Texto

Descripción generada automáticamente

Universidad Internacional de La Rioja

Escuela Superior de Ingeniería y Tecnología

Grado en Ingeniería Informática

Módulo de cálculo de presupuestos para viajes en moto

|  |  |
| --- | --- |
| Trabajo fin de estudio presentado por: | Carlos Gabriel Lora Espinosa |
| Director/a: | Carlos Gregorio Tolmo |
| Fecha: | 9 de julio de 2025 |
| Repositorio del código fuente: |  |

Resumen

El presente Trabajo Fin de Grado aborda la carencia de herramientas tecnológicas especializadas en la presupuestación integral de viajes en motocicleta, detectada tras analizar el panorama actual de aplicaciones para la planificación de rutas. Si bien estas aplicaciones ofrecen funcionalidades avanzadas para la definición de itinerarios, la estimación económica detallada de los costes asociados al moto-turismo brilla por su ausencia.

Con el objetivo de cubrir este vacío, se ha diseñado, desarrollado e implementado un Producto Mínimo Viable (PMV) de un módulo de cálculo presupuestario especializado. Este módulo, concebido para su integración con aplicaciones cliente existentes a través de una API, busca enriquecer la experiencia del usuario al proporcionar una visión económica completa durante la fase de planificación del viaje.

El desarrollo del proyecto ha comprendido las etapas de análisis de requisitos, diseño de la arquitectura y la lógica de cálculo, la implementación técnica de la solución y la API de integración, y finalmente, la validación y verificación mediante pruebas funcionales y el análisis de la precisión de los presupuestos generados.

La principal contribución de este trabajo reside en la creación de un módulo de presupuestación específico para viajes en moto, que considera tanto los gastos comunes como aquellos específicos del moto-turismo. Esta funcionalidad complementaria aporta un valor añadido significativo a las herramientas de planificación existentes, facilitando una gestión económica más eficiente para la comunidad motorista.

**Palabras Clave:** Presupuestos de viajes en moto, moto-turismo, estimación costes viaje moto, desarrollo software moto, integración vía API, módulo API presupuestos.

Abstract

This Final Degree Project addresses the identified lack of specialized technological tools for the comprehensive budgeting of motorcycle trips, following an analysis of the current landscape of route planning applications. While these applications offer advanced functionalities for itinerary definition, detailed economic estimation of the costs associated with motorcycle tourism is a missing feature.

To bridge this gap, a Minimum Viable Product (MVP) of a specialized budget calculation module has been designed, developed, and implemented. This module, intended for integration with existing client applications through an API, aims to enhance the user experience by providing a complete economic overview during the trip planning phase.

The project development encompassed the stages of requirements analysis, architecture and calculation logic design, technical implementation of the solution and the integration API, and finally, validation and verification through functional tests and analysis of the generated budgets' accuracy.

The main contribution of this work lies in the creation of a specific budgeting module for motorcycle trips, which considers both common expenses and those unique to motorcycle tourism. This complementary functionality provides significant added value to existing planning tools, facilitating more efficient economic management for the motorcycle community.

**Keywords**: Motorcycle trip budgets, Motorcycle tourism, Motorcycle trip cost estimation, Motorcycle software development, API integration, Budget API module.

Índice de contenidos

[1. Introducción 1](#_Toc202635745)

[1.1. Motivación 2](#_Toc202635746)

[1.2. Planteamiento del trabajo 2](#_Toc202635747)

[1.3. Estructura del trabajo 3](#_Toc202635748)

[2. Contexto y Estado del Arte 5](#_Toc202635749)

[2.1. Contexto 5](#_Toc202635750)

[2.2. Estado del Arte 7](#_Toc202635751)

[2.2.1. Aplicaciones especializadas en planificación de rutas para motoristas 8](#_Toc202635752)

[2.2.2. Aplicaciones especializadas en gestión económica 11](#_Toc202635753)

[3. Objetivos y metodología de trabajo 15](#_Toc202635754)

[3.1. Objetivo general 15](#_Toc202635755)

[3.2. Objetivos específicos 16](#_Toc202635756)

[3.3. Metodología de trabajo 17](#_Toc202635757)

[4. Análisis y definición del sistema a desarrollar 20](#_Toc202635758)

[4.1. Identificación y clasificación de conceptos y gastos de viajes en moto 21](#_Toc202635759)

[4.1.1. Clasificación según la naturaleza del concepto 21](#_Toc202635760)

[4.1.2. Clasificación según el tipo de flujo 22](#_Toc202635761)

[4.1.3. Clasificación según la especificidad del gasto 22](#_Toc202635762)

[4.2. Definición de requisitos funcionales y no funcionales 24](#_Toc202635763)

[4.3. Diagrama de casos de uso 25](#_Toc202635764)

[4.4. Descripción del flujo típico y del flujo alternativo de casos de uso 26](#_Toc202635765)

[4.4.1. Enviar datos a través de API 26](#_Toc202635766)

[4.4.2. Calcular presupuestos 27](#_Toc202635767)

[4.4.3. Procesar datos del viaje 28](#_Toc202635768)

[4.4.4. Generar formatos de salida 29](#_Toc202635769)

[4.4.5. Recibir resultados a través de API 29](#_Toc202635770)

[4.5. Diseño de la arquitectura de la solución 30](#_Toc202635771)

[4.6. Establecimiento del alcance del desarrollo 32](#_Toc202635772)

[5. Diseño y desarrollo técnico 34](#_Toc202635773)

[5.1. Herramientas utilizadas 34](#_Toc202635774)

[5.2. Implementación de la lógica de presupuestación 36](#_Toc202635775)

[5.2.1. Importación de módulos necesarios 36](#_Toc202635776)

[5.2.2. Funciones de validación de formatos 37](#_Toc202635777)

[5.2.3. Generación de formatos de resultados 38](#_Toc202635778)

[5.2.4. Procesamiento del presupuesto 40](#_Toc202635779)

[5.3. Implementación de la API de integración 42](#_Toc202635780)

[5.3.1. Importación de módulos necesarios 42](#_Toc202635781)

[5.3.2. Formatos de respuesta 43](#_Toc202635782)

[5.3.3. Endpoint Health Check 44](#_Toc202635783)

[5.3.4. Endpoint principal 44](#_Toc202635784)

[6. Validación y Verificación 46](#_Toc202635785)

[6.1. Pruebas funcionales del sistema 46](#_Toc202635786)

[6.1.1. Pruebas funcionales de la lógica de negocio calculos.py 47](#_Toc202635787)

[6.1.2. Pruebas funcionales de la interfaz de comunicación api.py 52](#_Toc202635788)

[6.2. Análisis de precisión de presupuestos 56](#_Toc202635789)

[6.2.1. Pruebas de Casos Reales 58](#_Toc202635790)

[6.2.2. Pruebas con Datos Extremos 60](#_Toc202635791)

[6.3. Integración con aplicaciones existentes 67](#_Toc202635792)

[6.3.1. Endpoints Disponibles 67](#_Toc202635793)

[6.3.2. Entrada y salida de datos 68](#_Toc202635794)

[6.3.3. Validación de Datos y Formatos Esperados 69](#_Toc202635795)

[6.3.4. Gestión de Errores y Excepciones 70](#_Toc202635796)

[7. Conclusiones y trabajo futuro 72](#_Toc202635797)

[7.1. Conclusiones del trabajo 72](#_Toc202635798)

[7.2. Líneas de trabajo futuro 74](#_Toc202635799)

[Referencias bibliográficas 77](#_Toc202635800)

[Anexo A. Comparativa de aplicaciones específicas para planificar rutas en moto 1](#_Toc202635801)

[Anexo B. Detalle de los gastos incluidos en el proyecto 1](#_Toc202635802)

[Anexo C. Detalle de los conceptos informativos incluidos en el proyecto. 6](#_Toc202635803)

[Anexo D. Requisitos Funcionales 9](#_Toc202635804)

[Anexo E. Requisitos no funcionales 12](#_Toc202635805)

[Anexo F. Archivos utilizados en las pruebas de análisis de precisión de resultados 13](#_Toc202635806)

[Índice de acrónimos 31](#_Toc202635807)

Índice de figuras

[Figura 1. Evolución de los sistemas de orientación y navegación por carretera. 5](#_Toc202635808)

[Figura 2. Esquema de los tipos de aplicaciones analizadas 7](#_Toc202635809)

[Figura 3. Diagrama de funcionalidades de las aplicaciones especializadas en planificación de rutas para motoristas. 8](#_Toc202635810)

[Figura 4. Imágenes de las aplicaciones: TravelSpend, Trabee y Moneyfy 11](#_Toc202635811)

[Figura 5. Imágenes de las aplicaciones: Kayak, Skyscanner y Hopper. 12](#_Toc202635812)

[Figura 6. Imágenes de las aplicaciones Tricount y Splitwise. 13](#_Toc202635813)

[Figura 7. Imágenes de las aplicaciones: Drivvo, Fuelio, Spritmonitor y ViaMichelin. 14](#_Toc202635814)

[Figura 8.Representación de la metodología Scrum 17](#_Toc202635815)

[Figura 9. Esquema de la metodología Scrum adaptada al trabajo individual. 19](#_Toc202635816)

[Figura 10. Esquema de requisitos funcionales y no funcionales. 24](#_Toc202635817)

[Figura 11. Diagrama de Casos de Uso del Módulo presupuestario. 25](#_Toc202635818)

[Figura 12. Esquema de la arquitectura de la solución. 30](#_Toc202635819)

[Figura 13. Ejemplo de código: Importación del módulo datetime. 37](#_Toc202635820)

[Figura 14. Código de la función validar\_fecha(fecha\_str). 38](#_Toc202635821)

[Figura 15.Función generar\_etiqueta\_viaje. 39](#_Toc202635822)

[Figura 16. Cabecera de la función generar\_resumen\_viaje. 39](#_Toc202635823)

[Figura 17. Cabecera de la función generar\_detalle\_viaje. 40](#_Toc202635824)

[Figura 18. Muestra de algunos cálculos de la función procesar\_presupuestovm. 41](#_Toc202635825)

[Figura 19. Generación de respuestas de la función procesar\_presupuestovm. 41](#_Toc202635826)

[Figura 20. Importación de módulos en api.py. 43](#_Toc202635827)

[Figura 21. Clase FormatoRespuesta en api.py. 43](#_Toc202635828)

[Figura 22. Código del endpoint Health Check en api.py. 44](#_Toc202635829)

[Figura 23. Código del endpoint principal en api.py. 45](#_Toc202635830)

[Figura 24. Fragmento del código de la función test\_validar\_numero. 48](#_Toc202635831)

[Figura 25. Código de la función test\_validar\_texto. 48](#_Toc202635832)

[Figura 26. Código de la función test\_valdiar\_fecha. 49](#_Toc202635833)

[Figura 27. Código de la función\_test\_total\_general\_viaje 51](#_Toc202635834)

[Figura 28. Código de test\_api.py. 52](#_Toc202635835)

[Figura 29. Fragmento de código de datos\_validos.json. 53](#_Toc202635836)

[Figura 30. Código de datos\_invalidos.json con marcas en los datos inválidos. 54](#_Toc202635837)

[Figura 31. Fragmento de código de conftest.py. 55](#_Toc202635838)

[Figura 32. Fragmento del resultado positivo de las pruebas realizadas mostrado por pantalla. 55](#_Toc202635839)

[Figura 33. Esquema de las pruebas de precisión de presupuestos realizadas. 57](#_Toc202635840)

[Figura 34. Captura de la hoja de cálculo con el resultado en formato Etiqueta de la prueba A3. 59](#_Toc202635841)

[Figura 35. Captura de la hoja de cálculo con el formato de resultado Resumen del viaje de 9 etapas. 60](#_Toc202635842)

[Figura 36. Captura de la hoja de cálculo con el resultado en formato Etiqueta de la prueba B1. 61](#_Toc202635843)

[Figura 37. Captura de la hoja de cálculo con el resultado en formato de Etiqueta de la prueba B3. 63](#_Toc202635844)

[Figura 39. Fragmento del archivo JSON ENTRADA Prueba B6 con el nombre del viaje vacío. 64](#_Toc202635845)

[Figura 40. Captura del error generado por la prueba B6. 64](#_Toc202635846)

[Figura 43.Fragmento del archivo JSON ENTRADA Prueba B8 con la fecha en formato incorrecto. 65](#_Toc202635847)

[Figura 44. Captura del error producido por la ejecución de la prueba B8. 65](#_Toc202635848)

[Figura 87. Captura del código de respuesta en formato Etiqueta de la prueba B9 (Anexo F). 66](#_Toc202635849)

[Figura 45. Captura de pantalla de la Interfaz Swagger con los dos endpoints disponibles. 68](#_Toc202635850)

[Figura 46. Ejemplos de gestión de Excepciones en las funciones de validación. 71](#_Toc202635851)

[Figura 47. Esquema de las posibles mejoras a futuro. 74](#_Toc202635852)

[Figura 48. Composición código archivo P.A3 JSON ENTRADA Viaje 9 Etapas (Anexo F). 13](#_Toc202635853)

[Figura 49. Composición del código de la respuesta Completa de la prueba B3 Viaje de 9 Etapas (Anexo F). 14](#_Toc202635854)

[Figura 50. Captura del código de respuesta Resumen de la prueba A3 Viaje 9 Etapas (Anexo F). 15](#_Toc202635855)

[Figura 51. Captura del código de la respuesta Etiqueta de la prueba A3 Viaje 9 Etapas (Anexo F). 15](#_Toc202635856)

[Figura 52. Captura de la hoja de cálculo con los resultados en formato Etiqueta y Resumen de la prueba A3 Viaje 9 Etapas (Anexo F). 16](#_Toc202635857)

[Figura 53. Captura de la hoja de cálculo con el resultado Completo de la prueba A3 Viaje 9 Etapas (Anexo F). 17](#_Toc202635858)

[Figura 54. Composición del código del archivo JSON ENTRADA Prueba B1 Todos los datos informados (Anexo F). 18](#_Toc202635859)

[Figura 55. Composición del código de la respuesta Completa de la prueba B1 Todos los conceptos informados (Anexo F). 19](#_Toc202635860)

[Figura 56. Captura código de respuesta en formato Resumen de la prueba B1 Todos los conceptos informados (Anexo F). 20](#_Toc202635861)

[Figura 57. Captura del código de respuesta en formato Etiqueta de la prueba B1 Todos los conceptos informados (Anexo F). 20](#_Toc202635862)

[Figura 58. Captura de la hoja de cálculo con los resultados Etiqueta y Resumen de la prueba B1 Todos los conceptos informados. (Anexo F). 21](#_Toc202635863)

[Figura 59. Composición con la respuesta en formato Completo de la prueba B1 Todos los conceptos informados (Anexo F). 21](#_Toc202635864)

[Figura 60. Composición del código del archivo JSON ENTRADA Prueba B3 (Anexo F). 22](#_Toc202635865)

[Figura 61. Composición de la respuesta en formato completo de la prueba B3 (Anexo F). 22](#_Toc202635866)

[Figura 62. Captura de la respuesta en formato Resumen de la prueba B3 (Anexo F). 23](#_Toc202635867)

[Figura 63. Captura de la respuesta en formato Etiqueta de la prueba B3 (Anexo F). 23](#_Toc202635868)

[Figura 64. Captura de la hoja de cálculo con los resultados Etiqueta y Resumen de la prueba B3 (Anexo F). 24](#_Toc202635869)

[Figura 65. Composición de la hoja de cálculo con la respuesta Completa de la prueba B3 (Anexo F). 25](#_Toc202635870)

[Figura 78. Composición del código del archivo JSON ENTRADA Prueba B6 (Anexo F). 26](#_Toc202635871)

[Figura 79. Captura del código de la respuesta común a todos los formatos de la prueba B6 (Anexo F). 26](#_Toc202635872)

[Figura 82. Composición del código del archivo JSON ENTRADA Prueba B8 (Anexo F). 27](#_Toc202635873)

[Figura 83. Captura de la respuesta de la prueba B8 en todos los formatos de salida (Anexo F). 27](#_Toc202635874)

[Figura 84. Composición del código del archivo JSON ENTRADA Prueba B9 (Anexo F). 28](#_Toc202635875)

[Figura 85. Composición código de respuesta en formato Completo de la prueba B9 (Anexo F). 29](#_Toc202635876)

[Figura 86. Captura del código de respuesta en formato Resumen de la prueba B9 (Anexo F). 30](#_Toc202635877)

[Figura 87. Captura del código de respuesta en formato Etiqueta de la prueba B9 (Anexo F). 30](#_Toc202635878)

Índice de tablas

[Tabla 1. Caso de uso: Enviar datos a través de API 26](#_Toc202635879)

[Tabla 2. Caso de uso: Calcular presupuestos 27](#_Toc202635880)

[Tabla 3. Caso de uso: Procesar datos del viaje 28](#_Toc202635881)

[Tabla 4.Caso de uso: Generar formatos de salida 29](#_Toc202635882)

[Tabla 5. Caso de uso: Recibir resultados a través de API. 29](#_Toc202635883)

[Tabla 6. Campos obligatorios y tipos de datos esperados. 70](#_Toc202635884)

[Tabla 7.Comparativa de aplicaciones específicas para planificar rutas en moto (Anexo A). 1](#_Toc202635885)

[Tabla 8. Gastos incluidos en el proyecto (Anexo B). 1](#_Toc202635886)

[Tabla 9. Conceptos informativos (Anexo C). 6](#_Toc202635887)

# Introducción

El presente Trabajo Fin de Grado aborda un problema relacionado con el ámbito del moto-turismo. Se trata de la ausencia de herramientas tecnológicas especializadas en el cálculo detallado de presupuestos para viajes en moto. Si bien existen diversas aplicaciones enfocadas en la planificación y optimización de rutas para motoristas, éstas no integran funcionalidades específicas para la estimación económica integral, considerando tanto los gastos comunes como aquellos particulares del uso de la motocicleta.

Con el objetivo de subsanar esta carencia, este proyecto se ha centrado en el diseño, desarrollo e implementación de un Producto Mínimo Viable (PMV) de un **módulo de cálculo presupuestario especializado en viajes en moto**. Este módulo se concibe como un componente complementario para aplicaciones cliente ya existentes, al que se accede a través de una interfaz de programación de aplicaciones (API). La integración de esta funcionalidad tiene el potencial de enriquecer significativamente la experiencia de planificación del usuario, proporcionando una visión económica completa y facilitando la toma de decisiones previas al viaje.

El desarrollo de este proyecto ha seguido un procedimiento estructurado que ha abarcado las **fases** de análisis y definición de los requisitos del sistema, el diseño y desarrollo técnico de la lógica de presupuestación y la API de integración, así como la validación y verificación de la solución implementada mediante pruebas funcionales y el análisis de la precisión de los presupuestos generados.

La **principal contribución** de este trabajo radica en la propuesta e implementación de un módulo de presupuestación especializado que aborda un vacío funcional existente en el mercado de aplicaciones para el moto-turismo. La **capacidad de estimar de forma precisa y personalizada los costes de un viaje en moto**, teniendo en cuenta las particularidades de este tipo de desplazamiento, aporta un valor añadido significativo a las herramientas de planificación actuales y facilita una gestión económica más eficiente por parte de los usuarios.

## Motivación

El creciente interés en el moto-turismo ha puesto de manifiesto una limitación en las soluciones tecnológicas disponibles. Aunque las aplicaciones de planificación de rutas ofrecen opciones muy avanzadas de personalización de itinerarios, la dimensión económica de la planificación del viaje no ha recibido la misma atención.

El problema principal es la falta de herramientas especializadas que permitan a los motoristas estimar, de manera exhaustiva, los costes asociados a sus viajes.

Esta ausencia de soluciones integradas obliga a los usuarios a combinar varias aplicaciones independientes o a realizar cálculos manuales, procesos que son ineficientes y propensos a errores. La relevancia de la planificación presupuestaria, como factor clave en la viabilidad de un viaje, evidencia la necesidad de una solución.

La complejidad inherente a la presupuestación de viajes en moto, motivada por la diversidad de variables y la especificidad de ciertos costes, justifica el desarrollo de una solución tecnológica a medida. La motivación para este proyecto surge del **interés por aplicar los principios de la ingeniería informática en un ámbito con una demanda creciente y un claro potencial de mejora en la experiencia del usuario**.

## Planteamiento del trabajo

Este proyecto ha implicado el diseño, desarrollo y prueba de un módulo de presupuestación económica para viajes en motocicleta, concebido para su integración con aplicaciones especializadas en la planificación de rutas y viajes en moto. El objetivo principal es cubrir la carencia existente en herramientas de cálculo de gastos específicos para el moto-turismo, proporcionando una solución que permita a los usuarios obtener una estimación detallada y personalizada del coste total de sus viajes.

Para alcanzar este objetivo, se ha optado por el **desarrollo de un Producto Mínimo Viable (PMV)**. Este PMV se caracteriza por una lógica de cálculo presupuestario que considera los principales conceptos de gasto relevantes para los viajes en moto, incluyendo combustible, peajes, alojamiento, alimentación, equipamiento y mantenimiento. El software desarrollado se ha diseñado para interactuar con aplicaciones cliente a través de una API, facilitando la recepción de los parámetros necesarios (duración del viaje, distancia, preferencias de gasto) y la entrega de una estimación presupuestaria detallada como respuesta.

La solución propuesta se basa en una **arquitectura modular** que busca facilitar su integración con sistemas ya existentes. La lógica de cálculo se ha implementado de manera que pueda ser invocada por aplicaciones cliente, gestionando los datos de entrada y generando la información presupuestaria correspondiente. La API de integración se ha diseñado con criterios de robustez y flexibilidad para asegurar una comunicación eficiente y segura entre el módulo de presupuestación y las aplicaciones de planificación de rutas.

En esencia, el planteamiento de este proyecto se centra en la creación de un módulo especializado que, mediante su integración con aplicaciones cliente, resuelve la actual falta de herramientas de presupuestación específicas para viajes en moto, ofreciendo a los usuarios una funcionalidad esencial para la planificación integral de sus experiencias en carretera.

## Estructura del trabajo

El presente documento se organiza en los siguientes capítulos, los cuales detallan el proceso de desarrollo y los resultados obtenidos en el proyecto:

* **Contexto y Estado del Arte**: Este capítulo establece el marco del proyecto, analizando el panorama de las soluciones tecnológicas existentes para la planificación de viajes por carretera. Se identifica la diferenciación funcional de las aplicaciones actuales y la ausencia de una solución integrada para la presupuestación específica de viajes en moto.
* **Objetivos y Metodología de Trabajo**: En este capítulo se definen los objetivos generales y específicos que se persiguen con el desarrollo del proyecto. Asimismo, se describe y justifica la metodología de trabajo seleccionada, una adaptación de la metodología ágil Scrum al contexto del trabajo individual.
* **Análisis y Definición del Sistema a Desarrollar**: Este capítulo aborda la fase de análisis, detallando la identificación y clasificación de los gastos asociados al moto-turismo, la definición de los requisitos funcionales y no funcionales del sistema, el diseño de los casos de uso y sus flujos, el diseño de la arquitectura de la solución para su integración y la delimitación del alcance del desarrollo del PMV.
* **Diseño y Desarrollo Técnico de la Solución**: En este capítulo se describen los aspectos técnicos de la implementación, incluyendo las herramientas de desarrollo utilizadas y la explicación de la arquitectura y el funcionamiento de la lógica de presupuestación y de la API de integración con aplicaciones cliente.
* **Validación y Verificación**: Este capítulo se centra en el proceso de aseguramiento de la calidad de la solución desarrollada, presentando las pruebas funcionales realizadas, el análisis de la precisión de los presupuestos generados y la estrategia para la integración del módulo con aplicaciones ya existentes.
* **Conclusiones y Trabajos Futuros**: Este capítulo final resume las principales conclusiones del proyecto, destacando el logro de los objetivos y la contribución al ámbito del moto-turismo. Además, se proponen diversas líneas de trabajo futuro para la mejora y expansión de la funcionalidad del módulo desarrollado.

