



DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

**TRABAJO DE TITULACIÓN, PREVIO LA OBTENCIÓN DEL
TÍTULO DE INGENIERO EN SISTEMAS E INFORMÁTICA**

**TEMA: DESARROLLO DE UN SISTEMA DISTRIBUIDO PARA
LA DIGITALIZACIÓN Y PROCESAMIENTO DE CHEQUES
USANDO ALGORITMOS DE RECONOCIMIENTO DE DÍGITOS
MANUSCRITOS EN LA EMPRESA DECISIÓN C.A.**

**AUTORES: BENALCÁZAR CABRERA, SANTIAGO DAVID
LLUMIQUINGA LUCERO, GERARDO VINICIO**

DIRECTORA: NOBOA MORALES, TATIANA KARINA

SANGOLQUÍ

2016



DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
CARRERA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

CERTIFICADO

Certifico que el trabajo de titulación, "DESARROLLO DE UN SISTEMA DISTRIBUIDO PARA LA DIGITALIZACIÓN Y PROCESAMIENTO DE CHEQUES USANDO ALGORITMOS DE RECONOCIMIENTO DE DÍGITOS MANUSCRITOS EN LA EMPRESA DECISIÓN C.A." realizado por el Sr. SANTIAGO DAVID BENALCÁZAR CABRERA y el Sr. GERARDO VINICIO LLUMIQUINGA LUCERO, ha sido revisado en su totalidad y analizado por el software anti-plagio con un resultado de cero por ciento (0%), el mismo cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de las Fuerzas Armadas ESPE, por lo tanto me permito acreditarlo y autorizar al Sr. SANTIAGO DAVID BENALCÁZAR CABRERA y el Sr. GERARDO VINICIO LLUMIQUINGA LUCERO para que lo sustenten públicamente.

Sangolquí, 30 de junio de 2016

ING. TATIANA NOBOA

DIRECTORA DE TESIS



DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
CARRERA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

AUTORÍA DE RESPONSABILIDAD

Nosotros, SANTIAGO DAVID BENALCÁZAR CABRERA, con cédula de identidad N° 1725030850 y GERARDO VINICIO LLUMIQUINGA LUCERO, con cédula de identidad N° 1722713136, declaramos que este trabajo de titulación "DESARROLLO DE UN SISTEMA DISTRIBUIDO PARA LA DIGITALIZACIÓN Y PROCESAMIENTO DE CHEQUES USANDO ALGORITMOS DE RECONOCIMIENTO DE DÍGITOS MANUSCRITOS EN LA EMPRESA DECISIÓN C.A." ha sido desarrollado considerando los métodos de investigación existentes, así como también se ha respetado los derechos intelectuales de terceros considerándose en las citas bibliográficas.

Consecuentemente declaramos que este trabajo es de nuestra autoría, en virtud de ello nos declaramos responsables del contenido, veracidad y alcance de la investigación mencionada.

Sangolquí, 30 de junio de 2016

Santiago David Benalcázar Cabrera

C.C. 1725030850

Gerardo Vinicio Llumiquinga Lucero

C.C. 1722713136



DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
CARRERA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

AUTORIZACIÓN

Nosotros, SANTIAGO DAVID BENALCÁZAR CABRERA y GERARDO VINICIO LLUMIQUINGA LUCERO, autorizamos a la Universidad de las Fuerzas Armadas ESPE la publicación, en la biblioteca virtual de la Institución, el trabajo: "DESARROLLO DE UN SISTEMA DISTRIBUIDO PARA LA DIGITALIZACIÓN Y PROCESAMIENTO DE CHEQUES USANDO ALGORITMOS DE RECONOCIMIENTO DE DÍGITOS MANUSCRITOS EN LA EMPRESA DECISIÓN C.A." cuyo contenido, ideas y criterios son de nuestra exclusiva responsabilidad y autoría.

Sangolquí, 30 de junio de 2016

Santiago David Benalcázar Cabrera

C.C. 1725030850

Gerardo Vinicio Llumiquinga Lucero

C.C. 1722713136

DEDICATORIA

A mis padres por todo el apoyo que me han brindado a lo largo de este proceso.

Santiago Benalcázar C.

AGRADECIMIENTO

A mi familia que han estado siempre conmigo para brindarme su apoyo incondicional y guía permanente.

A todas las personas que han estado involucradas en este proceso y han aportado para su culminación.

A la empresa DECISIÓN c.a. por su colaboración en el desarrollo de este proyecto.

A las comunidades de software libre y desarrolladores independientes que brindan su apoyo desinteresado y contribuyen al crecimiento del software.

DEDICATORIA

El presente proyecto está dedicado a todas las personas que estuvieron junto a mí en toda mi vida académica, especialmente a mi familia por estar en los momentos más difíciles.

Pero especialmente le dedico a mi madre por el esfuerzo y sacrificio que ha hecho para ser una persona de bien, también le agradezco por brindarme su confianza y apoyo incondicional en cada momento de mi vida.

Gerardo Llumiyinga L.

AGRADECIMIENTO

A mis padres, Gerardo Llumiquinga y María Lucero por el infinito amor y el constante soporte que me han brindado en el transcurso de mi vida.

También agradezco a mis hermanos, por sus consejos que han hecho que siga adelante, compartiendo tristezas y alegrías en este proceso de formación académica.

Del mismo modo agradezco a DECISIÓN c.a. por darme la oportunidad de crecer profesionalmente realizado el presente proyecto.

ÍNDICE

CERTIFICADO	ii
AUTORÍA DE RESPONSABILIDAD	iii
AUTORIZACIÓN	iv
DEDICATORIA	v
AGRADECIMIENTO	vi
RESUMEN	xviii
ABSTRACT	xix
CAPITULO I.....	1
INTRODUCCIÓN	1
1.1 Antecedentes.....	1
1.2 Problemática.....	2
1.3 Justificación	2
1.4 Objetivos.....	3
1.5 Alcance	3
CAPITULO II.....	5
MARCO TEÓRICO	5
2.1 Documento Financiero: Cheque	5
2.2 Caracteres de Tinta Magnética.....	6
2.3 Sistema de Cámara de Compensación de Cheques	6
2.4 Digitalización de imágenes	8
2.5 Computación Visual.....	9
2.5.1 Librería OpenCV	9
2.5.2 Librería Sklearn.....	10
2.5.3 Algoritmos de Reconocimiento de Caracteres	10

	x
2.6 Representational State Transfer (REST)	40
2.7 Autenticación Basada en Tokens	40
2.7.1 JSON Web Token (JWT).....	40
2.8 Metodología de Desarrollo De Software	44
2.8.1 Extreme Programming	44
2.9 Frameworks de Desarrollo	46
2.9.1 Django Framework	46
2.9.2 Django REST Framework	46
2.9.3 AngularJS.....	46
2.9.4 .NET Framework	46
2.10 Lenguajes De Programación	47
2.10.1 Python.....	47
2.10.2 Javascript.....	47
2.10.3 HTML5	47
2.10.4 CSS	47
2.10.5 C#	48
2.11 Motores de Base De Datos.....	48
2.11.1 PostgreSQL	48
2.11.2 SQLite.....	48
CAPITULO III.....	49
ANÁLISIS Y DISEÑO DE LA APLICACIÓN	49
3.1 Diagrama Físico del Sistema	49
3.2 Elaboración.....	50
3.2.1 Usuarios	50
3.2.2 Historias de Usuario	51
3.2.3 Diagramas De Secuencia.....	65
3.2.4 Diseño de la Base de Datos	70

	xi
CAPITULO IV	78
DESARROLLO E IMPLEMENTACIÓN	78
4.1 Arquitectura del Sistema.....	78
4.1.1 Cliente Web - AngularJS	79
4.1.2 Servidor Python - Django Rest API	81
4.2 Funcionamiento del Sistema.....	90
4.2.1 Pantalla de Inicio de Sesión	90
4.2.2 Vista Principal	91
4.2.3 Menú de Accesos.....	92
4.2.4 Operaciones CRUD.....	92
4.2.5 Digitalización	93
4.2.6 Aprobación de Depósitos	96
4.2.7 Reportes.....	97
CAPITULO V	99
CONCLUSIONES Y RECOMENDACIONES.....	99
5.1 Conclusiones	99
5.2 Recomendaciones	100
REFERENCIAS BIBLIOGRÁFICAS	101

ÍNDICE DE FIGURAS

Figura 1 Partes del cheque	5
Figura 2 Caracteres de tinta magnética	6
Figura 3 Sistema de Cámara de Compensación de Cheques	7
Figura 4. Antiguo proceso de la cámara de compensación de cheques.....	8
Figura 5. Digitalización de cheques	9
Figura 6 Las diferentes áreas del reconocimiento de caracteres.....	11
Figura 7. Algoritmo k-NN	12
Figura 8 Imagen de Prueba de Dígitos Manuscritos	13
Figura 9 Script Algoritmo k-NN	14
Figura 10 Estructura del modelo SVM linear con características de HOG...	15
Figura 11 Identificación de gradientes en una imagen.....	16
Figura 12 Histograma de gradientes.....	16
Figura 13 Support Vector Machines.....	17
Figura 14 Muestra Dígitos Manuscritos Base de Datos MNIST	18
Figura 15 Fragmento Script Algoritmo SVM - Modulos.....	19
Figura 16 Fragmento Script Algoritmo SVM - MNIST	19
Figura 17 Fragmento Script Algoritmo SVM - Arrays	19
Figura 18 Fragmento Script Algoritmo SVM – Características HOG.....	20
Figura 19 Fragmento Script Algoritmo SVM – Clasificador.....	20
Figura 20 Fragmento Script Algoritmo SVM - Modulos.....	20
Figura 21 Fragmento Script Algoritmo SVM – Cargar clasificador.....	20
Figura 22 Fragmento Script Algoritmo SVM – Cargar imagen	21
Figura 23 Fragmento Script Algoritmo SVM – Convertir imagen	21
Figura 24 Fragmento Script Algoritmo SVM – Calcular contornos.....	21
Figura 25 Fragmento Script Algoritmo SVM - Predicción.....	22
Figura 26 Fragmento Script Algoritmo SVM - Resultados	22
Figura 27 Prueba 01 Algoritmo SVM	23
Figura 28 Prueba 02 Algoritmo SVM	24
Figura 29 Prueba 03 Algoritmo SVM	25
Figura 30 Prueba 04 Algoritmo SVM	26
Figura 31 Prueba 05 Algoritmo SVM	27

	xiii
Figura 32 Prueba 06 Algoritmo SVM	28
Figura 33 Prueba 07 Algoritmo SVM	29
Figura 34 Prueba 08 Algoritmo SVM	30
Figura 35 Prueba 09 Algoritmo SVM	31
Figura 36 Prueba 10 Algoritmo SVM	32
Figura 37 Resultados Prueba Algoritmo SVM.....	33
Figura 38 Prueba 01 en Cheque Algoritmo SVM	34
Figura 39 Prueba 02 en Cheque Algoritmo SVM	34
Figura 40 Prueba 03 en Cheque Algoritmo SVM	35
Figura 41 Prueba 04 en Cheque Algoritmo SVM	35
Figura 42 Prueba 05 en Cheque Algoritmo SVM	36
Figura 43 Prueba 06 en Cheque Algoritmo SVM	36
Figura 44 Prueba 07 en Cheque Algoritmo SVM	37
Figura 45 Prueba 08 en Cheque Algoritmo SVM	37
Figura 46 Prueba 09 en Cheque Algoritmo SVM	38
Figura 47 Prueba 10 en Cheque Algoritmo SVM	38
Figura 48 Resultado de Pruebas en Cheques	39
Figura 49 Resultado de Pruebas en Cheques: Porcentajes	39
Figura 50 Estructura JSON Web Token.....	41
Figura 51 Header JSON Web Token	41
Figura 52 Payload JSON Web Token	42
Figura 53 Signature JSON Web Token.....	42
Figura 54 JSON Web Token Codificado	43
Figura 55 Authorization JSONN Web Token	43
Figura 56 Diagrama de Secuencia JSON Web Token.....	44
Figura 57 Fases Metodología Extreme Programming.....	45
Figura 58 Diagrama Físico del Sistema	49
Figura 59 Diagrama de Secuencia: Inicio de Sesión	65
Figura 60 Diagrama de Secuencia: Operación CRUD (Sucursal).....	66
Figura 61 Diagrama de Secuencia: Módulo Digitalización	67
Figura 62 Diagrama de Secuencia: Aprobación de Depósito	68
Figura 63 Diagrama de Secuencia: Reporte Entidad Financiera	69
Figura 64 Base de Datos: Diagrama Entidad Relación.....	70

	xiv
Figura 65 Base de Datos: Módulo Digitalización.....	71
Figura 66 Base de Datos: Módulo de Seguridad	71
Figura 67 Base de Datos: Tabla cabecera proceso	72
Figura 68 Base de Datos: Tabla cheque.....	72
Figura 69 Base de Datos: Tabla ciudad.....	73
Figura 70 Base de Datos: Tabla consulta cliente.....	73
Figura 71 Base de Datos: Tabla cuenta bancaria.....	74
Figura 72 Base de Datos: Tabla detalle de depósito	74
Figura 73 Base de Datos: Tabla empresa	74
Figura 74 Base de Datos: Tabla entidad financiera	75
Figura 75 Base de Datos: Tabla país.....	75
Figura 76 Base de Datos: Tabla persona	75
Figura 77 Base de Datos: Tabla proceso.....	76
Figura 78 Base de Datos: Tabla provincia	76
Figura 79 Base de Datos: Tabla scanner.....	76
Figura 80 Base de Datos: Tabla sucursal	77
Figura 81 Base de Datos: Tabla cuenta bancaria.....	77
Figura 82 Base de Datos: Tabla tipo de cuenta.....	77
Figura 83 Arquitectura del Sistema.....	79
Figura 84 AngularJS: Servicio.....	79
Figura 85 AngularJS: Controlador.....	80
Figura 86 AngularJS: Template	81
Figura 87 Django REST: Controlador	82
Figura 88 Django REST: Serializador	82
Figura 89 Django REST: Modelo	83
Figura 90 Pantalla de Inicio de Sesión.....	90
Figura 91 Vista Principal del Sistema	91
Figura 92 Menú de Accesos	92
Figura 93 Operación CRUD: Lista Registros	92
Figura 94 Operación CRUD: Formulario Ingreso/Edición del Registro	93
Figura 95 Operación CRUD: Validaciones formulario.....	93
Figura 96 Módulo Digitalización: Vista Principal	94
Figura 97 Módulo Digitalización: Vista de espera (digitalizando)	94

	xv
Figura 98 Módulo Digitalización: Proceso Completo.....	95
Figura 99 Módulo Digitalización: Controles de navegación de imágenes	95
Figura 100 Aprobación de Depósitos.....	96
Figura 101 Comprobante de Depósito	96
Figura 102 Parámetros de Búsqueda del Reporte de Entidad Financiera ...	97
Figura 103 Parámetros de Búsqueda del Reporte de Empresa	97
Figura 104 Detalle del Reporte	98

ÍNDICE DE TABLAS

Tabla 1 Información de dígitos manuscritos MNIST	18
Tabla 2 Resultado Prueba 01 Algoritmo SVM	23
Tabla 3 Resultado Prueba 02 Algoritmo SVM	24
Tabla 4 Resultado Prueba 03 Algoritmo SVM	25
Tabla 5 Resultado Prueba 04 Algoritmo SVM	26
Tabla 6 Resultado Prueba 05 Algoritmo SVM	27
Tabla 7 Resultado Prueba 06 Algoritmo SVM	28
Tabla 8 Resultado Prueba 07 Algoritmo SVM	29
Tabla 9 Resultado Prueba 08 Algoritmo SVM	30
Tabla 10 Resultado Prueba 09 Algoritmo SVM.....	31
Tabla 11 Resultado Prueba 10 Algoritmo SVM.....	32
Tabla 12 Historia de Usuario N° 1.....	51
Tabla 13 Historia de Usuario N° 2.....	52
Tabla 14 Historia de Usuario N° 3.....	53
Tabla 15 Historia de Usuario N° 4.....	54
Tabla 16 Historia de Usuario N° 5.....	55
Tabla 17 Historia de Usuario N° 6.....	56
Tabla 18 Historia de Usuario N° 7.....	57
Tabla 19 Historia de Usuario N° 8.....	58
Tabla 20 Historia de Usuario N° 9.....	59
Tabla 21 Historia de Usuario N° 10.....	60
Tabla 22 Historia de Usuario N° 11.....	61
Tabla 23 Historia de Usuario N° 12.....	62
Tabla 24 Historia de Usuario N° 13.....	63
Tabla 25 Historia de Usuario N° 14.....	64
Tabla 26 Servicios REST: Autorización	83
Tabla 27 Servicios REST: Usuarios.....	83
Tabla 28 Servicios REST: Personas.....	84
Tabla 29 Servicios REST: Países.....	84
Tabla 30 Servicios REST: Provincias	84
Tabla 31 Servicios REST: Ciudades.....	85

Tabla 32 Servicios REST: Entidades Financieras.....	85
Tabla 33 Servicios REST: Empresas	86
Tabla 34 Servicios REST: Sucursales	86
Tabla 35 Servicios REST: Escáneres	86
Tabla 36 Servicios REST: Tipos de Cuenta Bancaria.....	87
Tabla 37 Servicios REST: Cuentas Bancarias.....	87
Tabla 38 Servicios REST: Procesos	87
Tabla 39 Servicios REST: Cabecera de Proceso	88
Tabla 40 Servicios REST: Cheques.....	88
Tabla 41 Servicios REST: Consulta de Cliente.....	88
Tabla 42 Servicios REST: Consulta de Entidad Financiera	89
Tabla 43 Servicios REST: Consulta de Depósitos por Cliente.....	89
Tabla 44 Servicios REST: Detalle del Depósito	89

RESUMEN

En la actualidad el Banco Central del Ecuador permite el uso de imágenes digitalizadas para el proceso de cámara de compensación de cheques por lo que el presente trabajo tiene como objetivo desarrollar un sistema de digitalización y procesamiento de cheques para agilizar el proceso de depósito remoto, haciendo uso de lenguajes de alto nivel y estándares web acordes a las necesidades actuales del mercado. Se usó la metodología Extreme Programming para el desarrollo de un sistema web que consta de tres aplicaciones: la primera es aplicación web de una sola página utilizando el framework AngularJS junto con HTML5 y CSS3, la segunda es un servidor REST que usa el framework Django de Python y un gestor de base de datos PostgreSQL para manejar toda la lógica del negocio y la tercera es un servidor REST elaborado con C# que permite la comunicación del escáner de cheques con el aplicativo web. En el desarrollo del módulo de digitalización se analizó el índice de error de los algoritmos de reconocimiento de dígitos manuscritos: K Nearest Neighbors (k-NN) para el aprendizaje y Support Vector Machines (SVM) para las pruebas. El sistema permite la administración de depósitos remotos de cheques mediante una interfaz web y genera reportes que facilitan la búsqueda y clasificación de los depósitos por parte de administradores tanto de una empresa como de una entidad financiera. Con este proyecto se estandarizó y centralizó los procesos de depósito remoto de cheques obteniendo un ahorro de tiempo y recursos a los usuarios.

PALABRAS CLAVE:

- **RECONOCIMIENTO DE CARACTERES MANUSCRITOS**
- **COMPUTACIÓN VISUAL**
- **SVM**
- **K-NN**

ABSTRACT

Nowadays the Central Bank of Ecuador allows the use of digital images for the cheque compensation camera process, which is why the present work aims to develop a system of digitalization and cheque processing to streamline the process of remote deposit, using high-level languages and web standards according with the current market needs. Extreme Programming methodology was used for developing a web system that consists of three applications: the first is a Single Page Application (SPA) using the AngularJS Framework along with HTML5 y CSS3, the second is a REST Server that uses the Django Framework of Python and a PostgreSQL DBMS to handle the business logic and the third application is a REST Server made with C# which allows to communicate the scanner with the web app. In the development of the digitalization module the error index of the handwritten character recognition algorithms was analyzed: K Nearest Neighbors (k-NN) for the learning and Support Vector Machine for the tests. The system allows the management of cheque remote deposits through a web interface and generates reports that facilitate the search and classifying of deposits by company and financial entity administrators. This project aims to standardize and centralize the process of cheque remote deposits to save time and resources to the users.

KEYWORDS:

- **HANDWRITTEN CHARACTER RECOGNITION**
- **COMPUTER VISION**
- **SVM**
- **K-NN**

CAPITULO I

INTRODUCCIÓN

1.1 Antecedentes

Para las instituciones financieras el manejo y administración de los cheques supone un arduo y costoso trabajo, ya que el proceso desde que ingresa el cheque hasta que se hace efectivo, consta de varios pasos, algunos de ellos implican la manipulación directa del documento para el registro en los diferentes bancos, de tal manera que pueden ser modificados por personas inescrupulosas. Hoy en día se contrata empresas de seguridad dedicadas a la custodia y logística de los cheques, lo cual representa un gasto adicional para las entidades financieras, por tal motivo es necesario establecer un proceso que ayude a mantener la integridad de la información y reduzca tiempo y dinero en el transporte de estos documentos.

El 29 de septiembre de 2014 entró en vigencia el “Reglamento General de la Ley de Cheques” emitida por la Superintendencia de Bancos y Seguros en la resolución No. SBS-2014-234 (Superintendencia de Bancos y Seguros del Ecuador, 2014), el cual establece reglas al momento de efectivizar un cheque para que sea válido:

- El cheque no puede estar roto, mutilado ni deteriorado.
- No debe contener alteraciones, como borrones o tachones.
- Llenar los datos con esfero de tinta azul o negra.
- No escribir sellos o leyendas que condicionen la entrega del cheque a sus destinatarios.
- No usar cintas adhesivas sobre las cifras.

1.2 Problemática

Debido a la renovación del Sistema de Cámara de Compensación de Cheques (SCCC) por parte del Banco Central del Ecuador (BCE), DECISIÓN c.a. se ve en la necesidad de adaptar su sistema de procesamiento de cheques al nuevo Reglamento General de la Ley de Cheques y a los nuevos procesos que implantó el BCE.

La gestión que debe realizar una entidad financiera para el procesamiento de cheques es costosa, tanto en recursos financieros como tiempo; además, la información que éstos manejan es de vital importancia para su negocio, por lo cual se pretende desarrollar un aplicativo que automatice el proceso de digitalización, consultas y administración de estos documentos, que se adapte al nuevo sistema de cámara de compensación de cheques y que permita eliminar procesos y reducir el tiempo de entrega de los mismos.

1.3 Justificación

El aplicativo de digitalización de cheques hará que las empresas a las que DECISIÓN c.a. presta sus servicios puedan adquirir los siguientes beneficios:

- Agilizar el proceso de digitalización de los cheques, haciendo uso de nuevas tecnologías y escáneres.
- Disminuir costos al optimizar el proceso de transporte, así como el tiempo del personal encargado de realizar este trabajo.
- Aumentar la eficiencia en los servicios al disponer inmediatamente de las imágenes de cheques para ser procesados.
- Permitir el intercambio automático de imágenes de cheques y documentos entre diferentes entidades o centros de acopio.

