PROJECT REPORT

**DEVELOPING A BACKEND ADMIN FOR LEARNER’S ACADEMY**

Submitted in partial fulfillment of the requirement for the award of degree

OF

**FULL STACK JAVA DEVELOPER**

**PHASE-3 SESSION (05.12.2020-20.12.2020**)

**UNDER THE GUIDENCE OF**

**MR. WALEED & MR. WAHEED**

**COURSE INSTRUCTOR (SIMPLILEARN)**

**SUBMITTED BY**

**MANISH KUMAR ARORA**



**AT STUDY CENTRE**

**Learning Management System**

**SIMPLILEARN**

**CONTENTS**

1. Introduction
   1. Project Details
   2. Developer Details
   3. Problem Description
2. Sprint Plan
   1. Number of Sprints
   2. Duration of Sprint
   3. User Stories
   4. Task achieved in each Sprint
3. Algorithm of application
4. Flow Charts of application
5. Implementation
   1. Software and Tools used
   2. Methodology
   3. Coding
6. Output Test - Output Screens
7. Core Concept used in the Project
8. Links to the GitHub repository to track and verify the project
9. Conclusion
   1. Summary
   2. Future Enhancements
   3. Unique Selling Points(USPs)

**INTRODUCTION**

1. **Project Detail:**

The project “**Make an E-commerce Website for Sporty Shoes**” aims to increase the scope of user’s interactions by developing dynamic web pages and menu driven options in order to perform operations on database tables where admin can login and change password, set up products and categories, browse the registered members, can see the purchase report. Apart from that new user can sign up , and registered members can select the product and add to their carts, and checkout with payment gateway to buy the product, can see his order and edit his profile details if he wish to.

1. **Developer Detail:**

Manish Kumar Arora

Technical Support Associate, Engineering

MBA-IT, B.TECH-CSE

1. **Problem Description:**

Prototype of this project must include Java Concept like web pages(JSP), Java framework spring mvc with hibernate to build web application, database and SQL technology which is used to develop the application. Also, includes the Maven dependency injections, HQL Queries for source code optimization and increased performance and dynamic creation of tables in database. This application must have following basic features:

1. Admin Login
2. Manage the products
3. Categorizing the products
4. Browse the list of users
5. See purchase reports
6. Change passwords
7. New member can sign up
8. Registered members can choose the product and add to cart
9. Checkout via payment gateway
10. Edit his/her profiles
11. Check list of orders made.

**SPRINT PLAN**

1. **Number of Sprints: 2**
2. **Duration of each sprint: One Week**
3. **User Stories: 5 user stories.**

Sprint1: 15.12.2020 – 21.12.2020

User Story1:

**Title: Home Page**

**Priority: 1**

**Estimate: 1 D**

**As an** user(admin or member)

**I want to** have initial home page

**So that I can** login into dashboard securely according to role.

User Story2:

**Title: ADMIN LOGIN**

**Priority: 2**

**Estimate: 2 D**

**As an** Administrator

**I want to** have initial Login Page

**So that I can** login into dashboard securely.

User Story3:

**Title: Admin Dashboard**

**Priority: 3**

**Estimate: 4 D**

**As an** Administrator

**I want to** have all different tasks options to set up product, categories, see purchase reports and browse users and change password

**So that I can** choose between the multiple tasks to perform operations of adding new products and categories, edit or delete the existing products, browse the registered members and see purchase reports according to date.

Sprint2: 22.12.2020 – 27.12.2020

User Story4:

**Title: Member Dashboard**

**Priority: 1**

**Estimate: 4 D**

**As an** Member

**I want to** choose different products

**So that I can** add products into my cart and do the purchase.

User Story5:

**Title: Member Profile**

**Priority: 2**

**Estimate: 1 D**

**As an** Member

**I want to** edit my profile

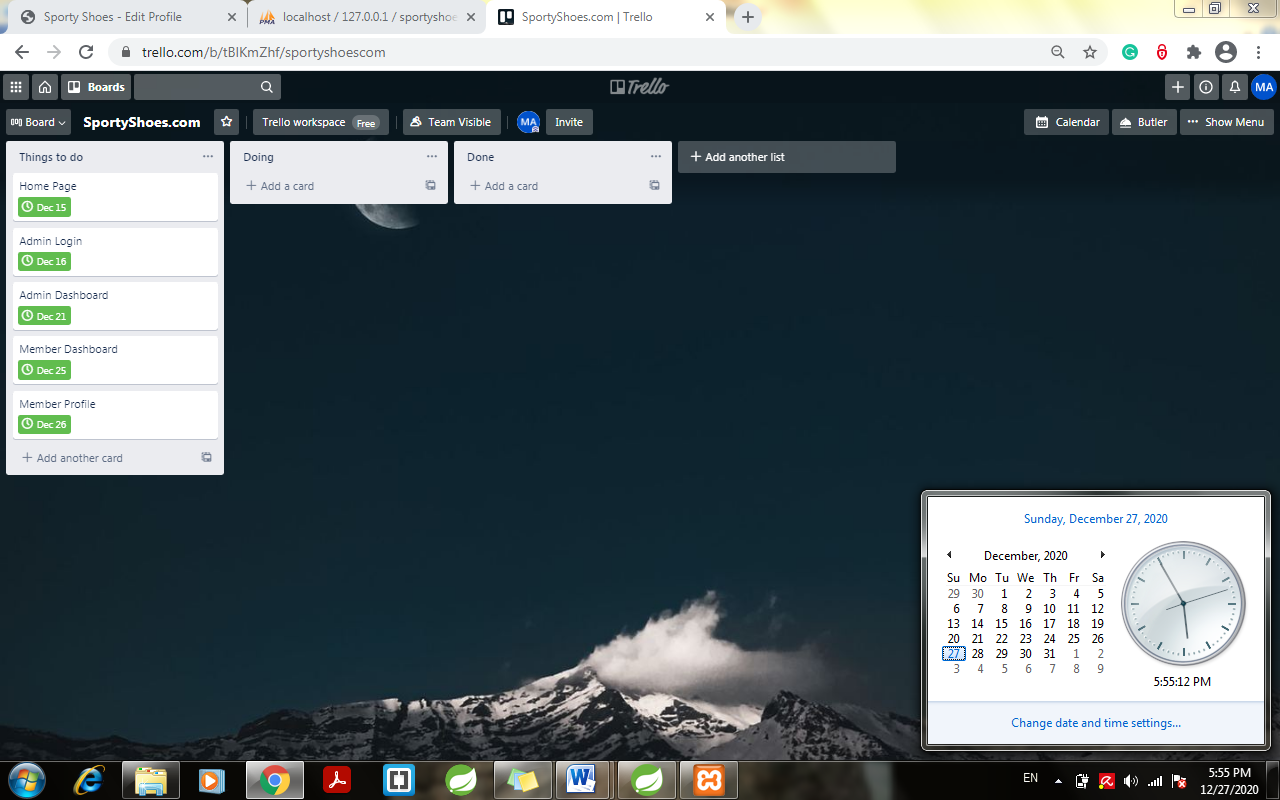
**So that I can** change password or my delivery address details and name.

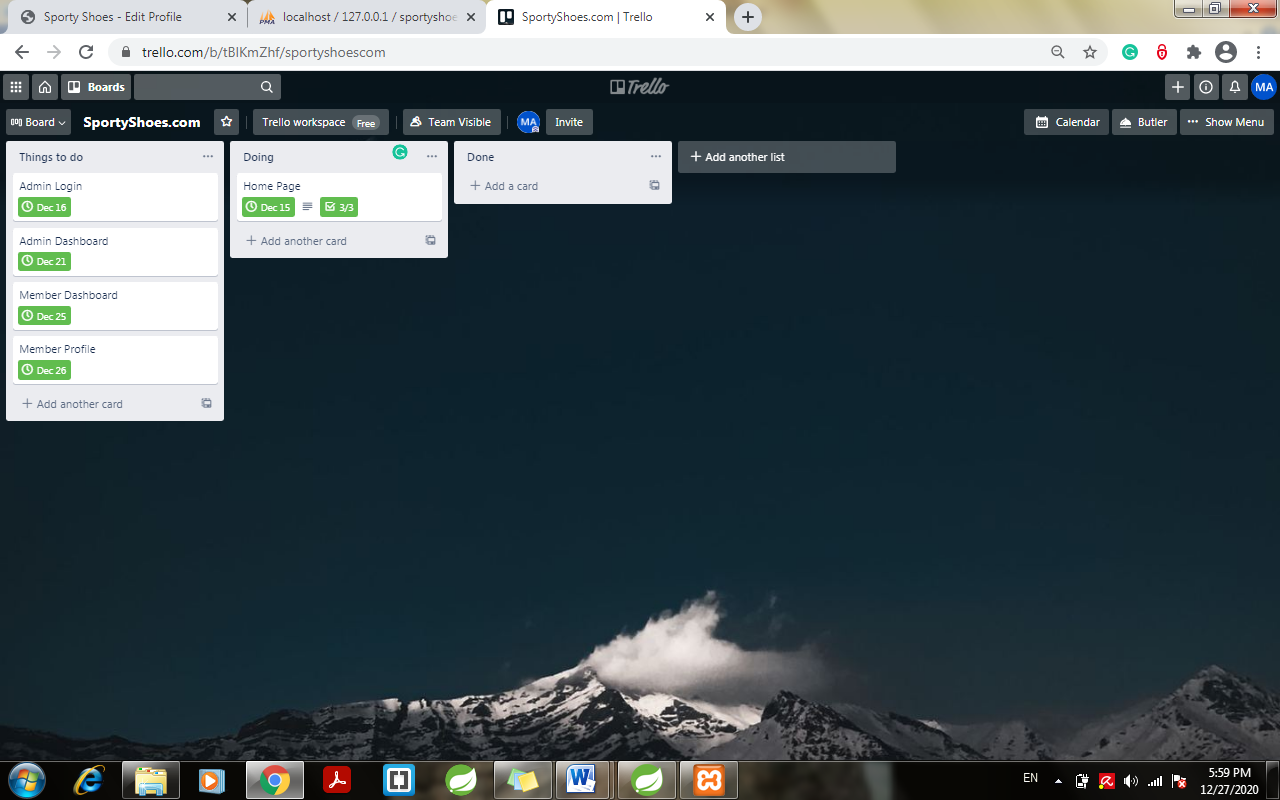
1. **Task achieved in each sprint:**
2. **Sprint One – Developed interactive home page, where admin and user can login to their respective dashboards, web pages for admin task i.e. set up products , categories, search registered members and see the purchase reports and created database “SportShoes” in with xampp. After that, developed develop business logic and uses HQL queries to dynamically create a tables in the database for admin and users operations and uses java DAO classes and beans to create database connections.**
3. **Sprint Two – Developed pages Members dashboard to choose products entered by admin and displayed on users home page, develop logic to add the products to the cart and checkout using payment gateway for purchasing the selected product.**

