

**BARRIOS CORNEJO SELENE CUEVA  
FLORES JONATHAN GARCIA DIAZ  
GERMAN HERRERA COOPER MIGUEL  
MONTESINOS APAZA SERGIO**

**EDUKIDS**



# EDUKIDS

- 1. EDUKIDS O INCENTIVA LOS NIÑOS A APRENDER Y REFORZAR CONOCIMIENTOS DE MATEMATICAS.**
- 2. EDUKIDS PERMITE EL REGISTRO DE USUARIOS PARA PODER GUARDAR SU PROGRESO EN EL JUEGO.**
- 3. EDUKIDS INCENTIVA AL USUARIO A CUMPLIR METAS MEDIANTE RECOMPENSAS ESPECIALES.**
- 4. EDUKIDS PRESENTA UNA INTERFAZ ADECUADA PARA LOS USUARIOS, LA CUAL CONTIENE AVATARES DIVERTIDOS.**



# PERFIL DEL USUARIO



# INTERFACE

- SE DEFINE COMO UNA “***CONEXIÓN FÍSICA Y FUNCIONAL ENTRE DOS APARATOS O SISTEMAS INDEPENDIENTES***”
- DESCRIBE LAS OPERACIONES QUE UNA ENTIDAD PUEDE REALIZAR, DE IGUAL MANERA ESTABLECE LOS LIMITES, NIVELES DE ACCESO Y LA MANERA EN QUE SE DESARROLLA LA COMUNICACIÓN ENTRE DOS ENTIDADES.



LOGIN && REGISTER

INICIAR SESION

REGISTRARTE

Genero : ☐ Female ☐ Male

Edad :

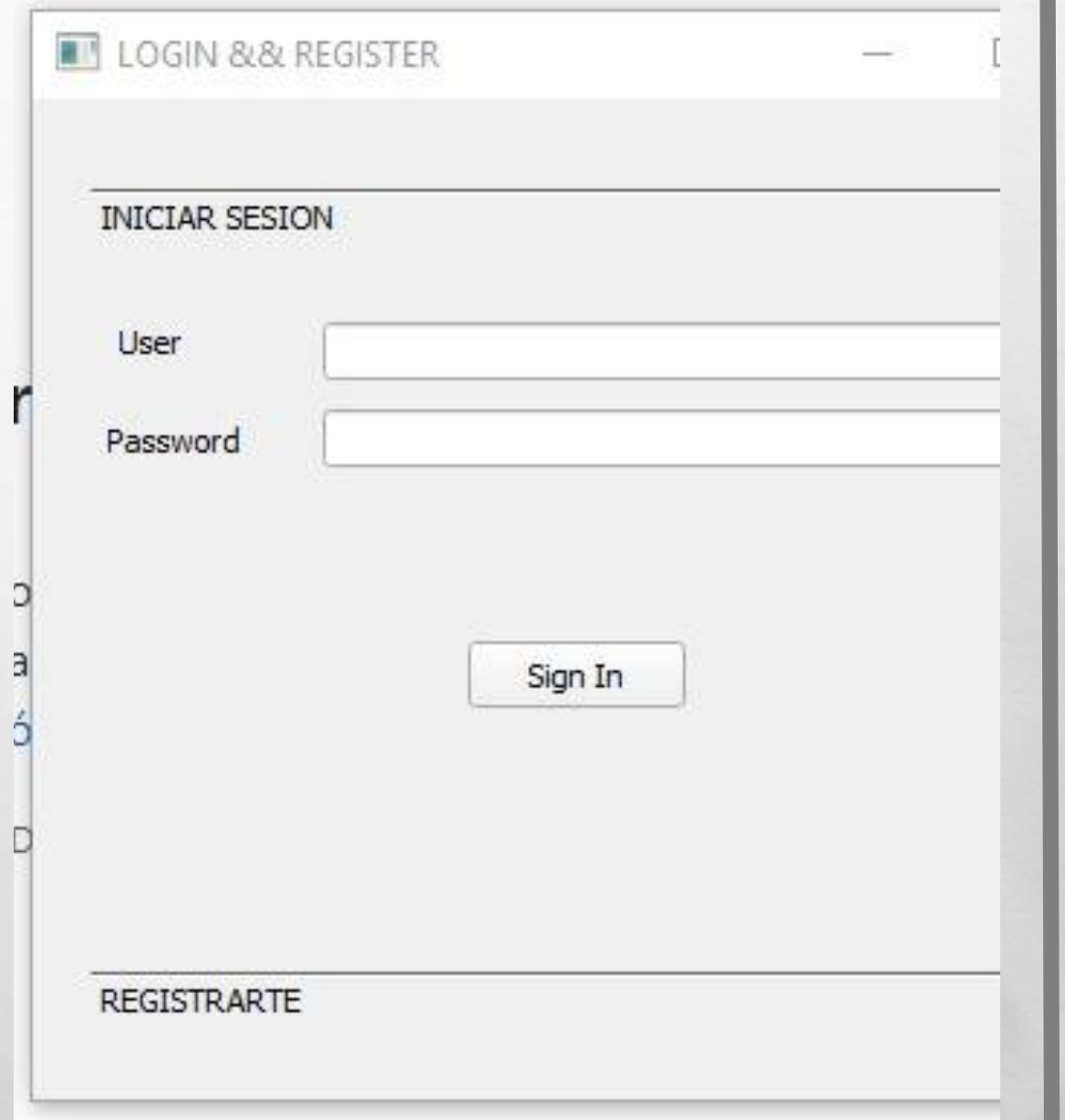
Ocupacion :