# Contexto y Estado del Arte

## Contexto

Profundizando en la problemática planteada en la introducción, nos encontramos con que el turismo en moto ha experimentado un notable crecimiento en los últimos años, impulsado por una comunidad de motoristas en expansión que busca experiencias de viaje más flexibles y aventureras que les permita disfrutar de la carretera, los paisajes, la gastronomía y la sensación de libertad que solo los viajes en moto te hacen sentir.

Afortunadamente, la tecnología aplicada a la planificación de viajes por carretera ha evolucionado notablemente en las últimas décadas en las que se ha pasado de los mapas impresos tradicionales a aplicaciones integradas en nuestros dispositivos móviles con infinidad de funciones que aportan información en tiempo real, pasando por los dispositivos o navegadores GPS (*Global Position System*) que hoy en día siguen siendo una muy buena opción como herramientas para el guiado y la planificación de rutas por carretera.

Los fabricantes de navegadores GPS también han desarrollado dispositivos especializados para su uso en motocicletas tanto a nivel de hardware, con dispositivos más resistentes y adaptados al tacto con guantes, como a nivel de software, desarrollando funcionalidades como la generación de rutas que permiten la selección del número de curvas y de desnivel.

Figura . Evolución de los sistemas de orientación y navegación por carretera.

Interfaz de usuario gráfica, Sitio web

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia. Basado en imágenes de pixabay: Epicioci. (17 de enero de 2021). <https://pixabay.com/es/photos/mapa-l%C3%A1piz-navegaci%C3%B3n-la-carretera-5911003/>), USA-Reiseblogger. (17 de julio de 2020). <https://pixabay.com/es/photos/auto-navegaci%C3%B3n-gps-veh%C3%ADculo-panel-5414514/> y freepik: Creativeart. (s.f.) <https://www.freepik.es/foto-gratis/movil-google-maps_926535.htm#fromView=image_search_similar&page=1&position=28&uuid=0c80a35e-d74b-41f2-8634-6df2fe7dca3b&query=navegador+GPS+smartphone>

Este proyecto se centra en las aplicaciones disponibles vía web o en aplicaciones móviles, por su mayor potencial respecto al desarrollo de funcionalidad relacionada con la planificación desde el punto de vista económico de un viaje en moto.

Aunque existen aplicaciones que facilitan la planificación de viajes y rutas en moto como, por ejemplo: *Calimoto, MyDrive*o*Rever*, estas aplicaciones se centran en el cálculo de las mejores rutas a seguir. Suelen contar con gran variedad de opciones como rutas más rápidas o cortas, con opción de evitar peajes o autovías, e incluso con la posibilidad de seleccionar trazados con más o menos curvas o desniveles. Sin embargo, dejan de lado el cálculo del presupuesto del viaje.

Teniendo en cuenta que el presupuesto de un viaje es fundamental para valorar su viabilidad y como parte importante de la planificación del mismo, resulta necesario que las aplicaciones especializadas en viajes en moto cuenten con esta funcionalidad.

Existe un vacío en el mercado en este sentido y, por lo tanto, sería de gran utilidad orientar este proyecto a cumplimentar las funcionalidades de otras aplicaciones especializadas en planificación de viajes en moto**,** integrando un módulo de presupuestos especializado que permita estimar el coste total del viaje en función de diversos parámetros personalizados por el usuario.

El público objetivo de esta herramienta son todos aquellos motoristas que deseen prever con detalle los costes de sus viajes tanto a nivel particular, para viajeros en solitario o para clubes y grupos de moteros, como a nivel profesional para agencias especializadas en turismo motero, que podrían incorporar la aplicación como un valor añadido para sus clientes, mejorando la experiencia de planificación de viajes.

La ausencia de herramientas especializadas para el cálculo de presupuestos en viajes en moto justifica plenamente el desarrollo de esta aplicación. Su enfoque en la especificidad de los costes propios del motociclismo y su potencial para complementar las plataformas de navegación existentes ofrecen una solución más completa y adaptada a las necesidades de la comunidad motera. Este proyecto, por tanto, se posiciona en un nicho de mercado con una clara utilidad práctica.

## Estado del Arte

El desarrollo de soluciones tecnológicas para el turismo en motocicleta se ha centrado, principalmente, en el desarrollo de aplicaciones de planificación y personalización de rutas para motoristas y en sistemas de seguridad, tanto para tratar de combatir los robos de estos vehículos como para localización y aviso en caso de accidente.

Por otro lado, existen aplicaciones orientadas a la gestión económica para el registro, análisis y control de: finanzas personales, contratación de viajes (vuelos y hoteles), compartición de gastos entre grupos de personas y para el control de los gastos en combustible y mantenimiento del vehículo.

A continuación, se incluye un análisis de las aplicaciones relacionadas con la planificación y presupuestación de viajes en el que se puede apreciar claramente la diferenciación de funcionalidad y la carencia o falta de aplicaciones que aúnen ambas funciones.

Figura . Esquema de los tipos de aplicaciones analizadas

Diagrama

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

### Aplicaciones especializadas en planificación de rutas para motoristas

En el mercado actual podemos encontrar aplicaciones especializadas en la **creación de rutas para motoristas** con múltiples opciones de personalización. Algunas de estas funcionalidades son genéricas o comunes a otros medios de transporte y otras funcionalidades son específicas para los motoristas.

Figura . Diagrama de funcionalidades de las aplicaciones especializadas en planificación de rutas para motoristas.

Diagrama

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

* Planificación de rutas:
  + Funciones genéricas:
    - **Planificador de rutas**: Suelen ofrecer varias opciones de mapas como vista satélite, relieve, tráfico, etc. Generan rutas desde un punto origen hasta un destino permitiendo incorporar puntos intermedios. Ofrecen trazados alternativos e información como distancias, tiempo estimado según el medio de trasporte, indicaciones paso a paso, etc.
    - **Puntos de interés**: Localizaciones recomendadas por su valor turístico, cultural o paisajístico.
    - **Selección de preferencias**: Se trata de criterios de exclusión o inclusión de elementos como: peajes, autovías o autopistas, túneles o caminos de tierra, que permiten personalizar el diseño de las rutas.
  + Funciones específicas:
    - **Rutas sinuosas o con curvas**: En moto, las curvas se asocian a diversión y por ello este tipo de aplicaciones permite generar rutas en las que los trazados que se priorizan son los trazados con más curvas.
    - **Rutas con desnivel**: Los trazados que pasan por zonas montañosas y puertos de montaña son trazados normalmente con desniveles y trazados con curvas, algunas muy cerradas. Estas aplicaciones permiten la selección de distintos grados de desniveles a incluir en la ruta.
* Disponibilidad de mapas y de rutas generadas:
  + Funciones genéricas:
    - **Base de datos de rutas**: La mayoría de estas aplicaciones ponen a disposición de los usuarios un amplio catálogo de rutas con información sobre localización, distancia, duración estimada, valoraciones, imágenes asociadas y comentarios.
    - **Registro de rutas**: Opción de grabar los recorridos realizados para ser reutilizados, editados o compartidos.
    - **Disponibilidad de mapas offline**: Esta es una característica muy importante ya que, en ocasiones, no se dispone de conexión a Internet y la posibilidad de utilizar los mapas de forma online es una ventaja respecto a las aplicaciones que solo trabajan en modo online.
* Comunidad:
  + Funcionalidad genérica:
    - **Noticias y/o Blogs**: Algunas aplicaciones ofrecen información a los usuarios de novedades, eventos, actividades o noticias relacionadas con la comunidad motera, el mundo del motor y de los viajes.
  + Funcionalidad específica:
    - **Comunidad Motera**: Funcionalidad orientada a la comunicación entre los usuarios mediante mensajes, foros y chats. Tratan de fomentar el uso de estas aplicaciones como redes sociales específicas para moteros.
    - **Compartición de información de rutas**: Permite la compartición de rutas con información y comentarios sobre las mismas.
    - **Compartición de ubicación a tiempo real**: Para avisar de la proximidad de otros usuarios de la aplicación y de la localización del propio usuario.
* Navegación:
  + Funcionalidad genérica:
    - **Navegación paso a paso**: Indicaciones de la ruta a seguir paso a paso: giros, desvíos, distancias, velocidad máxima permitida, etc.
    - **Aviso de radares**: Algunas aplicaciones incorporan una base de datos de localizaciones de radares fijos y zonas habituales de radares móviles, generando avisos para el motorista cuando se aproximan.
    - **Alertas de tráfico**: Se muestran mensajes sobre aglomeraciones de tráfico, aviso de accidentes y desvíos.
    - **Altitud**: Indicación de la altitud y de su variación a lo largo de la ruta.
  + Funcionalidad específica: Gracias a los sensores incorporados en los smartphones algunas aplicaciones incorporan estas funcionalidades que son apreciadas por los motoristas más extremos.
    - **Aceleración**: Indicación de la aceleración máxima experimentada a lo largo de la ruta.
    - **Ángulo de inclinación**: Indicación del ángulo máximo de inclinación experimentado durante la ruta.
    - **Alertas por caídas o accidentes**: En caso de accidente la aplicación puede emitir una alerta realizando una llamada de emergencia.
* Compatibilidad con otros sistemas:
  + Funcionalidad genérica:
    - **Importación y exportación de archivos GPX**: Muy útil a la hora de intercambiar rutas generadas por otras aplicaciones o dispositivos.

La mayoría de las aplicaciones analizadas ofrecen distintas versiones según el tipo de licencia contratada y a pesar de las numerosas funcionalidades con las que cuentan, ninguna permite calcular el presupuesto de las rutas y viajes.

En el anexo A. Comparativa de aplicaciones específicas para rutas en moto se incluye una tabla comparativa en la que se especifican las funcionalidades de las aplicaciones analizadas.

### Aplicaciones especializadas en gestión económica

Las aplicaciones disponibles en el mercado en relación con la gestión de gastos y presupuestos de viajes se pueden clasificar en cuatro grupos: gestión de finanzas y gastos propios, contratación de viajes, compartición de gastos entre varias personas y aplicaciones de gestión de costes de combustible y mantenimiento del vehículo.

#### Gestión de finanzas y gastos propios

Estas aplicaciones se especializan en el registro y control de los gastos de viajes. Las funcionalidades más frecuentes en este tipo de aplicaciones o herramientas de control financiero particular son las siguientes:

* Integración con las cuentas bancarias automatizando la introducción de datos y facilitando su actualización.
* Categorización de los gastos e ingresos.
* Análisis de los datos, generación de informes y gráficos de los gastos por categorías y por periodos de tiempo.
* Alertas o avisos según el límite de gastos configurado.

Ejemplos de este tipo de aplicaciones son: *TravelSpend*, *Trabee* *Pocket* o *Monefy*.

Figura . Imágenes de las aplicaciones: TravelSpend, Trabee y Moneyfy

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Fuente. Elaboración propia a partir de capturas de pantalla de las webs: <https://travel-spend.com/>, <https://trabeepocket.com/> y <https://monefy.com/>

#### Contratación de viajes

Son plataformas que permiten la contratación de vuelos, alojamientos, transporte, actividades culturales o de ocio y la realización de reservas en restaurantes.

Son muy útiles a la hora de comparar precios y buscar ofertas de viajes. Aunque proporcionan costes importantes en relación con los viajes en moto como alojamiento, actividades y restaurantes, no están especializadas en la planificación de viajes en moto ya que no contemplan la realización de rutas por carretera ni gastos de combustible ni conceptos específicos como la equipación necesaria o el mantenimiento.

Algunos ejemplos de este tipo de aplicaciones son: *Kayak*, *Skyscanner* o *Hopper*.

Figura . Imágenes de las aplicaciones: Kayak, Skyscanner y Hopper.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia a partir de capturas de pantalla de las webs: <https://www.kayak.es/mobile>, <https://play.google.com/store/apps/details?id=com.kayak.android&hl=es>, <https://www.skyscanner.es/movil.html> y <https://hopper.com/es?activeTab=stays>.

#### Compartición de gastos entre varias personas

Estas aplicaciones permiten registrar y repartir los gastos realizados entre grupos de personas como amigos, familiares o compañeros de viaje. Su funcionalidad principal es que permite calcular lo que cada uno debe pagar y a quién para que los gastos se dividan según se acuerde. *Tricount* y *Splitwise* son dos ejemplos de este tipo de aplicaciones.

Figura . Imágenes de las aplicaciones Tricount y Splitwise.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia a partir de capturas de pantalla de las páginas webs: <https://www.europapress.es/economia/noticia-espanoles-compartieron-tricount-mas-3300-millones-euros-2024-cerca-20-total-20250217102904.html>, <https://play.google.com/store/apps/details?id=com.tribab.tricount.android&hl=es>, <https://www.splitwise.com/> y <https://nucleo.news/2025/01/las-5-mejores-apps-para-mejorar-tus-finanzas-en-este-nuevo-ano/>

#### Gestión de costes de combustible y mantenimiento del vehículo

Este último grupo de aplicaciones son las que más se acercan a la funcionalidad objetivo de este proyecto ya que algunas como *GasAll*, *RoadTrippers* permiten la planificación de rutas o viajes por carretera, calcular una estimación del coste del combustible necesario e incluso incluir otros gastos como los peajes.

Otras aplicaciones como *Drivvo*, *Fuelio* y *Spritmonitor* se centran en el registro y control de los costes de combustible, selección de estaciones de servicio según sus precios y localización, registro de costes de mantenimiento y avisos de próximas revisiones. Son útiles para el registro y control de gastos relacionados con los vehículos pero no están desarrollados para el cálculo de presupuestos de viajes por carretera.

Una de las más completas es *ViaMichelin*, que también incluye la posibilidad de reservar alojamiento y restaurantes y sugiere puntos de interés cultural a lo largo de la ruta propuesta.

Figura . Imágenes de las aplicaciones: Drivvo, Fuelio, Spritmonitor y ViaMichelin.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia a partir de capturas de pantalla de las páginas webs: <https://www.drivvo.com/es-ES>, <https://play.google.com/store/apps/dev?id=8099138519625088613&hl=es_419>, <https://www.fuel.io/>, <https://www.spritmonitor.de/es/>, <https://play.google.com/store/apps/details?id=de.spritmonitor.smapp_mp&hl=es>, <https://www.viamichelin.es/> y <https://apps.apple.com/es/app/viamichelin-gps-radar-ruta/id443142682>.

Tras analizar las características funcionales de aplicaciones de este tipo se hace evidente la carencia de soluciones integradas que aborden ambos aspectos de forma conjunta, es decir, que permitan planificar tanto las rutas como el presupuesto de manera especializada para motoristas.

Este hecho implica que, hoy en día, la solución para planificar un viaje en moto de forma integral sea la combinación de varias aplicaciones, haciendo que esta labor sea más compleja y requiera más tiempo. Por lo que el desarrollo de una herramienta de planificación presupuestaria como complemento a la funcionalidad de diseño de las rutas personalizadas para motoristas supondría un importante valor añadido, convirtiéndose así en un elemento diferenciador.

# Objetivos y metodología de trabajo

En este apartado se definen los objetivos que se persiguen con el desarrollo del presente Trabajo Fin de Grado y la metodología utilizada para el desarrollo de la solución y la consecución de los objetivos planteados.

El objetivo principal es facilitar la planificación económica de viajes en moto mediante la creación de un módulo de presupuestación especializado que complemente a las aplicaciones de planificación de rutas ya existentes.

La metodología de trabajo utilizada para el desarrollo de este proyecto es una adaptación de Scrum al trabajo individual ya que resultan prioritarias la flexibilidad de priorización de las tareas y la agilidad en el avance del desarrollo, teniendo en cuenta la limitación del tiempo disponible para alcanzar los objetivos planteados.

## Objetivo general

El objetivo general de este proyecto es diseñar y desarrollar e implementar un Producto Mínimo Viable (PMV) de un módulo de presupuestación económica de viajes en moto que pueda integrarse con aplicaciones especializadas en planificación de rutas y viajes en moto, con el fin de cubrir el vacío actual en herramientas de cálculo de gastos específicos para el moto-turismo.

Este módulo de presupuestación está orientado a cumplimentar la funcionalidad de estas aplicaciones con una herramienta de presupuestación especializada que contemple todos los conceptos propios de un viaje en moto, de manera que permita ofrecer a los usuarios una herramienta de planificación completa e integral. Asimismo, la creación del PMV permitirá evaluar la propuesta de valor con usuarios potenciales, identificando sus necesidades reales, así como posibles mejoras o funcionalidades adicionales que puedan incorporarse en versiones futuras del producto.

La integración de este módulo presupuestario especializado representa, no solo una ampliación de funcionalidad, sino que también optimiza notablemente la experiencia del usuario al con solidar la planificación de viajes en una sola plataforma.

## Objetivos específicos

A continuación se definen los tres objetivos específicos necesarios para la consecución del objetivo general.

1. Analizar y definir el sistema a desarrollar: Este objetivo inicial es fundamental para sentar las bases del proyecto. En primer lugar, se busca **identificar y clasificar** de manera exhaustiva **todos los conceptos de gastos asociados a los viajes en moto** que deban ser tenidos en cuenta para la elaboración de los presupuestos, desde los elementos básicos como combustible y alojamiento, hasta costes más específicos del moto-turismo como equipación y mantenimiento.

En segundo lugar, se procederá a definición precisa de los **requisitos funcionales y no funcionales** del módulo, con el fin de establecer el alcance del sistema, según la definición de qué acciones y capacidades debe implementar el módulo presupuestario. Finalmente, se diseñará la **arquitectura de la solución** a desarrollar, estableciendo la estructura interna del módulo y su interacción con otros componentes externos.

1. **Desarrollar** los componentes necesarios como **la lógica de procesamiento** de los datos **y la API de integración** con otros sistemas. Este objetivo implica la construcción de la lógica central de procesamiento de datos, encargada de recibir la información del viaje facilitada por los usuarios y aplicar los algoritmos necesarios para calcular los presupuestos de manera precisa y adaptada a las necesidades. Paralelamente, se desarrollará la API que permitirá la comunicación fluida y segura entre el módulo de presupuestación y las aplicaciones cliente de forma que proporcione una experiencia de usuario cohesionada.
2. **Validar y verificar** la solución desarrollada mediante pruebas funcionales del sistema y el análisis de los resultados, comprobando la eficacia y precisión de los presupuestos resultantes. Este último objetivo es crucial para asegurar la **calidad y fiabilidad** del producto. Se llevarán a cabo pruebas funcionales exhaustivas para verificar que todas las funcionalidades del módulo operan según lo esperado y cumplen con los requisitos definidos. Además, se realizará un análisis riguroso de los resultados de los presupuestos comparándolos con estimaciones manuales o datos de referencia, para comprobar la eficacia y precisión de los cálculos.

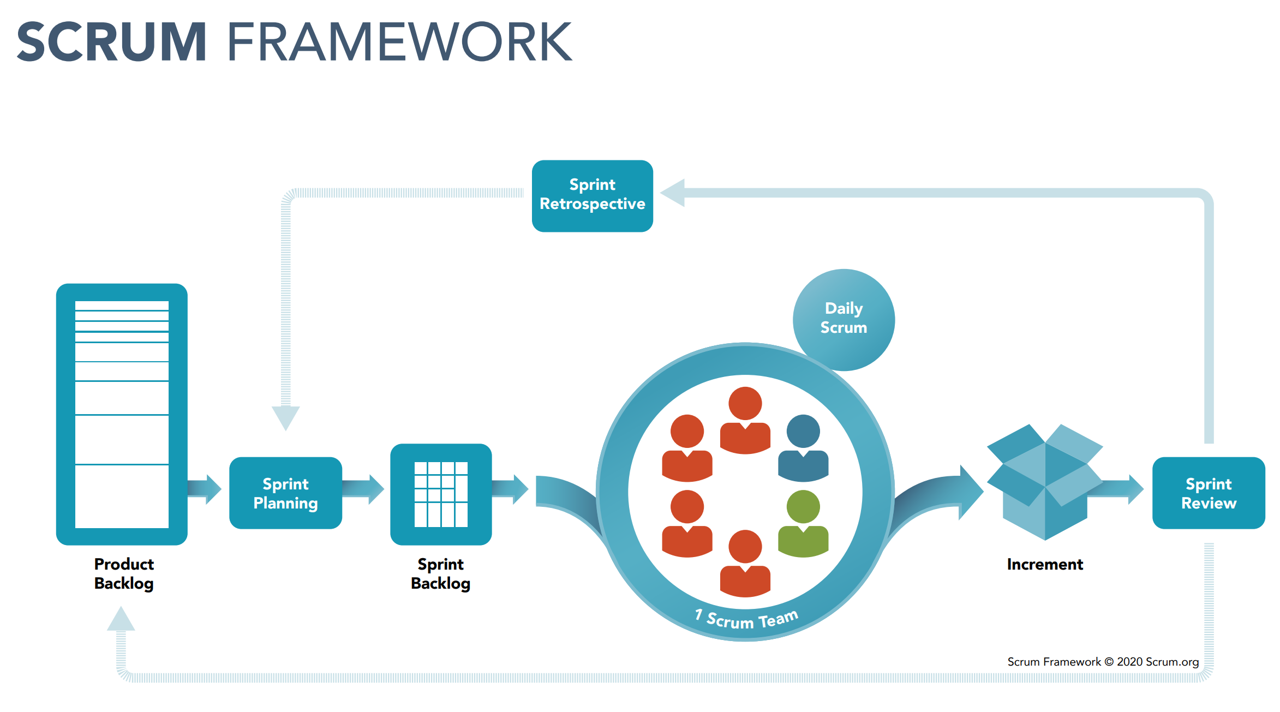
## Metodología de trabajo

La metodología de trabajo seleccionada para llevar a cabo los procesos propios de ingeniería de software necesarios para el desarrollo de este proyecto es la adaptación de la metodología ágil ***Scrum*** al trabajo individual.

Esta selección se debe a sus principales ventajas frente a otras metodologías como la mayor capacidad de gestión del tiempo y la flexibilidad a la hora de organizar y planificar las tareas, las cuales son necesarias para la realización de este proyecto.

Dado que Scrum está diseñado para el trabajo en equipos de tamaño pequeño o medio en los que las responsabilidades y tareas se asignan según el rol de cada participante, ya sea el propietario del producto (Product Owner), el director de scrum (Scrum Master), o parte del equipo de desarrollo, resulta necesario adaptar los procesos de esta metodología al trabajo individual.

Figura .Representación de la metodología Scrum



Fuente: ¿Qué es Scrum?, Joel Francia Huambachano, Perú, 25 de septiembre de 2017 (https://www.scrum.org/resources/blog/que-es-scrum)

Respecto a cómo adaptar la metodología Scrum, originalmente diseñada para el trabajo en equipos reducidos, se opta por adaptar cada una de las fases de esta metodología de la siguiente manera:

1. Definición de Roles: En este caso todos los roles propios de Scrum (Product Owner, Scrum Master y Equipo de desarrollo) y, por tanto, las tareas propias de cada uno son asumidos por un mismo actor.
2. Planificación. Product Backlog: Definición, mantenimiento y priorización de las tareas necesarias para el desarrollo del proyecto a realizar por la misma persona, en este caso.

Se parte de la definición de las tareas necesarias para el desarrollo del proyecto a un nivel alto de abstracción (tareas padres) y, a continuación, se concretan en tareas más específicas (tareas hijas o derivadas) con el fin de poder estimar el tiempo necesario para su realización y poder planificar el trabajo.

1. Sprint: Se seleccionan las tareas prioritarias del backlog a realizar cada semana, para ello se tiene en cuenta la estimación realizada anteriormente para tratar de adecuar las tareas a incluir según el tiempo disponible.
2. Daily Scrum: Dedicar diariamente el tiempo equivalente a una reunión breve para revisar el progreso conseguido durante el día y organizar el trabajo del día siguiente. Esto permite tener una visión diaria del avance del proyecto.
3. Revisión del Sprint (Sprint Review): Revisión de las tareas realizadas durante el sprint semanal para confirmar el grado de consecución de los objetivos planteados.
4. Retrospectiva del Sprint (Retrospective): Reflexionar sobre lo que se podría mejorar de cara al siguiente sprint con el fin de mejorar la metodología de trabajo.