**Develop logic to see the orders made and edit the profile and change password if user wish to.**

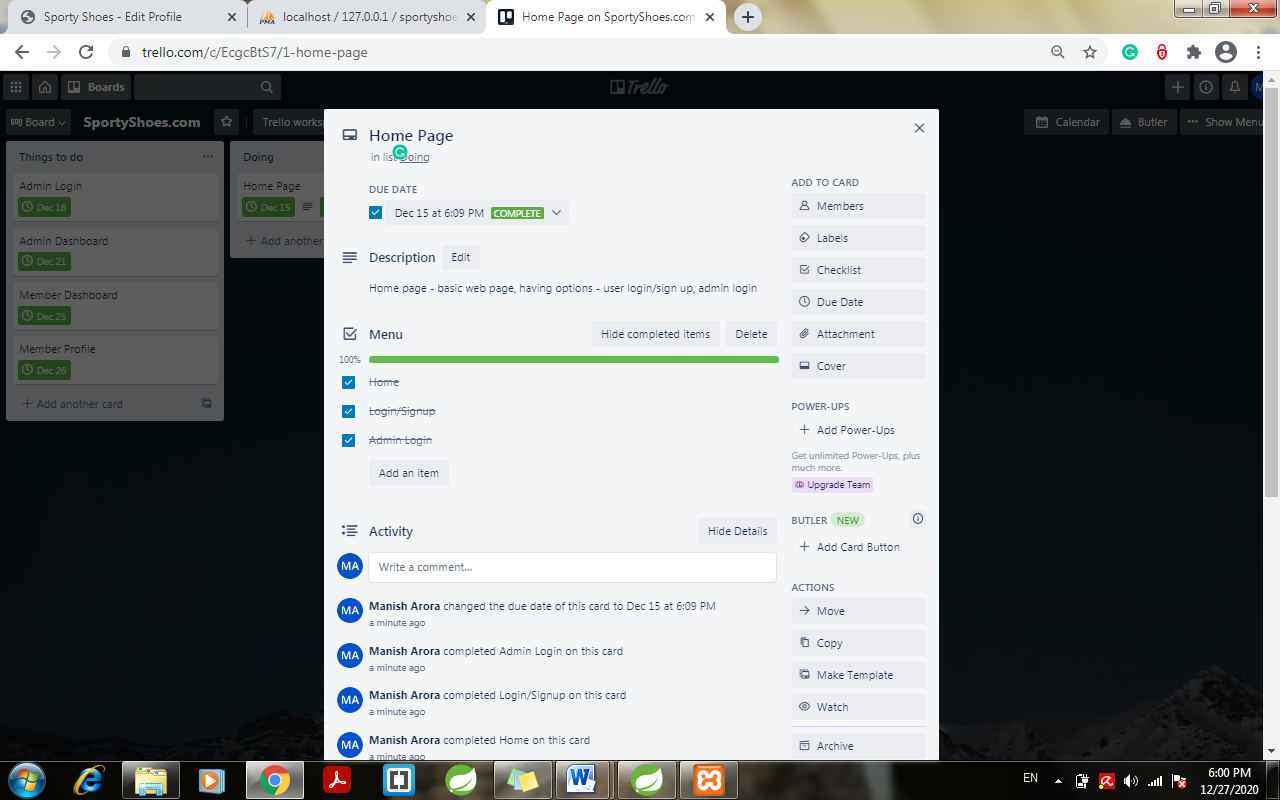
**Detailed Screenshots for each user stories used in scrum planning in each sprint:**

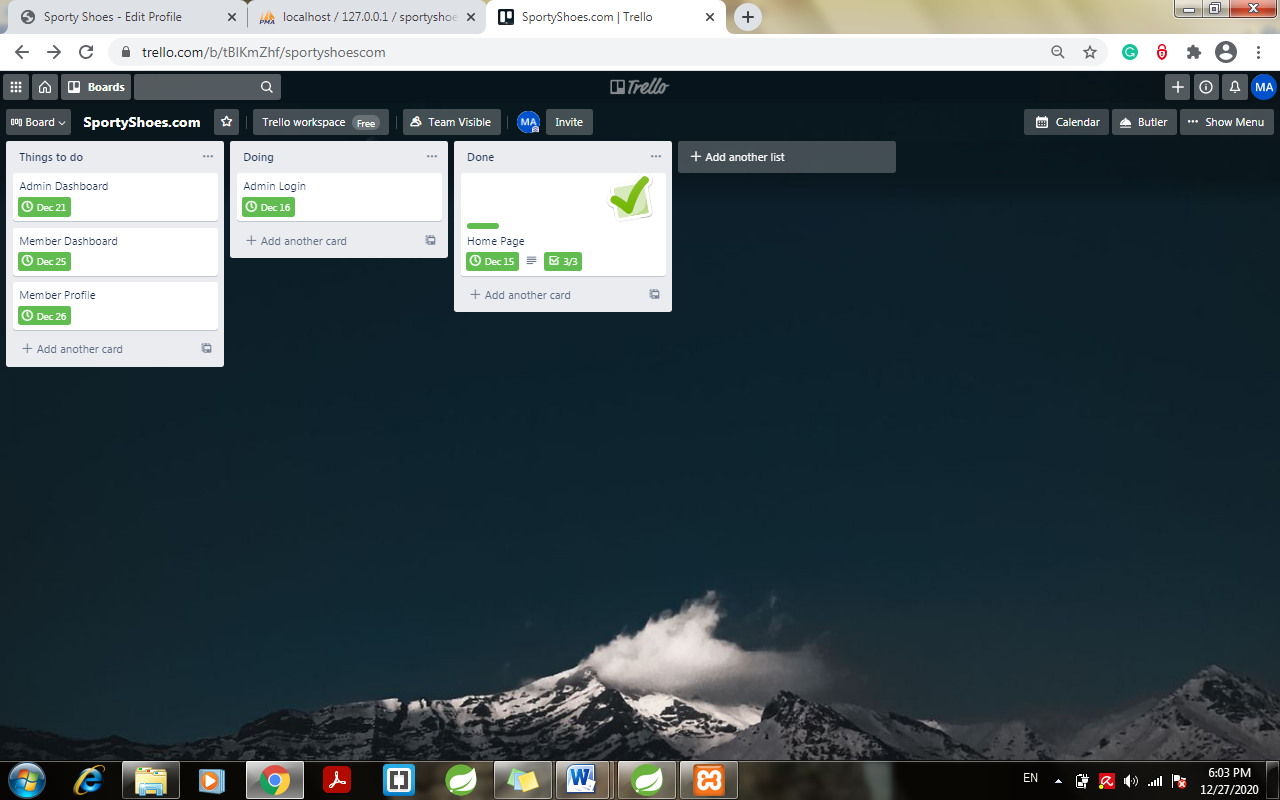
**Sprint 1: Date: 15.12.2020 – 21.12.2020**



