Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias de la Computación

PLATAFORMA COLABORATIVA DE APOYO AL ARTESANO BASADA EN SCRUM

Tesis presentada para obtener el grado de

Licenciado en Ciencias de la Computación

Presenta:

Cristian Jordán Nahuatlato De León

Asesora de Tesis:

María Luz Adolfina Sánchez Gálvez

Asesor de Tesis:

Mario Anzures García

Fecha de presentación:

Diciembre 2023

Índice

1. INTRODUCCIÓN. 6
2. APLICACIONES SIMILARES. 9
   2.1. MERCADO LIBRE 9
   2.2. AMAZON. 9
   2.3. ARTESALIZ. 10
   2.4. LA RANCHERITA. 10
   2.5. ARTESANÍAS DE MÉXICO 10
   2.6. EXIXO. 10
   2.7. ESTILO MEXICANO. 11
   2.8. FLOR DE PIÑA. 11
   2.9. ARTESANÍAS DE MÉXICO 11
   2.10. ENSAMBLE ARTESANO 12
   2.11. GAALXIM 13
   2.12. CONCLUSIÓN DE APLICACIONES SIMILARES 13
3. MARCO TEÓRICO. 15
   3.1 HERRAMIENTAS 15
   3.2 METODOLOGÍAS AGILES 18
4. DESARROLLO. 22
   4.1 HISTORIAS DE USUARIO CLIENTE Y VENDEDOR 23
      4.1.1 Diagrama de casos de uso usuarios vendedor y comprador 23
   4.2 HISTORIAS DE USUARIO ADMINISTRADOR 36
      4.2.1 Diagrama de casos de uso administrador 36
   4.3 DIAGRAMA DE BASE DE DATOS 44
   4.4 BACKLOG 54
   4.5 ARTXICANS 56
      4.5.1 Conexión a base de datos 56
      4.5.2 Inicio de sesión 58
         4.5.2.1 Inicio de sesión 59
         4.5.2.2 Registro de nuevo usuario 60
      4.5.3 Inicio o índex 62
      4.5.4 Categorías 66
      4.5.5 vender 68
         4.5.5.1 Usuario nuevo 70
         4.5.5.2 Usuario vendedor 73
      4.5.6 Ayuda 87
      4.5.7 Idioma 88
      4.5.8 Carrito de compras 89
      4.5.9 Administrador 100
1 CONCLUSIONES Y TRABAJO A FUTURO 113
6. BIBLIOGRAFÍA. 114

Índice de figuras.

Figura 1. Artesanías de México Página principal. 12
Figura 2. Ensamble Artesano inicio. 12
Figura 3. Gaalxim Página de inicio. 13
Figura 4. MySQL 16
Figura 5. Visual Studio Code 17
Figura 6. Git Bash y GitHub 17
Figura 7. xampp 18
Figura 8. Paypal 18
Figura 9. Diagrama de ASD 20
Figura 10. Kanban ejemplo de tablero 21
Figura 11. Diagrama casos de uso usuarios 23
Figura 12. Diagrama casos de uso Administrador. 36
Figura 13. Diagrama de Base de datos. 44
Figura 14. Archivo de conexión: "Conexion.php" 57
Figura 15. Formulario inicio de sesión. 59
Figura 16. Código de inicio de sesión. 59
Figura 17. Formulario de registro. 60
Figura 18. Código de registro 61
Figura 19. Inserción de datos de registro. 62
Figura 20: Índex de Artxicans. 62
Figura 21. Query de consulta. 63
Figura 22. Muestreo de información. 64
Figura 23. Footer o pie de página. 65
Figura 24. Archivo "pie.php" 65
Figura 25. porción de código en "pie.php" 65
Figura 26. Archivo "categories.php" 66
Figura 27. Parámetros para categorías. 66
Figura 28. Categoría: Alebrijes. 66
Figura 29. Archivo "pcategoria.php" 67
Figura 30. "pcategoria.php" código BackEnd. 67
Figura 31. Vender usuario "nuevo" 68
Figura 32. Menú de vendedor. 69
Figura 33. "validate-seller.php" 69
Figura 34. FAQ interfaz "vender". 70
Figura 35.Inicio del registro para vendedor. 71
Figura 36. "regseller.php" validación de datos 72
Figura 37. Validación de información. 72
Figura 38. Consulta query para inserción. 73
Figura 39. Mis productos, menú vendedor. 74
Figura 40. query de búsqueda lista de productos. 74
Figura 41. Formulario para editar información del producto. 76
Figura 42. Código de formulario Editar 77
Figura 43. Eliminar producto. 78
Figura 44. Código del formulario Eliminar 78
Figura 45. Formulario de Registro de producto. 79
Figura 46. BackEnd agregar producto. 80
Figura 47. Mis pedidos 81

Figura 48. Detalles de pedido. 81
Figura *49*. BackEnd estado de envío. 83
Figura 50. Especificación de mensaje para notificación. 83
Figura 51. Asignación de código de rastreo. 84
Figura 52. Interfaz perfil de vendedor. 85
Figura 53. Interfaz Editar perfil. 86
Figura 54. Actualizar/Guardar token de pago. 86
Figura 55. Formulario de Mi cuenta de pago. 87
Figura 56. FAQ. 87
Figura 57. Menú de cambio de idioma. 88
Figura 58. API Google Cloud Translation. 88
Figura 59. Botón comprar. 89
Figura 60. Desencriptación de datos. 90
Figura 61. Inserción de ítems al carrito vacío. 91
Figura 62. Insertar productos al carrito ya con ítems. 92
Figura 63. Interfaz Carrito de compras. 93
Figura 64. Botón actualizar cantidad. 94
Figura 65. Eliminar producto. 95
Figura 66. Muestra de dirección. 96
Figura 67. Dirección almacenada. 96
Figura 68. Formulario de actualizar datos. 97
Figura 69. Realizar compra API paypal. 97
Figura 70. Total, de la compra. 98
Figura 71. Almacenar los detalles de la venta. 98
Figura 72. Pago exitoso. 99
Figura 73. Formulario de inicio de sesión administrador. 101
Figura 74. Obtención de datos y quería de búsqueda. 101
Figura 75. Validación de inicio de sesión. 102
Figura 76. Inicio de panel de administrador. 103
Figura 77. lista de pedidos en panel de administrador. 103
Figura 78. ver detalle de la venta. 104
Figura 79. Interfaz de lista de productos administrador. 104
Figura 80. detalle de producto panel de administrador. 105
Figura 81. Listado de vendedores activos en plataforma panel de administrador. 105
Figura 82. detalles de vendedor panel de administrador. 106
Figura 83. Interfaz de reportes panel de administrador. 106
Figura 84. Detalle de reporte. 107
Figura 85. Notificación de prueba. 108
Figura 86. detalle de notificación, panel de usuario vendedor. 108
Figura 87. Reportes en espera de revisión. 109
Figura 88. Estatus de revisión activo. 109
Figura 89. detalles de producto actualizados. 109
Figura 90. Interfaz de solicitudes para nuevos vendedores. 110
Figura 91. Detalles de solicitud de vendedor. 110
Figura 92. solicitud de nuevos productos, panel de administrador. 111
Figura 93. Detalle de solicitud de producto. 111

Índice De Tablas

Tabla 1. Tabla Registro 45
Tabla 2. Tabla pay_account. 45
Tabla 3. Tabla stars. 46
Tabla 4. Tabla ventas. 46
Tabla 5. Tabla chats. 47
Tabla 6. Tabla detalleventa. 47
Tabla 7. Tabla direcciones. 48
Tabla 8. Tabla products. 49
Tabla 9.Tabla reg_sellers 50
Tabla 10. Tabla notifications. 51
Tabla 11. Tabla sellers_data. 51
Tabla 12. Tabla profile_comments. 52
Tabla 13. Tabla reports. 53
Tabla 14. Tabla admins 54
Tabla 15. Tabla de Backlog 55
Tabla 16. Validación de información 72
Tabla 17. Variables del formulario editar 77
Tabla 18. Validación de datos. 80
Tabla 19. Casos de estatus de envío 82
Tabla 20. Función de desencriptación 90
Tabla 21. Descripción Array_column. 92
Tabla 22. Descripción de in_array. 92
Tabla 23. Actualizar cantidad en carrito de compras. 94
Tabla 24. query para inicio de sesión 102

1. Introducción.

El presente proyecto de tesis consiste en realizar una plataforma colaborativa [1, 2, 3, 4, 5], que por una parte, permita desarrollar un sofisticado comercio electrónico de artesanías mexicanas, con el objetivo de impulsar el desarrollo social, económico y cultural en México, en donde los usuarios podrán vender diferentes tipos de artesanías mexicanas de diversas partes o estados de la república mexicana; describiendo detalles e historia de sus productos, actualizar cada uno de ellos y manipular sus propios productos. Por otra, se pueda conocer todo el arte, cultura y tradición que tiene México, además de hacer compras de los productos que se muestran a través de la plataforma, estableciendo negocios o proyectos con las personas u organizaciones que crean estas artesanías mexicanas, para poder brindar todo tipo de oportunidades a los usuarios. La plataforma se desarrolla de tal manera que sea robusta, segura y usable [6, 7].

La gran variedad de productos artesanales que se podrán encontrar como ejemplo, son:

- Talavera.
- Arte huichol.
- Diferentes artículos de oro, plata y piedras preciosas.
- Calzado.
- Muebles de madera, cerámica y artículos de decoración.
- Sombreros.

En el diagnóstico situacional del sector actual en México [8] durante el periodo de la pandemia la mayoría de las comunidades artesanales del país padecen de múltiples carencias, ya que no cuentan con ingresos sostenibles en su actividad, carecen de prestaciones laborales y de espacios indispensables para comercializar sus productos. Esto agregando que, desde antes de la pandemia, las personas que accedían a la opción de vender sus productos por internet se encontraban con los costos altos de otras plataformas o la falta de información de opciones para vender

sus productos, llegando a que muchas personas y comunidades estaban totalmente dependiendo del turismo.

Se realizaron encuestas a personas que se dedican a realizar y distribuir artesanías, obteniendo que las plataformas existentes carecen de innovación al arte, que se exceden en la comisión y costos que exigen las plataformas.

Respaldando esto con las investigaciones por parte del Instituto Nacional de Estadística, Geografía e Informática (INEGI) [9]. **Las artesanías son el sustento de un 0.78% de las familias mexicanas.**

Es por ello, que se hace necesario una plataforma no solo para la venta y compra de las artesanías, sino principalmente para fomentar las artesanías mexicanas tanto en nuestro país como en el mundo entero. Tomando en consideración las recomendaciones del Fondo Nacional para el Fomento de las Artesanías (FONART) [10]; en la cuales señala la importancia del desarrollo cultural en nuestro país a través de las artesanías mexicanas.

Por todo esto, nace la idea de crear una plataforma colaborativa de apoyo al artesano basada en SCRUM; La plataforma colaborativa se desarrollará con los recursos necesarios para su funcionamiento y que sea una plataforma exclusiva de artesanías mexicanas que contenga contenido y funciones que a los usuarios les sea más conveniente y satisfactorio, así como dar a conocer con mayor facilidad el contenido a usuarios que sean de diversos países. Además, se integra en el proyecto usabilidad para aplicaciones colaborativas, así como seguridad para las interacciones y transacciones que se realizaran; fomentando la fiabilidad, responsividad y robustez de dicha plataforma.

El objetivo general consiste en:

- Desarrollar una plataforma colaborativa de apoyo al artesano basada en SCRUM para implementar una plataforma eficaz, optimizada e impulsar el desarrollo social, económico y cultural en México.

Los objetivos específicos se centran en:

- Realizar una plataforma colaborativa que sea óptima, eficiente, segura, usable, robusta, fiable y responsiva para promocionar y dar a conocer las artesanías mexicanas.
- Elaborar el inicio del proyecto, determinando la visión del proyecto, los participantes, requisitos o backlog priorizado del producto y una planeación parcial de entregas.
- Especificar la planeación y estimación del producto, estableciendo las historias de usuario, tareas y pruebas conceptuales.
- Desarrollar la implementación mediante Sprints, que implican una iteración en la cual se analiza, diseña, codifica y prueba uno o más requisitos o funciones del sistema. Estos sprints se van controlando a través de la planificación de estos y de las reuniones diarias (daily stand ups meetings).
- Realizar la revisión y retrospectiva del proyecto, dando como resultado un backlog priorizado y mejorado del producto.
- Entregar el producto funcional y acorde a los requisitos establecidos anteriormente.
- Utilizar estándares en cada etapa del desarrollo de la plataforma para permitir y simplificar su adaptación y/o extensión.

La tesis está estructurada de la siguiente forma: La sección 2 presenta las aplicaciones similares destacando las funciones comunes, así como los pros y los contras de estas. La sección 3 describe, brevemente, los fundamentos teóricos del trabajo de tesis. La sección 4 explica el desarrollo de una plataforma colaborativa de apoyo al artesano basada en SCRUM. Finalmente, la sección 5 proporciona las conclusiones y el trabajo futuro.

2. Aplicaciones Similares.

Se han desarrollado múltiples plataformas, las cuales algunas personas usan para vender sus productos de artesanías, pero entrevistando a diversas personas y organizaciones que venden artesanías en diferentes estados, han concluido que en las plataformas e-commerce y Marketplace existentes se presentan algunos inconvenientes.

2.1. Mercado libre.

Empresa conocida en toda América latina, la cual genera ingresos en las ventas de artículos varios, entre ellos artesanías, sin embargo, no es una plataforma centrada en su totalidad en artesanías, por lo cual, dichos objetos no son mostrados entre los artículos más buscados, además de tener un alto costo al publicar una venta [11].

2.2. Amazon.

Empresa conocida mundialmente, de igual forma que la anterior no está centrada en su totalidad a la venta de artesanías, por ende, las publicaciones de dichos artículos son perdidas entre los artículos más vendidos y también tiene un alto costo al publicar una venta [12].

Que son plataformas de venta general donde los usuarios se pueden registrar y vender sus productos artesanales. Al igual existen plataformas que realizan personas u organizaciones, que cuentan con sus propias plataformas, en las cuales publican sus propias ventas, teniendo únicamente su propio inventario de productos, sin mencionar que los datos de los creadores de las artesanías no son conocidos, algunos sitios son:

2.3. Artesaliz.

Plataforma web creado por alguna organización, ya que dicha información no es pública, la cual se dedica a la venta de artesanías de diferentes estados de la república mexicana, sin embargo, las desventajas de dicha plataforma son que los datos de los artesanos no son públicos y no mencionan el estado del que proviene la artesanía, esto afecta a la economía del estado del que proviene la artesanía [13].

2.4. La Rancherita.

Sitio web administrado por una sola organización, dedicada a la venta y producción de artesanías, creadas por “ellos” mismos, esto quiere decir que pueden no estar distribuyendo artesanías provenientes de otros estados ayudando a la economía de los diferentes estados. Somos expertos en la fabricación de los estilos más populares con precios más bajos y calidad superior [14, 15]

2.5. Artesanías de México

Sitio web creado para la distribución de artesanías de toda la república mexicana, sitio de solo compra, no se pueden publicar ventas de personas externas a la organización creadora del sitio, impuestos elevados en la compra de artículos [16].

2.6. Exixo.

Sitio web diseñado por un grupo de personas que se dedican a la distribución de artesanías físicas y comestibles, no diseñado para la mayoría de los dispositivos, ya que, por tener diferentes animaciones, estas la vuelven lenta al cargar. Sin mencionar alto precio que imponen al promover una venta, ya que es necesario

mandar una solicitud para publicar una venta, Para vender con nosotros, solamente requerimos de un pago de membresía, más las diferentes comisiones de venta [17, 18].

2.7. Estilo mexicano.

Plataforma creada por una o varias organizaciones, se promueve las artesanías tanto de artesanos como también de arte urbano, donde, se promueve la venta de artesanías de diferentes tipos, para ello se necesita realizar una petición de venta, y así, poder publicarla. Precios demasiado elevados, el uso de los datos personales por parte de la plataforma no está clara, Recopilación de datos personales [19, 20].

2.8. Flor de piña.

Plataforma creada por dos entusiastas, para promover las artesanías de varios estados de la república mexicana, distribución de artículos por parte de personas u organización sin la posibilidad de publicar ventas. Los artículos publicados, como en la mayoría de los casos, no muestran la procedencia del artículo o información del artesano [21].

2.9. Artesanías de México

Artesanías de México [22] es un sitio web de solo ventas de diferentes artesanías, Creada y administrada por un grupo [23], uso excesivo de transiciones (véase la Figura 1) lo cual vuelven lento al sitio

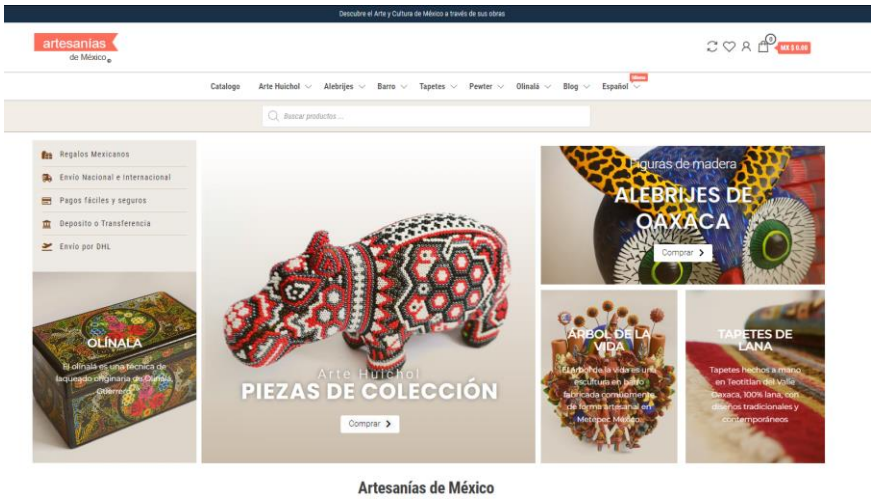


Figura 1. Artesanías de México Página principal.

2.10. Ensamble artesano

Ensamble artesano [24], es una plataforma en la cual colaboran 83 organizaciones, en donde se pueden realizar compras de artículos varios de tipo artesanal, sin embargo, esta plataforma está dedicada a la venta de productos de moda y el hogar (véase la Figura 2), dejando de lado muchas artesanías.

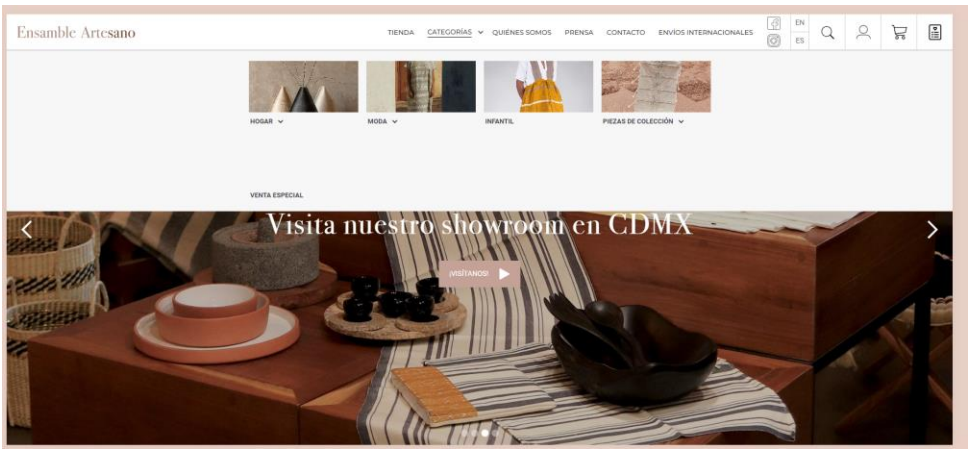


Figura 2. Ensamble Artesano inicio.

2.11. Gaalxim

Gaalxim [26], es una plataforma de venta de textiles de tipo artesanal (véase la Figura 3), dicho sitio pertenece a un solo grupo el cual se dedica a comercializar los productos.

Los productos aquí mostrados son inexactos en su descripción en general y especifica.

Figura 3. Gaalxim Página de inicio.

2.12. Conclusión de aplicaciones similares

Como se ha visto en las diferentes plataformas de venta de artesanías, algunas más visitadas que otras, muy pocas se enfocan al completo en la venta de estos artículos.

Algunos sitios colocan una descripción detalla de los artículos, sin embargo, no dan el crédito al artesano que los crea, además no dejan que él publique sus propias ventas, y, tenga esa flexibilidad

Las investigaciones por parte del Instituto Nacional de Estadística, Geografía e Informática (INEGI) [28], **En 2020, el PIB de las artesanías presentó una caída anual de 15.3 %, en términos reales.**

Por ello se propone la creación de este proyecto colaborativo que se centra en la venta de artesanías de todo tipo, en la cual los artesanos de los diferentes estados de la república mexicana pueden exponer sus obras con la seguridad de que el crédito será en su totalidad para ellos ya sea como vendedor al por mayor o por menor.

Con esto se logra que los artesanos sigan desarrollándose, manteniendo y haciendo crecer sus talleres, como también poder lograr el sustento para sus familias.

3. Marco Teórico.

En esta sección, se describen los fundamentos teóricos que sustentan este trabajo de tesis.

3.1 Herramientas

En los últimos años han aparecido una cantidad de herramientas para poder realizar la venta y compra de artesanías de forma física, sin embargo, con la ayuda de la internet y plataformas de venta en línea, ha sido más cómoda la experiencia para comprar diferentes cosas y que lleguen a la comodidad del hogar.

Para que esta experiencia se logre es necesario realizar estas plataformas con la ayuda del desarrollo web, las cuales ya son herramientas predefinidas dependiendo las necesidades de los diferentes sitios o plataformas.

Las herramientas utilizadas para la creación de esta plataforma web en el aspecto del BackEnd son las siguientes:

- **PHP**. El lenguaje encargado de la lógica de la plataforma, ya que por su versatilidad, robustez y escalabilidad es el lenguaje más popular para el uso en el desarrollo web.
- **MySQL**. Ya que, por el tipo de base de datos que manejamos necesitamos un lenguaje capaz de gestionar bases de datos relacionales, además de que con este gestor podemos almacenar una gran cantidad de información sin problema alguno.

Con la ayuda de estos dos lenguajes, diferentes, pero que se llevan de la mano sin problema alguno, hemos desarrollado nuestra plataforma de ventas sin ningún problema por su versatilidad y gran compatibilidad con una gran gama de dispositivos.

De igual modo son necesarios otras dos herramientas más, que se describen un poco más adelante.

1. **PHP.** Es un lenguaje de programación ampliamente utilizado en desarrollo web como ya antes hemos mencionado, sus principales características como lenguaje más utilizado es el ser de código abierto como también ser gratuito. Es utilizado para crear páginas web dinámicas, interactivas y personalizadas, permite interactuar con bases de datos y otros servicios. Es popular en el desarrollo web porque su sintaxis es sencilla, además, que por su amplia comunidad de desarrolladores y recursos en línea. Es un lenguaje muy versátil, ya que, puede ser ejecutado en una amplia gama de Sistemas operativos como: Windows, Linux como también en su amplia gama de distros que existen, Unix y Mac OS [33]. En definitiva, PHP es uno de los lenguajes más utilizados actualmente en el desarrollo de aplicaciones web y viene experimentado un constante crecimiento en su nivel de utilización en Internet.
2. **MySQL.** Es un sistema de gestión de bases de datos relacionales, el cual es ampliamente utilizado en la industria, es conocido y muy usado por su eficiencia y capacidad para gestionar grandes cantidades de datos de manera organizada, es un gestor de tipo gratuito el cual cuenta con una amplia gama de compatibilidad de dispositivos y usuarios (Logo véase la Figura 4). Se basa en el lenguaje SQL (Structured Query Language) para administrar y manipular datos, lo que facilita las consultas y operaciones. Ofrece una alta velocidad de acceso a datos y un rendimiento optimizado, lo que lo hace adecuado para aplicaciones que requieren una recuperación de datos rápida y confiable. [34]

Figura 4. MySQL

3. **Visual Studio Code.** Por su fácil instalación y uso VS Code será utilizado también por su entorno amigable, alta compatibilidad con los sistemas operativos, la amplia lista de lenguajes que puede editar y la variedad de plugins que se pueden utilizar externamente (logo véase la Figura 5).

Figura 5. Visual Studio Code

4. **GitHub y Git bash.** Herramientas para alojar y gestionar proyectos de desarrollo de software llámese de escritorio, web o móvil, con la ayuda de un bash o consola con ayuda de comandos, sin embargo, con su ayuda de su control de versiones la cual nos permite rastrear y administrar cambios en el código fuente que se esté trabajando en el momento. Proporciona una forma colaborativa y eficiente de trabajar en proyectos de código abierto o privados. Por su amplia compatibilidad con los diferentes Sistemas Operativos, es la herramienta más usada por la mayoría de los desarrolladores, y en este caso no es la excepción, ya que, para lograr una buena administración de versiones y el poder trabajar a distancia esta plataforma fue la numero 1 (logo véase la Figura 6).

Figura 6. Git Bash y GitHub

5. **Xampp.** Herramienta de desarrollo que permite probar páginas web (logo véase la Figura 7).

Figura 7. xampp

6. **API de Paypal.** PayPal API (Interfaz de programación de aplicaciones) [35] que nos provee un botón donde realizar una petición a la API y muestre una interfaz gráfica, para poder realizar pagos en línea por medio de PayPal. Se ingresa y realiza un proceso para poder integrar y consumir correctamente esta API (logo véase la Figura 8).

Figura 8. Paypal

3.2 Metodologías ágiles

Las metodologías ágiles son aquellas que permiten adaptar la forma de trabajo a las condiciones del proyecto, pretenden ofrecer el producto adecuado con una entrega incremental y frecuente de pequeños trozos de funcionalidad, por medio de pequeños equipos muy organizados.

Estas metodologías pretenden cambiar la tradicional forma de trabajo “cascada”, que consiste en la entrega de grandes entregables en largos periodos de tiempo, en los cuales pueden existir cambios y esto al final es un trabajo erróneo.

1. **SCRUM.** La metodología Scrum es una forma de trabajo sencillo para emplear en proyectos complejos, esta fue creada por Ken Schwaber y Jeff Sutherland [29]. Los proyectos se dividen en ciclos (tareas de no más de 3 semanas) mejor llamados Sprints. El cual representa un lapso, del cual, se debe desarrollar un conjunto de características, múltiples sprints

se pueden unir para formar una versión del trabajo. Al finalizar el proyecto se realiza una entrega al cliente u empresa. Las reuniones son el pilar de esta metodología, las cuales son [30]:

- Reunión de planificación: el equipo define los elementos de trabajo que se abordarán durante el sprint y se establece un objetivo claro.
- Reunión diaria: breves reuniones diarias en las que el equipo comparte actualizaciones sobre el progreso, los obstáculos y los planes para el día.
- Reunión de revisión: al final de cada sprint, se lleva a cabo una reunión en la que se muestra al cliente o al Product Owner el trabajo completado y se recopilan comentarios.
- Reunión de retrospectiva: también al final de cada sprint, el equipo reflexiona sobre el proceso y busca oportunidades de mejora para el próximo sprint.

