Serwer Knut - Dokumentacja

Wydanie 0.1

Wiktor Idzikowski

May 05 2010

Spis treści

Jest to dokumentacja techniczna przeznaczona dla programistów, którzy chcą zrozumieć jak działa serwer Knuta - repozytorium testów i odpowiedzi. Składa się ona z opisu klas, atrybutów i metod.

Program został napisany w języku Python, przy użyciu frameworka Django.

Zawartość:

Opis URL/API, ścieżek zasobów i sposobu komunikacji z serwerem

Komunikacja z serwerem odbywa sie przy pomocy protokołu HTTP, w szczególności metod GET i POST.

Metody edytujące testy weryfikują uprawnienia użytkownika za pomocą loginu i hasła, które muszą być przesłane w żądaniu.

Lista ścieżek zasobów:

- */test_upload - zapisuje test na serwerze*
- */test_list - listuje testy użytkownika*
- */test_list_public - listuje testy publiczne wszystkich użytkowników.*
- */test_delete/test_id - usuwa test o podanym id*
- */questions_download - Metoda zwraca plik z pytaniami do testu*
- */answers_download - Metoda zwraca plik z odpowiedziami do testu*
- */user_answers_download - Pobiera odpowiedzi ucznia w formacie xml*
- */results_upload - zapisuje wyniki ucznia na serwerze*
- */results_list - listuje wyniki testu wszystkich uczniów dla podanego testu*

Opis ścieżek zasobów:

- **/test_upload** - zapisuje test na serwerze. Metoda sprawdza czy użytkownik o podanym loginie i haśle istnieje w bazie danych a następnie zapisuje nowy test do bazy danych i przenosi odpowiednie pliki z pytaniami i odpowiedziami do katalogu "test_files/nr_testu/"

  Oczekuje na metodę POST zawierającą:

  - Login użytkownika - nauczyciela
  - Hasło użytkownika - nauczyciela
  - Tytuł testu
  - Instrukcje do testu
  - Hasło zabezpieczające test przed pobraniem - jęśli test prywatny
  - Kategorię testu

  - Wersję testu

- **/test_list** - listuje testy użytkownika. Metoda zwraca dokument xml z listą testów użytkownika, w szczególności nie pokazuje testów innych użytkowników.

  Oczekuje na metodę POST zawierającą:

  - Login użytkownika - nauczyciela
  - Hasło użytkownika - nauczyciela

- **/test_list_public** - listuje testy publiczne wszystkich użytkowników. Metoda zwraca dokument xml z listą wszystkich testów publicznych.

  Metoda nie oczekuje na argumenty.

- **/test_delete/<test_id>** - usuwa test o podanym id. Metoda sprawdza czy użytkownik o podanym loginie i haśle istnieje w bazie danych i jest autorem danego testu. Następnie usuwa test z bazy danych i pliki z dysku.

  Oczekuje na metodę POST zawierającą:

  - Login użytkownika - nauczyciela
  - Hasło użytkownika - nauczyciela

- **/questions_download** - Metoda zwraca plik z pytaniami do testu. Jeśli test jest prywatny sprawdza dodatkowo czy użytkownik (uczeń) podał poprawne hasło testu, lub czy użytkownik (nauczyciel) jest właścicielem testu. Jeśli żądanie wysłane przez ucznia zakończyło się odesłaniem testu, zapisuje tą informacje w bazie danych. Uczeń wysyła więc id testu, swój login i hasło do testu. Jeśli nauczyciel pobiera pytania do edycji to wysyła id testu, swój login i swoje hasło.

  Oczekuje na metodę POST zawierającą:

  - Id testu
  - Login użytkownika - ucznia lub nauczyciela
  - Hasło użytkownika - ucznia lub nauczyciela

- **/answers_download** - Metoda zwraca plik z odpowiedziami do testu. Logika jak w questions_download.

  Oczekuje na metodę POST zawierającą:

  - Id testu
  - Login użytkownika - ucznia lub nauczyciela
  - Hasło użytkownika - ucznia lub nauczyciela

- **/user_answers_download** - Pobiera odpowiedzi ucznia w formacie xml. Metoda sprawdza czy użytkownik o podanym loginie i haśle istnieje w bazie danych.