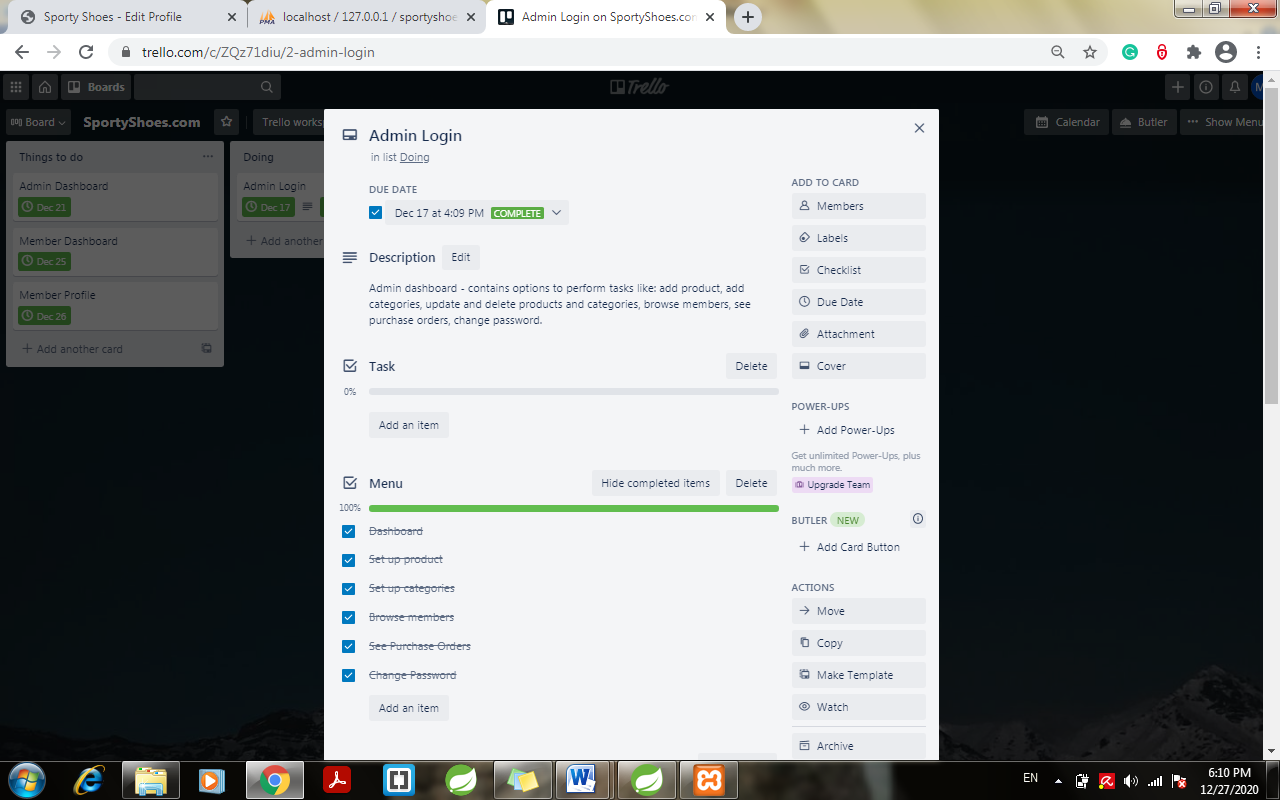


**User Story 1 Completed: 15.12.2020**

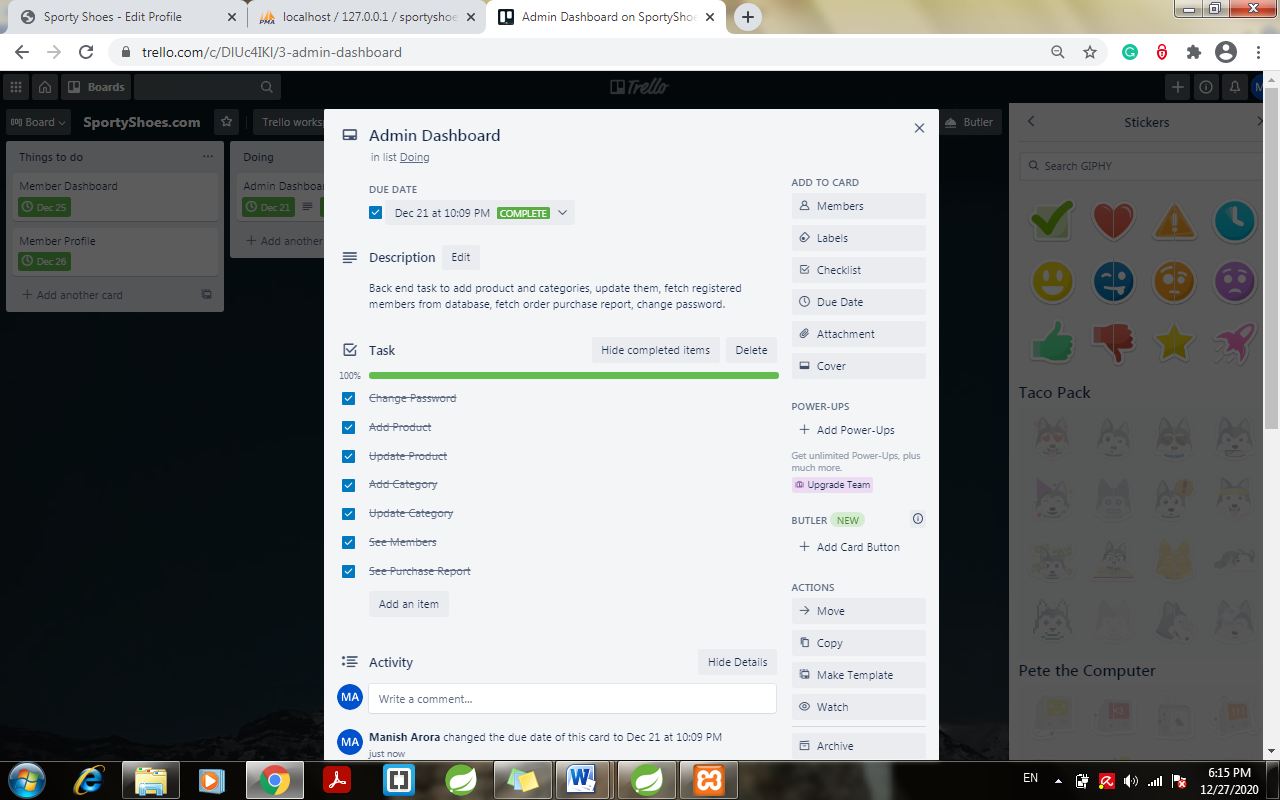




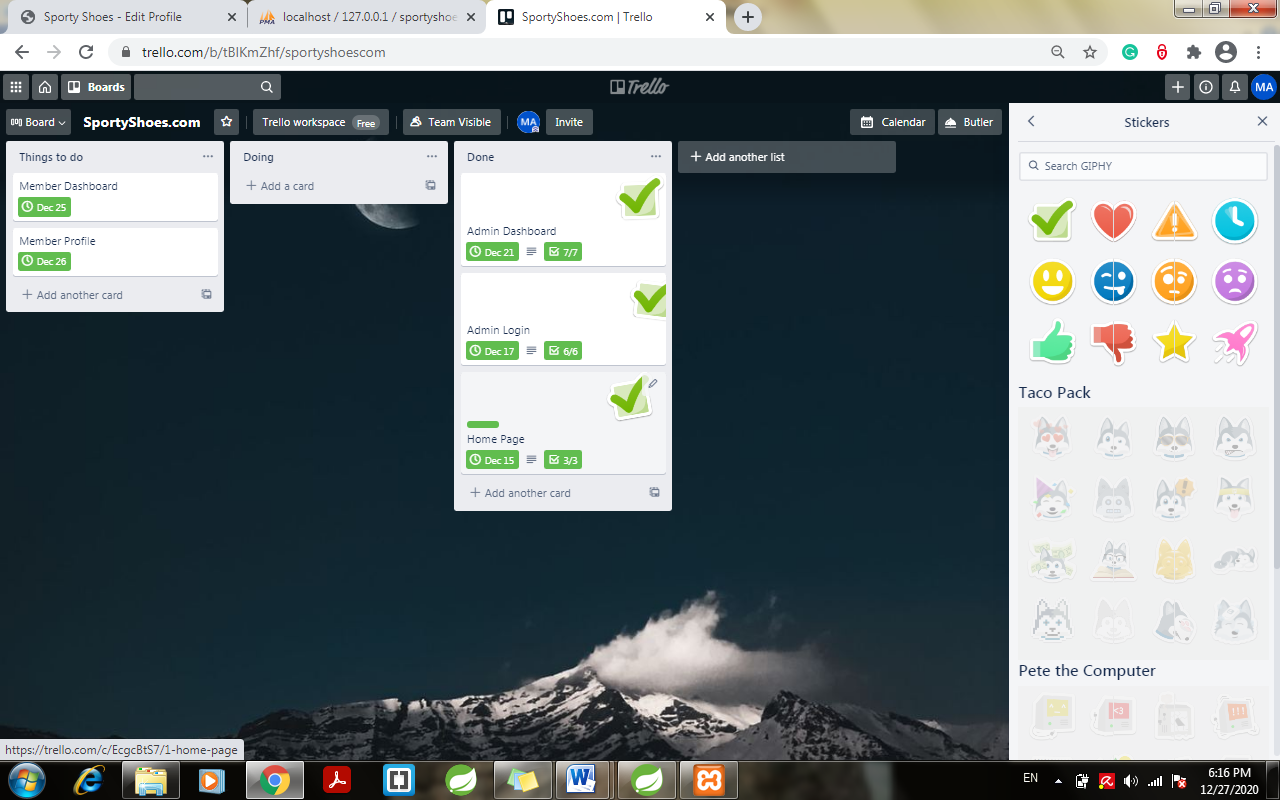
**User Story 2 Completed: 17.12.2020**



**User Story 3 Completed – 21.12.2020**

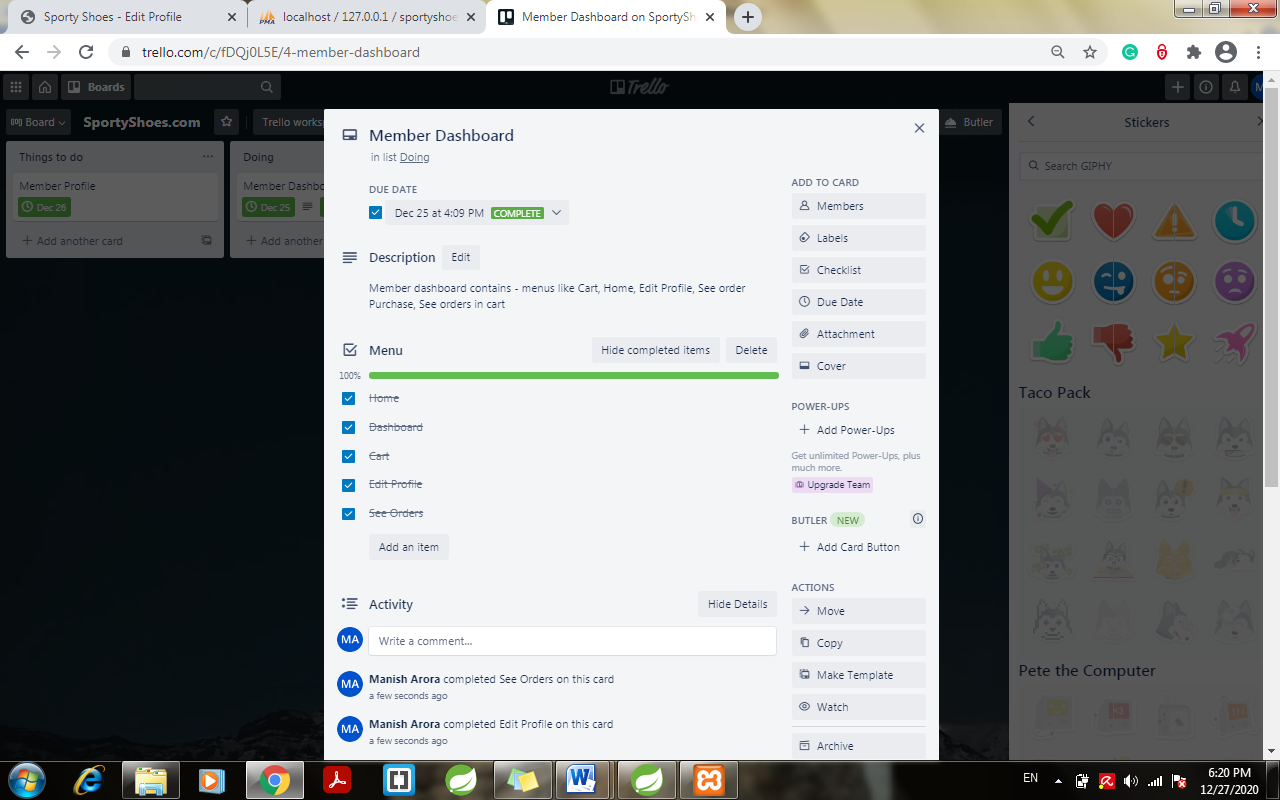


Sprint 1– Completed Date: 21.12.2020

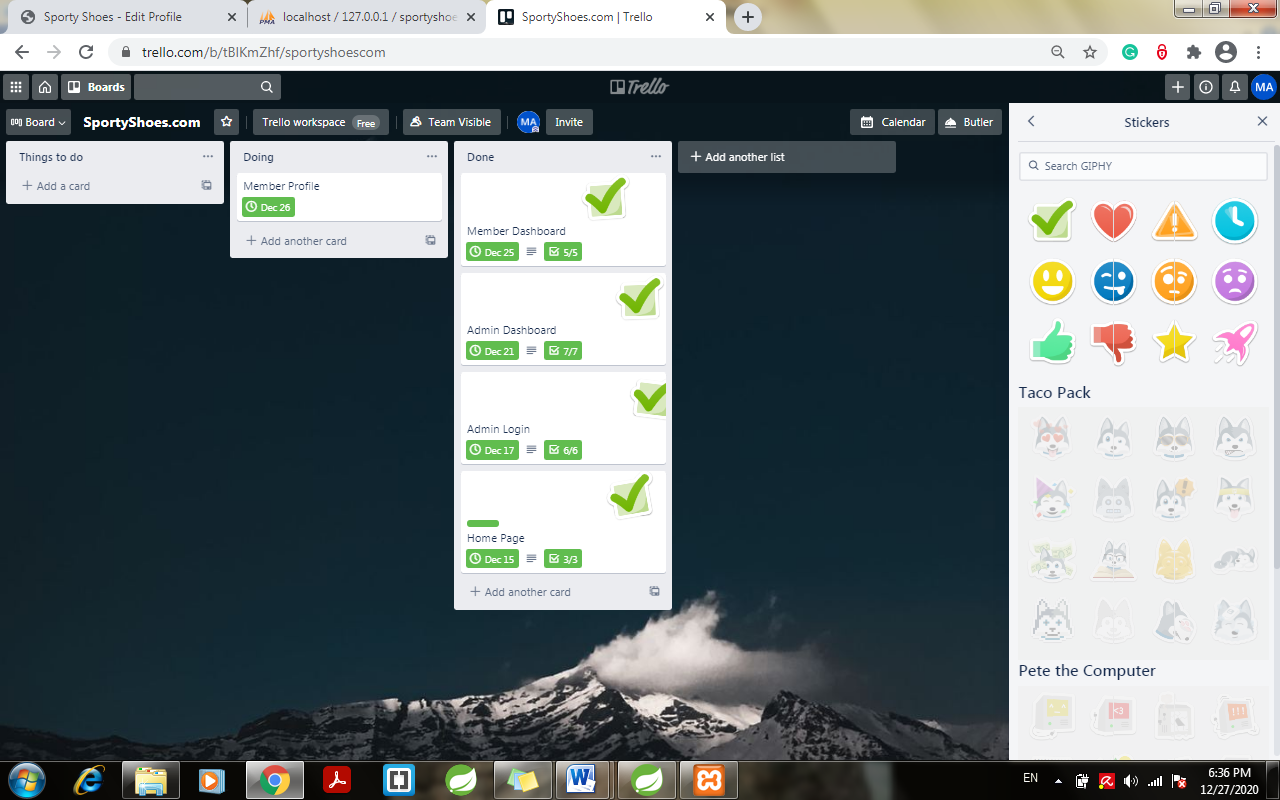


**Sprint 2: Date: 22.12.2020 – 27.12.2020**

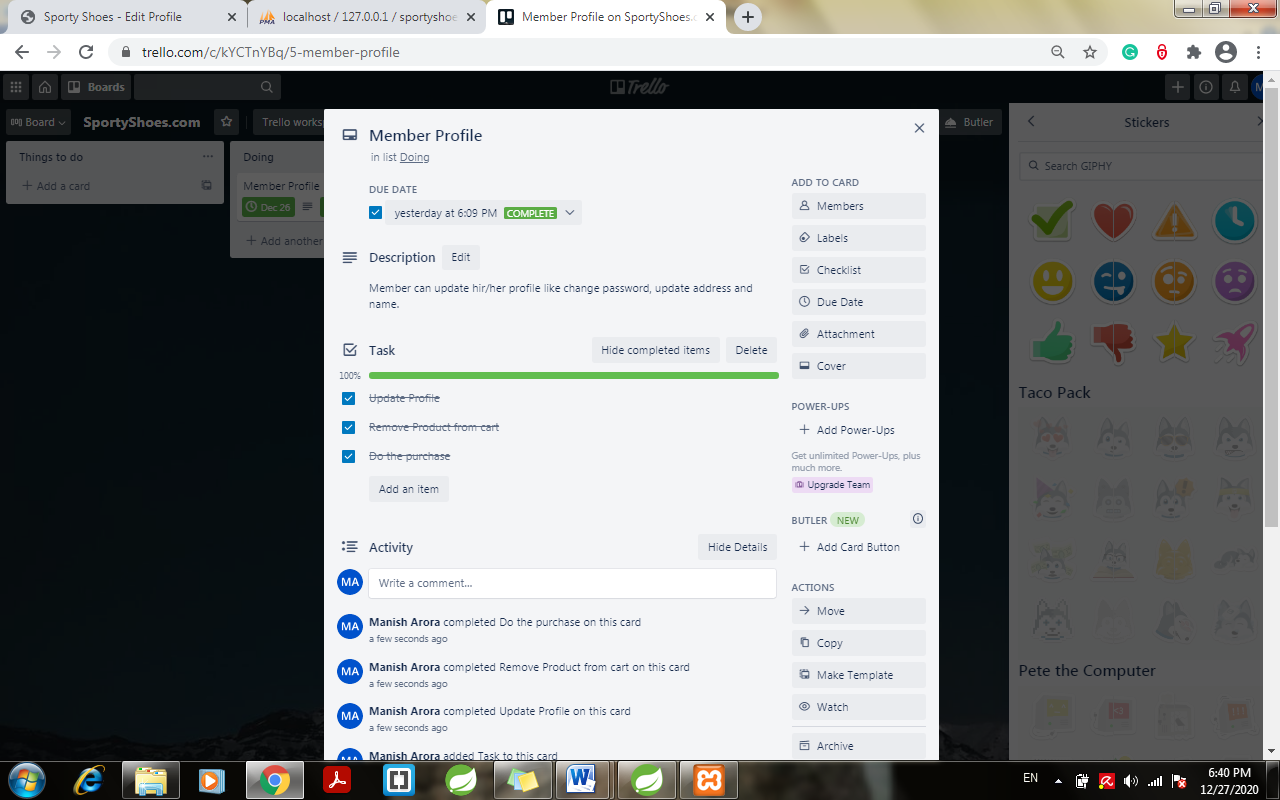
**User Story 4**



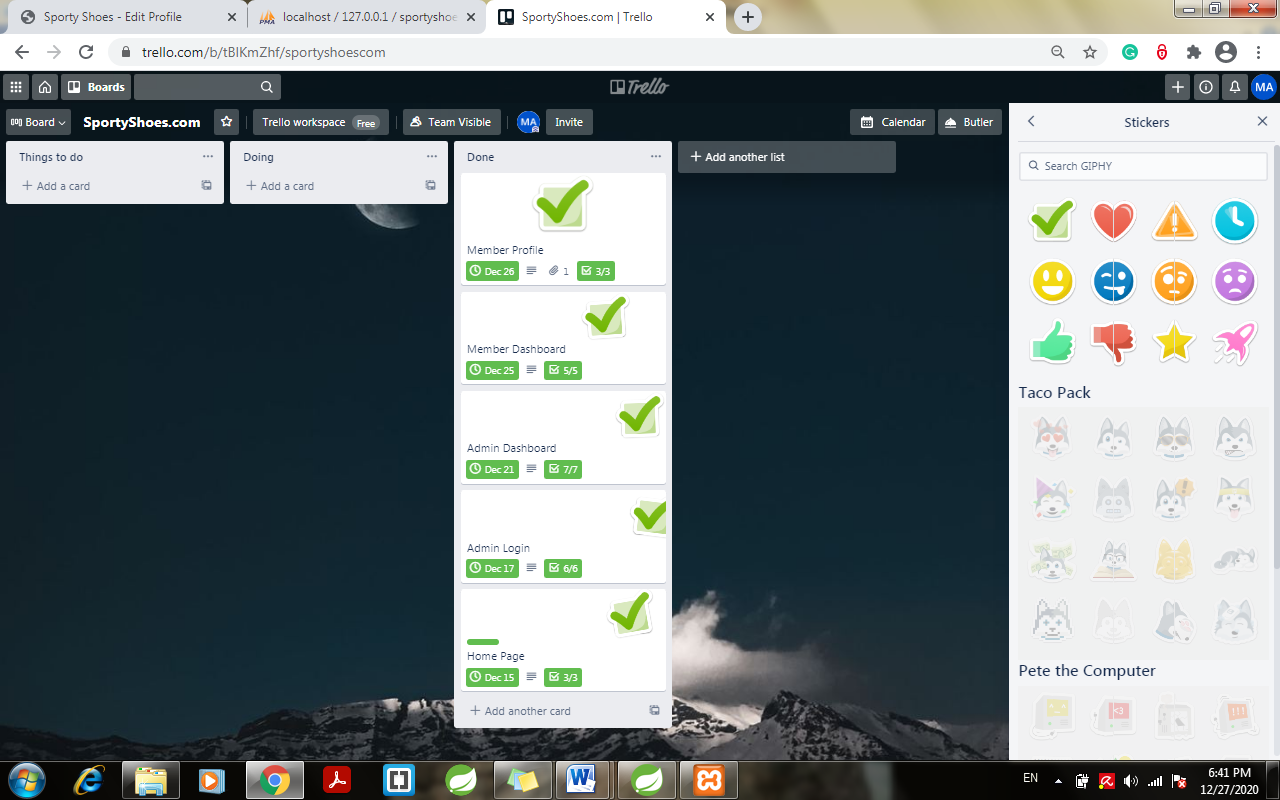
User Story 4 Completed – 25.12.2020



**User Story 5 Completed – 26.12.2020**



Sprint 2– Completed Date: 26.12.2020



**ALGORITHM OF APPLICATION**

**This application is developed using MVC architecture. The database and interactions done using DAO classes.**

**Application flow:**

**SportyShoes Home Page**

1. **User interacts dynamically with web pages (JSP Pages)**
2. **Home Page - > click on Login/Signup**
3. **Request sent with parameters to servlet-> MemberController.java**
4. **Return login.jsp**
5. **Enter user name password :: goto 3.**
6. **Authenticate with service class-> UserService.java**
7. **UserService.java interacts with UserDao.java**
8. **UserDao: list<- Createquery(hibernate)**
9. **If List not Empty :: return user Home Page**
10. **If empty :: “user not exist” :: goto :: 2 or Signup**
11. **If Signup -> sent to servlet “signup action ”-> goto :: 3**
12. **Return -> register.jsp**
13. **Enter details :: goto 3**
14. **Check if mail already exist with UserService.java**
15. **Else updateuser –>UserService.java**
16. **User Service interacts with UserDao.java**
17. **Goto::8**
18. **Return “registerconfirm.jsp”**