2. **Desarrollo de software adaptable (ASD).** El **desarrollo** de software adaptable (ASD), es una forma de trabajo que se centra en la adaptabilidad y la capacidad de respuesta a medida que se desarrolla un proyecto [Figura 9], la cual se enfoca en la **entrega** continua de software funcional y de calidad [31]. Esta metodología ágil busca abordar los desafíos cambiantes del desarrollo de software al enfocarse en la adaptabilidad, la colaboración y la entrega incremental. Al permitir una mayor flexibilidad y capacidad de respuesta, ASD busca mejorar la satisfacción del cliente y la calidad del software entregado. Los principios fundamentales del Desarrollo de Software Adaptable son los siguientes:
    - Adaptabilidad: ASD reconoce que los requisitos y las necesidades del cliente pueden cambiar a lo largo del tiempo y busca ser flexible y adaptable para responder a estos cambios.
    - Comunicación: la comunicación efectiva entre los miembros del equipo de desarrollo y los clientes es esencial en ASD.

- Entrega incremental: ASD se basa en la entrega incremental de funcionalidades y características del software. Se desarrollan y entregan incrementos de software en intervalos regulares y cortos, lo que permite obtener respuesta temprana del cliente y facilita la adaptación a los cambios.
- Calidad: Se centra en la calidad del software en ASD. Se lleva a cabo una planificación y una gestión cuidadosa para garantizar que se cumplan los estándares de calidad y que el software entregado sea funcional y confiable.
- Colaboración y autoorganización: ASD promueve equipos de desarrollo autoorganizados y multidisciplinarios. Se espera que los miembros del equipo trabajen juntos y colaboren en la toma de decisiones y en la solución de problemas.

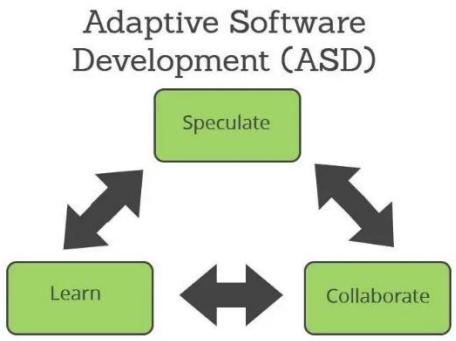


Figura 9. Diagrama de ASD

3. **Kanban.** Kanban, es una forma de trabajo utilizado en la gestión de proyectos y el desarrollo de software que se centra en la visualización y el flujo de trabajo. Kanban se basa en principios de transparencia, adaptabilidad y mejora continua para lograr una entrega eficiente de proyectos y productos [32]. Se utiliza un tablero visual [Figura 10] para representar el flujo de trabajo del proyecto o del equipo de desarrollo. El tablero se divide en columnas que representan las etapas del proceso, y las tareas o elementos de trabajo se representan como tarjetas o notas

adhesivas que se mueven a lo largo del tablero a medida que avanzan en el proceso. Los principios clave de Kanban son los siguientes:

- Visualización del flujo de trabajo: se basa en la visualización clara y transparente del flujo de trabajo. El tablero Kanban permite a todos los miembros del equipo ver el estado actual de las tareas y comprender cómo fluyen a través del proceso.
- Límites de trabajo en progreso (WIP): establece límites claros sobre la cantidad de tareas o elementos de trabajo que se pueden abordar simultáneamente en cada etapa del proceso.
- Enfoque en el flujo continuo: se busca lograr un flujo de trabajo continuo y constante, evitando interrupciones y esperas innecesarias.
- Mejora continua: fomenta la mejora continua a través de la revisión y el análisis regular del proceso y el flujo de trabajo.

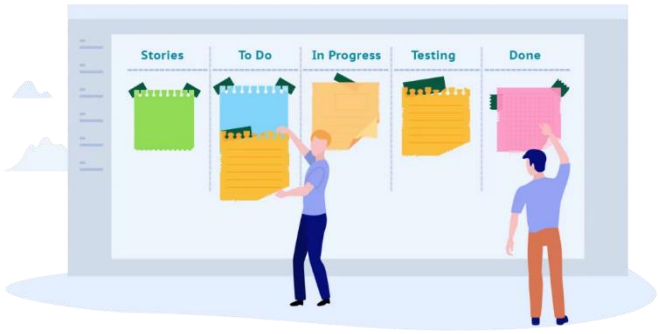


Figura 10. Kanban ejemplo de tablero

4. Desarrollo.

Para este proyecto utilizaremos la metodología ágil SCRUM con la ayuda de sus Sprints, ya que este se centra en proyectos complejos.

Hemos elegido esta metodología, ya que, los siguientes puntos de la metodología se centran en las necesidades de este proyecto:

- Productividad.
- Lapsos cortos de tiempo para entregas.
- Requerimientos cambiantes.
- Flexibilidad.

Estos puntos se lograrán gracias a las diferentes reuniones que podremos realizar con esta metodología:

- Reunión de planificación: desde el principio el equipo de trabajo estará enterado de las tareas a corto y largo plazo, como también de la meta a alcanzar.
- Reunión diaria: el trabajo en colaborativo se irá compartiendo, con esto se logrará una mejor interacción de trabajo y adaptación para el equipo.
- Reunión de revisión: al finalizar cada sprint o tarea el equipo podrá hacer revisión completa.

También se han utilizado todas las herramientas antes descritas en la parte del marco teórico, para lograr el desarrollo total de esta plataforma colaborativa de ventas artesanales. Primero presentaremos las historias de usuario, luego los diagramas de casos de uso, posteriormente, el modelo entidad-relación de la base de datos.

4.1 Historias de usuario cliente y vendedor

En este apartado se habla de los 3 tipos de usuarios, en primera instancia hablaremos del usuario normal el cual puede ver y comprar productos.

Como también tenemos al usuario de tipo vendedor, el cual, con algunos procedimientos adicionales puede realizar un perfil de vendedor con los datos de su empresa o negocio, como también, poder realizar ventas.

Y, por último, pero no menos importante, tenemos al usuario administrador, en este momento solo lo mencionaremos, sin embargo, en el siguiente apartado se describirá más a detalle a este tipo de usuario único.

De primera instancia describiremos cada una de las historias de usuario, más adelante en el siguiente apartado mostraremos los diagramas de casos de uso.

4.1.1 Diagrama de casos de uso usuario vendedor y comprador

El diagrama de casos de uso que se muestra en la Figura 11, lo estaremos hablando detalladamente a continuación.

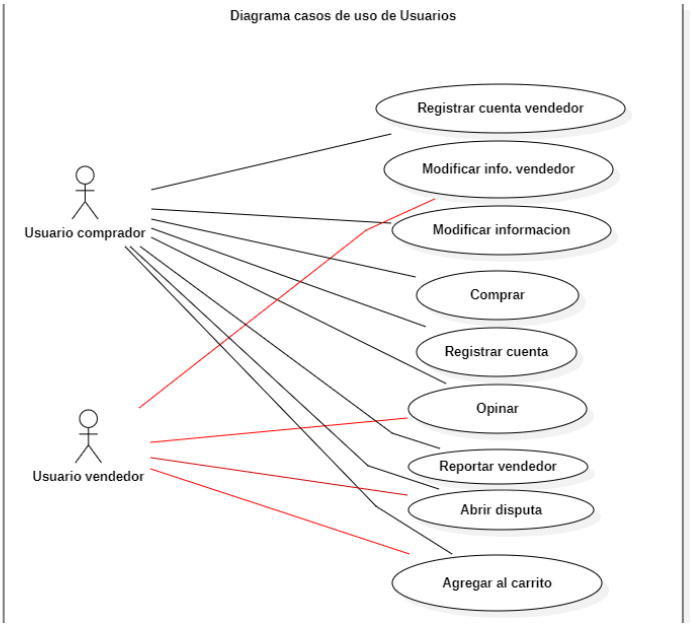


Figura 11. Diagrama casos de uso usuarios

- **Modificar información de vendedor.** El usuario vendedor tendrá la posibilidad de cambiar sus datos de tienda, dirección, razón social, cuenta bancaria, métodos de contacto, etc.

  - **Flujo básico:**
    1. El usuario vendedor da clic en la opción configuración de cuenta/Datos de “tienda”.
    2. El usuario ingresará sus datos y dará clic en guardar.
  - **Flujo alterno:**

    A1. Si el usuario deja campos vacíos que sean obligatorios el sistema no habilitará la opción guardar

    A2. Si el usuario olvida dar en guardar el sistema informara sobre si se quieren guardar los cambios
  - **Escenarios:**

  – **Escenario 1**
    - El usuario vendedor da clic en la opción configuración de cuenta/Datos de “tienda”.
    - El sistema desplegará un formulario donde el usuario deberá ingresar su información.
    - El usuario llena todos los campos con la información requerida y da en el botón guardar.
    - El sistema guarda los ajustes y muestra un letrero de proceso exitoso

  – **Escenario 2**
    - El usuario vendedor da clic en la opción configuración de cuenta/Datos de “tienda”.
    - El sistema desplegará un formulario donde el usuario deberá ingresar su información.
    - El usuario olvida llenar todos los campos requeridos.
    - El sistema no habilita la opción guardar, hasta que los campos sean completamente llenados.

  – **Escenario 3**

- El usuario vendedor da clic en la opción configuración de cuenta/Datos de “tienda”.
- El sistema desplegará un formulario donde el usuario deberá ingresar su información.
- El usuario llena todos los campos con la información requerida, olvida guardar y se mueve a otra opción del menú.
- El sistema alerta de que los cambios no se han sido guardados y se queda en la misma interfaz.

- **Clase entidad**: Usuario Vendedor.
- **Clases límite**: La pantalla modificar información, muestra los campos respectivos a datos de tienda, dirección, razón social, cuenta bancaria, teléfono, etc. y muestra el botón guardar.
- **Clases control**: Se creará la clase: ModificarInfVendedor.

- **Modificar información de comprador.** El usuario comprador tendrá la posibilidad de cambiar sus datos de usuario, como nombre, domicilio, datos de contacto, información de pago, etc.

  - **Flujo básico:**
    1. El usuario da clic en la opción configuración de cuenta/Datos de cuenta.
    2. El usuario ingresará sus datos y dará clic en guardar.
  - **Flujo alterno:**

    A1. Si el usuario deja campos vacíos que sean obligatorios el sistema no habilitará la opción guardar

    A2. Si el usuario olvida dar en guardar el sistema informara sobre si se quieren guardar los cambios
  - **Escenarios:**
  - **Escenario 1:**
    - El usuario da clic en la opción configuración de cuenta/Datos de cuenta.

- El sistema desplegará un formulario donde el usuario deberá ingresar su información.
- El usuario llena todos los campos con la información requerida y da en el botón guardar.
- El sistema guarda los ajustes y muestra un letrero de proceso exitoso.

– **Escenario 2:**
- El usuario da clic en la opción configuración de cuenta/Datos de cuenta.
- El sistema desplegará un formulario donde el usuario deberá ingresar su información.
- El usuario olvida llenar todos los campos requeridos.
- El sistema no habilita la opción guardar, hasta que los campos sean completamente llenados.

- **Escenario 3:**
- El usuario vendedor da clic en la opción configuración de cuenta/Datos de cuenta.
- El sistema desplegará un formulario donde el usuario deberá ingresar su información.
- El usuario llena todos los campos con la información requerida, olvida guardar y se mueve a otra opción del menú.
- El sistema alerta de que los cambios no se han sido guardados y se queda en la misma interfaz

▪ **Clase entidad:** Usuario comprador.
▪ **Clase límite**: La pantalla modificar información, muestra los campos respectivos a sus datos de usuario, como nombre, domicilio, datos de contacto, información de pago, etc.
▪ **Clases control:** Se creará la clase ModificarInfUsuario.

▪ **Agregar al carrito.** El usuario tendrá la opción de ver sus artículos en la tienda e ir agregándolos uno a uno al “carrito”, para que dé un solo movimiento compre varios artículos.

- **Flujo básico:**
  1. El usuario agrega artículos al “carrito”, seleccionando un artículo de la tienda y dando clic en el botón agregar al carrito.
  2. El usuario da clic en el icono de carrito para poder ver sus artículos.
  3. El usuario da clic en el botón comprar, para proceder a la compra.
- **Flujo alterno:**

  A1: El usuario agrega artículos al “carrito”, seleccionando un artículo de la tienda y dando clic en el botón agregar al carrito.

  A2: El usuario agrega artículos de más que ya no quiere comprar.

  A3: El usuario no procede a la compra.
- **Escenarios:**

– **Escenario 1**

- El usuario agrega artículos al “carrito”, seleccionando un artículo de la tienda y dando clic en el botón agregar al carrito.
- El usuario da clic en el icono de carrito para poder ver sus artículos.
- El usuario ya no quiere artículos de su carrito.
- El sistema habilita la opción de quitar artículos del carrito.

– **Escenario 2**

- El usuario agrega artículos al “carrito”, seleccionando un artículo de la tienda y dando clic en el botón agregar al carrito.
- El usuario da clic en el icono de carrito para poder ver sus artículos.
- El usuario solo ve los artículos, pero no procede a la compra
- El sistema guardara y mostrará la lista del carrito, hasta que el usuario elimine el carrito o los artículos queden sin existencia.

- **Clase entidad:** Usuario comprador.
- **Clases límite:** el sistema mostrará la interfaz de carrito con la lista de los artículos ingresados, dando la posibilidad de quitarlos, y mostrará el botón de comprar.
- **Clases control:** Se creará la clase AgregarCarrito.

- **Comprar.** El usuario tendrá la opción de proceder a la compra de sus artículos ingresados al carrito.
  - **Flujo básico:**
    1. El usuario agrega artículos al “carrito”, seleccionando un artículo de la tienda y dando clic en el botón agregar al carrito.
    2. El usuario procede a la compra dando clic en el botón comprar.
    3. El sistema muestra la cotización de los artículos.
    4. El usuario autoriza el pago.
  - **Flujo alterno:**

    A1: El usuario no ha agregado datos de dirección.

    A2: El usuario no ha agregado datos de pago.

    A3: El usuario no autoriza el pago.

    A4: El sistema notifica errores con el método de pago
  - **Escenarios:**
  - **Escenario 1**
    - El usuario agrega artículos al “carrito”, seleccionando un artículo de la tienda y dando clic en el botón agregar al carrito.
    - El usuario procede a la compra dando clic en el botón comprar.
    - El sistema no detecta datos de dirección.
    - El sistema muestra la opción de agregar una dirección.
  - **Escenario 2:**
    - El usuario agrega artículos al “carrito”, seleccionando un artículo de la tienda y dando clic en el botón agregar al carrito.
    - El usuario procede a la compra dando clic en el botón comprar.
    - El sistema no detecta datos de método de pago.
    - El sistema muestra la opción de agregar un método de pago.
  - **Escenario 3:**
    - El usuario agrega artículos al “carrito”, seleccionando un artículo de la tienda y dando clic en el botón agregar al carrito.
    - El usuario procede a la compra dando clic en el botón comprar.
    - El usuario solo ve la cotización de los artículos y no procede al pago

- El usuario no genera ningún tipo de pedido

- **Clase entidad:** Usuario comprador.

- **Clase límite**: el sistema mostrará la interfaz de agregar o seleccionar: dirección y método de pago, el usuario debe autorizar el pago.
- **Clases control:** Se creará la clase Comprar.

- **Registrar cuenta.** Los usuarios pueden registrarse y tienen la posibilidad de registrarse como un usuario vendedor
  - **Flujo básico:**
    1. El usuario da clic en registrarse.
    2. El sistema muestra el formulario de registro
    3. El usuario ingresa los datos correspondientes.
    4. El usuario da clic en el botón registrar
  - **Flujo alterno:**

    A1: el usuario deja campos obligatorios sin llenar.

    A2: la verificación de contraseña no coincide.
  - **Escenarios:**
  - **Escenario 1:**
    - El usuario da clic en registrarse.
    - El usuario ingresa los datos correspondientes, olvidando llenar un dato obligatorio.
    - El sistema no habilitará el botón registrar, ya que, faltan campos por rellenar y manda una alerta.
  - **Escenario 2:**
    - El usuario da clic en registrarse.
    - El usuario ingresa los datos correspondientes, ingresando su contraseña correctamente, pero la verificación de contraseña es incorrecta.
    - El sistema mandará una alerta de que las contraseñas no coinciden.
  - **Clase entidad:** Registrar
  - **Clase límite**: El sistema mostrará la interfaz de registro además del botón “registrar”.

- **Clases control:** Se creará la clase: Registrar.

- **Opinar.** El usuario tendrá la posibilidad de dejar su opinión respecto al vendedor y el artículo comprado, si el artículo se verificó como entregado
  - **Flujo básico:**
    1. El usuario da clic en el botón opinar.
    2. El sistema muestra la interfaz de opiniones.
    3. El usuario deja su opinión y su boto al artículo o tienda.
  - **Flujo alterno:**

    A1: El usuario solo deja su boto, pero no opina.

    A2: El usuario no opina.

    A3: El artículo no se ha verificado y no habilita la opción.
  - **Escenarios:**

  – **Escenario 1:**
    - El usuario no ha verificado su entrega
    - El sistema no autoriza el boto, por razones de seguridad.
  - **Clase entidad:** Opinar.
  - **Clase límite:** El sistema muestra la interfaz de opinar, además del botón opinar para guardar los datos.
  - **Clases control:** Se creará la clase Opinar.
- **Reportar vendedor.** El usuario tiene la opción de reportar al vendedor, por un mal trato, o por cualquier razón con cuestión a la venta.
  - **Flujo básico:**
    1. El usuario da clic en el botón “Reportar”
    2. El usuario llena el formulario y da clic en el botón “Enviar reporte”
  - **Flujo alterno:**

    A1: El usuario envía un reporte sin razón alguna.

    A2: El usuario inicia el proceso del reporte sin culminar.
  - **Escenarios:**
  – **Escenario 1:**
    - El usuario inicia el proceso de reporte.

        - El administrador verifica el reporte, sin encontrar aparente razón por la cual se pueda proceder.
    - **Escenario 2:**
        - El usuario inicia el proceso de reporte.
        - El usuario llena el formulario de reporte, pasando a otra interfaz sin enviar el reporte.
    - El sistema muestra una alerta de que el reporte no se ha enviado.
    - **Clase entidad:** Reportar vendedor.
    - **Clase límite**: El sistema habilita el botón “reportar vendedor”, muestra el formulario de reporte y el botón “reportar”.
    - **Clases control:** Se creará la clase: Reportar.
- **Abrir disputa.** Al momento de verificar la entrega del envío, el sistema habilitará el botón “abrir disputa”, en el cual mostrar un formulario y se describirá las posibles opciones del problema, Articulo equivocado, Articulo dañado, Cambiar artículo, Regresar artículo, etc.
    - **Flujo básico:**
        1. El usuario da clic en el botón “Abrir disputa”.
        2. El sistema muestra el formulario y el usuario lo llena.
        3. El usuario envía la disputa, dando clic en el botón “Enviar disputa”.
    - **Flujo alterno:**

      A1: El sistema no habilita la opción abrir disputa.

      A2: El usuario inicia el proceso, pero no lo culmina.

      A3: El usuario no verifica la entrega de su paquete.
    - **Escenarios:**
    - **Escenario 1**
        - El usuario inicia el proceso de disputa.
        - El usuario llena el formulario, pasando a otra interfaz sin enviar la disputa.
        - El sistema muestra una alerta de que la disputa no se ha enviado.
    - **Escenario 2**
        - El usuario no verifica la entrega de su paquete.

- El sistema esperará 15 días para verificar la entrega automáticamente.
- El usuario no podrá iniciar una disputa hasta verificar su entrega.

- **Clase entidad** Abrir disputa.
- **Clase límite**: El sistema habilitará el botón “Abrir disputa”, mostrando el formulario y el botón “enviar disputa”.
- **Clases control:** Se creará la clase Disputa.

- **Comentario en producto o perfil.** Al momento de querer dejar un comentario ya sea en algún producto o algún perfil de un vendedor, el sistema mostrará un formulario para poder llenar en el cual se tendrá habilitado la parte de agregar el comentario y ajustar un contador de estrellas, en los cuales el usuario puede dejar su comentario y calificación hacia el producto o el usuario vendedor.
  - **Flujo básico:**
    1. El usuario da clic en algún producto o perfil de usuario vendedor.
    2. El sistema muestra el formulario en la parte final de la página y el usuario lo llena.
    3. El usuario envía el comentario y calificación dando clic en el botón comentar
  - **Flujo alterno:**

    A1: El sistema no habilita la opción comentar.

    A2: El usuario inicia el proceso, pero no lo culmina.

    A3: El usuario llena la parte del comentario, pero no de la calificación de estrellas.
  - **Escenarios:**
  - **Escenario 2:**
    - El usuario inicia el proceso de comentario.
    - El usuario llena el formulario, pasando a otra interfaz sin enviar el comentario.
    - El sistema vaciara (borrara) los campos sin almacenar ninguna información.

- **Escenario 3:**
  - El usuario llena la parte del comentario, pero olvida ingresar calificación de estrellas.
  - El sistema ingresa la calificación como 0.
  - El usuario puede ocupar la plataforma sin ningún problema, sin embargo, esto afectaría al usuario vendedor.
- **Clase entidad:** comentario.
- **Clase límite**: El sistema habilitará los campos “comentario” y “stars”, para poder guardar la información.
- **Clases control:** Se creará la clase Comment.

- **Reportar producto.** Al momento de querer realizar un reporte hacia un producto ya sea por información engañosa, categoría errónea, información fuera de contexto a las normas establecidas. El usuario podrá realizar un reporte hacia dicho producto, dicho reporte será puesto como solicitud pendiente hasta que los administradores revisen dicha solicitud. Si la solicitud es aprobada, el producto pasara a un estado de reportado y dejara de mostrarse en el catálogo de productos, y se enviara la notificación al vendedor del producto para poder modificar la información de su producto o en caso contrario dar caso omiso. Si la solicitud es negativa, dicha solicitud solo será eliminada y el producto seguirá en catálogo sin ningún cambio.
  - **Flujo básico:**
    1. El usuario da clic en algún producto.
    2. El sistema muestra un botón de alerta, el cual mostrará un formulario que puede llenar el usuario.
    3. El usuario llena el formulario y lo envía, el sistema mostrará dicho reporte en el panel del administrador para que él pueda revisarlo.
  - **Flujo alterno:**

    A1: El sistema no habilita la opción reportar.

    A2: El usuario inicia el proceso, pero no lo culmina.

A3: El usuario quiere reportar el producto, pero el sistema no habilita la opción.

- **Escenarios:**

– **Escenario 2**

- El usuario inicia el proceso de reporte.
- El usuario llena el formulario, pasando a otra interfaz sin enviar el reporte.
- El sistema vaciara (borrara) los campos sin almacenar ninguna información.

– **Escenario 3**

- El usuario llena el formulario del comentario, pero ya había mandado un reporte anteriormente.
- El sistema bloqueará la opción de reporte hasta que el administrador de seguimiento al reporte anterior.
- El usuario puede ocupar la plataforma sin ningún problema.

- **Clase entidad** Reporte.
- **Clase límite**: El sistema habilitará el botón “Reporte” para poder mostrar el formulario de reporte.
- **Clases control:** Se creará la clase “Report”.

- **Reportar comentario.** Al momento de querer realizar un reporte hacia un comentario ya sea por palabras altisonantes, comentario fuera de contexto o simple hate, el usuario vendedor podrá hacer un reporte hacia dicho comentario. Dicho reporte será enviado al panel del administrador para que a si el administrador pueda revisar dicha solicitud. Si la solicitud es aprobada, dicho comentario será eliminado y el usuario que hizo el comentario será notificado de dicho reporte. Si la solicitud es rechazada, dicha solicitud será eliminada, y el comentario seguirá en el panel de comentarios de dicho producto o perfil.
  - **Flujo básico:**
    1. El usuario da clic en algún producto.

2. El sistema muestra un botón de alerta en el panel de cada comentario, el cual mostrará un formulario que puede llenar el usuario que desea realizar el reporte.
3. El usuario llena el formulario y lo envía, el sistema mostrará dicho reporte en el panel del administrador para que él pueda revisarlo.

- **Flujo alterno:**

A1: El sistema no habilita la opción reportar.

A2: El usuario inicia el proceso, pero no lo culmina.

A3: El usuario quiere reportar el comentario, pero el sistema no habilita la opción.

- **Escenarios:**

– **Escenario 1:**

  - El usuario inicia el proceso de reporte.
  - El usuario llena el formulario, pasando a otra interfaz sin enviar el reporte.
  - El sistema vaciara (borrara) los campos sin almacenar ninguna información.

– **Escenario 1:**

  - El usuario llena el formulario del comentario, pero ya había mandado un reporte anteriormente.
  - El sistema bloqueará la opción de reporte hasta que el administrador de seguimiento al reporte anterior.
  - El usuario puede ocupar la plataforma sin ningún problema.

- **Clase entidad** Reporte.
- **Clase límite**: El sistema habilitará el botón “Reporte” para poder mostrar el formulario de reporte.
- **Clases control:** Se creará la clase “Reportcomment”.

4.2 Historias de usuario Administrador

El usuario de tipo administrador es el más importante, ya que, como el nombre lo menciona es el tipo de usuario que administra la plataforma, en este caso, es el que da flujo a todas las peticiones de usuarios de tipo normal como de tipo vendedor.

En los siguientes apartados se describen los casos de uso del administrador.

4.2.1 Diagrama de casos de uso administrador

El diagrama de casos de uso Figura 12 se explicará detalladamente en los siguientes apartados.

