

4.1. Strategic-Level Attribute-Driven Design

4.1.1. Design Purpose

Para describir correctamente el Design Purpose de AdventureHub, es importante tener en cuenta los siguientes puntos:

Propósito general

El propósito general de AdventureHub es proporcionar una plataforma en línea fácil de usar que permita a los viajeros comprar paquetes de viaje personalizados que se adapten a sus intereses y presupuestos individuales. Nuestro objetivo es simplificar el proceso de planificación de viajes para los usuarios y ofrecer información confiable y transparente sobre precios y paquetes de viaje. Además, buscamos promover el turismo y apoyar a pequeñas agencias de viajes locales para fomentar un impacto positivo en la economía local y el turismo sostenible.

Público objetivo

Se tiene dos públicos objetivos distintos: 1. Personas que buscan experiencias de aventura. 2. Agencias de viajes que buscan publicar sus tours de viaje.

Para el primer grupo, se busca una aplicación que les ofrezca: - Información detallada sobre destinos y actividades de aventura disponibles. - Precios y paquetes de viaje competitivos y adaptados a sus necesidades. - Acceso a información sobre transporte, alojamiento y otros servicios necesarios para su viaje. - Una experiencia de usuario fácil de usar.

Por otro lado, las agencias de viajes buscan una plataforma en línea fácil de usar y eficiente para publicar sus tours de viaje y llegar a nuevos clientes potenciales, con funciones de gestión de reservas y pagos para facilitar la administración de los tours.

Características clave

Las características clave de AdventureHub incluyen: - **Usabilidad:** La aplicación debe ser fácil de usar e intuitiva para los usuarios, con una interfaz atractiva y funcionalidades claras. - **Rendimiento:** La aplicación debe ser rápida y confiable para una experiencia de usuario fluida. - **Seguridad:** La aplicación debe proteger la información personal y financiera de los usuarios. - **Escalabilidad:** La aplicación debe ser capaz de manejar un gran volumen de usuarios y transacciones sin afectar el rendimiento o la seguridad de la aplicación. - **Compatibilidad con múltiples plataformas:** La aplicación debe ser compatible con múltiples dispositivos, incluyendo computadoras de escritorio, laptops, tabletas y smartphones. - **Personalización:** La aplicación debe permitir la personalización de opciones de viaje para los usuarios. - **Integración de redes sociales:** La aplicación debe permitir la integración con las redes sociales para compartir información sobre experiencias de viaje y promocionar la aplicación.

Arquitectura del Sistema

La Arquitectura del Sistema propuesta para AdventureHub consiste en Microservicios bajo Attribute Driven Design (ADD). Se utilizarían diferentes tecnologías para cada capa, según las necesidades específicas de cada microservicio.

En la **Capa de presentación** se utilizaría una aplicación web o móvil como interfaz de usuario y un framework de JavaScript como React, Angular o Vue.js para la lógica de presentación. En la **Capa de servicios**, cada microservicio se implementaría utilizando un lenguaje de programación diferente y se utilizaría un protocolo de comunicación como REST o gRPC para la comunicación entre microservicios. En la **Capa de acceso a datos**, se utilizarían diferentes tecnologías de almacenamiento de datos, como bases de datos relacionales o NoSQL, y frameworks de ORM como Hibernate o Sequelize para el acceso a los datos. En la **Capa de infraestructura**, se utilizarían tecnologías de contenerización como Docker para gestionar los contenedores de los microservicios, y herramientas de automatización como Ansible o Puppet para la gestión de la configuración de la infraestructura.

4.1.2. Attribute-Driven Design Inputs.

4.1.2.1. Primary Functionality (Primary User Stories).

La siguiente tabla enumera las Historias de Usuario Principales que definen las funcionalidades clave de nuestro proyecto de software. Estas historias representan las necesidades y objetivos principales de los usuarios y administradores, y son esenciales para el funcionamiento exitoso de nuestra aplicación. Cada historia de usuario se identifica con un ID único.

ID

Historia de Usuario

US-04

Como usuario, quiero poder buscar paquetes de viaje en base a mi presupuesto, para encontrar opciones que se ajusten a mis necesidades financieras.

US-05

Como usuario, quiero poder explorar paquetes de viaje en base a la temporada del año, para encontrar opciones que se adapten al clima y las actividades disponibles en el momento.

US-06

Como usuario, quiero poder buscar paquetes de viaje en base al destino escogido, para encontrar opciones que se ajusten a mi ubicación y preferencias culturales.

US-07

Como usuario, quiero poder reservar un paquete de viaje en la aplicación, para poder asegurar mi lugar y mi itinerario.