En definitiva, se trata de adaptar la metodología Scrum al trabajo individual, tal y como se muestra en la *Figura 6*.

Para la gestión del proyecto se ha seleccionado ***Trello*** por su adecuación a metodologías ágiles como la propuesta. Su estructura basada en tableros, listas y tarjetas permite organizar y visualizar de forma clara las tareas, facilitando la planificación de esprints semanales y el control del avance del proyecto. Además, su interfaz intuitiva y su flexibilidad lo convierten en una herramienta eficaz para este propósito.

Figura . Esquema de la metodología Scrum adaptada al trabajo individual.

Diagrama

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia basada en la figura 1. ¿Qué es Scrum?, Joel Francia Huambachano, Perú, 25 de septiembre de 2017 (<https://www.scrum.org/resources/blog/que-es-scrum>)

# Análisis y definición del sistema a desarrollar

En el presente capítulo, abordamos la fase crucial de análisis y definición que sienta las bases para el desarrollo del módulo presupuestario especializado en viajes en moto. En el contexto de este proyecto, esta fase adquiere una relevancia especial al tratarse de un módulo de presupuestación especializado en viajes en moto, donde la precisión y la exhaustividad en la identificación de los elementos a considerar resultan cruciales para el éxito de la implementación.

El presente capítulo se estructura en seis apartados que abordan de manera sistemática los aspectos fundamentales del análisis del sistema. En primer lugar, se realiza un estudio detallado de los diferentes tipos de gastos asociados al moto-turismo, estableciendo una clasificación que permita su correcta integración en el sistema de cálculo.

Posteriormente, se procede a la definición formal de los requisitos del sistema, tanto funcionales como no funcionales. La correcta definición de estos requisitos es fundamental para establecer el alcance del desarrollo y asegurar que el producto final cumpla con las expectativas establecidas.

Seguidamente, y de forma complementaria, se incluye el diseño y definición de los casos de uso, así como la descripción del flujo típico y alternativos correspondientes a cada uno de los casos de uso definidos con el fin de obtener una comprensión clara y detallada de la funcionalidad del sistema desde la perspectiva de los usuarios.

Finalmente, se establece el alcance del desarrollo, delimitando claramente los límites del sistema y definiendo las interfaces necesarias para su integración con aplicaciones existentes de planificación de rutas.

El resultado de este proceso de análisis y definición sirve como base sólida para el diseño e implementación del sistema, garantizando que el producto final cumpla con los objetivos establecidos y aporte un valor añadido significativo al ecosistema de aplicaciones de planificación de viajes en moto.

## Identificación y clasificación de conceptos y gastos de viajes en moto

Para la elaboración de cualquier presupuesto es imprescindible tener en cuenta todos los conceptos implicados y si se trata de realizar presupuestos especializados y personalizados se requieren conocimientos acordes con dicha especialización para conocer cuáles son los conceptos necesarios y tratar de garantizar así un resultado completo y realista para el usuario final.

En el caso de los presupuestos de rutas o viajes en moto la experiencia, tanto propia como la de otros motoristas, resulta ser la mejor fuente de información para la recopilación y clasificación de los conceptos a valorar.

Una vez recopilada dicha información se procede a clasificarla y a seleccionar los conceptos más adecuados a incluir en el presupuesto.

Para organizar de forma coherente y útil los distintos conceptos asociados a los viajes en moto, se ha optado por aplicar un sistema de clasificación basado en tres criterios. Este enfoque permite estructurar los datos de manera lógica, facilitando tanto su análisis como su posterior tratamiento por parte del sistema de presupuestación. A continuación se explican los tres tipos de clasificación establecidos y el orden en el que se aplican.

### Clasificación según la naturaleza del concepto

El primer nivel de clasificación distingue entre:

* **Gastos propiamente dichos**: Son aquellos conceptos que implican una salida económica real y que deben ser tenidos en cuenta en el presupuesto final del viaje y de las etapas que lo componen. Los conceptos de este tipo utilizados en este proyecto se encuentran en el Anexo B. Detalle de los gastos incluidos en el proyecto.
* **Conceptos informativos**: Son conceptos que no representan un gasto en sí mismos, pero que resultan necesarios para calcular otros gastos o para identificar los viajes y las etapas. Ejemplos de estos son la distancia a recorrer, el consumo medio de la moto o el destino principal. Los conceptos de este tipo utilizados en este proyecto se encuentran en el Anexo C. Detalle de los conceptos informativos incluidos en el proyecto.

Esta primera clasificación es primordial, ya que delimita claramente qué elementos son costes directos a presupuestar y cuáles son variables necesarias para cuantificar dichos costes o como variables identificativas.

### Clasificación según el tipo de flujo

Una vez establecida la naturaleza del concepto, el segundo nivel de clasificación hace referencia al momento o tipo de intervención que representa dentro del procesamiento de los datos, es decir, si se trata de conceptos de entrada, de salida o de ambos:

* **Entrada**: Son conceptos introducidos por el usuario o que deben provenir de la aplicación cliente tanto si han sido introducidos por el usuario como si son datos obtenidos a partir de las rutas calculadas por la aplicación cliente.
* **Salida**: Son los resultados obtenidos a partir del procesamiento de los datos por la lógica del cálculo presupuestario.
* **Entrada y salida**: Son conceptos con carácter informativo que se muestran como resultados y que no varían en el procesamiento de los datos aunque sí hayan podido participar de los cálculos presupuestarios. Es decir, son datos de entrada que se muestran tal cual en los resultados como datos de salida informativos.

### Clasificación según la especificidad del gasto

Finalmente, se distingue si el concepto es un gasto común a cualquier viaje por carretera independientemente del vehículo utilizado o si se trata de un gasto específico propio de un viaje en moto:

* **Gasto común**: Aquí se incluyen gastos como alojamiento, alimentación, peajes o combustible.
* **Gasto específico**:Ejemplos de estos son los gastos relacionados con la equipación del piloto y pasajero, de la equipación de la moto necesaria para el viaje y de los conceptos asociados con el mantenimiento de la motocicleta.

Esta última clasificación es básica para el objetivo de desarrollar un presupuesto especializado en viajes en moto, ya que permite aislar y dar la importancia adecuada a los costes particulares asociados a este tipo de desplazamiento.

Al seguir este orden de clasificación, desde la distinción fundamental entre información y gasto, pasando por el flujo de los conceptos, hasta la especificidad del medio de transporte, se consigue lograr una estructura analítica robusta y detallada que facilitará la definición de los requisitos del sistema y el establecimiento preciso de su alcance.

Todos los conceptos seleccionados para la elaboración de los presupuestos de los viajes en este proyecto han sido incluidos debido a su importancia. Se podrían incluir más conceptos sobre todo en relación a las categorías de equipación del piloto y pasajero, equipación de la moto y mantenimiento. Sin embargo, los conceptos incluidos son una representación más que suficiente para la elaboración de presupuestos realistas y especializados en viajes en moto.

Dado el elevado número de conceptos se ha considerado la posibilidad de ofrecer los datos de salida en distintos formatos con el fin de aportar opciones con más detalle que incluyan todos los conceptos y otras opciones con datos más resumidos en categorías e incluso un conjunto limitado de datos e identificativos del viaje.

## Definición de requisitos funcionales y no funcionales

Se ha realizado la definición y el análisis exhaustivo de los requisitos funcionales y no funcionales propios del módulo de cálculo de presupuestos para viajes en moto, estableciendo las bases para su diseño y desarrollo, y teniendo en cuenta de que el objetivo es desarrollar un Producto Mínimo Viable que incorpore la funcionalidad principal y necesaria.

El detalle del análisis realizado en relación a los requisitos funcionales y a los requisitos no funcionales se encuentra en los anexos: Anexo D. Detalle de los requisitos funcionales y Anexo E. Detalle de los requisitos no funcionales, respectivamente.

A modo de resumen, se incluye un esquema de los requisitos funcionales y no funcionales resultado de dicho análisis.

Figura . Esquema de requisitos funcionales y no funcionales.

Una captura de pantalla de un celular con letras

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

## Diagrama de casos de uso

En este apartado se incluye el Diagrama de Casos de Uso correspondiente al módulo de elaboración de presupuestos de viajes en moto. Este diagrama representa, de manera gráfica y estructurada, las principales interacciones entre la aplicación cliente como actor externo y el sistema, así como los diferentes servicios y funcionalidades que ofrece el módulo.

A través de este diagrama, se facilita la comprensión global de los requisitos funcionales, permitiendo identificar de forma clara los procesos clave, los flujos de información y la integración mediante API.

Figura . Diagrama de Casos de Uso del Módulo presupuestario.

Diagrama

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

## Descripción del flujo típico y del flujo alternativo de casos de uso

A continuación se describen, de manera detallada, los flujos de ejecución asociados a los principales casos de uso del sistema, y los actores para alcanzar el objetivo principal de cada caso de uso, mientras que los flujos alternativos contemplan situaciones excepcionales, errores o variantes den la interacción que pueden surgir durante la ejecución.

La identificación y documentación de estos flujos permite anticipar posibles escenarios, mejorar la robustez del sistema y garantizar que se cubren adecuadamente tanto las necesidades habituales como las situaciones imprevistas. Además, facilita la comprensión de la lógica interna del sistema y sirve como base para el diseño, la implementación y la validación de los requisitos funcionales.

### Enviar datos a través de API

Tabla . Caso de uso: Enviar datos a través de API

|  |  |
| --- | --- |
| **Caso de uso: Enviar datos a través de API** | |
| **Requisito Funcional:** | RF4 – Integración con aplicación de origen mediante API. |
| **Requisitos previos:** | El usuario debe haber introducido los datos de entrada en la aplicación cliente. |
| **Flujo típico:** | La aplicación cliente recopila los datos y gastos del viaje.  La aplicación cliente realiza una petición HTTP a la API del sistema, enviando los datos en formato JSON.  El sistema recibe la petición y valida la estructura y el contenido de los datos recibidos.  Si los datos son correctos, el sistema continúa con el procesamiento de los datos de entrada. |
| **Flujo alternativo:** | Datos con formato incorrecto: El sistema detecta que el formato de los datos no es válido y responde con un mensaje de error solicitando que se vuelvan a introducir con el formato adecuado. |

### Calcular presupuestos

Tabla . Caso de uso: Calcular presupuestos

|  |  |
| --- | --- |
| **Caso de uso: Calcular presupuestos** | |
| **Requisito Funcional:** | RF1 – Cálculo de los presupuestos |
| **Requisitos previos:** | Datos de entrada recibidos vía API. |
| **Flujo típico:** | El sistema comprueba la validez de los datos recibidos y, si son correctos,  realiza los cálculos necesarios para la elaboración del presupuesto a partir de los importes de los gastos recibidos. |
| **Flujo alternativo:** | Error interno de cálculo: Si se produce un error inesperado durante la ejecución del cálculo de los presupuestos, el sistema informa del fallo.  Datos con formato incorrecto: El sistema detecta que el formato de los datos no es válido y responde con un mensaje de error solicitando que se vuelvan a introducir con el formato adecuado. |

Fuente: Elaboración propia.

### Procesar datos del viaje

Tabla . Caso de uso: Procesar datos del viaje

|  |  |
| --- | --- |
| **Caso de uso: Procesar datos del viaje** | |
| **Requisito Funcional:** | RF2 – Procesamiento de datos del viaje. |
| **Requisitos previos:** | Datos de entrada recibidos vía API. |
| **Flujo típico:** | El sistema comprueba la validez de los datos recibidos y, si son correctos, los procesa realizando los cálculos necesarios y preparando los resultados. |
| **Flujo alternativo:** | Error interno del procesamiento de los datos: Si se produce un error inesperado durante el procesamiento de los datos, el sistema informa del fallo.  Datos con formato incorrecto: El sistema detecta que el formato de los datos no es válido y responde con un mensaje de error solicitando que se vuelvan a introducir con el formato adecuado. |

Fuente: Elaboración propia.

### Generar formatos de salida

Tabla .Caso de uso: Generar formatos de salida

|  |  |
| --- | --- |
| **Caso de uso: Generar formatos de salida** | |
| **Requisito Funcional:** | RF3 – Generar formatos de salida. |
| **Requisitos previos:** | El usuario o la aplicación cliente deben haber seleccionado el formato de salida de los datos: Etiqueta, Resumen o Completo. |
| **Flujo típico:** | El sistema selecciona y organiza los datos calculados según el formato solicitado.  El sistema genera la respuesta estructurada y preparada para su envío. |
| **Flujo alternativo:** | Formato solicitado no soportado: Si el sistema recibe una solicitud de respuesta en un formato no válido, muestra un mensaje de error. |

Fuente: Elaboración propia.

### Recibir resultados a través de API

Tabla . Caso de uso: Recibir resultados a través de API.

|  |  |
| --- | --- |
| **Caso de uso: Recibir resultados a través de API** | |
| **Requisito Funcional:** | RF4 – Integración con aplicación de origen mediante API. |
| **Requisitos previos:** | El sistema debe haber generado los formatos de salida de resultados. |
| **Flujo típico:** | La aplicación cliente realiza una petición a la API para obtener los resultados del cálculo en el formato deseado.  El sistema envía los resultados en el formato solicitado a la aplicación cliente.  La aplicación cliente recibe y muestra los resultados al usuario final. |

Fuente: Elaboración propia.

## Diseño de la arquitectura de la solución

A partir del análisis previo de requisitos funcionales y no funcionales y casos de uso, se ha determinado que la solución a desarrollar debe basarse en una arquitectura sencilla y eficaz, acorde con las características de un Producto Mínimo Viable orientadas a la validación de propuesta de valor. El objetivo principal es garantizar un desarrollo ágil y la facilidad de integración con aplicaciones cliente mediante API.

Figura . Esquema de la arquitectura de la solución.

Diagrama

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Tras evaluar diferentes enfoques arquitectónicos, se han considerado dos opciones principales: arquitectura monolítica (con separación de responsabilidades) y arquitectura por capas (simplificada). Ambas opciones permiten estructurar el sistema de manera organizada, pero tienen algunas diferencias respecto a modularidad, acoplamiento y potencial de escalabilidad.

Dado que el planteamiento inicial ha sido el de separar la lógica de cálculos y procesamiento de los datos de la interfaz de comunicaciones (API) parece tener sentido optar por una arquitectura por capas. Sin embargo, la separación en capas de presentación, lógica de negocio y datos para un PMV podría implicar una mayor rigidez inicial y, además, en este caso la capa de datos no resulta necesaria.

Por otro lado, las características propias de la arquitectura monolítica coinciden con el enfoque del proyecto, es decir, un desarrollo en un único bloque ejecutable pero organizando el código en módulos diferenciados según su responsabilidad al separar la lógica de negocio de la API de comunicación. Además de ser una solución más sencilla que permite un desarrollo más ágil y flexible que permite cumplir con las expectativas propias de un Producto Mínimo Viable.

Por estos motivos, se ha optado por una arquitectura de tipo monolítica, eso sí, con separación entre la lógica de negocio y la API. Este hecho posibilita una escalabilidad futura pasando por una migración hacia arquitecturas más complejas. También permite una depuración y mantenimiento más sencillos que en el caso de una arquitectura monolítica pura.

## Establecimiento del alcance del desarrollo

Para delimitar el alcance del desarrollo se han tenido en cuenta los análisis previos realizados a lo largo de este capítulo, es decir, la identificación y clasificación de todos los conceptos y gastos a considerar, los requisitos funcionales y no funcionales, análisis de casos de uso y la arquitectura seleccionada. Todo ello, proporciona una visión del alcance del proyecto que, sumado al objetivo de desarrollar un Producto Mínimo Viable, permite delimitar el desarrollo necesario para llevarlo a cabo.

Dicho desarrollo se concreta en dos elementos principales: la lógica de negocio y la API como pasarela de comunicación con aplicaciones cliente a las que se pretende incorporar la funcionalidad del cálculo de presupuestos especializados en viajes en moto.

La lógica de negocio se encargará del procesamiento de los datos recibidos a desde la aplicación cliente realizando los cálculos necesarios para la obtención del presupuesto del viaje y la información relacionada. Tendrá en cuenta las distintas partidas de gastos y procesará los gastos propios de cada etapa y los gastos propios del viaje completo. Además, la lógica de negocio será responsable de generar los resultados en los diferentes formatos de salida (etiqueta, resumen y completo), adaptando la información según las necesidades de presentación y consulta tanto para la aplicación cliente como para el usuario final.

Por otro lado, la API constituirá la interfaz de comunicación entre la lógica de negocio y las aplicaciones cliente. Su funcionalidad principal es recibir las peticiones con los datos del viaje, validar su estructura y canalizar la información hacia la lógica de negocio para su procesamiento. Una vez obtenidos los resultados, la API se encargará de devolverlos a la aplicación cliente en el formato solicitado, asegurando así una integración sencilla y eficaz. Esta pasarela facilita la interoperabilidad y permite que la solución pueda ser incorporada como un módulo complementario en diferentes aplicaciones de planificación de rutas y viajes en moto.

La tecnología seleccionada para el desarrollo de ambos componentes es Python. Esta elección se fundamenta en varias razones alienadas con los objetivos del proyecto. Python es un lenguaje ampliamente utilizado en el ámbito de aplicaciones software. Destaca por su sintaxis clara y legible, lo que facilita la implementación rápida y el mantenimiento del código, aspectos relevantes en el contexto de un Producto Mínimo Viable. Además, Python cuenta con un ecosistema muy amplio de librerías y frameworks, entre los que destaca FastAPI para la construcción de APIs modernas, rápidas y seguras. Esta tecnología permite reducir los tiempos de desarrollo y simplificar la validación de datos, garantizando al mismo tiempo un rendimiento aceptable y una fácil integración con otras aplicaciones. Por todo ello, Python se presenta como la opción más adecuada para alcanzar los objetivos planteados en este proyecto.

# Diseño y desarrollo técnico

En este capítulo se abordan los aspectos clave relacionados con la construcción y la implementación del proyecto. En primer lugar, se presentan las principales herramientas empleadas durante el desarrollo del código, entre las que destacan Visual Studio Code, Postman y Github, cuya elección ha facilitado tanto la programación como la realización de pruebas del funcionamiento del código. Seguidamente, se detallarán los objetivos y la estructura de la lógica de negocio, centrándose en cada uno de los bloques funcionales del código. Por último, y de la misma manera que en el caso de la lógica de negocio, se detallarán los objetivos y el funcionamiento de la API de integración, que permite la comunicación con las aplicaciones cliente. Cada uno de estos aspectos se desarrollan en los siguientes apartados, proporcionando una visión clara y estructurada del proceso de diseño y desarrollo llevado a cabo para la consecución de los objetivos de este proyecto.

## Herramientas utilizadas

Para el desarrollo del código se han utilizado herramientas ampliamente reconocidas en el ámbito profesional y académico, que ofrecen un entrono robusto, flexible y adaptado a las necesidades de un desarrollo ágil y eficaz necesario para este proyecto. Estas herramientas son Visual Studio Code como entorno de programación, Postman para la generación y prueba de peticiones JSON y Github como repositorio de versiones del código fuente.

**Visual Studio Code** ha sido el entorno de desarrollo integrado (IDE) elegido para la programación de todo el proyecto. Este editor es conocido por su ligereza, velocidad y alta capacidad de personalización mediante extensiones. Entre sus características más destacables se encuentran la gestión de la sintaxis de múltiples lenguajes de programación que gracias a su función de autocompletado inteligente facilita la codificación de las instrucciones, también cabe destacar la capacidad de depuración integrada y la usabilidad de entorno gracias a su interfaz intuitiva. Cuenta con la posibilidad de instalar extensiones específicas para Python y FastAPI por lo que ha propiciado el desarrollo tanto de la lógica de negocio como de la API. Además, su gestión de archivos y carpetas ha resultado especialmente útil para mantener la estructura del proyecto de forma ordenada y simple.

Por otro lado, **Postman** se ha utilizado como herramienta para probar el funcionamiento de la API gracias a la generación de peticiones JSON con la estructura adecuada y la comprobación de las respuestas según los formatos desarrollados. Postman permite crear, enviar y analizar peticiones de HTTP de manera sencilla, lo cual resulta muy útil para probar APIs. Entre sus principales funcionalidades destaca la posibilidad de gestionar variables de entorno y visualizar las respuestas de la API en diferentes formatos. En el contexto de este proyecto, Postman ha facilitado la simulación de escenarios de uso permitiendo validar el correcto funcionamiento de los endpoints, la estructura de los datos intercambiados y la robustez de la lógica de negocio ante diferentes datos de entrada.

Por último, cabe destacar el uso de **Github** como repositorio para el almacenamiento y la gestión de distintas versiones del código fuente. Github permite almacenar de forma segura el código y documentar los cambios que se van realizando a lo largo de la evolución del proyecto. En este caso, ha sido utilizado para guardar manualmente las versiones del código en los diferentes hitos de desarrollo, lo que ha permitido tener un registro ordenado y seguro de los avances. Gracias a la posibilidad de recurar versiones anteriores se pudo reestablecer el código a una versión estable tras detectar una serie de fallos difícilmente corregibles cuya subsanación resultó más compleja de lo esperado y por tanto se optó por recuperar una versión anterior libre de estos errores para poder continuar con el desarrollo del código.

Respecto al tipo de licencia seleccionada a aplicar al código del repositorio, se optó por la MIT por su adecuación a los proyectos académicos.

Se puede acceder a dicho repositorio en el siguiente enlace:

<https://github.com/carloslora24/TFG_Presupuestos_viajes_moto>

## Implementación de la lógica de presupuestación

Este apartado aborda la implementación de la lógica de negocio o lógica de presupuestación, que es el elemento central del sistema desarrollado. El objetivo principal de esta lógica es garantizar el correcto procesamiento de los datos recibidos desde la aplicación cliente, así como la realización de todos los cálculos necesarios para la obtención de presupuestos precisos y adaptados a las características de cada viaje en moto. Además, la lógica de presupuestación está diseñada para generar los resultados en los tres formatos de salida definidos: Etiqueta, Resumen y Completo, permitiendo así que la información se adapte a las distintas necesidades de consulta y presentación tanto para la aplicación cliente como para el usuario final.

Entre los objetivos principales de la lógica implementada en el fichero “***calculos.py***” destacan el cálculo detallado de los gastos propios de las etapas (alojamiento, alimentación, peajes, actividades, aparcamiento, etc.) que componen el viaje y de los gastos asociados al viaje en general (tasas, seguros, equipación de piloto y pasajero, equipación de la moto, mantenimiento, etc.), el procesamiento de los datos informativos tanto del viaje como de cada etapa en concreto.

A continuación, se detalla la estructura del código, organizada en bloques funcionales, incluyendo fragmentos representativos del código y una explicación de la funcionalidad más relevante.

### Importación de módulos necesarios

En este caso, ha sido necesario importar el módulo estándar datetime de Python para gestionar y validar correctamente las fechas asociadas a las etapas del viaje y al inicio y fin del viaje. El módulo datetime se utiliza en la función de validación de fechas, donde se convierte una cadena de texto recibida en formato DD/MM/AAAA en un objeto de tipo datetime. Esta conversión permite comprobar que la fecha introducida es válida y la utilización de un formato estandarizado.

Figura . Ejemplo de código: Importación del módulo datetime.

