**МІНІСТЕРСТВО ОСВІТИ І НАУКИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»**



**ЗАПИСКА ДО КУРСОВОЇ РОБОТИ**

**З дисципліни «Мікроконтролери, частина 1» за темою «Датчики температури та тиску»**

**Виконав:**

ст. гр. ІР-23

ІКТА

Пазиняк В.

**Прийняв:** доцент КСА Павельчак А.Г.

**Львів 2022**

# Зміст

1. Мета та завдання до курсового проекту.
2. Вступ
3. Аналіз питання (короткий опис використаних технологій і т.п.)
4. Структурна та функціональна схема проекту з загальним описом складових.
5. Детальні описи окремих складових:
   1. Апаратна частина
   2. Back-end
   3. БД
   4. Веб додаток
6. Загальна ілюстрація працездатності проекту.
7. Висновки

# Мета та завдання до курсового проекту.

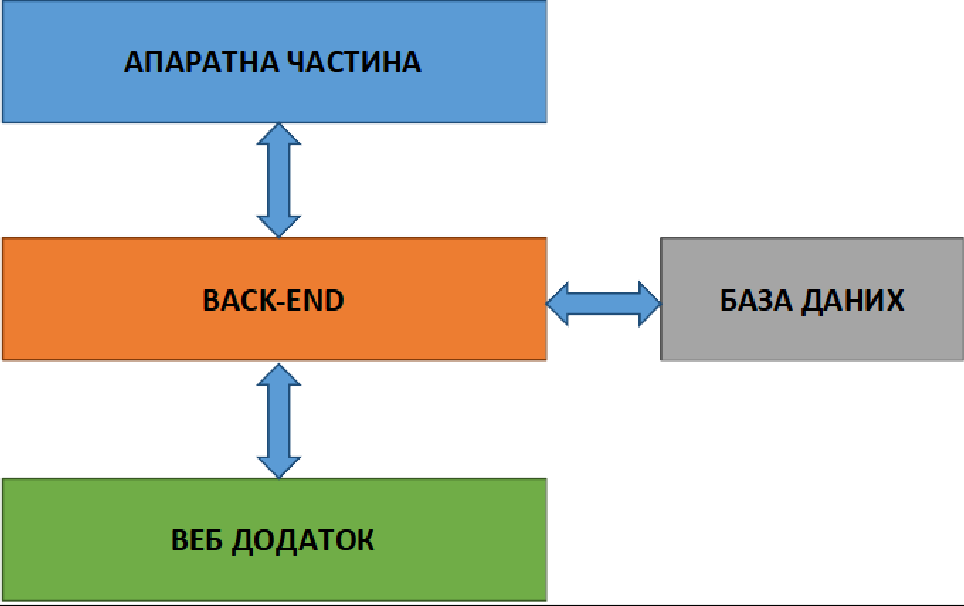
**ЗАВДАННЯ НА КУРСОВИЙ ПРОЕКТ**

Даний курсовий проект має на меті закріплення знань та навичок, здобутих під час навчання, та охоплює мікроконтролери, схемотехніку, бази даних, веб-програмування та інші.

## Проект має виконуватися на фізичному залізі (або Proteus), і реально працювати.

Тему курсового проекту можна обирати самостійно відповідно до вимог щодо складової, що продемонстровано нижче:

Основні складові курсового проекту:



# Вступ

У курсовій роботі виникла ідея реалізувати Барометр широкого призначення. Тобто цей пристрій можна застосувати в різній місцевості, ситуаціях чи використовувати як домашній пристрій для контролю

атмосферного тиску. Що таке саме барометр.

Барометр - прилад для вимірювання атмосферного тиску. Також може застосовуватися як альтиметр, для вимірювання висоти над рівнем моря.

Застосування:

Найчастіше, барометри застосовують для передбачення погоди. Також барометри застосовують для барометричного нівелювання в геодезії та маркшейдерській справі.

Перший барометр був ртутним. **Ртутний барометр** винайдений учнями Галілео Галілея, італійцями Еванджеліста Торрічеллі та Вінченцо Вівіані у 1643 році., це була тарілка з налитою в неї ртуттю та пробіркою (колбою), поставленою отвором вниз. Коли атмосферний тиск підвищувався, ртуть піднімалася в пробірці, коли вона знижувалася — ртуть опускалася. Через незручність така конструкція перестала застосовуватися і поступилася

місцем барометру-анероїду, але метод, за яким такий барометр був виготовлений, став застосовуватися в термометрах.

Барометр-анероїд - Так звані анероїди, у яких зміна атмосферного тиску змушує стискуватися або розширюватися гофровану металеву коробку (вакуумну камеру) з розрідженим повітрям усередині. Ці деформації, за допомогою системи важелів та шарнірів, передаються стрілці, що рухається по шкалі, на якій стоять позначки, які відповідають тиску. Показники анероїда з часом змінюються, внаслідок зміни пружності стінок коробки, тому його необхідно час від часу звіряти з ртутним барометром. Завдяки своїй портативності, широко застосовується в експедиціях, а також як висотоміри (тоді шкалу анероїда градуюють у метрах).

До шкали анероїда часто прикріплений дугоподібний термометр, який служить для внесення поправки на температуру. Похибка вимірювань

анероїд становить 1-2 мбар. Для отримання істинного значення тиску анероїд потребує три поправки:

на шкалу — зумовлена тим, що анероїд неоднаково реагує на зміну тиску в різних ділянках шкали;

на температуру — зумовлена залежністю пружних властивостей коробки і пружини від температури;

додаткова — обумовлена зміною пружних властивостей коробки і пружини з часом.

У наш час пристрій визначення атмосферного тиску – барометр, є майже в кожного. Вони бувають механічними, електронними, у вигляді датчиків і навіть вбудовані в годинники та смартфони.

Оскільки мене зацікавив такий пристрій, я вирішив його реалізувати, що також дало змогу підтягнути знання з Мікроконтролерів, Схемотехніка Веб- програмування, бази даних та освоєння інших технологій чи знань.

# Аналіз питання

Мій курсовий проект дасть змогу отримувати поточні дані в реальному часі, які будуть відображатись на зручному Веб інтерфейсі.

