A Project Report on

Anand ERP System

By

1. **Pranav Yadav** pranavyadav1603@gmail.com
2. **Rajesh Kumbhar** rajeshkumbhar.connect@gmail.com
3. **Vedang Pawar** vedangjpawar@gmail.com

Acknowledgement

The satisfaction and success of completion of this task would be incomplete without heartfelt thanks to people whose constant guidance, support and encouragement made this work successful.
This project has been a wonderful experience where we have learnt and experienced many beneficial things.

Abstract

The purpose of ERP system is to automate the existing manual system by the help of computerized equipment and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. The required software and hardware are easily available and easy to work with.

ERP System, as described above, can lead to error free, secure, reliable and fast management systems. It can assist the user to concentrate on their other activities rather than concentrating on the record keeping. Thus it will help organizations in better utilization of resources. The organization can maintain computerized records without redundant entries. That means that one need not be distracted by information that is not relevant, while being able to reach the information.

The aim is to automate its existing manual system by the help of computerized equipment and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same.

Declaration

We hereby declare that the work reported in this project work on “Anand ERP system” submitted is our original work done. The material contained in the report has not been submitted to any University or Institution for the award of any degree.

Table of Contents

Introduction

The "Anand ERP System" has been developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate and in some cases reduce the hardships faced by this existing system. Moreover this system is designed for the particular need of the company to carry out operations in a smooth and effective manner.

The application is reduced as much as possible to avoid errors while entering the data. It also provides an error message while entering invalid data. No formal knowledge is needed for the user to use this system. Thus by this all it proves it is user-friendly. Enterprise resource planning (ERP) System, as described above, can lead to error free, secure, reliable and fast management systems. It can assist the user to concentrate on their other activities rather than concentrating on the record keeping. Thus it will help organizations in better utilization of resources.

Every organization, whether big or small, has challenges to overcome and manage the information of Supplier, Inventory, Cost, Delivery. This is designed to assist in strategic planning, and will help you ensure that the organization is equipped with the right level of information and details for their future goals. These systems will ultimately allow the organization/shop keepers to better manage resources.

Features and Functionalities

There is one main actor of the application who will interact directly with the application, i.e., shop manager.

The Shop manager can add, edit, and delete the products. He can also manage the records of suppliers and transactions. Following are the features of this application :

1. Increases the efficiency of managing the Inventory.
2. Records are distinguished in categories, which help in easy retrieval of data.
3. It deals with monitoring the information and transactions of cost.
4. Editing, adding and updating of records is improved which results in proper resource management of Inventory data.

Used Tools and Technologies

Tools : Spring Tools Suite
Front-End : HTML, CSS,
Back-End : Java, Spring boot, JPA
Database : MySQL

Database Design

We use a MySQL database for this project. The Description of the function of each table is given below:
For this project, we use 5 tables.

1) categories
    a) category_id
    b) category_name

2) products
    a) product_id
    b) product_name
    c) category_name (Foreign key)
    d) product_stock
    e) product_costprice
    f) product_sellingprice

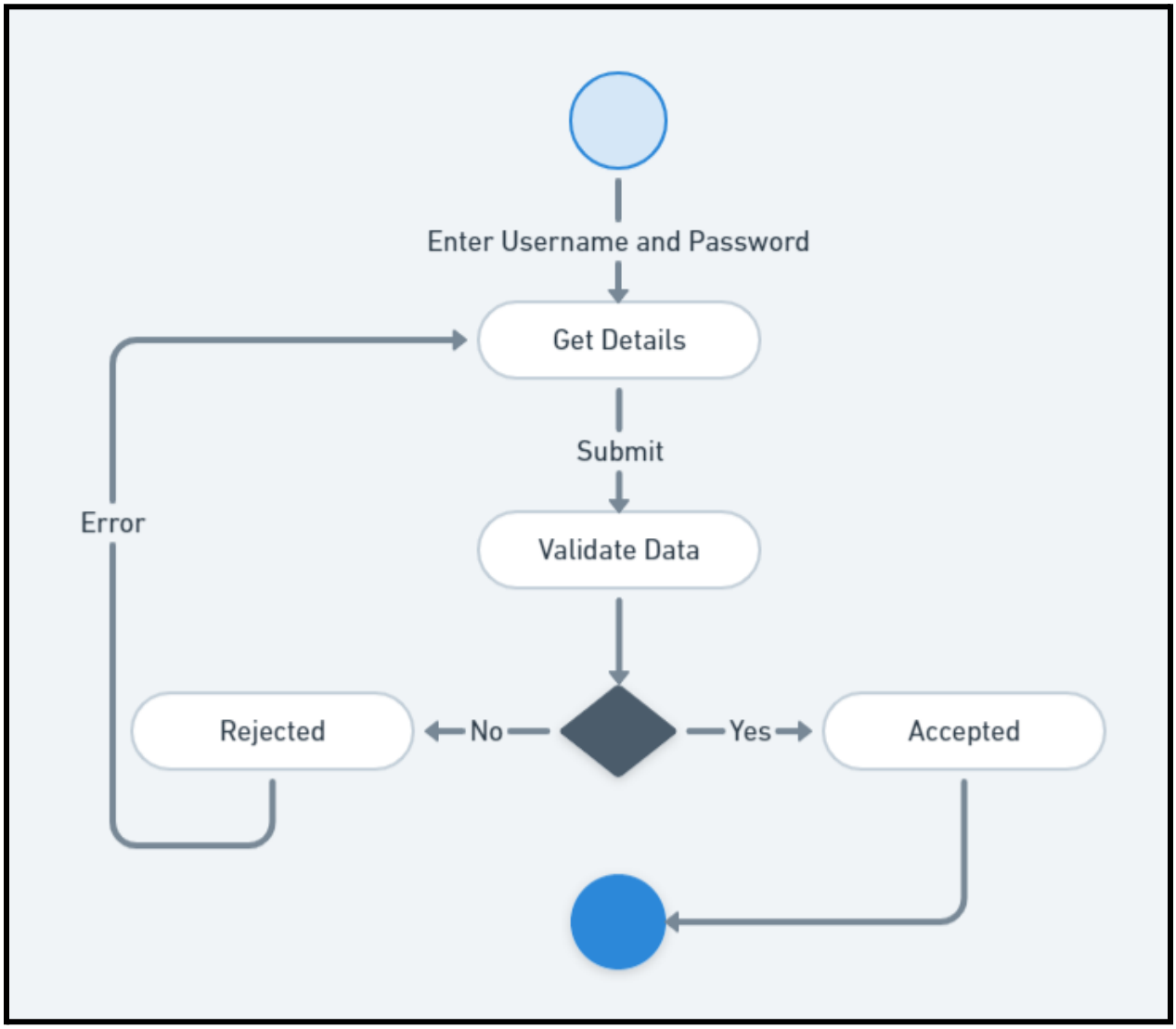
3) cart
    a) product_id (Foreign key)
    b) quantity