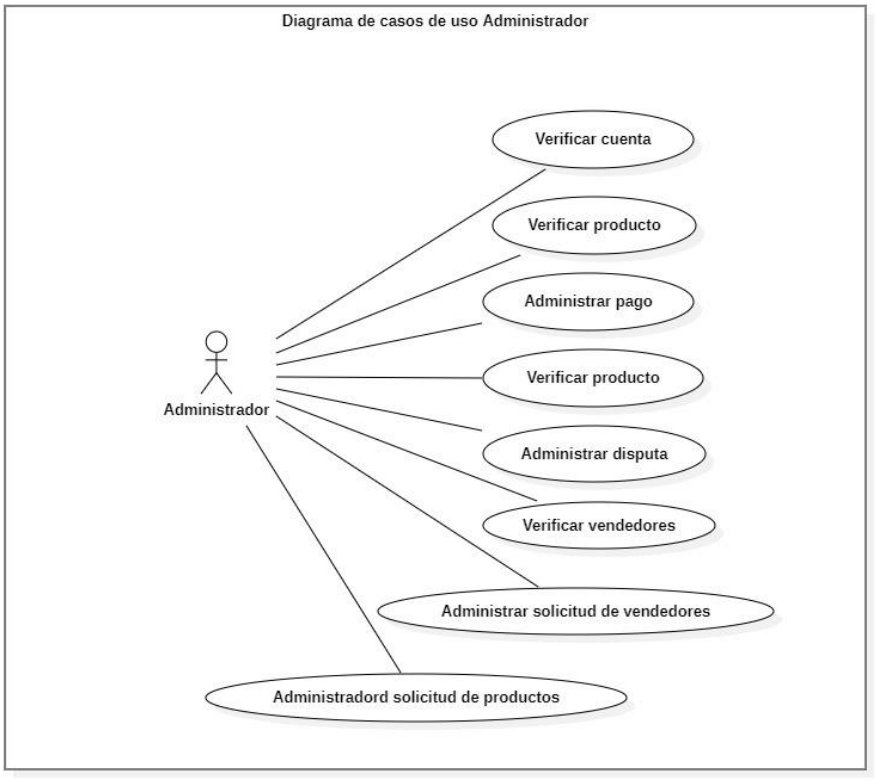


Figura 12. Diagrama casos de uso Administrador.

- **Verificar cuenta.** Permite al administrador verificar solicitudes de cuentas registradas por los usuarios como vendedores.
  - **Flujo básico**:
    1. El sistema mostrará un menú con la opción verificar cuentas.
    2. El sistema mostrará una lista con las solicitudes, facilitando un botón para ver los detalles de la solicitud.
    3. El administrador podrá aceptar o rechazar la solicitud.
  - **Flujo alterno:**

    A1. El administrador solo ve las solicitudes sin realizar ningún cambio.

    A2. El administrador ve la solicitud, pero cambia de lugar en el menú sin enviar cambios.
  - **Escenarios:**
  - **Escenario 1:**
    - El administrador da clic en menú/verificar cuentas.
    - El sistema muestra el listado de las solicitudes pendientes por revisar.
    - El administrador da clic para ver los detalles de dicha solicitud.
    - El administrador sale de la solicitud para revisarla después.
    - El sistema cambia el estado de la solicitud a “visto” y lo mueve de lugar en la lista.
  - **Escenario 2:**
    - El administrador da clic en menú/verificar cuentas.
    - El sistema muestra el listado de las solicitudes pendientes por revisar.
    - El administrador da clic para ver los detalles de dicha solicitud.
    - El administrador cambia de “interfaz” sin realizar un cambio.
    - El sistema manda una alerta de que no se ha realizado cambios.
  - **Clase entidad**: Administrador.
  - **Clases de control**: Se creará la clase: Verificarcuentas.
- **Verificar producto.** Permite al administrador verificar el producto que haya subido el usuario vendedor.

- **Flujo básico:**
    1. El sistema mostrará un menú con la opción verificar productos.
    2. El sistema mostrará una lista con las solicitudes, facilitando un botón para ver los detalles de la solicitud.
    3. El administrador podrá aceptar o rechazar la solicitud.
- **Flujo alterno:**

    A1. El administrador solo ve las solicitudes sin realizar ningún cambio.

    A2. El administrador ve la solicitud, pero cambia de lugar en el menú sin enviar cambios.
- **Escenarios:**

– **Escenario 1:**

- El administrador da clic en menú/verificar productos.
- El sistema muestra el listado de las solicitudes pendientes por revisar.
- El administrador da clic para ver los detalles de dicha solicitud.
- El administrador sale de la solicitud para revisarla después.
- El sistema cambia el estado de la solicitud a “visto” y lo mueve de lugar en la lista.

– **Escenario 2:**

- El administrador da clic en menú/verificar productos.
- El sistema muestra el listado de las solicitudes pendientes por revisar.
- El administrador da clic para ver los detalles de dicha solicitud.
- El administrador cambia de “interfaz” sin realizar un cambio.
- El sistema manda una alerta de que no se ha realizado cambios.

- **Clase entidad:** Administrador.
- **Clase de control:** Se creará la clase: Verificarproductos.

- **Administrar pagos.** El administrador podrá verificar los pedidos y envíos, para posteriormente autorizar los pagos de cada compra.
    - **Flujo básico:**

1. El sistema mostrará un menú con la opción de administrar pagos.
2. El administrador da clic en dicha opción y el sistema muestra un listado con los pagos disponibles para verificar.
3. El administrador puede ver los detalles dando clic en el botón “ver más”
4. El administrador puede autorizar los pagos o rechazarlos.

- **Flujo alterno:**

  A1. El administrador solo ve las solicitudes sin realizar ningún cambio.

  A2. El administrador ve la solicitud, pero cambia de lugar en el menú sin enviar cambios.

- **Escenarios:**

– **Escenario 1:**
  - El administrador da clic en menú/administrar pagos.
  - El sistema muestra el listado de las solicitudes pendientes por revisar.
  - El administrador da clic para ver los detalles de dicha solicitud.
  - El administrador sale de la solicitud para revisarla después.
  - El sistema cambia el estado de la solicitud a “visto” y lo mueve de lugar en la lista.

– **Escenario 2:**
  - El administrador da clic en menú/ administrar pagos.
  - El sistema muestra el listado de las solicitudes pendientes por revisar.
  - El administrador da clic para ver los detalles de dicha solicitud.
  - El administrador cambia de “interfaz” sin realizar un cambio.
  - El sistema manda una alerta de que no se ha realizado cambios.

- **Clase entidad:** Administrador.
- **Clase de control:** Se creará la clase: Administrarpagos.

- **Administrar disputa.** El administrador podrá revisar las solicitudes de disputas, que hayan enviado los usuarios por: fallas en producto, producto

no deseado, no es lo que esperaba, reportes a usuarios de diferentes tipos, etc.

- **Flujo básico:**
  1. El sistema mostrará un menú con la opción de administrar disputas.
  2. El administrador da clic en dicha opción y el sistema muestra un listado con las disputas para verificar.
  3. El administrador puede ver los detalles dando clic en el botón “ver más”
  4. El administrador puede autorizar o rechazar.
- **Flujo alterno:**

  A1. El administrador solo ve las solicitudes sin realizar ningún cambio.

  A2. El administrador ve la solicitud, pero cambia de lugar en el menú sin enviar cambios.
- **Escenarios:**

– **Escenario 1:**

- El administrador da clic en menú/administrar disputas.
- El sistema muestra el listado de las solicitudes pendientes por revisar.
- El administrador da clic para ver los detalles de dicha solicitud.
- El administrador sale de la solicitud para revisarla después.
- El sistema cambia el estado de la solicitud a “visto” y lo mueve de lugar en la lista.

– **Escenario 2:**

- El administrador da clic en menú/administrar disputas.
- El sistema muestra el listado de las solicitudes pendientes por revisar.
- El administrador da clic para ver los detalles de dicha solicitud.
- El administrador cambia de “interfaz” sin realizar un cambio.
- El sistema manda una alerta de que no se ha realizado cambios.

- **Clase entidad:** Administrador.

- **Clase de control:** Se creará la clase: Administrardisputas.

- **Verificar vendedores.** El administrador podrá revisar y administrar a los usuarios de tipo vendedor.
  - **Flujo básico:**
    1. El sistema mostrará un menú con la opción “vendedores”.
    2. El administrador da clic en el botón “vendedores” donde el sistema mostrará un listado con los nombres de cada vendedor.
    3. El administrador puede ver los detalles dando clic en el botón “Ver info”
    4. El administrador puede visualizar la información del vendedor o eliminarlo de la lista.
  - **Flujo alterno:**

    A1. El administrador solo ve las solicitudes sin realizar ningún cambio.

    A2. El administrador ve la solicitud, pero cambia de lugar en el menú sin enviar cambios.

  - **Escenarios:**
  - **Escenario 1:**
    - El administrador da clic en menú/vendedores.
    - El sistema muestra el listado de los vendedores actuales.
    - El administrador da clic para ver los detalles de un vendedor.
    - El administrador visualiza la información sin realizar ninguna acción.
    - El sistema no realiza ningún cambio a dicha solicitud.
  - **Escenario 2:**
    - El administrador da clic en menú/vendedores.
    - El sistema muestra el listado de los vendedores actuales.
    - El administrador da clic para ver los detalles de un vendedor.
    - El administrador cambia de “interfaz” sin realizar un cambio.
    - El sistema no realiza ningún cambio a dicho vendedor.
  - **Clase entidad:** Administrador.
  - **Clase de control:** Se creará la clase: sellers.

- **Administrar solicitud de vendedores.** El administrador podrá revisar y administrar las solicitudes de usuarios nuevos que quieran ser vendedores.
  - **Flujo básico:**
    1. El sistema mostrará un menú con la opción “Nuevos vendedores”.
    2. El administrador da clic en el botón “Nuevos vendedores” donde el sistema mostrará un listado con las solicitudes de nuevos vendedores.
    3. El administrador puede ver los detalles dando clic en el botón “Ver info”
    4. El administrador puede autorizar o rechazar la solicitud.
  - **Flujo alterno:**

    A1. El administrador solo ve las solicitudes sin realizar ningún cambio.

    A2. El administrador ve la solicitud, pero cambia de lugar en el menú sin enviar cambios.

  - **Escenarios:**
  - **Escenario 1:**
    - El administrador da clic en menú/nuevos vendedores.
    - El sistema muestra el listado de las solicitudes pendientes por revisar.
    - El administrador da clic para ver los detalles de dicha solicitud.
    - El administrador sale de la solicitud para revisarla después.
    - El sistema no realiza ningún cambio a dicha solicitud.
  - **Escenario 2:**
    - El administrador da clic en menú/ nuevos vendedores.
    - El sistema muestra el listado de las solicitudes pendientes por revisar.
    - El administrador da clic para ver los detalles de dicha solicitud.
    - El administrador cambia de “interfaz” sin realizar un cambio.
    - El sistema no realiza ningún cambio a dicha solicitud.
  - **Clase entidad:** Administrador.

- **Clase de control:** Se creará la clase: reg-vendedores.

- **Administrar solicitud de productos.** El sistema mostrará un listado con las peticiones de los usuarios vendedores para poder aceptar sus productos
  - **Flujo básico:**
    1. El sistema mostrará un menú con la opción solicitud de productos.
    2. El sistema mostrará una lista con las solicitudes, facilitando un botón para ver los detalles de la solicitud.
    3. El administrador podrá aceptar o rechazar la solicitud.
  - **Flujo alterno:**

    A1. El administrador solo ve las solicitudes sin realizar ningún cambio.

    A2. El administrador ve la solicitud, pero cambia de lugar en el menú sin enviar cambios.
  - **Escenarios:**
  - **Escenario 1:**
    - El administrador da clic en menú/solicitud de productos.
    - El sistema muestra el listado de las solicitudes pendientes por revisar.
    - El administrador da clic para ver los detalles de dicha solicitud.
    - El administrador sale de la solicitud para revisarla después.
    - El sistema no realiza ningún cambio.
  - **Escenario 2:**
    - El administrador da clic en menú/solicitud de productos.
    - El sistema muestra el listado de las solicitudes pendientes por revisar.
    - El administrador da clic para ver los detalles de dicha solicitud.
    - El administrador cambia de “interfaz” sin realizar un cambio.
    - El sistema no realiza ningún cambio.
  - **Clase entidad: Administrador.**
  - **Clase de control:** Se creará la clase:reg-products.

4.3 Diagrama de base de datos

El diagrama de la base de datos mostrado en la Figura 13, lo explicaremos en el siguiente apartado.

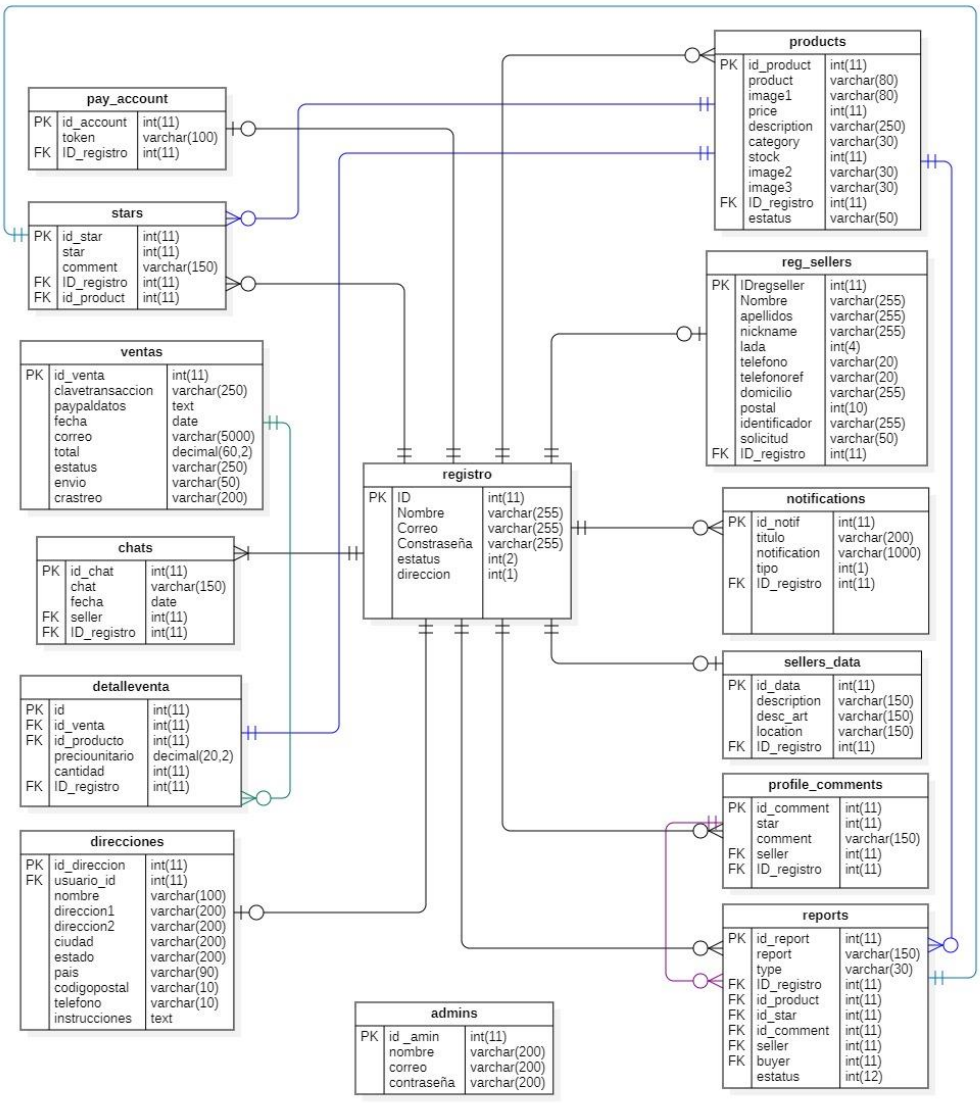


Figura 13. Diagrama de Base de datos.

La base se ha planeado de la siguiente manera:

- Registro
- Pay_account
- Stars
- Ventas
- Chats
- Detalleventa

- Direcciones
- Products
- Reg_sellers
- Notifications
- Sellers_data
- Profile_comments
- Reports
- Admins

Cada una de las tablas antes listadas serán explicadas una por una, ya que, cuentan con campos propios como también con campos de tipo llave foránea (foreign key). En la Tabla 1, observamos, la información correspondiente a la tabla Registro la cual se encarga de almacenar la información de todos los usuarios.

Tabla 1. Tabla Registro

Registro			
Llave	Identificador	Valor	Descripción
PK	ID	Int 11	Identificador de cada usuario
	Nombre	Varchar 255	Nombre del usuario
	Correo	Varchar 255	Correo del usuario
	Contraseña	Varchar 255	Contraseña del usuario
	estatus	Int 2	Estatus del usuario
	dirección	Int 1	Dirección del usuario

En la Tabla 2, observamos, la información correspondiente a las cuentas de pago de cada usuario de tipo vendedor.

Tabla 2. Tabla pay_account.

Pay_account			
Llave	Identificador	Valor	Descripción
PK	Id_account	Int 11	Identificador de la cuenta o del usuario vendedor
	token	Varchar 100	Campo para almacenar el token de cuenta que proporciona paypal
FK	ID_registro	Int 10	llave de tipo foránea para identificar a que usuario pertenece dicha información

En la Tabla 3, observamos, la información correspondiente a las calificaciones de cada artículo en venta.

Tabla 3. Tabla stars.

Stars			
Llave	Identificador	Valor	Descripción
PK	Id_star	Int 11	Identificador de la calificación
	star	Int 11	Cantidad de calificación
	comment	Varchar 150	Comentario de tipo texto
FK	ID_registro	Int 11	Llave de tipo foránea para identificar a que usuario corresponde dicha calificación
FK	Id_product	Int 11	Llave de tipo foránea para identificar a que producto corresponde dicha calificación

En la Tabla 4, observamos, la información correspondiente a cada descripción de venta que se realiza en la plataforma, donde, se almacena la información que proporciona la API (Application Programming Interface), de paypal.

Tabla 4. Tabla ventas.

Ventas			
Tipo	Identificador	Valor	Descripción
PK	Id_venta	Int 11	Campo que almacena él, ID de una venta
	clavetransaccion	Varchar 250	Campo que almacena la clave de verificación que envía paypal
	paypaldatos	text	Campo que almacena datos que envía paypal
	fecha	date	Campo que almacena la fecha de la compra
	correo	Varchar 5000	Campo que almacena el correo del comprador
	total	Decimal 60,2	Campo que almacena el total de la compra
	estatus	Varchar 250	Camp que almacena el estatus de la compra el cual puede ser pendiente, verificado u aprobado (estatus enviado por API de paypal)
	envio	Varchar 250	Campo que almacena el estatus del envío del producto comprado
	crastreo	Varchar 200	Campo que almacena el código de rastreo del producto comprado

En la **¡Error! La autoreferencia al marcador no es válida.** Observamos, la información correspondiente a cada dato, que se almacena cuando un usuario deja un chat, como él, ID del usuario que realiza él envío del mensaje, la información de fecha y hora, como también él, ID del usuario al que será enviado el mensaje.

Tabla 5. Tabla chats.

Chats			
Tipo	Identificador	Valor	Descripción
PK	id_chat	Int 11	Llave de tipo primaria que almacena él, ID del comentario
	chat	Varchar 150	Campo que almacena el texto del comentario
	fecha	date	Campo que almacena la fecha
FK	seller	Int 11	Campo que almacena el ID del usuario al que se le realiza el comentario
FK	ID_registro	Int 11	Campo que almacena él, ID del usuario que realiza el comentario

En la Tabla 6, observamos, la información que se almacena, la cual, es correspondiente a los detalles de una venta.

Tabla 6. Tabla detalleventa.

Detalleventa			
Tipo	Identificador	Valor	Descripción
PK	id	Int 11	ID del detalle de la venta
FK	Id_venta	Int 11	Llave de tipo foránea que almacena él, ID de una venta
FK	Id_producto	Int 11	Llave de tipo foránea que almacena él, ID del producto(s) comprado(s)
	preciounitario	Decimal 20,2	Campo que almacena el precio por unidad del producto comprado
	Cantidad	Int 11	Campo que almacena la cantidad del producto comprado
	ID_registro	Int 11	Campo que almacena él, ID del usuario comprador

En la Tabla 7, observamos, los datos correspondientes a la dirección que se pueden almacenar de un usuario comprador.

Tabla 7. Tabla direcciones.

direcciones			
Tipo	Identificador	Valor	Descripción
PK	Id_direccion	Int 11	Campo que almacena él, ID de los datos de cada comprador respecto a la dirección
FK	Usuario_id	Int 11	Campo que almacena él, ID del usuario al que corresponde esos datos de dirección
	nombre	Varchar 100	Campo que almacena el nombre del usuario
	Dirección 1	Varchar 200	Campo que almacena la dirección 1
	Dirección 2	Varchar 200	Campo que almacena la dirección 2 en caso de existir
	ciudad	Varchar 200	Campo que almacena la ciudad del usuario
	estado	Varchar 200	Campo que almacena el estado del usuario
	país	Varchar 90	Campo que almacena es país del usuario
	codigopostal	Varchar 10	Campo que almacena el código postal del usuario
	telefono	Varchar 10	Campo que almacena el teléfono del usuario
	instrucciones	text	Campo que almacena las instrucciones o referencias del domicilio

En la Tabla 8, observamos, los datos que se almacenan respecto a los productos de cada usuario vendedor, en este caso existen 3 tipos de estatus:

1. Aprobado: cuando el producto es revisado y aceptado por el administrador.
2. Pendiente: cuando el producto aún no ha sido revisado por el administrador.
3. Rechazado: cuando el producto ya fue revisado y ha sido rechazado por infringir las reglas de la plataforma.

Tabla 8. Tabla *products.*

Products			
Tipo	Identificador	Valor	Descripción
PK	Id_product	Int 11	Campo que almacena él, ID del producto
	product	Varchar 80	Campo que almacena el nombre del producto
	Image 1	Varchar 80	Campo que almacena la imagen 1 del producto
	price	Int 11	Campo que almacena el precio del producto
	description	Varchar 250	Campo que almacena la descripción del producto
	category	Varchar 30	Campo que almacena la categoría del producto
	stock	Int 11	Campo que almacena el stock del producto
	Image 2	Varchar 30	Campo que almacena la imagen 2 del producto en caso de existir
	Image 3	Varchar 30	Campo que almacena la imagen 3 del producto en caso de existir
FK	ID_registro	Int 11	Llave de tipo foránea que almacena él, ID del usuario vendedor
	estatus	Varchar 50	Campo que almacena el estatus del producto. Aprobado, Pendiente o Rechazado

En la Tabla 9, observamos, los campos correspondientes al registro de los usuarios de tipo vendedor. Como podemos observar, sucede algo similar que en la Tabla 8, ya que, tenemos un campo llamado solicitud el cual almacena el estatus del registro, en este caso:

1. Aprobado: cuando la solicitud del registro es revisada y aprobada por el administrador.

2. Pendiente: cuando la solicitud del registro no ha podido ser revisada por el administrador.
3. Rechazado: cuando la solicitud ya ha sido revisada por el administrador y esta ha sido rechazada, por infringir las normas de la plataforma.

Tabla 9.Tabla reg_sellers.

Reg_sellers			
Tipo	Identificador	Valor	Descripción
PK	IDregseller	Int 11	Campo que almacena él, ID del registro
	nombre	Varchar 255	Campo que almacena nombre del vendedor u organización
	apellidos	Varchar 255	Campo que almacena el apellido del vendedor u organización
	nickname	Varchar 255	Campo que almacena el alias del vendedor u organización
	lada	Int 4	Campo que almacena el número de lada correspondiente al número de contacto del vendedor u organización
	telefono	Varchar 20	Campo que almacena el teléfono de contacto del vendedor u organización
	teléfonoref	Varchar 20	Campo que almacena el teléfono de referencia del vendedor u organización
	domicilio	Varchar 255	Campo que almacena la dirección del vendedor u organización
	postal	Int 10	Campo que almacena el código postal del vendedor u organización
	identificador	Varchar 50	Campo que almacena la identificación del vendedor u organización
	solicitud	Varchar 50	Campo que almacena el estatus de la solicitud de registro. ()

FK	ID_registro	Int 11	Campo que almacena él, ID del usuario que realiza la solicitud de registro de tipo vendedor

En la Tabla 10, observamos, los datos almacenados que corresponden a las notificaciones. En este caso pueden corresponder a una notificación de comentario, notificación de solicitud, notificación de compra, notificación de reporte.

Tabla 10. Tabla notifications.

Notifications			
Tipo	Identificador	Valor	Descripción
PK	Id_notif	int 11	Campo que almacena él, ID de la notificación
	titulo	Varchar 200	Campo que almacena el texto que mostrará en el título de la notificación el cual corresponde al tipo de notificación
	notification	Varchar 1000	Campo que almacena el texto ingresado para la notificación en caso de existir
	Tipo	Int 1	Campo que almacena el tipo de la notificación
FK	ID_registro	Int 11	Campo que almacena él, ID el usuario al cual se le mandara la notificación

En la Tabla 11, observamos, los campos correspondientes a la información de los perfiles de los usuarios vendedores.

Tabla 11. Tabla sellers_data.

Sellers_data			
Tipo	Identificador	Valor	Descripción
PK	Id_data	Int 11	Campo que almacena él, ID de la información del vendedor
	Description	Varchar 150	Campo que almacena la descripción del perfil de vendedor
	Desc_art	Varchar 150	Campo que almacena la descripción de los artículos en general del vendedor
	location	Varchar 150	Campo que almacena la descripción de la dirección del vendedor u organización

FK	ID_registro	Int 11	Campo que almacena él, ID del usuario al que pertenece la información registrada

En la Tabla 12, observamos, los campos correspondientes a la información de los comentarios en los perfiles de los vendedores.

- Type: este campo se refiere al tipo de reporte en los cuales se encuentran
  - Comentario: si un reporte de comentario es aprobado el comentario es eliminado.

Tabla 12. Tabla profile_comments.

Profile_comments			
Tipo	Identificador	Valor	Descripción
PK	Id_comment	Int 11	Campo que almacena él, ID del comentario
	star	Ins 11	Campo que almacena la calificación del vendedor
	comment	Varchar 150	Campo que almacena el texto del comentario
FK	seller	Int 11	Campo que almacena él, ID del usuario vendedor
FK	ID_registro	Int 11	Campo que almacena él, ID del usuario que deja el comentario

  - Producto: si un producto es reportado el usuario tiene la posibilidad de actualizar su producto para devolverlo a revisión, sin embargo, el producto permanecerá dado de baja y no se mostrará en el catálogo.
  - Comentario en perfil: si este tipo de reporte es aprobado el comentario será eliminado.
  - Perfil de vendedor: si el reporte de perfil es aprobado, el perfil será dado de baja de la plataforma.
- Estatus: este campo se refiere al estatus de revisión por parte del administrador en los cuales se encuentran
  - Aprobado
  - Rechazado

En la Tabla 13, observamos, los campos correspondientes a la información de los reportes. Donde, encontramos los campos:

Tabla 13. Tabla reports.