Password

Register

# REGISTRO DE USUARIOS

# INICIO DE SESION



LOGIN && REGISTER

---

INICIAR SESION

User

Password

Sign In

---

REGISTRARTE

# MODULOS



MODULOS	REQUISITOS	ENCARGADO	ESTILOS
Implementar el registro de usuario	Registrar Usuario	<ul style="list-style-type: none"> <li>García Díaz, German Flavio</li> </ul>	<ul style="list-style-type: none"> <li>Letterbox</li> <li>Constructivist</li> <li>Spreadsheet</li> </ul>
Implementar la validación de Usuario	Ingresar al Sistema	<ul style="list-style-type: none"> <li>Cueva Flores, Jonathan Brandon</li> </ul>	<ul style="list-style-type: none"> <li>Bulletin Board</li> <li>Aspects</li> </ul>
Implementar la ventana del perfil de Usuario	Visualizar perfil y avatar de usuario	<ul style="list-style-type: none"> <li>Barrios Cornejo, Selene</li> </ul>	<ul style="list-style-type: none"> <li></li> </ul>
Implementar el módulo de Evaluación	Realizar Autoevaluación	<ul style="list-style-type: none"> <li>García Díaz, German Flavio</li> </ul>	<ul style="list-style-type: none"> <li>Letterbox</li> <li>Constructivist</li> <li>Aspects</li> </ul>
Recopilación de tutoriales y fuentes externas	Reproducir tutoriales del tema	<ul style="list-style-type: none"> <li>Barrios Cornejo, Selene</li> </ul>	<ul style="list-style-type: none"> <li></li> </ul>
Diseño GUI	Personalizar y Obtener accesorios del avatar	<ul style="list-style-type: none"> <li>Montesinos Apaza, Sergio</li> </ul>	<ul style="list-style-type: none"> <li>Complementos</li> </ul>
Recompensas Especiales	Recompensar Logro	<ul style="list-style-type: none"> <li>Herrera Cooper, Miguel Alexander</li> </ul>	<ul style="list-style-type: none"> <li></li> </ul>
Implementación de funcionalidad de recuperación de contraseña	Recuperar Contraseña	<ul style="list-style-type: none"> <li>Cueva Flores, Jonathan Brandon</li> </ul>	<ul style="list-style-type: none"> <li>Bulletin Board</li> <li>Aspects</li> </ul>
Definir e implementar lo módulos de aprendizaje	Teórico	<ul style="list-style-type: none"> <li>Cueva Flores, Jonathan Brandon</li> <li>Montesinos Apaza, Sergio</li> </ul>	<ul style="list-style-type: none"> <li></li> </ul>
Definir los Niveles de dificultad en los módulos	Teórico	<ul style="list-style-type: none"> <li>Montesinos Apaza, Sergio</li> <li>Cueva Flores, Jonathan Brandon</li> </ul>	<ul style="list-style-type: none"> <li></li> </ul>



# **TABLERO DE TAREAS EN TRELLO**



🏠

Tableros

🔍

Trello

+

🕒

🔔

SE

EduKids

🌟

Personal

🔒 Privado

SBC

GGD

JF

MC

SA

Invitar

... Mostrar menú

Por hacer

...

