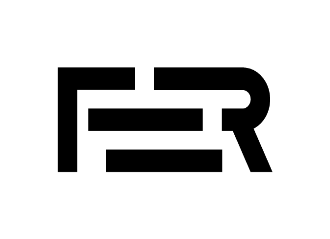
****

**Diplomski studij**

**Informacijska i komunikacijska tehnologija**

**Računarstvo**

Telekomunikacije i informatika  
Obradba informacija  
Računalno inženjerstvo

**Internet stvari  
  
Mjerenje zagađenja Jadranskog mora**

**Projekt**

**Tin Barić**

**Martin Požeg**

**Ivan Ceković**

**Dominik Marjanović**

**Branimir Pehar**

**Vjekoslav Matečić**

**Ak.g. 2019./2020.**

**Sadržaj**

[1. Uvod 3](#_Toc40821957)

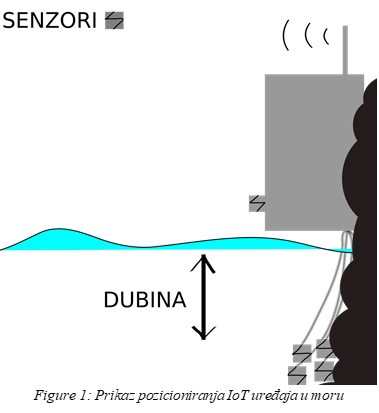
[2. Opis rješenja 7](#_Toc40821958)

[3. IoT platforma 8](#_Toc40821959)

[4. Korisničke aplikacije 10](#_Toc40821960)

# Uvod

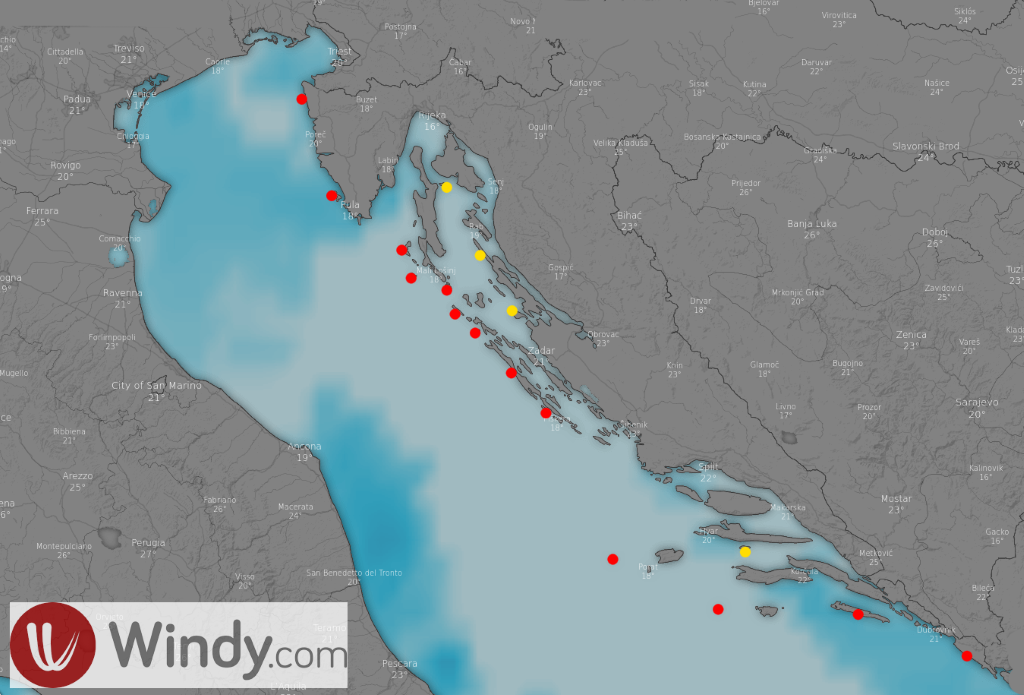
Jadransko more svakoga je dana izloženo novim ekološkim rizicima. Povećanje industrijske aktivnosti, povećani broj turista i brodova negativno utječe na složeni biološki svijet. Pristupačnost novih tehnologija, uređaja i senzora omogućila bi automatizirani nadzor i praćenje razine zagađenja Jadranskog mora. Korištenje automatiziranog sustava imalo bi i dodatnu prednost ranog upozoravanja na neposredno onečišćenje te omogućilo njegovo pravovremeno zbrinjavanje. Korištenjem modernih IoT tehnologija otvara se mogućnost povezivanja velikog broja senzora i uređaja koji bi u realnom vremenu pratili i bilježili razne parametre mora. Svi snimljeni podaci dostupni  su u realnom vremenu, što olakšava posao odgovornim službama i smanjuje potrebu za izlazak na otvoreno more, prikupljanje uzoraka i konačno analizu istih. Slika 1.1. prikazuje primjer pozicioniranja jednog takvog IoT uređaja na stijenu ili rub otoka. Na slici je jasno prikazano da se određeni senzori nalaze ispod površine mora na određenoj dubini.



