








WhatsApp Module - SaaS Version

Overview

The updated WhatsApp module for UltimatePOS now supports SaaS (Software as a Service) environment with separate management capabilities for each business. This module allows each business to configure and manage their own WhatsApp accounts independently and securely.

Key Features

-  Multi-tenant support
-  Complete data isolation between businesses
-  Multiple WhatsApp account management per business
-  Advanced permission system
-  Comprehensive API for account management
-  Comprehensive testing
-  Complete documentation

Requirements

- Laravel 8.0+
- PHP 8.0+
- MySQL 5.7+
- Guzzle HTTP Client

Installation

1. Apply Migration Files

```
bash  
  
php artisan migrate
```

2. Register Middleware

Ensure that the Middleware is registered in `WhatsAppServiceProvider`:

```
php
```

```
protected function registerMiddleware(): void
{
    $router = $this->app['router'];
    $router->aliasMiddleware('whatsapp.business', \Modules\WhatsApp\Http\Middleware\WhatsAppBusinessMiddlewa
}
```

3. Update Configuration

Make sure to update configuration files to include new settings.

Usage

WhatsApp Account Management

Display all accounts for current business

php

GET /api/whatsapp/my-accounts

Create new account

php

POST /api/whatsapp/my-accounts

```
{
    "sources": "WaSender",
    "wa_server": "https://api.wasender.com/send",
    "app_key": "your_app_key",
    "auth_key": "your_auth_key",
    "sender": "1234567890"
}
```

Update existing account

php

PUT /api/whatsapp/my-accounts/{id}

```
{
    "sources": "Updated WaSender",
    "wa_server": "https://api.updated-wasender.com/send"
}
```

Set account as default

php

```
POST /api/whatsapp/my-accounts/{id}/set-default
```

Delete account

php

```
DELETE /api/whatsapp/my-accounts/{id}
```

Sending Messages

Using the service in code

php

```
use Modules\WhatsApp\Services\WhatsAppServices;

$whatsappService = app(WhatsAppServices::class);

// Send notification message
$data = [
    'mobile_number' => '1234567890',
    'whatsapp_text' => 'Hello, this is a test message'
];

$whatsappService->requestData('notification', $data, null, null, $businessId);
```

Test message sending

php

```
POST /api/whatsapp/my-test-message
{
    "phone": "1234567890",
    "message": "Test message"
}
```

Database Structure

whatsapp_gateway Table

Column	Type	Description
id	INT	Unique identifier
business_id	INT	Business identifier
sources	VARCHAR(255)	Service source
wa_server	VARCHAR(500)	WhatsApp server URL
app_key	VARCHAR(255)	Application key
auth_key	VARCHAR(255)	Authentication key
sender	VARCHAR(255)	Sender number
is_default	BOOLEAN	Global default (deprecated)
is_default_for_business	BOOLEAN	Business default
created_at	TIMESTAMP	Creation date
updated_at	TIMESTAMP	Update date

Security

Permission System

- Each user can only access their business accounts
- Middleware verifies access permissions
- Complete data isolation between businesses

Data Validation

```
php

'sources' => 'required|string|max:255',
'wa_server' => 'required|url|max:500',
'app_key' => 'required|string|max:255',
'auth_key' => 'required|string|max:255',
'sender' => 'nullable|string|max:255'
```

Testing

Running Tests

```
bash

php artisan test Modules/WhatsApp/Tests/Feature/WhatsAppBusinessTest.php
```

Available Tests

- Test account display
- Test new account creation
- Test account updates
- Test account deletion
- Test setting default account
- Test security and permissions
- Test data validation

Troubleshooting

Common Errors

"No WhatsApp account configured for this business"

- Ensure a WhatsApp account exists for the business
- Verify that the account is set as default

"You are not authorized to access WhatsApp data for this business"

- Ensure the user belongs to the correct business
- Verify the business_id in the request

Message sending error

- Check server configuration accuracy
- Ensure API keys are correct
- Verify internet connection

Development

Adding New Features

1. Create a new Controller in `Http/Controllers/`
2. Add routes in `Routes/api.php`
3. Apply appropriate Middleware
4. Write tests

Contributing

1. Fork the project
2. Create a new feature branch
3. Write code and tests
4. Submit a Pull Request

Support

For support or to report issues:

- Create an Issue in the repository
- Review comprehensive documentation
- Check the tests

License

This module is licensed under the same license as the main project.

Updates

Version 2.0 (SaaS)

- Multi-tenant support
- Enhanced security system
- Updated API
- Comprehensive testing

Version 1.1 (Original)

- Basic WhatsApp support
- Single shared account
- Basic sending functions