**Corleone Calzone Pizza**

Sjabloon Ontwerp

Volgens de ontwikkel concepten UML zijn de hierna volgende producten

uitgewerkt:

* Functioneel ontwerp
* Technisch ontwerp
* Bouw ontwerp

Dit document is opgesteld op basis van de richtlijnen die zijn opgesteld voor het afnemen van MBO examens ontwerp en bouwen.

In dit document is ook opgenomen het itereren conform Agile/Scrum

MBO Rijnland Gouda

Versie : 0.1

Datum : 26 mei 2023

Auteurs : Roan van Dam, Max Shirazi

**Inhoud**

[1 Inleiding 2](#_Toc21590)

[2 Opdrachtomschrijving 3](#_Toc20690)

[2.1.1 Opdrachtomschrijving 3](#_Toc5475)

[2.1.2 Probleemstelling 3](#_Toc31380)

[2.1.3 Oplossing 3](#_Toc20353)

[2.2 Afstemmen met betrokkenen 3](#_Toc11181)

[2.3 Maak een overzicht User Stories en een product Backlog 4](#_Toc3329)

[3 Planning en begroting 5](#_Toc13159)

[4 Functioneel ontwerp 7](#_Toc7489)

[4.1 Uitvoeren iteraties 7](#_Toc21504)

[4.2 Context Klassen diagram (domein analyse) 8](#_Toc28815)

[4.3 Use case diagram met beschrijving (domein analyse) 9](#_Toc16236)

[4.4 Database Ontwerp (ERD) 10](#_Toc28777)

[4.4.1 Conceptueel model (ERD) 11](#_Toc8995)

[4.4.2 Normaliseren 12](#_Toc7824)

[4.4.3 Implementatie model 13](#_Toc871)

[4.4.4 Relationeel Representatie model 13](#_Toc22972)

[5 Technisch ontwerp 15](#_Toc27040)

[5.1 Inleiding 15](#_Toc23803)

[5.2 Wireframes 15](#_Toc18118)

[5.3 Mock-Ups 16](#_Toc30464)

[5.4 Keuze framework 16](#_Toc12785)

[6 Bouw 17](#_Toc8428)

[6.1 Architectuur Client/Server 17](#_Toc11395)

[6.2 Bouw ontwerp modules 17](#_Toc6229)

[6.3 Logische module beschrijving 18](#_Toc8331)

[6.4 Fysieke module beschrijving 18](#_Toc32410)

[7 Versie beheer 18](#_Toc13132)

[8 Testen 18](#_Toc20421)

[9 Opleveren (implementeren) 18](#_Toc5403)

[10 Resultaat 19](#_Toc21897)

[11 Versie beheer 19](#_Toc226)

# Inleiding

Dit document is voor de webshop van Corleone Calzone. In dit document staan de volgende punten omschreven:

1. Opdrachtomschrijving

* Probleemstelling
* Oplossing
* Planning en begroting
* Functioneel ontwerp
* Technisch ontwerp
* Bouw ontwerp
* Testen
* Presenteren

# Opdrachtomschrijving

### Opdrachtomschrijving

De heer Corleone Calzone wilt zijn pizzaketen genaamd Corleone Calzone Pizza uitbreiden door een webshop te maken. In de webshop wil Corleone Calzone dat de bezoeker kan inloggen, gegevens van hunzelf kan veranderen en producten in hun winkelwagen kan plaatsen. De bezoeker kan ook een bestellingen plaatsen die in de database wordt gezet. Corleone Calzone wilt ook dat er een admin rol is die producten kan aanmaken, veranderen en verwijderen. Bij een bestelling moet er een keuze gemaakt kunnen worden tussen een medium pizza, large pizza en een Calzone pizza. Als een 2e pizza is besteld wordt de prijs van de 2e pizza met 50% verlaagd.

