Campus Tecnológico Local San Carlos

Escuela de Ingeniería en Computación

"Plataforma web para la gestión de información en iniciativas sobre energías renovables"

Informe final de graduación para optar por el grado de Bachiller en la Escuela de Ingeniería en Computación

Andrés de Jesús García Salas

Mayo, Santa Clara, Costa Rica, 2019

Resumen ejecutivo

A través de los últimos años, las tecnologías de información promueven mejoras en una sin igual cantidad de campos, aumentando la precisión de labores, simplificando procesos e inclusive transformando prácticas laborales. En el caso de las instituciones que cuentan con sistemas de producción energética basados en energías renovables, se presenta la necesidad de administrar y gestionar los millones de datos generados cada segundo por dichos sistemas, creando alguna utilidad para estos en la inteligencia de negocios requerida por este proceso. Todo esto con el propósito de facilitar el proceso administrativo sobre este sistema.

Por lo anterior, se realizó un trabajo de dieciséis semanas, donde se implementaron múltiples procesos propios de un desarrollo de software acorde al estándar ingenieril, orientados en la construcción de una solución para este sector productivo. Estos procesos incluyen la propuesta de la solución, definición los requerimientos, diseño y desarrollo de la plataforma de software. Con ello se logró obtener una solución adaptada en conjunto con una solución anteriormente desarrollada, las cuales en conjunto permiten facilitar la manipulación de estos datos masivos.

Palabras claves— *Ciencia de los datos, Datos masivos, Modelo de predicción, Paneles Fotovoltaicos, R, RESTFul API*

Abstract

Over the last few years, information technologies have promoted improvements in an unrivaled number of fields, increasing the precision of tasks, simplifying processes and even transforming work practices. In the case of institutions that have energy production systems based on renewable energy, there is a need to manage the millions of data generated every second by these systems, creating some usefulness for them in the business intelligence required by this process. All this with the purpose of facilitating the administrative process on this system.

Due to the above, a sixteen-week work was carried out, where multiple processes of software development according to the engineering standard were implemented, oriented towards the construction of a solution for this productive sector. These processes include the proposal of the solution, the definition of the requirements, design, and development of the software platform. With this, it was possible to obtain a solution adapted in conjunction with a previously developed solution, which together makes it easier to manipulate this massive data.

Keywords— *Data Science, Big Data, Prediction model, Photovoltaic panels, R, RESTFul API*

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

1.1. Antecedentes

En el Instituto Tecnológico de Costa Rica, actualmente existe un proyecto que pretende gestionar los datos que se generan en los paneles solares de los Campus Tecnológicos San Carlos y Cartago. Se requiere de una eficiente gestión de todos los datos masivos que se generan en un tiempo determinado, ya que, también se deben de tomar en cuenta los datos históricos de meses o años atrás. Se están tomando registros generados desde el año 2017 a la actualidad, pero esos datos continúan creciendo día con día, así que de alguna manera se debe prever ese tipo de crecimiento. También, se debe tener en cuenta, que en un futuro la cantidad de paneles solares en ambos campus pueden aumentar, asi como la cantidad de registros que se generan.

Anteriormente, se contaba con una plataforma para la visualización de los datos, sin embargo, poco a poco se determinó que el rendimiento de esta herramienta tendía a reducirse considerablemente cuando se trataba una gran cantidad de datos o de registros. Además, el sistema actual solo cuenta con un número reducido de módulos ya predeterminados, y para efectos de administración, se requieren más funcionalidades que los existentes, además se requiere crear todo un sistema que pueda soportar todos los datos que se analizan, sin afectar el rendimiento de este. Para los encargados del monitoreo de los paneles solares es de vital importancia contar con una forma eficaz de poder acceder a los informes o gráficos.

1.2. Descripción de la empresa

El Instituto Tecnológico de Costa Rica es una universidad estatal, y fue fundada el 10 de junio del 1971. En el 2010 se le hizo un cambio de imagen, y a nivel de comunicación se pasó a llamar Tecnológico de Costa Rica (TEC), de manera promocional. En la ciudad de Cartago se encuentra el Campus Central, y cuenta con otros campus por todo el país, incluyendo el Campus en Santa Clara de San Carlos (provincia de Alajuela) campus en el cual se estará realizando el presente proyecto.

El campus de San Carlos está en la región tropical húmeda, su área de construcción es de aproximadamente 35.500 metros cuadrados, donde en sus instalaciones se incluye con un complejo académico-administrativo, aulas, biblioteca, laboratorios, oficinas, residencias para estudiantes y profesores, un amplio comedor, lavandería, edificios para investigación, áreas recreativas.

Entre los departamentos con los que se cuenta en la institución, existe el de Admisión y Registro, donde se gestiona toda la información con respecto a los estudiantes que desean ingresar a la institución y asimismo todo el proceso que deben completar, también llevan el control de estudiantes activos o inactivos en la institución. Existen departamentos administrativos en la parte financiera, legal, vida estudiantil, recursos humanos, servicios generales y demás. De igual forma, el campus además cuenta con seis carreras a disposición de la población estudiantil Administración de Empresas, Gestión en Turismo Rural Sostenible, Ingeniería en Computación, Ingeniería en Electrónica, Ingeniería en Agronomía y Ingeniería en Producción Industrial.

Se cuenta con departamentos de enfermería, donde hay profesionales encargados de valorar la salud a estudiantes y funcionarios. También se cuenta con un espacio para odontología, y de igual manera se atienden a estudiantes y a los funcionarios de la institución que requieran atención.

Además, el campus actual posee uno de los dos Centros de Transferencia Tecnológico y Educación Continua (CTEC) que tiene el TEC, este Centro es importante para contribuir con el desarrollo socioeconómico y de tecnología en la región norte. Ese mismo edificio, se utiliza para para graduaciones, presentaciones artísticas, entre otras actividades.
Sobre el eje de conocimiento bajo el que se desarrolla este proyecto (energías renovables), la institución hace especial enfoque sobre su interés en dicha materia, tomando en cuenta ya diversos proyectos existentes como lo son “la investigación sobre plasmas como futura fuente

de energía" y "Investigación sobre el uso de microalgas para la generación de biodiesel".

En conclusión, el concurrente proyecto será desarrollado en el edificio 'PROTEC', el cual es un sitio dedicado exclusivamente para que las carreras de electrónica, producción y computación desarrollen proyectos de investigación. Este proyecto de práctica se encuentra fuertemente relacionado al actual proyecto de investigación de la Escuela de Computación y Comunidad de Ciencia de los Datos, el cual se conoce como ICDER.

1.2.1. Organigrama de la empresa

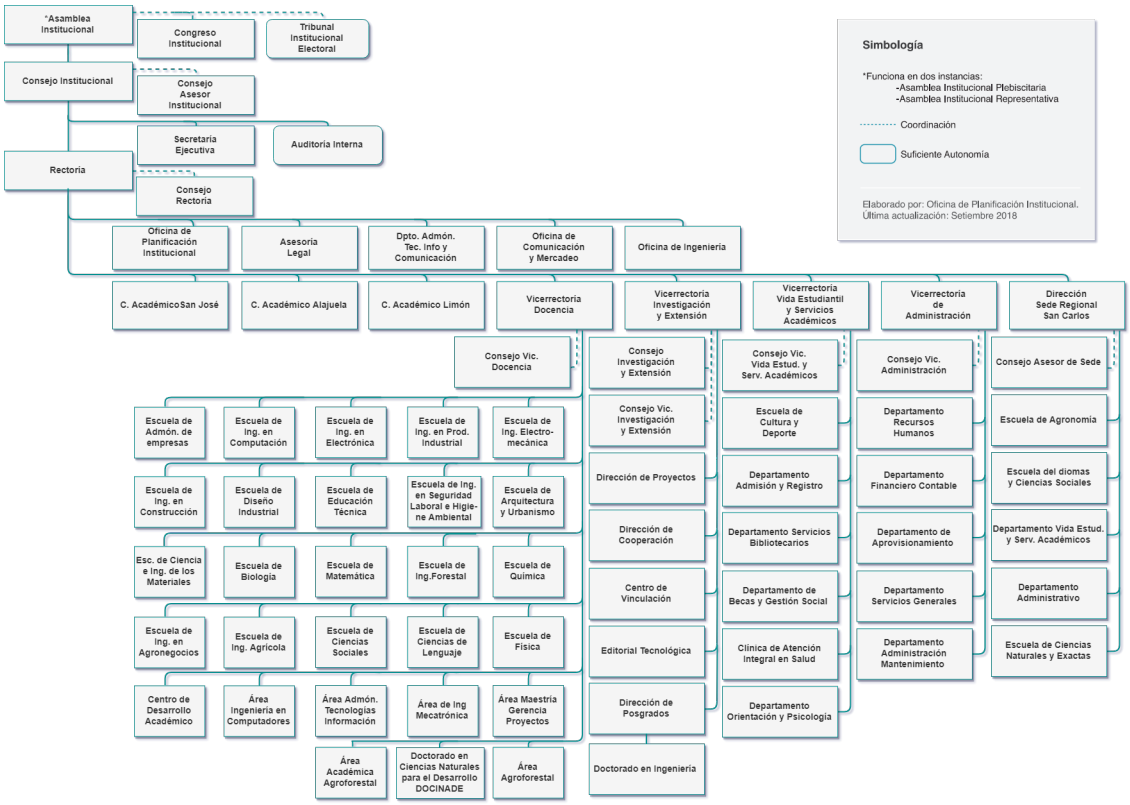


Figura 1.1: Organigrama de la Empresa

1.3. Problema

Por subsiguiente se muestra el análisis exhaustivo realizado en marco a la situación a analizar, esto permite esclarecer el contexto del problema, los procesos que intervienen en él, las especificaciones que requiere la solución, entre muchos otros detalles. Este conjunto de acciones permite obtener información que es de suma utilidad al momento de proponer

y desarrollar una solución. A continuación, se detalla la serie de acciones que permitieron comprender más a fondo el tema.

1.3.1. Contexto del problema

En el ITCR se ha promovido el uso de fuentes de energía renovables y limpias, de manera que, se han impulsado iniciativas que ayuden a crear alternativas tecnológicas amigables con el medio ambiente. Actualmente, la institución cuenta con una serie de paneles solares los cuales son aprovechados inyectando la energía producida por los mismos a la red eléctrica de la zona (Coopelesca), la idea principal de los mismos es llegar a producir para reducir los egresos del campus. Sin embargo, a pesar de que actualmente ya se les ha dado una utilidad, aún se desaprovechan en gran parte otras utilidades de esta herramienta, como los datos que se encuentran generando constantemente.

