JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

INSITUT TEKNOLOGI SEPULUH NOPEMBER

# **USULAN TUGAS AKHIR**

# IDENTITAS PENGUSUL

**Nama** : **Gigih Prio Nugroho**

**NRP** : **5109100002**

**Dosen Wali** : **Isye Arieshanti, S.Kom., M.Phil.**

# JUDUL TUGAS AKHIR

Sistem Pendeteksi Dini Banjir Menggunakan Sensor Kecepatan Air dan Sensor Ketinggian Air pada Mikrokontroler Arduino

*Early Detection System of Flood using Water Flow Sensor and Water Level Sensor in Arduino Microcontroller*

# URAIAN SINGKAT

Seperti yang kita ketahui, Indonesia adalah salah satu negara yang memiliki curah hujan tinggi. Curah hujan yang sangat tinggi tersebut seringkali menimbulkan masalah bagi masyarakat. Salah satunya adalah bencana banjir. Diperlukan suatu alat atau aplikasi untuk memonitor gejala awal terjadinya banjir. Salah satunya dengan mengukur kecepatan air beserta ketinggiannya.

Mikrokontroler Arduino Uno merupakan sebuah mikrokontroler ATMEGA328 yang memiliki *flash memory* dan RAM [[1](#Ard13)]. Mikrokontroler ini dapat dihubungkan dengan sensor yang dapat mendeteksi kondisi sekitarnya, salah satunya sensor kecepatan air dan sensor ketinggian air.

Aplikasi pemantauan kecepatan air ini dirancang untuk melakukan pemantauan kecepatan air dan ketinggian air untuk mendeteksi dini potensi banjir di titik yang sudah ditentukan. Data yang dibutuhkan pada aplikasi ini adalah besar kecepatan air, ketinggian air dan juga posisi lokasi geografis yang diukur kecepatan airnya. Dibutuhkan sensor kecepatan air dan sensor ketinggian air Arduino dalam perancangan arsitektur sistem ini.

Data kecepatan air, ketinggian air dan posisi lokasi disimpan di sebuah *database server*. Tujuannya agar data kecepatan air, ketinggian air dan posisi geografis dapat dibaca dalam sebuah aplikasi web. Untuk mengirim data ke *server,* digunakan protokol HTTP dengan fungsi GET. Pengiriman data didahului dengan melakukan koneksi ke *server*. Jika *server* tersedia, maka proses selanjutnya adalah melakukan pengiriman data dengan fungsi GET.

Aplikasi web nantinya digunakan oleh *end user*. Untuk mendapatkan data kecepatan air dan ketinggian air, aplikasi web harus melakukan *request* ke *server.* Ketika aplikasi mengirim data ke *server,* secara otomatis *server* akan mengirim respon ke aplikasi klien.

Tujuan aplikasi ini yakni untuk membantu memonitor kondisi kecepatan air dan ketinggiannya di tempat yang rawan terjadi banjir. Diharapkan dengan aplikasi ini dapat dilakukan pencegahan secara dini dari bencana banjir.

# LATAR BELAKANG

Banjir merupakan peristiwa terbenamnya daratan karena volume air yang meningkat [[2](#Hai09)]. Banjir dapat terjadi karena peluapan air yang berlebihan di suatu tempat akibat hujan besar, peluapan air sungai, atau pecahnya bendungan sungai. Di banyak daerah yang gersang di dunia, tanahnya mempunyai daya serapan air yang buruk, atau jumlah curah hujan melebihi kemampuan tanah untuk menyerap air. Ketika hujan turun, yang kadang terjadi adalah banjir secara tiba-tiba yang diakibatkan terisinya saluran air kering dengan air.

Pada umumnya banjir diakibatkan curah hujan. Dalam jangka waktu tertentu hujan mengakibatkan aliran air di permukaan bumi (*run off*). Volume *run off* dan kecepatan aliran air tidak sebanding dengan besarnya drainase dan kecepatan aliran air menuju sungai atau kanal dan selanjutnya menuju penampungan air berupa waduk atau laut. Untuk itu diperlukan pengukuran kecepatan air beserta ketinggiannya untuk mengetahui seberapa besar potensi terjadinya banjir. Cara melakukan pengukuran salah satunya dengan menggunakan alat pengukur arus atau yang disebut *current meter.*

Perkembangan mikrokontroler dewasa ini membuat mikrokontroler dilengkapi sensor kecepatan air yang dapat menggantikan *current meter*. Sebagai contohnya adalah mikrokontroler Arduino bisa menghitung kecepatan air pada tempat yang sudah ditentukan. Dengan pendekatan tersebut sensor kecepatan ini dapat digunakan sebagai alternatif penghitung kecepatan air. Data dari sensor dilengkapi data posisi lokasi geografis. Informasi lokasi geografis dan data kecepatan air yang didapatkan dari sensor dikirim ke *server*. Hasil dari pengolahan data ditampilkan ke pengguna melalui situs web.

