

Product Backlog

System do zarządzania egzaminami ustnymi

Product Owner

Zbigniew Rębacz

SCRUM Master

Michał Pachel

Kontrybutorzy

Michał Jancarz, Michał Gawryluk, Michał Szura, Konrad Welc, Arkadiusz Koszczan

I. Szkielet Aplikacji – PHP, HTML, CSS, Bootstrap

- **Stworzenie głównego menu strony**
 - W trakcie pracy nad tym zadaniem należy uwzględnić wszystkie podstawowe sekcje, które będzie zawierał nasz serwis takie jak kontakt, pomoc, strona tytułowa, autorzy projektu itp.
 - Ponadto na pasku menu powinna znajdować się zakładka „przenosząca” użytkownika do ekranu logowania.
- **Stworzenie strony tytułowej**
 - Strona tytułowa powinna zawierać informację marketingowe np. „Dlaczego warto zarejestrować się w naszym serwisie?”. Za przykład może posłużyć strona tytułowa serwisu github.com.
- **Stworzenie szkieletu strony**
 - Celem zadania jest stworzenie uniwersalnego szablonu, który pozwoli nam na szybkie tworzenie kolejnych podstron. Jednocześnie dzięki temu zadaniu unikniemy redundancji w kodzie.
- **Stworzenie globalnego pliku służącego do generowania znacznika „head”**
 - Klient programista powinien mieć możliwość wyboru tytułu dla swojej podstrony. Tytuł nie może być statyczny, czyli taki sam dla wszystkich podstron!
 - Powinna istnieć możliwość przekazania ścieżek do skryptów oraz ścieżek do arkuszy

CSS.

- **Stworzenie stopki**
 - Stopka powinna zawierać informację o prawach autorskich (COPYRIGHT wystarczy)
 - Jeżeli jest taka możliwość w stopce należy umieścić dodatkowe menu umożliwiające alternatywny sposób poruszania się po serwisie.
- **Stworzenie podstrony z autorami projektu**
 - Strona powinna zawierać listę wszystkich osób, które przyczyniły się do powstania aplikacji.
- **Stworzenie podstrony z informacjami kontaktowymi**
 - Strona powinna zawierać informację kontaktową do pomocy technicznej np. e-mail, telefon itp.
 - Na stronie powinna być wyszczególniona nazwa organizacji, która zajmuje się pomocą techniczną (nazwa naszej grupy?)
- **Stworzenie podstrony z formularzem kontaktowym**
 - Głównym celem tego zadania jest stworzenie formularza, który umożliwi wysłanie wiadomości e-mail do pomocy technicznej.
 - Formularz powinien znajdować się na podstronie z informacjami kontaktowymi.
 - Powinien zawierać zabezpieczenie przeciwko botom.
 - Dane powinny być sprawdzane.
- **Stworzenie formularza rejestrowania użytkowników (egzaminatorów)**
 - Głównym celem zadania jest stworzenie formularza rejestracyjnego.
 - Zabezpieczenie przeciwko botom.
 - Dane powinny być sprawdzane.
- **Stworzenie formularza logowania się do systemu**
 - Po zalogowaniu użytkownik powinien zostać przeniesiony na główną stronę aplikacji (Strona użytkownika).
- **Stworzenie mechanizmów umożliwiającego walkę z botami**
 - Celem zadania jest generowanie losowego obrazka na podstawie, którego użytkownik może odgadnąć kod dostępu.
 - Wizja: losujemy dwie liczby od 1 do 10 oraz działanie (dodawanie, odejmowanie itp.), generujemy obrazek z tym działaniem np. $2 * 4$, następnie pytamy się użytkownika jaki jest wynik tego działania.

- **Stworzenie podstrony, która wyświetli informację o zaistniałym błędzie (szablon)**
 - Jeżeli w systemie pojawi się błąd powinniśmy poinformować użytkownika o tym w cywilizowany sposób.
 - Przykładowy sposób wywołania: „showError(\$message);”

II. Zarządzanie aplikacją przez użytkownika

Generalna uwaga do zadań: Po zalogowani nie używamy mechanizmów stosowanych przeciwko botom.