Для реалізації цього проекту та всі потрібні для нього складові, а саме Апаратну частину, Front-end, Back-end та Базу даних, потрібно було ознайомитись з різними технологіями, фреймворками та засобами роботи

над проектами, а також використовувати знання з різних сфер програмування. Отже перейдемо до ознайомлення.

## Апаратна частина

Оскільки я не маю можливості зібрати фізичну схему пристрою, я буду використовувати Proteus Design - пакет програм для автоматизованого проєктування (САПР) електронних схем.

Розробляється компанією Labcenter Electronics (Велика Британія).

Використані компоненти:

**Мікроконтролер Arduino Uno** (у Proteus Design використано мікроконтролер Simulino UNO[v4.0] )

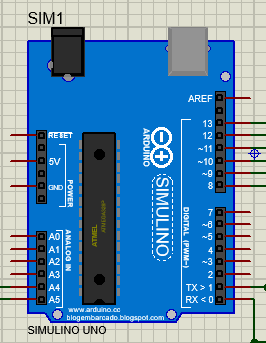
Arduino Uno — це широко використовувана плата мікроконтролерів з відкритим кодом на базі мікроконтролера ATmega328P. У його склад входить все необхідне для зручної роботи з мікроконтролером:

14 цифрових входів/виходів (з них 6 можуть використовуватися в якості ШІМ-виходів), 6 аналогових входів, кварцовий резонатор на 16 МГц, роз'єм USB, роз'єм живлення, роз'єм для програмування

всередині схеми (ICSP) і кнопка скидання.



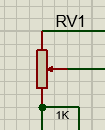
Фізична версія Arduino Uno



Arduiuno Uno Proteus Design

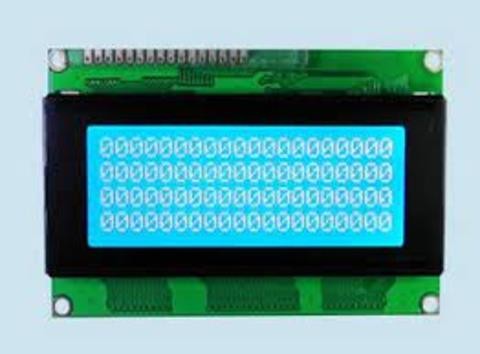
**Потенціометр** (potentiometer) — змінний резистор із трьома виводами, один із яких рухомий, що використовується, як дільник напруги.

Електричний струм проходить між кінцевими контактами, а потрібна споживачеві напруга знімається з повзунка, положення якого можна механічно змінювати.

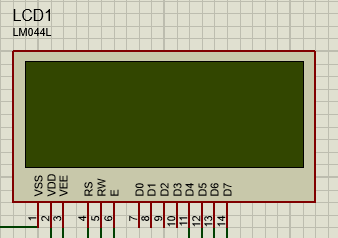


Потенціометр у Proteus Design

**LCD Дисплей (LCD Display) LM044L -** 20 символів x 4 рядки • Вбудований контроль LSI HD44780 типу • +5V один блок живлення MECH



Фізичний LCD Дисплей LM044L



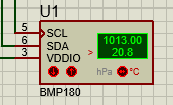
LCD Дисплей LM044L у Proteus Design

**Датчик Барометра BMP180** - Цей прецизійний датчик від Bosch є найкращим рішенням для вимірювання барометричного тиску та

температури. Це надмалопотужний цифровий датчик температури та тиску з високою точністю та стабільністю. Оскільки тиск змінюється з висотою, ви також можете використовувати його як висотомір.



Фізичний Датчик Барометра BMP180



Датчик Барометра BMP180 у Proteus Design

## Back-end

**Back-end** - двигун сайту з усіма його зв’язками з механізмами хостинга — базою даних, веб-сервером.

REST – це стиль архітектури API для розподілених систем. REST визначає, як дані надаються клієнту в зручному для нього форматі. Обмін даними відбувається у форматі JSON або XML (використовуватиму формат JSON та MVC-паттерн).

Для розробки Бекенду я використовував такі засоби:

1. **Мова програмування Java**. Використана версія для розробки 16 (Version 16).

Java - об'єктно-орієнтована мова програмування. Синтаксис мови багато в чому походить від C та C++. Java програми виконуються у середовищі віртуальної машини Java. Java програми компілюються у байткод, який при виконанні інтерпретується віртуальною машиною для конкретної платформи.

## Середовище розробки (IDE) - IntelliJ IDEA.

1. **Spring Boot** — це веб-фреймворк Java з відкритим вихідним кодом на основі мікросервісів. Фреймворк Spring Boot створює повністю готове до виробництва середовище, яке можна повністю налаштувати за допомогою попередньо створеного коду в його базі коду. Архітектура мікросервісів надає розробникам повністю закритий додаток, включаючи вбудовані сервери додатків.

Spring Boot — це лише розширення вже існуючих і розширених фреймворків Spring.

Цей фреймворк ідеально підходить для реалізації поставленої задачі.



Логотип Spring Boot

1. **Apache Maven** — це інструмент для управління проектами

програмного забезпечення та його розуміння. На основі концепції об’єктної моделі проекту (POM), Maven може керувати збіркою проекту, звітами та документацією з центральної частини інформації.

Використовуватиму для збірки проекту.



Логотип Apache Maven

1. **Python** - інтерпретована об'єктно-орієнтована мова програмування високого рівня зі строгою динамічною типізацією.

Використовуватиму для спілкування між мікроконтролером та бекендом, передачі та отримання даних через Serial Port.

