UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

ContractMe

Presentado por David Martínez Bahillo
en Universidad de Burgos — 11 de febrero de 2024
Tutor: Sandra Rodriguez Arribas

UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

D. nombre tutor, profesor del departamento de nombre departamento, área de nombre área.

Expone:

Que el alumno D. David Martínez Bahillo, con DNI 71566321G, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado «ContractMe».

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 11 de febrero de 2024

Vº. Bº. del Tutor:	Vº. Bº. del co-tutor:
D. nombre tutor	D. nombre co-tutor

Resumen

Este proyecto se enfoca en la creación de una solución innovadora para la contratación laboral, aprovechando la tecnología blockchain y los contratos inteligentes. Se busca agilizar el proceso de contratación en sectores como la agricultura y servicios domésticos mediante la automatización de la gestión de contratos desde su inició hasta su final. La solución propuesta incluye funcionalidades avanzadas como el seguimiento del tiempo de trabajo y el pago automático, integrando tecnologías de localización GPS para establecer zonas de trabajo específicas. El proyecto incorpora métodos de autenticación biométrica y escaneo de códigos QR, apoyándose en soluciones de identidad descentralizadas para proteger la privacidad de los usuarios.

El objetivo final es proporcionar una herramienta que promueva prácticas de empleo transparentes y legales, optimizando los procesos de contratación y verificación para empleadores y trabajadores garantizando la seguridad de todos los tramites.

Descriptores

Blockchain, contratos inteligentes, localización GPS, autenticación biométrica de teléfonos inteligentes, encriptación de datos, desarrollo de aplicación móvil, procesamiento de pagos.

Abstract

This project focuses on creating an innovative solution for employment hiring, leveraging blockchain technology and smart contracts. It aims to streamline the hiring process in sectors such as agriculture and domestic services by automating contract management from start to finish. The proposed solution includes advanced features such as work time tracking and automatic payment, integrating GPS technology to establish specific work zones. The project incorporates biometric authentication methods and QR code scanning, relying on decentralized identity solutions to protect user privacy.

The final goal is to provide a tool that promotes transparent and legal employment practices, optimizing hiring and verification processes for employers and workers, ensuring the security of all procedures.

Keywords

Blockchain, smart contracts, GPS tracking, smartphone biometric authentication, data encryption, mobile app development, payment processing.

Índice general

Índice de figuras

Índice de tablas

1. Introducción

Descripción del contenido del trabajo y del estructura de la memoria y del resto de materiales entregados.

2. Objetivos del proyecto

Este apartado explica de forma precisa y concisa cuales son los objetivos que se persiguen con la realización del proyecto. Se puede distinguir entre los objetivos marcados por los requisitos del software a construir y los objetivos de carácter técnico que plantea a la hora de llevar a la práctica el proyecto.

3. Conceptos teóricos

En aquellos proyectos que necesiten para su comprensión y desarrollo de unos conceptos teóricos de una determinada materia o de un determinado dominio de conocimiento, debe existir un apartado que sintetice dichos conceptos.

Algunos conceptos teóricos de LaTeX [1].

3.1. Secciones

Las secciones se incluyen con el comando section.

Subsecciones

Además de secciones tenemos subsecciones.

Subsubsecciones

Y subsecciones.

3.2. Referencias

Las referencias se incluyen en el texto usando cite [**?**]. Para citar webs, artículos o libros [**?**], si se desean citar más de uno en el mismo lugar [**?**, **?**].

[1] Créditos a los proyectos de Álvaro López Cantero: Configurador de Presupuestos y Roberto Izquierdo Amo: PLQuiz

3.3. Imágenes

Se pueden incluir imágenes con los comandos standard de LaTeX, pero esta plantilla dispone de comandos propios como por ejemplo el siguiente:

Figura 3.1: Autómata para una expresión vacía

3.4. Listas de items

Existen tres posibilidades:

- primer item.
- segundo item.

