LAMPIRAN 1 DAFTAR RIWAYAT HIDUP

Profil Pribadi

Nama : Adven Kristianata
TTL : Blitar 1 April 1998
Status : Mahasiswa
Bahasa : Indonesia
Inggris

Tentang Saya

Saya adalah seseorang yang tertarik kepada dunia Desain dan Teknologi. Dalam dunia Desain saya mengaplikasikannya dalam desain 2D dan sedikit kemampuan saya dalam 3D. Dan juga dalam desain tampilan pada WEB dan Android. Saya juga bisa mendesain mockup untuk aplikasi maupun web.

Dalam dunia Programmer saya berfokus untuk menjadi Front-End.

Villa Balaraja K2/17, Saga, Balaraja Kab. Tangerang, Banten.
fb.com/adven.kristianata
linkedin.com/advenkris
twitter.com/AdvenKris37
instagram.com/advenkristianata
advenkris37@gmail.com

Pendidikan

- SMPK Yohanes Gabriel — 2010 - 2013
- SMK Mandiri 2 Balaraja — 2013 - 2016
- Universitas Esa Unggul — 2016 - Sekarang

Pengalaman Kerja

- PT Futurax Global Group — 2018-2019
  - UI/UX Designer
  - Quality Assurance

Pengalaman Organisasi

- Indonesian Hoax Buster — 2015 -Sekarang
  Co-Founder
  Admin
- Masyarakat Anti Fitnah Hasut dan Hoax — 2016 -Sekarang
  Anggota
- Badan Eksekutif Mahasiswa — 2017 -2018
  Universitas Esa Unggul
  - (Wakil) Dept Kominfo

Kemampuan

CorelDraw
Adobe Photoshop
Adobe Illustrator
Android Studio
Figma
Sony Vegas
Bootstrap
CSS
Adobe Premiere

Sertifikasi

- CCNA Routing and Switching — Cisco — Nov - 2018
- CCNA Routing and Switching: — Cisco — Jan - 2019
- Junior Graphic Desain — Indonesian Professional Certification Authority — Mei - 2018

Aplikasi

- BelajarYuKak! (Released at Playstore) — Android
  UI/UX Designer
- Elizasby.net (Maintening) — WEB
  UI/UX Designer
- WarungBroker.com (Maintening) — WEB
  UI/UX Designer

LAMPIRAN 2 WAWANCARA

Pengembangan Aplikasi *Scheduler Team Meeting* Berbasis *Mobile*
Dengan Menggunakan *Push Notification*

Studi Kasus: PT NTX Solusi Teknologi

Wawancara secara langsung kepada *Product Manager* yang menangani perencanaan aplikasi internal maupun projek.

Waktu Pelaksanaan	:	Senin, 2 Desember 2019
Tempat	:	PT NTX Solusi Teknologi, Jl. HR. Rasuna Said Kav C-3 Jakarta 12920, Indonesia Tel. (62-21) 5204873 Fax. (62-21) 5204874

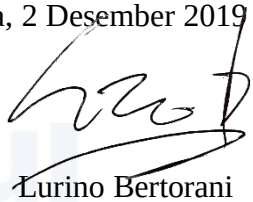
Identitas Informan

Nama	:	Lurino Bertorani
Pekerjaan	:	Product Manager di NTX Solusi Teknologi
Jenis Kelamin	:	Laki-laki
Pendidikan	:	S3

Peneliti	:	NTX Solusi Teknologi, bergerak dalam bidang apa ya?
Responden	:	Perusahaan kami bergerak sebagai Konsultan dalam bidang IT dan juga Analisis Data.
Peneliti	:	Dalam aktivitas setiap hari, team apakah banyak melakukan *Meeting* dengan *client* diluar kantor?
Responden	:	Betul, *team* hampir setiap hari melakukan *Meeting* dengan *client* diluar kantor.

