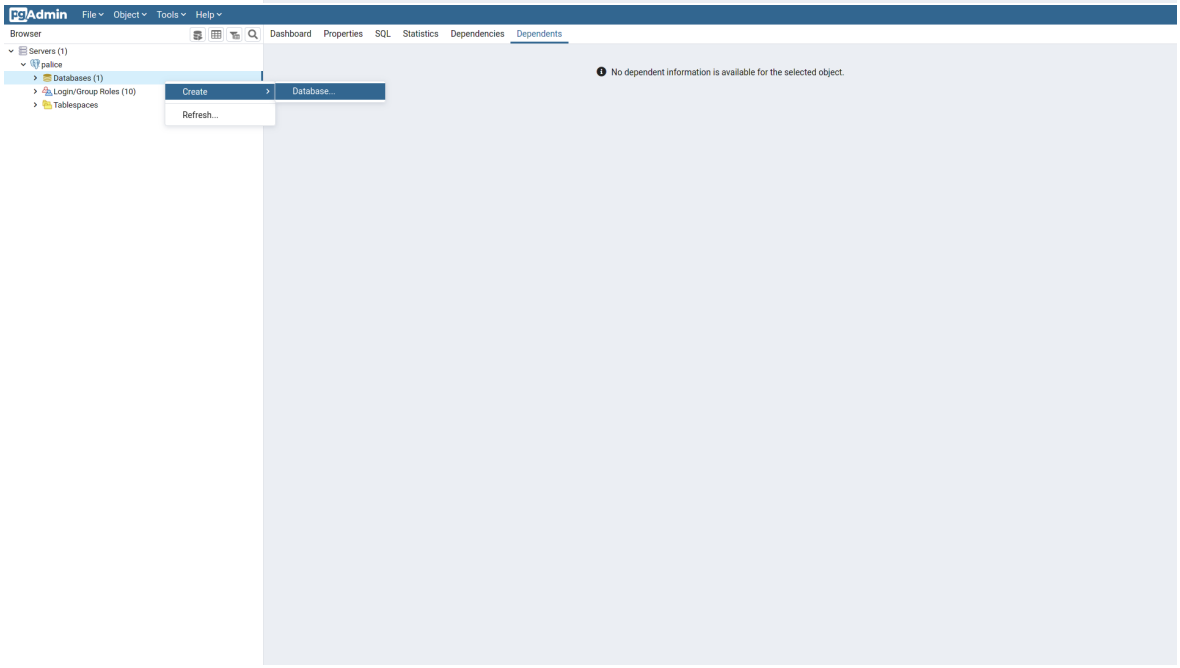
Stáhněte a nainstalujte si PostgreSQL
https://www.postgresql.org/download/
Stáhněte a nainstalujte si pgAdmin (Není vyžadovaný pro chod aplikace, ale budu ukazovat postup vytvoření databáze za jeho pomoci.)
https://www.pgadmin.org/download/