### Probleemstelling

1. De heer Corleone Calzone wilt zijn pizzaketen uitbreiden.
2. De bezoeker heeft een account nodig om te kunnen bestellen.
3. Corleone Calzone heeft een site nodig waarop hij producten kan aanmaken, veranderen en verwijderen.
4. Corleone Calzone moet korting kunnen toepassen aan de producten

### Oplossing

1. Wij maken een webshop voor de heer Corleone Calzone.
2. De bezoeker krijgt de kans om een account aan te maken om zo te kunnen bestellen. Bovendien kan de bezoeker ook inloggen om te kunnen bestellen.
3. Als er op de inlogpagina de gegevens van een admin zijn ingevuld krijgt de persoon toegang tot de adminpagina waarop je producten kan aanmaken, veranderen en verwijderen.
4. Als er van meerdere van hetzelfde product wordt verkocht krijgt de klant korting.

## Afstemmen met betrokkenen

Pitchen/presenteren

## Maak een overzicht User Stories en een product Backlog

***User story***

User stories passen goed in een Agile/Scrum frame-work.

Een user story is geen functionele beschrijving, maar het maakt duidelijk wat een (eind-) gebruiker wil en waarom.

Bij het opstellen van een user-story moet rekening gehouden worden met de hierna volgende punten:

* Het is een functionele wens van de gebruiker
* Het is in gebruikerstaal geschreven
* Het heeft een toegevoegde waarde voor de gebruiker
* Het is een discussie en geen contract

Een userstory wordt als volgt opgesteld:

Als {wie} wil ik {wat} zodat ik {waarom}

Hieronder wat voorbeelden:

*“Als een manager wil ik een dashboard hebben waarop ik de prestaties van mijn team kan zien, zodat ik de voortgang van projecten kan monitoren en bijsturen indien nodig.*

*“Als een student wil ik feedback ontvangen op mijn huiswerkopdrachten, zodat ik mijn prestaties kan verbeteren en mijn cijfers kan verhogen.”*

*“Als een softwareontwikkelaar wil ik toegang hebben tot de documentatie van de API, zodat ik mijn code kan schrijven op basis van de juiste specificaties.”*

*“Als een marketingmedewerker wil ik rapportages kunnen genereren van de websitebezoekers, zodat ik de effectiviteit van marketingcampagnes kan meten en verbeteren.”*

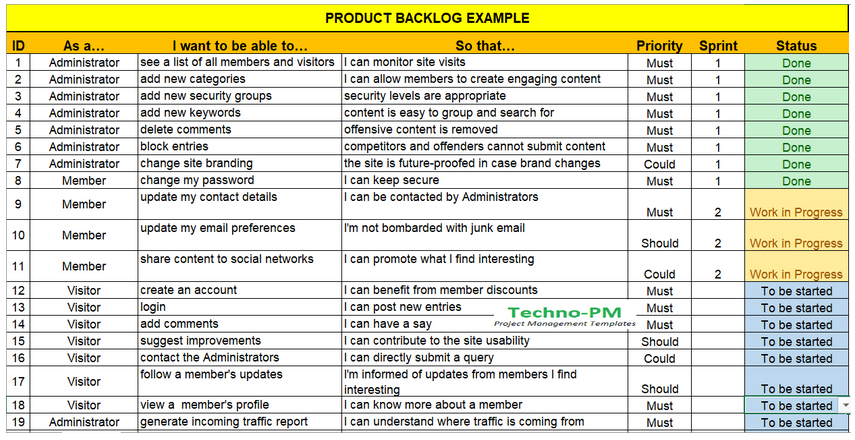
*“Als een forens wil ik real-time updates ontvangen over het openbaar vervoer, zodat ik op tijd op mijn bestemming aankom en mijn reistijd kan optimaliseren.”*

*“Als een online shopper wil ik de mogelijkheid hebben om producten te vergelijken, zodat ik de beste keuze kan maken en geld kan besparen.”*

***Product Backlog:***

Het ordenen van user-stories gebeurt in een zogenaamde “product backlog” en deze voorzien van prioriteiten conform, bijvoorbeeld MoSCoW.

Hieronder een voorbeeld:



# Planning en begroting

Planning en begroting betreffen de hieronder genoemde activiteiten:

**Projectfases:**

Initiatief Analyse (opdrachtsomschrijving, probleemstelling en oplossing

Ontwerp Functioneel- en technisch ontwerp

Realisatie Bouwen en testen

Implementatie implementatie plan (in productie nemen) en acceptatie-testen

Evaluatie Opleveren en resultaten evalueren

“Planning en Begroting” kan gewoon op 1 excel-lijst worden gezet.

