Faza konstrukcji

System Tutor (Tutor.pl)

Mateusz Markiewicz
Bartosz Sobocki
Czwartek, 8:15

Przypadki testowe

W celu przeprowadzenia wiarygodnych testów należy przygotować zestaw przykładowych, sztucznie wygenerowanych ogłoszeń obu stron, notatek, pomocy naukowych.

1. Przypadek 1

   ID: TC_1

   Tytuł: Dodanie ogłoszenia o chęci udzielania korepetycji

   Założenia wstępne: W systemie istnieje zalogowany użytkownik będący tutorem, mający możliwość dodawania nowych ogłoszeń

   Kroki:

   1. Wejdź w opcję dodawania nowego ogłoszenia
   2. Wprowadź poprawną dziedzinę: “matematyka”
   3. Wprowadź poprawny termin: 20.01.2020
   4. Wprowadź poprawną cenę korepetycji: 50zł
   5. Wprowadź poprawną lokalizację: Wrocław, ul. Joliot-Curie 15
   6. Potwierdź dodanie ogłoszenia

   Oczekiwany rezultat:

   1. Ogłoszenie zostało dodane do bazy, jest widoczne dla innych użytkowników
   2. Tutor widzi w swoim terminarzu swoje ogłoszenie, może otworzyć jego szczegóły

2. Przypadek 2

   ID: TC_2

   Tytuł: Rezerwacja spotkania z tutorem przez ucznia

   Założenia wstępne: W systemie istnieje zalogowany użytkownik będący uczniem, mogący rezerwować spotkania, istnieje testowe ogłoszenie tutora

   Kroki:

   1. Otwórz listę ogłoszeń

2. Znajdź testowe ogłoszenie tutora
3. Wejdź w szczegóły dotyczące tego ogłoszenia
4. Naciśnij przycisk rezerwacji tego ogłoszenia

Oczekiwany rezultat:

1. W terminarzach ucznia oraz tutora widnieje zaproponowane spotkanie
2. Tutor otrzymał powiadomienie o propozycji spotkania
3. Tutor widzi szczegóły dotyczące spotkania oraz ucznia, może zaakceptować oraz odrzucić propozycję

3. Przypadek 3

ID: TC_3

Tytuł: Akceptacja spotkania z uczniem przez tutora

Założenia wstępne: W systemie istnieje zalogowany użytkownik będący tutorem, istnieje testowa propozycja spotkania

Kroki:

1. Otwórz szczegóły dotyczące ogłoszenia
2. Naciśnij przycisk akceptacji propozycji

Oczekiwany rezultat:

1. W szczegółach widać zarówno dane dotyczące spotkania (podana data, lokalizacja, kwota, dziedzina), jak i dane dotyczące ucznia, które udostępnił (w naszym przypadku płeć, wiek, szkołę, mocne i słabe strony)
2. Wyświetlają się zarówno przyciski akceptacji oraz odrzucenia propozycji spotkania
3. Po naciśnięciu przycisku akceptacji wyskakuje komunikat “Potwierdzono spotkanie”
4. W terminarzu ucznia oraz tutora widać zaplanowane spotkanie
5. W szczegółach odnośnie spotkania uczeń widzi szczegółowe informacje o spotkaniu i tutorze
6. W szczegółach odnośnie spotkania tutor widzi szczegółowe informacje o spotkaniu i uczniu

Testowanie wymagań niefunkcjonalnych

- *szybki czas działania:* przygotowujemy dosyć słabe urządzenia testowe (smartphone i komputer połączony z wolną siecią, aby zasymulować najgorszy możliwy

przypadek), następnie sztucznie obciążamy naszą aplikację symulując największe przewidywane obciążenie, a następnie sprawdzamy ile czasu zajmują przykładowe scenariusze użycia aplikacji

Norma ISO/IEC:

Throughput time:

X = A / T

A -> liczba wykonanych zadań

T -> łączny czas testu

- *intuicyjny interfejs wersji mobilnej aplikacji oraz webowej* - przeprowadzamy ankietę na potencjalnych użytkownikach z różnych grup wiekowych oraz społecznych, sprawdzamy dla jakiej części ankietowanych interfejs był intuicyjny, prosząc ich o realizację przykładowych scenariuszy użycia aplikacji

Norma ISO/IEC:

Time Between Human Error Operations:

X= T / N

T -> czas działania

N -> liczba błędów użytkownika

- *bezpieczeństwo i anonimowość* - wszystkie wrażliwe dane naszych użytkowników będziemy szyfrować za pomocą odpowiednich algorytmów, uniemożliwiając (na tyle, na ile pozwala aktualna technika kryptografii) ich odszyfrowanie

Norma ISO/IEC:

Data encryption:

X=A/B

A -> ilość danych podlegających szyfrowaniu

B -> łączna ilość danych podlegających ochronie

- *bezawaryjność* - będziemy kontrolować jak często pojawiają się awarię (defekty) w naszej aplikacji, mierząc czas pomiędzy ich wystąpieniami

Norma ISO/IEC:

Failure density (Fault density):

X= NFAI / SIZE

Y= NFAU / SIZE

NFAI -> liczba wykrytych defektów

NFAU -> liczba wykrytych awarii

SIZE -> rozmiar projektu

- *spójność treści pomiędzy aplikacją mobilną i webową* - przeprowadzamy ankietę na dwóch grupach użytkowników - pierwsza używa wersji webowej przez określony czas, następnie dostaje aplikację w wersji mobilnej i przykładowe scenariusze użycia aplikacji do zrealizowania, druga grupa postępuje analogicznie dla drugiej wersji aplikacji, sprawdzamy dla jakiej części ankietowanych użycie innej, niż zazwyczaj, wersji aplikacji nie stanowiło problemu

  Norma ISO/IEC:

  Time Between Human Error Operations:

  X= T / N

  T -> czas działania

  N -> liczba błędów użytkownika
- *migracja systemu na inny serwer* - przeprowadzamy testową migracje systemu wraz ze sztucznie wygenerowanymi przykładowymi danymi, sprawdzamy czas takiej operacji oraz liczbę utraconych danych a następnie skalujemy to uwzględniając docelowy rozmiar systemu w stosunku do tego w testów

  Norma ISO/IEC:

  X= T

  T -> czas przenoszenia

Plan beta testowania

Pierwszym etapem beta testów dla naszej aplikacji, będzie przedstawienie jej naszej rodzinie oraz znajomym, prosząc ich o wcielenie się zarówno w rolę uczniów, jak i tutorów. Pomoże nam to zebrać wiele informacji zarówno na temat czytelności całej aplikacji, jak i wykrycia potencjalnych błędów. Będą oni testować aplikację na własnych telefonach oraz komputerach, co zapewni nam przetestowanie naszego produktu na różnego typu urządzeniach. Po tym etapie spodziewamy się uzyskać przede wszystkim opinii na temat tego, czy nasza aplikacja jest “user-friendly”, jak długo zajmuje nauka jej obsługi.

Następnym planowanym etapem jest zaangażowanie kilku aktywnych korepetytorów wraz z ich uczniami (obecnymi oraz potencjalnymi), którym udostępnimy naszą aplikację na okres 2 tygodni. Zapewni nam to przetestowanie naszego produktu przez docelowych użytkowników, spodziewamy się poznać ich zdanie zarówno na temat przejrzystości interfejsu, jak i przydatności oferowanych funkcjonalności. Będzie to pierwszy moment, w którym nasza aplikacja będzie wykorzystywana przez użytkowników końcowych, więc spodziewamy się również wykryć potencjalne błędy.