  Oczekuje na metodę POST zawierającą:

  - Login użytkownika - nauczyciela
  - Hasło użytkownika - nauczyciela
  - Id testu
  - Id użytkownika - ucznia
  - Id odpowiedzi

- **/results_upload** - zapisuje wyniki ucznia na serwerze. Metoda zapisuje login ucznia, id testu i wynik ucznia w bazie danych oraz plik xml z odpowiedziami na dysku w katalogu "results_files/<id_wyników>/".

  Oczekuje na metodę POST zawierającą:

  - Login użytkownika - ucznia
  - Id testu
  - Punkty uzyskane na teście
  - Punkty uzyskane na teście procentowo

- **/results_list** - listuje wyniki testu wszystkich uczniów dla podanego testu. Metoda sprawdza czy użytkownik (nauczyciel) o podanym loginie i haśle istnieje w bazie danych.

  Oczekuje na metodę POST zawierającą:

  - Login użytkownika - nauczyciela
  - Hasło użytkownika - nauczyciela
  - Id testu

Opis strony internetowej dostępnej z przeglądarki

Strona internetowa programu z opisem działania i możliwością pobrania wszystkich kodu. Użytkownik ma też możliwość przeglądanie testów.

Dostępne podstrony:

- **/** - strona główna programu z podstawowymi informacjami i linkami do programów do pobrania.
- **/categories** - strona listująca kategorie testów
- **/categories/<category-id>** - strona listująca wszystkie testy w wybranej kategorii

Opis klas bazy danych

3.1 Klasa opisująca tabelę użytkownika - nauczyciela

class `User` (**args, **kwargs*)

Klasa użytkownika wysyłającego testy - nauczyciela

Opis pól:

login

Login, identyfikator nauczyciela (30 znaków)

password

Hasło (30 znaków)

full_name

Imię i nazwisko (100 znaków)

3.2 Klasa opisująca tabelę kategorii testów

class `Category` (**args, **kwargs*)

Klasa kategori testów

Opis pól:

name

Nazwa kategorii (100znaków)

3.3 Klasa opisująca tabelę testów

class `Test` (**args, **kwargs*)

Klasa testu, dane opisujące test zapisywane są w bazie danych, natomiast pliki z pytaniami i odpowiedziami na dysku.

Opis pól:

user

Klucz obcy do użytkownika, który wysłał testy

category

Klucz obcy do kategorii testu

title

Tytuł testu (30 znaków)

instructions

Instrukcje do testu (256 znaków)

password

Hasło testu (30 znaków)

version

Wersja testu (liczba całkowita)

id_unq

Unikalne id testu (10 znaków)

3.4 Klasa opisująca tabelę wyników testu

class `Result` (**args, **kwargs*)

Klasa wyników testu. Wynik punktowy zapisywany jest bazie danych. Lista odpowiedzi udzielonych przez ucznia zapisywana jest na dysku w formacie xml.

Opis pól:

user_id_unq

Unikalne id użytkownika (30 znaków)

test_id_unq

Unikalne id testu (10 znaków)

points

Wynik z testu wyrażony w punktach (liczba zmiennoprzecinkowa)

points_percentage

Wynik z testu wyrażony w procentach (liczba zmiennoprzecinkowa)

ts_created

Stempel czasowy

3.5 Klasa opisująca tabelę użytkownika rozwiązującego testy - ucznia

class `TestUser` (**args, **kwargs*)

Klasa opisująca obiekt rozwiązywanego testu. Zapisuje datę rozpoczęcia i odesłania wyników testu.

Opis pól:

user_id_unq

Unikalne id użytkownika (30 znaków)

test_id_unq

Unikalne id testu (10 znaków)

ts_created

Stempel czasowy

ts_returned

Stempel czasowy

ROZDZIAŁ 4

Indeksy i tabele

- *Index*

Indeks modułów

K

Indeks