O'REILLY®

Wydanie II

# WordPress

## Tworzenie aplikacji internetowych

Brian Messenlehner
Jason Coleman

Helion

Tytuł oryginału: Building Web Apps with WordPress: WordPress as an Application Framework, 2nd Edition

Tłumaczenie: Agnieszka Górczyńska

ISBN: 978-83-283-6925-2



Helion SA
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *http://helion.pl* (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
*http://helion.pl/user/opinie/wordp2*
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

# Spis treści

ROZDZIAŁ 1.

# Tworzenie aplikacji internetowych w WordPressie

Ta książka pomoże Ci utworzyć w WordPressie witrynę internetową, motyw, wtyczkę, usługę sieciową i aplikację internetową. Zdecydowaliśmy się skoncentrować na aplikacjach internetowych, ponieważ można je postrzegać jako superwitryny internetowe pozwalające wykorzystywać wszystkie techniki przedstawione w tej książce.

Wiele osób jest przekonanych, że WordPress nie ma wystarczających możliwości do tworzenia aplikacji internetowych lub nie został opracowany w tym celu. Do tego tematu jeszcze powrócimy w dalszej części rozdziału. Od wielu lat tworzymy aplikacje internetowe WordPressa i wiemy, że doskonale nadaje się on do budowania skalowalnych aplikacji.

W tym rozdziale zaczniemy od zdefiniowania aplikacji internetowej, a następnie wyjaśnimy, dlaczego WordPress to doskonały framework przeznaczony do jej tworzenia. Zaprezentujemy również kilka wybranych sytuacji, w których użycie WordPressa *nie będzie* najlepszym rozwiązaniem podczas tworzenia aplikacji internetowej.

## Czym jest witryna internetowa?

Doskonale wiesz, czym jest witryna internetowa: to jedna lub więcej stron internetowych zawierających informacje i dostępnych za pomocą przeglądarki WWW.

## Czym jest aplikacja?

Lubimy definicję zamieszczoną w Wikipedii (*en.wikipedia.org/wiki/Application_software*): „aplikacja to oprogramowanie zaprojektowane do wykonywania grupy powiązanych ze sobą funkcji, zadań lub czynności, które oferują pewną korzyść użytkownikowi”.

## Czym jest aplikacja internetowa?

Aplikacja internetowa to po prostu *aplikacja uruchomiona w przeglądarce WWW*.

Zwróć uwagę na to, że w przypadku niektórych aplikacji internetowych technologia przeglądarki WWW pozostaje ukryta — np. podczas integracji aplikacji internetowej z natywną aplikacją systemu iOS lub Android, podczas uruchamiania witryny internetowej jako aplikacji w Google Chrome, a także podczas uruchamiania aplikacji w środowisku Adobe AIR. Jednak wewnątrz tych aplikacji nadal znajduje się system przetwarzający dane w HTML, CSS i JavaScripcie.

Aplikację internetową można traktować także jako *witrynę internetową wraz z dodatkami przypominającymi aplikację*. Nie istnieje wyraźna granica, po której przekroczeniu witryna internetowa staje się aplikacją internetową. To jest przykład jednej z tych sytuacji, w których zobaczywszy witrynę internetową, od razu będziesz wiedzieć, że masz do czynienia z aplikacją internetową.

Natomiast *można* wyjaśnić pewne funkcje aplikacji internetowej, przedstawić przykłady i spróbować zaprezentować skróconą definicję, aby Czytelnik ogólnie wiedział, co mamy na myśli, gdy w tekście książki stosujemy wyrażenie „aplikacja internetowa”.

Podczas lektury książki będziesz napotykać odwołania do SchoolPressa. Jest to aplikacja internetowa, którą budujemy, aby pomóc szkołom i nauczycielom w zarządzaniu uczniami i programem nauczania. Wszystkie przykładowe fragmenty kodu zostały przygotowane z myślą o funkcjonalności, która może istnieć w SchoolPressie. Więcej informacji na temat ogólnych koncepcji stojących za aplikacją SchoolPress znajdziesz w dalszej części rozdziału.

## Funkcje aplikacji internetowej

W tej sekcji wymieniamy typowe funkcje powiązane z aplikacjami internetowymi i ogólnie z aplikacjami. Im więcej tych funkcji znajdziesz w witrynie internetowej, tym bardziej odpowiednie jest nazywanie jej aplikacją internetową[1].

*Elementy interaktywne*

Typowa witryna internetowa oznacza poruszanie się między tworzącymi ją stronami, wczytywanie tych stron, przewijanie treści i klikanie łączy. Aplikacja internetowa również może mieć łącza i wymagać przewijania, ale zwykle wykorzystuje inne metody umożliwiające poruszanie się po aplikacji.

Witryny internetowe wraz z formularzami oferują możliwość obsługi transakcji. Przykładem może być formularz kontaktowy w witrynie internetowej lub formularz aplikacji na stronie zawierającej oferty pracy w danej firmie. Formularze pozwalają użytkownikom pracować z witryną za pomocą czegoś więcej niż tylko kliknięć.

Aplikacje internetowe oferują jeszcze bardziej interaktywne elementy interfejsu użytkownika. Do przykładów zaliczamy paski narzędzi, elementy typu „przeciągnij i upuść”, edytory bogatego tekstu i paski przewijania.

[1] Wiele idei przedstawionych w tej sekcji zostało zainspirowanych następującymi postami z blogów: „What is a Web Application?” napisany przez Dominique Hazae'l-Massieux i opublikowany na stronie *http://people.w3.org/~dom/archives/2010/08/what-is-a-web-application/* oraz „What is a Web Application?” napisany przez Boba Baxleya i opublikowany na stronie *http://boxesandarrows.com/what-is-a-web-application/*.

*Zadania zamiast treści*

Pamiętaj, że aplikacje internetowe są „zaprojektowane, aby pomóc użytkownikowi w wykonywaniu określonych zadań". Mapy Google dostarczają użytkownikom wskazówki ułatwiające dotarcie pod wskazany adres. Użytkownicy usługi Gmail tworzą wiadomości e-mail. Użytkownicy serwisu Trello zarządzają listami. Użytkownicy serwisu SchoolPress komentują zdarzenia zachodzące w klasie.

Część aplikacji nadal koncentruje się na treści. Typowa sesja w aplikacji Facebook lub Twitter obejmuje 90% czasu czytania. Jednak same aplikacje zapewniają możliwość przeglądania treści w inny sposób niż w przypadku tradycyjnej witryny internetowej.

*Loginy*

Loginy i konta pozwalają aplikacji internetowej zapisywać informacje o użytkownikach. Te informacje są później używane podczas wykonywania głównych zadań aplikacji i dla zapewnienia trwałego przechowywania danych. Po zalogowaniu się użytkownik serwisu SchoolPress widzi, których tematów nie przeczytał. Otrzymuje również nazwę użytkownika określającą jego aktywność w aplikacji.

Aplikacja internetowa może też mieć grupy użytkowników. W przypadku aplikacji SchoolPress istnieją administratorzy nadzorujący wewnętrzny sposób działania aplikacji, nauczyciele tworzący klasy oraz uczniowie biorący udział w dyskusjach prowadzonych w klasach.

*Wykorzystanie możliwości urządzeń*

Aplikacja internetowa uruchomiona w smartfonie może uzyskać dostęp do aparatu fotograficznego, książki adresowej, wewnętrznego magazynu danych, dostarczanych przez GPS informacji o położeniu itd. Z kolei aplikacja internetowa uruchomiona w tradycyjnym komputerze może mieć dostęp do kamery internetowej i dysku twardego. Ta sama aplikacja internetowa może reagować odmiennie w zależności od urządzenia, w którym została uruchomiona. Aplikacja internetowa dostosuje się do wielkości ekranu, rozdzielczości i możliwości urządzenia.

*Praca w trybie offline*

Gdy jest to możliwe, dobrym rozwiązaniem jest zapewnienie aplikacji możliwości działania w trybie offline. Oczywiście interaktywność internetu jest tym, co definiuje „internetową" część aplikacji. Jednak witryna internetowa, działająca, gdy jedziesz samochodem w tunelu, będzie przypominała bardziej aplikację.

W przypadku serwisu Gmail wiadomość e-mail można tworzyć w trybie offline. Evernote pozwala na tworzenie notatek w trybie offline, a następnie ich synchronizację z internetem po przywróceniu połączenia.

