

Worman API Technical Documentation

**Version:** 1

**Author:** Matimba Baloyi, Vodacom (External)

**Publication date:** 2023/08/01



|  |  |
| --- | --- |
|  |  |
|  |  |

I Revision History

| Revision Number | Revision Date | Author | Summary of Changes |
| --- | --- | --- | --- |
| Version 0.1 | 2023/08/01 | M BALOYI | Draft initial template |
|  |  |  |  |

II Related Documents

| Document Name | Date | Author |
| --- | --- | --- |
| Functional Design Specification | 2023/08/01 | M BALOYI |
|  |  |  |

III Abbreviations

| Abbreviation/Acronym | Definition |
| --- | --- |
| 2FA | 2 Factor Authentication |
| AD | Active Directory |
| Admin | Administrator |
| Android | Operating system based on Linux for mobile devices |
| API | Application Program Interface |
| App | Application |
| AWS | Amazon Web Services |
| CI | Corporate Identity |
| CI/CD | Continious Integration and Continious Deployment |
| CSOC | Customer Support Operations Centre |
| CSS | Cascading Style Sheet |
| DB | Database |
| DR | Disaster Recovery |
| EBU | Enterprise Business Unit |
| ECS | Enterprise Cloud Services |
| HTML | HyperText Markup Language |
| IAM | Identity and Access Management |
| iOS | Operating system used on Apple mobile devices |
| JS | JavaScript |
| MDM | Mobile Device Management |
| PiTR | Point in Time Recovery |
| SaaS | Software as a Service |
| SLA | Service Level Agreement |
| SMS | Short Message Service |
| SysAdmin | System Administrator |
| VCC | Vodacom Corporate Connect |
| VSDM | Vodacom Corporate Connect |
| XAML | Extensible Apllication Markup Language |

LogoCMYK2TABLE OF CONTENTS

Contents

[I Revision History I](#_Toc141945423)

[II Related Documents I](#_Toc141945424)

[III Abbreviations I](#_Toc141945425)

[TABLE OF CONTENTS III](#_Toc141945426)

[1 Introduction 5](#_Toc141945427)

[1.1 Aim 5](#_Toc141945428)

[1.2 Background 5](#_Toc141945429)

[1.3 Purpose 5](#_Toc141945430)

[2 Document Scope 5](#_Toc141945431)

[3 System Architecture 5](#_Toc141945432)

[3.1 Application environment 5](#_Toc141945433)

[3.2 Application Technology and Specification 6](#_Toc141945434)

[3.2.1 Application Server – Dev (1) 6](#_Toc141945435)

[3.2.2 Application Server – UAT (1) 6](#_Toc141945436)

[3.2.3 Application Server – Production (2) 6](#_Toc141945437)

[3.2.4 Database Server – Dev 7](#_Toc141945438)

[3.2.5 Database Server – UAT 7](#_Toc141945439)

[3.2.6 Database Server – Prod 7](#_Toc141945440)

[4 Application Functionality 8](#_Toc141945441)

[4.1 Order/Invoice Manipulation Process 8](#_Toc141945442)

[5 Database Design 9](#_Toc141945443)

[5.1 Data Objects and Resultant Data Structures 9](#_Toc141945444)

[5.2 Tables/Attributes 10](#_Toc141945445)

[6 Software Architecture 20](#_Toc141945446)

[6.1 Backend 22](#_Toc141945447)

[6.2 Software Technology 23](#_Toc141945448)

[7 Security 24](#_Toc141945449)

[7.1 Infrastructure 24](#_Toc141945450)

[7.2 Application 24](#_Toc141945451)

[7.3 Data 24](#_Toc141945452)

[8 Backups 24](#_Toc141945453)

[9 Disaster Recovery and Business Continuity 25](#_Toc141945454)

[10 Auditing 25](#_Toc141945455)

# Introduction

## Aim

The aim of this document is to present the technical specification for the Woerman Web API.

## Background

Woerman Coastal have developed an application to manipulate data (users, orders, invoices) in Swagger API. The application makes use of a Bearer token for authorization.