1.4 Objetivos

a. Objetivo General

Desarrollar un sistema de digitalización de cheques que permita la administración remota y local de los mismos.

b. Objetivos Específicos

- i. Investigar algoritmos de reconocimiento dígitos manuscritos utilizando librerías de computación visual.
- ii. Analizar y documentar el funcionamiento de los algoritmos para seleccionar el más idóneo para el sistema.
- iii. Implementar un algoritmo que permita el reconocimiento de caracteres de tinta magnética de un cheque utilizando la API¹ provista por CTS².
- iv. Desarrollar una aplicación web utilizando lenguajes de alto nivel para la administración de los cheques.
- v. Aplicar algoritmos de cifrado para la autenticación y transmisión de los datos de los usuarios del sistema.

1.5 Alcance

El presente proyecto tiene como finalidad establecer un proceso ágil para la digitalización de cheques que consta de tres aplicaciones:

La primera permite la comunicación entre el escáner CTS y una aplicación web, enfocado en la captura de la imagen digital del cheque y sus

¹ Application Program Interface (API) es un conjunto de rutinas, protocolos y herramientas para el desarrollo de software. (Richardson, Amundsen, & Ruby, 2013)

² CTS Electronics es una empresa que provee soluciones para el procesamiento de dinero y cheques para cajeros, bancos, etc. mediante dispositivos electrónicos: escáneres, cajeros automáticos, emisores de tarjetas bancarias. (CTS electronics, 2015)

metadatos haciendo uso de algoritmos de reconocimiento de caracteres de tinta magnética para su posterior clasificación y administración.

La segunda es un API REST³ que implementa servicios para la comunicación entre la base de datos y la lógica del negocio, así como la administración de usuarios y seguridad del sistema.

La tercera es una aplicación para administrar los cheques digitalizados desde la web, la cual permitirá que las empresas y entidades financieras puedan visualizar y procesar sus documentos. La aplicación web contará con:

- El uso e integración de escáneres CTS para la digitalización de cheques y reconocimiento de caracteres de tinta magnética.
- La generación de comprobantes para el control de los depósitos realizados.
- Reportes que facilitan la búsqueda y clasificación de los depósitos.

Las aplicaciones a desarrollar contarán con seguridades, controles de acceso para diferentes roles de usuarios y un módulo de administración con el objetivo de garantizar la integridad de la información.

³ Representational State Transfer (REST) (Ver página 40)

CAPITULO II

MARCO TEÓRICO

2.1 Documento Financiero: Cheque

El cheque es un documento financiero, contiene una orden de pago en determinada fecha que es cancelado por una entidad financiera, este documento es emitido por una persona natural o jurídica denominada girador a otra denominada beneficiario.

Las partes más importantes de un cheque son:

- a) La entidad financiera que emite el cheque.
- b) La persona o entidad a la cual está dirigido el cheque.
- c) La fecha de emisión.
- d) Monto.
- e) Firma de la persona o entidad que emite el cheque
- f) Caracteres de tinta magnética.

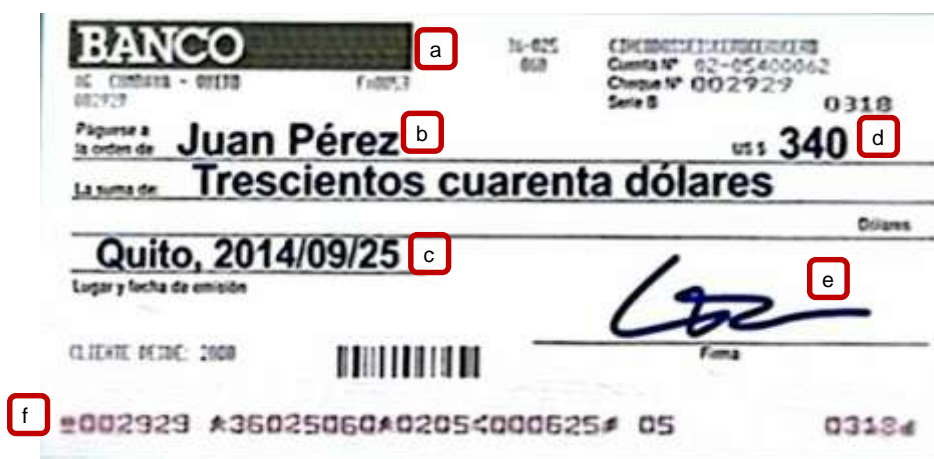


Figura 1 Partes del cheque

2.2 Caracteres de Tinta Magnética

El código de reconocimiento de caracteres de tinta magnética (MICR, por sus siglas en inglés) es una tecnología usada principalmente en las entidades financieras para facilitar el procesamiento de cheques y otros documentos. La codificación MICR se encuentra al pie del documento financiero y típicamente incluye los siguientes indicadores:

- a) Número de cheque
- b) Código de la entidad financiera
- c) Número de cuenta
- d) Moneda
- e) Indicador de control.

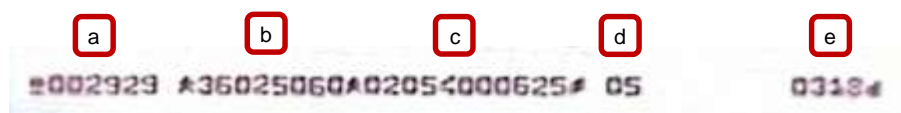


Figura 2 Caracteres de tinta magnética

La tecnología permite a los lectores MICR escanear y exportar la información directamente a una colección de datos. En el Ecuador la normativa dice que la banda de caracteres magnéticos debe estar ubicada en el área extrema inferior del documento y debe usar como fuente para la impresión la norma CMC7. (Superintendencia de Bancos y Seguros, 2006)

2.3 Sistema de Cámara de Compensación de Cheques

Sistema de Cámara de Compensación de Cheques (SCCC) es el conjunto de instrumentos, procedimientos y normas utilizados para la compensación, liquidación y el proceso de devolución de los cheques que las instituciones financieras presentan en la cámara de compensación, a través del intercambio de imágenes digitales e información de los cheques. (Banco Central del Ecuador, 2013)

El SCCC tiene como finalidad procesar los cheques digitalizados para que estos sean enviados al Banco Central del Ecuador y se hagan efectivo en menos de 24 horas.

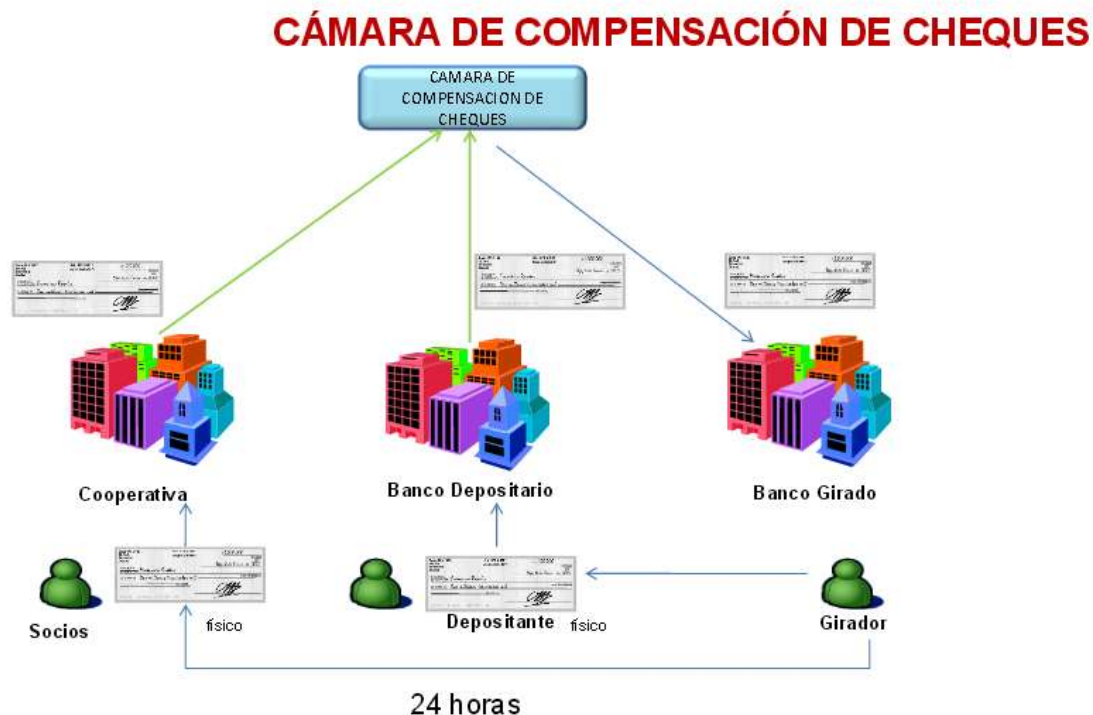


Figura 3 Sistema de Cámara de Compensación de Cheques

Fuente: (Banco Central del Ecuador, 2014)

Esto reemplaza al sistema anterior (ver figura 4), que para hacer efectivos los valores, las instituciones financieras debían realizar un proceso de compensación interbancaria que requería de reuniones presenciales para el intercambio físico de los cheques, mecanismo que tardaba 48 horas en el caso de los bancos y 5 días en el caso de las cooperativas afectando directamente al beneficiario. (Banco Central del Ecuador, 2014)



Figura 5. Digitalización de cheques

- **Escáner:** Es un periférico informático que es utilizado para transformar un documento físico en un archivo digital.
- **Escáner con alimentador de hojas:** En este tipo de escáner el sensor y la fuente de luz permanecen fijos mientras que lo que se mueve es el documento, ayudado por un transporte de rodillos, cinta, tambor o de vacío. Están diseñados para documentos que sean de un tamaño uniforme y con una solidez suficiente para soportar una manipulación brusca. (Publicaciones Vértice, 2008)

2.5 Computación Visual

La computación visual es un campo que incluye métodos para adquirir, procesar, analizar y entender imágenes y datos complejos del mundo real con el fin de producir información simbólica o numérica. (Visión artificial, 2016)

2.5.1 Librería OpenCV

OpenCV es una librería de software open source de computación visual y aprendizaje de máquina; fue construida para proveer una infraestructura común para aplicaciones de computación visual y para acelerar el uso de la percepción de las máquinas en productos comerciales.

La librería cuenta con más de 2500 algoritmos optimizados, estos pueden ser usados para detección y reconocimiento de caras, identificar

objetos, clasificar acciones humanas en videos, seguimiento de movimientos de cámara, seguimiento de movimiento de objetos, extracción de modelos 3D de objetos, enlazar imágenes en conjunto para producir imágenes de alta resolución de escenas enteras, seguir el movimiento de los ojos, reconocer escenarios y establecer marcadores para sobreponer realidad aumentada, reconocimiento de caracteres ópticos, etc. (OpenCV Developers Team, 2016)

2.5.2 Librería Sklearn

Sklearn es una librería de software open source de aprendizaje de máquina para Python, cuenta con herramientas simples y eficientes para minado y análisis de datos, identificación y categorización de objetos, predicción de valores asociados a objetos, agrupación de objetos similares, comparación, validación y escogimiento de parámetros y modelos, pre procesamiento de datos para extracción y normalización, etc. (scikit-learn developers, 2014)

2.5.3 Algoritmos de Reconocimiento de Caracteres

El reconocimiento de caracteres es la conversión mecánica o electrónica de imágenes de manuscritos o texto impreso en texto codificado para máquina.

2.5.3.1 Object Character Recognition (OCR)

El reconocimiento óptico de caracteres es necesario cuando la información debe poder ser leída tanto para los humanos como para las máquinas y no se cuenta con entradas alternativas. El reconocimiento óptico es realizado off-line (ver figura 6) después de que la escritura o impresión se ha completado y el rendimiento depende directamente de la calidad de los documentos de entrada. (Eikvil, 1993)

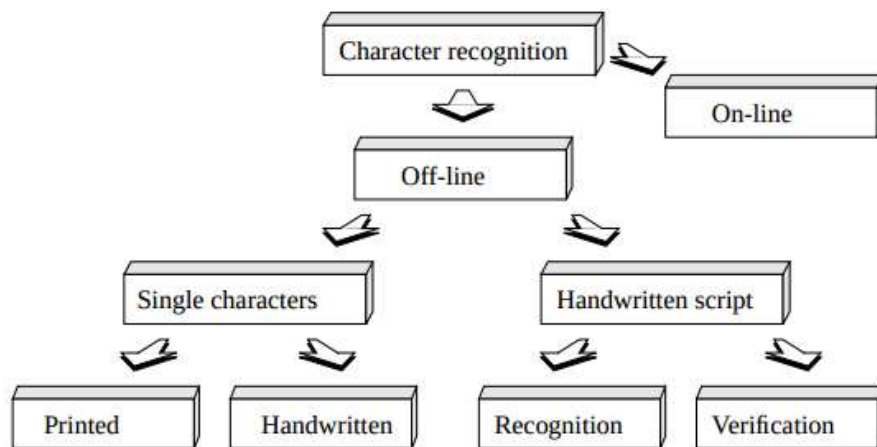


Figura 6 Las diferentes áreas del reconocimiento de caracteres

Fuente: (Eikvil, 1993)

Mientras más fiable es el documento, mejor será el reconocimiento de caracteres, sin embargo, cuando el texto es totalmente manuscrito, OCR está todavía muy lejos de igualar la fiabilidad humana para la lectura de letras o números. Sin embargo, las técnicas para lectura están siendo mejoradas continuamente haciendo uso de algoritmos de reconocimiento de patrones y caracteres como por ejemplo el método k-Nearest Neighbors (k-NN) o Support Vector Machine (SVM).

2.5.3.2 Algoritmo K-Nearest Neighbors (k-NN)

k-Nearest Neighbors (k-NN) es uno de los más simples algoritmos de clasificación disponibles para el aprendizaje supervisado. La idea es buscar la más cercana coincidencia entre unos datos de prueba en un espacio característico (Ver figura 7).

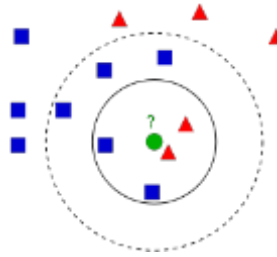


Figura 7. Algoritmo k-NN

Fuente: (Mordvintsev & K, 2014)

En la figura 7, tenemos dos familias, Cuadrados azules y Triángulos rojos. Se llama a cada familia una Clase las cuales están localizadas en un mapa que se lo llama un espacio característico. Se agrega un nuevo miembro al mapa, el cual es un Círculo Verde, este debe ser añadido a una de las familias Azul o Roja. Se llama a este proceso, Clasificación.

Un método para realizar esto es verificar cuál es su vecino más cercano. Se puede observar claramente que pertenece a la familia de triángulos rojos (ver figura 7). A este método se lo llama simplemente Vecino Cercano (Nearest Neighbor), porque la clasificación depende solo del vecino más cercano.

Pero hay un problema con esto, el triángulo rojo puede ser el más cercano, pero qué pasa si hay más cuadrados azules cerca de este. Entonces los cuadrados azules tendrán más peso que los triángulos rojos. Entonces solo verificar el más cercano no es suficiente, por lo tanto, se va a verificar las k familias más cercanas. En la figura 7, tomamos $k=3$ (primer círculo concéntrico), los 3 vecinos más cercanos. Entonces se tiene dos Rojos y solo un Azul, entonces tendrá que ser añadido a la familia Roja. Si tomamos $k=7$ (segundo círculo concéntrico, delimitado por una línea entrecortada), entonces se tiene 5 vecinos azules y 2 vecinos rojos, por lo tanto, éste será añadido a la familia azul. A este método se lo denomina k -Nearest Neighbors porque la clasificación depende de los k vecinos más cercanos.

Sin embargo, además del número de vecinos cercanos, se debe dar una ponderación dependiendo de la cercanía de estos a nuestro punto de

referencia. Entonces se añade un peso a cada vecino dependiendo de su distancia, para aquellos que están más cerca tienen mayor peso y aquellos que están más lejos tienen un menor peso. Luego se suma el total de pesos de cada familia separadamente y el que obtenga el peso mayor total, es a donde ira el punto de referencia. A este método se lo llama k-NN modificado. (Mordvintsev & K, 2014)

Prueba del Algoritmo

El algoritmo k-NN de OpenCV consiste en entrenarse y probarse con una imagen de prueba que contiene una muestra de 5000 dígitos manuscritos (500 por dígito) para después determinar el porcentaje de precisión del reconocimiento.

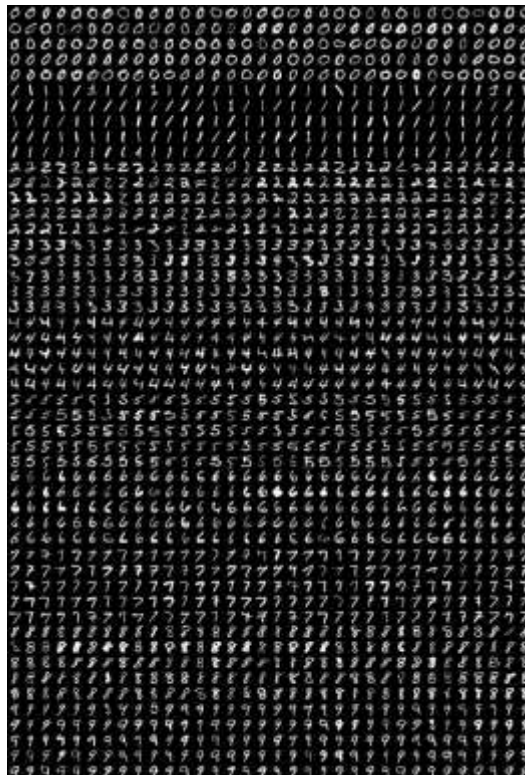


Figura 8 Imagen de Prueba de Dígitos Manuscritos

Cada dígito es una imagen de 20 x 20 píxeles, por lo tanto, el primer paso consiste en separar la imagen en 500 diferentes dígitos y cada imagen es convertida en un array de 400 píxeles. Después se usan las primeras 250 muestras para el entrenamiento y las siguientes 250 para pruebas.

```

1  import numpy as np
2  import cv2
3  from matplotlib import pyplot as plt
4
5  img = cv2.imread('digits.png')
6  gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
7
8  # Now we split the image to 5000 cells, each 20x20 size
9  cells = [np.hsplit(row,100) for row in np.vsplit(gray,50)]
10
11 # Make it into a Numpy array. It size will be (50,100,20,20)
12 x = np.array(cells)
13
14 # Now we prepare train_data and test_data.
15 train = x[:, :50].reshape(-1,400).astype(np.float32) # Size = (2500,400)
16 test = x[:, 50:100].reshape(-1,400).astype(np.float32) # Size = (2500,400)
17
18 # Create labels for train and test data
19 k = np.arange(10)
20 train_labels = np.repeat(k,250)[:,np.newaxis]
21 test_labels = train_labels.copy()
22
23 # Initiate knn, train the data, then test it with test data for k=1
24 knn = cv2.KNearest()
25 knn.train(train,train_labels)
26 ret,result,neighbours,dist = knn.find_nearest(test,k=5)
27
28 # Now we check the accuracy of classification
29 # For that, compare the result with test_labels and check which are wrong
30 matches = result==test_labels
31 correct = np.count_nonzero(matches)
32 accuracy = correct*100.0/result.size
33 print accuracy
34

```

Figura 9 Script Algoritmo k-NN

2.5.3.3 Algoritmo Support Vector Machines (SVM)

El algoritmo trabaja en tres etapas, primero procesa imágenes reales de dígitos manuscritos de la base de datos MNIST que cuenta con aproximadamente 60000 muestras de entrenamiento y 10000 de pruebas, después se aplica la extracción HOG que consiste en calcular la dirección del histograma de gradiente de la imagen para mejorar la predicción; finalmente se aplica el algoritmo linear SVM para clasificar los dígitos. (Ver figura 10)

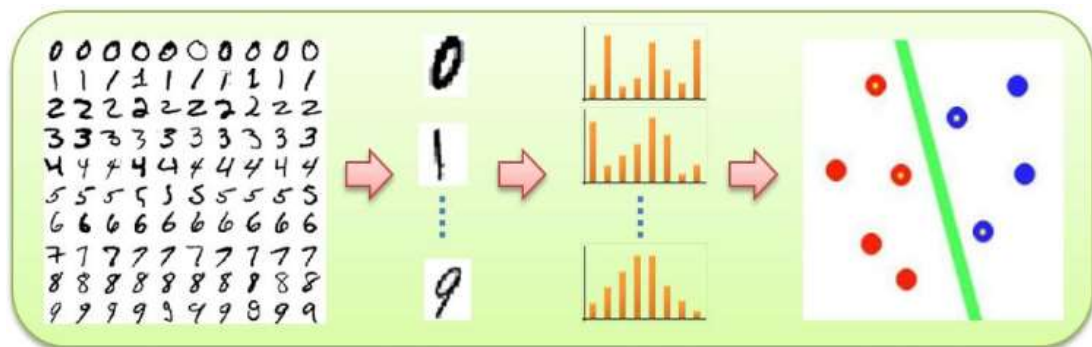


Figura 10 Estructura del modelo SVM lineal con características de HOG

Fuente: (Ebrahimzadeh & Jampour, 2014)

La aplicación del proceso hace que se tenga una exactitud en el reconocimiento del 97.5 % en comparación de otros algoritmos. (Ebrahimzadeh & Jampour, 2014)

Histogram of Oriented Gradients (HOG)

Es un algoritmo descriptor utilizado en computación visual para detectar y procesar objetos en una imagen, consiste en dividir a una imagen en pequeñas regiones (celdas), luego calcula el histograma de la dirección del gradiente o la orientación de los bordes en cada una de las celdas (ver figura 11), para posteriormente tener una estimación de cada región (ver figura 12). (Ebrahimzadeh & Jampour, 2014)

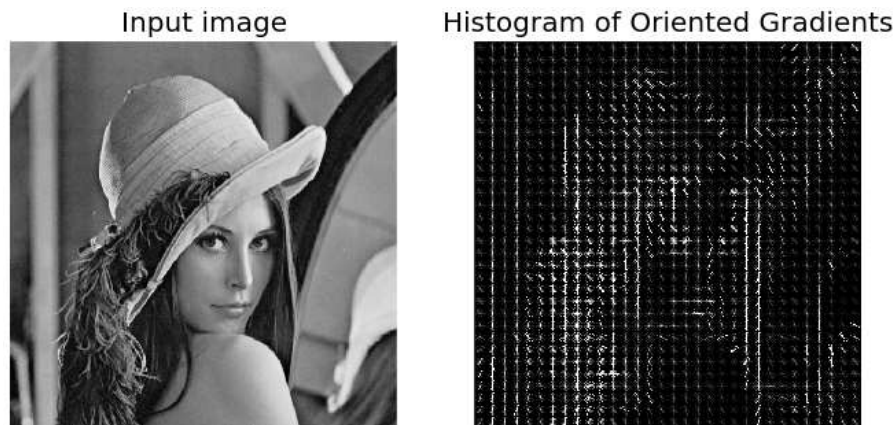


Figura 11 Identificación de gradientes en una imagen

Fuente: (Scikit-Image, 2012)

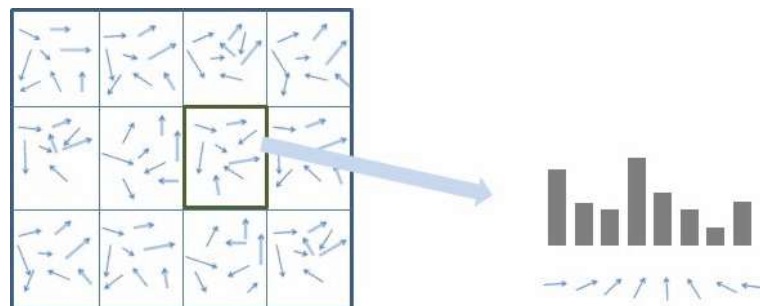


Figura 12 Histograma de gradientes

Fuente: (National Instruments Corporation, 2015)

Los resultados de los histogramas son usados para la clasificación de una imagen, especialmente para la detección de humanos.