*Mashupy*

Istnieje możliwość połączenia jednej lub więcej aplikacji internetowych. Aplikacja internetowa może wykorzystywać różne usługi sieciowe i API w celu przekazywania i pobierania danych. Możesz mieć aplikację internetową pobierającą dane o położeniu, np. długość i szerokość geograficzną, z serwisu Twitter lub Foursquare i przekazującą je do serwisu Mapy Google.

## Aplikacje mobilne

Wydanie pierwsze tej książki zostało opublikowane w 2012 roku — od tamtej chwili jesteśmy świadkami gwałtownego rozwoju aplikacji internetowych, w szczególności aplikacji mobilnych. W większości witryn internetowych urządzenia mobilne wyprzedziły tradycyjne komputery i generują najwięcej ruchu sieciowego (źródło: Perficient Inc., *https://www.perficientdigital.com/insights/our-research/mobile-vs-desktop-usage-study*).

W 2012 roku typowa aplikacja internetowa wyglądała jak Basecamp, oprogramowanie menedżera projektu dostępne w komputerze za pomocą przeglądarki WWW. Natomiast w 2019 roku typowa aplikacja internetowa wygląda jak Twitter, aplikacja komunikacyjna dostępna za pomocą smartfona działającego pod kontrolą systemu iOS lub Android.

Ponieważ w większości przypadków duża część użytkowników uzyskuje dostęp do witryny internetowej i aplikacji za pomocą urządzeń mobilnych, podczas projektowania i tworzenia aplikacji internetowych stosowane jest podejście „najpierw wersja mobilna". Temat natywnego uruchamiania aplikacji WordPressa zostanie omówiony w rozdziale 16. Natomiast podstawy projektu responsywnego i przygotowanie witryny internetowej do prawidłowego wyświetlania na ekranie o każdej wielkości dokładniej omówimy w rozdziale 4.

## Progresywne aplikacje internetowe

*Progresywne aplikacje internetowe* (ang. *progressive web apps*, PWA) to witryny internetowe, które wykorzystują zalety funkcji nowoczesnych przeglądarek WWW umożliwiające działanie tych witryn jak natywnych aplikacji w systemach iOS, Android oraz w tradycyjnych komputerach. W szczególności witryny internetowe używające tzw. *usług roboczych* do funkcjonowania w trybie offline mają plik *manifestu aplikacji internetowej*, który pozwala zdefiniować aplikację w systemie operacyjnym oraz spełnić kilka innych wymagań, aby możliwe było uruchamianie takiej aplikacji bezpośrednio z przeglądarki WWW.

Orędownikiem progresywnych aplikacji internetowych był zespół Google Chrome, a teraz te aplikacje są obsługiwane także w systemie iOS oraz w większości nowoczesnych przeglądarek WWW. Dostępna jest wtyczka (*https://wordpress.org/plugins/pwa/*) oferująca obsługę podstawowych funkcji PWA w WordPressie. Dzięki tej wtyczce można witrynę internetową WordPressa zmienić w aplikację PWA. Wprawdzie to jest dobry pomysł, ale w rzeczywistości tworzenie kodu PWA jest bardziej związane z nastawieniem niż z faktyczną konwersją. Podobnie jak w przypadku przedstawionych wcześniej „funkcji aplikacji internetowej" strona główna PWA w serwisie Google (*https://web.dev/progressive-web-apps/*) zawiera listę m.in. następujących funkcji oczekiwanych od większości aplikacji typu PWA (*https://web.dev/pwa-checklist/*):

- witryna korzysta z protokołu HTTPS;
- strony są responsywne w tabletach i innych urządzeniach mobilnych;
- wszystkie adresy URL aplikacji działają po jej przejściu do trybu offline;
- dostarczone są metadane pozwalające dodać witrynę do ekranu głównego urządzenia;
- pierwsze wczytanie witryny jest szybkie nawet w przypadku połączenia 3G;

- witryna działa w każdej przeglądarce WWW;
- przejścia między stronami nie blokują sieci;
- każda strona ma adres URL.

Poza wymienionymi tutaj podstawowymi funkcjami istnieje także lista rzeczy do sprawdzenia dla „wzorowych" aplikacji typu PWA obejmujących UX i wydajność działania. Narzędzie Lighthouse firmy Google (*https://developers.google.com/web/tools/lighthouse/*) oferuje testy zautomatyzowane i raporty dotyczące spełniania kryteriów PWA. Nawet aplikacje w pełni natywne lub utworzone dla przeglądarki WWW mogą czerpać korzyści z pewnych sugestii zamieszczonych na liście rzeczy do sprawdzenia PWA oraz w raportach Lighthouse.

# Dlaczego WordPress?

Żaden język programowania lub narzędzie nie będzie odpowiednim wyborem do wykonania każdego zadania. Wkrótce się dowiesz, dlaczego *nie* należy wybierać WordPressa, teraz natomiast chcemy się skoncentrować na sytuacjach, w których wykorzystanie WordPressa do utworzenia aplikacji internetowej *jest* dobrym rozwiązaniem.

## Jesteś już użytkownikiem WordPressa

Jeżeli już używasz WordPressa do udostępnienia głównej witryny internetowej, dodanie niezbędnej funkcjonalności będzie wymagało niewiele pracy. WordPress oferuje ogromny wybór wtyczek dla e-commerce (WooCommerce), forów (bbPress), witryn obsługi użytkowników (Paid Memberships Pro), funkcjonalności związanej z serwisami społecznościowymi (BuddyPress) i gamifikacją (BadgeOS).

Budowa aplikacji internetowej na podstawie istniejącej witryny internetowej WordPressa pozwoli na dużą oszczędność czasu i ułatwi pracę jej użytkownikom. Dlatego jeśli aplikacja jest dość prosta, można utworzyć dla witryny WordPressa własną wtyczkę zawierającą funkcjonalność niezbędną dla aplikacji internetowej.

Jeżeli jesteś zadowolony z użycia WordPressa w istniejącej witrynie, nie martw się, słysząc, że musisz uaktualnić coś jeszcze, aby dodać określoną funkcjonalność do witryny. Prawdopodobnie to nieprawda. Nie musisz pozbywać się wykonanej dotąd pracy w WordPressie, a wcześniej zdobyte doświadczenie to doskonały powód, by pozostać przy wyborze tej właśnie platformy.

## Zarządzanie treścią w WordPressie jest łatwe

Na początku WordPress opracowano jako platformę przeznaczoną do publikacji blogów. Na przestrzeni lat ewoluował i wraz z wprowadzeniem niestandardowych typów postów (ang. *custom post types*, CPT) w wydaniu 3.0 WordPress zyskał w pełni funkcjonalny system zarządzania treścią (ang. *content management system*, CMS). Każda strona lub post mogą być edytowane przez administratorów za pomocą panelu głównego (ang. *dashboard*), do którego dostęp odbywa się za pomocą przeglądarki WWW. Więcej informacji na temat pracy z CPT znajdziesz w rozdziale 5.

WordPress ułatwia dodawanie i edytowanie treści dzięki edytorowi działającemu w trybie WYSIWYG (ang. *what you see is what you get*), więc nie musisz się zwracać do projektanta internetowego za każdym razem, gdy chcesz wprowadzić prostą zmianę w witrynie internetowej. Ponadto możesz tworzyć własne menu i elementy nawigacyjne dla witryny nawet bez tworzenia jakiegokolwiek kodu.

Jeżeli aplikacja internetowa koncentruje się na treści (np. podstawowe funkcje aplikacji SchoolPress to zlecanie zadań i dyskusje), to omówione w rozdziale 5. API CPT dla WordPressa pozwala na łatwe i szybkie przygotowanie treści dla użytkowników i zarządzanie nią.

Nawet w przypadku aplikacji bardziej ukierunkowanych na zadania zwykle istnieje kilka stron zawierających informacje, dokumentację i inne dane, np. związane ze sprzedażą. Użycie WordPressa do utworzenia aplikacji internetowej oznacza, że aplikacją i jej całą treścią możesz zarządzać w jednym miejscu.

## Łatwe i bezpieczne zarządzanie użytkownikami w WordPressie

WordPress zawiera wszystko to, co jest potrzebne w celu dodawania użytkowników administracyjnych i końcowych do witryny internetowej.

