



Adventist University of Central Africa

P.O. Box 2461 Kigali, Rwanda | www.auca.ac.rw | info@auca.ac.rw

WEB TECHNOLOGY FINAL PROJECT DOCUMENTATION

BY NTORE NTWARI RUMENERA

24438

14th December 2023

CUSTOMER RELATIONSHIP MANAGEMENT SYSTEM

The NTJ Devices Customer Relationship Management (CRM) spring boot project has a primary focus on revolutionizing the online ordering experience for customers while ensuring seamless and timely delivery of cutting-edge devices. The key objectives include:

1. **User Authentication and Authorization:** The platform ensures secure user authentication and authorization processes. Customers can easily create accounts, securely log in, and utilize email services for product orders and notifications. The authentication system incorporates robust mechanisms to enhance user security, while role-based access control grants unique privileges to administrators through a separate login, with admin credentials hardcoded for heightened security.
2. **E-commerce Functionality:** The application supports essential e-commerce functionalities, allowing users to manage their shopping carts by double-clicking on desired products. Admins have the capability to perform CRUD operations on both customer accounts and the products customers interact. Additionally, efficient product search and filtering mechanisms contribute to an enhanced user experience.
3. **Admin Functionality:** A dedicated admin dashboard, accessible through a separate login, provides administrators with the tools to manage customer accounts and products efficiently, including CRUD operations for both.
4. **File Handling:** The platform facilitates file upload and download functionalities, enabling administrators to upload files, which customers can view and download for various purposes. This feature extends to product-related files, providing additional information to users.
5. **User Interaction:** The web application boasts responsive interfaces, ensuring an interactive and seamless user experience. Users can securely log out of their accounts when needed.
6. **Technology Stack:** The backend development is Java programming language, providing robust server-side logic. Frontend development leverages THYMELEAF, HTML, CSS, and JavaScript to create dynamic and engaging interfaces. The Spring Boot framework streamlines development processes, contributing to an efficient and effective application.

Requirements used in the production of the system include:

I developed the system using the Spring Tool Suite (STS) with POSTGRESQL as the database. Since the Spring Tool Suite is an IDE dedicated solely to spring boot projects, it is simpler to create a spring boot project using it.

Open STS and select "File" -> "New" -> "Spring Starter Project".

The service URL is the <https://start.spring.io> and you select the type, java version, packaging and language that you wish to use and then select the dependencies that your project will require then create the project. In my case I chose the type to be maven so that I can handle all the dependencies I will need easily, the java version used was 17, the packaging was jar and the language used was of course java. The dependencies used include:

- **THYMELEAF:** it allow you to create dynamic web pages by combining HTML templates with server-side data.
- **Spring Security:** If your application requires authentication and authorization, Spring Security provides robust security features.
- **Hibernate or Spring Data JPA:** If your application interacts with a relational database, Hibernate or Spring Data JPA can simplify database operations.
- **Spring web:** Spring Web provides a Model-View-Controller (MVC) architecture that allows for clear separation of concerns between the application's data, presentation, and user interaction.

- **Spring Dev-Tools**: this framework support remote development scenarios. It allows you to run your application on a remote server, provides automatic application restart capabilities.
- **Compatibility**: chosen database driver must be compatible with the database management system (DBMS) you are using. Different databases (e.g., MySQL, PostgreSQL, Oracle, and SQL Server) have their own specific drivers. As stated earlier I used POSTGRES.

For the other dependencies used despite those would be Jakarta mail and its activation jar with 2.0.1 version to help make the email process easier and efficient. Other dependencies where all automatically included by maven.

When developing a Spring MVC project, you typically need to include several packages from the spring framework such as:

- Controller
- Model
- Repository
- Service layer

The controller plays a crucial role in handling incoming HTTP requests, processing the requests, and returning an appropriate response to the client.

The main controller class consists of some annotations to indicate that it is responsible for handling HTTP requests and serving as a component of the MVC architecture.

@Controller: This annotation marks a class as a controller component in Spring MVC.