**Begroting**

Een begroting is een inschatting. Alle activitenten projectfases op een lijst zetten met inschatting uren en tarieven. Hieronder een voorbeeld:

Afbeelding met tafel

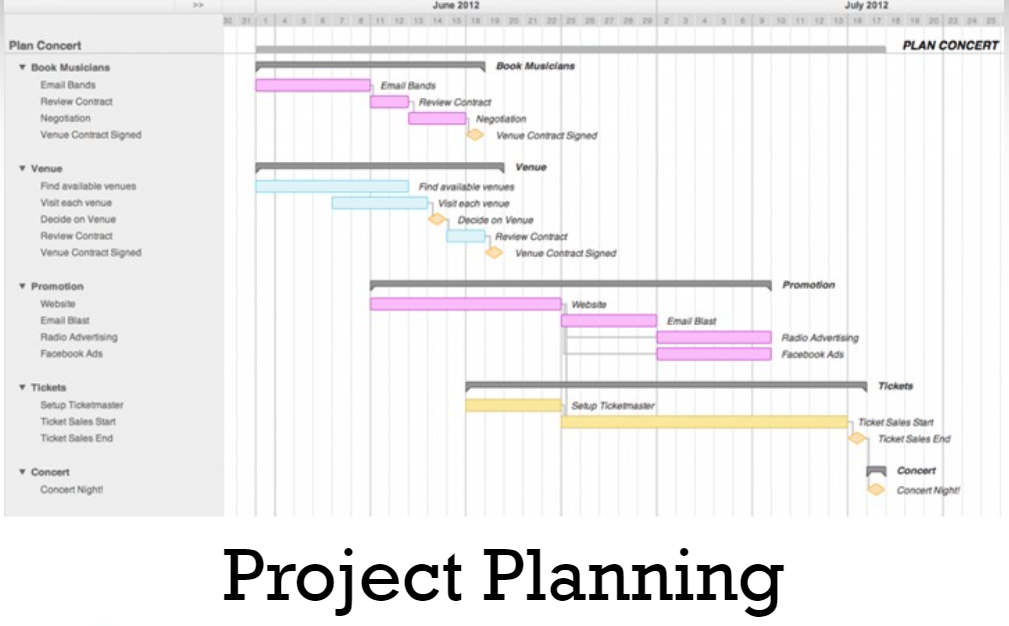
Automatisch gegenereerde beschrijving

**Planning:**

Een planning is een inschatting als het gaat om wanneer te beginnen aan welke taak, en wanneer is het af.

* Neem als basis de overzicht van de activiteiten (zie begroting)
* Bepaal prioriteiten (welke volgorde zal een activiteit worden uitgevoerd)
* Wie doet wat
* Een tijdslijn opstellen wanneer te beginnen en wanneer te eindigen

Hieronder een voorbeeld:



# Functioneel ontwerp

In dit hoofdstuk worden een x-aantal uit te voeren stappen/activiteiten uitgewerkt.

## Uitvoeren iteraties

Plan en plaats de user stories. Schat onder andere de tijd en de prioriteit/volgorde in (bijvoorbeeld met de MoSCoW-methode). Plaats de user stories op de iteratie/sprint backlog.

Verdeel binnen het team de eerste op te pakken user stories. Zorg voor een evenwichtige verdeling.