Poza kontrolą dostępu do treści system ról i możliwości w WordPressie jest rozszerzalny oraz pozwala kontrolować, które *akcje* są dostępne dla poszczególnych grup użytkowników. Na przykład domyślnie użytkownik w roli o nazwie *contributor* może *dodawać* nowe posty, ale nie może ich *publikować*. Podobnie można tworzyć nowe role i definiować nowe możliwości, aby w ten sposób zarządzać dostępem do własnego systemu funkcjonalności.

Wtyczki takie jak Paid Membership Pro rozszerzają wbudowane możliwości zarządzania użytkownikami i pozwalają na desygnowanie członków na różnych poziomach i kontrolowanie treści, do której użytkownicy mają dostęp. Przykładowo można utworzyć poziom zapewniający dostęp do treści dodatkowej witryny WordPressa tym użytkownikom, którzy wykupili taką możliwość.

## Wtyczki

W repozytorium WordPressa istnieje ponad 55 000 bezpłatnie dostępnych wtyczek (*https://wordpress.org/plugins/*). Znacznie większa liczba wtyczek, zarówno płatnych, jak i bezpłatnych, jest dostępna w innych witrynach internetowych. Gdy wpadniesz na pomysł, by utworzyć rozszerzenie dla witryny internetowej, istnieje duże prawdopodobieństwo, że taka wtyczka została już wcześniej opracowana, co pozwala zaoszczędzić czas i pieniądze.

W repozytorium znajdziesz wiele niezastąpionych wtyczek, które ostatecznie znajdą zastosowanie w każdej tworzonej przez Ciebie witrynie internetowej i aplikacji internetowej.

W praktycznie każdej tworzonej witrynie internetowej będziesz chciał m.in. zastosować buforowanie danych wyjściowych (co pozwala przyspieszyć operacje przeglądania stron), wykorzystać narzędzia takie jak Google Analytics do śledzenia odwiedzających, utworzyć mapę witryny, dostosować ustawienia strony w taki sposób, aby stała się przyjazna technikom optymalizacji dla wyszukiwarek internetowych (ang. *search engine optimization*, SEO).

Istnieje wiele doskonałych wtyczek przeznaczonych do wykonywania wymienionych funkcji. W książce przedstawimy nasze ulubione, a pełną ich listę znajdziesz w witrynie poświęconej książce (*https://web.archive.org/web/20191103020221/https://bwawwp.com/plugins/*).

## Elastyczność ma duże znaczenie

WordPress to w pełni wyposażony framework oferujący naprawdę potężne możliwości. Został zbudowany w oparciu o technologie PHP, JavaScript i MySQL, więc to, co można utworzyć za pomocą PHP i MySQL (czyli praktycznie wszystko), można także dość łatwo zrobić w aplikacji WordPressa.

Ogólnie rzecz biorąc, WordPress oraz PHP i MySQL to nie jest doskonałe rozwiązanie dla każdego zadania, choć to połączenie sprawdza się w wielu różnych przypadkach. Wykorzystanie jednej platformy rozwijającej się wraz z firmą pozwala na szybsze działanie i dostosowywanie się do zmian. Oto typowy proces rozwoju i rozbudowy witryny w przypadku witryny internetowej startupu zbudowanej w WordPressie:

1. Zaprezentowanie startupu za pomocą witryny internetowej składającej się z jednej strony.
2. Dodanie formularza umożliwiającego gromadzenie adresów e-mail.
3. Dodanie bloga.
4. Skoncentrowanie się na technikach SEO i optymalizacji całej treści.
5. Przekazywanie postów bloga do serwisów Twitter i Facebook.
6. Dodanie forum.
7. Użycie wtyczki Paid Membership Pro, aby umożliwić użytkownikom wnoszenie opłat za dostęp do treści witryny internetowej.
8. Dodanie niestandardowych formularzy, narzędzi i funkcjonalności aplikacji dla użytkowników płacących za dostęp do treści.
9. Uaktualnienie interfejsu użytkownika za pomocą frameworków i technik JavaScriptu.
10. Dostosowanie witryny internetowej i serwera do skalowania.
11. Lokalizacja witryny internetowej i aplikacji dla różnych krajów i języków.
12. Dodanie obsługi progresywnej aplikacji internetowej.
13. Przygotowanie opakowań iOS i Android dla aplikacji internetowej.

Pozytywnym aspektem przejścia przez ten proces jest to, że na każdym etapie masz do dyspozycji *tę samą bazę danych użytkowników* i korzystasz z *tej samej platformy programistycznej*.

## Częste uaktualnienia zabezpieczeń

Fakt wykorzystywania WordPressa przez witryny internetowe powoduje, że stanowi on cel dla hakerów próbujących złamać jego zabezpieczenia. W przeszłości części hakerów to się udało, na szczęście programiści tworzący WordPressa dość szybko reagują na znalezione luki w zabezpieczeniach i wydają usuwające je poprawki. Można to porównać do sytuacji, w której miliony osób nieustannie testują i poprawiają oprogramowanie, ponieważ dokładnie z tym mamy do czynienia.

Architektura stojąca za WordPressem powoduje, że stosowanie uaktualnień to proces szybki i bezproblemowy, nawet dla użytkowników dopiero rozpoczynających pracę z tą platformą. Jeżeli potrafisz zainstalować i uaktualnić WordPress do najnowszej dostępnej wersji, wówczas dla witryny internetowej masz do dyspozycji znacznie bezpieczniejszą platformę niż jakakolwiek inna aktualnie dostępna. Tematem zapewnienia bezpieczeństwa zajmiemy się w rozdziale 8.

## Koszt

WordPress jest bezpłatny, PHP jest bezpłatny, MySQL jest bezpłatny i większość wtyczek WordPressa jest bezpłatnych.

Wprawdzie serwery i hosting wymagają opłat, ale w zależności od wielkości aplikacji internetowej i natężenia generowanego przez nią ruchu sieciowego ten koszt może być względnie niewielki. Jeżeli wymagasz funkcjonalności niedostępnej w żadnej istniejącej wtyczce, wtedy będzie trzeba zatrudnić programistę i zlecić mu utworzenie odpowiedniej wtyczki. Jeżeli sam jesteś programistą, nie musisz ponosić tego kosztu i wystarczy, że poświęcisz trochę czasu na opracowanie wtyczki.

## Odpowiedź na często pojawiającą się krytykę wybranych aspektów WordPressa

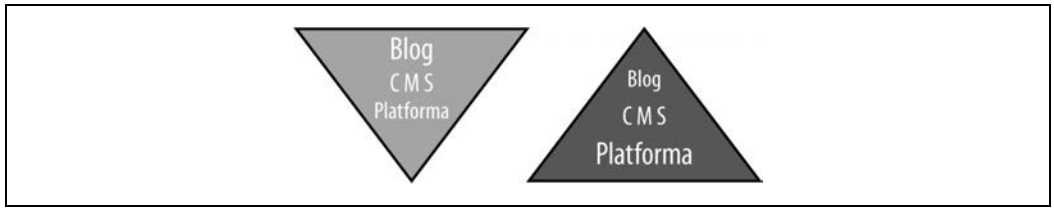
Dość często można spotkać się z opinią, że WordPress nie jest dobrym frameworkiem do tworzenia aplikacji internetowych lub nawet że w ogóle nie jest frameworkiem. Szanując prawo innych do wypowiedzi, w tym miejscu chcielibyśmy wyjaśnić, dlaczego nie zgadzamy się z takimi opiniami. W kolejnych punktach odniesiemy się do często pojawiającej się krytyki.

### WordPress jest przeznaczony tylko do tworzenia blogów

Spora grupa osób jest przekonana, że jeśli WordPress powstał w celu tworzenia blogów, to sprawdza się dobrze tylko w tym zakresie.

Takie stwierdzenia mogły być prawdziwe kilka lat temu. Jednak ostatnio w WordPressie zaimplementowano oferującą dość potężne możliwości funkcjonalność CMS, więc platforma stała się użyteczna także dla innych witryn internetowych skoncentrowanych na treści. Obecnie WordPress to najpopularniejszy system zarządzania treścią, a jego udział w rynku szacuje się na 60%[2]. Na rysunku 1.1 możesz zobaczyć slajd z prezentacji „State of WordPress", przygotowanej przez Matta Mullenwega na konferencję WordCamp San Francisco 2013. Widoczny po lewej stronie trójkąt skierowany do dołu przedstawia WordPress z mniej więcej 2006 roku, gdy większość kodu tworzącego ten framework była przeznaczona do obsługi aplikacji bloga, a tylko niewielka część dla systemu CMS i kodu platformy. Z kolei widoczny po prawej stronie trójkąt przedstawia aktualny stan platformy WordPress, w której większość kodu jest związana z samą platformą wraz ze znajdującą się nad nią warstwą CMS, aplikacja bloga zaś działa na warstwie CMS. Obecnie WordPress to znacznie stabilniejsza platforma niż kilka lat temu.