Support Vector Machines (SVM)

Es un conjunto de algoritmos de aprendizaje supervisados empleados para la clasificación y la regresión. Dado un conjunto de ejemplos de entrenamiento (muestras) podemos etiquetar las clases y entrenar una SVM para construir un modelo que prediga la clase de una nueva muestra. (Morán, 2013)

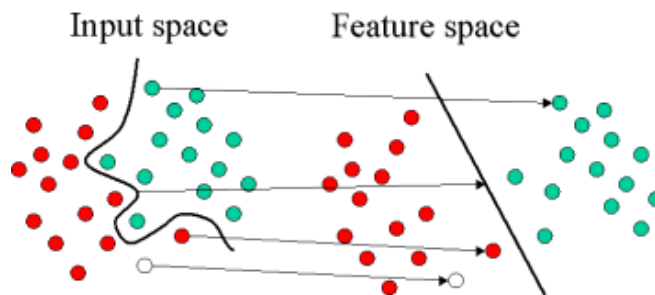


Figura 13 Support Vector Machines

Fuente: (StatSoft Inc., 2016)

En la figura 13 se visualiza el concepto básico de SVM, en la parte izquierda se ve los objetos originales, los mismos que pasan a ser reorganizados y mapeados. En este nuevo escenario, los objetos mapeados (ver figura 13 – lado derecho) son linealmente separados y, por lo tanto, en lugar de construir una curva compleja (ver figura 13 - lado izquierdo), todo lo que hace es encontrar una línea óptima que puede separar los objetos de color verdes de los rojos. (StatSoft Inc., 2016)

Base de datos MNIST

Mixed National Institute of Standards and Technology (MNIST) es una base de datos de dígitos manuscritos que comúnmente se usa para entrenar sistemas de procesamiento de imágenes, el tamaño de la base de datos es de 55.6 MB que contiene imágenes de 28 x 28 píxeles con dígitos del 0 al 9.

La base de datos contiene aproximadamente 70000 dígitos divididos en muestras para entrenamiento y pruebas.

Tabla 1**Información de dígitos manuscritos MNIST**

Digito	Número de muestras para entrenamiento	Número de muestras para pruebas	Total
0	5923	980	6903
1	6742	1135	7877
2	5958	1032	6990
3	6131	1010	7141
4	5842	982	6824
5	5842	892	6313
6	5918	958	6876
7	6265	1028	7293
8	5851	974	6825
9	5949	1009	6958
Total	60000	10000	70000

**Figura 14 Muestra Dígitos Manuscritos Base de Datos MNIST**

Fuente: (LeCun, Cortes, & Burges, 2013)

Prueba del Algoritmo

La prueba del algoritmo SVM Linear con características HOG consta de tres etapas:

- a) Crear una base de datos de dígitos manuscritos
- b) Para cada dígito manuscrito en la base de datos, extraer las características HOG y entrenar el algoritmo Linear SVM.
- c) Usar el clasificador entrenado para predecir los dígitos.

Entrenamiento del clasificador

Importar los módulos necesarios.

```
1  # Import the modules
2  from sklearn.externals import joblib
3  from sklearn import datasets
4  from skimage.feature import hog
5  from sklearn.svm import LinearSVC
6  import numpy as np
7  from collections import Counter
```

Figura 15 Fragmento Script Algoritmo SVM - Modulos

Cargar la base de datos MNIST.

```
9  # Load the dataset
10 dataset = datasets.fetch_mldata("MNIST Original")
```

Figura 16 Fragmento Script Algoritmo SVM - MNIST

Guardar las imágenes de los dígitos y sus etiquetas en arrays numéricos para Python.

```
12 # Extract the features and labels
13 features = np.array(dataset.data, 'int16')
14 labels = np.array(dataset.target, 'int')
```

Figura 17 Fragmento Script Algoritmo SVM - Arrays

Calcular de las características HOG para cada imagen en la base de datos y guardarlas en otro array numérico.

```

16 # Extract the hog features
17 list_hog_fd = []
18 for feature in features:
19     fd = hog(feature.reshape((28, 28)), orientations=9, pixels_per_cell=(14, 14), cells_per_block=(1, 1), visualise=False)
20     list_hog_fd.append(fd)
21 hog_features = np.array(list_hog_fd, 'float64')

```

Figura 18 Fragmento Script Algoritmo SVM – Características HOG

Crear un objeto linear SVM, realizar el entrenamiento y guardar el clasificador en un archivo para su posterior uso.

```

25 # Create an linear SVM object
26 clf = LinearSVC()
27
28 # Perform the training
29 clf.fit(hog_features, labels)
30
31 # Save the classifier
32 joblib.dump(clf, "digits_cls.pkl", compress=3)

```

Figura 19 Fragmento Script Algoritmo SVM – Clasificador

Prueba del Clasificador

Importar los módulos necesarios.

```

1 # Import the modules
2 import cv2
3 from sklearn.externals import joblib
4 from skimage.feature import hog
5 import numpy as np

```

Figura 20 Fragmento Script Algoritmo SVM - Módulos

Cargar el clasificador generado previamente.

```

7 # Load the classifier
8 clf = joblib.load("digits_cls.pkl")

```

Figura 21 Fragmento Script Algoritmo SVM – Cargar clasificador

Cargar una imagen de entrada para la prueba y tratarla digitalmente.

```
10 # Read the input image
11 im = cv2.imread("test_cheque009.jpg")
12
13 # Convert to grayscale and apply Gaussian filtering
14 im_gray = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
15 im_gray = cv2.GaussianBlur(im_gray, (5, 5), 0)
```

Figura 22 Fragmento Script Algoritmo SVM – Cargar imagen

Convertir la imagen en escala de grises a una imagen binaria con un valor de umbral de 90.

```
17 # Threshold the image
18 ret, im_th = cv2.threshold(im_gray, 90, 255, cv2.THRESH_BINARY_INV)
```

Figura 23 Fragmento Script Algoritmo SVM – Convertir imagen

Calcular los contornos en la imagen y aplicar un cuadro delimitador por cada contorno

```
20 # Find contours in the image
21 ctrs, hier = cv2.findContours(im_th.copy(), cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
22
23 # Get rectangles contains each contour
24 rects = [cv2.boundingRect(ctr) for ctr in ctrs]
```

Figura 24 Fragmento Script Algoritmo SVM – Calcular contornos

Para cada cuadro delimitador se calcula las características HOG y se predice el dígito usando Linear SVM

```

26 # For each rectangular region, calculate HOG features and predict
27 # the digit using Linear SVM.
28 for rect in rects:
29     # Make the rectangular region around the digit
30     leng = int(rect[3] * 1.6)
31     pt1 = int(rect[1] + rect[3] // 2 - leng // 2)
32     pt2 = int(rect[0] + rect[2] // 2 - leng // 2)
33     roi = im_th[pt1:pt1+leng, pt2:pt2+leng]
34     # Resize the image
35     roi = cv2.resize(roi, (28, 28), interpolation=cv2.INTER_AREA)
36     roi = cv2.dilate(roi, (3, 3))
37     # Calculate the HOG features
38     roi_hog_fd = hog(roi, orientations=9, pixels_per_cell=(14, 14), cells_per_block=(1, 1), visualise=False)
39     nbr = clf.predict(np.array([roi_hog_fd], 'float64'))
40     cv2.putText(im, str(int(nbr[0])), (rect[0], rect[1]), cv2.FONT_HERSHEY_DUPLEX, 2, (51, 102, 0), 3)

```

Figura 25 Fragmento Script Algoritmo SVM - Predicción

Mostrar el resultado gráficamente

```

42 cv2.imshow("Resulting Image with Rectangular ROIs", im)
43 cv2.waitKey()

```

Figura 26 Fragmento Script Algoritmo SVM - Resultados

Pruebas

Pruebas del algoritmo usando imágenes de entrada de dígitos manuscritos escritos normalmente.

Prueba 01

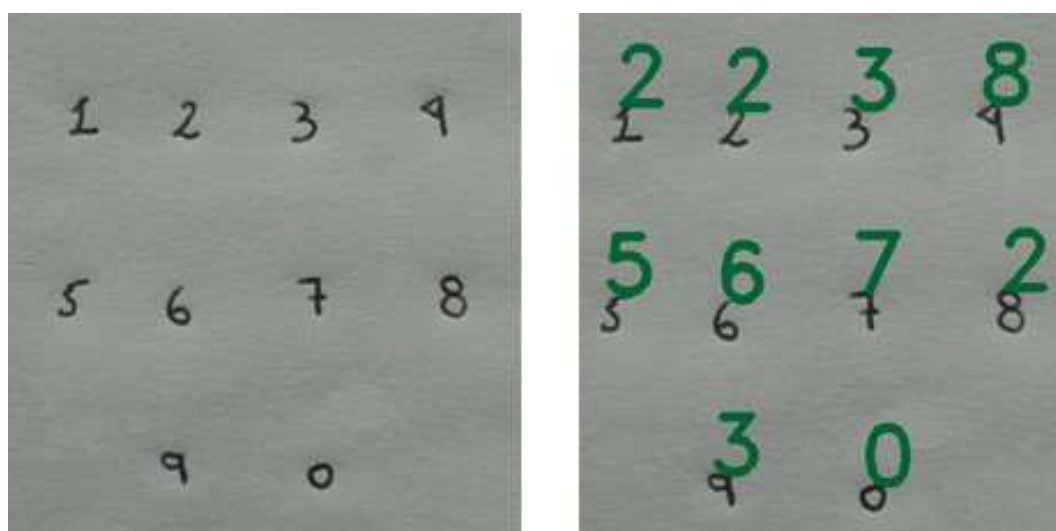


Figura 27 Prueba 01 Algoritmo SVM

Tabla 2

Resultado Prueba 01 Algoritmo SVM

Dígito Manuscrito	Dígito Encontrado
1	2
2	2
3	3
4	8
5	5
6	6
7	7
8	2
9	3
0	0

Prueba 02

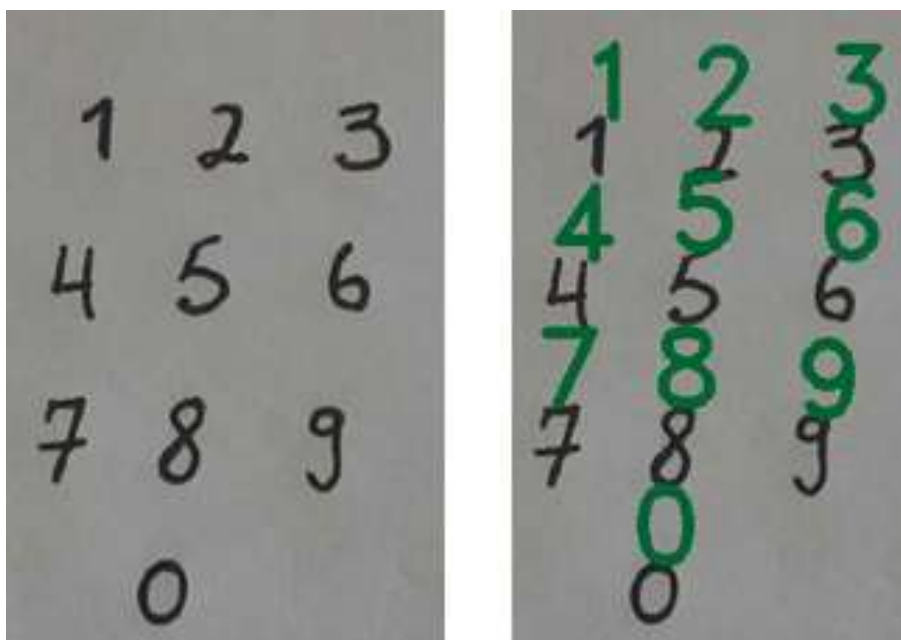


Figura 28 Prueba 02 Algoritmo SVM

Tabla 3

Resultado Prueba 02 Algoritmo SVM

Dígito Manuscrito	Dígito Encontrado
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
0	0

Prueba 03

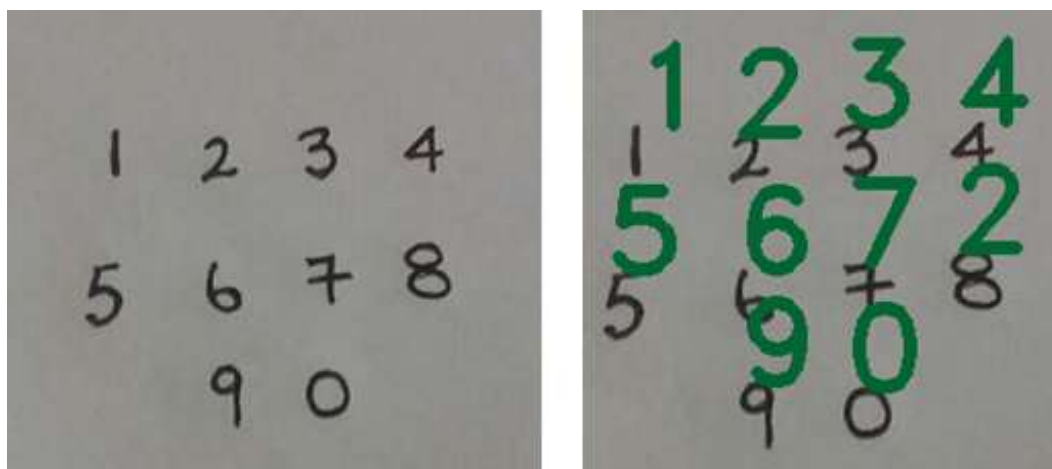


Figura 29 Prueba 03 Algoritmo SVM

Tabla 4

Resultado Prueba 03 Algoritmo SVM

Dígito Manuscrito	Dígito Encontrado
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	2
9	9
0	0

Prueba 04

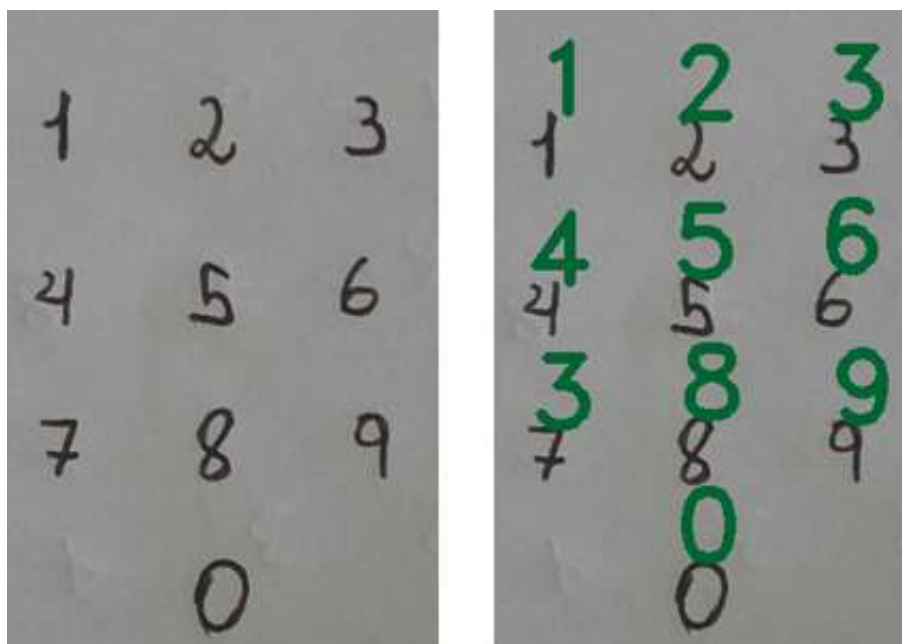


Figura 30 Prueba 04 Algoritmo SVM

Tabla 5

Resultado Prueba 04 Algoritmo SVM

Dígito Manuscrito	Dígito Encontrado
1	1
2	2
3	3
4	4
5	5
6	6
7	3
8	8
9	9
0	0

Prueba 05

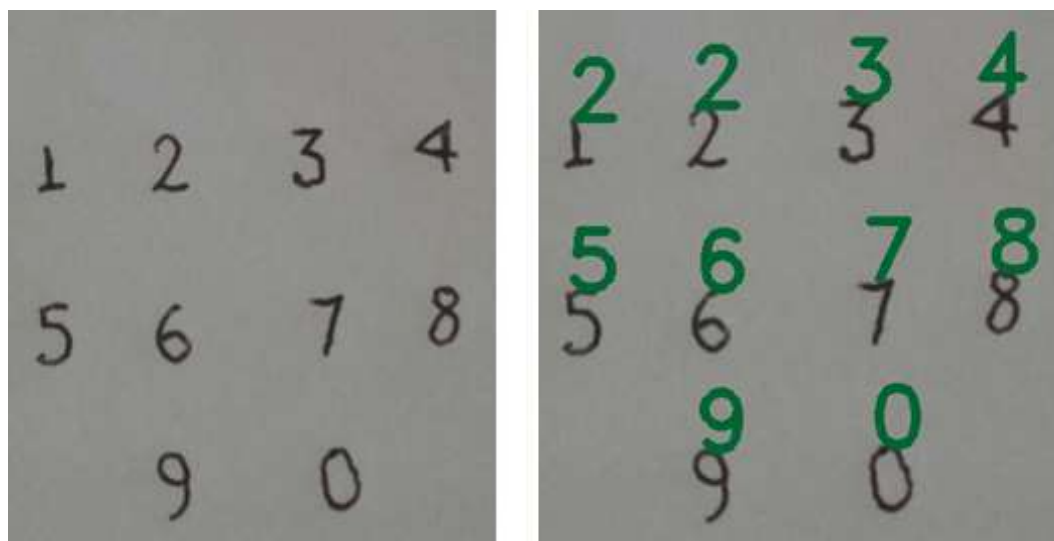


Figura 31 Prueba 05 Algoritmo SVM

Tabla 6

Resultado Prueba 05 Algoritmo SVM

Dígito Manuscrito	Dígito Encontrado
1	2
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
0	0

Prueba 06

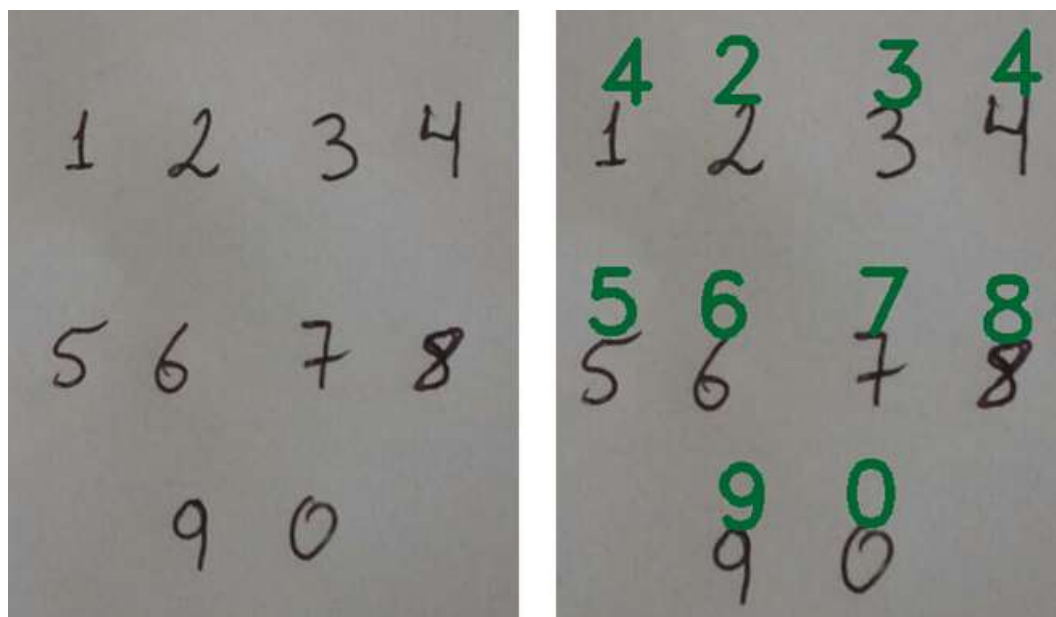


Figura 32 Prueba 06 Algoritmo SVM

Tabla 7

Resultado Prueba 06 Algoritmo SVM

Dígito Manuscrito	Dígito Encontrado
1	4
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
0	0