1. primer item.
2. segundo item.

Primer item más información sobre el primer item.

Segundo item más información sobre el segundo item.

-

Herramientas	App AngularJS	API REST	BD	Memoria
HTML5	X			
CSS3	X			
BOOTSTRAP	X			
JavaScript	X			
AngularJS	X			
Bower	X			
PHP		X		
Karma + Jasmine	X			
Slim framework		X		
Idiorm		X		
Composer		X		
JSON	X	X		
PhpStorm	X	X		
MySQL			X	
PhpMyAdmin			X	
Git + BitBucket	X	X	X	X
MikTEX				X
TEXMaker				X
Astah				X
Balsamiq Mockups	X			
VersionOne	X	X	X	X

Tabla 3.1: Herramientas y tecnologías utilizadas en cada parte del proyecto

3.5. Tablas

Igualmente se pueden usar los comandos específicos de LATEXo bien usar alguno de los comandos de la plantilla.

4. Técnicas y herramientas

4.1. Herramientas

Se muestra a continuación las herramientas usadas a lo largo del desarrollo del proyecto.

React Native

La elección del framework adecuado juega un papel crucial en el éxito de un proyecto. Dicha elección se complica aún más cuando se carece de experiencia previa en el desarrollo de aplicaciones móviles. Dentro del gran abanico de posibilidades React Native y Flutter emergen como grandes líderes en el sector gracias a sus grandes comunidades y la abundancia de recursos en línea. Por otro lado se pueden descartar directamente frameworks como Swift debido a su exclusividad para aplicaciones para iOS debido a que el objetivo de este proyecto es desarrollar una aplicación para Android.

React Native se presenta como mi elección favorita, este es un framework de código abierto creado por Facebook orientado a la creación de aplicaciones nativas tanto en iOS como en Android. React Native esta basado en JavaScript y React, una biblioteca de JavaScript destinada a la creación de interfaces de usuario. Fue lanzado en 2015 con el propósito de superar las limitaciones del desarrollo de aplicaciones móviles basado solo en HTML5, en las que simplemente se adapta aplicaciones web a un entorno móvil. React Native utiliza componentes nativos en lugar de WebViews para la interfaz de usuario, esto conlleva a que las aplicaciones se sientan y actúen como en una aplicación nativa. Uno de los elementos mas importantes de este framework es el React Native Bridge", el cual facilita la comunicación entre el código JavaScript y los elementos nativos del dispositivos. El puente

maneja de forma paralela dos flujos de trabajo, uno ejecuta la lógica de la aplicación en JavaScript y otro gestiona las operaciones de la interfaz de usuario nativa. Esto permite que las aplicaciones en React Native accedan a las características del dispositivo como la cámara o la ubicación manteniendo una experiencia fluida para el usuario.

Mi preferencia de este framework sobre el resto radica en mi familiaridad previa con la programación web y JavaScript, lo cual reducirá la curva de aprendizaje en la transición hacia React Native en contraste con otros frameworks que podrían requerir el aprendizaje de nuevos lenguajes de programación o paradigmas de programación. Por otro lado, uno de los grandes motivos de este framework es su gran popularidad, el cual se traduce en un gran riqueza de recursos disponibles, como bibliotecas, videotutoriales o foros, crucial a la hora de enfrentar los desafíos que puedan surgir. Finalmente aunque para este proyecto no sea un requerimiento hay que tener en cuenta la gran versatilidad de React Native, el cual permite la creación de aplicaciones nativas tanto en Android como en iOS a partir de un único código base. Optimizando así el proceso de desarrollo permitiendo en un futuro poder dar cobertura a un mercado más amplio sin esfuerzos duplicados.