4) customer
    a) customer_id
    b) customer_name
    c) cart (Foreign key)
    d) payment_mode

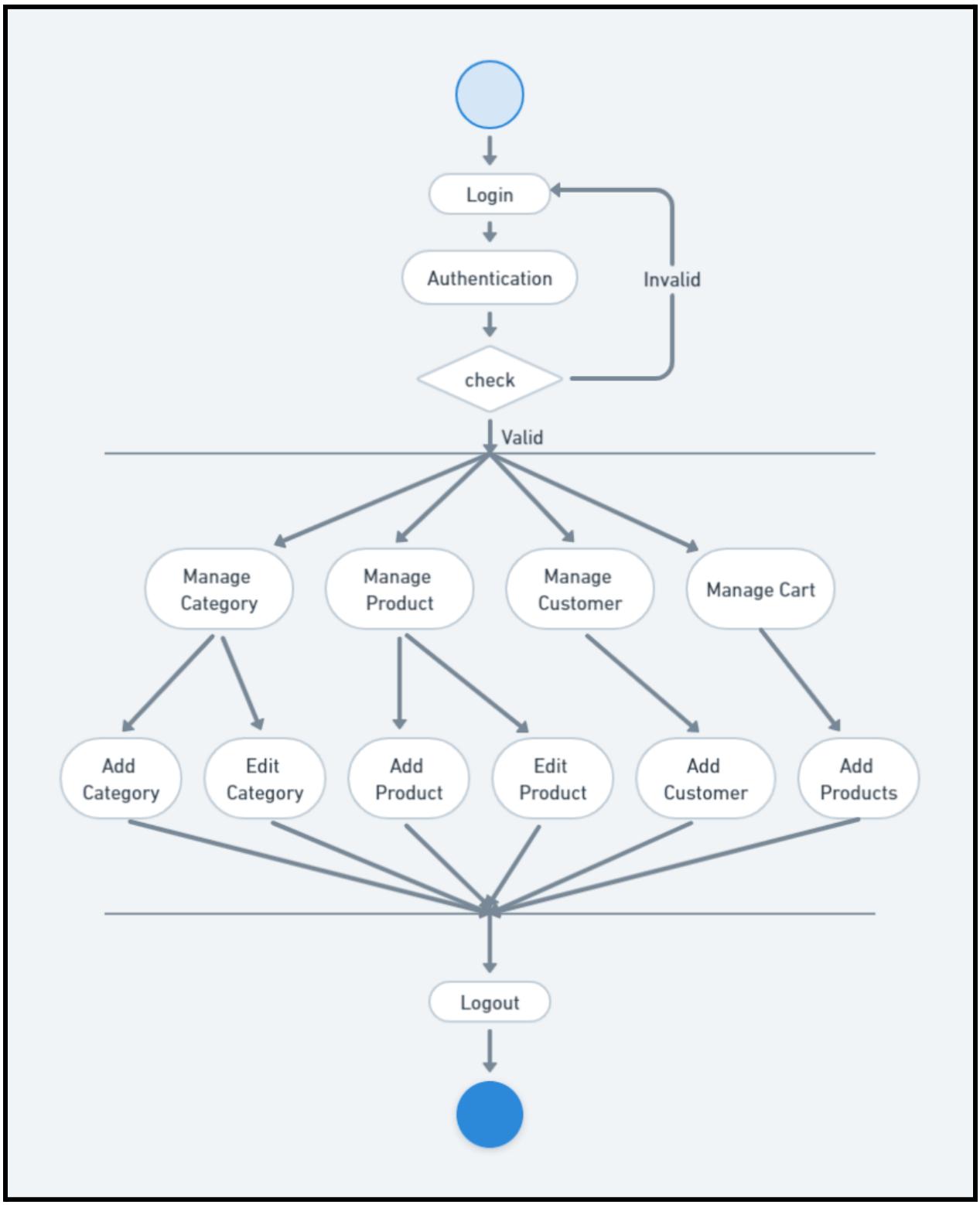
5) login
    a) login_id
    b) login_name

