System specification: MyFood

Document information

|  |  |
| --- | --- |
| Version | 0.3 |
| Classification | Confidential / **Internal /** Approved partner |

Modification history

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Changes | Author | Date |
| 0.1 | First draft | Herbert Scheffknecht | 2012-05-07 |
| 0.2 | Translation to english, more content | Herbert Scheffknecht | 2012-06-01 |
| 0.3 | Added content to production / technical implementation | Herbert Scheffknecht | 2012-06-08 |

This document is property of CreativeLion. CreativeLion is a brand by xite-media AG. No part of it maybe reproduced or used in any form or by any means without the written permission of the owner.

© 2012 CreativeLion - All rights reserved

**Content**

**1 Introduction**

**2 Frontend**

**2.1 User account**

**3 Backend**

**4 Payment**

**5 Production**

**6 Delivery**

**6.1 Deliver ytime**

**6.2 Duty**

**7 Technical implementation**

**7.1 Coding standards**

**7.2 Database conventions**

# Introduction

MyFood is the name of a website, which allows simple und beautiful ordering for food and drinks. The order process should be simple and intuitive, on the other hand a pretty product presentation is very important – the customer should become hungry when visiting MyFood.

The user have the choice of small number of foods, but he can configurate the food like he wish:

* He can add and remove side dishes or additions like ketchup
* He can additional order drinks

After that, he can order with a few steps. The system supports this with autmatic location detection, auto login, etc, to avoid unneeded user input. Every input should be saved, until he log out explicitly.

The user interface is splitted in 3 parts:

* The frontend, where the user orders his food
* The user account: This is a part of the frontend, where the user can log in, see his orders and changes his data.
* The backend: A own website, where only the employees of MyFood can log in. This is for administration, support and production the website to do their work.

# Frontend

The frontend is the main website for the order process and contains the following pages and functions:

* Homepage / location detection / choose dishes
* Food configuration
* Shopping basket
* Delivery address form
* Payment method form
* Checkout
* Static pages
  + AGB
  + Impressum
  + Kontaktformular
  + Hilfeseiten

## User account

The user account is the part of the frontend, where the user changes their data and shows his order history and status and contains the following pages and functions:

* Register page
* Login / Logout (accessible in page header)
* Password forgotten
* Change password and user data
* List of all addresses
* List of all bank cards
* Order history (inkluding order status)
* Pages for business customers
  + List of all employee users
  + Order history of all employee users (extending the existing order history)
  + List of all invoices (collective invoices) and payment status

# Backend

The backend is only accessible for the employees of MyFood. The access is protected through user/password and IP protection. The backend is a website independ to the Frontend, but uses the same MyFood database as the frontend.

The backend has the following pages and functions:

* Login
* Statistics of all orders and articles
* List / administration of all invoices
* List / administration of all users
* List / administration of all orders
* Possibility to login as a user on the frontend
* Interfaces to Zendesk for support
* Interface to accounting software

# Payment

The following payment methods are available:

* Credit card (pay during order process)
* Debit (pay during order process)
* Cash on delivery

If possible, the money should only tranfered, when the delivery is confirmed. Is this is not possible, the money is tranfered during order confirmation.

The delivery man has always to know if the user has paid during order process (credit card / debit) or if the delivery man has to collect the money.

# Production

The food is producted central in Feldkirch for the whole Rheintal and Liechtenstein. Some small kitchens only warm-up the dishes and deliver it to the customer.

The system support the production, to cook and deliver as many dishes as possible. The system supports them on the following ways:

* The chef shouldn’t cook each dish separately, they should cook in batches (many orders at the same time).
* The packaging of the orders should happen in the correct sorting, that the delivery man has no additional work.
* The system plans the routes of the delivery man, to plan the correct cooking sorting and batches.
* Creating and printing the receip for production and the customer (is the invoice for the customer)
* Printing labels to glue it on the food packaging.

## Production batches

One batch is a collection of orders, which are produces at the same time. The batch is generated by the system, the different orders are collected on one or more of the following similarities:

* Similar delivery target
* Similar dishes (same main dishes)
* Similar order time (the old first)

For each production batch, one A4 sheet with labels are printed (on a Zweckform labels sheet). The maximum batch size is to fill this paper. The batch generation interval is 10 minutes. That means, the system fetch every 10 minutes all orders and create as less batches as possible but group the orders by the criteria above.

# Delivery

The delivery is dependant on the number of orders within an area:

* For few orders, they drive with mopeds to the customers
* For more orders, they drive with car pairwise (1 is the driver, 1 deliverys the orders)
* For many orders, they drive with bus (1 is the driver, so many delivery men as required delivery the orders)

The delivery drivers are supported at the system. Each one has a mobile device with a MyFood app, which supports them with the following functions:

* Showing the calculated route to the next customer
* Allowing to customer to sign the delivery digitally
* Confirming the order to the server (inkl. the digital signature)
* Allowing to enter customer complaints and send them to the server and change the price (i.e. when the food is to late, cold, etc.)

## Deliver ytime

The system tries to estimate the duration of the food. This is possible through the real time order confirmation, distance, utilization, etc., which is logged with each order and is used for delivery time calculation.

## Duty

Delivery to Liechtenstein could become problematic, because they must transported through the custom office. This may be takes to long. This can be solved by doing the duty as ealy and possible and automatically.

# Technical implementation

For the technical implementation, the following technologies are used:

* Linux servers with Apache server
* PHP with Zend Framework (using MVC pattern)
* MySQL database server

On client side, we use the following technologies:

* LESS: <http://lesscss.org/>
* jQuery: <http://jquery.com/>
* Bootstrap: <http://twitter.github.com/bootstrap/>

The developed is supported with the following software:

* Code managing: Subversion (or Git?)
* Bugtracking: JIRA

The backend and frontend systems are separeted projects, common used components should handled within an own project “MyFood”, which is used from both projects as a common library.

For backend processes, there are 2 ways:

* Creating a CLI php script, which uses the “MyFood” library.
* If this is not possible because of the limits of PHP, the library can be ported to Mono, and a Mono application is build for this task.

## Coding standards

For development, we use the standards and recommendations of Zend Framework:

* “Zend Framework Coding Standard for PHP”: <http://framework.zend.com/manual/en/coding-standard.html>
* “Zend Framework Documentation Standard”: <http://framework.zend.com/manual/en/doc-standard.html>
* “Recommended Project Structure for Zend Framework MVC Applications”:  
  <http://framework.zend.com/manual/en/project-structure.html>

## Database conventions

* Tables are in plural and following the naming conventions of PHP classes (i.e. “Users” or “GuestbookEntries”)
* Columns are in singular and following the naming conventions of functions and methods (i.e. “userName”)
* Each table contains a primary key. This can be the “id” column (auto increment value) or one or more foreign keys.
* Relations should be used where possible and useful.
* Use “NO ACTION” for “On Update” / “On Delete” actions on relations.
* Use NULL values only when needed, otherwise set the column to “NOT NULL”.
* Use underscore on tablesnames for m:n relations (i.e. “Users\_Roles”)