Heath-Medicine-Quest  
Website

February 27, 2017

# Overview

## Project Background and Description

This is a website application for searching and buying health care products of a worldwide pharmaceutical companies.

## Project Scope

* E-Commerce Website application of searching and buying medicine online.
* Consists interactive interfaces to access different medicines of different category like Ayurveda, Homeopathy, Baby Care, Beauty Care, etc.
* Interfaces will enable to search medicine and add to cart hence will have option to place order with order cancellation.

## Software and Hardware Requirements

* Software
  + HTML5, CSS, Bootstrap, JSP
  + JavaScript, jQuery, JSON
  + Java, Spring MVC, Hibernate
  + H2 Database
  + Version Control
* Hardware
  + Processor - Intel Core i5/i7
  + Installed Memory(RAM) – At least 4.00GB
  + System Type – 64 bit OS, x64 based processor
  + Network - Internet

## High-Level Requirements

This system includes the following:

* Ability to allow both internal and external users to access the application without downloading any software.
* Ability to interface with the existing database application.
* Ability to incorporate automated routing and notifications based on business rules(optional).

## Deliverables

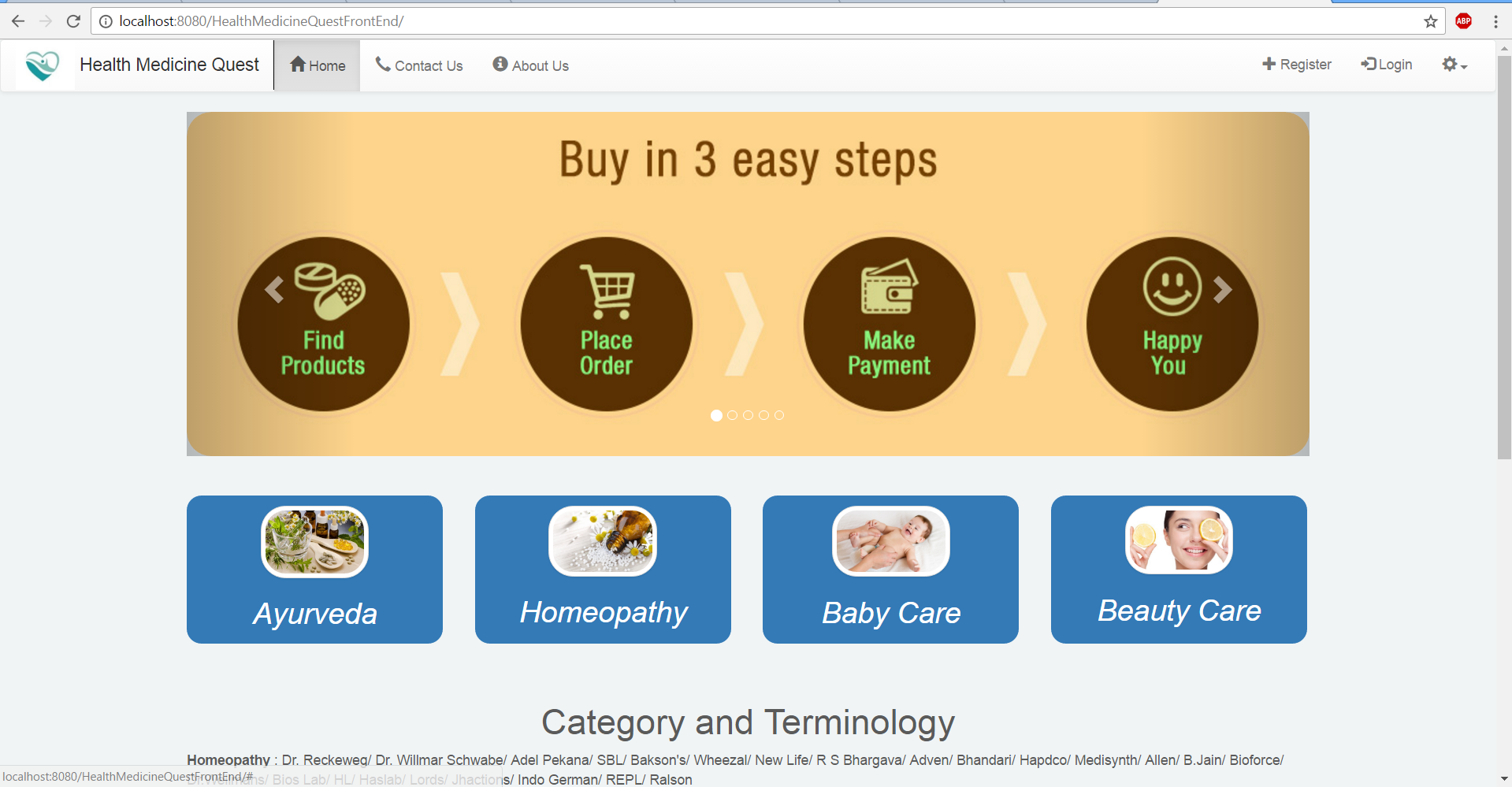
* Admin User will be able to insert, update and remove products.
* Customer User will be able to search product of different categories.
* Customer User will be able to view product detail with product name, effects, benefit of usage, brand name, price, available sizes, etc.
* Customer will be able to add product to the Cart to buy any product. More than one product can be added to the cart.
* Order can be placed by the Customer from cart.
* Placing order will be completed with Order Summary.

# Approval and Authority to Proceed

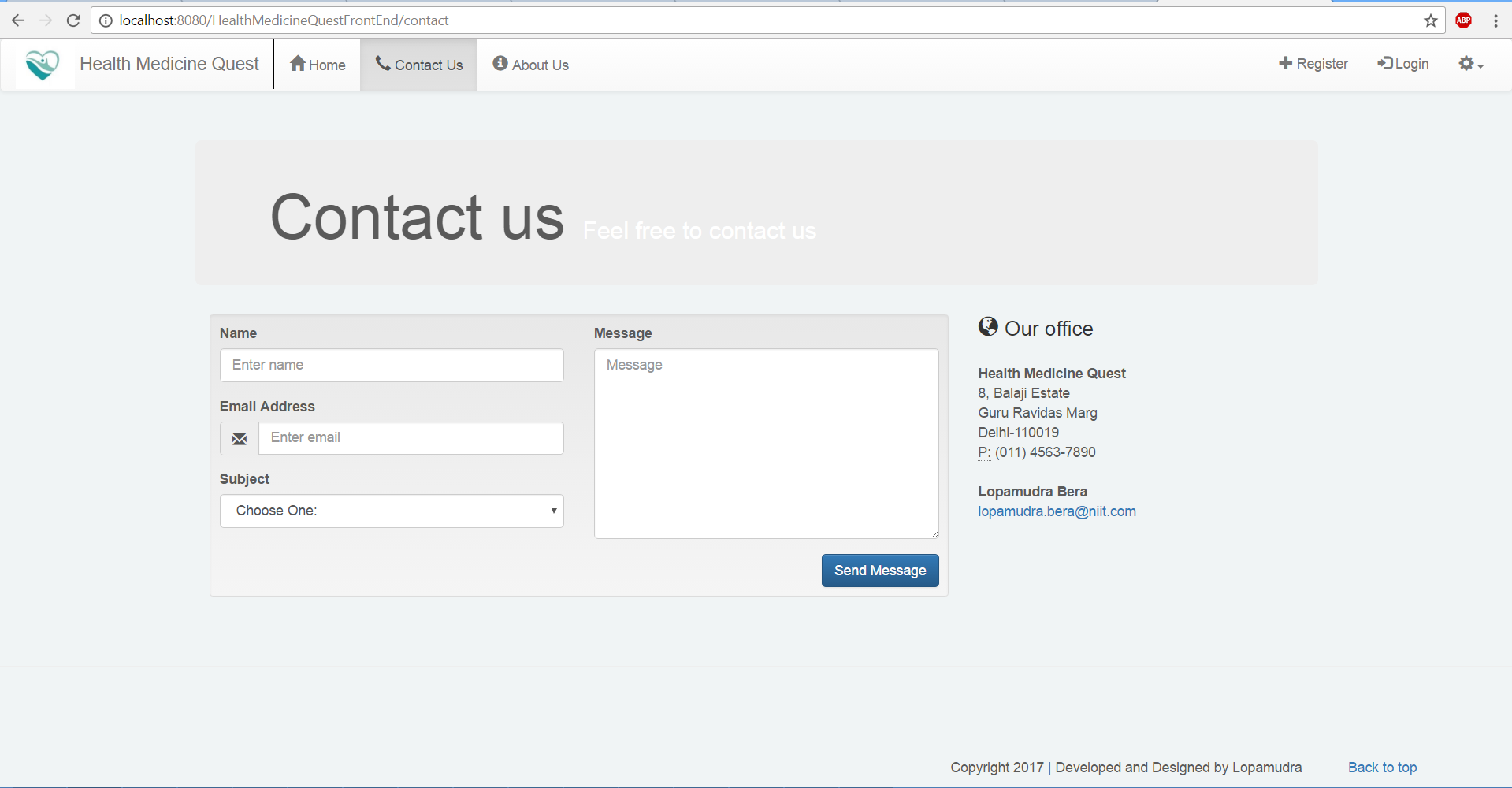
|  |  |  |
| --- | --- | --- |
| Name | Title | Date |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
| Approved By |  |  | Date |  | Approved By |  |  | Date |