Diagrams

Login activity diagram :



Shop Manager activity diagram :



Design and Implementation

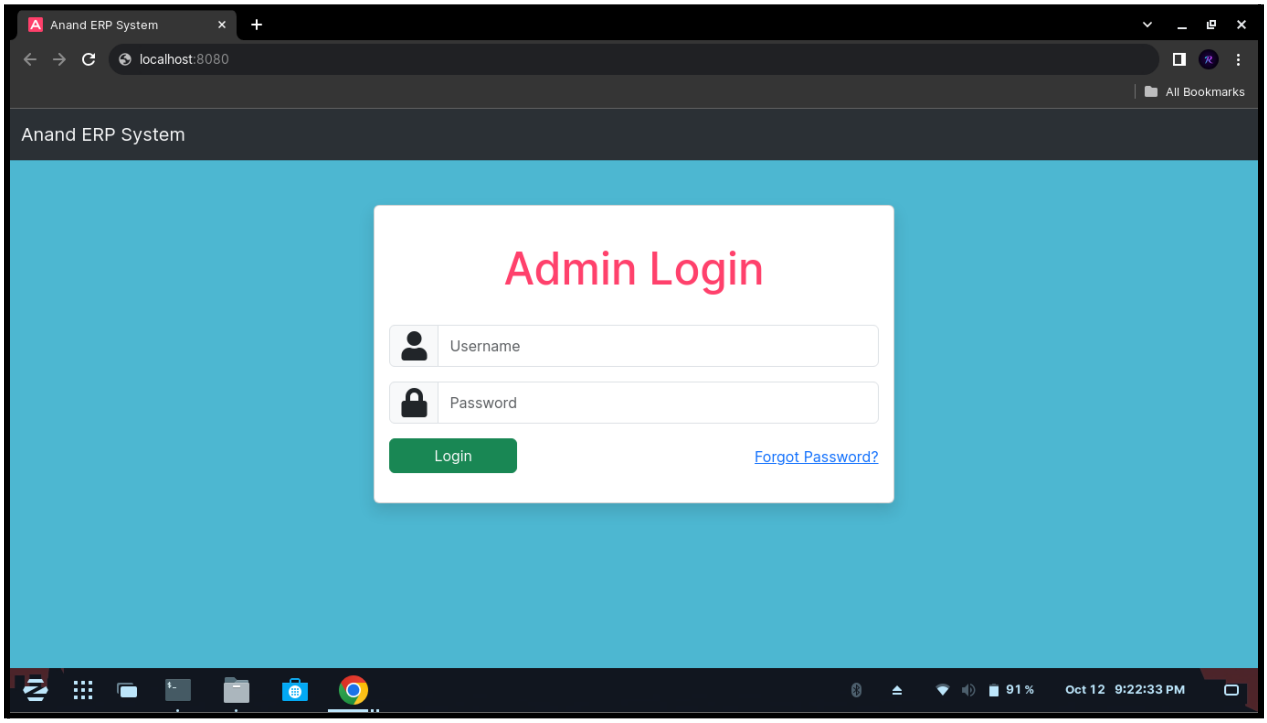


Figure : Admin Login page

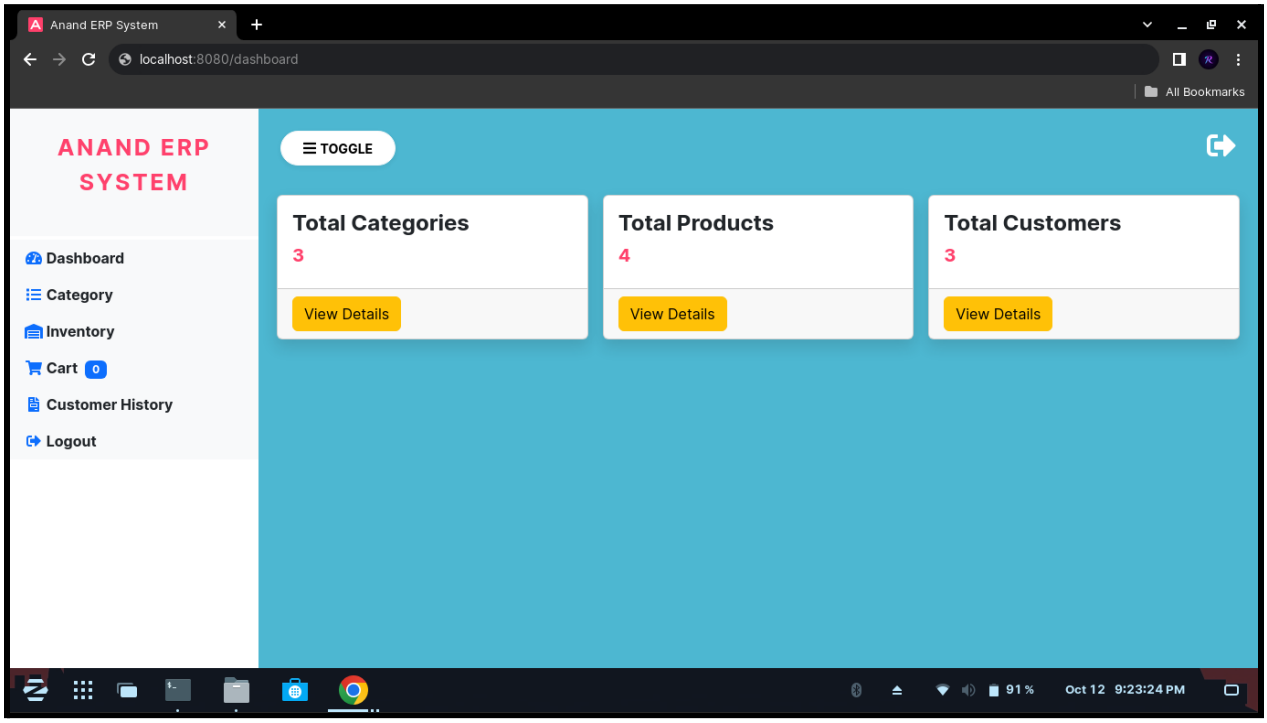


Figure : Dashboard page

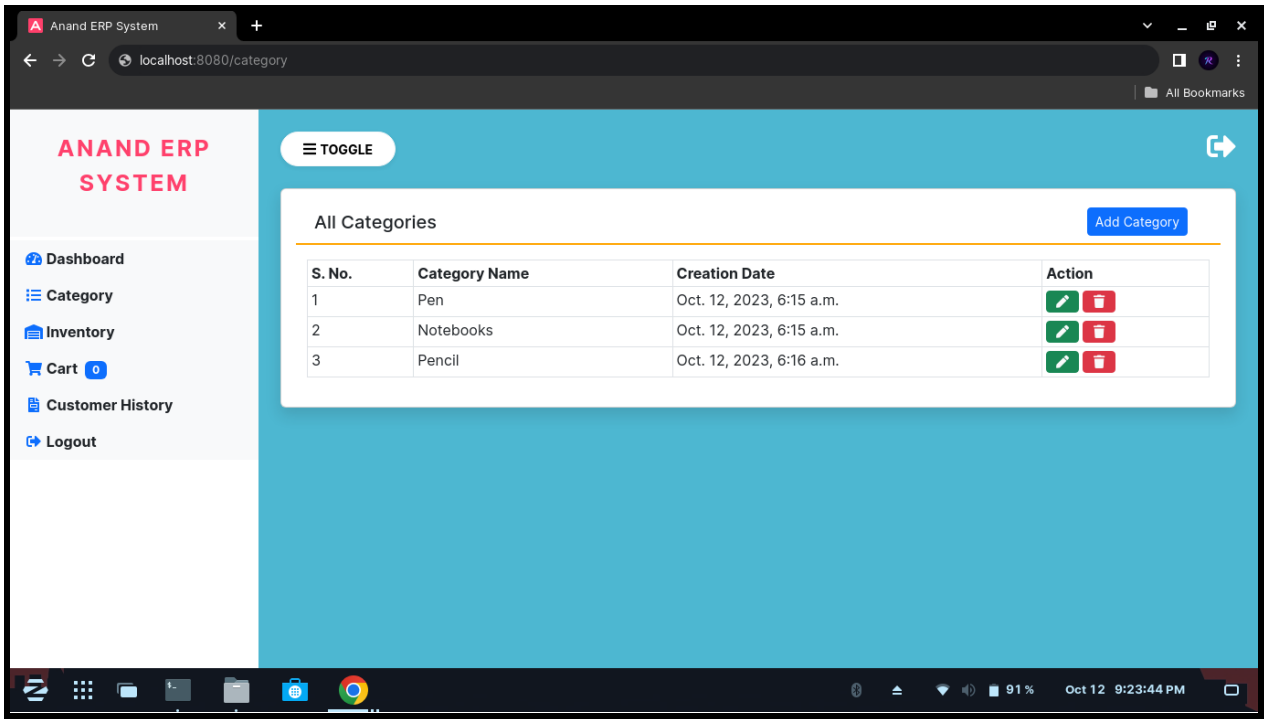


Figure : Category page

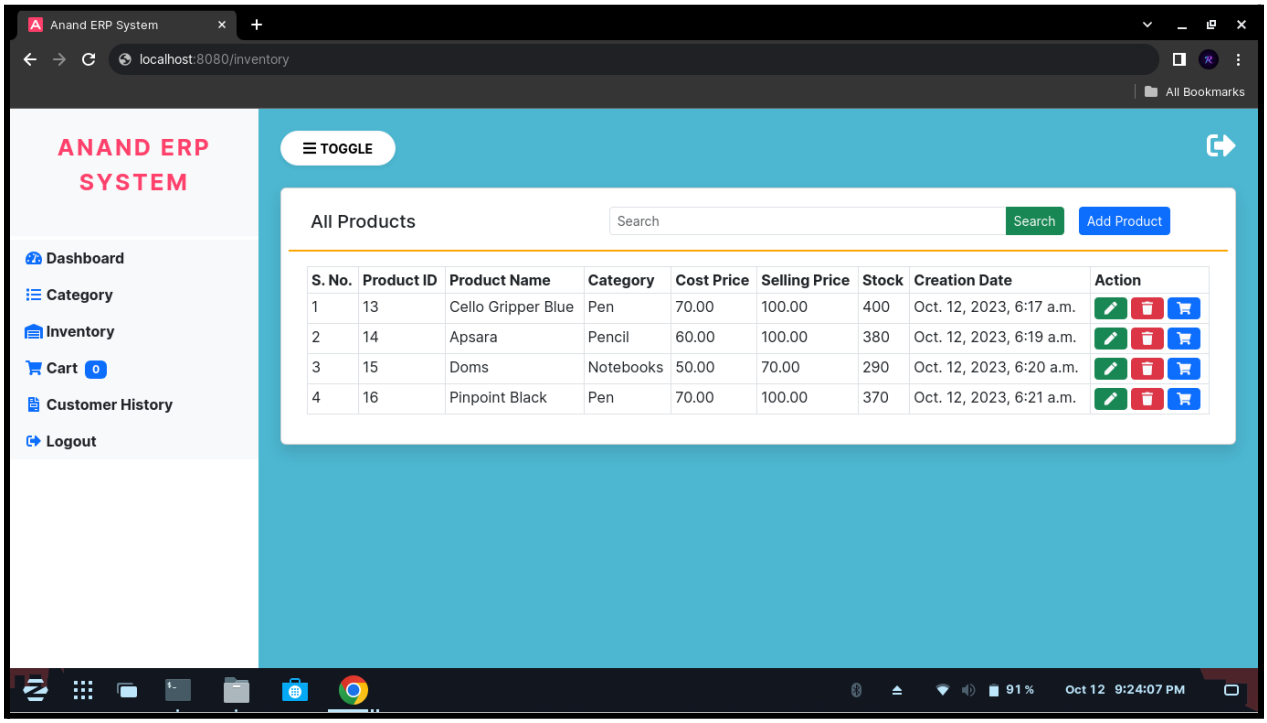


Figure : Inventory page

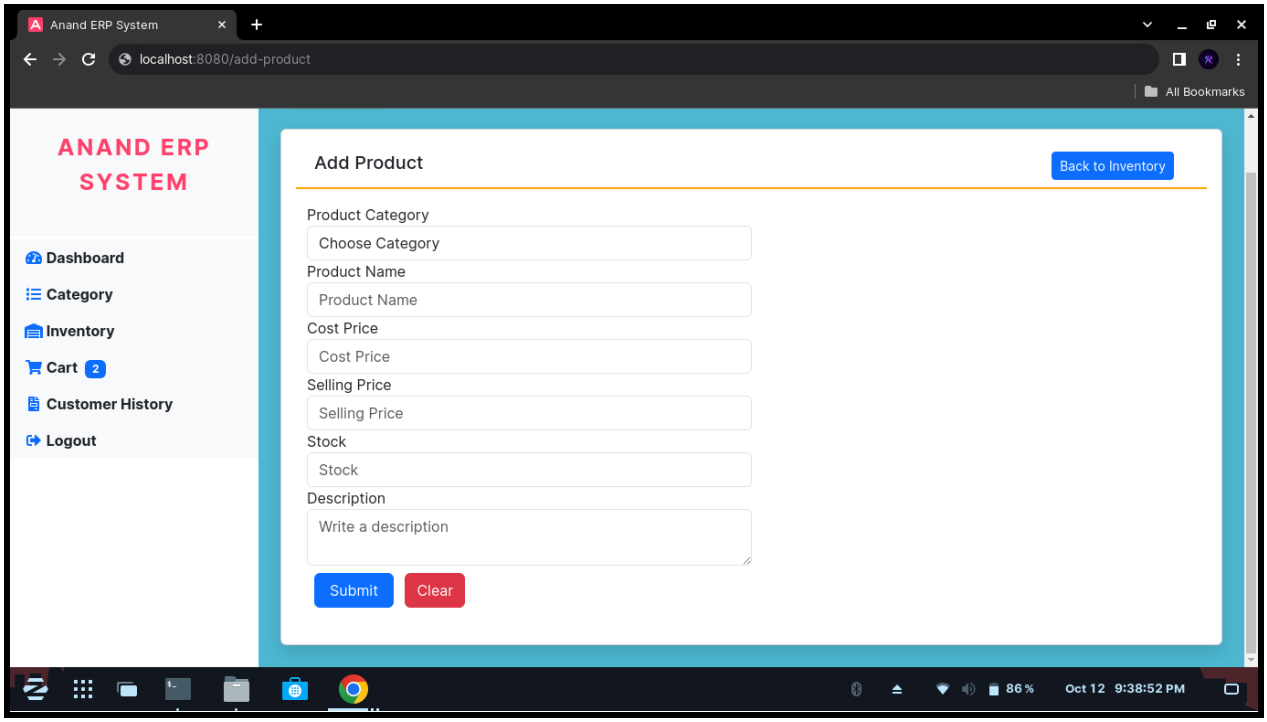


Figure : Add product page

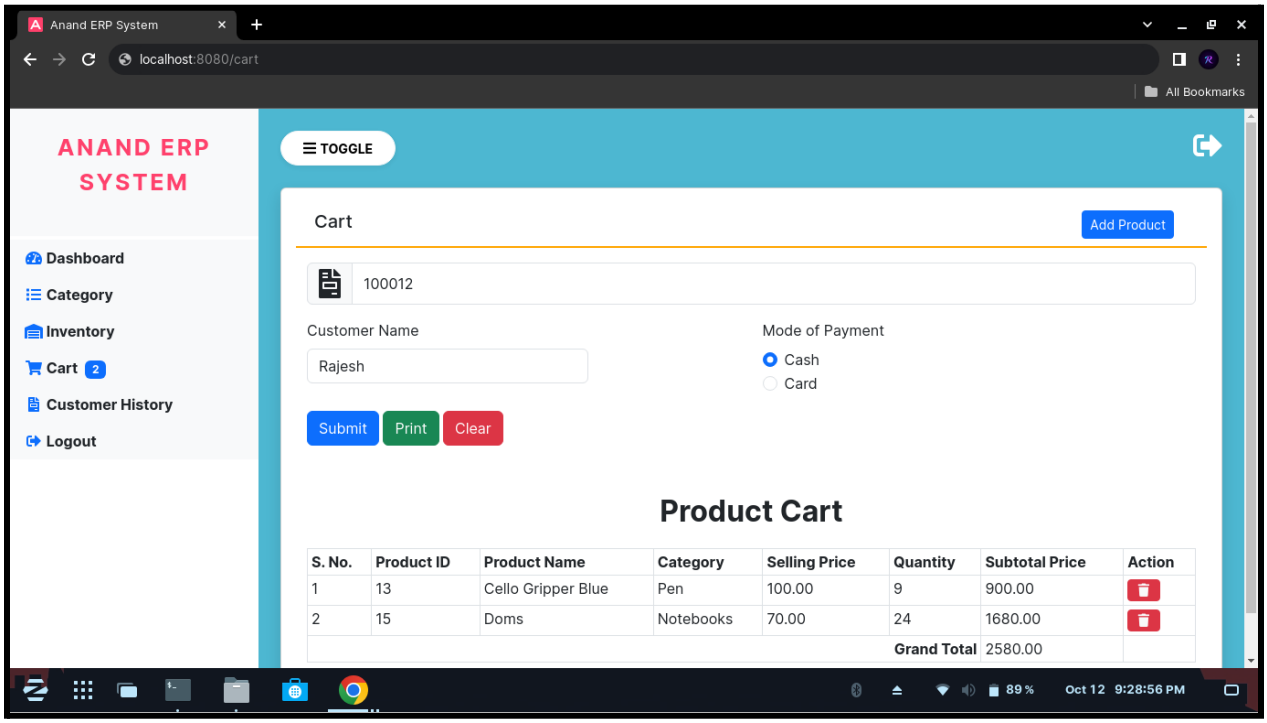


Figure : Cart page

Source Code

CategoryRepository.java :

```
package com.example.repository;

import com.example.entity.Category;
import org.springframework.data.repository.CrudRepository;
import org.springframework.stereotype.Repository;

@Repository
public interface CategoryRepository extends CrudRepository<Category, Integer> {


}
```

Category.java :

```
package com.example.entity;

import com.fasterxml.jackson.annotation.JsonIgnore;