+ Añada una tarjeta

Analisis

...

Recompensas Especiales

☰

MC

Recopilación de tutoriales y fuentes externas.

🔔 1 👁 ☰

SBC

+ Añada otra tarjeta

Desarrollo

...

Diseño e implementación de la tienda

🔔 3 👁 ☰

SA

+ Añada otra tarjeta

Pruebas

...

Implementación de la funcionalidad de recuperación de contraseña

☰

JF

Implementar el módulo de Evaluacion

☰

GGD

+ Añada otra tarjeta

Hecho

...

Validacion del documento de especificacion de requisitos.

Busqueda apropiada del diseño, aplicando metodologías de diseño centrado en el usuario.

Definir los Niveles de dificultad en los módulos

🕒 22 de jun. ☰

SA

Implementar la validación de Usuario

☰

JF

Implementar la ventana del perfil de usuario.

🔔 2 👁 ☰

SBC

Definir e implementar los Módulos de aprendizaje

🕒 20 de jun. ✉ 0/2

JF SA

+ Añada otra tarjeta

# **D.D.D**

**DOMAIN-DRIVEN DESIGN,**



# LENGUAJE UBICUO:

- **PROPUESTA PARA CREAR UN LENGUAJE COMÚN ENTRE LOS PROGRAMADORES Y USUARIOS; PROPONE NOMBRAR**
- **LAS VARIABLES, MÉTODOS Y CLASES CON LENGUAJE DEL DOMINIO DE MODO QUE SEA 'AUTO EXPLICABLE', HABITUALMENTE LOS NOMBRES REPRESENTAN OBJETOS Y LOS VERBOS REPRESENTAN MÉTODOS.**



# CAPAS DE LA ARQUITECTURA

- **USER INTERFACE: PRESENTA LA INFORMACIÓN AL USUARIO, INTERPRETA SUS ACCIONES Y LAS ENVÍA A LA APLICACIÓN.**

```
5 namespace Sergio
6 {
7     class AssetManager
8     {
9     public:
10         AssetManager() {}
11         ~AssetManager() {}
12         void LoadTexture(std::string name, std::string fileName);
13         sf::Texture &GetTexture(std::string name);
14
15         void LoadFont(std::string name, std::string fileName);
16         sf::Font &GetFont(std::string name);
17
18     private:
19         std::map<std::string, sf::Texture> _textures;
20         std::map<std::string, sf::Font> _fonts;
21     };
22 }
23
```

```
2  #include "SFML\Graphics.hpp"
3
4  namespace Sergio
5  {
6      class InputManager
7      {
8      public:
9          InputManager() {}
10         ~InputManager() {}
11
12         bool IsSpriteClicked(sf::Sprite object, sf::Mouse::Button button, sf::RenderWindow &window);
13         sf::Vector2i GetMousePosition(sf::RenderWindow &window);
14     };
15 }
```



```

namespace Sergio
{
    class MainMenuState : public State
    {
    public:
        MainMenuState(GameDataRef data);
        void Init();
        void HandleInput();
        void Update(float dt);
        void Draw(float dt);

    private:
        void AddSprite(std::string title, sf::Vector2i pos);
        void AddSpriteInput(std::string title, sf::Vector2i pos);
        void AddText(std::string font, std::string title, sf::Vector2i pos, sf::Color color, short int Size);
        void AddTextDynamic(std::string font, std::string title, sf::Vector2i pos, sf::Color color, short int Size);
        GameDataRef _data;

        sf::Sprite _background;
        sf::Text _title;
        int sw;
        sf::String Input1;
        sf::Clock _clock;

        std::vector<sf::Sprite> _sprites;
        std::vector<sf::Sprite> _spritesInput;
        std::vector<sf::Text> _texts;
        std::vector<sf::Text> _texts_ag;
    };
}