US-13

Como usuario, quiero recibir notificaciones sobre cambios en mi itinerario de viaje, para estar informado y preparado.

US-17

Como administrador, quiero poder enviar notificaciones personalizadas a los usuarios para informarles sobre nuevos paquetes de viaje y ofertas especiales.

US-24

Como usuario, quiero poder solicitar un paquete personalizado a través del chat de la aplicación, para poder encontrar la mejor opción de viaje para mí.

US-25

Como usuario, quiero recibir notificaciones en tiempo real cuando una agencia acepte mi solicitud de paquete personalizado, para poder comenzar la conversación de inmediato.

US-26

Como agencia de viajes, quiero poder ver una lista de solicitudes personalizadas de usuarios, para poder seleccionar las solicitudes que me interesan y comenzar una conversación.

US-28

Como usuario, quiero poder comunicarme directamente con la agencia de viajes a través del chat de la aplicación, para poder hacer preguntas y resolver cualquier problema relacionado con mi reserva.

US-30

Como usuario, quiero poder aceptar o rechazar los componentes del paquete personalizado que se me ofrecen, para poder tener una experiencia personalizada y a medida.

4.1.2.2. Quality attribute Scenarios.

Estos escenarios ayudan a definir y comprender las expectativas y requisitos clave en términos de calidad del sistema que estamos desarrollando. A continuación, se describirán los más importantes.

Atributo

Fuente

Estímulo

Artefacto

Entorno

Respuesta

Medida

Portabilidad

Usuario

Entrar a la plataforma web con diferente navegador

Frontend

Plataforma web en la que el usuario accede desde su navegador favorito

Logra cargarse el portal web en el dispositivo del usuario en su navegador favorito

Número de visitas por navegador web

Fiabilidad

Usuario

Entrar a la plataforma web en cualquier hora del día

Frontend

Plataforma web accedida por el usuario a cualquier hora del día

Logra cargarse el portal web en el dispositivo del usuario a cualquier hora del día

Número de visitas registradas a cualquier hora del día

Fiabilidad

Usuario

Backend con alta afluencia de usuarios

Backend

3 nodos para alta disponibilidad en los servidores Backend

Redirección de la carga de solicitudes a través de un Load Balancer

Número de solicitudes por servidor, esta debe presentarse de forma uniforme

Fiabilidad

Servidores

Fallas técnicas en los servidores

Frontend y Backend

3 nodos para alta disponibilidad en los servidores Frontend y Backend

Logran funcionar los servidores

Heartbeat, para saber si nuestros servidores están funcionando

Rendimiento

Usuario

Solicitud de procesar transacciones de pagos de paquetes de viaje

Frontend y Backend

Carga de 500 transacciones de pagos de paquetes de viaje, sin tiempo de espera para los usuarios o menor a 1 segundo

Logra procesarse con satisfacción las transacciones de pagos de paquetes de viaje

Número de solicitudes de pagos de viaje no mayores a 500 transacciones por hora

Rendimiento

Usuario

Frontend con alta afluencia de usuarios

Frontend

3 nodos para alta disponibilidad en los servidores Frontend

Redirección de la carga de solicitudes a través de un Load Balancer

Número de solicitudes por servidor, esta debe presentarse de forma uniforme

Seguridad

Usuario

Solicitud de autenticación y autorización menor a 2 segundos

Backend

Servidores Backend procesando la autenticación y autorización

Logran autenticarse y autorizarse los usuarios

Tiempo de respuesta de la solicitud en los servidores

Usabilidad

Usuario

Reserva de paquete de viaje en menos de 10 minutos

Frontend y Backend

Servidores Frontend y Backend procesando el proceso de reserva de paquete de viaje

Logran realizar la reserva

Tiempo en realizar la reserva desde que el usuario entra a la página

Escalabilidad

Usuario

Backend con muy altos niveles de afluencia prolongados

Backend

Servidores Backend con políticas de escalabilidad

Aumento de nodos en los servidores Backend

Número de instancias agregadas desde la plataforma de nube

Mantenibilidad

Servidores

Errores de producción resueltos en menos de 15 minutos

Backend

Servidores Backend desplegadas en una plataforma cloud que brinde herramientas de monitoreo

Métricas en la plataforma de despliegue que tenga servicios de alertas

Tiempo de duración de la alerta

4.1.2.3. Constraints.

En el proceso de desarrollo de nuestro proyecto de software, es esencial comprender y definir con claridad las restricciones que influirán en el diseño, implementación y funcionamiento del sistema. A continuación se detallarán cada una de ellas.

ID

Constraint

CON-1

Las cuentas de los usuarios y los permisos de estos mismos deben ser manejados por un servidor directo de usuarios.