Peneliti	:	Lalu dengan staff seperti BDO atau *Project Manager* apakah harus secepatnya mengetahui hasil dari pertemuan tersebut?
Responden	:	Sangat diharuskan secepatnya melaporkan, karena terkait dengan produk yang dibutuhkan oleh *client* tersebut, sehingga penyampaian informasi yang disampaikan oleh *client* tidak ada yang kurang/*miss*, memudahkan dalam koordinasi juga
Peneliti	:	Bagaimana penjadwalan yang dilakukan untuk memudahkan koordinasi tersebut?
Responden	:	Itu dia permasalahan disini, belum ada aplikasi yang menghandle jadwal untuk Meeting internal antara tim dan staff yang membutuhkan koordinasi lebih lanjut, selama ini kita hanya menggunakan chat melalui Whatsapp atau Telegram dalam berkoordinasi, dan hal itu terkadang membuat lupa jadwal *Meeting* yang akan berlangsung. Juga ketika ada *Meeting* diluar, terkadang mereka tidak mengabari staff sehingga tidak mengetahui jadwal mereka hari ini kemana saja.
Peneliti	:	Sistem seperti apa yang bapak harapkan untuk menangani permasalahan tersebut?
Responden	:	Sistem yang bisa memberitahu bahwa akan ada *Meeting*, lalu bisa membuat jadwal *Meeting* dan *team* lain bisa melihat jadwal antar *team*.
Peneliti	:	*Requirement* untuk berjalannya aplikasi tersebut di *Mobile* atau di PC?
Responden	:	Bisa berjalan disemuanya, tetapi yang paling penting adalah bisa mendapatkan notifikasi untuk jadwal pertemuan yang akan dilakukan

Jakarta, 2 Desember 2019



Lurino Bertorani

Product Manager

LAMPIRAN 3 SURAT PENELITIAN

Universitas
Esa Unggul
Fakultas Ilmu Komputer

Jakarta , 3 Desember 2019

Nomor : 136/SP/KAPRODI-TIF/FASILKOM/EXT/XII/2019
Lampiran : -
Perihal : ***Surat Permohonan Izin Untuk Penelitian***

Kepada Yth,
PT. NTX Solusi Teknologi
Mega Plaza 4th Floor
Jl.HR.Rasuna Said Kav C-3 Jakarta 12920, Indonesia

Sehubungan dengan mata kuliah Tugas Akhir yang memerlukan data dan informasi bagi mahasiswa **Fakultas Ilmu Komputer** Program Studi **Teknik Informatika**, bersama ini kami sampaikan bahwa mahasiswa kami bermaksud untuk melakukan penelitian. Adapun nama mahasiswa tersebut adalah :

Nama : Adven Kristianata
NIM : 2016-08-01-089

Demikianlah atas perhatian dan kerjasamanya, kami ucapkan terima kasih.

Hormat kami ,

Universitas
Esa Unggul

Malabay, S.Kom, M.Kom
Ketua Program Studi Teknik Informatika

Jl. Arjuna Utara 9, Tol Tomang, Kebon Jeruk, Jakarta 11510, Indonesia
(021) 567 4223 ext. 206, 207 (021) 567 4248
www.esaunggul.ac.id

LAMPIRAN 4 SURAT BALASAN PERUSAHAAN

NTX Solusi Teknologi
Next Technology

SURAT KETERANGAN
No 002/HR/NTX/XI/2019

Yang bertanda tangan di bawah ini Product Manager Pt. NTX Solusi Teknologi, dengan ini menerangkan bahwa :

Nama	: Adven Kristianata
Jurusan	: Teknik Informatika
NIM	: 20160801089
Universitas	: Esa Unggul

Adalah benar telah melaksanakan penelitian di PT. NTX Solusi Teknologi guna memperoleh data dalam rangka penulisan ilmiah yang berjudul "Pengembangan Aplikasi Scheduller Team Meeting Berbasis Mobile Dengan Menggunakan Push Notification"

Demikian surat keterangan ini dibuat dan di serahkan kepada yang bersangkutan untuk dipergunakan sebagimana mestinya.

Jakarta 12 Desember 2019

LURINO BERTORANI
Product Manager

/ARSIP

PT NTX Solusi Teknologi
Mega Plaza 4th Floor - Jl. HR. Rasuna Said Kav. C-3 Jakarta 12920 – Indonesia
Telp + 62 21 520 4873 - Fax + 62 21 520 4874 - Email : marketing@ntxsolution.com
www.NTXSOLUTION.com

