Audyt techniczny dla [redacted]

Rozwój i komercjalizacja aplikacji "[redacted]"

1 Stan obecny

1.1 Kontekst

1.1.1 Firma świadczy usługi poligraficzne (uszlachetnianie druku)

1.1.2 Aplikacja pełni kluczową rolę w firmie

Zarządza całym procesem technologicznym firmy, od zamówień przez stan magazynowy do rozliczeń i fakturowania

1.2 Cele

1.2.1 Możliwość bezpiecznego, dalszego rozwoju aplikacji

1.2.2 Możliwość bezpiecznego uaktualniania oprogramowania / urządzeń korzystających z tego oprogramowania

1.2.3 Kroki potrzebne w celu skomercjalizowania oprogramowania

1.3 Użytkownicy

1.3.1 Klienci – składający zamówienia

- Firmy (drukarnie), składające zamówienia za pośrednictwem „okrojonej” wersji aplikacji web
- Aktualnie to ok. [redacted] użytkowników tego typu

1.3.2 Pracownicy produkcji – realizujący zamówienia

1.3.3 Pracownicy sprzedaży

1.3.4 Urządzenia – na których zamówienia są realizowane

1.3.5 API publiczne

- Czy jakiekolwiek API publiczne istnieje obecnie ?

- Czy jest udostępniane klientom / firmom zewnętrznym, czy dostęp dla klientów odbywa się jedynie za pośrednictwem aplikacji web?

1.4 Wdrożenie

1.4.1 [redacted]

- W oddziale w [redacted] ok. [redacted] pracowników firmy korzysta z aplikacji

1.4.2 [redacted]

- W oddziale w [redacted] ok. [redacted] pracowników firmy korzysta z aplikacji

1.5 Stos technologii

1.5.1 Back-end

1.5.1.1 Język: PHP

- wersja: 5.4.24

1.5.1.2 Baza danych: MySQL

- wersja: 5.6.19

1.5.1.3 Biblioteki zewnętrzne

1.5.2 Front-end

1.5.2.1 Języki

- HTML
- CSS
- JavaScript

1.5.2.2 Biblioteki zewnętrzne

1.5.3 System Operacyjny: OS X

1.6 Problemy

1.6.1 Informatyk który stworzył aplikacje odszedł

- Aplikacja była stworzona i rozwijana jak dotychczas przez jedną osobę
- Znacznie utrudniony kontakt z tą osobą, brak jakiejkolwiek dokumentacji technicznej, testów automatycznych aplikacji oraz ogromny dług techniczny nagromadzony na przestrzeni ponad 5 lat jest ogromnym problemem

1.6.2 Ogromny dług techniczny

Nagromadzony na przestrzeni ponad 5 lat już

1.6.2.1 Stara wersja PHP (język)

1.6.2.2 Stara wersja MySQL (baza danych)

1.6.2.3 Stare wersje bibliotek zewnętrznych

1.6.2.4 Problemy z kodem aplikacji

1.6.2.4.1 Ponad 10 tysięcy poważnych i krytycznych problemów

Ponad 10 tysięcy poważnych i krytycznych problemów wykrytych przy analizie statycznej kodu aplikacji

1.6.2.4.2 Brak rozdziału na warstwy

- W całości aplikacji najniższe warstwy aplikacji (tzw. warstwa persystencji) „pomieszane” z najwyższymi warstwami (tzw. warstwami widoku)
- Znacznie utrudnia naprawianie błędów tym bardziej rozwój aplikacji

1.6.2.4.3 Brak rozdziału na klasy i moduły

Całość aplikacji pisana jest w tzw. stylu proceduralnym programowania, nie obiektowym, co tym bardziej utrudnia zapoznanie się z aplikacją w celu naprawy błędów oraz dalszego jej rozwoju

1.6.2.4.4 Brak wydzielonej konfiguracji aplikacji

- Całość konfiguracji aplikacji "zaszyta" jest w samym kodzie aplikacji
- Brak możliwości konfigurowania aplikacji bez ingerencji w kod
- Różnice pomiędzy środowiskami w których aplikacja jest wdrożona (tj. [redacted]) "zaszyte" są w kodzie źródłowym aplikacji, zamiast być zdefiniowanymi w zewnętrznym pliku konfiguracyjnym a ew. specyficzne dla danej lokalizacji funkcjonalności aplikacji wydzielone do osobnego modułu

1.6.2.5 Problemy z modelem danych wykorzystywanym w aplikacji

- Ponad 200 tabeli w bazie danych
- Kompletny brak jakichkolwiek powiązań zdefiniowanych pomiędzy tabelami („relacji” / „foreign keys”) znacznie utrudnia zrozumienie ich roli i zależności

1.6.3 Brak dokumentacji technicznej

1.6.4 Brak logów aplikacji

Oprócz standardowego logu PHP aplikacja nie loguje żadnych informacji diagnostycznych co znacznie utrudnia rozpoznanie i zdiagnozowanie problemów opisywanych