Reports			
Tipo	Identificador	Valor	Descripción
PK	Id_report	Int 11	Campo que almacena él, ID del reporte
	report	Varchar 150	Campo que almacena el texto del reporte
	type	Varchar 30	Campo que almacena el tipo de reporte
FK	ID_registro	Int 11	Llave foránea que almacena él, ID de registro del usuario que realiza reporte
FK	Id_product	Int 11	Llave foránea que almacena él, ID del producto reportado
FK	Id_star	Int 11	Llave foránea que almacena él, ID de la calificación del producto o perfil
FK	Id_comment	Int 11	Llave foránea que almacena él, ID del comentario reportado
FK	seller	Int 11	Campo que almacena él, ID del vendedor al que se reportara
FK	buyer	Int 11	Campo que almacena él, ID del usuario que deja el reporte
FK	estatus	Int 11	Campo que almacena el estatus del reporte

En la Tabla 14, observamos, los campos correspondientes a la información de los administradores. Ya que, por ser administradores la plataforma no dejara que la

tabla de la base de datos sea vaciada por completo, esto quiere decir que, siempre debe de haber por lo menos un solo administrador

Tabla 14. Tabla admins

Admins			
Tipo	Identificador	Valor	Descripción
PK	Id_admin	int 11	Campo que almacena él, ID del administrador
	nombre	Varchar 200	Campo que almacena el nombre del administrador
	Correo	Varchar 200	Campo que almacena el correo del administrador
	contraseña	Varchar 200	Campo que almacena la contraseña del administrador

4.4 Backlog

Para el backlog fue necesario iniciarlo con la reunión principal donde se dejan en claro los puntos más importantes de todo el proyecto los cuales son:

- Descripción del proyecto en general.
- Planeación de la documentación a entregar.
- Planeación de casos de uso a crear.
- Planeación de métodos de comunicación.
- Planeación de las herramientas a usar.

Esta reunión es la que afianzará las bases del buen trabajo en equipo y los objetivos para poder lograr el producto final el cual será la plataforma Artxicans.

La descripción completa del backlog se presenta en la Tabla 15.

.

Tabla 15. Tabla de Backlog.

Prioridad	Sprint	Entregables	Inicio	Duracion
1	Creacion de index y manejo de opciones basicas de la	Vinculos a cada opcion del menu y filtro para categorias.	23 de abril de 2023	2 dias
2	Creacion de formulario Inicio de sesion y registro de usuario	back inicio de sesion.	03 de mayo de 2023	2 dias
		back registro de usuarios.		
3	Creacion de catalogo en index	Creacion y conexión a base de datos y filtro para mostrar los productos.	05 de mayo de 2023	5 dias
4	Creacion de categorias en "categories.php" y filtro por	Filtro para muestra de productos según la categoria y existencia.	08 de mayo de 2023	1 dias
5	Creacion de formulario de registro de vendedores	Formulario de registro de vendedores back	06 de mayo de 2023	2 dias
6	Creacion de carrito de compras y conexión paypal	Carrito de compras "cart.php"	24 de mayo de 2023	2 dias
		Agregar al carrito	27 de mayo de 2023	1 dias
		Formulario de ingresar direccion	29 de mayo de 2023	1 dias
		Modificar carrito	31 de mayo de 2023	2 dias
		Conexión de API paypal	03 de junio de 2023	4 dias
		Pagar carrito	07 de junio de 2023	2 dias
		Pedidos	10 de junio de 2023	1 dias
7	Creacion de plantilla de Administrador	Inicio de sesion, panel administrador	11 de junio de 2023	2 dias
8	Funciones de panel de admistrador	Enlaces a funciones principales	19 de junio de 2023	1 dias
		Lista de pedidos	21 de junio de 2023	2 dias
		Lista de productos		
		Lista de vendedores		
		Reportes	24 de junio de 2023	6 dias
		Solicitud de vendedores		
		Solicitud de productos		
9	Funciones de panel de usuario	Notificaciones	01 de julio de 2023	2 dias
		Mis pedidos		
		Editar datos	04 de julio de 2023	1 dias
		Cerrar sesion	01 de julio de 2023	0 dias
10	Funciones de comentarios	Comentar	02 de julio de 2023	4 dias
		Calificar		
11	Funciones de reportes	Reportar comentario	08 de julio de 2023	16 dias
		Reportar producto		
		Reportar perfil de vendedor		
12	Funciones de usuarios vendedores	Lista de productos	24 de julio de 2023	1 dias
		Listado de pedidos		
		Listado de paqueteria		
		Registrar nuevos productos	26 de julio de 2023	2 dias
13	Funciones de perdil de usuario vendedor	Editar informacion perfil de vendedor	29 de julio de 2023	3 dias
14	Funciones de chat	Funcionamiento de chat	02 de agosto de 2023	2 dias
15	Funciones de manejo de pagos	registro de token para pago	05 de agosto de 2023	2 dias

4.5 Artxicans

Sección donde se describe el funcionamiento de la plataforma respecto al BackEnd. Al ser una página de ventas es imperativo que se muestre un catálogo random para poder mostrar productos y así poder llamar la atención del cliente a productos nuevos. De igual forma se incentiva a mostrar productos con buenas calificaciones como con calificaciones inferiores.

También se muestran los carruseles de categorías al azar, para mostrar productos un poco más exactos que el catálogo principal del índex.

Como menú principal se presenta la barra superior donde se muestra el logo de la página con sus respectivos submenús donde se incluyen:

- Conexión a base de datos
- Registro nuevo usuario
- Inicio o índex
- Categorías
- Vender
- Ayuda
- Idioma
- Carrito de compras

Todas estas partes se explican a continuación, pero antes de iniciar es necesario comentar y mostrar la parte de la conexión a la base de datos.

4.5.1 Conexión a base de datos

Para la implementación de la base de datos es necesario seguir el diagrama mostrado en la parte Diagrama BD, para así, no tener problema alguno en la creación y diseño.

Para incluir la base de datos a la plataforma fue necesario crear el archivo “Conexión.php” que contiene el código presentado en la Figura 14.

```php
<?php
    define("KEY","aRtXiCaNs");
    define("COD","AES-128-ECB");
    define("SERVIDOR","localhost");
    define("USUARIO","root");
    define("PASSWORD","");
    define("BD","artxicans");

    try{
        $conn = mysqli_connect(SERVIDOR,USUARIO,PASSWORD,BD);
    }catch(PDOException $e){
        echo "<script>alert('Error...')</script>";
    }
?>
```

Figura 14. Archivo de conexión: “Conexion.php”

Donde se definen las variables de conexión

- Servidor: donde se establece el nombre del servidor donde se realizará la conexión.
- Usuario: donde se define el usuario con el que se registrara la conexión.
- Password: donde se define la contraseña correspondiente al usuario especificado en la variable anterior.
- BD: donde se especifica el nombre de la base de datos a la cual se realizará la conexión.

Si alguno de estos datos falta o es erróneo ninguna query de búsqueda, actualización o guardado. funcionará.

También se definen las variables de encriptación las cuales son necesarias para encriptar los datos de las ventas, los cuales son obtenidos del api de paypal.

- Key: donde se define la llave para desencriptar los datos
- Cod: donde se define la variable del tipo de encriptación

Todo esto, para que al final se use un try-catch, donde, si por dado caso suceda un inconveniente con la conexión mande un mensaje de error, en caso contrario proceda a la conexión con ayuda de las variables antes descritas.

La variable final es “$conn”, con ella se realizan todas las consultas y modificaciones a la base de datos.

4.5.2 Inicio de sesión

En esta parte de la plataforma se inicia la explicación, ya que, por ser la parte más esencial de la plataforma donde se rige el tipo de usuario y los tipos de menús usados.

Para los registros de nuevos usuarios en la plataforma se cuenta con el formulario de inicio de sesión el cual será dividido en dos partes:

1 Inicio de sesión.
2 Registro de nuevo usuario.

Sin embargo, hay algo que comentar antes de pasar a la explicación, cada tipo de usuario tiene su identificador, con el cual se presentan los diferentes menús o muestreo de información. Lo cual consta de lo siguiente:

- Usuario nuevo: identificador 0, usuario de tipo comprador.
- Usuario vendedor: identificador 1, usuario de tipo vendedor.

La interfaz de inicio de sesión se muestra en la Figura 15.

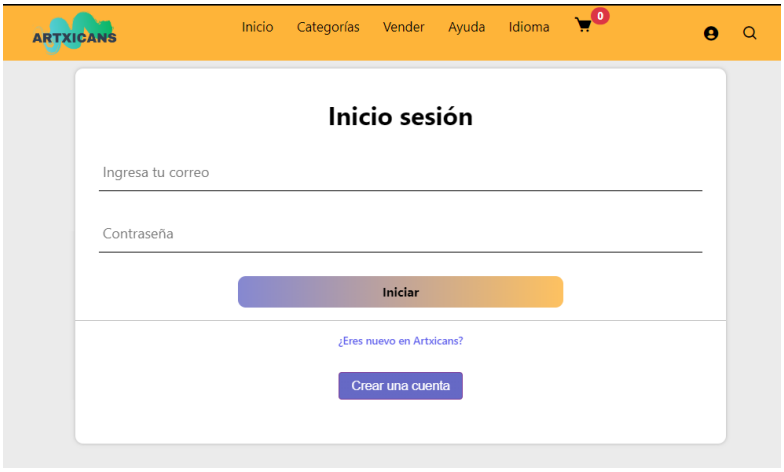


Figura 15. Formulario inicio de sesión.

4.5.2.1 Inicio de sesión

En la parte del BackEnd fue necesario crear el archivo llamado “login.php”, el cual consta del código que se presenta en la Figura 16.

```php
<?php
    if (isset($_REQUEST['login']))
        {
            $email = $_REQUEST['email']??'';        # variables obtencion de email del input html
            $password = $_REQUEST['pass']??'';  # variables obtencion de password del input html
            $sql = "SELECT * FROM registro WHERE Correo='".$email."' and Contraseña='".$password."' "; # query para obtener la busqueda de los campos
            $result = mysqli_query($conn,$sql);   # se obtiene el resultado de la query
            if ($result->num_rows > 0)  # se obtiene el valor de $result
                {
                    /* header('Location: index.php'); */
                    echo("<script>location.href = 'index.php';</script>");
                    $row = mysqli_fetch_assoc($result);    # registro de la peticion
                    $_SESSION['id']=$row['ID'];   # se almacena la sesion en un array
                    $_SESSION['user'] = $row['Nombre'];
                    include('./helpers/loader.php');
                }
            else
                {
                    ?>
                        <div class="alert alert-danger" role="alert" >Contraseña/Correo incorrecto</div>
                    <?php
                }
        }
?>
```

Figura 16. Código de inicio de sesión.

En el cual se inicia con un if, que verifica que se haya dado clic en el botón iniciar sesión, si dicho botón fue pulsado, se obtienen los datos, en este caso el correo ($email) y la contraseña ($password). Para después realizar una búsqueda en la base de datos en la tabla “registro”, donde se verifica que el correo y contraseña ingresados en los inputs coincidan con el usuario especificado.

Una vez que la consulta fue validada se procede a redireccionar al índex de la plataforma al usuario que inicio sesión y se guardan las variables de sesión ($SESSION) haciendo índice al ID y al nombre que se obtuvieron de la tabla “registro” con ayuda de la query de búsqueda.

4.5.2.2 Registro de nuevo usuario

Como se muestra en la Figura 15 se registra un nuevo usuario al dar clic en el botón “Crear una cuenta”. Lo que presenta nuevo formulario (Figura 17).

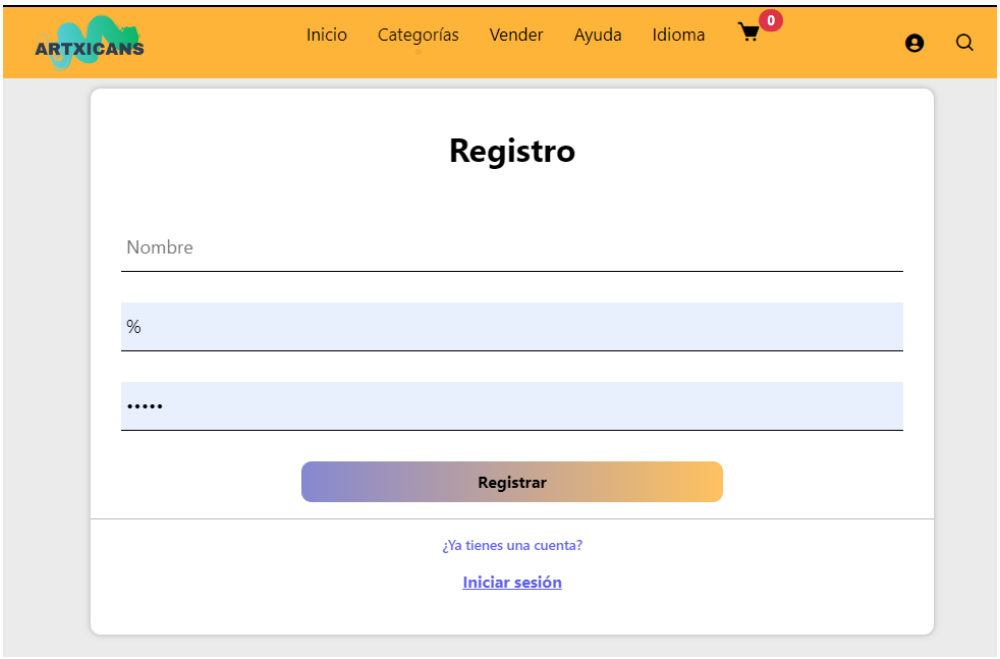


Figura 17. Formulario de registro.

En el cual se solicitarán 3 datos:

1 Nombre
2 Correo
3 Contraseña

Los cuales, una vez llenos el usuario deberá dar clic en registrar, dichos datos se almacenarán en la base de datos en la tabla “registro” (como se muestra en la Figura 17).

Esta parte se logró con ayuda del archivo llamado “registrar.php”, que contiene el código presentado en la Figura 18.”

```php
if(isset($_POST['reg']))
        {
            $nombre = $_POST['name'];       #
            $correo = $_POST['email'];      #   Obtencin de datos
            $password = $_POST['pass'];     #
            $sql = (" SELECT * FROM registro WHERE Correo='$correo' ");
            $result = mysqli_query($conn,$sql);
            if (!$result->num_rows > 0)
```

Figura 18. Código de registro.

Donde, primero se verifica que se dio clic en el botón registrar y si este clic es correcto se obtienen los datos correspondientes a los listados al principio de esta sección.

Después se realizan algunas verificaciones o filtros de seguridad, los cuales constan de lo siguiente:

- Duplicado de correo
- Nombre menor a 4 caracteres
- Contraseña menor a 4 caracteres

Si alguno de estos datos llega a ser incorrecto el sistema no dejara realizar el registro correctamente hasta que los datos solicitados sean correctamente ingresados.

Cuando los datos sean correctos se hará uso de una query de tipo inserción (el código correspondiente se presenta en la Figura 19), en este caso se insertan los datos en la tabla “registro”, cuando un usuario se registra por primera vez su estatus es igual a 0 lo que indica que no es de tipo vendedor, sin embargo, este estatus puede cambiar si el usuario decide realizar su registro para poder ser un usuario vendedor, lo cual se explicara más adelante.

```php
$sql = (" INSERT INTO registro VALUES(NULL,'$nombre','$correo','$password','0','0')");
$result = mysqli_query($conn,$sql);
if ($result)
    {
        include('./helpers/loader.php');
        echo("<script>location.href = 'login.php';</script>");
    }
```

Figura 19. Inserción de datos de registro.

4.5.3 Inicio o índex

Como se ha mencionado en el apartado anterior, el índex se compone de un catálogo sin filtros de muestreo de calificación o más vendido, pensado así, para que todos los vendedores tengan la misma oportunidad de sacar ventas de todos sus productos por igual (véase la Figura 20).

El código BackEnd se explica a lo largo de este apartado.

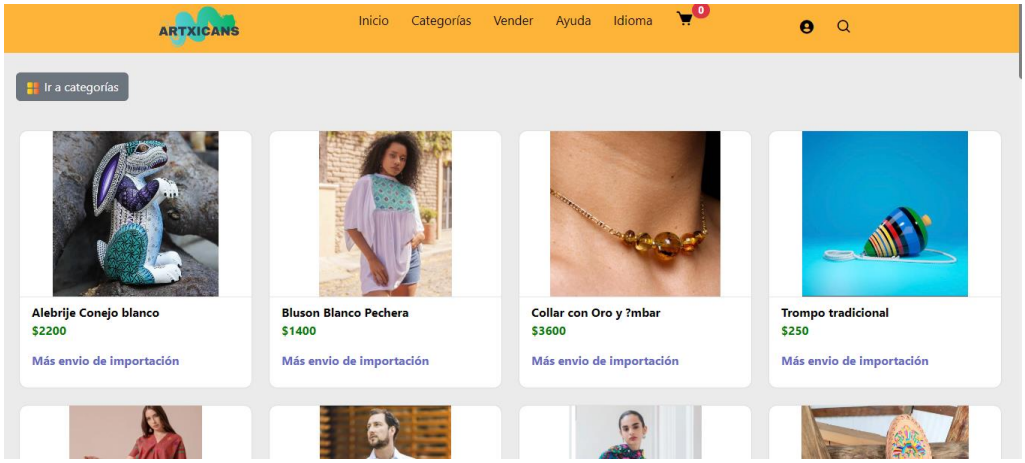


Figura 20: Índex de Artxicans.

Para realizar el catálogo en la parte del índex será necesario hacer uso del archivo descrito en la parte conexión a base de datos, además, se creará una consulta para poder acceder a los datos de la base de datos.

Para esta parte se hará uso de la variable “$result” la cual ayudará a obtener los valores de la consulta, el código se presenta en la Figura 21.

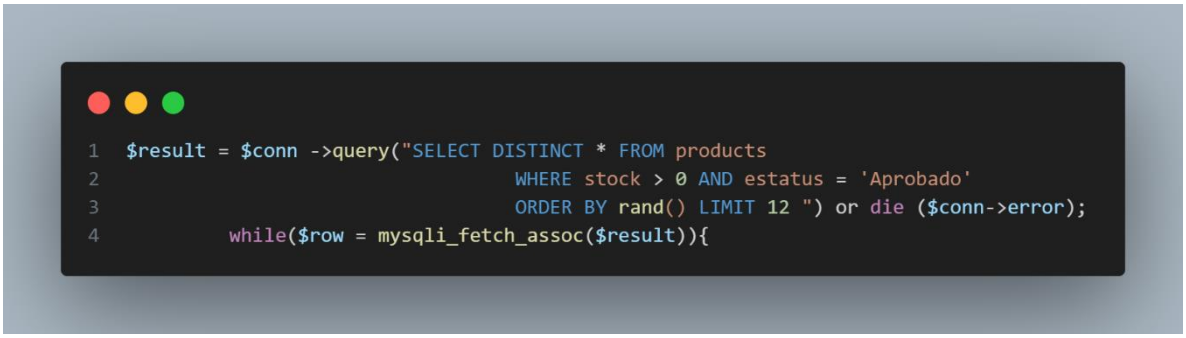


Figura 21. Query de consulta.

- $conn: variable de conexión descrita en la parte 5.6.1
- Query: variable de consulta donde se ingresa la consulta en lenguaje SQL
- $row: variable que obtiene los campos de la información obtenida basada en la query

Los parámetros que utilizados para la consulta se muestran en la Figura 21.

- Select distinct *: Seleccionar todos los campos de la tabla a excepción de los campos duplicados.
- From products: Seleccionar la tabla con la que trabajaremos.
- Where stock > 0 and estatus = 'Aprobado': Seleccionar los campos los cuales no tengan un stock igual a 0 y su estatus sea igual a Aprobado (estos campos se explican más adelante).
- Order by rand(): Todos los datos obtenidos se almacenan de forma aleatoria sin ningún filtro en especial.
- Limit 12: Se guardan 12 elementos en cada matriz de datos obtenida.

Para mostrar los datos obtenidos por la consulta, es necesario presentarlos con ayuda de un while, el cual, se describirá un poco más adelante.

Dentro del while (código de la Figura 22) se usa la variable $row, para obtener los datos que genera la consulta, en este caso como la consulta almacena un arreglo de resultados con los nombres y datos exactos que contiene la base de datos entonces solo es necesarios acceder a dicho arreglo con los nombres de los campos de la base de datos para que puedan ser visualizados con éxito.

Como ejemplo se muestra la Figura 22 donde se implementa el código de acceso a cada dato de información que almacena la variable $result de la query.

```php
while($row = mysqli_fetch_assoc($result))
        {
?>
            <div class="single-product">
            <a href="product.php?id_product=<?php echo $row['id_product'];?>" class='referencia'>
                 <div class='imgDiv'>
                    <img src="assets/products/<?php echo $row['image1'];?>"  alt="<?php echo $row['product']?>">
                </div>
                <div class="card-info">
                    <span class="product-title"><?php echo $row['product'];?></span>
                    <span class="price">$<?php echo $row['price'];?></span>
                    <span class="shipping">Más envio de importación</span>
                </div>
               </a>
            </div>
<?php
              $id = $row['id_product'];
        }
?>
```

Figura 22. Muestreo de información.

- $row['id_product']: donde se obtiene él, ID del producto seleccionado en el catálogo.
- $row['image1']; donde se obtiene el nombre de la imagen que se muestra en el catálogo.
- $row['product']: donde se obtiene el nombre del producto seleccionado.
- $row['price']: donde se obtiene el precio del producto seleccionado.

Para el Footer (pie de página de la Figura 23), solo fue necesario agregar los vínculos a las diferentes páginas que contienen la información solicitada.

Para el footer se crea el archivo “pie.php” (Figura 24) donde se muestra todo el contenido y según la opción se visualiza la página seleccionada. El código correspondiente se presenta en la Figura 25.

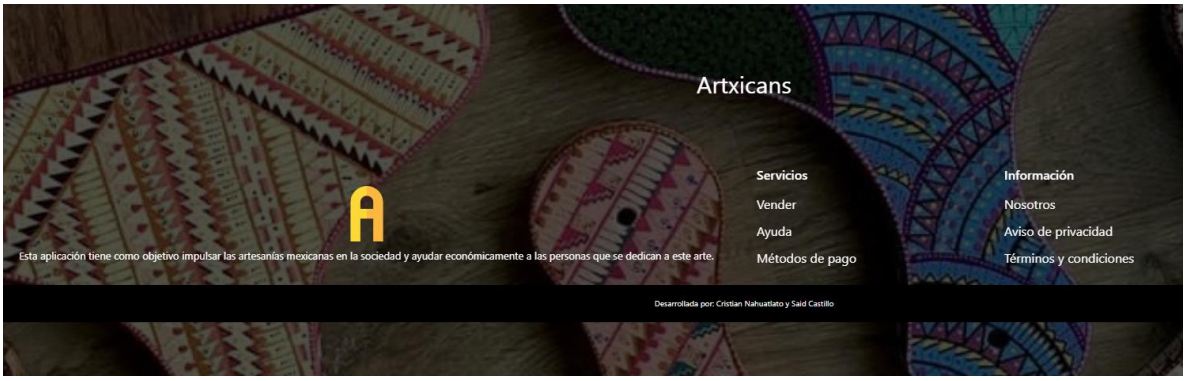


Figura 23. Footer o pie de página.

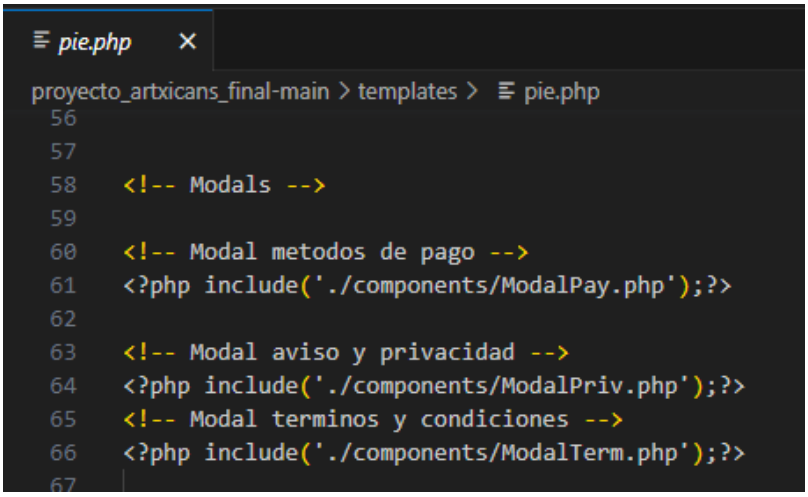


Figura 24. Archivo "pie.php"

Figura 25. Porción de código en "pie.php"

4.5.4 Categorías

Para las diferentes categorías que se muestran en la plataforma, se ha creado el archivo “categories.php”, donde, se usa una opción de filtro, o variable de opción la cual pasa como parámetro la variable “op”, la cual, almacena la palabra clave de los artículos a buscar (véase la Figura 26).

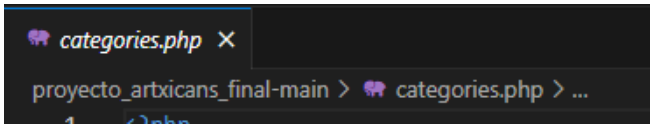


Figura 26. Archivo "categories.php"

Se dirige al archivo “pcategoria.php”, donde, se pasa como parámetro la palabra clave “Alebrijes” (como se muestra en la Figura 27), la cual, con ayuda de la variable “op” se presentan artículos que lleven por categoría “Alebrijes” (como se muestra en la Figura 28).

Figura 27. Parámetros para categorías.

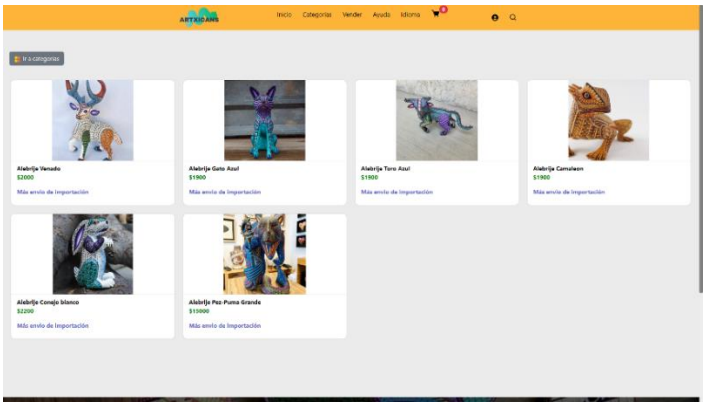


Figura 28. Categoría: Alebrijes.

Como este parámetro se obtiene por “pcategorias.php”, se debe crear creamos “pcategoria.php”, en el cual incluimos los archivos “conexión.php”, “cabecera.php” y “loader.php” (véase la Figura 29), descritos con anterioridad.

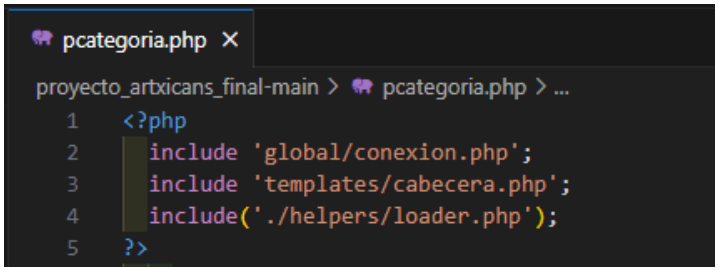


Figura 29. Archivo "pcategoria.php".

