Plan de Desarrollo: Sistema de Firma de Recibos de Sueldo

Estructura del Proyecto (CodeIgniter)

Para organizar mejor el proyecto, recomiendo la siguiente estructura de carpetas y archivos:

```
application/
├── config/
│   ├── config.php         (Configuración general)
│   ├── database.php       (Configuración de la BD)
│   └── email.php          (Configuración de emails)
├── controllers/
│   ├── Admin.php          (Panel administrador)
│   ├── Auth.php           (Login/registro)
│   ├── Dashboard.php      (Panel principal)
│   ├── Documentos.php     (Gestión de documentos)
│   ├── Empleados.php      (Gestión de empleados)
│   └── Firmas.php         (Gestión de firmas)
├── models/
│   ├── Usuario_model.php  (Modelo de usuarios/empleados)
│   ├── Documento_model.php(Modelo de documentos/recibos)
│   └── Firma_model.php    (Modelo de firmas)
├── views/
│   ├── admin/             (Vistas de administración)
│   ├── auth/              (Vistas de login/registro)
│   ├── dashboard/         (Vistas del dashboard)
│   ├── documentos/        (Vistas de documentos)
│   └── firmas/            (Vistas de firma)
└── libraries/
    └── Pdf_processor.php  (Librería para procesar PDFs)
```

Etapas y Pasos para el Desarrollo

Etapa 1: Configuración inicial (1 semana)

1. **Sistema de autenticación**
   - Configurar login para administradores
   - Configurar login para empleados
   - Gestión de sesiones
2. **Configuración de la BD**
   - Revisar la estructura existente
   - Confirmar que incluye tablas para:
     - Usuarios (con roles: admin/empleado)
     - Documentos
     - Firmas
3. **Estructura básica del panel admin**
   - Dashboard principal
   - Menú de navegación
   - Plantillas base

Etapa 2: Gestión de empleados (1 semana)

1. **CRUD de empleados**
   - Formulario para añadir empleados
   - Listado de empleados
   - Edición y eliminación
2. **Importación masiva**
   - Opción para subir CSV/Excel con datos de empleados
   - Procesamiento del archivo

Etapa 3: Procesamiento de documentos PDF (2 semanas)

1. **Subida de PDFs**
   - Formulario para subir PDFs
   - Validación de formatos
   - Almacenamiento en servidor
2. **Procesamiento de PDFs**
   - Desarrollo de algoritmo para identificar DNI en los PDFs
   - Separación en documentos individuales
   - Asignación a empleados según DNI
3. **Visualización de documentos**
   - Vista previa de PDFs
   - Listado de documentos asignados

Etapa 4: Sistema de firmas (1-2 semanas)

1. **Interfaz de firma**
   - Visualización del documento
   - Opciones de conformidad/no conformidad
   - Campo para comentarios (en caso de no conformidad)
2. **Proceso de firma**
   - Validación de identidad del empleado
   - Registro de timestamp
   - Almacenamiento de estado de firma

Etapa 5: Sistema de notificaciones (1 semana)

1. **Configuración de correo**
   - Integración con servidor SMTP
   - Diseño de plantillas de email
2. **Notificaciones automáticas**
   - Email cuando se asigna un documento
   - Email de recordatorio para documentos pendientes
   - Email de confirmación tras firma

Etapa 6: Dashboard y reportes (1 semana)

1. **Dashboard administrativo**
   - Estadísticas de documentos (pendientes, firmados, rechazados)
   - Filtros por fecha, empleado, tipo de documento
2. **Sistema de reportes**
   - Exportación a Excel/PDF
   - Listados de conformidad/no conformidad

Etapa 7: Pruebas y ajustes (1 semana)

1. **Pruebas de usuario**
   - Flujo completo del proceso
   - Casos de uso especiales
2. **Optimización y mejoras**
   - Ajustes de rendimiento
   - Mejoras en la UI/UX

Tecnologías y Herramientas Recomendadas

Para el procesamiento de PDFs:

- **FPDI/FPDF**: Para manipular archivos PDF
- **pdfparser/pdfparser**: Para extraer texto y contenido de PDFs

Para las notificaciones por email:

- Librería nativa de CodeIgniter para emails
- Alternativa: PHPMailer para opciones más avanzadas

Para el frontend:

- Bootstrap para UI responsiva
- jQuery para interactividad
- DataTables para listados

Para la firma digital:

- SignaturePad.js para captura de firma
- Alternativa: Sistema basado en autenticación con timestamp

