Final Project Documentation

Team Name: THE NEW GROUP
Course: IT490-004
Date of Submission: May 16th, 2025
Team Members & Primary Contributions:

- **Wisdom:** ZeroTier VPN hosting, Apache server setup, contributions to listener.php, login.php, register.php, and backend logic.
- **Jayden:** Database code (entirety, schema IT490DB.sql), contributions to listener.php.
- **Dennis:** Frontend development (HTML files: index.html, about.html, contact.html, product-detail.html, product.html, shoping-cart.html, thank-you.html, product_api.html)

1. Git Repositories

All Git repositories involved in this project are listed below.

- **Main GitHub Repository:** https://github.com/wui2Wisdom/IT490-NEW-CHANGES
  - *Note: This repository contains the PHP frontend, backend listener, database schema, and related configuration files.*
- **Deployment to Heroku (https://the-knive-store-9df170960c72.herokuapp.com/knives/auth.html, https://the-knive-store-9df170960c72.herokuapp.com/knives):**
  - Code is pushed from the main application repository to Heroku.

2. Project Management Report

Key tasks included: User authentication setup, RabbitMQ integration, database schema design, frontend HTML creation, API integration for product page, systemd service for listener, 2FA exploration, logging implementation, Heroku deployment troubleshooting.

Wisdom: Specialized in minimal frontend development as well as back end. Was responsible for listener configuration to rabbitmq. Established user authentication to website login as well as Heroku deployment and vpn configuration.

Dennis: Specialized in making the Website for the database which involved HTML and CSS styling and also integrated product API functionality. Established proper compatibility with the authentication server and the database further made sure that

the main website was working properly in the rabbitmq.

Jayden: Created the necessary functions for both the database and RabbitMQ listener. Also set up the use for the systemd service provided for the listener to track timestamps and relevant logs. Suggested the use of ZeroTier for VPN configuration when WireGuard seemed unusable.

3. Relevant Communications

Important technical decisions and delivery issues were mainly discussed in person or over an audio call between us. Below is a link to a document highlighting many scheduling plans in our Discord chat.

https://docs.google.com/document/d/1X8bAVzIVznPK10fFqfpOce44T9jpeRNXyn4hDIFN0dw/edit?usp=sharing

A. Technical Troubleshooting & Decisions:

- **Platform:** Ubuntu Terminal
- **Participants:** Wisdom, Jayden, Dennis
- **Summary/Log:**
  - **Database Setup:** Discussed creation of users table, adding email column (SQLSTATE[42S22] error resolved by adding column correctly), session table.
  - **PHP Listener (listener.php):**
    - Configuration for local DB (IT490, user IT490, pass password) and remote RabbitMQ (192.168.191.13:5672, user guest:guest).
    - Logic for handling login (checking username OR email) and registration requests.
    - Password hashing (password_hash, password_verify).
    - Session key generation and storage.
  - **Frontend Scripts (login.php, register.php):**
    - Sending messages to RabbitMQ.
    - Implementing RPC (request/reply) pattern for responses from the listener.
    - Debugging issues with message keys (username vs. identifier).
  - **Composer Issues:**
    - Troubleshooting pragmarx/google2fa-php installation ("Could not find a matching version", "404 not found, no packages here" from Packagist metadata URL). Identified as an external Packagist issue.
    - Installation of monolog/monolog and php-amqplib/php-amqplib.

- **Logging:**
  - Implementation of Monolog with JsonFormatter and StreamHandler in logger_setup.php.
  - Target log file: knives/logs/auth_operations.log.
  - Troubleshooting log file creation (permissions, path issues, auth_operations.log being a directory).
- **Systemd Service:**
  - Creation of rabbitmq-listener.service to manage listener.php.
  - Configuration for user, execution path, restart behavior, and logging via journald.
- **Heroku Deployment:**
  - Troubleshooting Failed to open stream: No such file or directory for logger_setup.php (path issue).
  - Troubleshooting Failed to open stream: No such file or directory for vendor/autoload.php (path issue from within logger_setup.php, corrected to __DIR__ . '/../vendor/autoload.php').
- **API Integration (product_api.html):**
  - Fetching HTML content from https://machinewise.store/products/serif-1 and displaying it.

4. Server Documentation

This section details the setup for each server environment used in the project.

4.1 Development Environment Server(s)

A. Primary Developer VM (Wisdom)

- **VM Provider/Type:** QEMU (Local Virtual Machine)
- **VM Specs:**
  - **CPU/Architecture:** Arm64 (Machine: QEMU 9.1 ARM virtual machine, alias of virt-9.1)
  - **RAM:** [Specify RAM, e.g., 4GB or 8GB]
  - **Storage:** [Specify Disk Size, e.g., 50GB or 100GB]
  - **Networking:** Bridged/NAT (as configured for local QEMU), ZeroTier for VPN access (Wisdom hosts the ZeroTier network).
- **OS Version:** Ubuntu (based on PHP 8.1.2-1ubuntu2.21, likely Ubuntu 22.04 LTS) (Linux)
- **Key Users & Permissions:**
  - ubuntu: Primary user with sudo privileges.
  - www-data: User for the Apache2 webserver process.