Todo lo que será descrito en las próximas líneas, hasta que finalice este apartado, será explicado basándonos en el código que se muestra en la Figura 30.

Primero, se verifica que la variable “op” no sea nula, lo cual se hace con la ayuda de una estructura “if” y $_GET, donde, si la variable es vacía el bloque de código no se ejecuta, pero si tiene un valor, se lleva a cabo todo el bloque de código:

- $opcion: consulta SQL para buscar los productos por categoría con ayuda de la variable “$opcion” la cual es igual a la variable “op”.
- $result: variable que ejecuta la consulta “$opcion”.

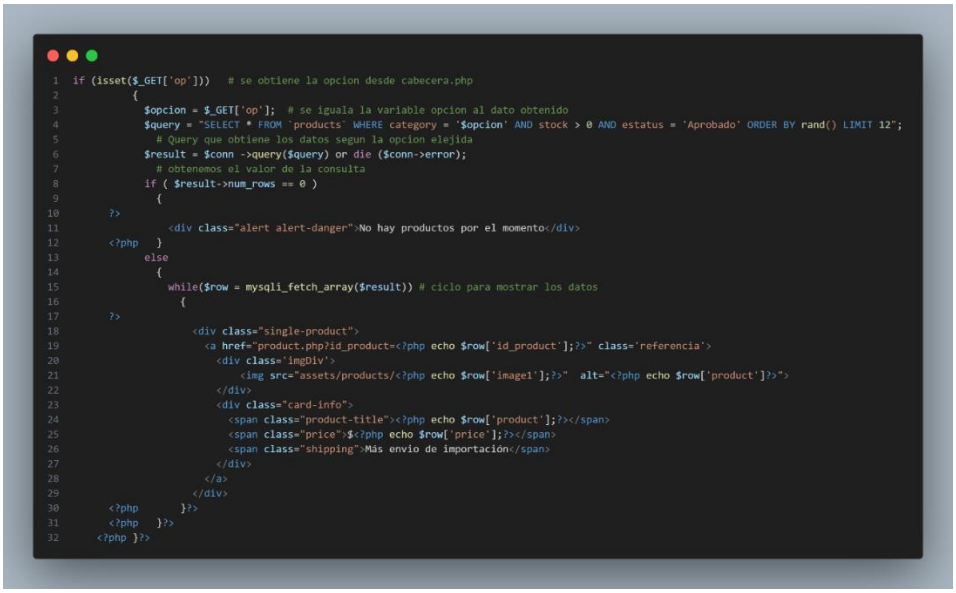


Figura 30. "pcategoria.php" código BackEnd.

Con ayuda de la consulta y la variable de resultados de esta, se crea otra estructura “if”, solo que ahora se verifica que la variable “$result” haya obtenido resultados con ayuda de la función “num_rows”.

Si la consulta, no obtuvo resultado alguno, solamente presenta un div con el texto “No hay productos por el momento”, con muestra los datos almacenados en la variable “$row” con ayuda de un ciclo de repetición while, en dado caso que los resultados sean mayores a 1.

4.5.5 vender

En esta parte de la plataforma será dividida en 2 partes, ya que, si un usuario nuevo (tipo 0) ingresa a esta parte de la plataforma, solo se visualiza una página que muestra la información y posibilidad de registrarse como un vendedor (Figura 31).

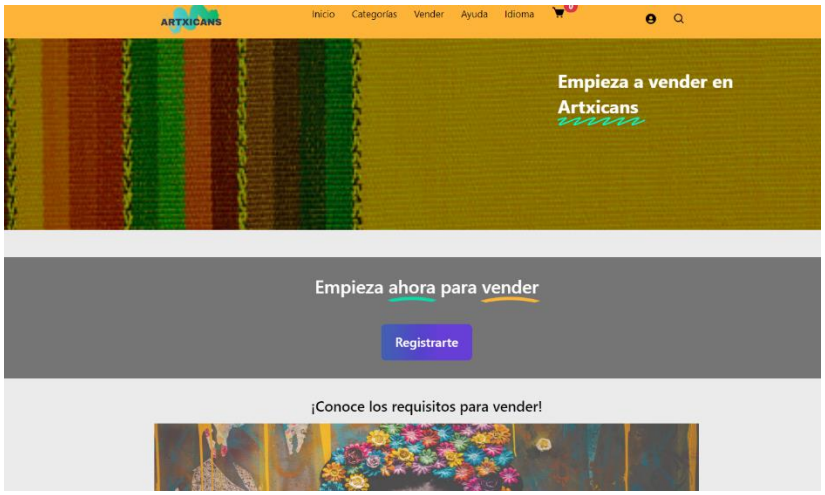


Figura 31. Vender usuario "nuevo”.

Si el usuario que ingrese a esta parte de la plataforma ya cuenta con un perfil de vendedor (tipo 1), esta parte cambia y presenta el menú de vendedor (véase la Figura 32), en el cual tiene las siguientes opciones:

- Mis productos
- Mis pedidos
- Mi perfil

- Mi cuenta de pagos

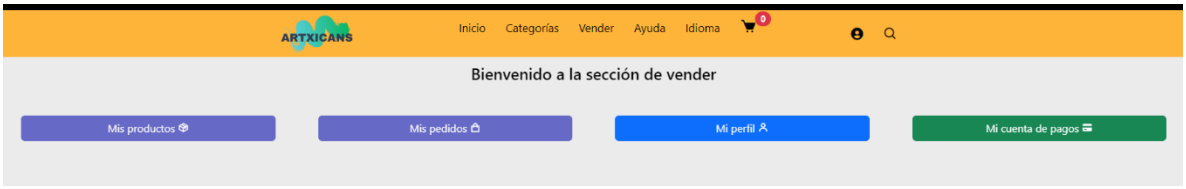


Figura 32. Menú de vendedor.

Pero ¿cómo se sabe que un usuario está registrado como un usuario vendedor? Para ello usamos el código de la Figura 33, el cual se compone de:

- If: el cual verifica que la variable de sesión no este vacía.
- $query: la cual con ayuda de la variable “$id_user”, realiza una búsqueda en la tabla registro, en base del ID almacenado en la variable.
- $res: variable que obtiene los datos encontrados.

```php
<?php
    include('../global/conexion.php');
    session_start();
    /* error_reporting(0);*/
    if(@!$_SESSION['user']){
        echo("<script>location.href = '../sell.php';</script>");
    } else{
        $id_user = $_SESSION['id'];
        $query = mysqli_query($conn,"SELECT * FROM registro WHERE ID = '$id_user'");
        $res = mysqli_fetch_array($query);

        if($res['estatus'] == '1'){
            $_SESSION['roll'] = $res['estatus'];
            echo("<script>location.href = '../seller.php';</script>");
        }else{
            echo("<script>location.href = '../sell.php';</script>");
        }
    }
?>
```

Figura 33. "validate-seller.php"

Cuando estos datos son obtenidos, se verifica la información obtenida con ayuda de la variable “$res” en el índice “estatus”, validando que el valor sea igual a 1.

Donde estatus se obtiene de la base de datos, en la tabla registro, el cual, cuando un usuario es aceptado como vendedor, su estatus cambia a 1 y si el usuario no es aceptado o simplemente no le interesa registrarse como vendedor, el estatus en la tabla queda en 0.

4.5.5.1 Usuario nuevo

Siendo un usuario nuevo tiene “estatus” de 0 en la base de datos, lo cual indica que no hay activo un perfil de vendedor en este caso la interfaz es la que presenta la Figura 34.

Dicha interfaz muestra el botón “Registrarte” y en las demás partes visualiza información respectiva a la misma página (Figura 34).

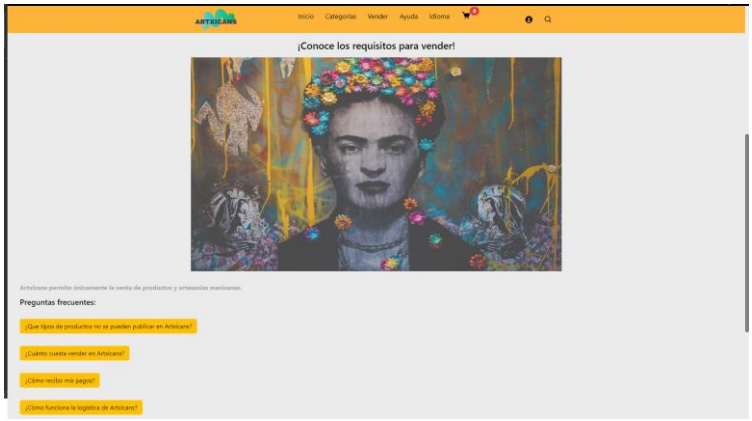


Figura 34. FAQ interfaz "vender".

Cuando un usuario nuevo quiera registrarse como vendedor, tendrá que realizar su registro dando clic en el botón “Registrarte” (Figura 35).

Una vez que sean aceptados los comentarios que se muestran en la ventana, el sistema lo re-direcciona hacia un formulario donde se piden los datos:

- Nombre
- Apellidos
- Identificaros (Nick name)

- Lada
- Número de teléfono
- Número de teléfono de referencia
- Domicilio
- Código postal
- Identificación oficial

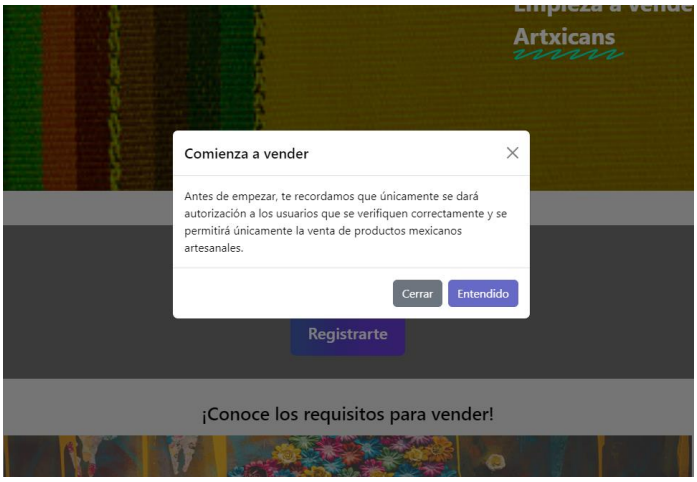
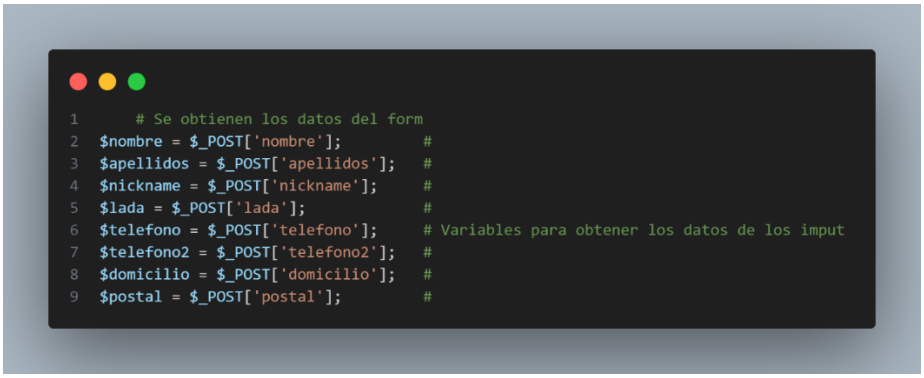


Figura 35. Inicio del registro para vendedor.

Cabe mencionar que estos datos son requeridos para poder validar la veracidad de la locación si es que existe una tienda física o de otra forma algún taller donde el artesano trabaje. Como también estos datos pueden coincidir con los datos registrados en el perfil de usuario “normal”.

Toda esta información se almacena en la base de datos, en la tabla “reg_sellers” como se describe en las siguientes líneas.

Antes de poder realizar la inserción de los datos a la tabla, es necesario realizar la obtención de los datos ingresados en cada input (como en el código presentado en la Figura 36), para ello se usa el método “$_POST”, que permite recibir lo que esté almacenado, haciendo referencia al atributo name o nombre de cada input, asignándolo a una variable correspondiente.



```php
    # Se obtienen los datos del form
$nombre = $_POST['nombre'];         #
$apellidos = $_POST['apellidos'];   #
$nickname = $_POST['nickname'];     #
$lada = $_POST['lada'];             #
$telefono = $_POST['telefono'];     # Variables para obtener los datos de los imput
$telefono2 = $_POST['telefono2'];   #
$domicilio = $_POST['domicilio'];   #
$postal = $_POST['postal'];         #
```

Figura 36. “regseller.php” validación de datos

Para verificar la información, antes de poder almacenarla a la tabla, se realizó un filtrado o verificación de información, que se presenta de la siguiente manera (descripción de validación en la Tabla 16 y presentación de código en Figura 37).

Tabla 16. Validación de información.

Variable	Validación
Todas las variables	Ningún campo quede vacío
Nombre	Que la longitud no sea menor a 4
Identificador o Nick name	Que no se repita el mismo identificador
Lada	No sea menor a 2 ni mayor a 4
Teléfono	Que no sea menor a 7 y no mayor a 10
Teléfono de referencia	Que no sea menor a 7 y no mayor a 10
Código postal	Que no sea menor a 4 ni mayor a 10

```php
if (!empty($_POST['nombre']) && !empty($_POST['apellidos']) && !empty($_POST['nickname'])
      && !empty($_POST['lada']) && !empty($_POST['telefono']) && !empty($_POST['telefono2'])
      && !empty($_POST['domicilio']) && !empty($_POST['postal']) )
      {
          if (strlen($nombre) < 4)
              {
              }
          elseif ($resultado->num_rows > 0)
              {
              }
          elseif ($cont < 2 or $cont >= 4)
              {
              }
          elseif (strlen($telefono) < 7 or strlen($telefono) > 10)
              {
              }
          elseif (strlen($telefono2) < 7 or strlen($telefono2) > 10)
              {
              }
          elseif (strlen($postal) < 4 or strlen($postal) >= 10)
              {
              }
```

Figura 37. Validación de información.

Para finalizar con la inserción de los datos a la tabla, se realiza el código de la Figura 38.

```
$sql = ("   INSERT INTO reg_sellers
            VALUES(NULL,'$nombre','$apellidos','$nickname','$lada','$telefono',
            '$telefono2','$domicilio','$postal','$name_imagen','Pendiente','$id_user')");
$result = mysqli_query($conn,$sql);
```

Figura 38. Consulta query para inserción.

- $sql: Variable de tipo query para almacenar las instrucciones SQL, insertando en la tabla “reg_sellers”, los datos que obtenidos con anterioridad y almacenarlos de forma ordenada en los campos correspondientes
- $result: Variable de tipo ejecución con la cual realizamos la conexión a la base de datos y se realiza la ejecución de la query.

Una vez ejecutada la query los datos han sido almacenados con éxito, sin embargo, el proceso aún no ha sido terminado, cuando el usuario realiza su registro como vendedor los datos ingresados tendrán que ser validados por un administrador. Cuando los datos sean aceptados el estatus del usuario pasará de 0 a 1 y con esto el registro estará completo, ahora al dar clic en el botón vender ya no se mostrará la interfaz de registro, sino que, se mostrará el panel de vendedor.

4.5.5.2 Usuario vendedor

Cuando un usuario ya es de tipo vendedor, esto quiere decir que su estatus es de tipo 1. El panel de vendedor se presenta como se describe en el apartado 4.5.5 vender (Figura 32).

Siendo un usuario de tipo vendedor, su “estatus” en la base de datos es de 1, lo cual indica que hay activo un perfil de vendedor en este caso la interfaz se visualiza en la Figura 32. Dicha interfaz tiene las siguientes opciones:

1. Mis productos
2. Mis pedidos
3. Mi perfil
4. Mi cuenta de pagos

Ya que, cada opción es diferente, la explicación la dividiremos en 4 partes siguiendo la lista

1. **Mis productos.** En esta opción se visualizan todos los productos que han sido registrados por el vendedor, en este caso se incluye una tabla, en la cual, se muestran los elementos de la tabla “products” perteneciente a la base de datos, junto con los botones “Editar”, “Eliminar” y por último el botón “Agregar producto” (como se muestra en la Figura 39).

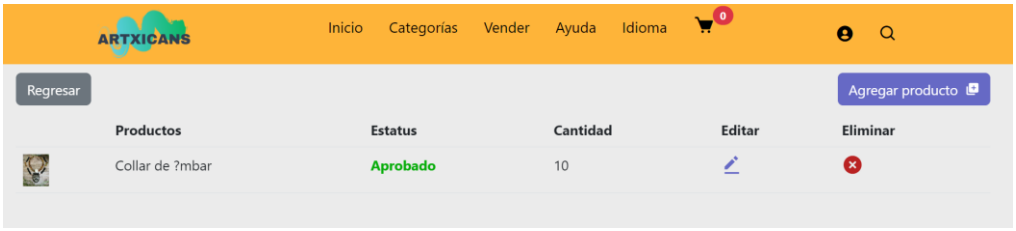


Figura 39. Mis productos, menú vendedor.

La lista de productos se genera realizando una consulta SQL dirigida a la tabla correspondiente, con ayuda de la variable “$_SESSION” que guarda el código de sesión del usuario actual (código en la Figura 40).

```
$id_user = $_SESSION['id'];
    $query = mysqli_query($conn,"SELECT * FROM products WHERE ID_registro = $id_user");
    if($query->num_rows > 0)
      {
        while($row = mysqli_fetch_array($query)){
```

Figura 40. query de búsqueda lista de productos.

Como se menciona anteriormente, se realiza un barrido a toda la tabla “products” y con la ayuda de la variable “id_user”, se obtendrán los productos asociados a ese usuario, los resultados serán mostrados con la ayuda de la variable “$row”, ya que, dicha variable almacena una matriz de datos, se muestran los valores con ayuda de una sentencia while, para recorrer la matriz haciendo referencia al nombre de la columna en la fila de resultados obtenidos (código mostrado en la Figura 40).

Ejemplos de acceso a los datos obtenidos:

- $nombre = $row['nombre'];
- $precio = $row['precio'];
- $cantidad = $row['stock'];

1.1. Editar. En esta interfaz del proyecto el usuario vendedor puede ver y modificar con precisión cualquier detalle de sus productos. La plataforma presentará un nuevo formulario, en el cual, el usuario vendedor puede modificar o no, la información de su producto (véase la Figura 41). Para realizar esta acción respecto al *Back-End* es necesario, obtener los datos que se almacenan en el formulario, para así después realizar el update respecto a la tabla en la base de datos. Para así poder terminar almacenando las imágenes, si es necesario. El código presentado en la Figura 42, corresponde a la obtención de los datos que se han almacenado en el formulario, se presenta a detalle en la Tabla 17.

1.2. Eliminar. Para esta parte de la interfaz es necesario obtener él, ID del producto, para ello seguimos haciendo uso de la variable “id_product” y con una consulta SQL de tipo DELETE(véase la Figura 43), se elimina el producto al que hacemos referencia. Como se observa en el código de la Figura 44, se obtiene él, ID del producto y con ayuda de mysqli_query se ejecuta la sentencia SQL, la cual elimina registros de la tabla “products”, en la cual, el valor de “id_product” sea igual al contenido de la variable “$id_product” la cual se obtiene de la interfaz de la tabla de productos como se muestra en la Figura 39.

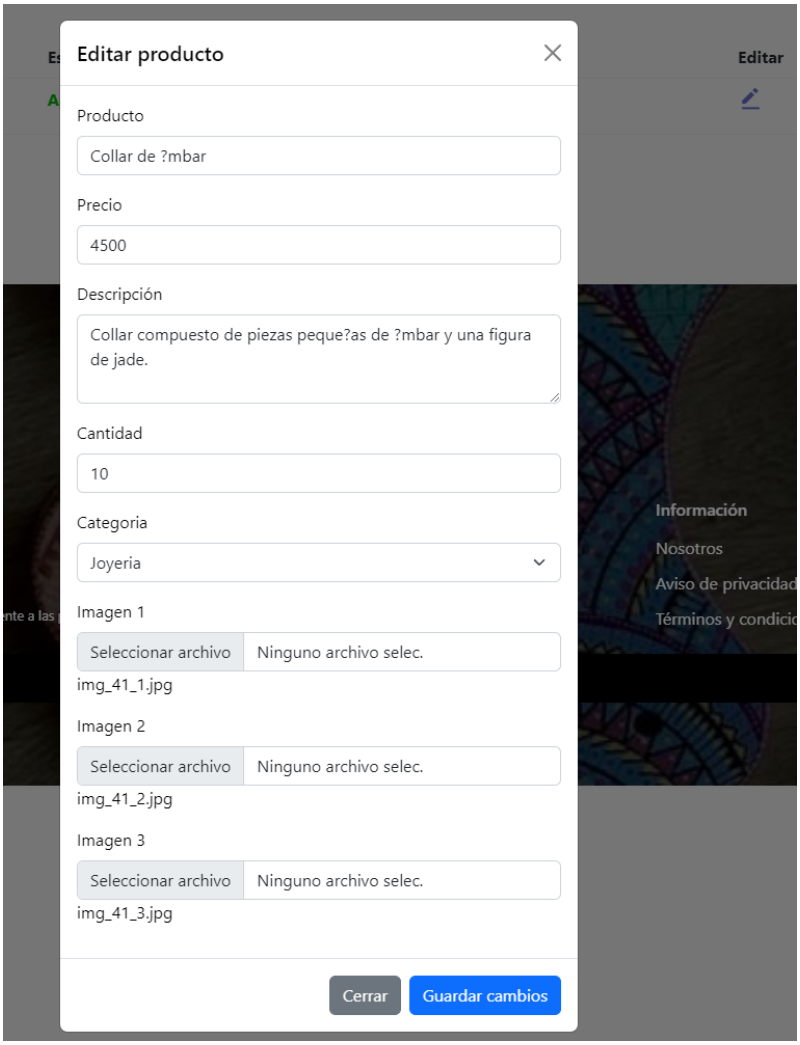


Figura 41. Formulario para editar información del producto.

```php
# Obtenemos todos los datos que mandamos por el formulario
$id_product = $_GET['id_product'];
$producto = $_POST['producto'];
$precio = $_POST['precio'];
$descripcion = $_POST['descripcion'];
$cantidad = $_POST['cantidad'];
$categoria = $_POST['categoria'];

# Editamos los atributos de la tabla con lo que obtuvimos en el formulario
mysqli_query($conn,"UPDATE products SET
    product = '$producto',
    price = '$precio',
    description = '$descripcion',
    category = '$categoria',
    stock = '$cantidad'
    WHERE id_product = $id_product");

# Las imagenes se hacen por separado, ya que son de tipo fyle.
# Si no ingresamos una imagen en el input 1 image, no hacemos algo y esto evitara eliminar la imagen
# Si esta vacio, dejarlo asi
if(empty($_FILES['imagen1']['name'])){

}else{ # Si se ingreso una imagen entonces hacer el UPDATE
    $imagen1 = $_FILES['imagen1']['name'];
    $sql = "UPDATE products SET
    image1 = '$imagen1' WHERE id_product = $id_product";

    move_uploaded_file($_FILES['imagen1']['tmp_name'], "../assets/products/".$imagen1."");
    mysqli_query($conn, $sql);
}
```

Figura 42. Código de formulario Editar

Tabla 17. Variables del formulario editar

Variable	Tipo	Descripción
$product	Obtención de datos	Con ayuda de esta variable obtenemos el dato del formulario correspondiente al nombre del producto “product”.
Mysqli_query	Ejecución de consultas SQL	Recibe 2 parámetros $conn: variable de conexión a la base de datos 2do parámetro: consulta SQL en este caso de tipo UPDATE.
Query	Consulta de tipo SQL	La query de tipo UPDATE quiero decir que se requiere actualizar los datos especificados con ayuda de la variable “id_product”, ya que, sin esa variable se actualizaría la información de un producto incorrecto.

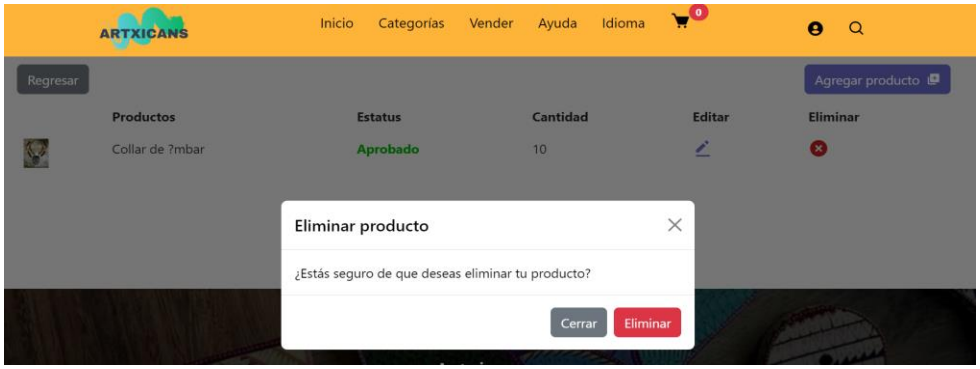


Figura 43. Eliminar producto.

```php
$id_product = $_GET['id'];
      # Eliminamos el producto que coincida con el id que se esta solicitando eliminar
      mysqli_query($conn,"DELETE FROM products WHERE id_product = '$id_product'");
```

Figura 44. Código del formulario Eliminar

1.3. **Agregar producto.** En esta interfaz de la plataforma, el usuario vendedor tendrá la capacidad de realizar el registro de nuevos productos, donde se presentará el nuevo formulario que se muestra en la Figura 45. Para completar el registro de forma exitosa, es necesario llenar todos los campos, ya que, si algún campo queda vacío (a excepción de las imágenes 2 y 3) el formulario no se podrá completar y el registro tendrá que repetirse. Los datos solicitados son los siguientes:

- Producto o nombre
- Descripción
- Categoría
- Precio
- Cantidad
- Imagen 1, 2 y 3

Para implementar el BackEnd en esta sección, se realizaron filtros para verificar que los datos ingresados en los inputs sean completos, y si algún

dato llega a faltar el registro no será completado, este proceso se llevó a cabo, de la forma como se muestra el código en la Figura 46, de esta manera, nos aseguramos de que sean ingresados todos los datos necesarios antes de realizar el registro. De igual forma que las demás partes de la interfaz del usuario vendedor, se inicia recolectando los datos que se almacenan en los inputs, para después, poder validar que los campos no estén vacíos. Esto lo podemos realizar con la ayuda de la sentencia “if” de la línea 16 como se puede observar en la Figura 46 para más detalle consulte la Tabla 18. Una vez que la validación de los datos sea exitosa y no falte ningún campo obligatorio, el servidor procederá a realizar la inserción de los datos, en la tabla “products” que se encuentra en la base de datos.

