# 3GIS MVP - План запуска за 1 неделю

## Минимальный функционал MVP

### Базовые возможности (Must Have)

* **Telegram Mini App** с автоматической авторизацией
* **8 категорий услуг** (рестораны, медицина, юристы, красота, авто, финансы, образование, недвижимость)
* **Список организаций** с базовой информацией (название, адрес, телефон, часы работы)
* **Карточка заведения** с фото, контактами и описанием
* **Простой поиск** по названию и категории
* **Фильтры**: город, категория, "открыто сейчас"
* **Геолокация** - показ расстояния до заведения
* **Кнопки действий**: позвонить, проложить маршрут
* **Добавление нового места** через форму

### Отложенные функции (v2)

* Система отзывов и рейтингов
* Интерактивная карта
* Премиум-подписки
* Push-уведомления
* Детальная аналитика

## Технологический стек (бесплатные сервисы)

### Frontend & Deployment

* **Vercel** (Free Plan) - хостинг Next.js приложения
  + 100 GB bandwidth/месяц
  + Автоматический SSL
  + Custom domain поддержка
  + Serverless functions

### База данных

* **Supabase** (Free Plan) - PostgreSQL + Auth
  + 2 проекта
  + 500MB хранилища
  + 2GB transfer/месяц
  + Built-in Auth (не нужен для Telegram, но есть)

### Файловое хранилище

* **Cloudinary** (Free Plan) - изображения заведений
  + 25GB хранилища
  + 25GB bandwidth/месяц
  + Автоматическая оптимизация

### Карты и геолокация

* **Google Maps API** (Free Tier)
  + $200 credits/месяц
  + Хватит на ~28,000 запросов карт
  + Places API для поиска заведений

### Мониторинг и аналитика

* **Vercel Analytics** (Free) - базовая аналитика
* **PostHog** (Free Plan) - продвинутая аналитика
  + 1M events/месяц
  + Session recordings

### Уведомления

* **EmailJS** (Free Plan) - email уведомления
  + 200 emails/месяц
* **Telegram Bot API** (Free) - уведомления в Telegram

## План разработки по дням

### День 1: Настройка инфраструктуры

**Задачи:**

* [ ] Создать Next.js проект с TypeScript
* [ ] Настроить Tailwind CSS + shadcn/ui
* [ ] Подключить Telegram Mini App SDK
* [ ] Создать репозиторий на GitHub
* [ ] Настроить Vercel deployment

**Код структуры:**

npx create-next-app@latest 3gis --typescript --tailwind --eslint

cd 3gis

npx shadcn-ui@latest init

npm install @telegram-apps/sdk @telegram-apps/sdk-react

**Результат:** Рабочая заготовка с CI/CD

### День 2: База данных и модели

**Задачи:**

* [ ] Настроить Supabase проект
* [ ] Создать Prisma схему для бизнесов
* [ ] Настроить подключение к БД
* [ ] Создать seed данные (50 заведений в Нью-Йорке)

**Prisma Schema:**

model Business {

id String @id @default(cuid())

name String

nameEn String?

category String

address String

city String @default("New York")

phone String?

website String?

description String?

// Геолокация

latitude Float?

longitude Float?

// Время работы

hours Json? // {mon: "9:00-18:00", tue: "9:00-18:00"}

// Метаданные

languages String[] @default(["ru", "en"])

verified Boolean @default(false)

createdAt DateTime @default(now())

updatedAt DateTime @updatedAt

@@map("businesses")

}

**Результат:** Рабочая БД с тестовыми данными

### День 3: Telegram Mini App + базовый UI

**Задачи:**

* [ ] Настроить Telegram Bot и Mini App
* [ ] Создать главную страницу с категориями
* [ ] Реализовать навигацию между экранами
* [ ] Добавить Telegram-specific функции (haptic feedback)

**Компоненты:**

* TelegramProvider - обертка для Telegram SDK
* CategoryGrid - сетка категорий 2x4
* BusinessCard - карточка заведения
* SearchInput - поиск с автокомплитом

**Результат:** Рабочая навигация в Telegram

### День 4: API и функциональность поиска

**Задачи:**

* [ ] Создать API routes для получения данных
* [ ] Реализовать поиск по категориям
* [ ] Добавить фильтрацию и сортировку
* [ ] Подключить React Query для кэширования

**API Endpoints:**

// pages/api/businesses/index.ts

GET /api/businesses?category=restaurants&city=New York&search=pizza

// pages/api/businesses/[id].ts

GET /api/businesses/123

// pages/api/categories.ts

GET /api/categories

**Результат:** Рабочий поиск и фильтрация

### День 5: Детальные страницы и действия

**Задачи:**

* [ ] Создать страницу заведения
* [ ] Добавить кнопки "Позвонить", "Маршрут"
* [ ] Интеграция с Google Maps для координат
* [ ] Геолокация пользователя

**Функции:**

* Определение расстояния до заведения
* Открытие телефона для звонка
* Построение маршрута через Google Maps
* Показ часов работы и статуса "открыто/закрыто"

**Результат:** Полнофункциональные карточки заведений

### День 6: Форма добавления + изображения

**Задачи:**

* [ ] Создать форму добавления нового места
* [ ] Настроить Cloudinary для загрузки фото
* [ ] Валидация и обработка ошибок
* [ ] Модерация новых заведений (простая)