Werk de eigen user stories uit in taken, criteria en een ontwerp. Stel hierbij, passend bij de opdracht, de volgende documenten op of voer de volgende activiteiten uit, zoals:

* Klassendiagram (context)
* Usecase diagram
* ERD (datamodel)
* Normaliseren
* Wireframes
* Mock-ups
* Activiteitendiagram
* Keuze framework

## Context Klassen diagram (domein analyse)

Lijst van objecten

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr** | **klasse** | **omschrijving** | **relevant** |
| 1 | pizzaketen | Het bedrijf van de heer Corleone Calzone | Nee |
| 2 | webshop | De website van de heer Corleone Calzone | Ja |
| 3 | bezoeker | De persoon die de website bezoekt | Ja |
| 4 | inloggen | Functie om toegang te krijgen tot bestellen | nee |
| 5 | gegevens | Waardes van de ingelogde | Nee |
| 6 | bestelling | Functie om te kunnen betalen voor producten | Ja |
| 7 | admin | Rol die toegang heeft tot het aanmaken, veranderen en verwijderen van een product op de website | Ja |
| 8 | Product | Materiaal dat je kan kopen | Ja |
| 9 | Pizza | Het hoofdproduct van de website | nee |
| 10 | prijs | De waarde van het product | nee |

### Context Klassen diagram

Zie bestand “klassendiagram webshop\_Corleone\_Calzone\_Pizza v0.1”.

## Use case diagram met beschrijving (domein analyse)

***Use Case Diagram***

Afbeelding met diagram, tekst, cirkel, patroon

Automatisch gegenereerde beschrijving

Data Dictionary Use Case:

|  |  |
| --- | --- |
| UseCase Naam | Login |
| Samenvatting | Functie gegevens bijhouden. |
| Actoren | Niet ingelogde bezoeker, ingelogde bezoeker, admin |
| Aannamen | De persoon is op het url van de website gekomen. |
| Beschrijving | Klant bezoekt website:  De persoon krijgt de kans om zijn/haar gebruikersnaam en wachtwoord in te vullen.  De persoon klikt op de login knop.  De persoon kan ook een knop aanklikken om naar een andere pagina te gaan waarop hij een nieuw account kan aanmaken. |
| Uitzonderingen | Als er misinformatie is ingevuld krijgt de persoon een foutmelding op het veld waar het fout is gegaan. |
| Resultaat | De persoon komt op de webpagina.  Als de gegevens van de admin zijn ingevuld komt de persoon op de adminpagina. |

|  |  |
| --- | --- |
| UseCase Naam | winkelwagen |
| Samenvatting | Opslag van nog niet betaalde producten waar de klant interesse voor heeft. |
| Actoren | Ingelogde bezoeker |
| Aannamen | De ingelogde bezoeker klikt op de knop “voeg toe aan winkelwagen” bij een product. |
| Beschrijving | Knop geklikt:  het product wordt toegevoegd aan een sessie genaamd “winkelwagen”.  Er kunnen meerdere producten worden toegevoegd aan de winkelwagen.  De prijs van alles in de winkelwagen wordt automatisch berekent.  De bezoeker kan ook een product verwijderen uit zijn/haar winkelwagen. |
| Uitzonderingen | Als een product niet goed is opgeslagen krijgt de bezoeker een foutmelding.  Als er geen product is krijgt de bezoeker niet de kans om zijn/haar bestelling te plaatsen. |
| Resultaat | De bezoeker kan de gehele inhoud van zijn/haar winkelwagen bestellen. |

|  |  |
| --- | --- |
| UseCase Naam | product |
| Samenvatting | Alle functies die gerelateerd zijn aan de producten. |
| Actoren | Admin |
| Aannamen | De admin is succesvol ingelogd. |
| Beschrijving | Er zijn 3 functies voor het product:  product aanmaken  product wijzigen  product verwijderen |
| Uitzonderingen | Als de functie niet volledig is afgemaakt is deze niet bereikbaar voor de admin |
| Resultaat | De admin kan kiezen uit de functies |

|  |  |
| --- | --- |
| UseCase Naam | 3a. product aanmaken |
| Samenvatting | Functie om een product toe te voegen aan de database. |
| Actoren | Admin |
| Aannamen | De admin heeft de functie “product aanmaken” gekozen door te klikken op de knop “product aanmaken”. |
| Beschrijving | Het product aanmaken:  alle producten zijn opgeslagen in een database.  het product heeft een naam, omschrijving, plaatje en prijs.  de admin moet alle gegevens invullen om het product aan te maken. |
| Uitzonderingen | als niet alle gegevens zijn ingevuld komt er een foutmelding bij het veld waar het is fout gegaan.  Als er een waarde is gegeven die niet opgeslagen kan worden in de database zoals een woord bij een cijferwaarde komt er een foutmelding bij het veld waar de incorrecte waarde is ingevuld. |
| Resultaat | Het product wordt opgeslagen in de database. |

