LMS "ЦУМ: Корпоративный университет" - Обзор проекта

Версия документа: 1.0
Дата обновления: 22 июля 2025
Текущий Sprint: 52
День проекта: 171

📋 Оглавление

🎯 О проекте

LMS ЦУМ - корпоративная система обучения для сотрудников ЦУМ (Центральный Универсальный Магазин). Система предназначена для:

- Онбординга новых сотрудников
- Повышения квалификации персонала
- Управления компетенциями
- Проведения аттестаций
- Отслеживания прогресса обучения

Ключевые особенности:

- 📱 Нативное iOS приложение (основной канал)
- 🏗️ Микросервисная архитектура backend
- 🎯 Domain-Driven Design
- 🧪 Test-Driven Development (обязательно!)
- 🤖 LLM-оптимизированная разработка

🛠️ Технологический стек

iOS приложение

- **Язык**: Swift 5.9+
- **UI Framework**: SwiftUI
- **Минимальная версия**: iOS 17.0
- **Архитектура**: Clean Architecture + MVVM-C
- **Зависимости**: SPM (Swift Package Manager)

- **Тестирование**: XCTest, ViewInspector

Backend

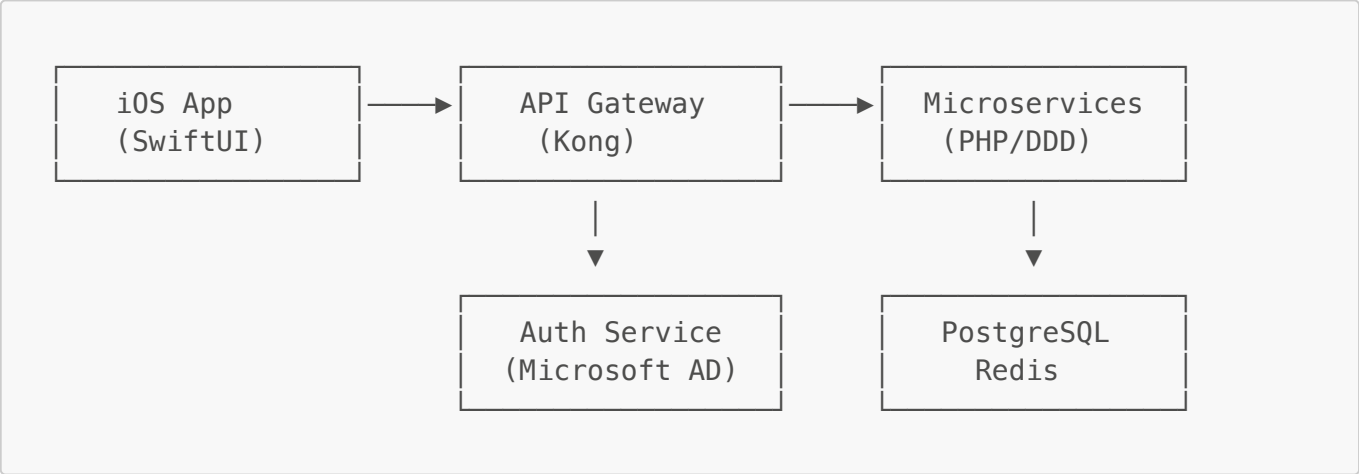
- **Язык**: PHP 8.1+
- **Framework**: Symfony 7.0 / Laravel (гибридный подход)
- **База данных**: PostgreSQL 15+ (основная), Redis (кэш)
- **Очереди**: RabbitMQ
- **API**: RESTful + OpenAPI 3.0
- **Контейнеризация**: Docker, Kubernetes

Инфраструктура

- **CI/CD**: GitHub Actions
- **Мониторинг**: Prometheus + Grafana
- **Логирование**: ELK Stack
- **API Gateway**: Kong
- **Хостинг**: Railway.app (production)

🏗️ Архитектура системы

Общая схема



Микросервисы (текущий статус)

