# 🔄 Обновление Solar Recorder: v1.1.0 → v1.1.1

**Исправление критических ошибок и улучшение производительности**

---

## 📋 Что исправлено в версии 1.1.1

### 🔴 Решены критические проблемы:

1. **Режим Whisper CPU** ✅
   - Исправлен сбой зависимости CUDA в Docker.
   - Добавлен явный `FORCE_CPU=true` параметр
   - Модель теперь надежно загружается в контейнерных средах.
2. **Согласованность переменных среды** ✅
   - Единый `MODEL` (был `WHISPER_MODEL`)
   - Единый `CORS_ORIGINS` (был `ALLOWED_ORIGINS`)
   - Все переменные теперь имеют единообразное наименование.
3. **Расширенный сценарий настройки** ✅
   - Автоматическое обнаружение Docker
   - Красивый цветной вывод
   - Автоматически генерирует `.env` файл
   - Показывает все URL-адреса после запуска
4. **Улучшения Docker Compose** ✅
   - Правильные зависимости проверки работоспособности
   - Явные переменные среды
   - Лучшая организация обслуживания

---

## 🚀 Быстрое обновление (рекомендуемый метод)

### Шаг 1: Остановите текущие услуги

```
баш

cd ~/solar-recorder  # or your project directory
docker compose down
```

### Шаг 2: Резервное копирование текущих файлов (безопасность прежде всего)

```
баш

# Backup current configuration
cp backend/.env backend/.env.backup
cp backend/transcribe.py backend/transcribe.py.v110
cp backend/main.py backend/main.py.v110
cp docker-compose.yml docker-compose.yml.v110
```

### Шаг 3: Заменить файлы

Замените следующие файлы новыми версиями:

**Внутренние файлы:**

- `backend/transcribe.py` → **Использовать новую версию из artifacts**
- `backend/main.py` → **Использовать новую версию из artifacts**

**Файлы конфигурации:**

- `docker-compose.yml` → **Использовать новую версию из artifacts**
- `setup.sh` → **Использовать новую версию из artifacts**
- **СОЗДАТЬ НОВЫЙ** : `backend/.env.example` → **Использовать новую версию из артефактов**

### Шаг 4: обновление переменных среды

Редактировать `backend/.env`:

```
баш

# OLD (v1.1.0) - REMOVE THESE:
# WHISPER_MODEL=base
# ALLOWED_ORIGINS=http://localhost:3000

# NEW (v1.1.1) - ADD THESE:
MODEL=base
CORS_ORIGINS=http://localhost:3000
FORCE_CPU=true

# Optional (keep if you have it):
DEEPSEEK_API_KEY=your_key_here
```

### Шаг 5: Перезапустите службы

```
баш
```

```
# Make setup.sh executable
chmod +x setup.sh

# Run the new installer
./setup.sh
```

---

## 📝 Обновление вручную (при желании)

### Изменения по файлам:

**1. `backend/transcribe.py`**

**Изменить строку 5-7:**

```
питон

# OLD:
MODEL_SIZE = os.getenv("WHISPER_MODEL", "base")
model = None

# NEW:
MODEL = os.getenv("MODEL", "base")
FORCE_CPU = os.getenv("FORCE_CPU", "true").lower() == "true"
model = None
```

**Изменение `get_model()` функции:**

```
питон
```

```
# OLD:
def get_model():
    global model
    if model is None:
        print(f"Loading Whisper model: {MODEL_SIZE}")
        model = whisper.load_model(MODEL_SIZE)
    return model

# NEW:
def get_model():
    global model
    if model is None:
        device = "cpu" if FORCE_CPU else ("cuda" if torch.cuda.is_available() else "cpu")
        print(f"🎙️ Loading Whisper model: {MODEL} on device: {device}")
        try:
            model = whisper.load_model(MODEL, device=device)
            print(f"✅ Whisper model '{MODEL}' loaded successfully on {device}")
        except Exception as e:
            print(f"❌ Failed to load Whisper model: {e}")
            raise
    return model
```

**Добавьте импорт вверху:**

питон

```
import torch  # Add this line
```

**2. backend/main.py**

**Изменить строку 17:**

питон

```
# OLD:
ALLOWED_ORIGINS = os.getenv("ALLOWED_ORIGINS", "http://localhost:3000").split(",")

# NEW:
CORS_ORIGINS = os.getenv("CORS_ORIGINS", "http://localhost:3000").split(",")
```

**Обновление промежуточного программного обеспечения CORS (строка 19):**

питон

```
# OLD:
app.add_middleware(
    CORSMiddleware,
    allow_origins=ALLOWED_ORIGINS,

# NEW:
app.add_middleware(
    CORSMiddleware,
    allow_origins=CORS_ORIGINS,
```