Dengan kombinasi data kecepatan air, ketinggian air dan informasi lokasi geografis, monitoring bencana banjir dapat dengan mudah dilakukan. Dengan diperolehnya informasi tersebut, dampak dari bencana banjir dapat diminimalisasi atau bahkan ditanggulangi. Oleh karena itu Tugas Akhir ini dikembangan sistem pemantauan kecepatan air dan ketinggian air yang dilengkapi posisi geografis untuk mengetahui daerah yang rawan banjir.

# RUMUSAN MASALAH

Adapun rumusan permasalahan yang ada saat ini adalah sebagai berikut:

1. Bagaimana cara mengukur kecepatan air dan ketinggian air yang dihasilkan dari sensor Arduino.
2. Bagaimana cara mengolah dan mengirim data ke *server*.
3. Bagaimana cara mengolah dan menyajikan informasi pada web.
4. Bagaimana cara menampilkan pesan peringatan apabila kecepatan air melebihi batas.

# BATASAN MASALAH

Permasalahan yang dibahas dalam Tugas Akhir ini memiliki beberapa batasan, diantaranya sebagai berikut:

1. Aplikasi hanya untuk menginfokan kondisi kecepatan dan ketinggian air di beberapa tempat.
2. Perangkat modul sensor menggunakan mikrokontroler Arduino.
3. Aplikasi menggunakan bahasa PHP dan C/C++.

# TUJUAN TUGAS AKHIR

Tugas Akhir ini memiliki tujuan yang rinciannya dapat dituliskan sebagai berikut:

1. Aplikasi bertujuan untuk memonitor besar kecepatan air dengan sensor kecepatan air.
2. Aplikasi bertujuan untuk memonitor ketinggian air dengan sensor ketinggian air.
3. Aplikasi bertujuan untuk mengirimkan koordinat posisi lokasi yang telah diukur kecepatan airnya.
4. Aplikasi bertujuan mengirim data ke *server* secara berkala.
5. Aplikasi bertujuan membaca data dari *server*.

# MANFAAT TUGAS AKHIR

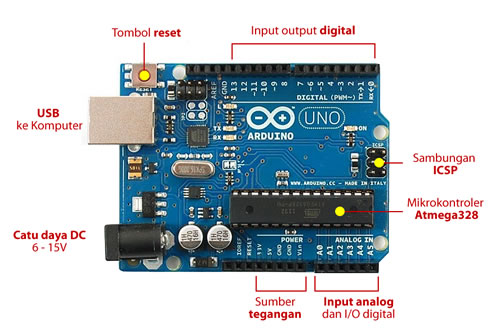
Manfaat yang diharapkan dari pengerjaan Tugas Akhir ini adalah untuk memberikan solusi pemantauan kecepatan air dan ketinggiannya di area yang rawan terjadi banjir dengan menggunakan sensor berbasis mikrokontroler Arduino.

# TINJAUAN PUSTAKA

## Mikrokontroler Arduino

Arduino adalah sebuah mikrokontroler *single-board* yang bersifat *open-source* seperti pada Gambar 1 [1]. *Hardware* mikrokontroler Arduino diprogram dengan menggunakan bahasa pemrograman *wiring-based* yang berbasiskan *syntax* dan *library*. Pemrograman *wiring-based* ini tidak berbeda dengan C/C++, tetapi dengan beberapa penyederhanaan dan modifikasi. Untuk memudahkan dalam pengembangan aplikasinya, mikrokontroler Arduino juga menggunakan *Integerated Development Environment* (IDE) berbasis *processing*.

Mikrokontroler Arduino dapat dipasangkan dengan bermacam-macam sensor dan aktuator lainnya. Adapun sensor dan aktuator yang dapat dipasangkan pada Arduino seperti sensor gerak, ultrasonik, panas, suara, Ethernet Shield, LED Display, dan yang lainnya.



Gambar 1. Mikrokontroller Arduino Uno

Dengan adanya sensor dan aktuator ini membuat mikrokontroler Arduino dapat berinteraksi dengan lingkungan sekitarnya. Penggunaan sensor maupun aktuator disesuaikan dengan kebutuhan dan tujuannya. Sebagai contoh adalah aplikasi Line Tracker, pada aplikasi ini perangkat keras yang digunakan yaitu sensor Line Tracker dan motor. Sehingga mikrokontroler Arduino dapat mengikuti jejak garis yang dilacak.

