Inżynieria Oprogramowania & Bazy Danych - Projekt

1. Wybór technologii

Opracował: Maciej Penar

Spis treści

[1. Wybór technologii 3](#_Toc2851208)

[2. Proof of Concept + Prezentacja 3](#_Toc2851209)

[3. Grupa L5 4](#_Toc2851210)

[4. Grupa L6 5](#_Toc2851211)

# 1. Wybór technologii

Przygotować rozdział do sprawozdania końcowego które:

1. **Krótko** opisuje rozważane dostępne technologie np.:
   1. Front-Endu
   2. Back-Endu
   3. Aplikacji typu Desktop
   4. Aplikacji mobilnych
   5. Konteneryzacja (np. Docker) / Wirtualizacja (np. Vagrant)
2. Zebrać zbiór języków w których piszemy w ramach wybranych technologii i znaleźć przydatne biblioteki. Opisać je zwięźle (1-3 zdania). Interesują nas przede wszystkim:
   1. Metoda zapisu/odczytu z Bazy Danych (np. ORM’y)
   2. Metoda prezentacji danych użytkownikowi
   3. **Biblioteki wspierające testy jednostkowe**
   4. Biblioteki wspierające serializacje/deserializacje JSON’ów/XML/YAML
   5. Biblioteki wspierające autentykację (np. z użyciem SSO)
   6. Pozostałe
3. Przedstawiona jest analiza SWOT dla każdej z technologii. Pomyśleć o:
   1. Warunkach w jakich będziecie stosować
   2. Powszechność języków, paradygmaty, dojrzałość języków / frameworków, zwięzłość kodu
   3. Bibliotekach zewnętrznych
4. Podana jest ostateczna rekomendacja dot. wybranej technologii – **dopóki nie ma tematu to można odpuścić.**

Każda technologia powinna być opisana na **minimum** 1 pełną stronę – dobrze jest pokazać przykład w kodzie, zobrazować diagramami. Dobór technologii odbędzie się na pierwszych zajęciach. Proszę o skoordynowanie się i sklejenie jednego zbiorczego dokumentu z opisem dostępnych narzędzi.

Szablon dokumentu powinien być w katalogu.

# 2. Proof of Concept + Prezentacja

Żeby dokument z zadania 1 był wiarygodny należy przygotować 5-10 minutowe prezentacje dot. dostępnych technologii wraz z prezentacją działającego kodu który realizuje:

* Umieszczenie bytu w bazie danych
* Prezentację tego bytu na liście
* Umożliwia usunięcie bytu z bazy danych
* Wystawienie aplikacji jako strony webowej na **jakimś porcie** na **localhost**

Rozważyć liczbę dostępnych bibliotek, integrację z chmurami/VCS’ami, wspieranie debugowania.

Szablon prezentacji powinien być w katalogu.

# 3. Grupa L5

Wybraliście tematy:

1. Java (3 osoby)
   * Jak wystawić jakiś port na świat / jak stworzyć stronę webową (np. Spring / Sockety 😊 / Servlety)
   * Jakiś ORM (np. OrmLite / Java Persistence API / Hibernate) lub inny sposób komunikacji z BD (np. JDBC)
2. **JS (+ HTML + CSS) (3 osoby)**
   * Server-side (np. NodeJS)
   * Client-side
   * Frameworki (np. ExtJS / Angular / React)
3. **Python (3 osoby)**
   * Jak wystawić jakiś port na świat / jak stworzyć stronę
   * Jakiś ORM lub inny sposób komunikacji z BD
4. Wirtualizacja (3 osoby) = **Vagrant** (hostowany np. na VirtualBox)
   * Jaki jest koncept?
   * Jak go wykorzystać w projekcie?
   * Postawić jakikolwiek tutorial lub usługę np. Jenkins / Redis / Memcached / Tomcat
5. Konteneryzacja (3 osoby) = **Docker** (hostowany na VirtualBox lub Hyper-V)
   * Jaki jest koncept?
   * Jak go wykorzystać w projekcie?
   * Postawić jakikolwiek tutorial lub usługę np. Jenkins / Redis / Memcached / Tomcat

# 4. Grupa L6

Wybraliście tematy:

1. Java (2 osoby)
   * Jak wystawić jakiś port na świat / jak stworzyć stronę webową (np. Spring / Sockety 😊 / Servlety)
   * Jakiś ORM (np. OrmLite / Java Persistence API / Hibernate) lub inny sposób komunikacji z BD (np. JDBC)
2. **C# (2 osoby)**
   * Jak wystawić jakiś port na świat / jak stworzyć stronę webową (**NIE**: WebForms, Silverlight) (np. ASP.NET MVC)
   * Jakiś ORM (np. Entity Framework / Dapper) lub inny sposób komunikacji z BD (np. surowe połączenie)
3. **JS (+ HTML + CSS) (3 osoby)**
   * Server-side (np. NodeJS)
   * Client-side
   * Frameworki (np. ExtJS / Angular / React)
4. **Python (2 osoby)**
   * Jak wystawić jakiś port na świat / jak stworzyć stronę
   * Jakiś ORM lub inny sposób komunikacji z BD
5. Wirtualizacja (3 osoby) = **Vagrant** (hostowany np. na VirtualBox)
   * Jaki jest koncept?
   * Jak go wykorzystać w projekcie?
   * Postawić jakikolwiek tutorial lub usługę np. Jenkins / Redis / Memcached / Tomcat
6. Konteneryzacja (3 osoby) = **Docker** (hostowany na VirtualBox lub Hyper-V)
   * Jaki jest koncept?
   * Jak go wykorzystać w projekcie?
   * Postawić jakikolwiek tutorial lub usługę np. Jenkins / Redis / Memcached / Tomcat