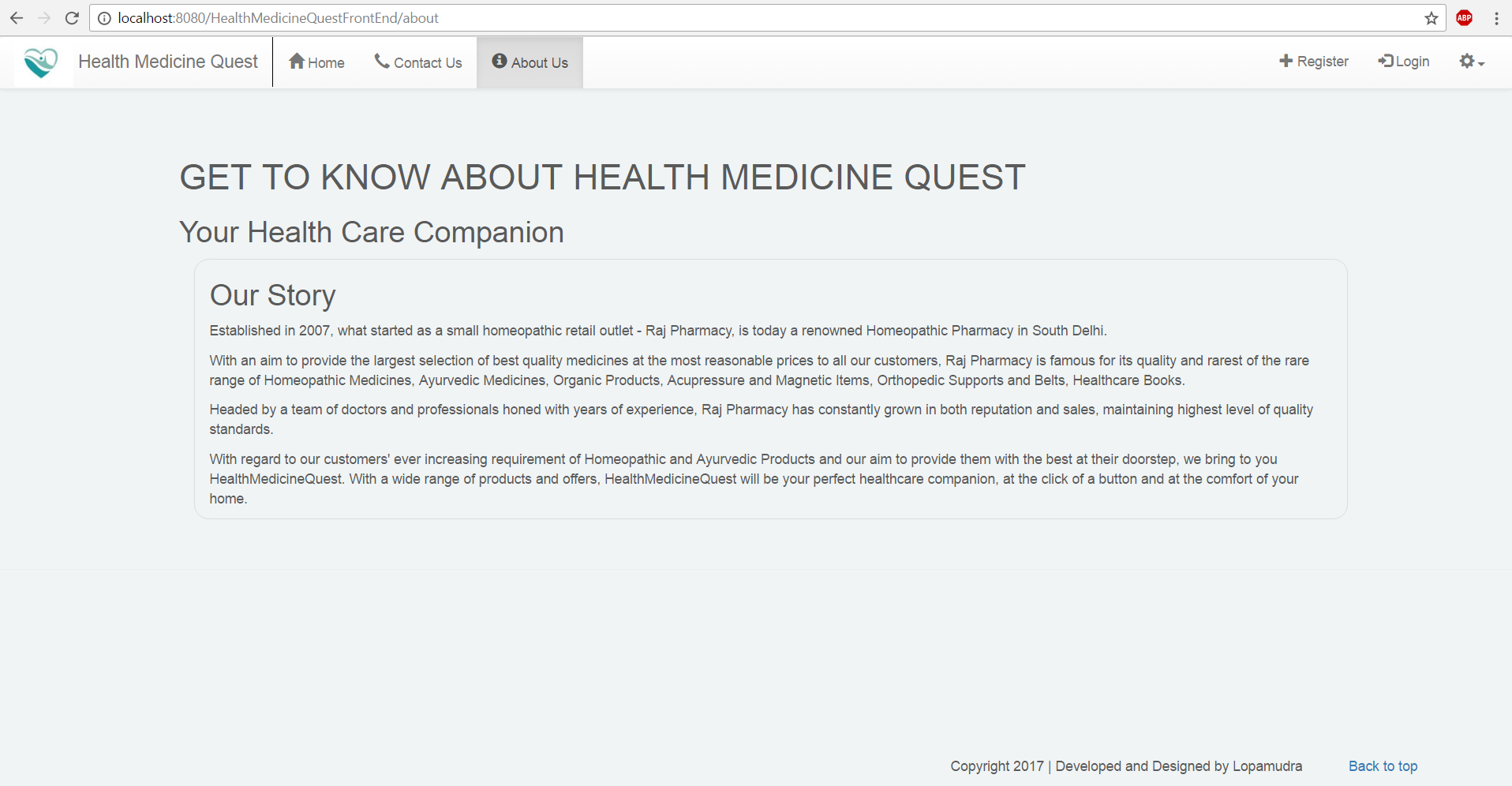
1. Landing Page



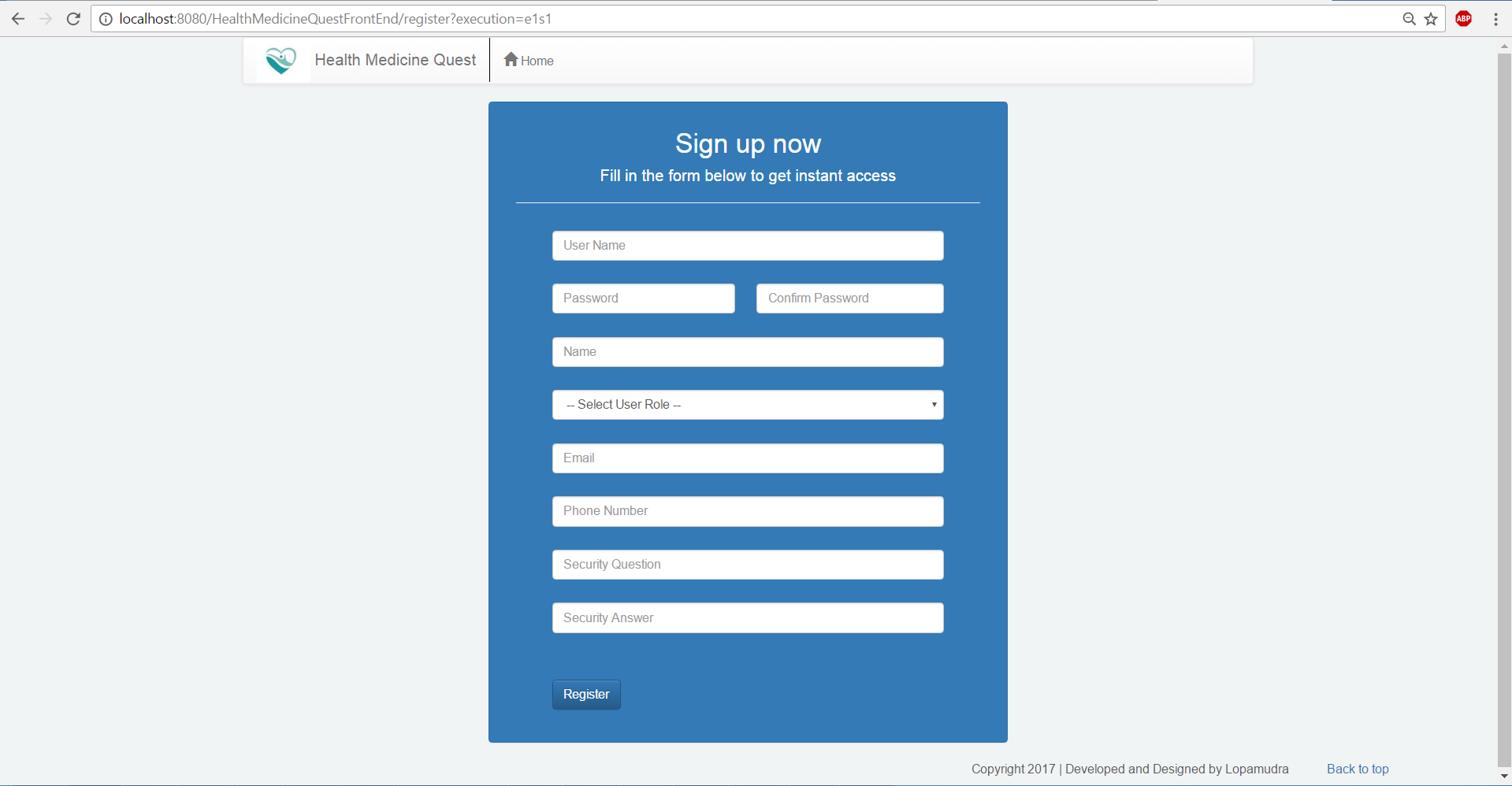
1. Technologies used to design the landing page are HTML, CSS, Bootstrap, JavaScript, JSP, Java, Spring MVC.
2. Landing page is built up on master page(page.jsp), navigation bar(navbar.jsp) , footer(footer.jsp) and content page(home.jsp).
3. Static Resource methodology is implemented to load resources like images, css, etc. which is configured in dispatcher-servlet using <mvc:annotation-driven /> and <mvc:resources> tags.
4. In FrontEndPageController class respective request mapping has been done.
5. Contact Us Page



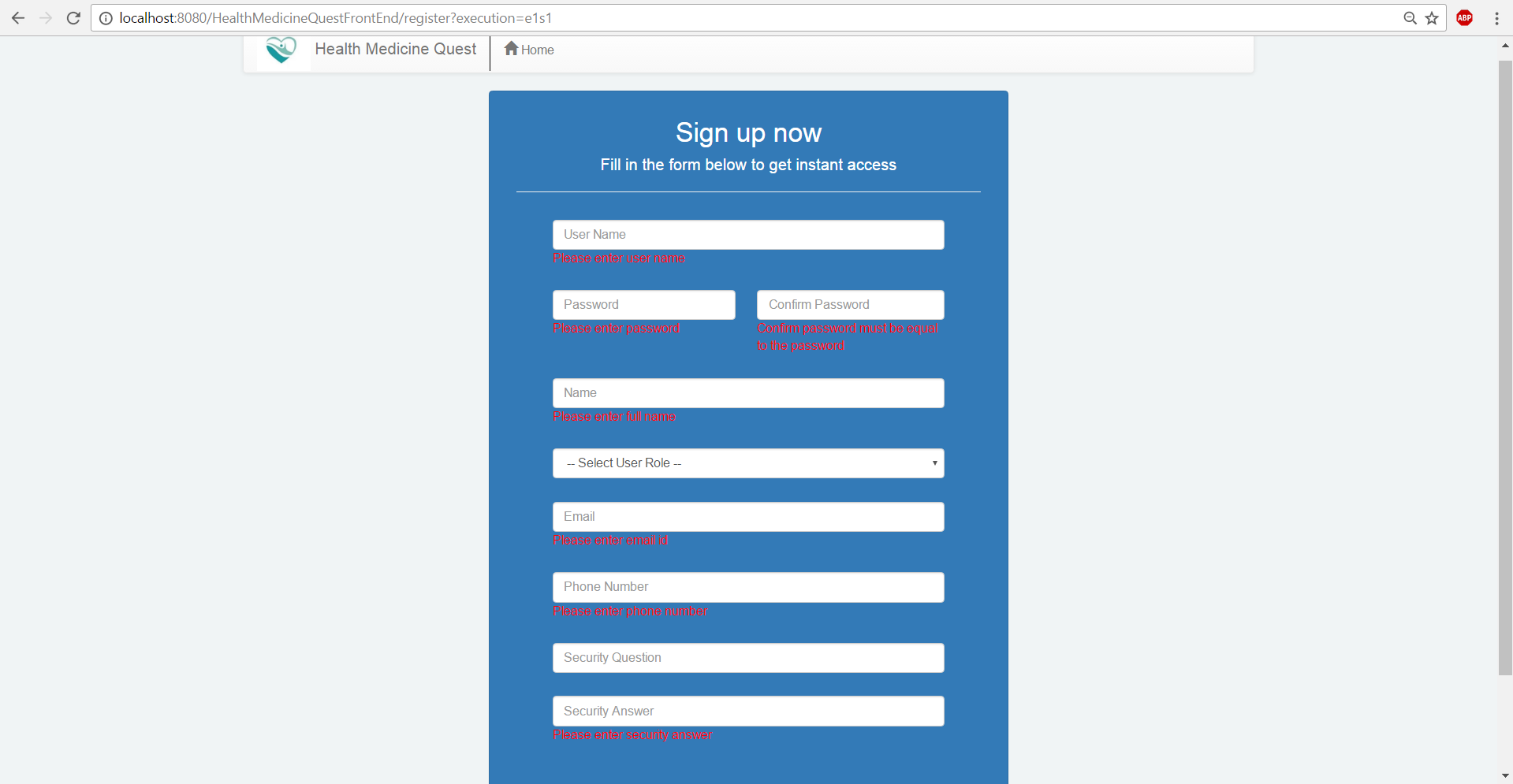
1. Technologies used to design the landing page are HTML, CSS, Bootstrap, JavaScript, JSP, Java, Spring MVC
2. Caontact Us page is built up on master page(page.jsp), navigation bar(navbar.jsp) , footer(footer.jsp) and content page(contact.jsp).
3. In FrontEndPageController class respective request mapping has been done.
4. About Us Page



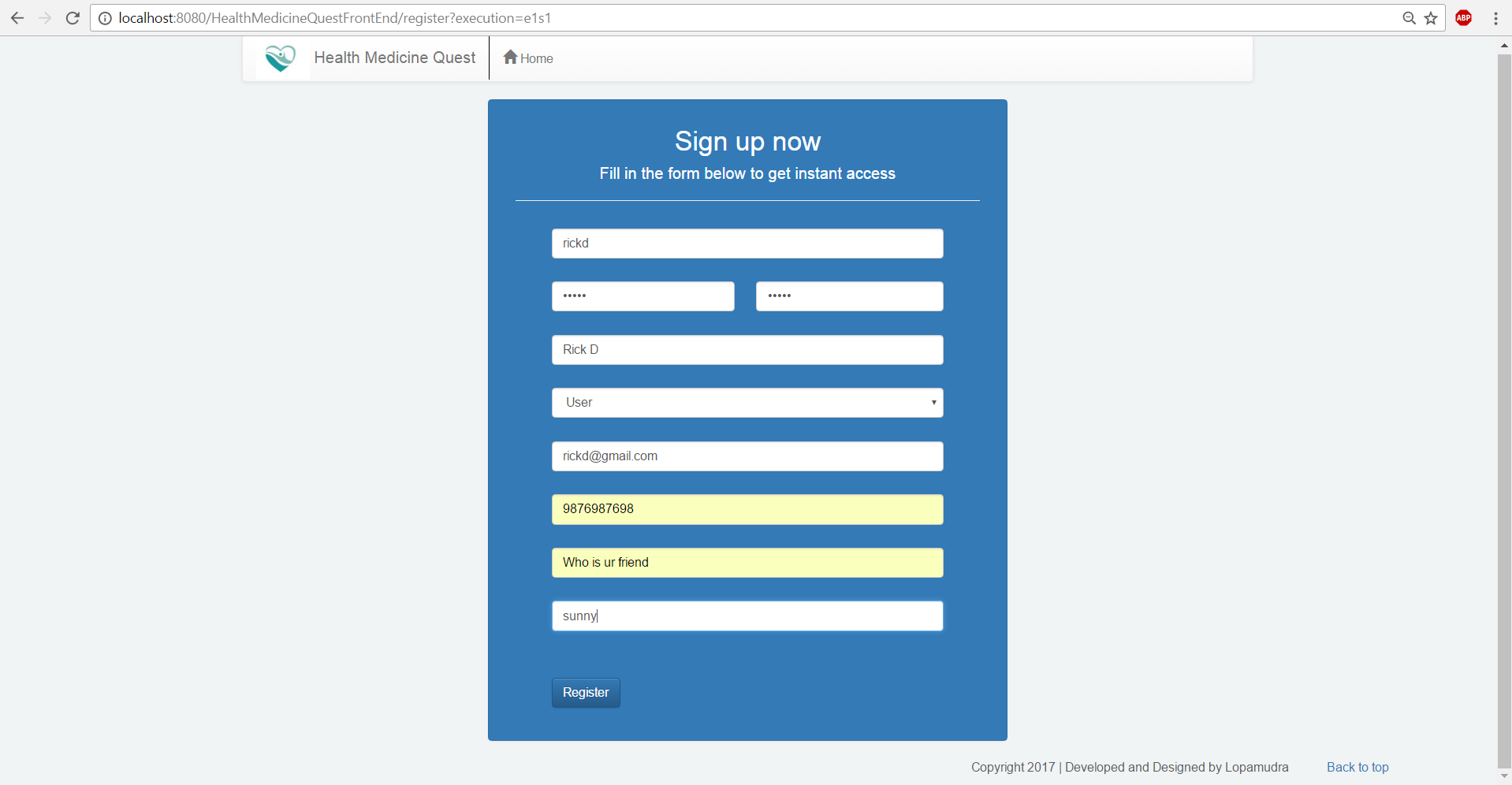
1. Technologies used to design the landing page are HTML, CSS, Bootstrap, JavaScript, JSP, Java, Spring MVC
2. About Us page is built up on master page(page.jsp), navigation bar(navbar.jsp) , footer(footer.jsp) and content page(about.jsp).
3. In FrontEndPageController class respective request mapping has been done.
4. Registration Page



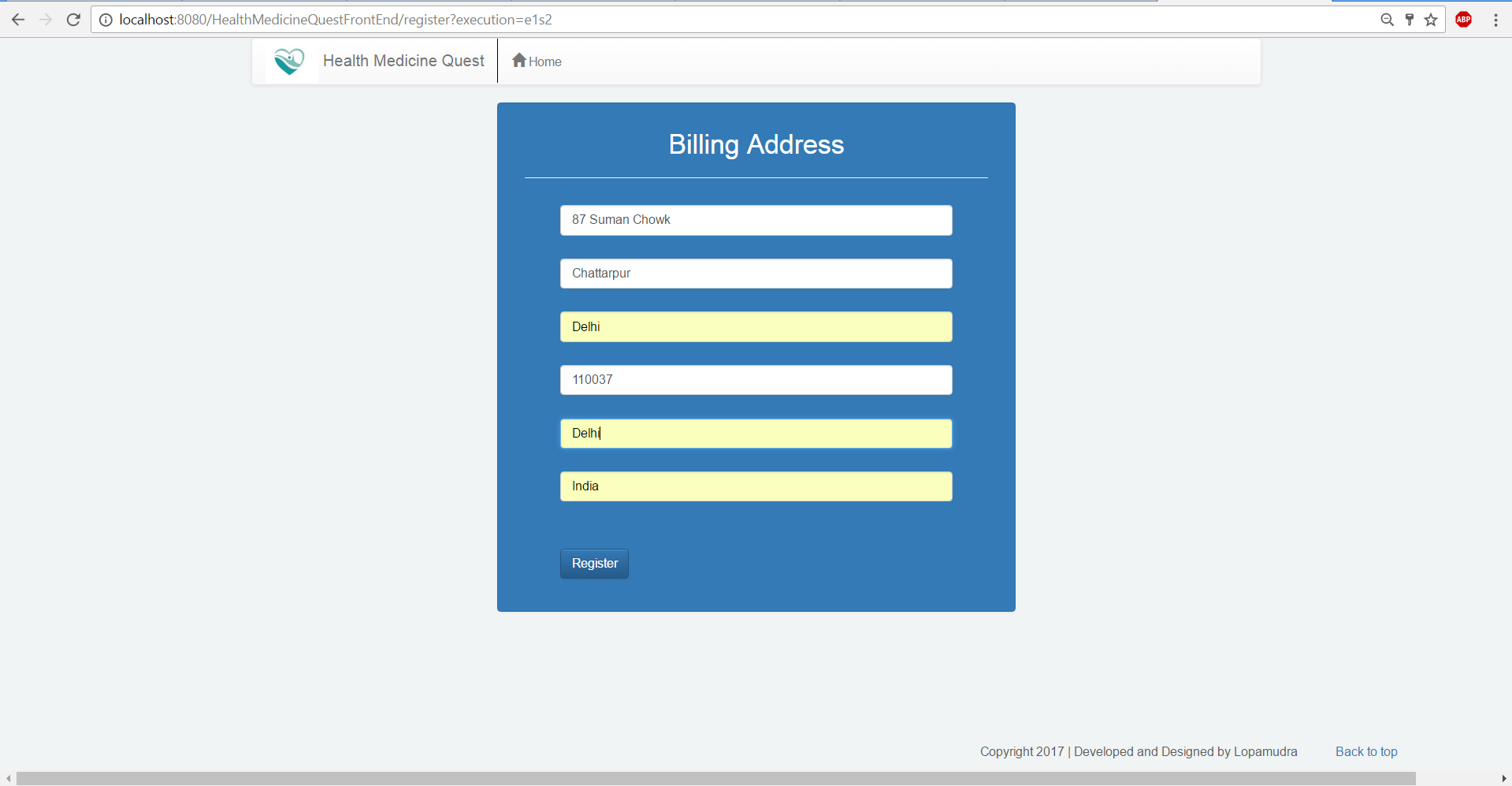
1. Technologies used to design the following page are HTML, CSS, Bootstrap, JavaScript, JSP, Java, Spring MVC.
2. This page is built up on navigation bar(navbar.jsp) , footer(footer.jsp) and content page(register.jsp).
3. Spring WebFlow is implemented to complete the whole registration process including accepting address. Hibernate Validation is used to validate data during, data acceptance from user.
4. Whole process is divided among 2 projects, FrontEnd and BackEnd:
   1. FrontEnd—register.jsp, billing.jsp, preRegister.jsp, welcome.jsp and exception.jsp pages configured through cartFlow.xml and corresponding handler code is mentioned in cartHandler.java and CartModel.java, mapping information is defines inside the dispatcher-servlet.xml file.
   2. WebFlow reduces overhead of carrying over the data from one request to another as it can be done by flow scope. It helps especially in cases like Breadcrumbs navigation. Flows can be further divided into Sub Flows to reduce the complexity.
   3. BackEnd—dao, daoimpl, model, hibernate configuration, test cases are implemented.
   4. Hibernate-core, h2-database, commons-dbcp2, junit, spring-orm, javax-persistence, hibernate-validator are the dependencies require to associate to the project through pom.xml file.
   5. H2 database is used to persist data which is light weighted and easy to configure with Hibernate.
   6. Entity classes are configured with persistence and hibernate validator related annotations.
5. Registration Flow with Error Validation



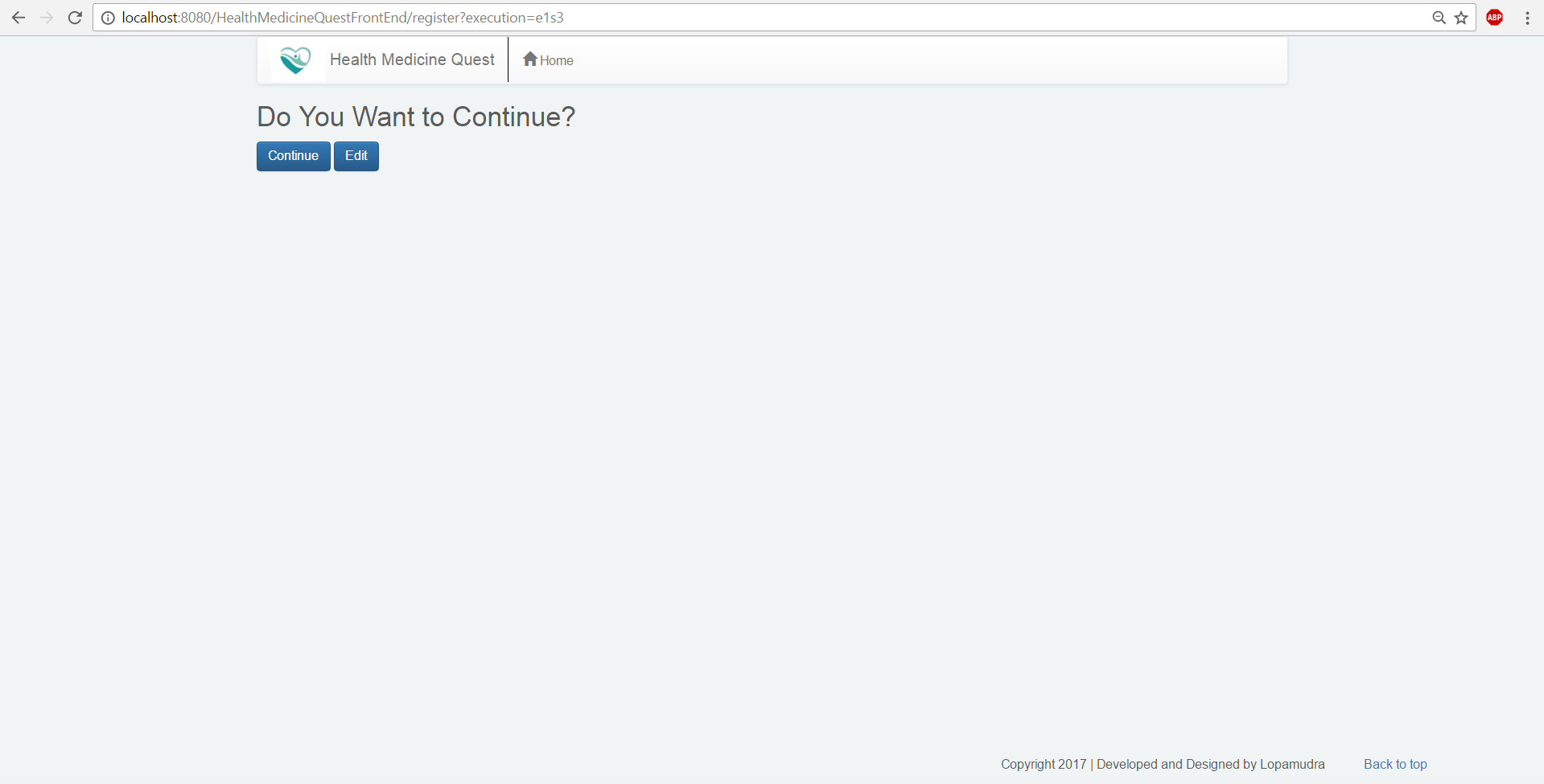
1. Registration Flow with sample data



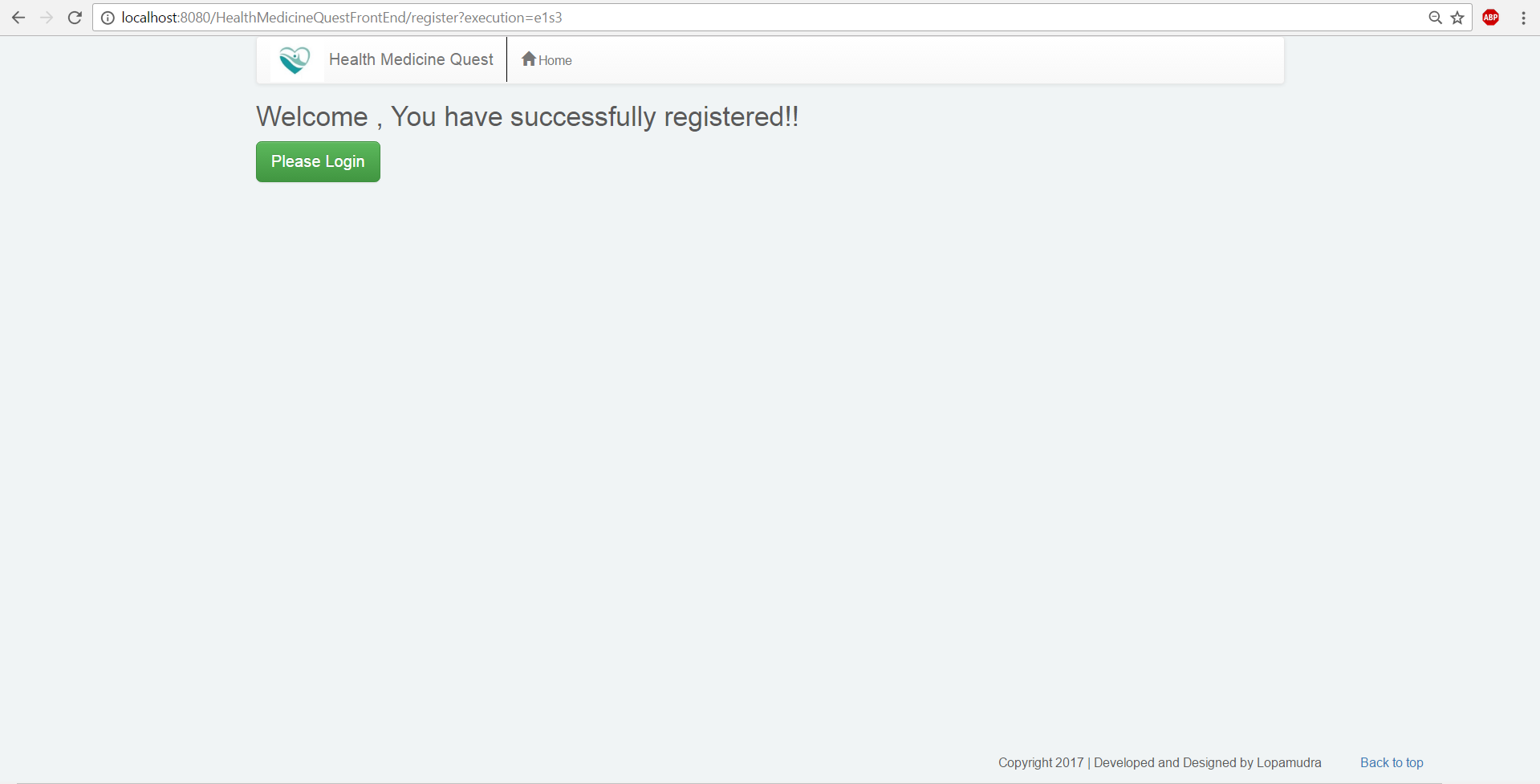
1. Billing Address page to accept user’s address



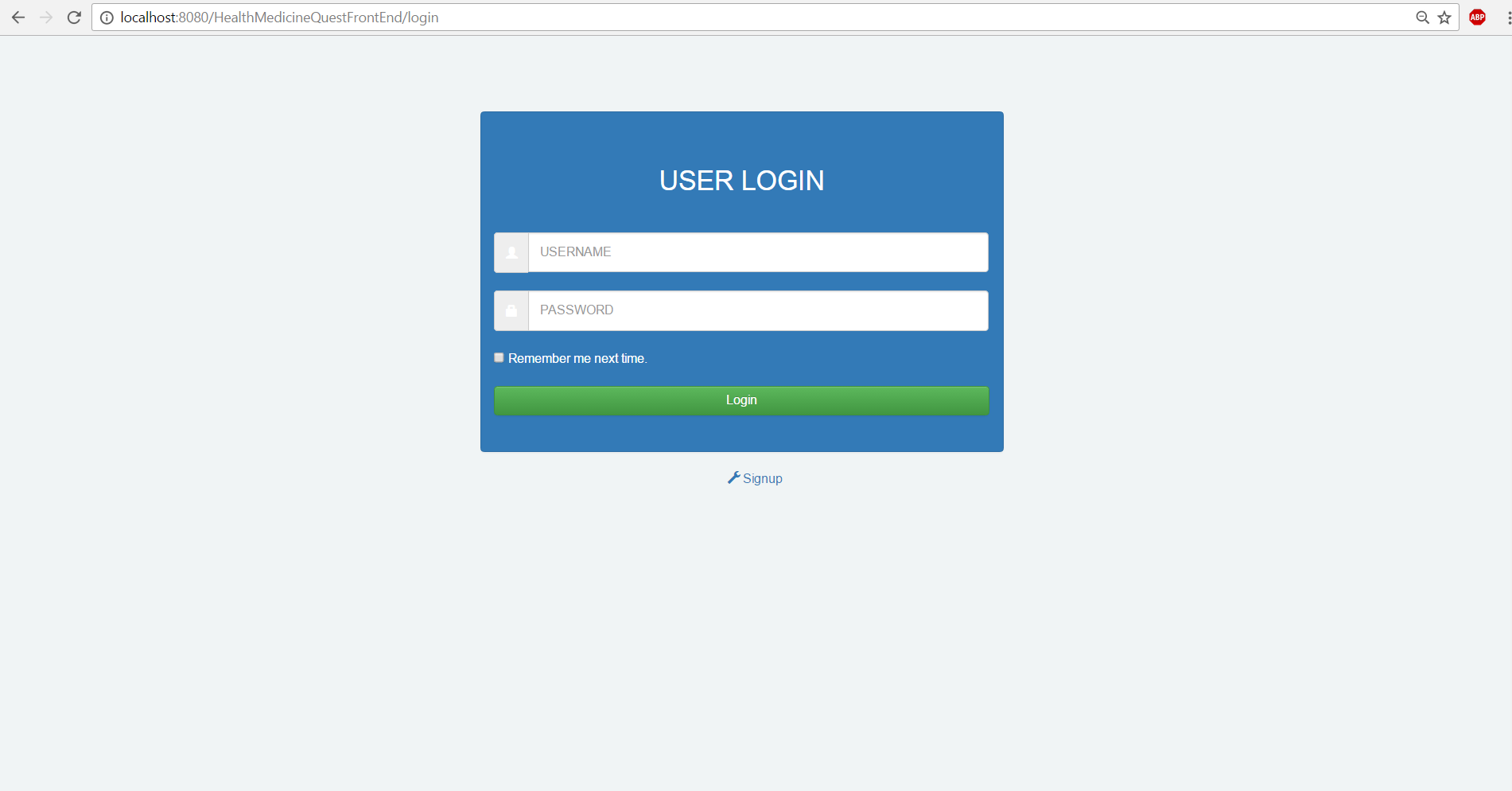
1. PreRegister page to accept confirmation to proceed the registration process.