**User Dashboard**

1. **Add Product -> Cart -> request sent to servlet “cart” -> CartController.java**
2. **Return cart.jsp**
3. **IF Remove -> request sent to servlet – “cartdeleteitem”**
4. **Return cart.jsp**
5. **If checknow - > request sent to servlet “checkout”**
6. **Return checkout.jsp**
7. **If Pay - > servlet “gateway”**
8. **Return gateway.jsp**
9. **If Complete - > servlet “confirm”**
10. **Return confirm.jsp**
11. **Your Order - > request sent to servlet “memberpurchases” - > PurchaseController.java**
12. **List <- getallitemsbyuserid using PurchaseService.java**
13. **PurchaseService.java interacts with PurchaseDao.java**
14. **List <- createquery using hibernate.**
15. **Return purchase list**
16. **Return purchase.jsp**
17. **IF “Edit Profile” - > request sent to servlet “editprofileaction” -> MemberController.java**
18. **Details set by entity class - > user.java**
19. **User - > updateuser - > UserService.java**
20. **UserService.java interacts with UserDao**
21. **Sql = update query using HQL**
22. **Return to dashboard.**
23. **If Logout -> request sent to servlet “logout”**
24. **Return home page.**

**Admin Login**

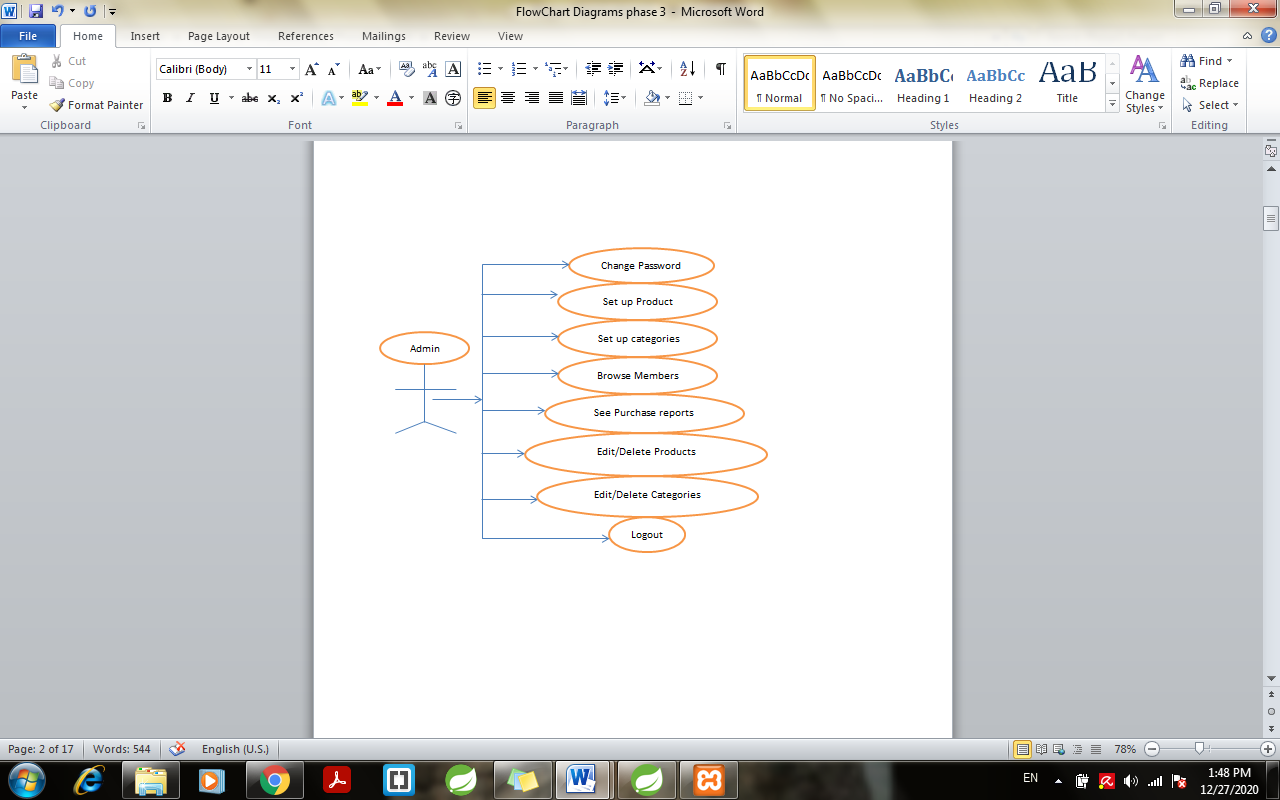
1. **Admin login -> enter user name and password - > request sent to servlet “adminloginaction” - > AdminController.java**
2. **Authenticate with service class-> AdminService.java**
3. **AdminService interacts with AdminDAO**
4. **AdminDao :: list <- createquery using HQL**
5. **Return Admin if exist - > AdminController**
6. **Return admin dashboard**
7. **If Admin Not exit -> “wrong Credentials.”**

**Admin Dashboard**

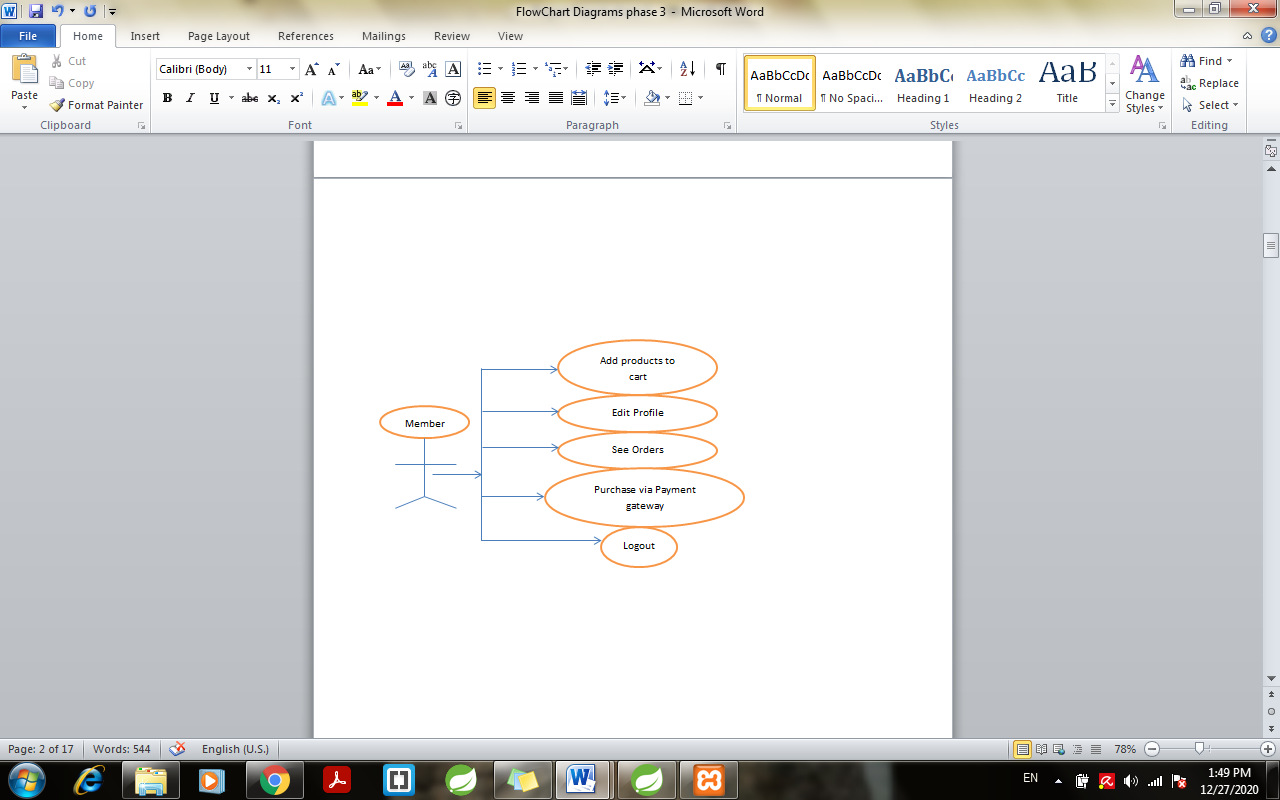
1. **Set Up product category - > request sent to servlet “admincategories” -> AdminController.java**
2. **List <- get all categories using CategoryService.java**
3. **CategoryService interacts with CategoryDao**
4. **CategoryDao interacts with db and returns Category**
5. **Return categories.jsp**
6. **If ADD CATEGORY**
7. **REQUEST sent to “admineditcat” -> AdminController.java**
8. **Return -> edit-category.jsp**
9. **Add category name - > parameter sent to -> SERVLET “admineditcataction”**
10. **RETURN - > edit-categories.jsp**
11. **If edit -> goto::7**
12. **If Delete -> request sent -> servlet “admindeletecat”**
13. **Return :: goto 10**
14. **IF setup products -> servlet “adminproducts”**
15. **Return products.jsp**
16. **If add products -> servlet “admineditproduct”**
17. **Add product name , price, category**
18. **Getproductby id using ProductService.java**
19. **ProductService interacts with ProductDao**
20. **ProductDao update the records using HQL**
21. **goto::15**
22. **If DELETE product - > servlet -> “admincategories”**
23. **Runs -> deleteproduct -> call-> ProductService.java**
24. **Goto:: 20**
25. **Returns 15.**
26. **If edit -> servlet “admineditproduct”**
27. **Return 15.**
28. **Browse Members -> servlet “adminmembers” -> AdminController.java**
29. **Getallusers -> UserService.java**
30. **Userservice.java interacts with UserDao to fetch data from DB using HQL**
31. **RETURN list**
32. **Return members.jsp**
33. **Purchase Report - > servlet “adminpurchases” -> AdminController.java**
34. **Get all items -> PurchaseService.java**
35. **PurchaseService.java interacts with PurchaseDao to fetch data from DB using HQL**
36. **Return list**
37. **Return- > admin/purchases.jsp**
38. **Change PASSWPRD -> SERVLET “adminchangepassword”**
39. **Return change-password.jsp**
40. **Enter new password - > servlet ->”adminchangepwdaction”**
41. **Run -> updatepwd from AdminService.java**
42. **AdminService interacts with AdminDao to update the records in to database**
43. **Return admin/login.jsp**
44. **If Logout -> return admin/login.jsp**

