# Детальный план реализации CRM для ЖКХ

## 1. Подготовительный этап

### 1.1 Анализ требований (5 дней)

* Сбор и анализ бизнес-требований
* Определение функциональных требований
* Определение нефункциональных требований
* Создание документации по требованиям

### 1.2 Проектирование архитектуры (5 дней)

* Проектирование базы данных
* Проектирование API
* Проектирование UI/UX
* Создание технической документации

### 1.3 Настройка окружения (4 дня)

* Настройка локального окружения
* Настройка CI/CD
* Настройка тестового окружения
* Настройка staging окружения

## 2. Разработка базового функционала

### 2.1 Мультитенантность (2 недели)

// app/Models/Tenant.php  
class Tenant extends Model  
{  
 protected $fillable = [  
 'name',  
 'domain',  
 'database',  
 'settings'  
 ];  
   
 public function users()  
 {  
 return $this->hasMany(User::class);  
 }  
}  
  
// app/Http/Middleware/InitializeTenancy.php  
class InitializeTenancy  
{  
 public function handle($request, Closure $next)  
 {  
 $tenant = Tenant::where('domain', $request->getHost())->first();  
 if (!$tenant) {  
 abort(404);  
 }  
   
 config(['database.connections.tenant.database' => $tenant->database]);  
 return $next($request);  
 }  
}

### 2.2 Управление абонентами (2 недели)

// app/Models/Subscriber.php  
class Subscriber extends Model  
{  
 protected $fillable = [  
 'name',  
 'address',  
 'phone',  
 'email',  
 'status'  
 ];  
   
 public function services()  
 {  
 return $this->hasMany(Service::class);  
 }  
}  
  
// app/Http/Controllers/SubscriberController.php  
class SubscriberController extends Controller  
{  
 public function index()  
 {  
 return Subscriber::with('services')  
 ->filter(request('filter'))  
 ->paginate(20);  
 }  
   
 public function store(StoreSubscriberRequest $request)  
 {  
 return Subscriber::create($request->validated());  
 }  
}

### 2.3 Биллинг (2 недели)

// app/Services/BillingService.php  
class BillingService  
{  
 public function calculateBill(Subscriber $subscriber)  
 {  
 $amount = 0;  
 foreach ($subscriber->services as $service) {  
 $amount += $this->calculateServiceAmount($service);  
 }  
 return $amount;  
 }  
   
 public function processPayment(Payment $payment)  
 {  
 DB::transaction(function () use ($payment) {  
 $payment->save();  
 $payment->subscriber->updateBalance($payment->amount);  
 });  
 }  
}

## 3. Разработка дополнительных модулей

### 3.1 Система тикетов (1 неделя)

// app/Models/Ticket.php  
class Ticket extends Model  
{  
 protected $fillable = [  
 'title',  
 'description',  
 'status',  
 'priority',  
 'subscriber\_id'  
 ];  
   
 public function comments()  
 {  
 return $this->hasMany(TicketComment::class);  
 }  
   
 public function attachments()  
 {  
 return $this->hasMany(TicketAttachment::class);  
 }  
}

### 3.2 Интеграция с платежными системами (1 неделя)

// app/Services/PaymentGatewayService.php  
class PaymentGatewayService  
{  
 public function processPayment($amount, $paymentMethod)  
 {  
 switch ($paymentMethod) {  
 case 'card':  
 return $this->processCardPayment($amount);  
 case 'cash':  
 return $this->processCashPayment($amount);  
 default:  
 throw new InvalidPaymentMethodException();  
 }  
 }  
}

### 3.3 Интеграция с SMS и email (1 неделя)

// app/Services/NotificationService.php  
class NotificationService  
{  
 public function sendSMS($phone, $message)  
 {  
 // Интеграция с SMS-шлюзом  
 }  
   
 public function sendEmail($email, $subject, $message)  
 {  
 // Отправка email  
 }  
}

## 4. Тестирование

### 4.1 Unit тесты

// tests/Unit/BillingServiceTest.php  
class BillingServiceTest extends TestCase  
{  
 public function test\_calculate\_bill\_amount()  
 {  
 $subscriber = Subscriber::factory()->create();  
 $service = Service::factory()->create([  
 'subscriber\_id' => $subscriber->id,  
 'tariff' => 100  
 ]);  
   
 $billingService = new BillingService();  
 $amount = $billingService->calculateBill($subscriber);  
   
 $this->assertEquals(100, $amount);  
 }  
}

### 4.2 Feature тесты

// tests/Feature/SubscriberManagementTest.php  
class SubscriberManagementTest extends TestCase  
{  
 public function test\_can\_create\_subscriber()  
 {  
 $response = $this->postJson('/api/subscribers', [  
 'name' => 'John Doe',  
 'address' => '123 Main St',  
 'phone' => '+1234567890'  
 ]);  
   
 $response->assertStatus(201)  
 ->assertJsonStructure([  
 'id',  
 'name',  
 'address',  
 'phone'  
 ]);  
 }  
}

## 5. Развертывание

### 5.1 Настройка сервера

# Установка необходимого ПО  
sudo apt update  
sudo apt install nginx mysql-server redis-server php8.1-fpm  
  
# Настройка PHP  
sudo apt install php8.1-mysql php8.1-redis php8.1-xml php8.1-curl  
  
# Настройка MySQL  
sudo mysql\_secure\_installation

### 5.2 Развертывание приложения

# Клонирование репозитория  
cd /var/www  
git clone git@github.com:your-org/crm.git  
  
# Установка зависимостей  
cd crm  
composer install --no-dev --optimize-autoloader  
npm install --production  
npm run build  
  
# Настройка окружения  
cp .env.example .env  
php artisan key:generate  
  
# Настройка базы данных  
php artisan migrate --force  
php artisan db:seed --force  
  
# Оптимизация  
php artisan config:cache  
php artisan route:cache  
php artisan view:cache

## 6. Мониторинг и поддержка

### 6.1 Настройка логирования

// config/logging.php  
'channels' => [  
 'stack' => [  
 'driver' => 'stack',  
 'channels' => ['daily', 'slack'],  
 'ignore\_exceptions' => false,  
 ],  
 'slack' => [  
 'driver' => 'slack',  
 'url' => env('LOG\_SLACK\_WEBHOOK\_URL'),  
 'username' => 'CRM Log',  
 'emoji' => ':boom:',  
 'level' => env('LOG\_LEVEL', 'critical'),  
 ],  
]

### 6.2 Настройка мониторинга

// app/Providers/AppServiceProvider.php  
public function boot()  
{  
 if (config('app.env') === 'production') {  
 DB::listen(function ($query) {  
 if ($query->time > 100) {  
 Log::warning('Slow query detected', [  
 'sql' => $query->sql,  
 'time' => $query->time  
 ]);  
 }  
 });  
 }  
}

## 7. Безопасность

### 7.1 Настройка аутентификации

// config/auth.php  
'guards' => [  
 'web' => [  
 'driver' => 'session',  
 'provider' => 'users',  
 ],  
 'api' => [  
 'driver' => 'sanctum',  
 'provider' => 'users',  
 ],  
]

### 7.2 Настройка авторизации

// app/Policies/SubscriberPolicy.php  
class SubscriberPolicy  
{  
 public function view(User $user, Subscriber $subscriber)  
 {  
 return $user->tenant\_id === $subscriber->tenant\_id;  
 }  
   
 public function update(User $user, Subscriber $subscriber)  
 {  
 return $user->tenant\_id === $subscriber->tenant\_id;  
 }  
}