Documento de Arquitectura

DrawIt!!!

# Integrantes:

* Nicole Vanessa Montaña Gómez
* Daniel Esteban Ramos Jiménez
* Yeison Steven Barreto Rodríguez

# Resumen:

La propuesta de nuestro proyecto es crear una plataforma interactiva que usa la técnica de gamificación que permite aprender mientras te diviertes, puede usarse en el ámbito académico, familiar, personal. Similar a Kakoot!, Pinturillo, Quizizz, ofrece la posibilidad de interactuar en tiempo real con los diversos participantes dentro de la sala.

# Descripción:

Para nuestra propuesta nos basamos en Kahoot!, Quizizz con referencia al estilo didáctico y evaluativo que estos tienen, y con Pinturillo nos guiamos en el formato lúdico y gráfico para que los participantes logren dibujar con libertar en el tablero propuesto.  
El objetivo de realizar este proyecto es crear un producto educativo divertido con el cual se podrán evaluar aspectos de dibujo o con el simple motivo de distraerse. Buscamos enfocarnos en la enseñanza o práctica de diversos conceptos mediante un mecanismo más artístico, para suplir con los deseos o necesidades del participante.

# Funcionalidades:

Las funcionalidades de que tiene nuestro juego según su rol son:

Diagrama

Descripción generada automáticamente Diagrama

Descripción generada automáticamente

# Detalles de las funcionalidades:

* **Visualizar pantalla participante:** El creador de la partida puede visualizar cada una de las pantallas de los jugadores que se unan a la parida, observando en tiempo real lo que los participantes están dibujando.
* **Agregar pregunta:** El creador de la partida debe agregar una pregunta que haga referencia al objetivo de la partida en el espacio correspondiente.
* **Agregar pista:** El creador de la partida podrá agregar una pista con el fin de ayudar a los demás participantes, utilizando el botón correspondiente.
* **Crear una partida:** Un usuario podrá crear una partida, si tiene los permisos necesarios, para iniciar con el juego, y permitir que los demás participantes se unan.
* **Seleccionar ganador:** El creador de la partida podrá escoger a un solo ganador entre los demás participantes, al seleccionar la opción que aparece al oprimir su nombre.
* **Visualizar la pregunta:**  Un participante podrá visualizar la pregunta dada por el creador de la partida en el espacio correspondiente, para poder empezar a pintar.
* **Visualizar la pista:** El participante podrá visualizar la pista dada por el creador de la partida si es el primero en oprimir el botón de Obtener pista.
* **Dibujar:** La aplicación cuenta con un espacio para que el participante pueda dibujar por medio de puntos lo deseado.
* **Se una a una partida:** Un usuario podrá unirse a una partida si tiene los permisos necesarios.

# Supuestos:

* Los usuarios poseen una conexión estable y veloz, evitando que se presenten inconsistencias por la latencia.
* El usuario ya esta previamente registrado en el directorio de ingreso del juego, para determinar los permisos necesarios.

# Restricciones:

* Los servicios usados para la realización de la aplicación deben pertenecer a Microsoft Azure.
* La aplicación debe cumplir con cada uno de los atributos de calidad especificados, uno de disponibilidad, dos de seguridad y uno de mantenibilidad.
* La arquitectura planteada e implementada debe permitir escalar horizontalmente.
* Al implementar la arquitectura se debe observar una separación física de los componentes.
* El profesor debe aprobar los dos atributos de calidad de seguridad planteados.
* Se debe implementar un ciclo de integración continua.
* Se debe escanear el código por medio de SonarQube.

# Atributos de calidad:

* **Disponibilidad:**

Se debe asegurar que la aplicación tiene una alta disponibilidad, para ello decidimos hacer uso del servicio de App Service que ofrece Azure, con el fin de que la aplicación pueda ser usada en cualquier plataforma o dispositivo, e implementar una infraestructura en la nube escalable y fiable.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Fuente** | **Estímulo** | **Artefacto** | **Ambiente** | **Respuesta** | **Medida de respuesta** |
| Servicio donde se encuentra desplegada la aplicación | Despliegue de la aplicación | Proceso de despliegue del código | De ejecución | Utilización de App Service para un funcionamiento correcto | Tiempo de ejecución del servicio |

