**DataHome**

**(DAS) Documento Arquitectura Sistemas**

**Versión 1.0**

**Identificación de Documento**

|  |  |
| --- | --- |
| **Identificación** |  |
| **Proyecto** | DataHome |
| **Versión** | 1.0 |

|  |  |
| --- | --- |
| **Documento mantenido por** | Marco Puga |
| **Fecha de última revisión** | 14/08/2014 |
| **Fecha de próxima revisión** | 21/11/2024 |

|  |  |
| --- | --- |
| **Documento aprobado por** | Alex Baeza (Gerente de Proyecto) |
| **Fecha de última aprobación** | 21/10/2024 |

**Historia de Revisiones**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor** |
| 21/10/2024 | 1.0 | Versión inicial del documento | Marco Puga |
| 22/10/2024 | 1.1 | Revisión de documento | Constanza Vilaza |
|  |  |  |  |
|  |  |  |  |

**Tabla de Contenidos**

Contenido

[1. Introducción 3](#_Toc180850385)

[2. Referencias 5](#_Toc180850386)

[3. Resumen ejecutivo 5](#_Toc180850387)

[4. Representación 5](#_Toc180850388)

[5. Metas y Restricciones de la Arquitectura 7](#_Toc180850389)

[6. Otros antecedentes y consideraciones 9](#_Toc180850390)

# Introducción

* 1. **Contexto del Problema**

El mercado inmobiliario en el sector oriente de la región Metropolitana, presenta una alta variabilidad de precios que dificulta la toma de decisiones tanto para compradores como para corredores. **DataHome** tiene como objetivo proporcionar predicciones precisas de precios de propiedades, análisis de mercado y estadísticas basadas en datos históricos y actuales, facilitando la planificación y evaluación del mercado inmobiliario.

* 1. **Propósito**

El propósito del documento es describir la arquitectura de la Api *DataHome*, una sistema que permite a los usuarios consultar precios de propiedades, generar reportes y realizar predicciones utilizando un modelo de IA.

* 1. **Ámbito**

El ámbito del proyecto incluye el desarrollo de una API basada en un modelo de predicción de precios de propiedades en comunas clave de la zona oriente de Santiago: Ñuñoa, Providencia, La Reina, Lo Barnechea, Vitacura y Las Condes.

* 1. **Definiciones, acrónimos y abreviaciones**

|  |  |
| --- | --- |
| **ACRÓNIMO** | **DESCRIPCIÓN** |
| ***API*** | Interfaz de Programación de Aplicaciones. |
| ***Dataset*** | Conjunto de datos de propiedades inmobiliarias. |
| ***Endpoint*** | Punto de acceso a un servicio de la API. |
| ***Modelo Predictivo*** | Algoritmo de Machine Learning utilizado para estimar valores futuros de propiedades. |
| ***OAuth*** | Protocolo de autenticación para proteger las APIs. |
| ***CSV (Comma Separated Values)*** | Formato de archivo que almacena datos tabulares en texto plano, separados por comas. |
| ***AI (Artificial Intelligence)*** | Inteligencia Artificial, rama de la informática que busca simular procesos de inteligencia humana mediante algoritmos y modelos matemáticos. |
| ***ML (Machine Learning)*** | Aprendizaje Automático, subcampo de la inteligencia artificial que permite a los sistemas aprender y mejorar automáticamente a partir de la experiencia. |
| ***API Gateway*** | Servicio que actúa como puerta de entrada para las API, gestionando el tráfico y controlando el acceso. |
| ***API Keys*** | Claves de autenticación utilizadas para acceder a una API y verificar la identidad de los usuarios. |
| ***Frontend*** | Parte del software que interactúa directamente con el usuario final, generalmente a través de una interfaz gráfica. |
| ***Backend*** | Parte del software que maneja la lógica de negocio, el procesamiento de datos y la interacción con la base de datos. |
| ***Django*** | Framework en Python. |
| ***Google Cloud Storage*** | Servicio de almacenamiento en la nube de Google para guardar y recuperar archivos, utilizado para almacenar datasets. |
| ***S3 (Simple Storage Service)*** | Servicio de almacenamiento en la nube de AWS, utilizado para almacenar y recuperar grandes cantidades de datos. |
| ***Cloud Run*** | Servicio de Google Cloud que permite ejecutar contenedores de forma serverless, ajustando la escalabilidad automáticamente. |
| ***Lambda*** | Servicio de AWS que permite ejecutar código sin necesidad de gestionar servidores (serverless), escalando automáticamente según la demanda. |
| ***GKE (Google Kubernetes Engine)*** | Servicio de Google Cloud para gestionar y orquestar contenedores en la nube. |
| ***ECS (Elastic Container Service)*** | Servicio de AWS para ejecutar y gestionar contenedores Docker en clústeres. |
| ***JSON (JavaScript Object Notation)*** | Formato ligero de intercambio de datos que utiliza texto legible por humanos para estructurar información. |
| ***HTTPS (Hypertext Transfer Protocol Secure)*** | Protocolo de comunicación segura en la web mediante el cifrado de los datos transferidos. |
| ***Python*** | Lenguaje de programación. |
| ***Scraping*** | Técnica de extracción de información de sitios web para obtener datos, generalmente mediante scripts o herramientas automatizadas. |
| ***UX (User Experience)*** | Experiencia de usuario, la forma en que los usuarios perciben la interacción con un producto o servicio. |
| ***KPIs (Key Performance Indicators)*** | Indicadores clave de rendimiento que miden el éxito de un producto o servicio en base a objetivos específicos. |
| ***Escalabilidad*** | Capacidad de un sistema para manejar un aumento de carga mediante la adición de recursos. |
| ***Alta Disponibilidad*** | Capacidad de un sistema para funcionar de manera continua sin fallos o tiempos de inactividad. |
| ***SLA (Service Level Agreement)*** | Acuerdo de nivel de servicio entre un proveedor de servicios y un cliente, especificando las expectativas de disponibilidad y rendimiento. |
| ***PaaS (Platform as a Service)*** | Plataforma como servicio, un modelo de computación en la nube que proporciona una plataforma para el desarrollo y ejecución de aplicaciones sin preocuparse por la infraestructura subyacente. |