|  |  |
| --- | --- |
| UseCase Naam | 3b. product wijzigen |
| Samenvatting | Functie om de gegevens van een product aan te passen. |
| Actoren | Admin |
| Aannamen | De admin heeft de functie “product wijzigen” gekozen door te klikken op de knop “wijzigen” onder het gewenste product. |
| Beschrijving | Het product wijzigen:  Alle producten zijn opgeslagen in een database.  Het product heeft een naam, omschrijving, plaatje en prijs  De bestaande gegevens van het product zijn ingevuld in de overeenkomende velden.  De admin krijgt de keuze om alle gegevens in de velden te wijzigen. |
| Uitzonderingen | Als een veld leeg is komt er een foutmelding bij dat veld en wordt de wijziging niet toegebracht.  Als een veld een waarde heeft die niet opgeslagen kan worden in de database komt er een foutmelding bij dat veld en wordt de wijziging niet toegebracht. |
| Resultaat | de wijziging is succesvol toegepast in de database. |

|  |  |
| --- | --- |
| UseCase Naam | 3c. product verwijderen |
| Samenvatting | Functie om een product te verwijderen in de database. |
| Actoren | Admin |
| Aannamen | De admin heeft de functie “product verwijderen” gekozen door te klikken op de knop “verwijderen” onder het gewenste product. |
| Beschrijving | Het product verwijderen:  Alle producten zijn opgeslagen in een database.  Als op de knop is gedrukt komt er een comfirm box die vraagt of de admin zeker weet dat hij het product wilt verwijderen.  Het product wordt verwijdert uit de database. |
| Uitzonderingen | Als de admin “nee” klikt bij de comfirm box “weet u zeker dat u dit product wilt verwijderen” wordt het product niet verwijdert. |
| Resultaat | Het product wordt verwijdert uit de database. |

|  |  |
| --- | --- |
| UseCase Naam | Foutmelding |
| Samenvatting | Melding bij uitzonderingen. |
| Actoren | Admin, ingelogde bezoeker, niet ingelogde bezoeker |
| Aannamen | Er is iets fout gegaan bij één van de functies. |
| Beschrijving | Er komt een foutmelding:  De foutmelding is gerelateerd aan de functie.  De foutmelding komt alleen tevoorschijn als de functie niet voldaan kan worden. |
| Uitzonderingen | - |
| Resultaat | Melding is succesvol gebracht bij de persoon. |

## Database Ontwerp (ERD)

### Conceptueel model (ERD)

Zie bestand: concept ERD corleone\_calzone\_pizza v0.1.drawio

### Normaliseren

0e normaalsvorm {

User: user\_id(pk), user\_name, password, is\_admin, created\_at, address, postal\_code

Product: product\_id(pk), product\_name, description, price, stock

Order: order\_id(pk), user\_id(fk), product\_id(fk), order\_date

}

1e normaalsvorm {

User: user\_id(pk), user\_name, password, is\_admin, created\_at, address, postal\_code

Product: product\_id(pk), product\_name, description, price, stock

Order: order\_id(pk), user\_id(fk), product\_id(fk), order\_date

}

2e normaalsvorm {

User: user\_id(pk), user\_name, password, is\_admin, created\_at, address, postal\_code

Product: product\_id(pk), product\_name, description, price, stock

Order: order\_id(pk), user\_id(fk), product\_id(fk), order\_date

}

3e normaalsvorm {

User: user\_id(pk), user\_name, password, is\_admin, created\_at

Product: product\_id(pk), product\_name, description, price, stock

Order: order\_id(pk), user\_id(fk), product\_id(fk), user\_order\_address(fk), order\_date

User\_order\_address: user\_address\_id(pk), user\_id(fk), address, postal\_code

}

### Implementatie model

Zie bestand: implementatie model ERD corleone\_calzone\_pizza v0.1.drawio

***Datadictionary ERD***

|  |  |  |
| --- | --- | --- |
| ***Nr*** | ***KLasse*** | ***Omschrijving*** |
| 1 | bezoeker | File Name: user |
| 2 | bestelling | File Name: order |
| 3 | product | File Name: product |
| 1 | bezoeker | File Name: user\_order\_address |

### Relationeel Representatie model

De namen van de velden ten behoeve van de op te bouwen bestanden zullen in groepen worden benoemd. Zo nodig zullen ook de primary- en secundary –keys worden beschreven.

