# Dokument analizy potencjalnych narzędzi potrzebnych do tworzenia projektu

**Projekt**: *Robotovo Dashboard*
**Wersja**: 1.0
**Data**: 2024-12-10
**Autorzy**: Tomasz Petrykowski, Maciej Peta, Dorota Duda, Ewa Węglewska, Paweł Żabczyński
**Zatwierdzone przez**: Powyższy zespół, dr Witold Bołt

---

## 1. Narzędzia i technologie użyte w projekcie

### Wstęp

W ramach tworzenia aplikacji webowej wybrano technologie i narzędzia, które pozwolą na efektywny i szybki proces rozwijania aplikacji. Przy wyborze kierowano się efektywnością, wsparciem społeczności oraz łatwością wdrożenia. Poniżej przedstawiono listę narzędzi wraz z uzasadnieniem ich wyboru oraz opisem integracji pomiędzy nimi.

---

### Język programowania - frontend

Frontendowy język programowania służy do tworzenia wizualnej części kodu. Musi działać na różnych urządzeniach, platformach oraz przeglądarkach. Istotne jest też wsparcie bibliotek pozwalających na tworzenie zrozumiałego lecz prostego w implementacji UI.

**JavaScript + React**
JavaScript, jako najpopularniejszy język frontendowy, umożliwia efektywne tworzenie aplikacji webowych dzięki swojej wszechstronności i prostocie. React,

jako framework, upraszcza projektowanie interfejsu użytkownika, oferując bogate wsparcie społeczności i liczne biblioteki wspomagające development.

**Uzasadnienie wyboru:**

- JavaScript i React są powszechnie stosowane w tworzeniu nowoczesnych aplikacji webowych.
- Duża społeczność gwarantuje wsparcie w przypadku problemów.
- React umożliwia tworzenie dynamicznego i łatwego w rozwoju UI.

## Język programowania - backend

Język programowania backendowy odpowiada za logikę, weryfikację, autoryzację i przetwarzanie danych aplikacji. Musi też posiadać wsparcie bibliotek pozwalające na komunikację z bazą danych oraz innymi API.

**JavaScript + Node.js**
Node.js pozwala pisać backend w JavaScript, co ujednolica stos technologiczny projektu, upraszczając przy tym rozwój aplikacji przez jeden zespół programistów.

**Uzasadnienie wyboru:**

- Użycie jednego języka w całym projekcie zmniejsza złożoność aplikacji oraz umożliwia rozwijanie strony backendowej i frontendowej przez jeden zespół programistów.
- Node.js jest wydajny i pozwala łatwo obsługiwać aplikacje oparte na React.

---

## IDE (środowisko programistyczne)

**IDE** ułatwia tworzenie i zarządzanie kodem poprzez udostępnianie narzędzi pomocniczych, jak wykrywanie błędów składni oraz pozwala integrować w jednej aplikacji narzędzia i pluginy używane przez zespół.

Kwestię wyboru środowiska programistycznego pozostawiamy zespołowi projektowemu. Są dwie opcje do wyboru: **Visual Studio Code** oraz **WebStorm**.

Visual Studio Code to lekkie, wszechstronne IDE z bogatym wsparciem wtyczek. Ułatwia debugowanie, integrację z GitHubem i zarządzanie kodem.

WebStorm to profesjonalne IDE przeznaczone dla programistów JavaScript, TypeScript oraz technologii front-endowych, takich jak React, Angular czy Vue.js.

**Uzasadnienie wyboru dla VSC:**

- Popularność wśród developerów JavaScript.
- Integracja z GitHubem pozwala na płynne zarządzanie kodem bez opuszczania IDE.

- Obsługa narzędzi takich jak Copilot przyspiesza pisanie kodu.

**Uzasadnienie wyboru dla IDE Webstorm:**
Narzędzie to oferuje szeroką gamę funkcji ułatwiających pracę nad kodem, takich jak:

- **Zaawansowana analiza kodu**: WebStorm automatycznie wykrywa błędy w kodzie i sugeruje poprawki dzięki wbudowanemu silnikowi analizy.
- **Wsparcie dla frameworków**: IDE zapewnia natywne wsparcie dla popularnych frameworków, takich jak React, Angular, Node.js, a także dla narzędzi do budowania (np. Webpack).
- **Integracja z systemami kontroli wersji**: WebStorm integruje się z GitHubem, GitLabem i innymi systemami kontroli wersji, co ułatwia zarządzanie projektami.
- **Refaktoryzacja kodu**: Wbudowane narzędzia do refaktoryzacji pozwalają na łatwe zmiany w kodzie, takie jak zmiana nazw zmiennych, funkcji czy plików, z automatycznym dostosowaniem zależności.
- **Debugowanie i testowanie**: IDE oferuje wbudowany debugger oraz wsparcie dla frameworków testowych, takich jak Jest i Mocha, co przyspiesza rozwiązywanie problemów i zapewnia wyższą jakość kodu.

