Инструкция по установке и запуску ПО «Beerжевик»

1. Назначение документа

Настоящий документ содержит инструкцию по установке и запуску программного обеспечения «Beerжевик» в инфраструктуре заказчика (on-premise).

2. Что входит в поставку

- архив с исходным кодом и конфигурационными файлами Docker Compose;
- файлы шаблонов переменных окружения (.env.example, .env-dev, .env-dokploy-template);
- инструкция по развертыванию (настоящий документ).

3. Системные требования

- ОС: Linux (рекомендуется Ubuntu 20.04 LTS и выше);
- CPU: от 2 ядер; RAM: от 4 ГБ (рекомендуется 8 ГБ);
- Диск: от 20 ГБ;
- Docker Engine (Docker CE) 20.10+ и Docker Compose v2+.

4. Установка (выполняет заказчик или специалисты правообладателя)

Если установку выполняют специалисты правообладателя, заказчик предоставляет доступ к серверу (SSH) и сведения для настройки домена/сертификата. При этом все шаги ниже выполняются теми же командами, разница только в исполнителе.

Экземпляр ПО «Beerжевик» предоставляется пользователю (клиенту) после приобретения ПО.

4.1. Установка Docker и Docker Compose

Docker может быть установлен, например, с использованием команд, приведенных ниже. Пример для Ubuntu/Debian:

```
curl -fsSL https://get.docker.com -o get-docker.sh
sudo sh get-docker.sh
sudo usermod -aG docker $USER
newgrp docker
docker --version
docker compose version
```

4.2. Распаковка дистрибутива

```
sudo mkdir -p /opt/beerzhevik
sudo chown $USER:$USER /opt/beerzhevik
cd /opt/beerzhevik
tar -xzf beerzhevik_v1.0.tar.gz
```

4.3. Настройка переменных окружения

Для развертывания ПО через docker-compose-dev.yaml используется файл .env-dev.
Рекомендуется сформировать его на основе .env.example и заполнить актуальными значениями.

```
cp .env.example .env-dev
nano .env-dev
```

4.4. Подготовка сети Docker

Если в docker-compose файле указана внешняя сеть, создайте ее:

```
docker network create app_main || true
```

4.5. Запуск сервисов

Запустите сервисы (docker-compose-dev.yaml, включает PostgreSQL и Redis):

Примечание: несмотря на название, файл docker-compose-dev.yaml используется для развертывания полного стека (включая PostgreSQL) в инфраструктуре заказчика.

```
docker compose -f docker-compose-dev.yaml up -d --build
```

4.6. Инициализация и миграции БД

После запуска выполните миграции:

```
docker compose -f docker-compose-dev.yaml exec app alembic upgrade head
```

4.7. Проверка запуска

```
docker compose -f docker-compose-dev.yaml ps
curl -fsSL http://localhost:8000/v1/healthcheck/
```

5. Развертывание клиентских веб-приложений

Клиентские части «Beergжевик» представлены веб-приложениями (PWA) и запускаются отдельно от серверной части. Критически важные компоненты: Client App (для посетителей), Waiter App (для персонала), TV Banners App (для ТВ-панелей).

5.1. Общие требования

- для запуска через Docker: установлен Docker и Docker Compose v2+;
- для запуска без Docker: установлен Node.js (LTS) и npm;
- для всех клиентских приложений необходимо задать BACKEND_URL (адрес API).

5.2. Настройка BACKEND_URL

Переменная BACKEND_URL должна указывать на адрес API в формате http(s)://<домен-или-ip>:<порт>/v1. Пример без внешней маршрутизации: http://<адрес-сервера>:8000/v1

5.3. Развертывание Client App (для посетителей)

Вариант A (рекомендуется): запуск через Docker Compose.

1. Перейдите в директорию Client App.
2. Создайте файл .env на основе .env.example и укажите BACKEND_URL.
3. При необходимости опубликуйте порт наружу (ports).
4. Запустите контейнер.

```
cd clientapp
cp .env.example .env
nano .env
# BACKEND_URL=http://<адрес-сервера>:8000/v1
docker network create app_main || true
docker compose up -d --build
```

Примечание: в docker-compose.yml используется директива expose. Чтобы открыть доступ снаружи, добавьте в сервис секцию ports, например:

```
ports:
  - "3000:3000"
```

Вариант B: запуск через npm (без Docker).

```
cd clientapp
cp .env.example .env
nano .env
npm ci
npm run build
npm run start
```

5.4. Развертывание Waiter App (для персонала)

Waiter App использует базовый путь /waiter/ в production-конфигурации.

Вариант A (рекомендуется): запуск через Docker Compose.

5. Перейдите в директорию Waiter App.
6. Создайте файл .env и укажите BACKEND_URL.
7. При необходимости опубликуйте порт наружу (ports).
8. Запустите контейнер.

```
cd waiter
cat > .env <<'EOF'
BACKEND_URL=http://<адрес-сервера>:8000/v1
EOF
docker network create app_main || true
docker compose up -d --build
```

Для доступа снаружи добавьте ports, например:

```
ports:
  - "3100:3100"
```

URL для доступа: http://<адрес-сервера>:3100/waiter/

Вариант B: запуск через npm (без Docker).

```
cd waiter
cat > .env <<'EOF'
BACKEND_URL=http://<адрес-сервера>:8000/v1
EOF
npm ci
npm run build
PORT=3100 HOST=0.0.0.0 npm run start
```

5.5. Развертывание TV Banners App (для ТВ-панелей)

TV Banners App использует базовый путь /banners/ в production-конфигурации.

Вариант A (рекомендуется): запуск через Docker Compose.

9. Перейдите в директорию TV Banners App.
10. Создайте файл .env на основе .env.example и укажите BACKEND_URL.
11. При необходимости опубликуйте порт наружу (ports).
12. Запустите контейнер.

```
cd tvbanners
cp .env.example .env
nano .env
```

```
# BACKEND_URL=http://<адрес-сервера>:8000/v1
docker network create app_main || true
docker compose up -d --build
```

Для доступа снаружи добавьте ports, например:

```
ports:
  - "3200:3200"
```

URL для доступа: http://<адрес-сервера>:3200/banners/

Вариант B: запуск через npm (без Docker).

```
cd tvbanners
cp .env.example .env
nano .env
npm ci
npm run build
PORT=3200 HOST=0.0.0.0 npm run preview
```

5.6. Проверка работоспособности (клиентские части)

- Client App: http://<адрес-сервера>:3000/
- Waiter App: http://<адрес-сервера>:3100/waiter/
- TV Banners App: http://<адрес-сервера>:3200/banners/
- Проверьте, что приложения загружают меню и выполняют запросы к API (BACKEND_URL).

6. Типовые проблемы и решения

- Сервисы не запускаются: проверьте логи `docker compose ... logs`, корректность .env и доступность внешних сервисов (например, БД).
- Порт занят: измените проброс портов и/или конфигурацию внешней маршрутизации (если используется).
- Ошибка миграций: убедитесь, что БД доступна и переменная DATABASE_URL указана корректно.

7. Обновление версии

13. Остановите сервисы: `docker compose -f docker-compose-dev.yaml down`.
14. Распакуйте новый дистрибутив поверх (или в новую директорию).
15. Запустите сервисы: `docker compose -f docker-compose-dev.yaml up -d --build`.
16. Выполните миграции: `docker compose -f docker-compose-dev.yaml exec app alembic upgrade head`.