* **Mantenibilidad:**

Se debe garantizar la mantenibilidad de la aplicación, para ello implementamos una inspección continua del código estático de la aplicación por medio de la herramienta de SonarQube, que detecta errores y vulnerabilidades de seguridad. Se encuentra implementada dentro del ciclo de integración continua.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Fuente** | **Estímulo** | **Artefacto** | **Ambiente** | **Respuesta** | **Medida de respuesta** |
| Código fuente de la aplicación | Despliegue de la aplicación | Proceso de análisis del código fuente | De despliegue | Inspección del código en la integración continua | Métricas de calidad para cada uno de los aspectos |

* **Seguridad:**
* Authenticate Actors: Se debe garantizar la táctica de seguridad para saber si una persona es quien dice ser, por medio de un Active Directory vamos a identificar si la persona que desea usar nuestra aplicación ya se encuentra registrada.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Fuente** | **Estímulo** | **Artefacto** | **Ambiente** | **Respuesta** | **Medida de respuesta** |
| Usuario de la aplicación | Querer acceder a la aplicación sin estar registrado | La pantalla del Máster o del participante | De ejecución | No poder participar en la partida | Identificación desde el Active Directory |

* Authorize Actors:Se debe asegurar el buen funcionamiento de la táctica de seguridad para verificar que una persona tenga los permisos necesarios para usar las diferentes funcionalidades de nuestra aplicación. Por medio de cookies vamos a guardar la información del usuario y se verificara constantemente si posee los permisos necesarios para usar alguna función.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Fuente** | **Estímulo** | **Artefacto** | **Ambiente** | **Respuesta** | **Medida de respuesta** |
| Usuario de la aplicación | Intento de acceder a alguna función de la cual no posee permisos | Las funciones especificadas para el creador de la partida | De ejecución | Denegar el acceso a la funcionalidad querida | Identificación del usuario que desea utilizar una función no permitida |

# Decisiones arquitectónicas:

* Los productos de Microsoft Azure que usamos son:
* Azure App Service: Para desplegar el proyecto, ya que es un servicio que hospeda aplicaciones web en el lenguaje deseado, y permite una ejecución fácil y escalable en diversos entornos.
* Azure Active Directory: Es un servicio de identidad que proporciona un inicio de sesión único, autenticación multifactorial y acceso condicional.
* Se realizo una implementación de un estilo arquitectónico de API-REST.
* Se utilizo el framework de Spring para la construcción de la aplicación con un patrón de MVN (Modelo - Vista - Controlador).
* Se hizo uso de un modulo STOMP para el aprovechamiento y configuración de los tópicos, con el fin de garantizar el real-time en nuestro proyecto, mediante la arquitectura de publicador - subscritor.

# Diagramas realizados:

* Diagrama de componentes:

Diagrama

Descripción generada automáticamente

* Diagrama de clases:

Diagrama, Esquemático

Descripción generada automáticamente

* Diagrama de contexto:

Diagrama, Dibujo de ingeniería

Descripción generada automáticamente

* Diagrama de despliegue:

Diagrama

Descripción generada automáticamente

* Diagrama de actividades:

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

# Descripción de la arquitectura:

Nuestro proyecto fue organizado en las siguientes capas:

* Aplicación: Para esta utilizamos Java, Maven y Spring Boot.
* Presentación: Para esta utilizamos HTML, CSS y JavaScript.

# Conclusiones:

Con esta aplicación estamos cumpliendo con los requerimientos principales, que son el uso de real-time y concurrencia. El primero se evidencia cuando el creador de la partida desea ver lo que esta haciendo cada uno de los participantes en ese momento, y el segundo se evidencia en la obtención de la pista.

Para su funcionamiento correcto hacemos uso de diversos servicios cloud suministrados por Microsoft Azure, con el fin de brindarle una alta disponibilidad al usuario. Brindándole un uso seguro, para una buena experiencia usando la aplicación, por medio de las distintas tácticas de seguridad explicadas anteriormente.

# Vínculo del proyecto:

<https://drawit-api-1670199012625.azurewebsites.net>

<https://github.com/Derjasai/DrawIt>