

*Semestrální práce pod názvem:*

**GAMENDAR**

KIV/WEB

*Vypracoval:*

**Tomáš Květoň** ([kvetont@students.zcu.cz](mailto:kvetont@students.zcu.cz))

**12.12. 2017**

## O APLIKACI

---

Gamendar. Již z názvu je možná zřejmé, že se název skládá ze dvou slov: Game a Calendar. Celá tahle aplikace je zaměřená na předem specifickou kategorií uživatelů. Jedná se o takovou aplikaci, která nabízí uživateli, možnost vytvořit si vlastní místnost, kterou může sdílet s ostatními. Tyto místnosti jsou pak používány pro zakládání akcí, do kterých se uživatelé mohou přidávat.

Aplikace by měla zajistit lepší organizaci a přehled o stavu účasti například nějaké herní komunity.

Aplikace byla navržena na základě mnou již existující aplikace s velmi podobnými funkcemi. Aplikace fungovala v rámci komunitního portálu a tento projekt přináší samostatnost aplikace a návrh takový, že aplikace má volnou ruku ve vytažení API, například pro Discord, se kterým je možné dané místnosti v budoucnu spárovat.

Stav, ve kterém je aplikace představena je pouze betou. Jedná se o demonstraci, která je v jádru plně funkční, ale různé funkčnosti okolo jsou stále ještě nedostupné, například uživatelská změna nastavení.

Tahle verze je zaměřená na demonstraci jádra aplikace – kalendář akcí.

Celá aplikace je také dostupná na stránce GitHub:

<https://goo.gl/MrPi8k>

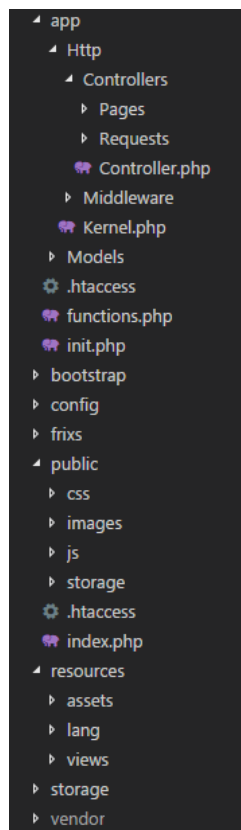
## TECHNOLOGIE

---

Aplikace na front-endu využívá klasické HTML5, CSS3 s pomocí preprocesoru SASS (SCSS), JavaScript především využívá framework JQuery a pro build projektu je využit template engine Blade z frameworku Laravel. Na back-endu máme dle zadání čisté PHP bez frameworku. Build a načtení tříd je pomocí Composeru.

# ADRESÁŘOVÁ STRUKTURA

---



Celá adresářová struktura je inspirována a velmi podobná struktuře frameworku Laravel.

**app** obsahuje samotnou aplikaci – controllery, requesty, atp.

**bootstrap** obsahuje spouštěč celé aplikace, která rozjede celý proces načtení.

**config** v sobě obsahuje, v souborech rozdělené, nastavené celé aplikace, například důležité konstanty.

**frixs** je mnou předhotovený framework, obsahující několik jednoduchých tříd, které zjednoduší vývoj. Některé jsou z předchozích starších projektů, některé jsou novější.

**public** obsahuje vše k čemu má klient přístup i na webu. JScripty, zkompilevané CSS, uploads a hlavní root celé aplikace, kde je zobrazován veškerý obsah – index.php.

**resources** obsahuje nezkompilované části front-endu, například CSS, JS, HTML (views) a překlady stránky.

**storage** obsahuje cache aplikace, cache například z views.

**vendor** obsahuje 3rd-party obsah. V našem případě Laravel Blade template engine.

# ARCHITEKTURA APLIKACE

---

Pokusím se architekturu nejprve představit od začátku requestu z klienta.

Veškerý obsah je odkazován na jeden root soubor a tím je *index.php* ve složce *public*. Ten načte všechny třídy z projektu spustí procedury, které jsou zapotřebí hned na začátku a spustí aplikaci pomocí třídy *App* ze složky *bootstrap*, která se o zbytek postará. Třída *App* se přesune do třídy *Route* z frameworku *frixs*, která se stará o rozpoznání URL a správné přesměrování a načtení korektního obsahu. *Route* má ještě přídružené třídy *RouteRequest* a *Router*. Všechny tyto 3 třídy jsou základem routování v naší aplikaci. Při routování dochází k využití třídy *Kernel*, ve které jsou nadefinované veškeré přístupy a práva. Na konci procedur routování je proces přesměrován už na daný controller stránky a ten se postará o správné nasměrování a načtení příslušného view.

