***LottoWeb***

LottoWeb busca enlazar usuarios que buscan servicios de mudanzas o acarreos de todo tipo con transportistas que cuentan con vehículos de carga que sastifagan dichan necesidad, logrando a su vez incrementar sus ingresos a partir de actividades por demanda

* **Integrantes**
  + Santiago Buitrago
  + Adriana Castañeda
  + Guillermo Castro
* **¿Por qué la idea?**
* **¿Qué Beneficios Brinda a la comunidad?**

Además de seguridad, se lo ofrece una gama abundante y variada tanto de servicios como objetos para casi cualquier necesidad.

* **¿Qué Necesidad resuelve?**

Las personas buscan una alternativa eficiente, segura y digital de solicitar el transporte de sus mudanzas o acarreos y Los transportistas que cuentan con camiones, camionetas, entre otros, necesitan generar ingresos con sus vehículos a partir de actividades por demanda. Actualmente estos servicios sólo los pueden realizar con familiares y conocidos, o en lugares comerciales esperando que usuarios lleguen a ellos (Estos lugares generalmente son de seguridad crítica).

* **Problemática**

En el servicio de mudanzas y acarreos se identificaron diferentes problemáticas que hacen que los usuarios no se encuentran satisfechos ni se sienten seguros. Entre estos se encuentran:

* Tiempos altos para solicitar el servicio (Se debe solicitar con aticipación)
* Alto nivel de inseguridad en la operación
* Altos costos

**Arquitectura Backend**

**El Backend de Lottoweb fue desarrollado en el siguiente repositorio: https://github.com/LottoWebTeam/ProyectoARSW2021\_LottoWeb**

**Diagrama de Clases – Modelo**

**(Foto)**

**Diagrama de Entidad – Relacion**

**(Foto)**

**Diagrama de Componentes**

**(Foto)**

**Arquitectura Frontend**

**Arquitectura de Despliegue**

**Casos de Uso**

**Atributos Funcionales**

* + - **Mostrar productos y su disponibilidad (Core)**
    - **Descripción del producto y su uso adecuado**
    - **Mostrar información del oferente(Core)**
    - **Comentar el Producto**
    - **Historial de productos alquilados**
    - **Hacer feedbacks al oferente.**
    - **Mostrar información del demandante(Core)**
    - **Control Pagos y Alquileres.**

**Atributos No Funcionales**

**Usabilidad**

**Escenarios**

Escenarios de solicitar/aceptar servicios.

1. **Source:** Usuario final (Conductor/Usuario del aplicativo).
2. **Stimulus:** Usar el sistema eficientemente.
3. **Artifact:** Servidores Frontend, servidor Backend, servidores de Bases de datos, servidores de Aplicativos externos.
4. **Enviroment:** Aplicación ejecutándose en condiciones normales.
5. **Response:** Es sencillo, tanto para solicitar como aceptar servicios, a cada tipo de usuario.
6. **Response Measure:** El conductor realiza 1 click para aceptar un servicio. El usuario, después de configurar sus preferencias y destino, realiza 1 click para solicitar el servicio.

(FOTO)

Escenarios de visualización de ofertas de servicios.

1. **Source:** Usuario final (Conductor/Usuario del aplicativo).
2. **Stimulus:** Verificar que el sistema haya registrado correctamente la toma de un servicio.
3. **Artifact:** Servidores Frontend, servidor Backend, servidores de Bases de datos.
4. **Enviroment:** Aplicación ejecutándose en condiciones normales.
5. **Response:** Facilidad para identificar los servicios que se han aceptado/solicitado.
6. **Response Measure:** Tanto el usuario como el conductor ingresan con un click a sus ofertas.

Escenarios de Iniciar un Viaje.

1. **Source:** Usuario final (Conductor/Usuario del aplicativo).
2. **Stimulus:** Verificar que el sistema haya registrado e iniciado correctamente el viaje para ambos actores
3. **Artifact:** Servidores Frontend, servidor Backend, servidores de Bases de datos, servidores de Aplicativos externos.
4. **Enviroment:** Aplicación ejecutándose en condiciones normales.
5. **Response:** Facilidad para mostrar los servicios que se han iniciado.
6. **Response Measure:** Tanto el usuario como el conductor tendrán en sus pantallaz el viaje en progreso

(FOTO)

Escenario Conductor

(FOTO)

Escenario Usuario

(FOTO)

**Escalabilidad**

**(Foto escalabilidad)**

**Vertical:** Escenario de escalamiento vertical para la base de datos de la aplicación principal.

* 1. **Source:**  Usuario final (Conductor/Usuario del aplicativo).
  2. **Stimulus:** Momento en el cual se solicitan servicios.
  3. **Artifact:** Servidor Backend, servidores Frontend, servidor de Base de datos.
  4. **Enviroment:** En condiciones de ejecución normales.
  5. **Response:** Los servicios solicitados son generados y se guardan en la base de datos.
  6. **Response Measure:** La respuesta del servidor es, en promedio, 10ms más rápida con el escalamiento vertical.

