# Milestone 3: Practical Implementation of LowTech Gmbh Webshop in CSP Platform

Wladymir Alexander Brborich Herrera (1437876)
`wladymir.brborich-herrera@stud.fra-uas.de`,
Vishwaben Pareshbhai Kakadiya (1471845)
`vishwaben.kakadiya@stud.fra-uas.de`,
Hellyben Bhaveshkumar Shah (1476905)
`hellyben.shah@stud.fra-uas.de`,
Heer Rakeshkumar Vankawala (1449039)
`heer.vankawala@stud.fra-uas.de`, and
Priyanka Dilipbhai Vadiwala (1481466)
`priyanka.vadiwala@stud.fra-uas.de`

Frankfurt University of Applied Sciences
(1971-2014: Fachhochschule Frankfurt am Main)
Nibelungenplatz 1
D-60318 Frankfurt am Main

## Abstract

LowTech GmbH Cloud Transformation Project, building upon the previous analyses and migration strategies. In this phase, the focus is on optimizing cloud operations, performance monitoring, and cost efficiency after the transition to Microsoft Azure. The objective is to ensure system reliability, security, and scalability while fine-tuning the deployed infrastructure. The report outlines key post-migration strategies, including performance assessment, security enhancements, and cost analysis. It introduces automation techniques using tools like Terraform and Ansible for infrastructure management and GitHub Actions for continuous integration and deployment (CI/CD). Special attention is given to monitoring solutions like Azure Monitor and Prometheus to track system performance and detect potential issues in real time. Furthermore, we evaluate cloud cost management techniques by analyzing usage patterns and identifying areas for optimization. The report also discusses future scalability strategies, ensuring that LowTech GmbH is well-equipped to handle growing business demands. This phase serves as the foundation for long-term cloud sustainability, enabling the company to leverage cloud-native solutions efficiently while maintaining operational resilience.

## 1 Introduction

### 1.1 Overview of the Project

LowTech GmbH, a medium-sized enterprise specializing in wooden furniture production, is modernizing its IT infrastructure as part of a comprehensive cloud transformation. Initially relying on traditional on-premises systems, the company faced challenges in scalability, security, and operational efficiency.

The transformation began with a thorough assessment of the existing infrastructure, which revealed the following key challenges:

- Limited scalability due to fixed hardware constraints.
- High operational costs and energy consumption of legacy systems.
- Outdated security measures, including basic firewall protection.
- Lack of automation, requiring manual interventions for maintenance and scaling.

To address these challenges, a private cloud migration strategy was adopted, focusing on hyper-converged infrastructure (HCI) and virtualization using Proxmox and Ansible. The aim was to improve scalability, security, and cost-efficiency while ensuring minimal downtime and business continuity.

### 1.2 Objectives of the Cloud Implementation of Webshop

The Webshop is a critical component for LowTech GmbH, serving as the company's primary sales platform. The goal of its cloud implementation is to enhance performance, scalability, and security, ensuring a seamless user experience.

Key objectives for the Webshop's cloud-based deployment include:

- **Scalability and Performance Optimization:** The Webshop is deployed on Azure App Service with auto-scaling capabilities, enabling efficient traffic handling during peak periods. A load balancer ensures even traffic distribution across multiple instances.
- **High Availability and Reliability:** Azure Virtual Machine Scale Sets provide fault tolerance with automatic failover. Azure Blob Storage is used to securely store digital assets with high availability.
- **Security and Compliance:** The Webshop integrates with Microsoft Entra ID for user authentication and Azure Security Center for enhanced threat protection. Encryption and Role-Based Access Control (RBAC) are employed to safeguard sensitive customer data.
- **Continuous Deployment and DevOps Automation:** A CI/CD pipeline powered by GitHub Actions automates code deployments, improving deployment speed and reducing manual intervention.
- **Cost Efficiency and Resource Optimization:** The dynamic allocation of resources optimizes compute and storage usage, reducing operational expenses. Azure's pay-as-you-go model aids in cost forecasting and budget management.
- **Future-Proofing and Cloud-Native Development:** The Webshop follows cloud-native best practices, utilizing Docker containers for portability and Azure Kubernetes Service (AKS) for container orchestration, positioning the Webshop for seamless integration with future cloud services.

These objectives ensure that the Webshop is scalable, secure, and cost-effective, delivering a fast and reliable shopping experience while adapting to the evolving needs of the business.