## Kontrola wersji

Narzędzia kontroli wersji ułatwiają pracę na tym samym projekcie pomiędzy wieloma użytkownikami. Pozwalają trzymać kopie kodu zdalnie, co ułatwia wymianę informacji i minimalizuje ryzyko utraty kodu.

**GitHub**
GitHub jest standardem w zarządzaniu wersjami, pozwalającym na przechowywanie kodu, śledzenie zmian oraz współpracę w zespole. Pozwala na integrację wielu etapów tworzenia aplikacji, takich jak np. testowanie, CI/CD, zarządzanie projektem.

**Uzasadnienie wyboru:**

- Dominująca rola na rynku minimalizuje czas wdrożenia nowych członków zespołu.
- Integracja z innymi narzędziami takich, jak VSC, GitHub Actions oraz Github Projects.
- Dodatkowe funkcje, np. Dependabot, zwiększają bezpieczeństwo projektu.

---

## CI/CD oraz testowanie

Narzędzia CI/CD służą do automatyzacji budowy, testów oraz deploymentu kodu do produkcji.

**GitHub Actions**
GitHub Actions to narzędzie zintegrowane z platformą GitHub, umożliwiające automatyzację budowy, testowania i wdrażania aplikacji. Do testowania aplikacji wykorzystany zostanie framework Jest, ponieważ jest przeznaczony do testowania aplikacji tworzonych w Javascripcie.

**Uzasadnienie wyboru:**

- Bezproblemowa integracja z repozytorium kodu.
- Automatyzacja procesów testowania zwiększa efektywność i zmniejsza ryzyko błędów.

## Project Management

Narzędzia do zarządzania projektem służą do przydzielania zadań do pracowników oraz ułatwienia planowania terminów i dalszych części projektu. Ułatwiają komunikację pomiędzy pracownikami, nawet jeśli projekt pokrywa kilka grup w innych fizycznych lokalizacjach. Pozwalają też śledzić postęp zadań w projekcie.

**GitHub Projects**
GitHub Projects to narzędzie do zarządzania zadaniami zintegrowane z platformą GitHub.

**Uzasadnienie wyboru:**

- Proste w obsłudze, mniej skomplikowane niż inne narzędzia, np. Jira.
- Bezpośrednia integracja z repozytorium pozwala na łatwe śledzenie postępów w projekcie.

---

## Narzędzia do designu

Narzędzia do designu pozwalają prototypować wygląd oraz interakcje UI. Pozwala to na planowanie designu strony bez faktycznego programowania danych funkcjonalności, co przekłada się na mniej zmian w kodzie a zatem szybszy development.

**Figma**
Figma to narzędzie do projektowania interfejsów użytkownika, które umożliwia współpracę w czasie rzeczywistym i przechowywanie historii zmian.

**Uzasadnienie wyboru:**

- Aplikacja przeglądarkowa niewymagająca skomplikowanego setupu.
- Współpraca wielu użytkowników w jednym projekcie.
- Możliwość integracji z frontendem za pomocą narzędzi takich jak Zeplin.
- Dodatkowym atutem tego narzędzia jest jego przystępna cena - od 15 € za osobę za miesiąc.

## 2. Integracja narzędzi

Wszystkie wybrane narzędzia uzupełniają się wzajemnie:

- **Visual Studio Code** oraz **Webstorm** integrują się z **GitHubem**, ułatwiając kontrolę wersji i współpracę w zespole.
- **GitHub Actions** wspiera proces CI/CD, bezpośrednio budując, testując i wdrażając kod z repozytorium.
- **Github Projects** pozwoli na efektywne zarządzanie projektem poprzez możliwość automatycznego dodawania problemów do listy "do zrobienia" i nie tylko.
- **Figma** ułatwia wymianę informacji między zespołem projektowym a developerami, umożliwiając szybkie prototypowanie i implementację UI.

Taki zestaw narzędzi pozwala na efektywne zarządzanie projektem, szybki development oraz łatwe wprowadzanie nowych członków zespołu.