Slika 1.1 Pozicioniranje IoT uređaja u moru

Prijedlog lokacija IoT uređaja:

* Vanjska strana vanjskih otoka (crvene točke na slici)
* Vanjske strane unutarnjih otoka (žute točke na slici)
* Sjeverni Jadran trpi veće zagađenje pa treba gušći razmještaj senzora (iako se možda otkrije da ova hipoteza uopće ne vrijedi te da zagađenje dolazi iz Mediterana)



Slika 1.2 Prijedlog lokacija IoT uređaja

Prijedlog dubine senzora:

* Optimalnu dubinu senzora je bitno odrediti zbog različite topljivosti plinova
* Površinski sloj se brzo grije i hladi i podložan je varijacijama umnažanja planktona koji apsorbira dio zagađivača
* Svi senzori trebali bi biti na više manje istoj dubini zbog konzistentnosti mjerenja
* Zbog stalnog miješanja valova dubina od 0 m (skroz uz površinu) nije baš vjerodostojna (npr. neki talog na vrhu vode od nafte ili nečega može zapljusnuti senzor)
* Razmotrimo li dubinu od -2 m, nema toliko temperaturnih promjena (iako se radi o otvorenom moru) i manja je vjerojatnost pojave vala od 2 m pa su očitanja uvijek u vodi
* Ispod toga na -3 m je konstantno hladnije i očitanja bi bila stabilnija
* Razmotrimo li dubinu od -5 m, na toj dubini je stabilnije miješanje struja s površine i očitanja su stabilnija (da neka struja ili val ne poremeti kontinuitet praćenja)
* Preporučena dubina senzora iznosila bi od -2 do -5 m u odnosu na razinu mora (uz mogućnost preciznijeg određivanja optimalne dubine s pomoću stvarnih podataka)

Prijedlog dodatnih senzora:

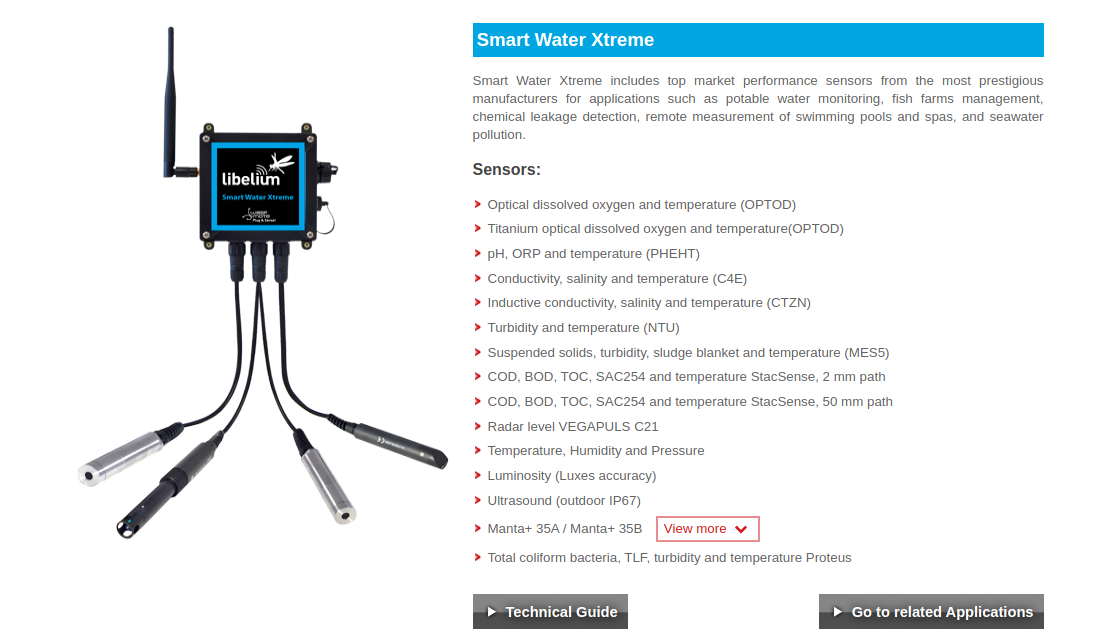
* Količina sunca (količina svjetlosti)
* Spektrometar (omogućavaju detekciju cvjetanja planktona, npr. <https://www.oceaninsight.com/products/spectrometers/> )
* Senzor za očitavanje životnih signala (npr. IR za toplokrvnjake i mikrovalni senzori)

Prijedlog fizičke sigurnosti:

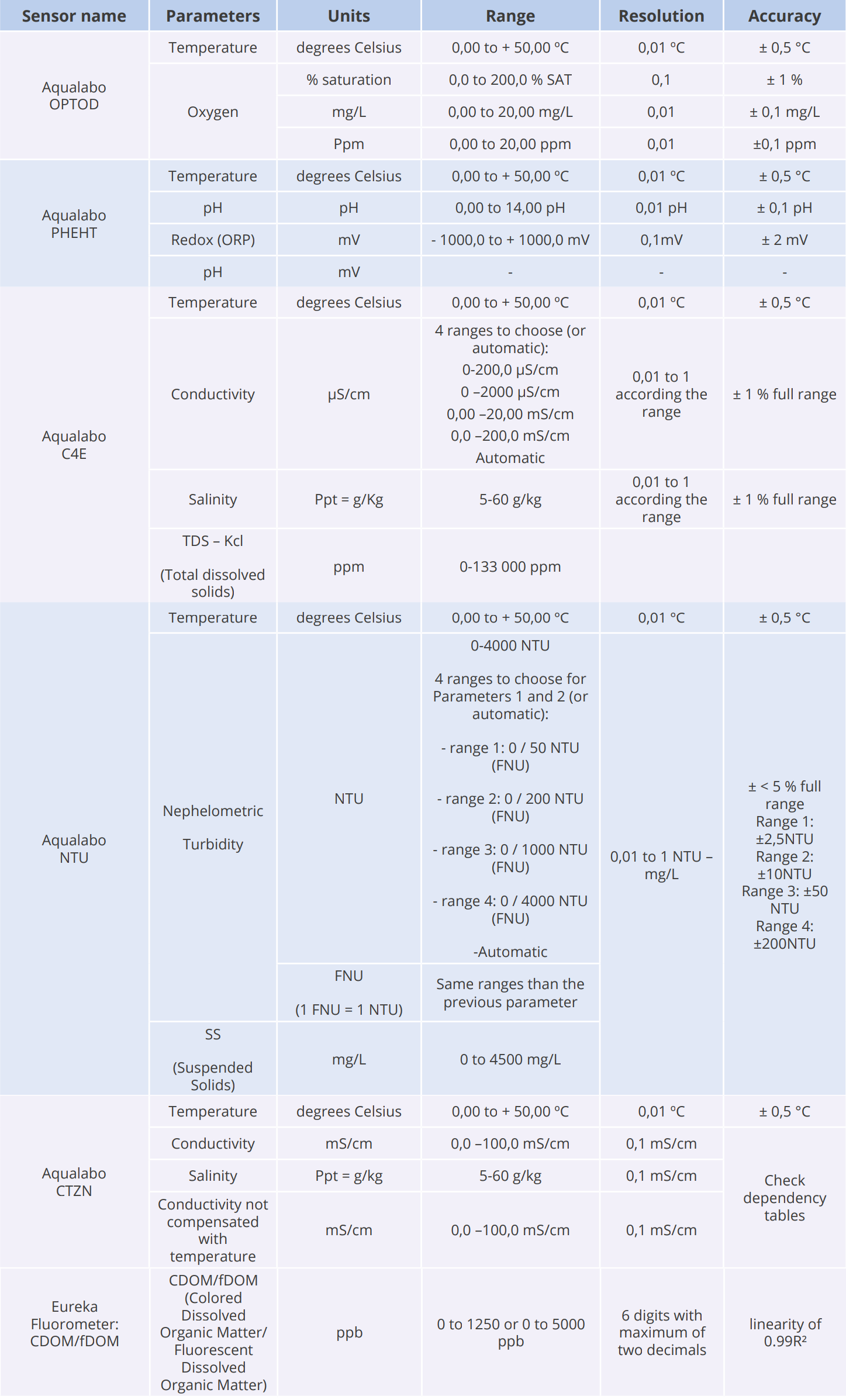
* Zamaskirati sve uređaje da ne privlače pozornost