        - Needs read access to web application files in ~/git/rabbitmqphp_example/ (or its served subdirectory like public).
        - Needs write access to ~/git/rabbitmqphp_example/knives/logs/ for Monolog.
        - Needs write access to PHP session save path (usually /var/lib/php/sessions).
    - IT490: MySQL user for the application database. (Jayden)
        - Privileges: SELECT, INSERT, UPDATE, DELETE on IT490 database tables. CREATE TABLE, ALTER TABLE during setup/migrations.
    - Permissions commands used:
        - sudo chown www-data:www-data ~/git/rabbitmqphp_example/knives/logs
        - sudo chmod 775 ~/git/rabbitmqphp_example/knives/logs
        - *Note on chmod 777*: While mentioned as a possibility during troubleshooting, this was advised against for general use. The aim was to use more specific permissions like 775.
- **List of Software to Install:**
    - Apache2 Web Server: sudo apt install apache2
    - PHP 8.1: sudo apt install php8.1 php8.1-cli php8.1-mbstring php8.1-xml php8.1-mysql php8.1-curl php8.1-zip php8.1-intl php8.1-bcmath php8.1-gd php8.1-json php8.1-pdo (ensure all necessary extensions are listed)
    - MySQL Server: sudo apt install mysql-server
    - Composer (latest version): Installation via script from getcomposer.org.
    - Git: sudo apt install git
    - ZeroTier Client: Installation via script from zerotier.com.
    - RabbitMQ Server: sudo apt install rabbitmq-server (for local development message queue)
    - PHP Dependencies (via composer install in ~/git/rabbitmqphp_example/):
        - php-amqplib/php-amqplib:^3.7 (from composer.json)
        - monolog/monolog
        - (pragmarx/google2fa-php - installation was attempted, but faced Packagist issues; if resolved, list here)
    - Nano text editor: sudo apt install nano
- **List of Software to Configure (and links/paths to backups of config files in the Git repo or final zip):**
    - **Apache2:**
        - Virtual Host configuration pointing to ~/git/rabbitmqphp_example/ (or a public subdirectory): /etc/apache2/sites-available/rabbitmqphp_example.conf (Link: [Path to rabbitmqphp_example.conf or its content])

        - Enabled modules: mpm_prefork, php8.1, rewrite, ssl. Commands: sudo a2enmod rewrite ssl
        - SSL Configuration: Self-signed certificates for localhost development. (Details: [Path to self-signed certs or generation steps])
    - **PHP:**
        - php.ini modifications:
            - /etc/php/8.1/apache2/php.ini: display_errors = On, error_reporting = E_ALL, date.timezone = America/New_York
            - /etc/php/8.1/cli/php.ini: display_errors = On, error_reporting = E_ALL, date.timezone = America/New_York
    - **MySQL:**
        - User IT490 created with password password, granted privileges on IT490 database.
        - Database schema: IT490DB.sql (located in ~/git/rabbitmqphp_example/)
    - **Application Code (~/git/rabbitmqphp_example/):**
        - knives/listener.php:
            - DB Config: host 127.0.0.1, name IT490, user IT490, pass password.
            - RabbitMQ Config: host 192.168.191.13 (ZeroTier IP of RabbitMQ server, could be localhost for dev), port 5672, user guest, pass guest.
        - knives/login.php & knives/register.php: RabbitMQ connection details similar to listener for sending messages.
        - knives/logger_setup.php: Configures Monolog to write to knives/logs/auth_operations.log.
    - **RabbitMQ Server (local instance):**
        - Default guest:guest user used for development.
        - Queues login, registration (durable, auto-declared by listener/publisher).
    - **Systemd Service for Listener:**
        - /etc/systemd/system/rabbitmq-listener.service (content as discussed, pointing to ~/git/rabbitmqphp_example/knives/listener.php, running as ubuntu user for dev). (Link: [Content of rabbitmq-listener.service])
    - **ZeroTier:**
        - Joined Network ID: [Wisdom's ZeroTier Network ID]
- **Anything else needed to make the server over from scratch:**
    - Clone Git repository: git clone [Your Git Repo URL] ~/git/rabbitmqphp_example
    - Install Composer dependencies: cd ~/git/rabbitmqphp_example && composer install
    - Import database: mysql -u IT490 -p IT490 < IT490DB.sql (after creating the database and user)

    - Configure and enable Apache virtual host: sudo a2ensite rabbitmqphp_example.conf && sudo systemctl reload apache2
    - Enable and start systemd service: sudo systemctl enable --now rabbitmq-listener.service
    - Firewall: sudo ufw allow 'Apache Full', sudo ufw allow 5672/tcp (RabbitMQ), sudo ufw allow 3306/tcp (MySQL, if accessed remotely).