**Обновить строку версии (строка 14):**

питон

```
# OLD:
app = FastAPI(title="Solar Recorder API", version="1.1.0")

# NEW:
app = FastAPI(title="Solar Recorder API", version="1.1.1")
```

**3. docker-compose.yml**

**Заменить раздел внутренней среды:**

yaml

```
# OLD:
environment:
  - WHISPER_MODEL=${WHISPER_MODEL:-base}
  - DEEPSEEK_API_KEY=${DEEPSEEK_API_KEY}
  - ALLOWED_ORIGINS=http://localhost:3000,http://localhost:3001

# NEW:
environment:
  - MODEL=${MODEL:-base}
  - DEEPSEEK_API_KEY=${DEEPSEEK_API_KEY:-}
  - CORS_ORIGINS=${CORS_ORIGINS:-http://localhost:3000}
  - FORCE_CPU=true
  - PYTHONUNBUFFERED=1
```

**Обновление интерфейса зависит_от:**

yaml

```
# OLD:
depends_on:
  - backend

# NEW:
depends_on:
  backend:
    condition: service_healthy
```

**4. Создать `backend/.env.example`**

Создайте новый файл с таким содержимым:

баш

```
# Solar Recorder v1.1.1 Configuration

# Whisper Model (tiny/base/small/medium/large)
MODEL=base

# DeepSeek API Key (optional - for translation)
DEEPSEEK_API_KEY=

# CORS Origins (comma-separated)
CORS_ORIGINS=http://localhost:3000

# Server Configuration
HOST=0.0.0.0
PORT=8000

# Force CPU mode (recommended for Docker)
FORCE_CPU=true

# Upload Settings
MAX_FILE_SIZE=500MB
```

---

## ✅ Шаги проверки

После обновления проверьте, что все работает:

### 1. Проверьте, запущены ли службы.

баш

```
docker compose ps
```

Обе службы должны отображаться как «Работающие (здоровые)».

### 2. Тестирование конечной точки работоспособности

```
баш

curl http://localhost:8000/health
```

Должен вернуть:

```
json

{
  "status": "healthy",
  "model": "base",
  "force_cpu": "true",
  ...
}
```

### 3. Проверьте журналы

```
баш

# Backend logs - look for:
docker compose logs backend | grep "Whisper"
# Should show: "Whisper model 'base' loaded successfully on cpu"

# No CUDA errors!
docker compose logs backend | grep "CUDA"
# Should return nothing
```

### 4. Тестовая запись

1. Откройте http://localhost:3000
2. Запишите короткое видео
3. Проверьте, что транскрипция выполнена без ошибок.

---

## 🔧 Устранение неполадок

### Проблема: ошибка «CUDA недоступна»

**Решение:** Убедитесь `FORCE_CPU=true`, что установлено в docker-compose.yml

### Проблема: «ALLOWED_ORIGINS» не распознан

**Решение:** Обновить `CORS_ORIGINS` в обоих случаях `.env` и `main.py`

### Проблема: службы не запускаются

**Решение:**

```
баш

# Clean rebuild
docker compose down -v
docker compose up --build
```

### Проблема: транскрипция по-прежнему не работает

**Решение:** Проверьте, правильно ли загружается модель:

```
баш

docker compose logs backend | grep -A 5 "Loading Whisper"
```

---

## 📊 Сводка изменений

| Компонент | Старый (v1.1.0) | Новое (v1.1.1) | Статус |
|---|---|---|---|
| Устройство шепота | Авто (CUDA/ЦП) | Явный ЦП | ✅ Исправлено |
| ENV: Модель | `WHISPER_MODEL` | `MODEL` | ✅ Унифицированный |
| ОКРУЖЕНИЕ: КОРС | `ALLOWED_ORIGINS` | `CORS_ORIGINS` | ✅ Унифицированный |
| CPU Force | Не установлено | `FORCE_CPU=true` | ✅ Добавлено |
| setup.sh | Базовый | Улучшенный | ✅ Улучшено |
| Проверка здоровья | Базовый | Подробный | ✅ Улучшенный |

---

## 🎯 Контрольный список после обновления

- [ ] Все службы работают ( `docker compose ps` )
- [ ] Проходит проверку здоровья ( `curl http://localhost:8000/health` )
- [ ] Ошибок CUDA в журналах нет.
- [ ] Запись произведений
- [ ] Транскрипция завершена
- [ ] PDF генерируется правильно
- [ ] Перевод работает (если установлен ключ API)

- [ ] Список файлов отображается правильно.

---

## 📞 Поддержка

Если у вас возникнут какие-либо проблемы во время обновления:

1. Проверьте логи: `docker