## Purpose

The application that will be developed will need to get data (User, order, invoice) from the API in JSON format, save the data provided, convert the data to xml format and send the xml data to Biz.

# Document Scope

The scope of work includes the following key activities:

* System Architectural design
* Application functional requirements
* Database design
* Software Architecture
* Security
* Backups
* DR and Business Continuity
* Auditing

# System Architecture

## Application environment

The Woerman API is built on the Secure by Design Vodacom principles and infrastructure deployed as per the network segmentation requirements. Infrastructure will be deployed and managed within the ECS VMWare virtualisation environment. Application and database servers are hosted within separate zones (vlans) and access to DB only allowed thru the application interface layer.

FW policies will be implemented for secure access of resources. The application servers are load balanced. The DB servers are hosted within the existing EC SQL Cluster on different nodes from existing databases and segregated from the application network. This design ensures a HA environment and monitoring will ensure business continuity.

## Application Technology and Specification

### Application Server – Dev (1)

|  |  |
| --- | --- |
| ServerName | xxx |
| OS | Microsoft Windows Server 2019 (64-bit) |
| CPU | 2 |
| Memory | 8GB |
| Storage | 100GB |
| Web server software | IIS |

### Application Server – UAT (1)

|  |  |
| --- | --- |
| ServerName | xxx |
| OS | Microsoft Windows Server 2019 (64-bit) |
| CPU | 2 |
| Memory | 8GB |
| Storage | 100GB |
| Web server software | IIS |

### Application Server – Production (2)

|  |  |
| --- | --- |
| ServerName | xxx |
| OS | Microsoft Windows Server 2019 (64-bit) |
| CPU | 4 |
| Memory | 8GB |
| Storage | 100GB |
| Web server software | IIS |

### Database Server – Dev

Need to determine if SQL DB for Dev can be hosted on the App Dev server or we will need a separate DB environment hosted on its own server as there currently is no Dev SQL Cluster.

### Database Server – UAT

|  |  |
| --- | --- |
| ServerName | UATSQLCLUSTER01 |
| OS | Microsoft Windows Server 2016 or later (64-bit) |
| CPU | 4 |
| Memory | 12GB |
| Storage | 500GB |

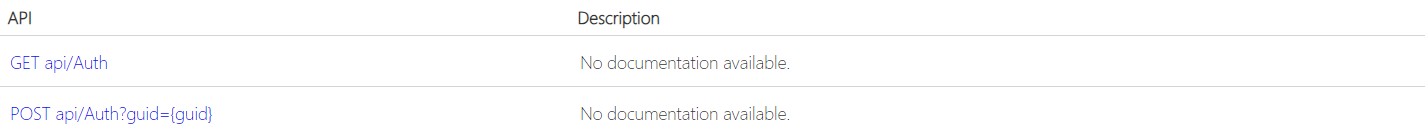
### Database Server – Prod

|  |  |
| --- | --- |
| ServerName | ECSQLPRD01 |
| OS | Microsoft Windows Server 2016 or later (64-bit) |
| CPU | 12 |
| Memory | 32GB |
| Storage | 2TB |

# Application Functionality

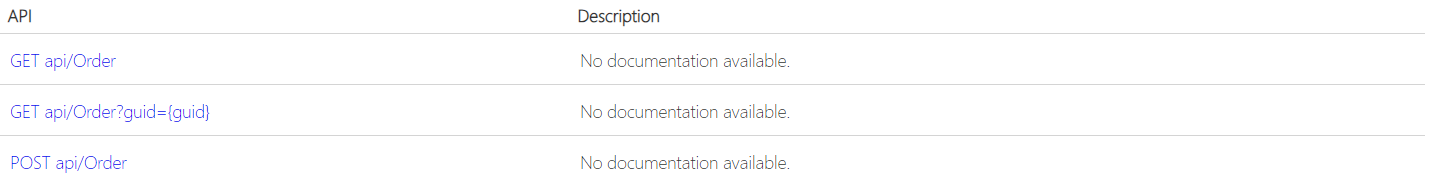
## Order/Invoice Manipulation Process

**Auth**



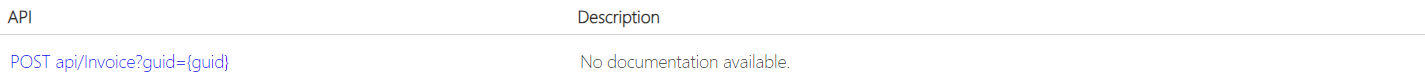
* Http GET method to get Users.
  + This method will call the URL <https://edi.yourwoermann.com:5000/api/Auth>to get a list of users from the Swagger API.
* Http POST method to save user details.
  + This method will call the URL <https://edi.yourwoermann.com:5000/api/Auth>to save user details from the Swagger API.