1.3.2. Descripción del problema

Debido a lo anteriormente mencionado, los campus tecnológicos de San Carlos y Cartago del Instituto Tecnológico de Costa Rica presentan ciertos inconvenientes al momento de aprovechar los datos generados por sus paneles solares. Hasta la fecha los datos provenientes de Rectoría en Cartago están siendo almacenados en un servidor local del Campus de San Carlos, a su vez se mantiene una réplica de estos datos en una máquina virtual en Amazon Web Services (AWS). Por otra parte, los datos generados por los inversores de paneles fotovoltaicos del Campus de Santa Clara de San Carlos aún se encuentran en fase de pruebas debido a que el medio para la recolecta de dichos datos se encuentra en desarrollo.

Actualmente se encuentra en desarrollo una infraestructura para la gestión de estos datos; esto incluye análisis de modelos estadísticos, modelos predictivos y algunas otras funcionalidades a futuro, no obstante, los modelos actualmente no son accesibles desde la plataforma. A causa de distintas eventualidades, como el set de tecnologías usadas, la introducción de los desarrolladores a dichas tecnologías, entre otras. Su integración no fue posible de manera inmediata por lo cual, si los usuarios de la infraestructura desean acceder a los modelos existentes no lo pueden llevar a cabo a través de la misma. Dicha interacción entre la infraestructura y los modelos no se da de manera automatizada, en el sentido de que las predicciones generadas a partir de un set de datos enviados al modelo no pueden ser renderizadas en la infraestructura de datos.

Actualmente existe un prototipo funcional, sin embargo, este prototipo solamente realiza consultas directas a la base de datos Postgresql, por lo cual se están desaprovechando distintas estadísticas y gráficas generadas por los modelos, y al no encontrarse dichos datos en el sistema se podrían ver afectada la toma de decisiones basadas en la infraestructura de datos.

1.4. Objetivos

1.4.1. Objetivo general

Contribuir con la inteligencia de negocios del proyecto ICDER (Infraestructura en ciencia de los datos para energías renovables del ITCR), automatizando el proceso de alimentación de los modelos predictivos sobre la Infraestructura de datos.

1.4.2. Objetivos específicos

a. Identificar los requerimientos para la comunicación entre los modelos y ICDER.

b. Definir el(los) módulo(s) para reportes provenientes de los modelos en el ICDER.

c. Desarrollar el(los) módulo(s) para reportes provenientes de los modelos en el ICDER.

1.5. Justificación

El presente proyecto se enfoca en la elaboración de uno o más módulos para la plataforma de gestión de datos de paneles fotovoltaicos. Debido a que en la actualidad dicha plataforma se encuentra aún en desarrollo, actualmente la plataforma cuenta con una sección de reportes, sin embargo, la misma solo trabaja consultas directas a la base de datos. Para crear una plataforma más robusta y efectiva se requieren reportes provenientes de algoritmos predictivos desarrollados en lenguaje R.

Esta iniciativa ayudará a administrativos o encargados de la gestión de paneles solares, poder visualizar datos mucho más efectivos y precisos. De esta manera, la toma de decisiones importantes respecto a la implementación de paneles solares en Campus Tecnológico se vería afectada al poder visualizar datos que responden a la factibilidad sobre alimentar el campus en gran parte con dicha energía. Cabe recalcar que esto beneficiaria a los gastos económicos

generados mensualmente por el campus respecto al consumo eléctrico, de modo que se reducirían en gran parte los egresos, los cuales podrían llegar a ser utilizados para otros fines por parte de la institución.

A gran escala, se pretende que con esta iniciativa se vea reflejada la viabilidad de los paneles no solamente dentro de nuestra institución, sino que se pueda pensar en la implementación del sistema en diferentes lugares del país. Esto debido a que la plataforma cuenta con la ventaja de poder ser utilizada en cualquier lugar donde se tengan paneles y se quiera gestionar los datos de los mismos. Contribuyendo con un país que poco a poco va tomando fuerza en implementación de energía renovable y de bajo impacto ambiental.

Capítulo 2

Revisión de literatura

2.1. Marco teórico

2.1.1. Ciencia de los datos (Data Science)

Según Liu (2015) la ciencia de los datos "*...es un campo interdisciplinario que trata sobre extraer conocimiento de procedimientos y sistemas o ideas de grandes volúmenes de información ya sea estructurada como no estructurada, además esto involucra algunos campos de análisis de datos como minería de datos y análisis predicativos.*"p.06. Por lo cual, se puede interpretar de lo anteriormente mencionado una fuerte relación entre la estadística, el análisis de datos y la consecutiva necesidad de comprender y analizar los datos. La implementación de este concepto requiere fuentes constantes de datos, poder computacional compartido, capacidad de almacenamiento considerable y un sistema de redundancia de datos ante cualquier eventualidad.

2.1.2. Big data

Como bien afirman Haider y Gandomi (2014), Big data es un concepto que usualmente hace mención a grandes volúmenes de datos, los cuales son producidos de manera masiva y con una complejidad y variabilidad alta. Es por ello que estos datos requieren técnicas y tecnologías avanzadas capaces de proveer captura, almacenamiento, distribución, gestión y análisis de la información.

2.1.3. Paneles Fotovoltaicos

Los paneles fotovoltaicos son dispositivos que son capaces de captar la energía de la radiación del sol para su aprovechamiento. De manera que a partir de la energía solar que se captura, se puede generar electricidad en aplicaciones domésticas o comerciales. En realidad se les llama paneles "solares"porque en la mayoría de ocasiones, la fuente más poderosa de luz que está disponible es la del sol. En Celsia (2018) se menciona que cuando hay luz, una célula solar se debe de comportar como una batería. La luz del sol que se recibe, separa los electrones de manera que se forma una capa de carga positiva y otra de carga negativa en la célula solar, de manera que genera una corriente eléctrica.

2.1.4. Modelo de Predicción

Tomando como punto de partida los tres componentes claramente delimitados de la palabra predicción, los cuales son:

1. '**pre-**': El cual significa 'antes'.
2. '**decir**'.
3. '**-ción**': Que viene a indicar 'acción y efecto'.

Por lo cual, esta claro que la expresión nos viene a indicar **anticipación de aquello que, supuestamente, va a suceder**. Como sugiere Chatti y cols. (2013), una predicción "tiene como objetivo el desarrollo de un modelo que intente predecir el conocimiento..., basado en una serie realizada de actividades.", esto en el ámbito computacional puede ser visto como el desarrollo de un modelo computacional entrenado, capaz de predecir bajo cierto intervalo de confianza, un conjunto de acciones a futuro.

2.1.5. PostgreSQL

Según afirma Momjian (2002), PostgreSQL es el más avanzado servidor de base de datos libre, claro el lo aseguro en el año 2002. Hasta la fecha sigue siendo una de las bases de datos más usadas en conjunto con MySQL y SQLServer, no esta demás aclarar que es una base de datos fuertemente relacional.

2.1.6. R (Lenguaje de Programación)

De acuerdo con Hornik (2018), R es un sistema para estadística y gráficas computacionales. Básicamente es un lenguaje interpretado el cual posee las características generales de cualquier lenguaje común, sin embargo su fuerte es la gran cantidad de librerías para cálculos estadísticos y su manejo de los datos, dándole la capacidad de realizar cálculos estadísticos sobre grandes cantidades de información.

2.1.7. Angular

Es un framework de desarrollo en JavaScript que fue creado por Google (*What is angular?* (2018)). La finalidad de la creación de esta nueva tecnología es para facilitar el desarrollo de aplicaciones web tipo SPA (Single Page Application), además de brindar las herramientas para trabajar con elementos web de una manera más óptima y sencilla. En este contexto, servirá como la parte de frontend de la aplicación, esto para poder separar funciones y sea mucho más eficaz y entendible. Además brinda la facilidad de crear una página en el cual la navegación entre secciones, se realiza dinámicamente y asincrónicamente, realizando llamadas al servidor sin tener que refrescar la página continuamente.

2.2. Trabajos relacionados

Con el paso de los años el uso de paneles fotovoltaicos como medio para generación de energía se ha vuelto bastante popular. Con anterioridad, se han propuesto distintos tipos de sistemas por ejemplo para verificar el estado de salud de los paneles y otros relacionados con los datos generados de los mismos. Actualmente se cuenta con un sistema en vías de desarrollo cuyo propósito es la recolección de datos, creación y entrenamientos de modelos para generar información de utilidad con fines de dar soporte a la toma de decisiones alrededor de un sistema de paneles fotovoltaicos.

En esta sección se desarrollan las bases teóricas basadas en trabajos relacionados al campo enfocado, las cuales sustentan la creación del sistema capaz de generar reportes a partir de los datos de los paneles.

Debido al crecimiento exponencial en la generación de datos durante los últimos años por distintos sistemas, nace la necesidad de crear una plataforma/sistema que sea capaz de consumir todo estos datos y retornar información útil para la toma de decisiones. A

este tipo de sistemas se les conoce como DSS (Decision support system), por ello Zhuang (2018) proponen la creación de un sistema basado en la arquitectura MEAN (Mongo-Express-Angular-Node) y el lenguaje estadístico R. Ellos lo llaman 'MEAN+R', un framework que facilita tomar todo lo anteriormente especificado he implementarlo. De igual manera, Park y cols. (2015) presentan una plataforma colaborativa de análisis de datos masivos para Big Data como un servicio. Los desarrolladores pueden colaborar entre sí en la plataforma compartiendo datos, algoritmos eficientes y servicios. En este último documento se describe la plataforma de análisis de datos, soportando de manera eficiente y efectiva el desarrollo de algoritmos, servicios de análisis, etc.

De manera similar Bartsch H y AM (2014), proponen el desarrollo de un portal interactivo para la exploración, visualización y testing hipotético de datos. Ahora, esta propuesta tomo conjuntos de datos provenientes de estudios de investigaciones clínicas, estos datos por naturaleza presentan grandes desafíos a la hora de intercambio de datos, exploración y visualización. Dicho sistema proporciona un enfoque interactivo para el análisis estadístico, extracción de datos y pruebas de hipótesis a lo largo de la vida útil de un estudio. Por ende la plataforma cumple con eliminar las barreras para la investigación y apoya la exploración de datos generados en curso.

Por último Liu y cols. (2017) presentan una plataforma web, la cual toma datos heterogéneos provenientes de múltiples fuentes con el objetivo de crear un gráfico de conocimiento de atención médica. Fuentes como datos clínicos de hospitales, datos de comportamiento y datos de salud personal obtenidos de dispositivos móviles, son algunos de los principales medios de obtención de datos. Ahora, ¿cual es la importancia de este artículo?, básicamente es la creación de una forma eficaz y precisa de analizar los datos masivos que se generan, aprovechándolos para generar una fusión que ayuda a mostrar gráficos de conocimiento. Ya que estos datos son provenientes del área de la salud, su nivel de importancia es un tema crítico en cuanto al procesamiento y el análisis de los mismos, ya que la micro-decisión tradicional, media o inclusive macro se verán afectadas según lo que se muestre como resultado.