## 3. Front-end

1. **Front-end** – це публічна частина web-додатків (веб-сайтів), з якою користувач може взаємодіяти і контактувати напряму. У Front-end входить відображення функціональних завдань призначеного для користувача інтерфейсу, що виконуються на стороні клієнта, а також обробка запитів користувачів. По суті, фронтенд – це все те, що бачить користувач при відкритті web-сторінки.
2. **Web-додаток** – клієнт-серверний додаток, в якому клієнтом виступає в основному браузер, а сервером – web-сервер. Логіка web-додатку розподілена між сервером і клієнтом, зберігання даних здійснюється переважно на сервері, обмін інформацією відбувається у мережі.
3. **HTML (HyperText Markup Language)** - це мова розмітки всіх елементів і документів на сторінці, і їх взаємодія в структурі сторінки.
4. **CSS (Cascading Style Sheets)** – це мова характеристики і стилізації зовнішнього вигляду документа. За допомогою CSS-коду браузер розуміє, як саме необхідно відображати елементи. CSS створює шрифти, кольори, визначає розташування блоків сайту, та інше. Також адаптує один і той же документ в різних стилях, виводить передачу на екран.
5. **JavaScript -** мова, створена оживляти веб-сторінки. Завдання JavaScript – відгукуватися на дії користувача, обробляти натискання клавіш, переміщення курсора, кліки мишкою. JavaScript також дає

можливість вводити повідомлення, посилати запити на сервер, а також завантажує дані без перезавантаження сторінки, і так далі.

## 4. База даних

**База даних (БД) —** це організована структура, призначена для зберігання, зміни й обробки взаємопов’язаної інформації, переважно великих обсягів. Бази даних активно використовують для динамічних сайтів зі значними обсягами даних — часто це інтернет-магазини, портали, корпоративні сайти.

В контексті баз даних варто розглянути поняття СКБД — система управління базами даних. СУБД — це комплекс програмних засобів, необхідних для створення структури нової бази, її наповнення, редагування вмісту і відображення інформації. Найбільш поширеними СУБД є MySQL, PostgreSQL, Oracle, Microsoft SQL Server.

## У своєму проекті використовуватиму MySQL 8.0.



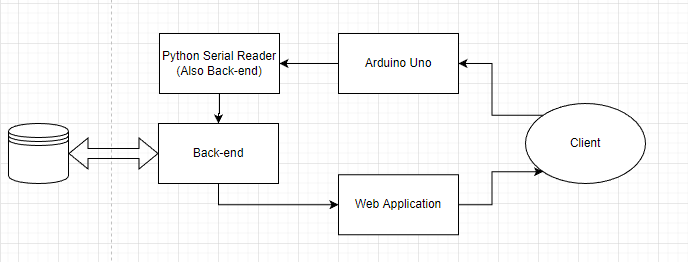
MySQL - це система управління реляційними базами даних (СУБД) Переваги:

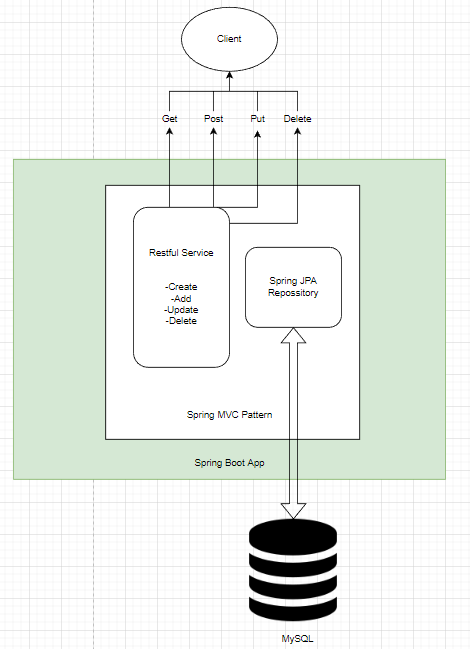
* простота у встановленні та використанні;
* підтримується необмежена кількість користувачів, що одночасно працюють із БД;
* кількість рядків у таблицях може досягати 50 млн.;
* висока швидкість виконання команд;
* наявність простої і ефективної системи безпеки.

# Структурна та функціональна схема проекту з загальним описом складових.

Як видно з діаграми користувач взаємодіє з мікроконтролером та веб- аплікацією, яка може вважатись нашим готовим Front-end і буде сприйматись користувачем. Також у нас є Бекенд, який є невидимим для користувача. Нижче подана докладніша схема Бекенду, де

продемонстровано роботу RestApi:





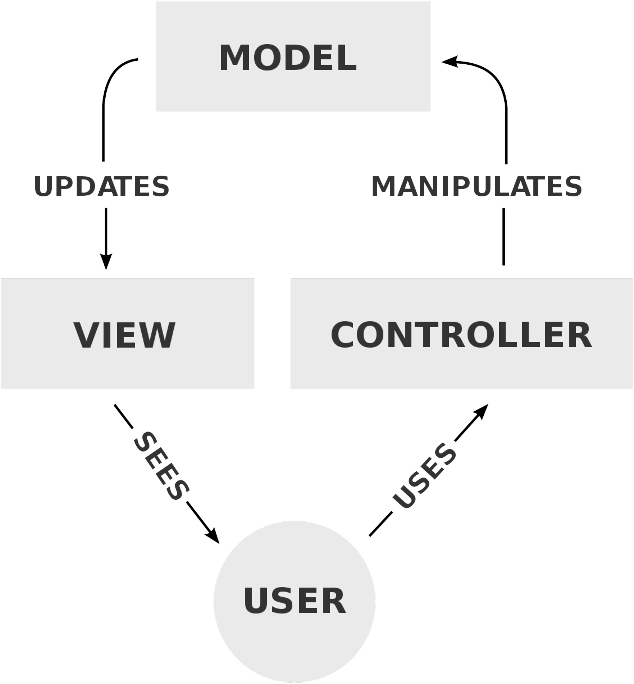
Діаграма RestApi

**MVC-Pattern** - архітектурний шаблон, який використовується під час проєктування та розробки програмного забезпечення.

Цей шаблон передбачає поділ системи на три взаємопов'язані частини: модель даних, вигляд (інтерфейс користувача) та модуль керування.

Застосовується для відокремлення даних (моделі) від інтерфейсу користувача (вигляду) так, щоб зміни інтерфейсу користувача мінімально впливали на роботу з даними, а зміни в моделі даних могли здійснюватися

без змін інтерфейсу користувача.