[2] Na stronie W3Tech pod adresem *https://w3techs.com/technologies/overview/content_management* dostępne są aktualne dane dotyczące użycia różnych systemów zarządzania treścią.




*Rysunek 1.1. Wykresy przygotowane przez Matta Mullenwega w 2013 roku na potrzeby prezentacji „State of WordPress". WordPress nie zawsze był stabilny*

API CPT można wykorzystać do usprawnienia instalacji WordPressa i zapewnienia obsługi typów treści innych niż posty bloga lub strony internetowe. Więcej informacji na ten temat znajdziesz w rozdziale 5.

### WordPress jest przeznaczony tylko dla stron z treściami

Podobnie jak w przypadku stwierdzenia „tylko dla bloga", część osób twierdzi, że WordPress jest przeznaczony tylko dla witryn internetowych skoncentrowanych na treści.

Po pierwsze, jeżeli WordPress byłby przeznaczony tylko dla skoncentrowanych na treści witryn internetowych i aplikacji, to i tak oznacza ogromną liczbę aplikacji. Na ekranie głównym smartfona prawdopodobnie masz wiele aplikacji koncentrujących się na treści, np. Netflix, Twitter, Facebook, Reddit, Evernote. To są przykłady bardzo popularnych aplikacji opracowanych przez ogromne firmy. Oczywiście nie sugerujemy, że *powinny* one działać w oparciu o WordPress, natomiast *sugerujemy*, że aplikacje o podobnym sposobie działania *można* zbudować, wykorzystując do tego framework WordPress.

Po drugie, jak się dowiesz z lektury niniejszej książki, WordPress to doskonały framework przeznaczony także do tworzenia bardziej interaktywnych aplikacji internetowych. Podstawową funkcją pozwalającą na wykorzystanie WordPressa jako frameworka jest API wtyczek, które umożliwia sprawdzenie domyślnego sposobu działania WordPressa i jego zmianę. Nie tylko masz dostęp do tysięcy wtyczek znajdujących się w repozytorium WordPressa i na różnych stronach w internecie, ale również za pomocą API wtyczek możesz przygotować własne wtyczki przeznaczone do wykonywania wszelkich zadań, które są możliwe do zrealizowania z użyciem PHP i MySQL.

### WordPress się nie skaluje

Niektórzy będą wskazywać na domyślną instalację WordPressa uruchomioną w usłudze hostingu o małych możliwościach i zwracać uwagę na spowolnienie działania witryny internetowej lub jej awarie w przypadku dużego obciążenia i na tej podstawie wyciągać wniosek, że WordPress się nie skaluje.

Być może uśmiałbyś się, gdybyś usłyszał o możliwości zbudowania witryny typu Facebook za pomocą WordPressa.

W rzeczywistości wiele witryn internetowych notujących ogromny ruch sieciowy zbudowano w oparciu o WordPress. Witryna *wordpress.com* została zbudowana w dokładnie taki sam sposób jak każda inna korzystająca z WordPressa i jest jedną z najczęściej odwiedzanych witryn internetowych na świecie.

Jeżeli masz zamiar utworzyć aplikację internetową na miarę Facebooka, nie jest to odpowiednia książka dla Ciebie. Zapytaj CTO, jaki odsetek wielomiliardowego budżetu jest przeznaczony na potrzeby związane z utworzeniem aplikacji i zatrudnieniem inżynierów z firm Google i Amazon, którzy będą potrzebni do pracy nad tą niestandardową aplikacją internetową.

Wraz ze wzrostem poziomu użycia aplikacji internetowej konieczne będzie uaktualnianie i wymienianie poszczególnych komponentów, aby sprostać tej skali. Problemy związane ze skalowaniem WordPressa są dokładnie takie same jak podczas skalowania każdej innej aplikacji: buforowanie stron i danych, znacznie szybsza obsługa zapytań do bazy danych i poprawienie wydajności działania sieci. Ogromne witryny takie jak *wordpress.com*, TechCrunch i blogi New York Timesa są skalowane w WordPressie. Podobnie większość wniosków wyciągniętych ogólnie ze skalowania aplikacji PHP i MySQL ma zastosowanie dla WordPressa. Tematem skalowania zajmiemy się w rozdziale 14.

### WordPress nie zapewnia bezpieczeństwa

Podobnie jak w przypadku innych produktów typu open source także WordPress wiąże się z pewnymi kompromisami w zakresie zapewnienia bezpieczeństwa.

Z jednej strony ogromna popularność WordPressa oznacza, że będzie on celem hakerów szukających luk w zabezpieczeniach. Skoro kod źródłowy został udostępniony jako open source, potencjalne luki w zabezpieczeniach są łatwiejsze do znalezienia.

Z drugiej strony, ponieważ kod źródłowy WordPressa jest właśnie typu open source, dość szybko dowiesz się o znalezionych lukach w zabezpieczeniach, a ktoś inny prawdopodobnie usunie je za Ciebie.

Czujemy się znacznie bezpieczniej, wiedząc, że choć wiele osób próbuje wykorzystać luki w zabezpieczeniach, inne starają się zabezpieczyć WordPressa przed wykorzystaniem tych braków. Nie wierzymy w „zapewnienie bezpieczeństwa przez niejawność", chyba że jest ona stosowana jako dodatkowy środek. Zamiast tego wolimy, aby znalezione luki w zabezpieczeniach oprogramowania wychodziły na światło dzienne, niż pozostały nieujawnione aż do najgorszego z możliwych momentów.

W rozdziale 8. znacznie dokładniej zajmiemy się omówieniem kwestii zapewnienia bezpieczeństwa — m.in. przedstawimy listę najlepszych praktyk w zakresie zabezpieczania instalacji WordPressa, a także podpowiemy, jak tworzyć bezpieczniejszy kod.

### Wtyczki WordPressa są beznadziejne

API wtyczek w WordPressie oraz tysiące dostępnych wtyczek opracowanych z wykorzystaniem tego API to tajna broń i według nas podstawowy powód, dla którego WordPress zyskał tak ogromną popularność i odniósł duży sukces jako platforma dla witryn internetowych.

Można się spotkać ze stwierdzeniem typu „oczywiście, że są dostępne tysiące wtyczek, ale większość z nich jest kiepska". W porządku, część wtyczek na pewno *jest* kiepska.

Jednak istnieje naprawdę wiele wtyczek, które zdecydowanie są użyteczne — wśród nich znajduje się AppPresser opracowana przez jednego z autorów tej książki, Briana Messenlehnera. Jeżeli używasz WordPressa do zarządzania treścią pisaną lub sklepem internetowym, wówczas wtyczka AppPresser i platforma to najszybszy sposób na otrzymanie tej treści lub sklepu internetowego w aplikacji mobilnej.

Paid Memberships Pro opracowana przez drugiego autora książki, Jasona Colemana, również zalicza się do użytecznych wtyczek. Pozwala ona obsługiwać rachunki wystawiane użytkownikom witryny internetowej, dzięki czemu można się skoncentrować na podstawowej funkcjonalności aplikacji zamiast zmagać się z integracją witryny z bramkami płatności.

Wiele wtyczek wykonuje bardzo proste zadania (np. ukrycie paska administracyjnego dla użytkowników niebędących administratorami), działa zgodnie z oczekiwaniem i naprawdę trudno je uznać za kiepskie.

Motywy i wtyczki znajdujące się w repozytorium WordPressa są przez wolontariuszy sprawdzane pod kątem zapewnienia bezpieczeństwa i jakości kodu. Proces *Theme Review Team* (*https://make.wordpress.org/themes/handbook/review/required/*) jest coraz bardziej rygorystyczny i rozbudowany niż pozostałe. Projekt Tide (*https://make.wordpress.org/tide/*) ma na celu opracowanie testów zautomatyzowanych do sprawdzania repozytoriów wtyczek i motywów, co powinno zapewnić dostęp do wysokiej jakości wtyczek i uaktualnień, a jednocześnie pozwolić na szybsze wykrywanie problemów związanych z zachowaniem zgodności i zapewnieniem bezpieczeństwa.