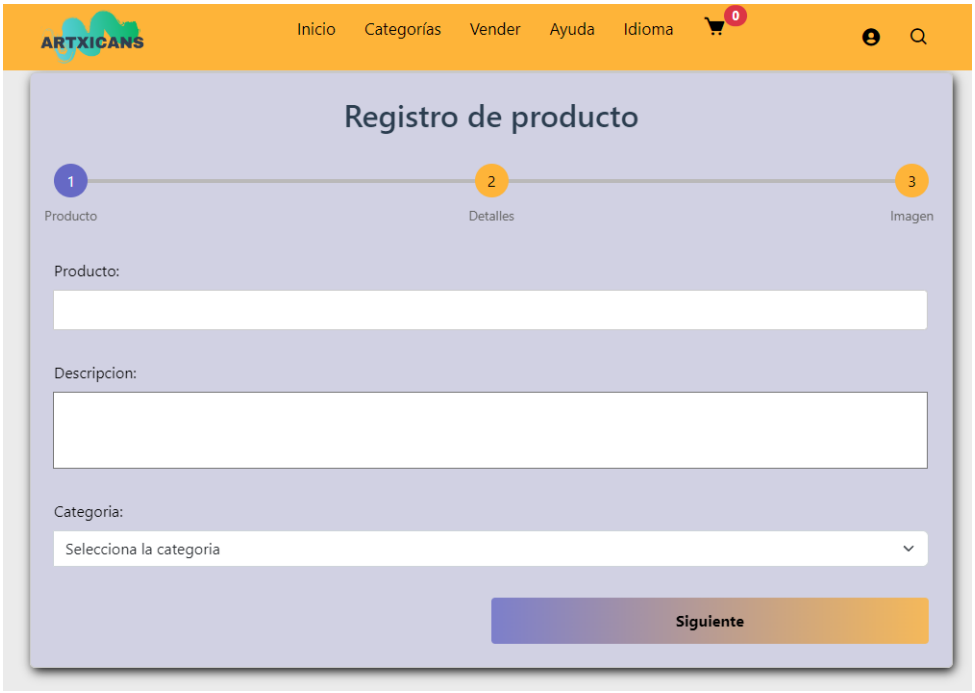


Figura 45. Formulario de Registro de producto.

```php
    # obtenemos los valores del formulario con POST
$idusuario =  $_SESSION['id'];
$producto = $_POST['producto'];
$descripcion = $_POST['descripcion'];
$categoria = $_POST['categoria'];
$precio = $_POST['precio'];
$cantidad = $_POST['cantidad'];
    # Se obtienen la direccion para almacenar las imagenes
$direccion = "./assets/products/";
    # Variables vacias para las imagenes
$image1 = "";
$image2 = "";
$image3 = "";

    # validacion de campos producto, categoria y precio, si estan vacios se salta al else
if (!empty($_POST['producto']) && !empty($_POST['categoria']) && !empty($_POST['precio']))
    {
            # validacion que el campo cantidad sea entero y no letra u otra cosa
        if (filter_var($cantidad, FILTER_VALIDATE_INT) == false)
            {
                $men .= "*La cantidad debe ser numerica.";
            }
```

Figura 46. BackEnd agregar producto.

Tabla 18. Validación de datos.

If (!empty($_POST['producto'])	
If	Estructura de control condicional.
!empty	Función para verificar si una variable no está vacía.
$_POST	Variable que obtiene los datos que se envían a la parte del servidor.
['producto']	Nombre del campo que se está verificando.

2. **Mis pedidos.** En la interfaz mostrada en la Figura 47, procedente del menú del usuario vendedor, el vendedor puede verificar los pedidos pendientes, por ejemplo:
    - Pedido pendiente.
    - Pedido en proceso de entrega.
    - Pedido enviado.
    - Pedido entregado.

    Esta parte del menú el vendedor puede realizar las acciones correspondientes al “envío” del paquete, o simplemente observar la lista de sus Pedidos pendientes (como se muestra en la Figura 47).

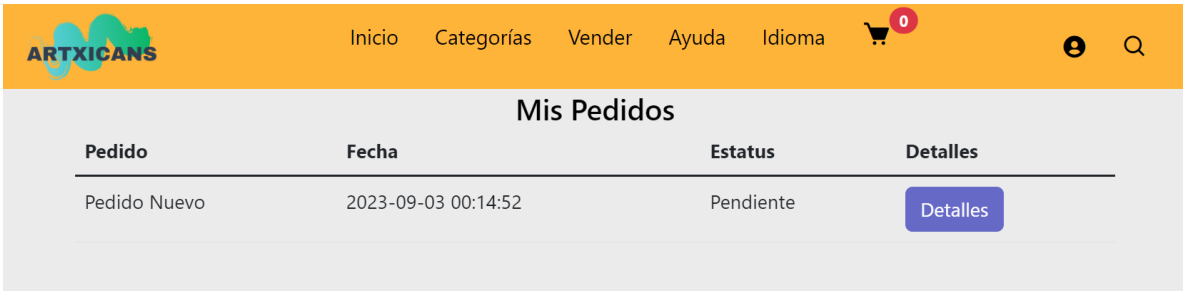


Figura 47. Mis pedidos

Esta lista muestra los pedidos tanto nuevos, como los pedidos antiguos y finalizados, cuando queramos ver los detalles del pedido tendremos que dar clic en el botón “detalles”, con el cual, el sistema mostrará otra interfaz donde se presentan los detalles tanto del producto vendido, como también, los datos del comprador como se muestra en la Figura 48.

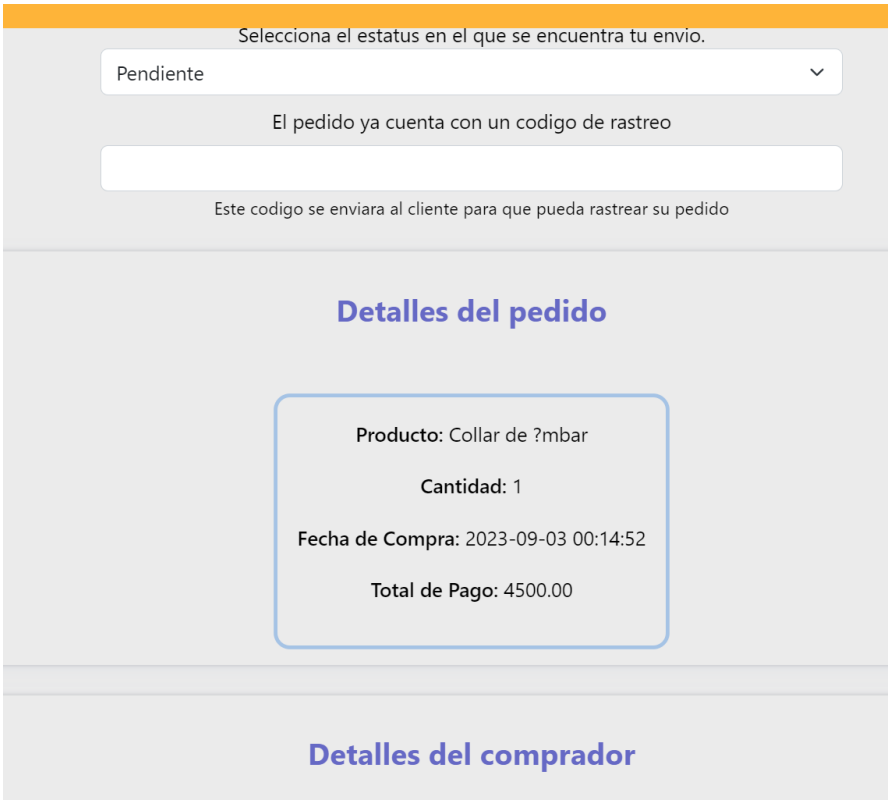


Figura 48. Detalles de pedido.

Después de dar clic en el botón, se presentan los detalles del estado del envío del paquete, mostrará la opción para poder ingresar un código de rastreo, como también, los detalles de la venta y del comprador.

Nota: esta parte se logró con ayuda del archivo llamado “Ssingle-order.php”, que contiene la porción de código, mostrado en la

Figura **49** y Figura 50.

En primero lugar hablaremos del estatus del envío, en este caso existen 3 tipos los cuales son:

1. Pendiente
2. Enviado
3. Entregado

Cada caso envía un mensaje de notificación diferente al usuario comprador, los mensajes son los siguientes (Tabla 19):

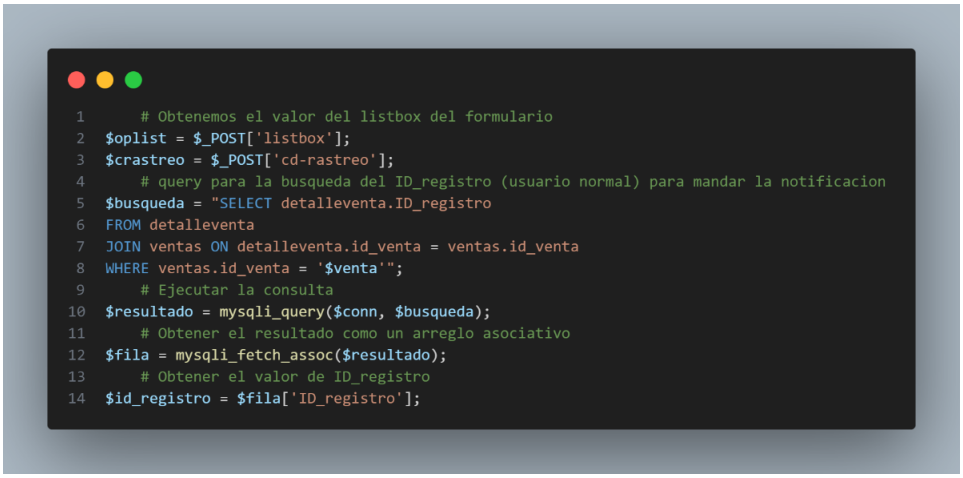
Tabla 19. Casos de estatus de envío.

Caso	mensaje
Pendiente	El paquete está siendo preparado para proceso de envío.
Enviado	El paquete ha sido enviado, tu ID de rastreo:
Entregado	El paquete ha sido entregado al domicilio registrado.

Estos mensajes se envían de la siguiente manera.

Como se muestra el código en la

Figura **49**, en primer lugar, se obtendrán los valores tanto del listbox y el input con ayuda de las variables “$oplist” y “crastreo”.



```
    # Obtenemos el valor del listbox del formulario
$oplist = $_POST['listbox'];
$crastreo = $_POST['cd-rastreo'];
    # query para la busqueda del ID_registro (usuario normal) para mandar la notificacion
$busqueda = "SELECT detalleventa.ID_registro
FROM detalleventa
JOIN ventas ON detalleventa.id_venta = ventas.id_venta
WHERE ventas.id_venta = '$venta'";
    # Ejecutar la consulta
$resultado = mysqli_query($conn, $busqueda);
    # Obtener el resultado como un arreglo asociativo
$fila = mysqli_fetch_assoc($resultado);
    # Obtener el valor de ID_registro
$id_registro = $fila['ID_registro'];
```

Figura 49. BackEnd estado de envío.

Después tenemos una consulta de búsqueda donde:

- Se selecciona el “ID_registro” de la tabla “detalleventa”.
- Se Utiliza la cláusula JOIN para unir las tablas “detalleventa” y “ventas”, donde, usaremos “id_venta” como índice para la unión
- Por último, realizamos la condición, el valor de la columna "id_venta" en la tabla "ventas" debe ser igual al valor almacenado en la variable "$venta".

Al finalizar la consulta también agregamos las variables de ejecución que hemos estado utilizando.

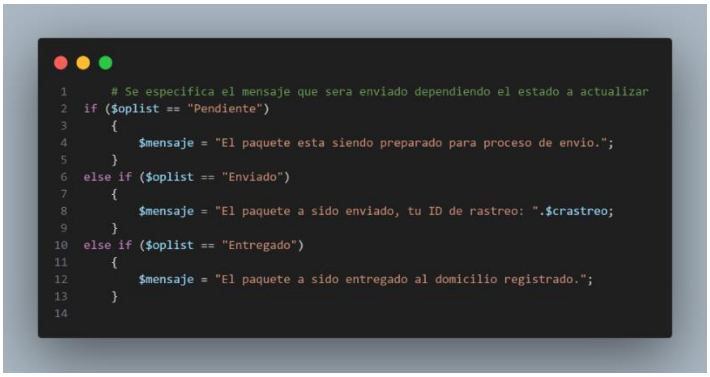
Y los resultados obtenidos se almacenan en un arreglo, que posteriormente se guarda en la variable "$fila"

En la última línea de la

Figura **49**, se obtiene él, ID del usuario, al cual se le mandara la notificación del estatus del pedido, con la ayuda de la variable “$id_registro”, la cual se iguala al valor obtenido al acceder al arreglo resultante del índice “ID_registro” de la tabla en la base de datos.

La especificación del mensaje la obtendremos con la ayuda de la variable “$oplist” de la siguiente manera:

Como se puede apreciar en la Figura 50 que solo es una sentencia “if” que dependiendo el valor obtenido con la ayuda de la variable “$oplist” ingresa el mensaje específico correspondiente a cada opción.



```
    # Se especifica el mensaje que sera enviado dependiendo el estado a actualizar
if ($oplist == "Pendiente")
    {
        $mensaje = "El paquete esta siendo preparado para proceso de envio.";
    }
else if ($oplist == "Enviado")
    {
        $mensaje = "El paquete a sido enviado, tu ID de rastreo: ".$crastreo;
    }
else if ($oplist == "Entregado")
    {
        $mensaje = "El paquete a sido entregado al domicilio registrado.";
    }

```

Figura 50. Especificación de mensaje para notificación.

La asignación del valor de la clave de rastreo se lleva a cabo con el siguiente proceso, nótese que el input se bloque cuando se detecta que ya se ha ingresado una clave de rastreo, lo que significa que no es posible modificarla una vez registrada.

En este caso se realiza la siguiente instrucción de tipo actualización o UPDATE en código SQL como se muestra en la Figura 51. Esto significa:

1. Se selecciona la tabla a actualizar en este caso “ventas”
2. Establecemos el campo a actualizar en este caso “crastreo”, en la tabla "ventas". El valor de "$crastreo" se asigna a la columna "crastreo" en los registros que cumplan con la condición especificada en la condición "WHERE".
3. La condición es que el valor de la columna "id_venta" en la tabla "ventas" debe ser igual al valor almacenado en la variable "$venta".

Y para ejecutar la consulta hacemos uso de la variable “$ result”.

Figura 51. Asignación de código de rastreo.

Para concluir la explicación de esta sección. Cuando se han completado los datos tanto de envío como el código de rastreo, solo se necesita dar clic en el botón “guardar”, para almacenar y actualizar la información ingresada.

3. **Mi perfil.** Esta parte es un poco extensa solo destacaremos las partes más importantes y necesarias de la interfaz mostrada en la Figura 52.

   En la interfaz Mi perfil, se presentará toda la información del usuario de tipo vendedor en este caso se mostrarán los siguientes puntos:

   1. Banner con nombre de usuario.
   2. Información importante del vendedor.
   3. Productos del vendedor.

4. Puntuación y cada de comentarios para el perfil del vendedor.

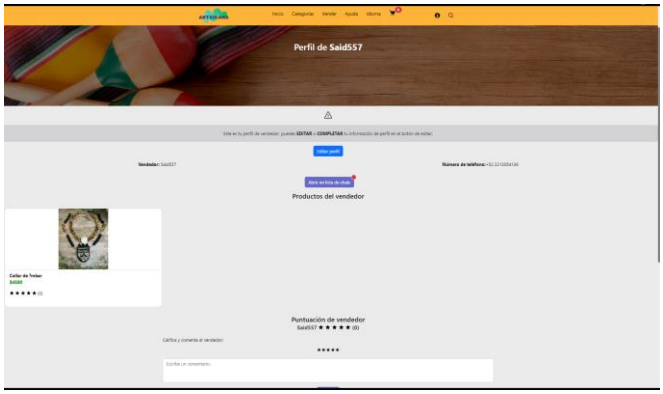


Figura 52. Interfaz perfil de vendedor.

Entre los puntos 1 y 2 se mostrará el botón “Editar perfil”, el cual, solo será visible cuando el usuario inicie sesión en su perfil de vendedor.

Cuando el usuario da clic en el botón “Editar perfil”, el vendedor tendrá la posibilidad de actualizar la información que proporciono al inicio de su registro (4.5.5 vender), solo que en este caso solo podrán ser editados los siguientes datos:

- Identificador o nickname
- Lada
- Teléfono

También se brinda la posibilidad de ingresar algunas descripciones, como lo son:

- Descripción acerca de ti y tu tiempo trabajando.
- Descripción acerca tus artesanías.
- Descripción sobre tú donde se encuentra tu lugar de trabajo.

Para concluir esta sección, una vez que se cuenten con los datos a actualizar, se deberá dar clic en el botón “Aceptar”, para así, poder enviar y guardar la información, como se muestra en la Figura 53.

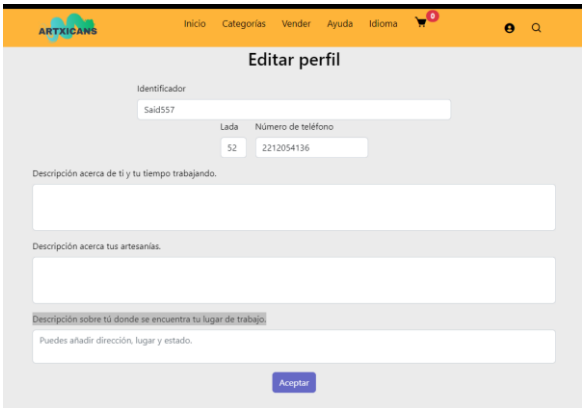


Figura 53. Interfaz Editar perfil.

4. **Mi cuenta de pagos.** En esta sección del menú del usuario vendedor, se brinda la oportunidad de ingresar su token para pagos que ofrece paypal, en esta parte de la interfaz, tenemos un formulario que consta de los siguientes componentes:

    - Input que almacena el token (string)
    - Botón Guardar
    - Botón Regresar

    Para que esta página funcione correctamente, se creó el archivo “Spay-account.php”, el código se muestra en la Figura 54

    En este archivo, para comenzar, se verifica en la base de datos, en la tabla “pay_account”, si el vendedor ya tiene un token guardado. Si este ya existe, se mostrará y se dará la opción de actualizarlo, en caso de que no exista un token, se habilitará la opción de guardar el token ingresado por primera vez. Este proceso se lleva a cabo como se muestra en el código de la Figura 54.

```php
# Consulta para saber si el usuario ya cuenta con un token
$query = mysqli_query($conn,"SELECT * FROM pay_account WHERE ID_registro = $idusuario");
$data = mysqli_fetch_array($query);
```

Figura 54. Actualizar/Guardar token de pago.

El formulario se presenta tal como se muestra en la Figura 55, en este formulario, como se menciona anteriormente, se incluye un campo de entrada de texto para ingresar el token de PayPal, junto con los botones de "Regresar" y "Guardar".

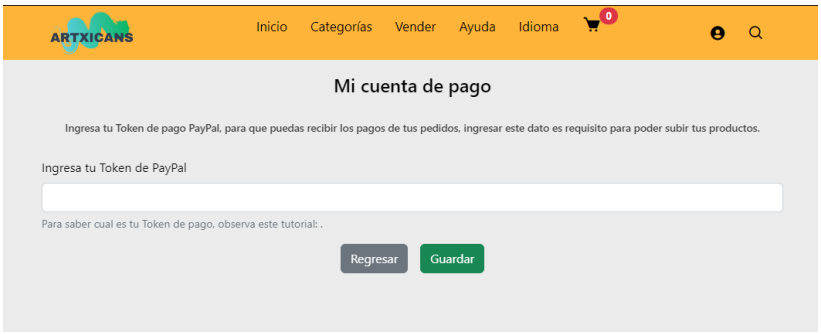


Figura 55. Formulario de Mi cuenta de pago.

4.5.6 Ayuda

En esta sección de la interfaz no hay mucho que comentar, sin embargo, es importante aclarar lo siguiente.

El menú de ayuda presenta las “Preguntas frecuentes” o “FAQ” (*Frequently Asked Questions*), donde se proporcionan algunas preguntas que muestran información o instrucciones más detalladas para realizar algunas tareas correspondientes a la plataforma (Figura 56).

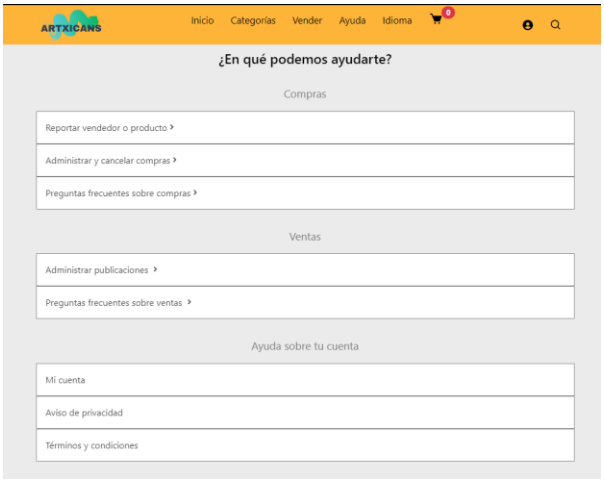


Figura 56. FAQ.

4.5.7 Idioma

En esta sección de la plataforma hicimos uso de una API de GCT (*Google Cloud Translation*) para permitir al usuario seleccionar el idioma deseado como se muestra en la Figura 57.

En esta ocasión, se ha habilitado únicamente la opción de cambio de idioma a inglés. Para integrar la API, fue necesario descargar los archivos necesarios y luego incorporarlos en los archivos principales de la plataforma, como se muestra en la Figura 58.

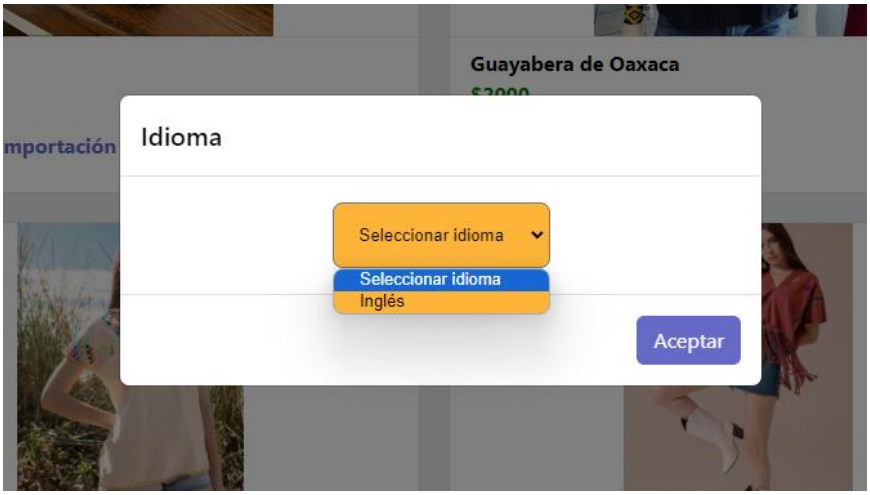


Figura 57. Menú de cambio de idioma.

```
/* Funcion para API de Google Translate, este codigo nos lo proposciona la API de Google*/
function googleTranslateElementInit() {
    new google.translate.TranslateElement({
        /* Idioma inicial */
      pageLanguage: 'es', layout:
      google.translate.TranslateElement.InlineLayout.VERTICAL, autoDisplay:
      /* en el included solo agregamos idioma ingles para traducir*/
      false, includedLanguages: 'en,es', gaTrack: true, gaId: 'YOUR_API_KEY'
      }, 'traducir');
 }
```

Figura 58. API Google Cloud Translation.

4.5.8 Carrito de compras

Esta sección de la plataforma será dividida en 2 partes como se detalla a continuación.

1 Agregar productos al carrito.
2 Ver y editar el carrito de compras.

1. **Agregar productos al carrito.** Para esta parte de la plataforma, se utiliza el archivo llamado “cart.php”, el cual, con ayuda del archivo llamado “conexión.php”, que describimos en el punto 4.5.1 conexión a base de datos, realizamos la recolección de los datos de los productos, en este caso encriptados, de esta forma damos un poco más de seguridad a la plataforma, ya que, para poder desencriptar estos datos hay que tener las credenciales necesarias.

   En todo caso de que traten de modificar los datos por otras formas les será un poco más difícil.

   Todo esto se logra después de dar clic al botón de “Comprar” que se muestra en la plataforma o las descripciones del producto como se muestra la interfaz de la Figura 59.

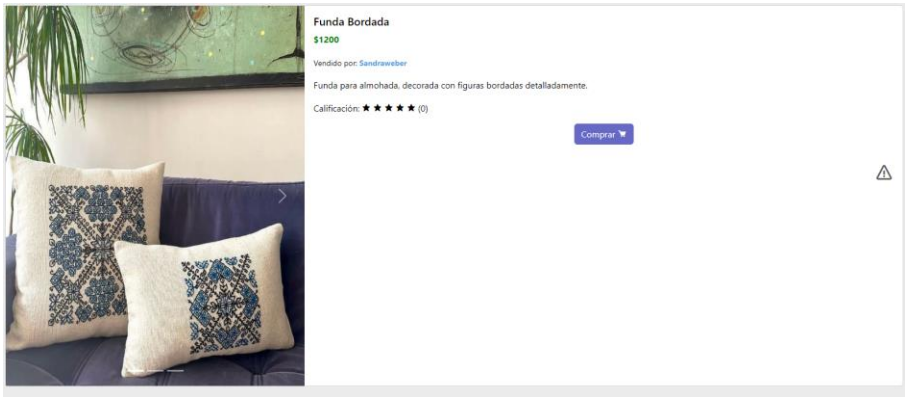


Figura 59. Botón comprar.

En este caso, los pasos de parámetros se logran al incluir inputs de forma oculta (hidden) al formulario que muestra los productos, como se puede

observar en la Figura 59, en estos inputs se almacena y se logra el envío de los datos específicos de cada producto como:

- Nombre
- Costo
- Imagen
- Id del producto
- Cantidad

Todos los datos listados anteriormente se almacenan y se envían encriptados directamente. Estos parámetros se envían al dar clic en el botón “Comprar”, realizando de primera instancia la desencriptación para poder trabajar con la información. En este caso desencriptamos los datos como se muestra el código en la Figura 60, con la ayuda de la función openssl_decrypt, la cual describiremos en la
Tabla *20*.

```php
if (is_numeric(openssl_decrypt($_POST['idproduct'],COD,KEY)))       # validacion de los datos del formulario del carrito id del producto
    {
        $idprod = openssl_decrypt($_POST['idproduct'],COD,KEY);
        #$mensaje .= "id correcto".$idprod;
    }
else
    {
        #$mensaje .= "id incorrecto".$idprod;
        break;
    }

if (is_string(openssl_decrypt($_POST['product'],COD,KEY)))          # validacion de los datos del formulario del carrito nombre del producto
    {
        $prod = openssl_decrypt($_POST['product'],COD,KEY);
        #$mensaje .= "producto correcto".$prod;

```

Figura 60. Desencriptación de datos.