MVC – Шаблон

* **Вигляд(View)** у мене виконує Front-end.
* **Контролер(Controller)** – його відповідальність полягає в отриманні, зміні та наданні даних користувачеві. По суті, контролер є сполучною ланкою між представленням і моделлю. За допомогою функцій геттера(Getter) і сетера(Setter) контролер витягує дані з моделі та ініціалізує уявлення.
* **Модель(Model)** це Бекенд, який містить всю логіку даних. Тобто з завдання моделі — просто керувати даними. Незалежно від того, чи надходять дані з бази даних, API чи об’єкта JSON, модель відповідає за керування ними. В моєму випадку це база даних MySQL.

# Детальні описи окремих складових

## Апаратна частина

Для реалізації поставленої задачі, один із ключових інструментів, які я використав це пакет програм Proteus Design. Тут я склав схему, де знаходяться такі елементи як

Мікроконтролер Arduino Uno. До нього було під’єднано датчик атмосферного тиску та температури BMP180. Він вимірює абсолютний тиск повітря навколо нього. Він має діапазон

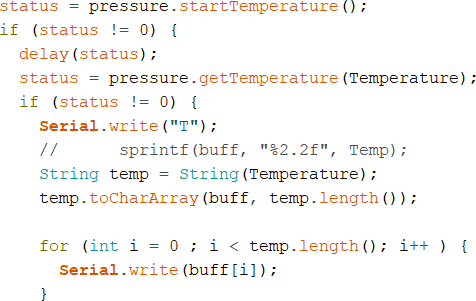
вимірювання від 300 до 1100 гПа з точністю до 0,02 гПа. Він також може вимірювати висоту і температуру. Сам датчик складається з п'єзорезистивного датчика, аналого-цифрового перетворювача і блоку управління з EEPROM і послідовним інтерфейсом I2C (SCL до рядка годинника I2C - на 328 Arduino Uno це Analog 5, SDA до лінії даних I2C – на 328 Arduino Uno це Analog 4). Необроблені вимірювання тиску та температури від датчика BMP180 мають бути компенсовані для температурних ефектів та інших параметрів за допомогою даних калібрування, збережених у EEPROM. Тут ми будемо з'єднувати датчик BMP180 з Arduino Uno. Використовуватиму плату Arduino для зчитування вимірювань температури та барометричного тиску з датчика BMP180 і відображення даних на РК-дисплеї 20х4.

Як і більшість датчиків тиску, BMP180 вимірює абсолютний тиск. Це фактичний тиск навколишнього середовища, який бачить пристрій, який буде змінюватися залежно від висоти та погоди.

Перед вимірюванням тиску необхідно виміряти температуру. Це робиться за допомогою Library

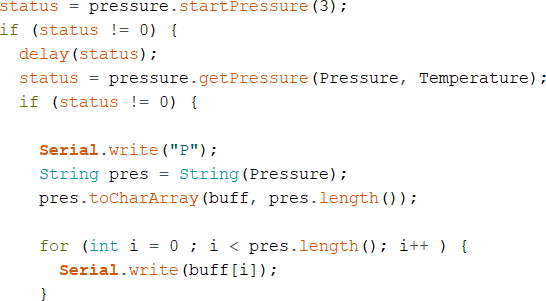


де використаю метод startTemperature() і getTemperature(). Результат виражається в градусах С.

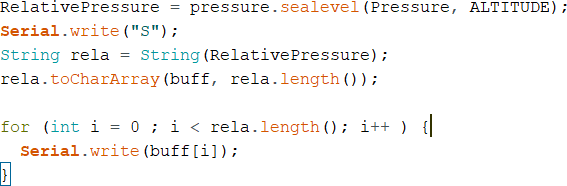


Після того як у вас буде показання температури, ви можете виміряти абсолютний тиск.

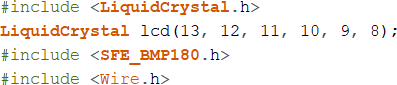
Це робиться за допомогою методів startPressure() і getPressure(). Результат виражається в мілібарах (мб) або гектопаскалях (гПа).



Щоб отримати середній тиск на рівні моря потрібно використати метод sealevel(), де як параметри передаємо наш абсолютний атмосферний тиск та нашу висоту.



Щодо прошивки, який написаний на мові програмування С++ Першим ділом підключив усі потрібні бібліотеки:

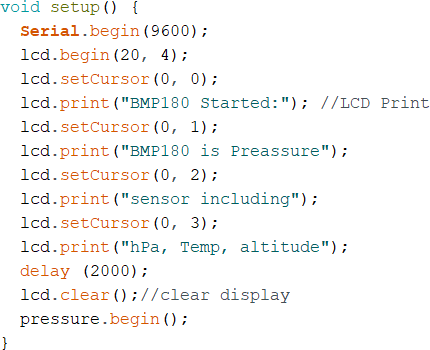


**LiquidCryslal.h** - дозволяє спілкуватися з алфавітно-цифровими рідкокристалічними дисплеями (РКД). Ця бібліотека дозволяє платі Arduino керувати дисплеями LiquidCrystalDisplay (LCD) (РК-дисплеї).

**SFE\_BMP.h -** дозволяє легко спілкуватися з датчиком BMP180 для Arduino. Ця бібліотека не входить до стандартного програмного забезпечення Arduino.

**Wire.h -** Ця бібліотека дозволяє спілкуватися з пристроями I2C/TWI. SDA (лінія даних) і SCL (рядок годинника)

В setup() налаштовуємо роботу дисплею та початок роботи датчика (BMP180):



Дані отримані з датчика буду надсилати через COMPORT.