Nawet słaba wtyczka może zostać poprawiona, utworzona na nowo lub stanowić punkt wyjścia do opracowania kolejnej, która będzie działała lepiej. Czasami okazuje się, że utworzenie wtyczki od nowa jest łatwiejsze niż poprawienie kiepskiej. Mimo to i tak jesteś w znacznie lepszej sytuacji, niż gdybyś musiał zupełnie od początku tworzyć wszystko to, co jest potrzebne do działania budowanej aplikacji internetowej.

Nikt nie zmusza Cię do stosowania wtyczek WordPressa bez ich wcześniejszego sprawdzenia. Jeżeli budujesz poważną aplikację internetową, kod wtyczki powinieneś dokładnie sprawdzić samodzielnie i poprawić go w taki sposób, aby spełniał Twoje standardy, a dopiero potem przejść do następnego zadania.

# Kiedy nie używać WordPressa?

WordPress na pewno nie jest rozwiązaniem dla każdej aplikacji. W tym podrozdziale przedstawimy kilka sytuacji, w których *nie należy* używać WordPressa do budowania aplikacji internetowej.

## Planujesz licencjonować lub sprzedawać technologię witryny internetowej

WordPress jest dostępny na licencji GPLv2, która nakłada pewne ograniczenia związane ze sposobem rozpowszechniania oprogramowania utworzonego za pomocą narzędzi na tej licencji. Przede wszystkim nie można ograniczać sposobów wykorzystania oprogramowania po jego sprzedaży lub dostarczeniu.

Wprawdzie to jest złożony temat, ale podstawowa idea polega na tym, że jeśli tylko sprzedajesz lub zapewniasz *dostęp* do aplikacji, nie musisz się przejmować GPLv2. Jeżeli jednak sprzedajesz lub rozpowszechniasz kod źródłowy aplikacji, wówczas licencja GPLv2 ma zastosowanie dla rozpowszechnianego kodu.

Przykładowo jeśli na własnym serwerze prowadzisz hosting aplikacji SchoolPress i sprzedajesz konta zapewniające dostęp do aplikacji, nie jest to uznawane za rozpowszechnianie i licencja GPLv2 nie ma tutaj znaczenia.

Jeżeli chcesz umożliwić instytucjom edukacyjnym instalowanie oprogramowania i uruchamianie go na własnych serwerach, wówczas musisz im udostępnić kod źródłowy. To jest uznawane za dystrybucję. Klient będzie mógł legalnie i bezpłatnie przekazać dalej ten kod źródłowy, nawet jeśli musiał zapłacić za oprogramowanie. Konieczne jest użycie licencji GPLv2, która nie zezwala na ograniczenie tego, co użytkownik może zrobić z kodem źródłowym po jego pobraniu.

## Inna platforma szybciej doprowadzi Cię do celu

Jeżeli masz zespół doświadczonych programistów języka Ruby, aplikację internetową powinieneś utworzyć właśnie w tym języku. Jeżeli istnieje platforma, framework lub paczka zawierająca 80% funkcji niezbędnych dla Twojej aplikacji internetowej, a WordPress nie oferuje niczego podobnego, wówczas prawdopodobnie powinieneś skorzystać z innej platformy.

## Elastyczność jest bez znaczenia

Jedną z najważniejszych zalet witryny internetowej opartej na WordPressie jest możliwość szybkiej zmiany jej poszczególnych komponentów, aby lepiej pasowała do Twoich potrzeb. Przykładowo jeśli „polubienia" na Facebooku przestaną być użyteczne, można odinstalować wtyczkę Facebook Connect i zainstalować wtyczkę zapewniającą obsługę serwisu Pinterest.

Ogólnie rzecz biorąc, uaktualnianie motywu lub zastępowanie wtyczek witryny WordPressa będzie odbywało się szybciej niż samodzielne opracowywanie danej funkcjonalności zupełnie od początku na innej platformie. Jeżeli jednak optymalizacja i wydajność działania mają dużo większe znaczenie niż możliwość szybkiego uaktualniania aplikacji, wtedy opracowanie aplikacji natywnej lub programowanie bezpośrednio w PHP będzie lepszym rozwiązaniem.

Jeżeli aplikacja będzie wykonywać *jedno proste zadanie*, powinna zostać utworzona na niższym poziomie. Przykładowo serwer licencji wtyczki Paid Membership Pro to w zasadzie pojedynczy plik JSON informacji dodatkowych oraz mały skrypt sprawdzający klucze licencji i dostarczający spakowane pliki. Jason zbudował serwer licencji bezpośrednio w kodzie PHP i w ogromnym stopniu wykorzystał buforowanie. Ten serwer działa na kosztującym 10 USD miesięcznie DigitalOcean Sroplet i obsługuje ponad 80 000 witryn internetowych wraz z działającą wtyczką Paid Membership Pro.

Podobnie jeśli masz zasoby dostępne dla Facebooka, możesz pozwolić sobie na samodzielne zbudowanie czegokolwiek, wykorzystanie własnych kompilatorów PHP do C, a także natywnych komponentów iOS w celu skrócenia o kilka milisekund czasu wczytywania witryny internetowej i aplikacji.

### Aplikacja musi działać w czasie rzeczywistym

Jedną z potencjalnych wad WordPressa, na której temat więcej dowiesz się z dalszej części książki, jest oparcie na typowej architekturze serwera WWW. W typowej konfiguracji WordPressa użytkownik przechodzi pod określony adres URL, co powoduje nawiązanie komunikacji z serwerem WWW (takim jak Apache) za pomocą protokołu HTTP, uruchomienie skryptu PHP generującego stronę, która następnie zostaje zwrócona użytkownikowi.

Jest wiele sposobów na poprawę wydajności działania tej architektury za pomocą technik buforowania i/lub optymalizacji konfiguracji serwera. Istnieje możliwość asynchronicznego działania WordPressa dzięki wywołaniom w technologii AJAX lub dostępu do bazy danych za pomocą alternatywnych klientów. Jeżeli jednak aplikacja musi działać w czasie rzeczywistym i być w pełni asynchroniczna (przykładem może być aplikacja czatu internetowego lub gra wieloosobowa), wówczas masz nasze błogosławieństwo i możesz dobrze się zastanowić, czy na pewno chcesz skorzystać z WordPressa.

Wielu programistów WordPressa, w tym także Matt Mullenweg — założyciel i przywódca duchowy tej platformy, zdaje sobie sprawę ze wspomnianego ograniczenia. Coraz więcej funkcjonalności jest przenoszonych do języka JavaScript pozwalającego na przeprowadzanie obliczeń w przeglądarce WWW oraz do frameworków takich jak React zapewniających wysoką interaktywność aplikacji internetowej. Nowy edytor Gutenberg, dodany w WordPressie 5.0, jest najlepszym przykładem tego trendu i wskazuje kierunek, w którym podąża platforma. Jednak na razie trzeba się zmagać z drogą pod górę, aby zapewnić asynchroniczne działanie WordPressa z wydajnością porównywalną do aplikacji natywnej, utworzonej całkowicie w Node.js lub za pomocą innych technologii przeznaczonych specjalnie dla aplikacji działających w czasie rzeczywistym.

## WordPress jako framework aplikacji

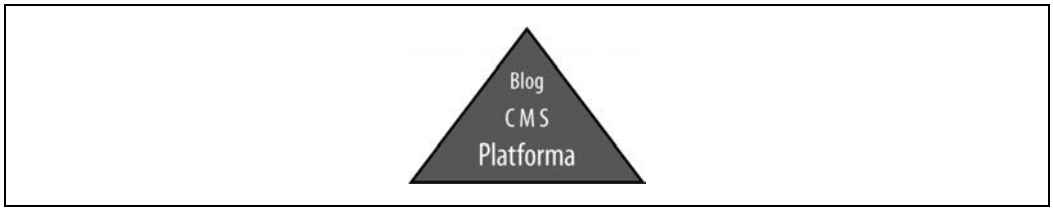
Systemy zarządzania treścią takie jak WordPress, Drupal i Joomla są często wykluczane z dyskusji związanych z frameworkami. Jednak w rzeczywistości WordPress (szczególnie) to doskonały przykład pokazujący, do czego są przeznaczone frameworki: do szybkiego tworzenia aplikacji.