- **Stworzenie strony użytkownika**
 - Po zalogowani użytkownik powinien zostać przeniesiony na swoją stronę domową.
 - Na tej stronie powinny znajdować się statystyki użytkownika takie jak liczba egzaminów itp.;
 - Ponadto na stronie powinna wyświetlać się lista najbliższych egzaminów.
- **Stworzenie menu użytkownika systemu**
 - Menu powinno znajdować się w górnej części podstrony tzn. nie powinno kolidować z głównym menu aplikacji.
 - Menu powinno być poziome.
 - Do elementów menu zaliczamy: „Mój Profil”(Strona użytkownika), „Dodaj egzamin”, „Lista egzaminów”, „Edytuj ustawienia osobiste”. Jeżeli użytkownik jest administratorem to po prawej stronie panelu pojawia mu się zakładka „Zarządzaj użytkownikami”, a pozostałe opcje są niedostępne.
- **Stworzenie podstrony służącej do modyfikowania ustawień osobistych przez użytkownika I**
 - Użytkownik powinien być w stanie zmienić kilka część parametrów podanych przy rejestracji i wprowadzić dodatkowe parametry(opcjonalne). Dlatego konieczne jest stworzenie odpowiedniego formularza.
 - Nie może zmienić adresu e-mail podanego przy rejestracji!
- **Stworzenie podstrony służącej do modyfikowania ustawień osobistych przez użytkownika II**
 - Celem tego zadania jest stworzenie formularza służącego do zmiany hasła przez użytkownika.
- **Stworzenie podstrony służącej do modyfikowania ustawień osobistych przez**

użytkownika III

- Stworzyć tylne wyjście dla administratora systemu (Powinien móc edytować wszystko nawet hasło).
- **Stworzenie formularza tworzącego egzamin I (Podstawowe parametry egzaminu)**
 - Głównym celem tego zadania jest stworzenie formularza, który umożliwi zarejestrowanie nowego egzaminu w systemie
 - W tym formularzu powinny znaleźć się tylko dane podstawowe takie jak „Nazwa”, „Termin”, „Semestr”, „Rok”, „Czas trwania egzaminu”;
 - Podstrona powinna oferować tryb edycji.
- **Stworzenie formularza tworzącego egzamin II (Kalendarz)**
 - Celem zadania jest zintegrowanie kontrolki „harmonogram” z formularzem.
 - Dodać możliwość dodawania kolejnych dni egzaminu w oparciu o kontrolkę „harmonogram”
 - Podstrona powinna oferować tryb edycji.
- **Stworzenie formularza tworzącego egzamin III (Studenci)**
 - Głównym celem formularza jest stworzenie możliwości dodawania tak
- **Napisać parser, którego celem będzie wyłuskanie adresów e-mail z zadanego tekstu**
 - Celem zadania jest napisanie parsera, który rozbije tekst na pojedyncze adresy e-mail.
- **Napisać skrypt, który roześle i wygeneruje kod do studentów**
 - Dla każdego studenta należy wygenerować 25 znakowy unikalny kod w skali całego systemu.
 - Należy stworzyć szkielet wiadomości, która będzie zawierała link umożliwiający rejestrację przez studenta na egzamin.
- **Stworzenie bocznego menu egzaminu**
 - W menu bocznym powinny znaleźć się trzy następujące opcje: edycja, kalendarz, studenci;
- **Stworzenie podstrony wyświetlającą listę egzaminów danego egzaminatora**
 - Lista powinna zawierać wszystkie egzaminy przypisane do danego egzaminatora
 - Lista powinna zawierać podstawowe informacje o egzaminie takie jak nazwa termin itp. Celem tego zadania jest ułatwienie wyszukiwania przez użytkownika rekordów w liście.
 - Lista powinna być uporządkowana po dacie.
 - Lista powinna zawierać dwa dodatkowe przyciski „Edytuj” (Użytkownik powinien zostać przeniesiony do podstrony z kalendarzem egzaminu”) oraz „Usuń”.

III. Zarządzanie aplikacją przez użytkownika – administracja

Administrator nie jest użytkownikiem systemu i nie może posiadać własnej listy egzaminów.

- **Stworzenie podstrony wyświetlającej wszystkich użytkowników systemu w postaci listy.**
 - Do każdego użytkownika na liście powinny być przypisane dwa przyciski edytuj i usuń.
 - Element lista powinna wyświetlać podstawowe informacje o użytkowniku takie jak: login, imię, nazwisko, login itp.
- **Rozwiązać problem modyfikacji**
 - Po naciśnięciu przycisku edytuj administrator tak jakby staje się użytkownikiem tzn. w menu użytkownika odblokowują mu się menu użytkownika z tą różnicą, że zamiast swoich ustawień wyświetlają mu się ustawienia aktualnie modyfikowanego użytkownika. I tak mamy dostęp do jego egzaminów. Możemy w jego imieniu rozsyłać e-mail-e do studentów, modyfikować kalendarz egzaminu, blokować terminy itp.