CON-2

Se debe realizar la conexión de la base de datos con los servicios de Amazon Web Services (AWS).

CON-3

La conexión de redes en las estaciones de trabajo de los usuarios puede tener baja latencia, pero generalmente es confiable.

CON-4

El sistema debe auto guardar los últimos 30 eventos, que se realizaron.

4.1.3. Architectural Drivers Backlog

Siguiendo las secciones previas que han abordado los Functional Drivers, Quality Attribute Drivers y todos los Constraints mencionadas, se han identificado los siguientes factores determinantes. Esto se logró al evaluar tanto la implementación de la solución tecnológica como su relevancia en el Business Core.

DRIVER ID

Título

Descripción

Importancia para Stakeholders (High, Medium, Low)

Impacto en la Complejidad Técnica de la Arquitectura (High, Medium, Low)

DR1

Experiencia del Usuario

Crear una interfaz de usuario intuitiva y atractiva para que los buscadores de aventuras encuentren fácilmente información sobre destinos, actividades y paquetes de viaje, y puedan planificar sus aventuras de manera eficiente.

High

Medium

DR2

Escalabilidad del Software

Asegurar que las agencias de viajes puedan publicar y gestionar fácilmente sus tours de viaje en la plataforma, incluso en momentos de alta demanda, para llegar a nuevos clientes y gestionar las reservas de manera eficiente.

High

High

DR3

Seguridad en los datos de los usuarios y las agencias de viajes

Proteger la información confidencial tanto de las agencias de viajes como de los usuarios que utilizan la plataforma, incluyendo detalles de tours y datos comerciales, para garantizar la confianza al uso de la aplicación.

Medium

Medium

DR4

Motor de Búsqueda por Presupuesto

Desarrollar un motor de búsqueda que permita a los usuarios buscar paquetes de viaje según su presupuesto, lo que facilitará encontrar opciones ajustadas a sus necesidades financieras.

High

High

DR5

Asistente Virtual con Inteligencia Artificial

Implementación de un asistente virtual con Inteligencia Artificial para ofrecer asistencia y recomendaciones personalizadas a los viajeros que buscan aventuras, mejorando la experiencia de búsqueda y planificación.

High

High

DR6

Reservas en la Aplicación

Permitir a los usuarios reservar paquetes de viaje directamente a través de la aplicación para garantizar su lugar y su itinerario, mejorando la experiencia de reserva.

High

High

4.1.4 Architectural Design Decision

DRIVER ID

Título del Driver

Pattern 1: MVC

Pattern 2: MVVM

Pro

Con

Pro

Con

DR1

Experiencia del Usuario

La separación clara entre lógica y presentación permite una fácil adaptación de la interfaz.

Puede requerir más trabajo manual para actualizar las vistas cuando cambian los modelos.

El binding bidireccional puede mejorar la interactividad y la reactividad de la interfaz. Facilita la creación de interfaces ricas y dinámicas.

Mayor complejidad en comparación con MVC debido al binding bidireccional.

DR2

Escalabilidad del Software

Modularidad que facilita la escalabilidad horizontal.

Puede ser menos eficiente en sistemas con muchas actualizaciones de vistas en tiempo real.

Mejor para aplicaciones ricas en cliente con muchas actualizaciones dinámicas.

Mayor overhead debido al binding bidireccional.

DR3

Seguridad en los datos de los usuarios y las agencias de viajes

El controlador actúa como un intermediario, lo que puede mejorar la seguridad al controlar qué datos se muestran.

Mayor riesgo si no se manejan adecuadamente las entradas/salidas.

El ViewModel puede filtrar y transformar datos antes de que lleguen a la vista, potencialmente mejorando la seguridad.

Como el ViewModel está más orientado a la presentación, podría existir un riesgo si no se manejan adecuadamente las transformaciones de datos.

DR4

Motor de Búsqueda por Presupuesto

Fácil de adaptar la lógica de búsqueda en el controlador.

La actualización de vistas puede no ser tan reactiva.

Las búsquedas y filtros pueden reflejarse en tiempo real en la vista gracias al binding.

La lógica de filtrado puede estar más dispersa entre el ViewModel y el Modelo.

DR5

Asistente Virtual con Inteligencia Artificial

Clara separación de lógica de IA (modelo) y su presentación (vista).

La comunicación entre la vista y el modelo puede ser menos fluida.

Mayor reactividad y actualizaciones en tiempo real en la interfaz de usuario.

La lógica del asistente puede estar dispersa entre ViewModel y Modelo, lo que puede complicar la implementación.