W ciągu zaledwie kilku minut można skonfigurować WordPressa i otrzymać w pełni funkcjonalną aplikację wraz z obsługą użytkowników, zarządzaniem sesją, zarządzaniem treścią oraz panelem głównym przeznaczonym do monitorowania aktywności witryny internetowej.

Różne API, obiekty i funkcje pomocnicze omówione w książce umożliwiają bardzo szybkie tworzenie skomplikowanych aplikacji, bez konieczności zajmowania się kwestiami związanymi z integracją tej aplikacji z systemami działającymi na niższym poziomie.

Na rysunku 1.2 pokazaliśmy trójkąt pochodzący z przedstawionej w 2013 roku prezentacji „State of WordPress”. Na podstawie tego rysunku WordPress jawi się jako platforma wraz z umieszczoną na niej warstwą CMS, na której z kolei znajduje się aplikacja bloga.

Rzeczywistość jest taka, że większość obecnej bazy kodu obsługuje platformę aplikacji. Każde wydanie WordPressa należy traktować jako framework aplikacji połączony z przykładową aplikacją bloga.



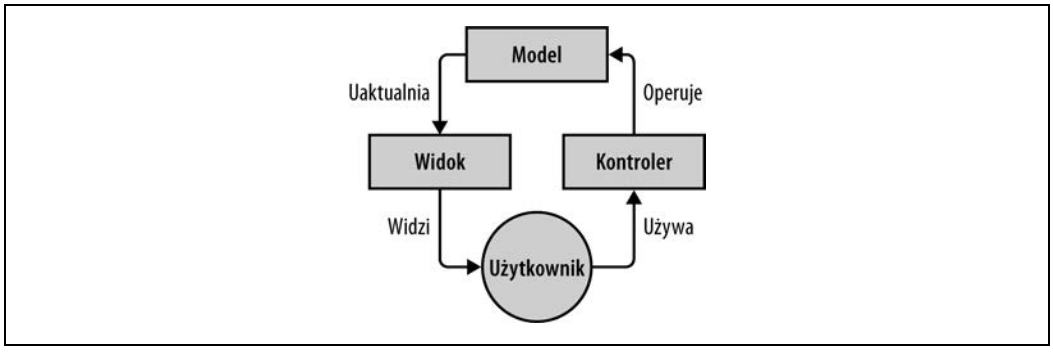

*Rysunek 1.2. Platforma WordPress*

## WordPress kontra frameworki MVC

Model – widok – kontroler (ang. *model-view-controller*, MVC) to wzorzec projektowy używany w wielu frameworkach przeznaczonych do tworzenia oprogramowania. Najważniejszą korzyścią wynikającą z użycia architektury MVC jest możliwość wielokrotnego użycia kodu i rozdzielenia zadań. WordPress nie stosuje architektury MVC, ale ma własny sposób zachęcający do wielokrotnego użycia kodu i separacji zadań.

Tutaj pokrótce wyjaśnimy architekturę MVC i sposób jej mapowania na proces programowania w WordPressie. Jeżeli masz doświadczenie w pracy z frameworkami MVC, to informacje przedstawione w tym punkcie powinny pomóc w zrozumieniu podejścia, które w WordPressie pozwoli programować w sposób podobny jak w przypadku MVC.

Na rysunku 1.3 pokazaliśmy typową aplikację zbudowaną na podstawie wzorca MVC. Użytkownik końcowy korzysta z *kontrolera* przeprowadzającego operacje wpływające na stan aplikacji za pomocą *modelu*, który następnie uaktualnia *widok* wyświetlany użytkownikowi. Przykładowo w aplikacji bloga użytkownik może patrzeć na stronę ostatnio opublikowanych postów (widok). Następnie może kliknąć tytuł posta, co spowoduje przejście pod nowy adres URL (kontroler), co z kolei oznacza wczytanie danych posta (model) i wyświetlenie pojedynczego posta (inny widok).




*Rysunek 1.3. Sposób działania architektury MVC*

Architektura MVC obsługuje wielokrotne użycie kodu i umożliwia współpracę modelom, widokom i kontrolerom. Przykładowo widok, zarówno ostatnio opublikowanych postów, jak i wyświetlający treść pojedynczego posta, może do wyświetlania danych posta używać tego samego modelu. Z kolei te same modele można wykorzystać we frontendzie do wyświetlania postów, w backendzie zaś do ich edycji. Architektura MVC obsługuje separację zadań przez umożliwienie projektantom koncentracji na widokach, a programistom — na modelach.

Możesz spróbować użyć architektury MVC w WordPressie. Istnieje wiele projektów, które mają w tym pomóc. Jednak uważamy, że próba stosowania architektury MVC w WordPressie może prowadzić do problemów, o ile oficjalna obsługa MVC nie zostanie wprowadzona w jądrze WordPressa. Dopóki tak się nie stanie, sugerujemy zastosowanie podejścia „zgodnego z WordPressem", które zostało omówione w tej książce.

Jeżeli mimo to wciąż jesteś zainteresowany podejściem MVC w WordPressie, zwróć uwagę na wtyczkę WP MVC (*https://wordpress.org/plugins/wp-mvc/*), która jest aktywnie rozwijana i pozwala stosować framework MVC podczas tworzenia wtyczek WordPressa. Jeżeli nie chcesz lub nie potrzebujesz pełnej architektury MVC, istnieje kilka sposobów mapowania procesu MVC na WordPressa.

### Modele = wtyczki

We frameworku MVC kod przechowujący struktury danych i logikę biznesową znajduje się w modelach. To na tworzenie modeli programiści poświęcają najwięcej czasu.

W WordPressie wtyczka to odpowiednie miejsce na przechowywanie nowych struktur danych, złożonej logiki biznesowej oraz niestandardowych typów postów.

Jednak takie porównanie jest nie najlepsze z kilku powodów. Po pierwsze, wiele wtyczek dodaje funkcjonalność przypominającą widok i zawiera elementy projektowe — rozważ np. wtyczkę dodającą widżet umieszczany na stronach internetowych. Po drugie, formularze i inne komponenty projektowe używane w panelu głównym WordPressa są, ogólnie rzecz biorąc, obsługiwane przez wtyczki.

Jednym ze sposobów na zapewnienie bardziej wyraźnie zarysowanej separacji zadań podczas dodawania przypominających widoki komponentów do wtyczek WordPressa jest utworzenie katalogu *templates* lub *pages* i umieszczenie w nim kodu frontendu. Często stosowana praktyka pozwala szablonom nadpisywać szablony wykorzystane przez wtyczkę. Przykładowo gdy używasz wtyczki Paid Membership Pro, katalog *paid-memberships-pro/pages* można umieścić w aktywnym motywie, aby w ten sposób nadpisać domyślne szablony stron. (Dokładniejsze omówienie tej techniki nadpisywania szablonów wtyczki znajdziesz w rozdziale 4.).

### Widoki = motywy

We frameworku MVC kod odpowiedzialny za wyświetlanie danych użytkownikowi zostaje umieszczony w widokach. To właśnie na ich tworzenie projektanci i programiści frontendu poświęcają najwięcej czasu.

W WordPressie motywy to odpowiednie miejsce na umieszczenie kodu i logiki szablonów.

Warto przypomnieć ponownie, że przedstawione tutaj porównanie nie jest do końca prawidłowe. Mimo to stwierdzenie *widoki* = *motywy* to dobry punkt wyjścia.

### Kontrolery = procedury wczytujące szablony

We frameworku MVC kod odpowiedzialny za przetwarzanie danych wejściowych użytkownika (w postaci adresów URL lub danych `$_GET` bądź `$_POST`) oraz ustalający, które z modeli i widoków będą obsługiwały dane żądanie, jest przechowywany w kontrolerach. Kod kontrolera jest, ogólnie rzecz biorąc, obsługiwany przez programistę, a po pierwotnym utworzeniu często pozostaje zapomniany. Sedno programowania w aplikacji MVC odbywa się w modelach i widokach. Jednak pomimo tego kontrolery to komponenty odgrywające ważne role w sposobie działania aplikacji.

W przypadku aplikacji WordPressa wszystkie żądania stron (o ile nie dotyczą buforowanych plików *.html*) przechodzą przez plik *index.php* i są przetwarzane przez WordPressa zgodnie z *hierarchią szablonów*. Procedura wczytująca szablon ustala, który plik szablonu powinien zostać użyty do wyświetlenia strony użytkownikowi końcowemu. Przykładowo plik *search.php* jest używany do pokazania wyników wyszukiwania, plik *single.php* do wyświetlenia pojedynczego posta itd.