Una captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

### Funciones de validación de formatos

Dentro de la lógica de presupuestación, las funciones de validación de datos desempeñan un papel fundamental para garantizar la integridad, coherencia y fiabilidad, de la información procesada por el sistema. Estas funciones se encargan de comprobar que los datos recibidos desde la aplicación cliente cumplen con los formatos y restricciones necesarios antes de ser utilizados en los cálculos y en la generación de resultados. En el fichero “*calculos.py*” se han implementado tres funciones de validación:

* ***validar\_numero(valor):*** Esta función verifica que el valor recibido sea un número válido de tipo *float*. Si el valor no puede convertirse lanza una excepción evitando posibles errores en los cálculos en los que se utiliza. Se utiliza para validar los datos numéricos del sistema como kilómetros, importes de gastos, consumos, precios, etc.
* ***validar\_texto(texto):*** Su objetivo es asegurar que los campos de texto no estén vacíos ni contengan únicamente espacios en blanco. Si el texto no es válido, se lanza una excepción. Esta función se emplea para validar nombres de viaje, destino, orígenes de etapa y cualquier otro campo de texto relevante, garantizando que la información descriptiva sea siempre significativa y útil.
* ***validar\_fecha(fecha\_str):*** Esta función comprueba que la fecha recibida en formato cadena (DD/MM/AAAA) sea válida y la convierte en un objeto datetime. Si el formato es incorrecto o la fecha no existe se lanza una excepción. Se utiliza para validar las fechas de cada etapa, así como las fechas de inicio y fin del viaje.

A continuación, se incluye una captura del código de esta función que sirve como ejemplo de cómo se han implementado las tres funciones de validación de datos.

Figura . Código de la función validar\_fecha(fecha\_str).

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Estas funciones de validación se emplean de manera transversal en todo el código, especialmente en la fase inicial del procesamiento de los datos recibidos y antes de realizar cualquier cálculo. Su aportación principal es la prevención de errores y la robustez del sistema, ya que evitan que datos incorrectos o inconsistentes puedan afectar al resultado final o provocar fallos en la ejecución.

### Generación de formatos de resultados

Una de las funcionalidades clave de la lógica de presupuestación desarrollada es la capacidad de ofrecer los resultados en diferentes formatos de salida, adaptados a las necesidades de la aplicación cliente y del usuario final. Para ello, se han implementado tres funciones principales: ***generar\_etiqueta\_viaje***, ***generar\_resumen\_viaje*** y ***generar\_detalle\_viaje***. Cada una de estas funciones construye una estructura de datos específica que organiza y presenta la información de manera diferenciada, permitiendo así una mayor flexibilidad y personalización de la consulta y visualización de los resultados generados.

* ***generar\_etiqueta\_viaje(datos\_viaje, etapas, total\_viaje)***: está orientada a proporcionar un formato de salida compacto y esencial pensado para ofrecer una visión rápida y resumida de los datos identificativos del viaje. Incluye el nombre del viaje, el destino principal, el número de etapas, el presupuesto total y el presupuesto por persona, en el caso de que se incluya el pasajero.

Figura .Función generar\_etiqueta\_viaje.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

* ***generar\_resumen\_viaje(datos\_viaje, etapas, otros\_gastos, equip\_moteros, equip\_moto, mantenimiento, totales)***: Está función amplía el nivel de detalle presentando un resumen estructurado de las partidas principales de gastos de la información general del viaje, sin llegar al detalle de cada concepto incluido en el presupuesto.

Figura . Cabecera de la función generar\_resumen\_viaje.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

* ***generar\_detalle\_viaje(datos\_viaje, etapas, otros\_gastos, equip\_moteros, equip\_moto, mantenimiento, totales):*** Por último, esta función incluye todos los conceptos y gastos incluidos en el presupuesto con detalle de cada etapa y totales del viaje. Es el formato de respuesta más completo y detallado.

Figura . Cabecera de la función generar\_detalle\_viaje.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

### Procesamiento del presupuesto

El último bloque funcional del código de la lógica de negocio se corresponde con la función principal llamada ***procesar\_presupuestovm***, dicha función es la encargada de orquestar todo el flujo de procesamiento, desde la recepción y validación de los datos de entrada hasta la generación y devolución de los resultados en el formato solicitado. Su funcionalidad principal consiste en recibir un conjunto de datos estructurados que describen el viaje, las etapas, los gastos y otros conceptos relevantes, así como el formato de salida deseado. A partir de esta información, valida los datos utilizando las funciones auxiliares de validación, realiza los cálculos necesarios para estimar los diferentes conceptos de gasto (combustible, alojamiento, alimentación, peajes, actividades, equipamiento, mantenimiento, etc.) y agrega los resultados tanto a nivel de etapa como a nivel global del viaje. Además, calcula totales, medias y otros indicadores útiles para la planificación y el análisis presupuestario del viaje.

Una de sus características más destacables es su capacidad para adaptarse a diferentes necesidades de presentación de resultados ya que, en función del formato de salida solicitado por la aplicación cliente (Etiqueta, Resumen o Completo), ***procesar\_presupuestomv*** delega la generación de la estructura de datos final a las funciones especializadas correspondientes. De este modo, se garantiza la flexibilidad y la modularidad del sistema, permitiendo que la misma lógica de procesamiento sirva para distintos contextos de uso y niveles de detalle.

Figura . Muestra de algunos cálculos de la función procesar\_presupuestovm.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Figura . Generación de respuestas de la función procesar\_presupuestovm.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Con el desarrollo de ***calculos.py*** se consigue alcanzar los objetivos planteados inicialmente para la lógica de presupuestación e incluso aportar una serie de datos adicionales y muy útiles a cerca del viaje en moto a presupuestar.

## Implementación de la API de integración

La API de integración constituye un componente fundamental dentro del proyecto, ya que actúa como la puerta de entrada y salida de información entre la lógica de presupuestación y las aplicaciones clientes con las que se integre. Los objetivos principales de la API son recibir los datos necesarios para el cálculo de presupuestos de viajes en moto, canalizar dicha información hacia la lógica de negocio y devolver los resultados procesados en el formato solicitado por la aplicación cliente. De este modo, la API proporciona una comunicación eficiente y estructurada, permitiendo que la solución desarrollada pueda ser utilizada como un módulo complementario en diferentes entornos y aplicaciones.

La relevancia de la API radica en su papel como intermediario entre el usuario final (a través de la aplicación cliente) y la lógica de negocio desarrollada. Gracias a la API, se facilita la interoperabilidad del sistema, asegurando que los datos se transmiten y procesan de maneara estandarizada. Además, la API permite abstraer la complejidad interna de los cálculos ofreciendo una interfaz sencilla y robusta para la integración con otros sistemas.

A continuación, se detalla la estructura del código de la API incluido en el fichero “***api.py***”, organizada en bloques funcionales, incluyendo fragmentos del código y la explicación de la funcionalidad más relevante para proporcionar una visión clara y comprensible de su implementación.

### Importación de módulos necesarios

En el fichero ***api.py*** se han importado los módulos siguientes:

* FastAPI: Se trata del framework principal utilizado para la creación de la API. FastAPI proporciona todas las herramientas necesarias para definir endpoints y gestionar peticiones y respuestas HTTP.
* Componentes Query y Body: Permiten definir y gestionar los parámetros de consulta en la URL y el cuerpo de las peticiones post, respectivamente. Estos componentes son fundamentales para recibir y validar los datos enviados por las aplicaciones cliente, asegurando que la información se procesa correctamente desde el momento de su recepción.
* Función procesar\_presupuestovm: Se importa esta función desde el módulo ***calculos.py*** para que la API pueda delegar en ella el procesamiento de los datos recibidos desde la aplicación cliente.
* Módulo Enum: Este módulo se utiliza para definir la clase FormatoRespuesta, que restringe y valida los posibles valores del parámetro de formato de salida, asegurando que la API solo acepte opciones válidas para la generación de resultados.

Figura . Importación de módulos en api.py.

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

### Formatos de respuesta

Para poder ofrecer distintos formatos de respuesta del cálculo presupuestario, en el código de api.py se define una enumeración denominada ***FormatoRespuesta***, que incluye los valores permitidos: etiqueta, resumen y completo.

El código correspondiente a esta definición utiliza la clase Enum para restringir y validar los valores de este parámetro, de modo que cuando una aplicación cliente realiza una solicitud para obtener el presupuesto puede especificar el formato deseado, y la API se encarga de comprobar que dicho formato es válido antes de procesar la petición.

Figura . Clase FormatoRespuesta en api.py.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

### Endpoint Health Check

Este endpoint es el encargado de permitir la verificación rápida del estado operativo de la API. Se define mediante un decorador de FastAPI que asocia la ruta /health a una función que responde a peticiones HTTP de tipo GET.

La funcionalidad principal de este endpoint consiste en devolver una respuesta simple indicando si el estado de la API es correcto o no, confirmando su disponibilidad.

Figura 22. Código del endpoint Health Check en api.py.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

### Endpoint principal

El endpoint principal de la API, definido bajo la ruta /presupuestovm, constituye el núcleo funcional de la integración entre al aplicación cliente y la lógica de negocio del sistema de presupuestación. Este endpoint está implementado como un método que responde a peticiones HTTP de tipo POST, permitiendo la recepción de datos estructurados en el cuerpo de la solicitud (en formato JSON), junto con un parámetro opcional que indica el formato de respuesta deseado. Su funcionalidad principal es la de recibir todos los datos necesarios para el cálculo del presupuesto de un viaje en moto, incluyendo la información general del viaje, las etapas, los distintos conceptos de gasto y el equipamiento. Una vez recibidos y validados, el endpoint delega su procesamiento a la función ***procesar\_presupuestovm***, que es la encargada de generar los resultados en el formato solicitado.

La relación entre el endpoint principal y el módulo ***calculos.py*** es fundamental ya que el endpoint actúa como intermediario entre la entrada de datos proporcionada por la aplicación cliente y la lógica de negocio que realiza el procesamiento de los datos y la generación de resultados. De esta manera, se consigue una clara separación de responsabilidades.

Además, el manejo de errores mediante bloques try/except garantiza que el sistema pueda responder de manera controlada ante posibles incidencias, mejorando su robustez.

Figura . Código del endpoint principal en api.py.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Con la funcionalidad desarrollada en cada uno de los bloques funcionales del código desarrollado en ***api.py*** se cumplen los objetivos planteados tanto para la API como para el conjunto del proyecto. Es decir, gracias a esta implementación se garantiza la interoperabilidad y la facilidad de integración del sistema desarrollado con aplicaciones cliente y dentro del marco de los requisitos definidos.

# Validación y Verificación

En este capítulo se abordan tres aspectos fundamentales que son esenciales para asegurar la calidad del sistema y su correcto funcionamiento en un entorno real.

En primer lugar, se presentan las pruebas funcionales del sistema, donde se detallan los procedimientos y metodologías empleadas para verificar que cada componente del sistema cumple con sus requisitos específicos y funciona de manera integrada. Estas pruebas son cruciales para identificar posibles fallos o inconsistencias en el software.

El segundo apartado se centra en el análisis de precisión de los presupuestos generados y en la fiabilidad de los datos. Este análisis es particularmente importante en un sistema de presupuestación, ya que la exactitud de las estimaciones económicas es fundamental para la toma de decisiones de los usuarios.

Finalmente, se aborda la integración con aplicaciones existentes, aspecto clave para asegurar que el sistema pueda funcionar de manera eficiente dentro del ecosistema tecnológico actual. Este apartado proporciona las directrices necesarias para la interoperabilidad del sistema con otras herramientas y plataformas.

A través de estos tres apartados, se pretende proporcionar una visión completa y detallada de la robustez del sistema, asegurando que cumple con los objetivos establecidos y que es capaz de proporcionar resultados confiables y útiles para los usuarios finales.

## Pruebas funcionales del sistema

El proceso de validación y verificación de la solución implementada comienza con la realización de pruebas funcionales realizadas sobre las funciones desarrolladas en la lógica de procesamiento de los datos (***calculos.py***) y en la interfaz de comunicación con aplicaciones clientes (***api.py***). Su objetivo es el de asegurar tanto su correcto funcionamiento como su robustez, contribuyendo así a obtener un adecuado nivel de fiabilidad. Estas pruebas permiten verificar que cada función cumple con los requisitos definidos y responde adecuadamente según las opciones de formato de salida seleccionado.

### Pruebas funcionales de la lógica de negocio calculos.py

En primer lugar, se comprueban las funciones de validación de los datos de entrada y, a continuación, se comprueban los cálculos principales de las funciones de procesamiento de los datos.

#### Pruebas de funciones de validación de datos de entrada

Para realizar las pruebas de las funciones de validación de datos de entrada se ha desarrollado el archivo “***test\_validacion\_calculos.py***” en el cual se han implementado las pruebas unitarias específicas para las funciones: ***validar\_numero***, ***validar\_texto*** y ***validar\_fecha***.

Estas pruebas abarcan tanto casos de uso habituales como situaciones límite y de error, al comprobar que las funciones:

* Devuelven los valores esperados ante entradas válidas de datos.
* Lanzan las excepciones adecuadas cuando se detectan datos incorrectos, nulos o vacíos.
* Mantienen la coherencia en el tipo de datos retornados.
* Gestionan correctamente los límites y casos especiales, como valores negativos, cadenas vacías o formatos de fechas incorrectos.

Las pruebas realizadas a la función ***validar\_numero*** tratan de verificar que la función tenga un comportamiento adecuado ante: distintos tipos de números (enteros, decimales, negativos), ante distintos valores (cero, vacíos o “*None*[[1]](#footnote-1)”) y ante otro tipo de datos no numéricos.

Figura . Fragmento del código de la función test\_validar\_numero.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

En el caso de los test realizados a la función ***validar\_texto***, se recogen en la función ***test\_validar\_texto***y se centran en comprobar que dicha función gestione correctamente texto normal, texto con espacios, texto vacío y texto con espacios en blanco.

Figura . Código de la función test\_validar\_texto.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Para la comprobación del funcionamiento de la función ***validar\_fecha*** se han realizado comprobaciones con: fechas válidas, fechas inválidas (días o meses inexistentes), fechas con formato incorrecto y fechas vacías o nulas. Estas pruebas están implementadas en la función ***test\_validar\_fecha***.

Cabe destacar que gracias a la realización de estas pruebas **se detectó que la validación de las fechas no era consistente ante la casuística de un valor vacío**, por lo que se modificó el código de la función ***validar\_fecha*** con el fin de corregirlo para incluir esta posibilidad y hacer más robusta la validación.

Figura . Código de la función test\_valdiar\_fecha.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

#### Pruebas funcionales de los cálculos principales

Una vez completadas las pruebas de validación de los datos de entrada, y contando con la seguridad de que los datos que llegan al sistema son correctos y están debidamente validados, el siguiente paso fundamental es garantizar la exactitud de los cálculos principales realizados por el módulo calculos.py. En este contexto, la validación de los resultados numéricos cobra especial relevancia, ya que de ellos depende la fiabilidad y utilidad del presupuesto generado para el usuario final.

Para abordar esta validación, se ha optado por una estrategia basada en una prueba global que verifica que el resultado total del presupuesto calculado es correcto. Esta prueba, implementada en el archivo ***test\_gestion\_calculos.py***, consiste en comparar el total general devuelto por la función principal del módulo con el resultado obtenido a partir de un cálculo manual, sumando todos los conceptos que intervienen en el presupuesto: gastos de etapas (alojamiento, alimentación, peajes, actividades, etc.), equipamiento, mantenimiento, otros gastos y el combustible.

La estructura de la prueba es sencilla y robusta. En primer lugar, se definen unos datos de ejemplo representativos, que incluyen diferentes valores para cada categoría de gasto. A continuación, se ejecuta la función de cálculo del presupuesto y se obtiene el resultado total. Paralelamente, se realiza el cálculo manual del total esperado, sumando uno a uno todos los importes de los distintos conceptos, utilizando los mismos datos de entrada. Finalmente, se compara el resultado obtenido por el sistema con el cálculo manual, asegurando que ambos coinciden.

La finalidad de esta prueba global es doble. Por un lado, permite validar de forma explícita que el cálculo del total general es correcto, lo que es esencial para la confianza del usuario en el sistema. Por otro lado, y de forma implícita, esta prueba verifica que todos los cálculos intermedios (suma de categorías, cálculo de combustible, equipamiento, mantenimiento, etc.) se están realizando correctamente, ya que cualquier error en alguno de estos pasos se reflejaría en una discrepancia en el total final.

Gracias a esta estrategia, se consigue una validación eficiente y fiable de la lógica de negocio principal del sistema, asegurando que el presupuesto generado es coherente y exacto, y que el usuario puede tomar decisiones fundamentadas a partir de la información proporcionada por la aplicación.

Figura . Código de la función\_test\_total\_general\_viaje

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

### Pruebas funcionales de la interfaz de comunicación api.py

Para asegurar el correcto funcionamiento de la API (***api.py***), se han implementado pruebas automatizadas utilizando el framework ***pytest***. Estas pruebas unitarias se han centrado en la validación del formato de respuesta y en la gestión de errores.

Para implementar estas pruebas ha sido necesario el desarrollo de los siguientes archivos:

* ***test\_api.py***: Es el archivo principal que contiene las pruebas unitarias. Tras importar las librerías necesarias para realizar las pruebas, realiza una carga de los datos de prueba previa a la ejecución de las pruebas de formato y gestión de errores.

Figura . Código de test\_api.py.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

* ***datos\_validos.json***: Proporciona datos de prueba válidos.

Figura . Fragmento de código de datos\_validos.json.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

* ***datos\_invalidos.json***: Proporciona datos de prueba inválidos. En este caso, se incluyen datos de tipo incorrecto como texto en vez de números (en el caso de ***consumo\_medio\_moto, kms\_etapa*** y ***alimentación\_desayuno***), texto en vez de true o false (en el caso de ***pasajero***) y valores negativos que afectan a los cálculos como en el caso del precio del combustible.

Figura . Código de datos\_invalidos.json con marcas en los datos inválidos.



Fuente: Elaboración propia.

* ***conftest.py***: Contiene la configuración del entorno de pruebas y el cliente de pruebas.

Figura . Fragmento de código de conftest.py.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Las pruebas implementadas en ***test\_api.py*** verifican el correcto procesamiento de los datos válidos y la estructura de respuesta, el correcto manejo de errores al recibir datos inválidos, la consistencia en los códigos de estado HTTP y la estructura correcta de las respuestas.

**Los resultados obtenidos han sido satisfactorios, con todas las pruebas pasadas correctamente**.

Figura . Fragmento del resultado positivo de las pruebas realizadas mostrado por pantalla.

Una captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Estas pruebas unitarias se complementan con las pruebas funcionales de validación de casos de uso reales y escenarios más complejos que se desarrollan en los siguientes apartados.

## Análisis de precisión de presupuestos

Con el objetivo de comprobar la **fiabilidad y la precisión** de los resultados generados por el módulo de presupuestación de viajes en moto, se han realizado una serie de pruebas a partir de datos realistas de viajes previamente presupuestados y con variación de datos más extremos.

La metodología empleada para llevar a cabo estas comprobaciones se ha basado en las **fases** de:

* Definición de casos de prueba
* Preparación previa de herramienta de comprobación de resultados y de elaboración de archivos JSON
* Ejecución de las pruebas
* Análisis de las pruebas y obtención de conclusiones.

Se han definido dos tipos de pruebas para este análisis, en primer lugar tenemos las **pruebas de casos reales** en los que se han elegido como referencia viajes de distinto número de etapas y en los que se han informado los datos correspondientes a cada tipo de viaje. En segundo lugar, se han realizado **pruebas con valores extremos** para poner a prueba el comportamiento del sistema y cómo afectan determinados datos a los resultados obtenidos.

De esta manera se han abarcado diferentes escenarios incluyendo situaciones comunes, casos límite y casos especiales para probar funcionalidades específicas del sistema, siempre desde el punto de vista de la validez y precisión de los resultados obtenidos.

Antes de la realización de las pruebas ha sido necesaria una preparación previa consistente tanto en la elaboración de una **herramienta de comparación de resultados** como en la **elaboración de los archivos JSON** configurados para cada una de las pruebas.

Se ha utilizado una hoja de cálculo como herramienta de comparación de resultados en la que por cada prueba realizada, según correspondiese o no, se han informado los datos de entrada y los tres formatos de salida con los resultados esperados. Dicha hoja de cálculo se llama “Pruebas Módulo Presupuestario.xlsx”, se ha elaborado con Microsoft Excel y puede consultarse en el repositorio Github del proyecto.

Para la elaboración y prueba de los archivos JSON con los datos de entrada de cada prueba se han utilizado Postman, Block de notas y la interfaz Swagger UI.

Una vez desarrollados preparativos necesarios, se procede a la **ejecución de las pruebas** en sí y al registro y documentación de los resultados para su comprobación. En este proceso, se han tenido en cuenta los tres formatos que ofrece el módulo de presupuestación por lo que se ha tenido que comprobar cada uno de ellos en todos los casos.

Con todos los datos registrados tras la ejecución de las pruebas se procede a su análisis mediante el contraste y la comprobación sistemática de cada uno de los resultados. Aunque se han comprobado todos los resultados obtenidos, se han contrastado con especial interés los cálculos más relevantes en la lógica de negocio implementada como, por ejemplo, el presupuesto total, el presupuesto por persona, en su caso, y el cálculo del coste en combustible y de las principales partidas que lo componen.

A continuación, se detallan las pruebas realizadas. En el **Anexo F. Archivos utilizados en las pruebas de análisis de precisión de resultados** se puede consultar cada uno de los archivos utilizados como datos de entrada, las respuestas obtenidas y las tablas correspondientes para su comprobación. Estos archivos y tablas también están disponibles en el repositorio Github del proyecto.

Figura . Esquema de las pruebas de precisión de presupuestos realizadas.



Fuente: Elaboración propia.

### Pruebas de Casos Reales

En este apartado se resumen las tres pruebas realizadas con datos de casos reales o pruebas de tipo A. Estas pruebas son:

* Prueba A1 – Viaje 1 Etapa
* Prueba A2 – Viaje 4 Etapas
* Prueba A3 – Viaje 9 Etapas

Se trata de probar el módulo presupuestario con datos reales de viajes de menor a mayor duración (en etapas) y en los que se incrementa la cantidad de datos a presupuestar introducidos a medida que el viaje es más largo y completo. Por ello, utilizamos como ejemplo la **Prueba A3 – Viaje 9 Etapas** al ser la más completa.

Los viajes medios y largos son los más difíciles de presupuestar, por lo que es en estos casos donde el módulo de presupuestación de viajes en moto resulta más útil. En este caso se ha optado por un viaje de nueve etapas que discurre por varios países (España, Italia, Suiza y Francia) y que implica gastos menos frecuentes como desplazamientos en ferry (Barcelona – Génova), pases de circulación como la viñeta suiza, peajes, ampliación del seguro de la moto y contratación de seguro de viaje. Además, se ha aprovechado para incluir pasajero y para incluir coste en la mayoría de los conceptos a presupuestar con el fin de poner a prueba la capacidad de cálculo del módulo.

Los resultados de la prueba obtenidos a partir de los datos de entrada del archivo “*P.A3 JSON ENTRADA Viaje 9 Etapas*” son correctos en los tres formatos de salida ofrecidos por el sistema, por lo que demuestra que el sistema admite sin problemas viajes con múltiples etapas y conceptos.

Los resultados obtenidos se recogen en los archivos:

* *Respuesta Completa Viaje 9 Etapas.txt*
* *Respuesta Resumen Viaje 9 Etapas.txt*
* *Respuesta Etiqueta Viaje 9 Etapas.txt*

Figura . Captura de la hoja de cálculo con el resultado en formato Etiqueta de la prueba A3.

Tabla

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Tras el análisis de los resultados de estas tres pruebas se observa que **la velocidad media total del viaje es quizás el dato menos realista**, debido a que está calculado a partir de una previsión inicial del usuario o de la aplicación cliente y no a partir de mediciones a posteriori, de los tiempos empleados para la realización de cada etapa. Quizás, de cara a futuras versiones, se podría implementar la posibilidad de que estos datos viniesen dados a partir de un cálculo basado en etapas similares para obtener una previsión más precisa o que pudiesen ser actualizados con carácter posterior a su realización. No obstante, al tratarse de un dato informativo o estadístico del viaje, no es de los más relevantes en lo que a la funcionalidad de presupuestación se refiere.