Prijedlog dostupnosti podataka:

* Omogućiti pristup prikupljenim podacima istraživačima ili komercijalnu pretplatu na podatke svima koji ih žele dodatno analizirati



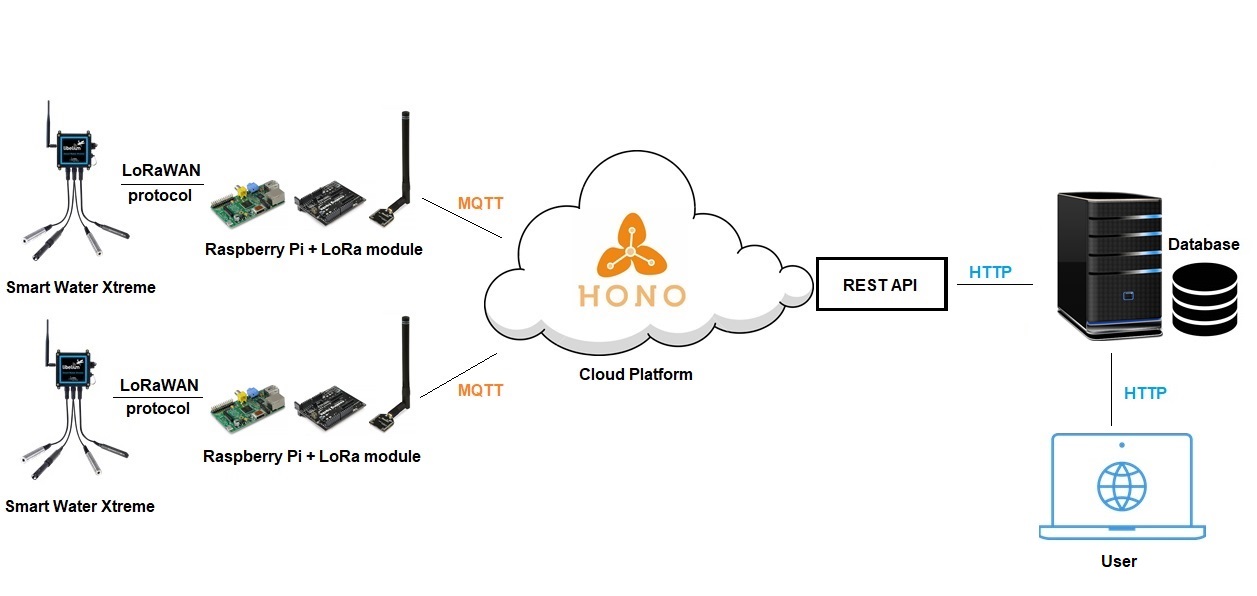
Slika 1.3 Senzor Smart Water Xtreme



Slika 1.4 Usporedba različitih senzora

# Opis rješenja

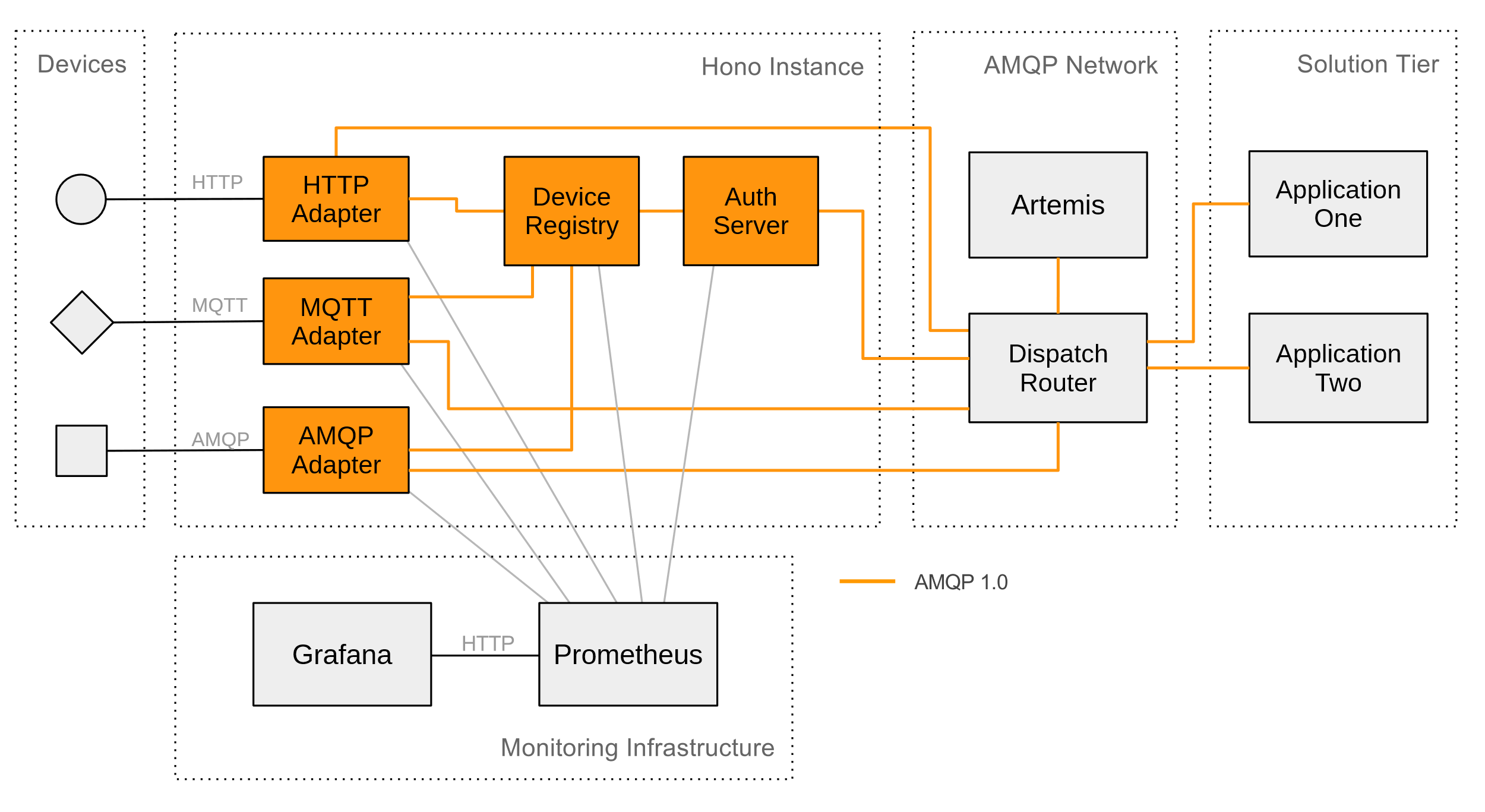
* Odabrani uređaj je Libelium Smart Water Xtreme koji ima nekoliko priključaka za senzore. Postoji puno raznih senzora koji se mogu priključiti, no odabrano ih je samo 6 kao što je vidljivo na slici 1.4. Oni mjere razne vrijednosti od temperature vode, provodnosti, slanoće, pH vrijednosti, količine otopljenih krutih tvari, količine otopljenih organskih tvari do zamućenja vode. Senzori direktno komuniciraju sa uređajem koji putem LoRaWAN mrežnog protokola odašilje podatke do mrežnog prilaza. U ovom projektu senzorske vrijednosti su simulirane preko skripte sensors.py. Svaki senzor predstavljen je klasom unutar koje su definirane konstante (minimalna vrijednost iznosa očitanja, maksimalna vrijednost iznosa očitanja, rezolucija te preciznost) po uzoru na podatke sa slike 1.4. te metode računanja slučajnih vrijednosti. Mrežni prilaz sastoji se od Raspberry Pi uređaja na koji je spojen LoRa modul kako bi mogao komunicirati sa senzorima. Raspberry Pi zatim šalje podatke putem MQTT protokola na IoT platformu (komunikacija između komponenti teče preko radnog okvira Mosquitto).
* Korištena IoT platforma je Eclipse Hono koja služi kao MQTT broker. Prima vrijednosti podataka sa senzora te ih šalje na web aplikaciju.
* Korisnička aplikacija je u obliku web aplikacije. Web aplikacija se pretplati na IoT platformu kao klijent te prima nove vrijednosti senzora koje zatim prikazuje na svom sučelju.



Slika 2.1 Arhitektura sustava

# IoT platforma

* Za izradu ovog projekta koristi se platforma otvorenog koda Eclipse Hono. Komponente Eclipse Hono instance su ostvarene preko više mikrousluga. Arhitektura komponenata prikazana je na slici 3.1.



