|  |  |  |
| --- | --- | --- |
|  | **Софийски университет „Св. Кл. Охридски”**  Факултет по математика и информатика  *Катедра „Информационни технологии”* |  |

**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**

София, 2023 г.

**Съдържание**

(препоръчителен обем без приложенията: от 60 до 80 стр.)

**Глава 1. Увод** (3-5стр.)

1.1. Актуалност на проблема и мотивация (0,5-1стр.)

1.2. Цел и задачи на дипломната работа (1-2стр.)

1.3. Очаквани ползи от реализацията (1-2стр.)

1.4. Структура на дипломната работа (0,5-1стр.)

**Глава 2. Преглед на предметната област (да се замени с конкретно заглавие според заданието)** (10-15стр.)

2.1. Основни дефиниции

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

2.3. Съществуващи решения (практически реализации)

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

2.5. Изводи

**Глава 3. Използвани технологии, платформи и/или методологии (за практическото решаване на проблема)** (10-15стр.)

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** (10-15стр.)

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** (10-15стр.)

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 (евентуално) внедряване** (10-15стр.)

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** (1-2стр.)

7.1. Обобщение на изпълнението на началните цели

7.2. Насоки за бъдещо развитие и усъвършенстване

**Използвана литература** (min 10 литературни източника – статии, книги, с, форматирани съгласно MLA Style - <http://www.library.mun.ca/guides/howto/mla.php>)

**Приложения (опционално)**

**Приложение 1: Терминологичен речник (опционално)** (1-2стр.)

**Приложение 2, 3, ....:** напр. код или извадки от кода, наръчник на потребителя, екрани от потребителския интерфейс, спецификации, диаграми, и др.

# Глава 1. Увод

# Актуалност на проблема и мотивация

Последните няколко години доста силно навлизат интелигентни сензори и платформи за събиране на данни за различни параметри на околната среда в реално време и тяхната обработка и анализ.

Подобен тип проекти са доста харесвани от любители на метеорологията и чрез споделените от тях данни, може да бъде изградена онлайн платформа за „open data”.

# Цел и задачи на дипломната работа

Основна цел на текущата дипломна работа е проектирането и разработването на платформа за събиране на данни в реално време от любителски измервателни устройства (Internet of Things).

При проектирането и разработката на платформата трябва да бъде взето под внимание, че тя ще бъде за крайни потребители и трябва да има удобен и лесен интерфейс за работа и комуникация.

Също основни фактори при проектирането и съответно разработването трябва да бъдат:

1. Сигурност
2. Пройзводителност
3. Комуникация между устройствата и платфомата
4. Управление и анализ на устройства и данни

Основна цел е разработването на платформа за събиране на данни от околната среда от IoT устройства на крайни потребители измерващи определени параметри например: температура, атмосферно налягане, качество на въздуха изразено от съдържанието на фини прахови частици, но също така наличието на определени газове във въздуха.

Последните години става и доста модерно измерването на UV индекс на слънчевите лъчи и шумовото замърсяване.

Също така основна идея на проекта е потребителя лесно да може да публикува своите данни в платформата и той да може да конфигурира лесно използваното устройство, но също така да гарантира за публикуваните данни.

При разработването на проекта, основен фокус ще има върху използването на продукти с отворен код.

# Структура на дипломната работа

Дипломната работа е съставена от следните глави, като всяка глава описва конкретна стъпка при разработването.

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

# Глава 2. Технически обзор на проблема

2.1. Основни дефиниции

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

2.3. Съществуващи решения (практически реализации)

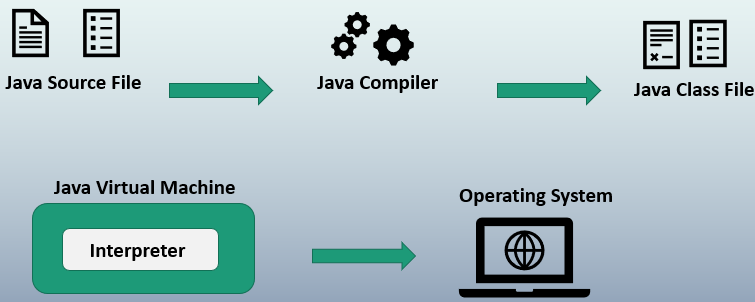
2.4. Избор на критерии за сравнение и сравнителен анализ на решения/методи/стандарти/...

2.5. Изводи

# Глава 3. Използвани технологии

# Java

Java е обектно-ориентиран език за програмиране и основната идея е веднъж компилирано приложение, да бъде платформено независимо. Създадените Java приложения не се изпълняват директно от операционната система, а от JVM (Java Virtual Machine – виртуална машина). Самият процес е показан на фигура 1.



Фигура 1. Java Virtual Machine

JVM се грижи за паметта за данни и инструкции, като освобождава паметта от ненужните ресурси. Също така се грижи за работата със файлова система и оптимизира използваното процесорно време.