Tabla 20. Función de desencriptación.

openssl_decrypt(1,2,3)	
Parámetro 1	Variable que contiene el código a desencriptar
Parámetro 2	Variable que contiene el tipo de encriptación
Parámetro 3	Variable que contiene la llave de desencriptación

Una vez desencriptado los datos, ahora pasamos a ingresarlos a la variable de carrito la cual llamamos “cart”, dicha variable es de tipo “SESSION” si la variable es vacía los elementos se almacenarán desde la posición 1, ya que, esta variable es de tipo array.

En dado caso que ya se encuentren ítems en el carrito entonces se almacenaran en la posición adecuada para el correcto funcionamiento de esta parte de la plataforma, todo esto se logra de la siguiente manera:

Como se logra observar en la Figura 61, tenemos la validación de que si la variable “cart” se encuentra vacía entonces haremos uso del array “$product” en el cual se almacenan el nombre de la imagen, el producto, id del producto, precio y la cantidad.

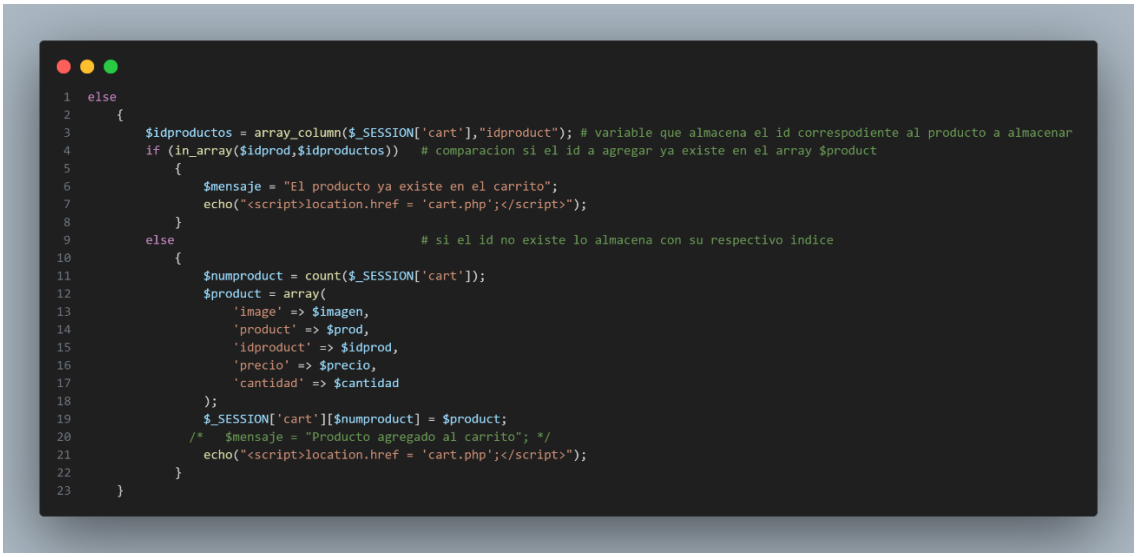
```php
if (!isset($_SESSION['cart']))          # si el carrito esta vacio
    {
        $product = array(               # array que almacena los datos del producto
            'image' => $imagen,
            'product' => $prod,
            'idproduct' => $idprod,
            'precio' => $precio,
            'cantidad' => $cantidad
        );
        $_SESSION['cart'][0] = $product;    # valida si la posicion del elemento a agregar es el 0 o incrementa
        $mensaje = "Producto agregado al carrito";
        echo("<script>location.href = 'cart.php';</script>");
    }
```

Figura 61. Inserción de ítems al carrito vacío.

En este caso si la posición del carrito es igual a cero entonces se almacenan los datos en esa misma posición (cero).

En el caso de que ya existan ítems o productos en el carrito entonces realizamos la validación de las posiciones, para seguir insertando productos como se muestra el código en la Figura 62).

En esta parte del código se creó el array llamado “$idproductos” (consulte el código en la Figura 62) con ayuda de la función array_column (descripción completa en la Tabla 21):



```
else
    {
        $idproductos = array_column($_SESSION['cart'],"idproduct"); # variable que almacena el id correspodiente al producto a almacenar
        if (in_array($idprod,$idproductos))   # comparacion si el id a agregar ya existe en el array $product
            {
                $mensaje = "El producto ya existe en el carrito";
                echo("<script>location.href = 'cart.php';</script>");
            }
        else                                  # si el id no existe lo almacena con su respectivo indice
            {
                $numproduct = count($_SESSION['cart']);
                $product = array(
                    'image' => $imagen,
                    'product' => $prod,
                    'idproduct' => $idprod,
                    'precio' => $precio,
                    'cantidad' => $cantidad
                );
                $_SESSION['cart'][$numproduct] = $product;
               /*   $mensaje = "Producto agregado al carrito"; */
                echo("<script>location.href = 'cart.php';</script>");
            }
    }
```

Figura 62. Insertar productos al carrito ya con ítems.

Tabla 21. Descripción Array_column.

array_column(1,2);	
Parámetro 1	Array que almacena los productos seleccionados
Parámetro 2	Parámetro que define el campo o columna en este caso “idproduct”

Después con la ayuda de la función in_array verificamos que él, ID del producto seleccionado ya exista en el array “$idproductos”, se explica la función en la siguiente tabla (descripción completa en la Tabla 22):

Tabla 22. Descripción de in_array.

in_array(1,2)	
Parámetro 1	Variable o dato por buscar
Parámetro 2	Lugar en donde buscar (Array)

En este caso como es verificación, si él, ID del producto ya existe en el carrito, dicho producto ya no se agregará de nuevo al carrito, y en caso contrario si será ingresado al carrito.

Esto se logra con un conteo de los productos en el carrito “$SESSION[‘cart’]” y almacenando dicho conteo en la variable “$numproduct”, después de esto se sigue el mismo método de almacenar los datos en un array llamado “$product” y almacenándolo en el carrito, pero en este caso con la posición indicada por la variable de conteo. Como se muestra en la línea 9 – 20 del código ilustrado en la Figura 62.

2. **Ver y editar el carrito de compras.** Esta sección de la explicación se dividirá en 5 partes, como se detalla a continuación:
    1 Actualizar cantidad
    2 Eliminar
    3 Ingresar dirección
    4 Ver dirección y editar
    5 Realizar compra

Llevaremos la explicación con forme a la lista, para que así, al finalizar lleguemos al botón de realizar compra sin perder alguna opción de vista (se muestra la interfaz en la Figura 63).

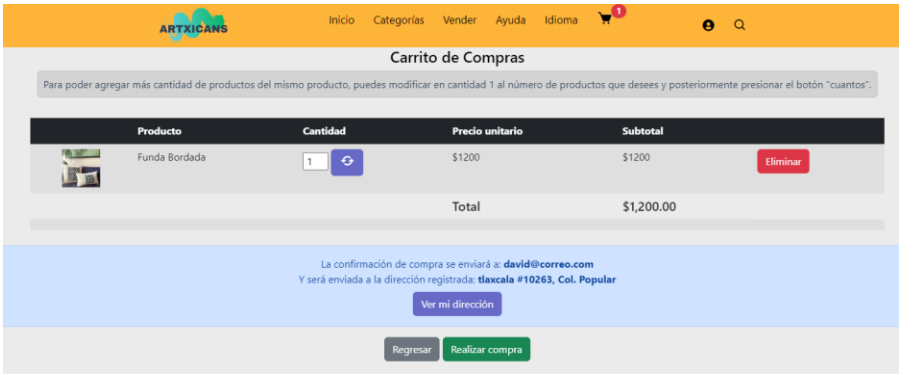
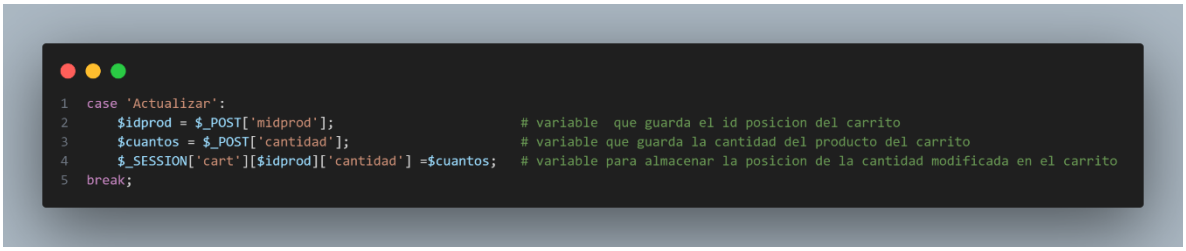


Figura 63. Interfaz Carrito de compras.

2.1 Actualizar cantidad. En esta parte de la interfaz como se muestra en la Figura 63 tenemos el carrito de compra con sus diferentes opciones, en este caso se explicará el botón de actualizar cantidad, no tiene gran ciencia, pero no es menos importante por ello. Este botón existe para la tarea de actualizar la cantidad de los productos que el usuario quiera comprar en este caso como se muestra en la Figura 59 el usuario comprara el producto denominado “Funda Bordada” pero en este caso

digamos que el usuario quiere comprar 3, Para ello hemos ingresado un input y un botón de actualizar para poder lograr esa función, sin que el usuario tenga que dar clic al botón comprar en la interfaz de descripción de producto. El código correspondiente para esta parte se presenta en la Figura 64.

En este caso se obtiene el valor tanto del ID del producto seleccionado, como de la cantidad que se almacena en el input, y al final se actualiza la cantidad que se muestra en el carrito de compras (descripción completa en la Tabla 23).



```
case 'Actualizar':
    $idprod = $_POST['midprod'];                    # variable  que guarda el id posicion del carrito
    $cuantos = $_POST['cantidad'];                  # variable que guarda la cantidad del producto del carrito
    $_SESSION['cart'][$idprod]['cantidad'] =$cuantos;   # variable para almacenar la posicion de la cantidad modificada en el carrito
break;
```

Figura 64. Botón actualizar cantidad.

Tabla 23. Actualizar cantidad en carrito de compras.

$_SESSION['cart'][$idprod]['cantidad'] =$cuantos;	
$_SESSION['cart']	Accedemos al carrito de compras almacenado en la variable del mismo nombre.
[$idprod]	Índice de búsqueda dentro del carrito
['cantidad']	Propiedad cantidad del producto seleccionado en la interfaz del carrito
$cuantos	Asignación del nuevo valor al producto del carrito.

Una vez ajustada la cantidad del producto tanto el subtotal como el total se ajustarán al instante.

2.2 Eliminar. Para poder eliminar un elemento de la interfaz, del carrito de compras en necesario cumplir con algunas instrucciones antes de realizar la eliminación del producto. Primero debemos de desencriptar él, ID del producto como se muestra en el apartado 4.5.8.1 en la Tabla 20. Después de esto se realiza una búsqueda completa en la variable multidimensional "$SESSION['cart']", para ello se hace uso de un foreach, cada elemento se asigna a la variable "product" para así poder acceder a cada elemento, como se muestra en el código presentado en la Figura 65. Después se valida, si el elemento a eliminar se encuentra en el carrito, si es el caso procedemos a eliminarlo del mismo, con ayuda de la función unset(), el cual elimina el producto con su respectivo índice.

```php
if (is_numeric(openssl_decrypt($_POST['idproduct'],COD,KEY)))
    {
        $idprod = openssl_decrypt($_POST['idproduct'],COD,KEY);
        foreach ($_SESSION['cart'] as $indice => $product)
            {
                if ($product['idproduct'] == $idprod)
                    {
                        unset($_SESSION['cart'][$indice]);
                        echo "<script> alert('elemento borrado'):</script>";

                    }
            }
    }
```

Figura 65. Eliminar producto.

2.3 Ingresar dirección. Esta parte se realizará automáticamente la primera vez que un usuario realice una compra, y este, no haya realizado el registro de su dirección (esta información se puede consultar en la interfaz de carrito como se muestra en la Figura 63). El cual, muestra un nuevo formulario, que solicitara capturar los datos relevantes a su dirección. Cuando el usuario proporcione la información completa, correspondiente a su dirección, la plataforma mostrará la dirección almacenada, al comento de comprar un producto como se muestra en la Figura 66.

Figura 66. Muestra de dirección.

2.4 Ver dirección y editar. Al dar clic en el botón editar en el formulario que muestra los datos de dirección ya almacenados, mandara a otro formulario, en el cual, se podrá ver y editar la dirección ya existente, ya sea, actualizando los datos o ingresando una nueva dirección (como se presenta en la Figura 67 y Figura 68).

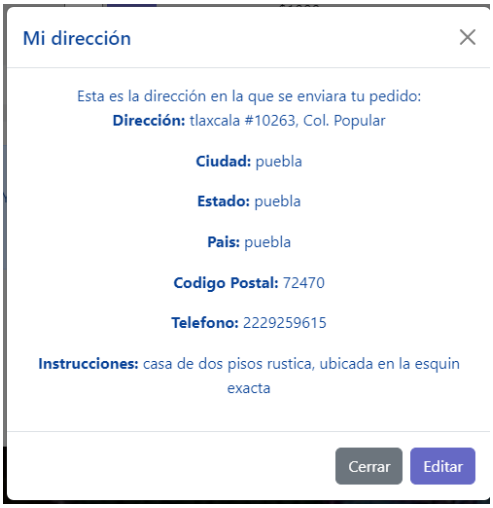


Figura 67. Dirección almacenada.

Figura 68. Formulario de actualizar datos.

2.5 Realizar compra. Para finalizar este apartado de explicación de código, procedemos a realizar la compra de un producto, para ello se creó el archivo “pago.php” el cual con ayuda de la API de paypal (se muestra el botón de pago en la Figura 69), obtenemos las claves de transacción y los datos que paypal manda para poder verificar una compra.

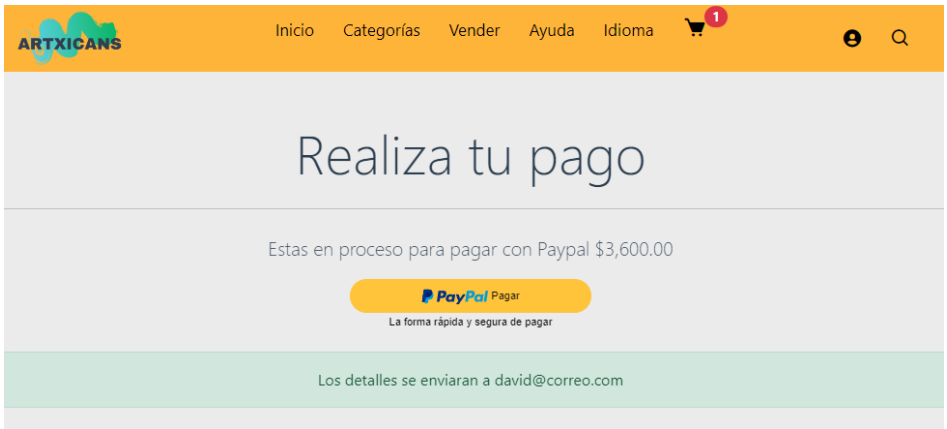


Figura 69. Realizar compra API paypal.

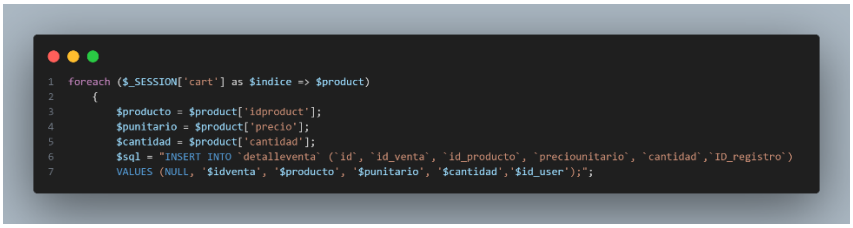
Para ello fue necesario realizar los siguientes pasos:

2.5.1 Calcular el total de la compra. En esta parte se hizo uso de un foreach para poder acceder a los datos almacenados en la variable “$_SESSION['cart']”, donde, realizamos una multiplicación simple (el código correspondiente se presenta en la Figura 70), donde se suma el valor de la variable “&total” con el producto del precio del producto y la cantidad.

```php
foreach ($_SESSION['cart'] as $indice => $product)
    {
        $total = $total + ($product['precio'] * $product['cantidad']);
    }
```

Figura 70. Total, de la compra.

2.5.2 Almacenar detalles del producto. En esta sección se hizo uso de un foreach para poder acceder a los datos almacenados en la variable “$_SESSION['cart']”, donde, realizamos una consulta de tipo “INSERT” en el lenguaje SQL, donde, los datos se almacenan en la tabla “detalleventa” (el código correspondiente se presenta en la Figura 71).



```php
foreach ($_SESSION['cart'] as $indice => $product)
    {
        $producto = $product['idproduct'];
        $punitario = $product['precio'];
        $cantidad = $product['cantidad'];
        $sql = "INSERT INTO `detalleventa` (`id`, `id_venta`, `id_producto`, `preciounitario`, `cantidad`,`ID_registro`)
        VALUES (NULL, '$idventa', '$producto', '$punitario', '$cantidad','$id_user');";
```

Figura 71. Almacenar los detalles de la venta.

Una vez que la API de payla haya finalizado su proceso exitosamente, el sistema redireccionará a la interfaz de “verificar.php”, la cual, mostrará que el pago se ha realizado con éxito, y el usuario podrá acceder a su compra para poder ver los detalles, la interfaz de pago exitoso se muestra en la Figura 72.

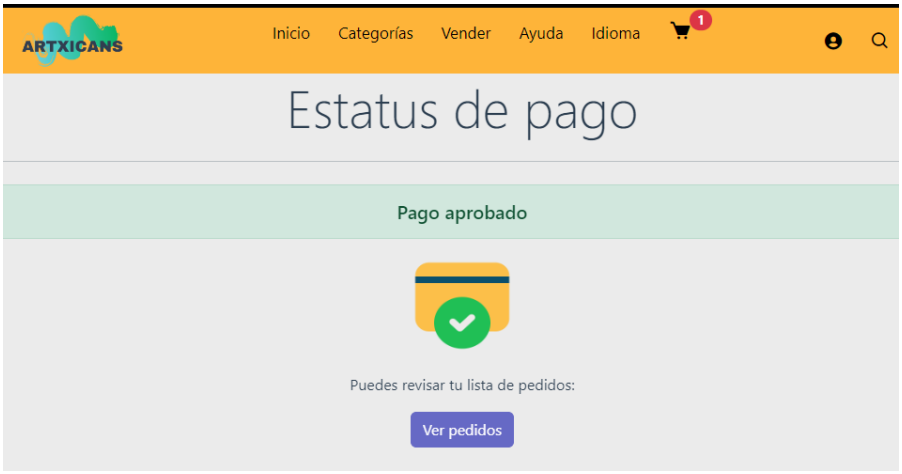


Figura 72. Pago exitoso.

4.5.9 Administrador

ahora toca la parte de explicar el funcionamiento del panel de administrador el cual se compone de las siguientes partes:

1. Login
2. Inicio
3. Pedidos
4. Productos
5. Vendedores
6. Reportes
7. Nuevos vendedores
8. Nuevos productos

Los puntos listados anteriormente se explicarán con forme avancemos en los apartados.

Cabe mencionar que algunas partes del menú del administrador son solo de visualizar información, ya que, por motivos obvios no es ético que sea posible modificar o visualizar información.

1. **Login.** Para realizar el login fue necesario hacer otro archivo diferente, aunque, similar al de la plataforma normal (la interfaz de inicio de sesión de administrador se muestra en la Figura 73), ya que, para hacer válido el acceso es necesario realizar una consulta, a la base de datos mostrada en el apartado Diagrama de base de datos en la Tabla 14, sin embargo, esta consulta necesita unas modificaciones para que no cualquiera pueda acceder a estos datos.

   Primero que nada, necesitamos obtener los datos que se ingresan en los inputs, para ello, es necesario el siguiente código (el código correspondiente se presenta en la Figura 74)

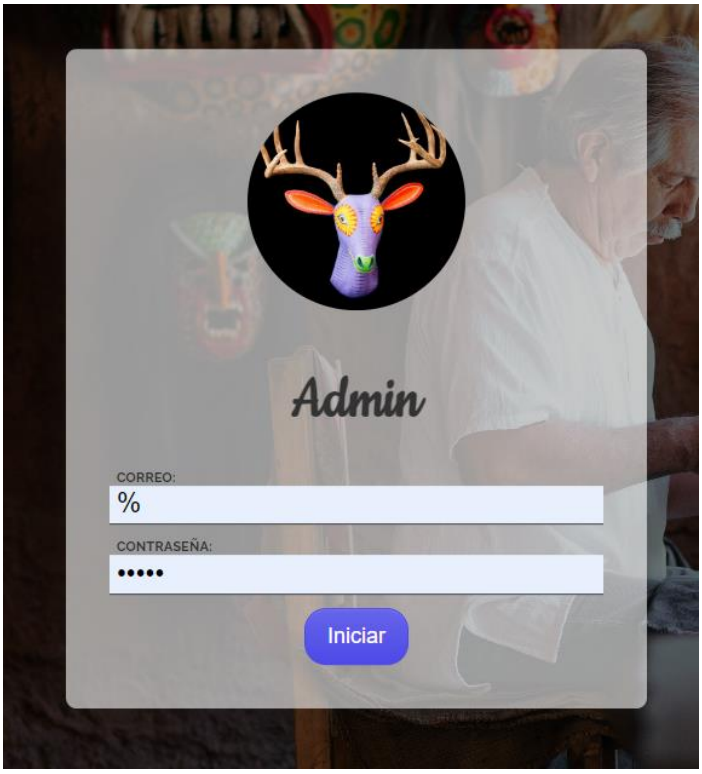


Figura 73. Formulario de inicio de sesión administrador.

```php
$email = $_REQUEST['email']??'';        # variables obtencion de email del input html
$password = $_REQUEST['pass']??'';  # variables obtencion de password del input html
$sql = "SELECT * FROM admins WHERE correo='".$email."' and contraseña='".$password."' "; # query para obtener la busqueda de los campos
$result = mysqli_query($conn,$sql);   # se obtiene el resultado de la query
```

Figura 74. Obtención de datos y quería de búsqueda.

Obtenemos el correo y la contraseña, para después, realizar la consulta para realizar la búsqueda de los datos en la tabla.

Una vez realizada la consulta, verificamos con ayuda de la variable “$result” si obtuvimos algún dato, procedemos a realizar otra consulta, para así, cerciorarnos que los datos son correctos.

Como podemos observar en código presentado en la Figura 75, validamos con la ayuda de la variable “$result”, con la cual, realizamos otra consulta en este caso parametrizada, ya que, esto ayuda a la seguridad del inicio de sesión, en este caso (descripción completa en la Tabla 24).

```php
if ($result->num_rows > 0 )  # se obtiene el valor de $result
            {
                $sql = "SELECT id_admin FROM admins WHERE correo = ?";
                $stmt = $conn->prepare($sql);
                    # "s" indica que se espera un parámetro tipo cadena (string)
                $stmt->bind_param("s", $email);
                $stmt->execute();
                    # Obtener el resultado de la consulta
                $result = $stmt->get_result();
                $admin = $result->fetch_assoc();
                if ($admin)
                    {
                        $id_admin = $admin['id_admin'];
                        $_SESSION['id_admin'] = $id_admin;
                            #se cierra la sentencia y la conexcion a la bd
                        $stmt->close();
                        $conn->close();
                        echo("<script>location.href = 'menu.php';</script>");
                        exit();
                    }
            }
```

Figura 75. Validación de inicio de sesión.

Tabla 24. query para inicio de sesión.

SELECT id_admin FROM admins WHERE correo = ?	
Id_admin	Seleccionamos el id_admin de los datos obtenidos
admin	De la tabla admin
Correo = ?	Consulta parametrizada para usar como marcador y pasar los parámetros de correo más delante de la consulta

Para finalizar si con la ayuda de la variable “$admin”, contiene datos o es verdadera entonces procederá a encolar él, ID del administrador asociado al correo que se obtuvo con las anteriores consultas, a un arreglo para iniciar sesión. Y redireccionará a la página de inicio del panel de administrador.

2. **Inicio.** Como podemos observar en la interfaz de inicio mostrada en la Figura 76, esta sección del panel del administrador solo es de visualizar un conteo rápido de todas y cada una de las partes listadas en el inicio de este apartado, mostrando los siguientes puntos:
    - Conteo de los nuevos pedidos realizados.
    - Conteo de los artículos en venta.
    - Conteo de los vendedores registrados.

- Conteo de los reportes en general.
- Conteo de las solicitudes de vendedores.
- Conteo de las solicitudes de nuevos artículos.

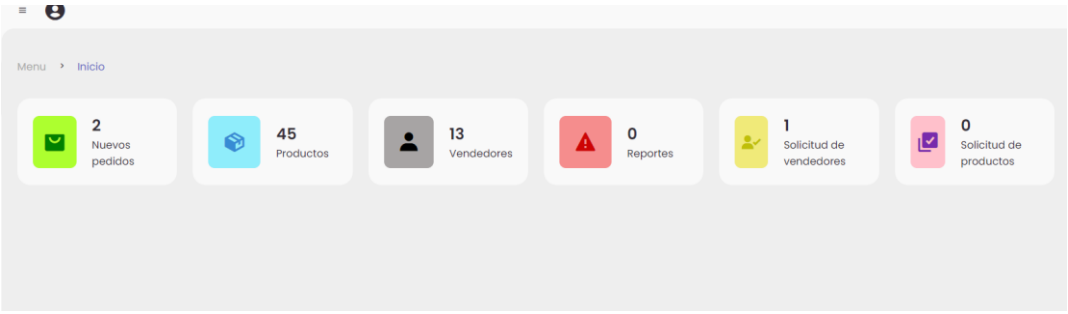


Figura 76. Inicio de panel de administrador.

3. **Pedidos.** Como podemos observar en la interfaz de la Figura 77, en esta parte de la plataforma, solo es de tipo visualización en este caso podemos ver el listado de todos y cada uno de los pedidos en proceso. Como también podemos ver los detalles de cada venta, eso dando clic en el botón “ver info” (interfaz de detalle de pedido mostrado en la Figura 78). Cuando se dé clic al botón, se mostrará la interfaz (como se muestra en la Figura 78), donde se muestra la siguiente información:
    - fecha y la hora de la compra
    - nombre del comprador y su correo
    - detalles del producto como nombre y precio total

Figura 77. Lista de pedidos en panel de administrador.

Figura 78. Ver detalle de la venta.

4. **Productos.** Como podemos observar en la interfaz mostrada en la Figura 79, en esta parte de la plataforma, solo es de tipo visualización en este caso podemos ver el listado de todos y cada uno de los productos que se encuentran activos en el catálogo.