B. Representative Developer VM (Jayden)

- **VM Provider/Type:** Oracle VirtualBox (Virtual Machine)
- **VM Specs:**
    - **CPU/Architecture:** x86_64
    - **RAM:** 4858 MB
    - **Storage:** 30.00 GB
    - **Networking:** NAT, ZeroTier client connected to Wisdom's network.
- **OS Version:** Ubuntu 64-bit (Linux)
- **Key Users & Permissions:**
    - Jayden: Standard user with sudo privileges.
    - MySQL user IT490 for connecting to the shared/local database.
- **List of Software to Install:**
    - PHP 8.1 (CLI primarily): sudo apt install php8.1-cli php8.1-mysql php8.1-mbstring php8.1-xml php8.1-bcmath php8.1-json
    - MySQL Client: sudo apt install mysql-client (or full server if running DB locally)
    - Composer
    - Git
    - ZeroTier Client
    - PHP Dependencies (via composer install in project directory): php-amqplib/php-amqplib, monolog/monolog.
- **List of Software to Configure:**
    - PHP CLI (/etc/php/8.1/cli/php.ini): date.timezone = America/New_York.
    - Application Code (~/git/rabbitmqphp_example/):
        - knives/listener.php configured to connect to appropriate DB (could be Wisdom's VM via ZeroTier, or a local instance) and RabbitMQ (Wisdom's VM via ZeroTier).
- **Anything else needed:**
    - Clone Git repository.
    - composer install.
    - Access to shared database or setup of local database instance.

C. Representative Developer VM (Dennis)

- **VM Provider/Type:** VMware (Virtual Machine)
- **VM Specs:**
  - **CPU/Architecture:**
  - **RAM:** 10 GB
  - **Storage:** 30GB
  - **Networking:** Zero Tier connected to Wisdom's network
- **OS Version:** Ubuntu 64-bit linux
- **Key Users & Permissions:**
- **List of Software to Install:**
  - PHP 8.1 (CLI primarily): sudo apt install php8.1-cli php8.1-mysql php8.1-mbstring php8.1-xml php8.1-bcmath php8.1-json
  - MySQL Client: sudo apt install mysql-client (or full server if running DB locally)
  - Composer
  - Git
  - ZeroTier Client
  - PHP Dependencies (via composer install in project directory): php-amqplib/php-amqplib, monolog/monolog.
- **List of Software to Configure:**
  - PHP CLI (/etc/php/8.1/cli/php.ini): date.timezone = America/New_York.
  - Application Code (~/git/rabbitmqphp_example/):
    - knives/listener.php configured to connect to appropriate DB (could be Wisdom's VM via ZeroTier, or a local instance) and RabbitMQ (Wisdom's VM via ZeroTier).

4.2 QA Environment Server(s)

[This environment is for testing before deploying to production.]

- **Platform Provider:** Heroku
- **Dyno Type(s):** Free Dyno
- **Region:** us-east
- **Stack:** heroku-22 (Ubuntu 22.04 based)
- **Key Users & Permissions:**
  - Heroku account owner(s) with deployment access.
  - Build process runs with Heroku's permissions.
- **Buildpacks Used:**
  - heroku/php (automatically detected for PHP projects)
- **Add-ons:**
  - **Database (MySQL):** JawsDB MySQL
  - **Message Queue (RabbitMQ):** CloudAMQP
- **Software Configuration (via Heroku Config Vars):**

    - DATABASE_URL: Provided by the MySQL add-on (parsed by application to get host, user, pass, db name).
    - CLOUDAMQP_URL (or similar): Provided by the RabbitMQ add-on (parsed for connection details).
    - APP_ENV: qa
    - LOG_LEVEL: DEBUG or INFO
    - [Any other API keys or environment-specific settings]
- **Procfile (~/git/rabbitmqphp_example/Procfile):**
    - web: vendor/bin/heroku-php-apache2 knives/ (or your specific public directory if knives/ contains frontend files like login.php)
    - worker: php knives/listener.php (for the backend listener)
- **Application Code Configuration:**
    - PHP scripts (listener.php, login.php, register.php) are written to parse DATABASE_URL and CLOUDAMQP_URL from environment variables to configure DB and RabbitMQ connections.
    - logger_setup.php: Configured to write logs to stderr or stdout for Heroku's logplex to capture:
      // In logger_setup.php for Heroku
      // $streamHandler = new StreamHandler('php://stdout', Logger::DEBUG);

- **Deployment Process:**
    - git push heroku main (or the branch designated for QA).
    - Heroku buildpack runs composer install --no-dev automatically.
- **QA Database Setup:**
    - Schema migrated using heroku run php your_migration_script.php or by connecting to the add-on DB and running IT490DB.sql.
    - Test data seeding process.