**FLOWCHART OF APPLICATION**

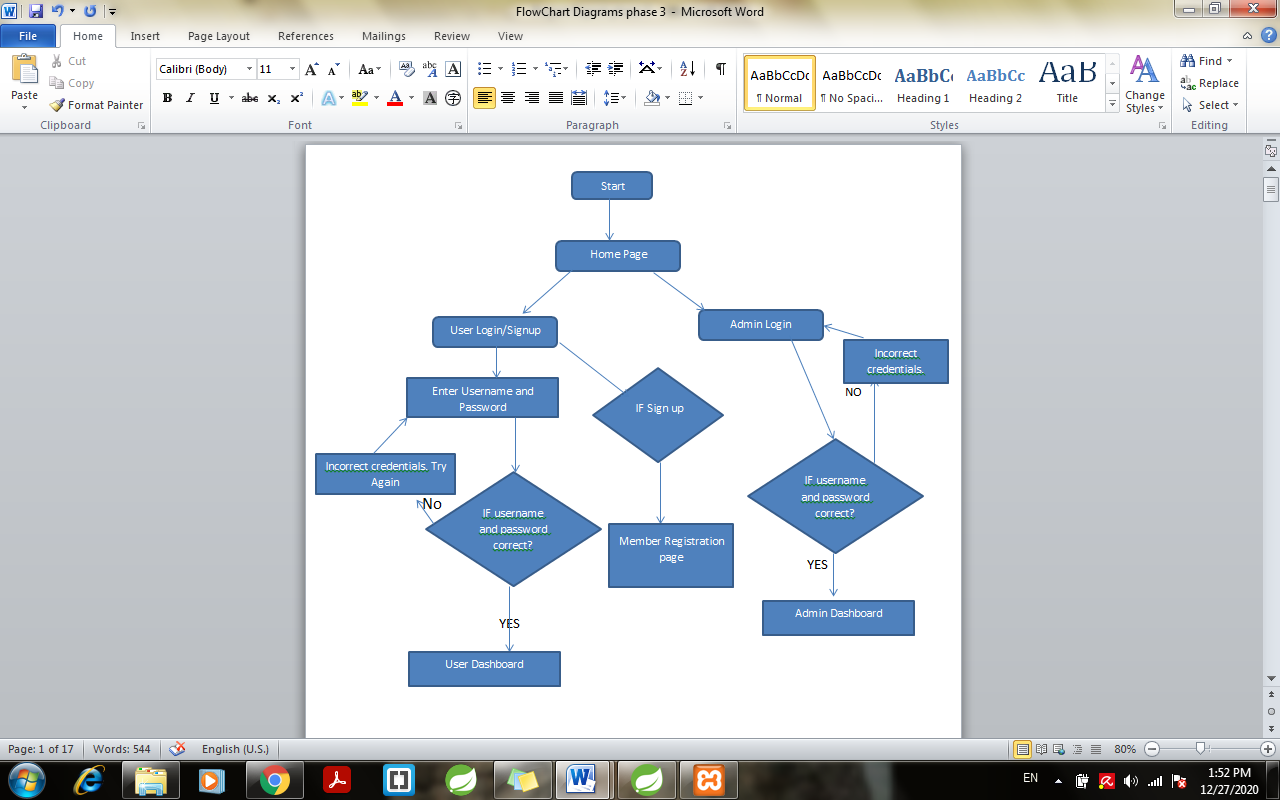
**USE CASE DIAGRAM 1**



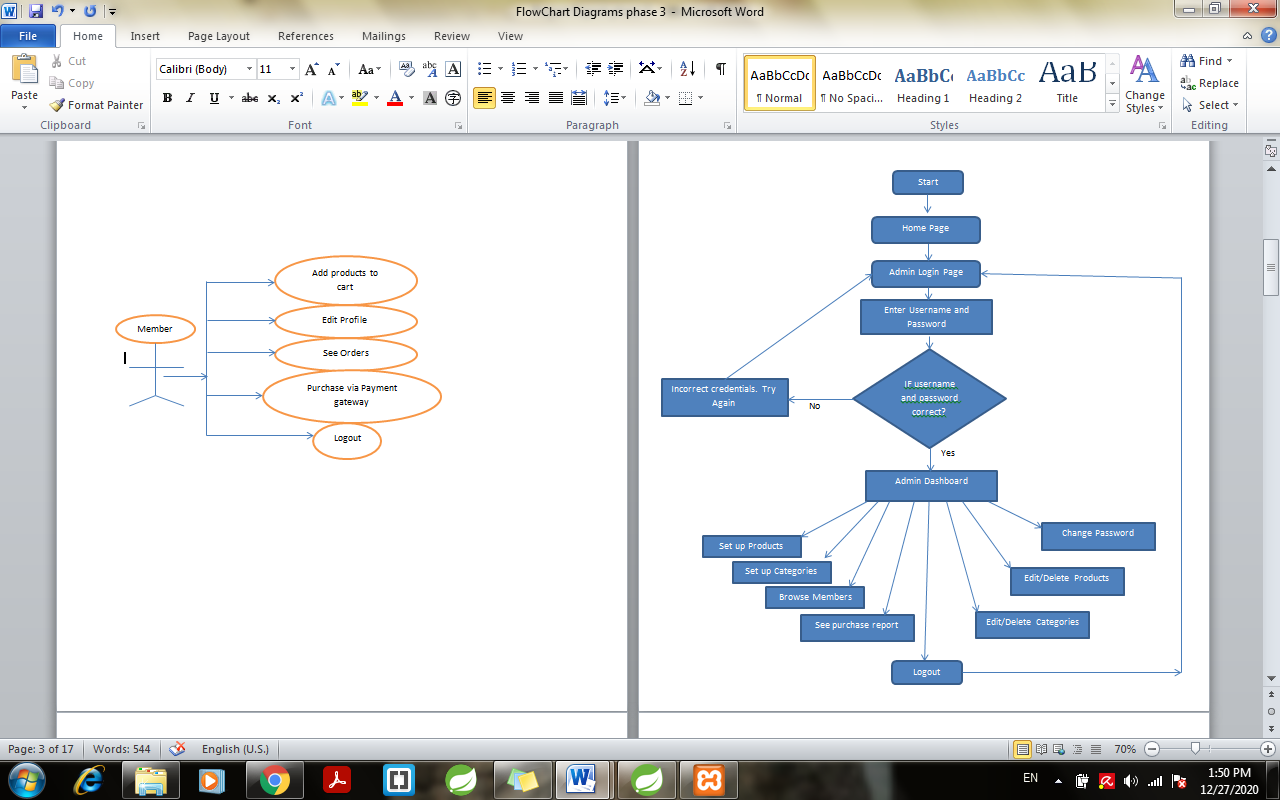
**USE CASE DIAGRAM 2**



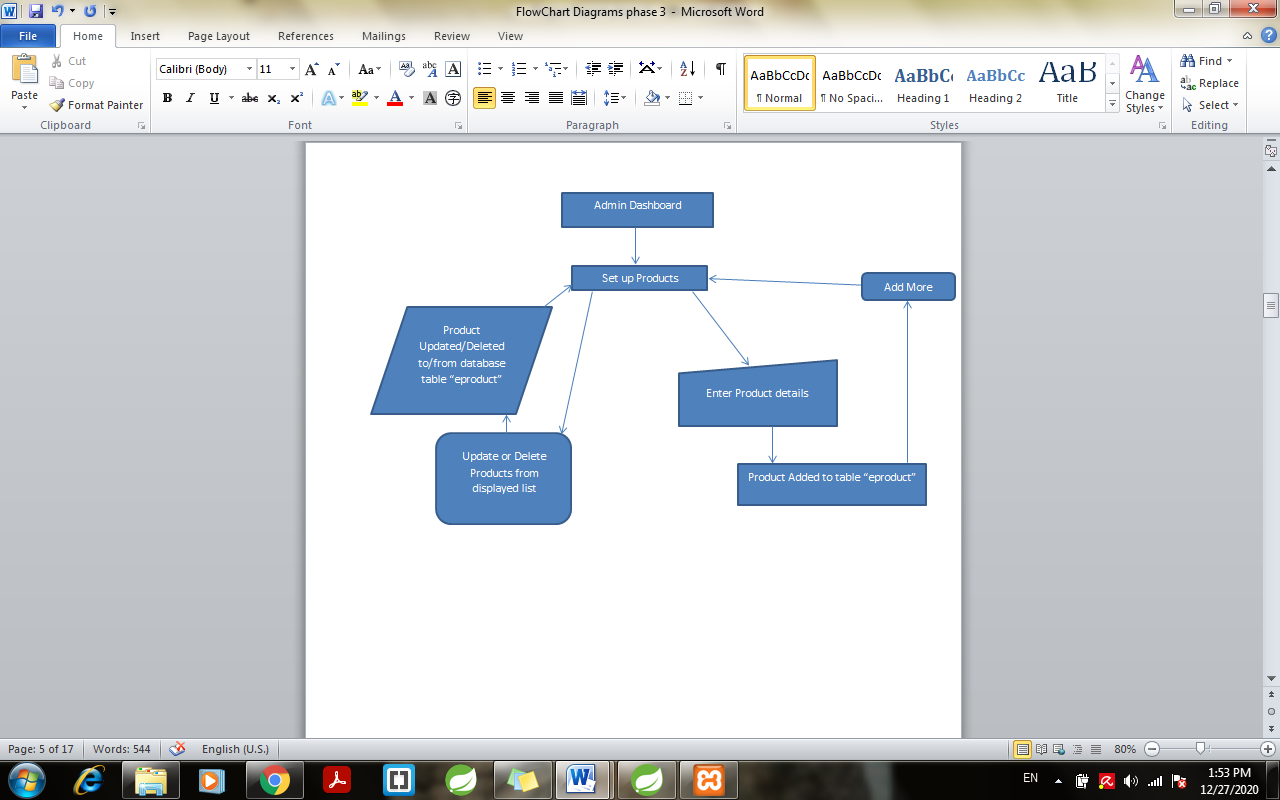
**Home Page**



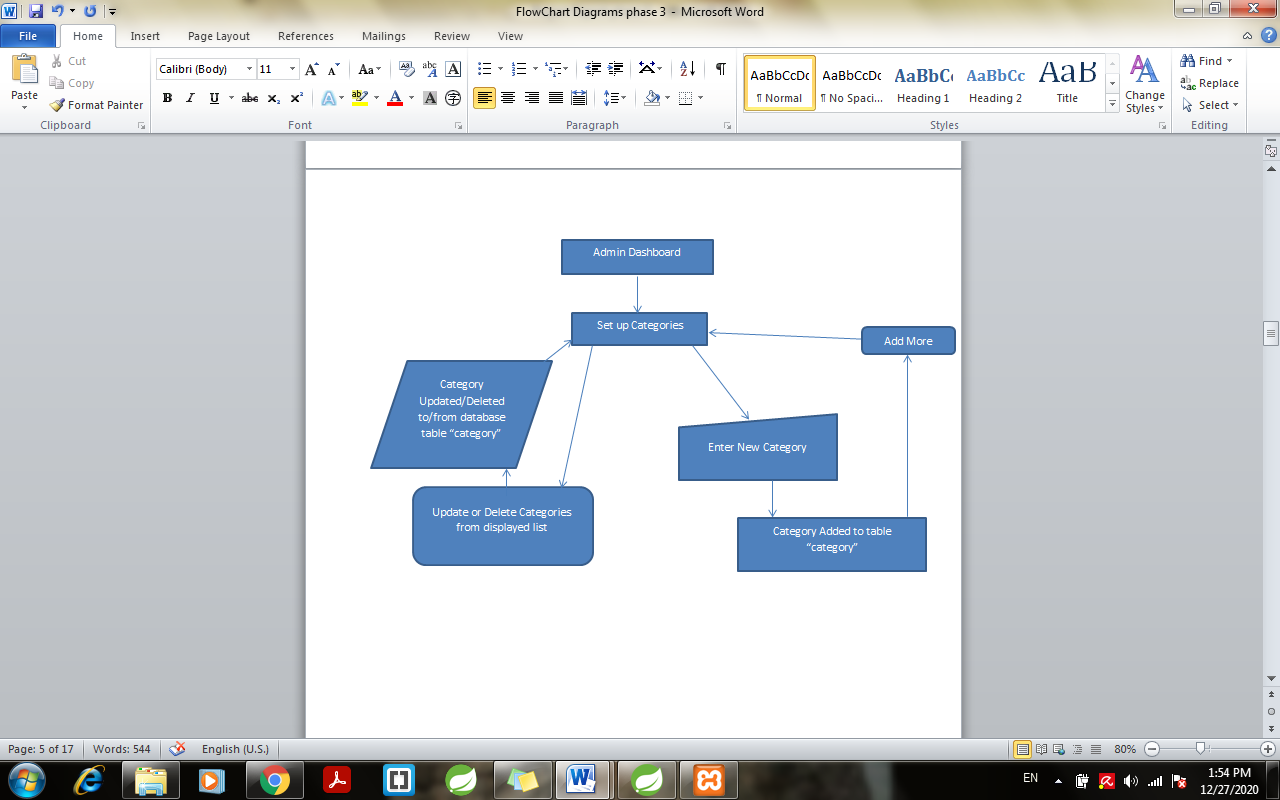
**Admin**



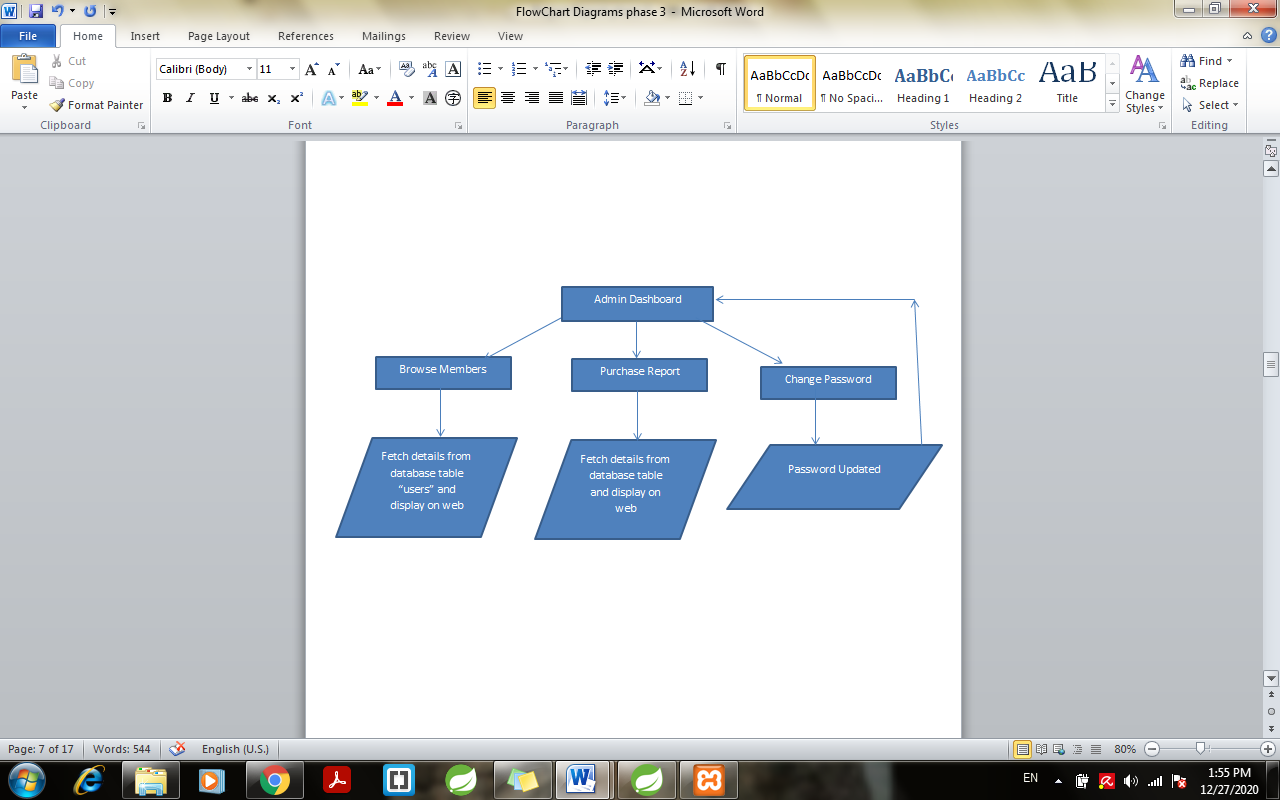
**Admin Task1**



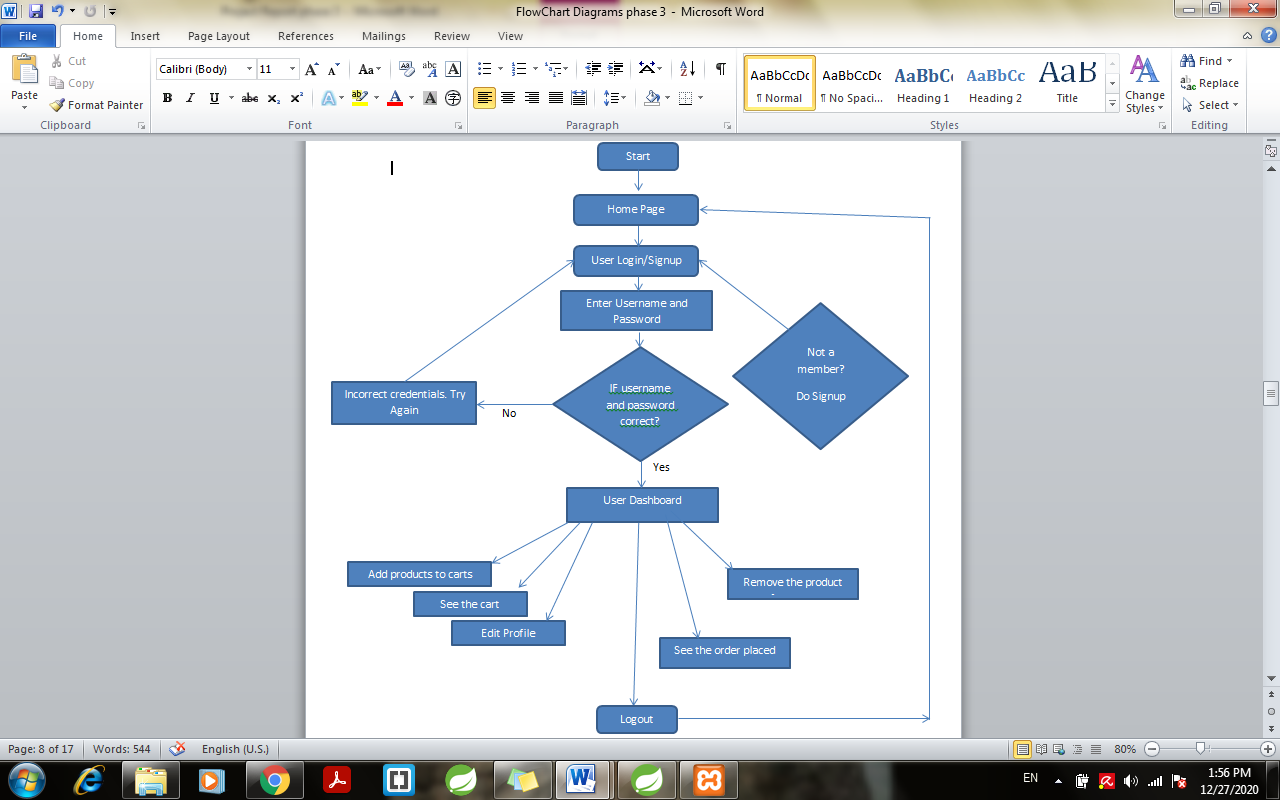
**Admin Task2**



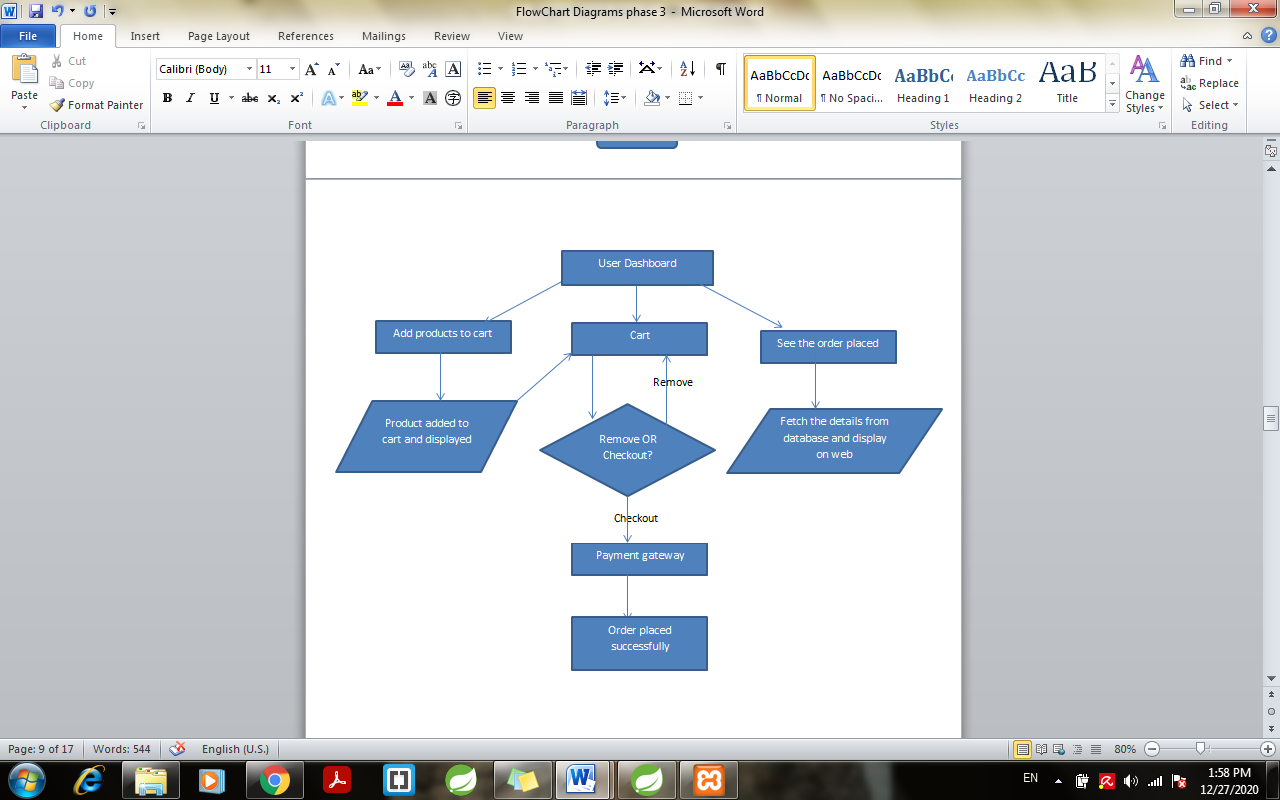
**Admin Task3**



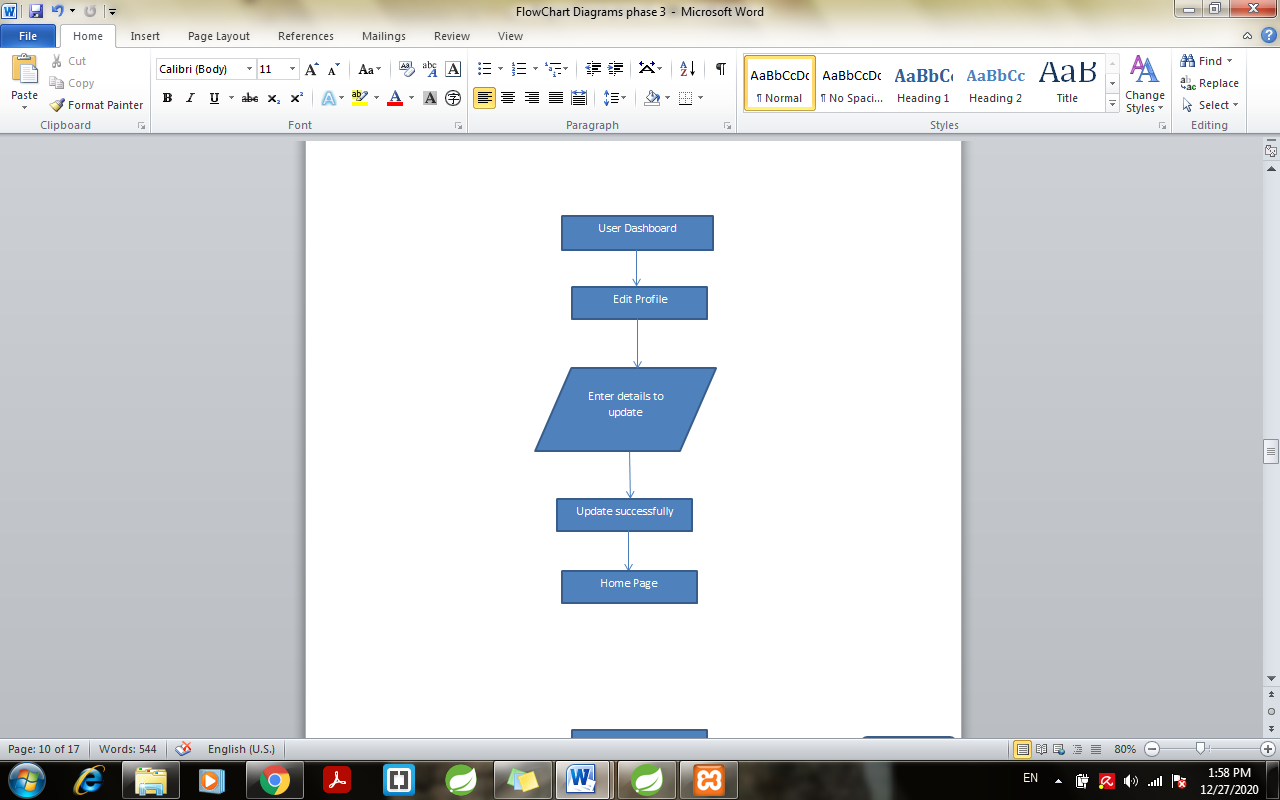
**User Page**



**User Task1**



**User Task2**



**IMPLEMENTATION**

1. **Software, Tools & Technology used: This is a Spring MVC-maven web project.**
2. Operating System - Windows 7 or upper version
3. STS as IDE - To write Java code for application
4. Hibernate as the ORM tool.
5. Spring MVC framework with Hibernate HQL query to make code optimization and increase performance.
6. XAMPP – For mysql and database connectivity
7. Trello – Online web app for sprint planning and make user cards.
8. Maven technology – for dependency injections
9. Front End – HTML and CSS style tags, JSP
10. GitHub – Online repository to track and submit the project
11. MS-Word – To create user stories and flow chart
12. Application hosted using local server Apache Tomcat
13. **Methodology: Agile-Scrum**
14. **Coding:**

Please refer folder “Source code and Screenshots” added in git hub.

**OUTPUT TEST – OUTPUT SCREENSHOTS**

Please refer folder “Source code and Screenshots” added in git hub.

**CORE CONCEPT USED IN THE PROJECT**

This project uses Java Programming language features. Brief of all concepts used in the project are as follow:

1. JSP for developing dynamic web pages.
2. Spring Servlet MVC to connect through JSP web pages.
3. Spring framework annotations like @Autowired, @RequestMappings to map the JSP pages and request through get and post methods, @controller - to executes the business logic, sets the processing result in Model and returns the logical name of view to HandlerAdapter.
4. Maven technology for dependency injections to get dependency jar – “mysql-connector-java”, “spring-core”, “spring-beans”, “spring-webmvc”, “hibernate-core” etc.
5. Use of html and CSS style tags coding in JSP pages to make it more user interacting web interface.
6. Xampp is selected as a database and hibernate as ORM tool.
7. HQL queries for database operations.
8. Spring MVC framework by separating Controller, DAO, Entity and Service class of java with hibernate to create table dynamically using HQL.
9. Use of Spring DispatcherServlet to receive the request and selecting an appropriate controller to HandlerMapping.
10. Core Java concepts like “If statements” for checking the conditions.
11. Java collection concepts like LIST and HASH map techniques , Exception handling, Sorting is used.

**LINK TO THE GITHUB REPOSITORY TO TRACK AND VERIFY THE PROJECT COMPLETION**

<https://github.com/Manish-K-Arora/simplilearnphase3project>

**CONCLUSION**

1. **Summary**

* The project has been developed mainly for company “sporty shoes” that manufactures and sell shoes, who wish to launch their e-commerce portal sportyshoes.com
* It is easy to use, as it uses dynamic JSP web pages.
* User friendly dynamic web based options are provided to user and admin with variable options for performing different users and admin operations.
* The usage of software increases the efficiency, decreases the effort.
* It has been thoroughly tested and implemented.

1. **Future Enhancements**

Some of the future enhancements that can be done to this system are:

* As the technology emerges, it is possible to upgrade the system and can be adaptable to desired environment.
* We can also add additional features like shoes size, availability, colors and dynamic images of product for more interactive experience.
* We can also add feature for admin to check the members buying history, add size, edit the quantity and availability of the products, and send notifications to privileged customers for ongoing discounts and offers.
* We can use Spring-boot with hibernate and JPA as well to make it more efficient.
* Based on the future security issues, security can be improved using the techniques to enter user-name and password using regex expressions of Java to use the application.

1. **Unique Selling Points(USPs)**

This is a Spring MVC based maven web project with dynamic web pages with user-interactive options which are easy to use and perform Admin-operations through user input via Dynamic JSP web-page-interface which enhances the interaction of user with application and includes features to edit, add and fetch records to and from database dynamically.