1.6.5 Brak testów jednostkowych i integracyjnych

- Testy automatyczne pełnią kluczową role „samo-dokumentowania” funkcjonalności aplikacji
- Umożliwiają automatyczne zweryfikowanie poprawnego działania aplikacji
- Umożliwiają również swobodny rozwój aplikacji z uwagi na możliwość automatycznego zweryfikowania czy po ingerencji w kod aplikacja nadal działa
- Z uwagi na ich kompletny brak w aplikacji, jak również brak dokumentacji technicznej aplikacji oraz ogromny dług techniczny, rozwój aplikacji w obecnym stanie jest wręcz niemożliwy

1.6.6 Zawieszanie się aplikacji

Widoczne problemy przywieszania się interfejsu aplikacji web – widoczne jako „smugi” na kranie

1.6.7 Konieczność częstego restartowania komputera na którym aplikacja jest uruchomiona

- Raz na kilka dni komputer na którym aplikacja jest uruchomiona musi być restartowany – jako powód wskazane „bufory się zapychają”
- Z uwagi na brak logów i monitorowania w aplikacji, zdiagnozowanie przyczyny tego problemu jest znacznie utrudnione czy wręcz niemożliwe z uwagi na 10+ tysięcy problemów wykrytych przy analizie statycznej kodu oraz problemy z modelem danych

1.6.8 Zagrożenia bezpieczeństwa

- Tylko analiza statyczna kodu aplikacji wykryła kilka tysięcy potencjalnych problemów z bezpieczeństwem aplikacji

- Dodatkowo, fakt nieuaktualniania systemu na którym aplikacja jest uruchomiona, wersji języka oraz bazy danych przez okres ponad 5 lat już, znacznie zwiększa ryzyko

1.6.9 Problemy ze sprzętem komputerowym na którym aplikacja jest uruchomiona

1.6.9.1 Częste samo-restartowanie się komputera

Tylko w okresie czasu niecałych dwóch godzin dnia [redacted] listopada 2019, sprzęt komputerowy na którym aplikacja jest uruchomia sam restartował się przynajmniej dwu-krotne z uwagi na „błąd systemowy”

- Zrzuty ekranu wykonane zdalnie, w tym m.in. raport awarii oraz poważne problemy z serwerem baza danych
- Próbki logów pobrane zdalnie

1.6.9.2 Prawdopodobieństwo fizycznego uszkodzenia

- Istnieje prawdopodobieństwo, iż dysk komputera jest fizycznie uszkodzony, stąd problemy z bazą danych widoczne przy prostej operacji wykonywania EKSPORTU danych dnia [redacted] listopada 2019 r.
- Ponadto, sprzęt komputerowy nie jest właściwie przechowywany obecnie

1.6.10 Brak możliwości aktualizacji sprzętu komputerowego na którym aplikacja jest uruchomiona

Obawa, iż jakakolwiek ingerencja w sprzęt komputerowy (uaktualnianie systemu, wersji oprogramowania) sprawi, że aplikacja przestanie działać

1.6.11 Niedokończone funkcjonalności aplikacji

Nie wszystkie funkcjonalności aplikacji z których użytkownicy korzystają są kompletne, w tym m.in. definiowanie uprawnień dla użytkowników, obsługa urządzeń wykorzystywanych w oddziale w [redacted], itp.

1.6.12 Brak systemu kontroli wersji kodu

2 Stan docelowy

2.1 Sprzęt komputerowy

Z uwagi na problemy ze sprzętem komputerowym na którym aplikacja jest obecnie uruchomiona – w tym częste samo-restartowanie się komputera oraz prawdopodobieństwo fizycznego uszkodzenia – priorytetem nr. 1 jest dostępność sprawnie działającego sprzęt komputerowego

2.2 Środowisko aplikacji

2.2.1 Klaster Kubernetesa

Wykorzystanie technologii wirtualizacji na nowy sprzęcie komputerowym wdrożonym w obydwu lokalizacjach gdzie aplikacja jest uruchomiona, [redacted] daje możliwość uruchomienia klastra Kubernetesa.

Korzyści jakie płyną z tego rozwiązania to m.in. bardziej wydajne wykorzystanie zasobów, możliwość uruchamiania wielu (dedykowanych) instancji aplikacji, możliwość automatycznego monitorowania i uruchamiania nowych instancji aplikacji w przypadku jakichkolwiek problemów, zabezpieczenie przed utrata dostępu do aplikacji i wiele innych.

Biorąc pod uwagę, iż w razie jakichkolwiek problemów z aplikacja, nowy sprzęt komputerowy użytkownicy nie będą mogli „restartować” fizycznie, dostarczenie technologii tzw. samo-leczenia jakie dostarcza klaster Kubernetesa jest konieczne.