# Referencias

* Documento de Casos de Uso
* Documento de Especificación de Requisitos
* Acta de Constitución del Proyecto

# Resumen ejecutivo

Este documento describe los escenarios de uso para el sistema DataHome, un servicio basado en inteligencia artificial que proporciona predicciones sobre el valor de propiedades inmobiliarias y análisis de precios. Los módulos principales del sistema incluyen:

* Predicción de precios de propiedades.
* Consulta de estadísticas de precios por comuna (mínimos, máximos, promedio, etc.).
* Reportes completos de las propiedades por comuna.

Los actores principales involucrados son los clientes inmobiliarios, corredores de propiedades, y administradores de la plataforma, quienes interactúan con la API para obtener los datos necesarios.

# Representación

La arquitectura del sistema *DataHome* se centra en la integración de datos inmobiliarios de la zona oriente de Santiago, con el objetivo de ofrecer predicciones de precios, análisis estadísticos, y visualizaciones que faciliten la toma de decisiones en el mercado inmobiliario. Para ello, el sistema emplea una representación modular y escalable. A continuación, se detalla cómo se representa cada parte del sistema, desde la entrada de datos hasta la visualización de resultados.

**Representación de los Datos**

Los datos manejados por el sistema provienen de múltiples fuentes relacionadas con el mercado inmobiliario, principalmente a través de un proceso de *web scraping* de sitios web de propiedades y la incorporación de Dataset históricos proporcionados por entidades inmobiliarias. Estos datos contienen información sobre:

* Precios de venta y arriendo.
* Características de las propiedades (superficie, número de dormitorios, baños, ubicación, etc.).
* Ubicación geográfica (coordenadas) de las propiedades.
* Tiempos históricos para análisis temporal.

**Estructura del Dataset**:

Los datos se almacenan en un sistema de base de datos relacional, utilizando Excel, con tablas diseñadas para relacionar las propiedades con sus características y valores a lo largo del tiempo. Cada registro incluye identificadores únicos, atributos de las propiedades y precios históricos o actuales.