## PHP CodeIgniter

CodeIgniter adalah aplikasi *open source* berupa *framework* yang menggunakan model MVC (*Model, View, Controller*) untuk membangun web dinamis dengan menggunakan PHP[[3](#Cod02)]. *Framework* dapat diartikan kumpulan dari fungsi-fungsi dan kelas-kelas untuk tujuan tertentu yang sudah siap digunakan sehingga dapat mempermudah dan mempercepat *programmer* tanpa harus membuat fungsi atau kelas dari awal. Beberapa alasan digunakannya *framewor* kini antara lain:

1. Mempercepat dan mempermudah pembangunan sebuah aplikasi web.
2. Relatif memudahkan dalam proses pemeliharaan karena sudah ada pola tertentu dalam sebuah *framework* (dengan syarat *programmer* mengikuti pola standar yang ada).
3. Umumnya *framework* menyediakan fasilitas-fasilitas yang umum dipakai sehingga kita tidak perlu membangun dari awal (misalnya validasi, ORM, *pagination*, *multiple basis data*, *scaffolding*, pengaturan *session*, penanganan *error*, dll).
4. Lebih bebas dalam pengembangan jika dibandingkan *Content Management System*(CMS).

MVC (*Model*, *View*, *Controller*) merupakan suatu konsep yang cukup populer dalam pembangunan aplikasi web, berawal pada bahasa pemrograman Small Talk. MVC memisahkan pengembangan aplikasi berdasarkan komponen utama yang membangun sebuah aplikasi seperti manipulasi data, antarmuka pengguna, dan bagian yang menjadi kontrol aplikasi. Komponen utama yang membentuk pola MVC adalah:

1. *View*, adalah bagian yang berhubungan langsung dengan pengguna menangani tampilan aplikasi. Pada aplikasi web bagian ini biasanya berisi file HTML yang diatur oleh *controller*. Fungsi lain *View* adalah menerima dan merepresentasikan data ke pengguna. Bagian *View* tidak memiliki akses langsung ke *Model*  melainkan melalui *Controller*.
2. *Controller*, merupakan bagian yang mengatur hubungan antara bagian *model* dan bagian *view*, *controller* berfungsi untuk menerima *request* dan data dari pengguna kemudian menentukan apa yang akan diproses oleh aplikasi.
3. *Model*, bagian ini berhubungan langsung dengan basis data. Biasanya menangani proses manipulasi data. Bagian ini tidak berhubungan langsung dengan *View*.

Dengan model MVC yang dapat mempermudah proses pembangunan aplikasi dan manfaat dari penggunaan *framework*, maka jelas mengapa penggunaan *framework* untuk membangun aplikasi sangat dibutuhkan.

Pemilihan *framework* CodeIgniter dalam membangun aplikasi web bukan tidak beralasan. Ada beberapa kelebihan menggunakan CodeIgniter dibandingkan dengan *framework* PHP lain:

1. Performa sangat cepat: salah satu alasan tidak menggunakan *framework* adalah karena eksekusinya yang lebih lambat daripada PHP *from the scratch*, tapi CodeIgniter sangat cepat bahkan mungkin bisa dibilang CodeIgniter merupakan *framework* yang paling cepat dibanding *framework* yang lain.
2. Konfigurasi yang sangat minim (*nearly zero configuration*). Tentu saja untuk menyesuaikan dengan basis data dan keleluasaan *routing* tetap diizinkan melakukan konfigurasi dengan mengubah beberapa file konfigurasi seperti *database.php* atau *autoload.php*, namun untuk menggunakan CodeIgniter dengan pengaturan standar, anda hanya perlu mengubah sedikit saja file pada folder *config*.
3. Banyak komunitas. Dengan banyaknya komunitas CodeIgniter ini, memudahkan pengguna untuk berinteraksi dengan yang lain, baik itu bertanya atau teknologi terbaru.
4. Dokumentasi yang sangat lengkap. Setiap paket instalasi CodeIgniter sudah disertai panduan pengguna yang sangat bagus dan lengkap untuk dijadikan permulaan, bahasanya pun mudah dipahami.

Untuk penggunaan *framework* CodeIgniter ini dapat diunduh secara gratis di situs web [www.codeigniter.com](http://www.codeigniter.com).