Uvedu klíčový obsah:

- **App\Http\Controllers\Pages** Controllery všech stránek webu.
- **App\Http\Controllers\Requests** Všechny Requesty aplikace. Request controller má 2 povinné metody *inputValidation* a *process*.
- **App\Http\Kernel::class** nadefinované oprávnění a přístupy do všech koutů aplikace.
- **App\Http\Middleware** Validace nadefinovaných přístupů ve třídě *Kernel*.
- **App\Models** Modely představující tabulky databáze. Modely jsou pouze statické a neudrží v sobě data opravdových „Modelů“ – Nebudeme vytvářet nový Laravel, když už ho někdo vytvořil.
- **Bootstrap\App::class** Inicializace.
- **Frixs\Routing** Přesměrování, rozpoznání URL, správné spuštění příslušného controlleru.
- **Frixs\Config\Config::class** Načtení config souborů.
- **Frixs\Language\Lang::class** Načtení obsahu v příslušném jazyce (zatím jen **en**).
- **Frixs\Database\Connection::class** Načtení správného wrapperu databáze (v našem případě jen MySQL - PDO).
- **Frixs\Validation\Validate::class** Wrapper, který snadněji zvaliduje pole (input data).
- **Resources\Views** Views aplikace.

# UŽIVATELSKÁ PŘÍRUČKA

---

Jak bylo již řečeno, aplikace je navržena pro specifickou skupinu uživatelů. Předpokládáme tedy, že uživatel je znalý v oblasti počítačů, přesněji v pohybu na internetu, registrace atp.

Pokud nejste přihlášení, tak jste automaticky přesunuti na stránku, kde je možné se zaregistrovat nebo přihlásit s existujícím účtem.

Pokud jste přihlášen, tak se ocitnete na dashboardu, což je hlavním rozcestníkem aplikace. Zde můžete vyhledat a přidat se do již existující skupiny nebo vytvořit svoji vlastní (pro demonstraci je plně funkční pouze PUBLIC místnost).

Přesunutím se do konkrétní místnosti můžete vidět hlavní menu, kde se zobrazují akce. Pokud máte příslušná práva, tak nahoře v hlavičce pod názvem místnosti je možnost nastavení. Je možné nastavit práva uživatelů v místnosti nebo změnit defaultní nastavení skupiny.

Při vytváření akce není možná zřejmě co znamenají sekce nebo k čemu slouží. Kdyby zde sekce nebyly, tak uživatelé se mohou přidávat do akcí, ale nijak neseřazeni. Vytvořením sekcí můžete tomuto předejít. Ve hrách, například může být akce na dungeon a je potřeba dopředu znát počet různých tříd hráčů, kteří o akci mají zájem. Tak vytvořím příslušnou akci se sekcemi, například: Léčitel, Bojovník, Obránce. Dané sekce mohou ještě omezit na počet uživatelů a oni se mohou do daných sekcí, v rámci akce, hlásit.

## **TESTOVÁNÍ**

Pro vyzkoušení aplikace přikládám předem vytvořené 4 účty. 2 z nich mají už nějaké nastavení ohledně toho, že patří do nějaké skupiny či eventu. Ale není problém si vytvořit účet nový nebo vytvořit novou místnost či event.

Registrace je nastavená bez ověřování emailem, z důvodu, prozatím, bez-produkčního vydání.

### **Účty:**

#### **TestObject1**

Email: [testobject1@testobject.xyz](mailto:testobject1@testobject.xyz)

PW: TestObject12345

#### **TestObject2**

Email: [testobject2@testobject.xyz](mailto:testobject2@testobject.xyz)

PW: HeSlo543

#### **JenJa**

Email: [testobject3@testobject.xyz](mailto:testobject3@testobject.xyz)

PW: hesloBEZhesla

#### **NickTester**

Email: [testobject4@testobject.xyz](mailto:testobject4@testobject.xyz)

PW: SpatneHeslo1