import java.io.Serializable;
import javax.persistence.*;
import java.math.BigDecimal;
import java.util.Date;
import java.util.List;

@Entity
@NamedQuery(name="Category.findAll", query="SELECT c FROM Category c")
public class Category implements Serializable {
	private static final long serialVersionUID = 1L;

	@Id
	@GeneratedValue(strategy=GenerationType.AUTO)
	private int categoryId;

	private String categoryName;

	@Temporal(TemporalType.TIMESTAMP)
	private Date createdDateTime;

	private String createdUser;

	@Temporal(TemporalType.TIMESTAMP)
	private Date lastModifiedDateTime;
```

```
private String lastModifiedUser;

private BigDecimal version;

//bi-directional many-to-one association to Product
@OneToMany(mappedBy="category")
private List<Product> products;

//bi-directional many-to-one association to Stock
@OneToMany(mappedBy="category")
private List<Stock> stocks;

public int getCategoryId() {
	return this.categoryId;
}

public String getCategoryName() {
	return this.categoryName;
}

public void setCategoryName(String categoryName) {
	this.categoryName = categoryName;
}

public Date getCreatedDateTime() {
	return this.createdDateTime;
}

public void setCreatedDateTime(Date createdDateTime) {
	this.createdDateTime = createdDateTime;
}

public String getCreatedUser() {
	return this.createdUser;
}

public void setCreatedUser(String createdUser) {
	this.createdUser = createdUser;
}

public Date getLastModifiedDateTime() {
	return this.lastModifiedDateTime;
}

public void setLastModifiedDateTime(Date lastModifiedDateTime) {
	this.lastModifiedDateTime = lastModifiedDateTime;
```

```
}

public String getLastModifiedUser() {
	return this.lastModifiedUser;
}

public void setLastModifiedUser(String lastModifiedUser) {
	this.lastModifiedUser = lastModifiedUser;
}

public BigDecimal getVersion() {
	return this.version;
}

public void setVersion(BigDecimal version) {
	this.version = version;
}

public void setCategoryId(int categoryId) {
	this.categoryId = categoryId;
}

@JsonIgnore
public List<Product> getProducts() {
	return this.products;
}

public void setProducts(List<Product> products) {
	this.products = products;
}


public Product addProduct(Product product) {
	getProducts().add(product);
	product.setCategory(this);

	return product;
}

public Product removeProduct(Product product) {
	getProducts().remove(product);
	product.setCategory(null);

	return product;
}
```

```
	public List<Stock> getStocks() {
		return this.stocks;
	}

	public void setStocks(List<Stock> stocks) {
		this.stocks = stocks;
	}

	public Stock addStock(Stock stock) {
		getStocks().add(stock);
		stock.setCategory(this);

		return stock;
	}

	public Stock removeStock(Stock stock) {
		getStocks().remove(stock);
		stock.setCategory(null);

		return stock;
	}

}
```