Lampiran 5 Source Code API ScheduleControler.php

```
<?php
class ScheduleController extends SecureRouteComponent {
    function __construct(){
        parent::__construct();
    }

    function index(){
        $model = new Scheduler();
        $this->data = ['success' => true, 'code'=> 200, 'payload' => $model->getData()];
    }

    function list(){
        $model = new Scheduler();
        $this->data = ['success' => true, 'payload' => $model->getList()];
    }

    function get_data_dashboard(){
        $start = $this->f3->get('GET.start');
        $end = $this->f3->get('GET.end');

        $model = new Scheduler();
        $lastSchedule = $model->afind(array('is_trash != ?', 1), array('order' =>
'schedule_start_date DESC', 'limit' => 3));
        $thisMonthSchedule = $model->afind(
            array('is_trash != ? AND DATE(schedule_start_date) >= ? AND
DATE(schedule_start_date) <= ?', 1, $start, $end),
            array('limit' => 100)
        );
        $this->data = [
            'success' => true,
```

```
            'code' => 200,
            'data' => [
                'last_schedule' => $lastSchedule,
                'this_mont_scheudule' => $thisMonthSchedule
            ]
        ];
    }

    function get_by_id(){
        $param = $this->f3->get('GET._id');

        $model = new Scheduler();
        $query = array('id = ?', $param);

        $model->load($query, $options);

        $this->data = ['success' => true, 'code'=> 200, 'data' => $model->cast()];
    }

    function summary(){
        $param = $this->f3->get('GET.type');

        $model = new Scheduler();

        if(empty($param)){
  $model->has('scheduler_detail', array('schedule_detail_user_id = ?', $this->f3->get('USER')->user->_id));
        }else{
            $user = new Users();
            $userTotal = $user->count(array('is_trash != ?', 1));
        }
```

```
        $all = $model->count();
        $day = $model->count(array('DATE(schedule_end_date) = ?', date("Y-m-d")));
        $complete = $model->count(array('DATE(schedule_end_date) <= ?', date("Y-m-d")));
        $scheduler = $model->count(array('DATE(schedule_end_date) >= ?', date("Y-m-d")));

        $this->data = [
            'success' => true,
            'code'=> 200,
            'payload' => [
                'all' => $all,
                'day' => $day,
                'complete' => $complete,
                'scheduler' => $scheduler,
                'user_total' => $userTotal
            ]
        ];
    }

    function approach(){
        $model = new Scheduler();
        $options = array('order' => 'created_at');
        $query = array('DATE(schedule_end_date) >= ?', date("Y-m-d"));

        $model->has('scheduler_detail', array('schedule_detail_user_id = ?', $this->f3-
>get('USER')->user->_id));
        $data = $model->afind($query, $options);
        $this->data = ['success' => true, 'payload' => $data];
    }

    function chart_user(){
        try{
            $model = new Scheduler();
```

```
        $start_date = date('Y-m-d',strtotime("-30 days"));
        $options = array('group' => 'created_at');
        $query = array('DATE(created_at) >= ? AND DATE(created_at) <= ?', $start_date, date("Y-m-d") );

        $model->has('scheduler_detail', array('schedule_detail_user_id = ?', $this->f3->get('USER')->user->_id));
        $model->count_schedule = 'COUNT(created_at)';
        $model->fields(array('created_at', 'count_schedule'));
        $data = $model->afind($query, $options);
        $this->data = ['success' => true, 'payload' => $data];
      } catch(\PDOException $e) {
        $err=$e->errorInfo;
        $this->data = ['success' => false, 'message' => 'Something went wrong', 'payload' => $err];
      }
    }

    function answer(){
      try{
      $model = new Scheduler_detail();
      $this->f3->get('DB')->begin();
      $id = $this->post['id'];
      $user_id = $this->f3->get('USER')->user->_id;
      $answer = $this->post['answer'];

      $query = array('schedule_id = ? AND schedule_detail_user_id = ?',  $id, $user_id);
      $model->load($query);
      $model->schedule_detail_approach = $answer;
      $model->save();

      $this->f3->get('DB')->commit();
```

```
        $this->data = ['success' => true, 'message' => 'Success', 'data' => $answer];

        } catch(\PDOException $e) {
            $err=$e->errorInfo;
            $this->data = ['success' => false, 'message' => 'Something went wrong', 'payload' =>
$err];
        }
    }

    function remind(){
        try{
        $model = new Scheduler_detail();
        $this->f3->get('DB')->begin();
        $id = $this->post['id'];
        $user_id = $this->f3->get('USER')->user->_id;
        $remind = (int) $this->post['remind'];

        $query = array('schedule_id = ? AND schedule_detail_user_id = ?',  $id, $user_id);

        $model->load($query);
        $model->schedule_detail_remind = $remind;
        $model->save();

        $this->f3->get('DB')->commit();

        $this->data = ['success' => true, 'message' => 'Success', 'data' => $remind];
        } catch(\PDOException $e) {
            $err=$e->errorInfo;
            $this->data = ['success' => false, 'message' => 'Something went wrong', 'payload' =>
$err];
        }
```

```
}

function create(){
    try {
        $payload = json_decode($this->f3->get('BODY'));
        $model = new Scheduler();
        $this->f3->get('DB')->begin();

        $this->f3->set('POST', $payload);
        $model->copyfrom('POST');
        //unset($model->schedule_detail_user_id);
        $model->schedule_code = '';
        $model->schedule_phone = $this->f3->get('USER')->user->user_phone;
        $model->save();

        foreach ($payload->schedule_detail_user_id as $value) {