**Orders**



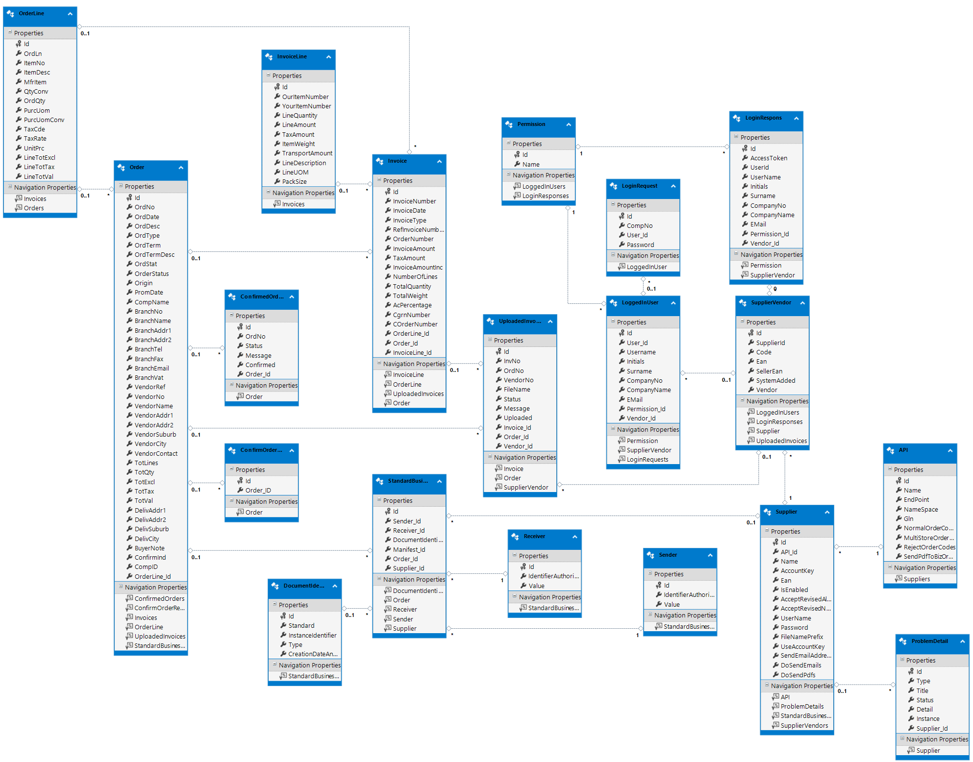
* Get json order from Woerman API in JSON format.
* Http GET method to get single order.
  + This method will call the URL <https://edi.yourwoermann.com:5000/api/Order/1>to get a single order from the Swagger API.
* Http GET method to get multiple orders.
  + This method will call the URL [https://edi.yourwoermann.com:5000/api/Order](https://edi.yourwoermann.com:5000/api/Order/1)to get a list of orders from the Swagger API.
* Http POST method to save a single order to the database.
  + This method will call the URL [https://edi.yourwoermann.com:5000/api/Order](https://edi.yourwoermann.com:5000/api/Order/1)/1to save a single order from the Swagger API.
* Http POST method to save multiple orders to the database.
  + This method will call the URL [https://edi.yourwoermann.com:5000/api/Order](https://edi.yourwoermann.com:5000/api/Order/1)to save orders from the Swagger API.
* Save Json order file.
* Convert json file to xml.
* Send order file in XML format to biz.

