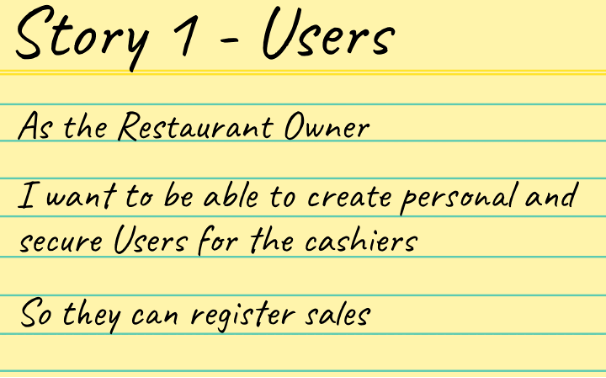
**3. Design of URIs**

Based on the User Stories and on the interview with the Restaurant Owner, we established 4 Business Rules that need to be implemented for the restaurant.

**3.1. User Story 1 Services**



**3.1.1. Service 1**

|  |  |
| --- | --- |
| **Method** | GET |
| **Description** | GET all the Users on DB |
| **URI** | http://{ip}:3005/restaurant/users |
| **Parameters** | None |
| **Format** | JSON |
| **Returns** | Http Codes:   * 200 for Success * 500 for Server Error |

**3.1.2. Service 2**

|  |  |
| --- | --- |
| **Method** | GET |
| **Description** | GET one User based on its username |
| **URI** | http://{ip}:3005/restaurant/users/{username} |
| **Parameters** | **username**: username of login for the User |
| **Format** | JSON |
| **Returns** | Http Codes:   * 200 for Success * 500 for Server Error |

**3.1.3. Service 3**

|  |  |
| --- | --- |
| **Method** | PUT |
| **Description** | PUT Update a User based on its username and for any attributes we want |
| **URI** | http://{ip}:3005/restaurant/users/{username} |
| **Parameters** | **username**: username of login for the User    You can use any of the parameters of a User and not necessarily all of them |
| **Format** | JSON |
| **Returns** | Http Codes:   * 200 for Success * 404 if there is no User with that username * 500 for Server Error at finding or Updating User |

**3.1.4. Service 4**

|  |  |
| --- | --- |
| **Method** | DELETE |
| **Description** | DELETE a User based on its username |
| **URI** | http://{ip}:3005/restaurant/users/{username} |
| **Parameters** | **username**: username of login for the User    You can use any of the parameters of a User and not necessarily all of them |
| **Format** | JSON |
| **Returns** | Http Codes:   * 200 for Success * 500 for Server Error at finding or Updating User |

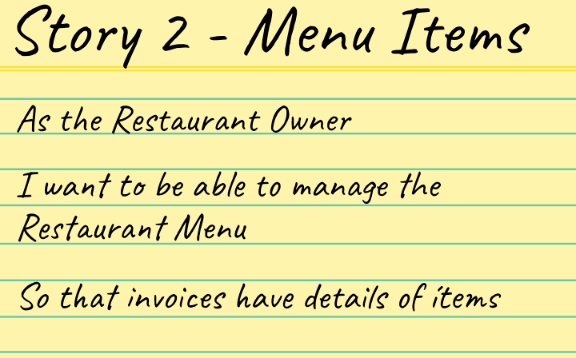
**3.1.5. Service 5 - Business Rule**

*For the Users to be safe, their passwords can’t be saved as plain text on the database.*

*We’ll use Bcrypt library to hash the passwords of new Users and we save these hashes which are secure in case someone gets that information.*

|  |  |
| --- | --- |
| **Method** | POST |
| **Description** | POST insert a new User but for security concerns, the password must be Hashed to save that and not the password |
| **URI** | http://{ip}:3005/restaurant/users |
| **Parameters** | You have to send all the User parameters and the password in plain text |
| **Format** | JSON |
| **Returns** | You can see the password has been Hashed and if someone has access to the User data, he can’t know the password.  Http Codes:   * 200 for Success * 500 for Server Error |

**3.2. User Story 2 Services**



**3.2.1. Service 6**

|  |  |
| --- | --- |
| **Method** | GET |
| **Description** | GET all the MenuItems on DB |
| **URI** | http://{ip}:3005/restaurant/menuItems |
| **Parameters** | None |
| **Format** | JSON |
| **Returns** | Http Codes:   * 200 for Success * 418 Not a Teapot. If you try to use the method for what it is not * 500 for Server Error |

**3.2.2. Service 7**

|  |  |
| --- | --- |
| **Method** | GET |
| **Description** | GET one MenuItem by its Code |
| **URI** | http://{ip}:3005/restaurant/menuItems/{code} |
| **Parameters** | **code**: code of menuItem |
| **Format** | JSON |
| **Returns** | Http Codes:   * 200 for Success * 500 for Server Error |

**3.2.3. Service 8**

|  |  |
| --- | --- |
| **Method** | POST |
| **Description** | POST a new MenuItem |
| **URI** | http://{ip}:3005/restaurant/menuItems |
| **Parameters** | You need to send all the attributes of a MenuItem |
| **Format** | JSON |
| **Returns** | We can see that the Decimal128 price parses correctly.  Http Codes:   * 200 for Success * 500 for Server Error |