Prueba 07

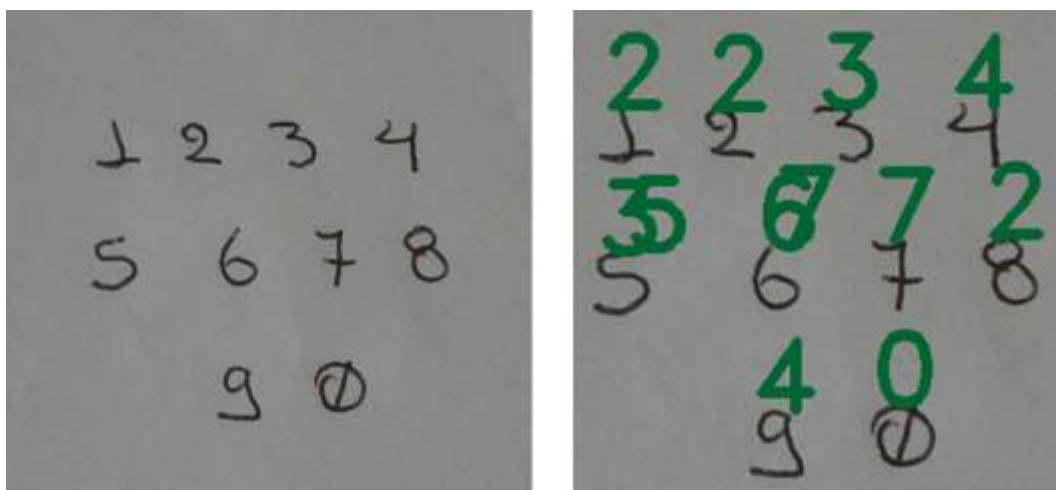


Figura 33 Prueba 07 Algoritmo SVM

Tabla 8

Resultado Prueba 07 Algoritmo SVM

Dígito Manuscrito	Dígito Encontrado
1	2
2	2
3	3
4	4
5	3-5
6	6-7
7	7
8	2
9	4
0	0

Prueba 08

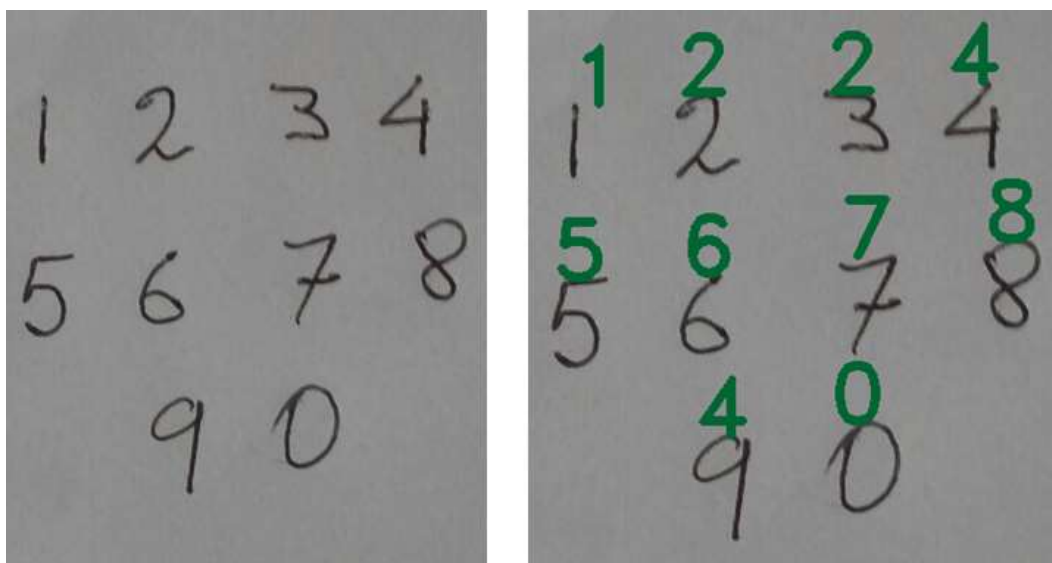


Figura 34 Prueba 08 Algoritmo SVM

Tabla 9

Resultado Prueba 08 Algoritmo SVM

Dígito Manuscrito	Dígito Encontrado
1	1
2	2
3	2
4	4
5	5
6	6
7	7
8	8
9	4
0	0

Prueba 09

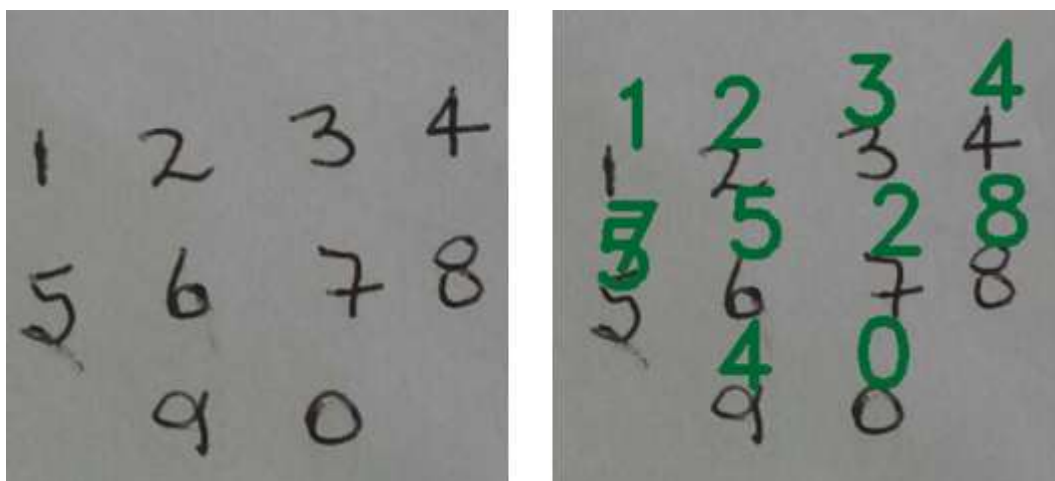


Figura 35 Prueba 09 Algoritmo SVM

Tabla 10

Resultado Prueba 09 Algoritmo SVM

Dígito Manuscrito	Dígito Encontrado
1	1
2	2
3	3
4	4
5	5-7
6	5
7	2
8	8
9	4
0	0

Prueba 10

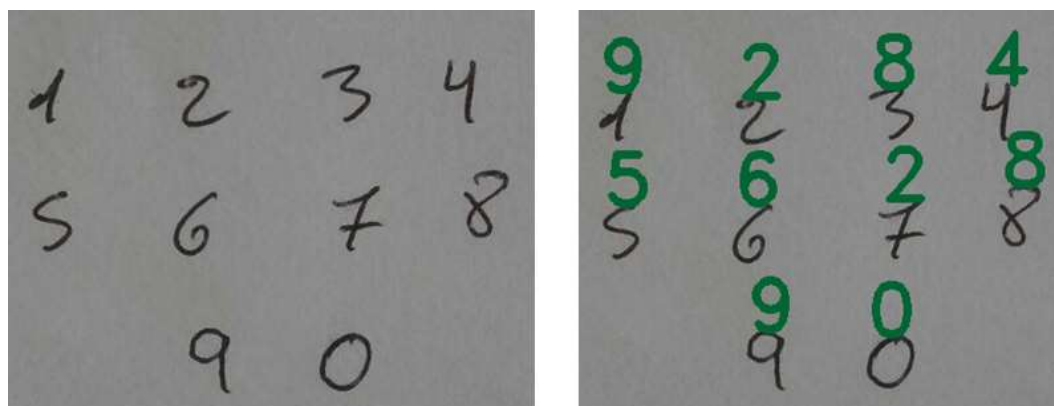


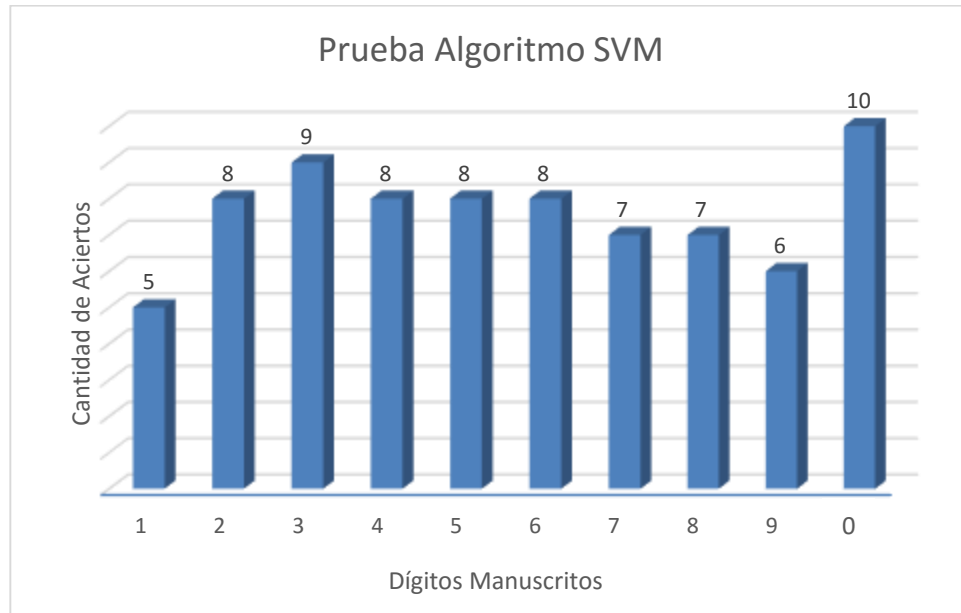
Figura 36 Prueba 10 Algoritmo SVM

Tabla 11

Resultado Prueba 10 Algoritmo SVM

Dígito Manuscrito	Dígito Encontrado
1	9
2	2
3	8
4	4
5	5
6	6
7	2
8	8
9	9
0	0

Resultados

**Figura 37 Resultados Prueba Algoritmo SVM**

Se puede observar altos índices de éxito, sobre todo en los dígitos 2 y 0 que obtienen 10 aciertos mientras que el 1 y 9 son los más bajos con 5 y 6 aciertos respectivamente, debido a que los números se encuentran bastante separados entre sí, además que el fondo es neutro por lo cual no interfiere con el reconocimiento.

Pruebas del algoritmo usando como imágenes de entrada el area del monto de un cheque

Prueba 01



Figura 38 Prueba 01 en Cheque Algoritmo SVM

Prueba 02



Figura 39 Prueba 02 en Cheque Algoritmo SVM

Prueba 03



Figura 40 Prueba 03 en Cheque Algoritmo SVM

Prueba 04



Figura 41 Prueba 04 en Cheque Algoritmo SVM

Prueba 05

BANCO DE LOJA
CHEQUE NACIONAL

29-001
300

CUENTA No. 2900-09487-7
SIETESIETEC CUATRONUEVECERO
CHEQUE No. U02458

PAGARSE A LA ORDEN DE Tamayo Alicia US.\$ 250.00

LA SUMA DE Doscientos cincuenta col/100

Ciudad Quito Fecha 25 de agosto del 2009

SISTEMAS DE INFORMACION

2900-09487-7 C17C17

0002458 *290001300* 2900094877 05 0586

250.00

250.00

Figura 42 Prueba 05 en Cheque Algoritmo SVM

Prueba 06

BANCO PICHINCHA C.A.
CHEQUE PAGADERO EN CUALQUIERA DE NUESTRAS OFICINAS

10-010
060

CUATROCERO CUATROCERO SIETECHO
CUENTA No. 30048784-04
CHEQUE No. 001051
4357

PAGARSE A LA ORDEN DE ANTONIA Juonita US.\$ 10.95

LA SUMA DE Diez noventa y cinco centavos

Ciudad y Fecha Risobamba 2015/04/21

MARIO HIDALGO CEDENO 02/05/1989
AG. RIO AMAZONAS QUITO NS. 0207.140814

Alcides Rios

001051 *10010060* 3004878404 05 4357

10.95

10.95

Figura 43 Prueba 06 en Cheque Algoritmo SVM

Prueba 07

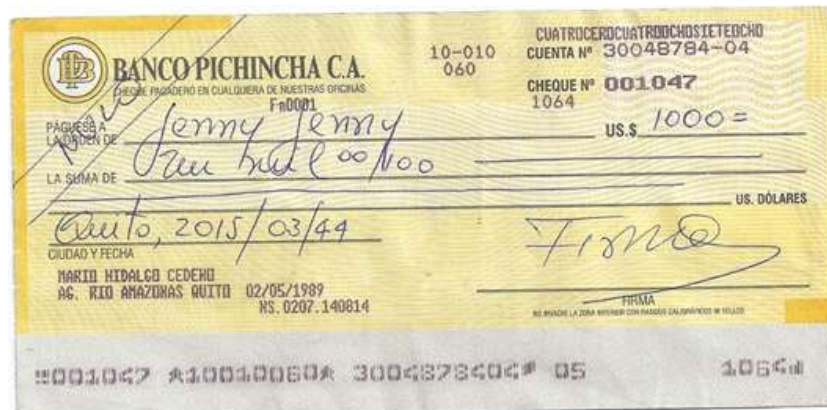


Figura 44 Prueba 07 en Cheque Algoritmo SVM

Prueba 08



Figura 45 Prueba 08 en Cheque Algoritmo SVM

Prueba 09



Figura 46 Prueba 09 en Cheque Algoritmo SVM

Prueba 10



Figura 47 Prueba 10 en Cheque Algoritmo SVM

Resultados

A pesar de los buenos resultados en las pruebas anteriores, se puede determinar que el algoritmo falla en entornos reales como el reconocimiento del monto de un cheque escaneado, esto se debe tanto a la calidad de la caligrafía como a la disposición de los dígitos; además el fondo y marcas de agua de los cheques no permite que el algoritmo diferencie correctamente los números confundiendo los dígitos.

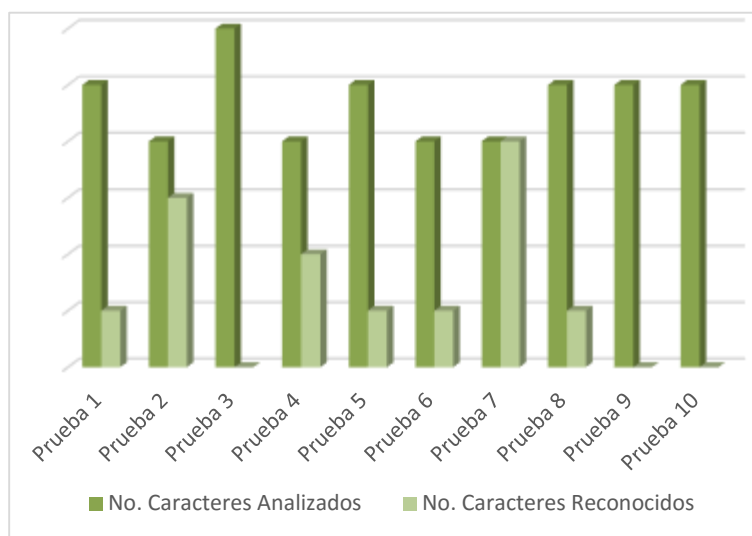


Figura 48 Resultado de Pruebas en Cheques

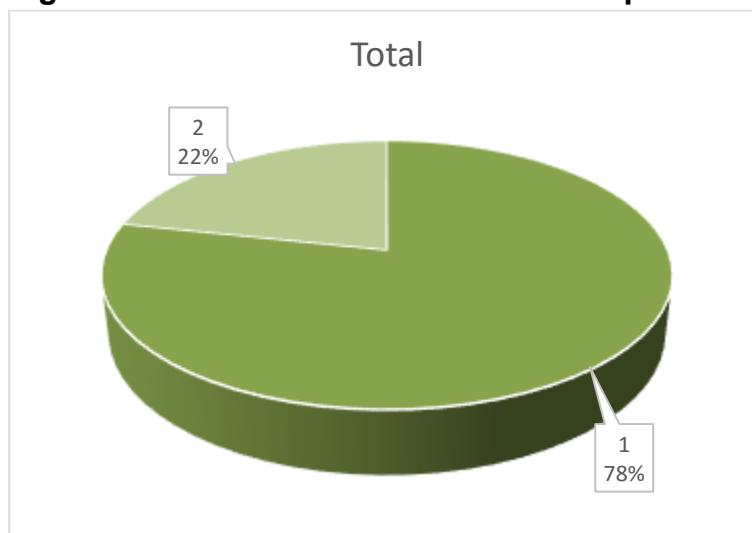


Figura 49 Resultado de Pruebas en Cheques: Porcentajes

2.6 Representational State Transfer (REST)

Representational State Transfer (REST) es un estilo de arquitectura de software para diseñar servicios web y aplicaciones distribuidas, se usa básicamente para conectar maquinas entre si haciendo uso del protocolo de comunicación HTTP.

Las aplicaciones RESTful usan peticiones HTTP para enviar datos (crear y/o actualizar), leer y eliminar datos. Por lo tanto, REST usa HTTP para las cuatro operaciones CRUD (Create/Read/Update/Delete). REST es una alternativa liviana a mecanismos como RCP (Remote Procedure Calls) y servicios web como SOA, WSDL, etc. (Elkstein, 2008)

2.7 Autenticación Basada en Tokens

Es una técnica de autenticación que permite a los usuarios ingresar sus credenciales y obtener a cambio un token que permite acceder a un recurso específico de un sitio web en un tiempo determinado por un servidor.

2.7.1 JSON Web Token (JWT)

2.7.1.1 Definición

JSON Web Token es un estándar abierto que define una compacta y auto contenida vía segura para transmisión de información entre partes como objetos JSON⁴. Esta información puede ser verificada y confiable debido a que está firmada digitalmente. Los JWTs pueden ser firmados usando una clave secreta (con el algoritmo HMAC⁵) o una clave pública/privada usando RSA⁶. (Jones, Bradley, & Sakimura, 2015)

⁴ JavaScript Object Notation (JSON) es un formato liviano para intercambio de datos que usa texto legible para los humanos. (ECMA International, 1999)

⁵ En criptografía, es un tipo de código de autenticación que involucra una función hash en combinación con una llave criptográfica secreta. (Krawczyk, Bellare, & Canetti, 1997)

⁶ Es un algoritmo criptográfico asimétrico usado por computadores para encriptar y des encriptar mensajes. (RSA Laboratories, 2012)

2.7.1.2 Estructura

JSON Web Token consiste en tres partes separadas por un punto (.), las cuales son:

- Header (cabecera)
- Payload (cuerpo)
- Signature (firma)

Por lo tanto, un JWT típicamente luce como lo siguiente

```
1  
2  xxxxx.yyyyy.zzzzz  
3
```

Figura 50 Estructura JSON Web Token

Header

La cabecera típicamente consiste de dos partes, el tipo de token, que en este caso es JWT, y el algoritmo hash usado, tales como HMAC, SHA256, por ejemplo:

```
1  {  
2    "alg": "HS256",  
3    "typ": "JWT"  
4  }
```

Figura 51 Header JSON Web Token

Payload

La segunda parte del token es el cuerpo, que contiene los datos y estos son estados acerca de una entidad (típicamente, el usuario) y metadatos adicionales. Un ejemplo puede ser:

```
1 {  
2   "sub": "1234567890",  
3   "name": "Santiago",  
4   "admin": true  
5 }
```

Figura 52 Payload JSON Web Token

Signature

Para crear la firma se debe tomar la cabecera y el cuerpo codificados, una frase secreta y el algoritmo especificado en la cabecera.

Por ejemplo, si se quiere usar el algoritmo HMAC SHA256, la firma debe ser creada de la siguiente manera:

```
1 HMACSHA256(  
2   base64UrlEncode(header) + "." +  
3   base64UrlEncode(payload),  
4   secret)
```

Figura 53 Signature JSON Web Token

La firma es usada para verificar que el remitente es quien dice ser y para asegurarse que el mensaje no ha sido cambiado en el camino.

Salida del Token

La salida consta de tres cadenas de texto en Base64 separadas por puntos que pueden ser fácilmente enviadas en entornos HTML y HTTP lo cual es más compacto comparado con estándares XML tales como SAML.

Lo siguiente muestra un JWT que tiene una cabecera y cuerpo codificados y firmados con un secreto:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaXNTb2NpYWwiOnRydWV9.4pcPyMD09olPSyXnrXCjTwXyr4Bsezdl1AVTmud2fU4
```

Figura 54 JSON Web Token Codificado

2.7.1.3 Como Trabaja JSON Web Token

En autenticación, cuando el usuario ingresa correctamente usando sus credenciales, un JSON Web Token será retornado y debe ser guardado localmente (típicamente en local storage, aunque también puede ocuparse cookies), en vez de la manera tradicional de crear la sesión en el servidor y retornarlas en una cookie.

Cualquiera que sea la forma en que el usuario quiera acceder a una ruta o recurso protegido, el user agent debe enviar el JWT, típicamente en la cabecera de la autorización usando el esquema Bearer. El contenido de la cabecera debe lucir como lo siguiente:

```
1
2 Authorization: Bearer <token>
3
```

Figura 55 Authorization JSON Web Token

Este es un mecanismo de autenticación sin estado ya que el estado nunca es guardado en la memoria del servidor. Las rutas protegidas del servidor se comprobarán con un JWT válido en la cabecera de autorización, y si está presente, el usuario podrá acceder a recursos protegidos.

Como los tokens JWT son auto contenidos, toda la información necesaria está ahí, reduciendo la necesidad de realizar queries a la base de datos por múltiples ocasiones.

El siguiente diagrama muestra el proceso completo:

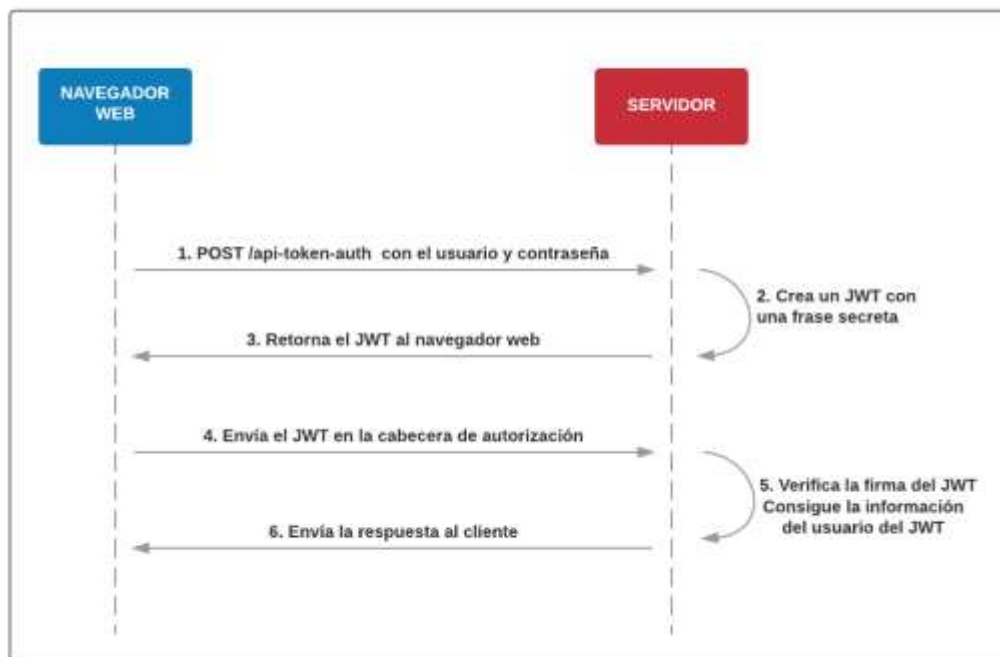


Figura 56 Diagrama de Secuencia JSON Web Token

2.8 Metodología de Desarrollo De Software

El uso de una metodología para el desarrollo de software es importante para mantener un adecuado control del proceso, optimizar los recursos, entre otras cosas, lo que permite obtener un producto final de calidad y acorde a las necesidades del usuario. Es por ello que en el presente proyecto se empleará una metodología ágil denominada Extreme Programming.

2.8.1 Extreme Programming

Es una metodología ágil de desarrollo de software que fue publicada en el año de 1996 y que se popularizó por hacer que los procesos de e licitación sean ágiles y simples, y no como las antiguas que se basan en un modelo de cascada.

El éxito de extreme programming es por su énfasis en la satisfacción del cliente, generando que el cliente se incluya en cada uno de los procesos de desarrollo, permitiendo así que no se tenga inconvenientes al verificar

software al final del proyecto. Extreme programming permite que en el proceso de desarrollo el programador pueda corregir a tiempo problemas en los requerimientos funcionales, además de que el cliente tenga pequeños entregables en cortos intervalos de tiempo para evaluar y aprobar el funcionamiento del software. El ciclo de vida de extreme programming es de forma circular permitiendo que haya una retroalimentación en cada una de las etapas, esto es sumamente útil para entregar un producto de calidad para el cliente. (Wells, 2013)

Extreme programming contempla cinco reglas las cuales son:

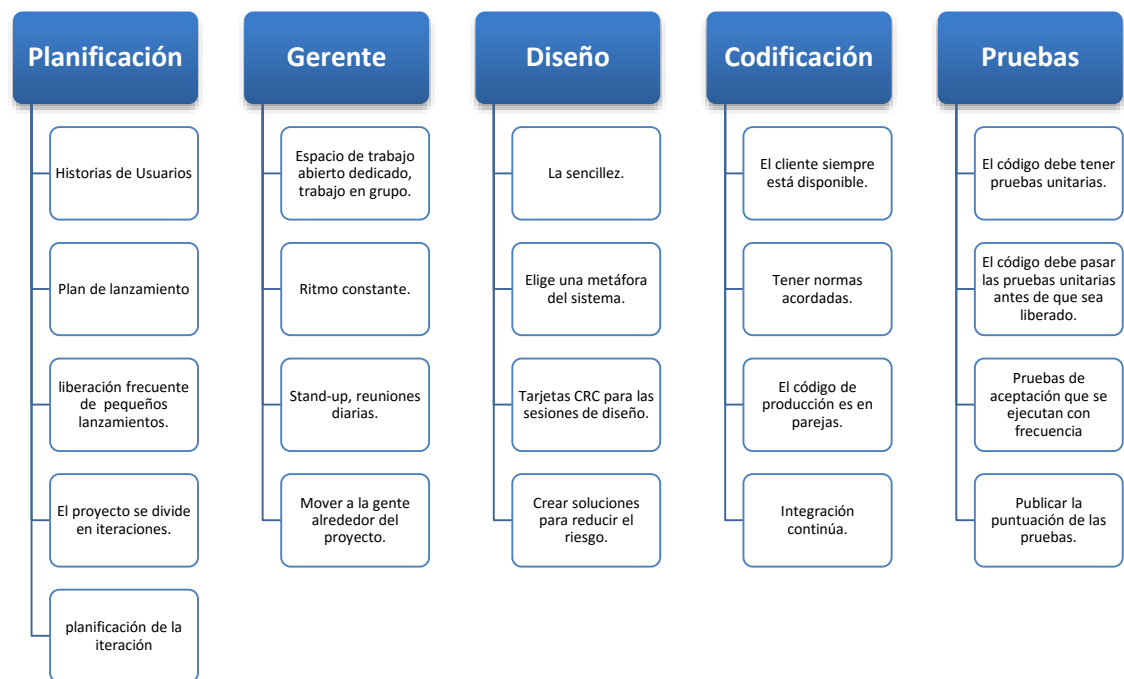


Figura 57 Fases Metodología Extreme Programming

2.9 Frameworks de Desarrollo

2.9.1 Django Framework

Django es un framework web de alto nivel para Python que incentiva el rápido y limpio desarrollo de software haciendo uso del patrón de diseño Modelo Vista Controlador (MVC). (Django Software Foundation, 2016)

2.9.2 Django REST Framework

Django REST Framework es un poderoso y flexible conjunto de herramientas para construir APIs Web.

Sus características proveen:

- Un API Navegable en la Web que provee una gran usabilidad para los desarrolladores.
- Políticas de autenticación entre los cuales están OAuth, JWT, etc.
- Serialización que soporta fuentes de datos ORM y No-ORM.
- Vistas regulares basadas en funciones totalmente personalizables. (Christie, 2016)

2.9.3 AngularJS

AngularJS es un framework Javascript de código abierto que usa el patrón de diseño Modelo Vista Controlador (MVC) y es utilizado para la creación de aplicaciones web de una sola página (Single Page Application). Extiende el lenguaje de marcado HTML añadiendo etiquetas personalizadas que obedecen a directivas y controladores externos. (Google Inc., 2016)

2.9.4 .NET Framework

El framework .NET es una tecnología que soporta la construcción y funcionamiento de aplicaciones y servicios web. Consta de dos componentes principales: Common Language Runtime (CLR) que es el motor de ejecución

que controla las aplicaciones en ejecución, y la biblioteca de clases de .NET Framework, que proporciona una biblioteca de código probado y reutilizable al que pueden llamar los desarrolladores desde sus propias aplicaciones. (Microsoft, 2016)

2.10 Lenguajes De Programación

2.10.1 Python

Python es un lenguaje de programación de código abierto interpretado, funcional, orientado a objetos e interactivo cuya filosofía hace hincapié en una sintaxis que favorezca un código legible. (Python Software Foundation, 2016)

2.10.2 Javascript

Javascript (JS) es un lenguaje de programación interpretado orientado a objetos. Se lo utiliza principalmente del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas. (Mozilla Developer Network, 2016)

2.10.3 HTML5

HTML5 es un lenguaje de marcado usado para la elaboración de la estructura y presentación de contenido en la Web. Define una estructura básica y un código para la definición de contenido de una página web, como texto, imágenes, videos, entre otros. (World Wide Web Consortium (W3C), 2014)

2.10.4 CSS

Las hojas de estilo en cascada o CSS es un lenguaje usado para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado como HTML.

Junto con HTML Y Javascript constituyen la piedra angular en el desarrollo de la mayoría de sitios y aplicaciones web. (World Wide Web Consortium (W3C), 2015)

2.10.5 C#

C# es un lenguaje de programación que se ha diseñado para compilar diversas aplicaciones que se ejecutan en .NET Framework. C# es simple, eficaz, con seguridad de tipos y orientado a objetos. Las numerosas innovaciones de C# permiten desarrollar aplicaciones rápidamente y mantener la expresividad y elegancia de los lenguajes de estilo de C. (Microsoft, 2016)

2.11 Motores de Base De Datos

2.11.1 PostgreSQL

PostgreSQL es un poderoso sistema de gestión de bases de datos relacional de código abierto con énfasis en la extensibilidad y cumplimiento de estándares. Como servidor de base de datos, su función primaria es almacenar datos de forma segura y permitir la recuperación mediante peticiones de datos por otras aplicaciones. (The PostgreSQL Global Development Group, 2016)

2.11.2 SQLite

SQLite es un sistema de gestión de bases de datos relacional contenido en una librería de programación escrita en C. En contraste con otros manejadores de bases de datos, SQLite no es un motor de bases de datos de tipo cliente-servidor, sino que se encuentra embebida en el programa final. (The SQLite Development Team, 2016)

CAPITULO III

ANÁLISIS Y DISEÑO DE LA APLICACIÓN

La metodología que se usó para el análisis, diseño, desarrollo e implementación del sistema de Digitalización Remota de Documentos está basada en Extreme Programming, la misma que permite adaptarse a los requerimientos cambiantes en la etapa del desarrollo. A continuación, se describe todos los procesos involucrados desde los requerimientos hasta la implementación del sistema.

3.1 Diagrama Físico del Sistema

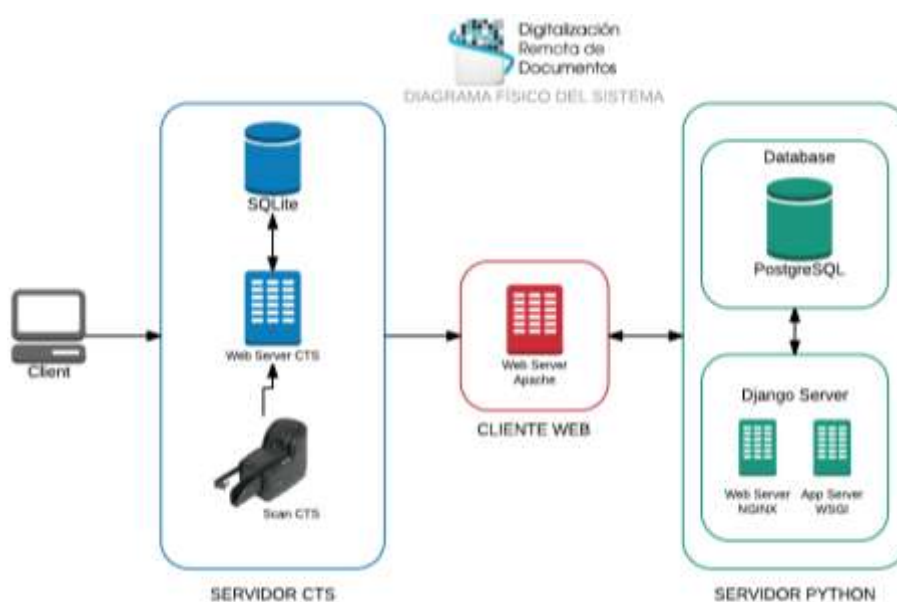


Figura 58 Diagrama Físico del Sistema

El diseño del sistema consta de tres aplicaciones: la primera, denominada *Servidor CTS*, se encarga de la comunicación entre el escáner y el cliente usando las librerías de desarrollo provistas por la empresa CTS Electronics, la segunda aplicación es un *Cliente Web* que corre en un servidor Apache cuya función es mostrar la interfaz gráfica al usuario, y la tercera es un *Servidor Python* que maneja la lógica del negocio mediante servicios REST.

3.2 Elaboración

En el siguiente apartado se definirán los roles y permisos de los usuarios, diagramas de secuencia, historias de usuario que describen los requerimientos del sistema y el diseño de la base de datos.

3.2.1 Usuarios

En el sistema se identificaron los siguientes roles:

- **Súper Administrador:** Persona encargada de parametrizar el sistema.
- **Administrador Entidad Financiera:** Persona que administra el sistema, tiene acceso para crear empresas y visualizar reportes.
- **Administrador Empresa:** Persona encargada de aprobar los depósitos realizados por el usuario, tiene acceso para crear sucursales, cuentas bancarias, procesos y visualizar reportes.
- **Usuario:** Persona que realiza la digitalización de cheques tiene acceso al módulo de digitalización.

3.2.2 Historias de Usuario

3.2.2.1 Entidad Financiera

Tabla 12

Historia de Usuario N° 1

HISTORIA DE USUARIO N° 1	
Nombre historia:	Entidad Financiera
Usuario:	Súper Administrador
Programador responsable: Gerardo Llumiquinga	
DESCRIPCIÓN	
<ul style="list-style-type: none"> • Crear una entidad financiera con los siguientes campos: <ul style="list-style-type: none"> ○ RUC: máximo 13 caracteres. ○ Nombre completo. ○ Nombre corto ○ Siglas ○ Teléfono fijo ○ Email contacto ○ Calle ○ Intersección ○ Numero de edificación. ○ Dato referencial ○ Descripción ○ Fecha creación ○ Ciudad: Seleccionar de una lista establecida anteriormente. ○ Estado. • Listar las entidades financieras. • Editar los campos de entidad financiera • Dar de baja a una entidad financiera (Cambiar de estado). 	
OBSERVACIONES	
<ul style="list-style-type: none"> • Todas las entidades financieras deben tener un ruc único. 	

3.2.2.2 Usuario Entidad Financiera

Tabla 13
Historia de Usuario N° 2

HISTORIA DE USUARIO N° 2	
Nombre historia:	Usuario Entidad Financiera
Usuario:	Súper Administrador
Programador responsable: Gerardo Llumiquinga	
DESCRIPCIÓN	
<ul style="list-style-type: none"> • Crear un usuario entidad financiera con los siguientes campos: <ul style="list-style-type: none"> ○ Cédula: máximo 13 caracteres. ○ Nombre de usuario. ○ Nombre. ○ Apellido. ○ Email. ○ Entidad Financiera: Seleccionar a que Entidad Financiera pertenece el nuevo usuario. • Listar los usuarios. • Editar los campos de usuario. • Dar de baja a un usuario Entidad Financiera. 	
OBSERVACIONES	
<ul style="list-style-type: none"> • Todos los usuarios entidad financiera deben tener un nombre de usuario único. 	

3.2.2.3 Empresa

Tabla 14
Historia de Usuario N° 3

HISTORIA DE USUARIO N° 3	
Nombre historia:	Empresa
Usuario:	Administrador Entidad Financiera.
Programador responsable: Santiago Benalcázar	
DESCRIPCIÓN	
<ul style="list-style-type: none"> • Crear una empresa con los siguientes campos: <ul style="list-style-type: none"> ○ RUC: máximo 13 caracteres. ○ Entidad Financiera: se debe asignar una entidad financiera anteriormente creada. ○ Nombre completo. ○ Nombre corto. ○ Teléfono fijo. ○ Email de contacto. ○ Calle ○ Intersección. ○ Numero de edificación. ○ Dato referencial. ○ Ciudad: Seleccionar de una lista establecida anteriormente. ○ Descripción. • Listar todas las empresas. • Editar los campos de usuario. • Dar de baja a una Empresa. 	
OBSERVACIONES	
<ul style="list-style-type: none"> • Todas las empresas deben tener un RUC único de identificación. 	

3.2.2.4 Usuario Administrador de Empresa.

Tabla 15
Historia de Usuario N° 4

HISTORIA DE USUARIO N° 4	
Nombre historia:	Usuario Administrador de Empresa.
Usuario:	Administrador Entidad Financiera.
Programador responsable: Santiago Benalcázar	
DESCRIPCIÓN	
<ul style="list-style-type: none"> • Crear un usuario administrador de empresa con los siguientes campos: <ul style="list-style-type: none"> ○ Cédula: máximo 13 caracteres. ○ Nombre de usuario. ○ Nombre. ○ Apellido. ○ Email. ○ Empresa: Seleccionar a que empresa pertenece el nuevo usuario. • Listar los usuarios administradores. • Editar los campos de usuario. • Dar de baja a un usuario. 	
OBSERVACIONES	
<ul style="list-style-type: none"> • Todos los usuarios administradores de empresa deben tener un nombre de usuario único. 	

3.2.2.5 Cuenta Bancaria

Tabla 16
Historia de Usuario N° 5

HISTORIA DE USUARIO N° 5	
Nombre historia:	Cuenta Bancaria
Usuario:	Administrador Empresa
Programador responsable: Santiago Benalcázar	
DESCRIPCIÓN	
<ul style="list-style-type: none"> • Crear una cuenta bancaria con los siguientes campos: <ul style="list-style-type: none"> ○ Tipo de cuenta: Seleccionar de una lista (Ahorros o Corriente). ○ Empresa: seleccionar la empresa a quien pertenece. ○ Entidad Financiera: Seleccionar la entidad financiera a quien pertenece. ○ Número de cuenta. • Listar las cuentas bancarias dependiendo de cada empresa. • Editar los campos de la cuenta bancaria. • Dar de baja a una cuenta bancaria. 	
OBSERVACIONES	
<ul style="list-style-type: none"> • Todas las cuentas bancarias deben ser únicas dependiendo de cada empresa y entidad financiera. 	

3.2.2.6 Proceso

Tabla 17
Historia de Usuario N° 6

HISTORIA DE USUARIO N° 6	
Nombre historia:	Cuenta Bancaria
Usuario:	Administrador Empresa
Programador responsable: Gerardo Llumiquinga	
DESCRIPCIÓN	
<ul style="list-style-type: none"> • Crear un proceso con los siguientes campos: <ul style="list-style-type: none"> ○ Sucursal: Seleccionar de una sucursal anteriormente ingresada. ○ Nombre. ○ Código: Un valor identificativo de cada proceso. ○ Descripción. • Listar los procesos dependiendo de cada empresa. • Editar los campos del proceso. • Dar de baja a un proceso. 	
OBSERVACIONES	
<ul style="list-style-type: none"> • Todos los procesos deben ser únicas dependiendo de cada sucursal a la que pertenecen. 	

3.2.2.7 Sucursal

Tabla 18
Historia de Usuario N° 7

HISTORIA DE USUARIO N° 7	
Nombre historia:	Sucursal
Usuario:	Administrador Empresa
Programador responsable: Gerardo Llumiquinga	
DESCRIPCIÓN	
<ul style="list-style-type: none"> • Crear un usuario administrador de empresa con los siguientes campos: <ul style="list-style-type: none"> ○ Nombre. ○ Empresa: Seleccionar a que empresa pertenece. ○ Lista de cuentas bancarias: Seleccionar varias cuentas bancarias anteriormente creadas. ○ Teléfono fijo. ○ Calle. ○ Intersección. ○ Numero de edificación. ○ Dato referencial. ○ Ciudad: Seleccionar de una lista establecida anteriormente. ○ Descripción. • Listar los usuarios administradores. • Editar los campos de usuario. • Dar de baja a un usuario. 	
OBSERVACIONES	
<ul style="list-style-type: none"> • Todos los usuarios administradores de empresa deben tener un nombre de usuario único. 	

3.2.2.8 Usuario

Tabla 19
Historia de Usuario N° 8

HISTORIA DE USUARIO N° 8	
Nombre historia:	Usuario.
Usuario:	Administrador Empresa.
Programador responsable: Santiago Benalcázar	