DR6

Reservas en la Aplicación

La lógica de reserva puede gestionarse de manera centralizada en el controlador.

Puede requerir más interacciones entre la vista y el controlador para actualizaciones.

Las actualizaciones de estado de la reserva pueden reflejarse en tiempo real en la vista.

Puede haber más complejidad al manejar estados en el ViewModel.

4.1.5. Quality Attribute Scenario Refinements

Scenario Refinement for US07 Scenario 1

Scenario

Como usuario, quiero poder reservar un paquete de viaje en la aplicación, para poder asegurar mi lugar y mi itinerario, se realizará en menos de 1 segundo la actualización

Business Goals

Sistema confiable, característica principal.

Relevant Quality Attribute

Funcionalidad y Performance

Scenario Components

Stimulus

Usuario selecciona el botón “Agendar reserva”

Stimulus Source

Botón en la app

Environment

Vista relacionada en la app

Artifact (if known)

Proceso Backend

Response

Se muestra en el mapa del sistema

Response Measure

<1 segundo

Questions

¿Qué paquetes se mostrarán disponibles para agendar la reserva?

Issues

Los paquetes de viaje dependen de las agencias que las brindan.

Scenario Refinement for US011 Scenario 1

Scenario

Como usuario registrado, quiero recibir un recibo de mi pago por correo electrónico después de hacer una reserva exitosa, para tener un registro de mi transacción, en menos de 1 segundo.

Business Goals

Sistema confiable, característica principal.

Relevant Quality Attribute

Operabilidad y Performance

Scenario Components

Stimulus

Usuario selecciona el botón “Pagar”

Stimulus Source

Botón en la app

Environment

Vista relacionada en la app

Artifact (if known)

Proceso Backend

Response

Se muestra en el mapa del sistema

Response Measure

<1 segundo

Questions

¿Qué se podría hacer en caso no se mande el correo?

Issues

El servicio de envío de correo se tendría que desarrollar o usar un servicio externo.

Scenario Refinement for US010 Scenario 1

Scenario

Como usuario registrado, quiero poder guardar múltiples métodos de pago en mi perfil, para poder hacer reservas con mayor facilidad y rapidez.

Business Goals

Sistema confiable, característica principal.

Relevant Quality Attribute

Operabilidad y Performance

Scenario Components

Stimulus

Usuario selecciona el botón “Agregar método de pago”

Stimulus Source

Botón en la app

Environment

Vista relacionada en la app

Artifact (if known)

Proceso Backend

Response

Se muestra en el mapa del sistema

Response Measure

<1 segundo

Questions

¿Qué acción tomaremos si el método de pago no es registrado correctamente?

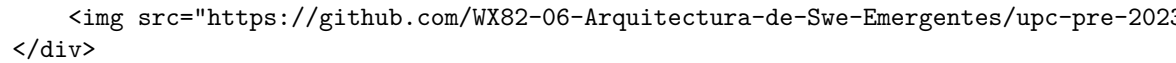
Issues

El método de pago podría no registrarse correctamente.

4.2.2. Candidate Context Discovery

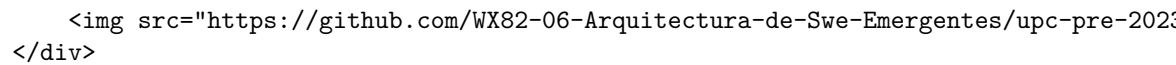
Con el EventStorming realizado, se utilizó la técnica de start-with-value para la identificación de aquellas partes del core del dominio que van a aportar un mayor valor a nuestro negocio. Por lo cual, se identificaron 3 principales que vendrían ser Customer Relationship & Communication, Traveling Experience Design and Maintenance y Traveling Experience Booking and Tracking.

El primer dominio indentificado es Customer Relationship & Communication, el cual utiliza un asistente virtual con el que los usuarios tourist podrán interactuar y realizar consultas, crear y comprar paquetes de viaje y turismo ofrecidos en la aplicación.



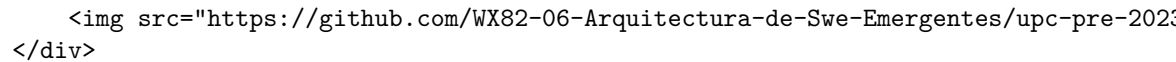
 </div>

El segundo dominio indentificado es Traveling Expirience Desing and Mainte-nance, el cual permite a los usuarios agency el poder resgistrar, actualizar y eliminar paquetes turisticos y vuelos, y a los usuarios tourist el poder elegir, comprar y ver los paquetes ofrecidos por las agencias.