            $modelDetail = new Scheduler_detail();
            $modelDetail->schedule_id = $model->_id;
            $modelDetail->schedule_detail_user_id = $value->id;
            $modelDetail->schedule_detail_user_name = $value->name;
            $modelDetail->schedule_detail_user_email = $value->email;
            $modelDetail->schedule_detail_remind = 0;
            $modelDetail->schedule_detail_approach = 0;
            $modelDetail->schedule_detail_disclaimer = 0;
            $modelDetail->schedule_detail_valid = 0;
            $modelDetail->save();
        }

        $this->f3->get('DB')->commit();

        $sendNotification = $model->sendNotification();
```

```
$this->data = ['success' => true, 'code' => 200, 'message' => 'Success', 'data' => $model->cast(), 'notification' => $sendNotification];
        } catch(\PDOException $e) {
            $err=$e->errorInfo;
            $this->errorData = ['success' => false, 'code' => 400, 'message' => 'Something went wrong', 'payload' => $err];
        }
    }

    function update(){
                try {
            $model = new Scheduler();
            $id =$this->f3->get('GET._id');
            $schedulerDetail = $this->post['schedule_detail_user_id'];
            $this->f3->set('action', 'update');
            $this->f3->set('POST', $this->post);

            if($model->validation($this->f3->get('POST'))){
                $model->reset();
                $model->edit($id);

                foreach ($schedulerDetail as $value) {
                    $modelDetail = new Scheduler_detail();

                    $modelDetail->schedule_id = $id;
                    $modelDetail->schedule_detail_user_id = $value['id'];
                    $modelDetail->schedule_detail_user_name = $value['name'];
                    $modelDetail->schedule_detail_user_email = $value['email'];
                    $modelDetail->schedule_detail_remind = 0;
                    $modelDetail->schedule_detail_approach = 0;
                    $modelDetail->schedule_detail_disclaimer = 0;
                    $modelDetail->schedule_detail_valid = 0;
```

```
            $modelDetail->save();
        }

        if($model->valid()){
            $this->data = ['success' => true, 'code'=> 200, 'data' => $model->cast()];
        }
    }else{
        $this->errorData = ['success' => false, 'code'=> 400, 'message' => 'not valid
request'];
    }

  } catch(\PDOException $e) {
    $err=$e->errorInfo;
    $this->errorData = ['success' => false, 'code'=> 400, 'message' => $err[2]];
  }
    }

 function upload_location(){
  try {
    $this->f3->set('UPLOADS','static/img/location/');

    $overwrite = true; // set to true, to overwrite an existing file; Default: false
    $slug = true; // rename file to filesystem-friendly version
    $web = \Web::instance();
    $files = $web->receive(function($file,$formFieldName){
        if($file['size'] > (4 * 1024 * 1024))// if bigger than 2 MB
            return false; // this file is not valid, return false will skip moving it

        return true;
    },
    $overwrite,
    $slug
```

```
        );

        $north = $this->f3->get('POST.bound_north');
        $west = $this->f3->get('POST.bound_west');
        $south = $this->f3->get('POST.bound_south');
        $east = $this->f3->get('POST.bound_east');

        $model = new Scheduler_detail();
        //var_dump($files);die();
        $fileName = array_keys($files)[0];
        $image = ImageComponent::get_exif($fileName);
        $location = ImageComponent::get_image_location($image);

        if(!$location){
           throw new Exception("location not found");
        }

        $software = ImageComponent::get_exif_software($image);
        $exifdate = ImageComponent::get_exif_date($image);

        $model->load(array('_id = ?', $this->f3->get('POST.scheduler_detail_id')));
        $model->schedule_detail_location  = $location[0].','.$location[1];

        if($north > $location[0] && $west > $location[1] && $south < $location[0] && $east
< $location[1]){
           $model->schedule_detail_valid  = 1;
        }
        $model->schedule_detail_image = $fileName;
        $model->save();

        $this->data = [
           'success' => true,
```

```
            'message' => 'Success',
            'data' => $model->cast(),
            'exif' => ['datetime' => $exifdate, 'software' => $software, 'location' => $location]
        ];
    } catch(\PDOException $e) {
        $err=$e->errorInfo;
        $this->f3->status(404);
        $this->data = ['code' => 400, 'message' => 'Something went wrong', 'payload' => $err];
    } catch (Exception $e) {
        $this->f3->status(404);
        $this->errorData = ['code' => 400, 'message' => $e->getMessage()];
    }
}


function delete_member(){
    try {
        $id = $this->f3->get('GET._id');

        $model = new Scheduler_detail();
        $model->delete($id);

        $this->data = ['success' => true, 'code'=> 200, 'message'=>'Deleted successfull'];

    } catch(\PDOException $e) {
        $err=$e->errorInfo;
        $this->errorData = ['success' => false, 'code'=> 400, 'message' => 'Something went
wrong'];
    }
}

function delete(){
```

```
        try {
            $id = $this->f3->get('GET._id');

            $model = new Scheduler();
            $query = array('id = ?', $id);

            $model->load($query);
            $model->is_trash = 1;
            $model->save();

            $this->data = ['success' => true, 'code'=> 200, 'message'=>'Deleted successfull'];

        } catch(\PDOException $e) {
            $err=$e->errorInfo;
            $this->errorData = ['success' => false, 'code'=> 400, 'message' => 'Something went wrong'];
        }
    }
}
```