**Invoices**



* Http POST method to save multiple invoices to the database.
  + This method will call the URL <https://edi.yourwoermann.com:5000/api/Invoice>to save multiple invoices from the Swagger API.
* Save Json invoice file.
* Convert json file to xml.
* Send invoice file to biz.

# Database Design



## Data Objects and Resultant Data Structures

Microsoft SQL will be used within the existing EC SQL Cluster.

Transparent Data Encryption to be implemented on sensitive information containing tables.

3rd From data normalisation is applied.

## Tables/Attributes

EndPointAuth

* Id
* Name
* EndPoint
* NameSpace
* Gln
* UserName
* Password
* SellerGln

Code

* Id
* SellerGln
* ReceiverGln
* Value
* Name

DocumentIdentification

* Id
* Standard
* InstanceIdentifier
* Type
* CreationDateAndTime

FileInformation

* Id
* FileSize
* FileName
* FileStore
* FileType

Receiver

* Id
* IdentifierAuthority
* Value

Sender

* Id
* IdentifierAuthority
* Value

StandardBusinessDocumentHeader

* Id
* Sender
* Receiver
* DocumentIdentification
* Manifest

Supplier

* Id
* AcceptRevisedAllocationOrders
* AcceptRevisedNormalOrders
* CompleteOrderResponse
* UseAccountKey
* DoSendEmails
* SendPdf
* AccountKey
* ApiEndPoint
* ApiGln
* ApiName
* ApiNameSpace
* Ean
* FileNamePrefix
* Name
* Password
* SellerEan
* SendEmailAddress
* UserName
* NormalOrderCodes
* MultiStoreOrderCodes
* RejectOrderCodes
* SendPdfToBizOrderCodes
* SupplierVendor

SupplierVendor

* Id
* Code
* Ean
* SellerEan

LogedInUser

* Id
* UserId
* Username
* Initials
* Surname
* CompanyNo
* CompanyName
* EMail
* Permission\_Id
* Vendor\_Id

ProblemDetails

* Id
* Type
* Title
* Status
* Detail
* Instance

LoginRequest

* Id
* CompNo
* UserId
* Password

LoginResponse

* Id
* AccessToken
* UserId
* UserName
* Initials
* Surname
* CompanyNo
* CompanyName
* EMail
* Permission\_Id
* Vendor\_Id

InvoiceLine

* Id
* OurItemNumber
* YourItemNumber
* LineQuantity
* LineAmount
* TaxAmount
* ItemWeight
* TransportAmount
* LineDescription
* LineUOM
* PackSize

Invoice

* Id
* InvoiceNumber
* InvoiceDate
* InvoiceType
* RefInvoiceNumber
* OrderNumber
* InvoiceAmount
* TaxAmount
* InvoiceAmountInc
* NumberOfLines
* TotalQuantity
* TotalWeight
* AcPercentage
* CgrnNumber
* COrderNumber
* OrderLine\_Id

UploadedInvoice

* Id
* InvNo
* OrdNo
* VendorNo
* FileName
* Status
* Message
* Uploaded

OrderLine

* Id
* OrdLn
* ItemNo
* ItemDesc
* MfrItem
* QtyConv
* OrdQty
* PurcUom
* PurcUomConv
* TaxCde
* TaxRate
* UnitPrc
* LineTotExcl
* LineTotTax
* LineTotVal

Order

* Id
* resend
* OrdNo
* OrdDate
* OrdDesc
* OrdType
* OrdTerm
* OrdTermDesc
* OrdStat
* OrderStatus
* Origin
* PromDate
* CompName
* BranchNo
* BranchName
* BranchAddr1
* BranchAddr2
* BranchTel
* BranchFax
* BranchEmail
* BranchVat
* VendorRef
* VendorNo
* VendorName
* VendorAddr1
* VendorAddr2
* VendorSuburb
* VendorCity
* VendorContact
* TotLines
* TotQty
* TotExcl
* TotTax
* TotVal
* DelivAddr1
* DelivAddr2
* DelivSuburb
* DelivCity
* BuyerNote
* ConfirmInd
* CompID
* OrderLine\_Id