***Opslagstructuur database namen velden: user***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nbr | Field name | From | To | Length | Type | Description |
| 1 | user\_id |  |  |  | medInt | Primary Key (Unique) |
| 2 | User\_name |  |  | 30 | Char |  |
| 3 | password |  |  | 15 | Char | Saved as password\_hash |
| 4 | Is\_admin |  |  |  | boolean | Gegeven dat kijkt of de user een adminrol heeft |
| 5 | Created\_at |  |  |  | date | Time\_Stamp van gecreëerd account |

***Opslagstructuur database namen velden user\_order\_address***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nbr | Field name | From | To | Length | Type | Description |
| 1 | User\_address\_id |  |  |  | medint | Primary Key (Unique) |
| 2 | User\_id |  |  |  | medint | Foreign key (van tabel user) |
| 3 | address |  |  | 255 | Char |  |
| 4 | Postal\_code |  |  | 6 | Char |  |

***Opslagstructuur database namen velden product***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nbr | Field name | From | To | Length | Type | Description |
| 1 | Product\_id |  |  |  | bigint | Primary Key (Unique) |
| 2 | Product\_name |  |  | 50 | Char |  |
| 3 | description |  |  |  | TEXT |  |
| 4 | price |  |  |  | float | Waarde in euro’s |
| 5 | stock |  |  |  | medint | Voorraad van het product |

***Opslagstructuur database namen velden order***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Nbr | Field name | From | To | Lenght | Type | Description |
| 1 | Oder\_id |  |  | 300 | bigint | Primary Key (Unique) |
| 2 | User\_id |  |  |  | medint | Foreign Key (van tabel user ) |
| 3 | Product\_id |  |  |  | bigint | Foreign Key (van tabel product) |
| 4 | user\_address\_id |  |  |  | medint | Foreign Key (van tabel user\_order\_address) |
| 5 | Order\_date |  |  |  | date | Gegeven dat opslaat wanneer de bestelling is geplaatst |

# Technisch ontwerp

## Inleiding

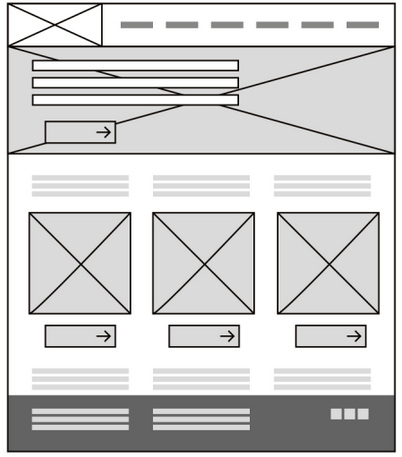
Dit hoofdstuk zal een gedetailleerde beschrijving bevatten van de technische specificaties van de te ontwikkelen applicaties inzake het *{Ontwikkel\_Systeem}*. Het is tevens een handvat voor de te bouwen applicaties ten behoeve van dit systeem met de hierna volgende inhoud:

* Wireframes
* Mock-ups
* Keuze Framework

## Wireframes

Wireframes is een bouwtekening van een website (schermprogramma) waarin een overzicht wordt gegeven van de verschillende onderdelen die op een website (schermprogramma) aanwezig zullen zijn.

Hieronder een voorbeeld:



Refereer puntgewijs de functionaliteiten vanuit de product-backlog (alle functionaliteiten uit het functioneel ontwerp)

## Mock-Ups

Een mock-up id een op schaal gemaakte model van een ontwerp of product. Voor een nog te bouwen website/schermprogramma maak je een “dummy website” zodat een goed beeld kan worden verkegen hoe de website (schermprogramma) in het echt eruit zal zien.