2.2.1.1 Wydajne wykorzystanie zasobów / redukcja kosztów

2.2.1.2 Automatyczna skalowalność / replikacja / samo-leczenie

W tym, automatyczne uruchamianie dodatkowych instancji aplikacji w przypadku wykrycia iż przestała działać ("disaster recovery" / "self-healing") lub liczba działających instancji spadła poniżej zdefiniowanego progu, automatyczna replikacja bazy danych, oraz możliwość uruchomienia dodatkowych instancji aplikacji, bazy danych – w miarę potrzeby / zwiększenia obciążenia.

2.2.1.3 Łatwość stawiania środowisk aplikacji

- Możliwość wykonania klonów obecnego środowiska w formie kontenera Docker uruchamianego w klastrze Kubernetesta
- Możliwość uruchomienia nowego, testowego środowiska w formie kontenera Docker w klastrze Kubernetesta, równolegle do działającej, produkcyjnej wersji aplikacji w celi wykonania testów i jej wdrożenia

2.2.1.4 Automatyczne monitorowanie

2.2.1.5 Dedykowane, osobne wersje aplikacji dla klientów firmy

Znacznie zmniejszy obciążenie aplikacji dla użytku wewnętrznego, zabezpieczy przed utrata możliwości składania zamówień, itd.

2.2.2 Klon obecnego środowiska

Z uwagi na brak jakiejkolwiek dokumentacji technicznej, testów automatycznych aplikacji oraz ogromny dług techniczny nagromadzony na przestrzeni ponad 5 lat, uruchomienie aplikacji w nowym środowisku (tj. nowe wersje języka PHP, nowa wersja bazy danych, nowe wersje bibliotek zewnętrznych, itd.) natychmiast po wdrożeniu nowego sprzętu komputerowego nie jest możliwe.

Wykonanie klonu obecnego środowiska umożliwi uruchomienie aplikacji na nowym sprzęcie natychmiast bez konieczności jakiejkolwiek ingerencji w kod aplikacji.

2.2.3 Nowe środowisko

- Na przestrzeni dwóch dni ([redacted]do[redacted] listopada 2019 r.) aplikacje udało się uruchomić w nowym środowisku innym – tj. z wykorzystaniem najnowszej wersji PHP, najnowszej wersji MySQL, najnowszej wersji Apache i aktualnej wersji systemu operacyjnego Linux ([redacted])
- Pełne uruchomienie aplikacji w nowym środowisku – tak aby wszystkie funkcjonalności i widoki użytkownika działały – wymaga jednak jeszcze sporej ingerencji w kod aplikacji i naprawę dużej ilości błędów.

2.3 System kontroli wersji

Chcąc wprowadzać jakiejkolwiek zmiany w kodzie aplikacji konieczne jest zabezpieczenie jakie daje system kontroli wersji, tj. łatwość cofnięcia jakichkolwiek zmian, jak również możliwość tworzenia nowych / naprawionych wersji wybranych funkcjonalności z możliwością „połączenia” ich z kompletnym kodem aplikacji.

2.4 API oraz integracja z zewnętrznymi aplikacjami i systemami

Z uwagi na brak jakiejkolwiek dokumentacji technicznej, testów automatycznych aplikacji oraz ogromny dług techniczny nagromadzony na przestrzeni ponad 5 lat, dalszy rozwój aplikacji wymaga najpierw spłacenia przynajmniej części długu technicznego.

Nie chcąc jednak opóźniać wdrożenia najważniejszych dla firmy funkcjonalności, jednocześnie chcąc ograniczyć do minimum ingerencje w istniejący kod, pierwszym krokiem po uruchomieniu aplikacji w nowym środowisku powinna być praca nad API oraz integracja z zewnętrznymi aplikacjami i systemami z wykorzystaniem tego API.

Praca nad API będzie obejmować tworzenie zupełnie nowego kodu, w języku PHP czy innym – który z obecną wersją aplikacji łączyć będzie jedynie baza danych, dzięki temu ograniczy potrzebę ingerencji w obecny kod aplikacji – następnie integracja z zewnętrznymi aplikacjami i systemami z wykorzystaniem tego API (np. program do księgowości w celu wykonania zestawienia miesięcznego, sklepy internetowe, itd.).

Sugerowane jest stopniowe wydzielanie funkcjonalności i integracja z zewnętrznym oprogramowaniem, tj. tak aby aplikacja skupiała się tylko i wyłącznie na tym czego np. oprogramowanie do księgowości, CRM, itd. nie są w stanie wykonać lepiej, i integracja z tymi programami. Aplikacja rozrosła się i obejmuje wiele funkcji, które dużo lepiej można wykonać przez dedykowane oprogramowanie (księgowość, CRM, itd. itp.), aplikacja wtedy skupia się tylko na tym co jest unikalne dla [redacted].

2.5 Splata długu technicznego

2.6 Naprawa błędów

2.7 Dalszy rozwój i komercjalizacja aplikacji

W tym wszelkie prace potrzebne w celu skomercjalizowania aplikacji / dostosowania do odsprzedawania / udostępniania w modelu SaaS dla innych firm działających w tej samej branży co [redacted]