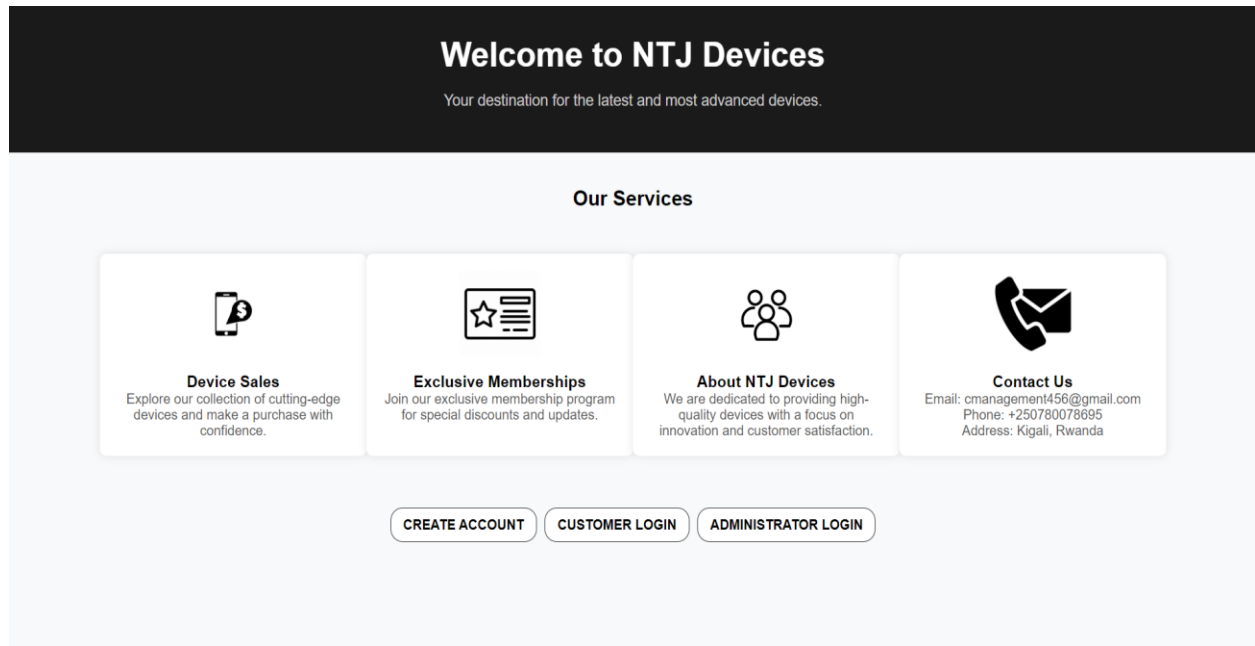
@RequestMapping: This annotation helps define the base URL or URL patterns that the controller handles.

@GetMapping, **@PostMapping**, **@PutMapping**, **@DeleteMapping**: These annotations are used to handle specific HTTP methods (GET, POST, PUT, DELETE) for a given URL pattern.

- **Model**: represents the data that passes between the controller and the view. It plays a crucial role in facilitating the exchange of information, enabling the controller to prepare the data and the view to render it appropriately.
- **Repository**: is responsible for defining and implementing methods to access and manipulate data stored in the database.
- **The service layer** plays a crucial role in implementing the business logic and coordinating the interactions between the controller and the repository (data access layer).
- **@Path-Variable**: This annotation helps to bind a URL variable to a method parameter. It extracts the value from the URL and maps it to the corresponding method argument.
- Templates are responsible for rendering the dynamic content seen by the user.
- Templates use template languages or frameworks (e.g., JSP, THYMELEAF) to access and display the data received from the controller.

I added all that in my project in order for it to function as intended and added static folder that contains the CSS that helped in better visualization.

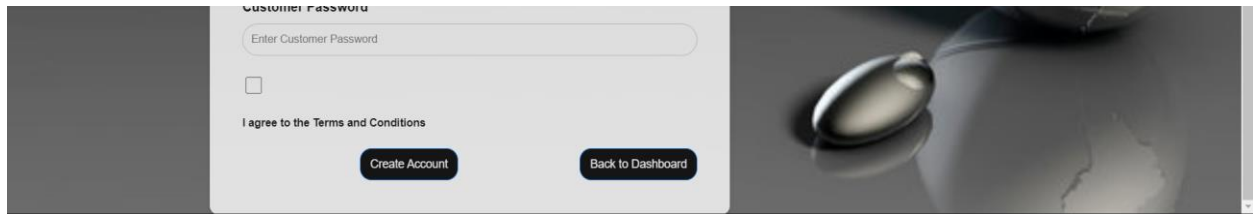
The homepage view



The customer creates an account and is then able to login into the portal. Login works for only customers in the database. The admin also logs in in order to access the admin portal and carry out various activities. The admin credentials are hardcoded because if they had the option of first going into the database it would give the customer the freedom of becoming an admin as well. The username is NTORE and the password is qqqq1111.