ConfirmOrdersRequest

* Id
* Order\_ID

ConfirmedOrder

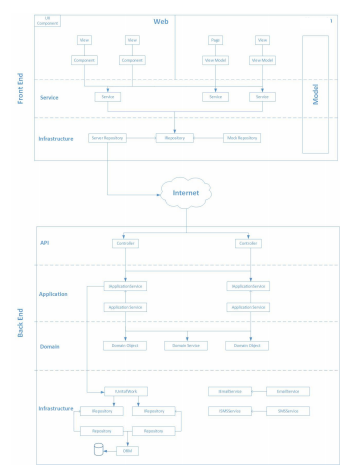
* Id
* OrdNo
* Status
* Message
* Confirmed

Vendor

* Id
* SupplierId
* Code
* Ean
* SellerEan
* SystemAdded
* Vendor

# Software Architecture

A layered architecture is used both within the website and backend design.



## Backend

* API Layer - responsible for exposing your system for consumption. The technology chosen to perform this function is a RESTful API. A RESTful API uses standard http verbs (GET, POST, DELETE, and PUSH) to communicate over the secure http protocol. The API layer is also responsible for defining the cacheing strategy to be used by the app (i.e with a GET request, how long can the result be cached client side before going back to the server).
* Application Layer - this can be seen as the public interface of the system, all domain logic is hidden from external users and exposed through the application layer. The consumer of the system is the technology being used to expose it to the outside world, which in this case is a RESTful API.
* Domain Layer - the core of the system where the state of the system is stored and the business rules reside. All process diagrams defined within this document will be executed within the domain layer. This layer is not interacted with by any external entity and can only be accessed through the application layer.
* Infrastructure Layer - the role of the infrastructure layer in the contains all code that is not part of the system but needed by the system in order to function correctly, for example SMS, Email, logging, integration and data access.

## Software Technology

Development Language:

Frontend/Backend – ASP.NET MVC CORE

Platform:

Microsoft ASP.Net 4.8/Net 6

Latest Microsoft .Net Framework Runtime

Development Tool:

Visual Studio 2019/22 – Microsoft development studio used to develop applications on .Net Framework.

# Security

The Woerman API is built on Secure by Design principles as prescribed within the various Vodafone Cyber Security policies. These policies include the following areas:

* Acceptable Usage
* Access and Authentication
* Cloud Computing Security
* Cyber Security Incident Management
* Cyber Security Testing
* Hardening Guidelines
* Information Security Classification and Protection
* Logging and Monitoring
* Secure System Management and Protection

## Infrastructure

Servers consist of application and database servers. Access is only possible thru the EC FW and policies determine access between the different servers. As per the design the DB servers are not accessible from the website and only from the application level where policies enforce this access. Server access is maintained by ECS-Infrastructure SysAdmin and has to be requested via Whitepages whereby the ECS line manager approves this request.

## Application

Application is managed with user/password and 2FA authentication. OWASP principles enforce token authentication.

Annual Penetration testing of the application is required by Vodacom Cyber Security which is conducted by external vendors.

## Data

Data is encrypted in transit over HTTPS protocol and encrypted at rest.

TLS1.2 applies which is enforced from the F5 and the following ciphers apply:

* TLS\_ECDHE\_RSA\_WITH\_AES\_128\_ GCM\_SHA256 (0xc02f)
* TLS\_ECDHE\_RSA\_WITH\_AES\_256\_ GCM\_SHA384 (0xc030)

As per Software Design data is only accessible thru the application layer and is not exposed externally. All ECS IoT & EM applications are built on the same architecture which

Provides secure decoupling of all layers within the application.

# Backups

Daily snapshots will be done on application and database servers and scheduled on Netbackup managed by ECS Backup team. A weekly full back will also be scheduled.

# Disaster Recovery and Business Continuity

This is an internal application and DR capability will not be required. Backups will be sufficient to restore.

2 Application servers to be deployed to fulfil application HA and load balancing.

The SQL DB will be hosted on the existing EC SQL Cluster which is built on HA failover capability.

# Auditing

Auditing needs to implemented for reporting capabilities required on any system and data changes. History needs to be kept on all data and should be timestamped that will tie in with auditing. Every change in terms of the CRUD model needs to be recorded, timestamped and which user implemented the change.