Java се използва широко за разработката на различни уеб приложения, мобилни приложения за Android, мрежови приложения и други.

# Spring Framework

Spring е най-използваната работна рамка „framework” за разработка на Java приложения, основно се използва за изграждането на уеб приложения и уеб услуги. Но последните няколко години доста се използва за разработката на различни автоматизации и бизнес процеси в реално време. Важно е да се отбележи, че е базирана на Java EE/Jakarta и надгражда в голяма степен.

* Jakarta EE
* Spring Boot
* Spring Batch
* Spring Cloud

Spring е една от най-популярните работни рамки – „framework” за разработка на уеб приложения с Java, а в последно време и на различни автоматизации и бизнес процеси. Създаден е от Pivotal, а в момента се подържа от VMware и самата работна рамка предоставя много инструменти и допълнителни билблиотеки за улеснение при разработката на уеб приложения. Spring e много гъвкав и лесно се интегрира с други библиотеки (например OAuth).

* Jakarta EE – това е стандарта за бизнес приложения за Java I самата работна рамка е изградена чрез него, в голяма степен го надгражда.
* Spring Boot – предлага готова конфигурация и автоматизация на процесите с разработката и стартирането на готовото приложение. За разлика от Spring Framework, тук е включен и приложен уеб сървър.
* Приложен уеб сървър – за Spring Framework може да се използва всеки един приложен уеб сървър, като единствено трябва да подържа съответната имплементация на Jakarta. Както беше споменато, в Spring Boot е включен приложен уеб сървър. Стандартно е Apache Tomcat, но може лесно да бъде заменен с друг, като Eclipse Jetty и Netty.

# Mongo DB

Използването на Apache Kafka за messaging платформа предлага редица предимства, включително

# Apache Kafka

Използването на Apache Kafka за messaging платформа предлага редица предимства, включително:

* **Висока производителност:** Kafka е проектирана да бъде много производителна и може да обработва големи обеми данни в реално време.
* **Гъвкавост:** Kafka може да се използва за различни приложения, включително обработка на събития, анализ на данни и интеграция на системи.
* **Скейлируемост:** Kafka е мащабируема система, която може да се разширява, за да поддържа нарастващия обем на данни.
* **Надеждност:** Kafka е надеждна система, която е проектирана да може да възстановява данните в случай на срив.
* **Отвореност:** Kafka е отворен код, което означава, че може да бъде разглеждана и променяна от всеки.

Като цяло, Apache Kafka е мощна и гъвкава messaging платформа, която може да се използва за различни приложения. Ако търсите високопроизводителна, мащабируема и надеждна система за обмен на данни, Kafka е чудесен избор.

Ето някои конкретни примери за приложения, за които Kafka може да се използва:

* **Обработка на събития:** Kafka може да се използва за събиране и обработка на събития от различни източници в реално време. Това може да се използва за подобряване на производителността, видимостта и вземането на решения.
* **Анализ на данни:** Kafka може да се използва за съхраняване и анализ на големи обеми данни от различни източници. Това може да се използва за идентифициране на тенденции, модели и аномалии.
* **Интеграция на системи:** Kafka може да се използва за интегриране на различни системи и приложения. Това може да се използва за подобряване на ефективността и гъвкавостта.

Ако търсите messaging платформа, която може да се използва за различни приложения, Apache Kafka е чудесен избор.

# TypeScript

# React

# Material UI

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** (10-15стр.)

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 (евентуално) внедряване** (10-15стр.)

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** (1-2стр.)

7.1. Обобщение на изпълнението на началните цели

7.2. Насоки за бъдещо развитие и усъвършенстване

**Изисквания за оформяне на дипломната работа:**

1. Това е препоръчителен шаблон, в зависимост от конкретното задание.
2. Йерархията на структуриране на съдържанието да не бъде повече от 3 нива, номерирани с арабски цифри – напр. 1.2.3.
3. Чуждестранните термини да бъдат преведени, а където това не е възможно – цитирани в *курсив* и нечленувани.
4. Страниците да бъдат номерирани с арабски цифри, в долния десен ъгъл.
5. Използваният шрифт за основния текст на описанието да бъде Times 12 или Arial 10, и Courier 9 за кода, с междуредие 16pt.
6. Да се избягват пренасянията на нова страница на заглавия на секции, фигури и таблици.
7. Да се избягват празни участъци на страници вследствие пренасянето на фигури на нова страница.
8. Всички фигури и таблици да бъдат номерирани и именовани (непосредствено след фигурата или таблицата).
9. Всички фигури и таблици да бъдат цитирани в текста.
10. Използваните фигури от други източници да бъдат цитирани.
11. Всички цитати да бъдат отразени в списъка на използваната литература.
12. Всички източници от списъка на използваната литература да бъдат цитирани в текста.
13. Използваната литература да се цитира съгласно MLA Style - <http://www.library.mun.ca/guides/howto/mla.php>