Slika 3.1 Komponente instance Eclipse Hono platforme

* Platforma čuva meta podatke o uređajima u registru uređaja. Uređaji se pri registraciji raspodjeljuju u svoje logičke dosege ili tenante. Unutar tenanta mu se postavlja jedinstveni ID. Platforma ima vlastiti autentikacijski server koji autenticira Hono komponente i izdaje tokene utvrđujući identitet i ovlaštenja. Adapteri za HTTP, MQTT i AMQP protokole izlažu Hononove API-je za telemetriju i događaje. Podaci poslani na platformu šalju se na prikaz nizvodnim aplikacijama preko „sjevernih“ (engl. *northbound*) sučelja koji su dio AMQP mreže.
* Postoje 2 načina korištenja Hono platforme. Jedan način je pokrenuti lokalnu instancu pomoću kontejnera. Drugi način, zamišljen samo za testiranje je javno dostupna i održavana od strane Eclipse organizacije, Hono Sandbox. U ovom projektu koristi se drugi način.
* Koriste se naravno registar uređaja i autentikacijski server i, a vlastito razvijenom aplikacijom spajamo se na sjeverna sučelja platforme.
* Platformi pristupamo preko tenanta (MY\_TENANT) koji je zapravo naš „ID“ MQTT brokera. Tenanta smo „stvorili“ pomoću curl naredbe:

curl -i -X POST http://hono.eclipseprojects.io:28080/v1/tenants

„ID“ naše skripte predstavlja konstanta MY\_DEVICE kao jedinstvena s obzirom da je riječ o simuliranim podacima. Dakako, da su postojali fizički uređaji kreirali bismo brojčano toliko senzora unutar tenanta koliko bi takvih uređaja za opažanje bilo. Kreirani uređaj MY\_DEVICE dobili smo sljedećom curl naredbom:

curl -i -X POST [http://hono.eclipseprojects.io:28080/v1/devices/<tenant\_id](http://hono.eclipseprojects.io:28080/v1/devices/%3ctenant_id)>

Nakon kreiranja uređaja potrebno je bilo još kreirati lozinku koja je predstavljena konstantom MY\_PWD te se takav uređaj mogao koristiti (pogledati isječke koda niže).