 </div>

El tercer dominio identificado es Traveling Expirience Booking and Tracking, el cual permitira a los usuarios agency el poder crear, remover, actulizar reservas de los paquetes y hacer seguimiento sobre el estado de estos.



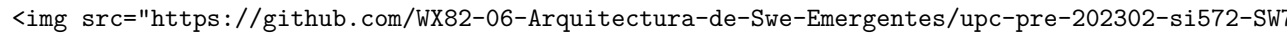
 </div>

4.2.3. Domain Message Flows Modeling

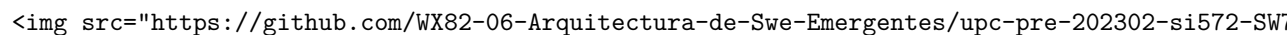
4.2.4. Bounded Context Canvases

- Travel Experience Booking and Tracking
- Travel Experience Design and Maintenance
- Customer Relationship & Communication Management
- Subscription and Payments
- Identity and Access Management
- Profiles & Social Interaction

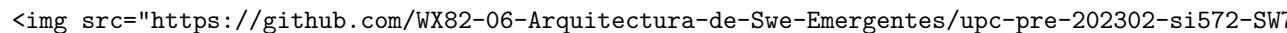
1. Scenario: Registering in the app



2. Scenario: Registering in the app



3. Scenario: Pay a Tour Package



4. Scenario: Check purchased tour packages

5. Scenario: Connect Tours Info and User chatbot to OpenAI

4.2.5. Context Mapping

- **Identity and Access Management (IAM) con Profiles & Social Interaction:**

Entre estos dos bounded contexts, se aplica el patrón Shared Kernel, ya que comparten información crítica para garantizar la consistencia. También tienen una relación de tipo Upstream/Downstream, dado que cualquier cambio en IAM podría afectar a Profiles & Social Interaction. Esto se debe a que IAM proporciona servicios de autenticación e identidad, datos fundamentales para el funcionamiento coherente de Profiles & Social Interaction.

- **Identity and Access Management (IAM) con Customer Relationship & Communication Management:**

Entre estos dos bounded contexts, se aplica la relación Upstream/Downstream, ya que los cambios en IAM podrían impactar a Customer Relationship & Communication Management. IAM actúa como el contexto “aguas arriba”, proporcionando servicios de autenticación e identidad que son esenciales para el correcto funcionamiento de Customer Relationship & Communication Management, el contexto “aguas abajo”.

- **Customer Relationship & Communication Management con Profiles & Social Interaction:**

Entre estos dos bounded contexts, se aplica el patrón Customer/Supplier, ya que Customer Relationship & Communication Management depende de los datos del perfil de usuario proporcionados por Profiles & Social Interaction para operar su asistente virtual con IA. También tienen una relación Upstream/Downstream, ya que cualquier cambio en Profiles & Social Interaction afectará a Customer Relationship & Communication Management.

- **Customer Relationship & Communication Management con Subscription and Payments:**

Entre estos dos bounded contexts, se aplica la relación Free, ya que no necesitan uno al otro para funcionar de manera independiente.

- **Customer Relationship & Communication Management con Travel Experience Design and Maintenance:**

Entre estos dos bounded contexts, se aplica el patrón Customer/Supplier, ya que Customer Relationship & Communication Management recibe información sobre

tours y vuelos disponibles desde Travel Experience Design and Maintenance para su asistente IA. También tienen una relación Upstream/Downstream, dado que cualquier cambio en Customer Relationship & Communication Management afectará a Travel Experience Design and Maintenance.

- **Travel Experience Design and Maintenance con Travel Experience Booking and Tracking:**

Entre estos dos bounded contexts, se aplica el patrón Customer/Supplier, ya que Travel Experience Booking and Tracking recibe información sobre tours y vuelos disponibles desde Travel Experience Design and Maintenance para la reserva. También tienen una relación Upstream/Downstream, ya que cualquier cambio en Travel Experience Design and Maintenance afectará a Travel Experience Booking and Tracking.

- **Travel Experience Booking and Tracking con Subscription and Payments:**

Entre estos dos bounded contexts, se aplica el patrón Customer/Supplier, y también tienen una relación Upstream/Downstream, ya que la reserva dentro de Travel Experience Booking and Tracking depende del pago correspondiente que se realiza dentro de Subscription and Payments.

- **Profiles & Social Interaction con Subscription and Payments:**

Entre estos dos bounded contexts, se aplica la relación Upstream/Downstream, ya que los datos del perfil de usuario son necesarios para el proceso de pago dentro de Subscription and Payments.

4.3 Software Architecture

4.3.1 Software Architecture System Landscape Diagram