За допомогою Serial.write(), записує двійкові дані в послідовний порт і ці дані надсилатимуться як байт або серія байтів. Через те що дані отримані з

датчика є типу Double, то я розпарсую їх у toCharArray() та передаю через Serial.write

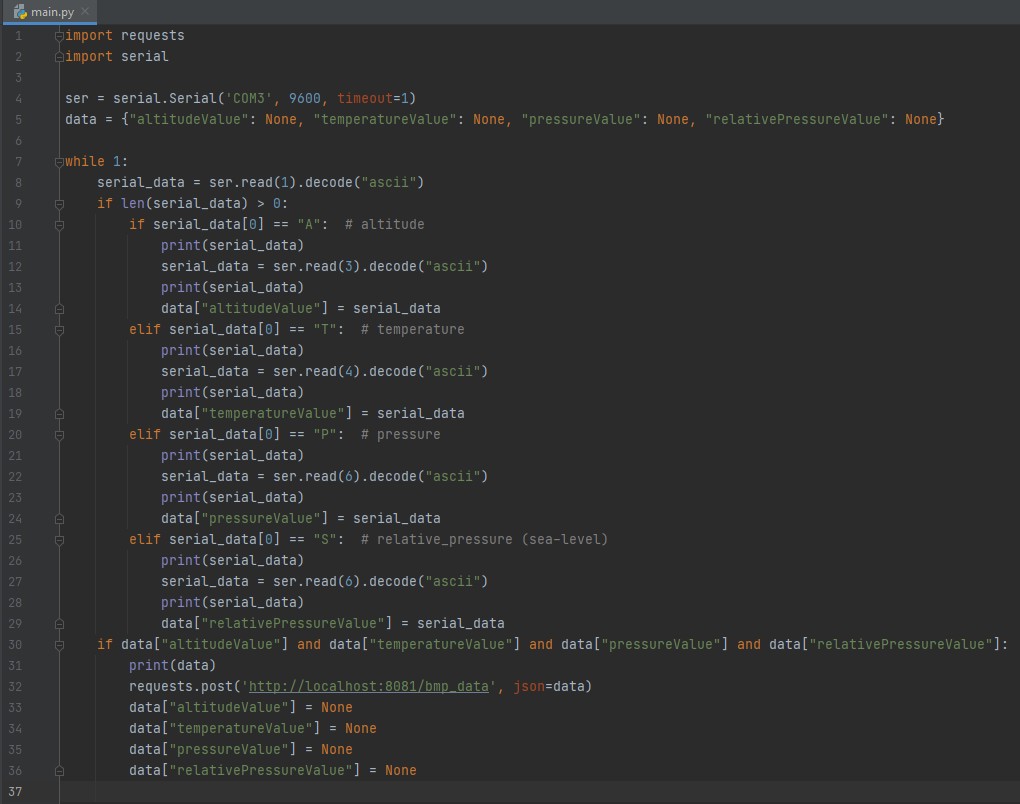
Один з парсів:

Дані надсилаються через COMPORT, де вже програма на бекенді приймає і записує ці значення у базу даних.

## 2.

**Back-end**

Почну відразу з програми написаній на мові програмування Python.



Це програмка, яка зчитує дані з мого Serial порта. Для цього я використовую такі бібліотеки:

Ця бібліотека надає доступ до послідовного порту. Він забезпечує Бекенд для Python , що працює на Windows, OSX, Linux, BSD (можливо, будь-яка система, сумісна з POSIX) і IronPython. Бібліотека під назвою «serial» автоматично вибирає відповідний бекенд.

Відповідно в моїй прошивці кожному значенню, яку було записано

(Температура, Абсолютний Атмосферний Тиск, Середній Тиск Рівня Моря та Висота) має свій унікальний префікс, через якого я можу визначити, що це за дані та їхні значення.



Requsts(Запити) - дозволяють надзвичайно легко надсилати запити

HTTP/1.1. Немає необхідності вручну додавати рядки запиту до ваших URL- адрес або кодувати свої дані у формі PUT чи POST, а використати цей

JSON метод, який запише та надішле дані на бекенд.



Back-end написаний на мові програмування Java.

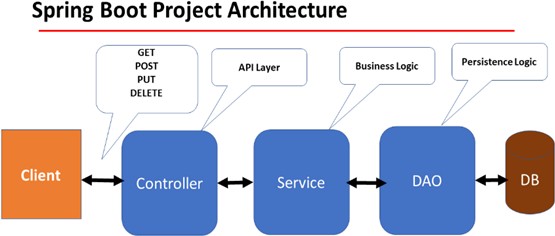
Використовувався Spring Boot фреймворк за допомогою якого було реалізовано весь необхідний функціонал для завдання.

Першим кроком за допомогою Maven створюю проект.

Наступним кроком буде створення pom.xml файлу, де відбудеться оголошення залежностей (Maven Dependencies).

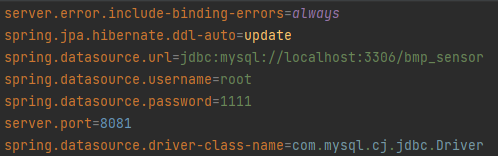
Dependencies — це фреймворки, які вам знадобляться для розробки вашого проекту.

Далі перейдемо до створення Spring Boot REST API



Структура проекту Spring Boot REST API

Для початку перейдемо у проект і налаштуємо базу даних MySQL. У файлі **application.properties** і додаєм до нього властивості.

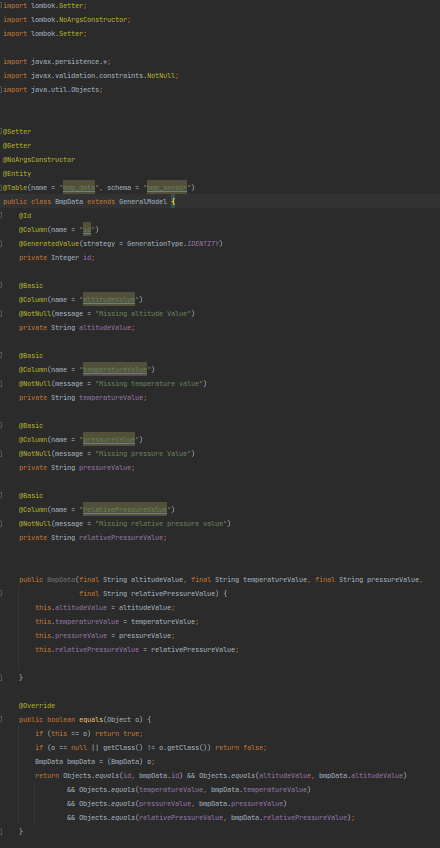


## application.properties

Щоб конфігурація була коректною нам залишається створити базу даних.