Lampiran 6 Source Code API SiteController.php

```
<?php
class SiteController extends BaseRouteComponent {

    function __construct(){
        parent::__construct();
        $this->model = new Users();
            }

            function index(){
                    $this->data = [
            'success'=> true,
            'payload'=> 'Is OK'
        ];
            }
    function register(){
        $data = $this->f3->get('POST');

        $validator = \Validate::instance();
        $validator->validation_rules(array(
                            'email' => 'required|valid_email|alpha_numeric',
                            'password' => 'required|max_len,100|min_len,6'
        ));

        if($validator->run($data)) {
            $this->data = [
                'success'=> true,
                'payload'=> $this->model->add()
            ];
        }else{
```

```
            $this->data = [
                'success'=> false,
                'payload'=> $validator->get_errors_array()
            ];
        }
            }

    function login(){

                    $validator = \Validate::instance();
        $validator->validation_rules(array(
                            'email' => 'required|email',
                            'password' => 'required|max_len,100|min_len,5'
        ));

>hash($_POST['user_password']));die()

        if($validator->run($this->post)) {
            $this->model->load(array('user_email = ?', $this->post['email']));

            if (\Bcrypt::instance()->verify($this->post['password'], $this->model->user_password)) {
$token = $this->model->genJWt($this->post['email']);
 $this->f3->set('COOKIE.auth', $token, 86400, '/', 'localhost'); // 1 day
$this->data = ['success' => true, 'payload' => ['messages'=>'Token generated successfully', 'token' => ''. $token]];
            }else{
                $this->data = ['success' => false, 'payload' => 'Invalid email and/or password'];
            }
        } else {
            $this->data = ['success' => false, 'payload' => $validator->get_errors_array()];
        }
```

```
    }


    function owner_login(){

                $validator = \Validate::instance();
        $validator->validation_rules(array(
                        'username' => 'required',
                        'password' => 'required|max_len,100|min_len,5'
        ));

        if($validator->run($this->post)) {
            $this->model->load(array('user_name = ? AND user_level = ?', $this->post['username'], 0));

            if (\Bcrypt::instance()->verify($this->post['password'], $this->model->user_password)) {
                $token = $this->model->genJWt($this->model->user_email);

                $this->f3->set('COOKIE.auth', $token, 86400, '/', 'localhost'); // 1 day
                $this->data = ['success' => true, 'code' => 200, 'payload' => ['messages'=>'Token generated successfully', 'token' => ''. $token]];
            }else{
                $this->errorData = ['code' => 404, 'message' => 'Invalid username and/or password'];
            }
        } else {
            $this->data = ['success' => false];
            $this->errorData = ['code' => 400, 'message' => $validator->get_errors_array()];
        }
    }
```

```
    function send_notification(){
        try {
            $model = new Scheduler();
            $query = array('NOW() >= DATE_ADD(schedule_start_date, INTERVAL -2 HOUR) AND NOW() <= schedule_start_date AND is_trash = ? AND schedule_send_notification = ?', 0, 0);

            $results = $model->find($query);
            foreach ($results as $value) {
                $value->sendNotification();
                $value->schedule_send_notification = 1;
                $value->save();
            }

            $this->data = ['success' => true, 'code' => 200, 'message' => 'Success'];
        } catch(\PDOException $e) {
            $err=$e->errorInfo;
            $this->errorData = ['success' => false, 'code' => 400, 'message' => 'Something went wrong', 'payload' => $err];
        }
    }
}
```