DESCRIPCIÓN	
<ul style="list-style-type: none"> • Crear un usuario con los siguientes campos: <ul style="list-style-type: none"> ○ Cédula: máximo 13 caracteres. ○ Nombre de usuario. ○ Nombre. ○ Apellido. ○ Email. ○ Sucursal: Seleccionar a que sucursal pertenece el nuevo usuario. • Listar los usuarios. • Editar los campos del usuario. • Dar de baja a un usuario. 	
OBSERVACIONES	
<ul style="list-style-type: none"> • Todos los usuarios deben tener un nombre de usuario único. 	

3.2.2.9 Digitalización del Cheque

Tabla 20
Historia de Usuario N° 9

HISTORIA DE USUARIO N° 9	
Nombre historia:	Digitalización del cheque
Usuario:	Usuario
Programador responsable: Santiago Benalcázar	
DESCRIPCIÓN	
<ul style="list-style-type: none"> • Listar los cheques digitalizados. • Permitir el ingreso del monto de cada cheque. • Visualizar al cheque en la pantalla y agregar las siguientes opciones: <ul style="list-style-type: none"> ○ Zoom +/- al cheque digitalizado. ○ Visualizar el cheque en ambas caras. ○ Visualizar el monto del cheque cuando se lo seleccione. • Eliminar un cheque de la lista. 	
OBSERVACIONES	
<ul style="list-style-type: none"> • El proceso de digitalización debe ser realizado desde el equipo CTS. • El dispositivo de CTS debe mandar alertas al usuario cuando ocurra problemas. 	

3.2.2.10 Depósito de Cheques

Tabla 21
Historia de Usuario N° 10

HISTORIA DE USUARIO N° 10	
Nombre historia:	Depósito de cheques
Usuario:	Usuario
Programador responsable: Santiago Benalcázar	
DESCRIPCIÓN	
<ul style="list-style-type: none"> • Crear un depósito con los siguientes parámetros: <ul style="list-style-type: none"> ○ Seleccionar proceso. ○ Seleccionar tipo de cuenta. ○ Seleccionar número de cuenta. ○ Ingresar monto a depositar. ○ Lista de cheques a depositar. • Agregar la lista de cheques (ver historia de usuario No. 9). 	
OBSERVACIONES	
<ul style="list-style-type: none"> • El usuario podrá ingresar el monto total a depositar. • Se debe realizar la prueba cero la que consiste en cuadrar cada valor de los cheques hasta que el valor total sea cero. 	

3.2.2.11 Comprobante de depósito

Tabla 22
Historia de Usuario N° 11

HISTORIA DE USUARIO N° 11	
Nombre historia:	Comprobante de depósito
Usuario:	Administrador Empresa
Programador responsable: Santiago Benalcázar	
DESCRIPCIÓN	
<ul style="list-style-type: none"> • Crear un documento PDF con los siguientes parámetros: <ul style="list-style-type: none"> ○ Nombre del banco. ○ Número de cuenta. ○ Número de cheques depositados (ver historia de usuario No 8 y 9). ○ Monto total del depósito (ver historia de usuario No 8 y 9). ○ Nombre de la empresa. ○ Nombre del operador. ○ Lugar y fecha. ○ Lista de cheques depositados (ver historia de usuario No 8 y 9). 	
OBSERVACIONES	
<ul style="list-style-type: none"> • El Administrador Empresa debe verificar el comprobante de depósito para su posterior aprobación del depósito. 	

3.2.2.12 Aprobación de Depósito

Tabla 23
Historia de Usuario N° 12

HISTORIA DE USUARIO N° 12	
Nombre historia:	Aprobación de depósito
Usuario:	Administrador Empresa
Programador responsable: Gerardo Llumiquinga	
DESCRIPCIÓN	
<ul style="list-style-type: none"> Listar los depósitos realizados por el usuario. Agrupar por los siguientes campos: <ul style="list-style-type: none"> Tipo de cuenta. Número de cuenta. Fecha de depósito. Nombre del usuario. Ciudad. Sucursal. Visualizar el comprobante de depósito. Visualizar los cheques de cada comprobante. Aprobar el depósito. Eliminar depósito. 	
OBSERVACIONES	
<ul style="list-style-type: none"> El Administrador Empresa debe aprobar el depósito para que se efectivice el depósito. El Administrador Entidad Financiera puede visualizar los depósitos efectivizados. 	

3.2.2.13 Reporte de Depósitos Empresa

Tabla 24
Historia de Usuario N° 13

HISTORIA DE USUARIO N° 13	
Nombre historia:	Reporte de Depósito Empresa
Usuario:	Administrador Empresa
Programador responsable: Gerardo Llumiquinga	
DESCRIPCIÓN	
<ul style="list-style-type: none"> • El Administrador Empresa debe realizar un reporte de depósitos basado en los siguientes criterios: <ul style="list-style-type: none"> ○ Rango de fechas. ○ Número de cuenta depositante. ○ Numero de cheque. ○ Nombre de usuario. • Exporta los datos de la consulta a Excel o PDF. • Agrupar los depósitos por los siguientes criterios: <ul style="list-style-type: none"> ○ Nombre del proceso. ○ Nombre Entidad Financiera. ○ Tipo de Cuenta. ○ Número de Cuenta. ○ Fecha de depósito. ○ Nombre de usuario. ○ Sucursal. • Visualizar el depósito. • Visualizar los cheques de cada depósito. 	
OBSERVACIONES	
<ul style="list-style-type: none"> • Ninguna. 	

3.2.2.14 Reporte de Depósitos Entidad Financiera

Tabla 25
Historia de Usuario N° 14

HISTORIA DE USUARIO N° 14	
Nombre historia:	Reporte de Depósito Empresa
Usuario:	Administrador Entidad Financiera
Programador responsable: Gerardo Llumiquinga	
DESCRIPCIÓN	
<ul style="list-style-type: none"> El Administrador Empresa debe realizar un reporte de depósitos basado en los siguientes criterios: <ul style="list-style-type: none"> Rango de fechas. Nombre de empresa. Número de cuenta depositante. Número de depósito. Número de cheque. Exporta los datos de la consulta a Excel o PDF. Agrupar los depósitos por los siguientes criterios: <ul style="list-style-type: none"> Nombre Empresa depositante. Tipo de Cuenta. Número de Cuenta. Fecha de depósito. Nombre de usuario. Ciudad. Sucursal. Visualizar el depósito. Visualizar los cheques de cada depósito. 	
OBSERVACIONES	
<ul style="list-style-type: none"> Ninguna. 	

3.2.3 Diagramas De Secuencia

3.2.3.1 Inicio de Sesión

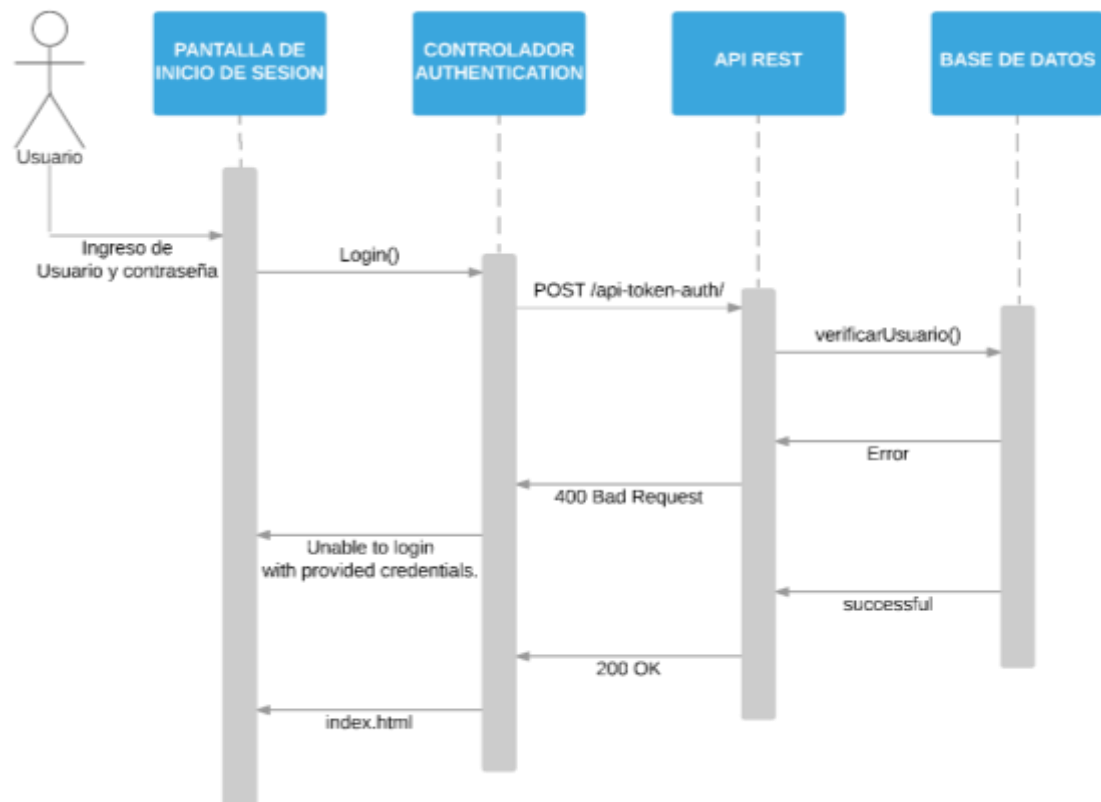


Figura 59 Diagrama de Secuencia: Inicio de Sesión

3.2.3.2 Operación CRUD (Sucursal)

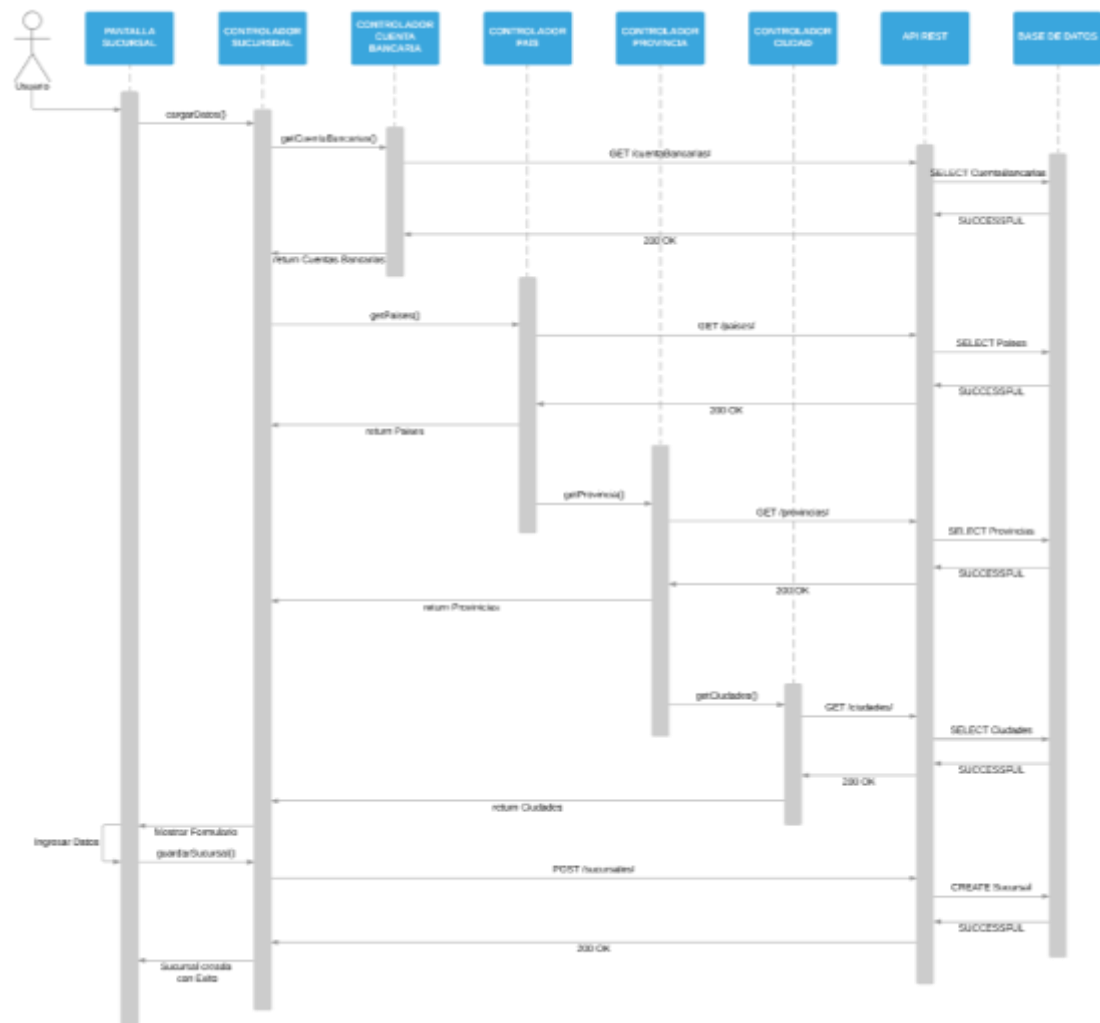


Figura 60 Diagrama de Secuencia: Operación CRUD (Sucursal)

3.2.3.3 Módulo Digitalización

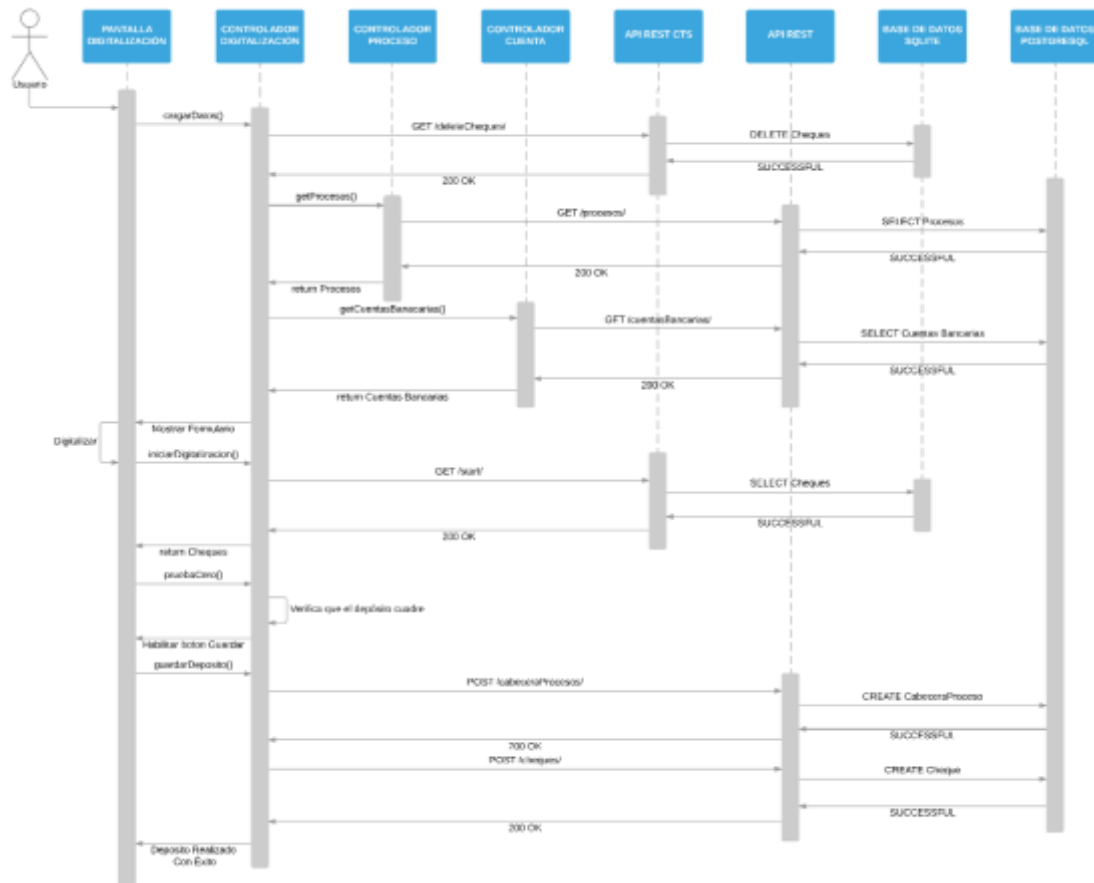


Figura 61 Diagrama de Secuencia: Módulo Digitalización

3.2.3.4 Aprobación de Depósito

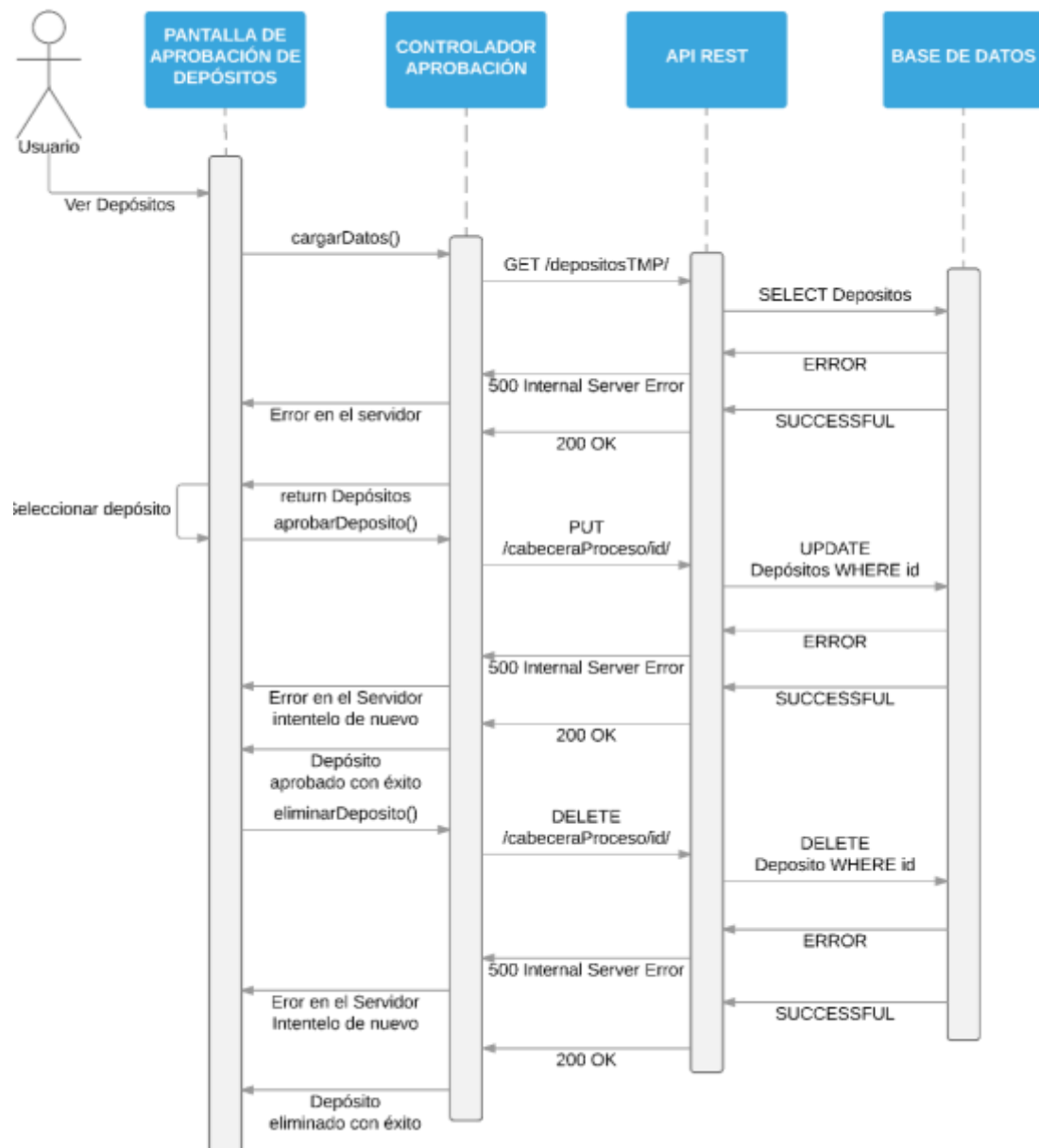


Figura 62 Diagrama de Secuencia: Aprobación de Depósito

3.2.3.5 Reporte Entidad Financiera

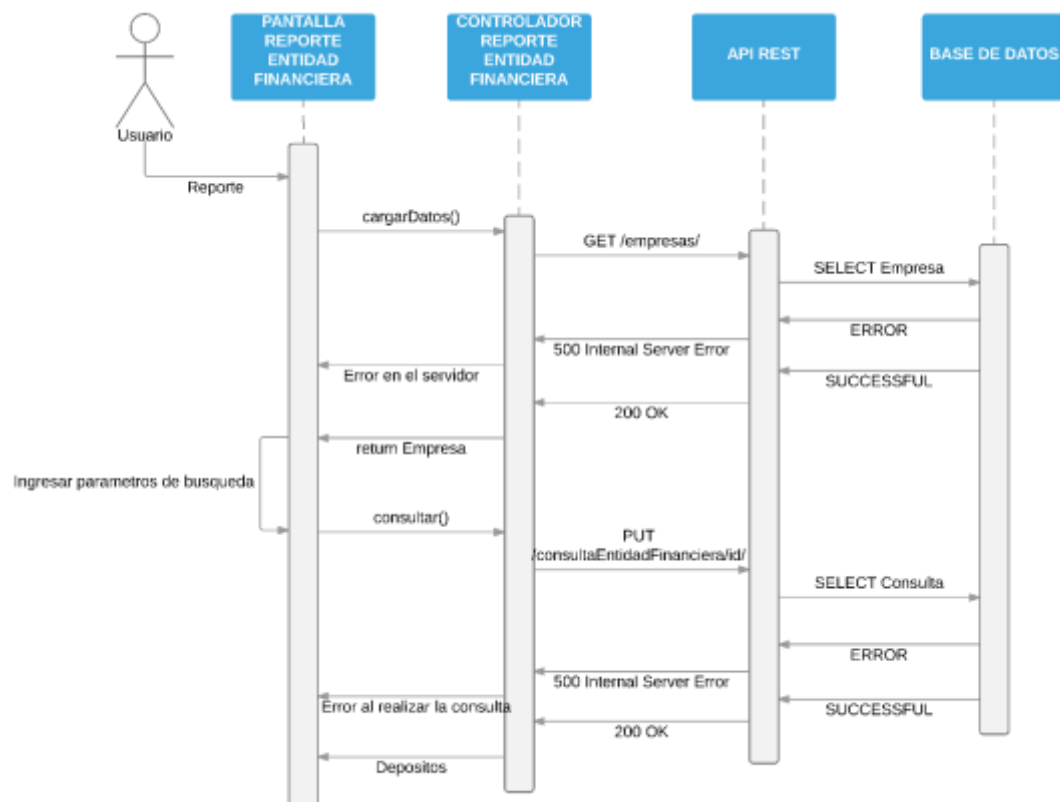


Figura 63 Diagrama de Secuencia: Reporte Entidad Financiera

3.2.4 Diseño de la Base de Datos

3.2.4.1 Diagrama Entidad Relación

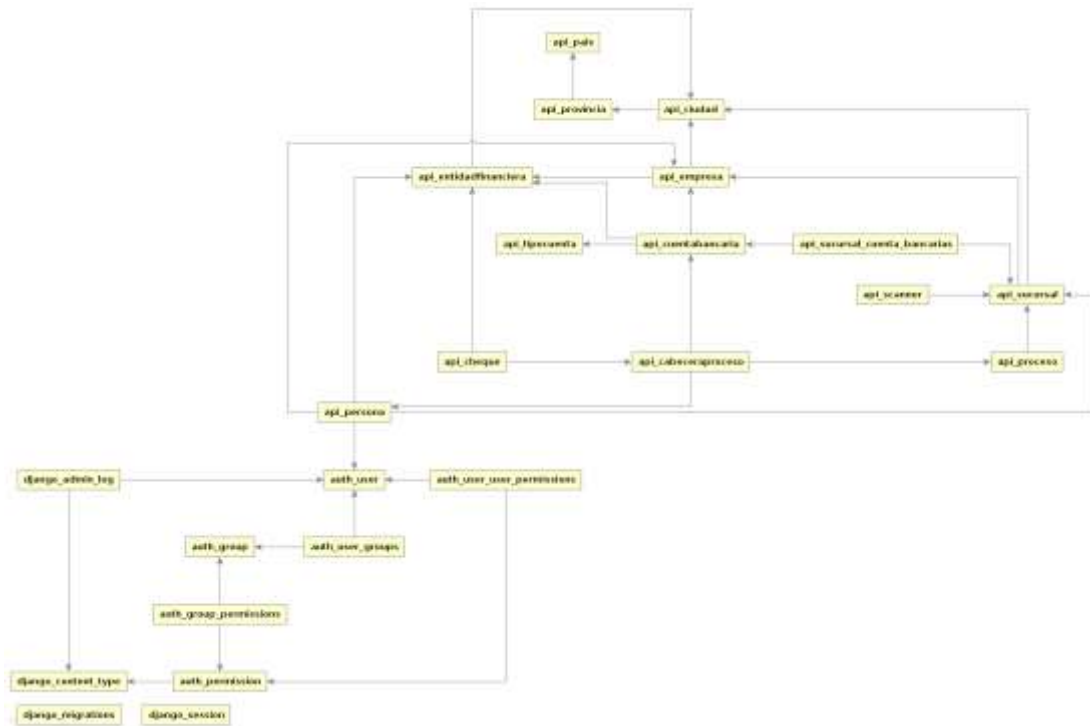


Figura 64 Base de Datos: Diagrama Entidad Relación

3.2.4.2 Módulo de Digitalización

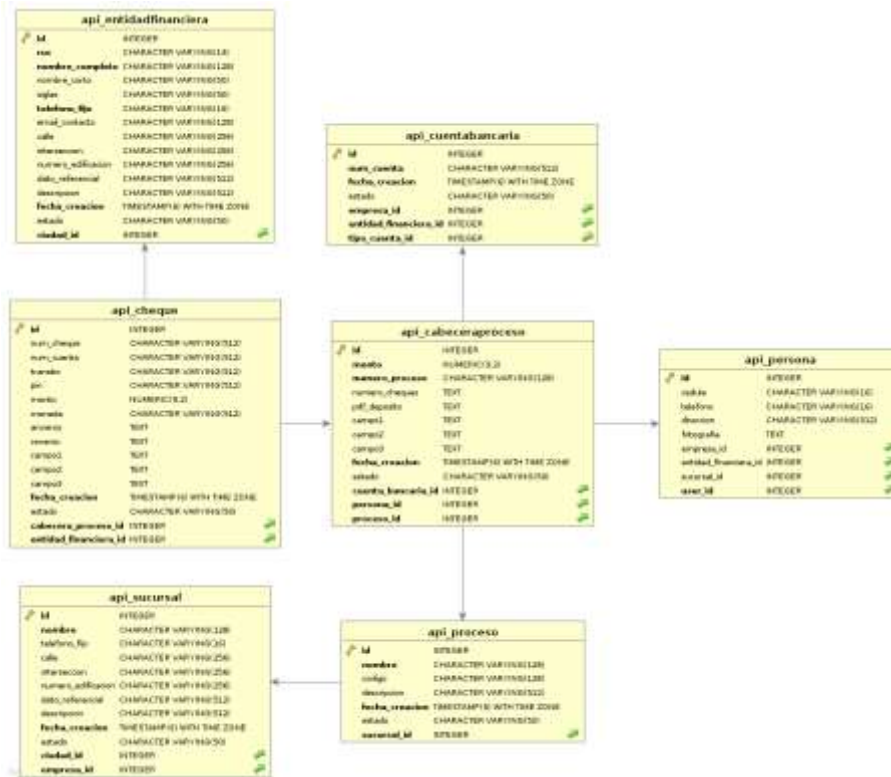


Figura 65 Base de Datos: Módulo Digitalización

3.2.4.3 Módulo De Seguridad

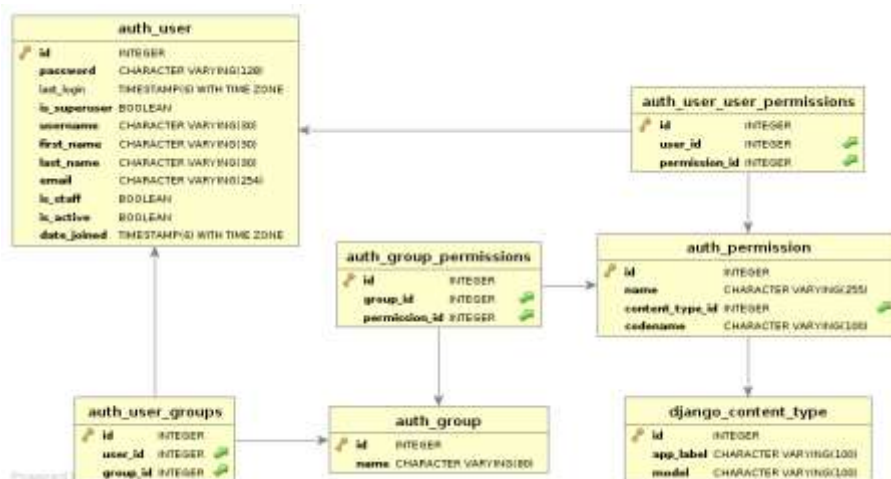


Figura 66 Base de Datos: Módulo de Seguridad

3.2.4.4 Diccionario De Datos





api_cabeceraproceso		
	id	INTEGER
	monto	NUMERIC(9,2)
	numero_proceso	CHARACTER VARYING(128)
	numero_cheques	TEXT
	pdf_deposito	TEXT
	campo1	TEXT
	campo2	TEXT
	campo3	TEXT
	fecha_creacion	TIMESTAMP(6) WITH TIME ZONE
	estado	CHARACTER VARYING(50)
	cuenta_bancaria_id	INTEGER 
	persona_id	INTEGER 
	proceso_id	INTEGER 

Figura 67 Base de Datos: Tabla cabeceraproceso




api_cheque		
	id	INTEGER
	num_cheque	CHARACTER VARYING(512)
	num_cuenta	CHARACTER VARYING(512)
	transito	CHARACTER VARYING(512)
	pin	CHARACTER VARYING(512)
	monto	NUMERIC(9,2)
	moneda	CHARACTER VARYING(512)
	anverso	TEXT
	reverso	TEXT
	campo1	TEXT
	campo2	TEXT
	campo3	TEXT
	fecha_creacion	TIMESTAMP(6) WITH TIME ZONE
	estado	CHARACTER VARYING(50)
	cabecera_proceso_id	INTEGER 
	entidad_financiera_id	INTEGER 

Figura 68 Base de Datos: Tabla cheque



api_ciudad	
 id	INTEGER
nombre	CHARACTER VARYING(50)
descripcion	TEXT
fecha_creacion	TIMESTAMP(6) WITH TIME ZONE
estado	CHARACTER VARYING(50)
provincia_id	INTEGER 

Figura 69 Base de Datos: Tabla ciudad

api_consultacliente	
id	INTEGER
proceso_id	INTEGER
proceso_nombre	CHARACTER VARYING(128)
entidad_id	INTEGER
entidad_nombre	CHARACTER VARYING(128)
tipocuenta_id	INTEGER
tipocuenta_nombre	CHARACTER VARYING(128)
cuenta_id	INTEGER
cuenta_num	CHARACTER VARYING(512)
cabproceso_id	INTEGER
fecha_creacion	TIMESTAMP(6) WITH TIME ZONE
monto	NUMERIC(9,2)
num_cheque	TEXT
user_id	INTEGER
user_first_name	CHARACTER VARYING(30)
user_last_name	CHARACTER VARYING(30)
sucural_id	INTEGER
sucursal_nombre	CHARACTER VARYING(128)
persona_id	INTEGER
pdf_deposito	TEXT
empresa_id	INTEGER
empresa_nombre_completo	CHARACTER VARYING(128)
empresa_nombre_corto	CHARACTER VARYING(50)
ciudad_id	INTEGER
ciudad_nombre	CHARACTER VARYING(50)
estado	CHARACTER VARYING(50)

Figura 70 Base de Datos: Tabla consultacliente





api_cuentabancaria		
 id	INTEGER	
num_cuenta	CHARACTER VARYING(512)	
fecha_creacion	TIMESTAMP(6) WITH TIME ZONE	
estado	CHARACTER VARYING(50)	
empresa_id	INTEGER	
entidad_financiera_id	INTEGER	
tipo_cuenta_id	INTEGER	

Figura 71 Base de Datos: Tabla cuentabancaria

api_detalledeposito	
id	INTEGER
banco	CHARACTER VARYING(128)
num_cuenta	CHARACTER VARYING(512)
empresa	CHARACTER VARYING(128)
fecha	TIMESTAMP(6) WITH TIME ZONE
ciudad	CHARACTER VARYING(50)
monto	NUMERIC(9,2)
tecniconombre	CHARACTER VARYING(30)
tecnicoapellido	CHARACTER VARYING(30)

Figura 72 Base de Datos: Tabla detalledeposito




api_empresa		
 id	INTEGER	
ruc	CHARACTER VARYING(13)	
nombre_completo	CHARACTER VARYING(128)	
nombre_corto	CHARACTER VARYING(50)	
telefono_fijo	CHARACTER VARYING(16)	
email_contacto	CHARACTER VARYING(128)	
calle	CHARACTER VARYING(256)	
interseccion	CHARACTER VARYING(256)	
numero_edificacion	CHARACTER VARYING(256)	
dato_referencial	CHARACTER VARYING(512)	
descripcion	CHARACTER VARYING(512)	
fecha_creacion	TIMESTAMP(6) WITH TIME ZONE	
estado	CHARACTER VARYING(50)	
ciudad_id	INTEGER	
entidad_financieras_id	INTEGER	

Figura 73 Base de Datos: Tabla empresa



api_entidadfinanciera	
 id	INTEGER
ruc	CHARACTER VARYING(13)
nombre_completo	CHARACTER VARYING(128)
nombre_corto	CHARACTER VARYING(50)
siglas	CHARACTER VARYING(50)
telefono_fijo	CHARACTER VARYING(16)
email_contacto	CHARACTER VARYING(128)
calle	CHARACTER VARYING(256)
interseccion	CHARACTER VARYING(256)
numero_edificacion	CHARACTER VARYING(256)
dato_referencial	CHARACTER VARYING(512)
descripcion	CHARACTER VARYING(512)
fecha_creacion	TIMESTAMP(6) WITH TIME ZONE
estado	CHARACTER VARYING(50)
ciudad_id	INTEGER 

Figura 74 Base de Datos: Tabla entidadfinanciera


api_pais	
 id	INTEGER
nombre	CHARACTER VARYING(50)
siglas	CHARACTER VARYING(3)
descripcion	TEXT
fecha_creacion	TIMESTAMP(6) WITH TIME ZONE
estado	CHARACTER VARYING(50)

Figura 75 Base de Datos: Tabla pais






api_persona	
 id	INTEGER
cedula	CHARACTER VARYING(16)
telefono	CHARACTER VARYING(16)
direccion	CHARACTER VARYING(512)
fotografia	TEXT
empresa_id	INTEGER 
entidad_financiera_id	INTEGER 
sucursal_id	INTEGER 
user_id	INTEGER 

Figura 76 Base de Datos: Tabla persona



api_proceso		
	id	INTEGER
	nombre	CHARACTER VARYING(128)
	codigo	CHARACTER VARYING(128)
	descripcion	CHARACTER VARYING(512)
	fecha_creacion	TIMESTAMP(6) WITH TIME ZONE
	estado	CHARACTER VARYING(50)
	sucursal_id	INTEGER 

Figura 77 Base de Datos: Tabla proceso



api_provincia		
	id	INTEGER
	nombre	CHARACTER VARYING(50)
	descripcion	TEXT
	fecha_creacion	TIMESTAMP(6) WITH TIME ZONE
	estado	CHARACTER VARYING(50)
	pais_id	INTEGER 

Figura 78 Base de Datos: Tabla provincia



api_scanner		
	id	INTEGER
	nombre	CHARACTER VARYING(128)
	modelo	CHARACTER VARYING(128)
	serie	CHARACTER VARYING(128)
	marca	CHARACTER VARYING(128)
	descripcion	CHARACTER VARYING(512)
	fecha_creacion	TIMESTAMP(6) WITH TIME ZONE
	estado	CHARACTER VARYING(50)
	sucursal_id	INTEGER 

Figura 79 Base de Datos: Tabla scanner




api_sucursal	
 id	INTEGER
nombre	CHARACTER VARYING(128)
telefono_fijo	CHARACTER VARYING(16)
calle	CHARACTER VARYING(256)
interseccion	CHARACTER VARYING(256)
numero_edificacion	CHARACTER VARYING(256)
dato_referencial	CHARACTER VARYING(512)
descripcion	CHARACTER VARYING(512)
fecha_creacion	TIMESTAMP(6) WITH TIME ZONE
estado	CHARACTER VARYING(50)
ciudad_id	INTEGER 
empresa_id	INTEGER 

Figura 80 Base de Datos: Tabla sucursal




api_sucursal_cuenta_bancarias	
 id	INTEGER
sucursal_id	INTEGER 
cuentabancaria_id	INTEGER 

Figura 81 Base de Datos: Tabla cuentabancaria


api_tipocuenta	
 id	INTEGER
nombre	CHARACTER VARYING(128)
descripcion	CHARACTER VARYING(512)
fecha_creacion	TIMESTAMP(6) WITH TIME ZONE
estado	CHARACTER VARYING(50)

Figura 82 Base de Datos: Tabla tipocuenta

CAPITULO IV

DESARROLLO E IMPLEMENTACIÓN

4.1 Arquitectura del Sistema

El sistema de Digitalización Remota de Documentos consta de tres aplicaciones independientes: Para el Cliente Web, se desarrolló una aplicación de una sola página o Single Page Application utilizando el framework AngularJS que se encarga de gestionar todos los recursos HTML, CSS y Javascript en el lado del cliente.

Por otra parte, el Servidor Python está desarrollado usando el framework Django REST que se encarga de manejar toda la lógica del negocio y la comunicación con la persistencia haciendo uso de una base de datos PostgreSQL y por otro lado el segundo servidor denominado Servidor CTS usa el framework .NET en conjunto con las librerías de desarrollo de CTS para la comunicación con el escáner y la administración de los cheques, usando para la persistencia una base de datos embebida SQLite.

A continuación, se presenta el esquema de la arquitectura del sistema:

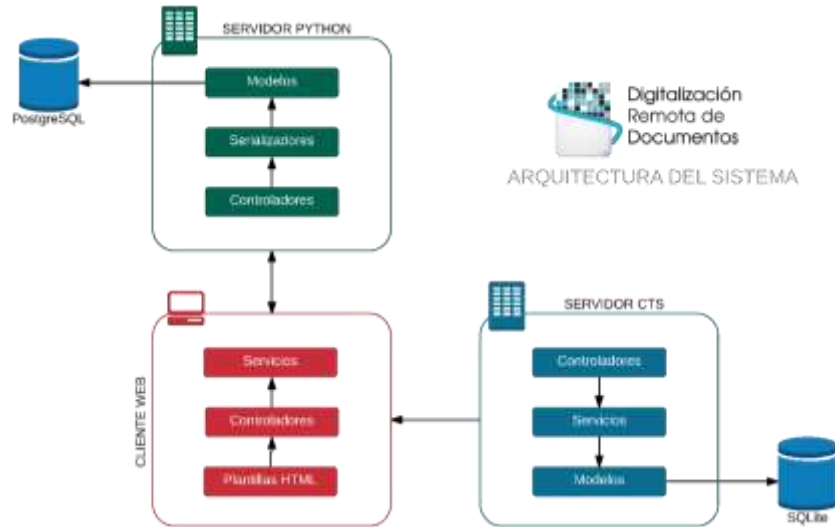


Figura 83 Arquitectura del Sistema

4.1.1 Cliente Web - AngularJS

El Cliente Web está desarrollado haciendo uso del patrón de diseño Modelo-Vista-Controlador (MVC) e inyección de dependencias que es la principal característica del framework AngularJS, consta de las siguientes capas:

4.1.1.1 Capa de Servicios

Los servicios son clases que interactúan directamente con el Servidor Python mediante una URI definida en el servidor y son los encargados de realizar las operaciones CRUD para la administración de los datos de la aplicación.