Capítulo 3

Solución planteada

3.1. Propuesta

Para el actual proyecto se propone el desarrollo de un API RESTful para interconectar la plataforma ya desarrollada bajo Angular 6 y el modelo matemático predictivo desarrollado en R. Además, se deberá tomar la aplicación web desarrollada y ajustar su sección de reportes, para mostrar los resultados de la predicción de igual manera a como lo haría un gráfico en Shiny. Dicho API deberá contar con algún método de seguridad como JSON Token(o similar), el cual permitirá mantener la información entre los dos puntos de manera segura. Las condiciones para elegir las tecnologías que se utilizaran se llevó a cabo mediante dos elementos básicos entre ellos la documentación (Etapa anterior del proyecto y investigación de trabajos relacionados) y comunidad de ayuda (foros y profesores).

3.1.1. RESTful API

Para el desarrollo del *RESTful API*, se utilizaran las tecnologías elegidas a conveniencia, debido a las razones anteriormente mencionadas. En el caso de *R*, es especialmente versátil para el manejo de elementos estadísticos, de manera mucho más específica en operaciones con matrices y vectores, lo cual facilita a gran escala el manejo de grandes fuentes de información. Además, es un lenguaje que fue diseñado especialmente para hacer análisis estadísticos, por lo cual es bastante exacto y preciso.

Por otro lado, también posee paquetes para desarrollar *RESTful API*s sin tener que migrar o llevar a cabo la ejecución del script, desde algún otro lenguaje. El paquete que utilizaremos sera *Plumber*, dicho paquete migra el código existente en *R* a un Web API, de

dos distintas maneras:

1. Haciendo uso de comentarios especiales.

2. Haciendo uso de la estructura programática que ellos establecen.

Figura 3.1: Tecnologías para RESTful API

3.1.2. Frontend

En la primera etapa del proyecto se desarrolló una aplicación web en Angular, la cual contiene entre ellas características como la gestión de usuarios, gestión de inversores y algunas consultas a la base de datos graficadas en el entorno principal de la aplicación. Una de las razones por las cuales se eligió esta tecnología son la creación de componentes web, este concepto permite modularizar pequeñas porciones de código, lo cual permite su reutilización en otras secciones de la aplicación o incluso en otros proyectos de manera mucho más sencilla.

La propuesta actual, incluye el desarrollo de uno o más módulo para la sección de reportes en la aplicación web, dicha sección obtendrá los datos directamente del *RESTful API* por razones de tiempo de respuesta. Los datos provenientes del modelo consultado deberán ser graficados en la sección de reportes y además este debe ofrecer la facilidad de poder hacer zoom sobre cierta región del mismo para poder ver de manera más detallada la información renderizada.

Figura 3.2: Tecnología para Frontend (Angular)

3.2. Stakeholders

Cuadro 3.1: Descripción del Stakeholder #1

Nombre	Andrés de Jesús García Salas.
Rol	Practicante (Desarrollador).
Responsabilidades	Desarrollo del proyecto asignado según los requerimientos definidos.
Objetivos	1. Emplear los conocimientos adquiridos durante la extensión de la carrera de Ingeniera en Computación en un ambiente laboral, con el objetivo de lograr un producto de calidad. 2. Entregar informes escritos de calidad de acuerdo a los estándares establecidos por el Tecnológico de Costa Rica.
Expectativas	Creación y integración de un sistema completamente funcional.

Cuadro 3.2: Descripción del Stakeholder #2

Nombre	Efrén Antonio Jiménez Delgado
Rol	Supervisor del proyecto programado (Contraparte empresa).
Responsabilidades	Dar seguimiento y retroalimentación al desarrollo del proyecto.

Objetivos	1. Aportar ideas y recomendaciones con respecto a las tecnologías usadas para la finalización del proyecto. 2. Evaluar el funcionamiento del sistema al finalizar el periodo del proyecto.
Expectativas	Ser un guía para el practicante, siendo capaz de brindarle facilidades y conocimiento para el desarrollo del producto final.

3.3. Perspectiva, supuestos y dependencias del producto

Del producto en general se espera que sea de gran importancia para la institución. De manera, que a partir de los resultados e informes generados se puedan tomar decisiones administrativas de gran relevancia. Se espera que el uso del sistema pueda generar los resultados en un formato comprensible y preciso.

3.3.1. Supuestos

A continuación, se detallan características o situaciones esperadas por partes de los desarrolladores tanto de los posibles usuarios como del entorno donde se encontrará el sistema.

- Los usuarios utilizarán cualquier dispositivo electrónico con pantalla y tenga acceso a internet. En un principio se pretende que la mayoría de los usuarios accedan a través de una computadora.
- La institución brindará acceso a la obtención de los datos de los paneles solares. De manera que se brindarán las credenciales para poder acceder a la base de datos que está en PostgreSQL.
- La institución proveerá acceso al servidor para alojar el sistema. Una vez creada la

aplicación web, podrá ser alojada en el servidor del ITCR u algún otro medio del cual la institución disponga.

3.3.2. Dependencias

Características y detalles los cuales serán críticos para el correcto funcionamiento y desarrollo del proyecto.

- Muy alta dependencia de acceso estable al servidor donde está alojada la base de datos.
- Alta dependencia a una conexión de internet estable.
- Muy alta dependencia de datos confiables de los paneles, ya que a partir de ellos se generan los reportes sobre las predicciones.

3.4. Metodología

En la tabla 3.3, se detalla el plan propuesto para abarcar cada objetivo definido.

Cuadro 3.3: Metodología propuesta para alcanzar los objetivos.

Objetivo general: Contribuir con la Inteligencia de Negocios de la Infraestructura en Ciencia de los datos para energías Renovables del ITCR (ICDER), mediante la automatización del proceso de alimentación de los modelos sobre la Infraestructura de datos.			
Objetivo específico	**Tarea**	**Meta**	**Indicador**
Identificar los requerimientos para la comunicación entre los modelos y ICDER.	1. Entrevistar a los responsables del ICDER. 2. Leer todo material bibliográfico consultado anteriormente para el actual proyecto. 3. Realizar una búsqueda bibliográfica en las bases de datos suscritas de la Institución para consultar trabajos realizados anteriormente con relación al proyecto actual.	1. Un documento con el reporte de la información obtenida en las entrevistas con los responsables del ICDER. 2. La sección de Trabajos relacionados.	1. Entrevista de al menos dos responsables del ICDER. 2. Consulta de al menos 4 artículos sobre proyectos relacionados. 3. Lista de al menos 30 requerimientos (funcionales y no funcionales) del ICDER.

Continúa en la siguiente página...

Cuadro 3.3 – continuación de la página anterior

Objetivo específico	Tarea	Meta	Indicador
Definir el(los) módulo(s) para reportes provenientes de los modelos en el ICDER	1. Definición y diseño de una arquitectura Backend(R-RestAPI). 2. Definición y diseño de Mockups. 3. Definición del módulo de Frontend.	1. La definición y el diseño de una arquitectura Backend(R-RestAPI). 2. La definición y el diseño de Mockups. 3. La definición del módulo de Frontend.	1. La definición y el diseño de una arquitectura Backend(R-RestAPI) que incluya al menos un 90 % de los requerimientos funcionales. 2. La definición y el diseño de Mockups que incluyan al menos un 90 % de los requerimientos funcionales. 3. La definición del módulo de Frontend con al menos un 90 % de los requerimientos funcionales.

Continúa en la siguiente página...

Cuadro 3.3 – continuación de la página anterior

Objetivo específico	Tarea	Meta	Indicador
Desarrollar el(los) módulo(s) para reportes provenientes de los modelos en el ICDER	1. Desarrollar el módulo(s) necesarios para la comunicación entre los modelos estadísticos y el ICDER. 2. Desarrollar el front-end respectivo al módulo de reportes en el ICDER.	1. El desarrollo de los módulo(s) necesarios para la comunicación entre los modelos estadísticos y el ICDER. 2. El desarrollo del front-end respectivo al módulo de reportes en el ICDER.	1. El desarrollo de los módulo(s) necesarios para la comunicación entre los modelos estadísticos y el ICDER con al menos el 90 % de los requerimientos funcionales. 2. El desarrollo del front-end respectivo al módulo de reportes en el ICDER con al menos el 90 % de los requerimientos funcionales.

3.5. Análisis de los riesgos

El desarrollo de proyectos nunca a estado exento de riesgos, y el actual no es mucho menos el caso. Por lo tanto, la identificación de los mismos es esencial en etapas tempranas del proyecto, esto permite desarrollar medidas en contra de dichos riesgos ahorrando tiempo y costos.

El cuadro 3.4 hace referencia a la plantilla usada para documentar los riesgos. En ella se pueden observar el nombre, categoría y causa de los riesgos, además se establece una estrategia de evasión y mitigación como medida contra los mismos. Por otro lado, en las tablas 3.5 y 3.6, se puede apreciar los grados disponibles de probabilidad y de estimación de impacto para los riesgos a establecer en el cuadro anteriormente mencionado.

Cuadro 3.4: Plantilla de riesgos

Nombre o Descripción del Riesgo	
Categoría del Riesgo	(Tecnológico, Humano, Políticas, etc.)
Causa del Riesgo	
Impacto (I)	Rango del 1 al 5
Probabilidad de ocurrencia (P)	Valores en el intervalo de [0, 1]
Exposición (I*P)	
Estrategia de evasión	
Estrategia de mitigación	

3.5.1. Estimación de la Probabilidad

Cuadro 3.5: Estimación de la probabilidad de los riesgos.

Frecuencia	Peso	Criterio
Improbable	0.2	El evento solo podría ocurrir excepcionalmente
Probable	0.4	El evento podría ocurrir en algún momento.

Cuadro 3.5: Estimación de la probabilidad de los riesgos.

Frecuencia	Peso	Criterio
Moderado	0.6	El evento podría ocurrir con cierta periodicidad.
Critico	0.8	El evento podría ocurrir en forma recurrente.
Catastrófico	1	El evento podría ocurrir en la totalidad de las circunstancias.

3.5.2. Estimación del Impacto

Cuadro 3.6: Estimación de la impacto de los riesgos.