Lampiran 7 Source Code API UserController.php

```
<?php
class UserController extends SecureRouteComponent {

   function __construct(){
      parent::__construct();
   }
   function index(){
      $model = new Users();
      $model->fields(array('user_password'), true); // all fields, but not these
      $data = $model->getData();

      $this->data = ['success' => true, 'code'=> 200, 'payload' => $data];
   }

   function get_by_id(){
      $param = $this->f3->get('GET._id');

      $model = new Users();
      $model->fields(array('user_password'), true); // all fields, but not these
      $query = array('id = ?', $param);

      $model->load($query, $options);


      $this->data = ['success' => true, 'code'=> 200, 'data' => $model->cast()];
   }

   function create(){
                  try {
```

```
        $model = new Users();
        $this->f3->set('POST', $this->post);
        $this->f3->set('action', 'create');
        $this->f3->set('POST.user_code', 1);
        $this->f3->set('POST.user_level', 1);
        $this->f3->set('POST.user_token', 1);
        $this->f3->set('POST.user_onesignal_id', '');
        $this->f3->set('POST.is_trash', 0);
        $this->f3->clear('POST.user_password_confirm');

        if($model->validation($this->f3->get('POST'))){
            $model->reset();
            $model->add();

            if($model->valid()){
                $this->data = ['success' => true, 'code'=> 200, 'data' => $model->cast()];
            }
        }else{
            $this->errorData = ['success' => false, 'code'=> 400, 'message' => 'not valid
request'];
        }

    } catch(\PDOException $e) {
        $err=$e->errorInfo;
        $this->errorData = ['success' => false, 'code'=> 400, 'message' => $err[2]];
    }
        }

        function update(){
                try {
        $model = new Users();
        $this->f3->set('action', 'update');
```

```
        $this->f3->set('POST', $this->post);
        $this->f3->set('POST.user_password', '');

        if($model->validation($this->f3->get('POST'))){
            $model->reset();
            $model->edit($this->f3->get('GET._id'));

            if($model->valid()){
                $this->data = ['success' => true, 'code'=> 200, 'data' => $model->cast()];
            }
        }else{
            $this->errorData = ['success' => false, 'code'=> 400, 'message' => 'not valid
request'];
        }

    } catch(\PDOException $e) {
        $err=$e->errorInfo;
        $this->errorData = ['success' => false, 'code'=> 400, 'message' => $err[2]];
    }
        }

  function name(){
    $param = '%'.$this->f3->get('GET.q').'%';

    $model = new Users();
    $model->fields(array('user_name', 'user_email')); // all fields, but not these
    $query = array('user_name LIKE ? OR user_email LIKE ?', $param, $param);
    $options = array('limit' => 10);

    $data = $model->afind($query, $options);
```

```
        $this->data = ['success' => true, 'code'=> 200, 'data' => $data];
    }

    function info(){
        try {
            $this->data = ['success' => true, 'code'=> 200, 'data' => $this->f3->get('USER')->user];
        } catch(\PDOException $e) {
            $err=$e->errorInfo;
            $this->errorData = ['success' => false, 'code'=> 400, 'message' => 'Something went
wrong'];
        }
    }



    function uploads(){
        try {
            $this->f3->set('UPLOADS','static/img/');

            $overwrite = true; // set to true, to overwrite an existing file; Default: false
            $slug = true; // rename file to filesystem-friendly version
            $web = \Web::instance();
            $files = $web->receive(function($file,$formFieldName){
                    if($file['size'] > (2 * 1024 * 1024)) // if bigger than 2 MB
                        return false; // this file is not valid, return false will skip moving it

                    return true;
                },
                $overwrite,
                $slug
            );
            //var_dump($files);die();
```

```
            $this->data = ['success' => true, 'message' => 'Success', 'data' =>
array_keys($files)[0]];
        } catch(\PDOException $e) {
            $err=$e->errorInfo;
            $this->errorData = ['success' => false, 'message' => 'Something went wrong'];
        }
    }

    function update_onesignal(){
        try {
            $id = $this->post['userId'];
            $user_id = $this->f3->get('USER')->user->_id;
            $model = new Users();
            $query = array('id = ?', $user_id);
            $model->load($query);
            $model->user_onesignal_id = $id;
            $model->save();
            $token = $model->genJWt($model->user_email);

            $this->data = ['success' => true, 'payload' => ['messages'=>'Token generated
successfully', 'token' => ''. $token]];

        } catch(\PDOException $e) {
            $err=$e->errorInfo;
            $this->errorData = ['success' => false, 'code'=> 400, 'message' => 'Something went
wrong'];
        }
    }

    function delete(){
        try {
            $id = $this->f3->get('GET._id');
```

```
            $model = new Users();
            $query = array('id = ?', $id);

            $model->load($query);
            $model->is_trash = 1;
            $model->save();

            $this->data = ['success' => true, 'code'=> 200, 'message'=>'Deleted successfull'];

        } catch(\PDOException $e) {
            $err=$e->errorInfo;
            $this->errorData = ['success' => false, 'code'=> 400, 'message' => 'Something went
wrong'];
        }
    }
}
```

