**ДИПЛОМНА РАБОТА**

на тема

„Платформа за събиране на данни в реално време”

Дипломант: **Иван Цветомиров Иванов**

Специалност: **Разпределени системи и мобилни технологии**

Факултетен номер: **5MI3400182**

Научен ръководител:

**Проф. д-р Милен Петров**

**Съдържание**

**Глава 1. Увод**

1.1. Актуалност на проблема и мотивация

1.2. Цел и задачи на дипломната работа

1.3. Очаквани ползи от реализацията

1.4. Структура на дипломната работа

**Глава 2. Технически обзор**

2.1. Основни дефиниции

2.2. Подходи, методи, модели и стандарти за решаване на проблемите

2.3. Съществуващи решения

2.4. Избор на критерии за сравнение и сравнителен анализ на решения/методи/стандарти/

2.5. Изводи

**Глава 3. Използвани технологии**

3.1. Изисквания към средствата (технологии, платформи и методологии)

3.2. Видове средства (технологии, платформи и методологии) и начин и място за използването им – сравненителен анализ

3.3. Избор на средствата (технологии, платформи и методологии)

3.4. Изводи

**Глава 4. Анализ**

4.1. Концептуален модел

4.2. Потребителски (функционални) изисквания (права, роли, статуси, диаграми, ...)

4.3. Качествени (нефункционални) изисквания (като напр. преносимост, използваемост, скалируемост, поддръжка, ...)

4.4. Работни (бизнес) процеси

4.5. Изводи

**Глава 5. Проектиране**

5.1. Обща архитектура – напр. слоеве, модули, блокове, компоненти...

5.2. Модел на данните (напр. база данни, файлова структура, ...)

5.3. Диаграми (на структура и поведение - по слоеве и модули, с извадки от кода)

5.4. Потребителски интерфейс (опционално)

5.5. Ресурсни и спомагателни модули (опционално)

**Глава 6. Реализация, тестване/експерименти и внедряване**

6.1. Реализация на модулите

6.2. Системна интеграция (опционално)

6.3. Планиране на тестването - тестови сценарии, процедури, ...

6.4. Модулно и системно тестване

6.5. Анализ на резултатите от тестването и начин на отразяването им

6.6. Експериментално внедряване (технологични изисквания, инсталиране, условия, използване, ...)

**Глава 7. Заключение**

7.1. Обобщение на изпълнението на началните цели

7.2. Насоки за бъдещо развитие и усъвършенстване

# Глава 1. Увод

# Актуалност на проблема и мотивация

Последните няколко години доста силно навлизат интелигентни сензори за измерване на различни параметри на околната среда, съответно и доста нашумяват различни уеб платформи за събиране, обработка и съхранение на различни измервания в реално време.

Подобни проекти са доста предпочитани от любители на метеорологията и чрез публикуваните данни, може да бъде реализирана мрежа за „отворени данни“, които да се използват за различни цели.

# Цел и задачи на дипломната работа

Основна цел на текущата дипломна работа е проектирането и разработването на платформа за събиране, обработване и съхранение на данни от различни измервателни IoT устройства в реално време.

При проектирането и разработването на проекта, трябва да бъде предвидено крайното предназначение – работа с любители на метрологията и да бъде предоставен удобен и лесен потребителски интерфейс, но също така и лесен, но сигурен начин за публикуване на данни в платформата.

Освен определяне на основни изисквания към платформата, трябва да бъдат проучени и подходящи бази от данни за съхранение на обработените данни.

При разработването на проекта ще бъде основен фокус върху използването на езици, работни рамки и проекти с отворен код.

Също така самият проект ще се разработва като проект с „отворен код“.

# Структура на дипломната работа

Дипломанта работа е структурирана от следните глави. Всяка глави описва конкретна стъпка от анализиране на основните изисквания към проекта, проектирането на основните модули и тяхната разработка с конкретно използвани технологии.

* Глава 1
* Глава 2
* Глава 3
* Глава 4
* Глава 5

# Глава 2. Технически обзор

…

# Основни дефиниции

Основните процеси в проекта ще бъдат основно върху комуникацията между IoT устройствата и платфомата, обработката и анализирането на данните и тяхното съхранение.

Всеки регистриран потребител с потвърден профил ще може да добавя неограничен брой устройства.

# Подходи за решаване на проблемите

….

# Примерни решения

Проекта Sensor Community е реализиран по следния начин:

# Изводи

# Глава 3. Използвани технологии

Основен фокус при избора на езици за разработка, софтуерни рамки, платформи и технологии е да бъдат с „отворен код“.

**Въведение и основни изисквания към използваните езици, работни рамки, бд, SQL срещу NoSQL**

# Java

Обектно ориентиран език за софтуерна разработка разработен и подържан от Oracle и има слените предимства:

**Платформено независим** – веднъж компилирано приложението, то може да бъде изпълнявано от всички операционни системи без да е необходима компилация за всяка от тях.