Por otro lado, **el formato** que parece ser **más práctico** a la hora de manejar la información de los resultados **es el formato Resumen**. Esto se debe a que contiene tanto la información estadística del viaje como el detalle de los costes por categorías o partidas presupuestarias, resultando ser el formato más equilibrado de respuesta.

Figura . Captura de la hoja de cálculo con el formato de resultado Resumen del viaje de 9 etapas.

Tabla

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

### Pruebas con Datos Extremos

#### Todos los conceptos informados: Pruebas B1 y B2

Con estas pruebas se ha confirmado que el módulo presupuestario es capaz de procesar correctamente todos y cada uno de los conceptos al haber informado un importe para todos ellos, sin excepción.

En el caso de la **Prueba B1 – Todos los conceptos informados**, se han introducido datos de entrada con importes más o menos realistas y los resultados muestran que la lógica ha procesado dichos importes correctamente.

En el caso de la Prueba B2 – Todos los conceptos a presupuestar a cero, excepto el precio del combustible, también se ha comprobado que los resultados son correctos.

Aunque se trate de unas pruebas básicas que puedan parecer innecesarias, son importantes ya que el considerable número de conceptos a tener en cuenta en el presupuesto ha implicado minuciosas comprobaciones tanto desde el diseño inicial de los datos de entrada como en el desarrollo de la lógica y la comprobación de los resultados.

Como ejemplo, se incluyen los resultados de la prueba B1 obtenidos a partir de los datos de entrada del archivo “*JSON ENTRADA Prueba B1*”. Éstos son correctos en los tres formatos de salida ofrecidos por el sistema, por lo que demuestra que el sistema tiene en cuenta todos los conceptos de entrada y los procesa adecuadamente.

Los resultados obtenidos se recogen en los archivos:

* *Respuesta Completa Prueba B1.txt*
* *Respuesta Resumen Prueba B1.txt*
* *Respuesta Etiqueta Prueba B1.txt*

Figura . Captura de la hoja de cálculo con el resultado en formato Etiqueta de la prueba B1.

Tabla

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

#### Conceptos Clave a Cero: Pruebas B3, B4 y B5

En estas pruebas se ha continuado probando el procesamiento de los datos con valores extremos como el cero y, especialmente, en conceptos clave para el cálculo presupuestario como son: el precio del combustible, el consumo medio de la moto y los kilómetros de etapa.

En el caso de la prueba **B3 – Precio de Combustible a cero**, se ha informado a cero tanto el precio del combustible como el resto de los conceptos presupuestables de etapas, equipamiento y mantenimiento, sin que se haya producido ningún resultado erróneo. Aunque no se corresponde con ningún caso útil desde el punto de vista de su posible aplicación real, se trata de probar si el sistema es capaz de procesar estos datos aunque den como resultado un presupuesto sin coste.

Siguiendo la línea de comprobación de datos a cero, en la **prueba B4 – Consumo medio moto a cero**, le toca el turno al consumo medio de la moto. Al informarse como cero no es posible calcular ni los litros de combustible necesarios para la realización del viaje, ni el coste que implica. De manera que el presupuesto resultante no incluye el coste del combustible y por lo tanto resulta incompleto.

En la **Prueba B5 – Kilómetros etapa a cero**, se informan los kilómetros del viaje a cero, que coinciden con los de la etapa al ser un viaje de una sola etapa. Este valor implica que no se puedan calcular ni costes como el del combustible, ni datos estadísticos del viaje como los kilómetros de media por etapa, ni los litros de combustible necesarios, ni los tiempos de desplazamiento total y de media diarios.

Los resultados de estas tres pruebas son los esperados, por lo que suma confianza respecto a la robustez de la lógica presupuestaria.

A continuación se incluye como ejemplo la captura de pantalla del resultado de la prueba B3 – Precio de Combustible a cero.

Figura . Captura de la hoja de cálculo con el resultado en formato de Etiqueta de la prueba B3.

Tabla

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

#### Datos Identificativos Vacíos: Pruebas B6 y B7

En estas pruebas se comprueban los resultados en caso de pasar **datos identificativos clave del viaje como vacíos**.

En el caso de la **prueba B6** se le pasa el **nombre del viaje vacío** (“*nombre\_viaje”: “”),* provocando un error controlado por la función ***validar\_texto()*** que es la encargada de validar el formato del dato introducido en este campo y en otros campos en los que se espera una cadena de texto.

En el caso de la **prueba B7** se le pasa el **destino principal** el que se introduce como **vacío** (*“destino\_principal”:””)*. El resultado obtenido también es el error controlado por la función *validar\_texto()*.

En ambos casos, se lanza la excepción con el mensaje “*Error: El texto no puede estar vacío”*. Este control resulta útil para evitar que se procese un presupuesto sin un nombre o sin un destino que lo identifique, lo que podría causar problemas de organización y seguimiento posterior.

Como ejemplo, se muestra el resultado de la prueba obtenido a partir de los datos de entrada del archivo “*JSON ENTRADA Prueba B6*”, que es el mismo en los tres formatos de salida ofrecidos por el sistema, y se recoge en el archivo: *Respuesta Todos los Formatos Prueba B6.txt.*

Figura . Fragmento del archivo JSON ENTRADA Prueba B6 con el nombre del viaje vacío.

Texto, Carta

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Figura . Captura del error generado por la prueba B6.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

#### Fecha en formato incorrecto: Prueba B8

En esta prueba se introduce una **fecha en formato incorrecto** según el formato definido, es decir, se introduce 2025/05/20 en vez de 20/05/2025. Esto genera un mensaje de error que avisa de que el formato de fecha es inválido. La función ***validar\_fecha()*** es la encargada de controlar y validar el formato de las fechas utilizadas en la lógica del módulo presupuestario. En concreto, cuando se trata de convertir la fecha con **datetime.strprime()**, Python no puede interpretar el formato y lanza la excepción.

Gracias a la función ***validar\_fecha()***el sistema se asegura de que todas las fechas tengan el mismo formato evitando confusiones al usuario e inconsistencias con la aplicación cliente y el posible sistema de almacenamiento de rutas que ésta disponga. Además, indica claramente el formato esperado para facilitar la corrección en la introducción de los datos.

El resultado de la prueba obtenido a partir de los datos de entrada del archivo “*JSON ENTRADA Prueba B8*” es el error comentado, en los tres formatos de salida ofrecidos por el sistema, y se recoge en el archivo: *Respuesta Todos los Formatos Prueba B8.txt.*

Figura .Fragmento del archivo JSON ENTRADA Prueba B8 con la fecha en formato incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Figura . Captura del error producido por la ejecución de la prueba B8.

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

#### Número como cadena de texto: Prueba B9

En esta prueba se pasa el concepto **kilómetros etapa como una cadena de caracteres** en vez de como un número *(“kms\_etapa”: “248”)*. Sin embargo, el sistema transforma el texto en un número tipo *float* y realiza los cálculos correctamente.

Esto ocurre gracias a la función ***validar\_numero()*** que es capaz de procesar tanto números como cadenas de texto que representan números, lo que la hace muy flexible. De esta manera, la cadena *“248”* se convierte automáticamente en el número 248.0.

En el caso de que la conversión fallase, la función lanzaría una excepción. Y si el valor fuese nulo o vacío, la función devolvería *0.0* como valor por defecto.

Los resultados de la prueba obtenidos a partir de los datos de entrada del archivo “*JSON ENTRADA Prueba B9*” son correctos en los tres formatos de salida ofrecidos por el sistema.

Los resultados obtenidos se recogen en los archivos:

* *Respuesta Completa Prueba B9.txt*
* *Respuesta Resumen Prueba B9.txt*
* *Respuesta Etiqueta Prueba B9.txt*

Figura 42. Captura del código de respuesta en formato Etiqueta de la prueba B9 (Anexo F).

Texto, Carta

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

## Integración con aplicaciones existentes

El sistema desarrollado pretende servir como complemento funcional para aplicaciones cliente que quieran incorporar la capacidad de calcular presupuestos de viajes en moto. Las aplicaciones cliente potenciales serían todas aquellas que se especializan en la elaboración y configuración de rutas y viajes por carretera y, en especial las orientadas a los viajes en moto.

Para facilitar la integración de esta funcionalidad se ha desarrollado una API (*api.py)* que junto con la lógica de presupuestación de viajes en moto implementada en calculos.py hacen que esto sea posible.

La documentación de la API es crucial para su correcta utilización por parte de aplicaciones cliente. A continuación, se especifican los elementos esenciales de la documentación desarrollada para el endpoint de presupuestación de viajes en moto, que los integradores de aplicaciones cliente deben tener en cuenta.

### Endpoints Disponibles

La API para el cálculo de presupuestos de viajes en moto dispone de dos endpoints: /health y /presupuestovm.

* **Endpoint /Health:** Se trata del endpoint de estado del sistema que se utiliza para verificar que la API está funcionando correctamente. Este endpoint es muy frecuente en el uso de APIs por lo que no es especialmente relevante. Se recomienda un uso equilibrado con el fin de no consumir más recursos de los necesarios.
* **Endpoint /presupuestovm:** Este es el endpoint principal. Acepta peticiones HTTP POST, a través de las cuales las aplicaciones cliente pueden enviar los datos necesarios para la generación del presupuesto. La estructura básica de la petición esperada es un objeto JSON que contiene los datos específicos del viaje en moto para el cual se desea obtener una estimación. Adicionalmente, la petición debe incluir un campo *formato\_respuesta* que indica el nivel de detalle deseado en la respuesta del servidor. Los valores válidos para este campo son "etiqueta", "resumen" y "completo", cada uno correspondiendo a un formato de salida diferente.

Figura . Captura de pantalla de la Interfaz Swagger con los dos endpoints disponibles.

Interfaz de usuario gráfica, Texto, Aplicación

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

### Entrada y salida de datos

La comunicación con el endpoint /***presupuestomv*** se realiza mediante el intercambio de objetos JSON.

Se han implementado tres formatos de salida distintos para adaptarse a las diferentes necesidades de las aplicaciones cliente. El formato **Etiqueta** proporciona una información concisa del viaje, incluyendo nombre, destino principal, kilómetros y presupuesto total, así como el presupuesto por persona. El formato **Resumen** ofrece un desglose de los gastos por categorías principales, útil para análisis preliminares. Finalmente, el formato **Completo** proporciona un detalle exhaustivo de cada etapa del viaje y todos los gastos asociados, ideal para análisis en profundidad.

En el Anexo B. Archivos utilizados en las pruebas de análisis de precisión de resultados se pueden encontrar múltiples ejemplos de archivos JSON utilizados tanto como entrada como para los distintos formatos de salida. Estos ejemplos son muy importantes porque reflejan la estructura de los archivos JSON que debe respetarse para asegurar su correcto funcionamiento e integración.

### Validación de Datos y Formatos Esperados

Para garantizar la precisión y la integridad de los cálculos de los presupuestos, la API implementa una serie de validaciones sobre los datos de entrada proporcionados por las aplicaciones cliente. Es importante que las aplicaciones cliente comprendan estos requisitos para enviar peticiones válidas.

En primer lugar, es necesario aclarar que existen ciertos campos que son considerados obligatorios para poder realizar el presupuesto. Estos campos incluyen información esencial del viaje y los costes asociados a los conceptos principales que son necesarios para la realización de los cálculos presupuestarios. La omisión de cualquiera de los campos obligatorios podrá ocasionar errores o fallos en los cálculos de los presupuestos, así como también en la información estadística del viaje.

En segundo lugar, cabe destacar que cada campo se espera un tipo de dato específico. Por ejemplo, los campos que representan cantidades de dinero, distancias o consumos deben ser de tipo numérico (preferiblemente float). Los campos que representan fechas deben seguir un formato específico (DD/MM/AAAA). Los campos para nombres de lugares o descripciones deben ser cadenas de texto. La API realiza validaciones para asegurar que los datos recibidos coincidan con los tipos esperados. El envío de datos con tipos incorrectos (por ejemplo, enviar una cadena de texto para el precio del combustible) generará un error de validación.

En la siguiente tabla se detallan los campos obligatorios y su tipo de dato esperado.

Tabla . Campos obligatorios y tipos de datos esperados.

|  |  |
| --- | --- |
| **Campo o Dato Obligatorio** | **Tipo de Dato Esperado** |
| **nombre\_viaje** | Texto no vació (str) |
| **destino\_principal** | Texto no vació (str) |
| **pasajero** | Booleano (true o false) |
| **precio\_combustible** | Numérico (float) |
| **consumo\_medio\_moto** | Numérico (float) |
| **fecha\_etapa** | Fecha en formato DD/MM/AAAA |
| **origen\_etapa** | Texto no vació (str) |
| **destino\_etapa** | Texto no vació (str) |
| **kms\_etapa** | Numérico (float) |
| **velocidad\_media\_etapa** | Numérico (float) |

Fuente: Elaboración propia.

Respecto a los demás datos de entrada, son conceptos que esperan datos de tipo numéricos (float) y que representen el importe del coste de adquisición o gasto, en cada caso. En el caso de que no se informen, serán interpretados como de valor cero.

El comportamiento de la API ante **valores nulos o vacíos** varía según el campo. Generalmente, los campos obligatorios no aceptan valores nulos o cadenas vacías, y su recepción resultará en un error de validación.

### Gestión de Errores y Excepciones

El sistema de presupuestación de viajes en moto implementa una gestión de errores básica pero efectiva que cubre los aspectos fundamentales necesarios para su funcionamiento, teniendo en cuenta que se trata de un producto mínimo viable. Esta implementación se centra en garantizar la robustez esencial del sistema tratando de mantener la simplicidad y la facilidad de mantenimiento.

La gestión de errores comienza con un conjunto de **funciones de validación** implementadas en calculos.py. Estas funciones, cubren los aspectos esenciales de la validación de datos: *validar\_numero(), validar\_texto()* y *validar\_fecha().* Cada función se encarga de verificar la integridad de los datos antes de su procesamiento, proporcionando una primera línea de defensa contra datos incorrectos.

Las **excepciones** se implementan principalmente en las funciones de validación, permitiendo detectar y manejar los errores más comunes que pueden surgir durante el procesamiento. Este enfoque, proporciona una base sólida para la detección de problemas.

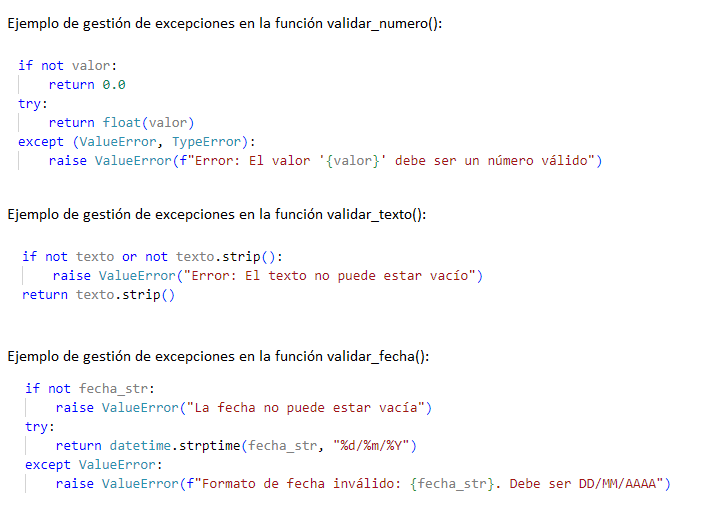
El sistema utiliza principalmente dos tipos de excepciones:

* ***ValueError***: para problemas de formato y validación
* ***TypeError***: para problemas de tipo de datos

Estas excepciones se capturan y transforman en respuestas de error estructuradas y con mensajes claros para facilitar su comprensión por parte del usuario.

La **comunicación de errores** se realiza a través de un formato simple y consistente. Las respuestas de error incluyen un estado y un mensaje descriptivo, proporcionando información suficiente para que los usuarios y desarrolladores puedan entender y resolver los problemas encontrados.

Figura . Ejemplos de gestión de Excepciones en las funciones de validación.



Fuente: Elaboración propia.

# Conclusiones y trabajo futuro

En este proyecto se ha abordado la problemática existente en el ámbito de las aplicaciones especializadas en planificación de viajes en moto: la ausencia de herramientas que permitan calcular, de manera precisa y personalizada, el presupuesto asociado a este tipo de viajes. Aunque existen soluciones tecnológicas avanzadas para la configuración y optimización de viajes y rutas por carretera, ninguna de ellas incorpora la funcionalidad específica para la estimación completa de costes, algo fundamental a la hora de planificar un viaje.

Ante esta situación, se ha optado por el desarrollo de un Producto Mínimo Viable (PMV) consistente en un módulo de cálculo presupuestario diseñado específicamente para viajes en moto. Este módulo ha sido concebido para interactuar a través de una API con aplicaciones cliente ya existentes, con el fin de complementar su funcionalidad y aportando un valor añadido y diferenciador al integrar una herramienta de presupuestación adaptada a las particularidades del moto-turismo.

La implementación de esta solución ha seguido una metodología ágil, adaptada al contexto del desarrollo individual, y ha permitido alcanzar satisfactoriamente los objetivos definidos al inicio del proyecto. A continuación, se exponen las principales conclusiones derivadas del trabajo realizado y se presentan las líneas de trabajo futuro que permitirán continuar mejorando la herramienta desarrollada.

## Conclusiones del trabajo

El desarrollo de este proyecto ha permitido alcanzar los objetivos planteados, validando la viabilidad técnica y práctica de un módulo de cálculo de presupuestos para viajes en moto.

A nivel general, se ha logrado diseñar, desarrollar e implementar un Producto Mínimo Viable de un módulo de presupuestación económica de viajes en moto, de forma integrable con otras aplicaciones vía API, y con el fin de cubrir el vacío existente en cuanto a herramientas de este tipo. Además, el desarrollo de este módulo ha sentado las bases para una futura evaluación de la propuesta de valor con usuarios potenciales, lo que permitirá confirmar sus necesidades reales y valorar posibles mejoras para versiones posteriores.

En cuanto a los objetivos específicos, se puede decir que durante la elaboración y desarrollo de este proyecto se han alcanzado satisfactoriamente los siguientes hitos u objetivos específicos:

* **Análisis y definición del sistema a desarrollar**: En las primeras fases del proyecto se ha llevado a cabo un análisis exhaustivo para identificar y clasificar los diversos conceptos de gastos asociados a los viajes en moto que son relevantes para la elaboración de presupuestos especializados. Muestra de ello es el contenido de los anexos B. Detalle de los gastos incluidos en el proyecto y C. Detalle de los conceptos informativos incluidos en el proyecto. También se definieron los requisitos funcionales y no funcionales necesarios para establecer el alcance del sistema y las capacidades que debía implementar el módulo presupuestario. El detalle de estos requisitos puede consultarse en los anexos D. Requisitos Funcionales y E. Requisitos No Funcionales. Finalmente, se diseñó una arquitectura de solución que permite la interacción del módulo con otras aplicaciones a través de una API de tipo API REST.
* **Desarrollo de los componentes necesarios**:Se han desarrollado los componentes fundamentales del módulo, incluyendo la lógica de procesamiento de los datos necesarios para realizar los cálculos presupuestarios y la API que facilita la comunicación e integración con sistemas externos, tal y como se había previsto en la fase de diseño.
* **Validación y verificación de la solución desarrollada**: Se ha procedido a la validación y verificación del sistema mediante la realización de pruebas funcionales exhaustivas. El análisis de los resultados obtenidos en estas pruebas ha permitido comprobar la eficacia y la precisión de los presupuestos generados por el módulo, confirmando así su correcto funcionamiento y su carácter realista y práctico. Además, ha implicado la revisión y modificación de la solución desarrollada inicialmente con el fin de mejorar su robustez y fiabilidad.

Cabe destacar que las fases de definición, análisis y diseño de la solución han resultado cruciales para delimitar el alcance de la solución a desarrollar. Por ejemplo, la selección de conceptos tanto informativos y de gastos a incluir para tratar de alcanzar el equilibrio entre un presupuesto lo más completo posible y un resultado práctico, han llevado a la conclusión de la idoneidad de que la solución desarrollada ofrezca tres formatos de salida de los datos: Etiqueta, Resumen y Completo. Esta decisión ha contribuido a la mejora de la funcionalidad aportada por la lógica desarrollada que permite cubrir un abanico mayor de soluciones al aportar respuestas más o menos detalladas según se demande por parte de las aplicaciones cliente y de los usuarios finales.

## Líneas de trabajo futuro

Partiendo de la solución desarrollada en el presente proyecto, y en el caso de constatar tanto el interés como el potencial de aplicación práctica que puede tener dentro del ecosistema de aplicaciones especializadas en la planificación de rutas para viajes en moto mediante pruebas con usuarios potenciales, cabe plantearse la posibilidad de realizar un estudio de viabilidad orientado a la comercialización del producto. En ese contexto, el sistema de presupuestación podría evolucionar desde un Producto Mínimo Viable (PMV) hacia una solución más robusta y versátil, capaz de integrarse plenamente en entornos reales y con usuarios finales. Para ello, sería recomendable abordar una serie de trabajos futuros, tanto desde el punto de vista técnico como funcional, que se detallan a continuación.

Figura . Esquema de las posibles mejoras a futuro.

Diagrama

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Respecto a las mejoras técnicas destacan la **recodificación basada en tecnologías más adecuadas para su uso en entornos web y de aplicaciones móviles** y el **refuerzo de la validación y seguridad de los datos de entrada**.

La solución actual, orientada a la validación del concepto y del modelo de cálculo, puede beneficiarse de una reimplementación utilizando tecnologías y frameworks específicos para el desarrollo multiplataforma. Esto permitiría una mayor adaptabilidad, escalabilidad y facilidad de integración con aplicaciones cliente existentes. Frameworks como ***React***, ***Flutter*** o tecnologías como ***Node.js*** para la lógica de procesamiento (backend) podrían ofrecer mejoras significativas en términos de rendimiento y compatibilidad.

Respecto a la **seguridad**, habría que considerar el incremento de la robustez del sistema mediante la incorporación de mecanismos avanzados de validación de entradas del usuario, con controles estrictos de tipo, rango y coherencia de datos. Igualmente, se propone implementar buenas prácticas de seguridad en las API (como autenticación, autorización y prevención de inyecciones), con el objetivo de garantizar la integridad y fiabilidad del sistema, especialmente en escenarios de acceso concurrente o de integración con servicios externos.

Algunas de las posibles **mejoras funcionales** a desarrollar son:

* **Base de datos con precios orientativos:** Se plantea la creación e integración de una base de datos que contenga precios orientativos o de media de componentes, costes de mano de obra para reparaciones, costes de revisiones frecuentes y costes de equipación de motoristas y de la motocicleta. Esta funcionalidad permitiría ofrecer al usuario valores de referencia para la elaboración de sus presupuestos, con la opción de aceptar los valores sugeridos o introducir sus propios datos personalizados. También permitiría la definición de rangos de precios con el fin de limitar la posibilidad de introducir datos incorrectos. Sin embargo, la recopilación y mantenimiento de esta base de datos requeriría una investigación continua de precios de mercado para que los precios fuesen realistas y actualizados.
* **Cálculo de mantenimiento predictivo y personalizado:** Se propone desarrollar algoritmos que permitan calcular el coste de mantenimiento de forma predictiva y personalizada, basándose en el kilometraje actual de la motocicleta, el historial de la última revisión (elementos sustituidos, reparaciones realizadas, previsiones de mantenimiento futuro) y los kilómetros planificados para el viaje. Esta funcionalidad alertaría sobre posibles mantenimientos o revisiones que podrían ser necesarios durante o después del viaje, ofreciendo una estimación de su coste.
* **Presupuesto como criterio de búsqueda:** Se sugiere la implementación de la capacidad de utilizar el presupuesto estimado de un viaje como un criterio de búsqueda dentro del histórico de rutas almacenadas en las aplicaciones de planificación de viajes en moto. Esto aportaría un valor añadido significativo al permitir a los usuarios encontrar viajes similares o alternativas dentro de un rango de presupuesto determinado.
* **Integración con APIs de previsión meteorológica:** Se plantea el desarrollo de APIs que permitan la integración con servicios externos que proporcionen datos del clima previsto para las localizaciones geográficas por las que discurren las etapas del viaje en las fechas correspondientes. Esta funcionalidad permitiría a los usuarios anticipar posibles gastos adicionales relacionados con condiciones climáticas adversas (equipamiento adicional, cambios de ruta, etc.).
* **Integración con APIs de precios de combustible:** Se propone la creación de APIs para la integración con servicios externos que ofrezcan precios actualizados del combustible en las estaciones de servicio ubicadas a lo largo de la ruta planificada. Esta funcionalidad permitiría ofrecer una estimación más precisa del coste de combustible, teniendo en cuenta las variaciones de precio en diferentes localizaciones.