**Форма включает:**

* Основную информацию (название, категория, адрес)
* Контакты (телефон, сайт)
* Часы работы
* Загрузку 1-3 фотографий
* Описание на русском

**Результат:** Пользователи могут добавлять места

### День 7: Финальная доработка + деплой

**Задачи:**

* [ ] Оптимизация производительности
* [ ] Тестирование на разных устройствах
* [ ] Настройка custom domain
* [ ] Подготовка к production запуску
* [ ] Базовая аналитика

**Финальные штрихи:**

* Обработка ошибок и loading states
* SEO оптимизация
* Мобильная адаптация
* Тестирование в реальном Telegram

**Результат:** Готовый к запуску MVP

## Конфигурация сервисов

### 1. Vercel Setup

# В корне проекта

npm i -g vercel

vercel login

vercel --prod

# Environment Variables в Vercel Dashboard:

DATABASE\_URL=postgresql://...

CLOUDINARY\_URL=cloudinary://...

GOOGLE\_MAPS\_API\_KEY=...

NEXT\_PUBLIC\_TELEGRAM\_BOT\_TOKEN=...

### 2. Supabase Configuration

// lib/supabase.ts

import { createClient } from '@supabase/supabase-js'

const supabaseUrl = process.env.NEXT\_PUBLIC\_SUPABASE\_URL!

const supabaseKey = process.env.NEXT\_PUBLIC\_SUPABASE\_ANON\_KEY!

export const supabase = createClient(supabaseUrl, supabaseKey)

### 3. Telegram Bot Setup

// lib/telegram.ts

import { useTelegramWebApp } from '@telegram-apps/sdk-react'

export function useTelegram() {

const webApp = useTelegramWebApp()

return {

user: webApp?.initDataUnsafe?.user,

hapticFeedback: () => webApp?.HapticFeedback?.impactOccurred('medium'),

showAlert: (message: string) => webApp?.showAlert(message),

close: () => webApp?.close()

}

}

### 4. Cloudinary Integration

// lib/cloudinary.ts

import { v2 as cloudinary } from 'cloudinary'

cloudinary.config({

cloud\_name: process.env.CLOUDINARY\_CLOUD\_NAME,

api\_key: process.env.CLOUDINARY\_API\_KEY,

api\_secret: process.env.CLOUDINARY\_API\_SECRET

})

export default cloudinary

## Seed данные для тестирования

### Начальные категории:

1. 🍽️ Рестораны и кафе
2. ⚕️ Медицина
3. ⚖️ Юридические услуги
4. 💄 Красота и здоровье
5. 🔧 Автосервисы
6. 🏦 Финансовые услуги
7. 🎓 Образование
8. 🏠 Недвижимость

### Тестовые заведения (по 5-7 в каждой категории):

* Brighton Beach рестораны
* Русскоговорящие врачи в Brooklyn
* Иммиграционные адвокаты в Manhattan
* Салоны красоты в Queens
* Автосервисы с русскими механиками

## Стратегия запуска

### Неделя 1 (после MVP): Soft Launch

* Тестирование с 20-30 пользователями
* Сбор обратной связи
* Исправление критических багов

### Неделя 2-3: Public Beta

* Запуск в русскоязычных Telegram-каналах Нью-Йорка
* Сбор первых 100 заведений
* Привлечение первых 500 пользователей

### Неделя 4: Официальный запуск

* PR в русскоязычных СМИ США
* Запуск партнерской программы
* Начало монетизации

## Бюджет запуска

### Первый месяц (все сервисы Free Tier):

* **$0** - Vercel hosting
* **$0** - Supabase database
* **$0** - Cloudinary images
* **$0** - Google Maps API (в лимитах)
* **$0** - Telegram Bot API
* **$15** - Domain (.com домен)

**Итого: $15** для запуска MVP

### При росте (когда нужно будет масштабироваться):

* Vercel Pro: $20/месяц
* Supabase Pro: $25/месяц
* Cloudinary: $89/месяц
* Google Maps: ~$100/месяц
* **Итого: ~$234/месяц** при 10,000+ пользователей

## Риски и митигация

### Технические риски:

* **Лимиты Free Tier** → Мониторинг использования, готовые планы upgrade
* **Telegram API ограничения** → Кэширование, оптимизация запросов
* **Производительность БД** → Индексы, оптимизация запросов

### Продуктовые риски:

* **Низкое качество данных** → Система модерации, пользовательские отчеты
* **Медленный набор заведений** → Партнерства, импорт из открытых источников
* **Конкуренция** → Фокус на UX и скорость разработки

## Готовность к масштабированию

После успешного MVP следующие функции добавляются по приоритету:

1. **Система отзывов** (неделя 2)
2. **Премиум-подписки** (неделя 3)
3. **Интерактивная карта** (неделя 4)
4. **Push-уведомления** (неделя 5)
5. **Расширение на другие города** (месяц 2)

**Заключение:** За 1 неделю реально создать работающий MVP с базовым функционалом, который решает главную проблему пользователей - поиск русскоязычных услуг в удобном мобильном формате через Telegram.