Для цього візьмемо реляційну реляційну базу даних та створимо бд з назвою bmp\_sensor. Тепер наша аплікація буде мати доступ за допомогою вказаного шляху, користувача та паролю.

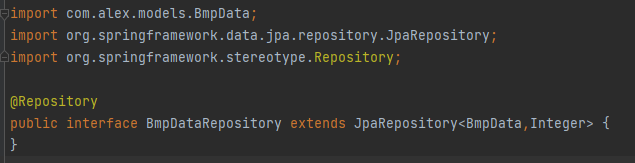
Наступним кроком буде створення об’єкта JPA. Створюємо пакет під назвою models, а ньому файл BmpData.



## BmpData.java

Використовуємо анотацію Lombok @Data, щоб зменшити шаблонний код (гетери/сетери)

Далі потрібно створити репозиторій для доступу до даних **BmpData** з бази даних. Інтерфейс JpaRepository визначає методи для всіх операцій CRUD над сутністю та реалізацію.



## BmpDataRepository.java

**BmpDataService** – у цій частині знаходиться бізнес логіка аплікації. Сервіс Spring визначається як файл класу, який містить анотацію @Service і дозволяє додавати бізнес-функціональні можливості.

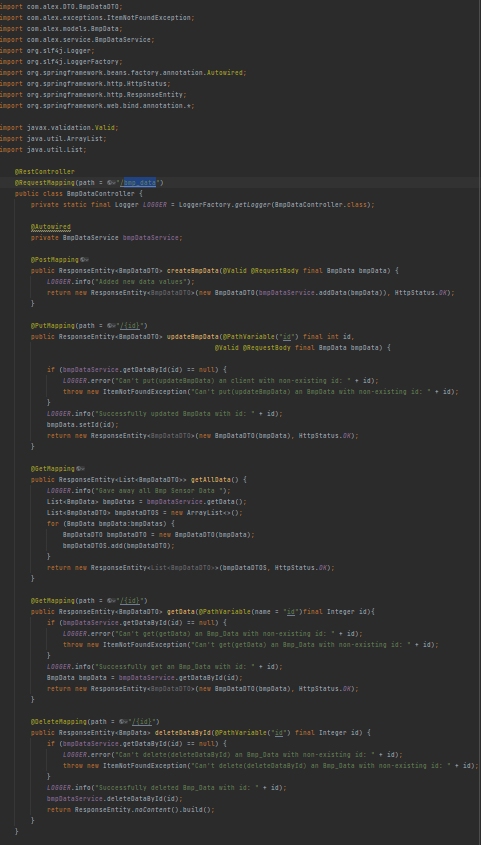
Анотація використовується з класами, які надають ці бізнес- функціональні можливості.



## BmpDataService.java

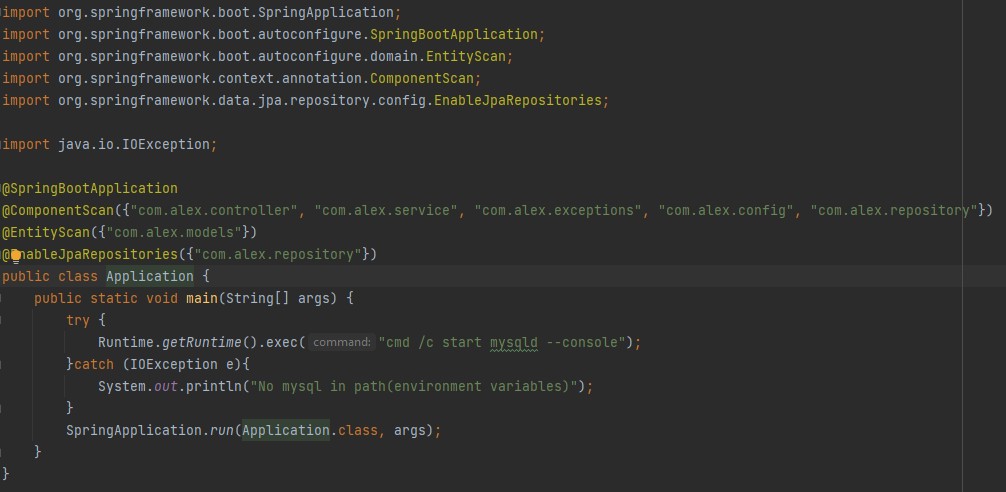
**BmpDataController -** у Spring методи контролера є кінцевою точкою призначення, тобто працює з ендпоїнтами, до яких може досягнути веб-запит. Після виклику метод контролера починає обробляти веб- запит, взаємодіючи з рівнем сервісу, щоб завершити роботу, яку необхідно виконати. Також контролер містить в собі об’єкт сервісу,

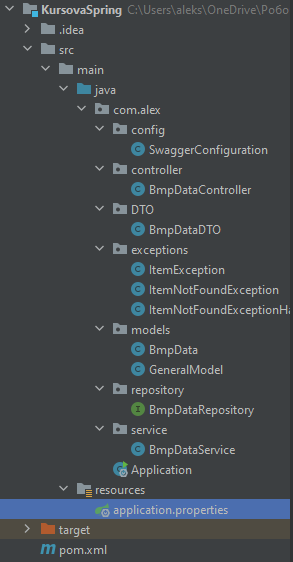
таким чином взаємодіючи з ним і вказує йому його роль.



## BmpDataController.java

**Application –** цей файл допоможе запустити аплікацію та відобразить в консолі результат роботи аплікації та помилки, які виникнуть.



Щодо структури проекту, то вище було згадано, що проект реалізовано на MVC-Pattern (Model-View-Controller)

Структура мого проекту

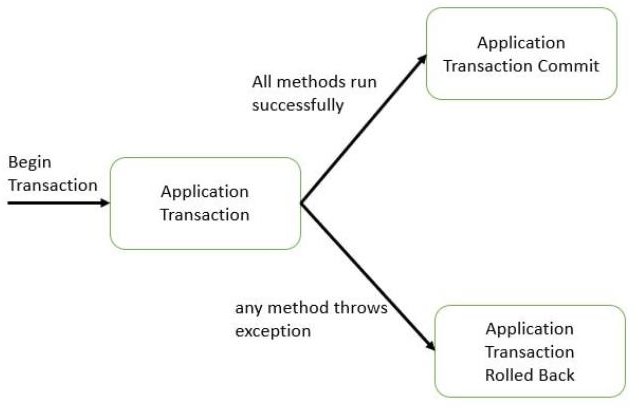
## БД

В аплікації, яку я описував вище вже було налаштовано підключення до MySQL бази даних та можливість проводити транзакції.