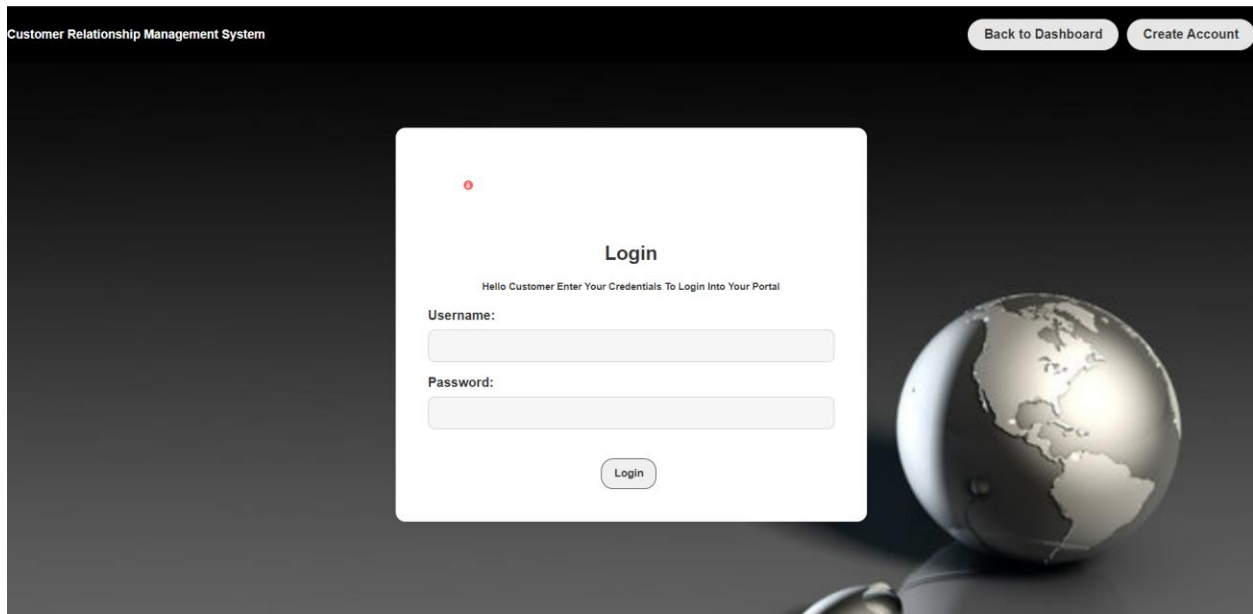
CREATE ACCOUNT VIEW

The screenshot shows a "Create New Customer Account" form. The form is titled "Create New Customer Account" and contains several input fields with placeholder text: "Customer First Name" (placeholder: "Enter Customer First Name"), "Customer Last Name" (placeholder: "Enter Customer Last Name"), "Customer User Name" (placeholder: "Enter Customer User Name"), "Customer Email" (placeholder: "Enter Customer Email"), "Customer Phone Number" (placeholder: "Enter Customer Number"), "Customer Address" (placeholder: "Enter Customer Address"), and "Customer Password" (placeholder: "Enter Customer Password"). The form is set against a dark background with a globe image on the right.

A screenshot of a web form titled "Customer Password". It features a text input field with the placeholder "Enter Customer Password", a checkbox, and the text "I agree to the Terms and Conditions". Below these are two buttons: "Create Account" and "Back to Dashboard". The form is set against a dark background with a globe and a pen.