MQTT\_ADAPTER\_IP="hono.eclipseprojects.io"

MY\_DEVICE="636ea23e-45db-4106-b1c7-bdeb14f6861c"

MY\_TENANT="4082acd7-4e7e-4c3d-b2b5-d9e8e056a147"

MY\_PWD="pass"

TIME\_TO\_SLEEP=900

TIME\_TO\_SLEEP\_BETWEEN\_SENDING\_MESSAGE=2

def send(location, value):

print("Sending... " + location + " -> " + value)

cmd = "mosquitto\_pub -h " + MQTT\_ADAPTER\_IP + " -u " + MY\_DEVICE + "@" + MY\_TENANT + " -P " + MY\_PWD + " -t telemetry -m '{\"" + location + "\": " + value + "}'"

os.system(cmd)

time.sleep(TIME\_TO\_SLEEP\_BETWEEN\_SENDING\_MESSAGE)

Isječak koda 3.2

# Function that checks if the parsed temperature is within range

def isTempWithinRange(temp):

if (temp > 10 and temp < 30):

return True

return False

# Function that sends data from a virtual device

def virtualDevice(name):

# Data from sensor AqualaboOPTOD

if (IS\_DHMZ\_PARSING\_ACTIVE):

parsedValue = parseDHMZ(name)

try:

if (isTempWithinRange(parsedValue)):

send(name + "/" + "Temperature", str(parsedValue))

except:

print("Can't parse DHMZ temperature")

else:

send(name + "/" + "Temperature",str(myAqualaboOPTOD.getTemperature()))

send(name + "/" + "OxygenSaturation", str(myAqualaboOPTOD.getOxygenSaturation()))

send(name + "/" + "OxygenMgL", str(myAqualaboOPTOD.getOxygenMgL()))

send(name + "/" + "OxygenPpm", str(myAqualaboOPTOD.getOxygenPpm()))

Isječak koda 3.3: Primjer slanja podataka sa senzora AqualaboOPTOD

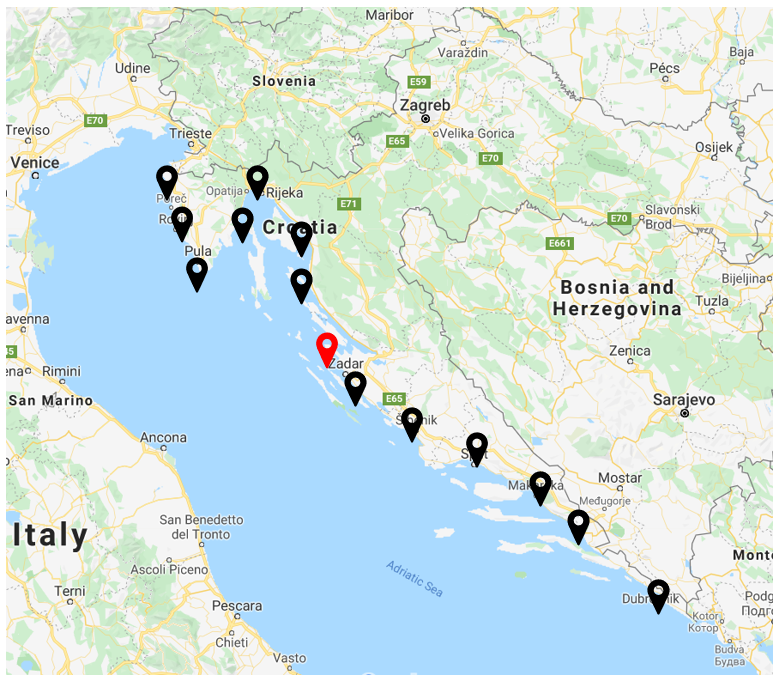
# Korisničke aplikacije

Korisničke aplikacije:

* Web aplikacija (postoji mogućnost progresije na PWA - *Progressive Web Application*)

Tehnologije:

* Spring, Thymeleaf, JavaScript, HTML5, CSS, (baza podataka - NoSQL – MongoDB)



|  |  |
| --- | --- |
| Zasićenost kisikom | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| Nitrati | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| pH | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| Razina soli | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| Temperatura | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