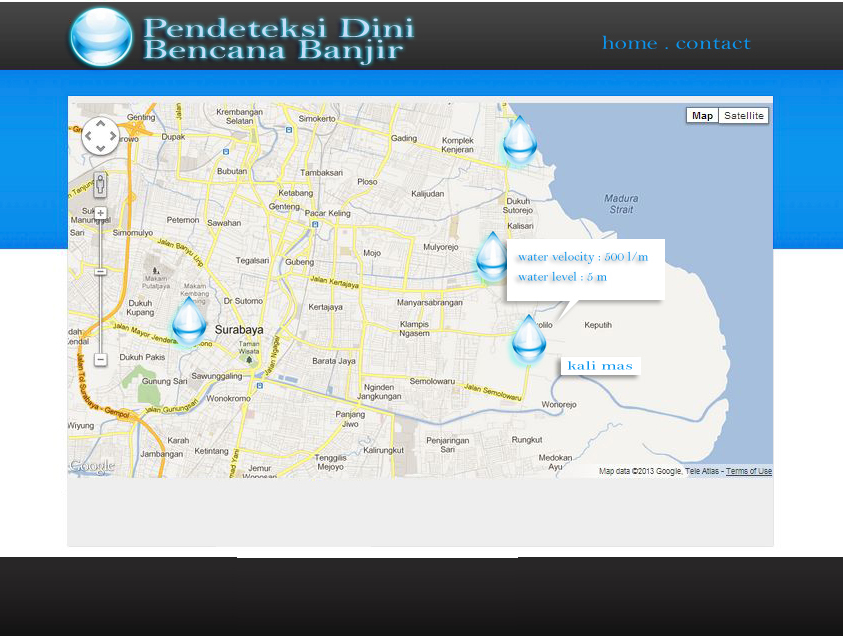
## Google Maps

Google Maps merupakan sebuah *Application Programming Interface* (API) yang dibuat oleh Google untuk menampilkan informasi lokasi berbasis peta. Google Maps dapat diakses melalui Java Script sehingga dapat ditampilkan pada halaman web [[4](#GooMaps)]. Untuk dapat mengakses Google Maps, pengembang harus melakukan pendaftaran API Key terlebih dahulu dengan data pendaftaran berupa nama domain web yang kita bangun.

# METODOLOGI

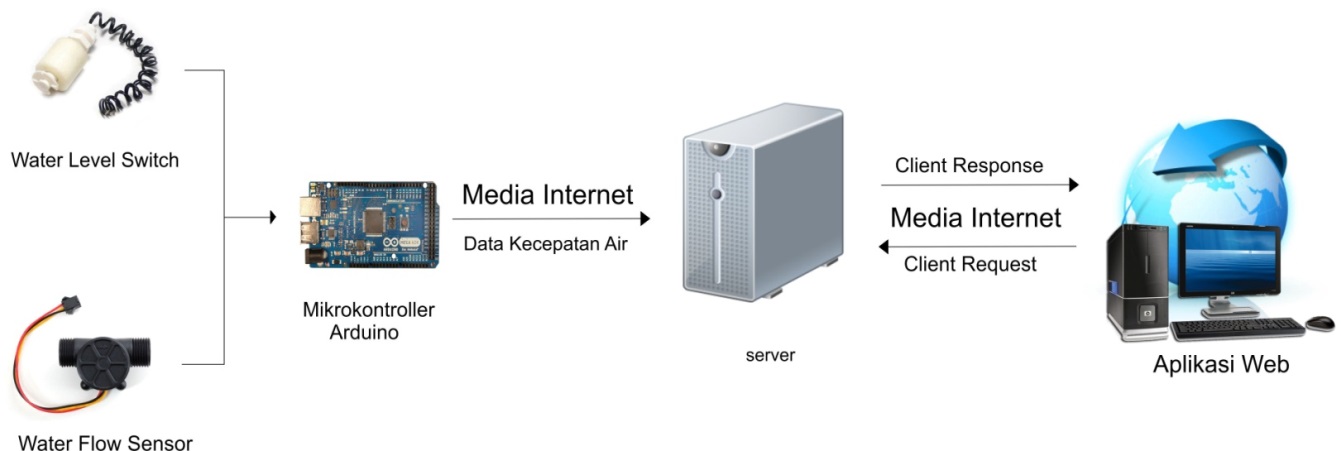
Dalam Tugas Akhir ini penulis akan membuat sebuah aplikasi yang berjalan pada sebuah web. Aplikasi web ini berfungsi untuk memantau kecepatan air pada lokasi tertentu. Aplikasi ini bekerja dengan menerima masukan (*input*) dari sensor kecepatan aliran air dan ketinggian air. Dengan mikrokontroler Arduino pada di beberapa titik yang rawan banjir mengirimkan datanya ke sebuah *server* pusat. *Server* tersebut menyimpan semua informasi mengenai data kecepatan air, ketinggian air dan juga lokasi geografis dari mikrokontroler Arduino. *Server* pusat tersebut juga akan menjadi sumber data bagi aplikasi klien yang berbasis web. Setiap mikrokontroler Arduino akan mengirimkan datanya ke *server* menggunakan perangkat Ethernet Shield atau Ethernet Card. Ethernet Shield menggunakan protokol HTTP sebagai komunikasi datanya.

