Plan zarządzania konfiguracją

Projekt: Aplikacja do wypożyczania samochodów

Technologie: C#, XAML (WPF)

Repozytorium: Git (np. GitHub)

# 1. Zakres dokumentu

Niniejszy dokument opisuje zasady zarządzania konfiguracją w projekcie aplikacji okienkowej do wypożyczania samochodów. Dokument obejmuje identyfikację, kontrolę, rejestrację oraz audyt elementów konfiguracji na każdym etapie cyklu życia projektu.

# 2. Referencje

• PN-EN ISO/IEC 12207  
• Materiały z przedmiotu "Inżynieria Oprogramowania"  
• Wewnętrzna dokumentacja projektu (pliki README, changelog, komentarze commitów)

# 3. Terminologia dokumentu

• CI (Configuration Item) – element konfiguracji (np. folder, plik, biblioteka)  
• Baseline – uzgodniony, oficjalny zbiór elementów konfiguracji na danym etapie  
• CSCI – element konfiguracji oprogramowania  
• CCB – Change Control Board (osoby zatwierdzające zmiany)  
• Repozytorium – zdalna lub lokalna baza przechowująca wersje projektu (np. GitHub)

# 4. System zarządzania konfiguracją

- W projekcie wykorzystano system kontroli wersji Git.  
- Repozytorium główne przechowuje kod źródłowy, pliki konfiguracyjne, dokumentację oraz zależności.  
- Zarządzanie zmianami odbywa się przez branchowanie (np. main, dev, gałęzie funkcjonalne), pull requesty i code review.  
- Każda zmiana jest opatrzona komentarzem i automatycznie wersjonowana.

# 5. Proces zarządzania konfiguracją

## 5.1 Identyfikacja konfiguracji (CI)

### 5.1.1 Struktura produktu:

Projekt składa się z kilku głównych modułów:  
- CarRental – aplikacja kliencka  
- CarRentalAPI – warstwa komunikacji z bazą (jeśli występuje)  
- Models, ViewModels, Views – kod źródłowy (logika, prezentacja, dane)  
- System/Configuration – pliki konfiguracyjne  
- Commands, Images – logika sterująca i zasoby graficzne  
- Pliki rozwiązania: .sln, .csproj, packages.config, App.config, App.xaml

### 5.1.2 Wybór elementów konfiguracji (CSCI):

Elementy konfiguracji to:  
- Wszystkie foldery z kodem (Models, ViewModels, Views, Commands, System/Configuration)  
- Pliki konfiguracyjne i projektu (App.config, App.xaml, CarRental.csproj, packages.config)  
- Skrypty wdrożeniowe, dokumentacja techniczna  
- Biblioteki i zależności (packages)

### 5.1.3 Dokumentowanie elementów konfiguracji:

Każdy CI posiada unikalną nazwę, ścieżkę oraz opis (np. w README.md lub komentarzu do commita).

### 5.1.4 Wersje:

Zmiany wersjonowane przez Gita. Każda wersja (release) oznaczona tagiem zgodnie z konwencją semver (np. v1.0.0).

### 5.1.5 Definicja zbioru baseline:

Baseline ustalany po każdej zakończonej funkcjonalności lub zakończonym sprincie – tag w repozytorium oraz archiwizacja wersji.

## 5.2 Kontrola konfiguracji

- Każda zmiana kodu lub plików jest weryfikowana przez code review (pull request) i zatwierdzana przez lidera lub innego członka zespołu (CCB).  
- Zmiany są szczegółowo opisywane w komentarzach commitów.

## 5.3 Rejestracja stanu konfiguracji

- Stan konfiguracji rejestrowany automatycznie przez Git – historia commitów, tagi, changelog.  
- Ważniejsze zmiany wpisywane do pliku CHANGELOG.md.

## 5.4 Audyt konfiguracji

### 5.4.1 Audyt funkcjonalny:

Sprawdzenie, czy wersja systemu działa zgodnie z wymaganiami i dokumentacją (testy funkcjonalne).

### 5.4.2 Audyt fizyczny:

Weryfikacja kompletności wszystkich plików projektu (brakujące biblioteki, poprawność zależności w packages.config).

# 6. Organizacja procesu zarządzania konfiguracją

- Lider zespołu odpowiada za zgodność z procedurami oraz zarządza repozytorium.  
- Każdy członek zespołu odpowiada za opisywanie swoich commitów i przestrzeganie ustalonych standardów.

# 7. Procedury zarządzania konfiguracją

## 7.1 Procedura identyfikacji konfiguracji:

Nowe elementy i zmiany są oznaczane w repozytorium oraz dokumentowane.

## 7.2 CCB:

W skład CCB wchodzi lider oraz wyznaczeni członkowie zespołu.  
Każda większa zmiana (np. architektoniczna) wymaga zatwierdzenia przez CCB.

## 7.3 Procedura kontroli konfiguracji:

Wszelkie zmiany wprowadzane przez pull requesty.  
Code review przed merge’m do głównej gałęzi.

## 7.4 Procedura rejestracji stanu konfiguracji:

Wszystkie zmiany rejestrowane przez system Git i opisane w changelogu.

## 7.5 Procedura audytu konfiguracji:

Audyt wykonywany po zakończeniu każdej dużej funkcjonalności lub przed wydaniem release.

## 7.6 Harmonogram zarządzania konfiguracją:

Baseline po zakończeniu każdej większej części projektu (np. co sprint lub po zakończeniu funkcjonalności).  
Audyt co najmniej raz przed wydaniem wersji końcowej.

# 8. Kontrola procesu zarządzania konfiguracją

Regularne spotkania zespołu i monitorowanie zgodności z procedurami.  
Przegląd statusu zmian i rozwiązywanie potencjalnych problemów na bieżąco.