```

1  angular.module('Usuario', [])
2  .factory('UsuarioService', ['urls', '$resource', function (urls, $resource) {
3      return $resource(urls.api + '/users/:id/', {id: '@id'}, {
4          update: {
5              method: 'PUT'
6          },
7          drdremove: {
8              method: 'PUT',
9              url: urls.api + '/users/delete/:id/'
10         }
11     });
12 });

```

Figura 84 AngularJS: Servicio

4.1.1.2 Capa de Controladores

Los controladores responden a las entradas que realice el usuario y ejecuta interacciones con los servicios. Los controladores reciben entradas, validan los datos y ejecutan operaciones para modificar el estado de los datos en la aplicación.

```

1  angular.module('Usuario', [])
2  .controller('UsuarioController', ['UsuarioService', '$scope',
3  function (UsuarioService, $scope) {
4      // READ
5      $scope.users = UsuarioService.query();
6
7      $scope.save = function (user) {
8          if ($scope.userForm.$valid) {
9              if (!user.id) {
10                 var record = new UsuarioService();
11                 record.username = user.username;
12                 record.first_name = user.first_name;
13                 record.last_name = user.last_name;
14                 record.email = user.email;
15                 // CREATE
16                 record.$save(function (e) {
17                     $scope.users = UsuarioService.query();
18                 });
19             } else {
20                 // UPDATE
21                 user.$update();
22             }
23             $scope.users = UsuarioService.query();
24             $('#myModal').modal('hide');
25         }
26     };
27
28     // DELETE (Update with status "DES")
29     $scope.remove = function (user) {
30         user.$rdremove(function () {
31             $scope.users = UsuarioService.query();
32         });
33     };
34 }
35 });

```

Figura 85 AngularJS: Controlador

4.1.1.3 Capa de Vistas

Las vistas están compuestas por templates HTML y directivas de AngularJS las cuales son manejadas por los controladores y definen la forma con la que se presentan los datos al usuario. (LibrosWeb, 2016)

```

1 <div ng-controller="UsuarioController">
2   <div class="panel panel-primary">
3     <div class="panel-heading"><h3 class="panel-title">Usuarios</h3></div>
4     <div class="panel-body">
5       <button type="button" class="btn btn-primary pull-right" ng-click="showWindow()">
6         <i class="glyphicon glyphicon-plus"></i>
7       </button>
8       <table class="table table-hover">
9         <thead>
10          <tr>
11            <th>Username</th>
12            <th>Nombre</th>
13            <th>Apellido</th>
14            <th>Email</th>
15            <th>Fecha Creación</th>
16          </tr>
17        </thead>
18        <tbody>
19          <tr ng-repeat="user in users">
20            <td>{{user.username}}</td>
21            <td>{{user.first_name}}</td>
22            <td>{{user.last_name}}</td>
23            <td>{{user.email}}</td>
24            <td>{{user.date_joined | date}}</td>
25            <td>
26              <button type="button" class="btn btn-primary" ng-click="showWindow(user)"><i
27                class="glyphicon glyphicon-pencil"></i></button>
28              <button type="button" class="btn btn-danger" ng-click="remove(user)"><i
29                class="glyphicon glyphicon-remove"></i>
30            </button>
31            </td>
32          </tr>
33        </tbody>
34      </table>
35    </div>
36  </div>

```

Figura 86 AngularJS: Template

4.1.2 Servidor Python - Django Rest API

La primera aplicación es una API REST que usa el patrón de diseño Modelo-Vista-Controlador (MVC), consta de las siguientes capas:

4.1.2.1 Capa de Controladores

Esta capa contiene la lógica para administrar los datos y además se comunica con la configuración de URLs (URLconf) manejada por el framework para llamar a la función apropiada de Python para la URL obtenida.

```

1 class PaisList(generics.ListCreateAPIView):
2     queryset = Pais.objects.all().filter(estado='ACT')
3     serializer_class = PaisSerializer
4
5
6 class PaisDetail(generics.RetrieveUpdateDestroyAPIView):
7     queryset = Pais.objects.all()
8     serializer_class = PaisSerializer
9
10
11 class PaisDelete(generics.UpdateAPIView):
12     queryset = Pais.objects.all()
13     serializer_class = PaisSerializer
14
15     def update(self, request, *args, **kwargs):
16         request.data['estado'] = 'DES'
17         return super(PaisDelete, self).update(request, *args, **kwargs)

```

Figura 87 Django REST: Controlador

4.1.2.2 Capa de Serializadores

Actúa como una capa intermedia entre los controladores y la persistencia y se encarga de convertir datos complejos como querysets o modelos a tipos de datos que puedan ser fácilmente leídos como JSON.

```

1 class PaisSerializer(serializers.ModelSerializer):
2     class Meta:
3         model = Pais
4         fields = '__all__'

```

Figura 88 Django REST: Serializador

4.1.2.3 Capa de Persistencia

La capa de persistencia usa Modelos que son tipos de datos especiales que contienen toda los campos y comportamiento de los datos que tenemos almacenados, es decir que cada modelo mapea una clase de Python hacia una Tabla de la base de datos.