```



```
#include <SFML/Graphics.hpp>
#include "StateMachine.hpp"
#include "AssetManager.hpp"
#include "InputManager.hpp"
```

```
namespace Sergio
```

```
{
```

```
    class GameData
```

```
    { public:
```

```
        StateMachine machine;
```

```
        sf::RenderWindow window;
```

```
        AssetManager assets;
```

```
        InputManager input;
```

```
    };
```

```
    typedef std::shared_ptr<GameData> GameDataRef;
```

```
    class Game
```

```
    {
```

```
    public:
```

```
        Game(int width, int height, std::string title);
```

```
    private:
```

```
        // Updates run at 60 per second.
```

```
        const float dt = 1.0f / 60.0f;
```

```
        sf::Clock _clock;
```

```
        GameDataRef _data = std::make_shared<GameData>();
```

```
        void Run();
```

```
    };
```

```
}
```



```

4  #include <SFML/Graphics.hpp>
5  #include "StateMachine.hpp"
6  #include "AssetManager.hpp"
7  #include "InputManager.hpp"
8
9  namespace Sergio
10 {
11     class GameData
12     { public:
13         StateMachine machine;
14         sf::RenderWindow window;
15         AssetManager assets;
16         InputManager input;
17     };
18
19     typedef std::shared_ptr<GameData> GameDataRef;
20
21     class Game
22     {
23     public:
24         Game(int width, int height, std::string title);
25
26     private:
27         // Updates run at 60 per second.
28         const float dt = 1.0f / 60.0f;
29         sf::Clock _clock;
30         GameDataRef _data = std::make_shared<GameData>();
31         void Run();
32     };
33 }
34

```

# APPLICATION

- ENCARGADA DE VERIFICAR E INTERACTUAR CON LOS DIFERENTES USUARIOS DURANTE EL USO DE LA APLICACIÓN.

# DOMAIN

- **ES EL NÚCLEO DE LA APLICACIÓN QUE CONTIENE LAS REGLAS DE NEGOCIO Y EL RESPONSABLE DE MANTENER EL ESTADO DE LOS OBJETOS DE NEGOCIO.**
- **EN NUESTRO CASO NUESTRO NÚCLEO ES EL MANEJO DE DATA Y LAS OPERACIONES MATEMÁTICAS QUE LA APLICACIÓN REALIZA.**

# INFRASTRUCTURE

- **CAPA DE SOPORTE PARA EL RESTO DE CAPAS, PROVEE LA COMUNICACIÓN CON LAS OTRAS CAPAS E IMPLEMENTA LAS PERSISTENCIAS DE LOS OBJETOS DE NEGOCIO Y LAS LIBRERÍAS DE SOPORTE PARA LAS OTRAS CAPAS.**
- **INTERCONEXIÓN EN VENTANAS DE SFML Y QT HACIENDO UNA APLICACIÓN CON DIFERENTES CAPAS DE APLICACIÓN.**

# SERVICES

- **EN NUESTRO CASO AL MOMENTO DE REALIZAR “OPERACIONES” LA FUNCIÓN QUE VA A RESOLVER ESTO NO ES LA MISMA, ES DECIR QUE UNA MULTIPLICACIÓN NO PUEDE SER RESUELTA CON RESTAS.**



# MODULES

- **PERMITEN SEGUIR CON FACILIDAD EL CONCEPTO DE ACOPLAMIENTO DÉBILES Y ALTA COHESIÓN.**
- **CONTAMOS CON DISTINTOS MÓDULOS:**
- **REGISTRO DE USUARIO**
- **INTERFAZ DE LA APP**
- **VERIFICACIÓN DE DATOS**
- **REALIZACIÓN** **DE** **OPERACIONES**

# AGGREGATES

- **AL GENERAR UN INICIO DE SESIÓN POR PARTE DEL USUARIO ESTE SOLO VISUALIZA PARTE DE SUS INFORMACIONES EL RESTO SIGUE ESTANDO OCULTO DEL RESTO DE LOS USUARIOS.**

# FACTORIES

- **EN NUESTRA APLICACIÓN APLICAMOS FACTORÍA EN EL MOMENTO QUE EL USUARIO INTERACTÚA CON LA APLICACIÓN, YA SEA EN EL MOMENTO QUE INICIE SESIÓN O REALICE OPERACIONES.**

# REPOSITORIES

- **LA APLICACIÓN NECESITA MANTENER UN PUNTERO A CADA OBJETO O ENTIDAD CREADA. RESPONSABLE DE PROPORCIONAR A LA APLICACIÓN A ESOS PUNTEROS.**



# ARQUITECTURA DEL SOFTWARE