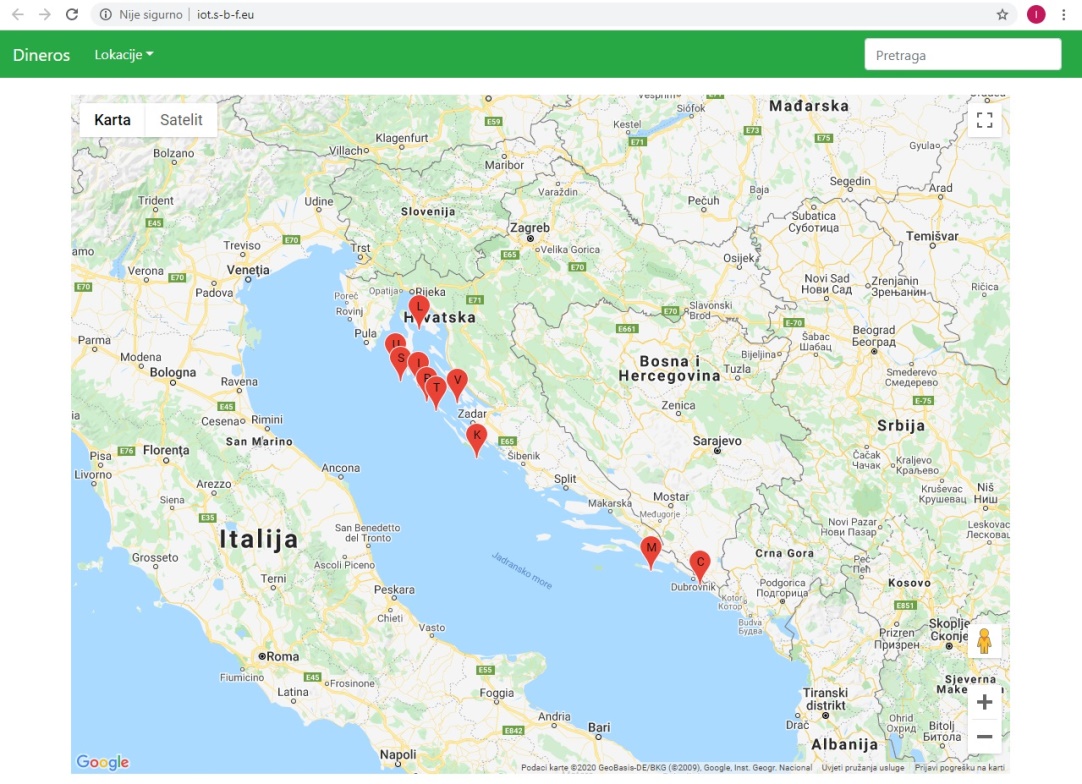
Senzor

Zadar-01

*Slika 4.1 Nacrt Web-aplikacije*

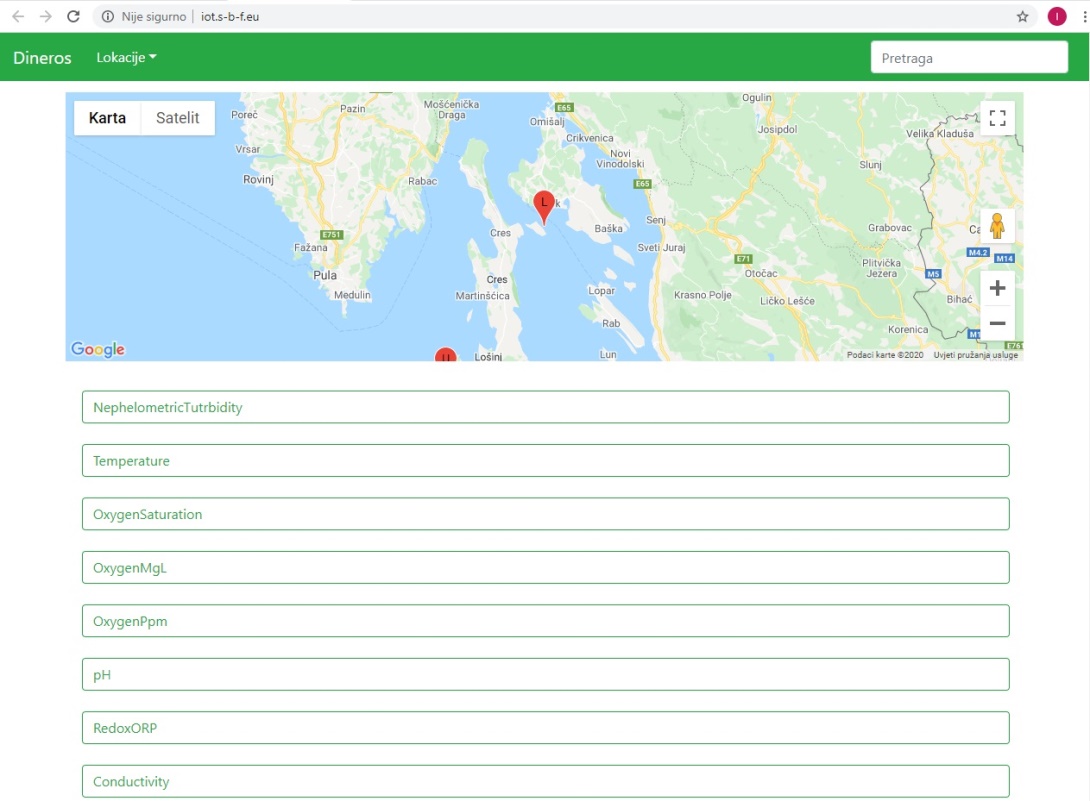
Izgled implementirane web aplikacije:

* Početni zaslon



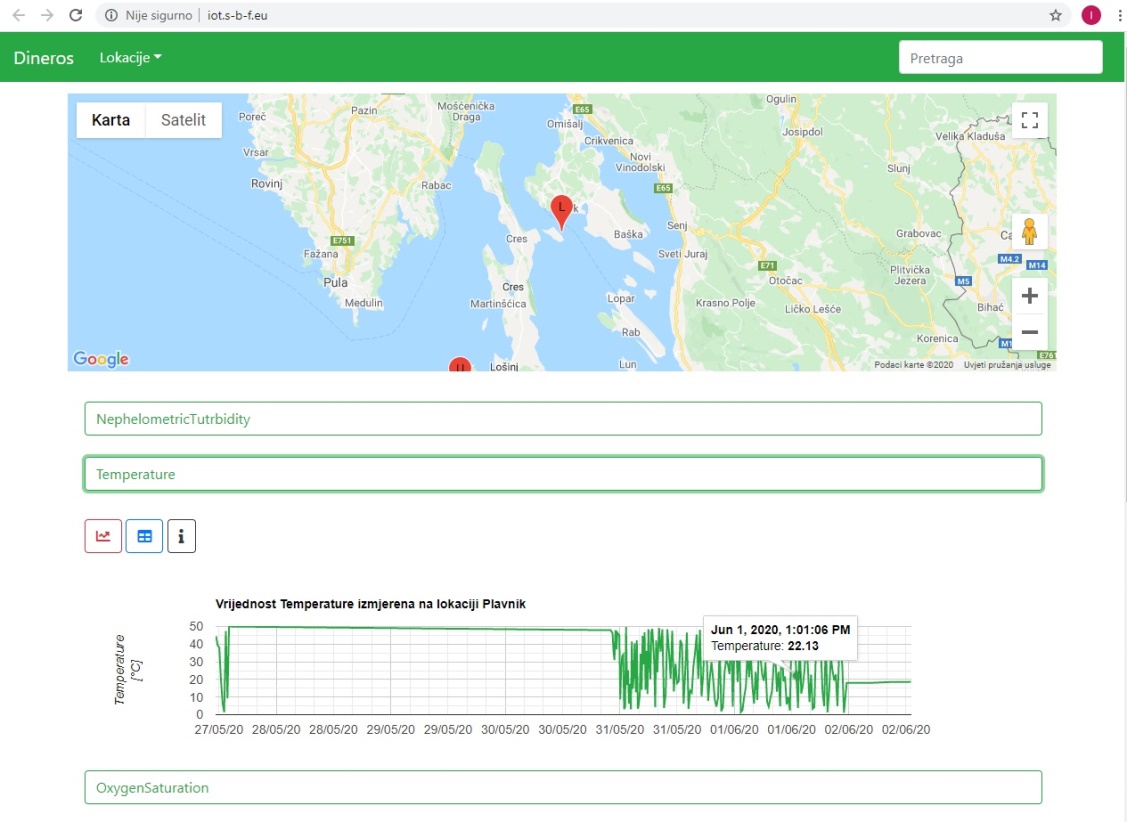
*Slika 4.2. Početni zaslon*

* Izbor senzora (Oznaka L- lokacija Plavnik)



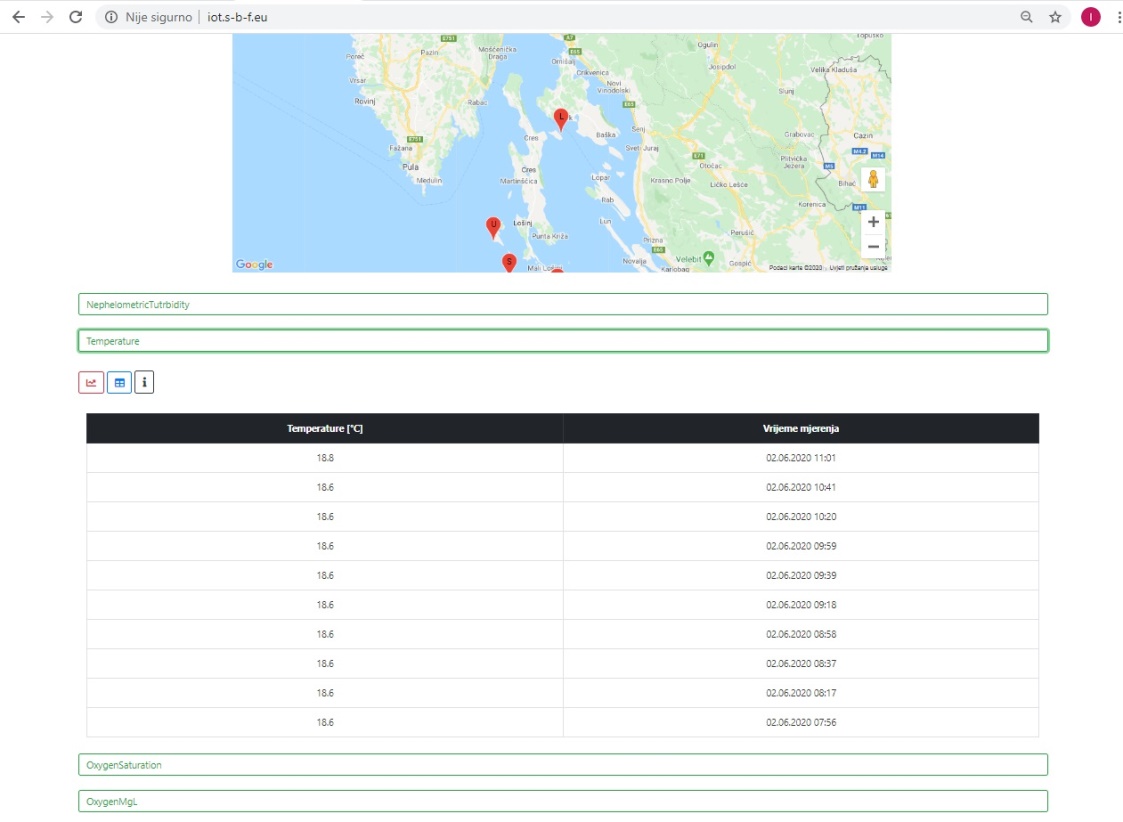
*Slika 4.3. Izbor senzora na lokaciji Plavnik*