Como había nombrado anteriormente Flutter es otro framework que destaca sobre el resto y ofrece ciertas ventajas notables sobre Reac Native, como un rendimiento superior gracias al uso de su motor de renderizado propio y la gran capacidad de personalización de la interfaz de usuario. Aunque Flutter pudiera ser superior en algunos aspectos técnicos sigo decantándome por React Native debido a su gran comunidad y la familiaridad que tengo con las tecnologías web, priorizando así un aprendizaje más sencillo frente a posibles mejoras en el rendimiento. Por otro lado existen frameworks Ionic y Xamarin que los he descartado debido a sus limitaciones en términos de acceso a funciones nativas o una comunidad bastante inferior comparado con React Native o Flutter. Kotlin era un opción con gran potencial para el desarrollo nativo de aplicaciones Android pero la descarte rápidamente debido a que disponía de una curva de aprendizaje más pronunciada que su competencia y iba a representar un obstáculo significativo en cuanto al tiempo de desarrollo sin garantizar beneficios proporcionales a dicho esfuerzo.

Respecto a Angular, si bien este framework ofrece un ecosistema robusto para el desarrollo de aplicaciones web dinámicas y complejas utilizando TypeScript, es importante mencionar que también es posible desarrollar aplicaciones móviles con Angular, ofreciendo una experiencia cercana a la nativa, aunque a través de un enfoque diferente al de las aplicaciones nativas

desarrolladas con React Native. Sin embargo, para el desarrollo específico de aplicaciones móviles nativas, React Native encaja mejor con los objetivos del proyecto. Angular no deja de ser una elección excelente, pero debido a la preferencia del proyecto de enfocarse en una aplicación móvil sin la necesidad de dar soporte de navegador, Angular puede no ser la opción más adecuada para el proyecto. Por lo tanto he descartado Angular buscando una solución más enfocada y optimizada para el desarrollo móvil, aprovechando las capacidades nativas de los dispositivos móviles.

Expo

Expo es un conjunto de herramientas, librerías y servicios para desarrollar aplicaciones nativas utilizando en Javascript y React Native. Proporciona un entorno de trabajo rico en funcionalidades que evita las complicaciones asociadas con la configuración nativa. Expo ha sido fundamental en el proyecto para desarrollar la interfaz de usuario permitiendo probar y visualizar cambios en tiempo real en diferentes plataformas. Una de las funcionalidades significativas de Expo ha sido su capacidad para facilitar la ejecución y testeo de la aplicación directamente en dispositivos móviles personales sin necesidad de emuladores. Esto se logra únicamente descargando la aplicación ExpoGo en el teléfono móvil y escaneando un código QR generado por el entorno de desarrollo Expo. Expo tiene un papel fundamental en las fases de prueba y depuración, permitiendo probar la aplicación en un entorno real y ajustar la interfaz la interfaz de usuario con un ciclo de retroalimentación casi instantáneo.

Solidity

Solidity es un lenguaje de programación orientado a objetos de alto nivel diseñado para escribir Smart Contracts que se ejecutan en la Ethereum Virtual Machine (EVM). Solidity consta de una sintaxis similar a lenguajes como JavaScript, C++ y Python, lo que facilita su aprendizaje. Es un lenguaje de tipado estático, lo que significa que el tipo de cada variable se define y no cambia durante al ejecución del contrato. Una de las potencialidades más destacadas de Solidity es su soporte para la herencia, una característica común en la programación orientada a objetos. Esto permite que los Smart Contracts hereden propiedades y comportamientos de otros contratos, beneficiándose de la reutilización de código y la organización de la lógica del mismo. En mi caso de gran utilidad para utilizar las funcionalidades del estándar ERC721 proporcionado por OpenZeppelin, que ofrece un conjunto de contratos inteligentes auditados y aprobados por su comunidad.