Source Code Master Model

```
<?php
class MasterModel extends \DB\Cortex {

function __construct() {
		//$this->f3 = \Base::instance();
		//$this->virtual('token', TRUE);
		$this->beforeinsert(function($self, $pkeys){
			if ($self->exists('created_at')) {
				$self->created_at = date("Y-m-d H:i:s");
			}
			if ($self->exists('created_by')) {
				$self->created_by = isset(\Base::instance()->get('POST')->created_by) ? \Base::instance()->get('POST')->created_by : \Base::instance()->get('USER')->user->_id;
			}
			if ($self->exists('updated_at')) {
				$self->updated_at = date("Y-m-d H:i:s");
			}
			if ($self->exists('updated_by')) {
				$self->updated_by = \Base::instance()->get('USER')->user->_id;
			}
		});
		$this->beforeupdate(function($self){
			if ($self->exists('updated_at')) {
				$self->updated_at = date("Y-m-d H:i:s");
			}
			if ($self->exists('updated_by')) {
				$self->updated_by = \Base::instance()->get('USER')->user->_id;
```

```
                }
                //$self->clear('token');
            });
    parent::__construct();
        }

        public function add(){
    $this->copyfrom('POST');
    $this->save();
        }

        public function edit($id){
    $this->load(array('id = ?',$id));
    $this->copyfrom('POST');
    $this->save();
}

public function getById($id){
    $this->load(array('id = ?',$id));
    $this->copyto('POST', 1);
}

        public function delete($id){
                $this->load(array('id = ?',$id));
                $this->erase();
        }

//public function get_created_at($value) {
    //return date("d/m/Y - H:i", strtotime($value));
        //}
```

```
    public function get_created_by($value) {
            return (new Users)->findone(['id = ? ', $value])->user_name;
        }

        public function getPrefixNumber($s){
            return \Base::instance()->get('config')->$s;
        }

        public function generatePad($value){
            return LabelComponent::generateNumber($value);
        }

        public function generateNumber(){
            return date('ymd').LabelComponent::generateNumber($this->count() + 1);
        }
}
```

