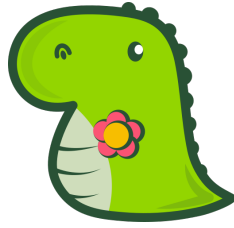


# Gardenzilla development notes

Peter Mezei

Tuesday 5<sup>th</sup> May, 2020



*Gardenzilla* development notes, questions and future development directions.  
The main goal of this document is to document the development steps and  
collect all the ideas for future evaluation.

All right reserved 2020  
All the logos, images and all the other  
used visual materials are copyrighted.

# Contents

<b>1</b>	<b>Introduction</b>	<b>4</b>
<b>2</b>	<b>Modules</b>	<b>5</b>
2.1	Customers . . . . .	5
2.1.1	Methods . . . . .	6
	<b>Todo list</b>	<b>7</b>

## **1 Introduction**

## 2 Modules

### 2.1 Customers

We manage customers as independent data objects, and manage a customer object through its implemented methods, or via public functions inside `core::customer` module.

Figure 1: Customer object

```
pub struct Customer {  
    id: String,  
    related_users: Vec<String>,  
    name: String,  
    tax_number: String,  
    address: InvoiceAddress,  
    phone: String,  
    email: String,  
    date_created: DateTime<Utc>,  
    created_by: String,  
}
```

Where:

**id**

Customer unique identifier

**related\_users**

vector of related user ID. User and Customer objects are independent of each other, as we can have customer without a user, and we can have - at least in theory - a user without a customer. Customer can purchase, User can login and interact withing the client. To manage Customer <-> User relations, we store them as a related ID list in each other. Here in a Customer object thats why we have this `related_users` vector.

**name**

name as string of the customer

**tax\_number**

tax number as string

**address**

address as `InvoiceAddress` object

**phone**

phone number as string

Describe and set the way we manage IDs.

display InvoiceAddress

**email**

email as string

**date\_created**

date as chrono::DateTime&lt;Utc&gt;

**created\_by**

userid as string, who created this object

**2.1.1 Methods****new(...) -> Self**

creates a new instance of Customer. ID cannot be changed after creation

**get\_id() -> &str**

returns customer id as &amp;str

**get\_name() -> &str**

returns customer name as &amp;str

**set\_name(name: String)**

set customer name to the given new one

**has\_user() -> bool**

returns bool if there is at least one related user

**get\_users() -> &Vec<String>**

returns related user vector reference

**remove\_user(userid: &str)**

remove a user by its userid

**get\_tax\_number() -> &str**

returns tax number as &amp;str

**set\_tax\_number(tax\_number: String)**

set tax number

describe  
datecreated  
and created  
by some-  
where as we  
use them  
everywhere

should re-  
turn bool, or  
result

## Todo list

Describe and set the way we manage IDs. . . . .	5
display InvoiceAddress . . . . .	5
describe datecreated and created by somewhere as we use them everywhere	6
should return bool, or result . . . . .	6

*Gardenzilla*  
Development Notes

All right reserved  
Peter Mezei  
2020