```

1 class Pais(models.Model):
2     nombre = models.CharField(max_length=50, unique=True)
3     siglas = models.CharField(max_length=3, blank=True, null=True)
4     descripcion = models.TextField(blank=True, null=True)
5     fecha_creacion = models.DateTimeField(auto_now_add=True)
6     estado = models.CharField(default='ACT', max_length=50, blank=True, null=True)
7
8     def __unicode__(self):
9         return self.nombre

```

Figura 89 Django REST: Modelo

4.1.2.4 Servicios RESTful

A continuación, se listan los servicios RESTful que maneja el Servidor Python

Tabla 26
Servicios REST: Autorización

AUTORIZACIÓN				
HTTP	URI	Parámetros		Resultado
POST	/api-token-auth/	JSON: Usuario y contraseña		JWT con los datos de usuario autenticado

Tabla 27
Servicios REST: Usuarios

USUARIOS			
HTTP	URI	CRUD	Resultado
GET	/users/	READ	Listar todos
GET	/users/:id	READ	Listar por id
POST	/users/	CREATE	Crear registro
PUT	/users/:id	UPDATE	Actualizar registro
DELETE	/users/:id	DELETE	Eliminar registro

Tabla 28
Servicios REST: Personas

PERSONAS			
HTTP	URI	CRUD	Resultado
GET	/personas/	READ	Listar todos
GET	/personas/:id	READ	Listar por id
POST	/personas/	CREATE	Crear registro
PUT	/personas/:id	UPDATE	Actualizar registro

Tabla 29
Servicios REST: Países

PAÍSES			
HTTP	URI	CRUD	Resultado
GET	/países/	READ	Listar todos
GET	/países/:id	READ	Listar por id
POST	/países/	CREATE	Crear registro
PUT	/países/:id	UPDATE	Actualizar registro
PUT	/países/delete/:id	DELETE	Actualizar estado (desactivado)

Tabla 30
Servicios REST: Provincias

PROVINCIAS			
HTTP	URI	CRUD	Resultado
GET	/provincias/	READ	Listar todos
GET	/provincias/:id	READ	Listar por id
POST	/provincias/	CREATE	Crear registro
PUT	/provincias/:id	UPDATE	Actualizar registro
PUT	/provincias/delete/:id	DELETE	Actualizar estado (desactivado)

Tabla 31
Servicios REST: Ciudades

CIUDADES			
HTTP	URI	CRUD	Resultado
GET	/ciudades/	READ	Listar todos
GET	/ciudades/:id	READ	Listar por id
POST	/ciudades/	CREATE	Crear registro
PUT	/ciudades/:id	UPDATE	Actualizar registro
PUT	/ciudades/delete/:id	DELETE	Actualizar estado (desactivado)

Tabla 32
Servicios REST: Entidades Financieras

ENTIDADES FINANCIERAS			
HTTP	URI	CRUD	Resultado
GET	/entidadFinancieras/	READ	Listar todos
GET	/entidadFinancieras/:id	READ	Listar por id
POST	/entidadFinancieras/	CREATE	Crear registro
PUT	/entidadFinancieras/:id	UPDATE	Actualizar registro
PUT	/entidadFinancieras/delete/:id	DELETE	Actualizar estado (desactivado)

Tabla 33
Servicios REST: Empresas

EMPRESAS			
HTTP	URI	CRUD	Resultado
GET	/empresas/	READ	Listar todos
GET	/empresas/:id	READ	Listar por id
POST	/empresas/	CREATE	Crear registro
PUT	/empresas/:id	UPDATE	Actualizar registro
PUT	/empresas/delete/:id	DELETE	Actualizar estado (desactivado)

Tabla 34
Servicios REST: Sucursales

SUCURSALES			
HTTP	URI	CRUD	Resultado
GET	/sucursales/	READ	Listar todos
GET	/sucursales/:id	READ	Listar por id
POST	/sucursales/	CREATE	Crear registro
PUT	/sucursales/:id	UPDATE	Actualizar registro
PUT	/sucursales/delete/:id	DELETE	Actualizar estado (desactivado)

Tabla 35
Servicios REST: Escáneres

ESCÁNERES			
HTTP	URI	CRUD	Resultado
GET	/scanners/	READ	Listar todos
GET	/scanners/:id	READ	Listar por id
POST	/scanners/	CREATE	Crear registro
PUT	/scanners/:id	UPDATE	Actualizar registro
DELETE	/scanners/:id	DELETE	Eliminar registro

Tabla 36

Servicios REST: Tipos de Cuenta Bancaria

TIPOS DE CUENTA			
HTTP	URI	CRUD	Resultado
GET	/tipoCuentas/	READ	Listar todos
GET	/tipoCuentas/:id	READ	Listar por id
POST	/tipoCuentas/	CREATE	Crear registro
PUT	/tipoCuentas/:id	UPDATE	Actualizar registro
DELETE	/tipoCuentas/:id	DELETE	Eliminar registro

Tabla 37

Servicios REST: Cuentas Bancarias

CUENTAS BANCARIAS			
HTTP	URI	CRUD	Resultado
GET	/cuentaBancarias/	READ	Listar todos
GET	/cuentaBancarias/:id	READ	Listar por id
POST	/cuentaBancarias/	CREATE	Crear registro
PUT	/cuentaBancarias/:id	UPDATE	Actualizar registro
PUT	/cuentaBancarias /delete/:id	DELETE	Actualizar estado (desactivado)

Tabla 38

Servicios REST: Procesos

PROCESOS			
HTTP	URI	CRUD	Resultado
GET	/procesos/	READ	Listar todos
GET	/procesos/:id	READ	Listar por id
POST	/procesos/	CREATE	Crear registro
PUT	/procesos/:id	UPDATE	Actualizar registro
PUT	/procesos/delete/:id	DELETE	Actualizar estado (desactivado)

Tabla 39
Servicios REST: Cabecera de Proceso

CABECERA DE PROCESO			
HTTP	URI	CRUD	Resultado
GET	/cabeceraProcesos/	READ	Listar todos
GET	/cabeceraProcesos/:id	READ	Listar por id
POST	/cabeceraProcesos/	CREATE	Crear registro
PUT	/cabeceraProcesos/:id	UPDATE	Actualizar registro
DELETE	/cabeceraProcesos/:id	DELETE	Eliminar registro

Tabla 40
Servicios REST: Cheques

CHEQUES			
HTTP	URI	CRUD	Resultado
GET	/cheques/	READ	Listar todos
GET	/cheques/:id	READ	Listar por id
POST	/cheques/	CREATE	Crear registro
PUT	/cheques/:id	UPDATE	Actualizar registro
DELETE	/cheques/:id	DELETE	Eliminar registro

Tabla 41
Servicios REST: Consulta de Cliente

CONSULTA DE CLIENTE			
HTTP	URI	CRUD	Resultado
PUT	/consultaCliente/	READ	Listar todos

Tabla 42

Servicios REST: Consulta de Entidad Financiera

CONSULTA DE ENTIDAD FINANCIERA			
HTTP	URI	CRUD	Resultado
PUT	/consultaEntidadFinanciera/	READ	Listar todos

Tabla 43

Servicios REST: Consulta de Depósitos por Cliente

CONSULTA DE DEPÓSITOS POR CLIENTE			
HTTP	URI	CRUD	Resultado
GET	/depositosTMP/	READ	Listar todos

Tabla 44

Servicios REST: Detalle del Depósito

DETALLE DEL DEPÓSITO			
HTTP	URI	CRUD	Resultado
GET	/detalleDepositos/	READ	Listar todos
GET	/detalleDepositos/:id	READ	Listar por id
POST	/detalleDepositos/	CREATE	Crear registro
PUT	/detalleDepositos/:id	UPDATE	Actualizar registro
DELETE	/detalleDepositos/:id	DELETE	Eliminar registro

4.2 Funcionamiento del Sistema

A continuación, se muestra el funcionamiento general del sistema

4.2.1 Pantalla de Inicio de Sesión

Muestra el formulario de ingreso al sistema, permite el acceso a los usuarios de acuerdo a su rol (administradores o digitalizadores).



Figura 90 Pantalla de Inicio de Sesión

4.2.2 Vista Principal

Vista general de la interfaz principal del sistema, en la parte superior una barra principal que contiene el logo de la aplicación y el usuario que usa el sistema; a la izquierda el menú de accesos del usuario; a la derecha el contenedor principal en donde se muestran los diferentes módulos del sistema y en la parte inferior el pie de página con información relacionada con la empresa.



Figura 91 Vista Principal del Sistema

4.2.3 Menú de Accesos

Cada usuario cuenta con un menú de accesos acorde con sus permisos, el sistema cuenta con tres roles: administrador de entidad financiera, administrador de empresa y usuario digitalizador.

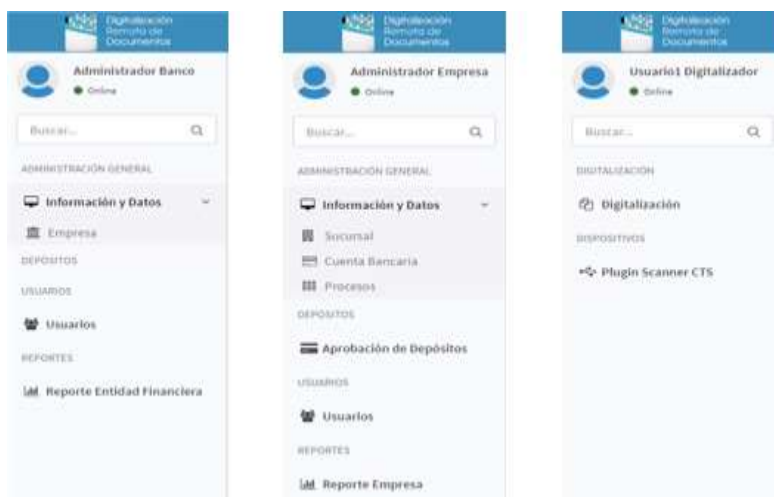


Figura 92 Menú de Accesos

4.2.4 Operaciones CRUD

Plantilla genérica para todas las Tablas de la base de datos que requieran operaciones CRUD; muestra todos los registros que contiene una Tabla de la base de datos, cuenta con las operaciones: añadir, editar y eliminar.

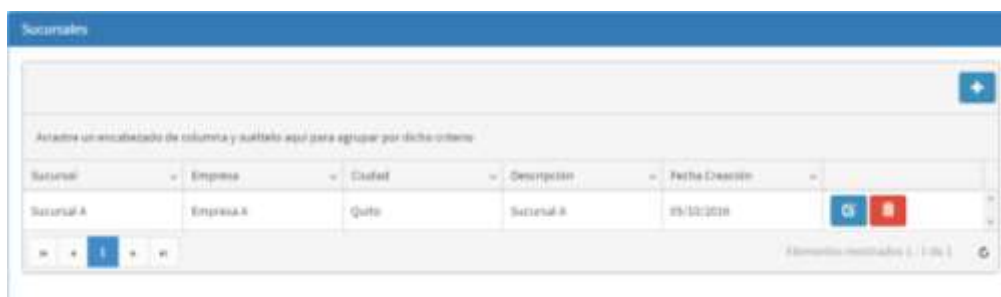


Figura 93 Operación CRUD: Lista Registros

Vista del formulario de crear/editar un registro

Figura 94 Operación CRUD: Formulario Ingreso/Edición del Registro

Formulario con datos y validaciones correspondientes

Figura 95 Operación CRUD: Validaciones formulario

4.2.5 Digitalización

Módulo principal del sistema, permite la digitalización y administración de los cheques. La parte de Información General permite el ingreso de los datos generales de un depósito como proceso, entidad financiera, cuenta bancaria y monto del depósito.

Figura 96 Módulo Digitalización: Vista Principal

Después de ingresar los datos requeridos para el depósito se activa el botón *Digitalizar* para iniciar el proceso. Una vez presionado empieza la digitalización de los cheques mostrando la siguiente interfaz:

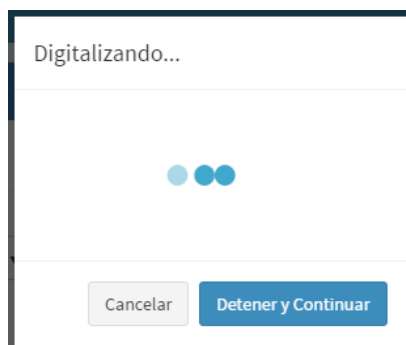


Figura 97 Módulo Digitalización: Vista de espera (digitalizando)

Finalizada la digitalización, se listan los cheques en una Tabla con sus respectivas imágenes pudiendo eliminar un cheque si este presentara errores.

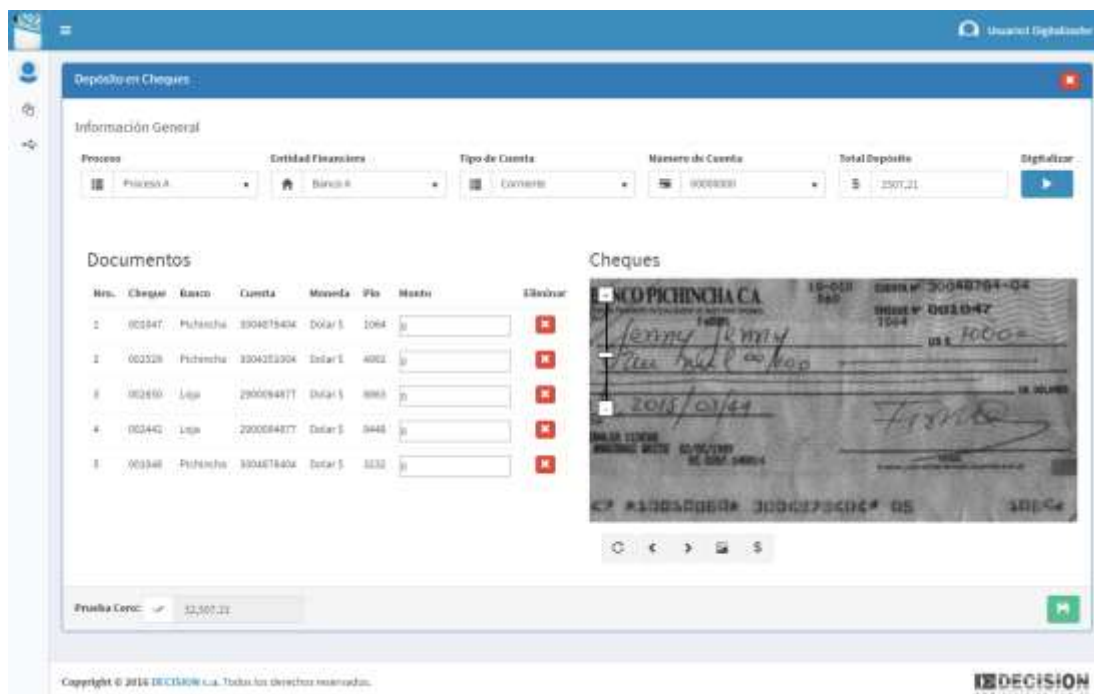


Figura 98 Módulo Digitalización: Proceso Completo

El área que muestra las imágenes de los cheques cuenta con algunos controles de ayuda al usuario como voltear el cheque (1), flechas de navegación (2), mostrar cheque completo (3), mostrar área del monto (4)

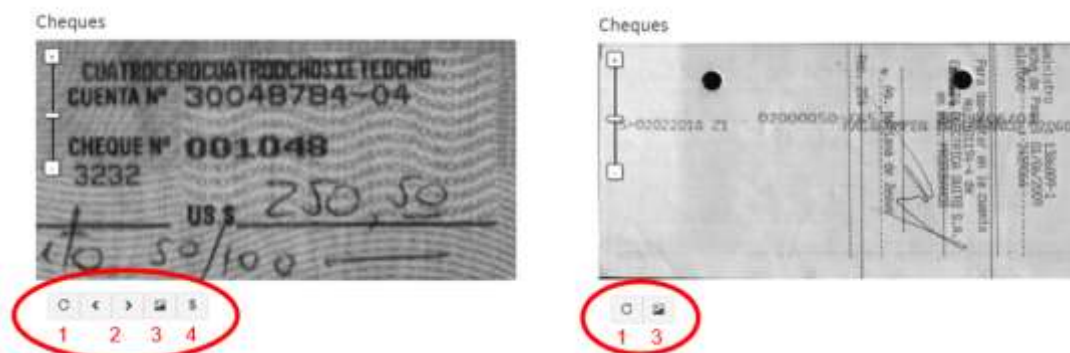


Figura 99 Módulo Digitalización: Controles de navegación de imágenes

4.2.6 Aprobación de Depósitos

Muestra una Tabla con los depósitos pendientes de aprobación con los datos generales de cada uno, un botón para visualizar el documento de confirmación del depósito en PDF y controles para aprobar o eliminar el depósito.

N° Depósito	Empresa	Tipo Cuenta	N° Cuenta	Fecha Depósito	Operador	Ciudad	C/Cheque	Monto Total	N° de Cheque		
1	Empresa A	Corriente	0000000	16/05/2016	Usuario1 Digitalizador	Quito	Secursal A	0.00	A	PDF	✓ ✕
2	Empresa A	Corriente	0000000	16/05/2016	Usuario1 Digitalizador	Quito	Secursal A	0.00	B	PDF	✓ ✕
3	Empresa A	Corriente	0000000	16/05/2016	Usuario1 Digitalizador	Quito	Secursal A	0.00	C	PDF	✓ ✕
4	Empresa A	Corriente	0000000	16/05/2016	Usuario1 Digitalizador	Quito	Secursal A	0.00	D	PDF	✓ ✕
5	Empresa A	Corriente	0000000	16/05/2016	Usuario1 Digitalizador	Quito	Secursal A	0.00	E	PDF	✓ ✕

Figura 100 Aprobación de Depósitos

DEPÓSITO EN CHEQUES

Nº: 10

Banco.....: Banco A

N° Cuenta.....: 000000000

N° Cheques.....: 2

Monto.....: 0.00

Empresa.....: Empresa A

Operador.....: Usuario1 Digitalizador

Lugar y Fecha.: Quito miércoles, 18 de mayo de 2016 12:29:19

DETALLE DE CHEQUES

Numero de Cuenta	Numero de Cheque	Transito	Pin	Valor
3004878404	001051	10010060	4357	0
3004878404	001050	10010060	2531	0

Figura 101 Comprobante de Depósito

4.2.7 Reportes

El reporte de la entidad financiera permite realizar consultas en base a los siguientes parámetros: empresa, cuenta depositante, numero de depósito y numero de cheque.

Figura 102 Parámetros de Búsqueda del Reporte de Entidad Financiera

Por otra parte, el reporte de la empresa permite realizar consultas con los siguientes parámetros: número de cuenta depositante, numero de cheque y operador.

Nº Depósito	Proceso	Cuenta	Tipo Cuenta	Nº Cuenta	Fecha Depósito	Operador	Oficina	Valor Total	Nº de Cheques	PDF
10	Proceso A	Cuenta A	Corriente	0000000	10/05/2016	Usuario1 Iniciado	Sucursal A	0.00	1	PDF
11	Proceso B	Cuenta B	Corriente	0000000	10/05/2016	Usuario2 Iniciado	Sucursal A	0.00	20	PDF

Figura 103 Parámetros de Búsqueda del Reporte de Empresa

Una vez realizada la consulta se visualizan los depósitos correspondientes con sus respectivos detalles

Consulta de Depósitos

Rango de Fechas

01-05-2016 31-05-2016

Parámetros de Búsqueda

Nombre Empresa: Seleccione...

Nro. Cuenta Depositante: Ingrese el Número Cuenta Depositante.

Nro. Depósito: Ingrese el número de Depósito.

Nro. Cheque: Ingrese el Número Cheque.

Q

Export to Excel Export to PDF

Arrastre un encabezado de columna y suéltelo aquí para agrupar por dicho criterio

N° Depósito	Empresa	Tipo Cuenta	N° Cuenta	Fecha Depósito	Operador	Ciudad	Oficina	Valor Total	N° de Cheques	
1	Empresa A	Corriente	00000000	16/05/2016	Usuario1 Digitalizador	Quito	Sucursal A	0.00	4	PDF
2	Empresa A	Corriente	00000000	16/05/2016	Usuario1 Digitalizador	Quito	Sucursal A	0.00	4	PDF
3	Empresa A	Corriente	00000000	16/05/2016	Usuario1 Digitalizador	Quito	Sucursal A	0.00	4	PDF
4	Empresa A	Corriente	00000000	16/05/2016	Usuario1 Digitalizador	Quito	Sucursal A	0.00	2	PDF
5	Empresa A	Corriente	00000000	16/05/2016	Usuario1 Digitalizador	Quito	Sucursal A	0.00	3	PDF

Elementos mostrados: 1 - 5 de 5

Figura 104 Detalle del Reporte

CAPITULO V

CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

- Las pruebas realizadas con el algoritmo de reconocimiento de dígitos manuscritos muestran una alta funcionalidad cuando la imagen a tratar tiene un fondo neutro y una caligrafía legible, teniendo un factor de éxito del 80%, mientras que cuando se prueba el algoritmo en entornos reales como la digitalización de cheques el factor de éxito se reduce a un 20% debido a las características de las imágenes como marcas de agua, sellos de seguridad, etc. por lo que no puede ser usado en el sistema ya que siempre se va a tener que considerar el factor humano para la confirmación del monto ingresado en cada cheque.
- El uso de una metodología ágil como Extreme Programming permitió un desarrollo rápido y una retroalimentación constante por parte de los usuarios permitiendo obtener un aplicativo fiel a los requerimientos solicitados.
- El uso de la arquitectura REST hace posible que el cliente y el servidor sean independientes contribuyendo al desarrollo de un sistema con alto rendimiento en la transmisión de los datos, además que facilita la escalabilidad y mantenimiento del mismo.
- La autenticación basada en tokens ayudó a simplificar el proceso de desarrollo del módulo de seguridad y además hace posible el uso de roles y permisos específicos para cada usuario lo que permite que se realicen menos peticiones al servidor mejorando el desempeño del sistema.
- El desarrollo e integración de varias tecnologías para conseguir el resultado final fue posible gracias al uso de librerías y frameworks de software libre ya que estas aportan con excelente código y documentación lo que agiliza la construcción de aplicaciones.

5.2 Recomendaciones

- Para las empresas representa un costo considerable el realizar el depósito de cheques físicamente, por lo que se recomienda el uso de este tipo de sistemas para estandarizar un proceso de depósito remoto de cheques y así ahorrar tiempo y recursos.
- El formato que manejan las entidades financieras al imprimir los cheques no es el más adecuado ya que incluyen demasiadas imágenes y logotipos que dificultan el uso de algoritmos de reconocimiento de caracteres, por lo que se recomienda elaborar un estándar por parte de las entidades gubernamentales para la emisión de documentos financieros.
- Debido al tamaño de las imágenes de los cheques digitalizados que maneja el sistema se recomienda el uso algoritmos de compresión de imágenes y manejar un servicio de almacenamiento especializado en archivos grandes para reducir la carga en la transmisión de información desde y hacia la base de datos.
- En el mundo laboral el desarrollo de software por su naturaleza es acelerado y cambiante por lo que se recomienda utilizar metodologías ágiles para acelerar los procesos de desarrollo e ir a la par con los requerimientos cambiantes de los usuarios y así conseguir productos que cuenten con una mayor aceptación por parte de los clientes.
- El uso de librerías como AngularJS aportan de sencillez y rapidez en el desarrollo de software ya que consiguen mayor fluidez en el diseño de la experiencia de usuario lo cual es una ventaja tanto para los desarrolladores como para el usuario final.
- Las comunidades de software libre son un eje esencial en el desarrollo de software ya que estas aportan con su conocimiento y retroalimentación en diversos proyectos; es por esto que se recomienda su uso y sobretodo la contribución a estas para ayudar con su crecimiento y mantener una mejora continua del software.

REFERENCIAS BIBLIOGRÁFICAS

- Banco Central del Ecuador. (2013). *Resolución No. 046-2013*. Obtenido de Banco Central del Ecuador: https://contenido.bce.fin.ec/documentos/PublicacionesNotas/Catalogo/Regulaciones/Regulacion46_2013.pdf
- Banco Central del Ecuador. (Agosto de 2014). *Banco Central del Ecuador*. Obtenido de <http://www.bcrp.gob.pe/docs/Publicaciones/Seminarios/2014/inclusion-financiera/if-paez.pdf>
- Beck, K. (2000). *Extreme Programming Explained: Embrace Change*. Indianapolis: Addison Wesley.
- Christie, T. (2016). *Django REST Framework*. Obtenido de Django REST Framework: <http://www.django-rest-framework.org/>
- CTS electronics. (2015). *CTS electronics*. Obtenido de <http://ctselectronics.ctsgroup.it/>
- Django Software Foundation. (2016). *Why Django?* Obtenido de Django Project: <https://www.djangoproject.com/start/overview/>
- Domínguez, A. (1996). *Procesamiento digital de imágenes*. Obtenido de <http://www.redalyc.org/articulo.oa?id=13207206>
- Ebrahimzadeh, R., & Jampour, M. (2014). Efficient Handwritten Digit Recognition based on Histogram of Oriented Gradients and SVM. *International Journal of Computer Applications*, 10-13.
- ECMA International. (1999). *ECMA-404 The JSON Data Interchange Standard*. Obtenido de Introducing JSON: <http://www.json.org/>
- Eikvil, L. (1993). *Optical Character Recognition*. Obtenido de <https://www.nr.no/~eikvil/OCR.pdf>

- Elkstein, M. (2008). *Learn REST: A Tutorial*. Obtenido de What is REST?: <http://rest.elkstein.org/>
- Google Inc. (2016). *AngularJS*. Obtenido de AngularJS by Google: <https://angularjs.org/#>
- Harrington, P. (2012). *Machine Learning in Action*. New York: Manning Publications Co.
- Jones, M., Bradley, J., & Sakimura, N. (2015). JSON Web Token (JWT). *Internet Engineering Task Force*, 1-30.
- Krawczyk, H., Bellare, M., & Canetti, R. (1997). HMAC: Keyed-Hashing for Message Authentication. *Network Working Group*, 1-11.
- LeCun, Y., Cortes, C., & Burges, C. (2013). Obtenido de The MNIST Database of handwritten digits: <http://yann.lecun.com/exdb/mnist/>
- LibrosWeb. (2016). *El libro de Django*. Obtenido de El patrón de diseño MTV: http://librosweb.es/libro/django_1_0/capitulo_5/el_patron_de_diseno_mtv.html
- Microsoft. (2016). *Microsoft Developer Network*. Obtenido de Introducción a .NET Framework: [https://msdn.microsoft.com/es-es/library/hh425099\(v=vs.110\).aspx](https://msdn.microsoft.com/es-es/library/hh425099(v=vs.110).aspx)
- Microsoft. (2016). *Microsoft Developer Network*. Obtenido de C#: <https://msdn.microsoft.com/es-es/library/kx37x362.aspx>
- Morán, N. (2013). *Desarrollo de un sistema avanzado de asistencia a la conducción en tiempo real para la detección de peatones en entornos urbanos complejos*. Leganés: Universidad Carlos III de Madrid.
- Mordvintsev, A., & K, A. (31 de Octubre de 2014). *Understanding k-Nearest Neighbour*. Obtenido de OpenCv Foundation: http://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_ml/py_knn/py_knn_understanding/py_knn_understanding.html#knn-understanding

- Morovia Corporation. (2016). *Basic Knowledge MICR*. Obtenido de MICR E-13B: <http://www.morovia.com/manuals/micr4/ch02.php#N1016B>
- Mozilla Developer Network. (2016). *Javascript*. Obtenido de Javascript: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>
- National Instruments Corporation. (2015). *Histogram of Oriented Gradients Concepts*. Obtenido de National Instruments Corporation: http://zone.ni.com/reference/en-XX/help/372916T-01/nivisionconcepts/feature_extraction/
- OpenCV Developers Team. (2016). *About OpenCV*. Obtenido de <http://opencv.org/about.html>
- Publicaciones Vértice. (2008). *Tratamiento de la fotografía digital*. Málaga: Publicaciones Vértice S.L.
- Python Software Foundation. (2016). *Python*. Obtenido de Python: <https://wiki.python.org/moin/SpanishLanguage>
- Richardson, L., Amundsen, M., & Ruby, S. (2013). *RESTful Web APIs*. Sebastopol: O'Reilly Media, Inc.
- RSA Laboratories. (2012). RSA Cryptography Standard. *RSA*, 1-63.
- Scikit-Image. (2012). *Histogram of Oriented Gradients*. Obtenido de Scikit-Image: http://sharky93.github.io/docs/gallery/auto_examples/plot_hog.html#algorithm-overview
- scikit-learn developers. (2014). *scikit-learn*. Obtenido de scikit-learn : <http://scikit-learn.org/stable/>
- StatSoft Inc. (2016). *Support Vector Machines (SVM) Introductory Overview*. Obtenido de StatSoft Inc.: <http://www.statsoft.com/Textbook/Support-Vector-Machines#index>
- Superintendencia de Bancos y Seguros. (2006). *Normas generales para las instituciones del sistema financiero*. Obtenido de Superintendencia de

Bancos y Seguros:
http://www.sbs.gob.ec/medios/PORTALDOCS/downloads/normativa/nueva_codificacion/todos/L1_I_cap_X.pdf

Superintendencia de Bancos y Seguros del Ecuador. (2014). *Resolución No. SBS-2014-234*. Obtenido de http://www.sbs.gob.ec/medios/PORTALDOCS/downloads/normativa/2014/SBS/resol_SBS-2014-234.pdf

The PostgreSQL Global Development Group. (2016). *PostgreSQL*. Obtenido de About PostgreSQL: <https://www.postgresql.org/about/>

The SQLite Development Team. (2016). *SQLite*. Obtenido de About SQLite: <https://www.sqlite.org/about.html>

Visión artificial. (2016). Obtenido de Wikipedia: https://es.wikipedia.org/wiki/Visi%C3%B3n_artificial

Wells, D. (2013). *Extreme Programming*. Obtenido de Extreme Programming: A gentle introduction: <http://www.extremeprogramming.org/>

World Wide Web Consortium (W3C). (2014). *HTML5*. Obtenido de A vocabulary and associated APIs for HTML and XHTML: <https://www.w3.org/TR/html5/>

World Wide Web Consortium (W3C). (2015). *CSS*. Obtenido de CSS Snapshot 2015: <https://www.w3.org/TR/CSS/>