**Representación de los Servicios**

El sistema está diseñado en una arquitectura de microservicios desplegada sobre **Google Cloud Run**, lo que permite el manejo de contenedores con una escalabilidad automática basada en la demanda de consultas a la API. Cada servicio tiene una responsabilidad bien definida:

1. **Servicio de Predicción de Precios**: Este componente recibe solicitudes con características de una propiedad (ubicación, número de habitaciones, superficie, etc.) y devuelve una predicción del precio estimado utilizando un modelo de regresión lineal entrenado en datos históricos.
2. **Servicio de Análisis de Distribución de Precios por Comuna**: Procesa solicitudes para mostrar la distribución de precios de las propiedades en una comuna específica, facilitando la comparación entre comunas y la identificación de precios fuera de lo común.
3. **Servicio de Reportes y Estadísticas**: Genera reportes con análisis estadísticos detallados (media, mediana, máximo, mínimo) sobre los precios de propiedades en una comuna determinada. Los reportes pueden ser descargados en formato PDF o Excel, orientados a analistas inmobiliarios y corredores.

**Representación de la Lógica del Negocio**

La lógica del negocio de *DataHome* se basa en la personalización y segmentación de los datos inmobiliarios según las necesidades de cada tipo de usuario. Los diferentes módulos incluyen:

1. **Histórico de Precios por Comuna**: Utilizando los datos temporales, el sistema puede generar gráficos de líneas que muestran las tendencias de precios en cada comuna a lo largo del tiempo, ayudando a los usuarios a entender la evolución del mercado.
2. **Comparativa de Precios por Características**: Se implementan análisis de dispersión (boxplots) para visualizar cómo varía el precio de una propiedad en función de características como el número de dormitorios, baños o la superficie.
3. **Predicción de Precios Futuros**: A partir del histórico de precios y un modelo predictivo, se genera una proyección de los precios futuros para una comuna específica, útil para inversores y compradores que buscan tendencias a largo plazo.

**Representación del Despliegue**

Para el proyecto **DataHome**, se utilizará una arquitectura basada en microservicios, que permite una mayor escalabilidad, mantenibilidad y flexibilidad. A continuación, se detalla la estructura de despliegue del sistema:

1. **Componentes del Sistema**

* **API RESTful**: Desplegada en **Google Cloud Run**, que permite el escalado automático y el manejo eficiente de las solicitudes.
* **Modelo Predictivo**: Implementado como un **endpoint** en **Google AI Platform**, permitiendo la ejecución de inferencias sobre datos de propiedades.
* **Archivo CSV**: Los datos de propiedades se almacenarán y gestionarán en un archivo CSV ubicado en Google Cloud Storage, desde donde se leerán para alimentar el modelo predictivo y realizar análisis.
* **Almacenamiento de Datos**: Los datasets y resultados generados serán almacenados en **Google Cloud Storage**, asegurando accesibilidad y seguridad.

**2. Arquitectura de Despliegue**

La arquitectura de despliegue del sistema está representada de la siguiente manera:



**3.Seguridad y Control de Acceso**

El acceso a la API y al modelo predictivo estará asegurado mediante **Google API Gateway**, que gestionará las autenticaciones con OAuth y claves API. Además, se implementarán políticas de control de acceso para proteger los datos sensibles almacenados en la base de datos y en el almacenamiento en la nube.

**4.Alta Disponibilidad y Escalabilidad**

La arquitectura propuesta garantiza alta disponibilidad mediante la distribución de recursos en múltiples zonas dentro de Google Cloud. **Cloud Run** y **Cloud SQL** están configurados para escalar automáticamente, asegurando que el sistema pueda manejar picos de tráfico y ofrecer un rendimiento óptimo.

# 5. Metas y Restricciones de la Arquitectura

**Metas de la arquitectura**

La arquitectura de la API de *DataHome* está diseñada para cumplir con una serie de metas clave, las cuales se alinean con los objetivos del proyecto y las necesidades de los usuarios finales. Estas metas guiarán el diseño, desarrollo y despliegue de la API, asegurando que esta sea eficiente, escalable y de fácil mantenimiento.