Estas propuestas de mejora representan una evolución natural del sistema hacia una solución más completa, adaptable y preparada para integrarse con el ecosistema digital actual. Su implementación contribuiría significativamente a consolidar el valor añadido del módulo de presupuestación como una funcionalidad diferenciadora dentro del mercado de aplicaciones de planificación de rutas para motoristas.

Referencias bibliográficas

Blancarte, Oscar (s.f.). *Estilos arquitectónicos: Arquitectura en Capas*. [*https://reactiveprogramming.io/blog/es/estilos-arquitectonicos/capas*](https://reactiveprogramming.io/blog/es/estilos-arquitectonicos/capas)

Blancarte, Oscar (s.f.) *Estilos arquitectónicos: Monolítico*. [*https://reactiveprogramming.io/blog/es/estilos-arquitectonicos/monolitico*](https://reactiveprogramming.io/blog/es/estilos-arquitectonicos/monolitico)

Bunq. (3 de abril de 2025). *Tricount – Gastos compartidos*. [*https://play.google.com/store/apps/details?id=com.tribab.tricount.android&hl=es*](https://play.google.com/store/apps/details?id=com.tribab.tricount.android&hl=es)

Calimoto. (s.f.). *Planificador Web*. [*https://calimoto.com/es/planificador-de-rutas*](https://calimoto.com/es/planificador-de-rutas)

Calimoto GmbH. (28 de marzo de 2025). *Calimoto – GPS para motos*. [*https://play.google.com/store/apps/details?id=com.calimoto.calimoto&hl=es*](https://play.google.com/store/apps/details?id=com.calimoto.calimoto&hl=es)

Cavadas, D., Horcajada, L. y Ramos, A. (9 de abril de 2024). *Cómo aplicar la metodología Scrum y qué es el método Scrum*. [*https://www.apd.es/metodologia-scrum-que-es/*](https://www.apd.es/metodologia-scrum-que-es/)

Creativeart. (s.f.). *Fotografía de un smartphone con mapa de carreteras*. <https://www.freepik.es/foto-gratis/movil-google-maps_926535.htm#fromView=image_search_similar&page=1&position=28&uuid=0c80a35e-d74b-41f2-8634-6df2fe7dca3b&query=navegador+GPS+smartphone>

Danfoss A/S. (7 de febrero de 2020). *MyDrive Connect*. [*https://play.google.com/store/apps/details?id=com.danfoss.mydriveconnect&hl=es\_419*](https://play.google.com/store/apps/details?id=com.danfoss.mydriveconnect&hl=es_419)

Europa Press Economía Finanzas (17 de febrero de 2025). *Los Españoles compartieron en Tricount más de 3.300 millones de euros en 2024, cerca del 20% del total*. <https://www.europapress.es/economia/noticia-espanoles-compartieron-tricount-mas-3300-millones-euros-2024-cerca-20-total-20250217102904.html>

Epicioci. (17 de enero de 2021). *Fotografía de un mapa de carreteras*. <https://pixabay.com/es/photos/mapa-l%C3%A1piz-navegaci%C3%B3n-la-carretera-5911003/>

García López, Aarón. (9 de diciembre de 2024). *Inroute*. [*https://play.google.com/store/apps/details?id=com.adsum.inroute&hl=es*](https://play.google.com/store/apps/details?id=com.adsum.inroute&hl=es)

GasAll. (s.f.). Web oficial. *El mejor repostaje en tu móvil*. [*www.gasallapp.com*](http://www.gasallapp.com)

Gas Biker. (s.f.). *Home*. [*https://www.gasbiker.com*](https://www.gasbiker.com)

Gas Biker Worldwide S.L. (23 de septiembre de 2024). *Gas Biker, la app para moteros*. [*https://play.google.com/store/apps/details?id=com.gasbiker&hl=es*](https://play.google.com/store/apps/details?id=com.gasbiker&hl=es)

Harley-Davidson App. (s.f.). *Harley-Davidson Ride Planner: Key Features*. [*https://www.timbertownhd.com/Service/Tips/Harley-Davidson-App*](https://www.timbertownhd.com/Service/Tips/Harley-Davidson-App)

Harley-Davidson Motor Company. (2 de abril de 2025). *Harley-Davidson*. [*https://play.google.com/store/apps/details?id=com.harley\_davidson.ride\_planner*](https://play.google.com/store/apps/details?id=com.harley_davidson.ride_planner)

Harley-Davidson. (s.f.). *Ride Planner*. [*https://www.harley-davidson.com/rideplanner*](https://www.harley-davidson.com/rideplanner)

Harris, Chandler (s.f.). Atlassian. *Comparación entre la arquitectura monolítica y la arquitectura de microservicios*. [*https://www.atlassian.com/es/microservices/microservices-architecture/microservices-vs-monolith*](https://www.atlassian.com/es/microservices/microservices-architecture/microservices-vs-monolith)

Hopper. (s.f.). Web oficial. *Singin, save money*. [*https://www.hopper.com/*](https://www.hopper.com/)

Huet, Pablo (24 de agosto de 2022). OpenWebinars. *Arquitectura de software: Qué es y qué tipos existen.* [*https://openwebinars.net/blog/arquitectura-de-software-que-es-y-que-tipos-existen/*](https://openwebinars.net/blog/arquitectura-de-software-que-es-y-que-tipos-existen/)

Inesdi – Business Techschool. (14 de agosto de 2024). *Arquitectura de software: 5 patrones principales*. <https://www.inesdi.com/blog/arquitectura-de-software-5-patrones-principales/>

In Route. (s.f.). *Intelligent Routing*. [*https://inroute.com*](https://inroute.com)

Instituto IEA. (16 de enero de 2025). *¿Cuáles son las 5 fases de Scrum?* [*https://www.institutoiea.edu.ar/publicaciones/cuales-son-las-5-fases-de-scrum*](https://www.institutoiea.edu.ar/publicaciones/cuales-son-las-5-fases-de-scrum)

Kayak. (s.f.). *Web oficial*. [*https://www.kayak.es/*](https://www.kayak.es/)

Kayak. (s.f.). *Web oficial: Aplicación móvil*. [*https://www.kayak.es/mobile*](https://www.kayak.es/mobile)

Kayak.com (16 de abril de 2025). *KAYAK: Vuelos, Hoteles, Coches*. [*https://play.google.com/store/apps/details?id=com.kayak.android&hl=es*](https://play.google.com/store/apps/details?id=com.kayak.android&hl=es)

Keepcoding (s.f.) *Conoce el concepto de None en Python*. [*https://keepcoding.io/blog/concepto-de-none-en-python/*](https://keepcoding.io/blog/concepto-de-none-en-python/)

Kurviger. (s.f.). *Tu planificador de rutas en moto: planifica Tourer con curvas y belleza con facilidad*. [*https://kurviger.com/es/features*](https://kurviger.com/es/features)

Kurviger navigation moto. (12 de marzo de 2025). [*https://play.google.com/store/apps/details?id=com.kurviger.app&hl=es\_419*](https://play.google.com/store/apps/details?id=com.kurviger.app&hl=es_419)

Monefy. (s.f.). *Take back control of your money*. [*https://www.monefy.me/*](https://www.monefy.me/)

Núcleo. (2 de enero de 2025). *Las 5 mejores apps para mejorar tus finanzas en este nuevo año.*

<https://nucleo.news/2025/01/las-5-mejores-apps-para-mejorar-tus-finanzas-en-este-nuevo-ano/>

Pirelli & C. S.p.A. (12 de diciembre de 2025). *Diablo Super Biker*. [*https://play.google.com/store/apps/details?id=it.pirelli.diablosuperbiker&hl=es*](https://play.google.com/store/apps/details?id=it.pirelli.diablosuperbiker&hl=es)

Pitet, Sten (s.f.). Atalassian. *Los distintos tipos de pruebas de software*. <https://www.atlassian.com/es/continuous-delivery/software-testing/types-of-software-testing>

Rever. (s.f.). *Planificador web de rutas*. [*https://rever.co*](https://rever.co)

Rever Owner. (17 de enero de 2025). *Rever-GPS motocicleta/paseos*. [*https://play.google.com/store/apps/details?id=com.reverllc.rever&hl=es\_419*](https://play.google.com/store/apps/details?id=com.reverllc.rever&hl=es_419)

Riding moto. (s.f.). *Web oficial*. [*https://ridingmoto.com*](https://ridingmoto.com)

Riser. (s.f.). *Web oficial*. [*https://riserapp.com*](https://riserapp.com)

Riser Gmbh. (2 de abril de 2025). *Riser – Tu aplicación de moto*. [*https://play.google.com/store/apps/details?id=com.riserapp&hl=es*](https://play.google.com/store/apps/details?id=com.riserapp&hl=es)

RoadStr Enthusiast SL. (7 de febrero de 2025). *Rockr – Rutas en Moto*. [*https://play.google.com/store/apps/details?id=io.rockr.app&hl=es*](https://play.google.com/store/apps/details?id=io.rockr.app&hl=es)

RoadTrippers. (s.f.). *Web oficial. Turn your road trip into an adventure*. [*www.roadtrippers.com*](http://www.roadtrippers.com)

Rockr. (s.f.). *Web oficial*. [*https://rockr.io/*](https://rockr.io/)

Scenic. (s.f.). *Web oficial*. [*https://scenicapp.space*](https://scenicapp.space)

Scrum.org. (enero de 2021). *Online Nexus Guide*. [*https://www.scrum.org/resources/online-nexus-guide*](https://www.scrum.org/resources/online-nexus-guide)

Skyscanner. (s.f.). *Web oficial*. [*https://www.skyscanner.es/*](https://www.skyscanner.es/)

Skyscanner (s.f.). *Web oficial: Aplicación móvil*. <https://www.skyscanner.es/movil.html>

Splitwise. (s.f.). *Web oficial. Menos estrés a la hora de compartir gastos con compañeros de piso*. [*https://www.splitwise.com/*](https://www.splitwise.com/)

Tamushi (20 de diciembre del 2025). Testing IT. *¿Qué son las pruebas de software y cómo aplicarlas?*. <https://www.testingit.com.mx/blog/que-son-las-pruebas-de-software>

TomTom: MyDrive. (s.f.). *Planificador web de rutas*. [*https://mydrive.tomtom.com*](https://mydrive.tomtom.com)

Trabee. (s.f.). *Manage your travel Budget with TRABEE POCKET*. [*https://trabeepocket.com/*](https://trabeepocket.com/)

TravelSpend. (s.f.). *Web oficial*. [*https://www.travelspend.com/*](https://www.travelspend.com/)

Tricount. (s.f.). *Web oficial. Easily track and Split group expenses*. [*https://tricount.com/*](https://tricount.com/)

USA-Reiseblogger. (17 de julio de 2020). *Fotografía del navegador GPS de un vehículo*. <https://pixabay.com/es/photos/auto-navegaci%C3%B3n-gps-veh%C3%ADculo-panel-5414514/>

ViaMichelin. (s.f.). *Web oficial*. [*https://www.viamichelin.es/*](https://www.viamichelin.es/)

Yamaha. (s.f.). *Yamaha Motor MYRIDE*. [*https://www.yamaha-motor.eu/es/es/experience/applications/myride/*](https://www.yamaha-motor.eu/es/es/experience/applications/myride/)

Yamaha Motor Europe. (4 de marzo de 2025). *MyRide – Motorcycle Routes*. [*https://play.google.com/store/apps/details?id=com.yamaha.myride&hl=es*](https://play.google.com/store/apps/details?id=com.yamaha.myride&hl=es)

1. Comparativa de aplicaciones específicas para planificar rutas en moto

En este anexo se incluye una tabla en la que se especifica las funcionalidades de las aplicaciones analizadas en relación con la planificación de rutas y viajes en moto.

Tabla .Comparativa de aplicaciones específicas para planificar rutas en moto (Anexo A).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Nombre de la aplicación** | **Calimoto** | **Diablo Super Biker** | **Gas Biker** | **H-D Ride Planner** | **InRoute** | **Kurviger** | **MyDrive** | **Rever** | **Riding Moto** | **Riser** | **Rockr** | **Scenic** | **Yamaha MyRide** |
| **Planificador de rutas** | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí |
| **Selección de preferencias** | Sí | Sí | No | No | Sí | Sí | Sí | Sí | Sí | Sí | No | Sí | Sí |
| **Puntos de interés** | Sí | Sí | No | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí |
| **BD de rutas** | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí |
| **Registro de rutas** | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí |
| **Nombre de la aplicación** | Calimoto | Diablo Super Biker | Gas Biker | H-D Ride Planner | InRoute | Kurviger | MyDrive | Rever | Riding Moto | Riser | Rockr | Scenic | Yamaha MyRide |
| **Mapas Offline** | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | No | Sí | Sí |
| **Comunidad** | Sí | Sí | No | Sí | Sí | Sí | No | Sí | Sí | Sí | Sí | Sí | Sí |
| **Compartir rutas** | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí |
| **Compartir ubicación en tiempo real** | Sí | Sí | Sí | Sí | No | No | No | Sí | Sí | Sí | Sí | No | No |
| **Navegación paso a paso** | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí |
| **Aviso de radares** | No | No | No | No | No | No | Sí | No | No | No | No | No | No |
| **Alertas** | Sí | Sí | Sí | No | Sí | No | Sí | Sí | Sí | No | No | No | No |
| **Altitud** | Sí | Sí | No | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí |
| **Aceleración** | Sí | Sí | No | No | No | No | No | No | No | Sí | Sí | Sí | Sí |
| **Nombre de la aplicación** | Calimoto | Diablo Super Biker | Gas Biker | H-D Ride Planner | InRoute | Kurviger | MyDrive | Rever | Riding Moto | Riser | Rockr | Scenic | Yamaha MyRide |
| **Inclinación** | Sí | Sí | No | No | No | No | No | No | No | No | No | No | Sí |
| **Noticias/ Blog** | Sí | Sí | No | Sí | Sí | Sí | No | Sí | Sí | Sí | Sí | No | Sí |
| **Importar archivos GPX** | Sí | Sí | No | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí |
| **Exportar archivos GPX** | Sí | Sí | No | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí | Sí |
| **Presupuesto** | No | No | No | No | No | No | No | No | No | No | No | No | No |

*Nota: Los datos indicados en esta tabla han sido recogidos entre los días 3, 4 y 5 de marzo de 2025 por lo que pueden variar según la evolución de futuras versiones de las aplicaciones analizadas.*

1. Detalle de los gastos incluidos en el proyecto

Los gastos incluidos en este proyecto son los mostrados en la siguiente tabla en la que también se indica si son datos de entrada, de salida o de entrada y salida:

Tabla . Gastos incluidos en el proyecto (Anexo B).

|  |  |  |
| --- | --- | --- |
| **Concepto** | **Descripción y utilidad** | **Entrada/ Salida** |
| **Gastos Combustible** | Gasto en combustible necesario para la realización del viaje. Se calcula a partir de la suma del gasto en combustible de las etapas que componen el viaje. | Salida |
| **Presupuesto por etapa** | Importe total de los gastos propios de cada etapa. Incluye todos los gastos de una etapa. Se utiliza como concepto de agrupación de gastos por etapa. | Salida |
| **Gasto de combustible por etapa** | Importe del gasto en combustible necesario para recorrer cada etapa. Se calcula multiplicando los kilómetros de cada etapa por el consumo medio de la moto y por el precio del litro de combustible. | Salida |
| **Gasto de alojamiento por etapa** | Importe del gasto en alojamiento de cada etapa, ya sea en hoteles o campings. | Entrada y salida |
| **Gasto de alimentación por etapa** | Importe del gasto total en alimentación y bebida de cada etapa. | Salida |
| **Gasto de desayuno** | Importe del gasto en el desayuno de cada etapa. | Entrada y salida |
| **Gasto de almuerzo** | Importe del gasto en el almuerzo de cada etapa. | Entrada y salida |
| **Gasto de comida** | Importe del gasto en la comida de cada etapa. | Entrada y salida |
| **Gasto de merienda** | Importe del gasto en la merienda de cada etapa. | Entrada y salida |
| **Concepto** | **Descripción y utilidad** | **Entrada/ Salida** |
| **Gasto de cena** | Importe del gasto en la cena de cada etapa. | Entrada y salida |
| **Otros gastos en alimentación** | Importe de otros gastos relacionados con alimentación y bebida en cada etapa. | Entrada y salida |
| **Gasto en parking por etapa** | Importe en gastos de aparcamiento en cada etapa. | Entrada y Salida |
| **Gasto en peajes por etapa** | Importe de gastos en peajes de cada etapa. | Entrada y salida |
| **Gasto en actividades por etapa** | Importe de gastos en actividades de ocio y culturales de cada etapa. | Entrada y salida |
| **Gastos en transporte por etapa** | Importe de gastos en transporte adicional de cada etapa. Por ejemplo, en transporte público o en transporte para la motocicleta (ferry, grúa, etc.). | Entrada y salida |
| **Gastos extra por etapa** | Importe de gastos extra de cada etapa. Permite prever un fondo de gastos adicionales. | Entrada y salida |
| **Total otros gastos** | Importe del total de gastos en compras, equipo de acampada, tasas y seguros asociados al viaje. | Salida |
| **Gastos en compras y souvenirs** | Importe de los gastos en compras y souvenirs a realizar en el viaje. | Entrada y salida |
| **Gastos de equipo de acampada** | Importe de los gastos en equipo de acampada en caso necesario. | Entrada y salida |
| **Gastos en visados** | Importe de los gastos en visados en el caso de que el recorrido del viaje pase por países que lo requieran para acceder a ciudadanos extranjeros. | Entrada y salida |
| **Gastos en otras tasas** | Importe de los gastos en otras tasas o impuestos que puedan ser necesarios a lo largo del viaje. | Entrada y salida |
| **Concepto** | **Descripción y utilidad** | **Entrada/ Salida** |
| **Gastos en pases de circulación** | Importe de los gastos en pases de circulación y otros permisos para circular por carretera que sean requeridos en los países por los que se lleve a cabo el viaje. | Entrada y salida |
| **Gastos en seguros** | Importe total de los gastos en seguros. | Salida |
| **Gastos en seguro de moto** | Importe de los gastos en el seguro de la motocicleta o de ampliación de coberturas necesarias para la realización del viaje. | Entrada y salida |
| **Gastos en seguro de viaje** | Importe de los gastos en contratación de un seguro de viaje. | Entrada y salida |
| **Gastos en seguro de salud** | Importe de los gastos en contratación de un seguro de salud. Es conveniente en viajes al extranjero. | Entrada y salida |
| **Gastos en otros seguros** | Importe de otros gastos derivados de la contratación de pólizas de seguros que se necesiten contratar según cada viaje. | Entrada y salida |
| **Gastos en equipación para piloto y pasajero** | Importe total en gastos de equipación necesaria para el piloto y el pasajero. La equipación adecuada es crucial para que el viaje pueda realizarse con las medidas de protección y confort suficientes. Los siguientes conceptos son los que se han considerado más relevantes. El usuario debería indicar el importe correspondiente a cada uno de ellos en el caso de que sea necesario adquirirlo para la realización del viaje, ya sea por falta o por renovación. Algunos de ellos son obligatorios. | Salida |
| **Casco** | Importe del casco a adquirir, en su caso. | Entrada y salida |
| **Guantes de invierno** | Importe del casco a adquirir, en su caso. | Entrada y salida |
| **Concepto** | **Descripción y utilidad** | **Entrada/ Salida** |
| **Guantes de verano** | Importe de los guantes de verano a adquirir, en su caso. | Entrada y salida |
| **Botas de invierno** | Importe de los guantes de invierno a adquirir, en su caso. | Entrada y salida |
| **Botas de verano** | Importe de las botas de verano a adquirir, en su caso. | Entrada y salida |
| **Chaqueta de invierno** | Importe de la chaqueta de invierno a adquirir, en su caso. | Entrada y salida |
| **Chaqueta de verano** | Importe de la chaqueta de invierno a adquirir, en su caso. | Entrada y salida |
| **Pantalón de invierno** | Importe del pantalón de invierno a adquirir, en su caso. | Entrada y salida |
| **Pantalón de verano** | Importe del pantalón de verano a adquirir, en su caso. | Entrada y salida |
| **Ropa térmica** | Importe de la ropa térmica a adquirir, en su caso. | Entrada y salida |
| **Traje de agua** | Importe del traje de agua a adquirir, en su caso. | Entrada y salida |
| **Airbag** | Importe del chaleco o dispositivo airbag a adquirir, en su caso. | Entrada y salida |
| **Protecciones adicionales** | Importe de las protecciones adicionales a adquirir, en su caso. | Entrada y salida |
| **Kit de primeros auxilios** | Importe del kit de primeros auxilios a adquirir, en su caso. | Entrada y salida |
| **Intercomunicadores** | Importe del sistema de comunicación y/o manos libres a adquirir, en su caso. | Entrada y salida |
| **Otros gastos en equipación piloto y pasajero** | Importe de otros elementos de equipación que puedan ser necesarios tanto para el piloto como para el pasajero. | Entrada y salida |
| **Concepto** | **Descripción y utilidad** | **Entrada/ Salida** |
| **Gastos de mantenimiento** | Importe total de los gastos en mantenimiento relacionados con el viaje | Salida |
| **Kit de reparación de pinchazos** | Importe del gasto en kit de reparación de pinchazos a adquirir, en su caso. | Entrada y salida |
| **Herramientas** | Importe del gasto en herramientas a adquirir. | Entrada y salida |
| **Compresor portátil** | Importe del gasto en compresor portátil a adquirir. | Entrada y salida |
| **Arrancador de batería** | Importe del gasto en arrancador de batería a adquirir, en su caso. | Entrada y salida |
| **Revisión previa** | Importe del gasto en la revisión previa al viaje. | Entrada y salida |
| **Talleres** | Importe de mano de obra de mecánico. | Entrada y salida |
| **Neumáticos** | Importe del gasto en neumáticos a adquirir. | Entrada y salida |
| **Frenos** | Importe del gasto en pastillas de frenos a adquirir, en caso necesario. | Entrada y salida |
| **Transmisión** | Importe del gasto en mantenimiento del kit de transmisión, en su caso. | Entrada y salida |
| **Otros gastos en mantenimiento** | Otros importes relacionados con el mantenimiento de la motocicleta de cara al viaje. | Entrada y salida |
| **Revisión posterior al viaje** | Importe de la revisión posterior al viaje. | Entrada y salida |
| **Presupuesto total** | Importe de la suma total de todos los gastos incluidos en el presupuesto del viaje. | Salida |
| **Presupuesto por persona** | Importe del presupuesto total repartido entre piloto y pasajero. | Salida |

Fuente: Elaboración propia.

1. Detalle de los conceptos informativos incluidos en el proyecto.

Algunos de estos conceptos pueden ser utilizados como criterios de búsqueda o selección de rutas en el caso de que la aplicación cliente disponga de una base de datos de rutas y viajes, hecho bastante común como se puede ver en la tabla comparativa del Anexo A. Comparativa de aplicaciones específicas para planificar rutas en moto.

