

# Generátor úloh do aplikované kryptografie Dokumentace

Michal Homola,  
Dominik Chrenčík,  
Jiří Marák,  
Vojtěch Lukáš

22. dubna 2023

# Obsah

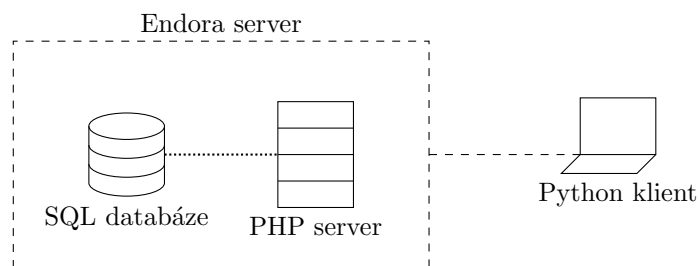
Úvod	1
1 Architektura	1
1.1 Konstrukce databáze . . . . .	1
1.2 Generátor hodnot . . . . .	2
1.3 API . . . . .	2
2 Komentáře	3

## Úvod

Předmětem této dokumentace je představit vizi projektu s názvem „Generátor kryptografických úloh“. První část bude věnována teoretickému popisu systému jako celku. ...

## 1 Architektura

Schéma systému lze vidět na obr. 1. Úlohy jsou uloženy v SQL databázi. K této databázi má přístup pouze webový PHP server. Ten slouží jako „prostředník“ mezi klientem a databází. Dále do úloh vkládá generované hodnoty (klíče apod.). Klientská aplikace funguje jako přístupový bod a sehrává roli prezentační vrstvy. Pro jednoduchost je vyvinuta v jazyce Python, využívá pouze konzolové prostředí.



Obrázek 1: Schéma systému

### 1.1 Konstrukce databáze

V tabulce 1 lze vidět strukturu SQL databáze. Sloupec **ID** slouží jako primární klíč databáze, **Kód** úlohy pak slouží pro snazší rozlišení úloh. V buňce **Zadání** se nachází textový popis úlohy. Zde stojí za povšimnutí, že všechny číselné hodnoty důležité k výpočtu jsou nahrazeny zástupnými znaky „\$n“. Na místa těchto znaků bude logika v back-endu vkládat vygenerované hodnoty. Díky tomu bude

Tabulka 1: Struktura SQL databáze

ID INT	Kód VARCHAR(5)	Zadání TEXT	Nápověda TEXT	Výsledek TEXT
1	PR	Rozhodněte (ano/ne) zda je číslo $n = \$1$ prvočíslo	...	NULL
2	RS Ae	Zašifrujte zprávu $m = \$4$ , pomocí RSA kryptosystému. Prvočísla jsou $p = \$1$ ; $q = \$2$ , a soukromý klíč je $e = \$3$	...	NULL
⋮	⋮	⋮	⋮	⋮

možno jednu úlohu řešit vícekrát, pokaždé s jinými parametry. Pole **Výsledek** je záměrně prázdné – správný výsledek zde vloží až server, který tuto hodnotu vypočítá podle vygenerovaných parametrů.

Uživatel si bude moct vybrat jaký typ bude chtít řešit, back-end si tuto úlohu podle jejího kódu vytáhne z databáze, opatří ji vygenerovanými operandy a spolu se správným výsledkem a nápovědou ji zašle uživateli, jak lze vidět v diagramu na obr. 2.

## 1.2 Generátor hodnot

Modul generace hodnot je pro tento projekt zcela klíčový. Byl implementován přímo v rámci back-end serveru, taktéž v jazyce PHP. Pro každý typ úlohy byla vytvořena jedna funkce, která vygeneruje pseudonáhodné operandy a předá je jako svou návratovou hodnotu.

Server pak podle kódu žádané úlohy zažádá o její prototyp SQL server a zavolá příslušnou funkci pro doplnění vygenerovaných hodnot. Takto upravenou úlohu zabalí jako JSON objekt a pošle uživateli.

## 1.3 API

Architektura back-endu je navržena podle doporučení REST API. Celé řešení je založeno na [1]. Od začátku byl projekt vyvíjen přímo na serveru pro usnadnění přístupu. URL pro zasílání dotazů: <http://vut-fekt-mpckry-gr14.8u.cz/index.php>. Všechny implementované dotazy jsou zmíněny v tabulce 2.

Jako odpověď na tyto dotazy server zašle JSON objekt, který bude již obsahovat vygenerované hodnoty i výsledek.

## Vývojový diagram

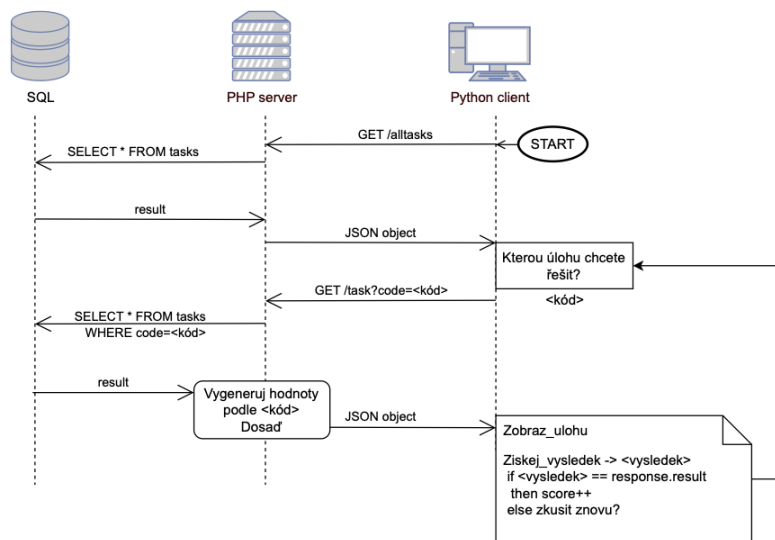
Tabulka 2: API funkce serveru

URL	popis	použití
/alltasks	zašle všechny úlohy z DB	<url>/alltasks
/task?code=<code>	zašle úlohu s daným kódem	<url>/task?code=pr
/randomtask	zašle náhodnou úlohu	<url>/randomtask

## 2 Komentáře

### Reference

- [1] SONI, Sajal. How to build a simple REST API in PHP. *Envato Tuts+* [online]. 27-5-2021 [cit. 2023-03-25]. Dostupné z: <https://code.tutsplus.com/tutorials/how-to-build-a-simple-rest-api-in-php--cms-37000>



Obrázek 2: Vývojový diagram systému