**Horizontal:** Escenario de escalamiento horizontal para servidor externo “Google”.

* 1. **Source:**  Usuario final (Conductor/Usuario del aplicativo).
  2. **Stimulus:** Interactuar con cualquier funcionalidad de la aplicación.
  3. **Artifact:** Servidor externo Google.
  4. **Enviroment:** En condiciones de ejecución normales.
  5. **Response:** El balanceador de carga, dependiendo del estado de la máquina, realiza la petición a dicha máquina.
  6. **Response Measure:** La respuesta es aproximadamente 200ms más rápida que tener un solo servidor.

**Disponibilidad**

#### Escenarios

Escenario de escalamiento horizontal para servidor externo “Google”.

1. **Source:**  Usuario final (Conductor/Usuario del aplicativo).
2. **Stimulus:** Pedir un servicio.
3. **Artifact:** Servidor externo Google.
4. **Enviroment:** En condiciones de falla de un servidor externo.
5. **Response:** El balanceador de carga detecta que una máquina quedó inactiva.
6. **Response Measure:** Ninguna petición realizada al servidor externo “Google” es denegada.
7. **Source:**  Usuario final (Conductor/Usuario del aplicativo).
8. **Stimulus:** Actualizar datos del viaje
9. **Artifact:** Servidor Frontend y Servidor backend
10. **Enviroment:** En condiciones de falla en la base de datos.
11. **Response:** El cache del servidor backend funciona sin necesidad de guardar hasta que la base datos se recupere.
12. **Response Measure:** El usuario no detecta ninguna anomalía al momento de actualizar su viaje mientras que la base de datos no tome más de 5 minutos para recuperarse.

**Rendimiento (Performance)**

#### Escenarios

Escenario de realizar una oferta

1. **Source:** Usuario final (Usuario)
2. **Stimulus:** El tiempo que toma entre realizar una oferta por parte de un conductor y se actualice para el usuario quien solicito dicho viaje.
3. **Artifact:** Servidores Frontend, servidor backend, servidores de bases de datos.
4. **Enviroment:** Aplicación ejecutandose en condiciones normales.
5. **Response:** La oferta realizada por cada conductor podrá ser vista por el usuario
6. **Response Measure:** Cuando un conductor realiza una subasta para un servicio se demora menos de 1 segundo en actualizarce para el usuario.

Escenario de aceptar servicio

1. **Source:** Usuario final (Conductor)
2. **Stimulus:** El tiempo que toma entre aceptar una oferta un usuario y se actualice para todos los conductores.
3. **Artifact:** Servidores Frontend, servidor backend, servidores de bases de datos.
4. **Enviroment:** Aplicación ejecutandose en condiciones normales.
5. **Response:** La oferta aceptada por el usuario no podrá ser vista por el resto de conductores.
6. **Response Measure:** Cuando un usuario acepta una oferta se demora menos de 1 segundo en desaparecer para los demás conductores.

Escenario notificación que se ha publicado un nuevo viaje

1. **Source:** Usuario final (Conductor)
2. **Stimulus:** El tiempo que toma un conductor en ver que un usuario ha solicitado un servicio.
3. **Artifact:** Servidores Frontend, servidor backend, servidores de bases de datos.
4. **Enviroment:**Aplicación ejecutandose en condiciones normales.
5. **Response:** El conductor puede observar los diferentes viajes disponibles
6. **Response Measure:** Cuando un usuario solicita un servicio, el conductor observa dicha solicitud en menos de 1 segundo.

**Seguridad**

Escenario de restricción de vistas para usuarios no delegados.

1. **Source:** Usuario final del aplicativo.
2. **Stimulus:** Los usuarios que no estén registrados en la pagina no tienen acceso a funcionales como: alquilar un producto o subir un producto.
3. **Artifact:** Servidores Frontend y servidor backend(Spring Security).
4. **Enviroment:** Aplicación ejecutándose en condiciones normales.
5. **Response:** Petición denegada por parte del servidor.
6. **Response Measure:** Aproximadamente 10ms para redirigirlo a la página de login.

Escenario de garantizar el acceso a usuarios registrados.

1. **Source:** Usuario final del aplicativo.
2. **Stimulus:** Los usuarios que estén registrados en la página tienen acceso a funcionales como: alquilar un producto o subir un producto.
3. **Artifact:** Servidores Frontend y servidor backend(Spring Security).
4. **Enviroment:** Aplicación ejecutándose en condiciones normales.
5. **Response:** una vez laPetición es aceptada por parte del servidor se redirige a la página principal.
6. **Response Measure:** Aproximadamente 10ms para redirigirlo a la página principal.

Escenario de integridad de imágenes.

1. **Source:** Usuario final del aplicativo.
2. **Stimulus:** Agregar imágenes que sean coherentes con el producto.
3. **Artifact:** Servidores Frontend y servidor backend(Spring Security).
4. **Enviroment:** Aplicación ejecutándose en condiciones normales.
5. **Response** Registro satisfactorio de la imagen que desea agregar el usuario una vez verificada su coincidencia con el nombre del producto.
6. **Response Measure:** aproximadamente 10ms haciendo la verificación y subiendo la imagen en base de datos..

**Continuidad de desarrollo - Github**

El proyecto se está desarrollando en la organización [LottoWebTeam](http://github.com/arsw-starsoft)

**Despliegue - Heroku**

El despliegue se realizó en [Heroku](https://arswsynchdrive.herokuapp.com/)

**Manual de Uso**

**Enlace a las Historias de Usuario**

**Estado del Backend**