IV. Rejestracja na egzamin przez studenta

- **Stworzenie listy wyświetlającej egzaminy na które student może się zarejestrować**
 - Po kliknięciu na link aktywacyjny student powinien zostać przeniesiony na stronę z listą egzaminów na które może się zarejestrować.
 - Student wybiera egzamin i zostaje przeniesiony na stronę, w której może dokonać zapisu na egzamin.
- **Stworzenie kodu odpowiedzialny za rejestrację na konkretny termin na egzamin**
 - Należy wyświetlić studentowi wszystkie wolne terminy.
 - Student wybiera termin z listy i naciska przycisk rejestruj.
- **Napisać mechanizm transakcji odpowiedzialny za ochronę terminu.**
 - Termin nie może być przypisany do dwóch lub większej ilości osób.

IV. Kontrolka Harmonogram – PHP, GD

- **Stworzenie skryptu, który wygeneruje obrazek z harmonogramem**
 - Przez harmonogram rozumiemy jeden dzień.
 - Obrazek powinien być skalowalny. Jeżeli przedziały czasowe przewidziane na egzamin są mniejsze to obrazek powinien być większy.

- **Stworzenie mechanizmu umożliwiający zaznaczanie pustych elementów na harmonogramie**
 - Harmonogram ma oferować możliwość zaznaczania dowolnego przedziału czasowego.
- **Stworzyć pojedynczy element harmonogramu**
 - Element harmonogramu powinien zawierać podstawowe informacje o zapisie (Jaka osoba jest do niego przypisana).
 - Element harmonogramu powinien dać się zaznaczać.
- **Napisać mechanizm, który będzie umieszczał elementy harmonogramu na harmonogramie**
 - Każdy element powinien znajdować się w odpowiednim miejscu na harmonogramie.

V. Bazy Danych – SQL, Skrypty

- **Stworzenie schematu bazy danych**
 - Wymodelować schemat bazy np. przy pomocy programu Edith.
 - Następnie model należy skonsultować z zespołem.
 - Uwzględnić uwagi zespołu.
 - Na koniec schemat należy wyeksportować do pliku graficznego i umieścić w miejscu widocznym dla wszystkich programistów np. repozytorium git.
- **Stworzenie kodu odpowiedzialnego za tworzenie bazy danych**
 - głównym celem tego zadania jest napisanie dwóch skryptów. Pierwszy z nich powinien tworzyć pustą bazę danych, natomiast drugi powinien utworzyć wszystkie potrzebne tabele (Patrz zadanie: „Stworzenie schematu bazy danych”).
- **Stworzenie diagramu encji**
 - Na podstawie schematu bazy danych oraz skryptu tworzącego bazę danych należy stworzyć diagram encji.
- **Stworzenie skryptu odpowiedzialnego za instalację/reinstalację bazy danych na systemie operacyjnym MS Windows**
 - Powinniśmy być w stanie zainstalować bazę pod Windowsem przy pomocy wywołania jednego skryptu powłoki (Technologię, które należy użyć w tym przypadku to „Batch File” lub „Power Shell”)
- **Stworzenie skryptu odpowiedzialny za instalację/reinstalację bazy danych na systemie operacyjnym GNU/Linux**
 - Powinniśmy być w stanie zainstalować bazę pod Windowsem przy pomocy wywołania jednego skryptu powłoki (Technologię, które należy użyć w tym

przypadku to „Bash”)

- **Dodanie do bazy kilku przykładowych danych**
 - Należy stworzyć skrypt sql, który wypełni bazę danych na dowolnej maszynie.
 - Zadanie jest bardzo ważne ponieważ pozwoli nam ono odtworzyć te same warunki testowe na wszystkich maszynach developerskich.

VI. Bazy Danych w kodzie PHP

- **Stworzenie kodu odpowiedzialny za łączenie się z serwerem baz danych**
 - Jeżeli połączenie nie dojdzie do skutku należy o tym poinformować użytkownika.
- **Opakowanie encji w klasy**
 - Należy napisać kod w PHP, który opakuje encje w klasy.
 - Każda klasa z encją to osobny plik.
 - Wszystkie zmienne należące do encji powinny być prywatne/chronione.
 - Dostęp do składowych powinien być realizowany przy pomocy metod z rodziny get/set.
- **Stworzyć kodu odpowiedzialną za komunikację z bazą danych**
 - Na każdym obiekcie z zadania „Opakowanie encji w klasy” powinienem być w stanie wykonać następujące operację insert, delete oraz update przy pomocy funkcji lub metod.
 - Celem zadania jest ukrycie przed klientem programistą wywołań sql.

V. Grafika na stronie

- **Stworzenie loga aplikacji.**
- **Stworzenie miniaturowej ikony aplikacji**
- **Stworzenie grafiki tylnego tła aplikacji**
- ...

VII. Przetestowanie zabezpieczeń aplikacji

- **Zabezpieczyć się przed atakami typu SQL Injection**
 - Celem zadania jest gruntowne przetestowanie systemu na podatność na ataki tego typu.