Відповідно після початку транзакції, може відбутися два варіанти А) Всі транзакції пройшли успішно.

Б) Якийсь з методів викинув помилку.

Нижче продемонстровано схему до вище написаного:



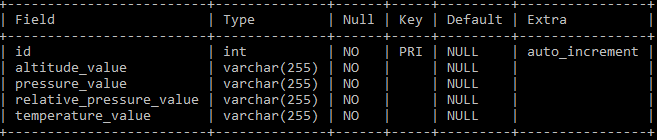
В табличці є 5 полів (1 поле id та 4 поля з даними з датчика):

-altitude\_value (висота)

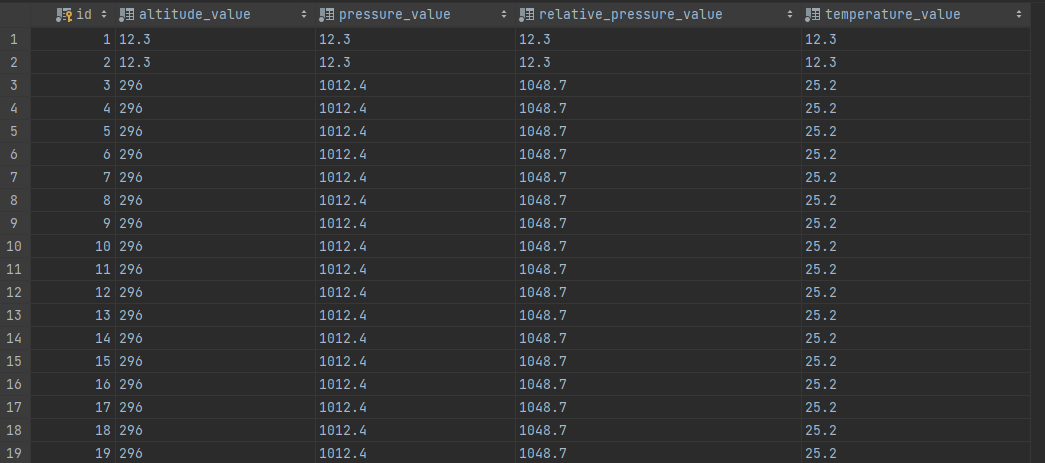
-pressure\_value (абсолютний атмосферний тиск)

-relative\_pressure\_value (середній тиск рівня моря)

-temperature\_value (температура)



Записані дані у таблиці:



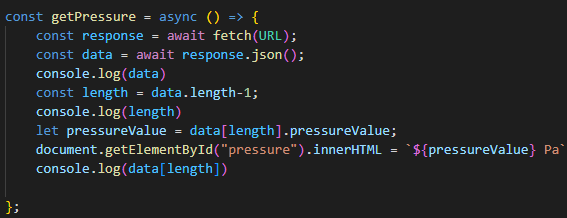
## Веб додаток

Веб додаток складається з HTML(HyperText Markup Language), CSS та JavaScript.

У цій частині реалізовано Front-end частину. Це основна складова для кінцевого користувача, оскільки, вона дає змогу побачити, протестувати реалізований проект та отримати результати на красивій і зручній картинці.

Використовую який Fetch API, який надає інтерфейс JavaScript для доступу та керування частинами конвеєру HTTP, такими як запити та відповіді. Він також забезпечує глобальний fetch()метод, який

забезпечує простий, логічний спосіб асинхронного отримання ресурсів у мережі.



Один з fetch()

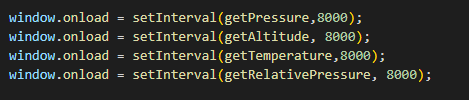
Також використовується Window: load event Також використовується **Window.onload** event і результатом

зображення для window.onload, тобто простішими словами, ми

можемо сказати, що подія завантаження вікна запускається щоразу, коли веб-сторінка успішно запущена. Подія Onload використовується для запуску сценарію після повного запуску його посуті (що включає файли CSS, зображення та сценарії) сторінки.

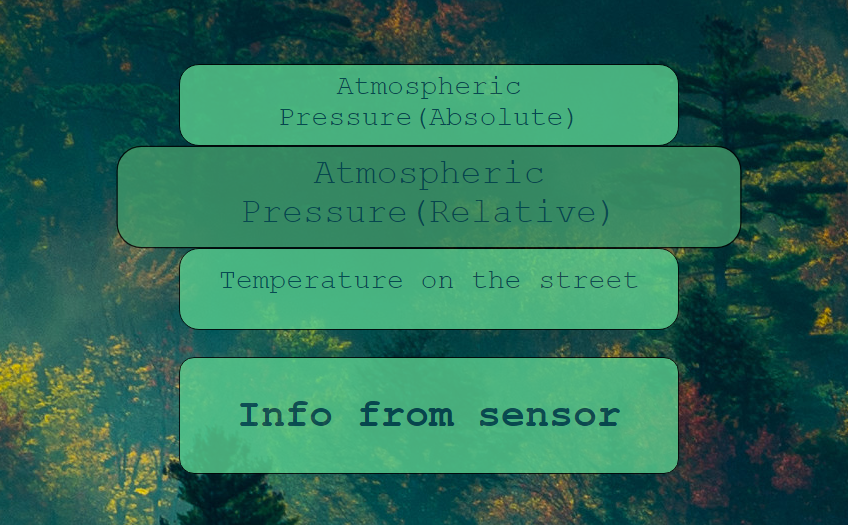
Це на відміну від DOMContentLoaded, який запускається відразу після завантаження DOM сторінки, не чекаючи, поки ресурси закінчаться.

До window.onload я виставляю інтервал у 8сек оновлення даних на сторінці, оскільки, на мікроконтролері дані зчитуються один раз у вісім секунд.