Ostatnim etapem beta testowania jest stworzenie wersji beta aplikacji oraz rozpowszechnienia jej za pomocą mediów społecznościowych. Dzięki temu nasz produkt dotrze do wielu docelowych użytkowników. Celem tego etapu jest wykrycie większości błędów, w czym pomoże duża liczba użytkowników oraz sprawdzenie jaka część użytkowników postanawia używać naszego produktu przez dłuższy czas. Jest to również ostatni moment na poznanie opinii o interfejsie i wprowadzenie w nim ewentualnych poprawek. Duża liczba użytkowników pozwoli nam również sprawdzić jak nasza aplikacja zachowuje się pod większym obciążeniem, czy jest wystarczająco responsywna.

Plan zarządzania ryzykiem

Świadomość tego, jakie czekają na nas zagrożenia i dobrze przemyślany plan działania w sytuacji wystąpienia któregoś z nich znacząco zwiększą szansę sukcesu naszej aplikacji. Główne ryzyka to:

Typ ryzyka	Nazwa ryzyka	Prawdopodobieństwo	Konsekwencje
1. Biznesowe	a) Brak zainteresowania ze strony uczniów	4/10	Poważne
	b) Brak zainteresowania ze strony tutorów	3/10	Poważne
	c) Zły model płatności	3/10	Bardzo poważne
2. Techniczne	a) Mało czytelny interfejs	1/10	Średnie
	b) Mała skuteczność algorytmu dopasowującego uczniów i tutorów	4/10	Bardzo poważne
	c) Zbyt mała responsywność aplikacji	4/10	Poważne

Plan działania w przypadku wystąpienia:

1.a) Kampania reklamowa w mediach społecznościowych oraz serwisie YouTube.com. Promocja na dodatkowe darmowe propozycje konsultacji w danym miesiącu oraz pokrycie 10% kosztów pierwszego spotkania danego użytkownika zaaranżowanego za pomocą naszej aplikacji.

1.b) Kampania reklamowa w środowisku studentów oraz nauczycieli. Promocja na dodatkowe darmowe propozycje konsultacji w danym miesiącu oraz pokrycie 10% kosztów pierwszego spotkania danego użytkownika zaaranżowanego za pomocą naszej aplikacji.
1.c) Próba balansu stosunku darmowych propozycji spotkań na miesiąc, a ceną dokupienia nowych. Zmniejszenie cen kosztem zwiększenia ilości reklam wyświetlanych w czasie działania aplikacji.
2.a) Zmiana interfejsu uwzględniając skargi użytkowników, ponowne testy nowego interfejsu
2.b) Analiza przyczyny niepoprawnego działania obecnego systemu rekomendacji, zaprogramowanie nowego (bądź użycie gotowego) systemu rekomendacji, wybranego w oparciu o w.w analizę.
2.c) Przeniesienie się na wydajniejszy model serwera, bądź całkowita zmiana serwera w przypadku braku takiej opcji.

Plan zarządzania jakością wytwarzania oprogramowania

Aby odpowiednio zarządzać jakością wytwarzania oprogramowania niezbędna będzie kontrola wersji programu oraz odpowiednich modułów.
Do tego posłużymy się systemem kontroli wersji **Git**.
Serwisy takie jak **GitHub** lub też **Gitea** pomogą nam prowadzić zdalne repozytoria.

Będziemy pracować w zespołach zajmujących się odpowiednio:

- aplikacją webową i jej testowaniem (zespół A),
- aplikacją mobilną i jej testowaniem (zespół B),
- bazą danych (zespół C),
- serwery, hardware firmy (Zespół D).

Będziemy pracować w scrumowych zespołach, które będą przy każdym sprincie starały się budować kolejną część systemu/aplikacji.

Dodatkowo będziemy posługiwać się diagramami ER bazy danych, aby zarządzanie nią było skuteczniejsze, wprowadzanie zmian prostsze, oraz aby można łatwiej było wykryć błędy jej konstrukcji itd.