Criterio	Retraso en planificación (Tiempo)	Valor numérico
Muy Bajo	Una Semana	1
Bajo	Dos Semanas	2
Moderado	Un Mes	3
Alto	Dos Meses	4
Muy Alto	Más de dos Meses	5

3.5.3. Lista de los Riesgos

Cuadro 3.7: Riesgo 1

Nombre o Descripción del Riesgo	No finalizar el proyecto.
Categoría del Riesgo	Humano
Causa del Riesgo	El desarrollador no dispone de las capacidades para finalizar el proyecto solicitado.
Impacto (I)	5
Probabilidad de ocurrencia (P)	0.2
Exposición (I*P)	1
Estrategia de evasión	No existe estrategia de evasión.
Estrategia de mitigación	Revisiones constantes con los encargados y solicitar ayuda en caso de ser necesario.

Cuadro 3.8: Riesgo 2

Nombre o Descripción del Riesgo	Proyecto puede perder prioridad.
Categoría del Riesgo	Humano
Causa del Riesgo	Por carga de trabajo de los encargados.
Impacto (I)	2
Probabilidad de ocurrencia (P)	0.4
Exposición (I*P)	0.8
Estrategia de evasión	No existe estrategia de evasión.
Estrategia de mitigación	Revisiones constantes con los encargados.

Cuadro 3.9: Riesgo 3

Nombre o Descripción del Riesgo	Mala interpretación de los requerimientos.
Categoría del Riesgo	Humano
Causa del Riesgo	Los requerimientos no están claros.

Impacto (I)	5
Probabilidad de ocurrencia (P)	0.4
Exposición (I*P)	2
Estrategia de evasión	Retroalimentación constante en los requerimientos.
Estrategia de mitigación	Retroalimentación constante en los requerimientos.

Cuadro 3.10: Riesgo 4

Nombre o Descripción del Riesgo	Usuarios no logren adaptarse al sistema.
Categoría del Riesgo	Humano
Causa del Riesgo	Resistencia al cambio.
Impacto (I)	4
Probabilidad de ocurrencia (P)	0.2
Exposición (I*P)	0.8
Estrategia de evasión	Capacitación a los usuarios del sistema y manual de uso.
Estrategia de mitigación	Diseño mas intuitivo.

Cuadro 3.11: Riesgo 5

Nombre o Descripción del Riesgo	Falla en la estimación del tiempo.
Categoría del Riesgo	Humano.
Causa del Riesgo	Mala planificación.
Impacto (I)	4
Probabilidad de ocurrencia (P)	0.4
Exposición (I*P)	1.6
Estrategia de evasión	Contar con juicio experto en el diseño del sistema.
Estrategia de mitigación	Hacer uso de estrategias ágiles en la administración de proyectos.

Cuadro 3.12: Riesgo 6

Nombre o Descripción del Riesgo	Ausencia de datos de los paneles fotovoltaicos, esto causaría un retraso inminente del proyecto. No se mostrarían los gráficos e informes requeridos, los modelos no podrían ser reentrenados o actualizados.
Categoría del Riesgo	Humano
Causa del Riesgo	Indisponibilidad de los encargados de la gestión de los datos de los paneles solares.
Impacto (I)	4
Probabilidad de ocurrencia (P)	0.4
Exposición (I*P)	1.6
Estrategia de evasión	No existe estrategia de evasión para este riesgo.
Estrategia de mitigación	Establecer reuniones con suficiente tiempo de anticipación y contar con medios de comunicación estables y constantes entre los considerables.

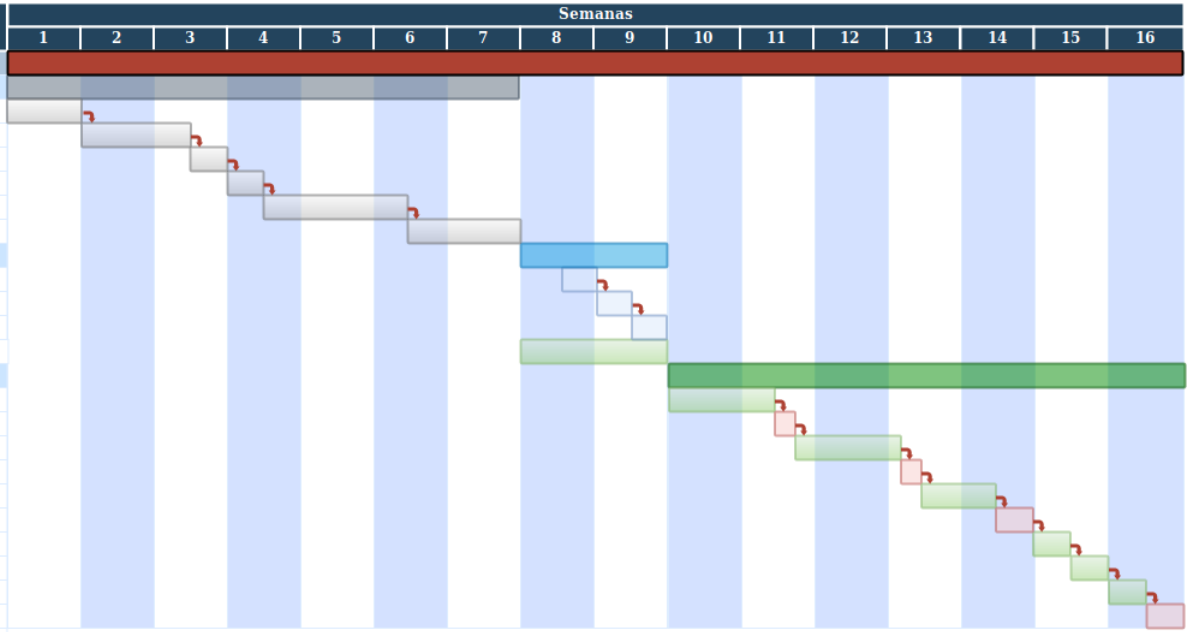
Cuadro 3.13: Riesgo 7

Nombre o Descripción del Riesgo	Disponibilidad de la conexión a internet .
Categoría del Riesgo	Tecnológico.
Causa del Riesgo	Caída de la red.
Impacto (I)	3
Probabilidad de ocurrencia (P)	0.4
Exposición (I*P)	1.2
Estrategia de evasión	No existe manera de evadir el riesgo.
Estrategia de mitigación	No existe manera de mitigar el riesgo.

3.6. Cronograma de trabajo

En la figura 3.3, se establece el cronograma de trabajo para el desarrollo del proyecto en una duración de aproximadamente 16 semanas. Este se divide en 3 etapas fundamentales las cuales son etapa de requerimientos (7 semanas), desarrollo del sistema (5 semanas) y evaluación del sistema (4 semanas).

	Tareas	Duracion	Comienza	ETA
1	Ejecución completa del proyecto	16 semanas	4.02.19	26.05.19
2	Etapa de requerimientos	7 semanas	4.02.19	24.03.19
3	Investigación y recopilación de trabajos relacionad	1 semana	4.02.19	10.02.19
4	Corrección de scripts y procesos en la recolección d	& 1/2 sem.	11.02.19	20.02.19
5	Instalación de entorno para proyecto(s) desarrollad	1/2 semana	20.02.19	24.02.19
6	Pruebas y ejecución de proyecto desarrollados ant.	1/2 semana	25.02.19	27.02.19
7	Pruebas de concepto con tecnologías tentativas	2 semanas	27.04.12	13.03.19
8	Desarrollo de documentación*	1 & 1/2 sem.	13.03.19	24.03.19
9	Etapa de diseño	2 semanas	25.03.19	07.04.19
10	Definición y diseño de una arquitectura Backend	1/2 semana.	25.03.19	27.03.19
11	Definición y diseño de Mockups	1/2 semana	27.03.19	31.03.19
12	Definición del modulo de Frontend	1/2 semana	01.04.19	04.04.19
13	Pruebas de concepto	2 semanas	25.03.19	07.04.19
14	Etapa de desarrollo	7 semanas	08.04.19	..19
15	Desarrollo del RESTful API (S.S. incluida)	1 & 1/2 sem.	08.04.19	24.04.19
16	Pruebas ligeras sobre RESTful API	1/4 semana	24.04.19	25.04.19
17	Desarrollo del Modulo de reportes	1 & 1/2 sem.	26.04.19	07.05.19
18	Pruebas ligeras sobre reportes	1/4 semana	07.05.19	08.05.19
19	Desarrollo del Modulo para Re-entrenamiento de m	1 semana	08.05.19	15.05.19
20	Pruebas ligeras modulo de re-entrenamiento	1/2 semana	15.05.19	19.05.19
21	Montaje de RESTful API en AWS with Docker	1/2 semana	20.05.19	22.05.19
22	Montaje de Angular 6 App en AWS	1/2 semana	23.05.19	26.05.19
23	Ensamble de ambos sistemas en UAT	1/2 semana	27.05.19	29.05.19
24	Pruebas ligeras sobre sistemas en UAT (AWS)	1/2 semana	30.05.19	02.06.19



* Todo lo realizado hasta el momento, requerimientos. Todo lo respectivo a la primera entrega. Se establecen, la mayoria de tareas a seguir para el transcurso del proyecto.

S.S = Semana Santa

Figura 3.3: Diagrama de Gantt - Cronograma

Capítulo 4

Definición de requerimientos

4.1. Introducción

La toma de requerimientos es siempre un punto crucial para la correcta comprensión de las tareas a realizar. Es por ello, que se debe hacer un buen planteamiento tanto de los requerimientos funcionales como no funcionales, de manera que los mismos sean comprensibles, tanto para el desarrollador como para cualquier tercero que desee estar en contexto con el proyecto.

En este capítulo se especificarán los requerimientos que se plantean inicialmente de manera clara y concisa, con la facilidad de poder adaptar pequeños cambios en los mismos, siempre y cuando estos cambios no afecten de manera drástica los requerimientos ya detallados. Además, es en este capítulo que se abarca el primer objetivo específico, el cual está relacionado con la identificación y definición de requerimientos para la interacción entre los modelos predictivos y ICDER.

4.2. Tareas realizadas para definir los requerimientos

Para la comprensión de cómo se deben gestionar los datos generados por los paneles fotovoltaicos y donde están alojados, se consultó al personal encargado del proyecto, con el fin de obtener la forma de acceder a los datos requeridos. Para, de esta forma, poder generar los reportes de predicción con información fiable.

En cuanto a la arquitectura utilizada para el manejo de datos masivos, se consultó mediante diferentes revisiones de literatura, documentaciones y especificaciones de las diferentes tecnologías existentes, para la parte del servidor, para el cliente se desarrollará bajo las tecnologías usadas en las etapas anteriores del proyecto.

4.3. Resultados obtenidos en la definición

4.3.1. Diagrama de caso de uso

En la figura 4.1, se describe de manera general mediante un diagrama de caso de uso las acciones del flujo a desarrollar. Por lo cual, a continuación, se presentan tres actores los cuales son el Administrador general del sistema (root), administrador de institución y usuario cliente común.