**Сигурност** – базирайки се на платформената независимост, то реализираните приложения не се изпълняват директно от операционната система, а от виртуална машина наречена “Java Virtual Machine” или JVM.

Тази виртуална машина се грижи за заделяне на ресурси като памет, оптимизиране на изчислителното време на процесора, освобождаване на паметта.

**Голям набор от софтуерни пакети** – платформата предоставя широк набор от класове за работа с различни ресурси от операционната система, файлова система и други.

Както всички обектно ориентирани езици софтуерната реализация се организира в класове, обекти и пакети.

**Java се организира в класове и пакети.**

**Структура на Java приложения.**

Процесът на компилация и изпълнение на Java приложение е показан на долната фигура. [link]

# 

*Фигура 1.*

Както беше споменато приложенията не се изпълняват директно от операционната система, а от виртуална машина, като за тази цел при компилация не се компилира до машинен език, а до “java byte code”, за което изпълнение е необходима инсталирана Java RE.

Java е широко използван за реализация на различен тип приложения като:

1. Уеб приложения
2. Уеб услуги
3. Мобилни приложения
4. Разпределени и облачни приложения
5. Десктоп приложения

Java основно се разработва и подържа от Oracle и софтуерната общност на Java екосистемата (SAP, VMware, Eclipse, Apache, …).

Като платформа за софтуерна разработка има следните версии/editions:

**Java Standard Edition (SE**) – съдържа основните пакети и основните API на езика за разработка на приложения.

**Java Enterprise Edition (EE)** – пакети и спецификация за разработка на бизнес приложения. В последствие преименуван на Jakarta EE и се разработва и подържа от Eclipse Software Foundation. https://www.scaler.com/topics/types-of-java/



*Фигура 2*

Съществуват и други допълнения или видове на Java като JavaFX, JavaME но те не са от особен интерес при разработката на проекта.

# Основни Java пакети

**Java Standard Edition (SE**) – съдържа основните пакети и основните API на езика за разработка на приложения.

# Jakarta ЕЕ

Java EE или Jakarta ЕЕ е набор от спецификации и стандарти за разработката на бизнес приложения и е широко използван за реализация на различни облачни приложение, уеб услуги и приложения.

При Jakarta EE основен фокус е сигурността и различни оптимизации при внедряването в облак.

В момента Jakarta се разработва и подържа от Eclipse Software Foundation и софтуерната общност на IBM, Apache Software Foundation, VMware, SAP и други.

На долната фигура са показани основните Jakarta EE профили.[2]



*Фигура 6. Jakarta EE профили*

# Основни спецификации за уеб приложения

**Jakarta Servlet –** основата на другите спецификации от екоситемата на Jakarta за обработка на HTTP заявки. Тук буквално се дефинира „Как се управляват HTTP заявките?“ чрез синхронна или асинхронна обработка.

**Jakarta WebSocket –** спецификация за разработването на WebSocket приложения изискващи комуникация в „реално време“.

**Jakarta Faces –** спецификация позната още като: Jakarta Server Faces или Java Server Faces и един от първите стандарти за създаването на уеб приложения използвайки MVC шаблон.

**Jakarta MVC –** това е стандарт базиран на идеологията на Jakarta Faces но реализиран посредством …. [todo]

# Основни спецификации за уеб услуги

**Jakarta RESTful Web Service** – основната спецификация за разработка на уеб услуги използвайки REST шаблон за комуникация… [todo] – also Microprofile.

**Jakarta XML Web Service** – основната спецификация за разработка на уеб услуги използвайки SOAP протокол.

# Enterprise спецификации

# 

# Spring

**Въведение в Spring, история на Spring и Как Spring надгражда Jakarta EE.**

Spring e широко използвана софтуерна рамка с отворен код за разработка на уеб приложения, уеб услуги, автоматизации на бизнес процеси, работа с бази от данни базирайки се на Java EE стандарта.

В голяма степен надгражда Java EE/Jakarta и предлага широк спектър от оптимизации и автоматизации при реализацията на основните бизнес процеси, работа с бази от данни, допълнителни библиотеки и услуги от трети страни.

# Spring Framework

Реализация на високо ниво за разработка на уеб приложения чрез Java EE/Jakarta и в голяма степен е алтернатива на Enterprise Java Beans.

На фигура 2 е показана основната структура на приложната рамка за разработка.



Фигура 2. Архитектура на Spring Framework

Spring Framework предлага следните неща:

* Уеб приложения използвайки MVC шаблона
* Поддръжка на JSP/JSTL
* REST уеб услуги
* Работа с релационни БД
* Работа с нерелационни БД
* Средства за автирузация и аут…
* Предаване на съобщения с различни протоколи

Spring Framework подържа слените Servlet приложни сървъри:

* Apache Tomcat
* Eclipse Jetty

# Spring Boot

Spring Boot предлага готова конфигурация на Spring Framework базирана на шаблона „Convention over configuration“, включително предлага и вграден приложен уеб сървър – Apache Tomcat, Eclipse Jetty и Netty.