## Keuze framework

Een framework is een soort sjabloon (een voorbeeld/macro) voor het bouwen van een website (schermprogramma).

***De voordelen hiervan:***

* De basis bestaat al, je hoeft niet vanaf 0 te beginnen
* Het scheelt tijd
* Het zou een standaard kunnen zijn
* Het is minder foutgevoelig

# Bouw

In dit hoofdstuk worden een x-aantal uit te voeren stappen/activiteiten uitgewerkt.

## Architectuur Client/Server

Web-Architectuur:

Zie document: “client\_server model Corleone Calzone Pizza v0.1”.

**Legenda:**

|  |  |  |
| --- | --- | --- |
| **Nr** | **Naam** | **Omschrijving** |
| 1 | Web Browser | In de website worden de volgende codeertalen gebruikt: html,css, javascript en php. Html wordt gebruikt om de basisinformatie van de website vast te stellen. Css wordt gebruikt voor alle styling van de website. Javascript wordt gebruikt om alle lastige animaties te regelen. Php wordt gebruikt om de database opdrachten uit te voeren. |
| 2 | HTTPS server | De website wordt gehost op de https server. |
| 3 | ODBC mySQL | De interactieve database wordt gemaakt en geregeld in mySQL |
| 4 | DBMS | De database is beveiligd |

## Bouw ontwerp modules

**Module- modellen:**

Het modulemodel komt tot uiting tijdens de bouw van de programmatuur en zal worden opgedeeld in een logische- en een fysieke module.

[

Hoe te beginnen:

* Ontwerp logische module
* logische modules -> Software Program Flow {program structure design}
* fysieke modules -> implementatie van de logische module {welke software taal}

]

## Logische module beschrijving

[

Maak een kopie van het concept klassendiagram, werk entiteiten/attributen bij zoals bestanden ontworpen volgens het ERD model.

Maak klassenmodel compleet voor het OOP model (voeg bij kenmerken de nodige methoden(functies) toe. (zie ook les documentatie ontwerpen en bouwen volgens OOP)

Beschrijvingen middels Program Flow diagrammen

]

Flowchart: zie bestand “flowchart\_logische module Corleone Calzone Pizza v0.1”.

## Fysieke module beschrijving

In dit hoofdstuk zullen de aan te passen cq nieuw te bouwen programmatuur worden beschreven.

**De Programmas**

|  |  |  |
| --- | --- | --- |
| Nr | Programmanaam | Omschrijving |
| 1 | Pgm\_naam\_html001 |  |
| 2 | Pgm\_naam\_html002 |  |

***Controles Pgm\_naam\_htlm001:***

De hierna volgende controles moeten worden uitgevoerd:

[

Speficiceer de controles voor invoer

]

# Versie beheer

Het op na houden van versie beheer. Dit betreft alle documenten, rapporten en programmatuur

# Testen

Maak van alle functionaliteiten (zie ook usecase, product backlog) een testplan en de per functie een test-scenario. Stel de test-scripten op.

# Opleveren (implementeren)

Lever het product op aan de klant (review).

* Houd bijvoorbeeld een demo of een presentatie. Ieder lid van het projectteam presenteert hierbij een of meer user stories.
* Stel naar aanleiding van de oplevering verbetervoorstellen voor het product op indien nodig.
* Bundel de verbetervoorstellen van de integratietest en review van jouw user stories in een document.

# Resultaat

Als resultaat van deze opdracht lever je de volgende producten en/of diensten op.

* Document uitgangspunten, eisen en wensen
* Notulen reflectiemeeting
* Applicatie/user stories
* Document met verbetervoorstellen

# Versie beheer

Revisie en opmerkingen

|  |  |  |  |
| --- | --- | --- | --- |
| Nr | Datum | Auteur | Omschrijving |
| 1 | 17-04-2023 | An | Initiële set-up / concept ontwerp |
| 2 | 26-05-2023 | Roan van Dam, Max Shirazi | Ontwerp ingevuld |