**Design Document – My Beauty Scheduler**

Introduction

My Beauty Scheduler app is made for the use of beauticians and/or manicurists, to have better client management. The app functionalities are containing a list of clients, products, and treatments that exist in the clinic, with an option to watch each client's details, including her treatment history and purchased products. Moreover, the app contains a calendar that allows scheduling appointments with clients.

Requirements

Functional requirements

Use Cases

|  |  |
| --- | --- |
| Use Case Name: | Add an appointment to the calendar |
| Preconditions | The user should be logged in to the app |
| Main Flow | the user presses "Add Appointment", and a window will open in which the user is required to fill in the treatment, the client's name, and the start and end of the event. |
| Postcondition | The event will be added to the calendar, and if the client doesn't exist it will be added to the clients' page. |

|  |  |
| --- | --- |
| Use Case Name: | Add a new client |
| Preconditions | The user pressed "clients" in the navbar, and then presses "+" |
| Main Flow | A window of "Add Client" is opened and the user is required to fill in the client's details. |
| Postcondition | The client will appear on the clients' list |

|  |  |
| --- | --- |
| Use Case Name: | Delete a client |
| Preconditions | The client exists in the clients' list |
| Main Flow | The user presses the delete icon near the client's name, and a window will pop and ask the user if he is sure that he wants to delete the client |
| Postcondition | If the user presses "Yes" the client will be deleted from the list, and if he presses "No" nothing will happen. |

|  |  |
| --- | --- |
| Use Case Name: | Add a new product |
| Preconditions | The user pressed "products" in the navbar, and then presses "+" |
| Main Flow | A window of "Add Product" is opened and the user is required to fill in the product's details. |
| Postcondition | The client will appear on the clients' list |

|  |  |
| --- | --- |
| Use Case Name: | Delete a product |
| Preconditions | The product exists in the products list |
| Main Flow | The user presses the delete icon near the product's name, and a window will pop and ask the user if he is sure that he wants to delete the product. |
| Postcondition | If the user presses "Yes" the product will be deleted from the list, and if he presses "No" nothing will happen. |

|  |  |
| --- | --- |
| Use Case Name: | Add a new treatment |
| Preconditions | The user pressed "treatments" in the navbar, and then presses "+" |
| Main Flow | A window of "Add Treatment" is opened and the user is required to fill in the treatment's details. |
| Postcondition | The treatment will appear on the treatments list |

|  |  |
| --- | --- |
| Use Case Name: | Delete a treatment |
| Preconditions | The treatment exists in the treatments list |
| Main Flow | The user presses the delete icon near the treatment's name, and a window will pop and ask the user if he is sure that he wants to delete the treatment. |
| Postcondition | If the user presses "Yes" the treatment will be deleted from the list, and if he presses "No" nothing will happen. |

|  |  |
| --- | --- |
| Use Case Name: | Add purchased products for a client |
| Preconditions | The user enters a client card and presses the "+" near purchased products |
| Main Flow | A window will open, and the user may choose a product from a list of products, using a dropdown menu |
| Postcondition | The chosen product is added to the purchased products of the client. |

Interfaces

SW Interfaces

Register user:

POST /api/auth/register

Request will be sent with the POST

{

firstName: value,

lastName: value,

email: value,

password: value

}

Login user:

POST /api/auth/login

Request will be sent with the POST

{

email: value,

password: value

}

Add new client:

POST /api/clients/newClient

Request will be sent with the POST

{

fullName: value,

phoneNumber: value

}

Add new client from calendar event:

POST /api/clients/newClientFromCalendar

Request will be sent with the POST

{

fullName: value,

phoneNumber: value

}

Get the list of all clients:

GET /api/clients/getClients

Delete a client:

DELETE /api/clients/deleteClient

Request will be sent with the DELETE

{

clientId: value

}

Add event to calendar:

POST /api/events/newEvent

Request will be sent with the POST

{

    title: value,

    fullName: value,

    phoneNumber: value,

    start: value,

    end: value,

}

Get all events:

GET /api/events/getEvents

Delete event:

 DELETE /api/events/deleteEvent

Request will be sent with the DELETE

{

eventId: value

}

Add new product:

POST /api/products/newProduct

Request will be sent with the POST

 {

    productName: value,

    manufacturer: value,

    productType: value,

    productGroup: value,

    price: value,

}

Get all products:

GET /api/products/getProducts

Delete product:

DELETE /api/products/deleteProduct

Request will be sent with the DELETE

{

productId: value

}

Add new treatment:

POST /api/treatments/newTreatment

Request will be sent with the POST

{

treatmentName: value,

    price: value,

}

Get all treatments:

GET /api/treatments/getTreatments

Delete treatment:

Request will be sent with the DELETE

{

treatmentId: value

}

Get treatment history by client's name:

GET /api/treatmentHistory/getTreatmentHistoryByName

Request will be sent with a query including

{

phoneNumber: value

}

User interfaces

Diagram

Description automatically generated

Design

Data models

MongoDB Schemas

Client

{

  fullName:  String,

  phoneNumber:  String,

  productsPurchased: [String],

}

Event

{

  start: Date,

  end:  Date,

  title: String,

  clientName: String,

  phoneNumber: String,

}

Product

{

  productName:  String,

  manufacturer: String,

  productType: String,

  productGroup: String,

  price: Number,

}

Treatment

{

  treatmentName: String,

  price:  Number,

}

Treatment History

{

  fullName: String,

  phoneNumber: String,

  treatmentName:  String,

  date:  Date,

}

User

{

  firstName: String,

  lastName: String,

  password: String,

  email: String,

}

UML diagram

Diagram

Description automatically generated