Product.java :

```
package com.example.entity;

import java.io.Serializable;
import javax.persistence.*;
import java.math.BigDecimal;
import java.util.Date;
import java.util.List;

@Entity
@NamedQuery(name="Product.findAll", query="SELECT p FROM Product p")
public class Product implements Serializable {
	private static final long serialVersionUID = 1L;

	@Id
	@GeneratedValue(strategy=GenerationType.AUTO)
	private int productId;

	@Temporal(TemporalType.TIMESTAMP)
	private Date createdDateTime;
```

```
	private String createdUser;

	@Temporal(TemporalType.TIMESTAMP)
	private Date lastModifiedDateTime;

	private String lastModifiedUser;

	private double productbuyingPrice;

	private byte productIsService;

	private String productName;

	private double productsellingPrice;

	private BigDecimal version;

	//bi-directional many-to-one association to Category
	@ManyToOne
	@JoinColumn(name="categoryId")
	private Category category;

	//bi-directional many-to-one association to ProductInvoice
	@OneToMany(mappedBy="product")
	private List<ProductInvoice> productInvoices;

	//bi-directional many-to-one association to ProductPricing
	@OneToMany(mappedBy="product")
	private List<ProductPricing> productPricings;

	//bi-directional many-to-one association to Stock
	@OneToMany(mappedBy="product")
	private List<Stock> stocks;

	public int getProductId() {
		return this.productId;
	}

	public void setProductId(int productId) {
		this.productId = productId;
	}

	public Date getCreatedDateTime() {
		return this.createdDateTime;
	}
```

```
public void setCreatedDateTime(Date createdDateTime) {
	this.createdDateTime = createdDateTime;
}

public String getCreatedUser() {
	return this.createdUser;
}

public void setCreatedUser(String createdUser) {
	this.createdUser = createdUser;
}

public Date getLastModifiedDateTime() {
	return this.lastModifiedDateTime;
}

public void setLastModifiedDateTime(Date lastModifiedDateTime) {
	this.lastModifiedDateTime = lastModifiedDateTime;
}

public String getLastModifiedUser() {
	return this.lastModifiedUser;
}

public void setLastModifiedUser(String lastModifiedUser) {
	this.lastModifiedUser = lastModifiedUser;
}

public double getProductbuyingPrice() {
	return this.productbuyingPrice;
}

public void setProductbuyingPrice(double productbuyingPrice) {
	this.productbuyingPrice = productbuyingPrice;
}

public byte getProductIsService() {
	return this.productIsService;
}

public void setProductIsService(byte productIsService) {
	this.productIsService = productIsService;
}

public String getProductName() {
	return this.productName;
```

```
}

public void setProductName(String productName) {
	this.productName = productName;
}

public double getProductsellingPrice() {
	return this.productsellingPrice;
}

public void setProductsellingPrice(double productsellingPrice) {
	this.productsellingPrice = productsellingPrice;
}

public BigDecimal getVersion() {
	return this.version;
}

public void setVersion(BigDecimal version) {
	this.version = version;
}

public Category getCategory() {
	return this.category;
}

public void setCategory(Category category) {
	this.category = category;
}

public List<ProductInvoice> getProductInvoices() {
	return this.productInvoices;
}

public void setProductInvoices(List<ProductInvoice> productInvoices) {
	this.productInvoices = productInvoices;
}

public ProductInvoice addProductInvoice(ProductInvoice productInvoice) {
	getProductInvoices().add(productInvoice);
	productInvoice.setProduct(this);

	return productInvoice;
}
```

```
	public ProductInvoice removeProductInvoice(ProductInvoice productInvoice) {
		getProductInvoices().remove(productInvoice);
		productInvoice.setProduct(null);

		return productInvoice;
	}

	public List<ProductPricing> getProductPricings() {
		return this.productPricings;
	}

	public void setProductPricings(List<ProductPricing> productPricings) {
		this.productPricings = productPricings;
	}

	public List<Stock> getStocks() {
		return this.stocks;
	}

	public void setStocks(List<Stock> stocks) {
		this.stocks = stocks;
	}

	public Stock addStock(Stock stock) {
		getStocks().add(stock);
		stock.setProduct(this);

		return stock;
	}

	public Stock removeStock(Stock stock) {
		getStocks().remove(stock);
		stock.setProduct(null);

		return stock;
	}

}
```