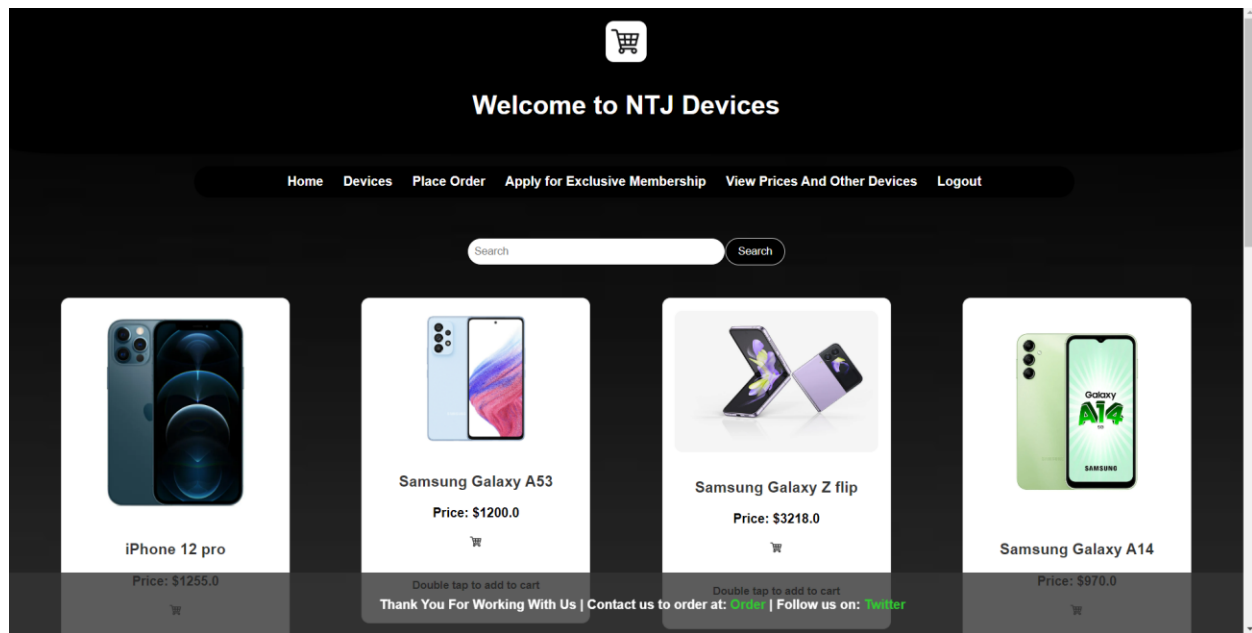
Here is what the customer sees as he creates the account. Once the customer creates the account and it is a success, there appears a success page giving them options to either go to the login page or return to the dashboard page. The create account form is long hence making the images come as two.

THE LOGIN VIEW

A screenshot of the "Login" view in a "Customer Relationship Management System". The page has a dark header with the system name on the left and "Back to Dashboard" and "Create Account" buttons on the right. The main content area features a white login form with a red error indicator, the title "Login", a greeting "Hello Customer Enter Your Credentials To Login Into Your Portal", and fields for "Username:" and "Password:". A "Login" button is at the bottom of the form. The background is dark with a globe and a pen.

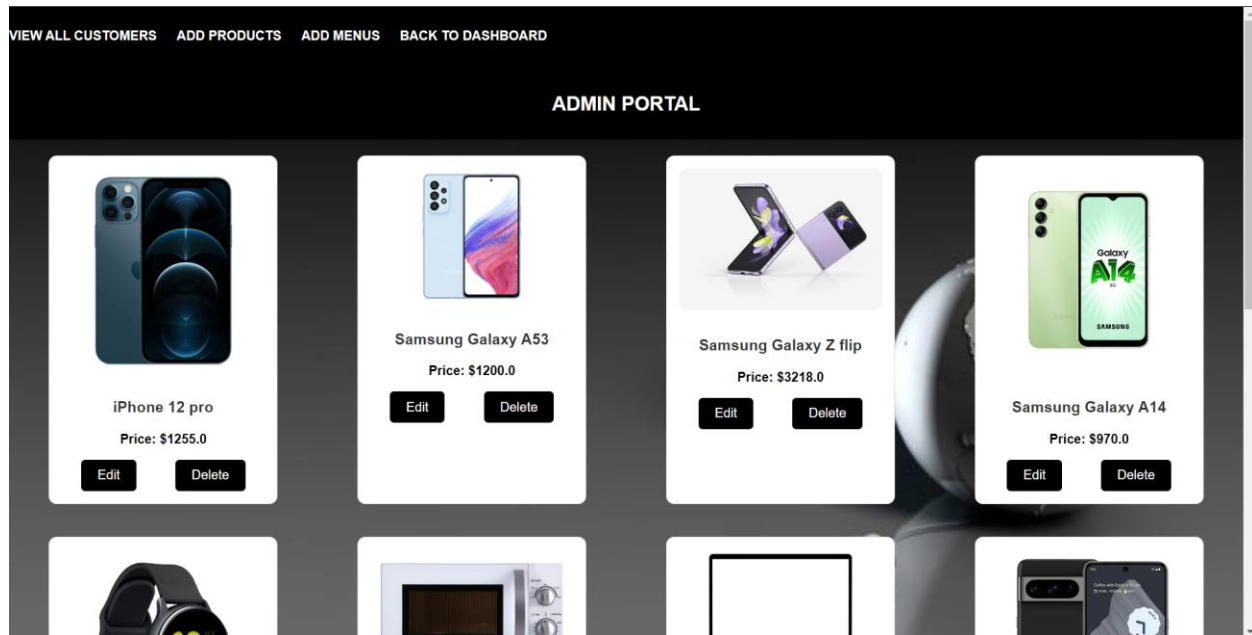
Here is where the login enters his/her credentials in order to enter and access the customer portal. Only the customers in the database can login and if not are directed to an error page telling them incorrect credentials. In case the customer has no account, there is a button in the navigation bar that would take them to the create account page.

THE CUSTOMER PORTAL OR HOMEPAGE VIEW



After successfully logging in, the customer sees this and can start exploring the devices, selecting the ones they like, and placing orders for the ones they want. An incredible feature allows them to apply for exclusive memberships, which allows them to receive updates on discounts and other offers. Included is also a search functionality that allows them to know whether the desired product is available without having to scroll through the entire selection. Finally, the customer has an option of adding products to his/her cart by double clicking the product of desire. After completing all of their desired tasks, the user can log out.

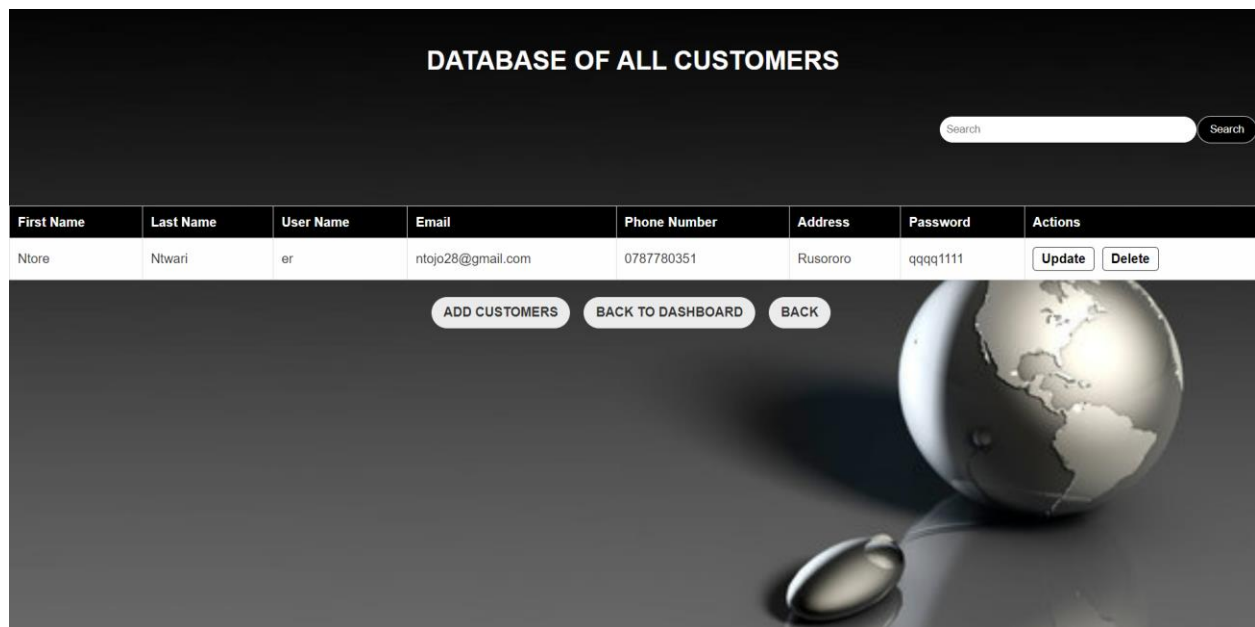
THE ADMIN PORTAL VIEW



As you can see, the admin has credibility to perform CRUD on both the products that the customer views as well as the customers in the database. He also uploads the files and documents that the customer might need to view and download and then the customer will hence be able to do it.

The admin has a lot of power on how the project performs and makes this CRM (Customer relationship management system) work efficiently.

THE DATABASE VIEW AND THE ERD.



Here the admin is able to search for customers in the database as well as perform all the necessary CRUD operations well and efficiently.

Although the system is simple, it is a system done with the knowledge acquired from our studies and does according to how it has work.

THE DATABASE SCHEMA OF MY PROJECT.

Finally, the database schema of my project showing the entities and relationships in my project. Each entity relies on the other to work in accordance. Below is the diagram