Remix

Remix es un entorno de desarrollo integrado (IDE) diseñado para el desarrollo de Smart Contracts escritos en Solidity. Proporciona una interfaz accesible y fácil de usar para escribir, compilar, probar y desplegar Smart Contracts directamente desde el navegador, sin la necesidad de instalar software adicional. Remix también ofrece funcionalidades avanzadas como la compilación en tiempo real, el despliegue de contratos en diversas redes de prueba (testnets) y la interacción con SmartContracts ya desplegados. Una de las características más útiles de este IDE es su análisis estático del código, pudiendo así identificar posibles errores de programación o vulnerabilidades de seguridad. Crucial para el desarrollo de Smart Contracts donde los errores pueden suponer consecuencias financieras significativas. Además, Remix se integra con herramientas y plugins adicionales, ofreciendo un entorna mas rico y extenso permitiendo conectarse con herramientas y servicios como MetaMask, Truffle y Ganache ampliamente utilizados en mi proyecto. Remix ha tenido una gran protagonismo en el desarrollo de mi proyecto permitiéndome iterar rápidamente a través de diferentes versiones de contratos inteligentes. Pudiendo ejecutar pruebas unitarias con diversas redes Ethereum como Rinkeby y Goerli, indispensable para validar la lógica del Smart Contract antes de su despliegue final. Dándome una gran capacidad de testeo en un entorno controlado pero realista el cual ha sido crucial para la corrección de errores y la optimización del uso de gas, asegurando así la eficiencia y seguridad de los contratos inteligentes desarrollados.

Truffle

Tras una fase inicial de desarrollo de los Smart Contracts usando Remix, la transición al uso de Truffle ha marcado un punto de inflexión en la complejidad de mi proyecto. Truffle consiste en una suite de desarrollo avanzada para Ethereum, ofreciendo un conjunto de herramientas diseñadas para facilitar el desarrollo, gestión y despliegue de Smart Contracts. Truffle proporciona un entorno de desarrollo estructurado generando una jerarquía de carpetas para favorecer la implementación de proyectos blockchain complejos. Ahorra mucho tiempo con su sistema de migraciones y scripts de despliegue el cual automatiza y simplifica el proceso de lanzamiento de contratos. Aunque Remix es una excelente herramienta, Truffle eleva la posibilidad de hacer pruebas unitarias a otro nivel con un marco de prueba mucho mas sofisticado, permitiendo ejecutar test en Solidity o JavaScript. Finalmente, uno de las mayores ventajas de usar Truffle en el desarrollo

de aplicaciones descentralizadas es en cómo simplifica el proceso de unir el trabajo que se realiza en el backend con el frontend. Cuenta con herramientas que ayudan a conectar Smart Contracts con aplicaciones móviles haciendo que la conexión sea mas directa y menos propensa a errores.

OpenZeppelin

OpenZeppelin es una biblioteca para el desarrollo seguro de Smart Contracts en Ethereum. Ofrece implementaciones auditadas y probadas para minimizar riesgos de seguridad. Su uso ha sido crucial para el desarrollo de mi proyecto, usando el estándar ERC721 para la representación de los contratos como tokens no fungibles (NFTs) asegurando que la aplicación cumpla con los estándares de seguridad.

Ganache

Ganache funciona como un nodo de Ethereum personal, permitiendo a los desarrolladores simular un entorno de blockchain que opera localmente ofreciendo un espacio seguro y controlado para experimentar sin ningún costo real ni tiempos de espera con las redes públicas de Ethereum. Proporciona diez cuentas cargadas con 1000 ETH cada una y con su clave pública y privada correspondiente. Ganache se puede usar tanto en la línea de comandos como en en la aplicación de escritorio y proporciona una vista de las transacciones, bloques y estado de la red, mostrando el feedback en tiempo real, vital para realizar una iteración rápida. Ganache se integra perfectamente con Truffle, siendo Ganache el entorno de desarrollo local predeterminado de Truffle.