CategoryService.java :

```
package com.example.service;

import com.example.entity.Category;
import com.example.repository.CategoryRepository;
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.stereotype.Service;

import javax.transaction.Transactional;
import java.util.Optional;

@Transactional
@Service
public class CategoryService {

    @Autowired
    private CategoryRepository categoryRepository;

    public void insert(Category category) {
        categoryRepository.save(category);
    }


    public Optional<Category> findById(int id) {
        return categoryRepository.findById(id);
    }

    public Iterable<Category> findAll() {
        return categoryRepository.findAll();
    }

    public void updateCategory(Category category) {

        categoryRepository.save(category);
    }

    public void deleteCategory(Category category) {

        categoryRepository.delete(category);
    }

}
```

ProductService.java :

```
package com.example.service;

import com.example.entity.Product;
import com.example.repository.ProductRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
```

```
import javax.transaction.Transactional;
import java.util.Optional;

@Transactional
@Service
public class ProductService {

   @Autowired
   public ProductRepository productRepository;

   public void insert(Product product) {
      productRepository.save(product);
   }

   public Optional<Product> find(int id) {
      return productRepository.findById(id);
   }

   public Iterable<Product> findAll() {
      return productRepository.findAll();
   }

   public void updateProduct(Product product) {
      productRepository.save(product);
   }

   public void deleteProduct(Product product) {
      productRepository.delete(product);
   }
}
```

CategoryController.java :

```
package com.example.controller;

import com.example.entity.TheLogConverter;
import com.example.entity.Category;
import com.example.service.CategoryLogService;
import com.example.service.CategoryService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

import java.util.Optional;

@RestController
@RequestMapping("/categories")
```

```
public class CategoryController {

    @Autowired
    public CategoryService categoryService;
    @Autowired
    private CategoryLogService categoryLogService;

    @RequestMapping("")
    public Iterable<Category> getAllCategory() {
        return categoryService.findAll();
    }

    @RequestMapping("/{id}")
    public Optional<Category> searchCategory(@PathVariable int id) {
        return categoryService.findById(id);
    }

    @RequestMapping(method = RequestMethod.POST, value = "")
    public void addCategory(@RequestBody Category category) {
        categoryService.insert(category);
        categoryLogService.insert(TheLogConverter.categoryLogConverter(category));
    }

    @RequestMapping(method = RequestMethod.PUT,value ="/{id}")
    public void updateCategory(@RequestBody Category category) {
        categoryService.updateCategory(category);
        categoryLogService.insert(TheLogConverter.categoryLogConverter(category));
    }

    @RequestMapping(method = RequestMethod.DELETE,value ="/{id}")
    public void deleteCategory(@RequestBody Category category) {
        categoryService.deleteCategory(category);
        categoryLogService.insert(TheLogConverter.categoryLogConverter(category));
    }

}
```

ProductController.java :

```
package com.example.controller;

import com.example.entity.Product;
import com.example.entity.TheLogConverter;
import com.example.service.ProductLogService;
import com.example.service.ProductService;
import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.web.bind.annotation.*;

import java.util.Optional;

@RestController
@RequestMapping("categories/{id}/products")
public class ProductController {

    @Autowired
    public ProductService productService;
    @Autowired
    private ProductLogService productLogService;

    @RequestMapping("")
    public Iterable<Product> getAllProducts() {
        return productService.findAll();
    }

    @RequestMapping("/{id}")
    public Optional<Product> searchProduct(@PathVariable int id) {
        return productService.find(id);
    }

    @RequestMapping(method = RequestMethod.POST, value = "")
    public void addProduct(@RequestBody Product product) {
        productService.insert(product);
        productLogService.insert(TheLogConverter.productLogConverter(product));
    }

    @RequestMapping(method = RequestMethod.PUT,value ="/{id}")
    public void updateProduct(@RequestBody Product product) {
        productService.updateProduct(product);
        productLogService.insert(TheLogConverter.productLogConverter(product));
    }

    @RequestMapping(method = RequestMethod.DELETE,value ="/{id}")
    public void deleteProduct(@RequestBody Product product) {
        productService.deleteProduct(product);
        productLogService.insert(TheLogConverter.productLogConverter(product));
    }

}
```

Conclusion

The goal of the "Anand ERP System" project was to make it simpler for shopkeepers to maintain and assess inventory. Despite several obstacles, our team managed to create a highly beneficial and user-friendly programme with the aid of our guide and others. Taking into account how well this project's various components performed overall.

We have the chance to learn about many technologies through this project and work on a project that could be extremely beneficial in the actual world. We discovered the value of cooperation, time management, consistency, and tenacity.

Future Scope

In future, we can increase the scope of this project by following enhancements :

1) Adding an online payment gateway.
2) Printing customized invoices.