* Odabir opcije Temperature i pozicioniranje na graf vrijednosti



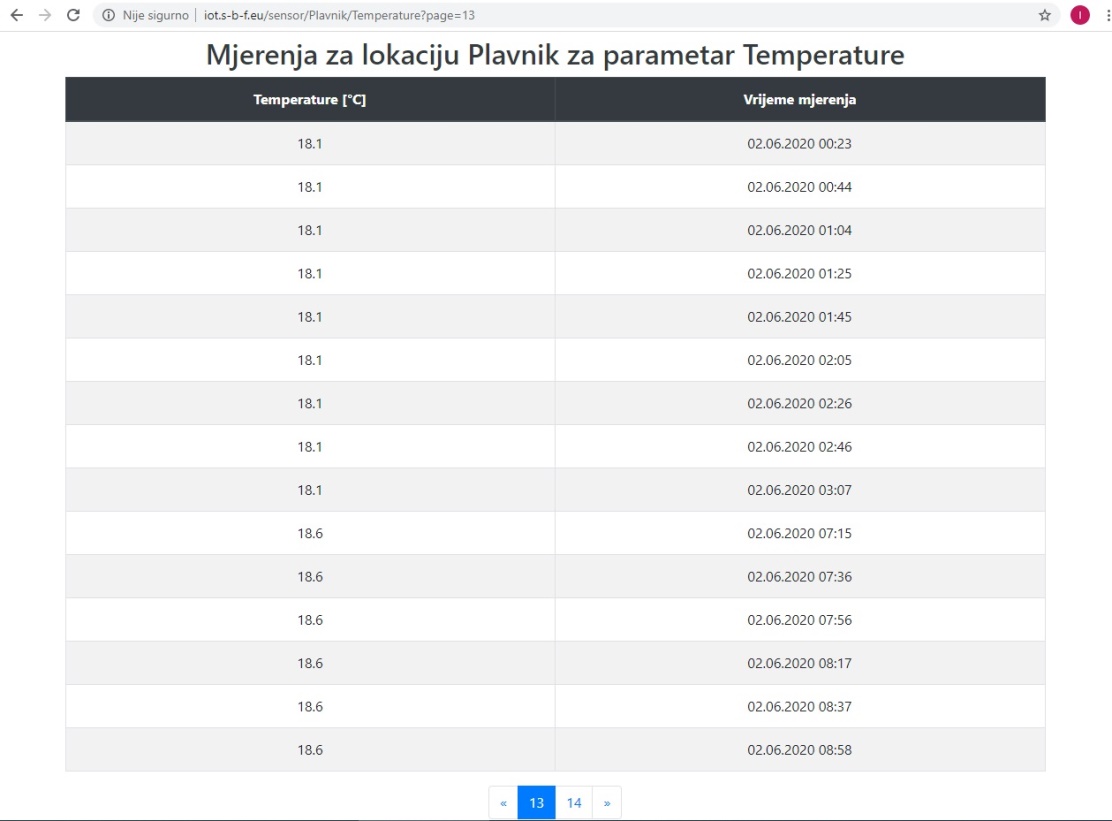
*Slika 4.4. Odabir opcije Temperature*

* Odabir prikaza podataka u obliku tablice



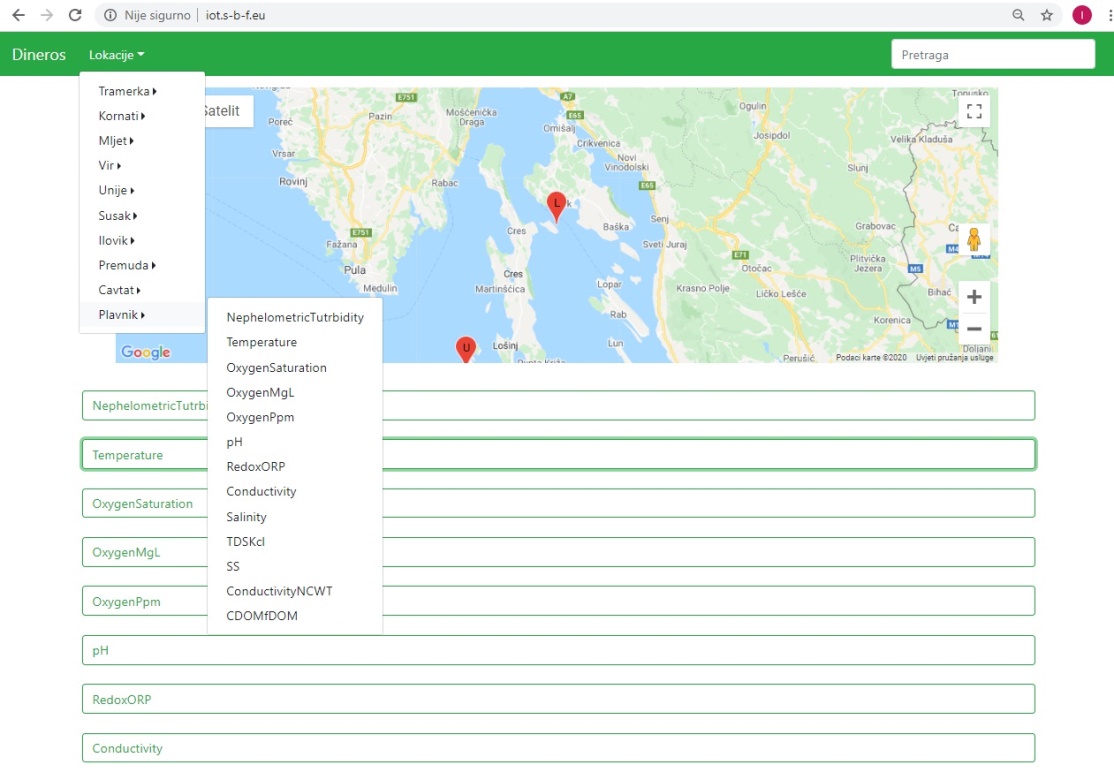
*Slika 4.5. Odabir prikaza podataka u obliku tablice*

* Odabir gumba „i“- informacije o povijesti zabilježenih vrijednosti



*Slika 4.6. Odabir povijesti zabilježenih vrijednosti*

* Odabir lokacije putem izbornika na početnom ekranu



*Slika 4.7. Odabir lokacije putem izbornika*