To zachowanie domyślne można zmodyfikować za pomocą API WP_Rewrite (jego dokładne omówienie znajdziesz w rozdziale 7.) oraz innych zaczepów i filtrów. Informacje dotyczące hierarchii szablonów (*https://developer.wordpress.org/themes/basics/template-hierarchy/*) znajdziesz w książce *WordPress Theme Handbook*. W niniejszej książce temat hierarchii szablonów będzie dokładniej omówiony w rozdziale 4.

Jeżeli chcesz lepiej zrozumieć sposób działania frameworków MVC, to na stronie frameworka PHP o nazwie Yii (*https://www.yiiframework.com/doc/guide/1.1/en/basics.best-practices*) znajduje się doskonałe wyjaśnienie, jak najlepiej używać architektury MVC[3].

Więcej informacji na temat tworzenia aplikacji internetowych za pomocą frameworka WordPress znajdziesz w dalszej części tej książki.

## Anatomia aplikacji internetowej WordPressa

W tym podrozdziale przedstawimy aplikację, która będzie budowana w książce: SchoolPress. Zaprezentujemy funkcjonalność aplikacji SchoolPress, sposób jej działania i użytkowników, dla których została przeznaczona. Co najważniejsze z pespektywy czytelnika tej książki, dokładnie pokażemy, jak każdy fragment tej aplikacji został utworzony w WordPressie.

Nie przejmuj się, jeśli nie będziesz rozumieć znaczenia pewnych pojęć w stosowanej tutaj terminologii. W dalszych rozdziałach książki powrócimy do omówienia aplikacji SchoolPress i przedstawimy wszystko znacznie dokładniej. Gdy tylko będzie to możliwe, będziemy starać się wskazywać rozdział zawierający dokładniejsze omówienie danej funkcjonalności.

Celem tej książki nie jest pokazać krok po kroku, jak utworzyć aplikację SchoolPress. Gdy ma to sens, fragment aplikacji SchoolPress został użyty w przykładowych fragmentach kodu prezentowanych w książce. Dzięki temu nie musisz poświęcać czasu na próbę zrozumienia każdego z przedstawionych przykładów.

[3] Yii to oparty na architekturze MVC framework PHP. Wprawdzie inne frameworki PHP, takie jak Laravel (*https://laravel.com/*), są popularniejsze wśród programistów WordPressa i ogólnie społeczności PHP, ale dotycząca MVC dokumentacja w witrynie Yii została doskonale napisana.

## Czym jest SchoolPress?

SchoolPress to aplikacja internetowa ułatwiająca nauczycielom pracę z uczniami poza klasą w szkole. Nauczyciel może tworzyć *klasy* i zapraszać do nich uczniów. Każda klasa ma przypisane forum przeznaczone na dyskusje, a bardziej strukturalny system dla nauczycieli umożliwia im publikowanie *zadań*, które muszą być wykonywane przez uczniów.

Działanie aplikacji SchoolPress możesz zobaczyć w witrynie internetowej *https://schoolpress.me/*. Kod źródłowy aplikacji znajduje się w archiwum dostępnym pod adresem *ftp://ftp.helion.pl/przyklady/wordp2.zip*.

## SchoolPress działa w sieci zawierającej wiele witryn WordPressa

SchoolPress to działająca wersja WordPressa składająca się z wielu witryn. Witryna główna zapewnia obsługę bezpłatnych kont pozwalających nauczycielom zapisać się do systemu i rozpocząć tworzenie klas. Obsługuje również wszystkie informacje marketingowe dla witryn poszczególnych szkół w sieci, łącznie ze stronami przeznaczonymi m.in. do zapisywania się do płatnych usług.

Szkoły mogą tworzyć unikatowe poddomeny przeznaczone dla klas organizowanych przez nauczycieli z danej szkoły. Takie podejście zapewnia dokładniejszą kontrolę i raportowanie informacji dla wszystkich klas w szkole. Szczegóły dotyczące używania sieci zawierającej wiele witryn internetowych WordPressa znajdziesz w rozdziale 12.

## Model biznesowy SchoolPressa

Aplikacja SchoolPress używa wtyczek Paid Memberships Pro, PMPro Register Helper i PMPro Network w celu dostosowania do własnych potrzeb procesu rejestracji oraz akceptowania płatności kartami kredytowymi za korzystanie z serwisu.

Szkoła może wykupić dla siebie unikatową poddomenę za roczną opłatą. Inni użytkownicy SchoolPressa nie muszą płacić za dostęp do serwisu. Gdy administrator szkoły zapisze się do aplikacji, może podać nazwę szkoły i usług dla jej poddomeny, np. *<nazwa_szkoły>.schoolpress.me*. Dla szkoły zostaje utworzona nowa witryna w sieci, a szkole jest zapewniony dostęp do uproszczonej wersji panelu głównego WordPressa przeznaczonego dla danej witryny internetowej.

Następnie administrator szkoły rozpoczyna zapraszanie uczniów do systemu. Nauczyciele mogą również żądać zaproszenia do szkoły, które musi być zaakceptowane przez administratora danej szkoły. Nauczyciel może zapraszać uczniów do utworzonej przez siebie klasy. Z kolei uczniowie mogą żądać zaproszenia do klasy, które musi być zaakceptowane przez tworzącego ją nauczyciela.

Nauczyciele mogą bezpłatnie zapisywać się do serwisu SchoolPress i organizować w nim klasy. Strony wyświetlane w poddomenie klasy mogą zawierać reklamy lub stosować inne rozwiązania pozwalające zarabiać pieniądze. Szczegóły związane z konfiguracją e-commerce za pomocą WordPressa będą dokładnie omówione w rozdziale 15.

## Poziomy członkostwa i role użytkowników

Nauczyciele otrzymują członkostwo na poziomie *Teacher* (za pomocą Paid Membership Pro) i własną rolę o nazwie *Teacher*, która pozwala tworzyć i edytować klasy, moderować dyskusje na forach klasy oraz przygotowywać zadania dla uczniów klasy.

Nauczyciele nie mają dostępu do panelu głównego WordPressa. Organizowanie klas i zarządzanie nimi odbywa się za pomocą utworzonych do tego celu formularzy frontendu.

Uczniowie otrzymują członkostwo na poziomie *Student* i domyślną rolę *Subscriber* w WordPressie. Uczeń może przeglądać zajęcia tylko tych klas, do których został zaproszony przez nauczyciela, i tylko w nich uczestniczyć. Szczegóły związane z rolami i możliwościami użytkowników zostaną omówione w rozdziale 6., natomiast w rozdziale 15. znajdziesz informacje na temat poziomów członkostwa i kontroli dostępu.

## Klasy są grupami BuddyPress

Gdy nauczyciel tworzy „klasę", tak naprawdę tworzy *grupę BuddyPress* i zaprasza do niej uczniów. Używając tej grupy, zyskuje dostęp do forów dla klasy, prywatnego systemu komunikatów oraz eleganckiego sposobu organizacji użytkowników.

Fora dyskusyjne klas działają w oparciu o wtyczkę bbPress. Dla każdej klasy jest generowane nowe forum, BuddyPress zaś zarządza dostępem do tych forów. Szczegóły związane z wykorzystaniem wtyczek zewnętrznych i wtyczki bbPress znajdziesz w rozdziale 3.

## Zadanie to przykład CPT

Zadanie to przykład niestandardowego typu posta (ang. *custom post type*, CPT) pozwalającego nauczycielom na użycie formularza frontendu do opublikowania nowych zadań. Zadanie przypomina domyślny post bloga w WordPressie — zawiera tytuł, treść i dołączone pliki. Nauczyciel publikujący dane zadanie jest autorem danego posta.

WordPress ma wbudowane typy, takie jak posty i strony, oraz wbudowane taksonomie, takie jak kategorie i tagi. W przypadku aplikacji SchoolPress tworzone są niestandardowe typy CPT i taksonomie. Więcej informacji na temat tworzenia niestandardowych typów postów i taksonomii znajdziesz w rozdziale 5.

## Rozwiązania zadań są podtypami CPT zadań

Uczeń może publikować komentarze dotyczące zadania, a także za pomocą innego formularza frontendu przesyłać rozwiązanie zadania.