Código Base para Empezar

1. Controlador de Documentos

php

```php
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Documentos extends CI_Controller {

    public function __construct() {
        parent::__construct();
        // Cargar modelos
        $this->load->model('documento_model');
        $this->load->library('pdf_processor');

        // Verificar login
        if (!$this->session->userdata('logged_in')) {
            redirect('auth/login');
        }
    }

    public function index() {
        $data['documentos'] = $this->documento_model->get_all();
        $this->load->view('documentos/listado', $data);
    }

    public function upload() {
        $this->load->view('documentos/upload');
    }

    public function process_upload() {
        // Configuración para subida de archivos
        $config['upload_path'] = './uploads/';
        $config['allowed_types'] = 'pdf';
        $config['max_size'] = 5000; // 5MB

        $this->load->library('upload', $config);

        if (!$this->upload->do_upload('documento')) {
            $error = $this->upload->display_errors();
            $this->session->set_flashdata('error', $error);
            redirect('documentos/upload');
        } else {
            // Archivo subido correctamente
            $upload_data = $this->upload->data();
            $file_path = $upload_data['full_path'];

            // Procesar el PDF
            $result = $this->pdf_processor->process_recibos($file_path);

            if ($result['status'] == 'success') {
                $this->session->set_flashdata('success', 'Se procesaron ' . $result['count'] .
            } else {
                $this->session->set_flashdata('error', 'Error al procesar: ' . $result['message
            }

            redirect('documentos');
        }
    }
```

```
    public function view($id) {
        $data['documento'] = $this->documento_model->get_by_id($id);
        $this->load->view('documentos/view', $data);
    }
}
```

2. Librería para procesar PDFs

php

```php
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Pdf_processor {

    protected $CI;

    public function __construct() {
        $this->CI =& get_instance();
        $this->CI->load->model('documento_model');
        $this->CI->load->model('usuario_model');
    }

    public function process_recibos($file_path) {
        // Aquí implementaremos la lógica para:
        // 1. Extraer texto del PDF para buscar DNIs
        // 2. Separar el PDF en archivos individuales
        // 3. Asignar a los empleados correspondientes

        try {
            // Este es un esquema básico, se implementará completamente más adelante
            $parser = new \Smalot\PdfParser\Parser();
            $pdf = $parser->parseFile($file_path);

            $pages = $pdf->getPages();
            $processed = 0;

            foreach ($pages as $page) {
                $text = $page->getText();

                // Buscar DNI en el texto (patrón simplificado)
                preg_match('/DNI[:\s]*([0-9]{7,8})/i', $text, $matches);

                if (isset($matches[1])) {
                    $dni = $matches[1];
                    $empleado = $this->CI->usuario_model->get_by_dni($dni);

                    if ($empleado) {
                        // Guardar este recibo para este empleado
                        // Aquí falta implementar la lógica para extraer la página individual

                        $documento_data = array(
                            'usuario_id' => $empleado->id,
                            'nombre' => 'Recibo ' . date('Y-m-d'),
                            'ruta' => 'ruta/al/archivo/individual.pdf', // Esto se implementará
                            'estado' => 'pendiente',
                            'fecha_creacion' => date('Y-m-d H:i:s')
                        );

                        $this->CI->documento_model->insert($documento_data);
                        $processed++;
                    }
                }
            }

            return array('status' => 'success', 'count' => $processed);
```

```
        } catch (Exception $e) {
            return array('status' => 'error', 'message' => $e->getMessage());
        }
    }
}
```

3. Modelo de documentos

php

```php
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Documento_model extends CI_Model {

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

    public function get_all() {
        return $this->db->get('documentos')->result();
    }

    public function get_by_id($id) {
        return $this->db->where('id', $id)->get('documentos')->row();
    }

    public function get_by_usuario($usuario_id) {
        return $this->db->where('usuario_id', $usuario_id)->get('documentos')->result();
    }

    public function insert($data) {
        $this->db->insert('documentos', $data);
        return $this->db->insert_id();
    }

    public function update($id, $data) {
        $this->db->where('id', $id);
        return $this->db->update('documentos', $data);
    }

    public function delete($id) {
        $this->db->where('id', $id);
        return $this->db->delete('documentos');
    }

    public function count_by_estado($estado) {
        $this->db->where('estado', $estado);
        return $this->db->count_all_results('documentos');
    }
}
```

Próximos Pasos Recomendados

1. Revisar y ajustar la estructura de la base de datos si es necesario

2. Implementar el sistema de autenticación
3. Crear los modelos básicos para usuarios y documentos
4. Configurar el panel de administración
5. Implementar la subida y procesamiento básico de PDFs

Una vez que estos componentes estén funcionando, podemos avanzar con la lógica más compleja para:

- Procesar y separar PDFs por DNI
- Implementar el sistema de firma digital
- Configurar las notificaciones por email