Web pada Tugas Akhir ini menggunakan API dari Google Maps untuk memudahkan memonitor lokasi yang rawan terjadi banjir. Untuk tampilan awalnya terdapat peta yang menunjukkan daerah yang dimonitor beserta data kecepatan air dan juga ketinggiannya. Untuk lebih jelasnya dapat dilihat pada Gambar 3. Terdapat simbol air yang menunjukkan posisi sensor.



Gambar 2. Rancangan Tampilan Web

Pada Tugas Akhir ini digunakan sebuah mikrokontroler Arduino yang dilengkapi dengan sensor kecepatan air dan ketinggian air. Perancangan sistem arsitektur untuk Tugas Akhir ini, secara garis besar dapat digambarkan seperti pada Gambar 4.



Gambar 3. Arsitektur Sistem

Seperti pada Gambar 4, data kecepatan air didapat dari sensor kecepatan air (*water flow sensor*) yang diolah oleh mikrokontroler Arduino. Sensor aliran air ini sendiri terbuat dari plastik dimana di dalamnya terdapat rotor dan sensor *hall effect* [[5](#htt)]*.* Saat air mengalir melewati rotor, rotor akan berputar. Kecepatan putaran ini akan tergantung dengan kecepatan aliran air. Sensor *hall effect* akan mengeluarkan keluaran pulsa sesuai dengan besarnya aliran air. Begitu juga dengan data ketinggian air. Data ketinggian air didapat dari sensor ketinggian air (*water level switch*) yang diolah dengan mikrokontroler Arduino [[6](#Wat13)].

Pada Tugas Akhir ini, penulis juga menggunakan posisi secara geografis untuk menunjukkan letak daerah yang rawan banjir. Proses pengumpulan data berhenti sampai disini. Data akan dikirimkan mikrokontroler Arduino ke *server* melalui koneksi internet dengan menggunakan protokol HTTP.

Proses selanjutnya adalah pengolahan data yang dilakukan oleh *server*. *Server* mendapatkan data dari Arduino yang mengirimkan data secara berkala. Setiap data yang diterima akan disimpan pada basis data jika data yang diterima benar. Untuk menentukan benar atau tidaknya data yang diterima, data tersebut akan disesuaikan dengan format data yang sudah ditentukan oleh *server*.

# JADWAL KEGIATAN TUGAS AKHIR

Tugas Akhir ini diharapkan dapat dikerjakan menurut jadwal sebagai berikut:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Kegiatan** | **Bulan** | | | | | | | | | | | | | | | |
| **Maret 2013** | | | | **April 2013** | | | | **Mei 2013** | | | | **Juni 2013** | | | |
| 1. | Penyusunan Proposal Tugas Akhir |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2. | Studi Literatur |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3. | Analisa dan Perancangan |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4. | Implementasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5. | Pengujian dan Evaluasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6. | Penyusunan Buku Tugas Akhir |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# DAFTAR PUSTAKA

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| x   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | x   |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | x   |  |  | | --- | --- | | [1] | (2013, Maret) Arduino. [Online]. <http://www.arduino.cc/> | | [2] | Zhang Hailin, "GIS-Based Risk Assessment for Regional Flood Disaster," *Environmental Science and Information Application Technology*, vol. 2, pp. 564-567, July 2009. | | [3] | (2013, Februari) CodeIgniter. [Online]. <http://codeigniter.com/> | | [4] | Google Developer. (2012, Desember) Google Maps. [Online]. <https://developers.google.com/maps/documentation/> | | [5] | (2013, maret) gerai cerdas. [Online]. <http://www.geraicerdas.com/water-flow-sensor-g1/2> | | [6] | (2013, februari) gerai cerdas. [Online]. <http://www.geraicerdas.com/water-level-switch> |   x |  | |  |  | |  |  | |  |  | |  |  |   x |  | |  |  | |  |  | |  |  | |  |  |   x |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |