3.2 Objectives

This project aims to enhance the functionality and user experience of WordPress websites by implementing custom features based on client's needs. By flexibility of WordPress, we focus on optimizing various aspects of web development through advanced customization using PHP, JavaScript, HTML and CSS. We aim to:

i. Enhance the Shop and Product Archive pages' navigation by using an AJAX – based pagination system, speeding up page loads, and dynamically loading new products without requiring page refreshes. This will improve user experience and greatly improve sales.

ii. Integrate additional custom fields into WooCommerce registration forms to get more personalized and detailed user information, ensuring the relevant data is collected and stored effectively in the database to improve the user management process.

iii. Integrate Contact Form 7 with WooCommerce. A custom registration form will be developed to parse data seamlessly into the WooCommerce database, streamlining the user registration process.

iv. Customize the Checkout page based on language. This will improve user – friendliness as many multilingual users will be able to access the website using various languages. GTranslate will be utilized to dynamically hide, display, or add custom fields depending on the user's selected language.

v. Prepare for future client needs by designing scalable features. This will allow for easy modifications and enhancements as requirements evolve.

3.3 Hardware & Software Requirements

3.3.1 Hardware Requirements

Table 3.1 My Personal Laptop

Details	Description
Device Name	Asus Zephyrus G14
Operating System (OS)	Windows 11
Processor	AMD Ryzen 5 4600HS with Radeon Graphics (3.00 GHz)
Graphics Processing Unit (GPU)	GeForce GTX 1650
Memory	8GB RAM
Storage	480GB SSD Memory
Network	Wi-Fi 6

Table 3.2 Other Hardware

Details	Description
Mobile Devices	Real Android and iPhone Operating System (iOS) devices are used to check responsiveness, and test the features to ensure performance, and compatibility across a range of screen sizes and operating system versions.
External Monitor	Acer monitor is used to provide additional screen size for multitasking, such as coding, debugging, layout checking, and referencing documentation simultaneously.

3.3.2 Software Requirements

Table 3.3 Software Requirements

Software	Description
Integrated Development Environment (IDE)	IDE such as VS Code Studio is used for writing and debugging code, with many extensions for WordPress development. For example, PHP IntelliSense, Hooks IntelliSense for WordPress, WordPress Snippets and WPCS Whitelist Flags.
Platform	WordPress and Elementor Pro as page builder.
Database	MySQL (phpMyAdmin) is used for database management, as WordPress typically uses MySQL for data storage and scripting.
Server Hosting	Software like cPanel is used for live server hosting, while MAMP and XAMPP are used for local host (staging site).
Programming Language	PHP is used for server-side scripting and hooks, JavaScript for client-side scripting and interactivity, HTML for structuring web content, and CSS for styling and designing the web pages.
Web Browser	Google Chrome, Microsoft Edge, and Safari are used to test the responsiveness and cross-browser compatibility of WordPress sites.
Plugins	Key plugins used for the WordPress are WooCommerce, Elementor Pro, GTranslate, Contact Form 7, Contact Form 7 Add Password field, ReCAPTCHA V2 and WP Mail SMTP.

3.4 Results of Project

3.4.1 AJAX Load More with Filters for Shop and Category Pages

3.4.1.1 Introduction

AJAX Load More Pagination and filtering are advanced techniques used to improve user experience on Shop and Category pages in WordPress websites. AJAX, which stands for Asynchronous JavaScript and XML, enables dynamic updates to web pages without requiring a full-page refresh. By implementing Ajax Load More functionality, users can seamlessly load additional products as they scroll or click a button, eliminating the need for traditional pagination that involves navigating through numbered pages. Filtering, when combined with AJAX, allows users to dynamically refine product listings based on specific attributes, such as price, category, or rating, in real time.

This feature has been developed using PHP, JavaScript, HTML, and CSS, leveraging WordPress and WooCommerce hooks to ensure seamless integration. WooCommerce powers the eCommerce functionality, providing robust tools to manage products, categories, and pricing, while Elementor Pro has been utilized to design and structure the shop and category pages. Together, these tools create a visually appealing and highly functional experience for both website administrators and end users.

The adoption of AJAX Load More Pagination and filtering ensures that visitors can navigate and explore products effortlessly, contributing to a faster and more responsive browsing experience. This enhancement is particularly valuable for eCommerce websites where user convenience and speed play a critical role in driving engagement and sales.

3.4.1.2 Problem Statement

Traditional pagination methods, such as numbered links or "Next" and "Previous" buttons, are often cumbersome and can negatively impact user experience. Each click on a pagination link triggers a full page reload, which not only interrupts the browsing flow but also increases loading times. This can be particularly frustrating for users on slower internet connections or when dealing with large product catalogues, leading to potential abandonment of the shopping process.

AJAX pagination offers a superior alternative by enabling users to load additional products dynamically without reloading the entire page. This approach ensures smoother and faster browsing experience, keeping users engaged and reducing bounce rates. Additionally, combining AJAX with filtering functionality allows real-time updates to product listings based on user preferences, eliminating the need for repetitive page loads and providing a more intuitive and enjoyable shopping experience.

3.4.1.3 Flowchart

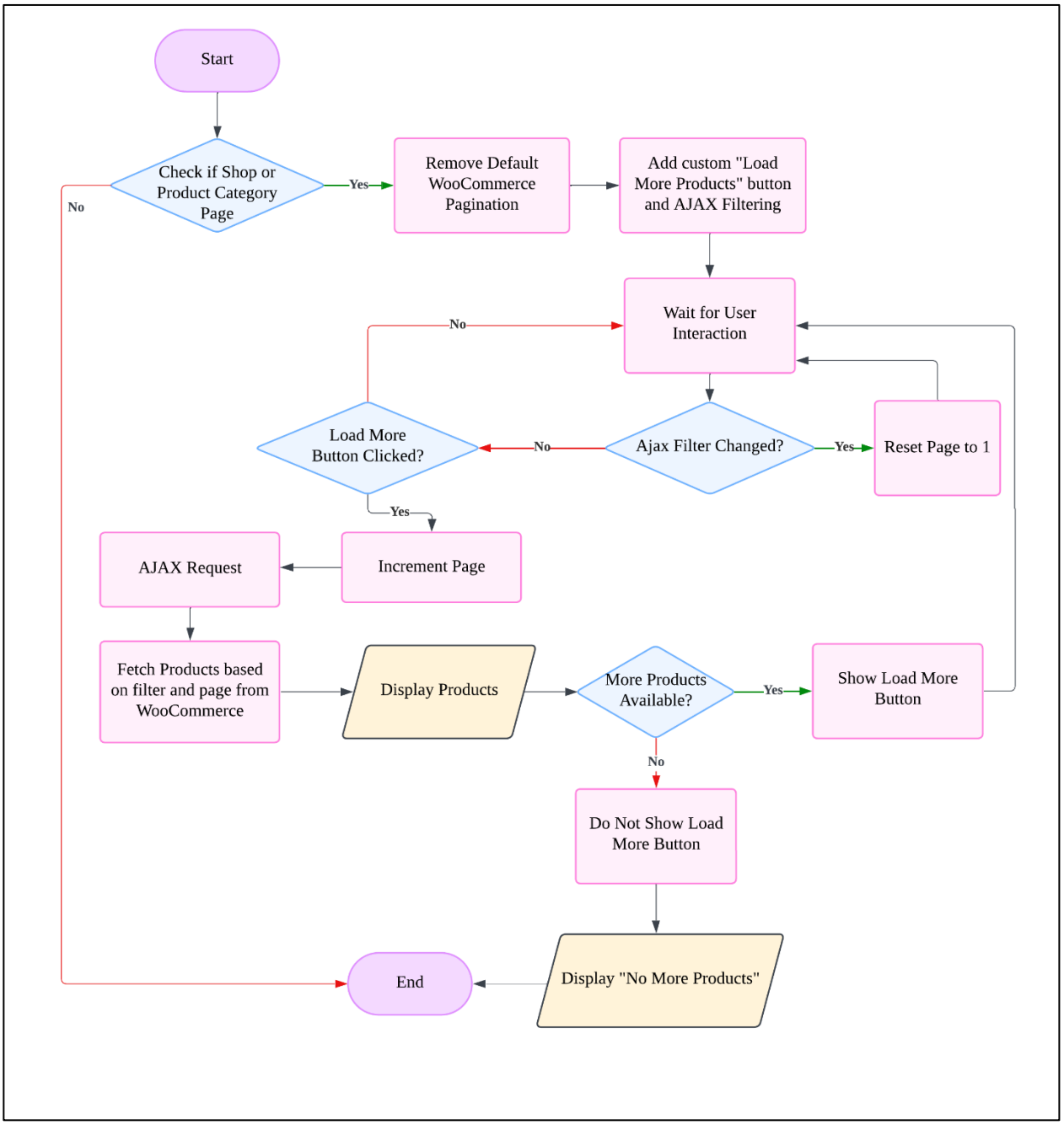

Figure 3.1: Flowchart for AJAX Load More with Filters for Shop and Product Category Archive Pages

3.4.1.4 Sample Interface

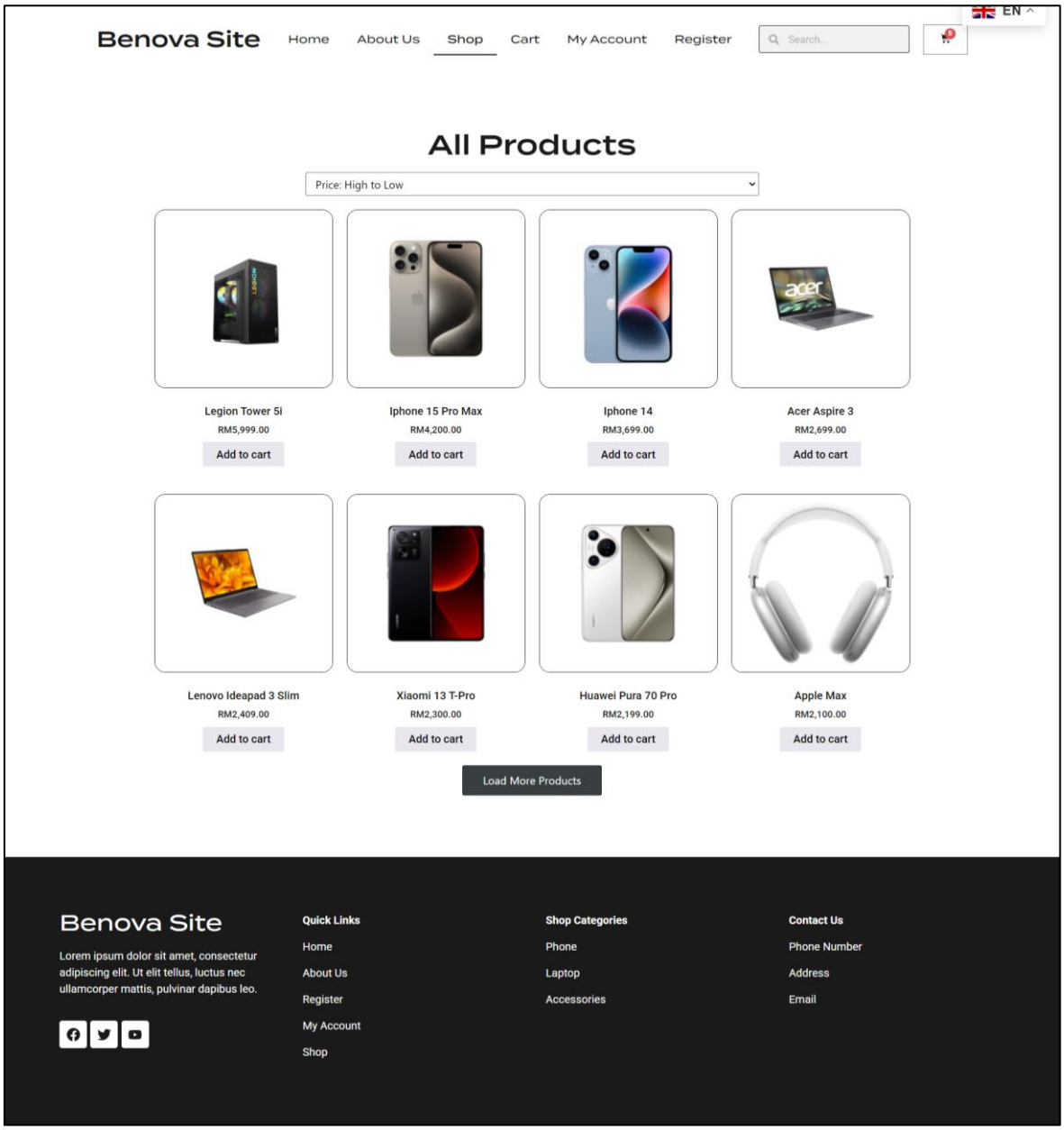

Figure 3.2: Sample Interface for AJAX Load More with Filters for Shop and Product Category Archive Pages

3.4.2 Adding Custom Field to WooCommerce Registration Form

3.4.2.1 Introduction

WooCommerce is the top e-commerce plugin for WordPress, running about 30 percent of the world's online stores. It is a flexible open-source platform used to convert WordPress websites into completely functional online stores. These stores are endowed with considerably enhanced features, including inventory management, secure payment processing and advanced shipping calculations.

Improving the WooCommerce registration system sometimes requires extra custom fields to get more user details than the default choices. Some clients need more data based on their needs, such as gender, birthday, postal code, full name, or other more personalized inputs. This can be done by collecting the data using custom fields in the registration form. The registration process is guaranteed to be adjusted with several business requirements by this flexibility, while at the same time more personalized experience is offered to an important number of users.

Although it seems like a simple thing to do, integrating and modifying WooCommerce can be quite complex, as we need to always refer to their documentation throughout the process. In this case, I implemented PHP for parsing the data to the database, using multiple WooCommerce and WordPress hooks, HTML and CSS for the layout and styling. The key outcomes from this include:

i. A better and enhanced registration process using custom fields in WooCommerce. For example, I added custom fields such as first name, last name, birthday, postal code, and gender to the WooCommerce registration form to provide more flexibility to collect more information from the user (customers).
ii. Database integration using PHP WordPress and WooCommerce hooks. I have successfully parsed the data and stored it in suitable tables in the database to maintain compatibility and effectiveness.
iii. Used secure data handling such as robust validation, error handling, and others to handle the database scripting to ensure security and safeness.
iv. The customers will be able to update their details through ‘My Account’ Page.

3.4.2.2 Problem Statement

The standard WooCommerce registration form has limited choices for getting user details, so businesses cannot easily collect the data they need to manage users well and connect with them personally. The lack of an ability to add at least two custom fields presents website administrators with several challenges in changing the registration process to their precise needs. This limitation frequently results in at least three negative consequences, such as bad user experience, the loss of several opportunities for data-driven understandings and a noticeable decrease in the efficiency of customer relationship management processes.

Adding custom fields to the registration form helps businesses collect and store specific information during onboarding, solving these problems. This feature smoothly integrates into the database, easing intuitive workflows. This integration allows at least several businesses to improve user experience as well as backend efficiency.

3.4.2.3 Flowchart

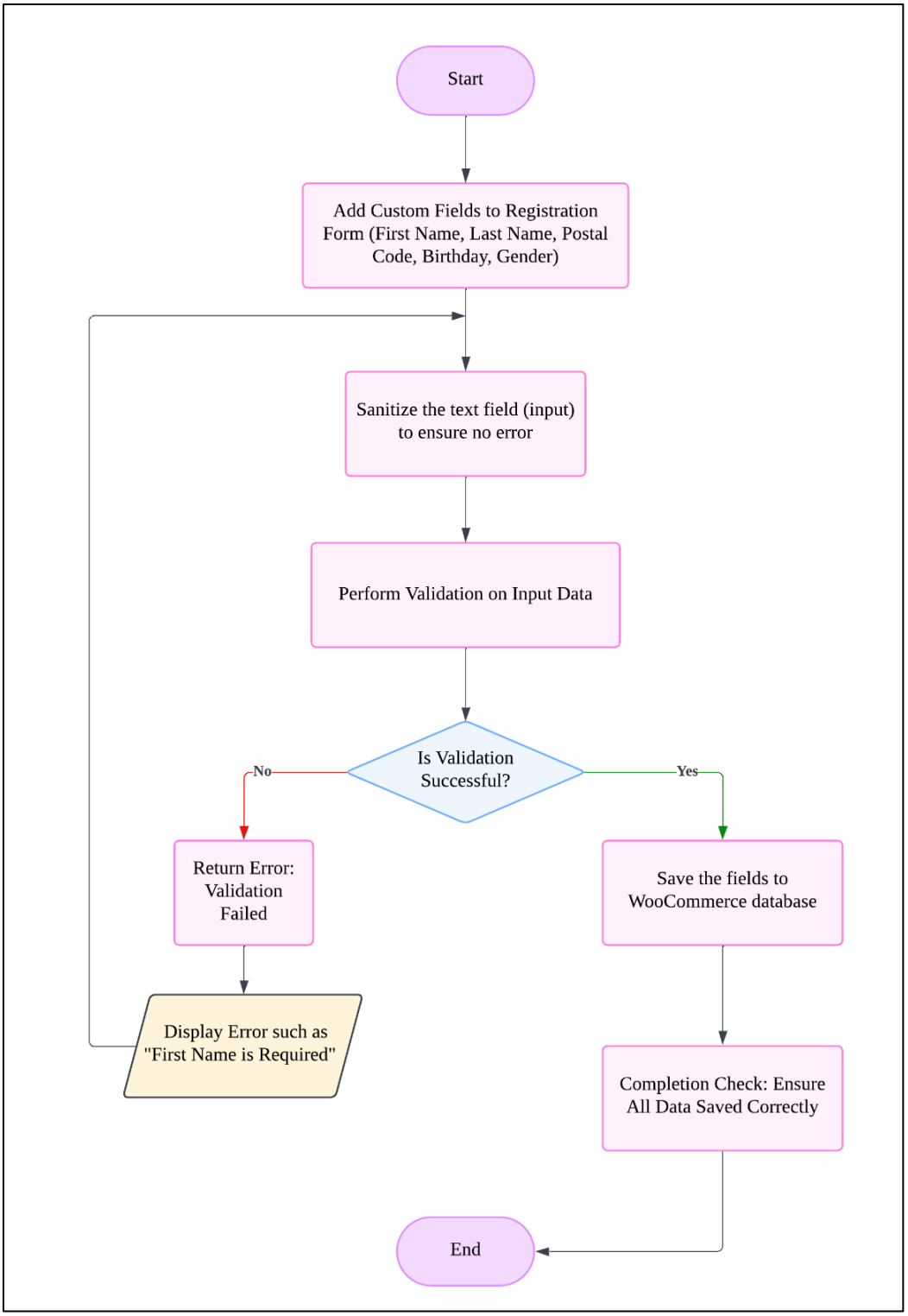

Figure 3.3: Flowchart for Registration Process at Registration Page with Additional Custom Fields with WooCommerce

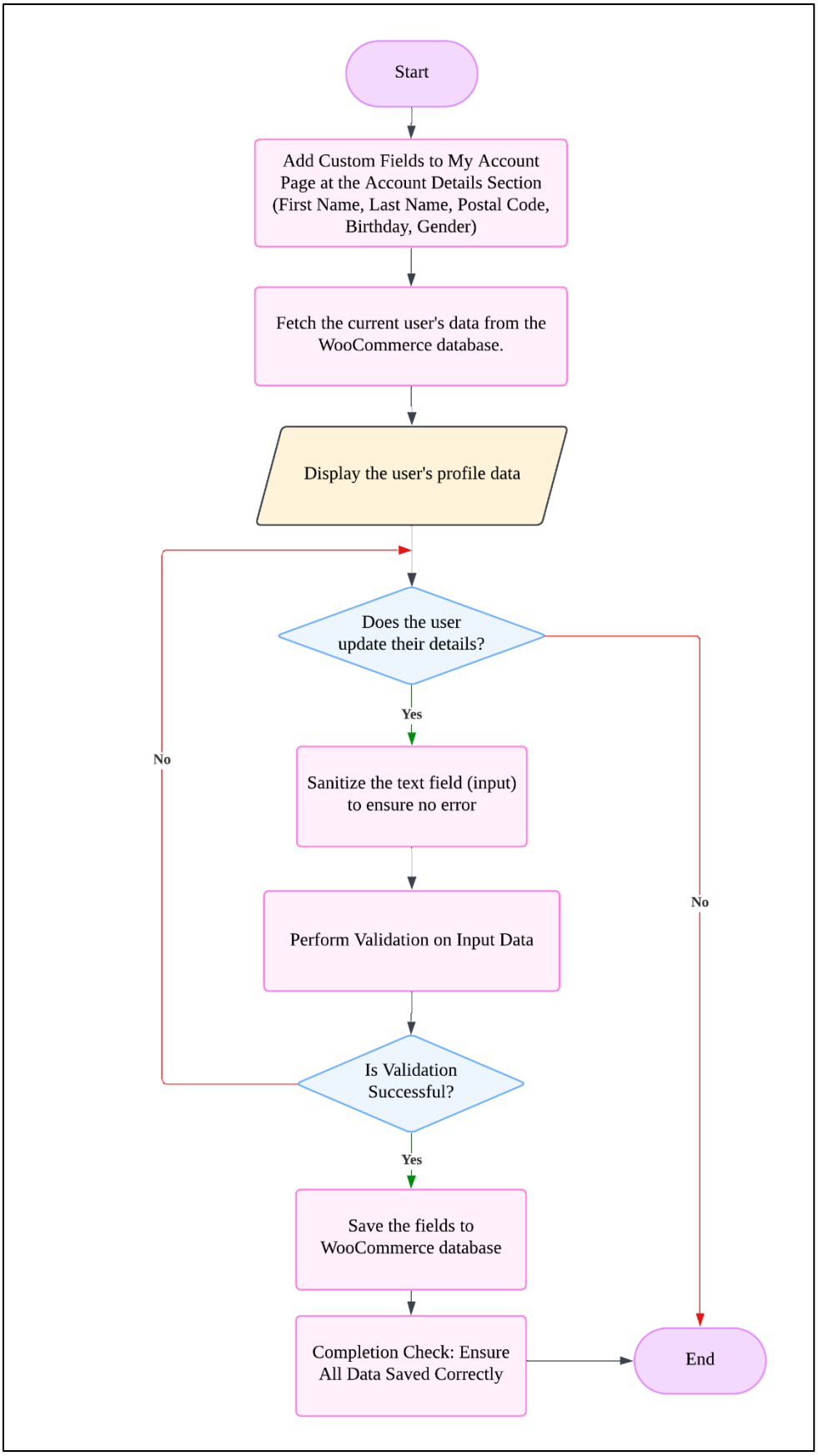

Figure 3.4: Flowchart for ‘My Account’ Page where the user can update their personal information including the new additional custom fields.

3.4.2.4 Sample Interface

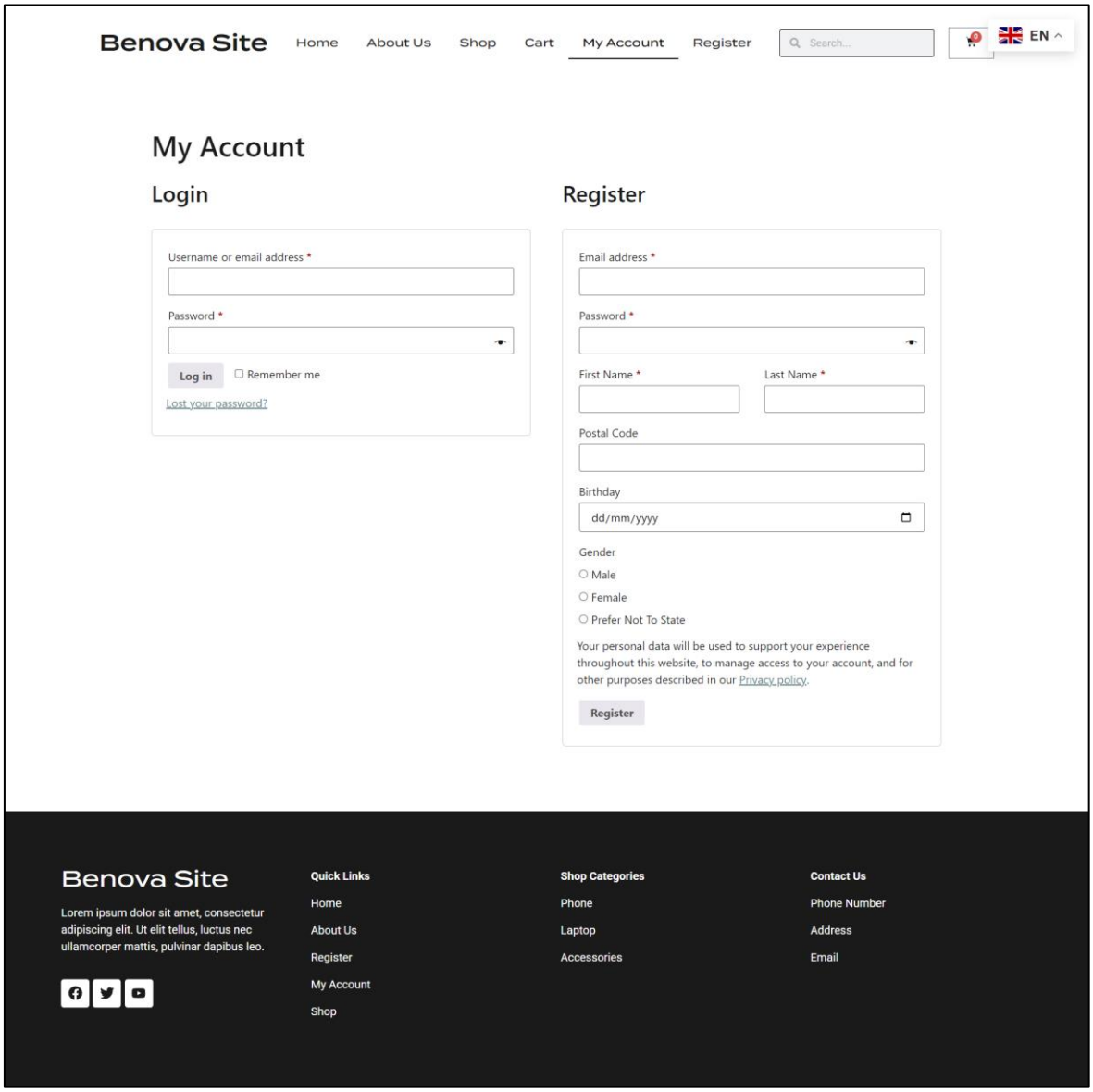

Figure 3.5: Sample Interface for the Registration Page with Custom Additional Fields

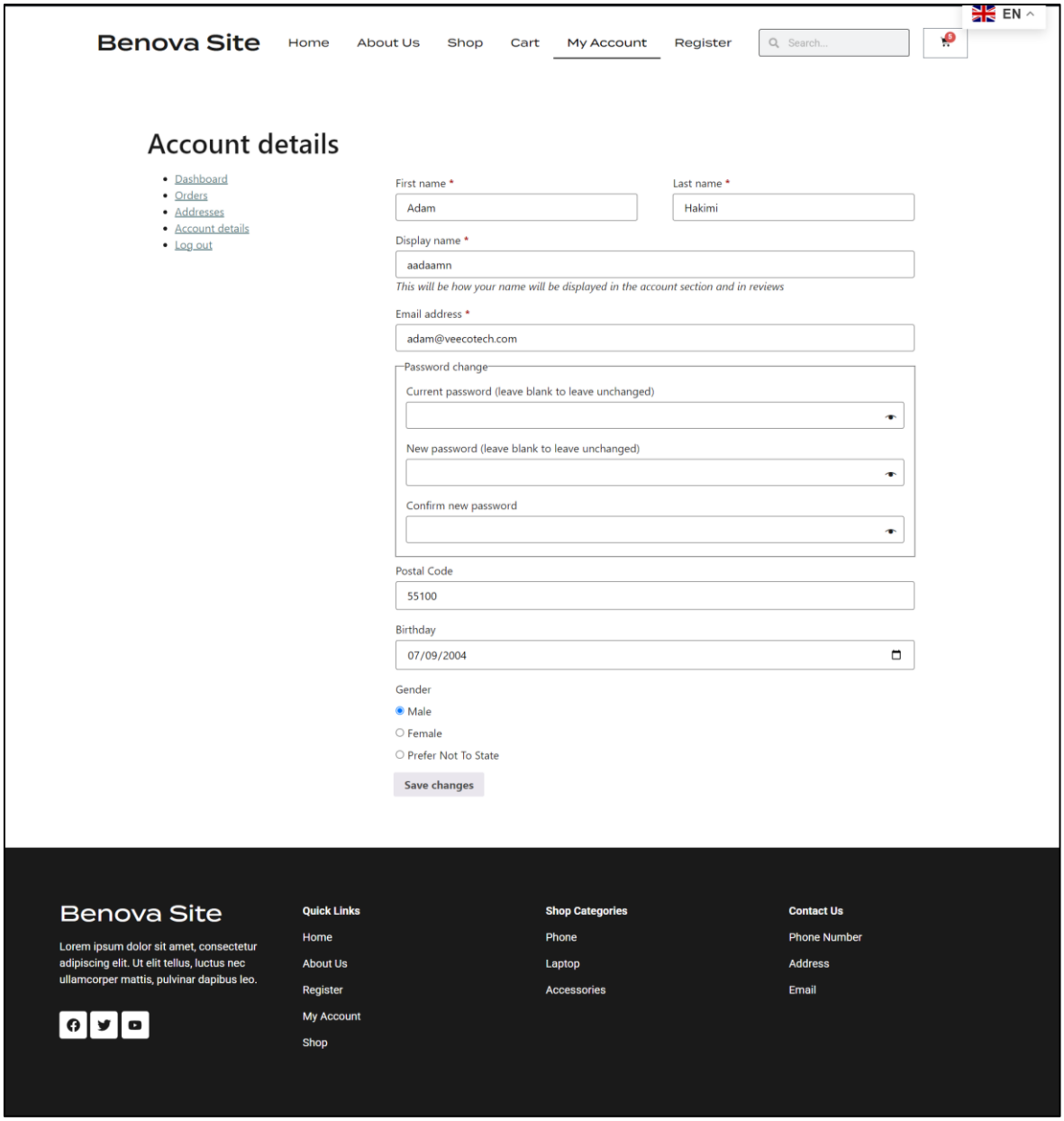

Figure 3.6: Sample Interface for the ‘My Account’ Page where the user will be able to update their personal information

3.4.3 Integrate Contact Form 7 with WooCommerce Registration

3.4.3.1 Introduction

WooCommerce and Contact Form 7 are very popular WordPress plugins for online stores. WooCommerce, which powers more than 5,000,000 online stores globally, holds an important position in the e-commerce platform market, commanding approximately 28 percent of the market share. Contact Form 7, boasting over 5 million active installations, powerfully stands as the leading form builder plugin.

WooCommerce has built-in registration, but using Contact Form 7 for registration is another option if the standard WooCommerce process is too complicated or clients want a simpler system. This alternative implementation retains WooCommerce's strong user management capabilities, offering considerably improved flexibility in form design and greatly enhanced user experience.

PHP hooks and filters create a smooth connection between Contact Form 7 and WooCommerce. The solution uses several WordPress actions, such as 'wpcf7_mail_sent', along with WooCommerce's customer management system to create a new registration process. This process guarantees data accuracy whilst improving the user experience. The key points of this feature include:

i. Added secure data handling techniques, including strong validation, error handling, and others, to database scripting to guarantee safety and security.
ii. Seamless integration between Contact Form 7 and WooCommerce's user management system, allowing custom form fields while maintaining WooCommerce's customer data structure
iii. Improved user experience with automatic login, redirection, and proper page structure.

3.4.3.2 Problem Statement

WooCommerce's standard signup form offers only basic user details, which greatly restricts how much customer information businesses can gather when customers sign up. Limited customization capabilities pose several important challenges for website administrators when they change the registration process to fulfil precise business requirements. Three important problems result from the lack of capacity to augment the registration form with custom fields: incomplete user profiles, inefficient collection of customer data and a severely limited capacity for personalization.

3.4.3.3 Flowchart

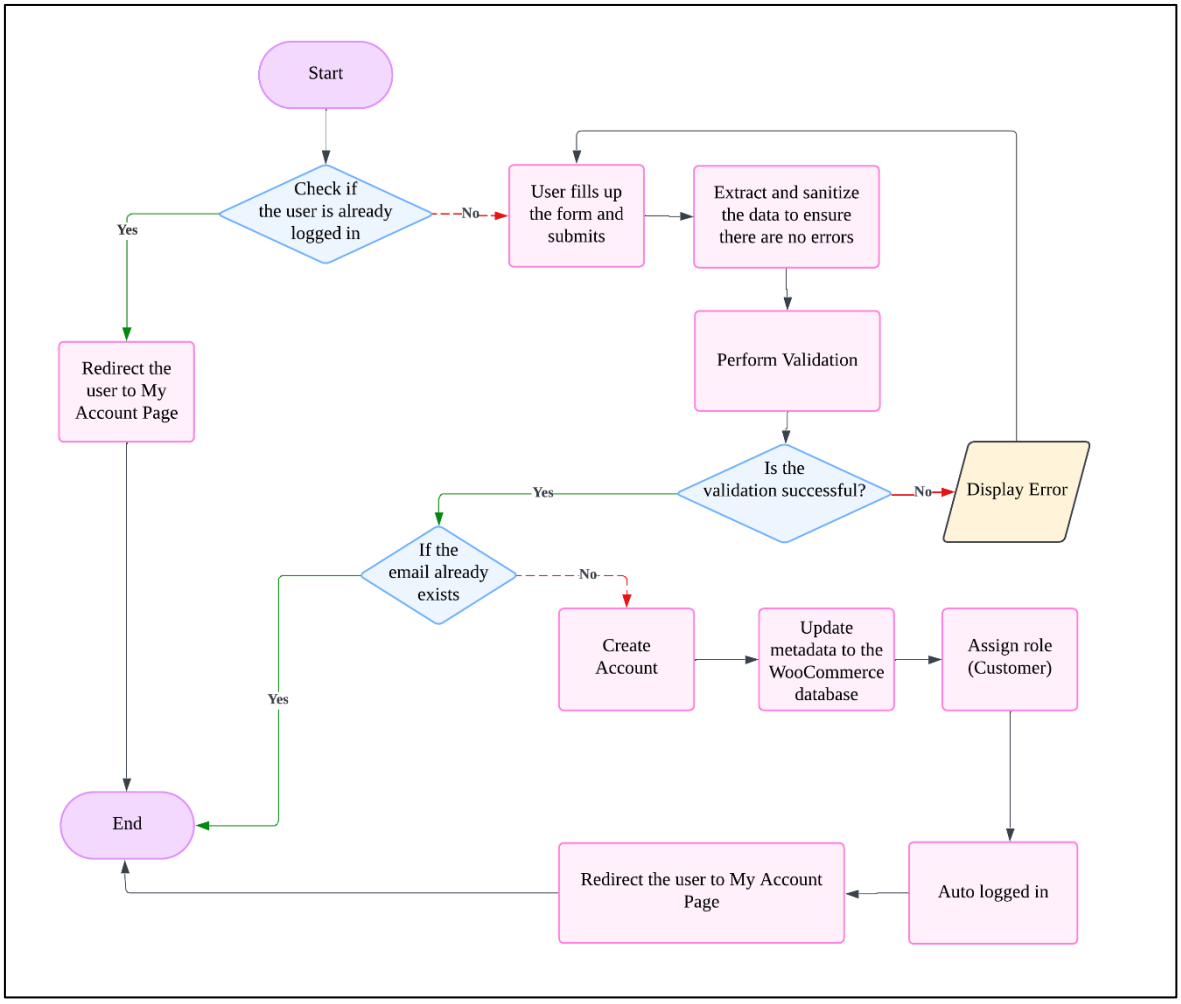

Figure 3.7: Flowchart for Registration Process using Contact Form 7

3.4.3.4 Sample Interface

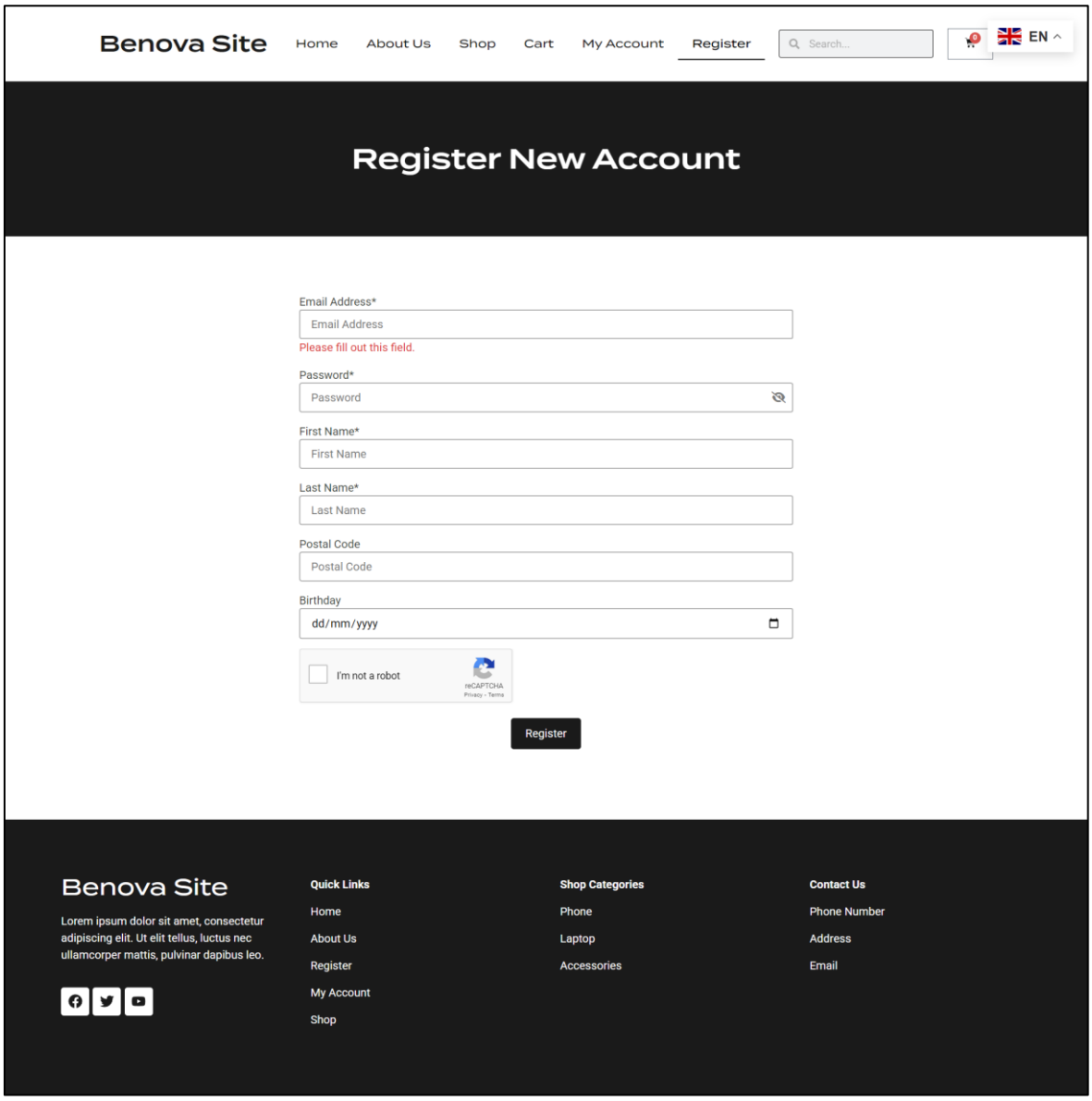

Figure 3.8: Sample Interface of the Registration Form using Contact Form 7

3.4.4 Dynamic Checkout Fields by Language with GTranslate Integration

3.4.4.1 Introduction

WooCommerce has a built – in checkout system to allow the user to purchase products that are available on the website. However, in certain cases, we need to change the checkout fields based on certain needs and requirements from the client. In this case, language barrier difference. By using GTranslate plugin, we will be able to enable the websites to offer multilingual content manually and automatically. GTranslate uses Google Translate API for real – time content translation which supports over 100 languages across the globe.

This dynamic checkout fields feature is a feature that customizes the checkout fields based on the language selection by the customer. It can hide, display, or even create new additional fields based on the language selection. Hence, for this project, I decided to use Tagalog, a Filipino language as an example how we can implement this feature for a client that have customers from the Philippines. If the user is from the Philippines and choose Tagalog as the language, custom fields such as Barangay and Building name will appear as optional. And it will also hide several fields such as Billing Company and Second Billing Address line.

For this feature, I use PHP for backend development and custom WordPress and WooCommerce hooks with HTML and CSS for layout and styling. The key outcomes include:

i. Language – based dynamic checkout field management using GTranslate's translation capabilities.
ii. Great integration with WooCommerce to fully customize the checkout form fields for specific language users such as hiding, displaying, or creating new additional fields.
iii. Improved the user experience since it will provide the user a more personalised sense.
iv. Comprehensive data handling with secure parsing to the WooCommerce database, with validation.

3.4.4.2 Problem Statement

The default checkout system that is offered by WooCommerce is quite limited in its ability to adapt to diverse users all around the world. This limitation creates many problems for our website such as a lack of personalization for users from at least a few countries, the failure to fulfill different needs for each customer, and will result of at least 10% of customer satisfaction because of inefficiency and flexibility during the checkout process. Hence, if our clients need to cater to customers or clients from more than one customer's language, there will be some issues with them which will result in losses in revenue, customer satisfaction, and more.

By integrating dynamic checkout fields with GTranslate plugin for real – time content translation, we can solve this problem to our clients. This solution will empower businesses to modify, hide, or add any custom checkout fields based on the language preferences by the customers. This will result in greater customer satisfaction, enhance user experience, increase personalized feeling to the overall checkout process, and others.

3.4.4.3 Flowchart

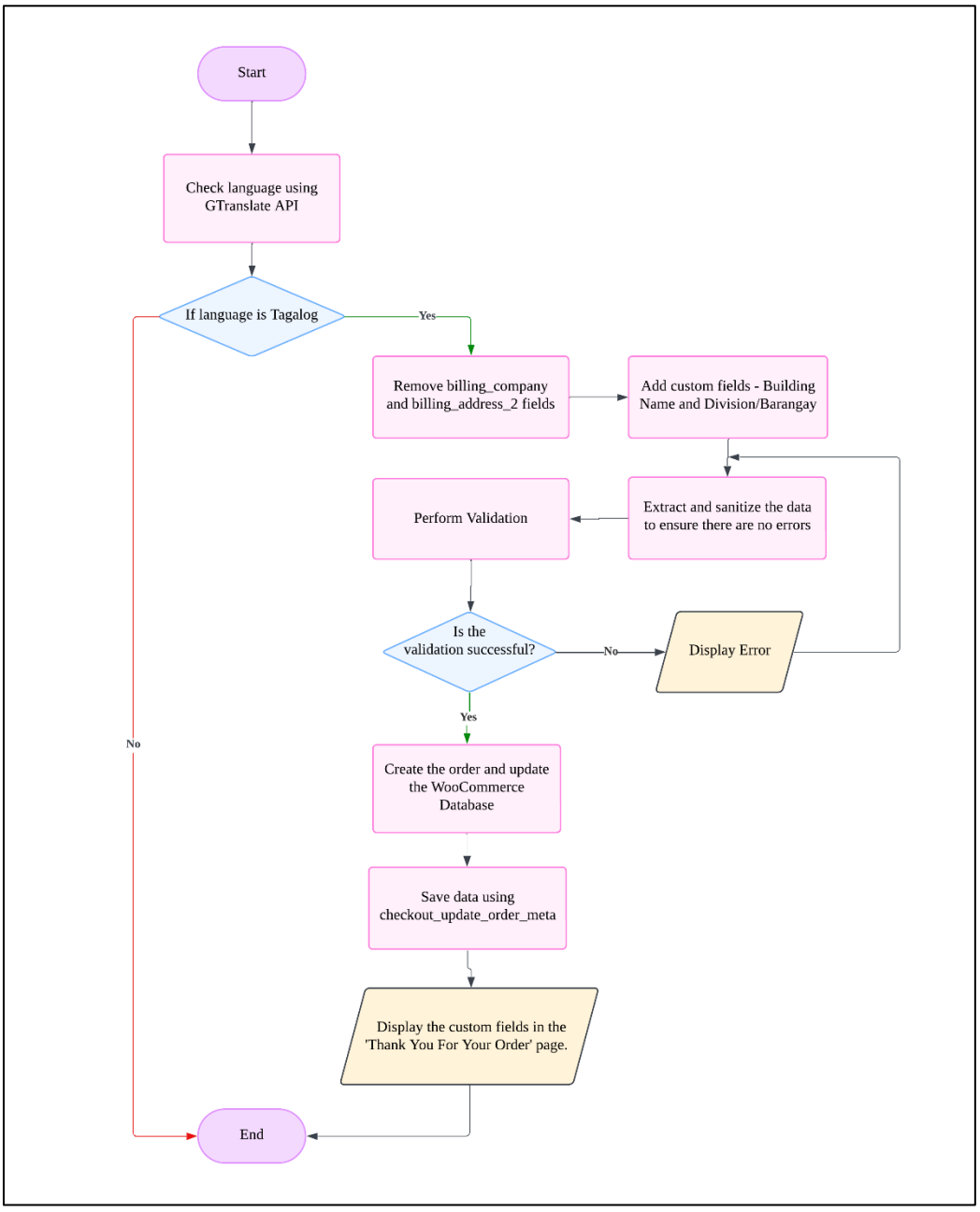

Figure 3.9: Flowchart of the Checkout Process if the user chooses Tagalog as the language

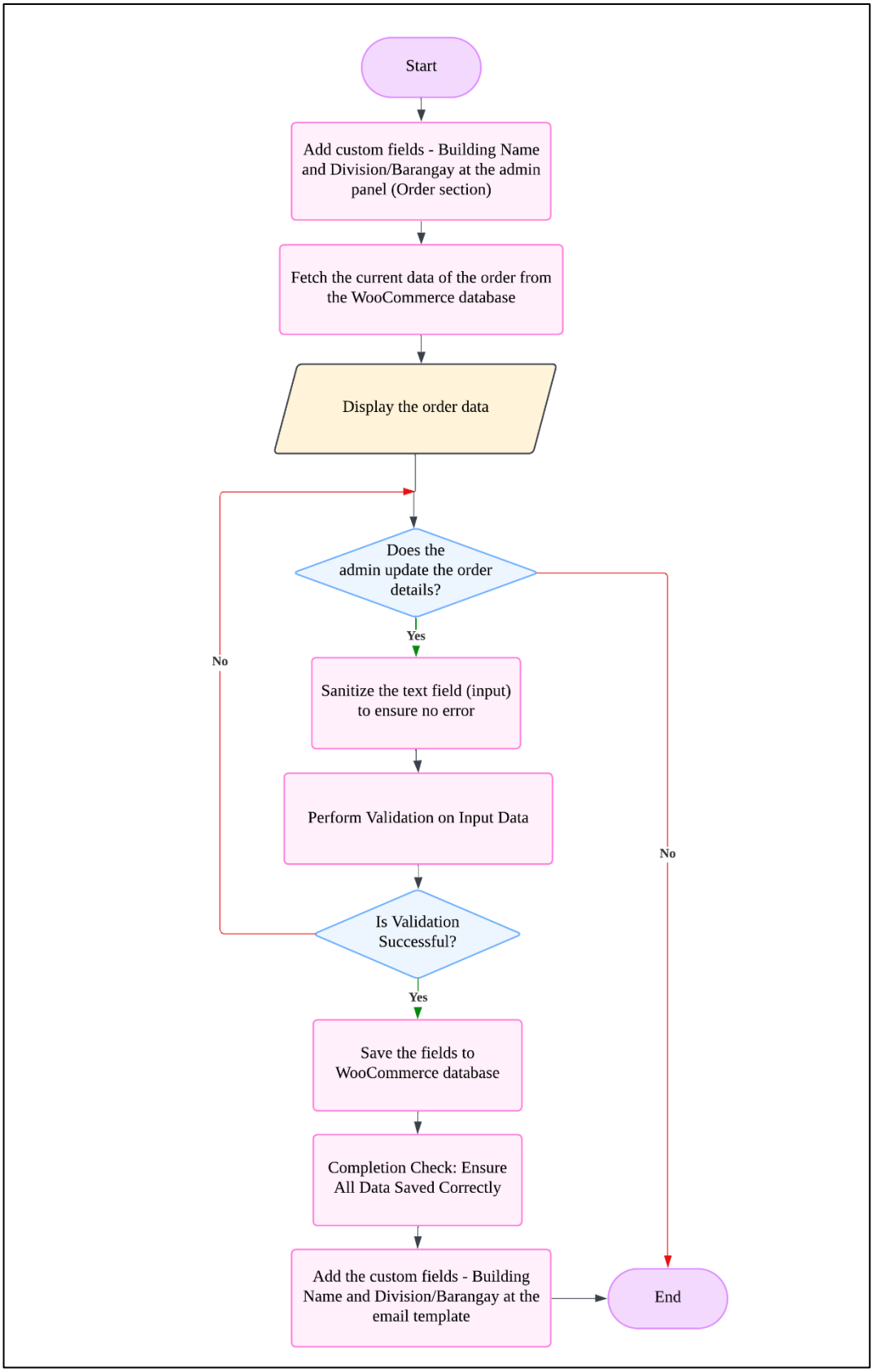

Figure 3.10: Flowchart of the Process to add and show the custom fields in the Admin Dashboard (In Order Section)

3.4.4.4 Sample Interface

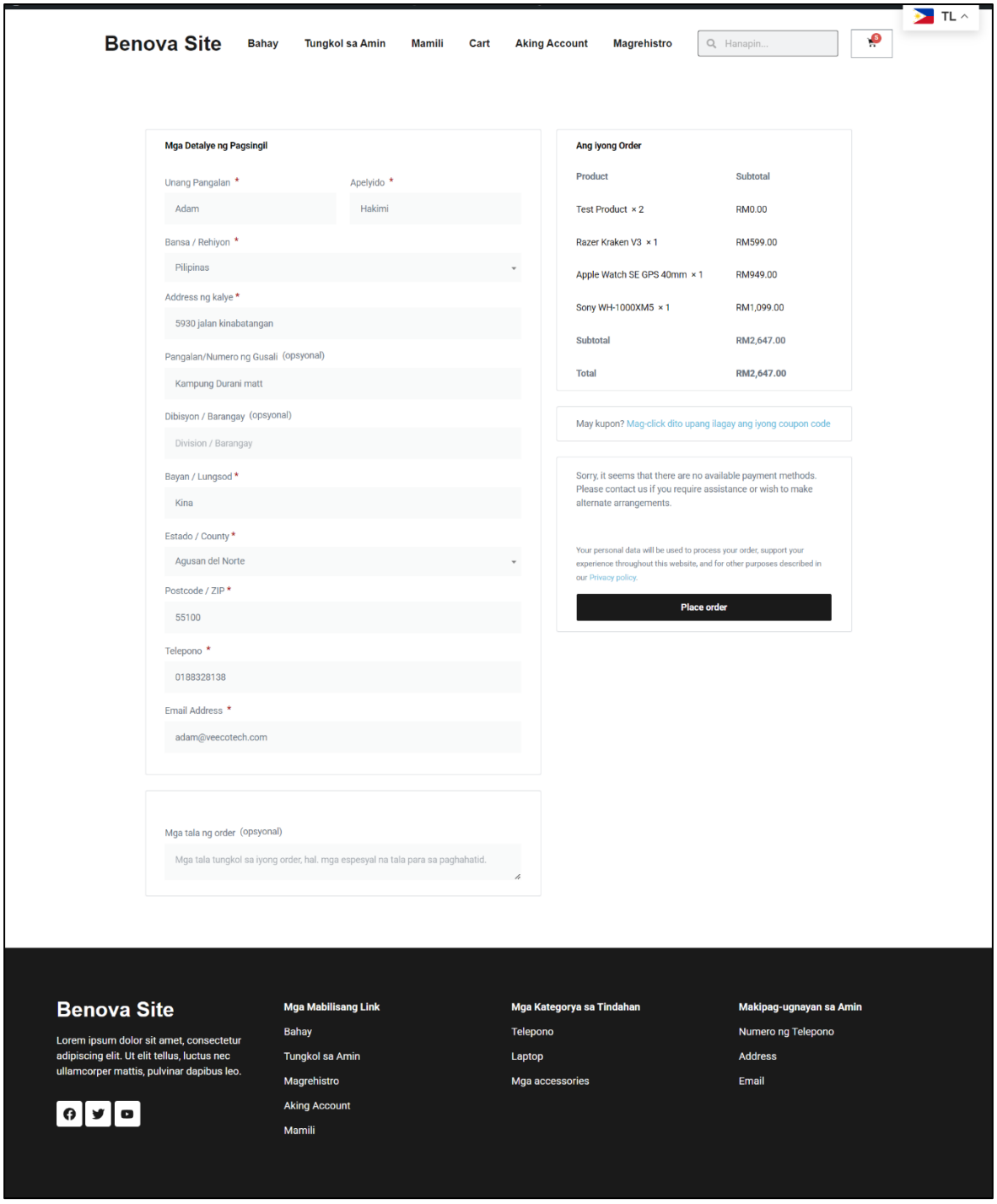

Figure 3.11: Sample Interface of the Checkout Page in Tagalog with some additional custom fields

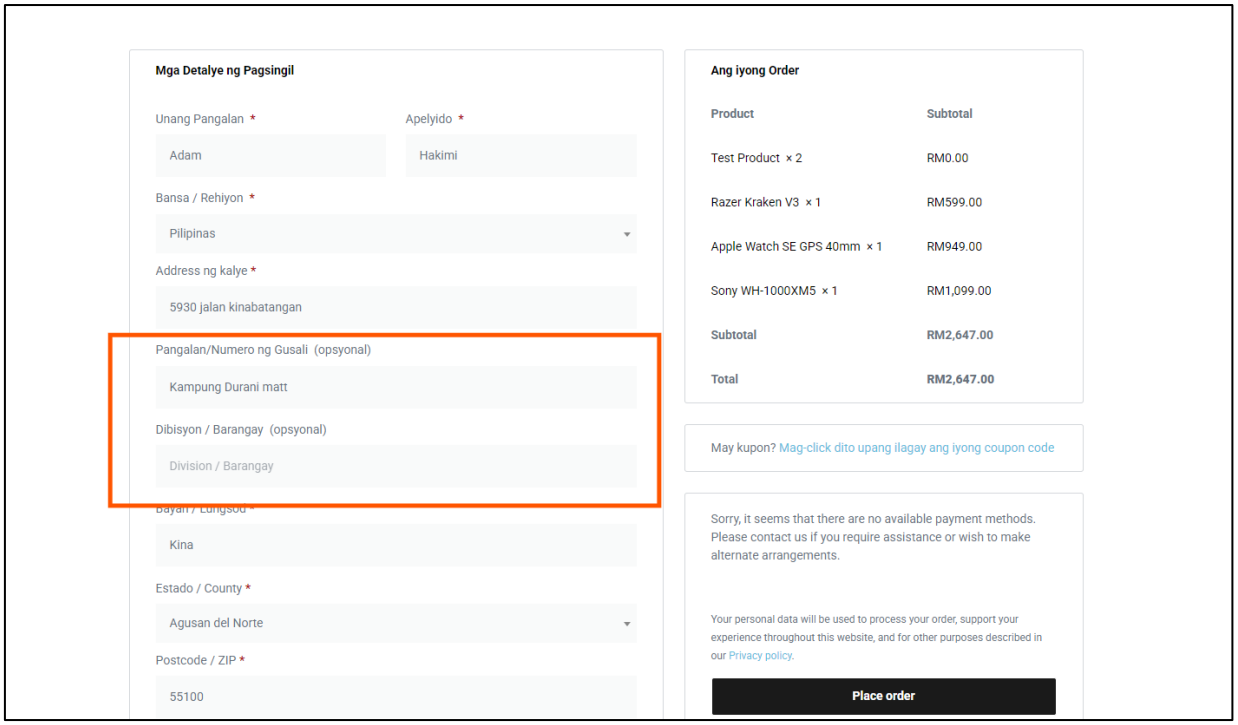

Figure 3.12: The added custom fields if users use Tagalog (Optional)

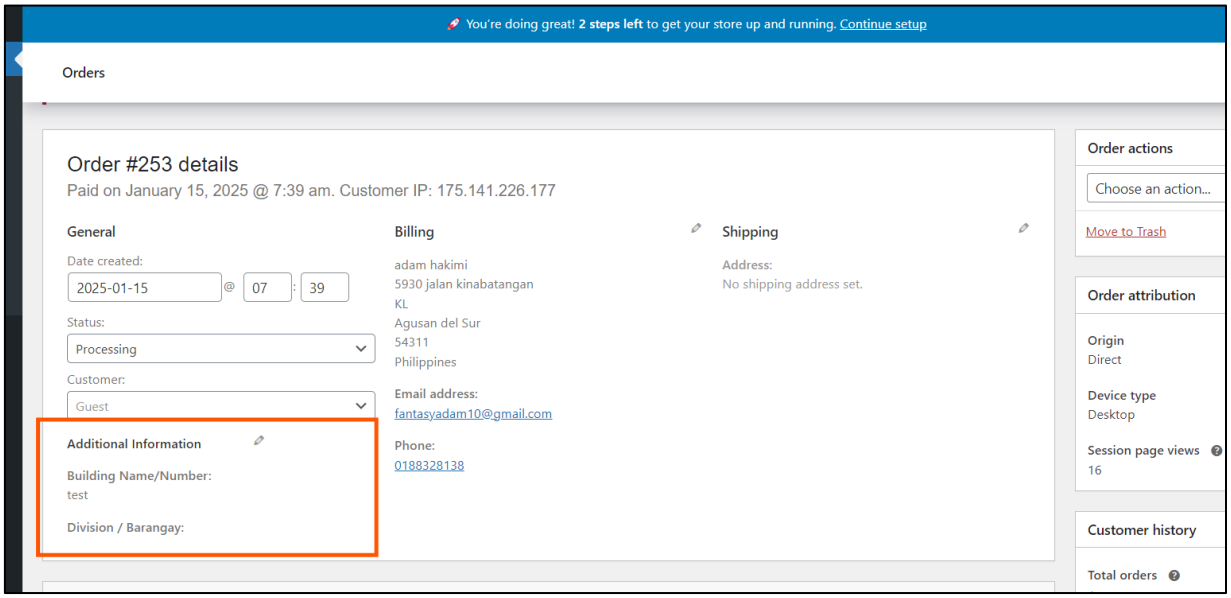

Figure 3.13: The added custom fields in the Order details section in Admin Dashboard