1. **Escalabilidad**: La API debe ser capaz de manejar un volumen variable de solicitudes de manera eficiente, especialmente en situaciones de alta demanda. Para garantizar la escalabilidad, la API se desplegará en un entorno serverless como **Google Cloud Run**, lo que permitirá el ajuste automático de recursos en función del tráfico. Esto evitará cuellos de botella y tiempos de respuesta lentos cuando el número de usuarios crezca.
2. **Bajo costo de operación**: La API debe ser económica de operar y mantener, optimizando el uso de recursos y reduciendo costos innecesarios. Al emplear tecnologías como **Google Cloud Run**, el costo operativo se ajustará al uso real, permitiendo pagar solo por el tiempo en que la API esté procesando solicitudes. Esto es particularmente importante para mantener la rentabilidad del proyecto a largo plazo.
3. **Disponibilidad y alta confiabilidad**: La API debe ser altamente disponible, con un tiempo de inactividad mínimo. Para lograr esto, se utilizarán prácticas de despliegue en múltiples zonas dentro de **Google Cloud**, garantizando la disponibilidad de la API incluso en caso de fallos en alguna región. Esto asegura un servicio continuo y confiable para los usuarios.
4. **Rendimiento óptimo**: La API debe proporcionar tiempos de respuesta rápidos, garantizando una experiencia fluida para los usuarios finales. Las consultas a la base de datos y las predicciones del modelo de precios de propiedades deben ser procesadas de manera eficiente, utilizando tecnologías como **Excel** para el dataset y un modelo de machine learning entrenado en la plataforma **AI Platform** de Google Cloud.
5. **Seguridad**: La protección de los datos y la privacidad de los usuarios son una prioridad. La API debe cumplir con los estándares de seguridad más estrictos, utilizando **API Gateway** y **OAuth** para controlar el acceso a los endpoints y proteger la información sensible. Además, se implementarán medidas como el cifrado de datos en tránsito y en reposo, así como la autenticación de usuarios y el control de acceso basado en roles.
6. **Mantenibilidad y facilidad de actualización**: La arquitectura de la API debe ser modular y de fácil mantenimiento, permitiendo actualizaciones y mejoras sin afectar el funcionamiento del sistema. Se seguirá una estructura de microservicios que permitirá la modificación independiente de cada componente de la API, reduciendo el tiempo de inactividad durante las actualizaciones y facilitando la incorporación de nuevas funcionalidades.
7. **Flexibilidad y extensibilidad**: La API debe ser flexible para adaptarse a cambios en los requisitos del negocio o a la incorporación de nuevas funcionalidades. La arquitectura debe ser lo suficientemente extensible como para integrar nuevas fuentes de datos, modelos de predicción y funcionalidades adicionales en el futuro, como la predicción de precios ajustados por inflación o la generación automática de reportes.
8. **Compatibilidad con diferentes plataformas**: La API debe ser accesible desde diferentes plataformas y dispositivos, lo que implica que debe cumplir con los estándares REST y ofrecer respuestas en formatos como JSON. Esto facilitará la integración con aplicaciones web, móviles y de terceros, permitiendo su uso por parte de distintos tipos de usuarios, como corredores de propiedades, analistas inmobiliarios y compradores interesados.
9. **Documentación clara y accesible**: La API debe contar con una documentación detallada y accesible, lo que permitirá a los desarrolladores integrar fácilmente en sus propias aplicaciones.

**Restricciones de la Arquitectura**

La arquitectura de la API de *DataHome* estará sujeta a una serie de restricciones que deberán ser consideradas a lo largo de su diseño, implementación y despliegue. A continuación, se detallan las principales restricciones:

1. **Presupuesto limitado**: Dado que el proyecto de *DataHome* tiene un presupuesto limitado, las decisiones tecnológicas deben enfocarse en maximizar el uso eficiente de los recursos financieros. Esto implica optar por soluciones en la nube de **bajo costo**, como **Google Cloud Run**, que ofrecen escalabilidad bajo demanda y facturación basada en uso, minimizando los costos operativos y evitando gastos innecesarios en infraestructura.
2. **Dependencia de servicios Cloud**: La API estará alojada en la infraestructura de Google Cloud, lo que significa que se depende de los servicios específicos de esta plataforma, como **Cloud Run** para el despliegue, **Excel** para la base de datos y **AI Platform** para el modelo predictivo. Esta dependencia puede limitar la portabilidad de la API a otras plataformas en el futuro, y cualquier cambio en los precios o políticas de Google Cloud afectará directamente los costos y la disponibilidad de la API.
3. **Disponibilidad de datos**: La API de *DataHome* depende de la disponibilidad y calidad de los datos inmobiliarios. Si los datos que se utilizan para entrenar el modelo predictivo o realizar consultas no están actualizados o son insuficientes, la precisión de las predicciones y análisis puede verse comprometida.
4. **Tiempo limitado para el desarrollo**: El cronograma del proyecto está acotado por fechas específicas, como el despliegue de la API el 4 de noviembre de 2024. Esta restricción de tiempo implica que el desarrollo debe ser ágil y centrarse en las funcionalidades críticas, priorizando las características que proporcionen el mayor valor a corto plazo, dejando las mejoras y nuevas funcionalidades para futuras versiones.
5. **Recursos humanos disponibles**: El equipo de desarrollo está compuesto por tres personas clave: **Alex Baeza** como Product Owner, **Constanza Vilaza** como Scrum Máster y QA, y **Marco Puga** como desarrollador Back-End. El tamaño reducido del equipo limita la capacidad para abordar múltiples tareas en paralelo, lo que exige una gestión eficiente de las responsabilidades y un enfoque en la priorización de tareas durante el desarrollo de la API.
6. **Rendimiento limitado del modelo predictivo**: El modelo de predicción de precios está basado en un algoritmo de **regresión lineal**, lo que, aunque eficiente en términos de velocidad, puede tener limitaciones en cuanto a la precisión para captar relaciones más complejas en los datos inmobiliarios. Esta restricción en el tipo de modelo utilizado puede afectar la exactitud de las predicciones para ciertos casos, especialmente cuando se presenten outliers o comportamientos atípicos en el mercado, aunque estas restricciones pueden ser tratadas en versiones futuras del proyecto.
7. **Limitación en el tiempo de respuesta**: Dado que la API necesita ofrecer predicciones y consultas en tiempo real, las solicitudes no deben superar un tiempo de respuesta definido de **500 ms a 1 segundo** en la mayoría de los casos. Este límite puede verse afectado por la carga de tráfico en momentos de alta demanda, lo que requerirá medidas adicionales de optimización y balanceo de carga.
8. **Seguridad y cumplimiento de normativas**: La API debe cumplir con las normativas de protección de datos y privacidad, lo que impone restricciones en cuanto al manejo, almacenamiento y transmisión de información sensible. Se requerirá el uso de **OAuth** para la autenticación de usuarios y la protección de los endpoints, además del cifrado de datos en tránsito y en reposo para garantizar la seguridad de la información.
9. **Compatibilidad con sistemas heredados**: Los clientes potenciales de *DataHome*, como corredores de propiedades y empresas inmobiliarias, pueden utilizar sistemas heredados que deben integrarse con la API. Esto impone restricciones en cuanto a la compatibilidad de formatos de respuesta (como **JSON** o **CSV**) y el diseño de los endpoints, que deben ser lo suficientemente flexibles para soportar estas integraciones sin generar fricciones.
10. **Alcance geográfico limitado**: Inicialmente, la API solo brindará soporte para propiedades ubicadas en las comunas de Ñuñoa, Providencia, La Reina, Lo Barnechea, Vitacura y Las Condes. Esto restringe la utilidad del sistema a usuarios que operen en estas zonas, lo que podría limitar la expansión del mercado en etapas tempranas del proyecto. La limitante geográfica puede ser cambiada en futuras versiones donde se puedan incluir más comunas de la región Metropolitana en el análisis.

# Otros antecedentes y consideraciones

A lo largo del desarrollo del proyecto de *DataHome*, se deben considerar algunos antecedentes clave que, si bien no encajan estrictamente en las restricciones o metas, son esenciales para consideraciones futuras y su desarrollo:

1. **Evolución del Modelo Predictivo**: A medida que se recopilan más datos y se ajustan las predicciones, el modelo actual podría ser reemplazado por algoritmos más avanzados. Esto permitiría mejorar la precisión y cubrir un mayor rango de escenarios, adaptándose a cambios en el mercado inmobiliario.
2. **Posible Expansión Geográfica**: Aunque inicialmente se centra en comunas específicas de la zona oriente de Santiago, existe la posibilidad de escalar la API para cubrir otras comunas y regiones. Esta expansión requerirá ajustes en el scraping de datos y la base de datos, así como nuevas optimizaciones en el modelo predictivo.
3. **Integración con Terceros**: La API debe considerar futuras integraciones con otras plataformas inmobiliarias o sistemas de corredores, lo que implica tener una arquitectura flexible y modular, con capacidad para exponer nuevas funcionalidades mediante endpoints adicionales.
4. **Monitorización y Mantenimiento**: Se deben implementar mecanismos de monitorización continua del rendimiento, tanto del modelo predictivo como del servicio en sí. Esto permitirá detectar cuellos de botella, fallos en tiempo real y realizar ajustes proactivos en la infraestructura.