Los datos informativos incluidos en este proyecto son los mostrados en la siguiente tabla en la que también se indica si son datos de entrada, de salida o de entrada y salida:

Tabla . Conceptos informativos (Anexo C).

|  |  |  |
| --- | --- | --- |
| **Concepto** | **Descripción y utilidad** | **Entrada/ Salida** |
| **Nombre del viaje** | Se trata de un nombre para identificar cada viaje o ruta. | Entrada y salida |
| **Destino Principal** | Identifica el destino principal del viaje. | Entrada y salida |
| **Kilómetros totales** | Calculados a partir de la suma de los kilómetros de cada etapa. En este proyecto se considera como dato de salida, aunque también podría ser un dato de entrada y salida en el caso de que este dato fuese facilitado por la aplicación cliente. | Salida |
| **Kilómetros de media por etapa** | Se calcula a partir del número total de kilómetros del viaje dividido entre el número de etapas. | Salida |
| **Etapas o días de viaje** | Número de etapas o días del viaje. Se calcula a partir del número de etapas que componen el viaje. | Salida |
| **Fecha inicial del viaje** | Es la fecha de inicio del viaje. Se corresponde con la fecha de la primera etapa del viaje. | Salida |
| **Concepto** | **Descripción y utilidad** | **Entrada/ Salida** |
| **Precio del combustible** | Este dato hace referencia al precio medio del combustible adquirido a lo largo del viaje. Se trata de un valor esencial para el cálculo del gasto de combustible tanto a nivel de etapas como respecto al total del viaje. | Entrada y salida |
| **Pasajero** | Informa acerca de si el viaje lo realiza solamente el piloto o si va acompañado de otra persona. Se utiliza como criterio a la hora de repartir los costes del viaje por persona. | Entrada y salida |
| **Fecha final del viaje** | Es la fecha de fin del viaje. Se corresponde con la fecha de la última etapa del viaje. | Salida |
| **Consumo medio de la moto** | Consumo medio de combustible de la motocicleta cada 100 kilómetros. Hoy en día la mayoría de las motocicletas informan de este dato gracias a los ordenadores de abordo que incorporan, por lo que es un dato fácil de obtener por el usuario. | Entrada y salida |
| **Litros de combustible** | Son los litros de combustible necesarios para la realización del viaje. Se calcula a partir de la suma de los litros de combustible a consumir en cada etapa. | Salida |
| **Velocidad media total** | Se corresponde con la velocidad media de todas las etapas. | Salida |
| **Tiempo total en moto** | Se trata del tiempo total de desplazamiento en moto necesario para la realización del viaje. Se calcula sumando el tiempo de desplazamiento de cada etapa. | Salida |
| **Tiempo en moto de media por etapa** | Se corresponde con el tiempo medio de desplazamiento en moto por etapa. | Salida |
| **Número de etapa** | Se corresponde con el orden de realización de la etapa. | Salida |
| **Concepto** | **Descripción y utilidad** | **Entrada/ Salida** |
| **Destino de la etapa** | Ciudad origen de la etapa. | Entrada y salida |
| **Origen de la etapa** | Ciudad origen de la etapa. | Entrada y salida |
| **Kilómetros de la etapa** | Número de kilómetros de cada etapa | Entrada y salida |
| **Fecha de la etapa** | Se trata de la fecha de realización de la etapa. En formato (dd/mm/aaaa). | Entrada y salida |
| **Velocidad media por etapa** | Velocidad media estimada para cada etapa en kilómetros por hora (kms/h). | Entrada y salida |
| **Tiempo en moto por etapa** | Tiempo de desplazamiento en moto de cada etapa. Se calcula a partir de los kilómetros y de la velocidad media de cada etapa. | Salida |
| **Litros de combustible por etapas** | Se trata de los litros de combustible necesarios para recorrer cada etapa. Se calcula a partir de los kilómetros de la etapa y del consumo medio de la motocicleta. | Salida |

Fuente: Elaboración propia.

1. Requisitos Funcionales

Los requisitos funcionales definidos para el diseño y desarrollo del módulo de presupuestación especializado en viajes en moto son los siguientes:

* **RF1- Cálculo de los presupuestos**: El sistema tiene que calcular el coste de viajes en moto a partir del importe de los gastos de entrada y de los datos informativos necesarios recibidos vía API desde la aplicación cliente.
  + **RF1.1- Cálculo del gasto en combustible**: El cálculo del gasto en combustible deberá realizarse tanto a nivel de cada etapa del viaje como a nivel del total del viaje. El gasto en combustible por etapa se realizará a partir de la multiplicación de los litros de combustible consumidos en la etapa por el precio del litro de combustible. El cálculo del gasto en combustible total se calculará multiplicando el número total de litros de combustible del viaje por el precio del litro de combustible.
  + **RF1.2- Cálculo del presupuesto por etapa:** Existen gastos que son propios de cada etapa y gastos que son propios del viaje en general. En este caso, el sistema deberá ser capaz de calcular los gastos de cada etapa por separado a partir de la suma de todos los gastos propios de cada etapa.
  + **RF1.3- Cálculo del gasto en alimentación por etapa**: Se calculará a partir de la suma de los gastos relacionados con alimentación y bebida de cada etapa, es decir, se calcula sumando los importes de los gastos de: Desayuno, Almuerzo, Comida, Merienda, Cena y otros gastos en alimentación y bebida.
  + **RF1**.**4- Cálculo del total de “otros gastos”:** El sistema deberá calcular el total de los gastos considerados como “otros gastos” que son los relacionados con las compras, gastos en equipación de acampada, tasas, impuestos y seguros, todo ello en relación al viaje a presupuestar.
  + **RF1**.**5- Cálculo del total del gasto en seguros:** Se calculará a partir de la suma de los gastos de los seguros relacionados con el viaje, es decir, el seguro de la motocicleta o la ampliación de coberturas necesarias y otros seguros como el de viaje y salud.
  + **RF1**.**6- Cálculo del gasto total en equipación para el piloto y el pasajero:** Se calculará a partir de la suma de los gastos relacionados con la equipación del piloto y del pasajero. En esta partida se incluyen: casco, guantes, botas, chaquetas, pantalones, ropa térmica, traje de agua, airbags, protecciones adicionales, kit de primeros auxilios, intercomunicadores, y cualquier otro elemento relacionado.
  + **RF1**.**7- Cálculo del gasto total en equipación para la moto:** Se calculará a partir de la suma de los gastos en equipación para la moto como: navegador GPS, maletas, bolsas de equipaje y sus soportes, funda, extintor, etc.
  + **RF1.8- Cálculo del gasto total en mantenimiento:** Se calculará a partir de la suma de los gatos relacionados con el mantenimiento de la motocicleta de cara a la preparación y realización del viaje incluyendo, si fuese necesario, el coste del mantenimiento una vez realizado el viaje.
  + **RF1.9- Cálculo del presupuesto total:** Es el cálculo principal que debe proporcionar el sistema. El presupuesto total del viaje se calculará a partir de la suma de todos los gastos contemplados en el presupuesto.
  + **RF1.10- Cálculo del presupuesto por persona:** En el caso de que se indique que el viaje incluye pasajero/a el sistema deberá indicar el presupuesto por persona que se corresponderá con el 50% del presupuesto total del viaje.
* **RF2- Procesamiento de datos del viaje:** El sistema deberá ser capaz de procesar los datos informativos de entrada con el fin de que sean tenidos en cuenta tanto para la realización de los cálculos necesarios como para ser mostrados como resultados. Estos datos proporcionan una información valiosa a la hora de planificar un viaje en moto por lo que el sistema también debe procesarlos.
  + RF2.1- Cálculo de los kilómetros totales del viaje: Los kilómetros totales se calculan a partir de la suma de los kilómetros de las etapas que componen el viaje.
  + **RF2.2- Cálculo de los kilómetros de media por etapa:** Se calculará a partir del número total de kilómetros dividido entre el número de tapas del viaje.
  + **RF2.3- Número de etapas:** El sistema deberá calcular el número total de etapas del viaje a partir de los datos de entrada y también deberá identificar cada etapa con un número correspondiente al orden previsto para su realización.
  + **RF2.4- Cálculo de las fechas de inicio y fin del viaje:** La fecha de inicio del viaje se corresponderá con la fecha de la primera etapa y la fecha de fin de viaje se corresponderá con la fecha de la última etapa del viaje.
  + **RF2.5- Cálculo del consumo de litros de combustible:** El sistema calculará el total de litros de combustible necesarios para cada etapa y el total de litros de combustible para la realización del viaje completo.
  + **RF2.6- Cálculo de la velocidad media del viaje:** Se calculará a partir de la media de la velocidad media de cada etapa del viaje, recibida como dato de entrada.
  + **RF2.7- Cálculo del tiempo de desplazamiento:** El sistema deberá calcular el tiempo de desplazamiento en moto dando como resultado el tiempo total, el tiempo medio por etapa y el tiempo concreto por etapa.
* **RF3- Formatos de salida:** El sistema deberá proporcionar tres formatos de salida de los datos con el fin de facilitar la obtención de los datos según las necesidades del usuario y de la aplicación cliente.
  + **RF3.1- Formato Etiqueta:** Se trata de un formato con unos pocos datos identificativos del viaje y de su presupuesto correspondiente. Este formato se podría utilizar como información mínima a mostrar al usuario a la hora de realizar búsquedas de rutas disponibles, en el caso de que la aplicación cliente disponga de una base de datos de rutas o viajes.
  + **RF3.2- Formato Resumen:** Este formato debe incluir, además de los datos del formato anterior, otros datos adicionales acerca del viaje y un presupuesto resumido por partidas, pero sin entrar en el detalle de cada concepto.
  + **RF3.3- Formato Completo:** Este formato debe incluir todos los conceptos detallados tanto acerca de la información del viaje como de cada uno de los gastos incluidos estructurados por partidas. Este formato sería el que proporcionaría todos los datos contemplados en este proyecto.
* **RF4- Integración con aplicación de origen mediante API:** El módulo de cálculo presupuestario debe comunicarse con aplicaciones clientes vía API con el fin de establecer la entrada de los datos necesarios para su procesamiento y la posterior salida de los datos resultantes.

1. Requisitos no funcionales

Teniendo en cuenta que el objetivo es el desarrollo de un Producto Mínimo Viable, los requisitos **no** funcionales necesarios para el diseño y desarrollo del módulo de presupuestación especializado en viajes en moto son los siguientes:

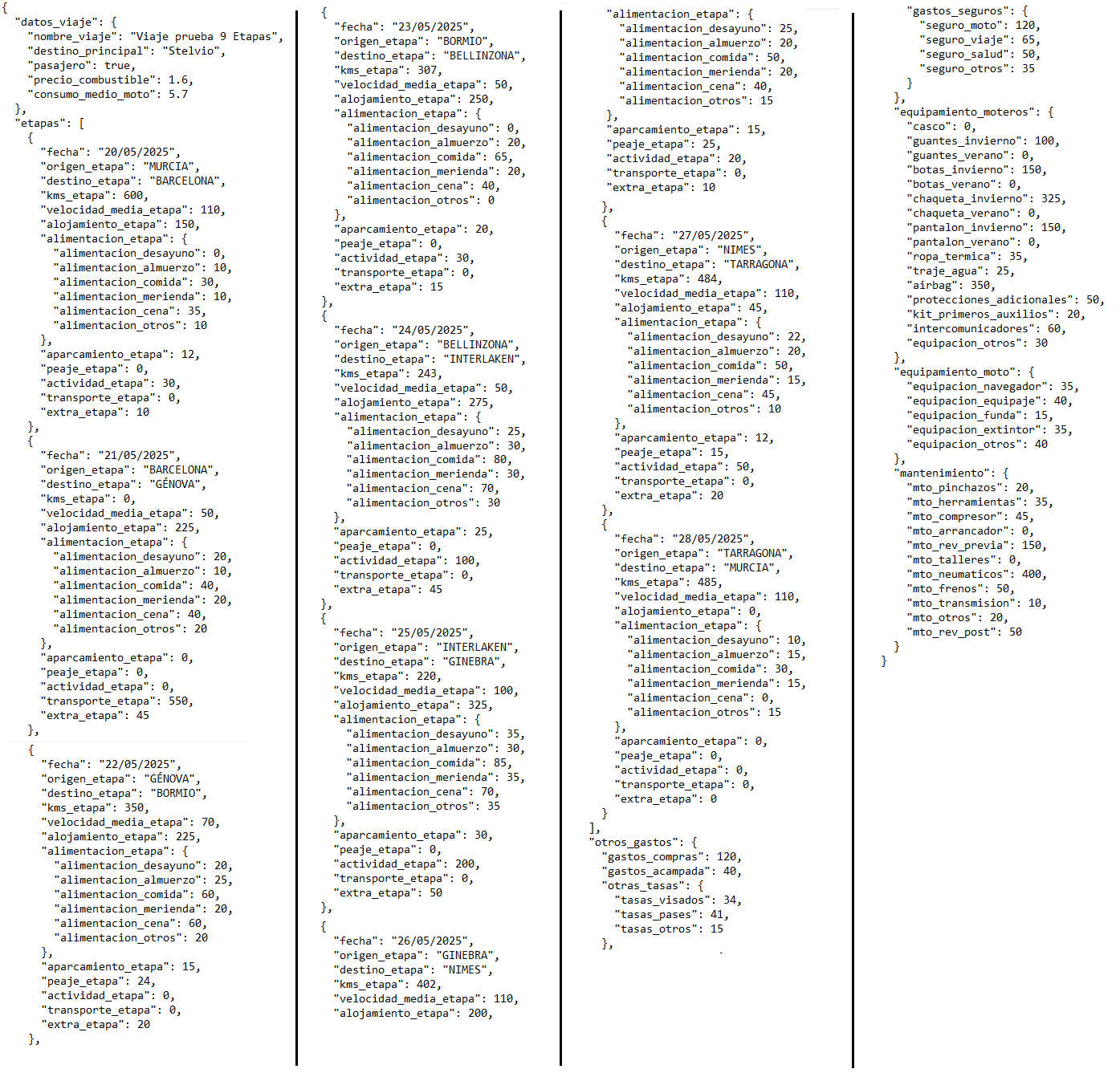
* **RNF1- Fiabilidad:** El módulo de presupuestación debe garantizar que, ante los mismos datos de entrada, los resultados obtenidos sean siempre consistentes y reproducibles. Esto implica que los cálculos y estimaciones generados por el sistema mantengan coherencia lógica y matemática, evitando variaciones injustificadas. Además, los resultados deben ser lo más realistas posibles, aproximándose a los costes reales que un usuario podría afrontar en un viaje en moto, dentro de los márgenes razonables que permiten los datos disponibles.
* **RNF2- Usabilidad:** La interfaz y la lógica del módulo deben estar diseñadas para facilitar su comprensión y uso tanto por parte de desarrolladores, encargados de integrarlo o ampliarlo, como de usuarios finales, quienes interactuarán con la herramienta a través de aplicaciones de planificación de viajes. La experiencia de usuario debe ser intuitiva, clara y accesible, permitiendo que las funcionalidades principales se utilicen sin necesidad de formación especializada.
* **RNF3- Interoperabilidad:** El módulo debe incorporar una API que actúe como pasarela de comunicación, permitiendo la integración fluida con aplicaciones de terceros que gestionan la planificación de rutas y viajes en moto. Esta API debe soportar mecanismos estándar de intercambio de datos para facilitar la entrada y salida de información sin requerir modificaciones complejas en los sistemas externos.
* **RNF4- Rendimiento:** Aunque el proyecto se centra en el desarrollo de un Producto Mínimo Viable y no se requiere una optimización avanzada, el módulo debe ofrecer tiempos de respuesta adecuados para no afectar negativamente la experiencia del usuario ni el rendimiento general de la aplicación cliente. El procesamiento de datos y la generación de presupuestos deben realizarse de manera ágil, asegurando una interacción fluida durante su uso.

1. Archivos utilizados en las pruebas de análisis de precisión de resultados

En este anexo se incluyen capturas de pantalla de los archivos utilizados como entrada de datos para la realización de las pruebas de precisión de los presupuestos, los archivos resultantes en los tres formatos disponibles: Completo, Resumen y Etiqueta, y las capturas de los resultados correspondientes recogidos en la hoja de cálculo para su comprobación y contraste.

* Capturas de la prueba A3 Viaje 9 Etapas: Figuras 46 a 51.

Figura . Composición código archivo P.A3 JSON ENTRADA Viaje 9 Etapas (Anexo F).



Fuente: Elaboración propia.

Figura . Composición del código de la respuesta Completa de la prueba B3 Viaje de 9 Etapas (Anexo F).

Fuente: Elaboración propia.

Figura . Captura del código de respuesta Resumen de la prueba A3 Viaje 9 Etapas (Anexo F).

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Figura . Captura del código de la respuesta Etiqueta de la prueba A3 Viaje 9 Etapas (Anexo F).

Texto, Carta

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

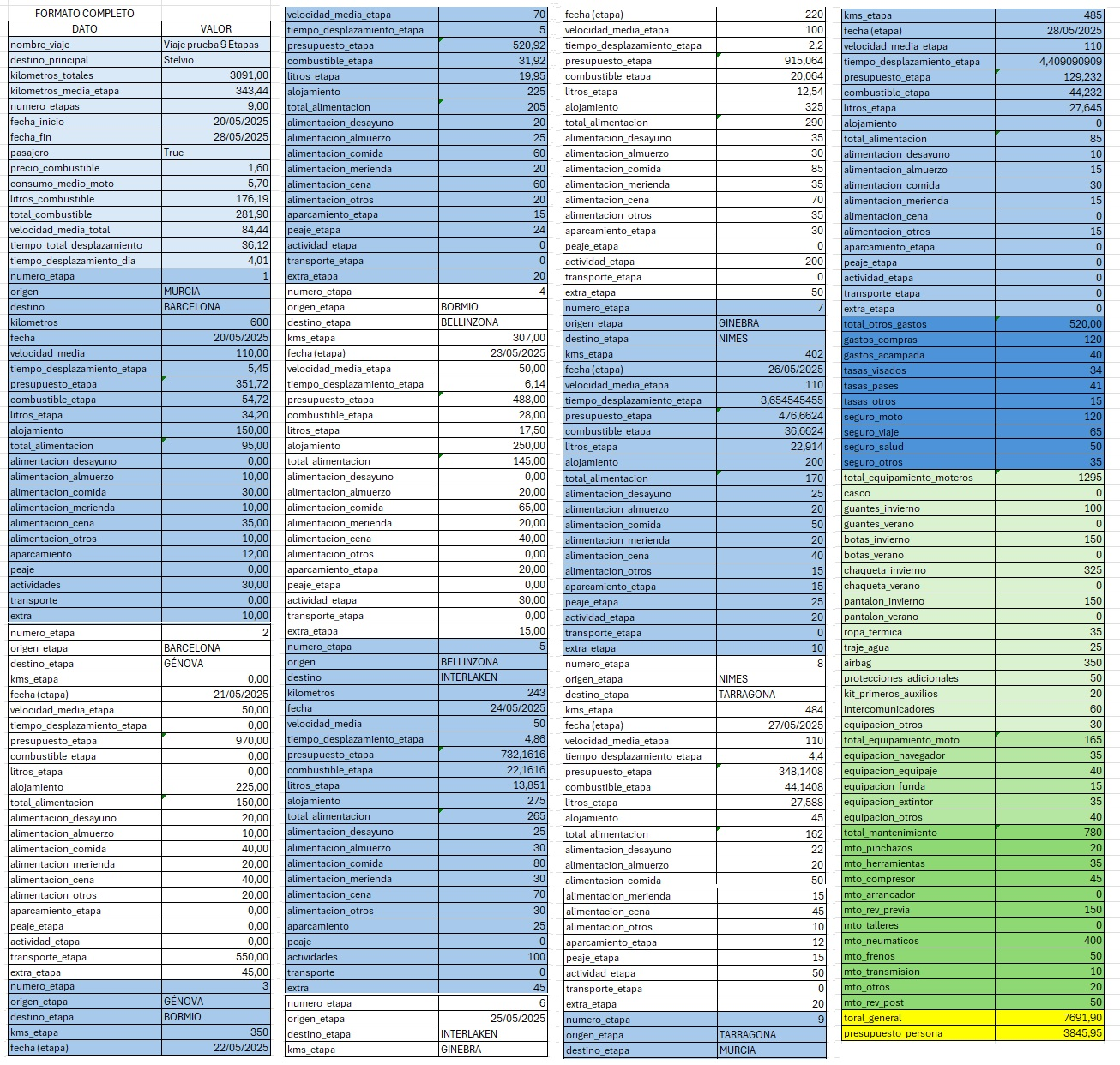
Figura . Captura de la hoja de cálculo con los resultados en formato Etiqueta y Resumen de la prueba A3 Viaje 9 Etapas (Anexo F).

Interfaz de usuario gráfica, Aplicación, Tabla, Excel

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

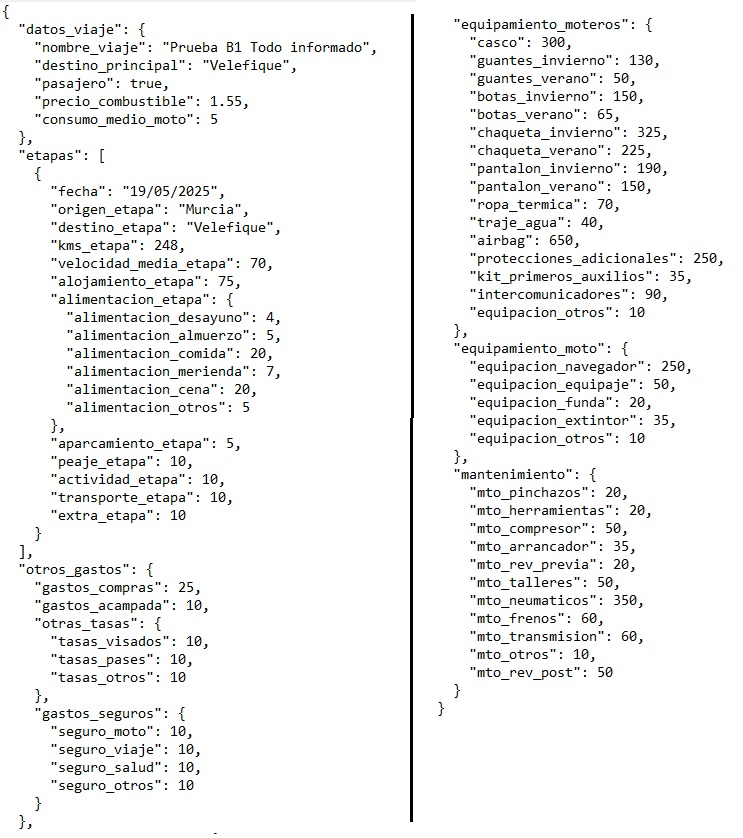
Figura . Captura de la hoja de cálculo con el resultado Completo de la prueba A3 Viaje 9 Etapas (Anexo F).



Fuente: Elaboración propia.