Сервис	Статус	Описание
AuthService	✅ Production	Аутентификация, JWT, интеграция с AD
UserService	✅ Production	Управление пользователями и ролями
CourseService	🚧 Разработка	Управление курсами и модулями
CompetencyService	🚧 Разработка	Матрица компетенций
NotificationService	📋 Планируется	Push-уведомления
OrgStructureService	📋 Планируется	Оргструктура компании

iOS архитектура

```
LMS/
├── Features/           # Feature modules (Vertical Slices)
│   ├── Auth/          # ✅ Готово
│   ├── Onboarding/    # ✅ Готово
│   ├── Feed/          # ✅ Готово (Telegram-style)
│   ├── Courses/       # ✅ Готово
│   ├── CourseManagement/ # ✅ Готово
│   ├── Competencies/  # ✅ Готово
│   ├── OrgStructure/  # ✅ Готово
│   ├── Cmi5/          # ✅ Готово
│   └── Scorm/         # 🚧 В разработке
├── Services/          # Бизнес-логика
├── Core/              # Общие компоненты
└── Resources/         # Ассеты и конфигурация
```

📊 Текущий статус разработки

Прогресс по модулям

Модуль	iOS	Backend	Интеграция	Тесты
Аутентификация	✅ 100%	✅ 100%	✅ 100%	✅ 95%
Онбординг	✅ 100%	✅ 100%	✅ 100%	✅ 98%
Лента новостей	✅ 100%	✅ 100%	✅ 100%	✅ 92%
Курсы (просмотр)	✅ 100%	✅ 100%	✅ 100%	✅ 90%
Управление курсами	✅ 100%	🚧 70%	🚧 60%	✅ 88%
Компетенции	✅ 100%	🚧 80%	🚧 70%	✅ 85%
CMI5 контент	✅ 100%	✅ 100%	✅ 100%	✅ 90%
SCORM контент	🚧 30%	📋 0%	📋 0%	🚧 20%

Ключевые достижения

- ✅ 200+ сборок в TestFlight
- ✅ 15 основных модулей реализовано
- ✅ 85%+ покрытие тестами
- ✅ Vertical Slice архитектура
- ✅ Полная система логирования
- ✅ Feedback система встроена

В работе (Sprint 52)

- 🚧 CourseService микросервис

- 🚧 CompetencyService микросервис
- 🚧 SCORM импорт
- 🚧 Kubernetes манифесты

📁 Структура проекта

```
lms_docs/
├── LMS_App/LMS/         # iOS приложение
│   ├── LMS/             # Исходный код
│   ├── LMSTests/        # Unit тесты
│   ├── LMSUITests/      # UI тесты
│   └── scripts/         # Автоматизация
├── src/                 # Backend код (DDD)
│   ├── Auth/            # Bounded Context
│   ├── User/
│   ├── Course/
│   └── Competency/
├── tests/               # Backend тесты
├── reports/             # Отчеты о разработке
│   ├── sprints/         # Sprint планы и отчеты
│   ├── daily/           # Ежедневные отчеты
│   └── methodology/     # Обновления методологии
├── technical_requirements/ # Техническая документация
└── docs/                # Проектная документация
```

🔄 Методология разработки

TDD (Test-Driven Development) - ОБЯЗАТЕЛЬНО!

```
1. RED: Написать тест → Запустить → Увидеть красный
2. GREEN: Написать минимальный код → Тест зеленый
3. REFACTOR: Улучшить код → Тест остается зеленый
4. LOG: Добавить логирование
```

⚠ **КРИТИЧНО**: Код без запущенных тестов = код не существует!