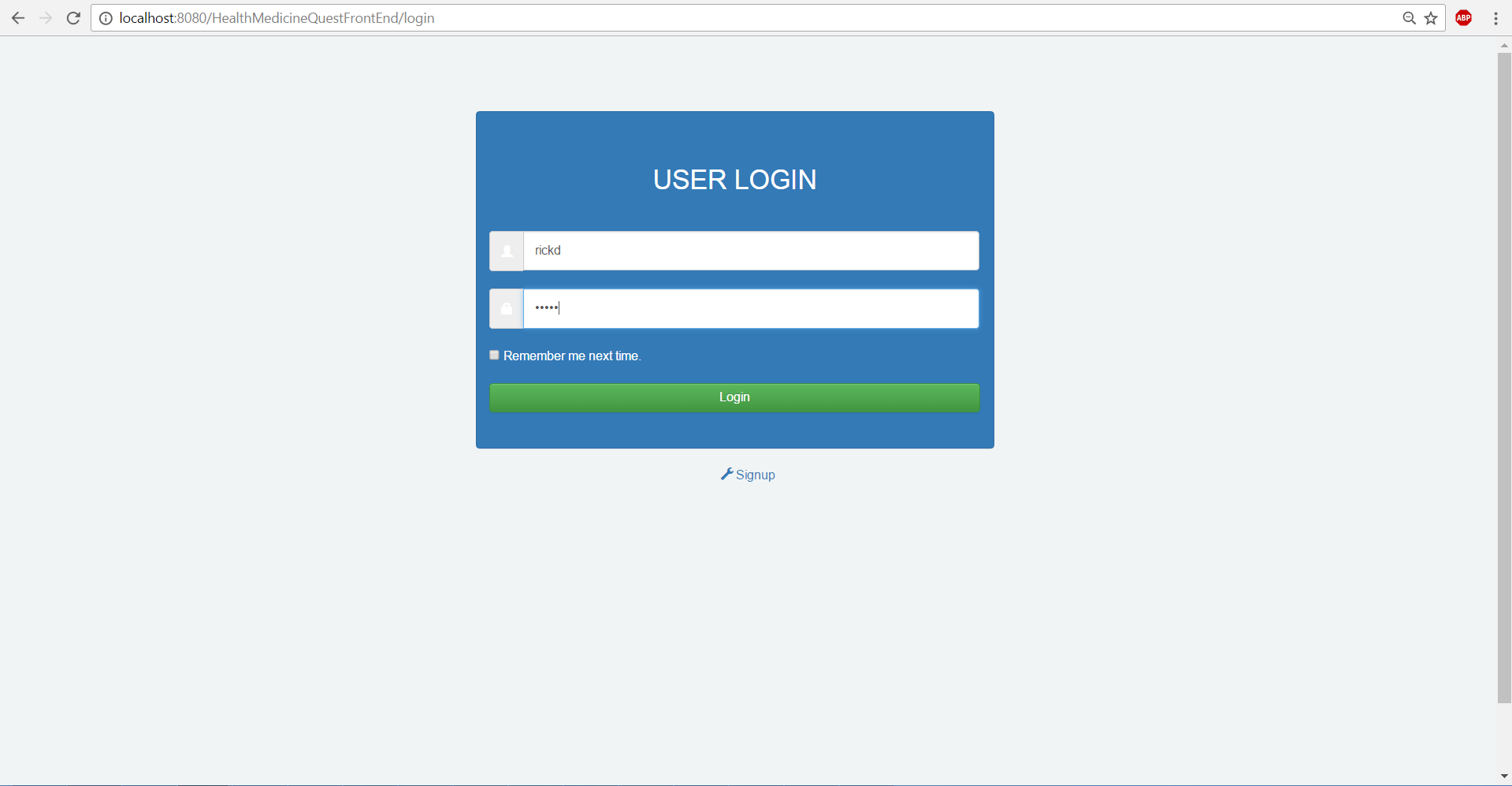
1. Success page at the end of registration flow.

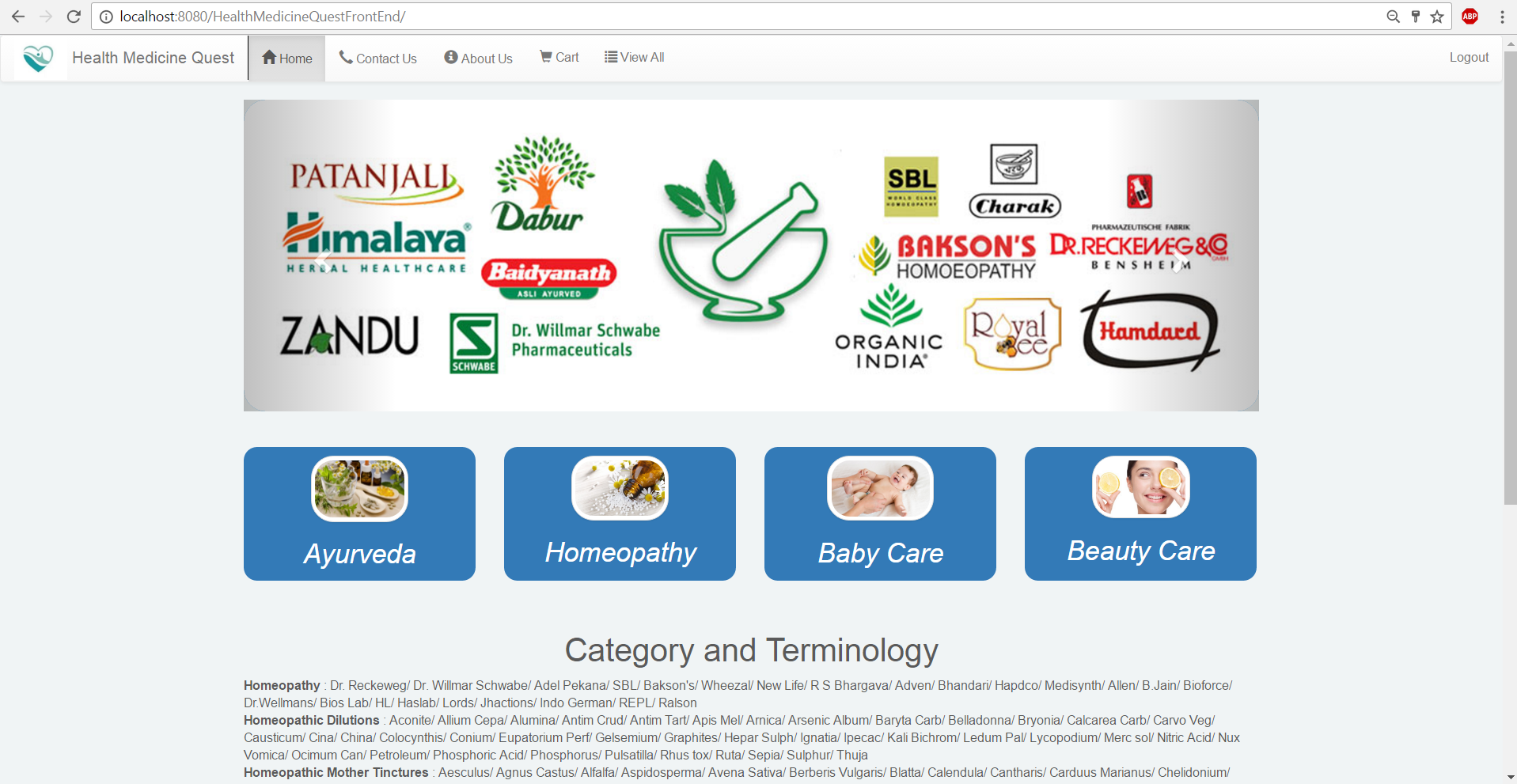
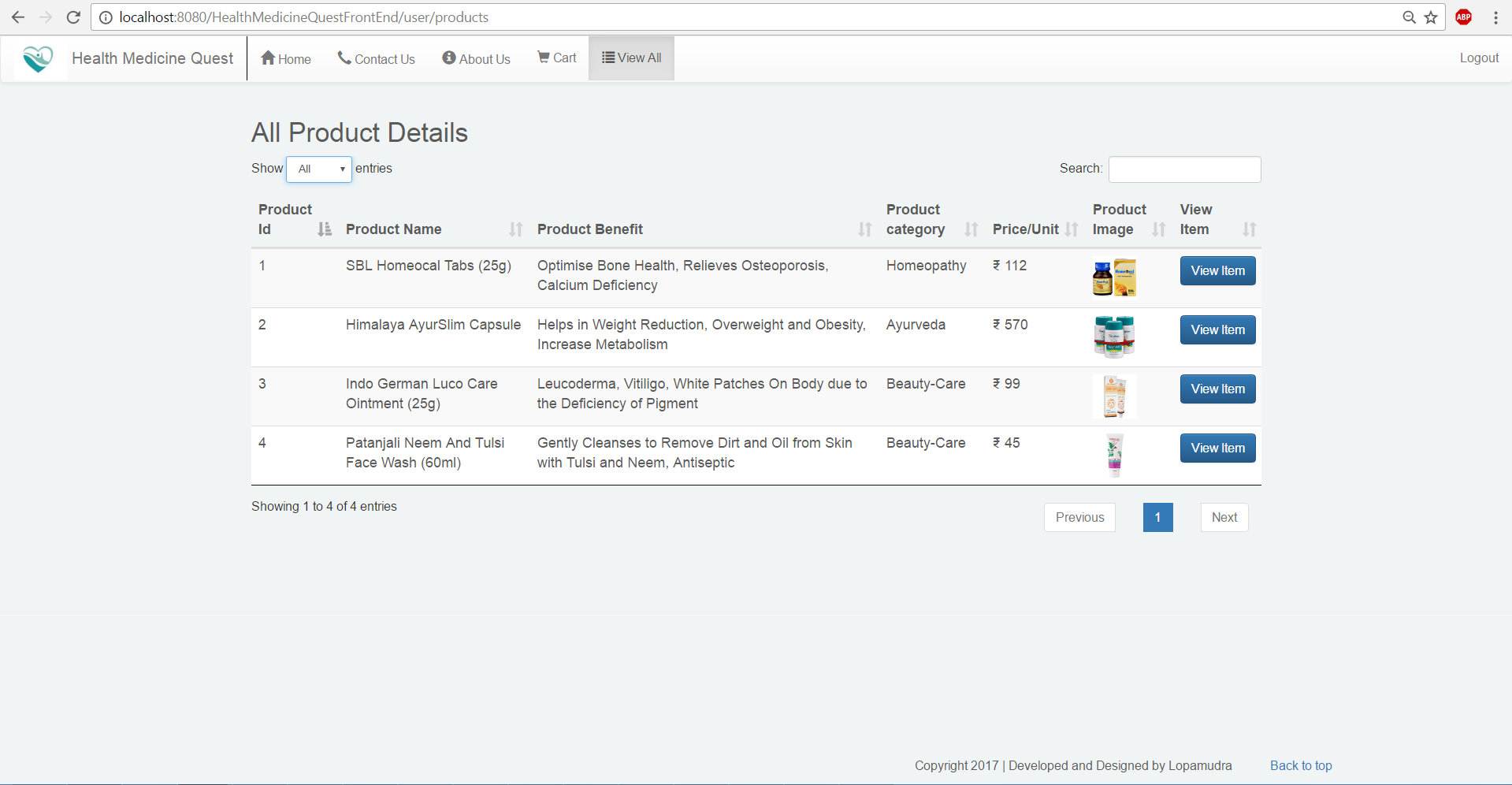
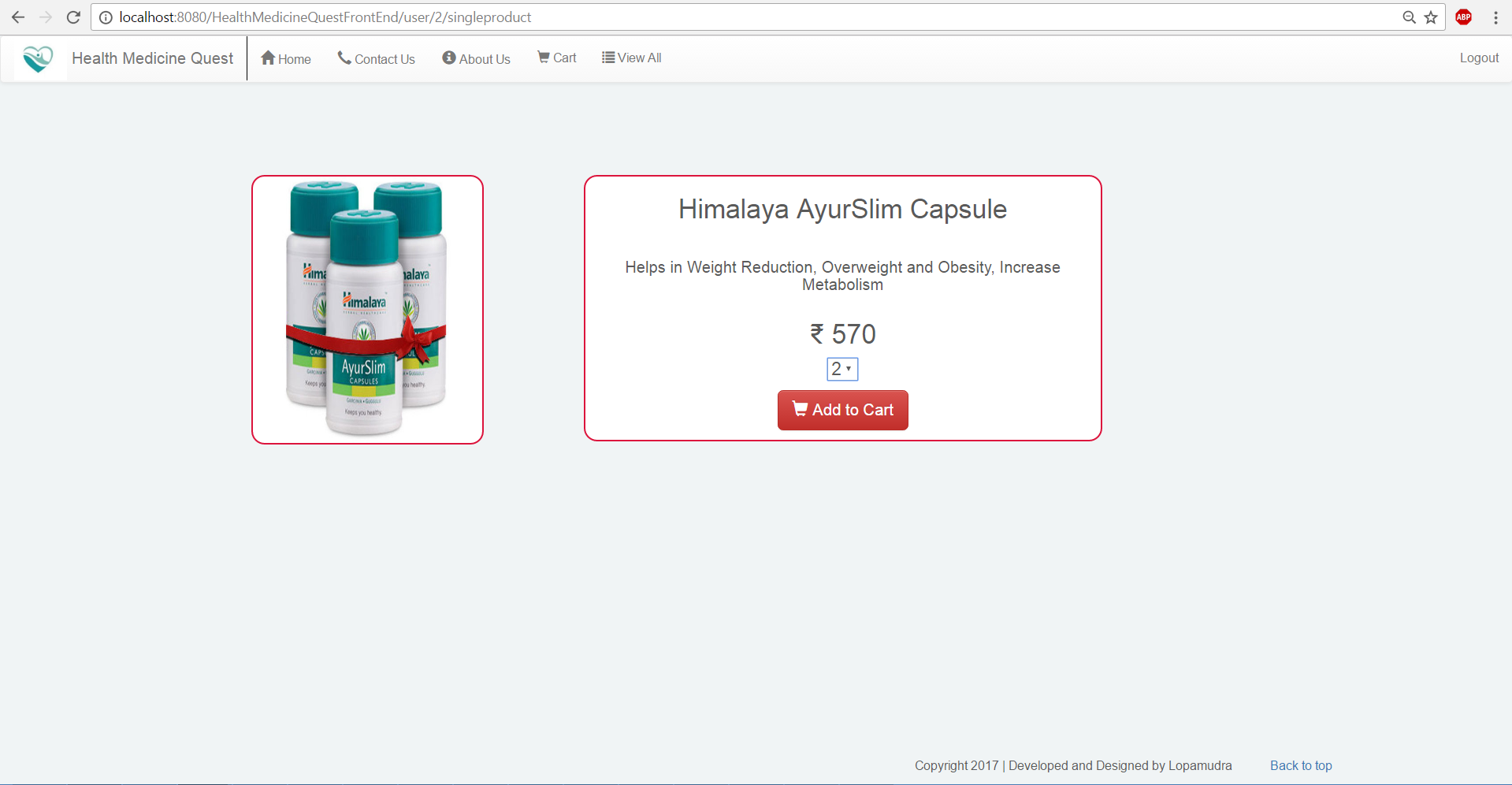
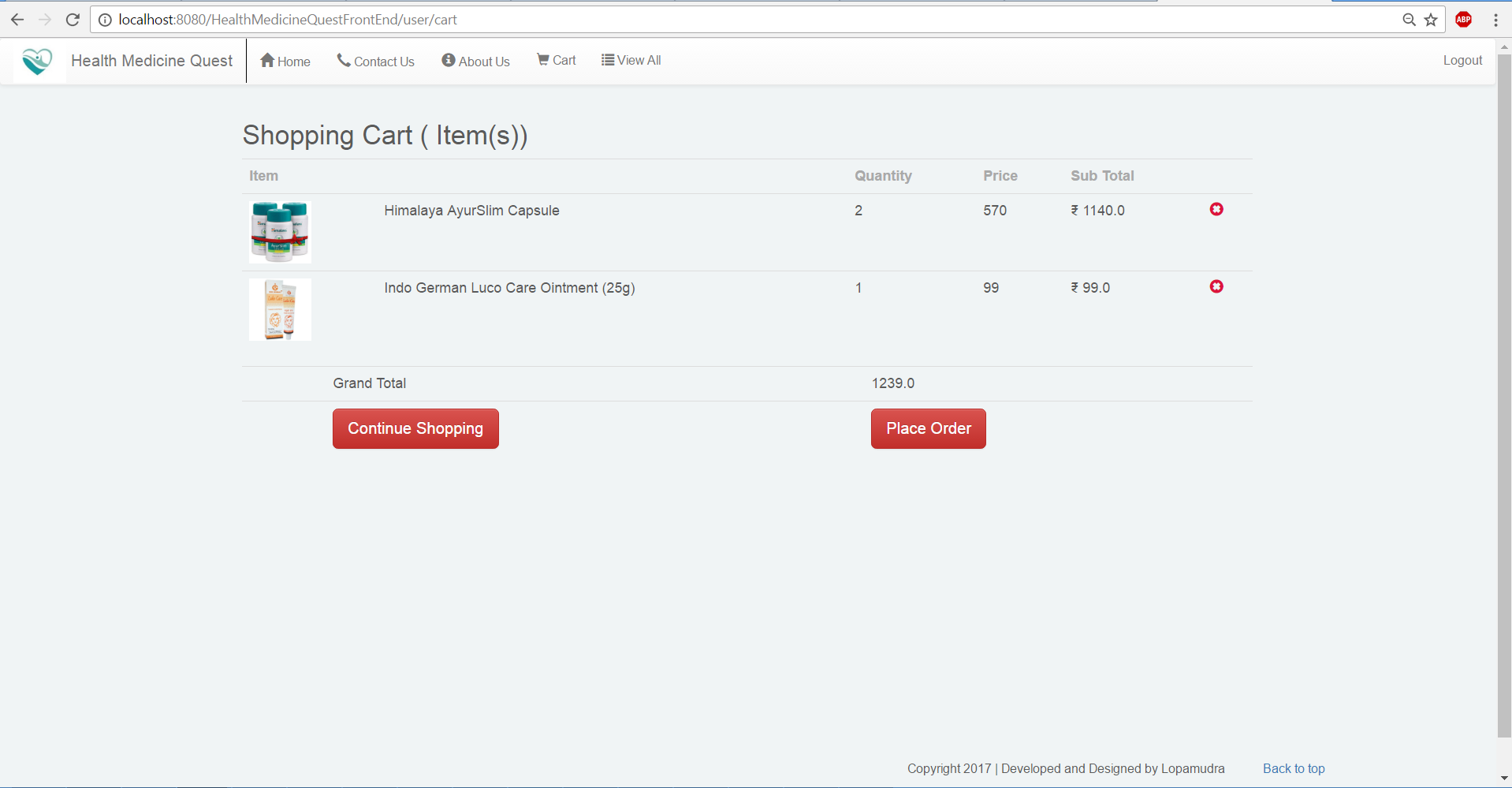


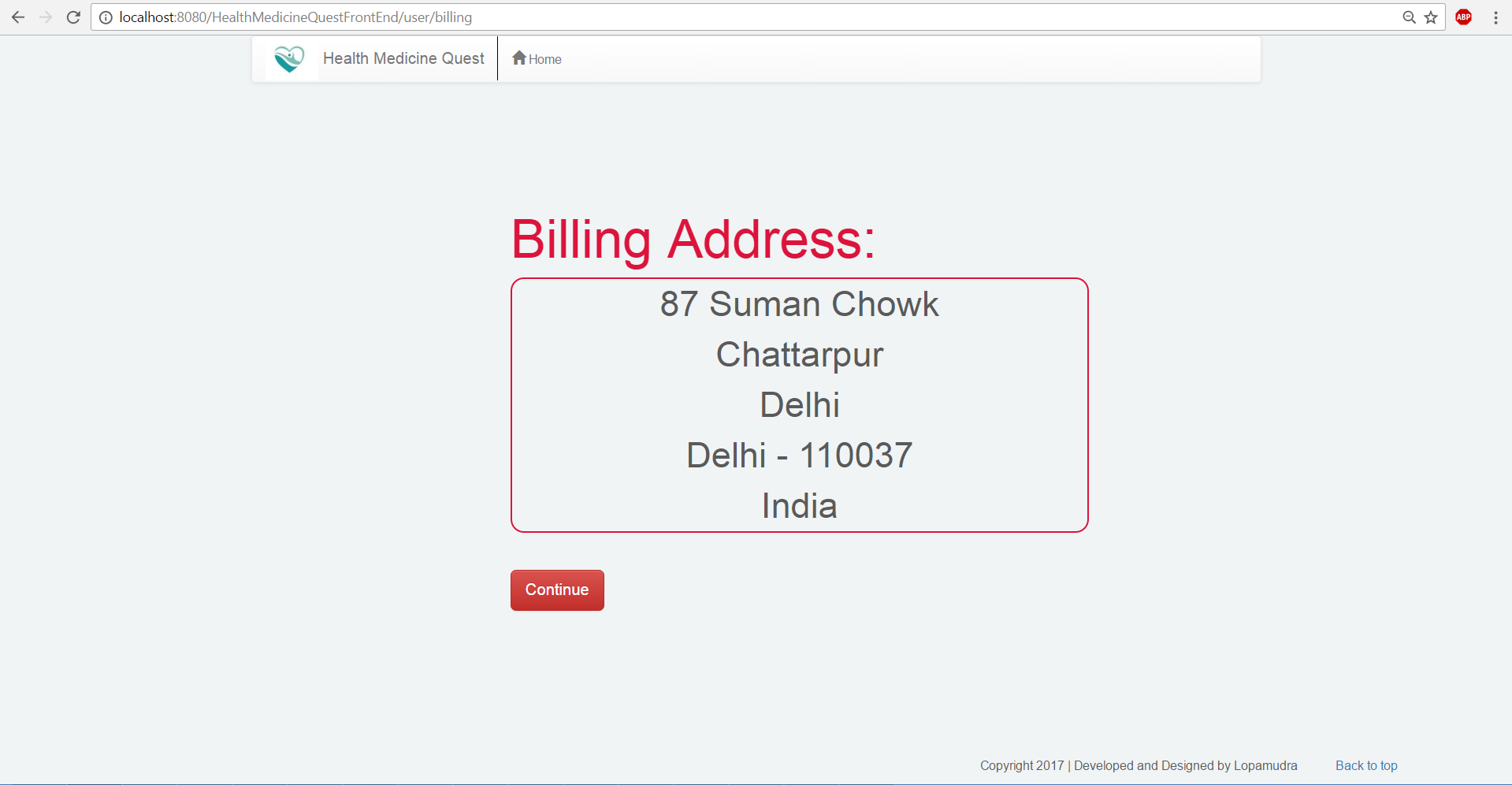
1. User can redirect to home page or to login page from this page.
2. Login page



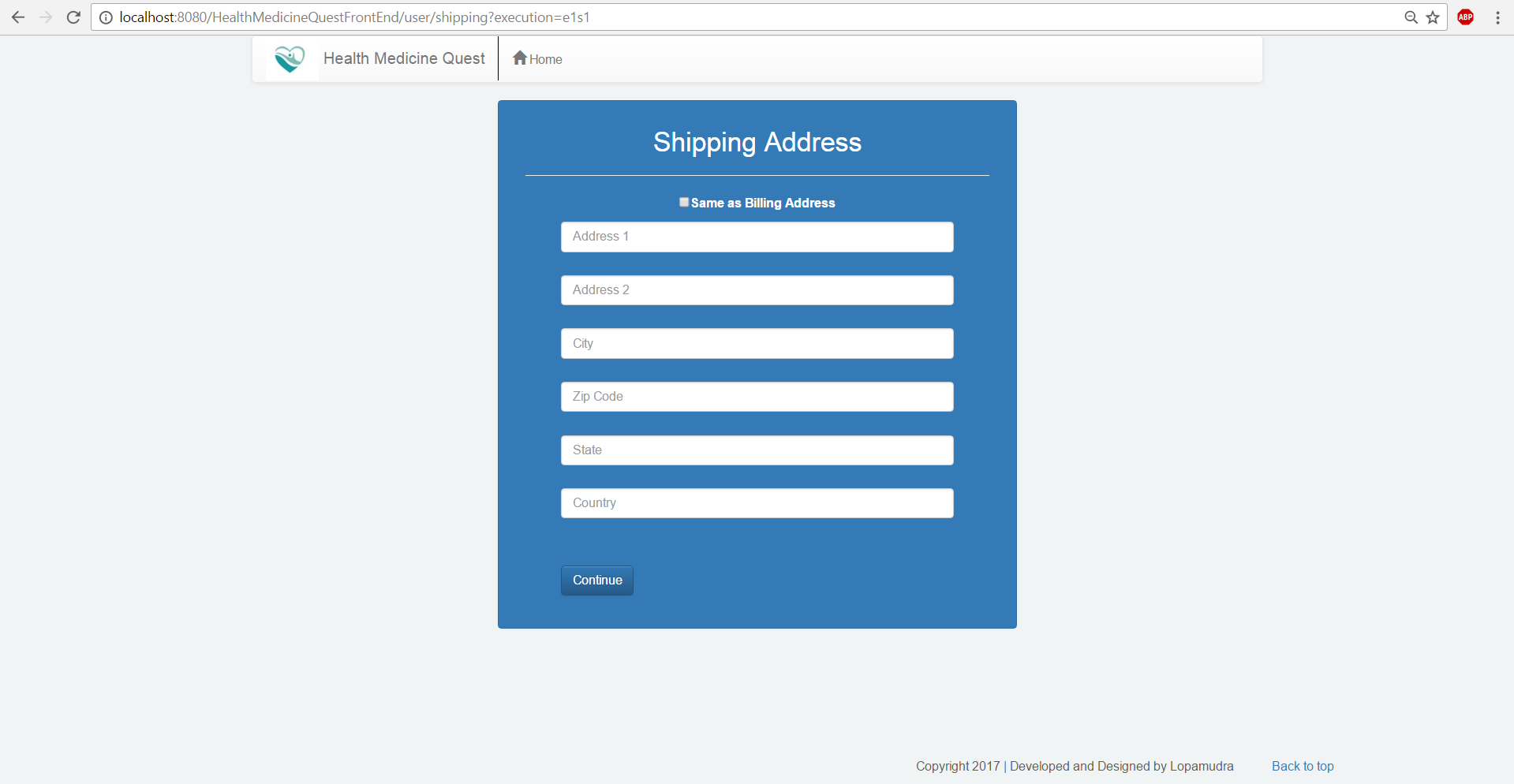
1. Spring-security is implemented in login process.
2. Whole process is configured with web.xml, applicationContext.xml and application-security.xml configuration files.
3. SecurityFilterChain is filtering each request through the spring security to deny annonymous and unauthorized access in web.xml file.
4. applicationContext.xml is intercepting user specific request by the help of hasAuthority method. Aslo CSRF(Cross Site Request Forgery) is implemented to avoid redirecting to other website. Enabling Logout link after each user’s login is emplemented. User’s credential is validated with the database data through <authentication-manager>.
5. Centralized navbar.jsp is modified with <security:authorize> tag to provide respective services to specified user, like Product management to only to the Admin, View product and cart flow to user, etc.
6. Final mapping to accommodate roles , admin and user is implemented in controller page to login().
7. Login Page with sample data of user



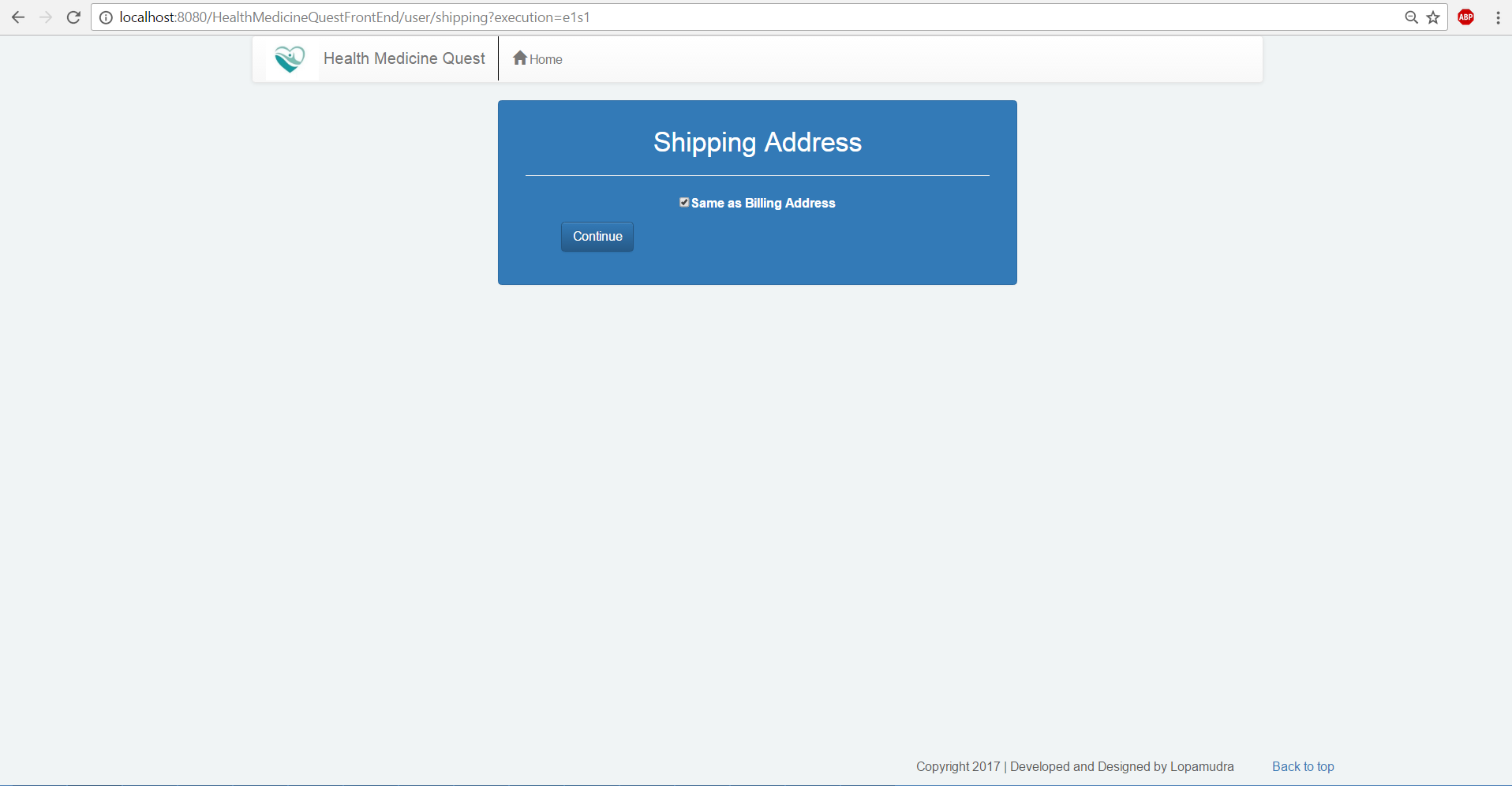
1. Home page after logged in by user. 
2. Cart and view all product services are available on the navbar.
3. Product Page to view all available products with sample data. 
4. User can click on “View Item” button to navigate to view single product detail page.
5. jquery datatable and corresponding bootstrap styling are used to populate the list of products in this page.
6. 3 jackson(core, databind and annotations) dependencies are user here to transfer data from backend to frontend in JSON format.
7. DataTables is a powerful jQuery plugin for creating table listings and adding interactions to them. It provides searching, sorting and pagination without any configuration. In this article we’ll go through the basics of DataTable and how to use some of the advanced features.
8. SingleProduct Page to view single product detail. 
9. User can insert product into user specific cart from this page by clicking “Add to Cart” button.
10. Selected item would be added to the cartItem table via CartItem dto, dao. And grant total and items will be added to the Cart table.
11. CartView Page to view the products, are selected by the user to buy. 
12. User can deselect the product by clicking on red cross button present at the end of each row.
13. By clicking “Place Order” button user can navigate to View the Billing Address. Or by clicking “Continue Shopping” user can navigate to view all products.
14. Data is retrieving from Cart and CartItem table through the FrontEndCartController.
15. Billing Address Page. Cart flow is triggering from this page.



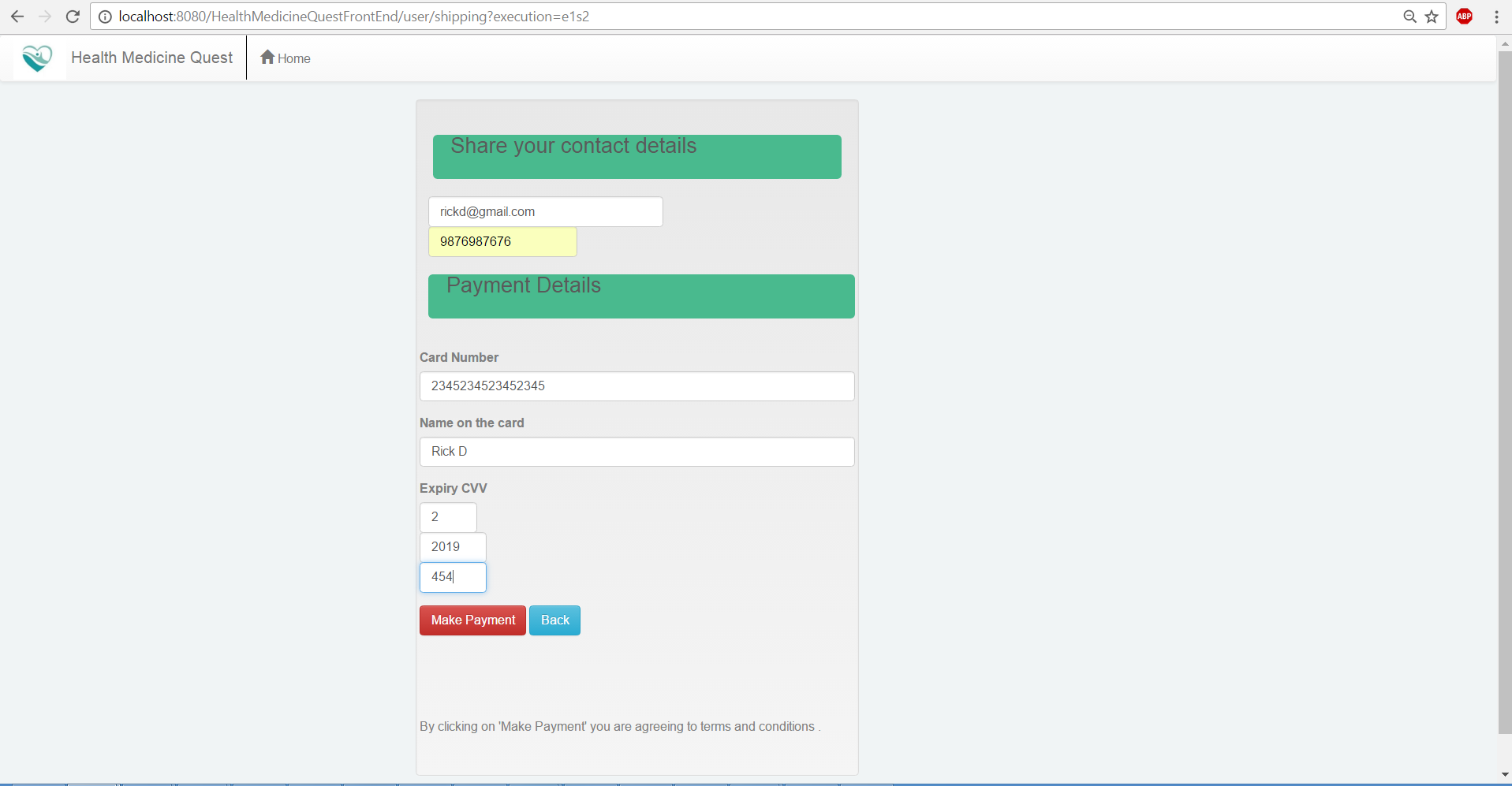
1. Shipping Address page.