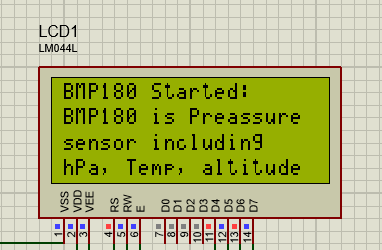


# Загальна ілюстрація працездатності проекту

(Загальна ілюстрація працездатності проекту (скріншоти з веб-додатку і тп.))



Веб-додаток для виводу даних



LCD перші 4 секунди

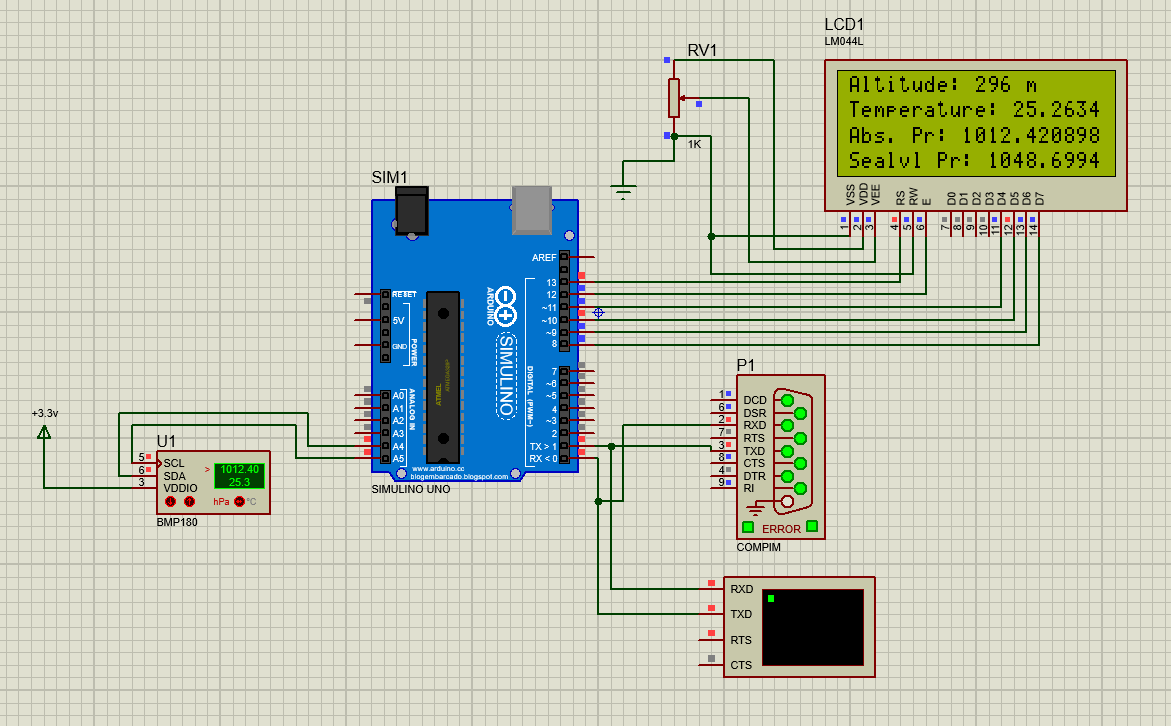
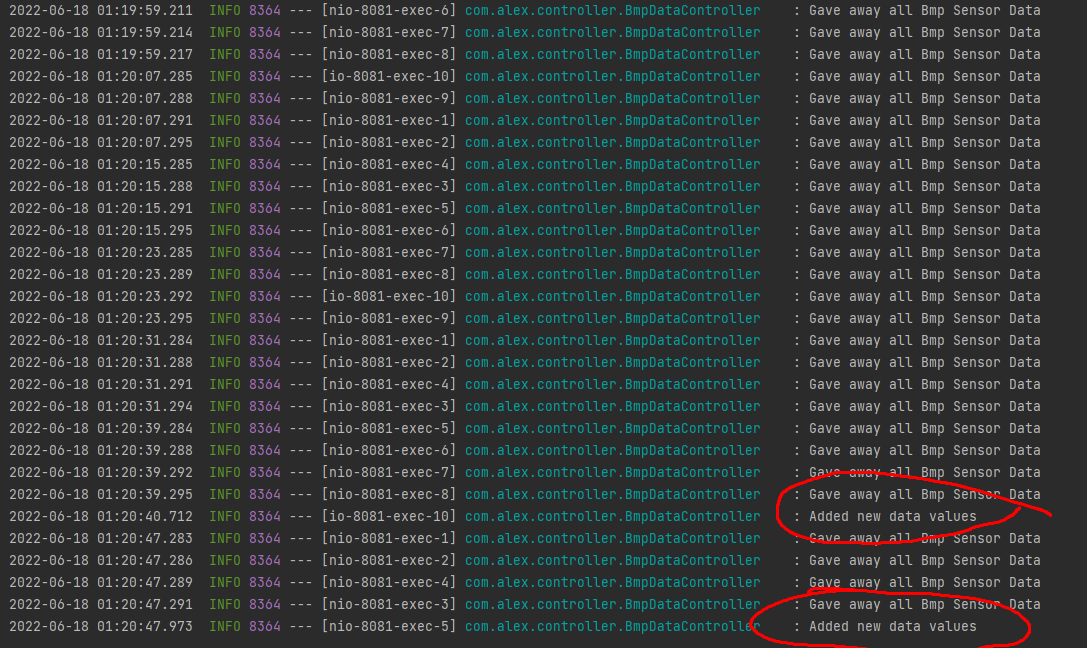


Схема проекту



Робочий Back-end

# Висновок

Виконуючи дану курсову роботу, я навчився збирати схему у програмі Proteus та збирати інформацію з Arduino-датчиків, імплементувати взаємодію з мікроконтролером, оброблення даних, які були отримані з

нього. Також зміг закріпити знання з таких дисциплін як веб-програмування, бази даних, мікроконтролери, схемотехніку.

Також, зміг поглибити свої знання у пізнанні будови барометра, його роботи та зумів імплементувати його у житті.