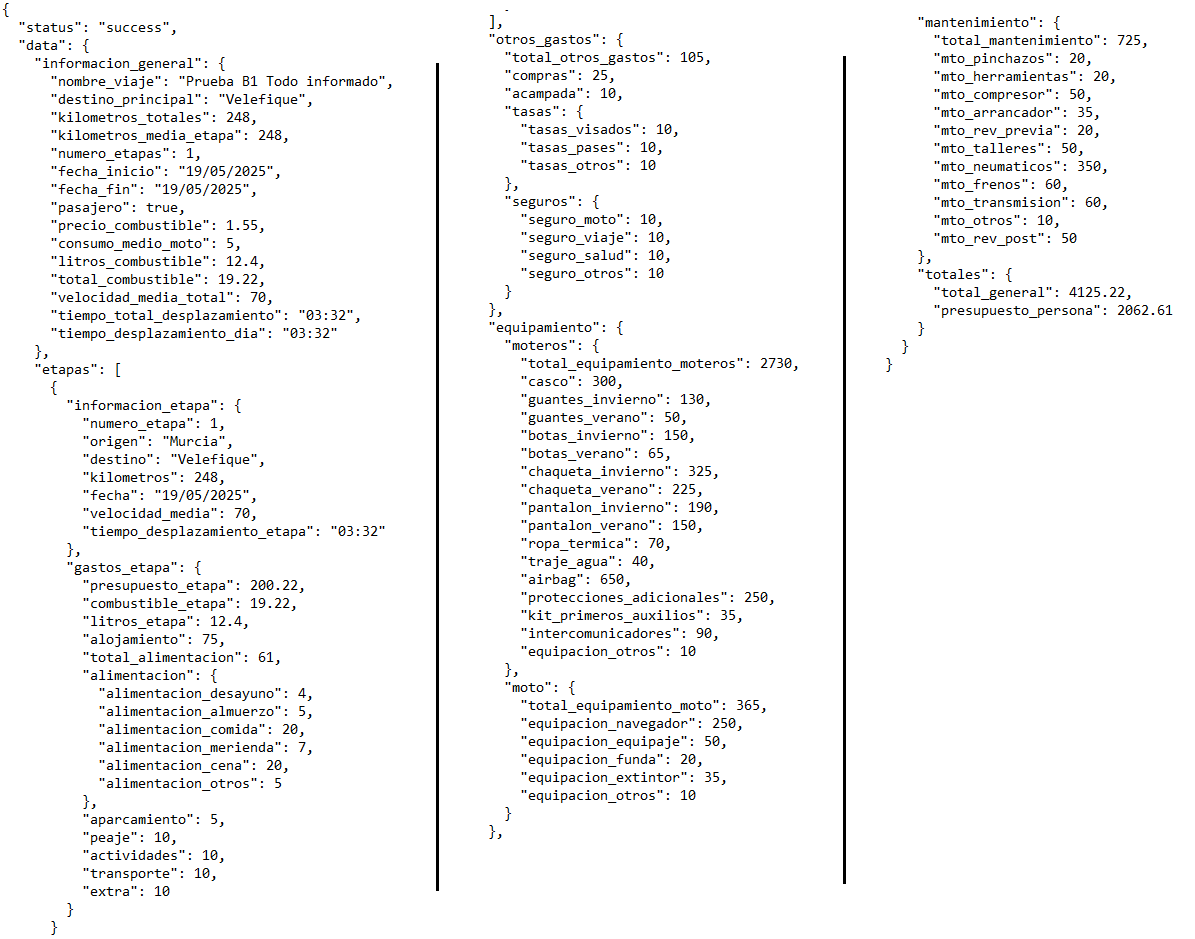
* Capturas de la prueba B1 : Figuras 52 a 57.

Figura . Composición del código del archivo JSON ENTRADA Prueba B1 Todos los datos informados (Anexo F).



Fuente: Elaboración propia.

Figura . Composición del código de la respuesta Completa de la prueba B1 Todos los conceptos informados (Anexo F).



Fuente: Elaboración propia.

Figura . Captura código de respuesta en formato Resumen de la prueba B1 Todos los conceptos informados (Anexo F).

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Figura . Captura del código de respuesta en formato Etiqueta de la prueba B1 Todos los conceptos informados (Anexo F).

Texto, Carta

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

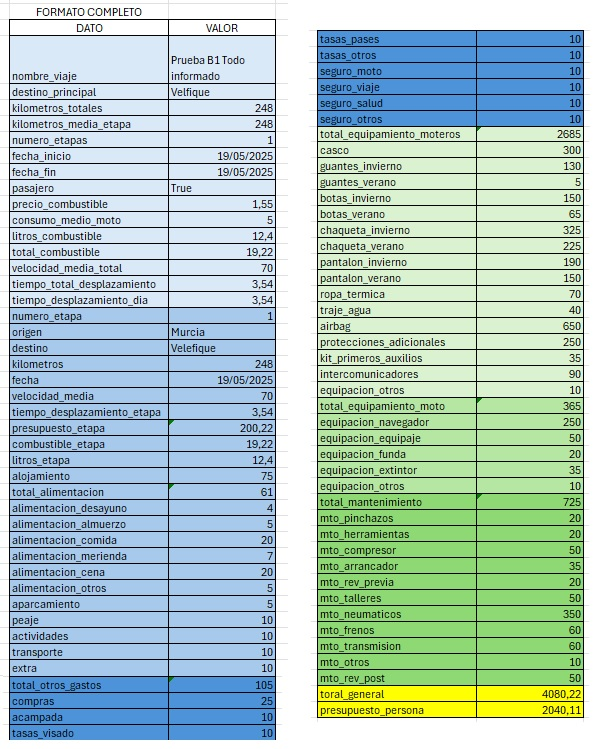
Figura . Captura de la hoja de cálculo con los resultados Etiqueta y Resumen de la prueba B1 Todos los conceptos informados. (Anexo F).

Tabla

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

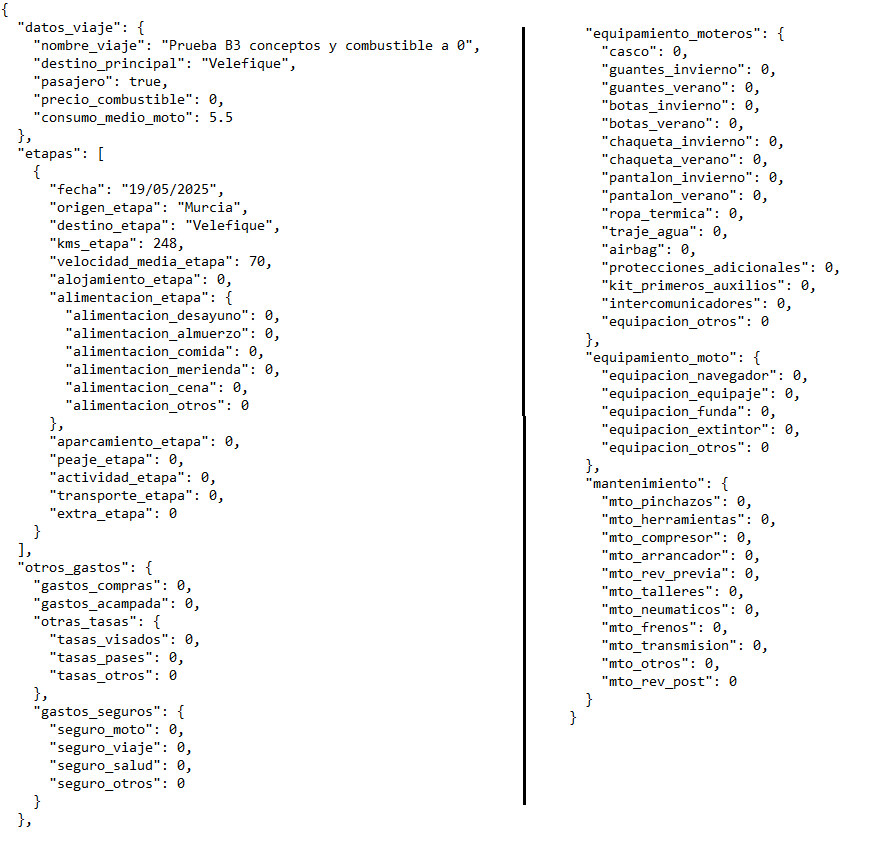
Figura . Composición con la respuesta en formato Completo de la prueba B1 Todos los conceptos informados (Anexo F).



Fuente: Elaboración propia.

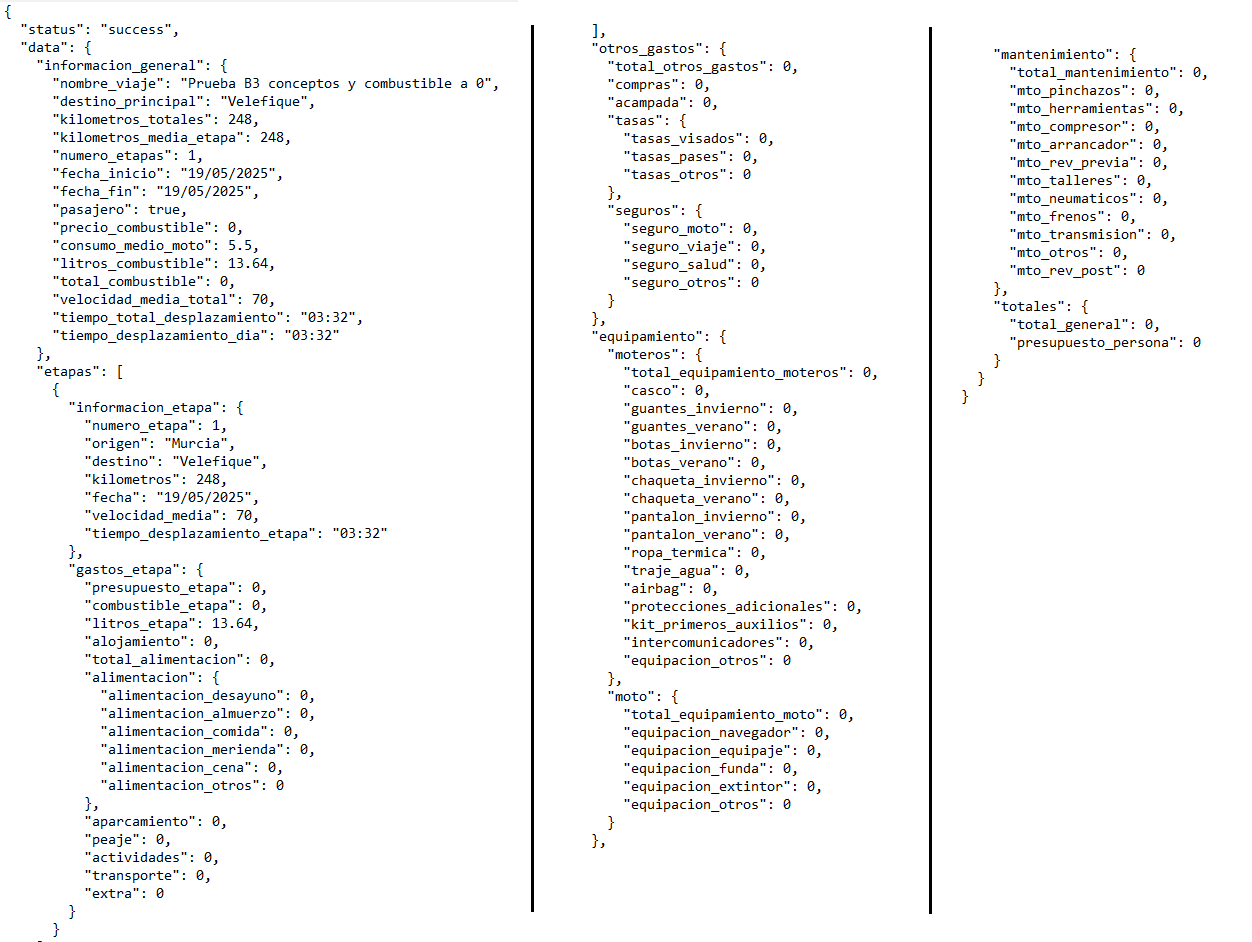
* Capturas de la prueba B3 : Figuras 58 a 63.

Figura . Composición del código del archivo JSON ENTRADA Prueba B3 (Anexo F).



Fuente: Elaboración propia.

Figura . Composición de la respuesta en formato completo de la prueba B3 (Anexo F).



Fuente: Elaboración propia.

Figura . Captura de la respuesta en formato Resumen de la prueba B3 (Anexo F).

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Figura . Captura de la respuesta en formato Etiqueta de la prueba B3 (Anexo F).

Texto, Carta

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Figura . Captura de la hoja de cálculo con los resultados Etiqueta y Resumen de la prueba B3 (Anexo F).

Tabla

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

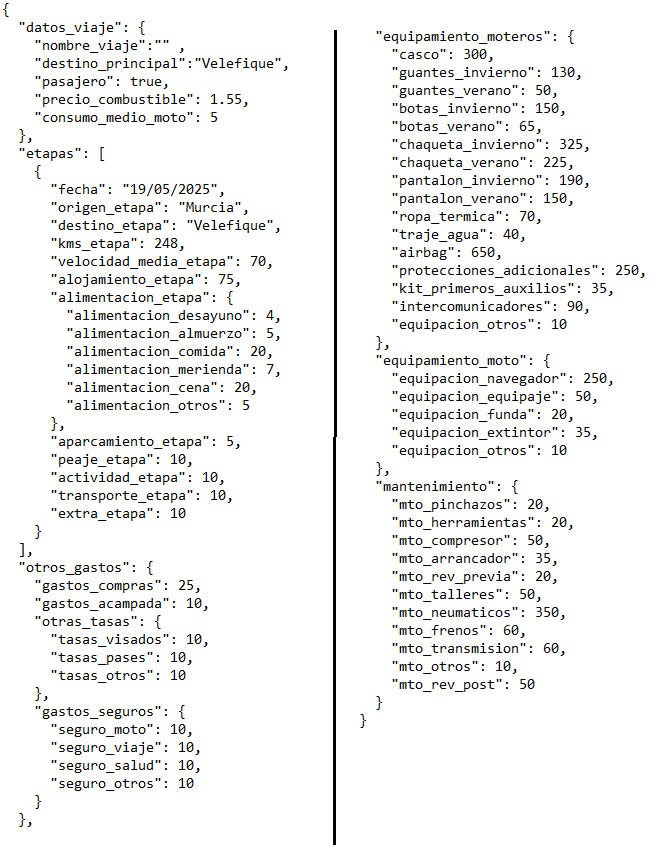
Figura . Composición de la hoja de cálculo con la respuesta Completa de la prueba B3 (Anexo F).



Fuente: Elaboración propia.

* Capturas de la prueba B6 : Figuras 103 y 104.

Figura . Composición del código del archivo JSON ENTRADA Prueba B6 (Anexo F).



Fuente: Elaboración propia.

Figura . Captura del código de la respuesta común a todos los formatos de la prueba B6 (Anexo F).

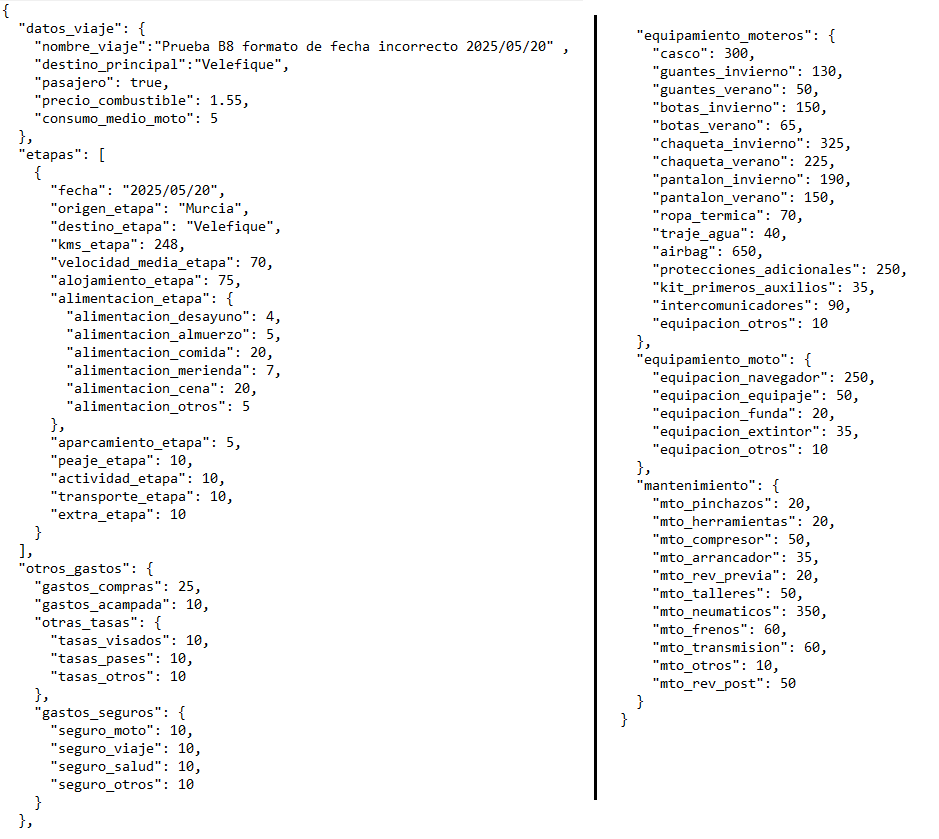
Imagen que contiene Logotipo

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

* Capturas de la prueba B8 : Figuras 66 y 67.

Figura . Composición del código del archivo JSON ENTRADA Prueba B8 (Anexo F).



Fuente: Elaboración propia.

Figura . Captura de la respuesta de la prueba B8 en todos los formatos de salida (Anexo F).

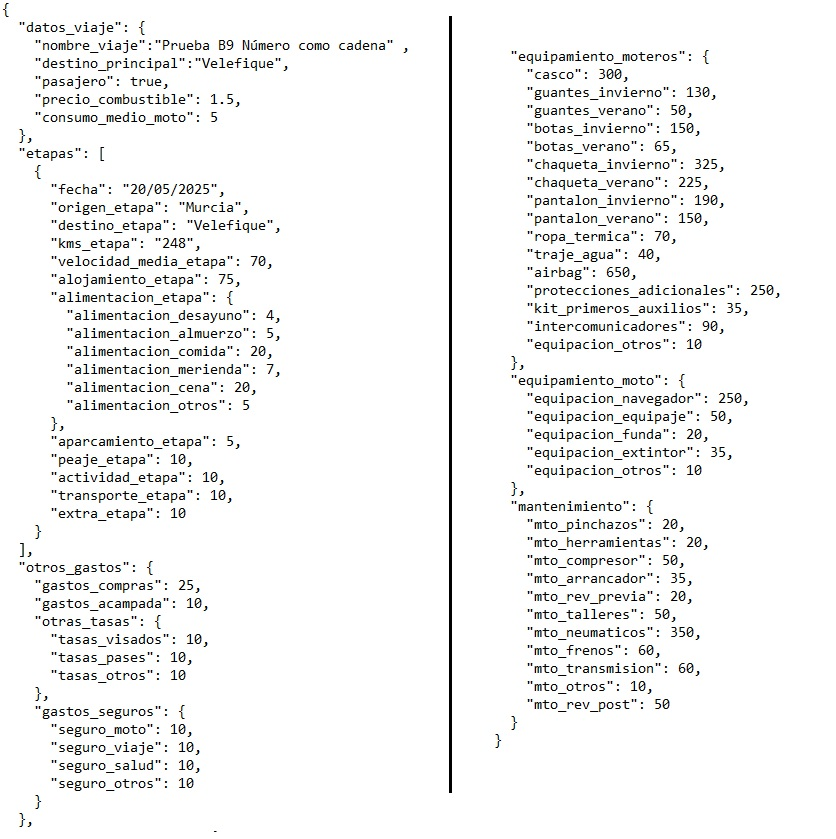
Logotipo

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

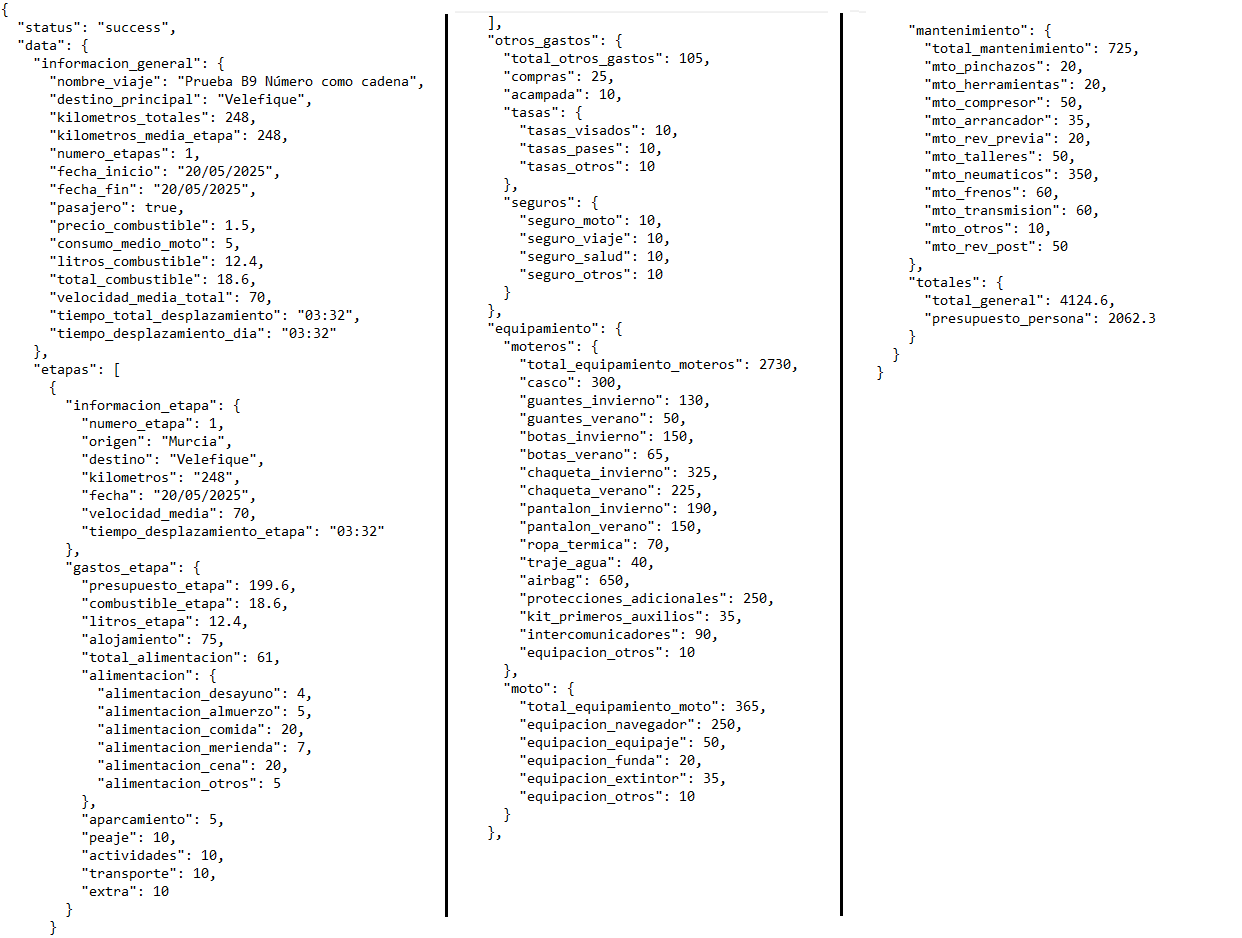
* Capturas de la prueba B9 : Figuras 68 a 71.

Figura . Composición del código del archivo JSON ENTRADA Prueba B9 (Anexo F).



Fuente: Elaboración propia.

Figura . Composición código de respuesta en formato Completo de la prueba B9 (Anexo F).



Fuente: Elaboración propia.

Figura . Captura del código de respuesta en formato Resumen de la prueba B9 (Anexo F).

Texto

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Figura . Captura del código de respuesta en formato Etiqueta de la prueba B9 (Anexo F).

Texto, Carta

El contenido generado por IA puede ser incorrecto.

Fuente: Elaboración propia.

Índice de acrónimos

**A**

API: Application Programming Interface (Interfaz de Programación de Aplicaciones)

**F**

FastAPI: Framework moderno para construir APIs con Python

**G**

GPS: Global Positioning System

**H**

HTTP: HyperText Transfer Protocol

**I**

IDE: Integrated Development Environment (Entorno de Desarrollo Integrado)

**J**

JSON: JavaScript Object Notation

**M**

MIT: Massachusetts Institute of Technology (Licencia de software libre)

**P**

PMV: Producto Mínimo Viable

**R**

REST: Representational State Transfer

**U**

UI: User Interface (Interfaz de Usuario)

1. None: En Python, este valor representa la ausencia de un valor o la falta de contenido de una variable. (Fuente: Keepcoding (s.f.) Conoce el concepto de None en Python. https://keepcoding.io/blog/concepto-de-none-en-python/ [↑](#footnote-ref-1)