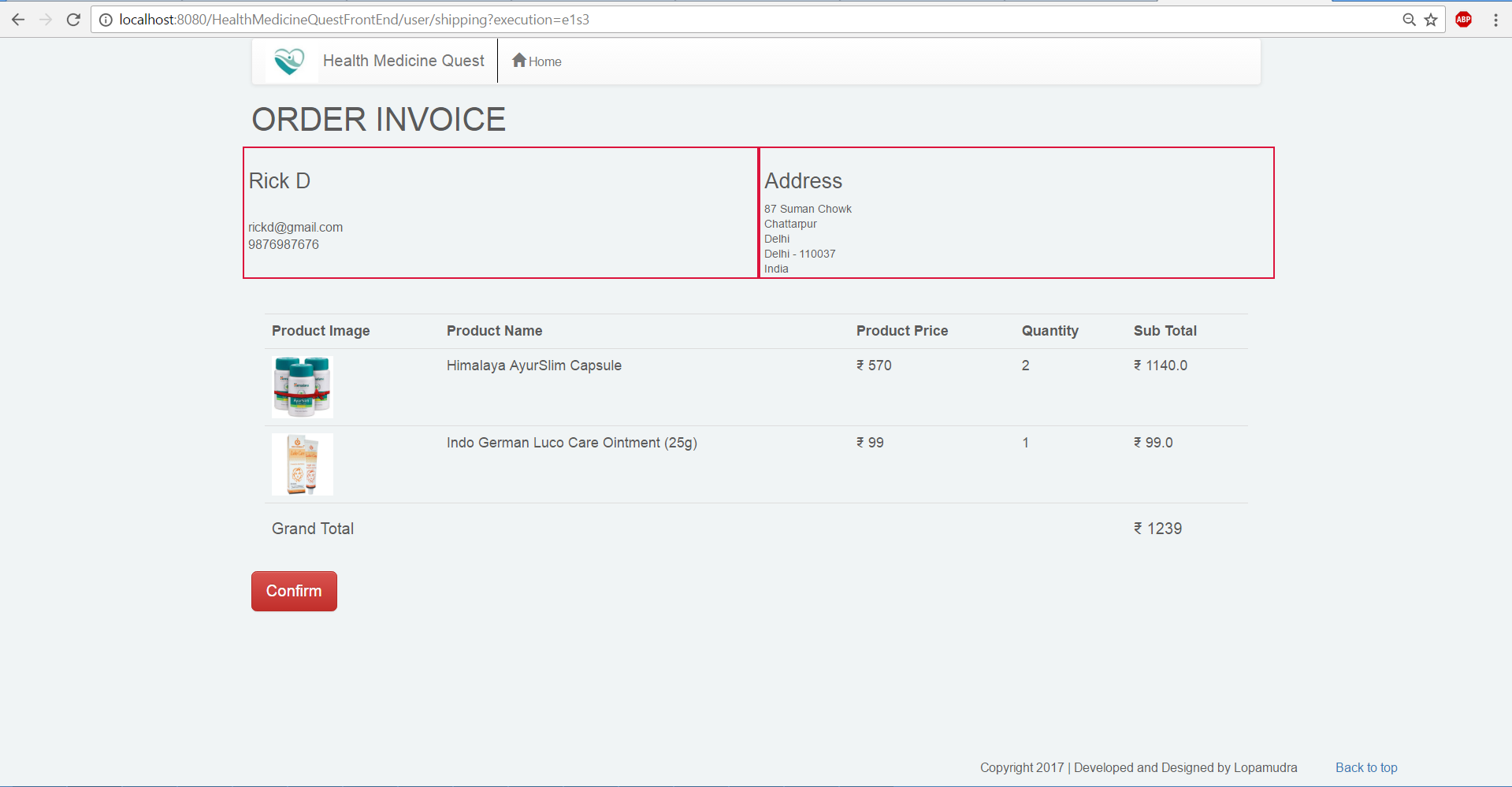
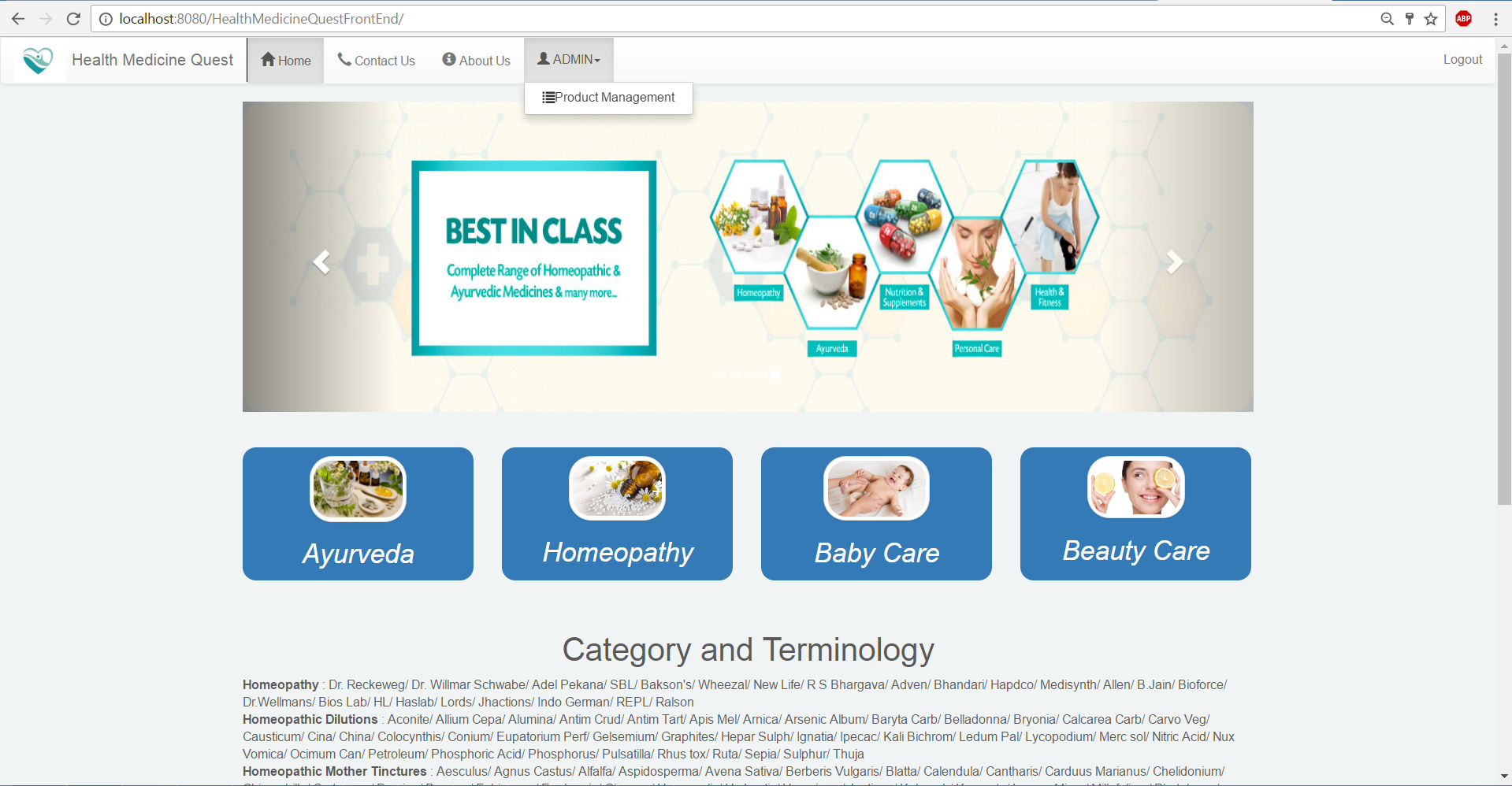


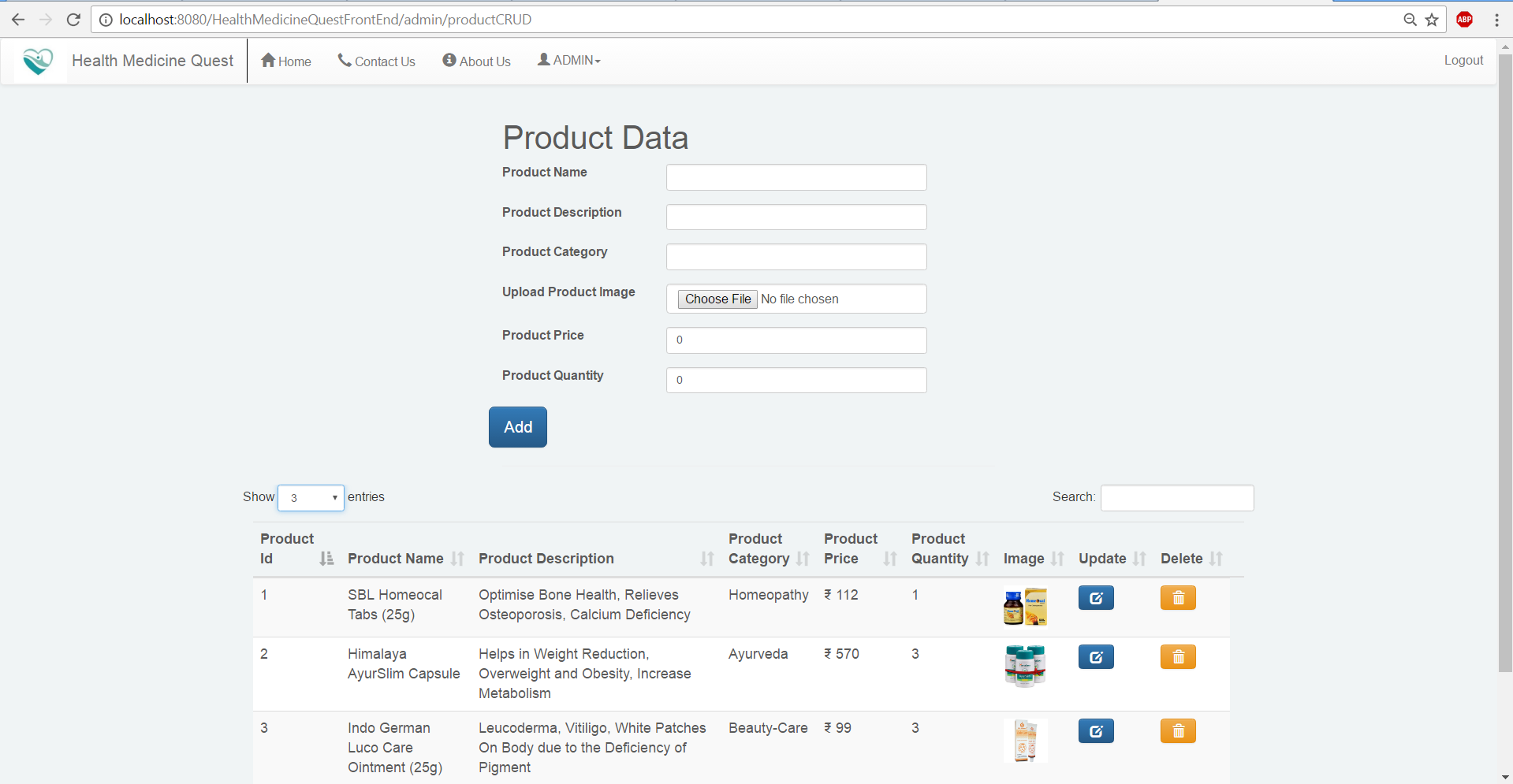
1. User can provide shipping address if shipping address is different from billing address. Or user can check the checkbox to opt out the shipping address option. In this case user’s billing and shipping both would be the same.
2. Cart flow is configured in cartFlow.xml and handler code is mentioned in CartHandler.java and CartModel.java file.
3. The whole cart flow comprises with billingCart.jsp, shippingCart.jsp, Payment.jsp and Invoice.jsp pages.
4. Cart flow is terminated by executing storeDetail() of CartHandler class. Which performs following set of actions.
5. Inserting shipping address to the database if billing address is not the same with shiiping.
6. Deleting CartItem detail to user specific and updating Cart.
7. Inserting payment detail to the Payment table.
8. End transition of this flow is to redirect to the home page.
9. Shipping Address Page with opt off different shipping address.



1. Payment Page to enter payment information.



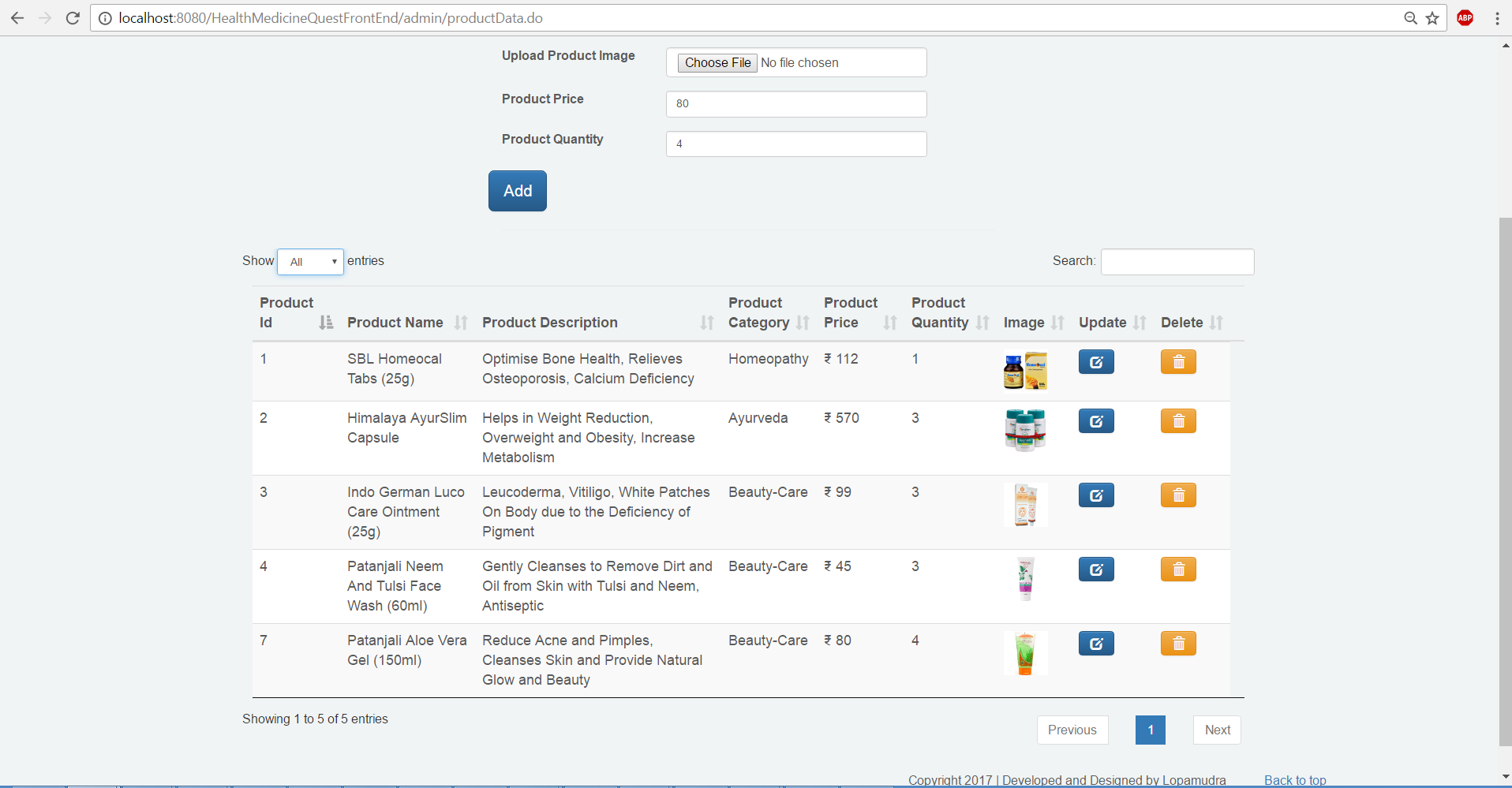
1. Final Order Invoice page. 
2. By clicking confirm buttom user can complete the complete cart flow.
3. Home page with Admin login. 
4. Administrator can click on Product management menu to manage the product like inserting, updating, deleting product records through interfaces.
5. ProductCRUD page to manage products by the admin user.