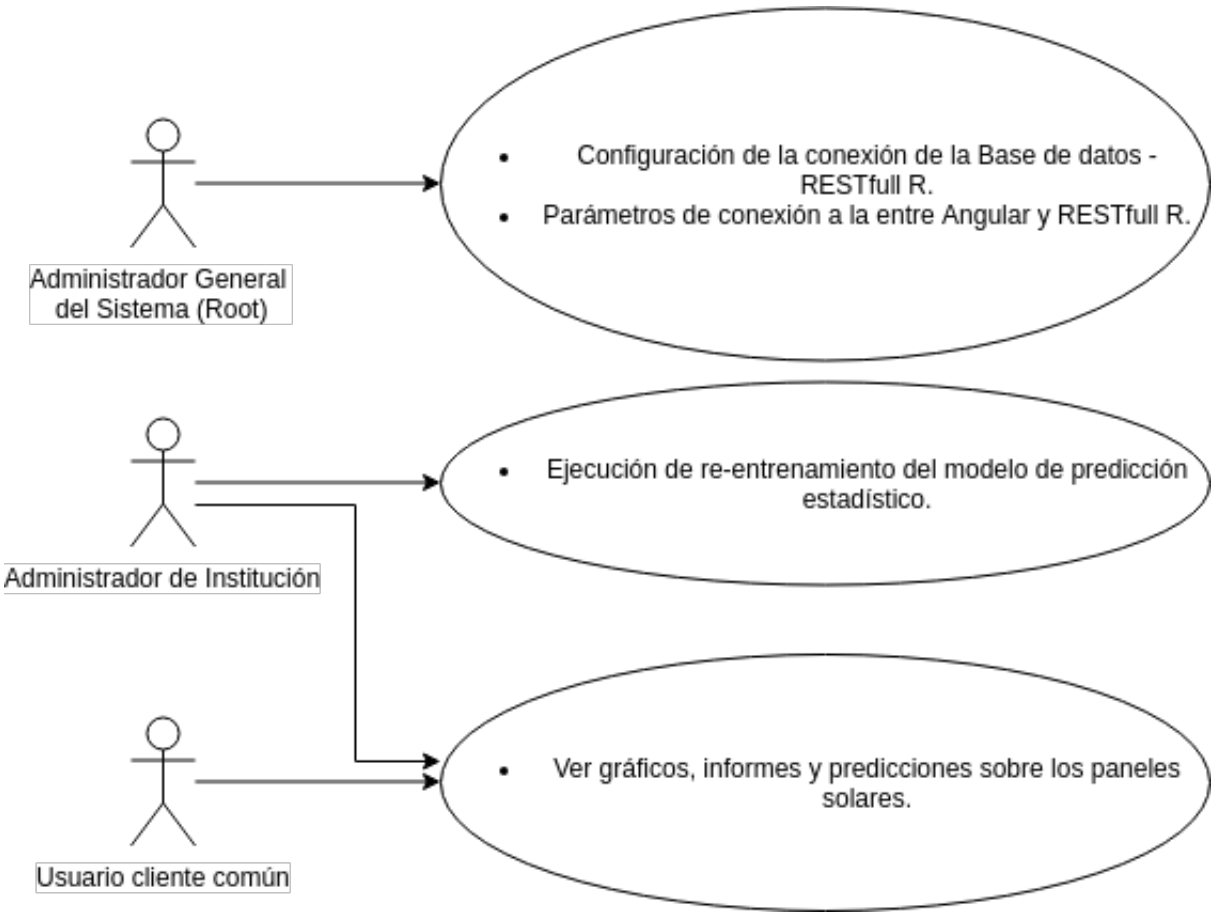


Figura 4.1: Diagrama simplificado de caso de uso

4.3.2. Requerimientos funcionales

En la tabla 4.1, se detallan los requerimientos funcionales requeridos para abarcar cada objetivo definido del proyecto.

Cuadro 4.1: Cuadro de requerimientos funcionales.

Identificador	Nombre	Descripción	Rol
RF-01	EL API RESTful deberá contar con modelos de predicción.	EL API RESTful deberá contar con mínimo uno o más rutas para modelos de predicción, para poder ser consultado desde la opilación angular, específicamente el módulo de reporte(s).	Administrador y usuario cliente.
RF-02	EL API RESTful deberá contar con rutas para reentrenamiento.	EL API RESTful deberá contar con una o más rutas para reentrenar cada uno de los modelos de predicción, cuando el administrador lo encuentre necesario.	Administrador
RF-03	EL API RESTful deberá contar con una alerta para error durante la consulta de datos.	EL API RESTful deberá contar con un mensaje de error para notificar cualquier eventualidad durante la consulta de datos provenientes de la base de datos PostgreSQL, todo esto posterior al entrenamiento.	Administrador

Continúa en la siguiente página...

Cuadro 4.1 – continuación de la página anterior

Identificador	Nombre	Descripción	Rol
RF-04	EL API RESTful deberá contar con una alerta para error de entrenamiento.	EL API RESTful deberá contar con un mensaje de error para notificar cualquier eventualidad durante la etapa de entrenamiento del modelo.	Administrador
RF-05	EL API RESTful deberá contar con una alerta para error en caso de parámetros inválidos.	EL API RESTful deberá contar con un mensaje de error para notificar cualquier eventualidad sobre parámetro(s) invalido(s) en el http request (Error 400).	Administrador.
RF-06	EL API RESTful deberá contar con una alerta para error en caso de ruta invalida.	EL API RESTful deberá contar con un mensaje de error para notificar sobre una ruta invalida en el http request (Error 404).	Administrador.
RF-07	EL API RESTful deberá contar con una alerta para errores internos del servidor.	EL API RESTful deberá contar con un mensaje de error para notificar cualquier eventualidad interna sobre el servidor (Error 500).	Administrador

Continúa en la siguiente página...

Cuadro 4.1 – continuación de la página anterior

Identificador	Nombre	Descripción	Rol
RF-08	EL API RESTful deberá contar con una alerta para errores desconocidos.	EL API RESTful deberá contar con un mensaje de error para notificar sobre cualquier eventualidad durante la ejecución de un modelo (Contacte con el administrador).	Administrador
RF-09	La aplicación en Angular debe contar con un módulo de reportes.	La aplicación en Angular debe contar con un módulo de reportes, el cual se encargará de trasladar las predicciones provenientes del(los) modelo(s) desarrollados en el RESTful API.	Administrador y usuario cliente
RF-10	Cada reporte debe contar con los input's adecuados.	Cada reporte debe contar con los input's adecuados según los parámetros requeridos por las ruta a consultar del modelo de predicción en el RESTfull API.	Administrador y usuario cliente.
RF-11	La aplicación en Angular debe contar con una sección para reentrenar modelos.	La aplicación en Angular debe contar con una sección para que el administrador de Institución pueda re-entrenar los modelos cada vez que considere necesario.	Administrador

Continúa en la siguiente página...

Cuadro 4.1 – continuación de la página anterior

Identificador	Nombre	Descripción	Rol
RF-12	El gráfico deberá contar con la capacidad de realizar zoom sobre si.	El gráfico generado a partir de los datos de la predicción deberá tener la capacidad de realizar zoom sobre cualquier sección de sí mismo.	Administrador y usuario cliente.

4.3.3. Requerimientos no funcionales

Todo proyecto de desarrollo durante algún periodo de su ciclo de vida necesita considerar su conjunto de requerimientos no funcionales junto muchos otros factores que están asociados a los mismos. En la tabla 4.2, se detallan los requerimientos no funcionales requeridos para abarcar cada objetivo definido del proyecto.

Cuadro 4.2: Cuadro de requerimientos funcionales.

Identificador	Descripción	Prioridad
RNF-01	EL RESTful API deberá de conectarse con una base de datos en PostgreSQL (Datos de paneles).	Alta
RNF-02	EL RESTful API deberá contar con seguridad para las request bajo el protocolo 'https'.	Media
RNF-03	La obtención de información proveniente de la base de datos deberá de darse de la manera más óptima posible, para asegurar tiempos de respuesta aceptables.	Media
RNF-04	EL RESTful API deberá brindar cada response a cada request bajo formato JSON.	Alta
RNF-05	EL RESTful API deberá poder ser consultado desde cualquier medio capaz de realizar consultas vía http requests.	Alta
RNF-06	La aplicación Angular podrá ejecutarse al menos en los navegadores Edge, Firefox y Chrome.	Media

Continúa en la siguiente página...

Cuadro 4.2 – continuación de la página anterior

Identificador	Descripción	Prioridad
RNF-07	El módulo(s) de reporte gráfico de predicción deberá ser intuitivo (Menos de una hora de aprendizaje)	Baja
RNF-08	El módulo(s) de reporte gráfico de predicción deberá ser 'Responsive', a fin de poder ser visualizado correctamente en diferentes tipos de dispositivos.	Media
RNF-09	El web API deberá contar con una conexión directa a la aplicación de angular, por factores en el tiempo de respuesta.	Alta
RNF-10	EL RESTful API debe ser lo suficientemente flexible para poder adicionar más requerimientos o módulos (modelos) al mismo.	Media
RNF-11	EL RESTful API deberá soportar una gran cantidad de datos para el entrenamiento del modelo(s).	Alta
RNF-12	El RESTful API deberá ser desarrollado en R utilizando el paquete de Plumber, en cuanto al módulo(s) de reporte gráfico este debe ser desarrollado en Angular.	Alta
RNF-13	El diseño del módulo(s) de reporte gráfico y el RESTful API deberán adaptarse a los estándares de la institución.	Media

Continúa en la siguiente página...

Cuadro 4.2 – continuación de la página anterior

Identificador	Descripción	Prioridad
RNF-14	El RESTful API deberá contar métodos aplicados para evitar ataques Denial Of Service(DOS).	Media
RNF-15	El RESTful API deberá contar métodos aplicados para la desinfección de parámetros, esto con tal de evitar la extracción de datos sensibles.	Media
RNF-16	El RESTful API deberá hacer uso de Cross-Origin Resource Sharing (CORS) para permitir el consumo del API desde la aplicación Angular ubicada en un origen diferente.	Alta
RNF-17	El gráfico generado a partir de los datos de la predicción deberá ser de líneas	Alta
RNF-18	El RESTful API deberá permitir la resolución de múltiples requests, provenientes de distintas conexiones.	
RNF-20	El código fuente y demás documentación del proyecto deberán ser redactados en español, con excepción del abstract.	Media

Capítulo 5

Diseño de la plataforma de software

Para esta etapa se planteó el diseño de la solución software al problema planteado en la sección 1.3. Por lo cual, para modular el desarrollo del producto, se realizaron un conjunto de tareas 5.1, las cuales ayudaran a converger el producto en un desarrollo exitoso.