   Podemos ver solo el listado o también ingresar a los detalles de cada artículo, dando clic en el botón “ver info”

   En la interfaz mostrada en la Figura 80, podemos observar los siguientes detalles:

   - imágenes correspondientes al producto seleccionado
   - nombre del producto
   - precio del producto
   - descripción del producto
   - categoría correspondiente al producto

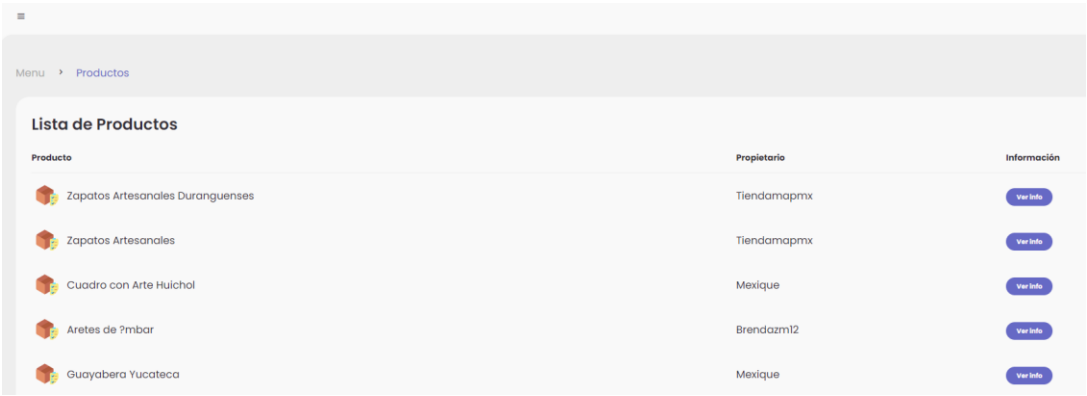


Figura 79. Interfaz de lista de productos administrador.

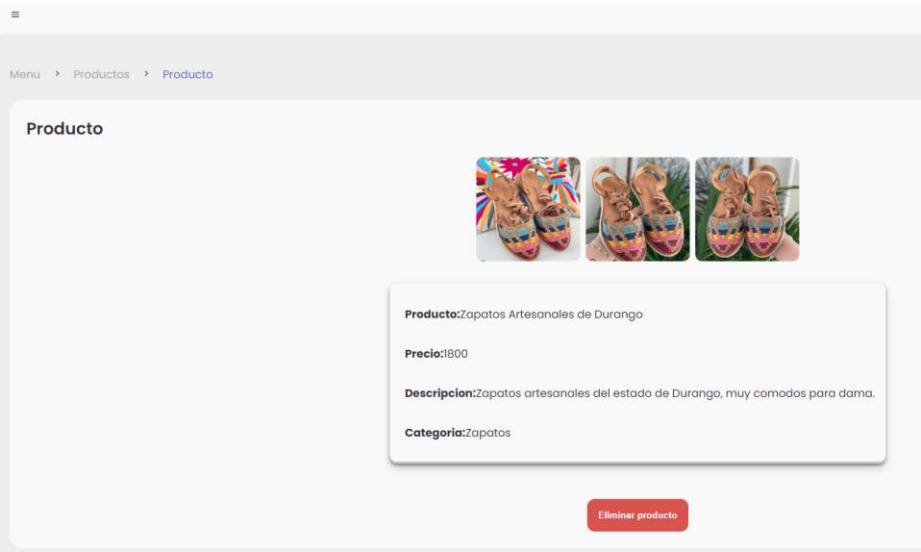


Figura 80. Detalle de producto panel de administrador.

Cabe mencionar, que por motivos de ser panel de administrador también cuenta con el botón eliminar, ya que por motivos de seguridad de ser necesaria la eliminación de algún producto. Al instante el administrador la puede realizar.

5. **Vendedores.** Como podemos observar en la interfaz de lista de vendedores presentada en la Figura 81, en esta parte de la plataforma, solo es de tipo visualización en este caso podemos ver el listado de todos y cada uno de los usuarios de tipo vendedor que se encuentran activos.

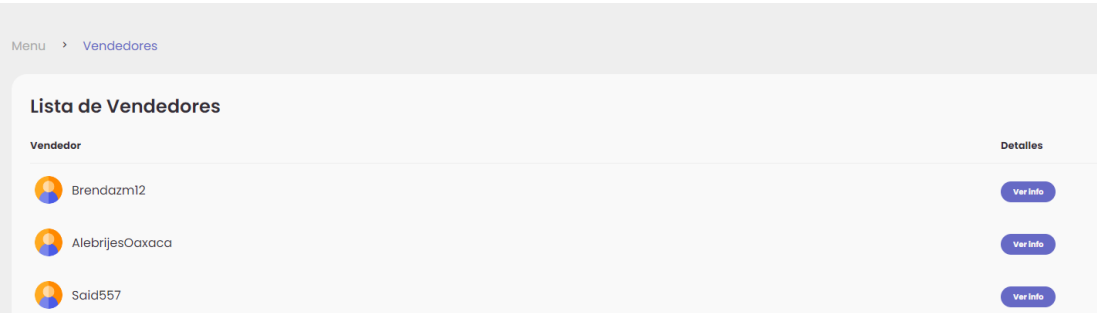


Figura 81. Listado de vendedores activos en plataforma panel de administrador.

Como también se pueden ver los detalles de cada vendedor dando clic en el botón “ver info”. Interfaz de detalle de vendedor, mostrada en la Figura 82 también se muestran los detalles del vendedor seleccionado, como también el botón eliminar, esto por motivos de seguridad.

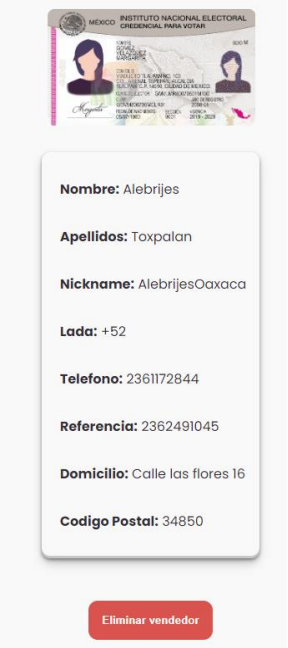


Figura 82. Detalles de vendedor panel de administrador.

6. **Reportes.** Como podemos ver en la interfaz presentada en la Figura 83 de reportes en el panel de administrador se divide en dos partes:
    1. Lista de reportes.
    2. Reportes en revisión.

Cada uno de estos puntos será explicado más adelante.

Figura 83. Interfaz de reportes panel de administrador.

6.1 Lista de reportes. En esta parte de la interfaz solo se muestran los reportes que no han sido revisados por el administrador, ya que, como se muestra en la interfaz presentada en la Figura 82, solo se muestra la lista de los reportes y se pueden ver los detalles de dicho reporte. Dando clic en el botón “ver info” (véase la Figura 84).

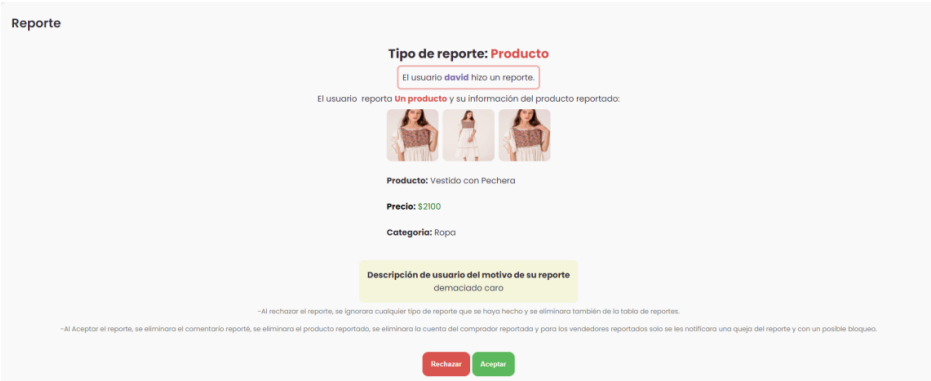


Figura 84. Detalle de reporte.

En la Figura 84 podemos observar los siguientes datos:

1. Tipo de reporte.
2. Usuario que reporta.
3. Imágenes del producto reportado.
4. Nombre de producto.
5. Precio del producto.
6. Categoría del producto.
7. Descripción del motivo del reporte.
8. Botones de aceptar y rechazar.

Antes de seguir con la explicación de esta interfaz es bueno comentar que existen cuatro tipos de reportes en este caso son los siguientes:

- Comentario: si un reporte de comentario es aprobado el comentario es eliminado.
- Producto: si un producto es reportado el usuario tiene la posibilidad de actualizar su producto para devolverlo a revisión, sin embargo, el producto permanecerá dado de baja y no se mostrará en el catálogo.
- Comentario en perfil: si este tipo de reporte es aprobado el comentario será eliminado.
- Perfil de vendedor: si el reporte de perfil es aprobado, el perfil será dado de baja de la plataforma.

6.2 Reportes en revisión. Continuando con la explicación, en este caso si aceptamos el reporte, como es de tipo producto entonces será enviada una notificación al vendedor (panel del vendedor presentado en la Figura 85), donde dará la opción de actualizar los datos del producto o dar caso omiso y el producto será quitado del catálogo (interfaz de notificación presentada en la Figura 86).

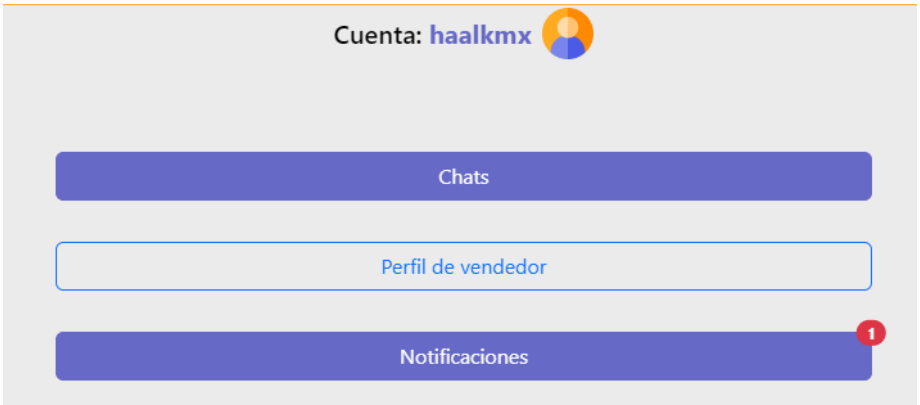


Figura 85. Notificación de prueba.

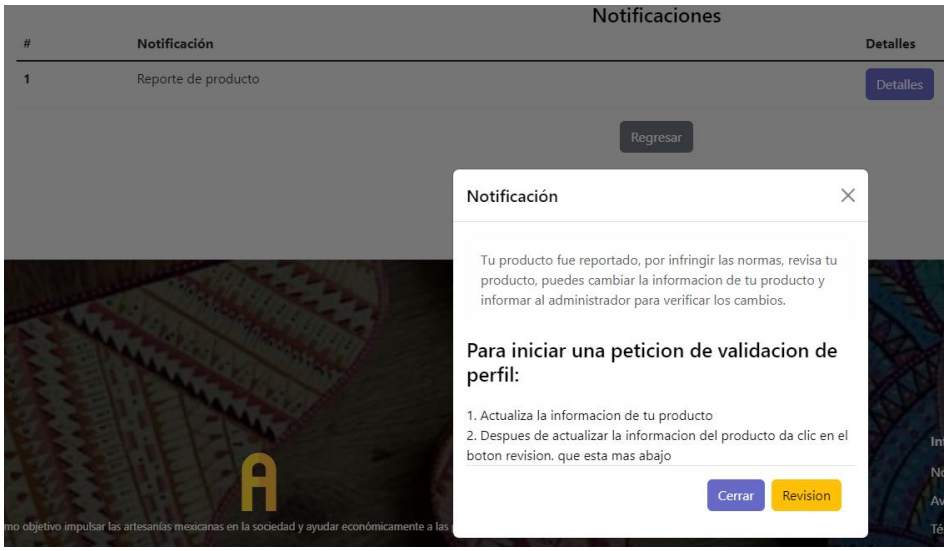


Figura 86. Detalle de notificación, panel de usuario vendedor.

Hasta que el vendedor actualice la información de su producto y de clic en el botón “revisión” que se muestra en la interfaz mostrada en la Figura 86**,** el reporte en el panel de administrador cambiará de estatus a en

revisión y se mostrará en la otra pestaña (la interfaz de detalle se presenta en la Figura 87).

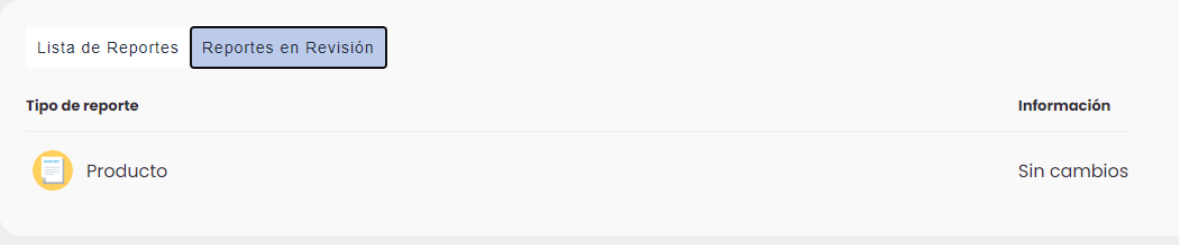


Figura 87. Reportes en espera de revisión.

Después de que el usuario vendedor haya actualizado los datos de su producto el administrador podrá o no aceptar los cambios. Dando clic en el botón “ver info” mostrado en la Figura 88 y aceptando o rechazando como se muestra en la interfaz presentada en la Figura 89.

Figura 88. Estatus de revisión activo.

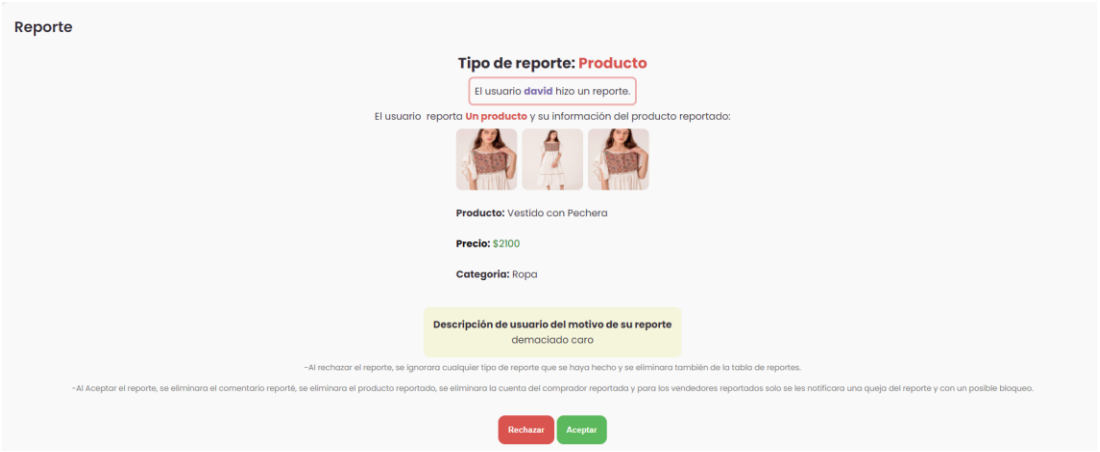


Figura 89. Detalles de producto actualizados.

7. **Nuevos vendedores.** Como se observa en la interfaz de muestra presentada en la Figura 90, esta interfaz muestra las solicitudes para nuevos vendedores, en este caso aparece el nombre del nuevo vendedor u organización, dando clic en el botón “ver info” se podrá ver la solicitud más detalladamente (interfaz de detalle presentada en la Figura 91).

   En la Figura 91 podemos observar, los datos del vendedor que se registran en el apartado 4.5.5.1 usuario nuevo, además, el administrador puede aceptar o rechazar dicha solicitud, lo cual, depende de la información presentada por el solicitante.

   Si la solicitud es aceptada, al usuario le llegara una notificación a su perfil, notificando que ha sido aceptada su solicitud, o de caso contrario que ha sido rechazada, dependiendo cuál sea la opción el administrador puede adjuntar un mensaje.

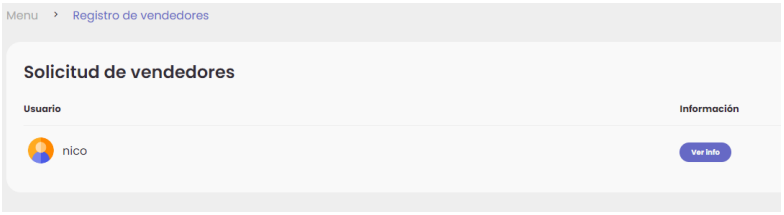


Figura 90. Interfaz de solicitudes para nuevos vendedores.

Figura 91. Detalles de solicitud de vendedor.

8. **Nuevos productos.** Como podemos observar en la interfaz presentada en la Figura 92, esta sección es de muestreo de información, donde se listan las solicitudes de los productos que pueden vender suben usuario. En este caso se puede ver los detalles dando clic en el botón “ver info”, donde, se mostrará la siguiente información (interfaz de detalle presentada en la Figura 93).

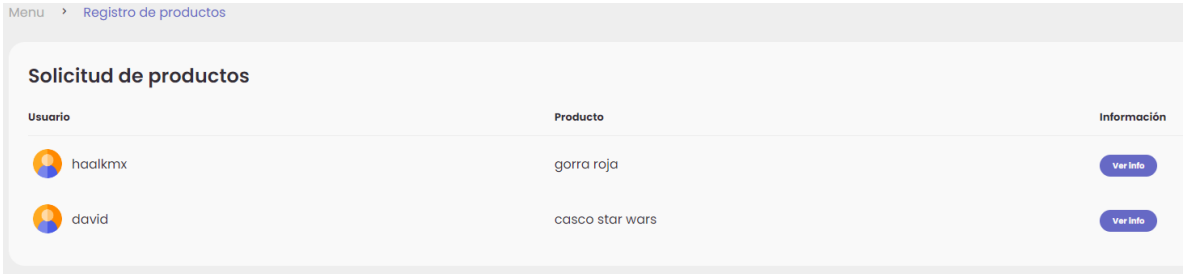


Figura 92. Solicitud de nuevos productos, panel de administrador.

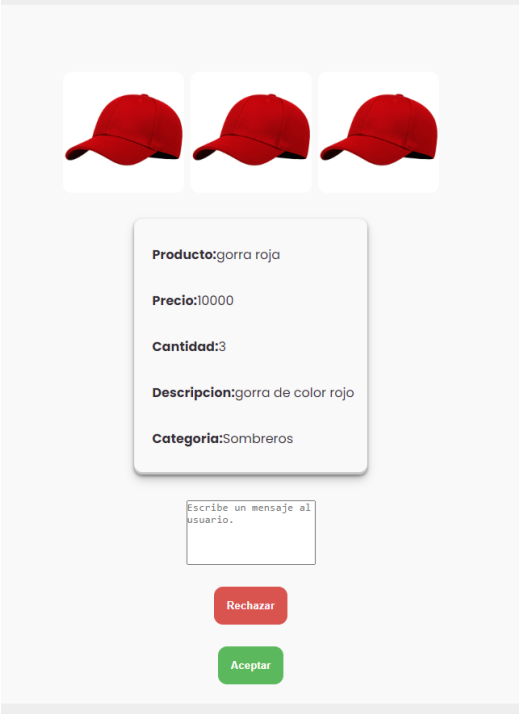


Figura 93. Detalle de solicitud de producto.

- Imágenes correspondientes al producto.
- Nombre del producto.

- Precio del producto.
- Cantidad del producto.
- Descripción del producto.
- Categoría del producto.

Además, el administrador puede aceptar o rechazar dicha solicitud, lo cual, depende de la información presentada por el solicitante. Si la solicitud es aceptada, al usuario le llegara una notificación a su perfil, notificando que ha sido aceptada su solicitud, o de caso contrario que ha sido rechazada, dependiendo cuál sea la opción el administrador puede adjuntar un mensaje. Si la solicitud ha sido aceptada, el producto se pondrá en línea y se mostrará en el catálogo automáticamente.

5. Conclusiones y trabajo a futuro

En este trabajo de tesis, se desarrolló una plataforma colaborativa, que suministró la comunicación directa de los artesanos mexicanos con todos los estados de México y con personas de otros países, también apoyó a las nuevas generaciones de artesanos, para así, seguir con la creación de las emblemáticas piezas y creaciones de cada estado, aunado a esto se logra evitar enormemente la deserción de nuevos y antiguos artesanos. Además de que contribuyó a mejorar la economía de los artesanos, a quienes es imposible promover sus productos de forma autónoma. Respecto al desarrollo de esta plataforma se finaliza de forma exitosa, cumpliendo los requisitos y prospectos de una forma satisfactoria, el *Back-End* se realiza con un soporte de SCRUM, concluyendo con tareas y Sprints que cumplen las necesidades del cliente y de cada usuario, dejando un código rápido, robusto y entendible para todo desarrollador. Como resultado de este trabajo el capítulo de libro denominado "*ARTXICANS: Plataforma Colaborativa Web y Móvil para apoyar a las ARTesanías meXiCANaS*" ha sido aceptado para ser publicado en la editorial Alfa-Omega.

Como trabajo a futuro, se optimizará el código, la administración para la base de datos, he implementarán más aspectos sobre la seguridad, como certificados para pagos y base de datos para evitar las peligrosas inyecciones SQL.

6. Bibliografía.

[1] Ellis, C.A., Gibbs, S.J. and Rein, G.L. Groupware: some issues and experiences. Communications of the ACM, vol. 34-1, pp. 39-58, (1991)

[2] Anzures-García, M., Sánchez-Gálvez, L.A., Hornos, M.J. and Paderewski, P. Tutorial function groupware based on a workflow ontology and a directed acyclic graph. IEEE Latin American Transactions, vol. 16-1, pp. 294-300. (2018)

[3] Mario Anzures-García, and Luz A. Sánchez-Gálvez. PROMISE: PRoposing an Ontological Model for developing collaboratIve SystEms. Journal of Intelligent & Fuzzy Systems, Vol. 39 (2), 2020.

[4] Anzures-García, M., Sánchez-Gálvez, L.A., Hornos, M.J. and Paderewski, P. A workflow ontology to support knowledge management in a group's organizational structure, Computación y Sistemas, vol. 22-1, pp. 163–178. (2018)

[5] Anzures-García, M., Sánchez-Gálvez, L.A., Hornos, M.J. and Paderewski, P. A workflow ontology to support knowledge management in a group's organizational structure, Computación y Sistemas, vol. 22-1, pp. 163–178. (2018)

[6] Luz A. Sánchez-Gálvez, Juan Manuel Fernández-Luna and Mario Anzures-García. A Groupware Usability-oriented Evaluation Methodology based on a Fuzzy Linguistic Approach. Serie Communications in Computer and Information Science Springer, vol. 1114, pp. 1-16, 2019, HCI-COLLAB.

[7] Luz A. Sánchez-Gálvez y Juan M. Fernández-Luna, (2015). A Usability Evaluation Methodology of Digital Library. eKNOW 2015: The Seventh International Conference on Information, Process, and Knowledge Management, pp. 23-28, IARIA.

[8] Sistema de información legislativo de la secretaría de gobernación, Revisado el día 13 de octubre de 2022: http://sil.gobernacion.gob.mx/Archivos/Documentos/2021/04/asun_4166156_20210408_1618241733.pdf

[9] INEGI. diputados.gob.mx, 2021, p01. Revisado el día 13 de octubre de 2022

[10] Fondo Nacional para el Fomento de las Artesanías (FONART, 2021), Revisado el día 22 de octubre de 2022: https://www.gob.mx/cms/uploads/attachment/file/596992/Diagno_stico_Pandemia_Fonart.pdf .

[11] Mercadolibre, Revisado el día 24 de octubre de 2022: https://www.mercadolibre.com.mx/#from=homecom

[12] Amazon, Revisado el día 24 de octubre de 2022: https://www.amazon.com.mx

[13] Artesaliz, Revisado el día 25 de octubre de 2022: https://www.artesaliz.com/

[14] Tienda Mex, Revisado el día 26 de octubre de 2022: https://tienda-mex.com/
[15] La Rancherita tienda-mex, p. sobre nosotros, Revisado el dia 07 de febrero de 2023.
[16] Artesanias de mexico, Revisado el dia 07 de febrero de 2023.
[17] Exixo, Revisado el dia 07 de febrero de 2023: https://artesaniasdemexico.com.mx
[18] Exixo, p. ¿Quiene Somos?, Revisado el dia 07 de febrero de 2023.
[19] Estilo mexicano, Revisado el día 07 de febrero de 2023: https://www.estilomexicano.com.mx
[20] Estilo mexicano, p. Recopilación de datos personales, Revisado el día 07 de febrero de 2023.
[21] Flor de piña, Revisado el día 07 de febrero de 2023: https://www.flordepina.mx
[22] Artesanías de México, Revisado el día 17 de mayo de 2023: https://artesaniasdemexico.com
[23] Artesanías de México, p. Política de privacidad, Revisado el día 17 de mayo de 2023.
[24] Ensamble artesano, Revisado el día 17 de mayo de 2023: https://www.ensambleartesano.org/index
[25] Ensamble Artesano, p. Quienes somos, Revisado el día 17 de mayo de 2023.
[26] Gaalxim, Revisado el día 17 de mayo de 2023: https://www.gaalxim.com
[27] Gaalxim, p. ¿Por qué comprar con nosotros?, Revisado el día 17 de mayo de 2023.
[28] INEGI. inegi.org.mx, 2023, Comunicado de prensa num 159/23, p. la artesanía y la covid-19, Revisado el día 17 de mayo de 2023
[29] Scrum.org, Revisado el día 18 de mayo de 2023: https://www.scrum.org/about
[30] Scrum.org, Revisado el día 18 de mayo de 2023: https://www.scrum.org/learning-series/what-is-scrum/the-scrum-events/what-is-a-sprint
[31] Ingeniería de software, Revisado el día 18 de mayo de 2023: https://ingenieriadesoftware.mex.tl/61154_asd.html
[32] Kanbanize, Revisado el día 18 de mayo de 2023: https://kanbanize.com/es/recursos-de-kanban/primeros-pasos/que-es-kanban
[33] Php, Revisado el día 11 de agosto de 2023: https://www.php.net/sites.php#www
[34] Help center, Revisado el día 11 de agosto de 2023: https://docs.oracle.com/en-us/iaas/mysql-database/doc/db-system.html
[35] PayPal API, Revisado el dia 15 de agosto de 2023: https://developer.paypal.com/dashboard/