**3.2.4. Service 9**

|  |  |
| --- | --- |
| **Method** | PUT |
| **Description** | PUT Update a MenuItem using its Code and the new attributes |
| **URI** | http://{ip}:3005/restaurant/menuItem/{code} |
| **Parameters** | **code**: code of menuItem    You don’t need to send all the attributes of a MenuItem, just the ones you want |
| **Format** | JSON |
| **Returns** | Http Codes:   * 200 for Success * 404 if there is not a Menu Item with that Code * 500 for Server Error at finding or Updating |

**3.2.5. Service 10**

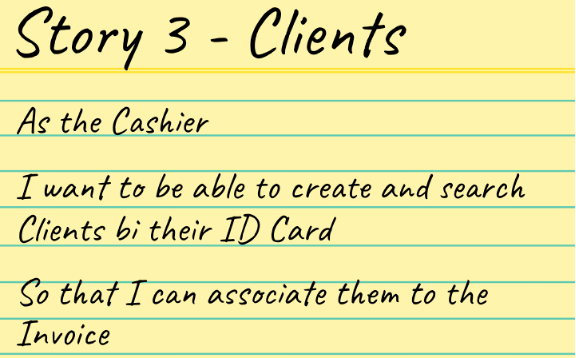
|  |  |
| --- | --- |
| **Method** | DELETE |
| **Description** | DELETE a MenuItem using its Code |
| **URI** | http://{ip}:3005/restaurant/menuItem/{code} |
| **Parameters** | **code**: code of menuItem |
| **Format** | JSON |
| **Returns** | Http Codes:   * 200 for Success * 500 for Server Error at finding or Updating |

**3.2.6. Service 11 - Business Rule**

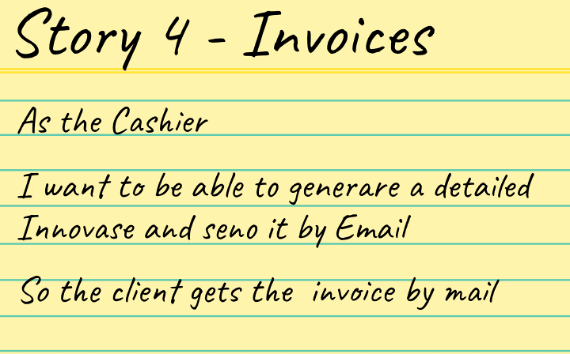
*The Restaurant Owner wants to be able to give discounts to specific clients to engage them. So, we want a Service that allows us to Send a discount on a specific item of the menu, to a specific client, and for a specific percentage and we will use WhatsApp.*

|  |  |
| --- | --- |
| **Method** | GET |
| **Description** | Send discount on a certain menuItem to a certain client’s WhatsApp Number for a certain percentage |
| **URI** | http://{ip}:3005/restaurant/menuItem/{code}/discount/{percentage}/client/{idCard} |
| **Parameters** | **code**: code of menuItem  **percentage:** percentage of Discount offered. i.e. “50”  **idCard:** “Cédula” or “RUC” of the Client |
| **Format** | JSON |
| **Returns** | Http Codes:   * 200 for Success * 500 for Server Errors   And in the Clients Whatsapp: |

**3.3. User Story 3 Services**



**3.4. User Story 4 Services**



**3.4.1. Service 12**

|  |  |
| --- | --- |
| **Method** | GET |
| **Description** | GET all the Invoices on DB |
| **URI** | http://{ip}:3005/restaurant/invoices |
| **Parameters** | None |
| **Format** | JSON |
| **Returns** | Http Codes:   * 200 for Success * 500 for Server Error |

**3.4.2. Service 13**

|  |  |
| --- | --- |
| **Method** | GET |
| **Description** | GET one Invoice by its InvoiceNumber |
| **URI** | http://{ip}:3005/restaurant/invoices/{invoiceNumber} |
| **Parameters** | **invoiceNumber**: code of Invoice |
| **Format** | JSON |
| **Returns** | Http Codes:   * 200 for Success * 500 for Server Error |

**3.4.3. Service 14**

|  |  |
| --- | --- |
| **Method** | POST |
| **Description** | POST a new Invoice |
| **URI** | http://{ip}:3005/restaurant/invoices |
| **Parameters** | You need to send all the attributes of an Invoice |
| **Format** | JSON |
| **Returns** | We can see that the Decimal128 price parses correctly.  Http Codes:   * 200 for Success * 500 for Server Error |

**3.4.4. Service 15**

|  |  |
| --- | --- |
| **Method** | PUT |
| **Description** | PUT Update an Invoice using its InvoiceNumber and the new attributes |
| **URI** | http://{ip}:3005/restaurant/invoice/{invoiceNumber} |
| **Parameters** | **invoiceNumber**: code of Invoice    You don’t need to send all the attributes of an Invoice, just the ones you want |
| **Format** | JSON |
| **Returns** | Http Codes:   * 200 for Success * 404 if there is not a Menu Item with that Code * 500 for Server Error at finding or Updating |

**3.4.5. Service 16**

|  |  |
| --- | --- |
| **Method** | DELETE |
| **Description** | DELETE an Invoice using its InvoiceNumber |
| **URI** | http://{ip}:3005/restaurant/menuItem/{invoiceNumber} |
| **Parameters** | **invoiceNumber**: code of Invoice |
| **Format** | JSON |
| **Returns** | Http Codes:   * 200 for Success * 500 for Server Error at finding or Updating |