Cuadro 5.1: Tareas realizadas en el diseño

1. Definición de arquitectura Backend(R-RestAPI)
2. Definición de Mockups
3. Definición de módulo del Frontend

A continuación, se detalla cada una de las tareas en listadas anteriormente.

5.1. Definición de arquitectura Backend(R-RestAPI)

En este caso, la Infraestructura cuenta con dos Backend:

1. Backend de la aplicación Angular.

2. Backend en Plumber R.

Este último, responde las consultas del módulo de predicción. Además, tal como se muestra en la Fig 5.1, dicho Backend funciona como controlador entre la base de datos y la vista correspondiente en el Frontend.

Server.R es el archivo principal encargado de definir los handlers para errores y de montar models.R. Cada petición que llega es redirigida a este último archivo, el cual la toma y la

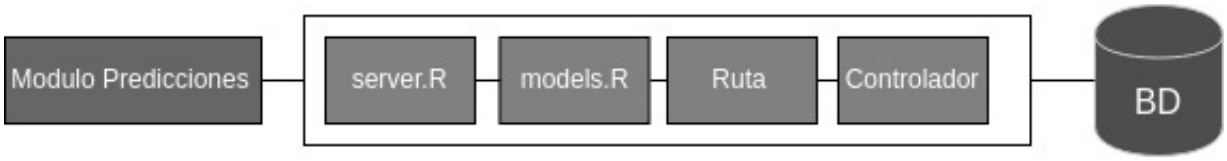


Figura 5.1: Estructura Backend

coloca según corresponda. La ruta trabaja como enlace entre la petición http y el respectivo controlador ubicado en models.R. Por último, el controlador es un método el cual realiza consultas a la base de datos y procesa dichos datos según la necesidad del request.

Por lo tanto, con la estructura descrita en la Fig 5.1 y la descripción del problema (véase 1.3), se concluye que por cada módulo de predicción que se desarrolle se requerirán dos controladores y dos rutas en models.R para su correcto funcionamiento. Hasta la fecha se dispone de un solo algoritmo de predicción por lo cual solo existe una ruta para su reentrenamiento y otra para su uso.

En la Tabla 5.2 se aprecia el diseño de las consultas a realizar por parte del controlador del backend en R, sobre las predicciones existentes.

Cuadro 5.2: Cuadro de consultas del API R.

Controlador	Nombre del método	Parámetro	Acción en la BD
Meses	1. trainPredictModel() 2. predict()	1. Sin parametros 2. meses:'m' / Intervalo de Confianza:'pIntervalLvl'	1. Get 2. Get

5.2. Definición de Mockups

Un Mockup es un fotomontaje que permite a diseñadores gráficos y web mostrar al cliente u desarrolladores sobre como quedaran sus diseños. En la sección anterior, se hizo referencia al manejo de los datos del sistema, de mayor manera en el backend; por lo tanto, a continuación, se muestra el aspecto que tendrán visualmente los datos al ser presentados al usuario.

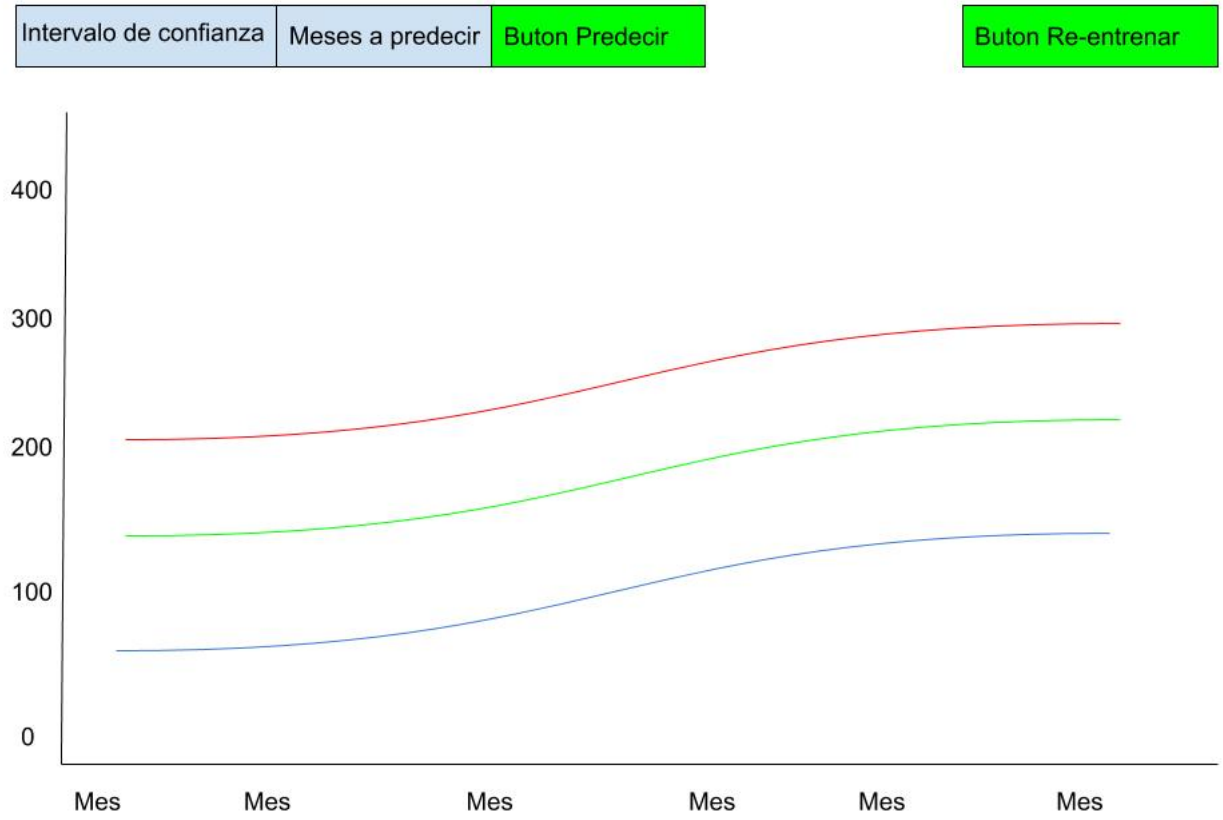


Figura 5.2: Mockup gráfico y form request para predicción.

En la Fig 5.2 se presenta el estándar para realizar los módulos gráficos de reportes de predicciones provenientes del RESTful en R, esta vista estará compuesta por un gráfico que mostrará los datos predichos y en la parte superior izquierda tendrá un form para ingresar los parámetros para cualquier otra predicción que se requiera, a la derecha del mismo habrá un botón para re-entrenar el patrón de predicción, este solo sera visible para usuarios administradores.

5.3. Definición de módulo de Frontend

Para finalizar con los diseños expuestos en las secciones anteriores y poder brindar un medio de representación de los datos y comunicación entre el usuario y el sistema, se debe diseñar una estructura que soporte los componentes gráficos de la definición de mockups (Sección: 5.2) en conjunto con la parte lógica (Sección: 5.1).

Hasta la fecha se cuenta con una aplicación desarrollada en Angular 6, por lo cual no se procederá a analizar sobre que tecnología desarrollar. Se requiere desarrollar un módulo para mostrar el reporte gráfico definido en la sección 5.2. Además, con tal de facilitar la definición de dicha arquitectura se parte de la abstracción de componente, este contiene distintos archivos uno se encarga del código a renderizar en la vista (*component.html / component.css—.scss*) y el otro se encarga de manejar la parte lógica y las acciones de dicha vista. Existen algunos otros conceptos como módulos, services. Para este caso, se desarrollara un módulo llamado predicciones, el cual tendrá un conjunto de (.html—.ts—.scss) encargados de las vista para predicción de meses y además un archivos llamado services, el cual se encargará de hacer los http request al API RESTful.

Con estos conceptos como pilares en el frontend, se permitirá realizar una estrategia que facilite la reutilización de código tanto en la etapa de desarrollo, como en caso de que a futuro se requiera agregar cualquier otra vista para predicción.

Capítulo 6

Desarrollo de la plataforma de software

6.1. Recolección de datos

Atendiendo a trabajos anteriormente realizados, se procedió a visualizar ligeramente los datos recolectados hasta la fecha, mediante esta acción se logró notar que el indexado de los datos, entre algunos otros detalles se encontraban mal estructurados, unido a esto, también los datos tenían algunos meses de no ser recolectados debido a que los scripts desarrollados anteriormente para el proyecto no estaban siendo ejecutados en el servidor local del Campus Tecnológico San Carlos.

Por lo cual, se decidió desarrollar un nuevo script que unificaba todos los existentes y agregaba una estructura más modular para el código facilitando la programación de una ejecución automática por parte del servidor con Crontab (Comando UNIX que facilita la ejecución de una lista de comandos bajo un tiempo u acción especifica), además con estas rutinas calendarizadas el script seguiría siendo ejecutado todos los días a las 11pm de la noche, recolectando todos los datos diarios generados por los paneles fotovoltaicos y guardándolos en el servidor local de la institución.

Las funciones básicas del script son recolectar los datos de los paneles bajo un rango de fechas dado y subirlos a un servidor en específico o generar un archivo con formato JSON el cual contiene todos los datos generados en el rango especificado. Para la base de datos, se realizó una limpieza completa de sus datos, se reiniciaron los identificadores seriales de las tablas y se realizó una recuperación completa de todos los datos a partir de marzo del 2017.

6.2. Desarrollo del Backend

En la sección de diseño se comprendió a fondo la arquitectura del *Backend* a desarrollar, además se especificaron las consultas que este debe realizar a la base de datos, las cuales para recalcar, son básicamente un '*Select **' de una tabla en específico según lo requiera el modelo para su respectivo entrenamiento.

En esta sección se muestra el detalle de cada dirección del RestAPI con el controlador, el nombre del método y las acciones que ejecuta, así como su valor de retorno. (Véase cuadro 6.1)

Cuadro 6.1: Cuadro de consultas del API R.

Controlador	Nombre del método	Acción del método	Retorno
Meses	1. trainPredictModel() 2. predict()	1. Get 2. Get	1. Objeto JSON con Código del status de la consulta y el summary de la misma. 2. Objeto JSON con lista de objetos que representan los meses predichos por el API a partir del último mes con el cual el algoritmo fue entrenado.

Este API cuenta además con Swagger para pruebas en cualquiera de sus endpoints desarrollados. (Véase Fig. 6.1 & Fig. 6.2)

Con anterioridad se consideró implementar un API bajo la tecnología de NodeJS en conjunto con un paquete llamado R-node para la ejecución de scripts desarrollados en R, todo esto debido a algunas pruebas de concepto desarrolladas, sin embargo, el paquete mencionado ya tenía algunos años deprecado, por lo cual, al encontrarse fuera de mantenimiento se procedió a buscar una solución óptima, y se encontró el paquete de Plumber en R.