# 2 Application Design

## 2.1 Architectural Overview

Detailed description of the three-tier structure with CSP service mapping

### 2.1.1 Presentation-Tier (Frontend) - User Interface (UI)

*Technology Stack*

- Frontend Framework
- State Management
- Communication with Backend
- Hosting & Deployment,

*component-based architecture*

1. Navigation & Routing
2. API Communication
3. Data Fetching

*Key Features of the UI*

1. Product Catalog
2. Product Search and Filtering
3. Product Details Page
4. Shopping Cart
5. Checkout Process

#### 2.1.2 Application-Tier (Backend) - Business Logic

#### 2.1.3 Data-Tier (Database) - Databases

### 2.2 Technology Stack

- Frontend:
- Backend:
- Database:

### 2.3 System Diagrams

## 3 Implementation Process

### 3.1 Cloud Environment Setup

Step-by-step account configuration and resource provisioning

### 3.2 Service Integration

- Azure Load Balancer configuration
- Database replication setup
- Blob storage integration patterns

### 3.3 Development Challenges

- State management in scaled environments
- Database connection pooling
- CSP-specific limitations encountered

## 4 Operational Characteristics

### 4.1 Performance Metrics

#### 4.1.1 Functional Test Cases

| Test Case ID | Test Scenario | Expected Outcome | Status (Pass/Fail) |
|---|---|---|---|
| TC-001 | Load homepage and verify product listing | Homepage loads with products displayed correctly. | |

| Test Case ID | Test Scenario | Expected Outcome | Status (Pass/Fail) |
|---|---|---|---|
| TC-002 | Apply price filter (Ascending/Descending) | Products reorder correctly based on selected price. | |
| TC-003 | Apply Öut of Stockffilter | Only out-of-stock items are displayed. | |
| TC-004 | Apply FFast Deliveryffilter | Only products eligible for fast delivery show up. | |
| TC-005 | Filter by category | Products are filtered correctly by selected category. | |
| TC-006 | Search product by name or category | Products matching search are shown correctly. | |
| TC-007 | Clear filter functionality | All filters are removed, showing the full product list. | |
| TC-008 | Product Detail Page - Click Product | Clicking a product opens its detailed page. | |
| TC-009 | Product Detail Page - Load Product Details | Product details (name, description, price) are displayed. | |
| TC-010 | Add a product to cart | Product appears in cart with correct details. | |
| TC-011 | Increase product quantity in cart | Quantity updates and is reflected in the cart. | |
| TC-012 | Remove product from cart | Product is removed from the cart immediately. | |
| TC-013 | Proceed to checkout | Checkout page loads with the correct order summary. | |
| TC-014 | Select payment method - Stripe | Stripe payment option is selected and processed. | |
| TC-015 | Select payment method - PayPal | PayPal payment option is selected and processed. | |
| TC-016 | Complete order processing | Order confirmation message is displayed. | |
| TC-017 | Order confirmation email is received | Email is sent after order is placed. | |
| TC-018 | Shipment notification email is received | Email is sent when the order is shipped. | |
| TC-019 | Toggle Dark/Light Theme | Application switches between themes successfully. | |

### 4.2 Security Considerations

– Network security groups configuration
– Database encryption implementation
– Access control mechanisms

## 5 Critical Analysis

### 5.1 Cloud Service Evaluation

Cost-benefit analysis of selected Azure services

### 5.2 Architectural Decisions

Trade-off discussion between containerized vs serverless approaches

# 6 Repository Documentation

## 6.1 GitHub Structure

1. **Branching Strategy:**
   The project follows a simple main branch strategy where all contributors work directly on the main branch. This ensures seamless integration without managing multiple branches. All changes should be committed with meaningful messages. Code should be reviewed and tested before pushing to the main branch.

2. **Continuous Integration/Continuous Deployment:**
   - **Continuous Integration (CI):**
     Every code commit triggers an automated build and testing process. This includes:
     - Running unit tests to validate individual components.
     - Performing integration tests to ensure compatibility between different modules.
     - Static code analysis for linting and security vulnerabilities.

   - **Continuous Deployment (CD):**
     Once the CI stage passes successfully, the application is automatically deployed to the appropriate environment. This process includes:
     - Deploying to a staging environment for final testing.
     - Running automated acceptance tests before production deployment.
     - Deploying to production with rollback mechanisms in case of failure.

   - **Documentation Standards**

## 6.2 Contribution Tracking

Commit history analysis and individual contribution breakdown

# 7 Conclusion

## 7.1 Project Outcomes

Summary of achieved objectives and demo capabilities

## 7.2 Future Enhancements

Potential improvements for production readiness

## References