```
using SplashKitSDK;
using System.Collections.Generic;
using static DragonBallGame.RecruitSystem;
using DragonBallGame.CharacterClass;

namespace DragonBallGame
{
    public class GameMenu
    {
        private Window _menuWindow;
        private int _currentCharacterIndex;
        private ImageManager _imageManager;
        private Player _player;
        private InputHandler _inputHandler;
        private BattleManager _battleManager;
        private MessageManager _messageManager;

        public GameMenu(Player player)
        {
            _menuWindow = new Window("Dragon Ball Game", 800, 600);
            _player = player;
            _imageManager = new ImageManager();
            _inputHandler = new InputHandler();
            _battleManager = new BattleManager(_player, _menuWindow);
            _messageManager = new MessageManager();

            // Player starts with only Son Goku
            _player.AddRecruitedCharacter(new Goku());

            _currentCharacterIndex = 0;
        }

        public void Run()
        {
            while (!_menuWindow.CloseRequested)
            {
                SplashKit.ProcessEvents();
                SplashKit.ClearScreen(Color.White);

                DrawCharacterInfo();
                _messageManager.DrawMessages();

                if (SplashKit.MouseClicked(MouseButton.LeftButton))
                {
                    Point2D mousePosition = SplashKit.MousePosition();
                    _inputHandler.HandleMouseClick(mousePosition, this);
                }

                SplashKit.RefreshScreen(60);
            }

            _menuWindow.Close();
        }

        private void DrawCharacterInfo()
        {
            if (_player.RecruitedCharacters.Count > 0)
            {
                Character currentCharacter = _player.RecruitedCharacters[_currentCharacterIndex];
                Bitmap characterImage = _imageManager.GetCharacterImage(currentCharacter);

                // Draw Character Image
                SplashKit.DrawBitmap(characterImage, 150, 50);

                // Draw Character Stats
                SplashKit.DrawText($"{currentCharacter.Name}", Color.Black, 450, 150);
                SplashKit.DrawText($"Power: {currentCharacter.Power}", Color.Black, 450, 180);
                SplashKit.DrawText($"Health: {currentCharacter.Health}", Color.Black, 450, 210);
                SplashKit.DrawText($"Evolution: {currentCharacter.Form}", Color.Black, 450, 240);
                SplashKit.DrawText($"Skill: {currentCharacter.SpecialAbility}", Color.Black, 450, 270);
            }

            // Draw Arrow Buttons
            SplashKit.FillRectangle(Color.LightGray, 50, 250, 40, 40); // Left Arrow
            SplashKit.FillRectangle(Color.LightGray, 710, 250, 40, 40); // Right Arrow
            SplashKit.DrawText("<", Color.Black, 65, 265);
            SplashKit.DrawText(">", Color.Black, 725, 265);

            // Draw Recruit and Battle Buttons
            SplashKit.FillRectangle(Color.LightGray, 250, 550, 100, 40);
            SplashKit.DrawText("Recruit", Color.Black, 275, 565);
            SplashKit.FillRectangle(Color.LightGray, 450, 550, 100, 40);
            SplashKit.DrawText("Battle", Color.Black, 475, 565);

            // Draw Zeni
            SplashKit.DrawText($"Zeni: {_player.Zeni}", Color.Black, 650, 50);
        }

        public void NavigateLeft()
        {
            _currentCharacterIndex--;
            if (_currentCharacterIndex < 0) _currentCharacterIndex = _player.RecruitedCharacters.Count - 1;
        }

        public void NavigateRight()
        {
            _currentCharacterIndex++;
            if (_currentCharacterIndex >= _player.RecruitedCharacters.Count) _currentCharacterIndex = 0;
        }

        public void RecruitCharacter()
        {
            RecruitResult result = RecruitSystem.Instance.RecruitRandomCharacter(_player);

            switch (result.Status)
            {
                case RecruitStatus.NotEnoughZeni:
                    _messageManager.AddMessage("Not enough Zeni.");
                    break;

                case RecruitStatus.NoAvailableCharacters:
                    _messageManager.AddMessage("All characters are at their highest form and cannot be recruited further.");
                    break;

                case RecruitStatus.NewCharacterRecruited:
                    _player.AddRecruitedCharacter(result.Character);
                    _messageManager.AddMessage($"{result.Character.Name} recruited!");
                    break;

                case RecruitStatus.CharacterEvolved:
                    _messageManager.AddMessage($"{result.Character.Name} has transformed into {result.Character.Form}!");
                    break;
            }
        }

        public void Battle()
        {
            if (_player.RecruitedCharacters.Count == 0) return;

            Character currentCharacter = _player.RecruitedCharacters[_currentCharacterIndex];
            string battleResult = _battleManager.StartBattle(currentCharacter);
            _messageManager.AddMessage($"{battleResult}");
        }
    }
}
```