Source Code User Model

```
<?php
use Firebase\JWT\JWT;

class Users extends MasterModel
{
    protected $fieldConf = [
        'user_code' => [
            'type' => 'VARCHAR128',
            'nullable' => false
        ],
        'user_name' => [
            'type' => 'VARCHAR128',
            'nullable' => false
        ],
        'user_password' => [
            'type' => 'VARCHAR128',
            'validate'=> 'required',
            'nullable' => false
        ],
        'user_email' => [
            'type' => 'VARCHAR128',
            'validate'=> 'required|unique',
            'nullable' => false
        ],
        'user_phone' => [
            'type' => 'VARCHAR128',
            'nullable' => false
        ],
        'user_address' => [
```

```
    'type' => 'VARCHAR128',
    'nullable' => false
],
'user_position' => [
    'type' => 'VARCHAR128',
    'nullable' => false
],
'user_token' => [
    'type' => 'VARCHAR128',
    'nullable' => false
],
'user_level' => [
    'type' => 'INT1',
    'validate'=> 'required',
    'nullable' => false
],
'user_desc' => [
    'type' => 'VARCHAR128',
    'nullable' => false
],
'user_onesignal_id' => [
    'type' => 'VARCHAR128',
    'nullable' => false
],
'is_trash' => [
    'type' => 'INT1',
    'nullable' => false
],
'created_at' => [
    'type' => 'DATETIME',
    'nullable' => false
```

```
    ],
    'updated_at' => [
        'type' => 'DATETIME',
        'nullable' => false
    ],
    'created_by' => [
        'type' => 'INT4',
        'nullable' => false
    ],
    'updated_by' => [
        'type' => 'INT4',
        'nullable' => false
    ]
],
$db = 'DB',
$table = 'users';

const ADMIN_LEVEL = 0;
const ADMIN_FAKE_LEVEL = 1;
const USER_LEVEL = 2;

function __construct() {
            parent::__construct();
        }

public function set_user_password($value) {
    return !empty($value) ? \Bcrypt::instance()->hash($value) : $this->user_password;
}

public function set_user_token($value) {
    return sha1($this->user_name);
```

```
}

public function set_user_level($value) {
   return $this->user_position === 'staff' ? 1 : 0;
}

public function set_user_code($value) {
   return $this->generateCode();
}

public function generateCode(){
            return 'US'.$this->generateNumber();
}

public function getLevelLabel($status = NULL)
      {
            $getStatus = array(
                  self::ADMIN_LEVEL => 'Owner',
                  self::USER_LEVEL => 'Staff',
            );

            if ($status === NULL) {
                  return $getStatus;
            } else {
                  return isset($getStatus[$status]) ? $getStatus[$status] : '';
            }
}

      public function validation($post){
   $validator= \Validate::instance();
```

```
$validator->add_validator("unique", function($field, $input, $param = NULL) {
    if(\Base::instance()->get('action') !== 'update'){
        $uniq = [["$field = ?", $input[$field]]];
        if(!empty($this->_id))
                        array_unshift($uniq, ["id != ?", $this->_id]);

        $filter = $this->mergeFilter($uniq, 'and');
        return $this->count($filter) ? false : true;
    }else{
        return true;
    }
});

if(\Base::instance()->get('action') === 'update'){
    $rmv = ['user_code', 'user_password'];
    foreach ($rmv as $value) {
        $this->fieldConf[$value]["validate"] = '';
    }
}

$validator->add_cortex($this);
            if($validator->run($post))
    $isTrue = true;
            else
    $isTrue = false;

            if(!empty($post)){
    foreach ($validator->get_errors_array() as $field=>$error) {
        \Flash::instance()->setKey($field, $error);
                    }
            }
```

```
                return $isTrue;
    }

    public function genJWT($email) {
        $this->fields(array('user_password', 'user_password_old', 'created_by', 'updated_by'),
true); // all fields, but not these
        $this->load(array('user_email = ?', $email));
        $payload = array(
          "user" => $this->cast(),
          "exp" => time()+604800
        );

        return JWT::encode($payload, UrlComponent::$privateKey, 'RS256');
      }



    public function auth() {
        $this->load(array('user_name = ?', $_POST['user_name']));
        //echo 'asd';
                if (\Bcrypt::instance()->verify($_POST['user_password'], $this-
>user_password)) {
            return json_encode($this->genJWT());
        }
    }


        public function searchBy($v){
    return empty($_GET[$v])?":array("$v LIKE ?", '%'.$_GET[$v].'%');
    }


        public function getData(){
                $sf = $_GET['sort'];
```

```
                $so = $_GET['order'];
                $limit = $_GET['limit'];
                $offset = $_GET['offset'] * $_GET['limit'];
        $total = $this->count();
        $query = ['is_trash != ?', 1];

                $filter = array_filter(array_map([get_called_class(), 'searchBy'], $this->fields()), 'ArrayComponent::notEmpty');

        if(isset($_GET['for'])) array_unshift($filter, ['id != ?', 1]);

        //array_unshift($filter, ['AND is_trash != ?', 1]);

                $merfil = !empty($filter) ? $this->mergeFilter([$query, $this->mergeFilter($filter ,'or')]) : $query;
                $options = array('limit' => $limit, 'offset' => $offset, 'order' => empty($sf)?"":"$sf $so");

        $model = $this->find($merfil, $options);

        return ['data' => $model ? $model->castAll() : [] , 'total' => $total];
    }

}
```

Source Code CrownJob SendNotification

```
<?php

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, "https://api.scheduler.my.id/xxxx");
curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type: application/json;
charset=utf-8'));
curl_setopt($ch, CURLOPT_RETURNTRANSFER, TRUE);
curl_setopt($ch, CURLOPT_HEADER, FALSE);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, FALSE);
$response = curl_exec($ch);

curl_close($ch);
print(json_encode($response));
```