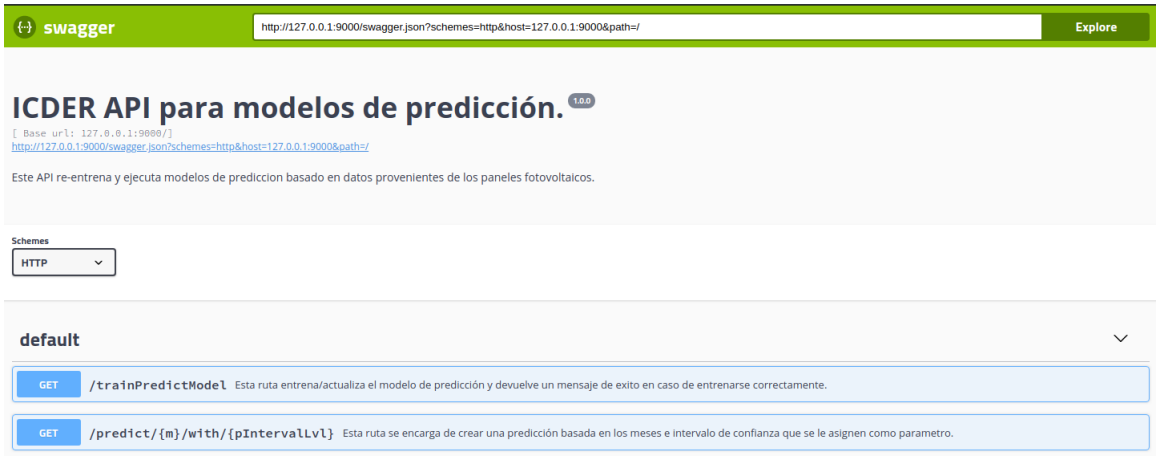


Figura 6.1: #1 Swagger UI.

6.3. Desarrollo del módulo en el Frontend

En la sección de diseño se estableció el comportamiento de la vista de este módulo en el programa, aprovechando la arquitectura ya definida en la aplicación la cual es orientada a contenedores y componentes. Por lo cual, como se menciona en la sección 5.3, en etapas anteriores del proyecto se desarrolló una aplicación en Angular 6, por lo cual no se requirió definir con que tecnologías trabajar esta sección.

El módulo desarrollado es totalmente visible para todos los tipos de usuarios del sistema 'Administrador General', 'Administrador de Institución' y Clientes, sin embargo, el botón de 'reentrenar' solo se encuentra disponible para los usuarios de tipo administrador, solo ellos pueden decir cuando actualizar el estado del modelo, esto debido a que el proceso de reentrenamiento conlleva un costo computacional mayor para el equipo que este alojando el R-API, por ello no cualquier usuario dispone de esta funcionalidad. Esto se puede apreciar en las Figs. 6.3 & 6.4.

Cada vez que al módulo se le solicita una acción ('Las acciones son aquellas instrucciones que normalmente repercuten en una consulta al backend node o R'), las posibles acciones a ejecutar en este módulo son una predicción o un reentrenamiento, cada vez que es solicitada una acción el módulo se dirige al servicio respectivo, el cual contiene los paquetes y procedimientos necesarios para realizar la consulta. Ahora, cada vez que se ejecuta una acción se hace de manera asincrónica por lo cual se pueden ejecutar otras acciones en otros módulos. Claro existe una excepción para el mismo, ya que no se puede ejecutar ninguna predicción si el reentrenamiento se está llevando a cabo esto debido a que se depende del modelo que está

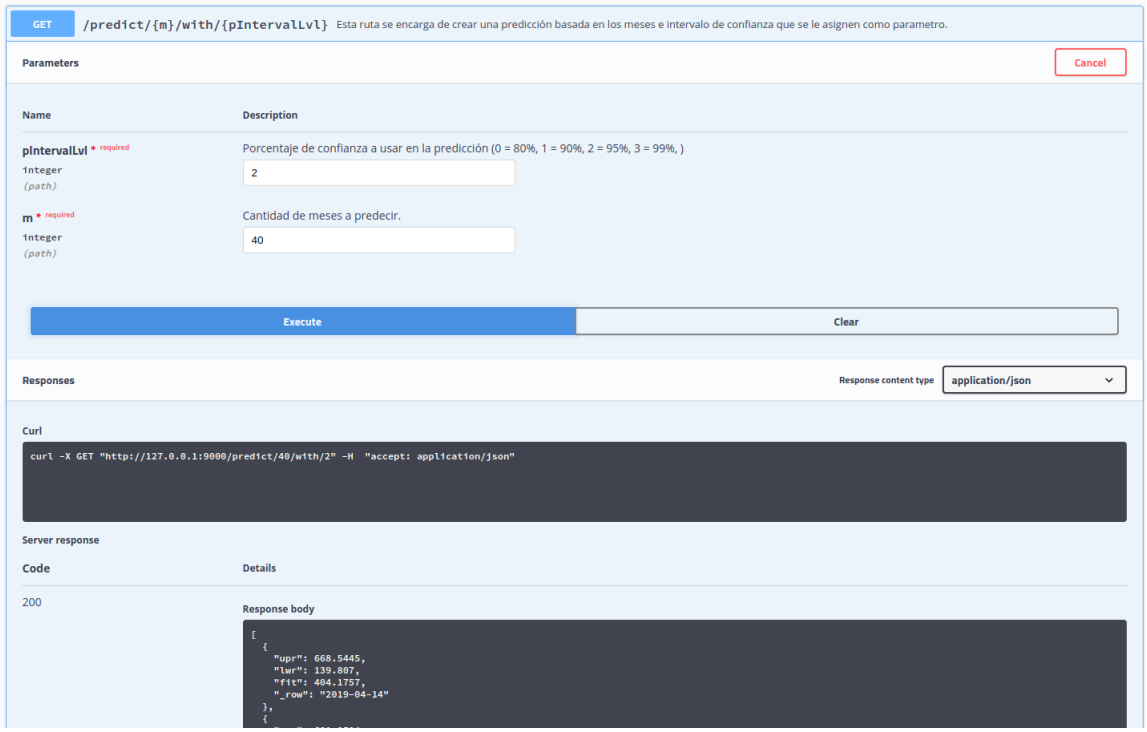


Figura 6.2: #2 Swagger UI.

siendo actualizado en ese momento para poder realizar predicciones, pero fuera de esa acción todas las demás son válidamente ejecutables mientras se mantiene la ejecución de cualquiera de las dos acciones del módulo de predicciones. Véase Figs. 6.5 & 6.6

6.4. Docker's

En la presente sección y consecuentes, se incluye todo lo respectivo a ensamble, montaje en AWS y creación de Docker's para el ambiente UAT(User acceptance testing). Una vez que estuvieron listos tanto el módulo de predicción como el RestAPI, se procedió a investigar y montar dichas aplicaciones en un Docker.

La principal razón del montaje de las aplicaciones mediante Docker, se debe a la facilidad que el mismo genera a la hora de ensamblar una aplicación sobre un ambiente vació, ya que el incluye todo lo necesario para la ejecución exitosa de la aplicación con la que se le configure. La configuración de un Docker es sumamente sencilla, lo único que se requiere es un archivo llamado 'Dockerfile' donde se le especifica todos los componentes, comandos y ubicación de la aplicación, necesarios para la ejecución y construcción exitosa aplicación. Ver ejemplo de Dockerfile para R-API Fig 6.7.

Figura 6.3: Administrador UI.

Una vez finalizado el archivo de configuración 'Dockerfile', lo único necesario a llevar a cabo es la construcción del mismo, etiquetarlo y subirlo a la plataforma 'Docker Hub', el cual es un sitio para el manejo de repositorios de Docker semejante a Github. Normalmente el comando es ejecutado con la terminal situada en la misma ubicación del Dockerfile, esto explica el punto al final del comando. En caso contrario se debería establecer la ubicación del Dockerfile que se desea construir.

```
$ docker build -t nombre-del-docker .
$ docker tag docker-image-number dockerhub-user/docker-name:tag
$ docker push dockerhub-user/docker-name
```

El proceso de montaje es relativamente sencillo debido a la gran variedad de Docker's existentes, lo cual facilita el trabajo al basarse en un Docker construido con anterioridad que se ajusta a las necesidades de lo que se necesita montar. Un ejemplo claro es el Dockerfile que se ocupó para montar la aplicación de Angular el luce así.

```
FROM httpd:2.4
MAINTAINER Andres Garcia <jgarsalas@gmail.com>

COPY ./dist/ /usr/local/apache2/htdocs/
```

El único requerimiento hasta este punto fue Docker, dado que el mismo ya poseía sumas características completamente necesarias para alojar los diferentes programas que componen

Figura 6.4: Cliente UI

la aplicación. posteriormente la aplicación se alojó en la raíz del servidor web Apache ubicado en el Docker.

6.5. Publicación en Amazon Web Services(AWS)

Previo al ensamble de los Docker's en UAT, se requerían máquinas virtuales que alojaran las aplicaciones en sus Docker's respectivos. Por ello, se procedió a crear dos instancias escalables en AWS, por el momento dichas instancias cuentan con características de una máquina virtual FREE en AWS, por lo cual no cuentan con mucho rendimiento, sin embargo, en cualquier momento pueden ser modificadas con mayores especificaciones.

Una de las dos instancias alojará el API de R debido a que reentrenar el modelo requiere la mayoría del poder computacional de la instancia y a futuro cuando se cuente con mayor volumen de datos esta tarea se vería mayormente comprometida en una instancia que contase con diversas aplicaciones en ejecución. Por otra parte, la segunda instancia contará con el API en Node ya elaborado y con la aplicación en Angular con el módulo desarrollado. Ahora es importante mencionar que si se llegan a apagar/desactivar las instancias en AWS, las mismas algún tiempo después de iniciadas varían su dirección ip publica, por lo cual se requerirá que se vuelva a construir el Docker de la aplicación de Angular con las direcciones ip de los Backends actualizadas para que el correcto funcionamiento, de lo contrario su ejecución

Figura 6.5: Consulta de Predicción.

no sera del todo correcta. Para ello, solamente se requiere actualizar la dirección ip en el *enviroment.prod.ts* y por último construir de nuevo el Docker para su nueva publicación en el sitio 'Docker hub'.

```
1    $ ng build --prod
```

6.6. Ensamble en UAT

En la sección 3.6, Cronograma de trabajo, se comentó la distribución de tareas a realizar en el proyecto. En dicha sección, se mostró la segmentación de las etapas en tres apartados, mostrando un comportamiento incremental en el tipo de desarrollo del proyecto. Como cierre de cada uno de esos apartados se realizó un proceso semejante a llevar una aplicación a un ambiente de pruebas seguro, sin aun estar bajo producción.

