Serenitune Deployment Summary

Slide 1: Introduction

Deploying Serenitune: A Quick Guide

This presentation summarizes the key steps to deploy the Serenitune full-stack web application using Docker and Docker Compose.

Slide 2: Prerequisites

Before You Begin

Ensure you have the following installed on your deployment machine:

- **Docker**: For containerization of the application components.
- **Docker Compose**: For defining and running multi-container Docker applications.
- **Git** (Optional): For cloning the repository.

Slide 3: Step 1: Get the Code

Clone or Download the Repository

Obtain the Serenitune source code:

```
 git clone <repository-url>
cd serenitune
```

If you downloaded a zip, extract it and navigate into the `serenitune` directory.

Slide 4: Step 2: Configure Environment Variables

Set Up Your Environment

Navigate to the `deployment` directory and create a `.env` file from the example:

```
cd deployment
cp .env.example .env
```

Edit the `.env` file to configure: - `SECRET_KEY`: A strong, unique secret key. - `FLASK_ENV`: Set to `production`. - `DATABASE_URL`: Database connection string (SQLite by default, or PostgreSQL).

Slide 5: Step 3: Build and Start Containers

Launch the Application

From the `deployment` directory, execute the Docker Compose command:

```
docker-compose up -d
```

This command will: - Build the Docker images for both frontend and backend. - Start the containers in detached mode (runs in the background).

Slide 6: Step 4: Verify Deployment

Check if Everything is Running

Confirm that your containers are active:

```
docker-compose ps
```

Access the application: - **Frontend**: Typically `http://localhost` (or your server's IP/domain). - **Backend API**: `http://localhost:5000/api`.

Slide 7: Step 5: Initialize Database (First-time)

Seed Your Database

For the initial setup, you need to populate the database:

```
docker-compose exec backend python seed_database.py
```

This command runs the database seeding script inside the backend container.

Slide 8: Stopping the Application

Shutting Down Serenitune

To stop and remove the running containers:

```
docker-compose down
```

Slide 9: Deployment Options & Considerations

Beyond Localhost

- **VPS/Cloud Providers**: DigitalOcean, AWS EC2, Linode.
- **Container Platforms**: Heroku, AWS Elastic Beanstalk, Google Cloud Run.
- **Database**: SQLite for simple setups, PostgreSQL recommended for production.
- **SSL/TLS**: Essential for production (Nginx with Let's Encrypt, cloud load balancers).
- **Monitoring**: Set up logging and health checks.

Slide 10: Conclusion

You're Ready to Deploy!

By following these steps, you can successfully deploy your Serenitune application.

For more detailed information, refer to the `DEPLOYMENT_GUIDE.md` file in the `deployment` directory.