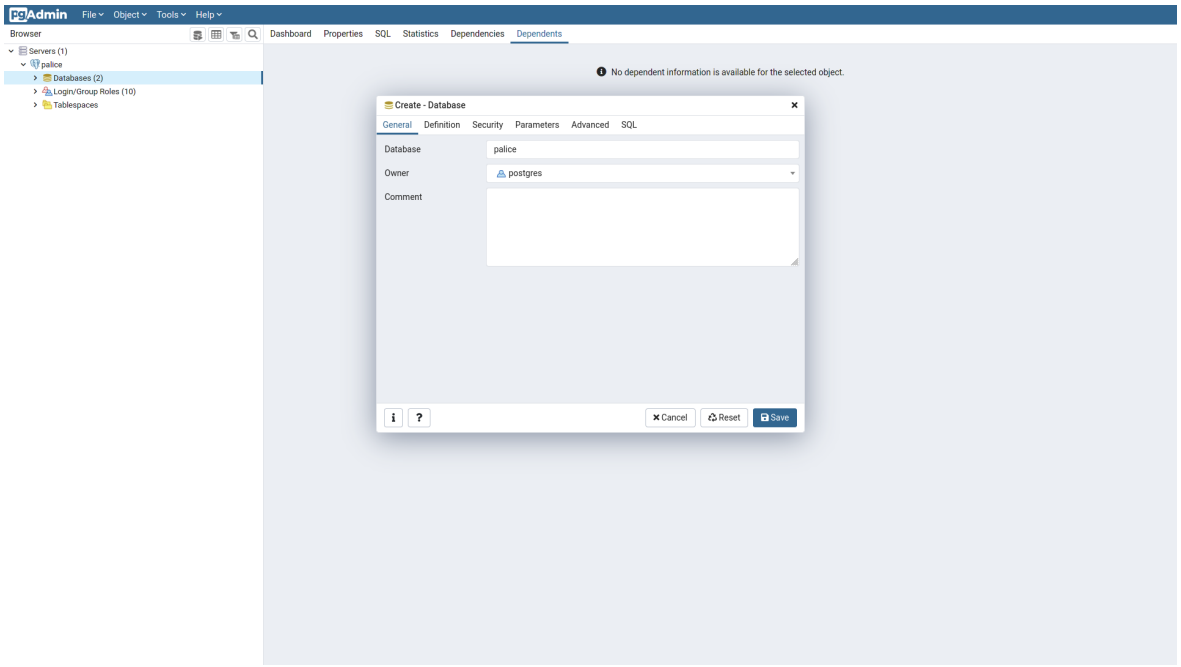
1. Spusťe pgAdmin
2. Vytvořte server se jménem **palice** a jako Hostname nastavte **localhost** s defaultním portem **5432**
Jako Username nastavte **postgres** a jako heslo **root**
3. Vytvořte databázi s uživatelem **postgres** a pojmenujte ji **palice**
Obrázkový návod:


pgAdmin
File
Object
Tools
Help
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Servers
Create
Server Group...
Server...
Refresh...
Remove Server Group
Properties...
No dependent information is available for the selected object.


pgAdmin
File
Object
Tools
Help
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Servers
No dependent information is available for the selected object.
Create - Server
General
Connection
SSL
SSH Tunnel
Advanced
Name
palice
Server group
Servers
Background
Foreground
Connect now?
Comments
Either Host name, Address or Service must be specified.
Cancel
Reset
Save


pgAdmin
File
Object
Tools
Help
Browser
Servers
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
No dependent information is available for the selected object.
Create - Server
General
Connection
SSL
SSH Tunnel
Advanced
Host name/address
localhost
Port
5432
Maintenance database
postgres
Username
postgres
Password
Save password?
Role
Service
Cancel
Reset
Save


pgAdmin
File
Object
Tools
Help
Browser
Servers (1)
pallice
Databases (1)
Login/Group Roles (10)
Tablespaces
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
No dependent information is available for the selected object.
Create
Database...
Refresh...



Palice

Jestliže máte nainstalované a nakonfigurované všechny programy, tak můžete stáhnout a spustit samotnou webovou aplikace.

```
git clone https://github.com/Daybringer/chytra-palice --recursive
cd chytra-palice
sh build.sh
```

https://www.pgadmin.org/download/
Po úspěšné instalaci balíčků vytvořte v složce **server** soubor ***.env*** s následujícími proměnnými:

```
GOOGLE_CLIENT_ID=922077049204-lh8mmn4bak6bj81v9r2n3ir522c2gqu3.apps.googleusercontent.com
GOOGLE_SECRET=tajnyUdajKteryVamZaslu
JWT_SECRET=VamiZvolenyNahodnyKlic
```

Poté z kmenové složky spusťte skript pojmenovaný start.sh, který se nachází v té samé kmenové složce.

```
sh start.sh
```

3.2 Uživatelský manuál

3.2.1 Články

3.2.2 Soutěž

3.2.3 Práce

3.2.4 Hlavní panel

Závěr

Závěr obsahuje shrnutí práce a vyjadřuje se k míře splnění jejího zadání. Dále by se zde mělo objevit sebehodnocení studenta a informace o tom, co nového se naučil a jak vnímal svou práci na projektu.

Seznam použité literatury

[Fow03] Martin Fowler. *Patterns of enterprise application architecture*. Boston: Addison-Wesley Professional, 2003.

[Mar00] Robert C. Martin. *Design Principles and Design Patterns*. www.objectmentor.com, 2000.

Seznam obrázků

Seznam tabulek