Vertical Slice подход

Каждый Sprint = готовый функционал от UI до БД:

- iOS UI с анимациями
- Backend API endpoints
- База данных с миграциями
- Тесты (>90% покрытие)
- TestFlight релиз

LLM-оптимизированная разработка

- Файлы до 150 строк (оптимально 50-100)
- Один класс = один файл
- Методы до 30 строк
- Четкие имена и структура

🚀 Быстрый старт

Для iOS разработчиков

```
# 1. Клонировать репозиторий
git clone <repository-url>
cd lms_docs/LMS_App/LMS

# 2. Открыть в Xcode
open LMS.xcodeproj

# 3. Запустить тесты
./scripts/test-quick-ui.sh

# 4. Собрать и запустить
Cmd+R в Xcode
```

Для Backend разработчиков

```
# 1. Запустить Docker
docker-compose up -d

# 2. Установить зависимости
composer install

# 3. Запустить миграции
php bin/console doctrine:migrations:migrate

# 4. Запустить тесты
./test-quick.sh
```

Полезные скрипты

```
# iOS
./scripts/test-quick-ui.sh          # Быстрый запуск UI тестов
./scripts/run-tests-with-timeout.sh # Тесты с таймаутом
./scripts/check_integration.sh      # Проверка интеграции

# Backend
./test-quick.sh                     # Быстрый запуск PHP тестов
./scripts/report.sh                 # Генерация отчетов
```

```
# Общие
./sync-methodology.sh                # Синхронизация методологии
```

📱 TestFlight

Текущая версия: 2.3.0 (Build 207)

Новые сборки выпускаются каждый Sprint (еженедельно).

Как получить доступ:

1. Отправить Apple ID менеджеру проекта
2. Принять приглашение в TestFlight
3. Установить приложение
4. Тестировать и отправлять feedback

📚 Документация

Обязательно к прочтению:

1. `.cursorrules` - правила для AI-ассистентов
2. `technical_requirements/TDD_MANDATORY_GUIDE.md` - гайд по TDD
3. `technical_requirements/llm_development_guide.md` - разработка с LLM
4. `docs/ARCHITECTURE_V3.md` - архитектура системы

Технические спецификации:

- `docs/api/*.yaml` - OpenAPI спецификации
- `technical_requirements/v1.0/*.md` - требования MVP
- `docs/releases/*.md` - история релизов

🔧 Инструменты разработки

Рекомендуемые:

- **IDE**: Xcode 15.4+ (iOS), PHPStorm (Backend)
- **API клиент**: Postman / Insomnia
- **Git GUI**: SourceTree / GitKraken
- **DB клиент**: TablePlus / DBeaver

Системы:

- **Задачи**: GitHub Issues
- **CI/CD**: GitHub Actions
- **Мониторинг**: Grafana dashboards
- **Логи**: Встроенная система + ELK

🤝 Команда и контакты

Роли в проекте:

- **Product Owner**: Определяет требования
- **iOS разработчики**: Основной фронт
- **Backend разработчики**: API и сервисы
- **QA**: Тестирование через TestFlight
- **DevOps**: Инфраструктура и CI/CD

Коммуникация:

- Daily standup: 10:00 MSK
- Sprint Review: Пятница
- Документация: На русском языке
- Код: Комментарии на английском

⚡ Важные правила

1. **TDD обязателен** - сначала тест, потом код
2. **Все тесты должны проходить** - 100%, без исключений
3. **Логирование обязательно** - каждое действие логируется
4. **Feature Registry** - все модули регистрируются
5. **Vertical Slice** - полный функционал за Sprint
6. **TestFlight каждый Sprint** - обязательный релиз

🎓 Обучающие материалы

Для новичков:

1. Изучить примеры в `/examples/`
2. Прочитать antipatterns.md
3. Посмотреть существующие тесты
4. Начать с простой задачи

Продвинутый уровень:

1. Clean Architecture patterns
2. DDD tactical patterns
3. Микросервисные паттерны
4. Performance optimization

🚨 Куда обращаться

При проблемах:

1. Проверить существующие Issues
2. Поискать в документации
3. Спросить в чате команды
4. Создать новый Issue

Для предложений:

1. Обсудить с командой

2. Создать RFC документ
3. Получить одобрение
4. Реализовать через TDD

Добро пожаловать в команду LMS ЦУМ! 🎉

Мы создаем современную систему обучения, которая поможет тысячам сотрудников ЦУМ развиваться профессионально. Следуйте методологии, пишите чистый код, и вместе мы построим отличный продукт!