Pomocne również będą tablice **User Story Mapping** dla każdego z zespołów. Zależy nam aby jak najsprawniej działał proces wytwarzania oprogramowania dopasowanego do

użytkownika już od rozpoczęcia prac, aby ograniczyć wprowadzanie dużych zmian do minimum.

Tablice pomogą nam także lepiej określać, w którym momencie wytwarzania oprogramowania się znajdujemy, oraz planować kolejne kroki, sprinty.

Przed każdym wprowadzeniem zmiany w gałęzi master będzie wykonywany **Code Review**, aby kod był czysty, przejrzystszy, podatny na rozszerzenia i zmiany, jak najprostszy w zrozumieniu oraz aby każdy najmniejszy błąd został wyłapany zanim zmiany ujrzą światło dzienne.

Plan wykonania produktu

Luty:

01 - 08 – określenie wymagań aplikacji

09 - 11 – wytworzenie User Story Mapping

12 - 24 – produkcja grafiki (grafik komputerowy)

12 - 24 – wstępne projektowanie architektury systemu (zespoły A, B)

12 - 24 – wstępne projektowanie bazy danych (zespół C)

24 - 29 – rozpisanie backlogów scrum'a

** - ** – tworzenie architektury serwerów (zespół D)

** - ** – tworzenie środowiska pracy (zespół D)

Marzec:

01 - 10 – projektowanie User Interface aplikacji webowej (zespół A)

01 - 10 – projektowanie User Interface aplikacji mobilnej (zespół B)

Marzec - Sierpień:

** - ** – tworzenie szkieletu i wersji testowych backend'u aplikacji webowej *(zespół A)*

** - ** – tworzenie szkieletu i wersji testowych backend'u aplikacji mobilnej *(zespół B)*

** - ** – tworzenie User Interface aplikacji webowej *(zespół A)*

**** - ** –** tworzenie User Interface aplikacji mobilnej *(zespół B)*

**** - ** –** praca nad tworzeniem i rozwojem bazy danych *(zespół C)*

**** - ** –** utworzenie i rozwój chmury, podłączenie przetestowanych elementów systemu z bazą i chmurą *(zespół D)*

**** - ** –** testowanie zaimplementowanych/zastosowanych rozwiązań *(zespoły A, B, C, D)*

Wrzesień:

01 - 15 – Kontakt z potencjalnymi klientami, prowadzenie ankiety, w celu uzyskania niezbędnych informacji na temat pożądanych funkcjonalności

16 - 30 – Wprowadzenie dodatkowych funkcjonalności, poprawa jakości aplikacji

Październik - Listopad:

**** - ** –** wypuszczenie wersji BETA wśród potencjalnych klientów, oczekiwanie na wyniki testów, rozeznanie wśród grupy testujących na temat błędów systemu, wydajności aplikacji, wyglądem oraz jakości oprogramowania

**** - ** –** poprawa zgłoszonych nieprawidłowości

Grudzień:

**** - ** –** prowadzenie kampanii reklamowej, promowanie produktu w internecie, prasie, mediach społecznościowych, telewizji

**** - ** –** prowadzenie kampanii reklamowej na zasadzie promowania produktu pomiędzy uczniami w szkołach oraz potencjalnymi tutorami na uczelniach na terenie największych miast uczelniach

**** - ** –** wypuszczenie produktu

Ocena zgodności

Czas niezbędny na wytworzenie aplikacji okazał się dużo dłuższy, niż oczekiwaliśmy. Różnica czasowa wynosi około 4 miesięcy (zamiast 7 miesięcy mamy prawie rok! 11 miesięcy).

Powstają dodatkowe koszty, np. korzystanie z serwisów GitHub lub Gitea.

Rozważamy więcej możliwości na poniesienie ryzyka, ale także więcej możliwości pozyskania niezbędnych informacji na temat użytkowania aplikacji przez użytkowników (prowadzenie testów wśród potencjalnych użytkowników, prowadzenie ankiet na temat funkcjonalności i wyglądu aplikacji).