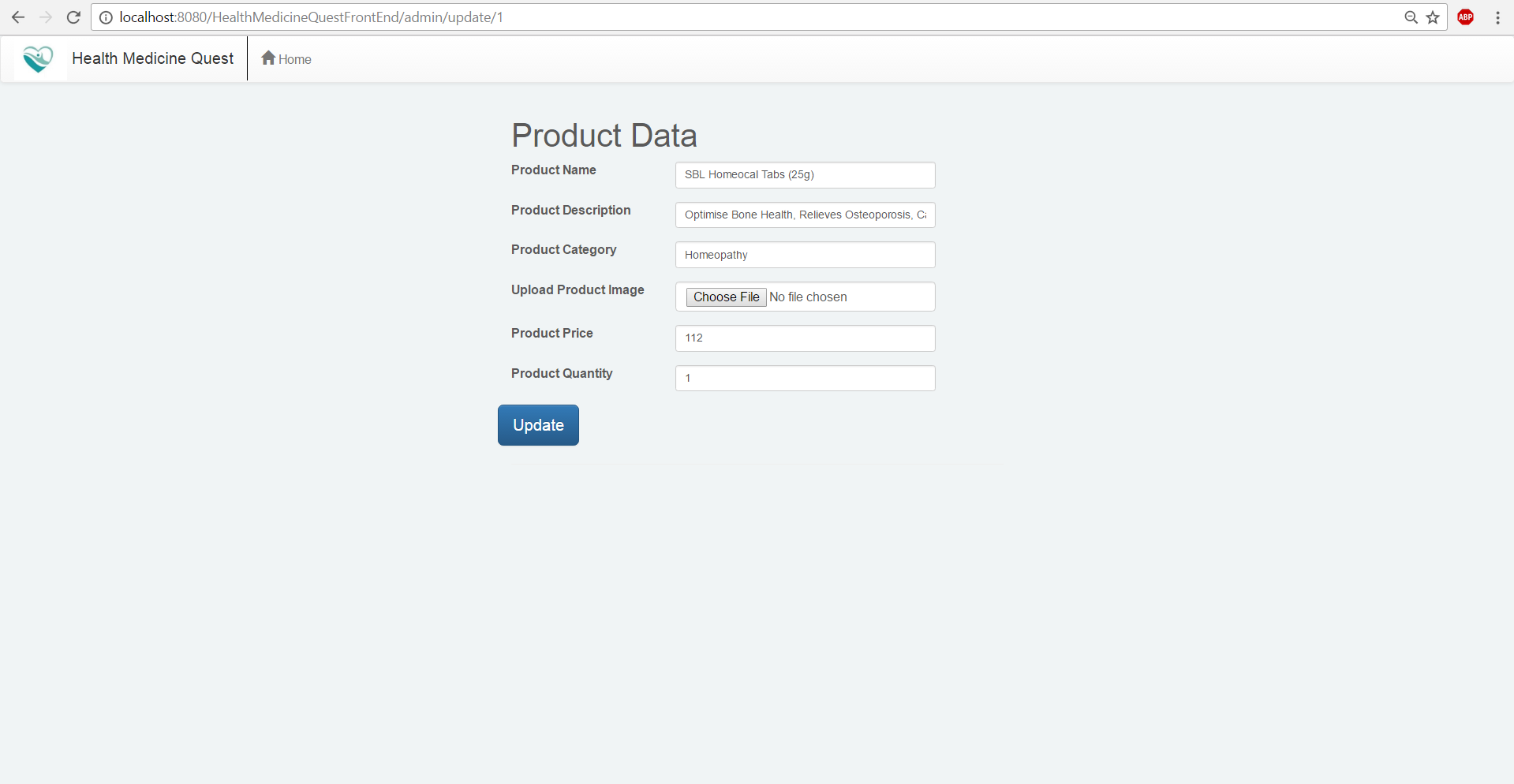
1. Spring form is used to design the data insertion form.
2. Jquery datatable is used to populated the existing products from the Product table.
3. Implemented Image Upload Functionality using Multipart File.
4. Spring-web, commons-fileupload, commons-io dependencies are used to implement this file upload feature into the application.
5. Product table is updated with a transient field of type MultipartFile.
6. Add product operation is combined with to images upload and store imagea at destination location using getOriginalFileName(), getRealPath() and transferTo() methods.
7. ProductCRUD page with image upload.



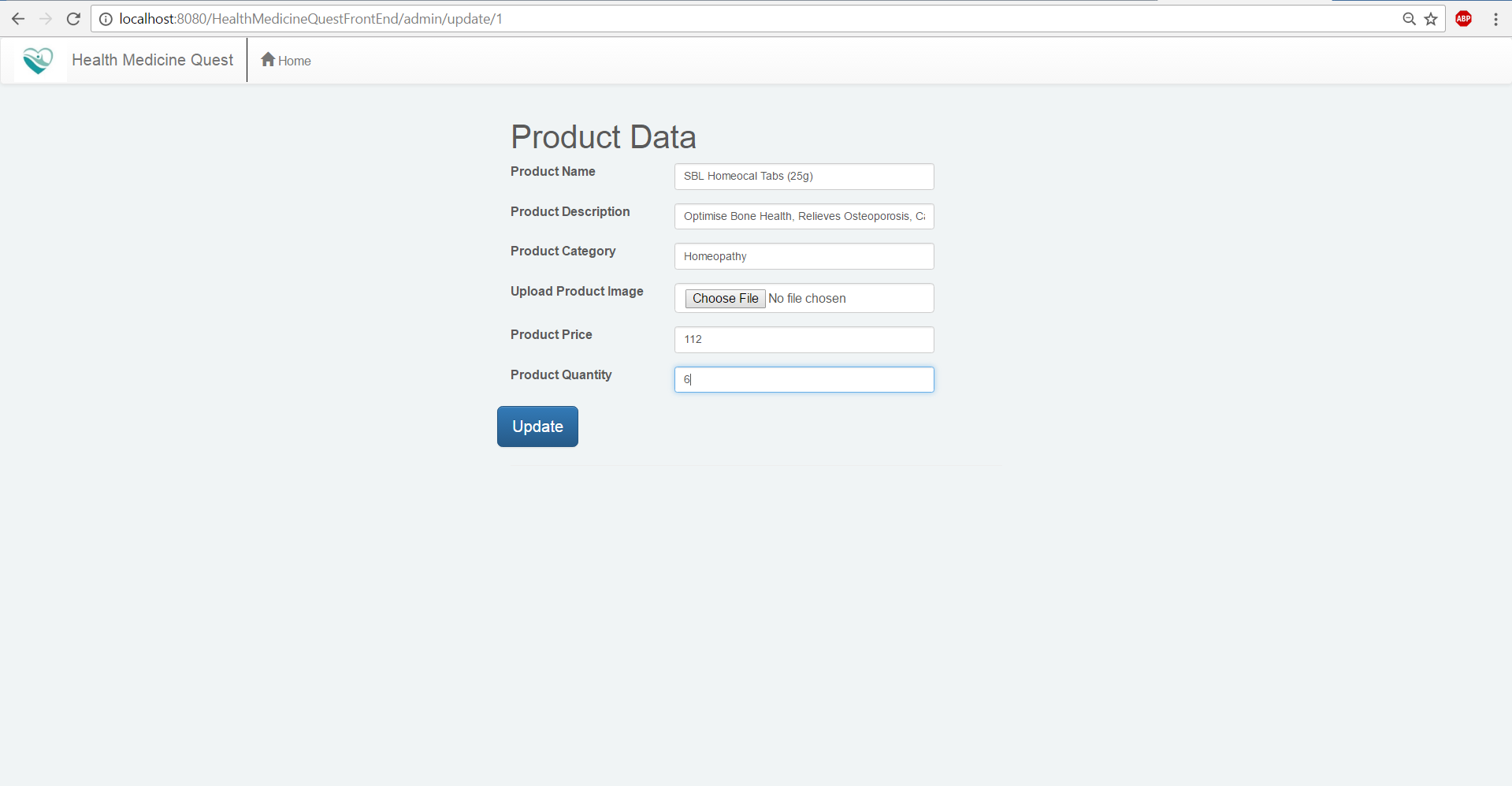
1. ProductCRUD page with newly inserted product.



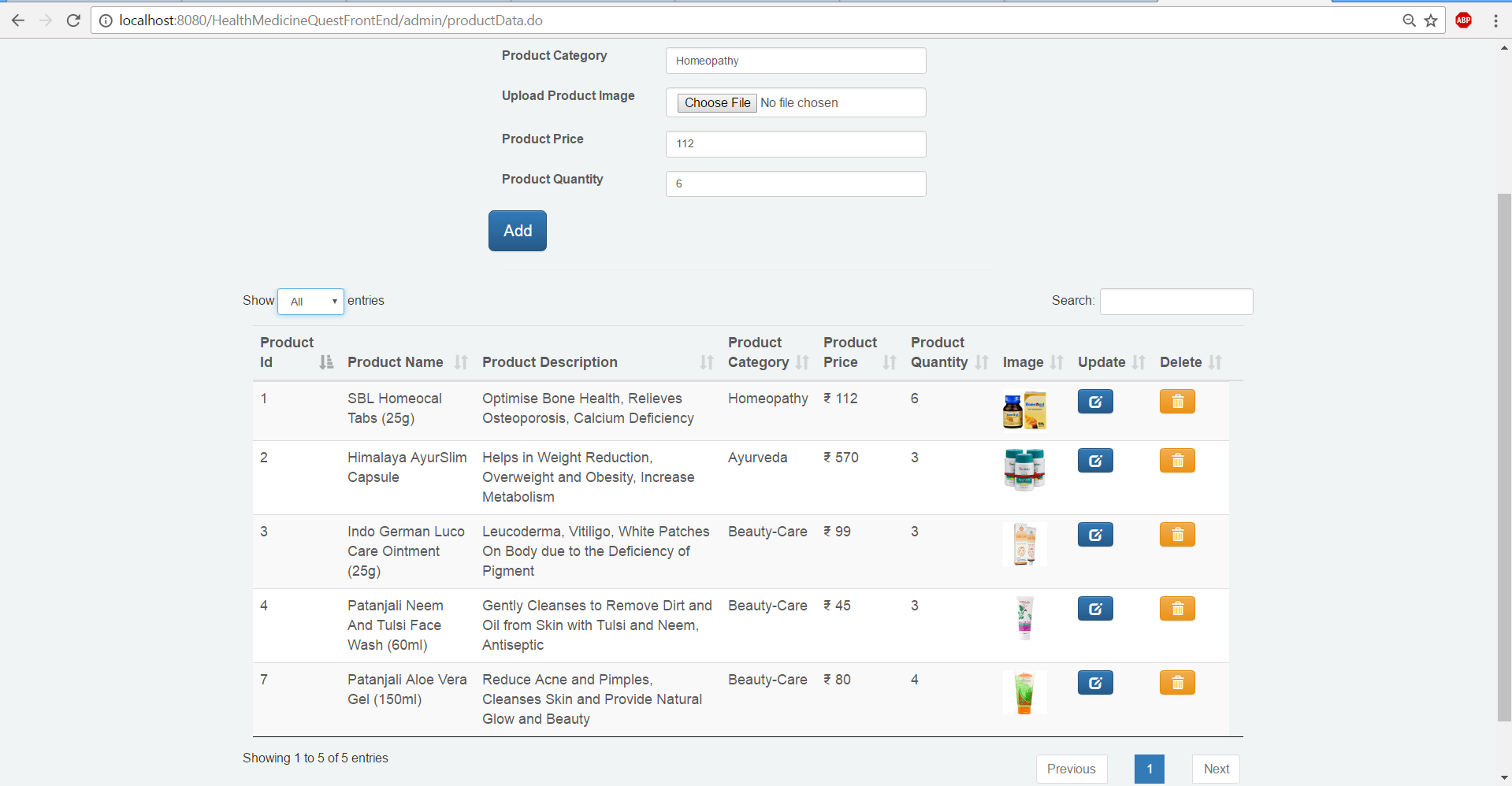
1. Each product row consists with update and delete button.
2. User can click update button to navigate ProductUpdate page.
3. User can click delete button to delete product.
4. ProductUpdate page with existing old data.



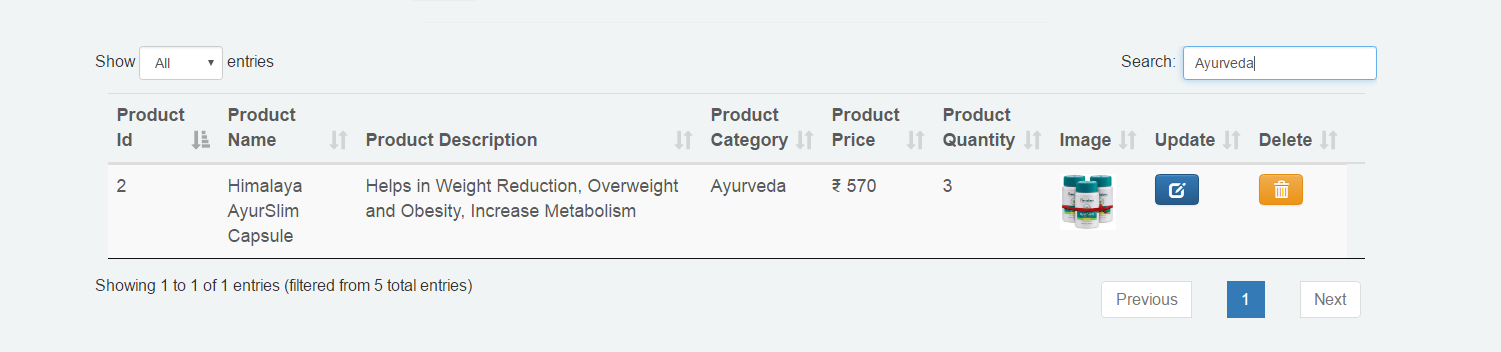
1. ProductUpdate page with modified data.



1. ProductCRUD page after data updation.



1. User can view category wise product with search option of datatable.



Conclusion

It was a wonderful and learning experience for me while working on this project. This project took me through the various phases of project development, implement known and unknown technologies and gave me real insight into the world of software engineering. The joy of working and thrill involved while tackling the various jargons and challenges gave me a feel of developers industry.

It was due to this project I came to know how professional software’s are designed and developed. I am confident enough to handle student’s query during their practices. I will every time make myself up to date to guide students.

I enjoyed each and every bit of work I had put into this project. This project is further extendable.

==============================================================================