Todo producto debe ser probado antes de ser lanzado a producción, por lo cual UAT resulta ser un proceso fundamental y no opcional, en el desarrollo de productos de software. Para concluir, se describirá de manera breve el proceso que se llevó a cabo para ensamblar los contenedores en el ambiente de pruebas.

Primero se procede a levantar las instancias de AWS, una vez levantadas ambas instancias se requiere descargar los containers respectivos a cada una de las instancias como describimos en la sección anterior, para descargar los containers se ejecuta un comando similar al siguiente.

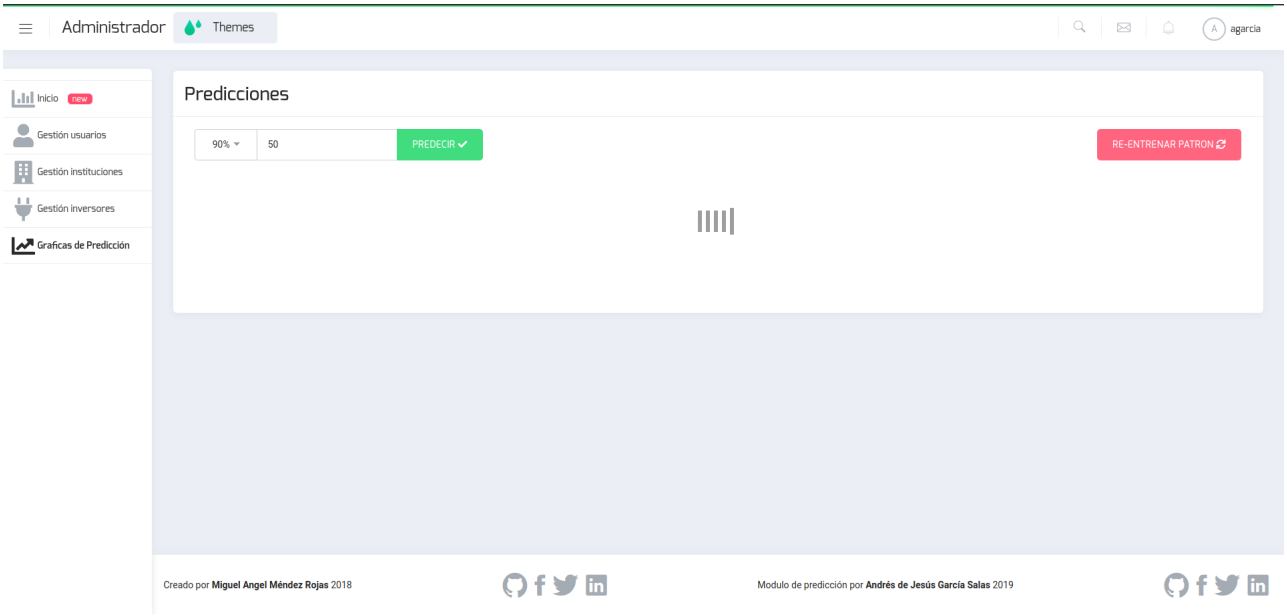


Figura 6.6: Reentrenamiento.

```
$ docker pull docker-user/docker-name:tag

$ sudo docker run -dit --name nombre-container-local -p num-puerto-instancia:num-puerto-container docker-user/docker-name:tag # Ejecuci n del docker
```

La ejecución del docker lleva una estructura sencilla y bastante comprensible a primera vista. Por el momento, el Docker encargado de correr el API de R es una simple instancia sin balanceo de cargas, se intentó seguir la guía detallada por *'Plumber IO - Advanced Hosting Docker'*, sin embargo no se logró con éxito debido a problemas a la hora de configurar varias instancias saliendo por el mismo puerto del docker, por lo cual, ni siquiera se pudo llegar a la última sección la cual conllevaba a balancear las cargas entre las instancias.

Como resultado actualmente solo existe una instancia del API corriendo, lo cual afecta la ejecución de múltiples consultas provenientes de múltiples clientes en instancias de la aplicación de Angular, todo esto debido a que R es single-threaded (Un solo hilo). Para ejecutar tareas multi-threaded es necesario hacer uso de paquetes en otras tecnologías como Open BLAS, sin embargo, no se logró contar con el tiempo necesario para su investigación he implementación.

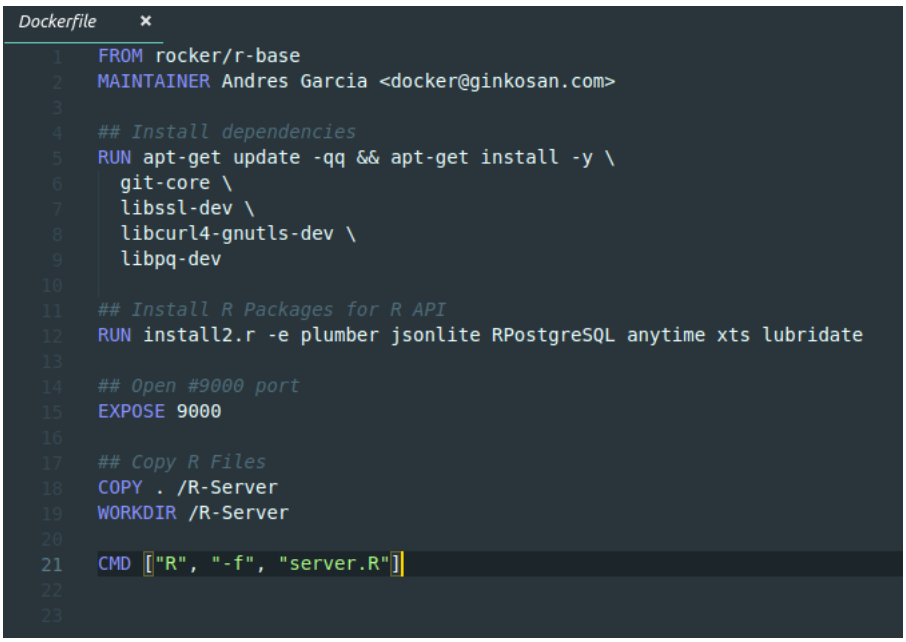


Figura 6.7: Ejemplo Dockerfile R-API .

Capítulo 7

Conclusiones

La ausencia de datos de utilidad para la toma de decisiones en los cuerpos administrativos encargados de los paneles fotovoltaicos es realmente elevada, y con el paso del tiempo las decisiones tomadas alrededor de estos equipos sin dichas bases, conlleva a acciones que en algunos momentos no son del todo, las adecuadas. Debido a dichas acciones se requirió iniciar el desarrollo de una plataforma encargada de dar solución a esta problemática y se concluye que con el avanza de las herramientas ya desarrolladas existirá un importante cambio en la lógica de negocios llevada a cabo por los encargados de los paneles fotovoltaicos. La solución desarrollada gira entorno a los sistemas de generación de energía (paneles fotovoltaicos), enfocada en tomar el conjunto de datos antes sin ninguna utilidad a ahora un modelo entrenado que genera estadísticas de suma utilidad para los usuarios de la infraestructura, facilitando la toma de decisiones.

No esta demás mencionar que el desarrollo del proyecto se vio favorecido por el conjunto de pasos elegido para el desarrollo del mismo, normalmente estos son los más adecuados para el avance de un proyecto ingenieril de desarrollo de software, haber tomado primero los requerimientos evito el malgasto de tiempo en las etapas consecuentes, debido a que de este proceso nació una mejor visualización de la vía de producción que debía tomar para desarrollarlo de la manera más efectiva. Por consecuencia, la etapa de diseño también evito el malgasto de tiempo durante el desarrollo, gracias a haber tomado tiempo anteriormente dedicado a armonizar las tecnologías elegidas con la solución desarrollada hasta la fecha y la proveniente para el problema que se abarco.

El desarrollo del proyecto debido a las etapas anteriores se llevó a cabo dentro del tiempo estimado en un 93.75 % de sus requerimientos, por lo cual, es seguro mencionar que, si en un

futuro cercano el Campus Tecnológico de San Carlos trabaja con esta plataforma para la administración de los paneles fotovoltaicos, los datos generados por los mismos serán explotados de una manera beneficiosa por parte de la lógica del negocio.

Referencias

Bartsch H, J. T., Thompson WK, y AM, D. (2014, Dec). A web-portal for interactive data exploration, visualization, and hypothesis testing. *Neuroinform*. Descargado de `https://www.frontiersin.org/articles/10.3389/fninf.2014.00025/full` doi: 10.3389/fninf.2014.00025

Celsia. (2018). *¿qué son y cómo son los paneles solares?* Descargado de `https://blog.celsia.com/que-son-como-son-paneles-solares`

Chatti, M. A., Dyckhoff, A. L., Schroeder, U., y Thüs, H. (2013). A reference model for learning analytics. *International Journal of Technology Enhanced Learning*, *4*(5-6), 318–331.

Haider, M., y Gandomi, A. (2014, Dec). *Beyond the hype: Big data concepts, methods, and analytics.* International Journal of Information Management. Descargado de `https://www.sciencedirect.com/science/article/pii/S0268401214001066#!`

Hornik, K. (2018, oct). *R faq.* Descargado de `https://cran.r-project.org/doc/FAQ/R-FAQ.html#What-is-R_003f`

Liu, A. (2015). *Data science and data scientist.* IBM. Descargado de `http://www.researchmethods.org/DataScienceDataScientists.pdf`

Liu, J., Zhang, Y., y Xing, C. (2017, Jan). Medical big data web service management platform. En *2017 ieee 11th international conference on semantic computing (icsc)* (p. 316-321). doi: 10.1109/ICSC.2017.69

Momjian, B. (2002, jan). Postgresql: Introduction and concepts. *postgresql.org*, 10. Descargado de http://sgbdr.fr/SGBDR/PostgreSQL/Documentation/PostgreSQL%20Concepts.pdf

Park, K., Nguyen, M. C., y Won, H. (2015, July). Web-based collaborative big data analytics on big data as a service platform. En *2015 17th international conference on advanced communication technology (icact)* (p. 564-567). doi: 10.1109/ICACT.2015.7224859

What is angular? (2018, oct). Descargado de https://angular.io/docs.

Zhuang, Y. L. L. M., ZY. (2018, feb). 'mean+r': implementing a web-based, multi-participant decision support system using the prevalent mean architecture with r based on a revised intuitionistic-fuzzy multiple attribute decision-making model. *Springer Berlin Heidelberg*. Descargado de https://doi.org/10.1007/s00542-018-3755-z doi: 10.1007/s00542-018-3755-z