Rozwiązanie zadania, podobnie jak zadanie, również jest przykładem CPT. Rozwiązanie zadania jest powiązane z zadaniem poprzez przypisanie polu `post_parent` rozwiązania wartości identyfikatora zadania, dla którego dane rozwiązanie zostało przekazane. Uczeń może przekazywać treść w postaci tekstu, a także jednego lub więcej załączników.

## Semestry to taksonomie dla CPT klasy

Niestandardowa taksonomia o nazwie *Semester* jest skonfigurowana dla grupy/klasy CPT. Administrator szkoły może dodać nowe semestry do witryn. Przykładowo można utworzyć semestr „jesień 2019", do którego następnie będą przypisywani nauczyciele organizujący klasy. Uczniowie mogą bardzo łatwo przeglądać listę wszystkich klas semestru np. jesień 2019.

## Wydział to taksonomia dla CPT klasy

Niestandardowa taksonomia o nazwie *Department* jest skonfigurowana dla grupy/klasy CPT. Jest dostępna w postaci rozwijanej listy dla nauczycieli podczas tworzenia przez nich klas. Uczniowie mogą przeglądać klasy według wydziałów.

## Aplikacja SchoolPress ma jedną główną niestandardową wtyczkę

W tle poszczególne fragmenty aplikacji SchoolPress są kontrolowane za pomocą pojedynczej niestandardowej wtyczki o nazwie SchoolPress. Ta wtyczka główna zawiera definicje dla różnych CPT, taksonomii i ról użytkowników. We wtyczce znajduje się również kod przeznaczony do modyfikacji sposobu działania innych wtyczek używanych przez SchoolPress, takich jak Paid Memberships Pro i BuddyPress.

Wtyczka główna zawiera też klasy dla administratorów szkoły, nauczycieli i uczniów, które rozszerzają klasę `WP_User`, oraz klasy przeznaczone do tworzenia klas uczniów, zadań i rozwiązań — wszystkie one opakowują klasę `WP_Post`. Te klasy (PHP) umożliwiają zdefiniowanie kodu w sposób obiektowy, co ułatwia kontrolowanie sposobu wprowadzania zmian w kodzie i jego przyszłą rozbudowę. Tworzenie tych klas przynosi radość, podobnie jak praca z nimi. Przykład takiej klasy przedstawiliśmy na listingu 1.1.

*Listing 1.1. Potencjalne zdarzenia dotyczące logowania użytkownika*

```
if($class->isTeacher($current_user))
{
    // To jest nauczyciel, należy wyświetlić narzędzia dostępne dla nauczyciela
    //…
}
elseif($class->isStudent($current_user))
{
     // To jest uczeń, należy wyświetlić narzędzia dostępne dla ucznia
    //…
}
elseif(is_user_logged_in())
{
     // Jeżeli użytkownik jest niezalogowany, należy wyświetlić formularz logowania,
     // a dopiero później powrócić na tę stronę
     wp_redirect(wp_login_url(get_permalink($class->ID)));
     exit;
} else {
     // Jeżeli użytkownik nie należy do klasy, należy go przekierować na stronę zaproszenia
     wp_redirect($class->invite_url);
     exit;
}
```

Temat tworzenia niestandardowych wtyczek zostanie omówiony w rozdziale 3., a rozszerzenia klasy `WP_User` — w rozdziale 6.

## Aplikacja SchoolPress używa kilku innych niestandardowych wtyczek

Czasami się zdarza, że fragmenty kodu utworzone dla konkretnej aplikacji są użyteczne także w innych projektach. Jeżeli kod może być uruchomiony poza kontekstem bieżącej aplikacji i wtyczki głównej, wówczas można go umieścić w oddzielnej, niestandardowej wtyczce.

Przykładem może być tutaj wtyczka o nazwie Force First and Last Name as Display Name (*https://wordpress.org/plugins/force-first-last/*), która okazała się niezbędna dla omawianego projektu. Do działania nie wymaga uruchomienia jakiegokolwiek innego kodu wtyczki głównej i jest użyteczna także w innych witrynach internetowych WordPressa poza kontekstem aplikacji SchoolPress. Dlatego zdecydowaliśmy się na utworzenie dla tej funkcjonalności oddzielnej wtyczki i umieszczenie jej w repozytorium WordPressa, aby również inni użytkownicy mieli dostęp do wymienionej wtyczki.

## Aplikacja SchoolPress używa motywu Memberlite

Witryna główna aplikacji SchoolPress wykorzystuje dostosowany do własnych potrzeb motyw potomny Memberlite. Jeżeli administrator zdecyduje się na wykupienie domeny premium, będzie miał do wyboru więcej motywów potomnych Memberlite, a także możliwość zmiany kolorów, czcionek i logo, aby w ten sposób lepiej dopasować witrynę do marki danej szkoły. Wszystkie motywy wykorzystują projekt responsywny, więc strona prezentuje się dobrze w urządzeniach o różnych wielkościach ekranu: zarówno w tradycyjnych komputerach, jak i urządzeniach mobilnych.

Kod tworzący motyw Memberlite (*https://memberlitetheme.com/*) jest ściśle ograniczony do zadań związanych z wyświetlaniem treści. Kod motywu oczywiście zawiera kod HTML i CSS tworzące układ witryny, a także prostą logikę pozwalającą na integrację z wtyczką główną SchoolPress. Natomiast cały kod odpowiedzialny za przeprowadzanie operacji na niestandardowych typach postów, rolach użytkowników lub wymagający wielu obliczeń znalazł się we wtyczce głównej.

Skoro wiesz już, na czym polega działanie aplikacji SchoolPress, być może zechcesz utworzyć podobną za pomocą WordPressa, zgodnie z podziałem zadań w sposób „zgodny z WordPressem”. Przechodzimy więc do poznawania WordPressa, jego zawartości i sposobu działania.

# WordPress: zbudujesz o wiele więcej!

WordPress, platformę przeznaczoną do tworzenia blogów, obsłuży nawet osoba niemająca pojęcia o programowaniu. Dziś jednak jest on czymś znacznie więcej niż systemem do zarządzania treścią. Kto zna PHP, HTML, CSS i JavaScript, może wykorzystać tę platformę do projektowania wydajnych, skalowalnych, bezpiecznych i elastycznych aplikacji internetowych oraz mobilnych, a także usług sieciowych. Wystarczy poza podstawowymi funkcjami i schematami poznać techniki tworzenia własnych wtyczek, motywów i usług. Łatwo się przekonać, że WordPress jest świetnym narzędziem do tworzenia funkcjonalnych aplikacji — i małych, i rozbudowanych!

W tej książce znajdziesz wyczerpujący opis funkcjonalności WordPressa w wersji 5.4. Dowiesz się też, czy ta platforma spełni Twoje oczekiwania. Zapoznasz się z podstawami WordPressa i z bardziej zaawansowanymi zagadnieniami, takimi jak niestandardowe typy postów, metadane i taksonomie. Nauczysz się organizować kod zgodnie z zasadami programowania zorientowanego obiektowo, a także zapewniać swoim aplikacjom wysoki poziom bezpieczeństwa. Opisano tu również używanie JavaScriptu i technologii AJAX w aplikacji WordPressa oraz API REST i możliwości integracji z aplikacjami zewnętrznymi. Sporo miejsca poświęcono tematyce e-commerce i wtyczkom, które będą najodpowiedniejsze do tego typu aplikacji. Na końcu zamieszczono rozdział poświęcony przyszłości i perspektywom WordPressa.

## W książce między innymi:

- WordPress a standardowe frameworki
- motywy i wtyczki WordPressa
- zarządzanie kontami i rolami użytkowników oraz dostępem do danych
- rozwiązania asynchroniczne, integracja z bibliotekami PHP, zewnętrznymi API i wtyczkami
- obsługa płatności
- skalowanie aplikacji WordPressa

**Brian Messenlehner** programuje od dwóch dekad. Od 2008 roku wdraża rozwiązania oparte na WordPressie. Zajmował się tworzeniem niestandardowych rozwiązań dla takich klientów jak Discovery Channel, Uber, Starbucks, YMCA oraz National Park Services.

**Jason Coleman** jest szefem Stranger Studios i głównym programistą Paid Memberships Pro, czyli platformy obsługi członkostwa w WordPressie. Od ponad pięciu lat zajmuje się tworzeniem aplikacji PHP opartych na WordPressie.

Helion
helion.pl
HELION SA
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl



ISBN 978-83-283-6925-2

INFORMATYKA W NAJLEPSZYM WYDANIU

Cena: 89,00 zł