MetaMask

MetaMask es una extensión de navegador y una aplicación móvil que permite a los usuarios interactuar con la blockchain de manera segura y sencilla. Actúa como puente entre los navegadores web y la blockchain. Su funcionamiento es análogo a una cartera digital, permitiendo a los usuarios almacenar sus cuentas de Ethereum. Al conectarse a dApps, los usuarios pueden usar sus cuentas de MetaMask para autenticarse eliminando la necesidad de copiar claves privadas manualmente. Por otro lado, aplica una capa adicional de seguridad ya que esta herramienta encripta la información del usuario y almacena las claves privadas directamente en el dispositivo del usuario. Esto asegura que solo el usuario tenga acceso a sus fondos y datos. La integración de MetaMask en mi proyecto no solo mejora la

experiencia del usuario final, sino que también agiliza el desarrollo de mi proyecto al proporcionar una manera fácil y segura de acceder a los activos de los usuarios.

Node.js

Node.js es un entorno de ejecución en JavaScript, tradicionalmente un lenguaje de programación del lado del cliente, para desarrollar aplicaciones del lado del servidor. Es conocido por su capacidad para manejar operaciones asíncronas y por su escalabilidad siendo de gran popularidad en el desarrollo de aplicaciones web modernas. Una de las ventajas de de Node.js es su ecosistema de paquetes gestionados por npm (Node Package Manager), que proporciona acceso a miles de librerías. En el proyecto se ha usado de manera frecuente, más allá de ser un requisito para utilizar Truffle, a través del uso de npm install se ha usado "npm install"para descargar librerías y paquetes necesarios para el desarrollo del frontend y backend. Bien es así, que aunque React Native sea el framework principal del desarrollo del frontend, la gestión de sus dependencias, librerías adicionales se realiza mediante npm, el cual opera sobre Node.js. A su vez, por ejemplo el uso de la biblioteca Web3.js, una de las mas importantes del proyecto ya que es fundamental para interactuar con la blockchain, se ha instalado y gestionado usando npm. Por tanto, Node.js ha sido una pieza crucial en el desarrollo del proyecto, simplificando la configuración del proyecto.

5. Aspectos relevantes del desarrollo del proyecto

Este apartado pretende recoger los aspectos más interesantes del desarrollo del proyecto, comentados por los autores del mismo. Debe incluir desde la exposición del ciclo de vida utilizado, hasta los detalles de mayor relevancia de las fases de análisis, diseño e implementación. Se busca que no sea una mera operación de copiar y pegar diagramas y extractos del código fuente, sino que realmente se justifiquen los caminos de solución que se han tomado, especialmente aquellos que no sean triviales. Puede ser el lugar más adecuado para documentar los aspectos más interesantes del diseño y de la implementación, con un mayor hincapié en aspectos tales como el tipo de arquitectura elegido, los índices de las tablas de la base de datos, normalización y desnormalización, distribución en ficheros3, reglas de negocio dentro de las bases de datos (EDVHV GH GDWRV DFWLYDV), aspectos de desarrollo relacionados con el WWW... Este apartado, debe convertirse en el resumen de la experiencia práctica del proyecto, y por sí mismo justifica que la memoria se convierta en un documento útil, fuente de referencia para los autores, los tutores y futuros alumnos.

6. Trabajos relacionados

Este apartado sería parecido a un estado del arte de una tesis o tesina. En un trabajo final grado no parece obligada su presencia, aunque se puede dejar a juicio del tutor el incluir un pequeño resumen comentado de los trabajos y proyectos ya realizados en el campo del proyecto en curso.

7. Conclusiones y Líneas de trabajo futuras

Todo proyecto debe incluir las conclusiones que se derivan de su desarrollo. Éstas pueden ser de diferente índole, dependiendo de la tipología del proyecto, pero normalmente van a estar presentes un conjunto de conclusiones relacionadas con los resultados del proyecto y un conjunto de conclusiones técnicas. Además, resulta muy útil realizar un informe crítico indicando cómo se puede mejorar el proyecto, o cómo se puede continuar trabajando en la línea del proyecto realizado.