# 5. Linki

1. Java:
   1. Java SDK
      1. SE: <https://www.oracle.com/technetwork/java/javase/downloads/index.html>
      2. EE: <https://projects.eclipse.org/proposals/eclipse-jakarta-ee-platform>
   2. IDE:
      1. Intellij: <https://www.jetbrains.com/idea/download/>
      2. Eclipse: <https://www.eclipse.org/downloads/packages/release/mars/r/eclipse-ide-java-developers>
      3. Netbeans: <https://netbeans.org/>
   3. Framework/kombajn Spring: <https://spring.io/>
   4. Dostęp do danych:
      1. ORMLite: <http://ormlite.com/>
      2. JPI: <https://docs.oracle.com/javaee/6/tutorial/doc/bnbpz.html>
      3. Hibernate: <http://hibernate.org/>
   5. Serwery aplikacji:
      1. Tomcat: <http://tomcat.apache.org/>
      2. Glassfish: <https://javaee.github.io/glassfish/>
   6. Inne:
      1. Maven (budowanie): <https://maven.apache.org/>
      2. Gradle (budowanie): <https://gradle.org/>
      3. Apache commons (productivity): <https://commons.apache.org/>
      4. Guava (productivity): <https://github.com/google/guava>
      5. Lombok (productivity): <https://projectlombok.org/>
   7. Testy:
      1. Junit: <https://junit.org/junit5/>
      2. Mockito: <https://site.mockito.org/>
2. C#
   1. SDK:
      1. .NET 4.7.2: <https://dotnet.microsoft.com/download/dotnet-framework-runtime/net472>
   2. IDE:
      1. Visual Studio: <https://visualstudio.microsoft.com/pl/?rr=https%3A%2F%2Fwww.google.com%2F>
      2. Visual Studio Code: <https://code.visualstudio.com/>
   3. Dostęp do danych:
      1. ORMLite: <https://github.com/ServiceStack/ServiceStack.OrmLite>
      2. Dapper: <https://github.com/StackExchange/Dapper>
      3. Entity Framework: <https://docs.microsoft.com/pl-pl/ef/>
   4. Serwery aplikacji IIS: <https://www.iis.net/> (poszukać instalacji z poziomu Panelu Sterowania w Windows Home)
   5. Inne:
      1. MSBuild (budowanie): <https://docs.microsoft.com/pl-pl/visualstudio/msbuild/msbuild?view=vs-2017>
      2. ~~NANT:~~ [~~http://nant.sourceforge.net/~~](http://nant.sourceforge.net/)
      3. LINQ: <https://docs.microsoft.com/pl-pl/dotnet/csharp/programming-guide/concepts/linq/introduction-to-linq-queries>
   6. Testy:
      1. Nunit: <https://nunit.org/>
3. JS
   1. NodeJS: <https://nodejs.org/en/>
   2. ExtJS: <https://www.sencha.com/products/extjs/>
   3. Angular: <https://angular.io/>
   4. React: <https://reactjs.org/>
   5. ExpressJS: <https://expressjs.com/>
4. Python
   1. SDK:
      1. SDK: <https://www.python.org/downloads/>
   2. IDE:
      1. Eclipse (patrz: Java)
      2. Visual Studio (patrz: C#)
      3. Visual Studio Code (patrz: C#)
   3. Framework: <https://www.djangoproject.com/start/>
   4. Dostęp do danych:
      1. ORM: <https://www.fullstackpython.com/object-relational-mappers-orms.html>
   5. Testowanie:
      1. Unittest: <https://docs.python.org/3/library/unittest.html>
5. Wirtualizacja
   1. Wiki: <https://pl.wikipedia.org/wiki/Wirtualizacja>
   2. Wiki: <https://pl.wikipedia.org/wiki/Hyper-V>
   3. VirtualBox: <https://www.virtualbox.org/>
   4. Vagrant: <https://www.vagrantup.com/>
6. Konteneryzacja:
   1. Wiki: <https://pl.wikipedia.org/wiki/Wirtualizacja>
   2. Wiki: <https://pl.wikipedia.org/wiki/Hyper-V>
   3. Wiki: <https://pl.wikipedia.org/wiki/Docker_(oprogramowanie)>
   4. VirtualBox: <https://www.virtualbox.org/>
   5. Docker: <https://www.docker.com/>

# 6. W pozostałych rolach

1. Upublicznianie adresów lokalnych na potrzeby developementu: <https://ngrok.com/>
2. Repozytoria kodu źródłowego **VCS**:
   1. Oprogramowanie:
      1. GIT: <https://git-scm.com/>
      2. Tortoise: <https://tortoisesvn.net/>
   2. Repozytoria:
      1. Github: <https://github.com/>
      2. Bitbucket: <https://bitbucket.org/>
3. Organizacja pracy:
   1. Trello: <https://trello.com/>
   2. JIRA: <https://pl.atlassian.com/software/jira>
4. **Single-Sign On**, czyli lenistwo na wyższym poziomie:
   1. Wiki: <https://en.wikipedia.org/wiki/Single_sign-on>
   2. Protokoły: WS-FED / SAML / SAML2 / OAuth / OAuth2 / CAS
   3. Via facebook: <https://developers.facebook.com/docs/facebook-login/web>