Архитектура на Spring Boot приложение e показана на фигура 3.



Фигура 3.

Тук се подържат два стандарта за разработка на уеб приложения и услуги:

**Spring web/Servlet**

**Spring Webflux/Reactive**

**Spring Security**

Тук се предлагат голям набор от класове и конфигурации за authentication и authorization на разработеното приложение. Принципът на работа на Spring Security e показан на фигура 4.



Фигура 4.

**Spring Data**

…

# Spring Cloud

# Mongo Db

# Redis

# Docker

# React.js и Next.js

# Други

Тук ще се изброят всички помощни библиотеки и средства.

Може да се включат IDE и описание на някой процеси.

* TypeScript
* Material UI
* Formik
* Yup

# Глава 4. Анализ

4.1. Концептуален модел

4.2. Потребителски (функционални) изисквания (права, роли, статуси, диаграми, ...)

4.3. Качествени (нефункционални) изисквания (като напр. преносимост, използваемост, скалируемост, поддръжка, ...)

4.4. Работни (бизнес) процеси

4.5. Изводи

# Концептуален модел

# Функционални изисквания

**Изисквания на потребителите към платформата.**

**Use-case диаграма и анализ**

**5 страници, детайлно разглеждане на всеки случай на употреба**

# Нефункционални изисквания

1. **Сигурност**
2. **Производителност**
3. **Комуникация между IoT и платформата**
4. **Управление и анализ на устройства и данни**

**3 страници**

# Основни бизнес процеси

Use case диаграми.

# Изводи

# Глава 5. Проектиране

Тук се разглежда и описва процеса на проектиране на отделните модули от платформата основавайки се на основните изисквания и ограничения към тях.

# Обща архитектура

Общата софтуерна архитектура на разработеното приложение е показана на долната фигура.



*Фигура 4: Обща архитектура на приложението*

За софтуерна архитектурна реализация е избран подхода с използването на уеб услуги и разделянето на отделни три слоя, като всеки слой да има определени функции и изпълнява определени роли и изисквания.

1. Слой UI
2. Сървърен слой
3. Слой за съхранение

Frontend – това е клиентката част на приложението, тук е включен и потребителския интерфейс UI, също така той изцяло двупосочно комуникира с сървърната част.

Backend – сървърната част на приложението и тук е реализация на бизнес логиката, в конкретния случай е важен и подбора на софтуерна архитектура, модел за управление на данните в БД и т.н.

Database – отговаря за съхранение на обработените данните от сървърната част.

Избран е подхода за използване на уеб услуги, по този начин имаме разделение между клиентската и сървърната част. За разлика от класическия метод за разработка на уеб приложения, и изгледите да бъдат генерирани от сървърното приложение, то тук имаме следните предимства:

1.

2.

3.

4.

Както беше разгледано има доста предимства избрания принцип за реализация чрез използване на уеб услуги, пред класическия модел за реализация на уеб приложения за директно изпращане на изгледи.

# Реализация на клиентката част (UI)

Spring e широко използван framework с отворен код за разработка на Java приложения (уеб приложения и услуги, автоматизации на бизнес процеси и работа с бази от данни). В голяма степен надгражда Java EE/Jakarta и предлага широк спектър от оптимизации и автоматизации при реализацията на основните бизнес процеси, работа с бази от данни, допълнителни библиотеки и услуги от трети страни.

# Реализация на сървърната част (Server)

Spring e широко използван framework с отворен код за разработка на Java приложения (уеб приложения и услуги, автоматизации на бизнес процеси и работа с бази от данни). В голяма степен надгражда Java EE/Jakarta и предлага широк спектър от оптимизации и автоматизации при реализацията на основните бизнес процеси, работа с бази от данни, допълнителни библиотеки и услуги от трети страни.

# Модел на данните

Spring e широко използван framework с отворен код за разработка на Java приложения (уеб приложения и услуги, автоматизации на бизнес процеси и работа с бази от данни). В голяма степен надгражда Java EE/Jakarta и предлага широк спектър от оптимизации и автоматизации при реализацията на основните бизнес процеси, работа с бази от данни, допълнителни библиотеки и услуги от трети страни.

# Глава 6. Софтуерна реализация

Реализация на модулите

Системна интеграция

Модулно и системно тестване

Анализ на резултатите от тестването

Внедряване на проекта (технологични изисквания, инсталиране, условия, използване, ...)

Spring Resource Server

Spring Authorization Server

Микросървисна архитектура и концепция за реализация

Zookeeper, Spring Cloud предимства, Spring Eureka

# Глава 7. Заключение

7.1. Обобщение на изпълнението на началните цели

7.2. Насоки за бъдещо развитие и усъвършенстване

# Използвана литература

1. Източник 1
2. <https://jakarta.ee/>
3. Източник 2
4. Източник 3
5. Източник 4

# Приложение № 1: Терминологичен речник

# Приложение № 2: Потребителски наръчник